



## CARRERA DE ESPECIALIZACIÓN EN INTERNET DE LAS COSAS

MEMORIA DEL TRABAJO FINAL

### Desarrollo de un dispositivo embebido localizador y botón antipánico

**Autor:**  
**Ing. Erik Hromek**

Director:  
Mg. Ing. Lucas Dórdolo (FIUBA)

Jurados:  
Nombre del jurado 1 (pertenencia)  
Nombre del jurado 2 (pertenencia)  
Nombre del jurado 3 (pertenencia)

*Este trabajo fue realizado en la Ciudad de Quilmes,  
entre agosto de 2023 y abril de 2024.*



## *Resumen*

En la presente memoria se realiza una descripción del diseño y desarrollo de un prototipo de botón antipánico y una plataforma web básica para la recepción de alertas. Este trabajo hace énfasis en desarrollar un dispositivo que pueda ser construido con módulos de bajo costo y adquiribles en el mercado local, con foco en el consumo energético.

Para su desarrollo se aplicaron gran parte de los conocimientos adquiridos en la carrera tales como el desarrollo sobre sistemas embebidos, implementación de protocolos de comunicación y uso de tecnologías web.



## *Agradecimientos*

Esta sección es para agradecimientos personales y es totalmente **OPCIONAL**.



# Índice general

<b>Resumen</b>	<b>I</b>
<b>1. Introducción general</b>	<b>1</b>
1.1. Planteo básico . . . . .	1
1.1.1. Botones antipánico . . . . .	1
1.2. Motivación . . . . .	2
1.3. Objetivos y alcance . . . . .	3
1.4. Soluciones comerciales similares . . . . .	4
1.4.1. LK109 . . . . .	4
1.4.2. TK102B . . . . .	5
<b>2. Introducción específica</b>	<b>7</b>
2.1. Particularidades de los sistemas de botones antipánico . . . . .	7
2.1.1. Características del hardware . . . . .	7
2.1.2. Características de los sistemas web . . . . .	8
2.2. Componentes de hardware utilizados . . . . .	8
2.2.1. Microcontrolador ESP32 . . . . .	8
2.2.2. Módulo GPS NEO-6M . . . . .	9
2.2.3. Módulo GSM SIM800L . . . . .	10
2.2.4. Batería 18650 y chip cargador TP4056 . . . . .	11
2.3. Componentes de software utilizados . . . . .	11
2.3.1. Django REST framework . . . . .	11
2.3.2. Angular . . . . .	12
2.3.3. PostgreSQL . . . . .	13
2.3.4. Heroku . . . . .	13
2.4. Herramientas y servicios de software utilizados . . . . .	13
2.4.1. Docker . . . . .	13
2.4.2. Visual Studio Code . . . . .	13
2.4.3. PyGPSClient . . . . .	14
2.4.4. Moserial . . . . .	14
2.4.5. Twilio . . . . .	15
<b>3. Diseño e implementación</b>	<b>17</b>
3.1. Análisis del software . . . . .	17
<b>4. Ensayos y resultados</b>	<b>19</b>
4.1. Pruebas funcionales del hardware . . . . .	19
<b>5. Conclusiones</b>	<b>21</b>
5.1. Conclusiones generales . . . . .	21
5.2. Próximos pasos . . . . .	21
<b>Bibliografía</b>	<b>23</b>



# Índice de figuras

1.1.	Diagrama en bloques del dispositivo y el sistema web . . . . .	1
1.2.	Imagen del botón LK109 . . . . .	4
1.3.	Imagen del botón TK102B . . . . .	5
2.1.	Microcontrolador ESP32-S . . . . .	9
2.2.	Módulo GPS NEO-6M con una antena externa . . . . .	10
2.3.	Módulo SIM800L . . . . .	11
2.4.	Batería 18650 . . . . .	11
2.5.	Circuito Mp4056 . . . . .	11
2.6.	Aplicación PyGPSClient . . . . .	14
2.7.	Aplicación Moserial . . . . .	15



# Índice de tablas

1.1.	Resumen de objetivos . . . . .	3
1.2.	Tabla comparativa . . . . .	6
2.1.	Tabla comparativa . . . . .	9
2.2.	Tabla comparativa . . . . .	10
2.3.	Tabla comparativa . . . . .	12
2.4.	Tabla comparativa . . . . .	13



*Dedicado a... [OPCIONAL]*



# Capítulo 1

## Introducción general

Este capítulo presenta una introducción a los conceptos básicos de la temática del trabajo y explica las principales causas que motivaron el desarrollo del proyecto, junto con los objetivos que se buscaron alcanzar.

### 1.1. Planteo básico

Existen en la actualidad diferentes soluciones en lo que respecta a la prevención o seguridad ciudadana [1]. Estos sistemas centralizan información en tiempo real de cámaras de seguridad, móviles de las fuerzas de seguridad, entre otros. Además, suelen incorporar alguna funcionalidad para que un ciudadano, ante una situación de emergencia, pueda dar aviso a las autoridades. El mecanismo para notificar puede ser una aplicación móvil, un dispositivo geolocalizador u otra alternativa pero el objetivo es el mismo. En la figura 1.1 se pueden observar los componentes de alto nivel del sistema web y del dispositivo geolocalizador.

