UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE QUITO

CARRERA DE TELECOMUNICACIONES

DISEÑO DE UN SISTEMA GEOLOCALIZADOR PARA VEHÍCULOS DE CARGA EN UNA RED IOT Y CLOUD.

Trabajo de titulación previo a la obtención del
Título de Ingeniero en Telecomunicaciones

AUTOR: DIEGO JAVIER LEÓN SALGUERO

TUTOR: LUIS GERMÁN OÑATE CADENA

Quito, Ecuador

2022

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Yo, Diego Javier León Salguero con documento de identificación N° 0502965338; manifiesto que:

Soy el autor y responsable del presente trabajo; y, autorizo a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Quito, 01 de septiembre del año 2022

Atentamente,

_____

Diego Javier León Salguero

0502965338

CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

Yo, Diego Javier León Salguero con documento de identificación No. 0502965338, expreso mi voluntad y por medio del presente documento cedo a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que soy autor del proyecto técnico "Diseño de un sistema geolocalizador para vehículos de carga en una red IoT y Cloud", el cual ha sido desarrollado para optar por el título de: Ingeniero en Telecomunicaciones, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribo este documento en el momento que hago la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana

Quito, 01 de septiembre del año 2022

Atentamente,

_____

Diego Javier León Salguero

0502965338

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Luis Germán Oñate Cadena con documento de identificación N° 1712157401, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: DISEÑO DE UN SISTEMA GEOLOCALIZADOR PARA VEHÍCULOS DE CARGA EN UNA RED IOT Y CLOUD, realizado por Diego Javier León Salguero con documento de identificación N° 0502965338, obteniendo como resultado final el trabajo de titulación bajo la opción Proyecto Técnico que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Quito, 01 de septiembre del año 2022

Atentamente,

_____

Ing. Luis Germán Oñate Cadena, MSc
1712157401

DEDICATORIA

Con mucho orgullo yo Diego Javier León Salguero dedico mi trabajo a mis padres, Ángel León y Susan Salguero por el apoyo brindado y todo el enorme sacrificio que han puesto en mí para que logre todas mis metas y pueda alcanzar mis sueños, gracias por ayudarme a construir un camino de bien y de profesionalismo para defenderme en la sociedad, gracias por la educación y la atención especial puesta en mí, además de la confianza y cariño que me tienen. También dedico mi trabajo a mis hermanas María León y Paola León que con su voluntad pudieron ayudarme y gracias a su fuerte compañía hicieron más fácil mi camino universitario.

De igual forma quiero dedicar mi trabajo a Juan Carlos Paredes León que ha sabido darme fuerzas y ha tenido siempre palabras de aliento para que pueda salir adelante, que con su humildad, empatía y carisma supo alegrar mis días malos y supo aumentar de alegría mis días buenos.

AGRADECIMIENTO

Primeramente, agradezco a Dios por darme salud y constancia en todo mi periodo universitario, quiero agradecer a la empresa TRANSPSUR S.A por haber hecho posible la instalación de mi proyecto técnico en sus vehículos y por la confianza puesta en mi desde un inicio.

Agradecer a la Universidad Politécnica Salesiana y a cada uno de los docentes que me brindaron de su conocimiento e hicieron posible que logre culminar una meta más en mi vida.

Quiero agradecer a mi tutor Luis Germán Oñate por sus valiosos aportes, consejos, comentarios y por brindarme un poco de su experiencia en el desarrollo de mi proyecto técnico.

Finalmente, expreso mi profundo agradecimiento a toda mi familia cercana y a la que se encuentra lejos, por el apoyo incondicional y por el amor que me tienen.

ÍNDICE GENERAL

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN i
CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA ii
CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN iii
DEDICATORIA 1
AGRADECIMIENTO 2
ÍNDICE GENERAL 3
ÍNDICE DE FIGURAS 5
ÍNDICE DE TABLAS 6
RESUMEN 7
ABSTRACT 8
INTRODUCCIÓN 9
CAPÍTULO 1 11
1.1 Planteamiento del problema 11
1.2 Justificación 12
1.3 Objetivos 13
1.3.1 Objetivo General 13
1.3.2 Objetivos Específicos 13
1.4 Metodología 13
1.4.1 Metodología Descriptiva 13
1.4.2 Metodología Analítica 13
1.4.3 Metodología Experimental 14
CAPÍTULO 2 15
Fundamentación teórica 15
2.1 Tecnologías para el rastreo de vehículos 15
2.1.1 Sistema de posicionamiento básico 15
2.1.2 Sistema Sat-Nav 15
2.1.3 Sistema A-GPS 15
2.2 ESP32 16
2.3 Internet de las cosas (IoT) 16
2.3.1 Importancia del IoT para sistemas de posicionamiento 17
2.4 Servicio general de paquetes vía radio 17

2.5 Cloud 17
2.6 Firebase 18
2.7 Módulo GSM, GPRS SIM800L 18
2.8 Módulo GPS GY-NEO6MV2 19
CAPÍTULO 3.. 20
3.1 Diagrama Esquemático 20
3.2 Diagrama de bloques del dispositivo. 21
3.1.1 Bloque de alimentación. 22
3.1.2 Bloque de posicionamiento GPS 22
3.1.3 Bloque controlador 22
3.1.3 Bloque de transmisión 23
3.1.4 Bloque de la nube 23
3.1.5 Bloque de visualización de datos 23
3.3 Diagramas de Flujo 24
3.4 Aplicación para el rastreo de vehículos de carga. 26
3.5 Resultados obtenidos 29
CAPÍTULO 4.. 35
ANÁLISIS DE COSTOS 35
4.1 Costos de materiales 35
4.2 Servicios adicionales 36
4.3 Costo Fijo 36
4.4 Costo Variable 36
4.5 Punto de Equilibrio 37
CAPÍTULO 5.. 38
CONCLUSIONES 38
RECOMENDACIONES 39
REFERENCIAS BIBLIOGRÁFICAS 40
ANEXOS 43

ÍNDICE DE FIGURAS

Figura 1. ESP32 16
Figura 2. Módulo SIM800L 18
Figura 3. Módulo GPS Neo 6M 19
Figura 4. Diseño esquemático computarizado. 20
Figura 5. Estructura del Equipo. 21
Figura 6. Diagrama de bloque del dispositivo. 21
Figura 7. Diagrama de flujo para la transmisión de datos. 24
Figura 8. Diagrama de flujo para la recepción de datos 25
Figura 9. Pantalla principal de la aplicación. 27
Figura 10. Acceso directo al localizador en el mapa. 28
Figura 11. Tabla de distribución normal. 30
Figura 12. Precisión y exactitud del geolocalizador. 32
Figura 13. Prueba de campo 33

ÍNDICE DE TABLAS

Tabla 1. Muestras y distancias de referencia. 31
Tabla 2. Módulos y Materiales 35
Tabla 3. Costo total del proyecto. 36
Tabla 4. Cálculo del Punto de Equilibrio. 37

RESUMEN

Ante la creciente ola de asaltos a vehículos de carga pesada, que en los últimos años se han vuelto más constantes por lo que la vulnerabilidad a la mercadería y a la integridad de los conductores aumentan, surge el presente proyecto en el que se propone la implementación de un dispositivo geolocalizador para vehículos de carga pesada de la empresa Transpsur S.A. para lo cual se diseña un dispositivo portátil que mediante envío de datos de posicionamiento global al IoT y Cloud que es donde se van a almacenar los datos tipo string para poder conocer la ubicación del vehículo en caso de robo o sustracción. El dispositivo propuesto consta de módulos GSM y GPS controlados mediante un ESP32 que hace la función de un microcontrolador y para la creación de la aplicación móvil, se usa un software para desarrollares llamado Visual Studio con la que se podrá gestionar de manera remota la ubicación del vehículo en tiempo real y se podrá visualizar lugares de referencia con alta precisión, todo esto en vinculación con la plataforma de Firebase y conjuntamente con otras dependencias y librerías compatibles y multiplataforma. El dispositivo está diseñado para brindar seguridad, confianza y una salida para evitar posibles robos o peligros que enfrenten los conductores en la carretera.

Palabras clave: geolocalizador, IoT, Cloud, string, GSM, GPS, ESP32, Firebase, Visual Studio, dependencias, librerías, multiplataforma.

ABSTRACT

Faced with the growing wave of assaults on heavy-duty vehicles, which in recent years have become more constant, which is why the vulnerability of the merchandise and the integrity of the drivers increase, this project arises in which the implementation of a geolocator device for heavy cargo vehicles from the company Transpsur S.A. for which a portable device is designed that by sending global positioning data to the IoT and Cloud, which is where the string data will be stored in order to know the location of the vehicle in case of theft or theft. The proposed device consists of GSM and GPS modules controlled by an ESP32 that acts as a microcontroller and for the creation of the mobile application, a software for developers called Visual Studio is used with which the location of the vehicle can be managed remotely. In real time and it will be possible to visualize reference places with high precision, all this in connection with the Firebase platform and together with other compatible and cross-platform dependencies and libraries. The device is designed to provide security, confidence, and an outlet to avoid potential theft or dangers faced by drivers on the road.

Keywords: geolocator, IoT, Cloud, string, GSM, GPS, ESP32, Firebase, Visual Studio, dependencies, libraries, multiplatform.

INTRODUCCIÓN

El sistema de posicionamiento global (GPS) permite aproximar una ubicación en tiempo real de vehículos pequeños, medianos y grandes y gracias a esto se ha podido frustrar muchos robos y recuperar mercadería valuada en miles de dólares, según datos de la Fiscalía General del Estado los vehículos de carga pesada son considerados bienes de alto riesgo ya que tan solo de enero a noviembre del 2021 se produjeron 6129 robos de vehículos en todo el territorio Ecuatoriano generando pérdidas a los dueños de los vehículos y al estado.

La empresa Transpsur S.A hace algunos años vio la necesidad de implementar este tipo de tecnología para sus vehículos de carga, por la creciente demanda de robos, a nivel nacional, pero los servicios de este tipo son costosos en cuanto a la administración y a la implementación, ante esta problemática propuse la idea de implementar un GPS con equipamiento más barato, menos estético pero que realice las mismas funciones que un GPS comercial.

Se desarrolló un geolocalizador IoT y Cloud para la prevención de pérdidas vehiculares que supongan perdidas de bienes o servicios y que generen un gasto elevado a la empresa, al tratarse de una aplicación en tiempo real se puede actuar a tiempo y gestionar la información con las autoridades para la aprensión de los asaltantes, se puede conocer las rutas establecidas por la empresa y generar alertas ante posibles desvíos, todo esto gracias a la función de los dispositivos de envío de datos como el GPS, GSM y ESP32 y a la base de datos que recolecta la información enviada y la almacena en la nube de Firebase, estos datos se vinculan con la aplicación creada en Visual Studio y con dependencias de Xamarin para realizar la aplicación en un menor tiempo y de manera más fácil.

Este trabajo consta de cinco capítulos, los cuales se describen a continuación:

En el Capítulo 1: Planteamiento del problema, justificación, objetivo general y específicos y la metodología.

En el Capítulo 2: Fundamentación teórica, las características técnicas y físicas de los módulos para el desarrollo del dispositivo.

En el Capítulo 3: Se expone el diseño y la parte física y lógica del dispositivo.

En el Capítulo 4: Se analizan los costos y la aceptación del equipo.

En el Capítulo 5: Se formula la conclusión y las recomendaciones, se añaden las referencias bibliográficas y los anexos correspondientes al desarrollo del software en el lenguaje propio del IDE de arduino y Visual Studio.

CAPÍTULO 1

En este capítulo se describe el planteamiento del problema, la justificación, los objetivos y la metodología a seguir.

1.1 Planteamiento del problema

En la actualidad los vehículos de carga pesada se consideran bienes de alto riesgo ya que al contribuir con el desarrollo económico del país si sufren pérdidas de mercadería se generarán pérdidas a nivel macro y micro económico, es por eso que la necesidad de sofisticar a los vehículos con tecnologías que permitan gestionar de manera remota su localización y sus operaciones a nivel nacional, será de vital importancia a corto plazo. (Jeremy Lara, 2020)

La empresa TRANSPSUR S.A se ve forzado a la necesidad de la implementación de un dispositivo que indique la ubicación del vehículo mediante el uso de nuevas tecnologías, debido a que es primordial para la empresa salvaguardar la vida de los conductores ya que están expuestos a la delincuencia y además para cuidar la integridad del vehículo, el cual está valuado en miles de dólares, generando pérdidas considerables a la economía de la empresa si este fuese robado.

