ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS

IMPLEMENTACIÓN DE UN SISTEMA DE GEO-POSICIONAMIENTO DE MASCOTAS

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE TECNÓLOGO EN ELECTRÓNICA Y TELECOMUNICACIONES

Julio Cesar Diaz Carrera
julio.diaz@epn.edu.ec

German Danilo Tiupul Ashqui
german.tiupul@epn.edu.ec

DIRECTOR: ING. FERNANO VINICIO BECERRA CAMACHO, MSc.
fernando.becerrac@epn.edu.ec

CODIRECTOR: ING. FABIO MATÍAS GONZÁLEZ GONZÁLEZ, MSc.
fabio.gonzalez@epn.edu.ec

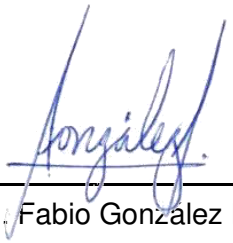
Quito, AGOSTO 2021

CERTIFICACIÓN

Certificamos que el presente trabajo fue desarrollado por el Sr. Diaz Carrera Julio Cesar y Sr. Tiupul Ashqui German Danilo como requerimiento parcial a la obtención del título de TECNÓLOGO EN ELECTRÓNICA Y TELECOMUNICACIONES, bajo nuestra supervisión:

Ing. Fernando Becerra MSc.

DIRECTOR DEL PROYECTO



Ing. Fabio Gonzalez MSc.

CODIRECTOR DEL PROYECTO

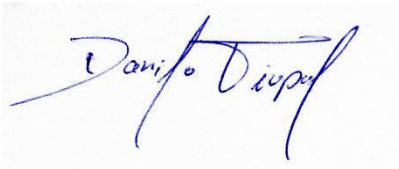
DECLARACIÓN

Nosotros Diaz Carrera Julio Cesar con CI: 1754588463 y Tiupul Ashqui German Danilo con CI: 606021236 declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y que hemos consultado las referencias bibliográficas que se incluyen en este documento.

Sin prejuicio de los derechos reconocidos en el primer párrafo del artículo 144 del Código Orgánico de la Economía Social de los Conocimientos, Creatividad e Innovación – COESC-, somos titulares de la obra en mención y otorgamos una licencia gratuita, intransferible y no exclusiva de uso con fines académicos a la Escuela Politécnica Nacional.

Entregamos toda la información técnica pertinente, en caso de que hubiese una explotación comercial de la obra por parte de la EPN, se negociará los porcentajes de los beneficios conforme lo establece la normativa nacional vigente.

Diaz Carrera Julio Cesar



Tiupul Ashqui German Danilo

DEDICATORIA

Dedicado a mis padres, hermana, compañeros, amigos que me han dado su apoyo para lograr cumplir uno de mis objetivos.

- Julio Cesar Diaz Carrera

AGRADECIMIENTO

Doy un agradecimiento a todas las personas que me dieron su apoyo en todo momento.

- Julio Cesar Diaz Carrera

DEDICATORIA

A ella, su manera de sonreír a pesar de sus problemas, su manera de ser tan fuerte cuando tenía todo el derecho de sentirse débil.

- German Danilo Tiupul Ashqui

AGRADECIMIENTO

Agradecido con Jehová que me permitió seguir cumpliendo los objetivos que me planteo cada día.

- German Danilo Tiupul Ashqui

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

El presente trabajo de titulación “Implementación de un sistema de geo-posicionamiento de mascotas” busca crear un sistema electrónico e informático que rastree la ubicación de una mascota en caso de llegar a perderse, poniendo en práctica los conocimientos adquiridos durante la carrera de Tecnología en Electrónica y Telecomunicaciones, añadiendo a esto el respectivo análisis y comprensión de la parte informática para presentar de manera visual este proyecto.

A continuación, se presentan cinco secciones en las cuales se sintetiza lo realizado durante el desarrollo de este trabajo de titulación:

La sección uno contiene la introducción en la cual se describe el problema que se quiere resolver en este trabajo de titulación, también se encuentran los objetivos a cumplirse, así como los fundamentos necesarios para comprender el desarrollo del sistema.

La sección dos contiene la metodología utilizada para alcanzar la construcción del sistema de geo–posicionamiento.

En la sección tres se encuentran los pasos seguidos para lograr los objetivos, como son: el diseño que contenga la parte electrónica, desarrollando la parte informática para presentación de datos, la programación para control de módulos y la unión de los anteriores para pruebas finales.

En la sección cuatro se presentan las conclusiones que se obtuvieron durante la realización del proyecto, así como las recomendaciones respectivas para realizar mejoras.

Para finalizar, la última sección contiene la bibliografía usada para cumplir los objetivos.

PALABRAS CLAVE: Geo-posicionamiento, GPS/GPRS, Servidor Web, Visual Studio, Google Maps.

ABSTRACT

The present titling plan "Implementation of a geo positioning system of pets" seeks to create an electronic and computer system that tracks the location of a pet in case of getting lost, putting into practice the knowledge acquired during the career of Technology in Electronics and Telecommunications, adding to this the respective analysis and understanding of the computer part to present this project visually.

Next, five sections are presented in which what has been done during the development of this degree work is synthesized:

Section one contains the introduction in which the problem to be solved in this degree work is described, also find the objectives to be met, as well as foundations necessary to understand the development of the system.

Section two contains the methodology used to achieve construction of the geo-positioning system.

Section three are the steps followed to finalize the objectives, such as: the design that contains the electronic part, developing the computer part for data presentation, programming for control of modules and the union of the previous ones for final tests.

Section four presents the conclusions that were obtained during the project, as well as the respective recommendations for making improvements.

Finally, the last section contains the bibliography used to fulfill the objectives.

KEYWORDS: Geo positioning, GPS/GPRS, Web Server, Visual Studio, Google Maps.

1 INTRODUCIÓN

En Ecuador varias familias poseen mascotas, por lo general estas son perros, es común ver carteles con imágenes de sus mascotas perdidas, familias ofrecen recompensas con la esperanza de recuperar a aquellos seres que consideran parte de su familia.

Aunque pueden existir varias formas de encontrar a las mascotas perdidas, el geoposicionamiento se encuentra entre las más precisas, este proyecto de titulación permite presentar las coordenadas geográficas en un mapa señalando los lugares que han sido recorridos por cada mascota.

El funcionamiento del dispositivo es simple, un módulo Arduino junto a un módulo GPS, este último puede enviar las coordenadas en longitud y latitud, estos datos pueden ser interpretados por la placa Arduino y como consecuencia enviar esos datos a un servidor para que el usuario pueda tener acceso a ellos mediante una interfaz amigable.

Para crear el sistema de geoposicionamiento, se realiza un análisis de los requerimientos que serán necesarios tanto en el uso del software como del hardware, mismos que serán la base de la implementación de este proyecto.

Con el software y hardware ya definidos se procede a realizar la adquisición de los elementos y la construcción de un prototipo que permita cumplir con el proyecto de manera que satisfaga las necesidades del usuario, con las pruebas del prototipo ya realizadas se procede a la construcción final.

Posteriormente se realizan las pruebas de funcionamiento del sistema de geo-posicionamiento, con ello se trata de evitar y corregir los posibles errores durante la conexión y fabricación del hardware, una vez verificada la funcionalidad del proyecto, estará listo para su uso.

1.1 Objetivos

Objetivo general

Implementar un sistema de geo-posicionamiento de mascotas.

Objetivos específicos

Realizar un estudio de las tecnologías utilizadas GPS y GPRS.

Diseñar un arnés para soporte de hardware de geo-posicionamiento.

Desarrollar una interfaz para la localización de mascotas con servidor WEB.

Implementar el sistema de geo-posicionamiento.

Realizar pruebas de funcionamiento del sistema diseñado.

1.2 Fundamentos

GPS (*Global Positioning System*)

Es un sistema que utiliza 24 satélites que orbitan el planeta tierra, de los cuales solo son requeridos cuatro satélites para lograr tener una ubicación, cada satélite envía la hora exacta y su posición por medio de una señal de radiofrecuencia hacia el receptor GPS que se encuentra en la superficie terrestre [1].

El método de trilateración visto en la Figura 1.1 usa la distancia que existe entre el receptor GPS y cada satélite para obtener la posición del receptor GPS, para ello al tiempo de recepción se le resta el tiempo de envío, multiplicado por la velocidad de la luz [2].

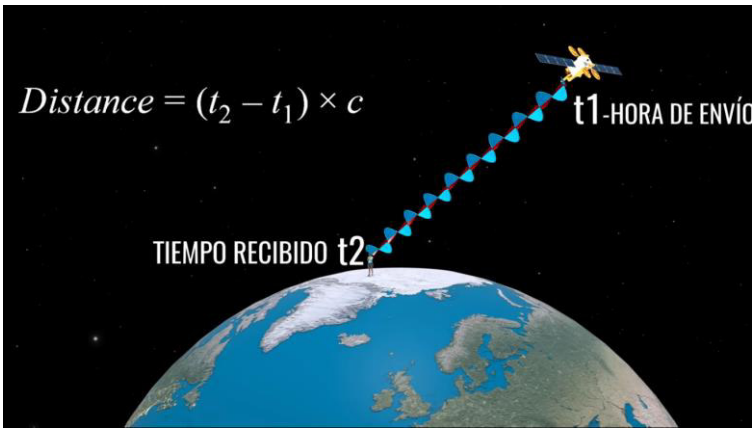


Figura 1.1 Método de trilateración GPS [1]

GPRS (*General Packet Radio Service*)

Es una extensión del sistema global de comunicaciones móviles, por ello utiliza el mismo sistema de los teléfonos celulares. El sistema GPRS permite que los datos sean almacenados en paquetes los cuales serán transmitidos por medio de una red de telefonía móvil, los dispositivos que posean GPRS pueden tener acceso no solo a telefonía móvil también pueden acceder a Internet [3].

Comandos AT (*Attention*)

Estos comandos son utilizados para establecer una configuración, comunicación y conexión con módems tradicionales. En la actualidad los teléfonos móviles son capaces de comportarse como módems por lo que aceptan estos comandos, al ser fáciles de implementar en varios sistemas de comunicación lo utilizan en la actualidad [4].

Arduino

Para el presente proyecto se utilizó la placa de procesamiento Arduino con su respectivo lenguaje de programación en plataforma de Arduino IDE. Arduino es una plataforma de código abierto que posee placas con un microcontrolador la cual proporciona la interacción con ella.

Se lo puede comparar a un ordenador el cual puede interactuar con distintos sensores y actuadores con el fin de ejecutar proyector en el área de electrónica, estos pueden ser para un fin estudiantil, universitario o laboral, cada placa cuenta con un puerto de conexión que permite escribir el programa desde un ordenador [5].

Entorno de programación Arduino IDE

Esta aplicación permite editar las instrucciones las cuales realizará la placa de Arduino, el lenguaje de programación que es usado está basado en C++. Hay que tener en cuenta que para escribir un código de programación usando este software se debe seguir ciertas reglas para un funcionamiento correcto. Ver Anexo 1.

AJAX

Es un término usado para describir el uso de varias tecnologías existentes las cuales se utilizan de manera conjunta entre las cuales se incluyen: HTML, XHTML, CSS, JavaScript, DOM, XML, XSLT. Logrando aplicaciones web capaces de actualizarse de manera constante evitando cargar la página continuamente [6].

2 METODOLOGÍA

2.1 Descripción de la metodología usada

Se realizó una investigación de módulos de geo-posicionamiento GPS para su uso en el proyecto de titulación para los cuales se seleccionaron entre un módulo SIM808 y SIM900, por costes se optó por el módulo SIM808, una vez seleccionado el modelo se procedió a buscar una fuente de alimentación conveniente para un ambiente exterior, que satisfagan las demandas de los circuitos a utilizarse, la elección es el uso de baterías de Li Ion Lito recargables adaptadas para el hardware y alimentación del Arduino que ejecutara la parte de software, se seleccionó una placa Arduino Nano, por sus cortas dimensiones con ello ahorrando espacio para la parte del armado.

Una vez seleccionado el módulo SIM808, placa Arduino Nano se procedió a buscar baterías que mejor se adapten para ser usadas en el sistema móvil, de tal manera que satisfaga las demandas de voltaje y corriente de los circuitos utilizados, por ello se eligió baterías de Li Ion Lito recargables con una capacidad de 3100 (mA) y 3.7 (V) cada una.

Tomando en cuenta las dimensiones máximas tanto del Arduino Nano como del módulo Sim808 EVB V3.2, se realizó la compra de arneses para mascotas los cuales soporten los equipos de geo-posicionamiento.

El arnés toma en cuenta la movilidad de la mascota, evitando así una posible incomodidad al transportar el dispositivo, el hardware de geo-posicionamiento ocupa el menor espacio posible siendo el módulo SIM808 el componente con mayores dimensiones.

Se usó Microsoft Azure el cual permitió alojar una página web desarrollada en *HTML5* y JavaScript, se creó una base de datos para usuarios y mascotas usando *MySQL* misma que junto a la página web fueron subidas a *Microsoft Azure*.

Haciendo uso de la plataforma *ThingSpeak* se procedió a captar datos de geo-posicionamiento arrojados por el Sim808, estos datos son enviados a la página web de manera que pueden ser visualizado mediante la API de Google *Maps*.

Se efectuó la programación de la placa Arduino NANO, la cual se comunica con el módulo SIM808, este módulo recepta información mediante el uso de su antena GPS, esos resultados son guardados para ser enviados al servidor de *ThingSpeak.*

La antena GPS está integrada en el SIM808, envía la información de las coordenadas de geo-posicionamiento a la placa Arduino, información que es guardada y enviada hacia el servidor *ThingSpeak*, estos datos viajan por medio de la red celular GPRS.

Una vez instalado el sistema de geo-posicionamiento se realizaron pruebas de toma de datos de posicionamiento, gestión de batería, precisión y monitoreo de la mascota. Se realizaron los cambios necesarios para que el sistema cumpla con las expectativas requeridas.

3 RESULTADOS Y DISCUSIÓN

El uso de una placa Arduino Nano junto a un módulo SIM808 EVB 3.2 son los componentes más relevantes del circuito, la placa Arduino Nano ejecuta un código con el cual se realiza el control del módulo SIM808.

Este código se encarga tanto de encender el módulo SIM808, verificar que se encuentre activa la bandeja para la SIM CARD, una vez que el módulo SIM808 esté activo y funcionando con normalidad procede a comunicarse con la placa Arduino.

Mediante los comandos AT, el módulo SIM808, solicita la ubicación al GPS, el cual envía información sobre: tiempo, fecha y posicionamiento. Cuando el Arduino posee esta información toma los valores numéricos del posicionamiento los cuales son longitud y latitud para guardarlos en variables.

Para enviar estos valores a la base de datos *ThingSpeak*, se lo hace usando comandos AT y usando la red celular GPRS, estos comandos ingresan el *API KEY* para entablar la conexión entre Arduino y *ThingSpeak*.

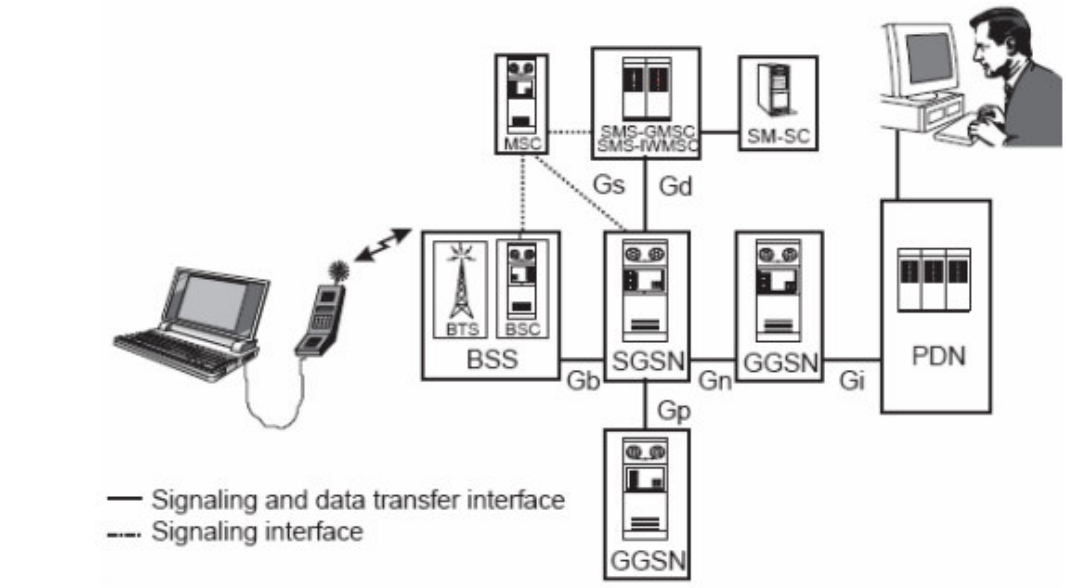
3.1 Diseño de un sistema de geo-posicionamiento para mascotas

Recepción, Almacenamiento y Envío de Datos de Geo-posicionamiento

Funcionamiento de una red GPRS

Se realizó el respectivo análisis sobre el funcionamiento del sistema GSM/GPRS, esta red permite la conexión de un dispositivo móvil a Internet, enviando una señal hacia el servidor usando las antenas BSS *Base Station Sub-system* se efectúa una comunicación con una SGSN *Serving GPRS Support Node*.

Este nodo SGSN permite la conexión de varias BSS con la puerta de red GPRS conocida como GGSN *Gateway GPRS Support Node,* para finalizar se realiza la conexión con el equipo PDN *Public Digital Network,* de esta manera con el uso de una APN *Acces Point Name* dada por el proveedor de red celular se puede realizar una conexión hacia el servicio de Internet. [7] La Figura 3.1 muestra un diagrama de conexión de los componentes que son parte de una red GPRS.



MSC: *Mobile Switching Center*
BTS: *Base transceiver Station*
BSC: *Base Station Controller*
BSS: *Base Station Sub-system*
SMS-GMSC: *Short Message Service-Gateway Mobile-services-Switching Center*
SMS-IWMSC: *Short Message Service-Interworking MSC*
SGSN: *Serving GPRS Support Node*
GGSN: *Gateway GPRS Support Node*
PDN: *Public Digital Network*

Figura 3.1 Red GPRS [7]

Este mismo principio se lo puede aplicar al uso del SIM808, al poseer un compartimento para una tarjeta SIM, y una antena GSM tiene la posibilidad de realizar este tipo de enlace utilizando la red de telefonía celular.

Placa Arduino Nano

Tomando en cuenta que el sistema de geo-posicionamiento será usado por una mascota es necesario que este sea compacto y robusto, por este motivo se usó una placa Arduino NANO, esta placa cuenta con características útiles para la recepción de información y control, debido a su tamaño, su número de pines, almacenamiento suficiente para el código y compatibilidad con distintos módulos. Esta placa es completa y amigable, basada en ATmega328P [8].

Las partes mencionadas anteriormente se las visualiza en la Figura 3.2.

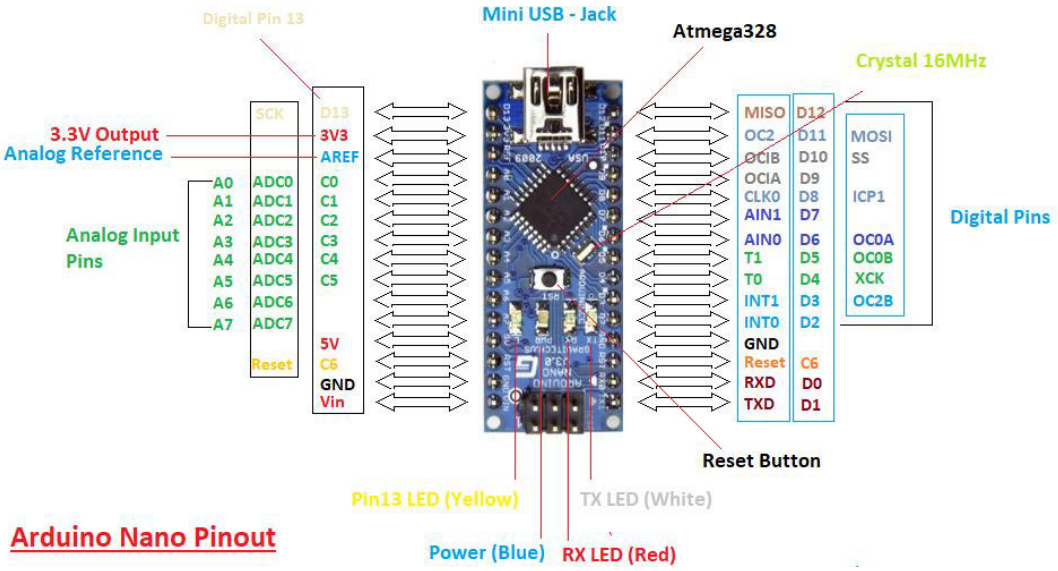


Figura 3.2 Pines Arduino Nano [9]

Para ver las características completas de la placa Arduino NANO estas se encuentran en el Anexo 2.

En la Tabla 3.1 se encuentran las características generales con las que cuenta la placa Arduino Nano.

Tabla 3.1 Características generales de Arduino Nano [10] [9]

Características	Valor
Microcontrolador	ATmega 328P
Voltaje de funcionamiento	7 (V)
Corriente DC por pin (recomendado)	20 (mA)
Largo	45 (mm)
Ancho	18 (mm)
Peso	7 (g)

La placa seleccionada para el presente proyecto de titulación se eligió por las características más convenientes, entre ellas se tiene: número de pines digitales, tamaño de la placa, voltaje de operación, peso, información para uso de la placa, precio accesible. Se realizó una comparación entre los modelos más comunes de placas Arduino, esta se presenta en la Tabla 3.2.

Tabla 3.2 Características Modelos de Arduino

Características Modelos Arduino		
Modelo	Arduino Uno	Arduino Nano
Número de pines digitales	14	14
Dimensiones	80 * 55.1 (mm)	45 * 18 (mm)
Voltaje de alimentación	7 (V)	7 (V)
Peso	41.75 (g)	7 (g)
Documentación del modelo	Sí	Sí
Precio	15 $	7 $

Módulo SIM808 GPS/GSM EVB V3.2

Para captar datos de geo-posicionamiento, latitud y longitud, se optó por utilizar un módulo compatible con la placa Arduino NANO el cual es el Módulo SIM808 EVB V3.2. Este módulo puede comunicarse, ser controlado y configurado por medio de la placa Arduino, esto se lo realiza por medio de UART y comandos AT, con ello solo se debe establecer la conexión física entre el Arduino y el módulo.

Se realizó una investigación de los módulos de geo-posicionamiento GPS que serán utilizados en el proyecto para los cuales se seleccionaron entre un módulo SIM808 y SIM900, por costo y características similares se optó por el módulo SIM808. Una comparación de estos dos módulos se presenta en la Tabla 3.3

Tabla 3.3 Características de módulo de Geo-posicionamiento

Rasgos buscados para Módulo de Geo-posicionamiento		
Tipo de modelo	SIM808	SIM900
Compatible con Arduino	Si	Si
Envío y recepción de paquetes GPRS (TCP/IP)	Si	Si
Cuatri banda 850/900/1800/1900 (MHz)	Si	Si
Voltaje de funcionamiento 7 (V)	Si	Si
Dimensiones	81 * 58 (mm)	85 x 57 (mm)
Incluye antenas GPS/GPRS	Si	Si
Precio	45 $	60 $

Entre las aplicaciones que se usó de módulo SIM808 se tiene: la conexión de las placas Arduino a una red GPRS, comunicación recursiva, envío de mensajes, sistemas IoT *Internet of Things*, puntos de control y ubicación, entre otros. Para la conexión a la red

GPRS este módulo trabaja en las frecuencias 850/900/1800/1900 (MHz) y cuenta con una ranura para insertar una SIM CARD y pines extras para poder realizar configuraciones adicionales [11].

El módulo SIM808 EVB V3.2 posee características que facilitan la interacción con él, entre las más notorias son:

- Pose conectores SMA, con estos se realiza la conexión de la antena celular y la antena GPS haciendo uso de cable coaxial.
- Led de encendido, el led (PWR) indica si la placa se encuentra energizada.
- Comunicación UART, protocolo para poder comunicarse con la placa Arduino, la velocidad de transmisión se encuentra en un rango de entre 120 y 115200 baudios por segundo.
- *Netligth led*, Indica la actividad del chip usado en el SIM 808, el parpadeo rápido indica que está buscando la red celular, el parpadeo cada 3 segundos indica que encontró la red celular, si está apagado no se encuentra activo el chip del SIM 808.
- Para realizar la alimentación del SIM, existen tres formas diferentes, la primera con el jack de alimentación la cual soporta entre 5 (V) a 12 (V), esta se puede activar con un interruptor, la segunda opción es usar los pines V_{IN} y GND y realizando la conexión al Arduino, la tercera opción es usar los pines de 3.5 (V) a 4 (V) para conectar una batería de Ion Litio.
- Ranura chip, acepta tarjetas chip de tamaño 2G (Figura 3.3).

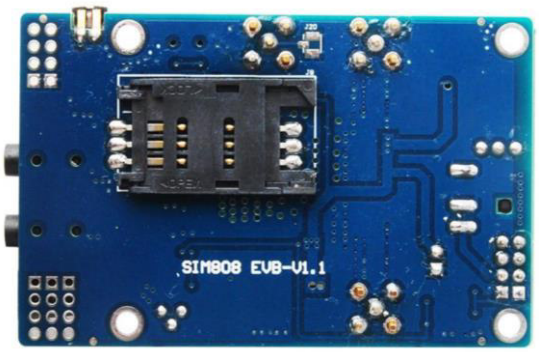


Figura 3.3 Ranura chip 2G

La ilustración de los componentes descritos se encuentra en la Figura 3.4.

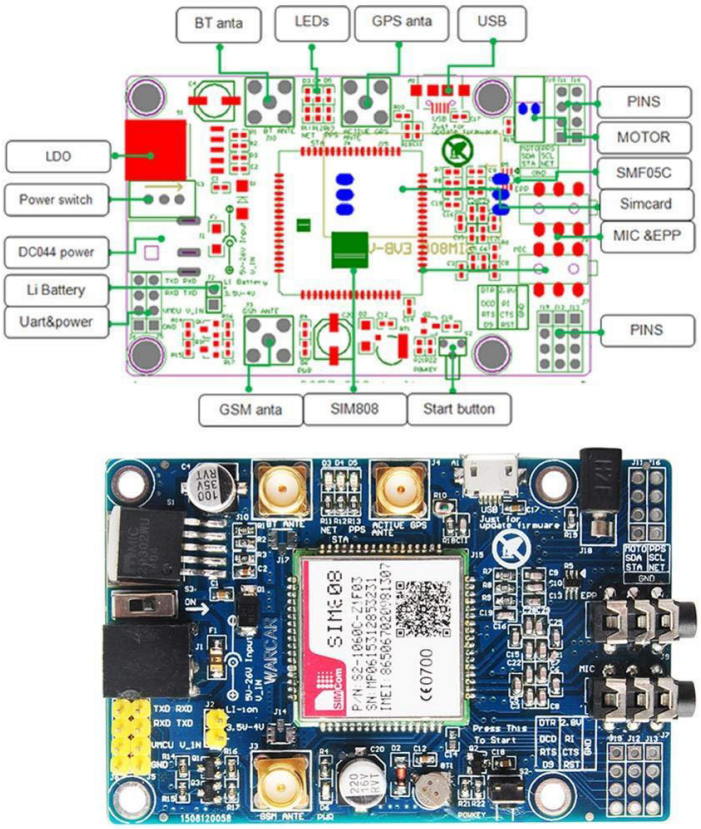


Figura 3.4 Componentes módulo SIM808 [12]

Las características que se necesitan para este proyecto usando el módulo SIM808 [13] se encuentran en la Tabla 3.4.

Tabla 3.4 Características del módulo SIM808 GSM [15]

Características del módulo GSM/GPRS	
Voltaje de operación	7 (V)
Puerto de comunicación	Serial UART
Bandas de frecuencia	850/900/1800/1900MHz
Corriente	1.5 (mA) Sleep
Dimensiones	81 x 58 x 19 (mm)
Peso	76 ± 2 g

En el Anexo 3 se puede observar datos técnicos del módulo SIM808 EVB V3.2.