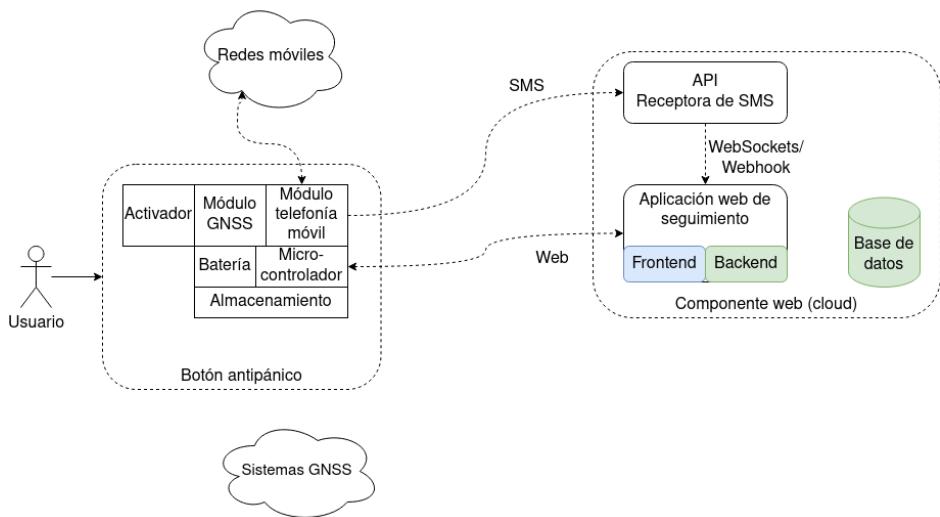


FIGURA 1.1. Diagrama en bloques del dispositivo y el sistema web

#### 1.1.1. Botones antipánico

Un botón antipánico hace referencia a un dispositivo pequeño y portátil, alimentado por batería, que permite a una persona, ante una situación de emergencia, presionar un pulsador con el objetivo de generar una alerta y que esta sea recepcionada en un sistema de monitoreo.

Estos dispositivos generalmente son brindados por las autoridades, aunque también pueden ser provistos por empresas de seguridad privadas. Sus beneficiarios suelen ser:

- Personas en situación de violencia de género.
- Personas en situación de violencia intrafamiliar.
- Establecimientos donde pueden ocurrir situaciones de violencia y es necesario dar rápidamente aviso a las autoridades.
- Agente o sereno cuyo recorrido debe ser registrado.

Entre otros.

Entre sus funcionalidades, suelen poseer:

- Conectividad a la red de telefonía móvil (*Global System for Mobile communications* o GSM).
- Acceso a datos móviles (*General Packet Radio Service* o GPRS).
- Conectividad a sistemas de geoposicionamiento satelital (*Global Navigation Satellite System* o GNSS).
- Botón de disparo de alerta.
- Reporte periódico de ubicación por mensaje de texto (SMS) o vía web.

Un detalle interesante respecto a estos dispositivos es que generalmente se comercializan en dos formatos:

- Dispositivo personal: dispositivo embebido que puede ser llevado como colgante, en la muñeca o en el bolsillo. Tiene una autonomía limitada.
- Dispositivo para móviles: también conocido como AVL o *Automatic Vehicle Location*; además de contar con antenas externas para mayor precisión, cuenta con cables para conexión a la batería de los vehículos, permitiendo tener una mayor autonomía.

En relación a este trabajo, se planteó el desarrollo del prototipo de un botón geolocalizador y una plataforma web a modo de prueba de concepto, para la recepción de alertas.

## 1.2. Motivación

El autor del trabajo posee experiencia en este tipo de soluciones, principalmente en la adquisición, configuración, implementación y soporte de botones antipánico disponibles en el mercado. Se detectó y se llegó a la conclusión, respecto a los dispositivos, de que estos presentan los siguientes inconvenientes:

- Especificaciones de hardware obsoletas.
- Documentación escasa, desactualizada o incluso incorrecta en algunos casos.
- Poca fiabilidad de dispositivo (dificultad para configurarlos, mala señal en ambientes *indoor*, desconfianza al momento de utilizarlo).
- Inconsistencias entre dispositivos idénticos.

- Poca duración de batería (desde aproximadamente 24 horas hasta 3 o 4 días como máximo, cuando lo deseable para el beneficiario final del botón es de al menos 5 o 6 días).
- Precio exageradamente elevado y muchas dificultades para adquirirlos mediante proveedores, más en períodos de alta inflación.
- Nulas características de seguridad.

Se investigaron y revisaron diferentes alternativas en el mercado, así como también se trabajó con dispositivos utilizados por sistemas de la competencia, llegando a conclusiones similares: no se encuentran dispositivos aceptables cuyo costo sea accesible.

Todas estas cuestiones nombradas anteriormente son las que motivaron la formulación del proyecto, siendo necesario disponer de un dispositivo embebido funcional y un sistema web que permita recepcionar la información del botón.

Otras motivaciones menores del trabajo, pero igualmente muy relacionadas a la especialización, fueron:

- Profundizar en el uso de frameworks para *Single Web Applications* o SPAs.
- Poder aplicar los conocimientos trabajados en relación a tecnologías *cloud* y desarrollo de APIs web.

### 1.3. Objetivos y alcance

En relación a los objetivos de alto nivel del trabajo, estos fueron:

- Investigación y determinación de un conjunto de módulos de hardware que sean adecuados para la construcción del prototipo.
- Diseño y construcción un prototipo de botón antipánico que permita validar las decisiones tecnológicas.
- Tener fiabilidad superior sobre ciertas características, en comparación con algunos de los dispositivos disponibles en el mercado.
- Desarrollo de un sistema web a modo de prueba de concepto para el uso del dispositivo.

Respecto a los objetivos por componente, estos se presentan en la tabla 1.1.