Se plantea un prototipo que sea capaz de enfrentar la problemática de la ubicación debido al gran número de robos de los vehículos de empresas de transporte en el Ecuador. Para gestionar esta problemática se trabajará en el diseño de un dispositivo electrónico que cuente con herramientas de localización y monitoreo constante para conocer su posición geográfica en tiempo real.

1.2 Justificación

El geolocalizador constituye una herramienta para la evolución de la seguridad y la integridad de las personas, donde los índices de delincuencia son altos y la seguridad es de vital importancia, en el Ecuador según datos estadísticos de la Fiscalía General del Estado de enero a noviembre del 2020 se produjeron 4142 robos de vehículos y para las mismas fechas del año 2021 estos robos aumentaron un 48% es decir se produjeron un total de 6129 robos, los cuales el 31.6% de los robos se realizan en la noche y el 54.7 % a modalidad de estruche, es decir dentro de las viviendas. (FGE, 2021)

Para precautelar dichos acontecimientos el equipo electrónico utilizado se usa para generar señales en tiempo real y permite una implementación fácil y rápida el dispositivo no ocupa mucho espacio pudiendo ocuparse en cualquier parte del vehículo y sin límite de tiempo. El dispositivo se puede implementar para todo tipo de vehículo y marcas, sin ocasionar cambios en la estructura del vehículo. El desarrollo generará confianza y tranquilidad a los dueños de los vehículos a la hora de realizar un viaje a nivel nacional e internacional.

La importancia de este proyecto radica en cubrir la necesidad de la empresa TRANSPSUR S.A la cual es poder localizar el vehículo si este fuera sustraído y evitar así pérdidas materiales y comerciales. El diseño y análisis del geolocalizador se realiza con tecnología ESP32 automatizado y mediante una red IoT y Cloud que gestionan los datos de las señales que genere el localizador portátil equipado en el vehículo de carga pesada en tiempo real y sin ningún costo adicional.

1.3 Objetivos

1.3.1 Objetivo General

- Desarrollar un dispositivo geolocalizador, IoT y Cloud, que determine la ubicación en tiempo real de un vehículo de carga pesada.

1.3.2 Objetivos Específicos

- Analizar las diferentes tecnologías de geolocalización de vehículos para la determinación de los requerimientos y las restricciones del diseño.
- Diseñar un dispositivo IOT en Cloud que pueda localizar la posición geográfica de vehículos en tiempo real para el cumplimiento de los requerimientos y restricciones analizadas.
- Implementar un dispositivo geolocalizador de vehículos para la comprobación de su correcto funcionamiento determinando la posición del vehículo y utilizando los servicios de la Cloud.
- Analizar el costo del dispositivo para la verificación de factibilidad de la implementación en vehículos de carga pesada.

1.4 Metodología

1.4.1 Metodología Descriptiva

La siguiente metodología se utilizará para establecer todas las características de los módulos, en este caso el módulo GPS y GSM estableciendo la respectiva relación entre ellos.

1.4.2 Metodología Analítica

En la metodología analítica se efectuará el análisis de cuan factible es el proyecto en base al costo-presupuesto que se generen a lo largo de la implementación.

1.4.3 Metodología Experimental

Mediante la experimentación se podrá comprobar el correcto funcionamiento del sistema, generando respuestas satisfactorias o excluyentes en tiempo real del vehículo de carga pesada.

CAPÍTULO 2

Fundamentación teórica

En este capítulo se describe principalmente los aspectos técnicos del hardware y los elementos básicos del software la cual consta de una interfaz de monitoreo y de usuario. Las tecnologías de las cuales forma parte el dispositivo son: microcontrolador ESP32 ligado al Internet de las cosas (IoT) y Cloud, Sistema de Posicionamiento Global (GPS) y Servicio General de Paquetes Vía Radio (GPRS).

2.1 Tecnologías para el rastreo de vehículos

2.1.1 Sistema de posicionamiento básico

Este tipo de sistema posee funciones muy básicas y esenciales, a la hora de reportar la ubicación en tiempo real, se trata de un dispositivo que puede reportar la ubicación según su latitud y longitud, con todos los detalles propios de un mapa como carreteras, calles, lugares de ocio, recreación, parques, avenidas, centros comerciales, etc. Una de las principales características es su bajo costo. (Jeremy Lara, 2020)

2.1.2 Sistema Sat-Nav

Este tipo de dispositivo es ideal para personas que conducen vehículos la mayor parte de su tiempo, este tipo de GPS tiene varias tareas que no solo te dirige por rutas alternas y descongestionadas, sino que tiene un asistente de voz e incluso te detalla la velocidad, el tiempo, el kilometraje. Este tipo de equipo es ideal para viajes largos pero la desventaja es que tiene un precio elevado por todas las características mencionadas. (Jeremy Lara, 2020)

2.1.3 Sistema A-GPS

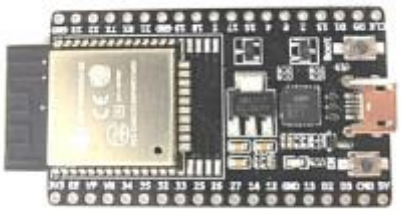
Este sistema es uno de los más vanguardistas, ya que utiliza no solo la triangulación satelital, sino que utiliza el posicionamiento mediante servidores de Google Maps, de esta manera si un usuario enciende el GPS y tiene buena señal el servidor le enviara la ubicación de la antena que posee buena señal al usuario con mala recepción y este dato seguirá estableciéndose para cualquier persona que desee establecer una conexión de GPS. (Esteban, 2019)

2.2 ESP32

Es un microcontrolador diseñado principalmente para la integración y aplicaciones basados en IoT de baja potencia. Tiene capacidades operáticas ideales para dispositivos portátiles y su principal función es la compatibilidad con el IDE de Arduino por lo que su programación y soporte son excelentes además de contar con vía WiFi, bluetooth, interruptores de antena, pines de comunicación y recepción de datos, amplificador de potencia, filtros y módulos de administración de energía, con este último se logra un consumo de energía muy bajo mediante la sincronización de reloj y los múltiples modos de operación. (Components101, 2020)

Sus principales características se pueden encontrar en el siguiente enlace: https://components101.com/microcontrollers/esp32-devkitc

Figura 1. ESP32



Autor (Components101, 2020)

En la figura 1 se observa el microcontrolador base y el cable de comunicación USB tipo mini B encargados de la configuración de los periféricos de entrada y salida.

2.3 Internet de las cosas (IoT)

El internet de las cosas no es sino una red de interconexión digital entre dispositivos, personas y la propia internet, el cual permite el intercambio de datos, permitiendo capturar información clave sobre el uso y el rendimiento de los dispositivos y los objetos para detectar patrones, hacer recomendaciones, mejorar la eficiencia y crear mejores experiencias para los usuarios, mediante un proceso conocido como M2M (machine to machine), en la cual dos máquinas interactúan entre si utilizando cualquier tipo de conectividad, sea por cable, Wifi, Bluetooth, etc. Dicha conectividad lleva información a la plataforma de la IoT que recolecta, procesa y analiza los datos. (Rodrigo Alonso, 2021)

2.3.1 Importancia del IoT para sistemas de posicionamiento

Este tipo de tecnología posibilita todo el control y conocimiento de datos de localización, control en tiempo real, genera una interacción entre máquina y humano con el fin de dar características propias y con sentido sobre el estado del vehículo, y a medida que se desarrolla genera beneficios como:

- Seguridad
- Reducción de gastos de operación
- Incremento de productividad
- Control de sensores y componentes vehiculares
- Prevención de errores.

Hablando en términos generales la inteligencia de las cosas ayuda a mejorar la eficiencia de los procesos de administración incorporados en el vehículo. (Ismael Jacobo, 2020)

2.4 Servicio general de paquetes vía radio

Conocido también como GPRS o tecnología 2.5G es un sistema de telefonía, mensajería instantánea y de correo electrónico el cual proporciona una cobertura inalámbrica completa, fue una de las primeras infraestructuras de la red móvil que permitió conectarse a la red de internet mediante los datos de un teléfono celular, entre sus servicios se tiene, servicios de mensajes multimedia, servicios de mensajes cortos, servicios punto a punto y servicios punto a multipunto, la velocidad de operación de transferencia es de 56 a 144 Kbps. (Wevar, 2005).

2.5 Cloud

Consiste en una serie de suministros de archivos o recursos a peticiones de usuario mediante la conexión de internet, es muy utilizable en un ecosistema de plataformas IoT, el cloudcomputing está especializada en soluciones de hardware y software conectados para monitorear, controlar y automatizar procesos remotamente, además permite dotar de infraestructura a las empresas para los recursos de: servidores, redes, bases de datos, archivos, etc. El principal objetivo de la nube es reducir la complejidad del procesamiento de los datos permitiendo hacer cualquier cosa más fácil, en el menor tiempo posible y para reducir costos. (Ángel Aller, 2019)

2.6 Firebase

Se trata de una plataforma móvil creada por Google y cuya función principal es la de desarrollar y facilitar la creación de espacios de aplicaciones de elevada calidad y de forma segura y rápida. La plataforma forma parte de la nube de Google y está disponible para iOS, Android y Web (Cardona, 2019). Firebase utiliza el formato JSON el cual posibilita la sincronización con varios desarrolladores de aplicaciones móviles y sin que el usuario acceda a las aplicaciones esta se mantiene actualizada, además si la conexión es nula esta se mantiene conectada a la nube hasta que se restablezca la conexión de la red y se sincronicen todos los datos, una de las ventajas es que ofrece seguridad al usuario con certificación SSL. (López, 2020)

2.7 Módulo GSM, GPRS SIM800L

Para aplicaciones inalámbricas de gran cobertura es útil la utilización del tipo de módulo GSM, GPRS que utiliza una tarjeta SIM la cual nos permite enviar y recibir servicios de banda, como llamadas, SMS e intercambio de datos a través de internet.

Este módulo se puede usar como un sistema global móvil que permite difundir información por voz, permite la interacción de aplicaciones M2M, seguimiento personal, mediciones inteligentes y otras aplicaciones relacionadas al monitoreo, todo esto mediante la comunicación serial de sus puertos, con una velocidad máxima de transmisión de 85.6 Kbps. (Naylamp mechatronics, s.f) Sus principales características se pueden encontrar en el siguiente enlace:
https://naylampmechatronics.com/inalambrico/115-modulo-gsm-sim800l

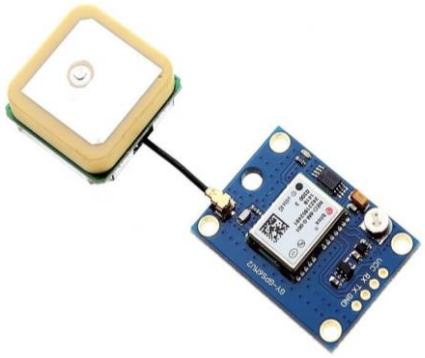
Figura 2. Módulo SIM800L

Autor: (Naylamp mechatronics, s.f)

2.8 Módulo GPS GY-NEO6MV2

Este módulo de la serie Ublox 6M está equipado con una configuración de fábrica EEPROM, posee indicadores LED y una antena de cerámica, este módulo nos permite observar datos de cualquier posición global en tiempo real. (Electronics, Naylamp, 2016). Sus principales características se pueden encontrar en el siguiente enlace: https://www.naylampmechatronics.com/blog/18_Tutorial-M%C3%B3dulo-GPS-con-Arduino.html