Por consiguiente, primero el sistema funciona de manera que el Arduino envía comandos AT al módulo SIM808 para la captación de longitud y latitud, datos que serán almacenados en Arduino. Segundo, se envían nuevos comandos AT al módulo para la preparación y transmisión de estos mediante la red GPRS hacia el servidor.

Alimentación eléctrica del sistema

Debido a que algunas mascotas suelen perderse por varias horas, la duración de la batería debe ser un requisito importante, se estimó que dicha duración debe ser mayor a 8 horas, por ello la capacidad considerada de la batería es de 3100 (mAh), el tiempo de duración se lo calculó conectando un multímetro al módulo Sim808 y la placa Arduino, cuyos valores más altos fueron de 420 (mA) y 100 (mA) respectivamente, estos valores se obtuvieron mientras se realizaba el envío de datos. Los cálculos realizados se muestran a continuación:

- Primero se calculó el tiempo de duración para la primera corriente, la cual se obtiene por medio de la ecuación:

$$t1 = \frac{\text{Carga de la bateria}}{I1}$$

Ecuación 1 Cálculo de duración de la batería primera carga

Donde:

Carga de la batería	:	3100 (mAh) capacidad de carga batería
I1	:	420 (mA) corriente consumida por Sim808
t1	:	(h) duración en horas

Usando la Ecuación 1 se obtiene:

$$t1 = 7.4\ (h)$$

- Segundo, se realiza el mismo cálculo para la segunda corriente usando la ecuación:

$$t2 = \frac{\text{Carga de la bateria}}{I2}$$

Ecuación 2 Cálculo de duración de la batería segunda carga

Donde:

Carga de la batería	:	3100 (mAh) capacidad de carga batería
I2	:	100 (mA) corriente del Arduino Nano
t2	:	(h) duración en horas

Usando la Ecuación 2 se obtiene:

$$\mathrm{t2} = 31\ (\mathrm{h})$$

- Tercero, se deben tener los valores ideales, para ello se utiliza la ecuación:

$$\text{tiempo ideal} = \frac{\mathrm{t1} + \mathrm{t2}}{2}$$

Ecuación 3 Cálculo de duración ideal de la batería

Donde:

$\mathrm{t1}$	:	tiempo en horas primera carga (SIM808)
t2	:	tiempo en horas segunda carga (Arduino Nano)
tiempo ideal	:	duración en horas (h) de la batería en condición ideal

Usando la Ecuación 3 se obtiene:

$$\text{tiempo ideal} = 19.2\ (\text{Horas})$$

- Para finalizar las baterías no deben tener una descarga por debajo del 71% para calcular el tiempo real de duración de la batería en el sistema se utiliza la ecuación:

$$\text{tiempo real} = (\text{tiempo ideal}) * \%$$

Ecuación 4 Cálculo de duración tiempo real de la batería

Donde:

tiempo ideal	:	tiempo ideal del sistema en horas
%	:	valor mínimo de descarga “0.71”, dado por el fabricante
tiempo real	:	duración de batería dado en horas y minutos

Usando la Ecuación 4 se obtiene:

$$\text{tiempo real} = 13\ \text{horas y}\ 6\ \text{minutos}$$

Para alimentar el sistema y que este tenga una operación óptima, debe trabajar con un voltaje recomendado de 7 (V), valor recomendado en las hojas de datos respectivas [13]. Para obtener este voltaje se lo realiza por medio de baterías recargables “Li Ion LITIO 18650”, este modelo de baterías puede ser visto en la Figura 3.5 [14].

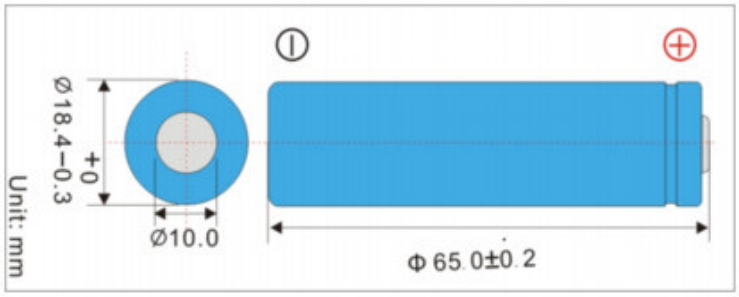


Figura 3.5 Batería 18650 [15]

Cada una de las baterías entrega un voltaje de entre 3.7 (V) a 4.2 (V), dependiendo del modelo, la capacidad de la batería puede ir desde los 2000 (mAh) hasta los 4000 (mAh), para este proyecto de titulación los valores son de 3100 (mAh), la media para recargar estas baterías es de 600 a 1000 veces, además que poseen auto descarga nula [16].

En el Anexo 4 se puede observar datos técnicos de las baterías usadas para el sistema construido.

Un factor importante es el tamaño de la mascota, por lo cual se diseñó dos sistemas de alimentación, uno haciendo uso de dos baterías y el otro de tres baterías.

Para mascotas de gran tamaño se realizó el sistema con tres baterías, se le dio una configuración en serie, con ello se suman los voltajes, cada batería posee 3,7 (V) dando en total 11.1 (V) aproximado a 12 (V) esta configuración se encuentra visible en la Figura 3.6, con esta configuración se puede obtener el voltaje de trabajo adecuado para el Arduino Nano y módulo SIM808.

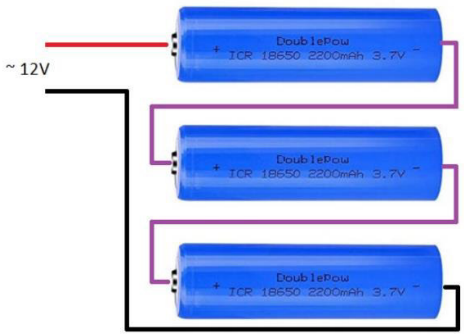


Figura 3.6 Conexión serie 3 Baterías 18650

Para mascotas de mediano tamaño se realizó el sistema de dos baterías, se usó la misma configuración en serie sin embargo el voltaje sumado es 7.4 (V), en la Figura 3.7 se puede observar la configuración realizada, en esta instalación se puede obtener un valor muy cercano al voltaje de trabajo recomendado para el Arduino Nano y módulo SIM808, con esta cantidad de baterías disminuye el peso del prototipo.

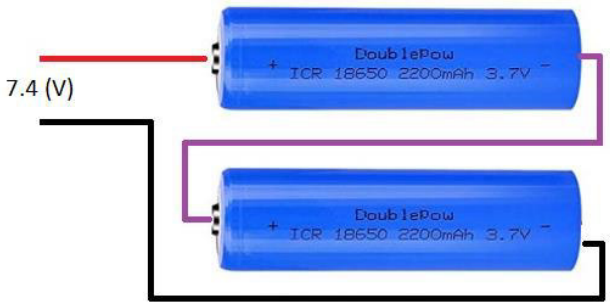


Figura 3.7 Conexión serie 2 Baterías 18650

Para tener un control del voltaje para el Arduino y módulo SIM808 se usa un módulo Step Down DC/DC LM2596 visible en la Figura 3.8, este se encarga de brindar un voltaje constante a la salida, para ello usa un integrado LM2596, entre sus características más destacables son: soportar un voltaje de entrada que va entre 4.5 (V) a 40 (V) (DC), da un voltaje de salida que puede ir entre 1.2 (V) a 37 (V), también soporta corrientes de salida hasta los 3 (A) [17]. Este convertidor DC/DC trasforma un voltaje mayor a uno menor, esto se lo realiza mediante el manejo del potenciómetro que trae incluido.

Siguiendo las recomendaciones que se encuentran en las hojas técnicas de Arduino y el módulo SIM808, se usó este conversor para tener un voltaje constante de 7 (V) en el sistema.

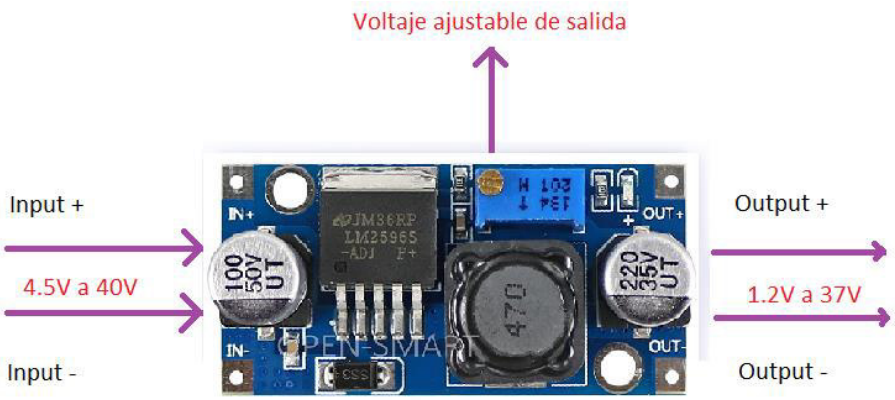


Figura 3.8 Módulo Step Down LM2596 [18]

En la Tabla 3.5 se puede observar las principales características que posee el módulo LM2596.

Tabla 3.5 Características del módulo Step Down LM2596

Características del módulo Step Down LM2596	
Convertidor DC-DC Buck	LM2596
Voltaje de entrada	4.5 (V) a 40 (V) DC
Voltaje de salida	1.23 (V) a 37 (V) DC
Corriente de Salida	máx. 3 (A), 2.5 (A) recomendado, usar disipador para corrientes mayores a 2 (A).
Eficiencia de conversión	92%
Protección de corto circuito	Hasta 5 (A)
Dimensiones	43 (mm) * 21 (mm) * 13 (mm)

Para ver las características completas del módulo Step Down LM2596 ver Anexo 5.

Para la alimentación de voltaje de la placa Arduino Nano se lo realizó de la siguiente manera:

- Pines V_{IN} (Voltaje) & GND (Ground), puede ser alimentada de entre 6(V) a 12(V), lo recomendado es 7(V).

Cargador de batería

Para que el sistema de carga sea más cómodo para el usuario se optó por un Balanceador de Cargas para Baterías conocido como BMS, este es un dispositivo que permite cargar baterías de Li Ion 18650.

Dependiendo de la necesidad del sistema se decidió por dos modelos: uno para dos baterías que se usa en mascotas medianas para lo cual se usó el módulo BMS 2S 10 (A) el cual es visible en la Figura 3.9.

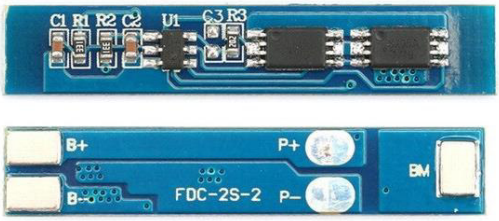


Figura 3.9 Cargador de 2 baterías 18650 Módulo BMS 2S 10 (A)

El segundo sistema de carga es para mascotas grandes, el cual usa el módulo de carga de tres baterías, BMS 3S 10 (A) apreciable en la Figura 3.10.

Figura 3.10 Cargador de 3 baterías 18650 Módulo BMS 3S 10 (A)

Estos dos módulos evitan que el voltaje de las baterías de Li Ion al descargarse caigan por debajo de las especificaciones técnicas las cuales son 3 (V) y que el voltaje de carga sea mayor al recomendado 4.2 (V) por cada batería [19].

Para realizar la conexión de las baterías de Litio 18650 al módulo BMS 3S 10 (A) se lo realiza mediante la configuración visualizada en la Figura 3.11, para cargar las respectivas baterías conectadas se requiere de un cargador externo de 12 (V).

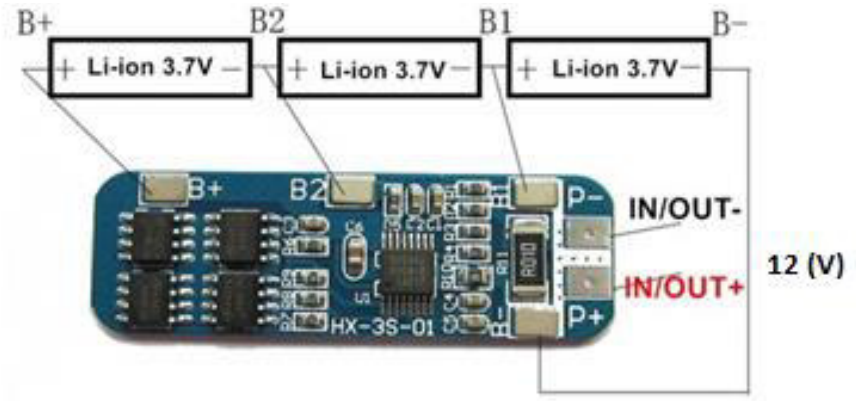


Figura 3.11 Conexión baterías18650 a BMS 3S 10A [20]

Por otro lado, la conexión de las baterías de litio 18650 al módulo BMS 2S 10 (A) se lo realiza mediante la disposición visible en la Figura 3.12 [21]. Para cargar las baterías conectadas a este módulo requiere de un cargador externo de 8 (V).

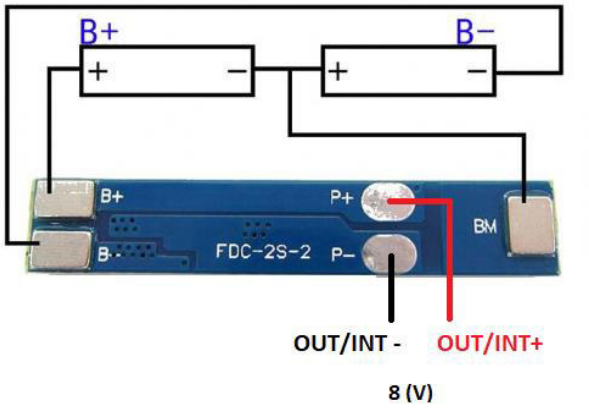


Figura 3.12 Conexión baterías18650 a BMS 2S 10A

En la Tabla 3.6 encuentran características seleccionadas para hacer uso del módulo BMS 3S 10A.

Tabla 3.6 Características del módulo BMS 3S 10A [21]

Características del módulo BMS 3S 10A	
Modelo	HX-3S-FL10A-A
Vida útil	>30.000 horas
Dimensiones	50 (mm) x 15 (mm) x 2.2 (mm)
Peso	3 (g)
Voltaje de carga	12.6 (V) – 13(V)
P+	Voltaje (+) de entrada / salida a carga
P-	Voltaje (-) de entrada / salida a carga
B-	0 (V) terminal negativa
B1	3.7 (V) a batería
B2	7.4 (V) a batería
B+	11.1 (V) a batería

Características adicionales sobre los módulos BMS 3S 10 (A) y BMS 2S 10 (A) se presentan en el Anexo 6.

3.2 Desarrollo de interfaz para la localización de mascotas con servidor WEB

Base de datos *ThingSpeak*

En el presente proyecto de titulación para evitar costos adicionales se utiliza el servidor gratuito *ThingSpeak* para guardar y visualizar la información del geo-posicionamiento enviada por el Arduino. Con el uso de este servidor se tiene la posibilidad de procesar los datos usando Matlab, caso contrario, por medio de su *API KEY* permite que una aplicación externa acceda a las coordenadas guardadas, en el caso del sistema se utilizó la segunda opción. El Arduino mediante comandos AT se encarga de controlar el SIM808 activando el sistema GPRS para el envío de la ubicación hacia *ThingSpeak*.

Los pasos que se requieren para poder realizar la transferencia de las coordenadas hacia *ThingSpeak* deben seguir un orden determinado para evitar errores en el envío, a continuación, se los enumera:

- En primer lugar, se realizar una consulta para observar el estado actual de la conexión para ello se usa el comando *AT+CIPSTATUS* visible en la Figura 3.13, este comando devuelve el estado del servidor aplicable, el estado del cliente, el número de conexión para IP y la información del portador GPRS.

```
71 Sim800Serial.println("AT+CIPSTATUS");/
72 delay(2000);
```

Figura 3.13 Comando AT+CIPSTATUS

- En segundo lugar, es necesario obtener una dirección IP esta puede ser única o múltiple, para obtener dicha dirección se usa el comando *AT+CIPMUX=0* el cual se lo puede observar en la Figura 3.14, para este trabajo de titulación es necesario que la dirección IP sea estática, por ello tiene un valor de “0”.

```
73 Sim800Serial.println("AT+CIPMUX=0");
74 delay(3000);
```

Figura 3.14 Comando AT+CIPMUX

- En tercer lugar, se procede a iniciar la conexión GPRS para esto se utiliza el comando *AT+CGATT=1* visible en la Figura 3.15, este comando se utiliza para

asociar o separar el dispositivo al servicio de dominio de paquetes, el valor de 1 implica que está conectado.

```
76 Sim800Serial.println("AT+CGATT=1")
77 delay(3000);
```

Figura 3.15 Comando AT+CGATT=1

- En cuarto lugar, se configura el APN respectivo para el acceso a la red móvil, se utiliza el comando *AT+CSTT=\"igprs.tuenti.com.ec\",\"ctigprs\",\"ctigpr999* visible en la Figura 3.16, el APN depende de cada operadora de telefonía móvil.

```
79 Sim800Serial.println("AT+CSTT=\"igprs.tuenti.com.ec\",\"ctigprs\",\"ctigpr999\"");
80 delay(1000);
```

Figura 3.16 Conexión con operadora celular

- En quinto lugar, se inicia la conexión GPRS para ello se usa el comando *AT+CIICR* el cual se lo observa en la Figura 3.17, esto depende de la configuración establecida previamente.

```
82 Sim800Serial.println("AT+CIICR");/
83 delay(3000);
84 mostrarDatosSeriales();
```

Figura 3.17 Inicio de conexión a GPRS

- En sexto lugar, se obtiene una dirección IP local utilizando *AT+CIFSR*, el comando usado se lo puede visualizar en la Figura 3.18, este devuelve una dirección IP local por parte del servidor móvil.

```
85 Sim800Serial.println("AT+CIFSR");
86 delay(2000);
```

Figura 3.18 Comandos AT+CIFSR obtención IP local

- En séptimo lugar, se establece un indicador para poder enviar datos, se usó AT*+CIPSPRT=0,* el uso de este comando se lo puede ver en la Figura 3.19, el símbolo que usa Arduino para indicar el inicio de datos es *">"*

```
88 Sim800Serial.println("AT+CIPSPRT=0");
```

Figura 3.19 Comando AT+CIPSPRT=0

- En octavo lugar, se inicia la conexión con el servidor de *ThingSpeak*, para ello se utiliza el puerto 80, el comando para realizar esta acción es

AT+CIPSTART=\"TCP\",\"api.thingspeak.com\",\"80, visible en la Figura 3.20, esta comunicación se lo realiza por medio de conexión TCP.

```
92 Sim800Serial.println("AT+CIPSTART=\"TCP\",\"api.thingspeak.com\",\"80\""
93 delay(6000);
```

Figura 3.20 Conexión al servidor ThingSpeak

- En décimo lugar, se realiza el envío de datos por medio de la conexión TCP establecida anteriormente, el comando para realizar esta acción es *AT+CIPSEND* el cual se lo puede ver en la Figura 3.21.

```
95 Sim800Serial.println("AT+CIPSEND");/
96 delay(4000);
```

Figura 3.21 Envío de datos al ThingSpeak

- En undécimo lugar, la función *String*, permite realizar el transporte de caracteres ya que los convierte en una cadena, se envía la *API KEY* junto a las coordenadas las cuales se almacenan las variables configuradas llamadas *Field*, el uso de *String* se lo puede observar en la Figura 3.22.

```
98 String datos="GET https://api.thingspeak.com/update?api_key=BRDQOASQ4BLF2SHK";
99 datos=datos+"&field1="+ String(lat);
100 datos=datos+"&field2="+ String(lon);
```

Figura 3.22 Escritura de coordenadas en ThingSpeak

- En duodécimo lugar, se envía el carácter Z para indicar que no se trasmitirá más información hacia el *ThinkSpeak*, para que este carácter pueda ser entendido se escribe en *ASCII* el valor correspondiente que es 26, esta configuración se la puede ver en la Figura 3.23.

```
104 Sim800Serial.println((char)26);   /,
105 delay(5000);//Ahora esperaremos una
```

Figura 3.23 Finaliza envío de información

- Finalmente se cierra la conexión GPRS, para ello se usa el comando *AT+CIPSHUT*, con esto se termina el ciclo de trasferencia al *ThingSpeak*, la configuración realizada se encuentra en la Figura 3.24.

```
108 Sim800Serial.println("AT+CIPSHUT");//Cierr
109 delay(5000);
```

Figura 3.24 Ciclo de transferencia terminado

Cabe recalcar que *ThingSpeak* posee licencias gratuitas y de paga, dependiendo de la escala del proyecto a realizarse se puede optar por una licencia comercial o gratuita, el desarrollo de este proyecto se realiza con una licencia gratuita [22].

Ahora se procede a verificar los valores del geo-posicionamiento en *ThingSpeak*, para ello es necesario tener una cuenta en *ThingSpeak*, se puede usar la cuenta institucional como se muestra en la en la Figura 3.25.

Figura 3.25 Acceso a *ThingSpeak*

Dentro de este servidor se tiene la pestaña *Channels* dentro de la cual se encuentra el ícono resaltado en verde *New Channel*, visible en la Figura 3.26, se debe crear un nuevo canal para poder almacenar los valores que reciba por parte del Arduino y del SIM808.

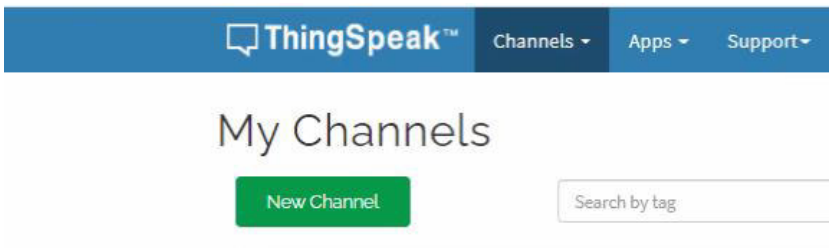


Figura 3.26 Crear nuevo canal

Es recomendable dar un nombre al canal para identificarlo en caso de utilizar más de uno, dependiendo de los datos recibidos se pueden activar cada una de las casillas

Field, para el presente proyecto de titulación es necesario guardar valores de longitud y latitud por lo cual es necesario activar el *Field 1* y *Field 2*, como lo indica la Figura 3.27.

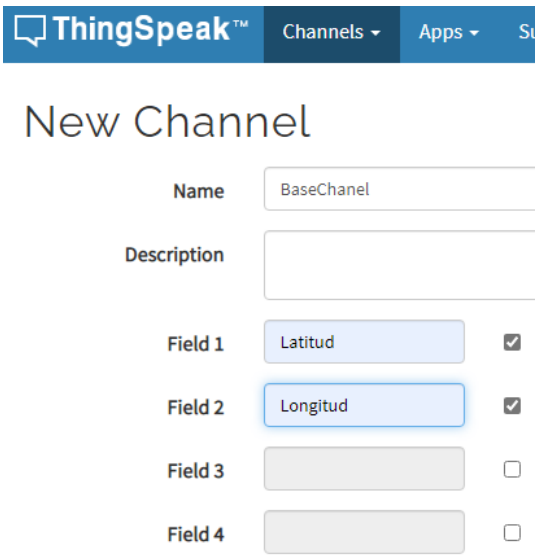


Figura 3.27 Activar *Field* para almacenamiento

Para finalizar la creación del canal se continúa llenando los espacios requeridos, esto depende de cada usuario, para el presente proyecto de titulación no es necesario, finalmente dar un clic en *Save Channel*, esto se lo puede ver en la Figura 3.28.

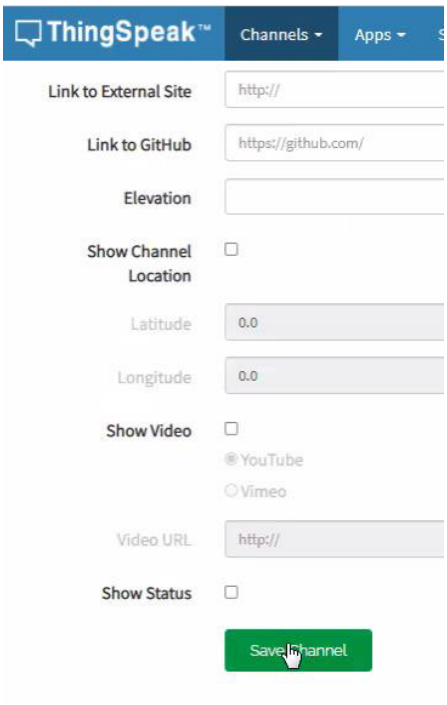


Figura 3.28 Salvar configuraciones

Culminado el paso anterior se obtiene el canal creado en el cual se guardará los valores que se envía a *ThingSpeak*, para acceder a la información almacenada se ingresó al canal como lo indica la Figura 3.29.

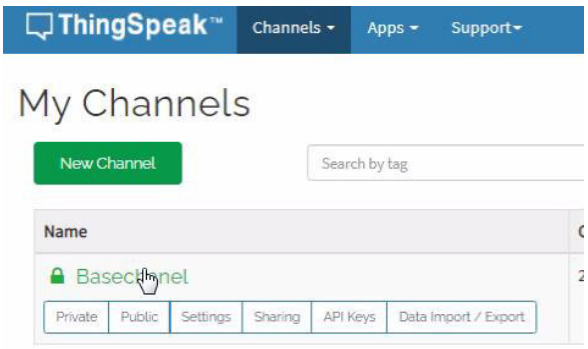


Figura 3.29 Canal creado

Este servidor muestra varias ventanas en las cuales se puede visualizar las coordenadas de latitud y longitud que han sido almacenadas, los valores de estas coordenadas son independientes, para este proyecto *Field 1* posee los datos de latitud mientras que *Field 2* posee los datos de longitud, lo mencionado anteriormente se lo puede observar en la Figura 3.30.

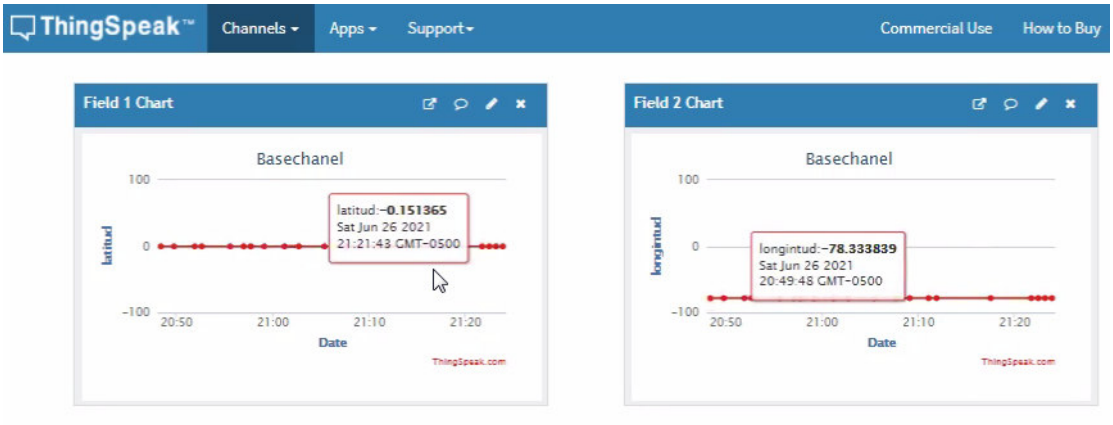


Figura 3.30 Datos latitud y longitud

Una de las características que brinda *ThingSpeak* es poder utilizar estos valores guardados en otros servidores, para tener una lectura de esta información se debe tener la *API KEY* de lectura, esta es una llave que autoriza el acceso para conocer los valores de geo-posicionamiento, así de igual forma para poder modificar los *Field 1* y *Field 2*

con nuevos resultados de posicionamiento se usa la *API KEY* de escritura, las *API KEY's* son visibles en la Figura 3.31.