TABLA 1.1. Resumen de objetivos del trabajo

Componente	Objetivo
Prototipo	Duración de la batería
Prototipo	Calidad de señal de GSM en ambientes cerrados y abiertos
Prototipo	Calidad de señal de GNSS en ambientes cerrados y abiertos
Prototipo	Tiempo de actualización de la ubicación ( <i>time-to-first-fix</i> )
Prototipo	Envío de alertas mediante pulsador
Sistema web	Recepción de SMS de alertas
Sistema web	Alta de dispositivos
Sistema web	Visualización de alertas en tiempo real
Sistema web	Despliegue en un entorno <i>cloud</i>

No se incluyeron, dentro del alcance del trabajo, las siguientes características:

- Aplicación web productiva para gestionar las activaciones del botón antipánico. Por productiva se refiere a que pueda ser ofrecida al mercado o utilizable por usuarios encargados del monitoreo.
- Desarrollo de un dispositivo listo para reemplazar dispositivos existentes, es decir que no sea un prototipo.
- Desarrollo de un contenedor físico para el dispositivo.

Además, durante el desarrollo del trabajo, se fueron detectando diferentes cuestiones técnicas a resolver, que modificaron algunos objetivos planteados inicialmente.

## 1.4. Soluciones comerciales similares

Se presentan a continuación algunos modelos conocidos y utilizados en el mercado local. Estos dos modelos sufren los inconvenientes nombrados en las secciones anteriores: el motivo de fondo es que existen muchos clones de estos modelos, resultando muy difícil determinar si se está trabajando con un dispositivo original o un clon con posibles problemas [2].

### 1.4.1. LK109

LK109 es el nombre de un dispositivo geolocalizador que trabaja con redes GSM y GPRS, y usa *Global Positioning System* (GPS) como sistema de GNSS. Existe también una versión que trabaja con redes 3G, cuyo costo es superior. En la figura 1.2 se puede observar el dispositivo en cuestión.



FIGURA 1.2. Imagen del botón LK109

Entre sus características más interesantes se encuentran [3]:

- Indicadores de estado: batería, señal de GPS y señal de GSM.
- Posibilidad de reportar la ubicación mediante TCP o SMS.
- Reporte de ubicación en tiempo real, con una periodicidad configurable.
- Posibilidad de monitorear de forma remota el entorno del dispositivo mediante un micrófono.
- Autonomía de 240 horas o 10 días.

#### 1.4.2. TK102B

TK102B es el nombre de otro dispositivo geolocalizador que trabaja con redes GSM y GPRS, y usa *Global Positioning System* (GPS) como sistema de GNSS. En la figura 1.3 puede observarse el diseño de este modelo de dispositivo.

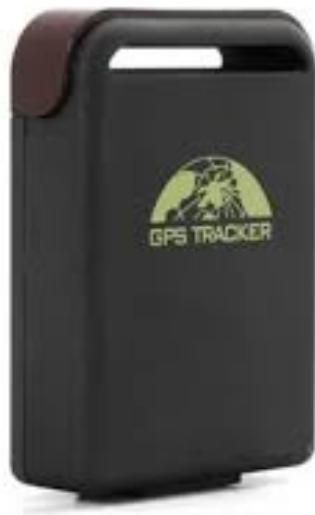


FIGURA 1.3. Imagen del botón TK102B

Entre sus características más relevantes se encuentran [4]:

- Indicadores de estado: batería, señal de GPS y señal de GSM.
- Posibilidad de reportar la ubicación mediante TCP o SMS, incluyendo la dirección aproximada, mediante geodecodificación inversa.
- Reporte de ubicación en tiempo real, con una periodicidad configurable.
- Posibilidad de monitorear de forma remota el entorno del dispositivo mediante un micrófono.
- Autonomía de 80 horas o 3 días aproximadamente.

Se presentan a continuación las principales características de estos dos modelos, contrastadas:

TABLA 1.2. Tabla comparativa de los dos modelos

Característica	LK109	TK102B
Conectividad	GSM, GPRS, SMS	GSM, GPRS, SMS
Reporte periódico	Sí	Sí
Autonomía (horas)	240	80
Indicadores de estado	Sí	Sí
Escucha remota	Sí	Sí
Protocolo de comunicación	h02	gps103

Respecto al ítem de autonomía, no se ha podido comprobar en la práctica que la duración sea la indicada (siempre es mucho menor, entre 12 y 24 horas), a excepción del uso mediante *sleep* o ahorro de batería, que no tiene utilidad para las aplicaciones de seguimiento de una persona. Esto es debido a que, en el modo de ahorro de batería, la ubicación se actualiza posterior al envío de una alerta y no antes, resultando inútil la ubicación reportada.

En relación al ítem de escucha remota, es decir la posibilidad de escuchar lo que ocurre alrededor del dispositivo, no se ha podido verificar para varios dispositivos; de todas maneras no resulta una funcionalidad útil en la mayoría de los casos de uso. Las características que se deben cumplir y validar obligatoriamente son:

- Precisión al momento de obtener la ubicación.
- Autonomía considerable de la batería.
- Rapidez para obtener señal de GSM y GNSS.

## Capítulo 2

# Introducción específica

En este capítulo se hace una presentación de cuestiones técnicas particulares del trabajo, una descripción de alto nivel de las tecnologías involucradas y las herramientas y servicios externos que formaron parte del desarrollo, así como también los componentes de hardware empleados.