Figura 3. Módulo GPS Neo 6M



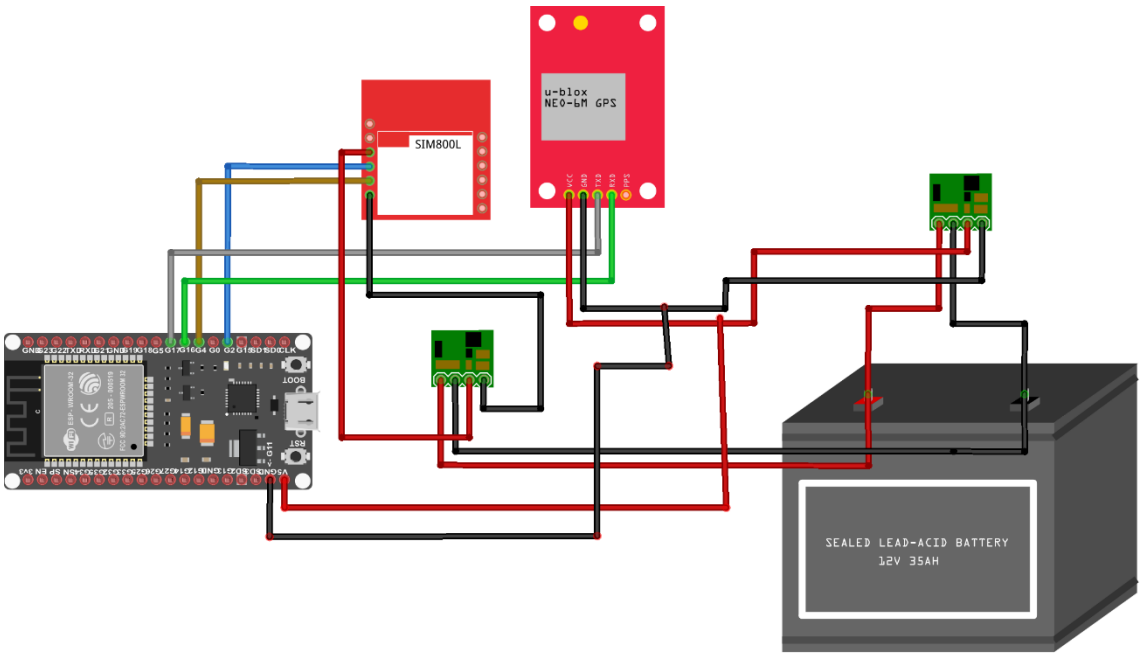
Autor: (Electronilab, s.f)

CAPÍTULO 3

En este capítulo se va a tratar el diseño del dispositivo, donde van a ejecutarse tanto la parte lógica como la parte física, además se modelan los diagramas de bloque y de flujo los cuales se considera parte fundamental del diseño, conjuntamente con las diferentes configuraciones de cada uno de los módulos donde se pone a prueba su arquitectura y por último la verificación de la señal de transmisión y recepción para validar su funcionamiento en el ambiente urbano de la ciudad.

3.1 Diagrama Esquemático

Figura 4. Diseño esquemático computarizado.



Elaborado por el autor

En la figura 4 se observa el diagrama esquemático del geolocalizador, al tratarse de un batería de 24V se debe usar reguladores de voltaje para evitar la fundición de los equipos.

Figura 5. Estructura del Equipo.

Elaborado por el autor

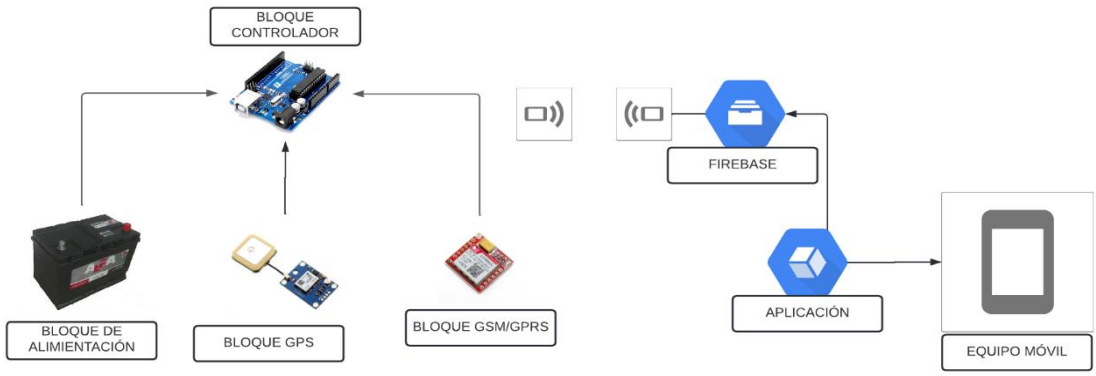
En la figura 5 se representa la estructura externa o armazón del dispositivo geolocalizador, se implementó con la finalidad de precautelar daños en las conexiones, módulos y evitar principalmente el deterioro de todo el dispositivo. Las dimensiones del armazón son:

- Ancho = 7 cm
- Alto = 13 cm

3.2 Diagrama de bloques del dispositivo.

En la figura 6 se puede observar el diagrama de bloques del dispositivo geolocalizador para vehículos de carga.

Figura 6. Diagrama de bloque del dispositivo.



Elaborado por el autor

El proyecto está conformado por bloques de envió y bloques de recolección de datos, como:

- Bloque de alimentación.
- Bloque GPS.
- Bloque del controlador.
- Bloque de transmisión.
- Bloque de la nube.
- Bloque de visualización de datos.

3.1.1 Bloque de alimentación.

Este bloque suministra la energía a los dispositivos, mediante una fuente de alimentación de 24 Voltios, se usa un regulador de voltaje para que disminuya de 24 Voltios a 5 Voltios para la alimentación de los módulos GPS y el circuido integrado ESP32 y para el caso del GSM un voltaje de 4.3 Voltios.

Las baterías de 24 Voltios son usadas en todos los camiones de carga de la empresa TRANSPSUR S.A. conformada por vehículos de gran tamaño y de uso diario y uso diario y al cubrir largos tramos, es necesario que los componentes eléctricos y mecánicos estén reforzados.

3.1.2 Bloque de posicionamiento GPS

Este bloque es el encargado de obtener datos en tiempo real de la posición del vehículo y como este se encuentra en movimiento, debe estar en constante actualización, los datos que se recolectan van a ser gestionados por el controlador ESP32 para el envió en formato de coordenadas geográficas con latitud y longitud.

3.1.3 Bloque controlador

Este bloque realiza todo el procesamiento de datos y es donde se realiza el manejo de los datos del GPS y él envió de datos mediante el GPRS, utiliza la comunicación serial y cada uno de estos módulos tienen sus propias librerías en el IDE de arduino, una vez cargado el programa en el módulo ESP32, este envía toda la información a la nube para que sea almacenada y pueda ser consultada por el cliente

3.1.3 Bloque de transmisión

Este módulo utiliza un chip Claro 4G con paquete de datos adicional para el acceso a internet, también establece un punto de acceso a la red destinada para todo el equipo con el que se puede enviar datos directamente a la nube. Este módulo se gestiona por los puertos seriales del ESP32 donde se verifica la conexión, acceso y navegación en la red.

3.1.4 Bloque de la nube

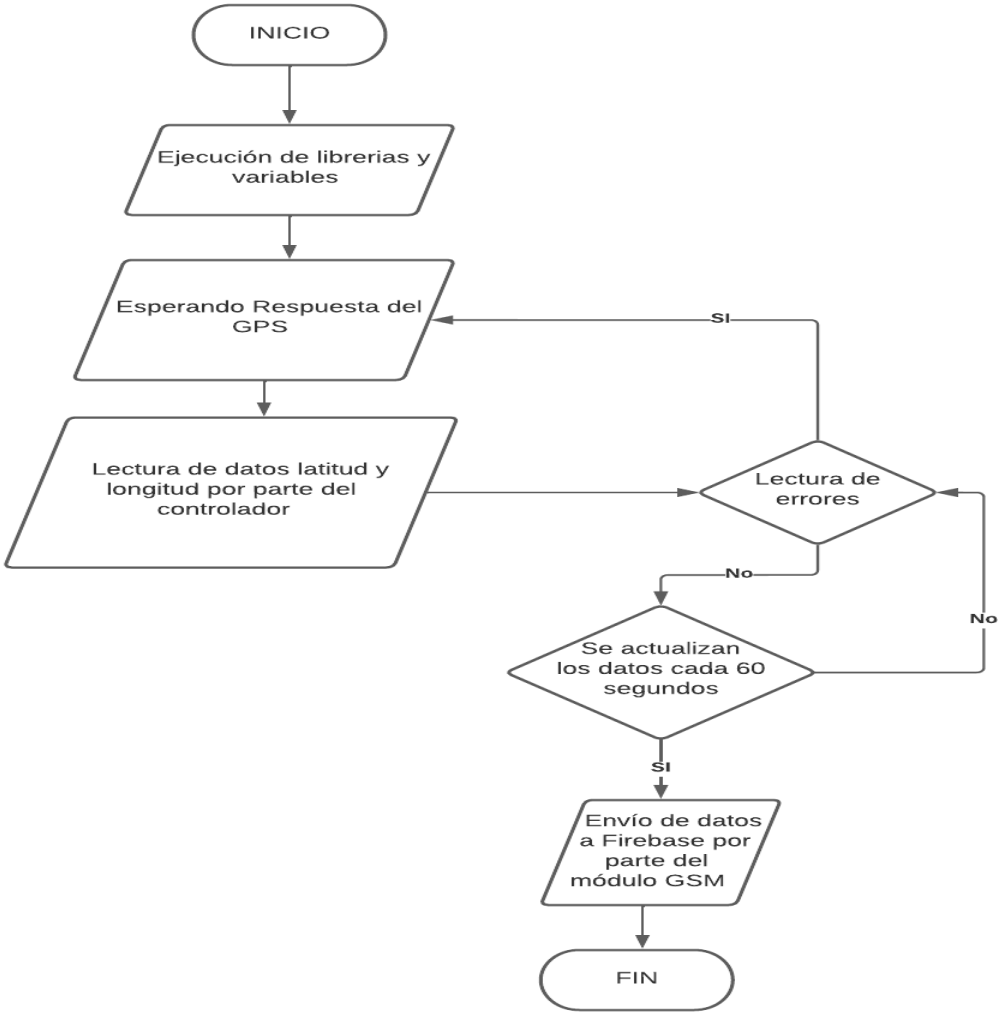
Firebase es el encargado de almacenar todos los datos enviados desde el geolocalizador, este servicio web ofrece una autentificación de usuario personalizada ya que se configura mediante el uso de una API dedicada especialmente al dueño de la cuenta y un Token con el que el usuario pueda informar al servidor que ya han sido validados los datos almacenados, todo lo antes mencionado se programa en el IDE de arduino, además Firebase transforma automáticamente el lenguaje propio de arduino a un lenguaje JSON, facilitando la lectura al cliente ya que este tipo de código es mucho más fácil de leer e interpretar.

3.1.5 Bloque de visualización de datos

Este bloque es el encargado de realizar la petición de los datos almacenados en la nube para poder visualizarlos en un entorno gráfico, se utilizó el software Visual Studio Code el cual permite crear la plantilla para la creación del entorno del geolocalizador compatible para todas las plataformas, sea iOS, Android, Windows, macOS, Linux, etc. Se basa en el lenguaje de programación de JavaScript y toda la información almacenada en la nube se visualiza en el entorno grafico creado, observando en un mapa la posición del vehículo.

3.3 Diagramas de Flujo

Figura 7. Diagrama de flujo para la transmisión de datos.