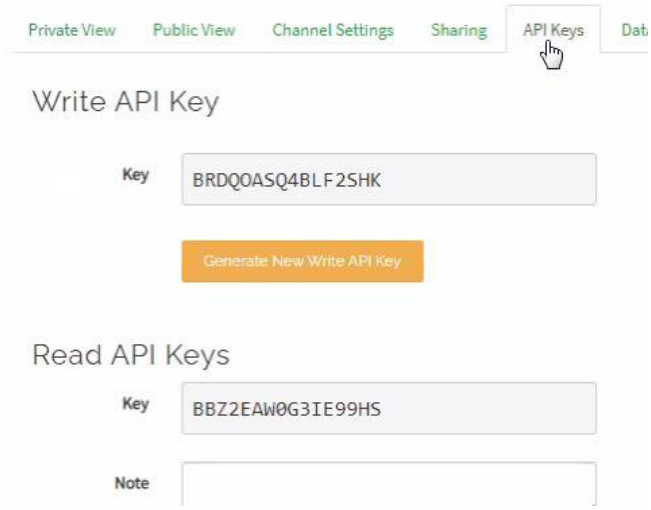


Figura 3.31 *API KEY* escritura y lectura

API *KEY Google Maps*

Una API KEY es un código de autorización que brinda permiso a una web para utilizar recursos y servicios, para el presente proyecto de titulación se usará una *API KEY* para tener permisos de Google *Maps*, para obtener una *API KEY* de Google *Maps* se procede a seguir 5 pasos.

- Primero se inicia sesión con una cuenta de Google, en caso de no poseer una cuenta se la puede crear de manera gratuita como se observa en la Figura 3.2

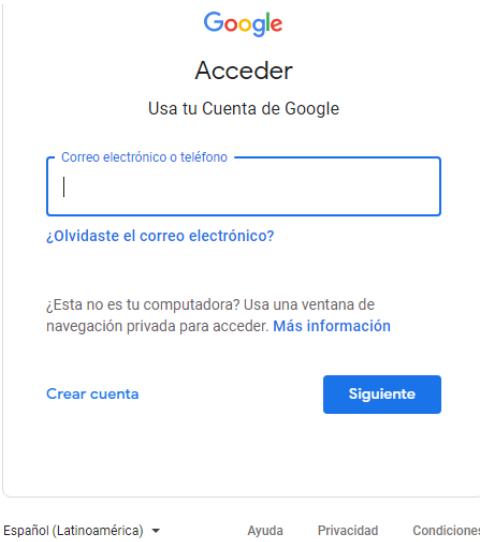


Figura 3.32 Inicio de sesión en Google

- En segundo lugar, se accede a Google Cloud y presionar clic en el recuadro azul que dice “Empezar” como lo muestra la Figura 3.33.

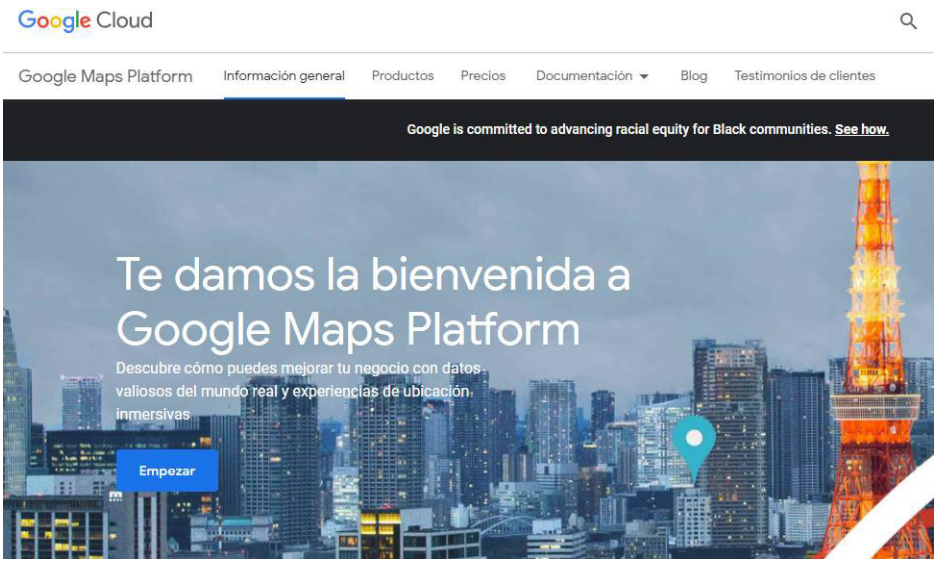


Figura 3.33 Inicio para adquirir *API KEY*

- En tercer lugar, seleccionar el país correspondiente y aceptar los términos de servicios y condiciones, posterior a esto se selecciona ACEPTA Y CONTINUAR, estas configuraciones se las puede ver en la Figura 3.34.

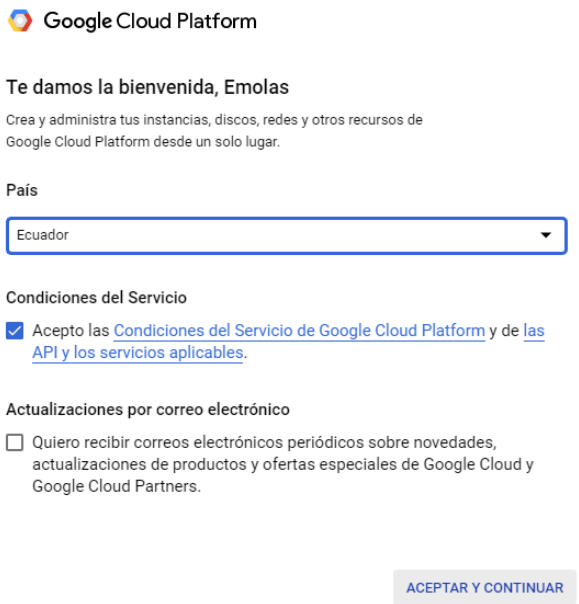


Figura 3.34 Confirmar ubicación

- En cuarto lugar, se procede a confirmar la cuenta a utilizarse, por consiguiente, hacer clic en el botón “Continuar”, como se observa en la Figura 3.35.

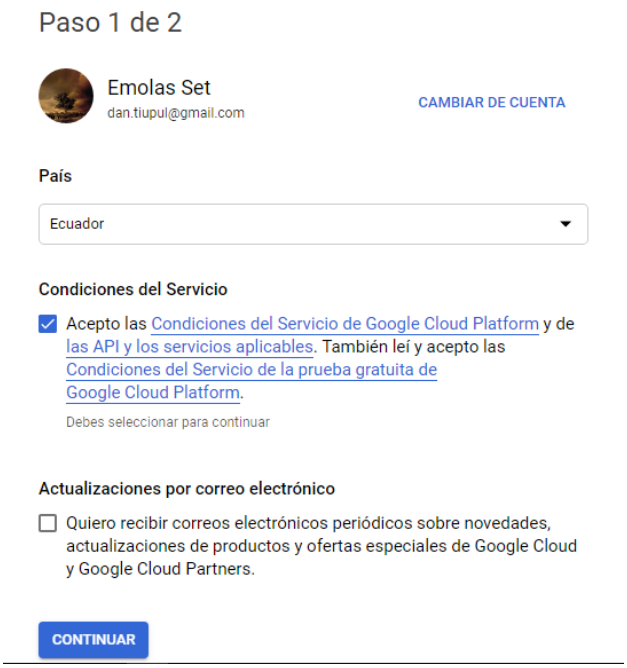


Figura 3.35 Terminar la configuración

- Para finalizar, terminado los pasos se obtiene la API KEY para usar los servicios de Google *Maps*, la API KEY obtenida se la puede ver en la Figura 3.36.

Figura 3.36 Obtención del *API KEY*

Página WEB

Para crear la página web se lo realizó utilizando el programa *Visual Studio Code*, debido a que la interfaz que presenta es amigable para el usuario como se lo ve en la Figura 3.37, adicional a esto el programa permite trabajar con varios lenguajes de programación web, los cuales se explicarán a medida que avance el documento.

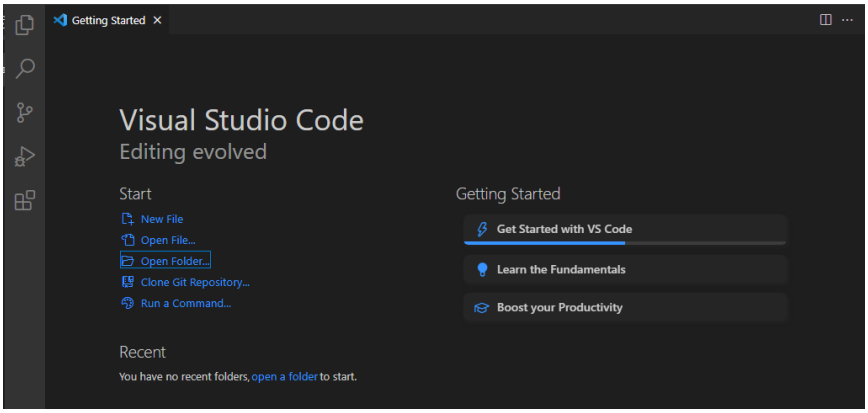


Figura 3.37 *Visual Studio Code*

Para hacer de la página web un entorno amigable con los usuarios se realizó un sistema para iniciar sesión por medio del cual los usuarios puedan registrase, este código se realizó mediante HTML para el manejo de los formularios, CSS3 para dar estilo y JavaScript para presentarlo de manera didáctica. Todos los datos fueron captados y almacenados mediante el método *POST,* que se encarga de preparar los datos para su procesamiento, la interfaz antes descrita se presenta en la Figura 3.38.

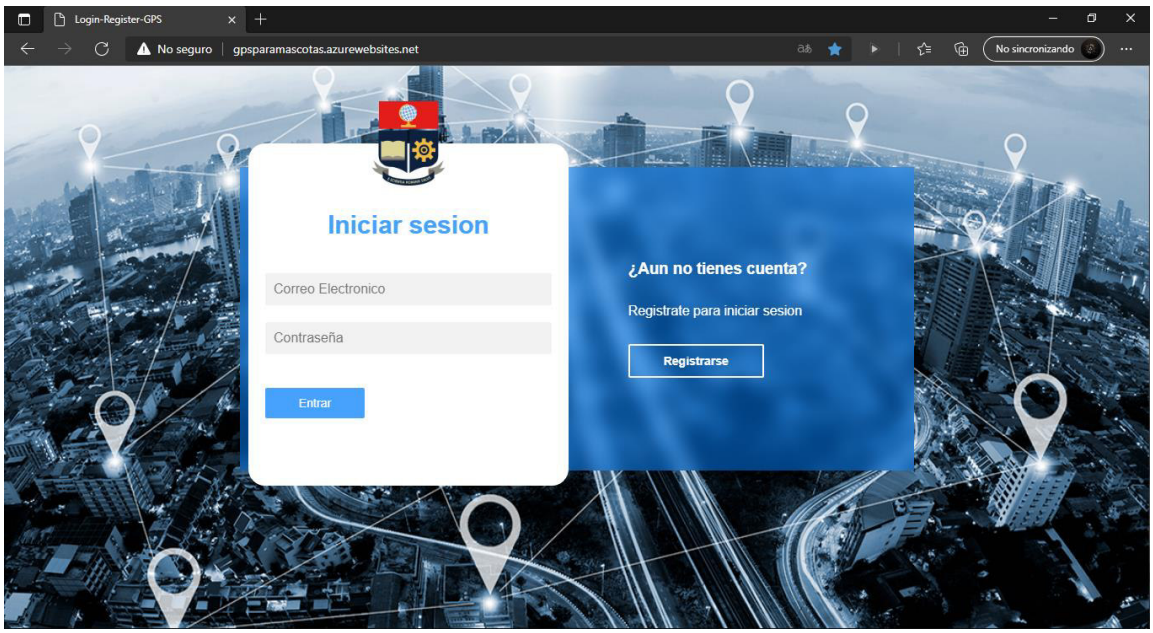


Figura 3.38 Página de *Login*

Una vez captados los datos iniciales estos deben ser almacenados, para ello se realizó una base de datos en MySQL. Dentro de esta base de datos se realizaron varias tablas, una de las cuales es la tabla usuarios cuyos campos están conformados por la unión de filas y columnas, estas se llenan con la información ingresada al momento de realizar el registro.

Los parámetros que se almacenan son: nombre, contraseña, dirección de correo electrónico, por cada fila que se llena se asigna una ID única, esta servirá como llave para el control de usuarios, esto se lo puede ver en la Figura 3.39.

Figura 3.39 PHP *My Admin*

Para la escritura, lectura y modificación de datos de MySQL se necesita un archivo de configuración que posea las credenciales necesarias para administrarlo, estas son: nombre del host, puerto, nombre de la base de datos, usuario, y contraseña. Con este registro se establece el enlace entre usuario y servidor y es llamado “Archivo de conexión” el cual se encuentra en la Figura 3.40. Este archivo se creó en PHP con la interfaz de objetos de datos PDO, la cual permitió el acceso de PHP hacia la base MySQL.

Figura 3.40 Archivo de conexión

Para que este proyecto pueda tener una conexión con el servidor Microsoft Azure debe tener un certificado que mantenga la actividad del usuario de manera privada, esto a pesar de ser una página web que no recopila información confidencial o recibe pagos, el archivo que valida esta conexión se lo puede ver en la Figura 3.41.

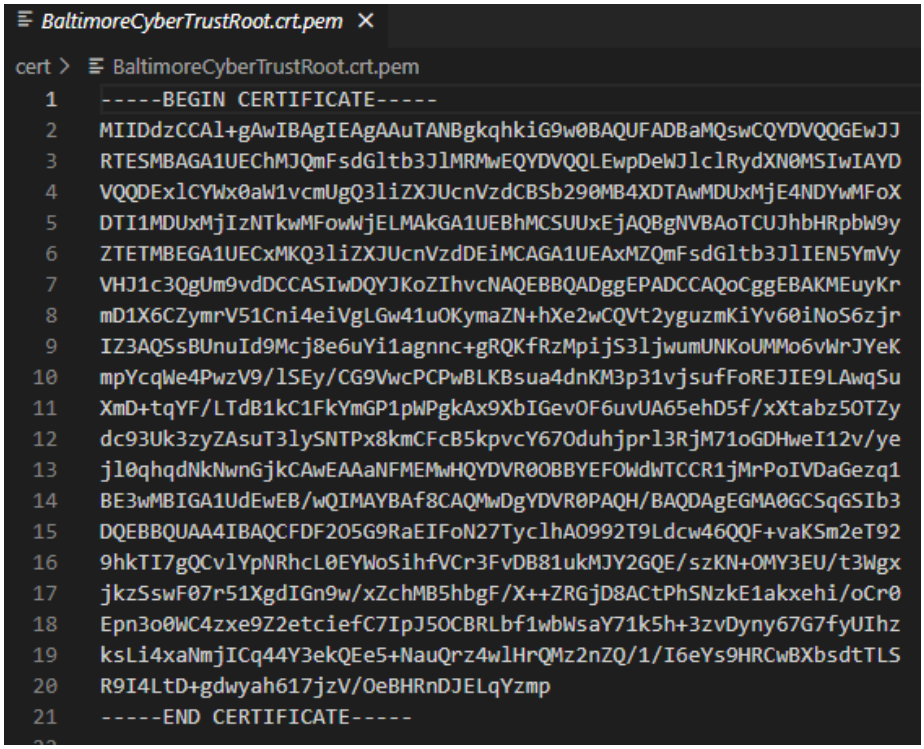


Figura 3.41 Archivo BaltimoreCyberTrustRoot.crt.pem

El código que realiza la acción de receptar y enviar la información del usuario hacia la base de datos se realizó en lenguaje PHP, este posee el archivo de conexión mostrado en la Figura 3.40, también recepta los datos por el método *POST,* el código usado está en la Figura 3.42.

```php
if(isset($_POST['nombre_completo'],$_POST['usuario'],$_POST['correo'],$_POST['contrasena'])
{
    $nombre_completo = trim($_POST['nombre_completo']);
    $usuario = trim($_POST['usuario']);
    $correo = trim($_POST['correo']);
    $contrasena = trim($_POST['contrasena']);

    $options = array("cost"=>4);
```

Figura 3.42 Método POST

Además, se asegura que no se repita información registrada y evita dejar campos vacíos durante el registro, también encripta la contraseña en *HASH*, parte del código diseñado se lo puede ver en la Figura 3.43.

```
        else
        {
            $valFirstName = $nombre_completo;
            $valLastName = $usuario;
            $valEmail = '';
            $valPassword = $contrasena;

            $errors[] = 'Email address already registered';
            echo '
            <script>
                alert("Correo ya esta registrado,Introduzca un nuevo correo");
                window.location="../index.php";
            </script>
            ';
        }
    }
    else
    {
        $errors[] = "Email address is not valid";
        echo '
            <script>
                alert("Correo no valido, Introduzca un nuevo correo");
                window.location="../index.php";
            </script>
            ';
    }
}
else
{
    if(!isset($_POST['nombre_completo']) || empty($_POST['nombre_completo']))
    {
```

Figura 3.43 Control para evitar campos vacíos

El archivo *login.php* se ejecuta de manera que en caso de haber tenido un inicio de sesión se procede a incluirse el archivo de conexión observado en la Figura 3.40, posterior a esto la clave se desencripta por medio de *HASH*, el código se lo puede ver en la Figura 3.44.

```
include("../php/conexion.php");

$txtCorreo=($_POST['correo']);
$txtContrasena=($_POST['contrasena']);
$contrasena = hash('sha512', $txtContrasena);

$sentenciaSQL=$pdo->prepare("SELECT*FROM usuarios
WHERE correo=:correo AND contrasena=:contrasena");
```

Figura 3.44 Código para inicio de sesión

Terminado esto se inicia una sentencia *PDO* la cual prepara la información de usuario y contraseña, a continuación, se procede a realizar una búsqueda y comparación con los registros guardados, si existe una coincidencia, se permite el inicio de sesión y permite identificar al usuario, caso contrario se regresa a la página principal. El código que describe esta función se encuentra en la Figura 3.45.

```php
    $sentenciaSQL=$pdo->prepare("SELECT*FROM usuarios
    WHERE correo=:correo AND contrasena=:contrasena");

    $sentenciaSQL->bindParam("correo",$txtCorreo,PDO::PARAM_STR);
    $sentenciaSQL->bindParam("contrasena",$contrasena,PDO::PARAM_STR);
    $sentenciaSQL->execute();

    $registro=$sentenciaSQL->fetch(PDO::FETCH_ASSOC);

    /*print_r($registro);*/


$numeroRegistros=$sentenciaSQL->rowCount();

if($numeroRegistros>=1){

    session_start();

    $_SESSION['usuario']=$registro;

    /*header('location:../DisePagina/blog.php');*/
```

Figura 3.45 Código login.php sentencia PDO

Una vez que el usuario inicie sesión de manera correcta, se le asigna sus respectivas credenciales y se lo redirige a la página de inicio, esta es la página principal creada en HTML, en la que se observa una presentación visible en la Figura 3.46.

Figura 3.46 Página principal

Se encuentran las opciones registro y ver mascotas registradas, en caso de pérdida se puede dirigir a *Ver ubicación* para obtener su localización en tiempo real. Estas características se las puede ver en la Figura 3.47.

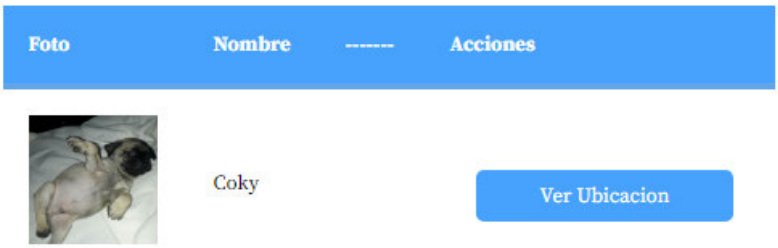


Figura 3.47 Opción ver ubicación

Si el usuario elige la opción *REGISTRO* lo traslada a la página *GPS PARA MASCOTAS*, allí se puede subir una foto, añadir un identificador único y colocar un nombre respectivamente.

La Figura 3.48 muestra el diseño de la página, el campo *ID_pet* será un identificador único que permite la vinculación con el proyecto. Igualmente, cada individuo puede modificar y eliminar según sea la necesidad.

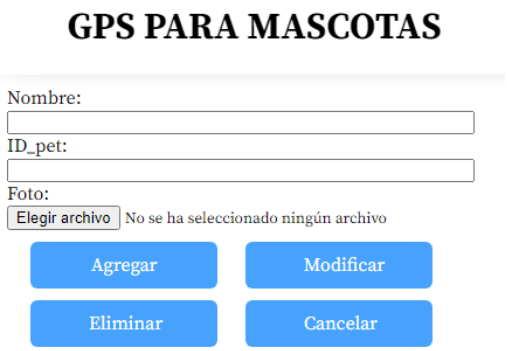


Figura 3.48 Registro GPS PARA MASCOTAS

La información de usuarios y sus mascotas registradas se encuentra guardada en la base de datos en MySQL visible en la Figura 3.49, esto se logra gracias al archivo de conexión, que también se observa en la Figura 3.40, usado con anterioridad.

Figura 3.49 Base de datos mascotas

La estructura Cardinalidad de base de datos es un modelo que permite asociar a los usuarios con sus respectivas mascotas, esto se lo consigue cuando el usuario realizó su registro, en donde se le asigna un identificador que se asocia a cada mascota que se desee registrar teniendo una relación de uno a muchos. Esto se lo puede ver en la Figura 3.50.

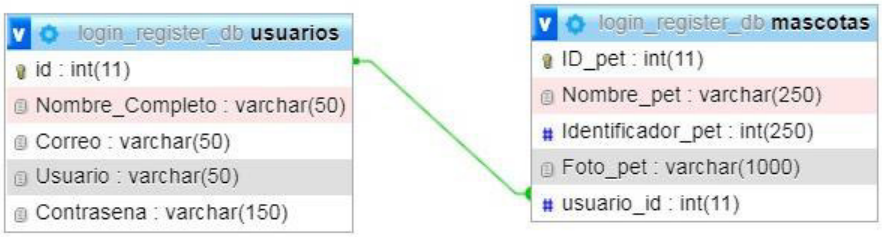


Figura 3.50 Cardinalidad usuarios y mascotas

Una vez ya registradas las mascotas en la página de inicio se puede ver la opción *Ver Ubicación,* en esta se inicia un proceso de verificación donde el identificador único debe ser validado, el código para esta acción es el identificador único visible en la Figura 3.51.

```
$txtIDpet = $_POST['txtIDpet'];
$txtNombrePet = $_POST['txtNombrePet'];
$txtIdentificadorPet = $_POST['txtIdentificadorPet'];
$idunica = 55974069;
$idunicax = 12345678;
$_SESSION['id']=$txtIdentificadorPet;



if($txtIdentificadorPet == $idunica){
    echo '
    <script>
        alert("Redirigiendo....");
        window.location ="../DisePagina/ej.php";
    </script>
    ';


}if($txtIdentificadorPet == $idunicax){

    echo '
    <script>
        alert("Redirigiendo....");
        window.location ="../DisePagina/ej1.php";
    </script>
```

Figura 3.51 Validación código único

Estos identificadores pertenecen a un solo Arduino, por ende, al ingresar una credencial no válida el acceso a la visualización será denegado, como lo indica la Figura 3.52, caso contrario se inicia un código de programación que permite obtener los datos almacenados en ThingSpeak.

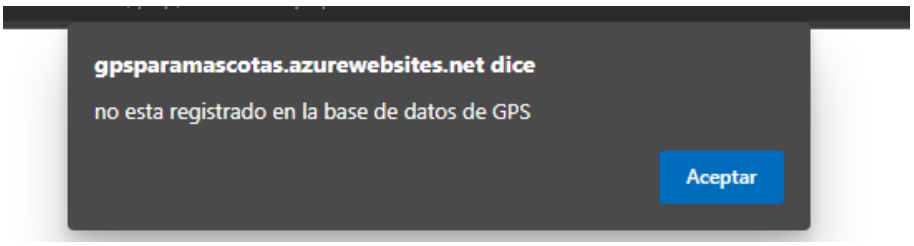


Figura 3.52 Identificador denegado

La información almacenada en *ThingSpeak* es recolectada mediante el método JavaScript *getJSON* que permite leer datos codificados en *JSON* desde el servidor mediante una solicitud *GET HTTP*, la cual se vincula mediante la *API KEY,* para este proyecto de titulación se usó la API KEY de ThingSpeak como se ve en la Figura 3.53, una vez tomados los datos de *JSON* se los almacena en variables que posteriormente serán necesarias para poder realizar la visualización en la *API* de Google *Maps*.

```
$.getJSON("https://api.thingspeak.com/channels/1004059/fields/1/last.json?api_key=BBZ2EAW0G3IE99
m1 = result;
console.log(result.field1);
x=Number(m1.field1);
console.log(x);
 iry();              //alert(x)
      });

function iry(){
  $.getJSON("https://api.thingspeak.com/channels/1004059/fields/2/last.json?api_key=BBZ2EAW0G3IE99
  m2 = result;
  console.log(result);
  y=Number(m2.field2);
  console.log(y);
  dataReady();
```

Figura 3.53 *API KEY* función JSON

Para que el método *JSON* sea implementado correctamente hace uso de librerías de la *API* de Google y *AJAX* como se lo ve en la Figura 3.54, esto permite trabajar de una manera asíncrona con las dos *API, de* Google *Maps* y *ThingSpeak*, al mismo tiempo.

```
<script src="js/scriptmap.js"></script>
<script async defer src="https://maps.googleapis.com/maps/api/js?key=AIzaSyB9R3mrFPTL8vUa2CqpbRTDg
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
<!--footer-->
```

Figura 3.54 Uso de librería AJAX

Mediante JavaScript es posible agregar marcadores, tamaño de mapa, cursores, etc. Gracias a esto se puede unir la latitud y longitud en un solo punto como se visualiza en la Figura 3.55. De esta manera se obtiene la ubicación de las mascotas registradas.

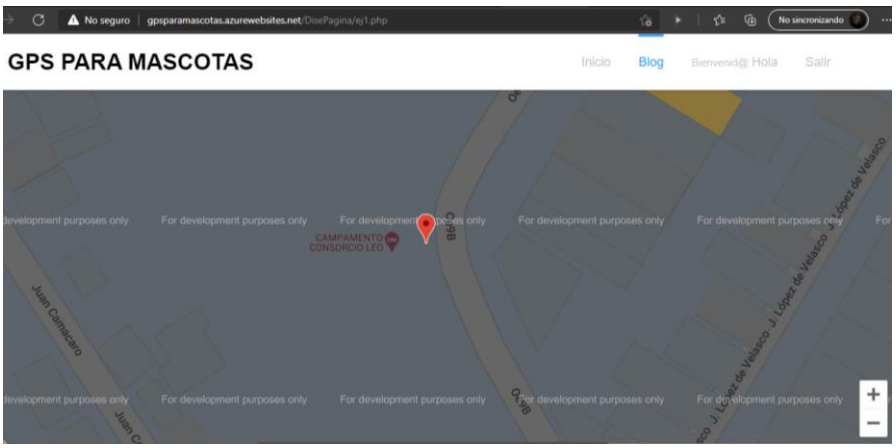


Figura 3.55 Visualización Google *Maps*

Se debe tener en cuenta que en todas las partes de navegación dentro de la página de inicio se cuenta con un encabezado que tiene la opción para cerrar sesión lo cual finaliza todas las actividades del usuario, eso se visualiza en la Figura 3.56.