### 2.1. Particularidades de los sistemas de botones antipánico

Es necesario, en primer lugar, hacer una introducción a las características que deben tener las soluciones de este tipo, junto con sus limitaciones y particularidades técnicas. Cualquier desarrollo de este tipo debe considerar las siguientes características para los módulos de hardware y software.

#### 2.1.1. Características del hardware

Los botones antipánico y dispositivos geolocalizadores suelen ser utilizados en zonas donde la cobertura de telefonía móvil es muy baja. Es decir, se cuenta con conectividad limitada para llamadas y mensajes, y prácticamente no hay cobertura para datos móviles o es de baja tasa de transferencia. La tecnología más presente es la de GPRS o 2G[5], ya que es la red de mayor alcance, pero también la de menor ancho de banda. Esto impone una restricción respecto a la conectividad, por lo que la mayoría de los dispositivos tienen las siguientes características respecto al método de comunicación:

- Comunicación vía SMS, TCP o UDP para configuración y envío de datos.
- Comunicación vía protocolos personalizados[6] para reducir el *overhead* de datos.
- Mecanismos para almacenar datos en caso de perder conectividad.

Debido a que, dentro del alcance del trabajo se apunta a configuración y envío de mensajes de alerta, se consideró innecesario, para estas funciones, requerir otro medio adicional. No se considera apropiado usar un método de comunicación que pueda no ser fiable, como lo son los datos móviles en zonas donde la conectividad es baja.

Por otra parte, otro punto importante respecto a estos dispositivos, es su capacidad de obtener señal aceptable tanto de GSM como GNSS en ambientes *indoor* y *outdoor*. Prácticamente todos los botones antipánico poseen antenas internas y sin alimentación activa, por lo que su capacidad para adquirir un nivel de señal aceptable es limitada; esto introduce varios puntos a favor y en contra:

- No generan un consumo de energía adicional.
- Su tiempo de *startup*, es decir, hasta que adquieren un nivel de señal operativo aceptable, es elevado si están en ambientes cerrados.
- Su precisión suele ser baja en ambientes cerrados, lo cual puede ser un problema en situaciones de emergencia.
- El tiempo de actualización de la posición de la persona puede ser elevado si la antena no logra captar una buena señal.

### 2.1.2. Características de los sistemas web

Un sistema web, al requerir estar en comunicación con un dispositivo y mostrar su información en tiempo real, debe contar, idealmente, con las siguientes características:

- No requerir ningún software adicional para su acceso y uso más allá de un navegador web y conexión a Internet.
- Poder recibir mensajes de texto, es decir, tener un número telefónico virtual asociado.
- Enviar actualizaciones *push* al navegador del usuario en tiempo real.

Se presentan algunos desafíos respecto al segundo y tercer punto. Para la recepción de SMS, es necesario tener integrada una central de telefonía virtual o implementar un *Short Message Service Center* o SMSC, un servicio específico para el ruteo de mensajes de texto desde y hacia la web. Respecto al tercer punto, resulta obligatoria la implementación de algún mecanismo de actualización en tiempo real desde el servidor hacia el cliente, el navegador web, y no al revés, como puede ser *WebSocket*.

En algunos casos, también debería permitir comunicación bidireccional con los dispositivos, para que un operador pueda configurarlos *on-demand*. De todas maneras, debido a que está fuera del alcance del trabajo, no se planteó el desarrollo de esta característica.

## 2.2. Componentes de hardware utilizados

### 2.2.1. Microcontrolador ESP32

Se utilizó un SoC (*System on a Chip*) de la familia ESP32 de Espressif debido a que resulta una opción ideal para el desarrollo de sistemas embebidos y aplicaciones de Internet de las Cosas. Entre las características que resultaron más atractivas para su elección, se destacan[7]:

- Bajo costo y disponibilidad en el mercado local.
- Prestaciones muy interesantes: Wi-Fi, Bluetooth, modos de ahorro de energía, varios periféricos, entre otros.
- Gran soporte de la comunidad y disponibilidad de bibliotecas, junto a su documentación.
- Ya utilizado en la carrera de especialización.

En la figura 2.1 puede observarse un ESP32-S (variante comercial utilizada).



FIGURA 2.1. Microcontrolador ESP32-S

Se presentan en la tabla 2.1 algunas alternativas comparadas. De todas maneras, esta comparación no debe ser interpretada linealmente ya que hay placas como Ai Thinker A9G y TTGO T-Call V1.3 que poseen ya módulos integrados, pero todos fueron analizados para el trabajo.

TABLA 2.1. Tabla comparativa de placas de desarrollo

Característica	ESP32	A9G	LoPy 4	TTGO T-Call
Disponibilidad	Baja	No	No	Baja
Comunidad	Muy grande	Grande	Poca	Sí
Costo	Muy bajo	Moderado	Muy elevado	Muy elevado
Curva aprendizaje	Muy baja	Alta	Baja	Baja

## 2.2.2. Módulo GPS NEO-6M

Para la implementación del módulo de posicionamiento, se decantó por el uso del módulo de GPS NEO-6M desarrollado por la compañía u-blox. Este módulo GPS posee un rango operativo de entre 2.7 V y 3.6 V, siendo muy similar al microcontrolador elegido anteriormente. Tiene modos de ahorro de energía configurables e interfaces de comunicación UART, SPI y I<sup>2</sup>C[8]. Se analizaron otras alternativas, encontrando mucha similitud, pero hallando mayor documentación y casos de uso por parte de la comunidad para el NEO-6M. En la figura 2.2 se puede observar una variante de fabricación del NEO-6M.