Elaborado por el autor

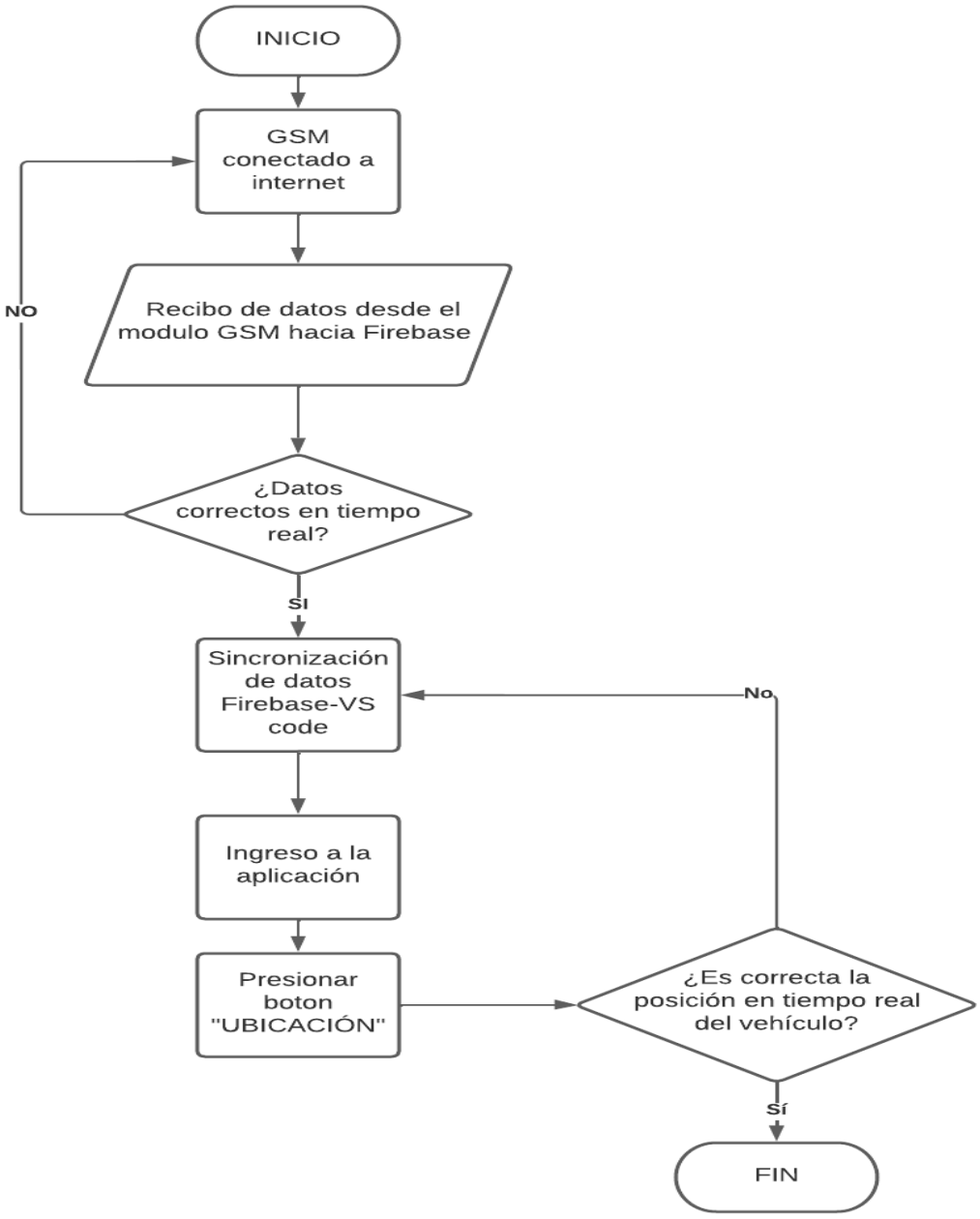
En la figura 7 se puede observar el diagrama de flujo donde se inicia ejecutando las librerías y variables del módulo GPS y GSM para que el controlador pueda ejecutar y leer comandos, además se inicializa la biblioteca serial para habilitar la comunicación digital de todos los pines de comunicación.

En el siguiente proceso y para que el dispositivo sirva, el módulo GPS debe inicializar la lectura con datos de latitud y longitud, este tipo de datos van a ser de tipo string, el módulo GPS va mandar la ubicación actual del dispositivo conectado al controlador quien

mediante programación en el entorno gráfico del IDE de arduino se visualizarán los datos enviados por el GPS, si no existe ningún error los datos se actualizarán cada 60 segundos.

El proceso de actualización de las coordenadas cada 60 segundo está programado en el controlador y es automática, el encargado del envió de los datos en tiempo real a la nube es el módulo GSM, este cual cumple la función del envió de datos a través de la conexión a la red, el envió de datos va a ser permanente y los resultados se van a ver reflejados la nube de firebase, y así sucesivamente hasta que el equipo se apague.

Figura 8. Diagrama de flujo para la recepción de datos.



Elaborado por el autor

En la figura 8 se observa el diagrama de flujo para la recepción de datos, donde el que da inicio al proceso es el módulo SIM800L, y una de las reglas para el funcionamiento del geolocalizador es que el módulo GSM siempre tenga acceso a la red y a internet, con eso se asegura la subida de datos a la nube.

Para el siguiente proceso y si todos los módulos de envió de datos están bien configurados y en correcto funcionamiento, los datos en firebase serán actualizados cada minuto según la posición real del vehículo, si todo este proceso es correcto la nube realizara una petición a la aplicación de Visual Studio para generar un entorno gráfico con la posición actual del vehículo, todo en tiempo real.

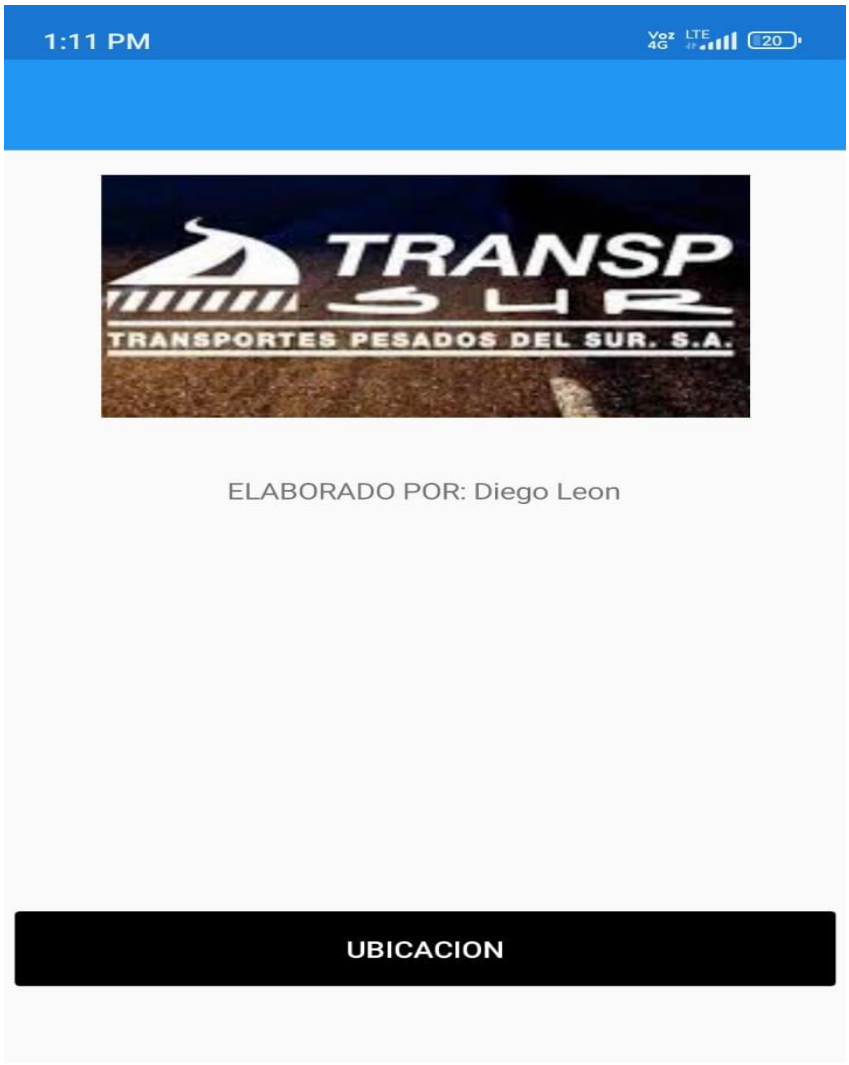
Para el ingreso a la aplicación que se realizó en un entorno de desarrolladores de aplicaciones Android se ejecutaron varias librerías disponibles, dependencias externas, repositorios globales a nivel solución todo esto compatible con firebase, se realiza el llamamiento de bibliotecas y lectura de código XML y JSON, todo esto con el fin de la creación del entorno gráfico y las funciones que tiene la aplicación. Mediante un botón denominado Ubicación dirigirá al usuario a la posición exacta del vehículo, quien comprobará que la posición sea verdadera.
Si todo este proceso es verdadero se da por concluida el rastreo y se generan nuevos datos y nuevas ubicaciones, todo esto en el periodo que el usuario desee.

3.4 Aplicación para el rastreo de vehículos de carga.

La aplicación es de acceso libre para el usuario que desee la disposición y funcionamiento del geolocalizador, la aplicación consta de dos interfaces; la primera es una pantalla principal donde se puede apreciar el logotipo de la empresa TRANSPSUR S.A, el autor del proyecto técnico y el botón de acceso a la ubicación en el mapa del vehículo en tiempo real, todo lo antes mencionado se sincroniza con las interfaces de la nube y las dependencias creadas en Visual Studio para crear el entorno gráfico y el acceso directo, como se puede observar en la figura 9.

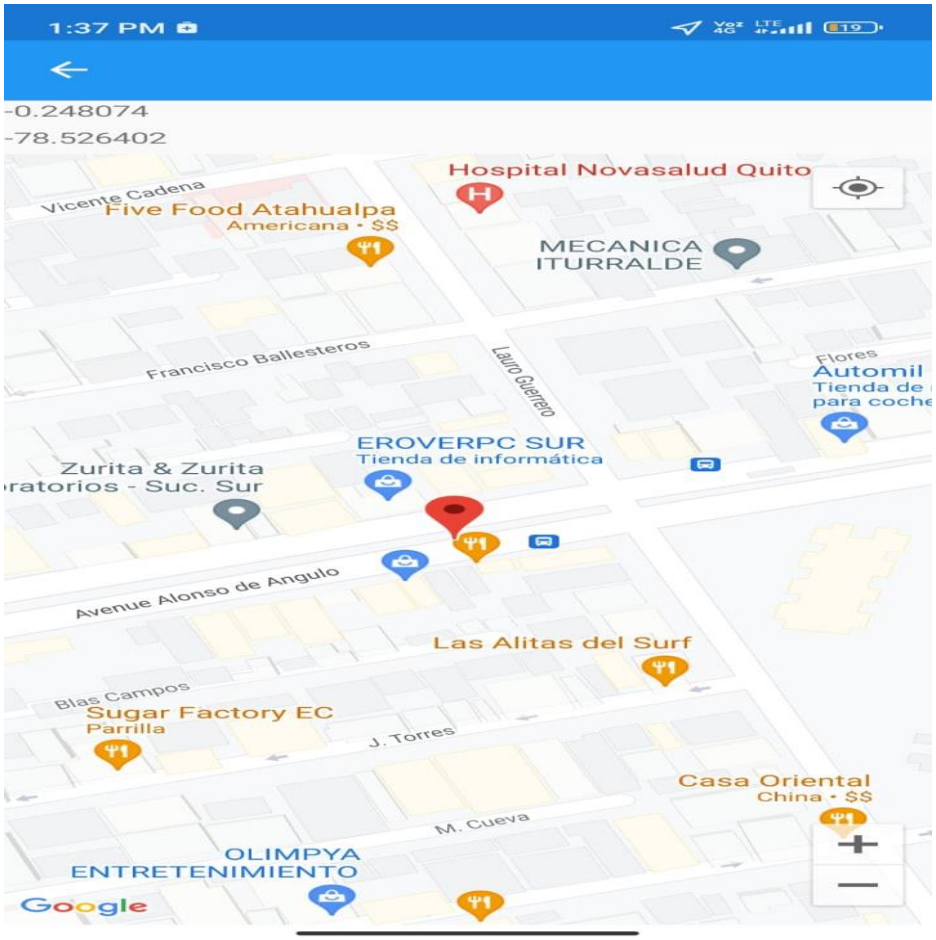
En la parte superior izquierda de la segunda interfaz se puede apreciar las coordenadas tipo string, una flecha para regresar a la pantalla principal y el mapa donde se evidencia la localización del vehículo con los datos proporcionados por la nube, además la aplicación cuenta con las funciones de Google Maps de acercar, alejar y conocer la ubicación del dispositivo móvil y la del geolocalizador al mismo tiempo tal y como se puede observar en la figura 10.

Figura 9. Pantalla principal de la aplicación.



Elaborado por el autor

Figura 10. Acceso directo al localizador en el mapa.



Elaborado por el autor

Todo lo antes mencionado es posible debido a que los módulos y el controlador envían la información mediante peticiones HTTP como se evidencia en el Anexo 1 y esta información es transformada en archivos y formatos JSON almacenados en la nube los cuales son codificados y enlazados para que el entorno desarrollador de Visual Studio mediante el lenguaje de programación de C# la cual conecta todas las dependencias habilitadas para la comunicación con Firebase como se observa en el Anexo 2, todas estas dependencias deben ser compatibles y programables con las versiones recientes de Google y de Android con el fin de evitar inconvenientes a la hora de la compilación de la aplicación en el dispositivo móvil, las dependencias disponibles para la creación de la aplicación deben permitir conectarse, ejecutar y desarrollar aplicaciones con la plataforma .NET.