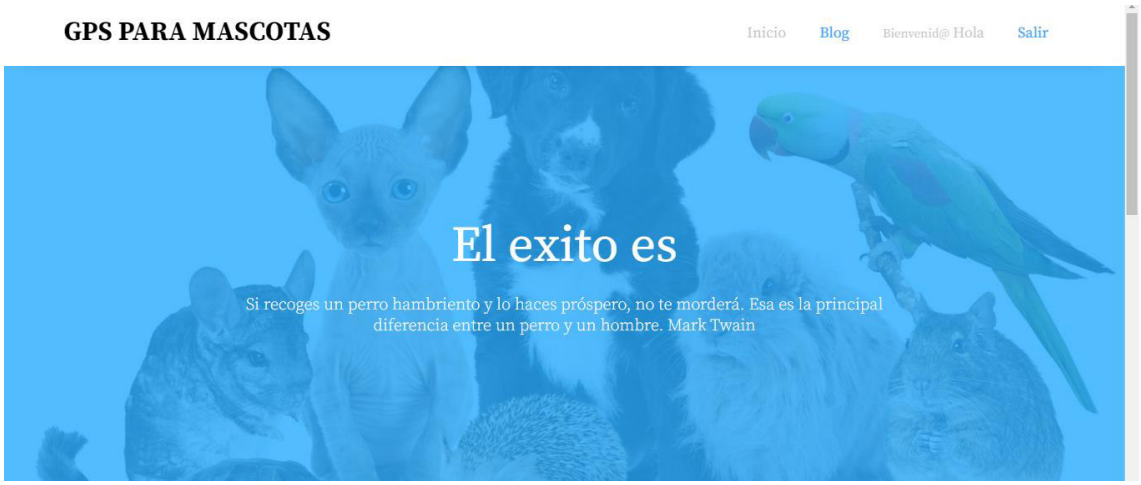


Figura 3.56 Encabezado

En caso de querer revisar toda la programación a detalle dirigirse al Anexo 7.

Servidor WEB

Para hospedar la página web creada se usa un servicio de nube tipo PaaS, este tipo de plataforma permite lanzar aplicaciones como bases de datos, herramientas de desarrollo, servicios de inteligencia, el usuario dueño de la página web solo administra las aplicaciones y servicios que desarrolla, el proveedor de servicio de nube se encarga de administrar lo demás [23].

Microsoft Azure es un servicio de nube tipo PaaS por lo que se paga solo por el uso de los servicios, no obstante, Azure provee de un crédito monetario para hacer uso de los recursos que posee esta plataforma, el crédito ofrecido se lo puede usar durante un año.

Para este proyecto de titulación se eligió la cuenta gratuita esto debido a que el proyecto no está hecho a gran escala, la página web que se creó no necesita de altos requisitos de hardware, por ello para poder subir a esta plataforma solo requiere de algunos pasos:

- En primer lugar, una vez registrado en la plataforma Microsoft Azure seleccionar la opción Portal que se ve en la Figura 3.57, dentro de esta opción se desplegará la pestaña crear recurso.

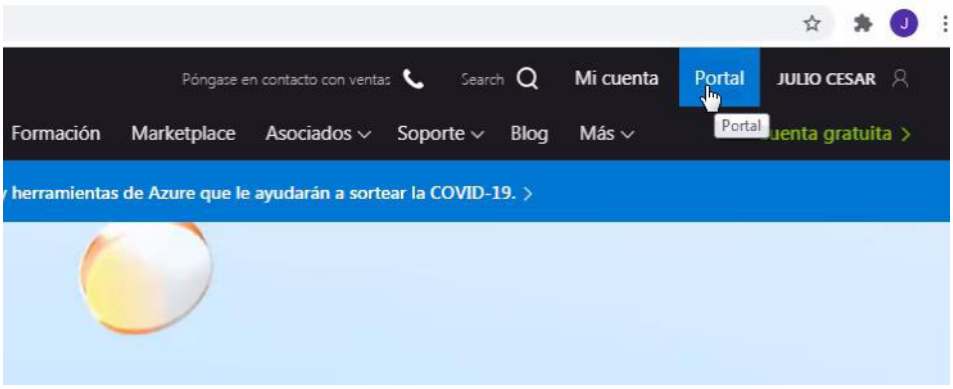


Figura 3.57 Microsoft Azure Portal

- En segundo lugar, en la pestaña “Crear un recurso” seleccionar la opción “Empezar” y seguido a esto hacer clic en la pestaña “Aplicación Web”, esto se lo puede ver en la Figura 3.58, estos pasos son para iniciar la configuración para crear la plataforma del servidor web.

Figura 3.58 Empezar Aplicación web

- En tercer lugar, se completan los datos solicitados, en la sección *Detalle de instancia* se coloca el nombre de dominio que tendrá el servidor, para este trabajo de titulación se usa la dirección [(gpsparamascotas.azurewebsites.net)] como se observa en la Figura 3.59.

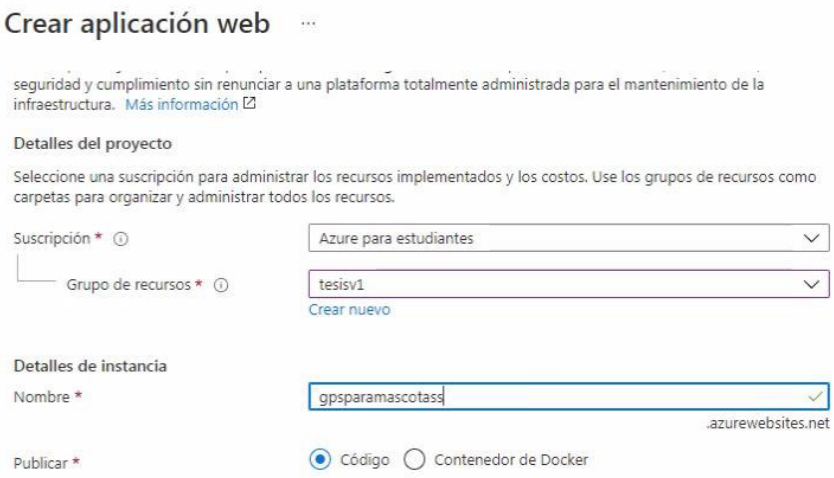


Figura 3.59 Completar datos

- En cuarto lugar, se continúa llenado los datos solicitados, la opción "Pila del entorno en tiempo de ejecución" se lo llena con *PHP 7.4*, esto debido a que la base de datos se la programó en PHP, en el ámbito "Región" se escoge el país de origen, se selecciona la opción "SKU y Tamaño", dentro de ella se encuentra la configuración para hardware del servidor visible en la Figura 3.60.

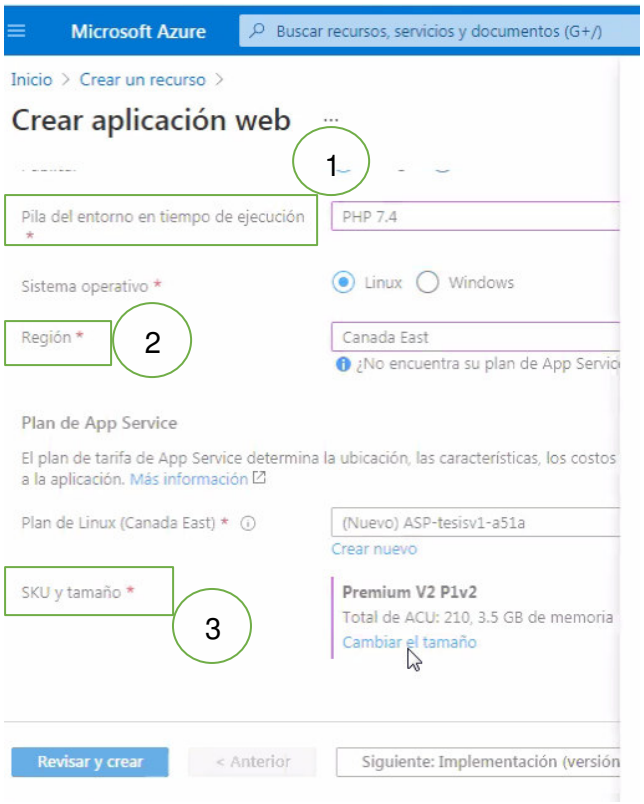


Figura 3.60 Configuraciones extras.

- En quinto lugar, dentro de la opción “SKU y Tamaño” ingresada en la configuración anterior, aparecen opciones para elegir el tamaño de almacenamiento de disco, tamaño de memoria RAM, estas características son para crear el servidor que alberga la página web, parte de las configuraciones se la puede ver en la Figura 3.61.

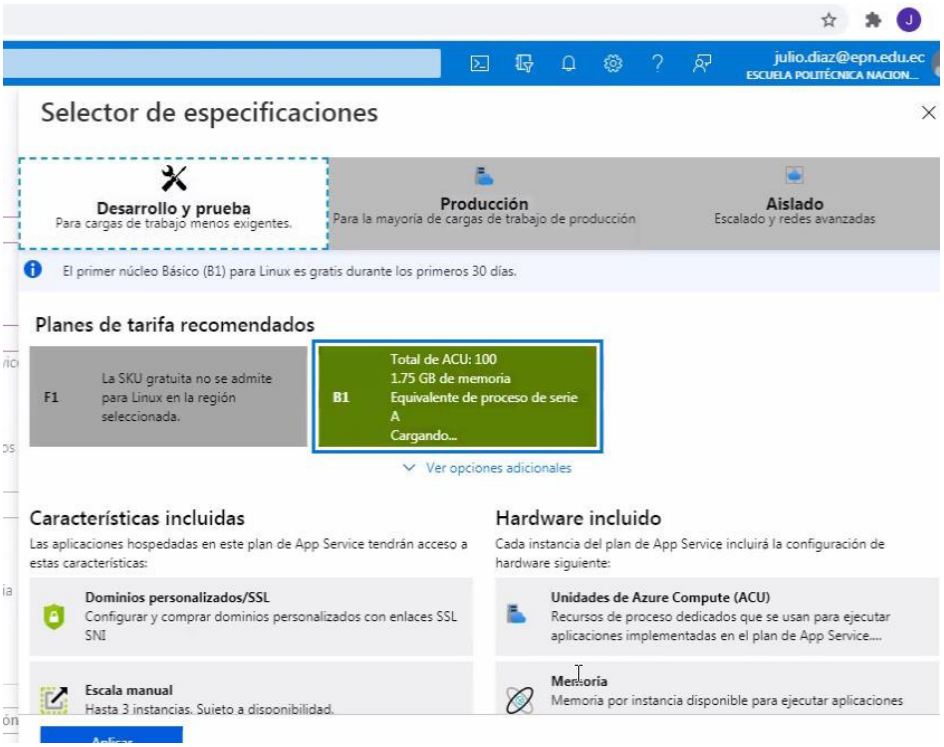


Figura 3.61 Elección de tamaño del servidor

- En sexto lugar, dirigirse hacia la pestaña *Revisar y crear* para ver términos y condiciones del uso de los servicios, finalizado esto se da clic en el botón color azul “Revisar y crear” para continuar con el proceso, esto se lo puede ver en la Figura 3.62.

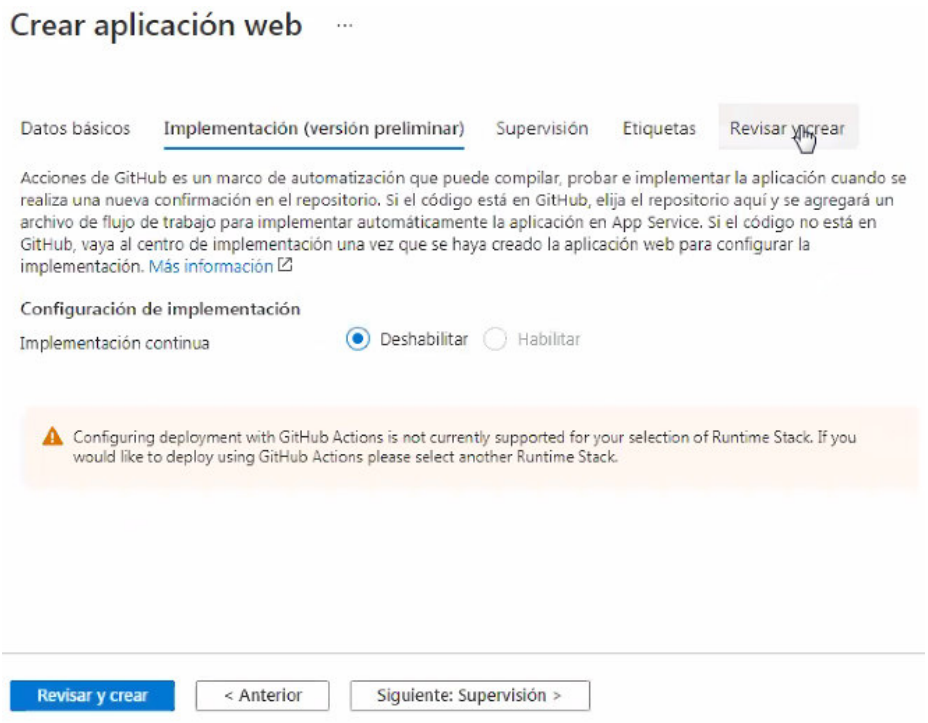


Figura 3.62 Revisión de condiciones de uso

- En séptimo lugar, se muestra todos los detalles configurados de antemano, en caso de no necesitar realizar cambios hacer clic en el botón “Crear”, visible en la Figura 3.63.

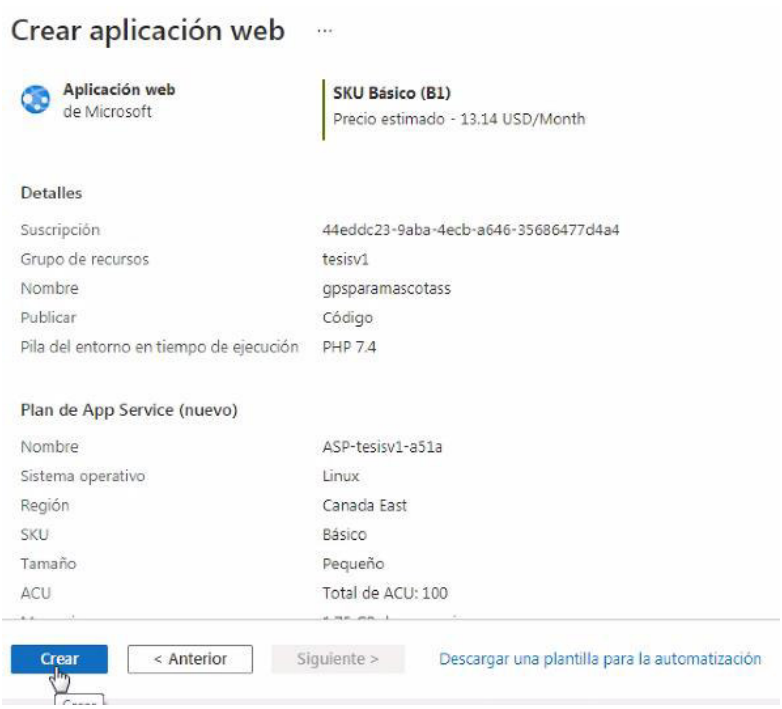


Figura 3.63 Crear servidor web

- En octavo lugar, se muestran los servicios de nube creados, estos serán los encargados de contener y actualizar la página web de manera automática, estos servicios se los puede ver en la Figura 3.64.

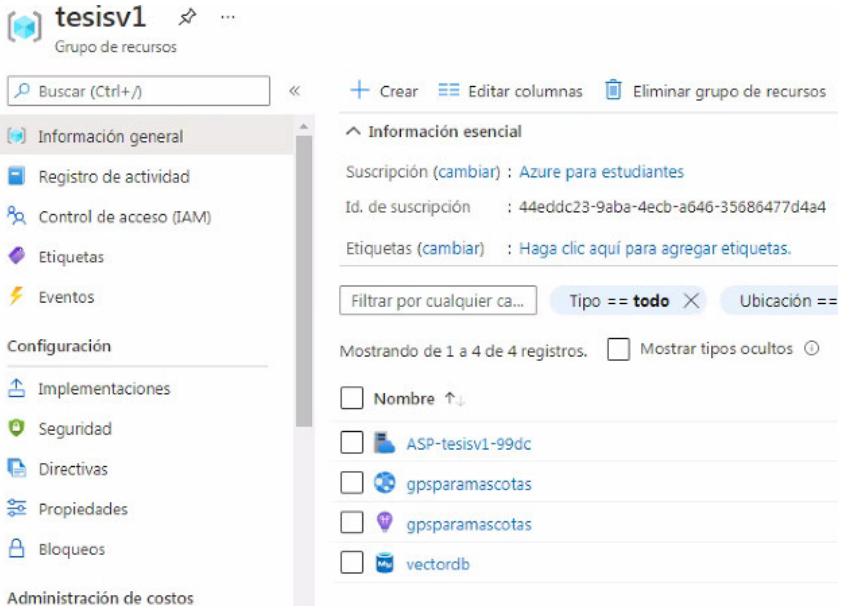


Figura 3.64 Servidor de nube creado

Una vez obtenido un servicio de Microsoft Azure creado en la nube, ya es factible subir la página web a la nube, para ello Azure permite realizar este proceso de una manera fácil, dentro del programa *Visual Studio Code* se debe dirigir al ícono de extensiones, el cual se encuentra en la columna en la parte izquierda como se lo ve en la Figura 3.65.

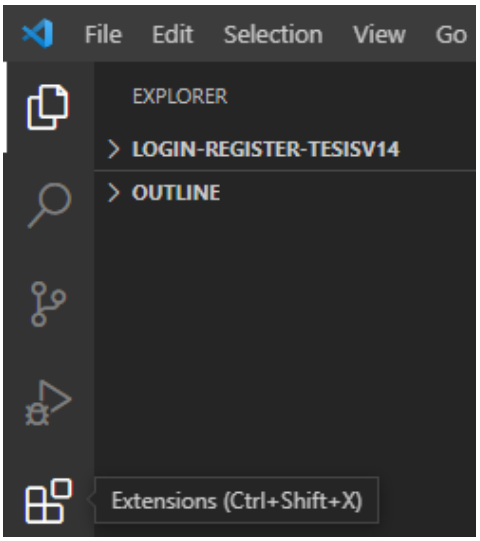


Figura 3.65 Extensión *Azure App Service*

Dentro de las extensiones buscar *Azure App Service* y proceder a instalar como se lo ve en la Figura 3.66, esta extensión facilita la manera de subir la página web a la plataforma Microsoft Azure.

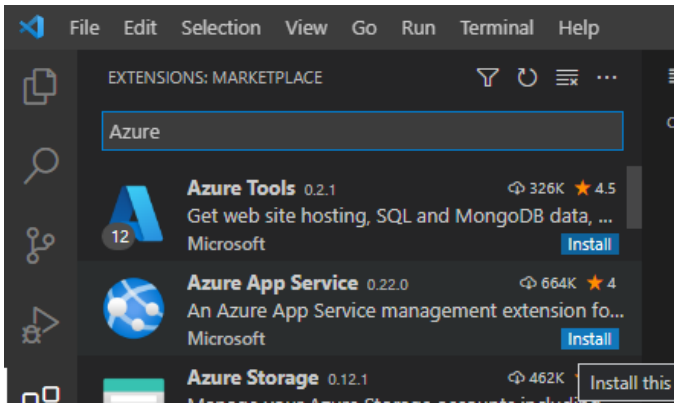


Figura 3.66 *Azure App Service*

Ya instalada la extensión en la barra izquierda aparece un ícono similar a la letra "A", seleccionando este ícono aparece una ventana, dentro de ella en la pestaña *APP SERVICE,* se registra la cuenta de Microsoft Azure y clic en el ícono de subir a la nube esto se lo puede ver en la Figura 3.67.

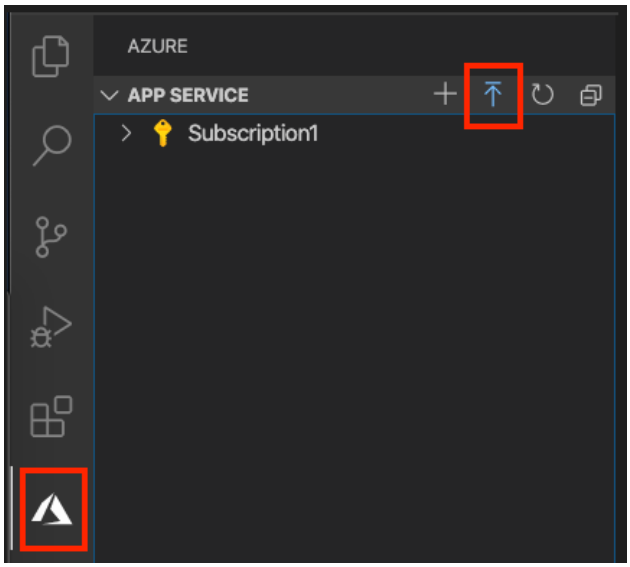


Figura 3.67 Subir página web a la nube.

Mientras se carga los archivos de la página web al servidor Microsoft Azure proporciona dos mensajes, seleccionar "*Yes"* en cada uno para continuar con el proceso, esto se lo puede observar en la Figura 3.68, la página web creada se subirá automáticamente al servidor Microsoft Azure.

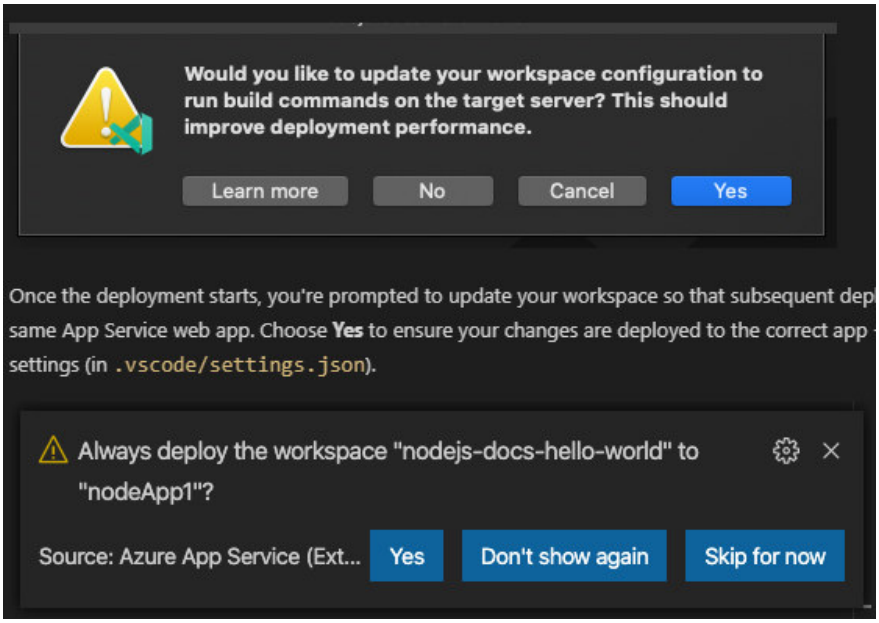


Figura 3.68 Mensajes de alerta

Finalmente, se verificar que la página web está subida a la nube de Microsoft Azure ingresando al dominio establecido en las configuraciones anteriores, de esta manera se puede comprobar que no ha existido error al momento de subir la página web, esto se visualiza en la Figura 3.69.

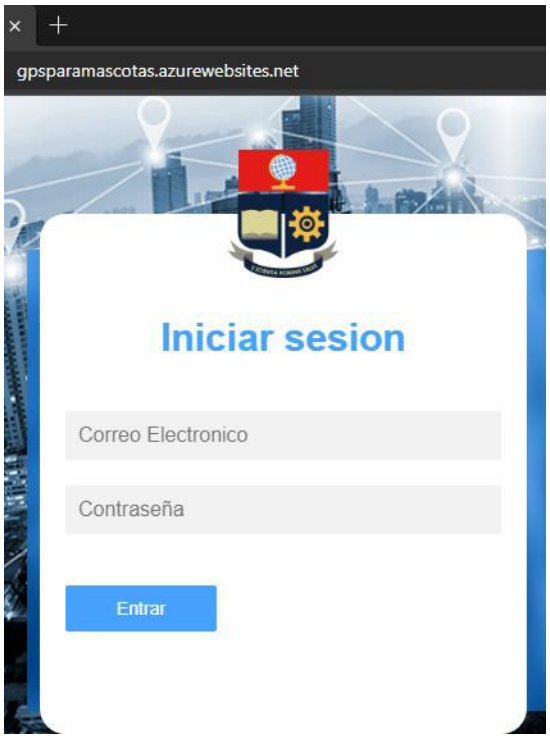


Figura 3.69 Página Web en línea

3.3 Uso de Arduino para la transmisión de datos mediante GPS y GPRS

Programación en Arduino IDE

El algoritmo utilizado en el presente proyecto de titulación se encarga de la adquisición de datos de longitud y latitud y procesamiento y transmisión de estos, esta información se obtiene por medio del módulo SIM808 y la antena GPS/GPRS que viene incluida.

En la creación del programa se utilizaron tres librerías: "DFRobot_sim808.h" permite que Arduino Nano reconozca los datos que le envía el módulo SIM808, "*SoftwareSerial.h*" permite que Arduino Nano use la comunicación serial con otros pines que sean distintos a los que viene por default (pin 0 y pin1) y "*String.h*", con esta librería es posible realizar el manejo y manipulación de cadenas de caracteres; estas librerías son visibles en la Figura 3.70.

```
#include <DFRobot_sim808.h>
#include <SoftwareSerial.h>
#include <String.h>
```

Figura 3.70 Librerías Arduino NANO

La conexión entre el módulo SIM808 y la placa Arduino Nano, se lo realiza mediante los pines digitales 10 y 11, para realizar esta configuración en la placa Arduino se usa el comando de la Figura 3.71.

```
SoftwareSerial Sim800Serial(10, 11);
```

Figura 3.71 Configuración pines digitales

Se crean variables del tipo *char* esta es una matriz sin inicializar que se usará para almacenar los datos tanto de longitud como latitud, estos serán provistos por el módulo SIM808, la función se la ve en la Figura 3.72.

```
char lat[15];
char lon[15];
```

Figura 3.72 Variable para almacenar datos

Estas variables global tipo *char* vistas en la Figura 3.72, posteriormente se encargan de almacenar los datos provistos por el módulo SIM808, después con el uso de la función *string* estos datos serán enviados en una sola cadena hacia el *ThingSpeak*.

Se debe verificar que el módulo SIM808 se encuentra energizado, otro punto a tener en cuenta es que se encuentre activo el led indicador de la tarjeta SIM, caso contrario por medio del pin 9 del Arduino se envía un valor lógico hacia el pin 9 del módulo Sim808 para que se active automáticamente, esto se lo ve en la Figura 3.73.

```
DFRobot_SIM808 sim808(&Sim800Serial);
static const int PowerPin =9; // PIN9
```

Figura 3.73 Inicio del Módulo SIM808 por medio de código

A continuación, se procede a realizar la comunicación entre el Arduino y módulo SIM808 para que exista esta comunicación se configura la velocidad de TX (trasmisión) y RX (recepción) a 19200 baudios como se lo observa en la Figura 3.74, una vez establecida la comunicación el monitor serial indica un mensaje confirmando el trabajo entre el Arduino y el módulo SIM808.

```
Sim800Serial.begin(19200);//
Serial.begin(9600);//Velocid

Serial.println("Empezando");
delay(20000);//Tiempo pruden
```

Figura 3.74 Comunicación Arduino y SIM808

Para mayor seguridad en la activación del chip SIM del módulo SIM808, el código presenta en el monitor serial el estado actual, mientras se no esté activando la bandeja de la tarjeta SIM mostrará un mensaje de error: "*SIM808 init error*".