FIGURA 2.2. Módulo GPS NEO-6M con una antena externa

Se muestra en la tabla 2.2 una comparativa respecto a otro módulo similar.

TABLA 2.2. Tabla comparativa de placas de desarrollo

Característica	Quectel L86	NEO-6M
Disponibilidad	Moderada	Baja
Costo	Moderado	Elevado
Sistemas GNSS	GPS	GPS, GLONASS, Galileo

### 2.2.3. Módulo GSM SIM800L

Para el desarrollo de la conectividad a las redes de telefonía móvil se utilizó el módulo SIM800L, un chip fabricado por la compañía Simcom. Funciona con un rango de voltaje entre 3.4 V y 4.4 V, y permite la utilización del módulo mediante comandos AT. Estas instrucciones consisten en cadenas de caracteres cortas utilizadas dentro de la industria desde la década de 1980 hasta el día de hoy para la comunicación con módulos, módems, entre otros dispositivos[9]. Posee mecanismos para optimizar el consumo de energía pero tiene picos elevados de consumo durante la transmisión. Respecto a la compatibilidad de generaciones de tecnologías de telefonía móvil, solo soporta hasta 2G[10]. De todas maneras, para los fines del trabajo, resultó suficiente. En la figura 2.3 se puede visualizar el módulo SIM800L con una antena helicoidal, aunque pueden utilizarse otros tipos de antenas con mayor ganancia de potencia.

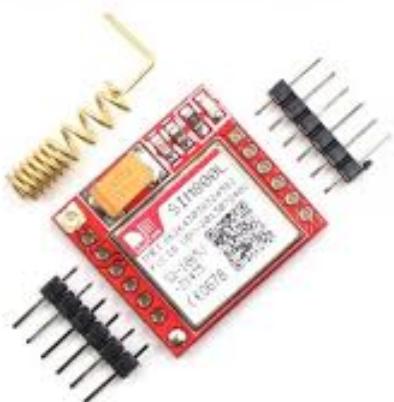


FIGURA 2.3. Módulo SIM800L

#### 2.2.4. Batería 18650 y chip cargador TP4056

Para la alimentación del microcontrolador y sus módulos, se decidió utilizar una batería recargable del tipo 18650 de Li-ion de 2600 mAh. Este tipo de baterías tiene la ventaja de trabajar con un rango de voltaje apropiado para microcontroladores y módulos, pudiendo además tener una elevada corriente máxima de descarga (hasta 2 A aproximadamente). En la figura 2.4 se puede apreciar el formato de estas, muy similar al de las pilas domésticas. Debido a que requieren un circuito integrado para la alimentación y el control ante sobrecargas, se utilizó un módulo Tp4056 que permite utilizar una entrada micro USB de 5 V, lo cual posibilita, por ejemplo, el uso de un cargador común de teléfono celular. Este circuito se puede visualizar en la figura 2.5.



FIGURA 2.4. Batería 18650



FIGURA 2.5. Circuito Mp4056

### 2.3. Componentes de software utilizados

#### 2.3.1. Django REST framework

Django REST framework es un conjunto de bibliotecas y herramientas del lenguaje Python para el desarrollo de aplicaciones web y APIs REST. Es el framework *de facto* para el desarrollo web en Python. Entre sus características más interesantes, se encuentran[11]:

- Soporte incluído para serialización de datos.
- Código modular y reusable mediante *views* y *viewsets*, que son clases de objetos para el armado de interfaces REST sobre el modelo de datos.

- Mapeo de bases de datos relacionados a objetos mediante un *Object-relational mapping* (ORM) propio.
- Documentación automática para las API por defecto.
- Estable desde hace más de una década.
- Combina la rapidez para la codificación y flexibilidad de un lenguaje de tipado dinámico junto a un conjunto de características y estructuras para escalar el desarrollo y que siga siendo mantenible.
- Soporte para múltiples mecanismos de autenticación ya incluídos.
- Bajo uso de recursos.

En la tabla 2.3 se puede ver una comparativa con uno de los frameworks web más utilizados de la actualidad[12], Express. Se eligió Django para el desarrollo de la API de la aplicación porque se deseó trabajar con un lenguaje como Python, pero con un framework fuertemente estructurado.

TABLA 2.3. Tabla comparativa de frameworks web

Característica	Django	Express
Lenguaje	Python	Javascript
Curva de aprendizaje	Moderada	Muy baja
Arquitectura	Definida	Flexible
Enfoque	Estructurado	Libre/Flexible
Performance	Muy alta	Extremadamente alta
Baterías incluídas <sup>1</sup>	Sí	No

### 2.3.2. Angular

Angular es un framework de código abierto desarrollado originalmente por Google para la construcción de aplicaciones web del tipo SPA[13]. Utiliza el lenguaje Typescript que es un *superset* de Javascript, dotándolo de tipado estático y características no presentes en Javascript. Entre sus características más potentes se pueden destacar[14]:

- Framework completo con una estructura definida.
- Arquitectura orientada en componentes.
- *Binding* de variables bidireccional.
- Disponibilidad de muchas bibliotecas y herramientas.

Para el desarrollo de las interfaces de usuario, se planteó el uso de *Angular Material*, que es una biblioteca de componentes de interfaces disponible para Angular. Su principal ventaja es la disponibilidad de componentes estándar ya armados y diseño común[15].

Para la incorporación de mapas en la aplicación, se definió el uso de Leaflet, una biblioteca para el desarrollo de mapas en aplicaciones web. Su incorporación a Angular puede hacerse mediante el paquete *ngx-leaflet*[16] junto con el proveedor gratuito y *open source* de mapas OpenStreetMap.