Para el proceso de compilación e implementación de la aplicación en el dispositivo móvil es necesario activar la depuración por USB, esto permite al usuario poder acceder a las

aplicaciones creadas por desarrolladores en diferentes entornos de programación, es decir permite la comunicación entre el teléfono móvil y un ordenador.

Visual Studio y todas las dependencias disponibles en Windows para desarrolladores permiten crear y leer todos los datos desde el inicio de la comunicación hasta el final en un solo idioma y para cualquier plataforma.

3.5 Resultados obtenidos

Para obtener la precisión y exactitud que ofrece el dispositivo se realiza un muestreo proporcional, es decir se escogen varias muestras arrojadas por un sistema, para estimar la muestra representativa de una población u objeto sin considerar el tamaño de la población. (Bernal Torres, 2010)

$$n = \frac{z^2 * p * q}{E^2}$$

Ec. (3.1)

n= Tamaño de la muestra por estimar

z= Representa el margen de confiabilidad

p= Proporción donde la nueva posición respecto al índice de referencia (menor o igual a 5 metros).

q= Proporción donde la nueva posición respecto al eje de referencia sea mayor a 5 metros.

E= Representa el error de estimación.

Para el proceso de formulación se toma un valor de confianza del 96 %, lo que corresponde a un nivel de confianza de 2.0, con un error estimado del 5 %, se desea para el proyecto un 98 % de precisión, es decir, una distancia menor a 5 metros y el 2 % restante a una distancia mayor a los 5 metros.

Primero calculamos el margen de confiabilidad, mediante la fórmula:

$$z = 1 - \frac{\alpha}{2}$$

Ec. (3.2)

Donde:

α= 1- (porcentaje de confianza)

$$z = 1 - \frac{(1 - 0.96)}{2}$$

$$z = 0.98$$

Una vez obtenemos el valor de confianza lo verificamos en la tabla de distribución normal como se puede observar en la figura 11. (Medwave, 2011)

Figura 11. Tabla de distribución normal.

2,0	0,9772	0,9777	0,9783	0,9788	0,9793	0,9798	0,9803	0,9807	0,9812	0,9816
2,1	0,9821	0,9825	0,9829	0,9834	0,9838	0,9842	0,9846	0,9849	0,9853	0,9857
2,2	0,9860	0,9864	0,9867	0,9871	0,9874	0,9877	0,9880	0,9883	0,9886	0,9889
2,3	0,9892	0,9895	0,9898	0,9900	0,9903	0,9906	0,9908	0,9911	0,9913	0,9915
2,4	0,9918	0,9920	0,9922	0,9924	0,9926	0,9928	0,9930	0,9932	0,9934	0,9936

Elaborado por: Medwave

$$n = \frac{2.0^2 * 0.98 * 0.02}{0.05^2}$$

$$n \approx 31$$

El resultado difiere que se necesita un total de 31 muestras con un error al 5 % para determinar cuan exacto y la precisión es el geolocalizador.

Tabla 1.Muestras y distancias de referencia.

Muestra	Latitud	Longitud	Distancia al eje de referencia (metros)
1	-0,239905	-78,525378	3,2
2	-0,239911	-78,525405	4,1
3	-0,239904	-78,525393	2
4	-0,239895	-78,525403	3,3
5	-0,239757	-78,525547	2,1
6	-0,236103	-78,526885	2,6
7	-0,235728	-78,526298	2,3
8	-0,233564	-78,52462	2,2
9	-0,231583	-78,523866	3,6
10	-0,230129	-78,52329	2,2
11	-0,227919	-78,522575	2
12	-0,2254	-78,52176	2,1
13	-0,219506	-78,520722	2,9
14	-0,214424	-78,516626	2,5
15	-0,207618	-78,512768	4,2
16	-0,213465	-78,515796	3,1
17	-0,207575	-78,512775	2,2
18	-0,205509	-78,51162	2
19	-0,204761	-78,510987	3,1
20	-0,220825	-78,520604	4,2
21	-0,201004	-78,51164	3,8
22	-19,1413	-78,512832	4,6
23	-18,2309	-78,507249	3,6
24	-0,18385	-78,501744	3,2
25	-0,185804	-78,495603	2,9
26	-0,183158	-78,495114	2,7
27	-0,177429	-78,494064	3,4
28	-0,174289	-78,49294	4,2
29	-0,174453	-78,49276	4
30	-0,174577	-78,491982	4,3
31	-0,17459	-78,491991	3,6

Elaborado por el autor

Como se observa en la tabla 1 para el cálculo del error absoluto se tomó 31 muestras de la latitud y longitud con respecto al eje de referencia en metros, a continuación, se procede a calcular el rango confiable en metros para el geolocalizador.

$$E = \sqrt{\frac{\sum(x - promedio)^2}{N(N-1)}}$$

Ec. (3.3)

$$E = \sqrt{\frac{9.61}{31(31-1)}} = 0.102$$

El error de todas las medidas de localización en torno al eje de referencia está entre los 3.1032 ±0.102 metros.

Figura 12. Precisión y exactitud del geolocalizador.

Elaborado por el autor

En la figura 12 se puede verificar que el porcentaje de error con respecto al eje de referencia es similar ya que la precisión de los datos está en un rango de similitud entre el dispositivo geolocalizador marcado con una etiqueta de color rojo y el eje de referencia marcado de color azul, cabe mencionar que para ambos dispositivos GPS la señal siempre tendrá errores en cuando a la precisión.

Figura 13. Prueba de campo

Elaborado por el autor

En la figura 13 se evidencia el recorrido que realizó el vehículo de carga y cada una de las posiciones que se marcan en la aplicación con etiquetas de color rojo, se evidencia que el equipo respondió de la mejor manera durante todo el trayecto sin generar fallas a nivel de hardware ni de software y con esto se verificó la posición vehicular en todo momento.

CAPÍTULO 4

ANÁLISIS DE COSTOS

Este capítulo detalla todos los costos de los elementos que conforman el dispositivo, el estudio determina cuan factible es el equipo para los usuarios de la empresa TRANSPSUR S.A, donde se determinan los costos fijos, costos variables y los puntos de equilibrio, con el fin de conocer la rentabilidad.

4.1 Costos de materiales

Tabla 2. Módulos y Materiales

EQUIPO	CANTIDAD	VALOR UNITARIO USD	VALOR TOTAL
Chip (Claro)	1	3,5	3,5
GPS NEO 6M V2	1	11,9	11,9
GSM SIM800L	1	11,5	11,5
STEP DOWN L4005	1	3,5	3,5
Step down mini 5V	1	2	2
Resistencia 330	1	0,1	0,1
Resistencia 1k	1	0,1	0,1
Resistencia 2k	1	0,1	0,1
Diodo LED	1	0,3	0,3
Capacitor 100 uF	3	0,4	1,2
Capacitor 1uF	1	0,1	0,1
Placa PCB	1	15	15
Diseño 3D	1	15	15
Impresión 3D	1	15	15
ESP32	1	14	14
Servicio de datos	1	10	10
		TOTAL	103,3

Elaborado por: Diego León.

En la tabla 2 se detallan todos los módulos, materiales y servicios adicionales que se utilizaron con el fin de una buena ejecución del dispositivo.

4.2 Servicios adicionales

Tabla 3. Costo total del proyecto.

INVERSIÓN	VALOR (USD)
Software de la aplicación	40.00
Equipo	103.3
Software GPS	25.00
Instalación	10.00
Soporte de rastreo	6.00
Servicio en la nube	5.00
Mano de Obra	15.00
TOTAL	204.3

Elaborado por: Diego León

En la tabla 3 se establece el precio final del dispositivo geolocalizador donde se suman los costos de los servicios de internet, equipo y mano de obra y servicios adicionales por la aplicación y la programación.

4.3 Costo Fijo

Representan a todos aquellos costos que van a permanecer constantes en un determinado periodo de tiempo, todo esto sin importar el volumen de la producción del proyecto que se va a realizar. (Mónica Ordaz, 2020)

4.4 Costo Variable

Representan a todos aquellos costos que se van a modificar de acuerdo al volumen de la producción del proyecto, es decir, si no existe una visión de mayor producción no van a existir costos variables y el costo con el número de producción van a ser directamente proporcionales, es decir, si la producción es a gran escala, los costos variables serán mayores. (Mónica Ordaz, 2020)

4.5 Punto de Equilibrio

Representa el nivel de actividad donde la empresa logra cubrir toda la totalidad de sus costes, sean costos fijos o costos variables para una escala de producción mucho mayor, todo esto con el fin de obtener un beneficio cero. (Mónica Ordaz, 2020)

Con los datos de los costos de la tabla 3 se procede a obtener los costos fijos y los costos variables con el fin de determinar el punto de equilibrio, con un margen de utilidad del 30% y un precio estimado de venta al público de 265.59 USD.

Tabla 4. Cálculo del Punto de Equilibrio.

CANTIDAD	VENTAS USD	COSTOS FIJOS USD	COSTOS VARIABLES USD	COSTO TOTAL USD	UTILIDAD USD
0	0	1924,5	0	1924,5	-1924,5
2	531,18	1924,5	152	2076,5	-1545,32
4	1.062,36	1924,5	304	2228,5	-1166,14
6	1.593,54	1924,5	456	2380,5	-786,96
8	2.124,72	1924,5	608	2532,5	-407,78
9	2.390,31	1924,5	684	2608,5	-218,19
10	2.695,96	1924,5	771,5	2696,0	0
12	3.187,08	1924,5	912	2836,5	350,58
14	3.718,26	1924,5	1064	2988,5	729,76
16	4.249,44	1924,5	1216	3140,5	1108,94

Elaborado por el autor

En la tabla 4 se evidencia que para obtener un logro de beneficio cero y proceder a obtener ganancias con la distribución y producción del proyecto se deben vender 10 dispositivos de geolocalización con un total en ventas de 2.695.96 USD, llevando un margen de ganancia del 24.22 %.

CAPÍTULO 5

CONCLUSIONES

Con el fin de desarrollar un dispositivo geolocalizador, se analizaron las diferentes tecnologías y funciones de cada módulo y se concluye que el IoT es una herramienta efectiva para el proceso de localización y posicionamiento global ya que realiza toda la interacción entre máquina y servidor, además esta no solo se encarga del envió de datos, sino que también utiliza las diferentes tecnologías GPS para realizar funciones de análisis, almacenamiento, conexión y automatización de todas las tareas para que el dispositivo funcione correctamente.

Se desarrolló un dispositivo geolocalizador IoT y Cloud para determinar la posición de vehículos de carga pesada de la empresa Transpsur S.A en un mapa, con la ayuda de los módulos de gestión y envío de datos GPS, GSM y ESP32, gracias a la comunicación serial y a la importación de librerías se logró realizar el envío de datos tipo string a la nube de Firebase, y vincularla con la plataforma de desarrollo de aplicaciones de Visual Studio y a todas sus dependencias las cuales fueron compatibles entre sí, facilitando la comunicación entre servicios. Se logró la creación del diseño de la interfaz gráfica con la que el usuario puede verificar la posición en tiempo real del vehículo de carga.

En cuanto a las pruebas del dispositivo en carretera se concluye que el dispositivo es 90% aceptable en cuanto a precisión y exactitud ya que de las 31 muestras tomadas alrededor de la ciudad de Quito y con espacios de poca recepción de señal se estableció una relación entre el geolocalizador y el eje de referencia y se concluyó que tan solo el 10% muestran un error significativo de 3,1032 metros entre ambos, con un nivel de confianza del 98% lo que le hace al dispositivo rentable a la implementación.

Dentro de los análisis de costos se evidencia una solvencia y rentabilidad del 24.22% del dispositivo ya que a partir a la onceava venta se empiezan a obtener ganancias y a partir de la décima venta se recupera lo invertido, concluyendo que el dispositivo de geolocalización es factible para una empresa que disponga de un gran número de vehículos.

RECOMENDACIONES

Para un futuro se puede implementar varias funciones más en cuanto a seguridad, tecnología y accesos directos con el fin de lograr una mejor satisfacción en el usuario con mayores costos, pero mejores garantías en el servicio.