Al momento de activarse la tarjeta SIM el monitor serial mostrará el mensaje: "*SIM808 init success*", el código para realizar esta acción se encuentra en la Figura 3.75, de igual manera la activación del sistema GPS se lo puede ver por medio del monitor serial, cuando no se encuentre activo el mensaje mostrado será: "*Open the GPS power failure*".

```
while(!sim808.init()) {
   delay(1000);
    Serial.print("Sim808 init error\r\n");
    pinMode(PowerPin, OUTPUT);
    //to turn on the GSM module -software switch instead Power Key
    digitalWrite(PowerPin, HIGH);
    delay (3000);
    digitalWrite (PowerPin, LOW);
    delay (10000); //delay set for 10 seconds

}
```

Figura 3.75 Mensaje estado tarjeta SIM

Dentro del código el mensaje "*Open the GPS power failure*" será constante hasta que empiece a recibir los datos de geo-posicionamiento, al momento de tener conexión el mensaje será el siguiente: "*Open the GPS power success*", para realizar esta acción el código usado se encuentra en la Figura 3.76.

```
if( sim808.attachGPS())
    Serial.println("Open the GPS power success");
 else
    Serial.println("Open the GPS power failure");
```

Figura 3.76 Mensaje conexión a la Red Móvil

En la función "*void comandosAT*" Figura 3.77 es la encargada de enviar los datos de posicionamiento en longitud y latitud para guardarlos en la base de datos en los respectivos "*Field*" del canal creado en *ThingSpeak*, para obtener una ejecución correcta de los comandos AT es necesario crear un retardo entre cada comando, se utilizó un retardo de dos segundos, los comandos utilizados en el código son los siguientes:

- AT+ CIPSTATUS: Muestra el estado de la conexión actual.
- AT+ CIPMUX: Configura el dispositivo para una conexión IP estática o dinámica.
- AT+ CGATT: Asocia o separa al dispositivo al servicio del dominio de paquetes.
- AT+ CSTT: Configura el APN, el nombre de usuario y contraseña, depende del operador móvil.
- AT+ CIICR: Abre el puerto para la conexión GPRS, acepta llamadas GPRS o CSD.
- AT+ CIFSR: Solicita una dirección IP local a la red móvil.
- AT+ CIPSPRT: Se usa para fijar el inicio de un paquete de transmisión, el símbolo es ">"
- AT+ CIPSEND: Se usa para el envío de datos a través de la conexión TCP o UDP.
- AT+ CIPSHUT: Cierra la conexión del GPRS.

```
void comandosAT(){
Sim800Serial.println("AT+CIPSTATUS");
delay(2000);
Sim800Serial.println("AT+CIPMUX=0");/
delay(3000);
mostrarDatosSeriales();
Sim800Serial.println("AT+CGATT=1");//
delay(3000);
mostrarDatosSeriales();
Sim800Serial.println("AT+CSTT=\"igprs
delay(1000);
mostrarDatosSeriales();
Sim800Serial.println("AT+CIICR");//RE
delay(3000);
mostrarDatosSeriales();
Sim800Serial.println("AT+CIFSR");// O
delay(2000);
mostrarDatosSeriales();
Sim800Serial.println("AT+CIPSPRT=0");
GPS();
delay(3000);
mostrarDatosSeriales();
Sim800Serial.println("AT+CIPSTART=\"T
delay(6000);
mostrarDatosSeriales();
Sim800Serial.println("AT+CIPSEND");//
```

Figura 3.77 Comandos AT envío de datos a ThingSpeak

Para ver el listado completo y una descripción más detallada de comandos AT se pueden revisar el Anexo 8.

Por último, la función final "*void GPS*" se encarga de tomar lectura de los datos obtenidos por el GPS, estos datos pueden ser visualizados por medio del monitor serial, se puede obtener información como: fecha, tiempo, ubicación. Estos datos se los puede apreciar en la Figura 3.78.

```
Serial.print(sim808.GPSdata.year);
Serial.print("/");
Serial.print(sim808.GPSdata.month);
Serial.print("/");
Serial.print(sim808.GPSdata.day);
Serial.print(" ");
Serial.print(sim808.GPSdata.hour);
Serial.print(":");
Serial.print(sim808.GPSdata.minute);
Serial.print(":");
Serial.print(sim808.GPSdata.second);
Serial.print(":");
Serial.println(sim808.GPSdata.centisecond);
Serial.print("latitude :");
Serial.println(sim808.GPSdata.lat,9);
Serial.print("longitude :");
Serial.println(sim808.GPSdata.lon,9);
```

Figura 3.78 Obtención de datos GPS

Después de obtener el valor de longitud y latitud, se almacenan en una variable llamada "*float la*" y "*float lo*" debido a que los valores se encuentran en decimales, con la función "*dtostrf*" estos datos se dividen en 7 valores para números enteros y 8 para números decimales, los datos obtenidos son guardados en las variables *lat* y *lon* visualizados en la Figura 3.79.

```
148     float la = sim808.GPSdata.lat;
149     float lo = sim808.GPSdata.lon;
150     float ws = sim808.GPSdata.speed_kph;
151
152   dtostrf(la, 7, 8, lat); //toma los valc
153   dtostrf(lo, 7, 8, lon); //toma los valc
```

Figura 3.79 Función dtostrf

Para finalizar los datos de longitud y latitud se pueden ver usando un dispositivo electrónico simplemente accediendo a la página web que se subió a la nube con anterioridad, para ello se accede a la dirección web creada para el servidor, la página se muestra de forma correcta como se visualiza en la Figura 3.80.

Figura 3.80 Página Web móvil

Con la mascota ya registraba se puede ir a verificar su respectiva ubicación y comprobar el funcionamiento de la página como se visualiza en la Figura 3.81.

Figura 3.81 Búsqueda de Mascota

Para concluir se despliega la *API* de *Google Maps* por medio del dispositivo electrónico, con ello el usuario puede ver la ubicación de una mascota registrada en un mapa como lo indica la Figura 3.82.

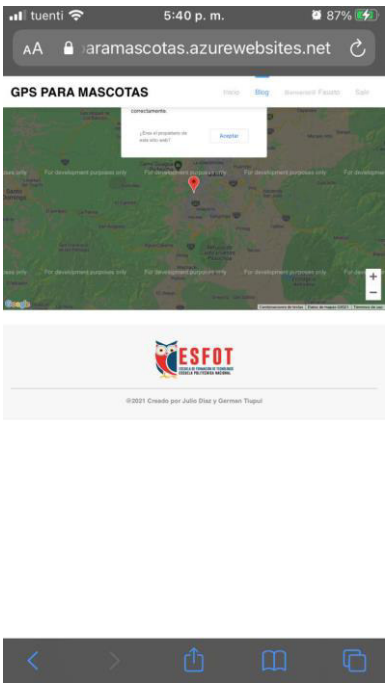


Figura 3.82 Google *Maps* en móvil

El código descrito se encuentra disponible en el Anexo 9.

Diagramas de flujo para código Arduino

Al moneto de enceder el sistema de geo posicionamietno, se energizara la placa Arduino nano y el módulo SIM808, ejecutandase el código descrito con anterioridad. Este código empieza con una comunicación serial entre al Arduino nano y SIM808 a 1920 (*Baudios*) y 9600 (*Baudios*), con esta comunicación la placa Arduino verifica que el módulo SIM808 se encuentre en fucnionanmiento, caso contrario ejecuta el código de reinicio.

A continuación se inicia la configuración para la recepción de datos GPS, estas configuraciones mecionadas con anterioridad se lo realiza una sola vez ya que pertenecen a la parte de configuración de *VOID SETUP*.

Continuando con el código, se ejecuta la parte del programa conocida como *VOID LOOP*, en donde por medio de comandos AT la parte del GPS toma datos que seran usados posteriormente. Estos datos seran almacenados hasta ser enviados usando la antena GPRS del SIM808. Lo descrito anteriormente se lo puede ver en la Figura 3.83.

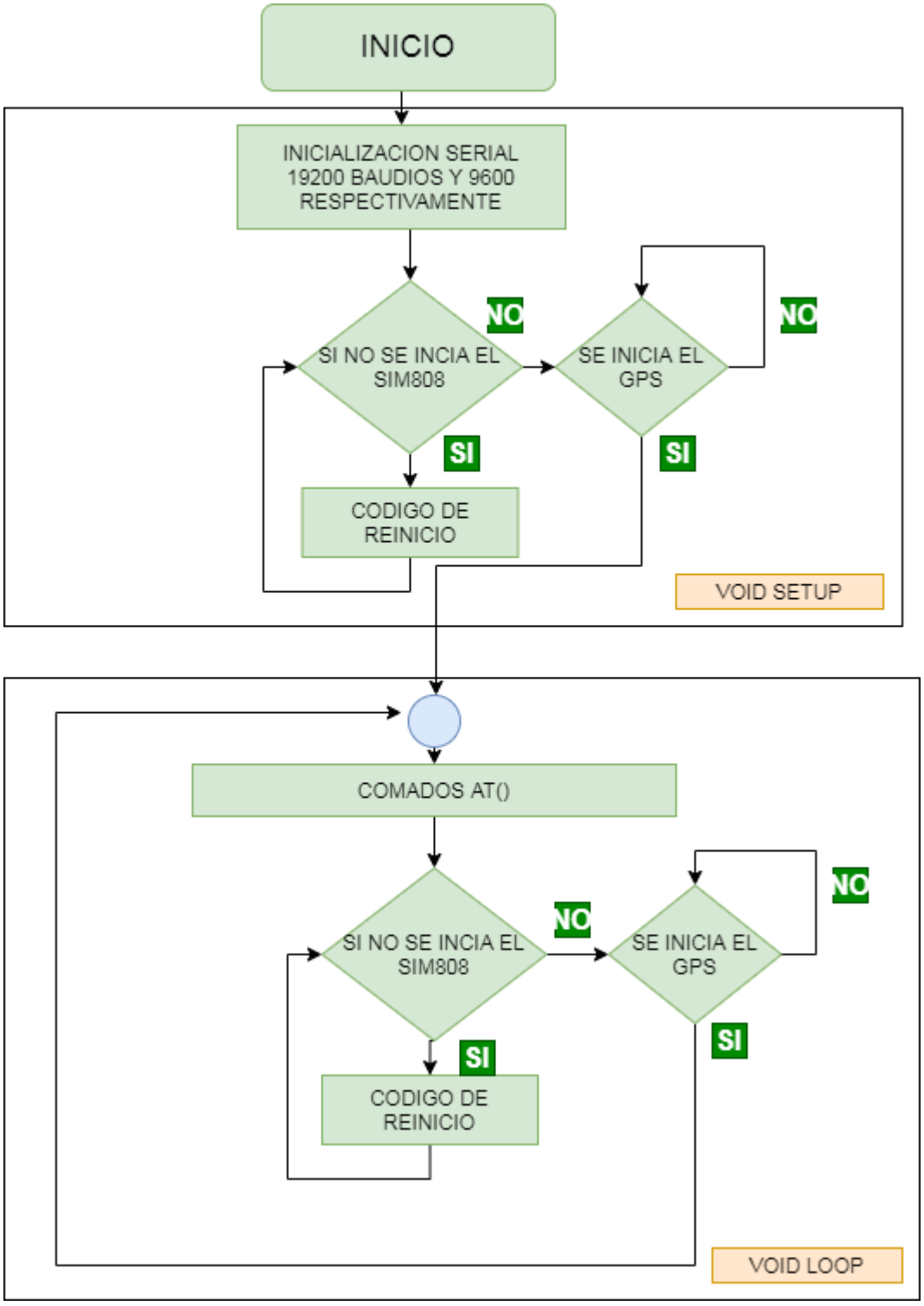


Figura 3.83 Diagrama de flujo *Void Set Up* y *Void Loop*

El código de reinicio se ejecuta en caso que el SIM808 no se encuentre conectado a la red de telefonía respectiva, para este caso Tuenti la cual trabaja en la freceuncia de 1900 (Mhz). En caso que el módulo SIM 808 no se conecte a su respectiva red se tomara un segundo para configurar el pin 9 de Arduino como un salida.

Haciendo uso de este Pin se envía un 1 lógico por un tiempo de 3 segundos hacia el módulo SIM808, de esta forma se reinicia automaticamente el lector de la terjeta SIM, para continuar se envia un 0 lógico por el Pin 9 por un tiempo de 10 segundos, de esta forma esperando que se conecte a la red de telefonía movil. Lo descrito anteriormete se lo puede ver en la Figura 3.84.

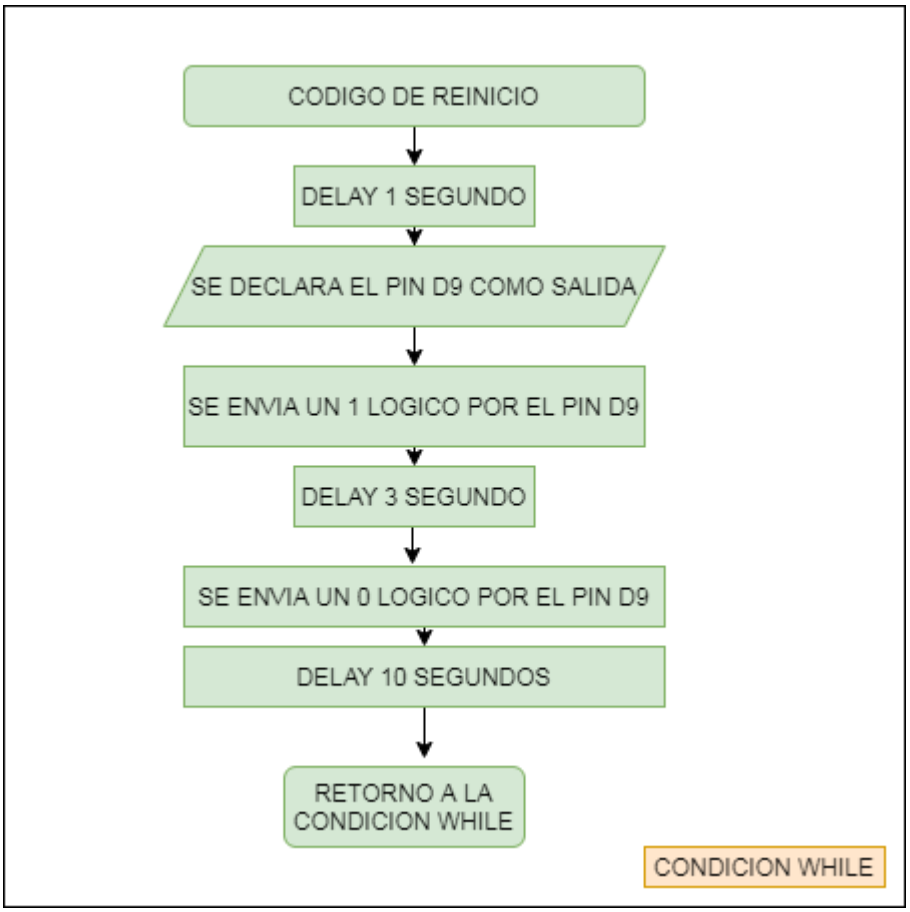


Figura 3.84 Código de reinicio

La configuración para la parte del GPS se ejecuta de manera que en caso de no poseer la información, se vuelva a realizar la solicitud. Una vez con los datos GPS obtenidos son procesados para obtener los valores de latitud y longitud, valores lo cuales serán guardados en las variables LAT y LONG respectivamente, estos valores serán enviados hacia el servidor ThingSpeak posteriormente. Esto se lo puede ver en la Figura 3.85.

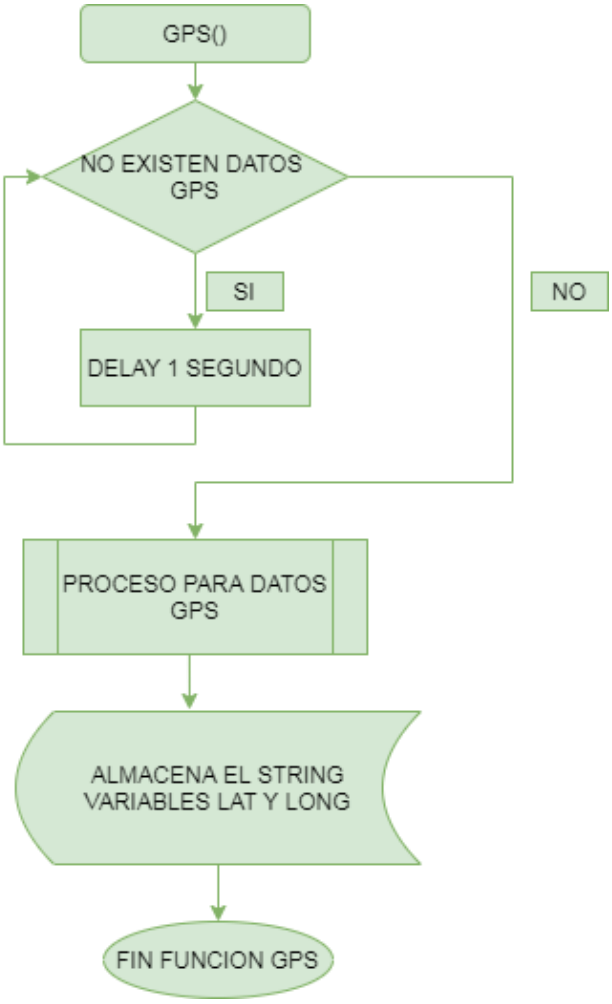


Figura 3.85 Función GPS

La función de comandos AT se encargan de realizar las configuraciones para conectarse a la red GPRS, revisando primero el estado actual de la conexión, una vez este activa la conexión se realiza la solicitud de una dirección IP única, a continuación se inicia la conexión GPRS y con ello al módulo SIM808 se le asigna una IP única.

Con la dirección IP ya obtenida se realiza la conexión con la APN de la operadora en este caso será Tuenti, después se inicia la conexión GPRS con la red telefónica de la operadora que se esté utilizando, una vez ya conectado se realiza en envío de un indicador para la transmisión de datos.

Los datos se lo obtienen con el uso de la función GPS descrita con anterioridad, los valores de longitud y latitud se los envían al servidor ThingSpeak usando TCP, una vez ya terminado el envío de esta información se debe cerrar la conexión por lo cual se usa el carácter 26 y se regresa al *Void Loop*. Lo descrito anteriormente se lo puede ver en la Figura 3.86.

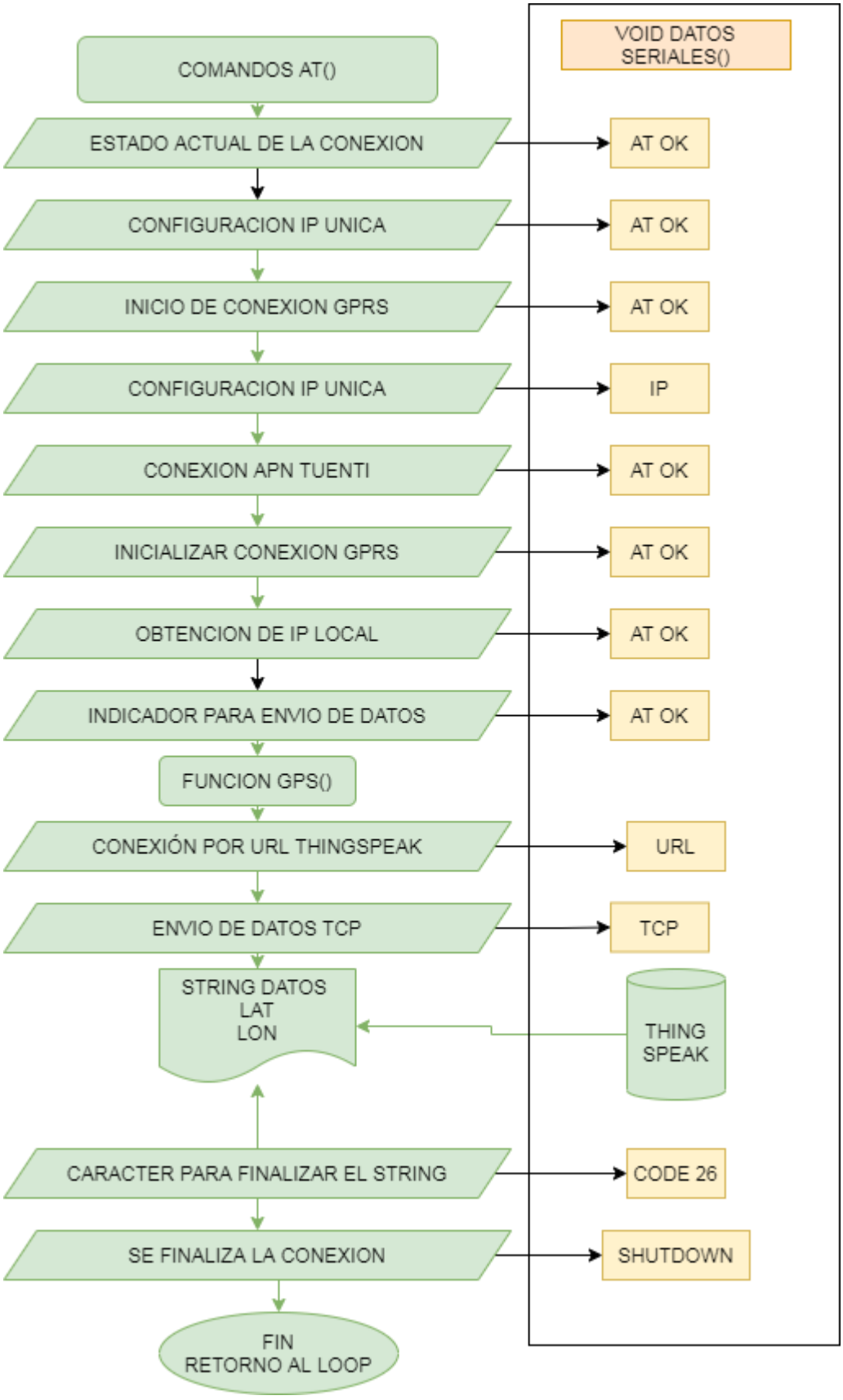


Figura 3.86 Función comandos AT

3.4 Implementación de dispositivo GPS que permita la detección y localización de las mascotas

Se inicia tanto el módulo SIM808 y la placa Arduino Nano, en esta última se crean variables globales para guardar la información de geo-posicionamiento que estará enviado por el módulo Sim808 por medio de la conexión serial que existe entre ambos.

Una vez iniciado el módulo Sim808 empezará la lectura de los datos de geo-posicionamiento por medio de su antena GPS, en caso de no encontrar una lectura seguirá leyendo, si el módulo encuentra datos se ejecuta la subrutina comandos AT.

En la subrutina “comandosAT” se hace lectura de los datos recibidos por la antena GPS para ello se utiliza “sim808.getGPS” con ello obtenemos toda la información y se la almacena en variables, los comandos AT también inician la conexión con la red móvil, por medio de su antena GSM realizan el envío de datos hacia el servidor *ThingSpeak*.

El servidor *ThingSpeak* recibe estos datos y los almacena dentro de unas variables llamadas *Field,* esta información puede ser usada por otros servidores en caso de ser requerido.

El Servidor Microsoft Azure Acede a la información de *ThingSpeak* y la presenta usando la API de Google *Maps*, el funcionamiento del sistema de geo-posicionamiento se lo puede representar por medio del diagrama de flujo mostrado en la Figura 3.87.

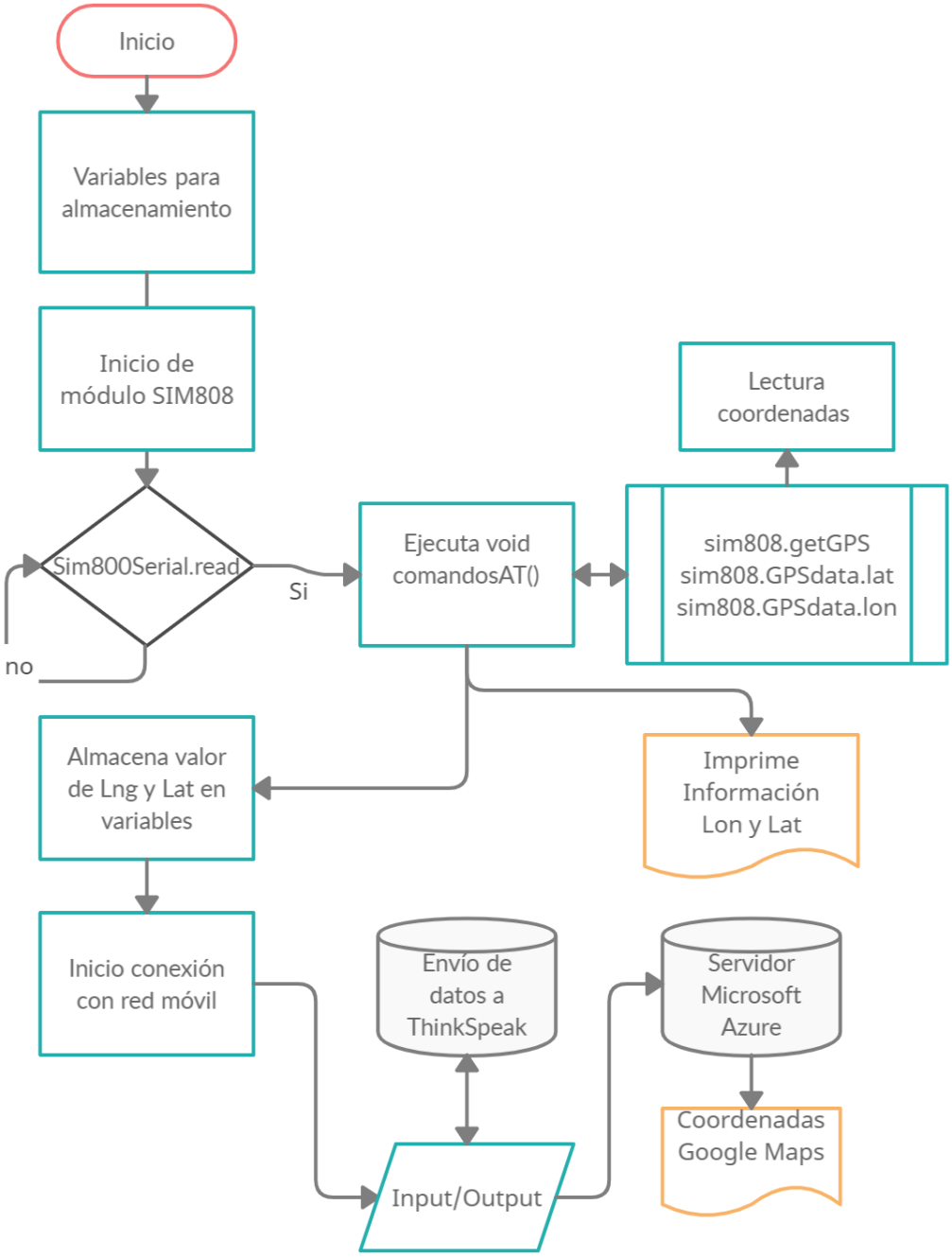


Figura 3.87 Diagrama de flujo del sistema de geo-posicionamiento

Elección de arnés

Durante el desarrollo del presente proyecto se optó por utilizar dos arneses de un tamaño considerable para la comodidad de las mascotas, se pudo acoplar el sistema de geo-posicionamiento en cada arnés [24].

El arnés utilizado para el sistema de geo-posicionamiento que emplea 3 baterías se lo puede ver en la Figura 3.88.

Figura 3.88 Arnés para sistema de 3 baterías

El arnés utilizado para el sistema de geo-posicionamiento que emplea 2 baterías se lo puede ver en la Figura 3.89.

Figura 3.89 Arnés para sistema de 2 baterías

Cajas para módulos usados

Se eligieron dos cajas plásticas ABS para proyectos de electrónica. Para determinar las dimensiones de cada una de las cajas se tomó en cuenta las medidas y cantidad de los elementos utilizados: Arduino Nano, SIM808, baterías 18650, módulo BMS 3S, módulo BMS 2S y módulo Step Down 2596.

Las dimensiones de la primera caja son 97 (mm) * 81 (mm) * 52 (mm), visible en la Figura 3.90, cuenta con espacio para almacenar toda la circuitería y elementos de electrónica, el espacio que posee permite el uso de tres baterías.

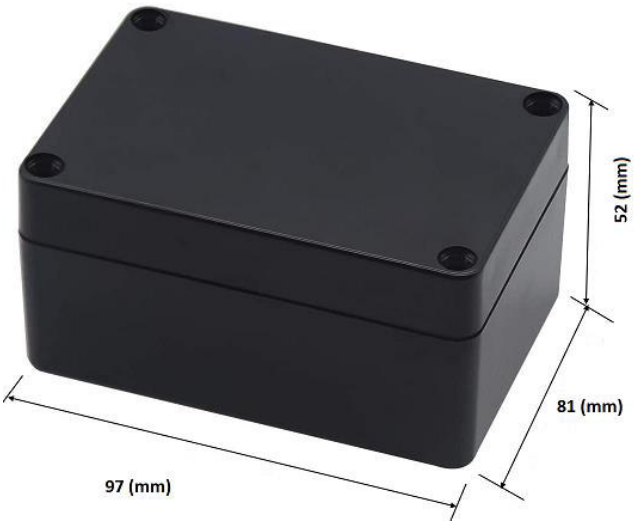


Figura 3.90 Caja para 3 baterías y módulo BMS 3S

Las dimensiones de la segunda caja son 130 (mm) * 70 (mm) * 40 (mm), visible en la Figura 3.91, cuenta con espacio para almacenar toda la circuitería y elementos de electrónica, el espacio que posee permite el uso de dos baterías.