### 2.3.3. PostgreSQL

PostgreSQL es un *relational database management system* o RDBMS de código abierto que hace uso de SQL y es famoso por ser *ACID-compliant*, su gran amplitud de tipos de datos, robustez y extensibilidad[17]. Es utilizado ampliamente por la industria del software[18] y se encuentra disponible como una opción por los mayores proveedores de *cloud* de la actualidad. Prácticamente todos los frameworks web permiten conectarse a este motor de bases de datos.

### 2.3.4. Heroku

Heroku es un proveedor de infraestructura y servicios en la nube y *Platform as a Service* (PaaS) focalizado en el rápido despliegue y mantenimiento de las aplicaciones web. La principal ventaja de Heroku es que apunta a abstraer al usuario de detalles técnicos y la infraestructura que da soporte a los servicios usados. Se presenta en la tabla 2.4 una comparativa frente a otros proveedores, como Amazon Web Services (AWS) y Google Cloud Platform (GCP).

TABLA 2.4. Tabla comparativa de proveedores *cloud*[19][20][21]

Característica	Heroku	AWS	GCP
Complejidad	Baja	Muy alta	Media-Alta
Costo	Medio	Bajo	Bajo-Medio
Flexibilidad	Baja-Media	Muy alta	Alta
Servicios	Poca oferta	Mucha oferta	Mucha oferta
Regiones	Limitadas	Amplias	Amplias

## 2.4. Herramientas y servicios de software utilizados

### 2.4.1. Docker

Docker es una plataforma *open source* para la contenerización de aplicaciones, con el objetivo de[22]:

- Generar versiones de sistemas idénticas entre diferentes ambientes, es decir que un sistema sea portable.
- Aislar los recursos del *host* de los sistemas que corren en él.
- Reducir los tiempos de despliegue, mantenimiento y soporte de aplicaciones.

Docker permite generar imágenes de sistemas siguiendo un conjunto de instrucciones definidas mediante un archivo *Dockerfile*, habilitando generar estas mismas imágenes en otros ambientes.

### 2.4.2. Visual Studio Code

*Visual Studio Code* es un editor de texto e IDE (Integrated development environment o Entorno de desarrollo integrado) de código abierto desarrollado originalmente por Microsoft[23]. Su flexibilidad y soporte para una cantidad considerable de extensiones para gran parte de los lenguajes y frameworks de la industria lo posicionaron en la última década como uno de los IDE más utilizados.

Permite desarrollar aplicaciones con Python, Javascript/TypeScript, incorporando incluso frameworks para el desarrollo de embedidos como es el caso de ESP-IDF[24] o PlatformIO[25]. La mayoría de proveedores de servicios en la nube han desarrollado extensiones para agilizar el despliegue de aplicaciones en sus servicios.

### 2.4.3. PyGPSClient

PyGPSClient es una herramienta de escritorio desarrollada en Python para el *debugging* y análisis de varios módulos GNSS de u-blox. Su principal ventaja frente a la herramienta estándar para las pruebas de estos módulos, u-center, es que está disponible para cualquier sistema que soporte Python, en comparación con la anterior que solo se encuentra para Windows[26]. En la figura 2.6 es posible observar la aplicación en ejecución, mostrando los mensajes recibidos por el módulo GPS directamente en una terminal y la ubicación geolocalizada del módulo.

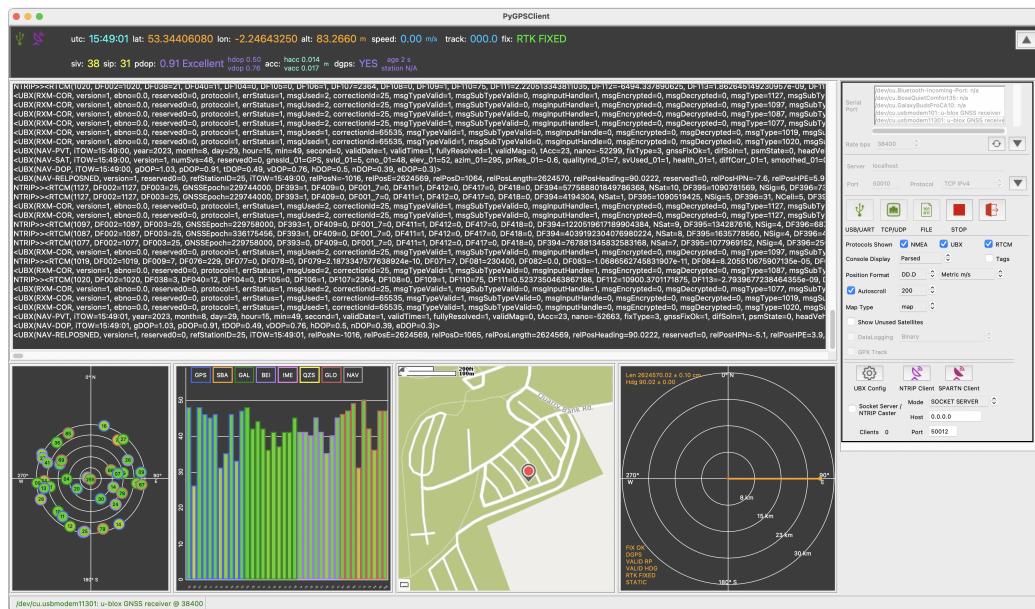


FIGURA 2.6. Aplicación PyGPSClient

### 2.4.4. Moserial

Moserial es una aplicación de escritorio orientada al *testing* de dispositivos embedidos, consolas y equipos electrónicos que permiten comunicación vía *Universal Asynchronous Receiver-Transmitter* o UART. Se encuentra disponible para el entorno gráfico GNOME, popular en sistemas GNU/Linux. En la figura 2.7 se muestra su uso, donde se pueden apreciar los datos recibidos vía comunicación serial. Resultó de importancia para la comunicación con el módulo de GSM, SIM800L.