Se recomienda que el equipo utilizado no funcione con una fuente independiente ya que los módulos de envío y recepción de datos consumen mucha corriente y eso hará que el sistema de rastreo falle en cualquier momento, es por eso que se recomienda conectarlo a la fuente de alimentación primara que posee el vehículo.

REFERENCIAS BIBLIOGRÁFICAS

Jeremy Lara. (2020). *Sistema de geolocalización satelital controlado mediante la app's móvil y raspberry pi ''Geo-Spe''*. Santiago de Chile: Universidad Andres Bello, 11-13.

Wevar, J. A. (2005*). Obtenido de Análisis y Estudio de Redes GPRS.* Chile: Universidad Austral de Chile: http://cybertesis.uach.cl/tesis/uach/2005/bmfcis211a/doc/bmfcis211a.pdf

Redacción KeepCoding. (2022). *Programar aplicaciones Android en Visual Studio Cod una buena idea:* https://keepcoding.io/blog/programar-aplicaciones-android-en-visual-studio/

Components101. (19 de octubre del 2020). *ESP32* – DevKitC Pinout Configuration https://components101.com/microcontrollers/esp32-devkitc

Naylamp mechatronics SAC. (2021). *Módulo GSM SIM800L, especificaciones técnicas:* https://naylampmechatronics.com/inalambrico/115-modulo-gsm-sim800l-2g.html#:~:text=Velocidad%20m%C3%A1xima%20de%20transmisi%C3%B3n%2085.6%20Kbps

Naylamp Mechatronics. (2016). *Obtenido de Modulo GPS NEO – 6M para Arduino:* https://www.naylampmechatronics.com/blog/18_Tutorial-M%C3%B3duloGPS-con-Arduino.html

Fiscalía General del Estado. (2021). *Comisión de estadísticas de la Seguridad, Justicia, Crimen y Transparencia.* Dirección de Estadística y Sistemas de Información: https://www.fiscalia.gob.ec/estadisticas-de-robos/

Carlos pozo. (2018). *Evaluación de las tecnologías, herramientas y los protocolos con aplicaciones antirrobo para el rastreo de dispositivos móviles:* http://repositorio.puce.edu.ec/bitstream/handle/22000/15423/Tesis.pdf?sequence=1&isAllowed=y

Ismael Villavicencio Jacobo. (2020*). plataforma IoT para la localización, rastreo y monitoreo remoto de parámetros de vehículos:* https://rinacional.tecnm.mx/bitstream/TecNM/1502/1/OK%20%20Ismael%20Villavicencio%20Jacobo.pdf

Revista EIA. (2018). Sistema de geolocalización de vehículos a través de red gsm/gprs gsm/gprs y tecnología arduino: https://www.redalyc.org/journal/1492/149258931011/149258931011.pdf

Angel Aller. (2019). *Todo sobre lo que es el cloud y para qué sirve:* Profesional review: https://www.profesionalreview.com/2019/12/29/que-es-cloud-y-para-que-sirve/

Manuel Cardona (2019) *Firebase, que es, para qué sirve la plataforma de google:* IEBS https://www.iebschool.com/blog/firebase-que-es-para-que-sirve-la-plataforma-desarroladores-google-seo-sem/

Sara López (2020) *Firebase: qué, para qué sirve, funcionalidades y ventajas*: Dgital55 https://www.digital55.com/desarrollo-tecnologia/que-es-firebase funcionalidades-ventajas-conclusiones/

Medwave (2011). *Estadística Aplicada a la Investigación en Salud.* Fernando Quevedo Ricardi. Facultad de Medicina y ciencias de la salud de la Universidad de Chile: https://www.medwave.cl/medios/medwave/mayo2011/1/medwave.2011.05.5033.pdf

Bernal Torres, C.A. (2010). *Metodología de la investigación.* Bogotá D.C. Colombia: PEARSON.

Mónica Ordaz (2020). *Clasificación de los Costos.* Facultad de contaduría y administración. Universidad Veracruzana: https://www.uv.mx/personal/alsalas/files/2013/02/Clasificacion-DE-los-Costos.pdf

Esteban (2019). *Que es el A-GPS y en qué se diferencia del GPS convencionalesteb.* El Android libre:https://www.elespanol.com/elandroidelibre/tutoriales/20190414/a-gps-diferencia-gps-convencional/390961621_0.html

ANEXOS

ANEXO 1. Código de programación en el IDE de arduino

Anexo 1.1 Inicialización de librerías y variables.

```
#define TINY_GSM_MODEM_SIM800

//Increase RX buffer
#define TINY_GSM_RX_BUFFER 256

//NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
#include <TinyGPS++.h> //https://github.com/mikalhart/TinyGPSPlus
#include <TinyGsmClient.h> //https://github.com/vshymanskyy/TinyGSM
#include <ArduinoHttpClient.h> //https://github.com/arduino-libraries/ArduinoHttpClient
//NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
int count=0;
```

Anexo 1.2 Configuración de los pines y comunicaciones seriales, tanto para el GSM como para el GPS.

```
//NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
//GSM Module RX pin to ESP32 2
//GSM Module TX pin to ESP32 4
#define rxPin 4
#define txPin 2
HardwareSerial sim800(1);
TinyGsm modem(sim800);
//NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN

//NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
//GPS Module RX pin to ESP32 17
//GPS Module TX pin to ESP32 16
#define RXD2 16
#define TXD2 17
HardwareSerial neogps(2);
TinyGPSPlus gps;

unsigned long previousMillis = 0;
long interval = 10000;

void setup() {
  Serial.begin(115200);
  Serial.println("esp32 serial initialize");

  sim800.begin(9600, SERIAL_8N1, rxPin, txPin);
  Serial.println("SIM800L serial initialize");

  neogps.begin(9600, SERIAL_8N1, RXD2, TXD2);
  Serial.println("neogps serial initialize");
  delay(3000);
```

Anexo 1.3 Código de tiempo de espera para subir los datos.

```
  Serial.println("Initializing modem...");
  modem.restart();
  String modemInfo = modem.getModemInfo();
  Serial.print("Modem: ");
  Serial.println(modemInfo);
  //NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN

  // Unlock your SIM card with a PIN
  //modem.simUnlock("1234");

  http_client.setHttpResponseTimeout(90 * 1000); //TIEMPO DE ESPERA
}
```

Anexo 1.4 Código para subir los datos a Firebase, mediante la API y el Token proporcionado por la nube y con el punto de acceso a la red de Claro.

```
const char FIREBASE_HOST[]  = "arduinogps[redacted]default-rtdb.firebaseio.com";
const String FIREBASE_AUTH  = "i2NwEqQtGFF8yC2lzp[redacted]bMsbVGliPSle";

const String FIREBASE_PATH  = "/Mediciones/"+String(count);
const int SSL_PORT          = 443;

char apn[]  = "internet.claro.com.ec";
char user[] = "";
char pass[] = "";

TinyGsmClientSecure gsm_client_secure_modem(modem, 0);
HttpClient http_client = HttpClient(gsm_client_secure_modem, FIREBASE_HOST, SSL_PORT);

void loop() {

  Serial.print(F("Connecting to "));
  Serial.print(apn);
  if (!modem.gprsConnect(apn, user, pass)) {
    Serial.println(" fail");
    delay(1000);
    return;
  }
  Serial.println(" OK");
  //NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN

  http_client.connect(FIREBASE_HOST, SSL_PORT);

  //MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
  while (true) {
    if (!http_client.connected()) {
      Serial.println();
      http_client.stop();// Shutdown
      Serial.println("HTTP  not connect");
      break;
    }
    else{
      gps_loop();
    }
  }
```

```
  delay(30000);
}
//*****************************************************************************************************

//*****************************************************************************************************
void PostToFirebase(const char* method, const String & path , const String & data, HttpClient* http) {
  String response;
  int statusCode = 0;
  http->connectionKeepAlive(); // Currently, this is needed for HTTPS

  //NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
  String url;
  if (path[0] != '/') {
    url = "/";
  }
```

Anexo 1.5 Código de respuestas y conexión.

```
void PostToFirebase(const char* method, const String & path , const String & data, HttpClient* http) {
  String response;
  int statusCode = 0;
  http->connectionKeepAlive(); // Currently, this is needed for HTTPS

  //NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
  String url;
  if (path[0] != '/') {
    url = "/";
  }
  url += path + ".json";
  url += "?auth=" + FIREBASE_AUTH;
  Serial.print("POST:");
  Serial.println(url);
  Serial.print("Data:");
  Serial.println(data);
  //NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN

  String contentType = "application/json";
  http->put(url, contentType, data);
```

Anexo 1.6 Código de mensajes de error, si el código esta bien el mensaje será 200 (OK), caso contrario se mantiene en lectura de datos.

```
  //si todo esta correcto el codigo es 200 y el mensaje es OK.
  //si el codigo de estado es 3 se mantiene en espera
  //statusCode-200 (OK) | statusCode -3 (TimeOut)
  statusCode = http->responseStatusCode();
  Serial.print("Status code: ");
  Serial.println(statusCode);
  response = http->responseBody();
  Serial.print("Response: ");
  Serial.println(response);
  //NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN

  //NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
  if (!http->connected()) {
    Serial.println();
    http->stop();// cerrar
    Serial.println("HTTP POST disconnected");
  }
  //NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
}
```

Anexo 1.7 Código para el tipo de dato String, latitud, longitud.

```
if(true){
newData = false;

String latitude, longitude;
//float altitude;
//unsigned long date, time, speed, satellites;

latitude = String(gps.location.lat(), 6); // Latitude in degrees (double)
longitude = String(gps.location.lng(), 6); // Longitude in degrees (double)

//altitude = gps.altitude.meters(); // Altitude in meters (double)
//date = gps.date.value(); // Raw date in DDMMYY format (u32)
//time = gps.time.value(); // Raw time in HHMMSSCC format (u32)
//speed = gps.speed.kmph();

Serial.print("Latitude= ");
Serial.print(latitude);
Serial.print(" Longitude= ");
Serial.println(longitude);

count++;
String FIREBASE_PATH  = "/Mediciones/"+String(count);

String gpsData = "{";
gpsData += "\"lat\":" + latitude + ",";
gpsData += "\"lng\":" + longitude + "";
gpsData += "}";

  PostToFirebase("PATCH", FIREBASE_PATH, gpsData, &http_client);


  }
  //NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
}
```

Anexo 1.8 Código para subir los datos cada minuto en tiempo real.

```
void gps_loop()
{
  //NNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNNN
  //toma de muestras cada 60 segundos
  boolean newData = false;
  for (unsigned long start = millis(); millis() - start < 2000;){
    while (neogps.available()){
      if (gps.encode(neogps.read())){
        newData = true;
        break;
      }
    }
  }
```

Anexo 2. Código de programación en Visual Studio.