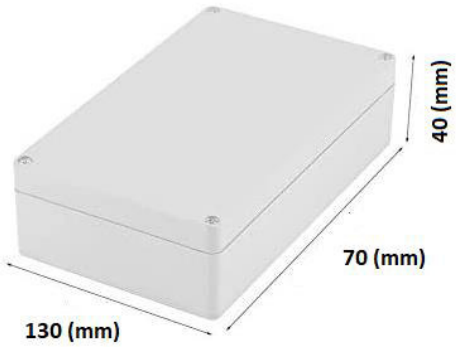


Figura 3.91 Caja para 2 baterías y módulo BMS 2S

Terminada la parte de programación del código y subido a la placa Arduino Nano, se realizó la conexión con el resto de compontes electrónicos, al tener dos sistemas de geo-posicionamiento con distinto tamaño se realizaron leves modificaciones que se verán a continuación.

Sistema Geo-posicionamiento 3 baterías y módulo BMS 3S 10(A)

En el presente sistema se observa el diseño físico y conexión respectiva entre los componentes utilizados y descritos en los capítulos anteriores, esta gráfica de conexiones se la puede apreciar en la Figura 3.92.

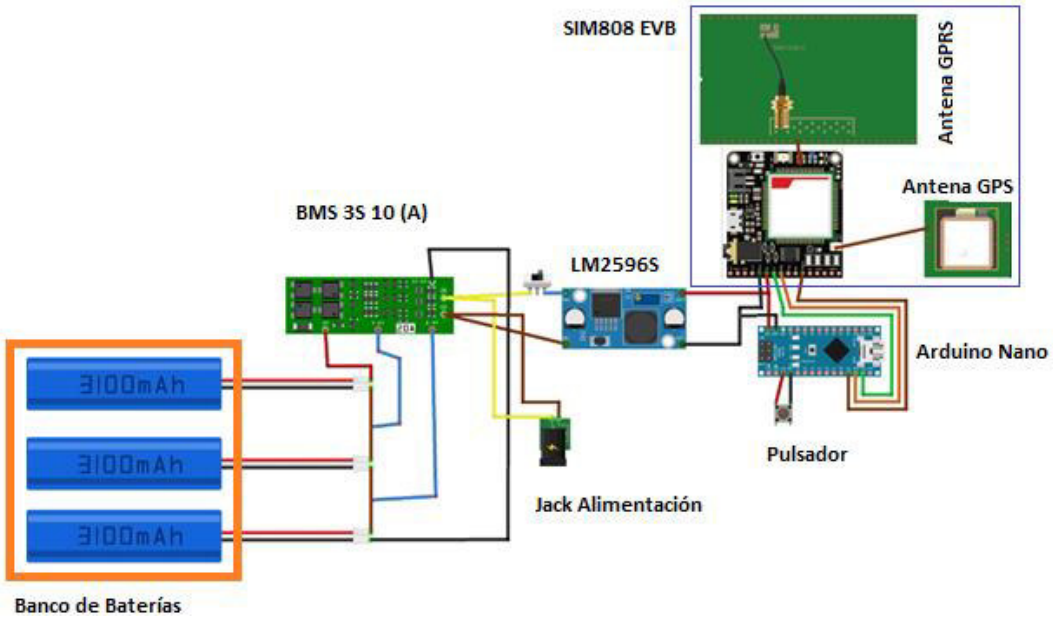


Figura 3.92 Geo-posicionamiento con 3 baterías

Con el uso del diagrama anterior se procede a realizar el armado físico del primer sistema de geo-posicionamiento, los componentes que se usaron son los descritos en capítulos anteriores, siguiendo el diseño se realizó el ensamblaje adecuado para que ingrese en su respectiva caja, este circuito se lo puede visualizar en la Figura 3.93.

Figura 3.93 Ensamble de geo-posicionamiento a 3 baterías

Una vez armado este primer módulo de geo-posicionamiento, se procede a acoplarlo a su respectivo arnés para comprobar que sus medidas encajen de manera óptima, esto se lo puede ver en la Figura 3.94.

Figura 3.94 Primer sistema geo-posicionamiento acoplado

Para concluir se procede a colocarlo en la mascota, con ello se asegura que esté bien sujeto y el sistema de geo-posicionamiento no llegue a separarse de su respectivo arnés, esto se lo puede ver en la Figura 3.95.

Figura 3.95 Mascota y primer sistema de geo-posicionamiento

Sistema Geo-posicionamiento 2 baterías y módulo BMS 2S 10(A)

En esta sección se puede visualizar el diseño físico y conexión respectiva entre los componentes utilizados y descritos para este sistema de geo-posicionamiento, utilizando una alimentación eléctrica de 2 baterías visible en la Figura 3.96.

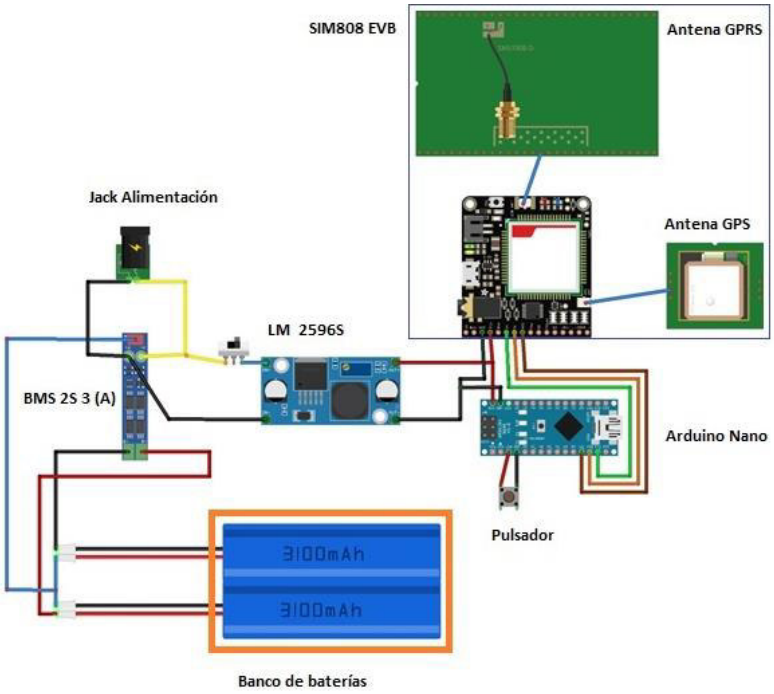


Figura 3.96 Geo-posicionamiento con 2 baterías

Con el uso del diagrama anterior se procede a ensamblar el segundo sistema de geo-posicionamiento, se utilizó los mismos componentes, pero con una batería menos, se usa el arnés correspondiente, así como su respectiva caja. El circuito armado se visualiza en la Figura 3.97.

Figura 3.97 Ensamble de geo-posicionamiento a 2 baterías

Una vez ya armado se procede a acoplarlo a su respectivo arnés para comprobar que sus medidas encajen de manera óptima, esto se lo puede ver en la Figura 3.98.

Figura 3.98 Segundo sistema geo-posicionamiento acoplado

Finalmente se procede a colocarlo en la mascota, con ello se comprueba que esté bien sujeto y el sistema de geo-posicionamiento no llegue a separarse de su arnés, esto se lo puede ver en la Figura 3.99.

Figura 3.99 Mascota y segundo sistema de geo-posicionamiento

3.5 Pruebas y Análisis de Resultados

Para verificar el correcto funcionamiento del sistema de geo-posicionamiento diseñado se realizó de manera práctica, para ello se ha diseñado una serie de pasos que debe cumplir el sistema para validar su funcionamiento.

- Primero, se debe confirmar que el servidor *ThingSpeak* reciba información de manera constante por parte del sistema de geo-posicionamiento.
- Segundo, en la página web se debe poder visualizar la API Google *Maps* con la información que contenga el servidor *ThingSpeak*, la página web debe mostrar la ubicación del sistema de geo-posicionamiento.

- Finalmente, comparar la ubicación con un sistema de geo-posicionamiento diferente, para lo cual se utiliza el sistema de ubicación GPS de un teléfono celular, con ello se puede ir a la misma dirección y buscar a la mascota.

Se realizó la búsqueda de geo-posicionamiento para la mascota llamada Blue para ello se ingresó a la opción Ver Ubicación. Figura 3.100.

Figura 3.100 Acceso a la API Google *Maps*

Pruebas de localización

Primera localización

Se confirma que exista un cambio en la base de datos *ThingSpeak* visible en la Figura 3.101.

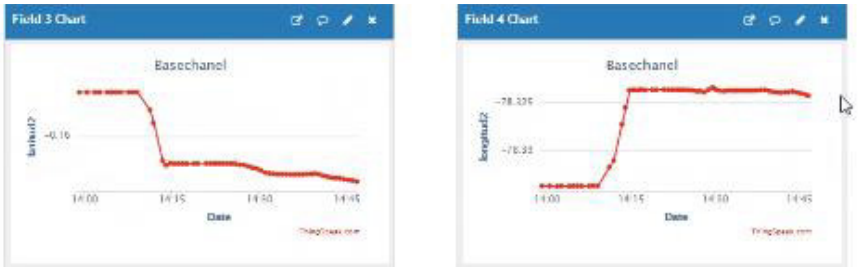


Figura 3.101 Primer cambio en los datos *ThingSpeak*

En la página web se puede visualizar la localización de la mascota, se encuentra una cuadra abajo de la Calle 2, cerca de una intersección visible en la Figura 3.102.

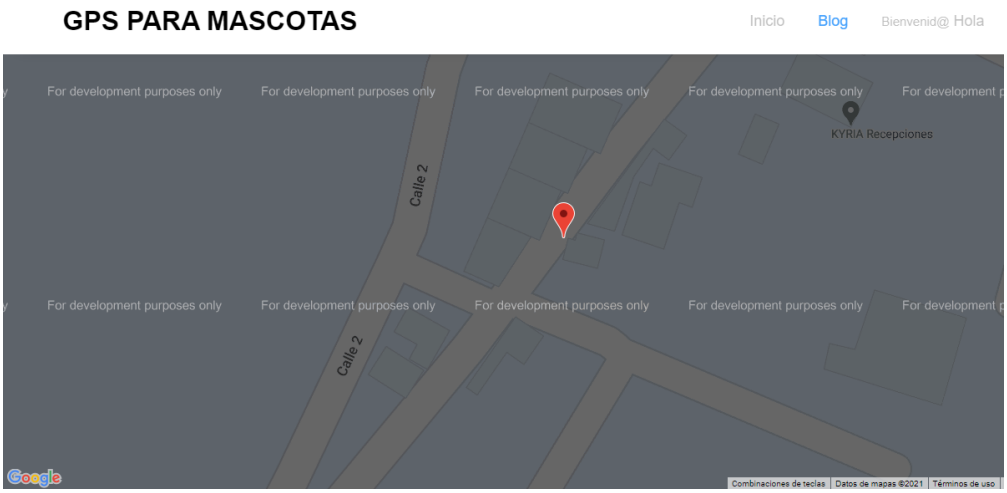


Figura 3.102 Primera localización

Se lo compara con un sistema de ubicación diferente, en este caso se utiliza el de un teléfono celular como se lo ve en la Figura 3.103, indicando la misma posición que se encuentra en la página web.

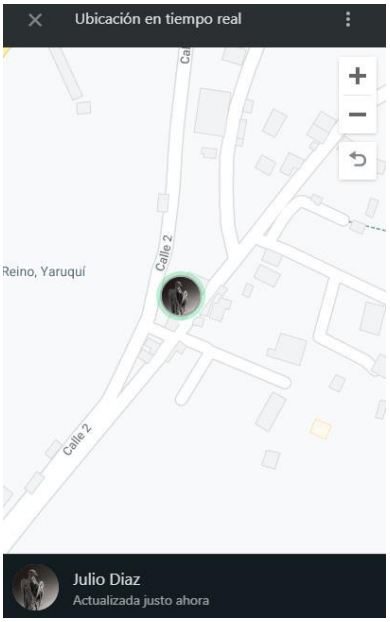


Figura 3.103 Comprobación de la Primera localización

La localización es correcta, las distancias son muy similares, la mascota llamada Blue fue encontrada cerca de una intersección, esto se visualiza en la Figura 3.104.

Figura 3.104 Mascota Blue encontrada primera localización

Segunda localización.

Se libera a la mascota para localizarla, pasado un tiempo se corrobora que el servidor *ThingSpeak* continúe con la recepción de datos esto se confirma en la Figura 3.105.

Figura 3.105 Segundo cambio en los datos *ThingSpeak*

Se observa en la página web la nueva localización de la mascota, la cual se encuentra en la intersección de la Calle 1 y Calle 2 como se lo observa en la Figura 3.106.

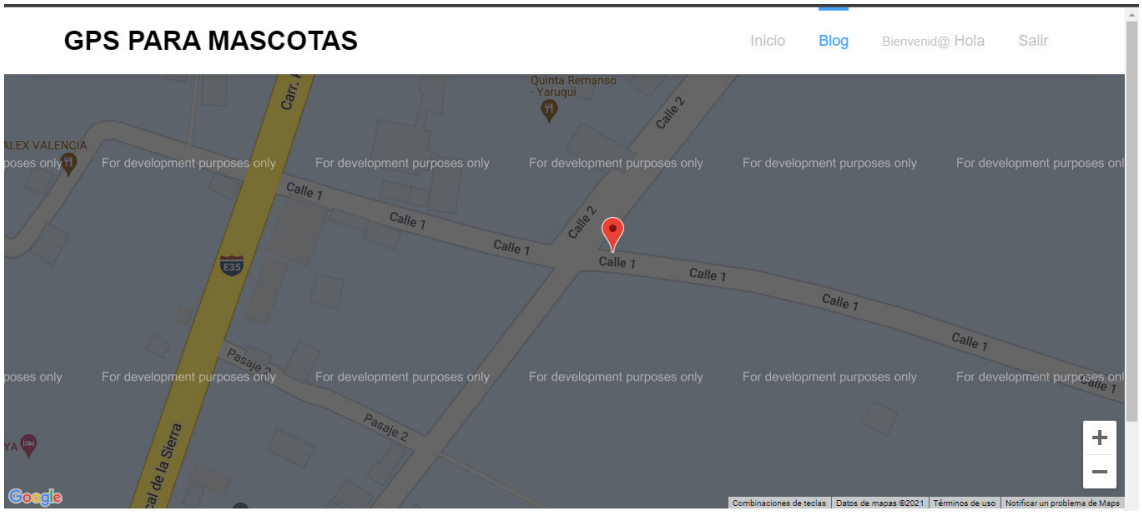


Figura 3.106 Segunda localización

Se compara la segunda ubicación con el uso del GPS del teléfono móvil visible en la Figura 3.107, el sitio coincide y muestra una pequeña variación en la ubicación.

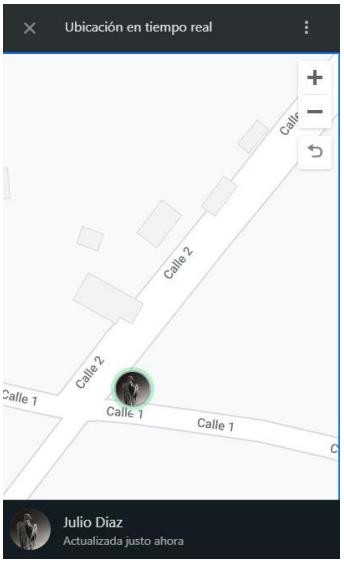


Figura 3.107 Comprobación de la segunda localización

La localización es correcta, las distancias son muy similares, la mascota llamada Blue fue encontrada cerca de una esquina que corresponde a Calle 1 y Calle 2, esto se lo puede apreciar en la Figura 3.108.

Figura 3.108 Mascota Blue encontrada segunda localización

Tercera localización

Se libera a la mascota para una tercera prueba, y se continúa comprobando que estén llegando datos al servidor *ThingSpeak*, en la Figura 3.109 se confirma el continuo envío de información.

Figura 3.109 Tercer cambio en los datos *ThingSpeak*

Se observa en la página web la tercera localización de la mascota, la cual se encuentra al finalizar la primera cuadra de la Calle 1 visible en la Figura 3.110.

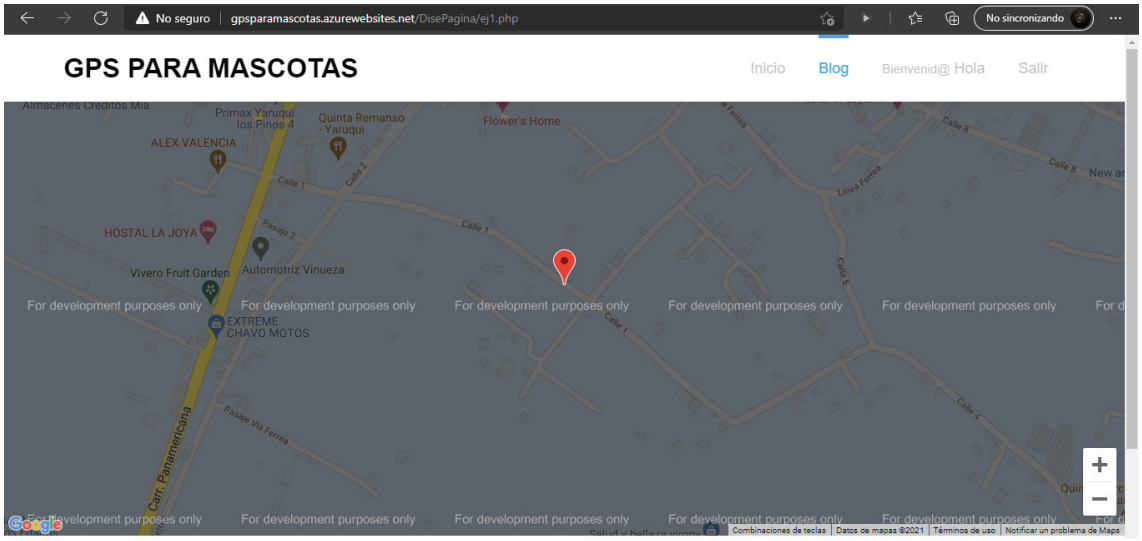


Figura 3.110 Tercera localización

Se compara la tercera ubicación con el uso del GPS del teléfono móvil, lo cual se visualiza en la Figura 3.111, el lugar coincide y muestra la misma ubicación, la cual es cercana a finalizar la Calle 1.

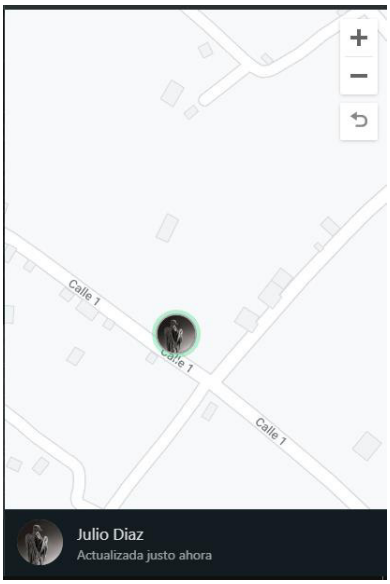


Figura 3.111 Comprobación de la tercera localización

La localización es correcta las distancias son muy similares la mascota fue encontrada cerca de finalizar la Calle 1 junto a una puerta de color negro visible en la Figura 3.112.

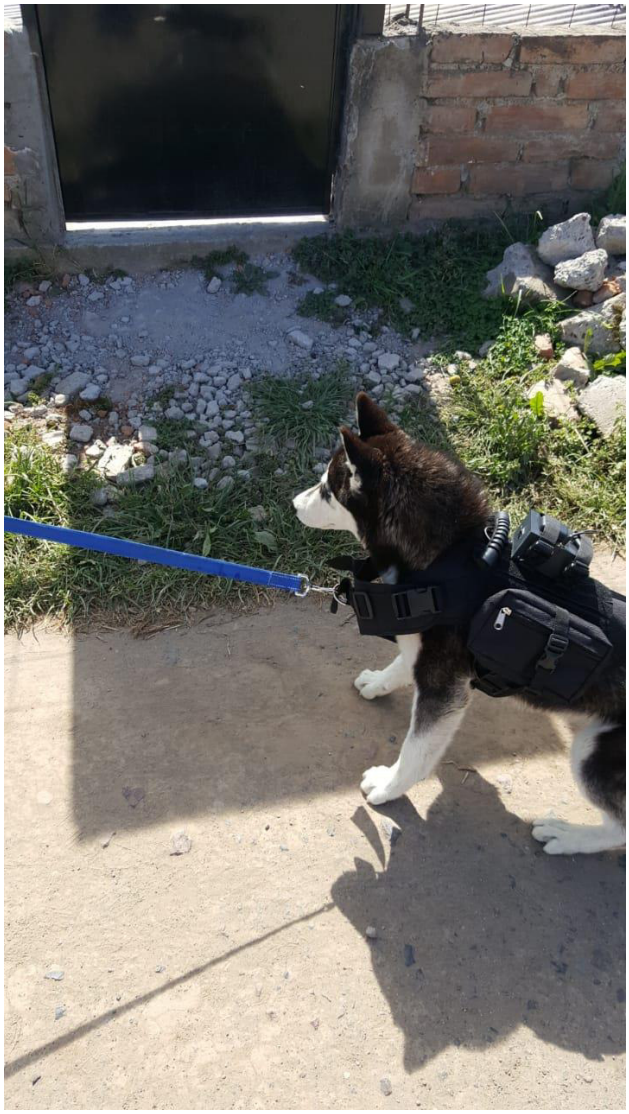


Figura 3.112 Mascota Blue encontrada tercera localización

Cuarta localización

La mascota conocida como Blue es llevada a su hogar, una vez en casa se verifica que el servidor *ThingSpeak* tenga los datos de su respectivo hogar visible en la Figura 3.113.

Figura 3.113 Cuarto cambio en los datos *ThingSpeak*

Se observa en la página web la localización de la mascota en su casa, la cual se encuentra una cuadra abajo de la Calle 2, se lo puede ver en la Figura 3.114.

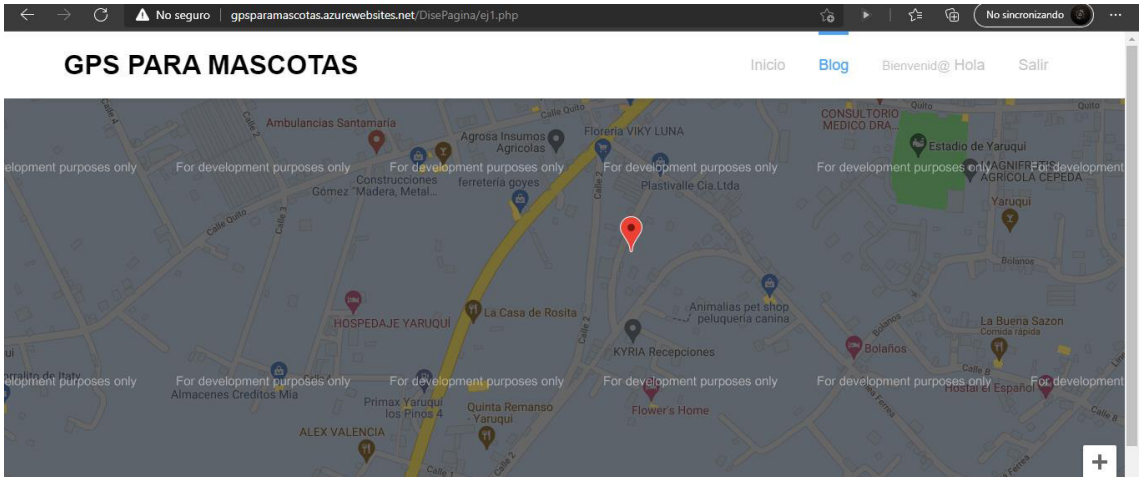


Figura 3.114 Cuarta localización

Se compara la cuarta ubicación con el uso del GPS del teléfono móvil, lo cual se visualiza en la Figura 3.115, la dirección coincide y muestra que la mascota se encuentra en su respectivo hogar.

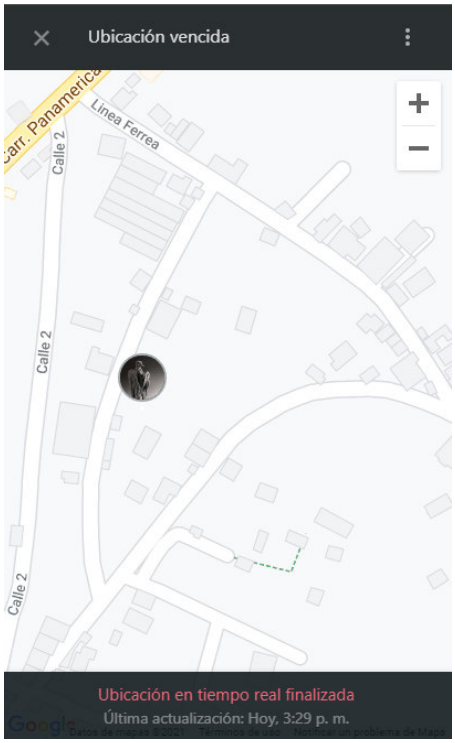


Figura 3.115 Comprobación de la cuarta localización

La localización es correcta el lugar es el mismo, la mascota se encuentra dentro de su hogar esto se lo puede observar en la Figura 3.116.

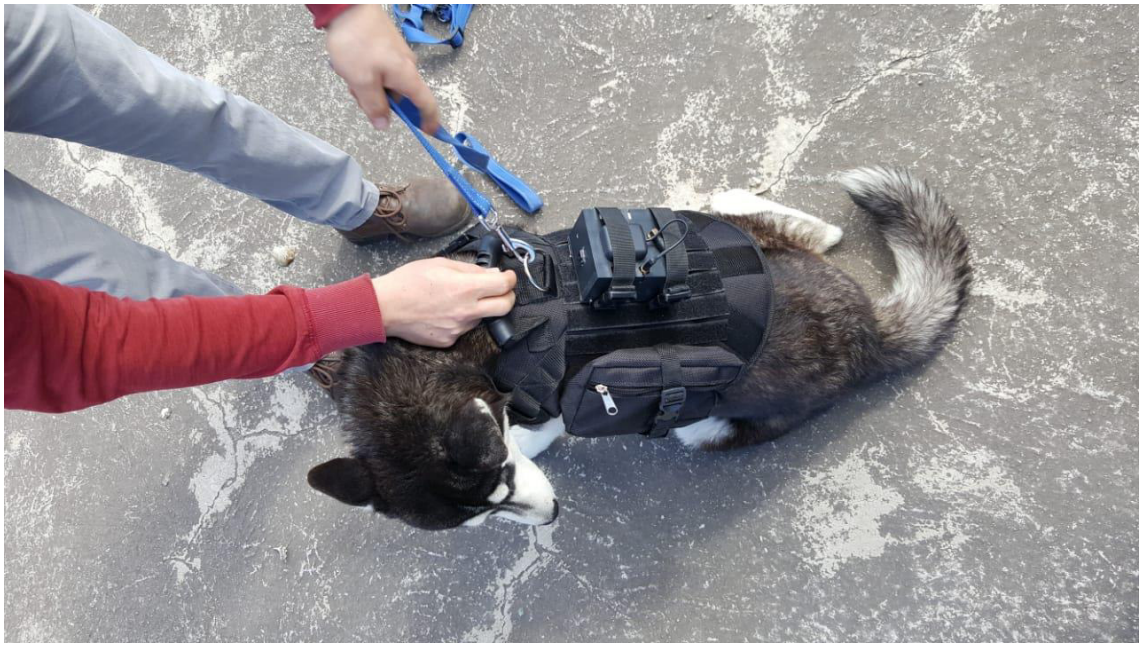


Figura 3.116 Mascota Blue en su hogar

Análisis de resultados obtenidos

En la Tabla 3.7 se muestra un resumen que generaliza lo realizado en las pruebas de funcionamiento, siguiendo los pasos establecidos para validar el correcto funcionamiento del sistema de geo-posicionamiento.