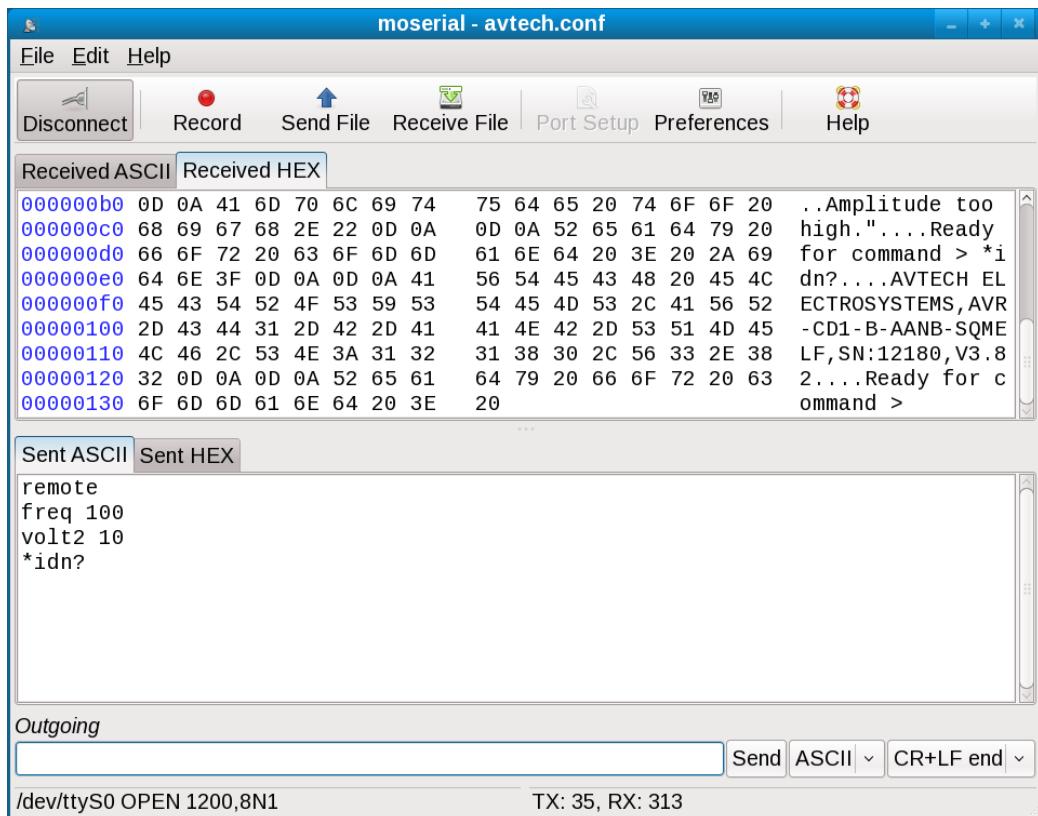


FIGURA 2.7. Aplicación Moserial

#### 2.4.5. Twilio

Twilio es un *Communications Platform as a Service* o CPaaS que ofrece diferentes servicios para el desarrollo de sistemas que requieran funcionalidades relacionadas a telefonía IP[27]. Dentro del trabajo, su principal uso fue para la adquisición de un número de teléfono virtual para la recepción de mensajes de texto provenientes de un dispositivo embebido. Permite acceder a sus servicios mediante varios mecanismos ofrecidos por la plataforma, como APIs web.



## Capítulo 3

# Diseño e implementación

### 3.1. Análisis del software

La idea de esta sección es resaltar los problemas encontrados, los criterios utilizados y la justificación de las decisiones que se hayan tomado.

Se puede agregar código o pseudocódigo dentro de un entorno lstlisting con el siguiente código:

```
\begin{lstlisting}[caption= "un epígrafe descriptivo"]
las líneas de código irían aquí...
\end{lstlisting}
```

A modo de ejemplo:

```
1 #define MAX_SENSOR_NUMBER 3
2 #define MAX_ALARM_NUMBER 6
3 #define MAX_ACTUATOR_NUMBER 6
4
5 uint32_t sensorValue[MAX_SENSOR_NUMBER];
6 FunctionalState alarmControl[MAX_ALARM_NUMBER]; //ENABLE or DISABLE
7 state_t alarmState[MAX_ALARM_NUMBER]; //ON or OFF
8 state_t actuatorState[MAX_ACTUATOR_NUMBER]; //ON or OFF
9
10 void vControl() {
11     initGlobalVariables();
12     period = 500 ms;
13
14     while(1) {
15         ticks = xTaskGetTickCount();
16         updateSensors();
17         updateAlarms();
18         controlActuators();
19
20         vTaskDelayUntil(&ticks, period);
21     }
22 }
```

CÓDIGO 3.1. Pseudocódigo del lazo principal de control.



## Capítulo 4

# Ensayos y resultados

### 4.1. Pruebas funcionales del hardware

La idea de esta sección es explicar cómo se hicieron los ensayos, qué resultados se obtuvieron y analizarlos.