Anexo 2.1 Dependencias y funciones para el acceso directo a herramientas para la creación de aplicaciones móviles, contiene todo el programa para la instalación en el teléfono móvil.

```
<Project Sdk="Microsoft.NET.Sdk">

  <PropertyGroup>
    <TargetFramework>netstandard2.0</TargetFramework>
    <ProduceReferenceAssembly>true</ProduceReferenceAssembly>
  </PropertyGroup>

  <PropertyGroup Condition="'$(Configuration)|$(Platform)'=='Debug|AnyCPU'">
    <DebugType>portable</DebugType>
    <DebugSymbols>true</DebugSymbols>
  </PropertyGroup>

  <ItemGroup>
    <PackageReference Include="Xamarin.Forms" Version="5.0.0.2478" />
    <PackageReference Include="Xamarin.Essentials" Version="1.7.3" />
    <PackageReference Include="MvvmLightLibsStd10" Version="5.4.1.1" />
    <PackageReference Include="FirebaseDatabase.net" Version="4.0.7" />
    <PackageReference Include="Xamarin.Forms.Maps" Version="5.0.0.2478" />
  </ItemGroup>
  <ItemGroup>
    <None Remove="ViewModel\" />
    <None Remove="View\" />
    <None Remove="MvvmLightLibsStd10" />
    <None Remove="Model\" />
    <None Remove="FirebaseDatabase.net" />
    <None Remove="Xamarin.Forms.Maps" />
  </ItemGroup>
  <ItemGroup>
    <Folder Include="ViewModel\" />
    <Folder Include="View\" />
    <Folder Include="Model\" />
  </ItemGroup>
</Project>
```

Anexo 2.2 Archivo C# para declarar la clase de lectura y propiedades que va a tener la función coordenada.

```
using System;
namespace Proyecto.Model
{
    3 referencias
    public class Coordenadas
    {
        1 referencia
        public string Lat { get; set; }
        1 referencia
        public string Lng { get; set; }
    }
}
```

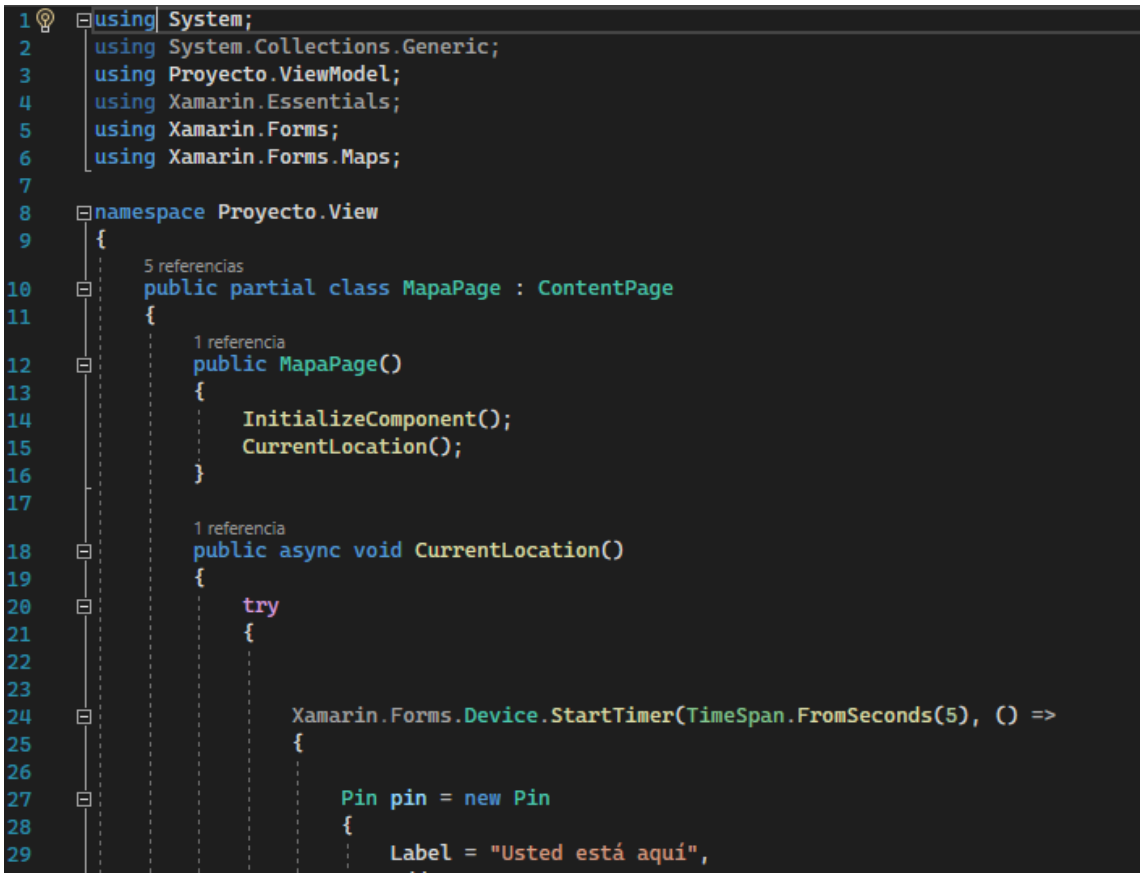
Anexo 2.3 Archivo de etiquetas XML para la creación del entorno de la aplicación, para la inserción de imágenes, letras y botones.

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<ContentPage
    xmlns="http://xamarin.com/schemas/2014/forms"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    x:Class="Proyecto.View.InicioPage"
    BindingContext="{Binding Main, Source={StaticResource Locator}}">
    <ContentPage.Content>
        <ScrollView
            BindingContext="{Binding Inicio}">
            <StackLayout
                Padding="5">
                <!--Logo habitare-->
                <Image
                    HeightRequest="150"
                    Margin="10,10,10,30"
                  Source="logo.jpeg">
                </Image>
                <Label Text="ELABORADO POR: Diego Leon"
                       VerticalOptions="Center"
                       HorizontalOptions="Center"/>
                <Button
                        BackgroundColor="Black"
                        HeightRequest="46"
                        Command="{Binding InicioCommand}"
                        Text="UBICACION"
                        TextColor="White"
                        VerticalOptions="CenterAndExpand">
                </Button>
            </StackLayout>
        </ScrollView>
    </ContentPage.Content>
</ContentPage>
```

Anexo 2.4 Archivo de etiquetas XML para la lectura de datos en la plataforma nativa de Xamarin.

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<ContentPage
    xmlns="http://xamarin.com/schemas/2014/forms"
    xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
    xmlns:maps="clr-namespace:Xamarin.Forms.Maps;assembly=Xamarin.Forms.Maps"
    x:Class="Proyecto.View.MapaPage"
    BindingContext="{Binding Main, Source={StaticResource Locator}}">
    <ContentPage.Content>
        <ScrollView
            BindingContext="{Binding Mapa}">

            <StackLayout>
                <!--
                <CollectionView ItemsSource="{Binding CoordenadasRecibidas}">
                    <CollectionView.ItemTemplate>
                        <DataTemplate>
                            <StackLayout>
                                <Grid>
                                    <Grid.ColumnDefinitions>
                                        <ColumnDefinition Width="Auto"/>
                                        <ColumnDefinition Width="Auto"/>
                                    </Grid.ColumnDefinitions>
                                    <Label
                                        Grid.Column="0"
                                    Text="{Binding Lat}"/>
                                    <Label
                                        Grid.Column="1"
                                    Text="{Binding Lng}"/>
                                </Grid>

                            </StackLayout>
                        </DataTemplate>
                    </CollectionView.ItemTemplate>
                </CollectionView>
                -->
                <Label Text="{Binding LatitudFin}" />
                <Label Text="{Binding LongitudFin}" />

                <maps:Map IsShowingUser="True"
                          x:Name="map"
                          >
                </maps:Map>

            </StackLayout>

        </ScrollView>
    </ContentPage.Content>
</ContentPage>
```

Anexo 2.5 Código para brindar colecciones genéricas que brindan seguridad a las dependencias de Xamarin Forms a la hora de la ejecución de mapas y para generar las propiedades de cada función agregada.



```
using System;
using System.Collections.Generic;
using Proyecto.ViewModel;
using Xamarin.Essentials;
using Xamarin.Forms;
using Xamarin.Forms.Maps;

namespace Proyecto.View
{
    5 referencias
    public partial class MapaPage : ContentPage
    {
        1 referencia
        public MapaPage()
        {
            InitializeComponent();
            CurrentLocation();
        }

        1 referencia
        public async void CurrentLocation()
        {
            try
            {


                Xamarin.Forms.Device.StartTimer(TimeSpan.FromSeconds(5), () =>
                {

                    Pin pin = new Pin
                    {
                        Label = "Usted está aquí",
```

```
                        Address = "",
                        Type = PinType.Place,
                        Position = new Position(MainViewModel.GetInstance().LatitudFin, MainViewModel.GetInstance().LongitudFin)
                    };
                    map.Pins.Add(pin);

                    Position position = new Xamarin.Forms.Maps.Position(MainViewModel.GetInstance().LatitudFin, MainViewModel.GetInstance().LongitudFin);

                    map.MoveToRegion(MapSpan.FromCenterAndRadius(position, Xamarin.Forms.Maps.Distance.FromKilometers(.1)));


                    return true;
                });

            }
            catch (Exception)
            {

                await DisplayAlert("Aviso", $"Active su ubicación", "Ok");
            }
        }
    }
}
```

Anexo 2.6 Código para la creación de planillas modelo-vista prediseñada en Visual Studio que interaccione con mapas.

```
using System;
using System.Windows.Input;
using GalaSoft.MvvmLight.Command;
using Proyecto.View;
using Xamarin.Forms;
namespace Proyecto.ViewModel
{
    3 referencias
    public class InicioViewModel
    {
        1 referencia
        public InicioViewModel()
        {
        }
        0 referencias
        public ICommand InicioCommand
        {
            get
            {
                return new RelayCommand(Inicio);
            }
        }
        1 referencia
        private async void Inicio()
        {
            MainViewModel.GetInstance().Mapa = new MapaViewModel();
            await Application.Current.MainPage.Navigation.PushAsync(new MapaPage());
        }
    }
}
```

Anexo 2.7 Código para la creación de propiedades del archivo *InstanceLocatos, MainViewModel.*

```
using System;
namespace Proyecto.ViewModel
{
    1 referencia
    public class InstanceLocator
    {
        1 referencia
        public MainViewModel Main
        {
            get;
            set;
        }
        0 referencias
        public InstanceLocator()
        {
            this.Main = new MainViewModel();
        }
    }
}
```

Anexo 2.8 Código para la creación de propiedades y ejecución de funciones, alertas y valores de respuesta para la lectura de datos con Firebase.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Runtime.CompilerServices;
using Xamarin.Forms;

namespace Proyecto.ViewModel
{
    13 referencias
    public class MainViewModel : INotifyPropertyChanged
    {
        private double latitudFin;
        private double longitudFin;
        3 referencias
        public double LatitudFin
        {
            get { return this.latitudFin; }
            set { SetValue(ref this.latitudFin, value); }
        }
        3 referencias
        public double LongitudFin
        {
            get { return this.longitudFin; }
            set { SetValue(ref this.longitudFin, value); }
        }
        1 referencia
        public InicioViewModel Inicio
        {
            get;
            set;
        }

        2 referencias
        public MainViewModel()
        {
            instance = this;

            this.Inicio = new InicioViewModel();
        }

        private static MainViewModel instance;
        7 referencias
        public static MainViewModel GetInstance()
        {
            if (instance == null)
            {
                return new MainViewModel();
            }
            return instance;
        }

        public event PropertyChangedEventHandler PropertyChanged;

        1 referencia
        protected void OnPropertyChanged([CallerMemberName] string propertyName = null)
        {
            PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(propertyName));
        }

        2 referencias
        protected void SetValue<T>(ref T backingField, T value, [CallerMemberName] string propertyName = null)
        {
            if (EqualityComparer<T>.Default.Equals(backingField, value))
            {
                return;
            }
            backingField = value;
            OnPropertyChanged(propertyName);
        }
    }
}
```

Anexo 2.9 Código para la ejecucion, lectura y actualizacion de los datos en tiempo real almacenados en la nube, contiene el enlace con Firebase.

```
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.ComponentModel;
using System.Runtime.CompilerServices;
using Firebase.Database;
using Firebase.Database.Query;
using Proyecto.Model;

namespace Proyecto.ViewModel
{
    3 referencias
    public class MapaViewModel : INotifyPropertyChanged
    {
        1 referencia
        public ObservableCollection<Coordenadas> CoordenadasRecibidas { get; set; } = new ObservableCollection<Coordenadas>();
        private string latitudFin;
        private string longitudFin;
        2 referencias
        public string LatitudFin
        {
            get { return this.latitudFin; }
            set { SetValue(ref this.latitudFin, value); }
        }
        2 referencias
        public string LongitudFin
        {
            get { return this.longitudFin; }
            set { SetValue(ref this.longitudFin, value); }
        }

        FirebaseClient firebaseClient = new FirebaseClient("https://arduinogps-33731-default-rtdb.firebaseio.com/");
```

```
        double conversio = Convert.ToDouble("855.65", System.Globalization.CultureInfo.InvariantCulture);
        1 referencia
        public MapaViewModel()
        {
            var collection = firebaseClient
                .Child("Mediciones")
                .AsObservable<Coordenadas>()
                .Subscribe((dbevent) =>
                {
                    if (dbevent.Object != null)
                    {
                        CoordenadasRecibidas.Add(dbevent.Object);
                        LatitudFin = dbevent.Object.Lat;
                        LongitudFin = dbevent.Object.Lng;
                        MainViewModel.GetInstance().LatitudFin = Convert.ToDouble(LatitudFin, System.Globalization.CultureInfo.InvariantCulture) ;
                        MainViewModel.GetInstance().LongitudFin = Convert.ToDouble(LongitudFin, System.Globalization.CultureInfo.InvariantCulture);


                        /*
                        var query = firebaseClient.Child("Mediciones").Child(dbevent.Key).OrderByKey().LimitToLast(1);
                        var get = query.AsObservable<Coordenadas>().Subscribe((dbeventchild) =>
                        {
                            CoordenadasRecibidas.Add(dbeventchild.Object);//get Null
                        });*/


                    }
                });

        }
```

Anexo 2.10 Código para la implementacion, control de versiones, reutilizacion, alcances de activacion y permisos de seguridad que requiere el geolocalizador. Este código es el encargado de correr la informacion de acuerdo a lo programado.