Tabla 3.7 Cumplimiento pruebas del sistema de geo-posicionamiento

Número de localizaciones	Envío de datos a ThingSpeak	Visualización en página web	Comparación con otro sistema GPS
Primera localización	✓	✓	✓
Segunda localización	✓	✓	✓
Tercera localización	✓	✓	✓
Cuarta localización	✓	✓	✓

Mantenimiento Módulos de Geo-posicionamiento

Para realizar el mantenimiento respectivo se lo puede realizar utilizando una brocha o una lata de aire comprimido, mediante esto se puede retirar el polvo o suciedad que pueda ingresar al circuito, para una explicación más detallada se puede ver en el Anexo 10.

4 CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- Por las características del receptor GPS utilizado en este proyecto, la ubicación no es totalmente precisa, esta posee una variación que va de los 3 (m) a 4 (m).
- Se pudo analizar que el uso y aplicaciones de tecnologías GPRS abarca una gran variedad de temas, esto debido a la facilidad de conectar un módulo a la red de telefonía móvil y a su vez a Internet.
- El sistema GPRS es una tecnología utilizada por parte de la telefonía celular la cual permite un acceso no solo a los servicios de telefonía móvil, también le da la capacidad de realizar una conexión con Internet, esta conexión es posible por la configuración de los nodos y equipos usados en el lado del proveedor.
- Se puede crear una red IoT, esto gracias a que existen módulos que permiten el acceso a una red GPRS, gracias a esta conexión es posible unirse a Internet y con el uso de una placa que controle un actuador se puede controlar cualquier dispositivo.
- Se consiguió armar un sistema físico que se adapte a utilizar los módulos seleccionados para este proyecto, y este a su vez pudo ser acoplado a un arnés para su uso de manera continua en exteriores.
- Con los cálculos respectivos se obtuvo el tiempo de duración de las baterías, la capacidad que poseen es suficiente para alimentar el sistema de manera constante por un tiempo de alrededor de 13 horas.
- Dependiendo de las características que sean necesarias se puede escoger entre varios módulos de geo-posicionamiento, así como tamaños, para la culminación de este proyecto de titulación se optó por el uso del SIM808 EVB V3.2 esto por traer incluida una antena de GPS y una antena GPRS, evitando realizar conexiones extras para poder utilizarlo.
- La alimentación de energía para el sistema se lo realiza calculando el consumo de corriente de cada módulo, los principales son el Sim808 EVB V3.2 y placa Arduino Nano, para los otros módulos como lo son el regulador de batería BMS y módulo Step Down el consumo de corriente es muy bajo, debido a eso no se lo consideró para el análisis.
- Al ingresar el código de Arduino por primera vez, la memoria fue insuficiente, lo que llevó a realizar cambios al código de manera que pueda realizar lo solicitado con un mejor rendimiento, el uso de librerías puede ahorrar o consumir espacio,

se debe tener en consideración los objetivos principales que debe cumplir Arduino y si es requerido el uso de librerías, en el trabajo de titulación desarrollado se usó dos librerías: una para obtener datos de geo posicionamiento para el Sim808 EVB V3.2 y otra para hacer uso de pines seriales utilizados para una mejor comunicación entre Arduino y módulo.

- Durante la investigación se encontraron placas Arduino de un menor tamaño en comparación con el Arduino Nano, pero en algunas de ellas era imposible subir el código ya que era necesario un módulo adicional para realizar la conexión con un computador, por este motivo se optó por el uso de Arduino Nano en este proyecto.
- Para tener las credenciales para el acceso a la respectiva red GPRS por medio de la antena GSM, se debe conocer la APN del proveedor de servicio de telefonía móvil, estos datos pueden cambiar dependiendo de cada operadora móvil, en algunos casos no son necesarios dichos parámetros, otro factor a tomar en cuenta es la cobertura que posee la operadora, de esto dependerá el envío de información hacia la base de datos, a pesar que el módulo Sim808EVB V3.2 pueda obtener los datos de posicionamiento si no llega a conectarse a la red GPRS será imposible el envío de información hacia la base de datos.
- El desarrollo de una interfaz que pueda mostrar la ubicación se lo consiguió realizando una programación en PHP y HTML5, estos dos fueron esenciales para crear una base de datos gracias al amigable trabajo con la plataforma Microsoft Azure la cual permite la visualización del sistema de geo-posicionamiento.
- Al realizar las pruebas del servidor web presentó problemas de conectividad con el servidor *TeamSpeak*, esto debido a que es necesario proteger la información que se está enviando usando Internet, para solucionar este problema se debe añadir un certificado de seguridad, para este caso se utiliza el *BaltimoreCyberTrustRoot.crt.pem*, este permite que los datos de geoposicionamiento viajen de manera segura por la red.
- Antes de subir la página web a Microsoft Azure fue necesario verificar su correcto funcionamiento, para ello se usó *PHPMyAdmin* este programa permite crear un host en la misma computadora, una vez comprobado el diseño y funcionalidad estará listo para subirlo a la nube.
- Para poder hacer uso de los recursos de Google *Maps* se debe tener una cuenta Google, con esta cuenta se puede hacer el pedido de una *API Google Maps*, la

cual se utilizó para poder visualizar de mejor manera las coordenadas obtenidas por el módulo Sim808 EVB V3.2.

- El uso de la tecnología *AJAX* permitió a la página web obtener la información de geoposicionamiento provista por el *ThingSpeak* sin la necesidad de cargar la página de manera continua.
- Microsoft Azure provee servicios de nube de tipo PaaS, se crea una máquina virtual en esta plataforma la cual ejecuta la página web de manera automática, para lo cual se usó el crédito monetario provisto por Azure en la creación de la cuenta gratuita de un año, si se desea mantener el uso de la página web el costo será variado, esto se debe a que la facturación se realiza por uso de los servicios que está dando Microsoft Azure.
- Los pines del Arduino Nano y Sim808 EVB V3.2 tienen un valor de voltaje y corriente máximo que pueden soportar, para no provocar un daño se revisó las hojas técnicas de manera detallada en donde se encontraron los valores máximos que soportan cada uno de los dispositivos, siendo 7 (V) el voltaje recomendado para un trabajo óptimo del Arduino Nano y Sim808 EVB V3.2.
- Para evitar problemas en la construcción de los arneses que soporten los sistemas de geo-posicionamiento diseñados, se optó por la compra de dos modelos a los cuales se les realizaron modificaciones para sujetar cada uno de los sistemas hechos.
- Para realizar el cálculo del tiempo que va a durar el sistema encendido se debe tener como dato el nivel mínimo de descarga de cada batería, así se evita causar daños en las baterías que se estén utilizando, para obtener este porcentaje se lo debe buscar en la hoja técnica, este dependerá de cada modelo de batería que se quiera utilizar, por ello en este proyecto de titulación el valor de descarga mínimo recomendado por el fabricante es de 3 (V) lo que equivale a un 71%.
- El sistema de geo-posicionamiento de mascotas puede ser implementado en otras áreas que se basen en la geo localización, para ello se cuentan con módulos más pequeños, varios de estos módulos solo se encuentran en otros países por lo cual si se desea utilizarlos se los tiene que importar.
- Con los conocimientos adquiridos durante la Carrera de Electrónica y Telecomunicaciones se creó un sistema físico robusto que permitió usar módulos para cumplir con el propósito de localización GPS usando la API de Google *Maps*.

4.2 Recomendaciones

- Para tener un buen funcionamiento de Arduino lo recomendado es trabajar con 7 (V), ya que en sus especificaciones técnicas es un voltaje que no afectaría al regulador interno, en nuestro proyecto se alimentó con un voltaje en los pines Vin y GND, en estos pines se debe tener cuidado ya que no cuentan con protección contra polaridad, ni diodos de protección contra caída de voltajes, por eso se optó por un Step Down que nos mantiene un voltaje constante y un buen funcionamiento del sistema.
- Hay que tener en cuenta que la placa Sim808 EVB V3.2 cuando esté detectando la señal tiene picos de corriente de 2 (A), por ende, la alimentación del circuito mínimo debe contar con 2,5 a 3 (A) para no tener mal funcionamiento del sistema.
- Durante el desarrollo del plan de titulación se encontraron antenas tanto para GSM, así como para GPS, estas poseen distintas longitudes y características, en caso de requerir una antena más pequeña con rasgos diferentes a las que el módulo Sim808 EVB V3.2 trae por default, se puede acceder a ellas tomando en cuenta los conectores del módulo.
- El Sim808 posee un pulsador que permite activar la red de telefonía celular y el GPS, en caso de fallo o pérdida de señal este se desactiva solo, dejando el programa obsoleto pero existe una forma de reinicio del Sim808 por código y es soldando un cable a un pad del pin D9 de la placa, de esa manera mediante código se puede controlar un pulso de 3 s que vendría a ser un reinicio de la red celular, en el programa del proyecto se tuvo en cuenta un sistema de fallo para evitar esos apagones del servicio ya que al estar el sistema en una carcasa se dificultaba el acceso a él, de este modo al momento de perder la señal se ejecutará el código contra caídas de señal el cual reinicia el Sim808 por código.
- En caso de querer aumentar la capacidad de las baterías basta con realizar una configuración en paralelo, de esta manera se sumarán las corrientes y aumenta la capacidad, para tener el nuevo tiempo de duración solo debe remplazarse el valor total de la capacidad en las fórmulas ya usadas con anterioridad.
- Al usar la operadora Tuenti no posee cobros por el uso de la red APN debido que es una red MNVO y en sus planes solo posee paquetes de datos móviles 4G, mensajería y llamadas, al usar GPRS que es 2.5G no detecta el cobro del servicio al utilizar su APN.

- Al utilizar *ThingSpeak* como base de datos esta nos permite crear una cuenta que permite receptar hasta 3 millones de mensajes/año de datos para su análisis, por 8 canales de datos y de forma gratuita, al poseer una API Key se pueden desarrollar diversos proyectos que requieran la trasmisión de datos por GPRS.
- Para desarrollar una aplicación o página web es recomendable probar todas sus funcionalidades en un servidor local ya que se pueden probar todos los fallos sin necesidad de tener que subir y actualizar códigos ya subidos a una nube o hosting.
- En el diseño de páginas web todos los estilos que se quieran utilizar mediante CS3, puede ocurrir que en algún momento la página no quiera adaptarse a la configuración y esto es debido a que los exploradores web como Chrome o Firefox almacenan un cache de estilo, por lo tanto, para evitar este problema es aconsejable recargar el cache de la página pulsando *Ctrl+ Shift + R*.
- En ciertas ocasiones al crear páginas web debemos darnos cuenta que se deben poder visualizar tanto en computadoras como en teléfonos móviles, en nuestro caso siendo una aplicación de rastreo de mascotas se utilizó instrucciones de JavaScript para hacer que la página sea responsiva.
- En algunos casos de ciertas plataformas al momento de subir archivos a una nube que posea una base de datos, estas piden certificados para tener una conexión segura para la transmisión de datos, en nuestro caso ese certificado permite la etapa de conexión con Microsoft Azure y este debe ir detallado en la etapa de conexión ya que ahí están todas las credenciales para la comunicación con las bases de datos.
- Es aconsejable tener archivos de control en la página web o dejar mensajes para así comprobar el correcto funcionamiento ya que con estos mensajes se pueden detectar ciertos problemas, en el código que está anexado en *One Drive* se pueden visualizar los códigos de control que están comentados y que ayudan mucho para ver si ciertas funciones se ejecutan.
- Por las características de la caja adquirida, así como las modificaciones realizadas, se debe evitar mojarla; para tener un módulo con mayor resistencia se debe comprar una caja con protección contra salpicaduras y polvo, también se puede remplazar las antenas GPS y GSM por unas de distintas características.

5 REFERENCIAS BIBLIOGRÁFICAS

[1] L. Española, «youtube.com,» Lesics Española, 28 Noviembre 2019. [En línea]. Available: https://www.youtube.com/watch?v=BOoY4rDRf0s. [Último acceso: 29 Junio 2021].

[2] J. Leceta, «autobild.es,» AXEL SPRINGER ESPAÑA, 8 Octubre 2015. [En línea]. Available: https://www.autobild.es/practicos/como-funciona-gps-tu-coche-268547. [Último acceso: 28 Junio 2021].

[3] mototraking, «mototraking.com,» 24 Mayo 2018. [En línea]. Available: https://www.mototraking.com/que-es-gprs-y-como-se-usa-con-dispositivos-de-seguimiento-gps/. [Último acceso: 28 Junio 2021].

[4] 330ohms, «blog.330ohms.com,» 330ohms, 18 Octubre 2019. [En línea]. Available: https://blog.330ohms.com/2019/10/18/guia-definitiva-de-comandos-at-gsmbt/. [Último acceso: 28 Junio 2021].

[5] Arduino_software, «arduino.cc,» Arduino, 25 Junio 2021. [En línea]. Available: https://www.arduino.cc/en/software. [Último acceso: 25 Junio 2021].

[6] G. B, «https://www.hostinger.es/,» hostinger.es - Servicios de Hosting Web Premium, Cloud, VPS & Registro de Dominios., 13 Mayo 2019. [En línea]. Available: https://www.hostinger.es/tutoriales/que-es-ajax. [Último acceso: 2 Julio 2021].

[7] M. K. W. M. W. O. Siegmund M. Red, GSM and Personal Communications Handbook, Boston London: 1998 ARTECH HOUSE, INC., 1998.

[8] Arduino_nano, «store.arduino.cc,» Arduino.cc, 17 Junio 2021. [En línea]. Available: https://store.arduino.cc/usa/arduino-nano. [Último acceso: 25 Junio 2021].

[9] A. Aqeel, «theengineeringprojects.com,» TheEngineeringProjects.com, 25 Junio 2018. [En línea]. Available: https://www.theengineeringprojects.com/2018/06/introduction-to-arduino-nano.html. [Último acceso: 25 Junio 2021].

[10] v. e. linea, «youtube.com,» 5 Diciembre 2017. [En línea]. Available: https://www.youtube.com/watch?v=BzUHAYvVQl0. [Último acceso: 25 Junio 2021].

[11] seeedstudio, «seeedstudio.com,» Seeed Technology Co.,Ltd, 18 Junio 2021. [En línea]. Available: https://wiki.seeedstudio.com/Mini_GSM_GPRS_GPS_Breakout_SIM808/. [Último acceso: 25 Junio 2021].

[12] O. S. O. S. Offered, «spanish.alibaba.com,» OEM Service OfferedDesign Service Offered, 18 Junio 2021. [En línea]. Available: https://spanish.alibaba.com/product-detail/sim808-module-gsm-gprs-gps-development-board-ipx-sma-gsm-gps-antenna-support-2g-network-for-arduino-raspberry-pi-62206322700.html. [Último acceso: 25 Junio 2021].

[13] servotronik, «servotronik.com.co,» 18 Junio 2021. [En línea]. Available: https://www.servotronik.com.co/index.php/producto/modulo-sim-808-gsm-gprs-gps/. [Último acceso: 25 Junio 2021].

[14] 330ohms, «330ohms.com,» 330ohms, 22 Junio 2020. [En línea]. Available: https://blog.330ohms.com/2020/06/22/que-diferencias-hay-entre-una-li-po-y-una-li-ion/. [Último acceso: 25 Junio 2021].

[15] L. EEMB Co., «https://www.eemb.com/,» EEMB Co., Ltd, Noviembre 2010. [En línea]. Available: https://www.ineltro.ch/media/downloads/SAAItem/45/45958/36e3e7f3-2049-4adb-a2a7-79c654d92915.pdf. [Último acceso: 25 Junio 2021].

[16] FerreHogar, «ferrehogar.es,» ferrehoga, 22 Septiembre 2019. [En línea]. Available: https://www.youtube.com/watch?v=dYTf_Joimrw. [Último acceso: 25 Junio 2021].

[17] onsemi.com, «http://onsemi.com,» Semiconductor Components Industries, LLC, Noviembre 2008. [En línea]. Available: https://www.onsemi.com/pdf/datasheet/lm2596-d.pdf. [Último acceso: 25 Junio 2021].

[18] Argos, «https://nomadaselectronicos.wordpress.com/,» Nómadas Electrónicos, 12 Abril 2015. [En línea]. Available:

https://nomadaselectronicos.wordpress.com/2015/04/12/convertidores-dcdc-buck/. [Último acceso: 25 Junio 2021].

[19] E. P. P. a. Paso, «https://www.youtube.com/,» 20 Febrero 2021. [En línea]. Available: https://www.youtube.com/watch?v=usGyOARttpQ. [Último acceso: 25 Junio 2021].

[20] U. Electronics, «https://uelectronics.com/,» UNIT Electronics, 22 Abril 2019. [En línea]. Available: https://uelectronics.com/producto/cargador-para-paquete-de-3-baterias-de-litio-10a-bms/. [Último acceso: 25 Junio 2021].

[21] H. Saavedra, «https://www.youtube.com/,» RECIxCLA & MAS, 3 Mayo 2020. [En línea]. Available: https://www.youtube.com/watch?v=PI1iLQ_k3MY. [Último acceso: 25 Junio 2021].

[22] MatLab, «https://thingspeak.com/,» The MathWorks, Inc., 18 Junio 2021. [En línea]. Available: https://thingspeak.com/pages/learn_more. [Último acceso: 25 Junio 2021].

[23] M. Azure, «https://azure.microsoft.com/,» Microsoft , [En línea]. Available: https://azure.microsoft.com/es-es/overview/. [Último acceso: 25 Junio 2021].

[24] F. &. Chelsee, «https://www.latiendadefrida.com/,» 9 Abril 2021. [En línea]. Available: https://www.latiendadefrida.com/blogs/juega-con-tu-perro/para-salir-a-pasear-que-es-mejor-para-un-perro-un-collar-o-una-pechera. [Último acceso: 25 Junio 2021].

[25] N. Tapia, «BaulPHP,» BaulPHP, 06 Febrero 2021. [En línea]. Available: https://www.baulphp.com/ventajas-y-desventajas-del-lenguaje-php/#Ventajas_del_lenguaje_PHP. [Último acceso: 25 Junio 2021].

[26] C. M. Orfali, «toadworld.com,» blog.toadworld.com, 04 Abril 2017. [En línea]. Available: https://blog.toadworld.com/2017/04/04/que-es-phpmyadmin-y-como-podemos-gestionar-la-base-de-datos-mysql-con-esta-herramienta. [Último acceso: 25 Junio 2021].

[27] I. d. Souza, «rockcontent.com,» rockcontent.com, 9 Marzo 2020. [En línea]. Available: https://rockcontent.com/es/blog/php/. [Último acceso: 25 Junio 2021].

[28] M. contributors, «developer.mozilla.org,» MDN, 22 Junio 2021 Web Docs. [En línea]. Available: https://developer.mozilla.org/es/docs/Learn/JavaScript/First_steps/What_is_JavaScript. [Último acceso: 25 Junio 2021].

[29] M. contributors, «developer.mozilla.org,» MDN Web Docs , 22 Junio 2021. [En línea]. Available: https://developer.mozilla.org/es/docs/Web/CSS. [Último acceso: 25 Juio 2021].

[30] M. contributors, «developer.mozilla.org,» MDN Web Docs, 22 Junio 2021. [En línea]. Available: https://developer.mozilla.org/es/docs/Glossary/CSS. [Último acceso: 25 Junio 2021].

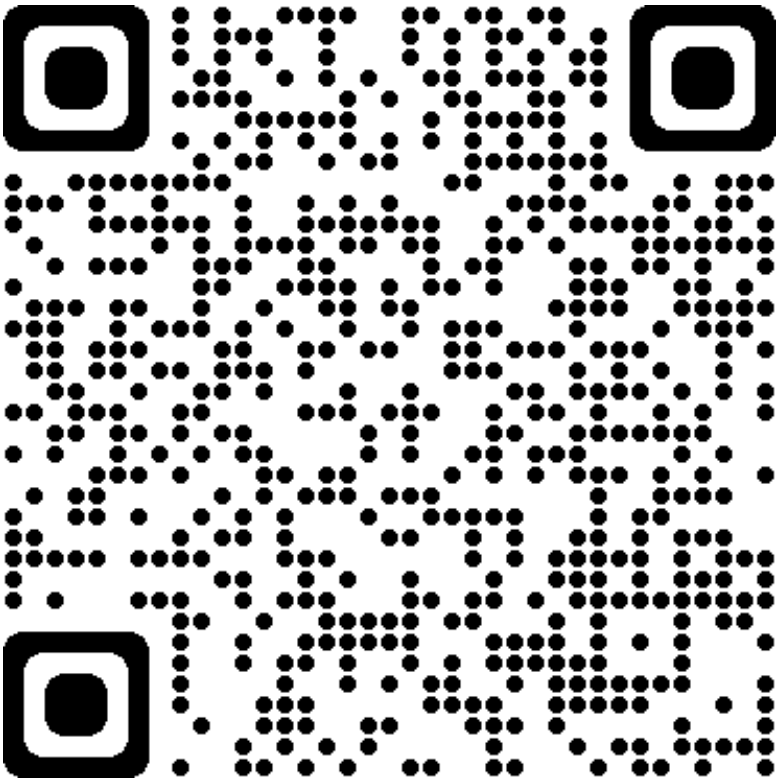
[31] M. contributors, «developer.mozilla.org,» MDN Web Docs , 1 Junio 2021. [En línea]. Available: https://developer.mozilla.org/en-US/docs/Web/HTML. [Último acceso: 25 Junio 2021].

[32] D. WEB, «blog.aulaformativa.com,» aulaformativa.com, 31 Octubre 2020. [En línea]. Available: https://blog.aulaformativa.com/definicion-usos-ventajas-lenguaje-html5/. [Último acceso: 25 Junio 2021].

ANEXOS

ANEXO 1: FUNCIONES ARDUINO NANO

Para acceder a la información escanear la imagen QR

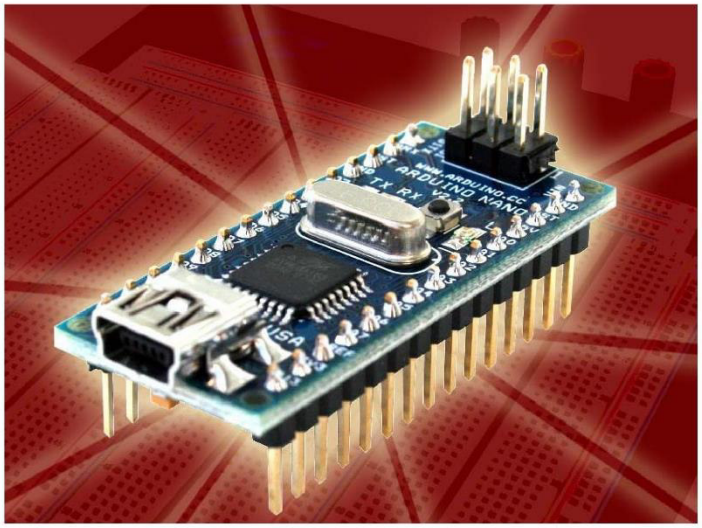


Anexo 1 Funciones Arduino Nano

ANEXO 2: HOJA DE DATOS ARDUINO NANO

Arduino Nano (V2.3)

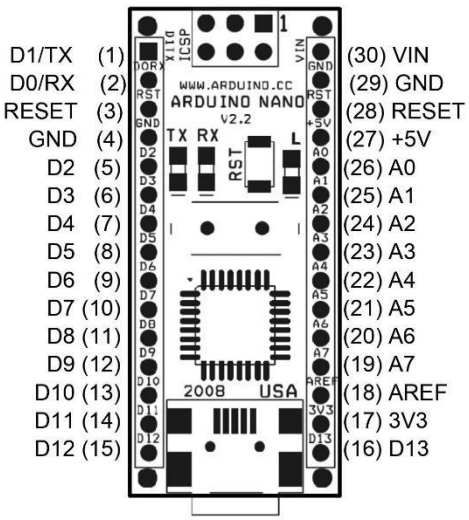
User Manual



More information:

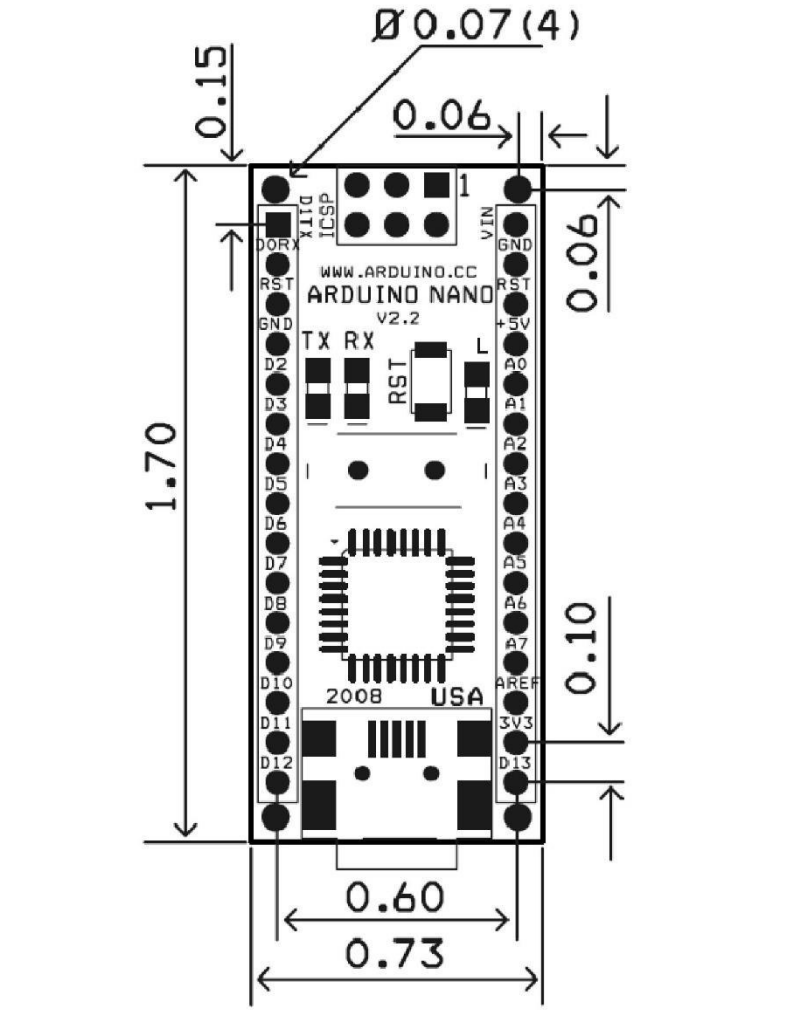
www.arduino.cc Rev. 2.3

Arduino Nano Pin Layout



Pin No.	Name	Type	Description
1-2, 5-16	D0-D13	I/O	Digital input/output port 0 to 13
3, 28	RESET	Input	Reset (active low)
4, 29	GND	PWR	Supply ground
17	3V3	Output	+3.3V output (from FTDI)
18	AREF	Input	ADC reference
19-26	A7-A0	Input	Analog input channel 0 to 7
27	+5V	Output or Input	+5V output (from on-board regulator) or +5V (input from external power supply)
30	VIN	PWR	Supply voltage