## Capítulo 5

# Conclusiones

### 5.1. Conclusiones generales

La idea de esta sección es resaltar cuáles son los principales aportes del trabajo realizado y cómo se podría continuar. Debe ser especialmente breve y concisa. Es buena idea usar un listado para enumerar los logros obtenidos.

Algunas preguntas que pueden servir para completar este capítulo:

- ¿Cuál es el grado de cumplimiento de los requerimientos?
- ¿Cuán fielmente se pudo seguir la planificación original (cronograma incluido)?
- ¿Se manifestó algunos de los riesgos identificados en la planificación? ¿Fue efectivo el plan de mitigación? ¿Se debió aplicar alguna otra acción no contemplada previamente?
- Si se debieron hacer modificaciones a lo planificado ¿Cuáles fueron las causas y los efectos?
- ¿Qué técnicas resultaron útiles para el desarrollo del proyecto y cuáles no tanto?

### 5.2. Próximos pasos

Acá se indica cómo se podría continuar el trabajo más adelante.



# Bibliografía

- [1] Programa de las Naciones Unidas para el Desarrollo. *Sinopsis: Seguridad Ciudadana.*  
<https://www.undp.org/es/publications/sinopsis-seguridad-ciudadana>.  
 Abr. de 2014.
- [2] Traccar. *Chinese Clones.* <https://www.traccar.org/clones/>.
- [3] TKSTAR LK109 *USER MANUAL.*  
<https://www.manualslib.com/manual/1035746/Tkstar-Lk109.html>.
- [4] *TK102B Manual.pdf.*  
<https://www.munstergps.ie/wp-content/uploads/TK102B-Manual.pdf>.
- [5] Cobertura 3G / 4G / 5G en Buenos Aires - nPerf.com.  
<https://www.nperf.com/es/map/AR>.
- [6] Traccar. *Protocols.* <https://www.traccar.org/protocols/>.
- [7] Espressif Systems. *ESP32 Series Datasheet.* [https://www.espressif.com/sites/default/files/documentation/esp32\\_datasheet\\_en.pdf](https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf).
- [8] u-blox. *NEO-6 Data Sheet.* [https://content.u-blox.com/sites/default/files/products/documents/NEO-6\\_DataSheet\\_%28GPS.G6-HW-09005%29.pdf](https://content.u-blox.com/sites/default/files/products/documents/NEO-6_DataSheet_%28GPS.G6-HW-09005%29.pdf).
- [9] Wikipedia. *Hayes AT command set.*  
[https://en.wikipedia.org/wiki/Hayes\\_AT\\_command\\_set](https://en.wikipedia.org/wiki/Hayes_AT_command_set).
- [10] Simcom. *SIM800L Hardware Design.* [https://www.makerhero.com/img/files/download/Datasheet\\_SIM800L.pdf](https://www.makerhero.com/img/files/download/Datasheet_SIM800L.pdf).
- [11] Sherpany. *Home - Django REST Framework.*  
<https://www.djangoproject-rest-framework.org/>.
- [12] Lionel Sujay Vailshery. *Most used web frameworks among developers worldwide, as of 2023.*  
<https://www.statista.com/statistics/1124699/worldwide-developer-survey-most-used-frameworks-web/>. Jul. de 2023.
- [13] Angular. *Angular - What is Angular?*  
<https://angular.io/guide/what-is-angular>.
- [14] Angular. *Angular - Introduction to Angular concepts.*  
<https://angular.io/guide/architecture>.
- [15] Angular. *Getting Started | Angular Material.*  
<https://material.angular.io/guide/getting-started>.
- [16] BlueHalo. *bluehalo/ngx-leaflet: Core Leaflet package for Angular.io.*  
<https://material.angular.io/guide/getting-started>.
- [17] The PostgreSQL Global Development Group. *PostgreSQL: About.*  
<https://www.postgresql.org/about/>.
- [18] solid IT gmbh. *DB-Engines Ranking - popularity ranking of database management systems.* <https://db-engines.com/en/ranking>.
- [19] Romaric Philogène. *Heroku vs AWS: What to choose as a startup in 2023?*  
<https://www.qovery.com/blog/heroku-vs-aws-what-to-choose-as-a-startup>. Dic. de 2020.

- [20] SumatoSoft. *Heroku vs AWS: Which Cloud Hosting to choose in 2022?* <https://sumatosoft.medium.com/heroku-vs-aws-which-cloud-hosting-to-choose-in-2022-a9f2f1959aeb>. Dic. de 2023.
- [21] Edward Jones. *Google Cloud vs AWS (Comparing the Giants).* <https://kinsta.com/blog/google-cloud-vs-aws/>. Sep. de 2023.
- [22] Docker Inc. *Overview of the get started guide.* <https://docs.docker.com/get-started/>.
- [23] Microsoft. *Documentation for Visual Studio Code.* <https://code.visualstudio.com/docs>.
- [24] Ltd. Espressif Systems (Shanghai) Co. *Get Started - ESP32 - ESP-IDF Programming Guide v5.1.2 documentation.* <https://docs.espressif.com/projects/esp-idf/en/v5.1.2/esp32/get-started/index.html>.
- [25] PlatformIO. *PlatformIO IDE - PlatformIO latest documentation.* <https://docs.platformio.org/en/latest/integration/ide/pioide.html>.
- [26] SEMU Consulting. *Python Graphical GPS Client Application supporting NMEA, UBX, RTCM3, NTRIP & SPARTN Protocols.* <https://github.com/semuconsulting/PyGPSClient>.
- [27] Wikipedia. *Twilio.* <https://en.wikipedia.org/wiki/Twilio>.