```
using Xamarin.Forms.Xaml;

[assembly: XamlCompilation(XamlCompilationOptions.Compile)]
```

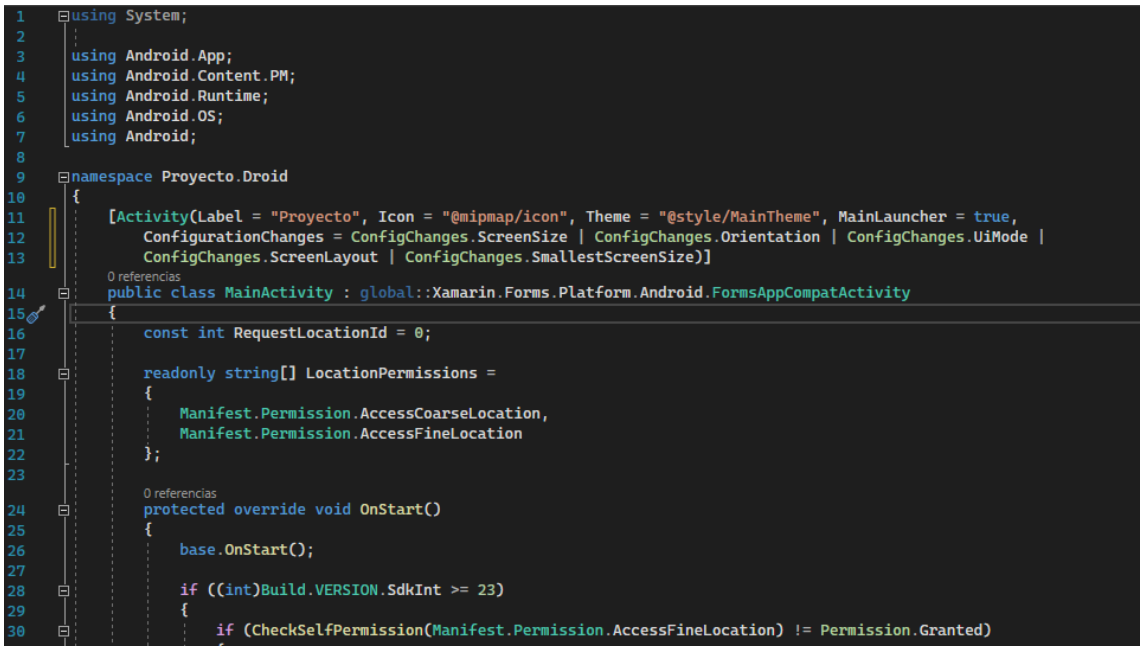
Anexo 2.11 Archivo para la creacion de etiquetas XML de plantillas vinculadas a la plataforma web de Xamarin Forms.

```xml
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="Proyecto.MainPage">

    <StackLayout>

        <Label Text="Start developing now" FontSize="Title" Padding="30,10,30,10"/>
        <Label Text="Make changes to your XAML file and save to see your UI update in the running app with XAML Hot Reload. Give it a try!"
               FontSize="16" Padding="30,0,30,0"/>
        <Label FontSize="16" Padding="30,24,30,0">
            <Label.FormattedText>
                <FormattedString>
                    <FormattedString.Spans>
                        <Span Text="Learn more at "/>
                        <Span Text="https://aka.ms/xamarin-quickstart" FontAttributes="Bold"/>
                    </FormattedString.Spans>
                </FormattedString>
            </Label.FormattedText>
        </Label>
    </StackLayout>

</ContentPage>

```

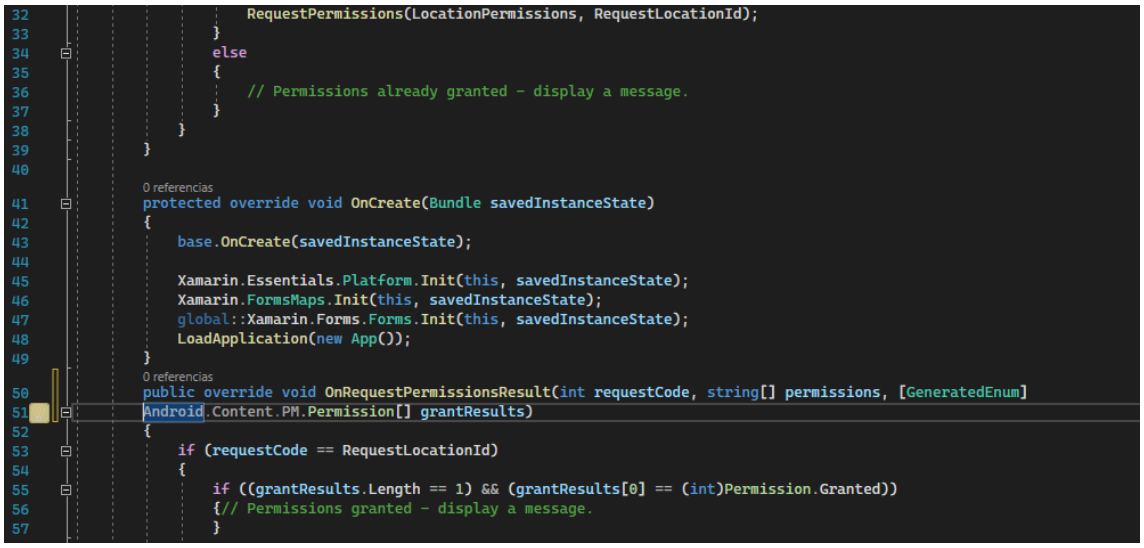
Anexo 2.12 Código para la lectura de etiquetas del archivo *MainPage* y la devolución de valores.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Xamarin.Forms;

namespace Proyecto
{
    4 referencias
    public partial class MainPage : ContentPage
    {
        0 referencias
        public MainPage()
        {
            InitializeComponent();
        }
    }
}

```

Anexo 2.13 Código para la instalación del software en cualquier plataforma móvil, con todos los permisos y compatible para todas las versiones disponibles por el SDK



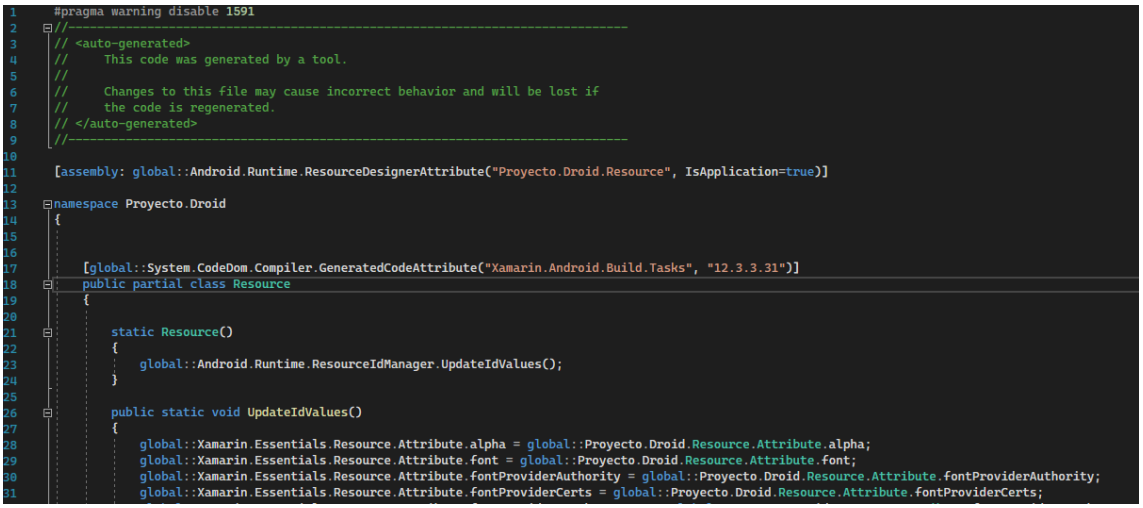
```
using System;

using Android.App;
using Android.Content.PM;
using Android.Runtime;
using Android.OS;
using Android;

namespace Proyecto.Droid
{
    [Activity(Label = "Proyecto", Icon = "@mipmap/icon", Theme = "@style/MainTheme", MainLauncher = true,
        ConfigurationChanges = ConfigChanges.ScreenSize | ConfigChanges.Orientation | ConfigChanges.UiMode |
        ConfigChanges.ScreenLayout | ConfigChanges.SmallestScreenSize)]
    0 referencias
    public class MainActivity : global::Xamarin.Forms.Platform.Android.FormsAppCompatActivity
    {
        const int RequestLocationId = 0;

        readonly string[] LocationPermissions =
        {
            Manifest.Permission.AccessCoarseLocation,
            Manifest.Permission.AccessFineLocation
        };

        0 referencias
        protected override void OnStart()
        {
            base.OnStart();

            if ((int)Build.VERSION.SdkInt >= 23)
            {
                if (CheckSelfPermission(Manifest.Permission.AccessFineLocation) != Permission.Granted)
```



```
                    RequestPermissions(LocationPermissions, RequestLocationId);
                }
                else
                {
                    // Permissions already granted - display a message.
                }
            }
        }

        0 referencias
        protected override void OnCreate(Bundle savedInstanceState)
        {
            base.OnCreate(savedInstanceState);

            Xamarin.Essentials.Platform.Init(this, savedInstanceState);
            Xamarin.FormsMaps.Init(this, savedInstanceState);
            global::Xamarin.Forms.Forms.Init(this, savedInstanceState);
            LoadApplication(new App());
        }
        0 referencias
        public override void OnRequestPermissionsResult(int requestCode, string[] permissions, [GeneratedEnum]
        Android.Content.PM.Permission[] grantResults)
        {
            if (requestCode == RequestLocationId)
            {
                if ((grantResults.Length == 1) && (grantResults[0] == (int)Permission.Granted))
                {// Permissions granted - display a message.
                }
```

```
                else
                {// Permissions denied - display a message.
                }

            }

            Xamarin.Essentials.Platform.OnRequestPermissionsResult(requestCode, permissions, grantResults);

            base.OnRequestPermissionsResult(requestCode, permissions, grantResults);
        }
    }
}
```

Anexo 2.13 Código autogenerado por la plataforma Xamarin respecto a los recursos de Android.



```
#pragma warning disable 1591
//------------------------------------------------------------------------------
// <auto-generated>
//     This code was generated by a tool.
//
//     Changes to this file may cause incorrect behavior and will be lost if
//     the code is regenerated.
// </auto-generated>
//------------------------------------------------------------------------------

[assembly: global::Android.Runtime.ResourceDesignerAttribute("Proyecto.Droid.Resource", IsApplication=true)]

namespace Proyecto.Droid
{


    [global::System.CodeDom.Compiler.GeneratedCodeAttribute("Xamarin.Android.Build.Tasks", "12.3.3.31")]
    public partial class Resource
    {

        static Resource()
        {
            global::Android.Runtime.ResourceIdManager.UpdateIdValues();
        }

        public static void UpdateIdValues()
        {
            global::Xamarin.Essentials.Resource.Attribute.alpha = global::Proyecto.Droid.Resource.Attribute.alpha;
            global::Xamarin.Essentials.Resource.Attribute.font = global::Proyecto.Droid.Resource.Attribute.font;
            global::Xamarin.Essentials.Resource.Attribute.fontProviderAuthority = global::Proyecto.Droid.Resource.Attribute.fontProviderAuthority;
            global::Xamarin.Essentials.Resource.Attribute.fontProviderCerts = global::Proyecto.Droid.Resource.Attribute.fontProviderCerts;
```