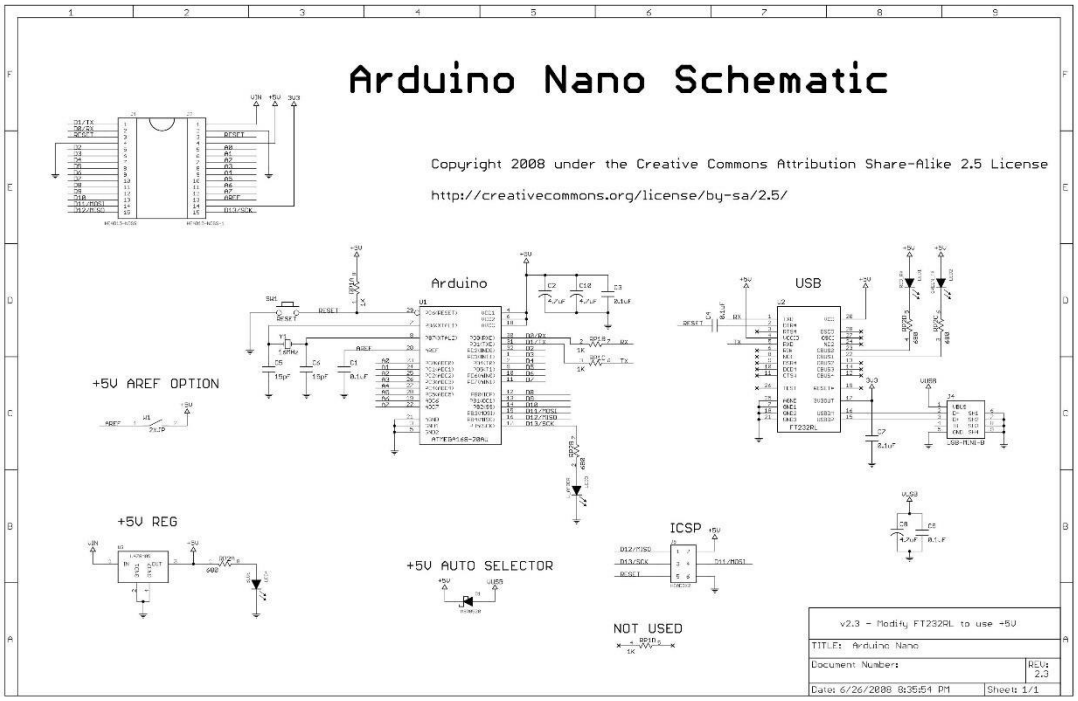
Arduino Nano Mechanical Drawing



ALL DIMENTIONS ARE IN INCHES

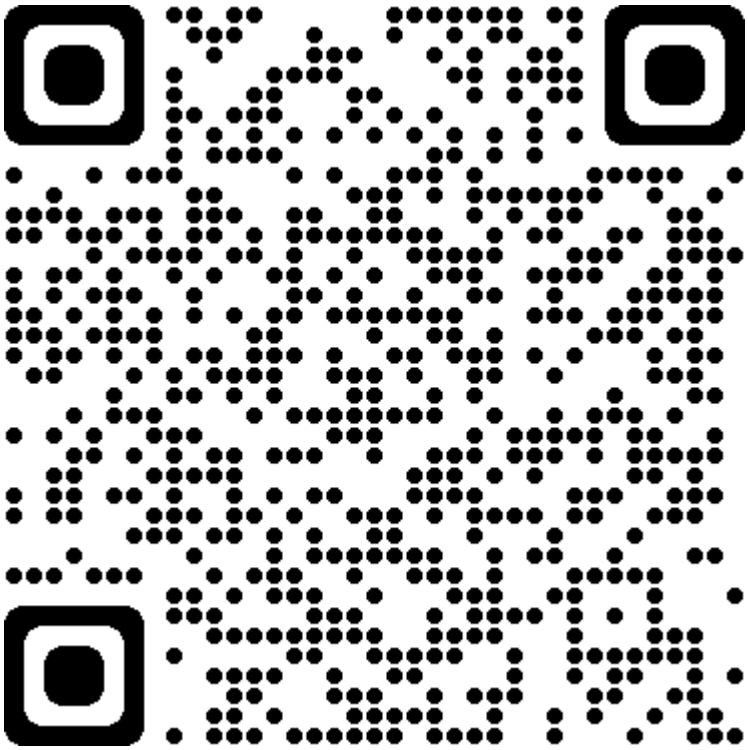
Arduino Nano Bill of Material

Item Number	Qty.	Ref. Dest.	Description	Mfg. P/N	MFG	Vendor P/N	Vendor
1	5	C1,C3,C4,C7,C9	Capacitor, 0.1uF 50V 10% Ceramic X7R 0805	C0805C104K5RACTU	Kemet	80-C0805C104K5R	Mouser
2	3	C2,C8,C10	Capacitor, 4.7uF 10V 10% Tantalum Case A	T491A475K010AT	Kemet	80-T491A475K010	Mouser
3	2	C5,C6	Capacitor, 18pF 50V 5% Ceramic NOP/COG 0805	C0805C180J5GACTU	Kemet	80-C0805C180J5G	Mouser
4	1	D1	Diode, Schottky 0.5A 20V	MBR0520LT1G	ONSemi	863-MBR0520LT1G	Mouser
5	1	J1,J2	Headers, 36PS 1 Row	68000-136HLF	FCI	649-68000-136HLF	Mouser
6	1	J4	Connector, Mini-B Recept Rt. Angle	67503-1020	Molex	538-67503-1020	Mouser
7	1	J5	Headers, 72PS 2 Rows	67996-272HLF	FCI	649-67996-272HLF	Mouser
8	1	LD1	LED, Super Bright RED 100mcd 640nm 120degree 0805	APT2012SRCPRV	Kingbright	604-APT2012SRCPRV	Mouser
9	1	LD2	LED, Super Bright GREEN 50mcd 570nm 110degree 0805	APHCM2012CGCK-F01	Kingbright	604-APHCM2012CGCK	Mouser
10	1	LD3	LED, Super Bright ORANGE 160mcd 601nm 110degree 0805	APHCM2012SECK-F01	Kingbright	04-APHCM2012SECK	Mouser
11	1	LD4	LED, Super Bright BLUE 80mcd 470nm 110degree 0805	LTST-C170TBKT	Lite-On Inc	160-1579-1-ND	Digikey
12	1	R1	Resistor Pack, 1K +/-5% 62.5mW 4RES SMD	YC164-JR-071KL	Yageo	YC164J-1.0KCT-ND	Digikey
13	1	R2	Resistor Pack, 680 +/-5% 62.5mW 4RES SMD	YC164-JR-07680RL	Yageo	YC164J-680CT-ND	Digikey
14	1	SW1	Switch, Momentary Tact SPST 150gf 3.0x2.5mm	B3U-1000P	Omron	SW1020CT-ND	Digikey
15	1	U1	IC, Microcontroller RISC 16kB Flash, 0.5kB EEPROM, 23 I/O Pins	ATmega168-20AU	Atmel	556-ATMEGA168-20AU	Mouser
16	1	U2	IC, USB to SERIAL UART 28 Pins SSOP	FT232RL	FTDI	895-FT232RL	Mouser
17	1	U3	IC, Voltage regulator 5V, 500mA SOT-223	UA78M05CDCYRG3	TI	595-UA78M05CDCYRG3	Mouser
18	1	Y1	Cystal, 16MHz +/-20ppm HC-49/US Low Profile	ABL-16.000MHZ-B2	Abracon	815-ABL-16-B2	Mouser


Arduino Nano Schematic
Copyright 2008 under the Creative Commons Attribution Share-Alike 2.5 License
http://creativecommons.org/license/by-sa/2.5/
Arduino
USB
+5V AREF OPTION
+5V REG
+5V AUTO SELECTOR
ICSP
NOT USED
v2.3 - Modify FT232RL to use +5V
TITLE: Arduino Nano
Document Number:
REV: 2.3
Date: 6/26/2008 8:35:54 PM
Sheet: 1/1

ANEXO 3: HOJA DE DATOS SIM808 EVB V3.2

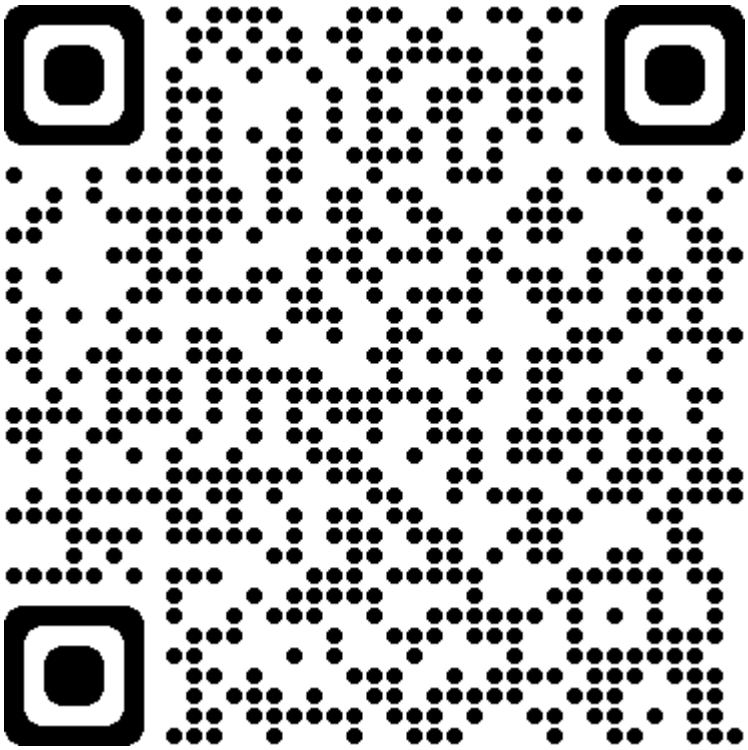
Para acceder a la información escanear el código QR



Anexo 3 Hoja de datos SIM808 EVB V3.2

ANEXO 4: HOJA DE DATOS BATERÍA LIO ION LITIO 1865

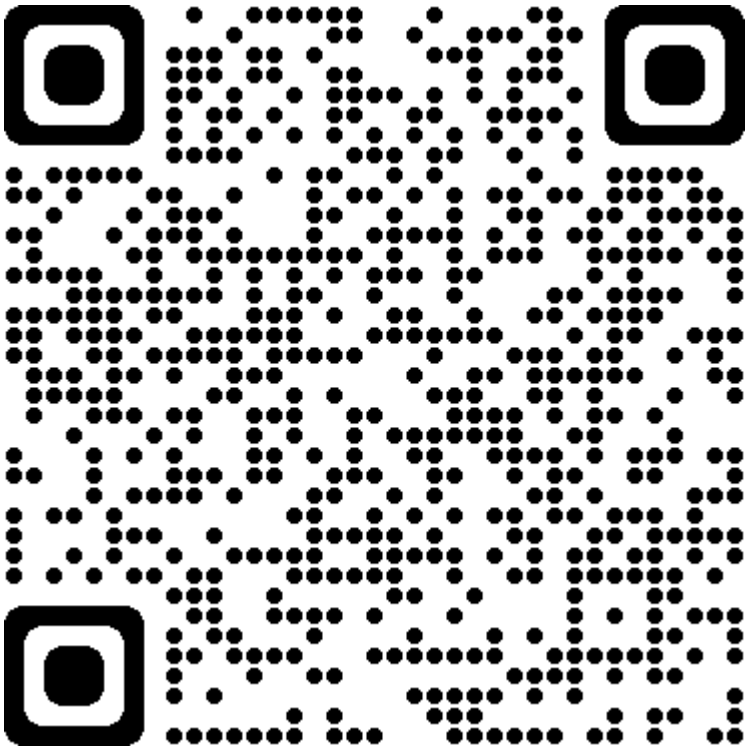
Para acceder a la información escanear el código QR



Anexo 4 Hoja de datos batería Li-Ion Litio 18650

ANEXO 5: HOJA DE DATOS MÓDULO LM2596S

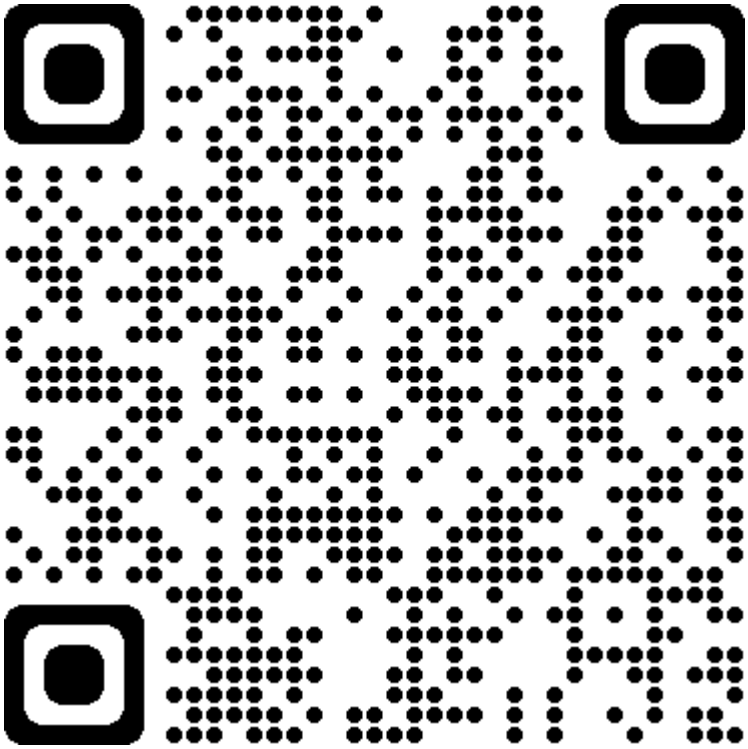
Para acceder a la información escanear el código QR



Anexo 5 Hoja de datos LM2596S

ANEXO 6: HOJA DE DATOS MÓDULO BMS 3S Y 2S

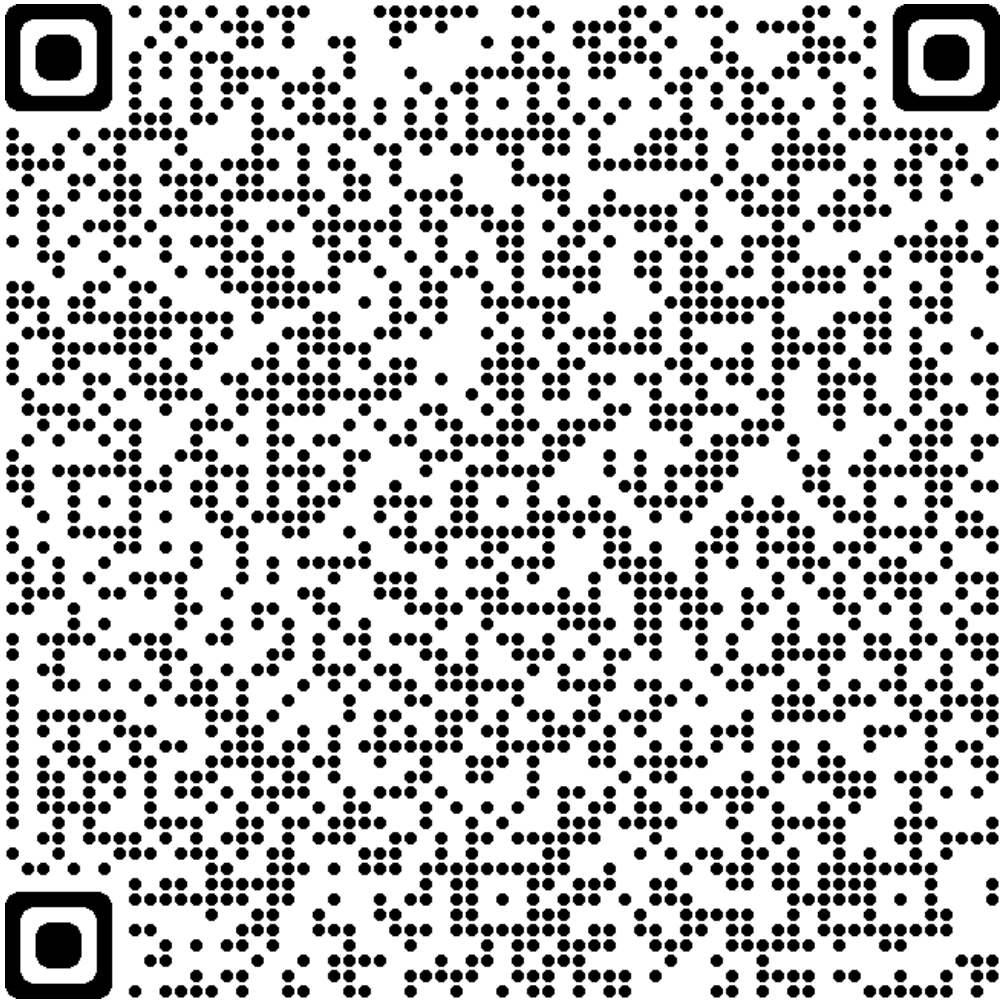
Para acceder a la información escanear el código QR



Anexo 6 Hoja de datos módulo BMS 3S y 2S

ANEXO 7: CÓDIGO PROGRAMACION PÁGINA WEB

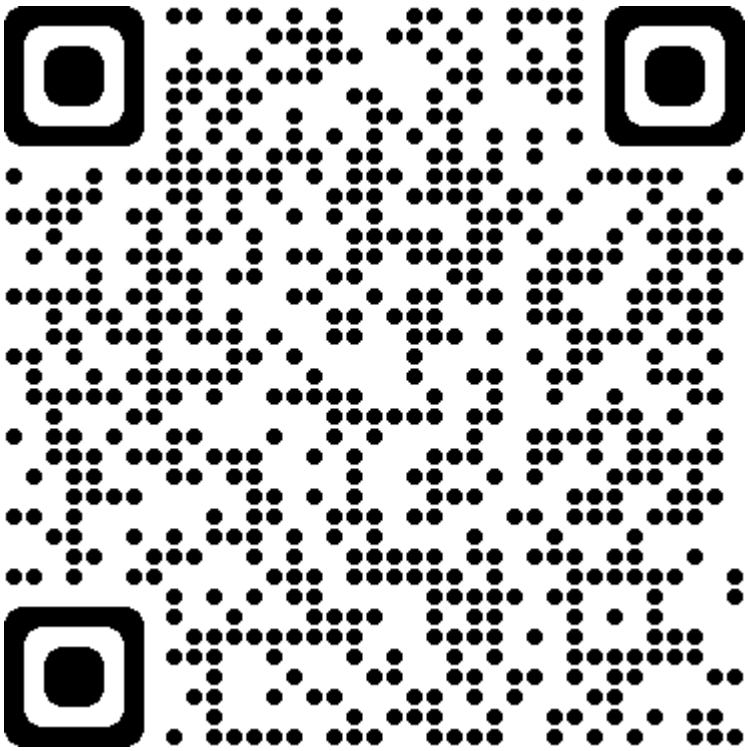
Para acceder a la información escanear el código QR



Anexo 7 Archivos de programación

ANEXO 8: COMANDOS AT

Para acceder a la información escanear el código QR



Anexo 8 Comandos AT para Sim808

ANEXO 9: CÓDIGO ARDUINO

```
#include <DFRobot_sim808.h>
#include <SoftwareSerial.h>
#include <String.h>
SoftwareSerial Sim800Serial(10, 11);//Configuración de los pines serial
por software
char lat[15]; //matriz con caracteres sin inicializar
char lon[15]; //matriz con caracteres sin inicializar


DFRobot_SIM808 sim808(&Sim800Serial);//Connect RX,TX,PWR,
static const int PowerPin =9; // PIN9 ARDUINO PIN D9 SIM800


void setup()
{
  Sim800Serial.begin(19200);//Arduino se comunica con el SIM800 a una
velocidad de 19200bps
  Serial.begin(9600);//Velocidad del puerto serial de arduino

  Serial.println("Empezando");
  delay(20000);//Tiempo prudencial para el escudo inicie sesión de red
con tu operador
    while(!sim808.init()) {
       delay(1000);
        Serial.print("Sim808 init error\r\n");
        pinMode(PowerPin, OUTPUT);
        //to turn on the GSM module -software switch instead Power Key
        digitalWrite(PowerPin, HIGH);
        delay (3000);
        digitalWrite (PowerPin, LOW);
        delay (10000); //delay set for 10 seconds

        //to turn off the GSM module
       // digitalWrite(PowerPin, HIGH);
        //delay (3000);
        //digitalWrite (PowerPin, LOW);
    }
    Serial.println("Sim808 init success");

  if( sim808.attachGPS())
      Serial.println("Open the GPS power success");
   else
      Serial.println("Open the GPS power failure");

}
```

```
void loop()
{
  comandosAT();//Llama a la función comandosAT
if(Sim800Serial.available())//Verificamos si hay datos disponibles desde
el SIM800
Serial.write(Sim800Serial.read());//Escribir datos
 while(!sim808.init()) {
       delay(1000);
        Serial.print("Sim808 init error\r\n");
        pinMode(PowerPin, OUTPUT);
        //to turn on the GSM module -software switch instead Power Key
        digitalWrite(PowerPin, HIGH);
        delay (3000);
        digitalWrite (PowerPin, LOW);
        delay (10000); //delay set for 10 seconds
    }
    Serial.println("Sim808 init success");

  if( sim808.attachGPS())
      Serial.println("Open the GPS power success");
   else
      Serial.println("Open the GPS power failure");

}
void comandosAT(){
Sim800Serial.println("AT+CIPSTATUS");//Consultar el estado actual de la
conexión
delay(2000);
Sim800Serial.println("AT+CIPMUX=0");//comando configura el dispositivo
para una conexión IP única o múltiple 0=única
delay(3000);
mostrarDatosSeriales();
Sim800Serial.println("AT+CGATT=1");///Iniciamos la conexión GPRS
delay(3000);
mostrarDatosSeriales();
Sim800Serial.println("AT+CSTT=\"igprs.tuenti.com.ec\",\"ctigprs\",\"ct
igpr999\"");//comando configura el APN, nombre de usuario y
contraseña."gprs.movistar.com.ar","wap","wap"->Movistar Arg.
delay(1000);
mostrarDatosSeriales();
Sim800Serial.println("AT+CIICR");//REALIZAR UNA CONEXIÓN INALÁMBRICA
CON GPRS O CSD
delay(3000);
mostrarDatosSeriales();
Sim800Serial.println("AT+CIFSR");// Obtenemos nuestra IP local
delay(2000);
mostrarDatosSeriales();
```

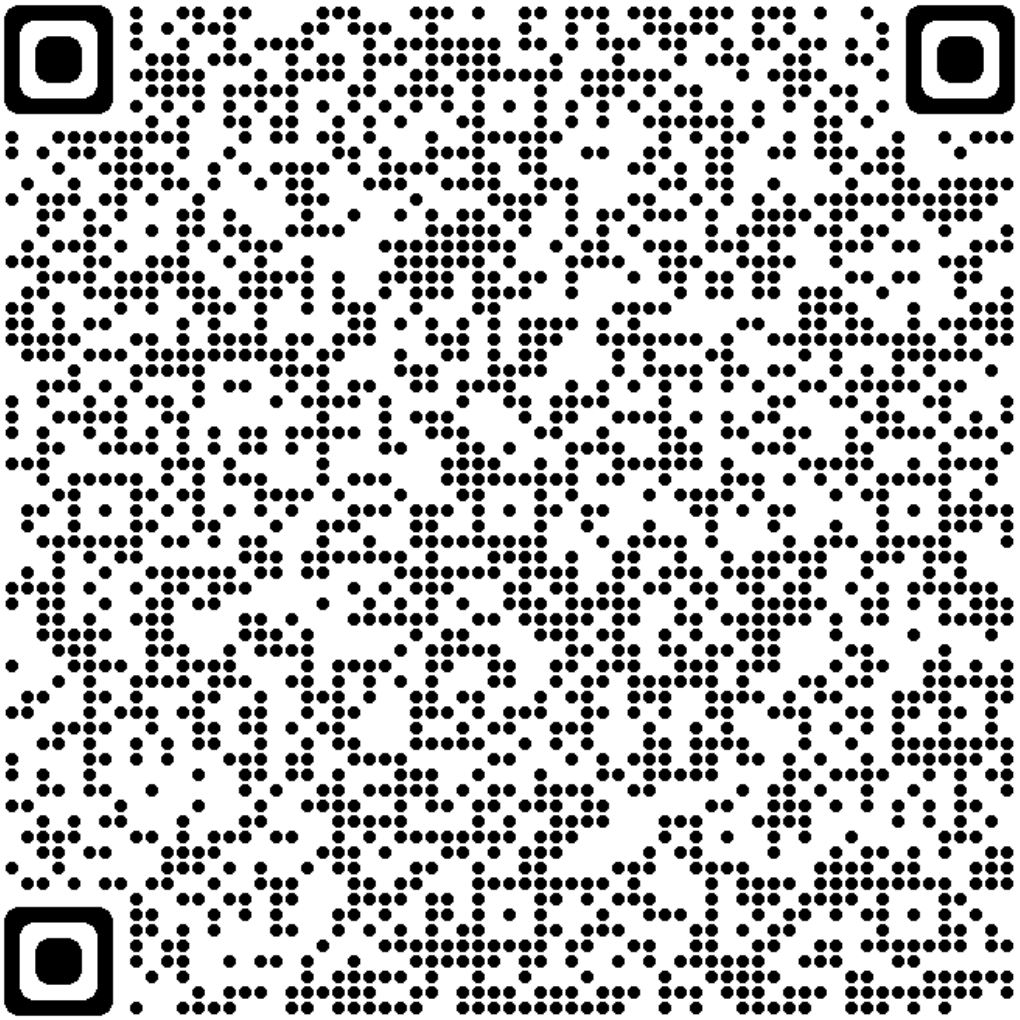
```
Sim800Serial.println("AT+CIPSPRT=0");//Establece un indicador '>' al
enviar datos
GPS();
delay(3000);
mostrarDatosSeriales();
Sim800Serial.println("AT+CIPSTART=\"TCP\",\"api.thingspeak.com\",\"80\
"");//Indicamos el tipo de conexión, url o dirección IP y puerto al que
realizamos la conexión
delay(6000);
mostrarDatosSeriales();
Sim800Serial.println("AT+CIPSEND");//ENVÍA DATOS A TRAVÉS DE una
CONEXIÓN TCP O UDP
delay(4000);
mostrarDatosSeriales();
String                                                    datos="GET
https://api.thingspeak.com/update?api_key=BRDQOASQ4BLF2SHK";    //envia
una cadena de datos,
datos=datos+"&field1="+ String(lat);
datos=datos+"&field2="+ String(lon);
Sim800Serial.println(datos);//Envía datos al servidor remoto
delay(4000);
mostrarDatosSeriales();
Sim800Serial.println((char)26); // End AT command with a ^Z, ASCII code
26
delay(5000);//Ahora esperaremos una respuesta pero esto va a depender
de las condiciones de la red y este valor quizá debamos modificarlo
dependiendo de las condiciones de la red
Sim800Serial.println();
mostrarDatosSeriales();
Sim800Serial.println("AT+CIPSHUT");//Cierra la conexión(Desactiva el
contexto GPRS PDP)
delay(5000);
mostrarDatosSeriales();
}
void mostrarDatosSeriales()//Muestra los datos que va entregando el
sim808
{
while(Sim800Serial.available()!=0)
Serial.write(Sim800Serial.read());
}

void GPS(){//Función para la lectura del GPS
   while(!sim808.getGPS())
    {

    }

      Serial.print(sim808.GPSdata.year);
      Serial.print("/");
```

```
      Serial.print(sim808.GPSdata.month);
      Serial.print("/");
      Serial.print(sim808.GPSdata.day);
      Serial.print(" ");
      Serial.print(sim808.GPSdata.hour);
      Serial.print(":");
      Serial.print(sim808.GPSdata.minute);
      Serial.print(":");
      Serial.print(sim808.GPSdata.second);
      Serial.print(":");
      Serial.println(sim808.GPSdata.centisecond);
      Serial.print("latitude :");
      Serial.println(sim808.GPSdata.lat,9);
      Serial.print("longitude :");
      Serial.println(sim808.GPSdata.lon,9);
      Serial.print("speed_kph :");
      Serial.println(sim808.GPSdata.speed_kph);
      Serial.print("heading :");
      Serial.println(sim808.GPSdata.heading);
      Serial.println();
//      Serial.println(datos1);

      float la = sim808.GPSdata.lat;         //
      float lo = sim808.GPSdata.lon;         //valores flotantes
      float ws = sim808.GPSdata.speed_kph;   //

    dtostrf(la, 7, 8, lat); //toma los valores de la y divide en 7
valores para entero y 8 valores para los decimales, almacena en las
variables lat.
    dtostrf(lo, 7, 8, lon); //toma los valores de lo y divide en 7
valores para entero y 8 valores para los decimales, almacena en las
variables lon.

}
```

Anexo 9 Código Arduino

ANEXO 10: MANTENIMIENTO MÓDULOS GEO-POSICIONAMIENTO

Para acceder a la información escanear el código QR



Anexo 10 MANTENIMIENTO MÓDULOS GEO-POSICIONAMIENTO