- 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 necesitar 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

A continuación se describirán los principales componentes de hardware empleados en el desarrollo del trabajo.

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.

8 Capítulo 2. Introducción específica

- 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 necesitar 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

A continuación se describirán principales componentes de hardware empleados en el desarrollo del trabajo.

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-WROOM-32 (variante comercial utilizada).

FIGURA 2.1. Placa de desarrollo ESP32-WROOM-32.

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 módulos integrados, pero todos fueron analizados para el trabajo.

TABLA 2.1. 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 I²C [8]. Se analizaron otras alternativas y se encontraron muchas similitudes, sin embargo se halló mayor cantidad de 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.

2.2. Componentes de hardware utilizados 9

- 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-WROOM-32 (variante comercial utilizada).

FIGURA 2.1. Placa de desarrollo ESP32-WROOM-32.

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 módulos integrados, pero todos fueron analizados para el trabajo.

TABLA 2.1. 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 I²C [8]. Se analizaron otras alternativas y se encontraron muchas similitudes, sin embargo se halló 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.

```
REST_FRAMEWORK = {
    "DEFAULT_SCHEMA_CLASS": "rest_framework.schemas.coreapi.AutoSchema",
    "DEFAULT_AUTHENTICATION_CLASSES": [
        "rest_framework_simplejwt.authentication.JWTAuthentication",
    ],
    "DEFAULT_PERMISSION_CLASSES": [
        "rest_framework.permissions.IsAuthenticated",
    ],
}
```

CÓDIGO 3.4. Configuración de *middlewares* para la autenticación de la API.

3.4. Desarrollo de módulo de *frontend*

El desarrollo de la aplicación web en *Angular* se estructuró siguiendo la convención estándar del framework [46], como se puede apreciar en la figura 3.11.

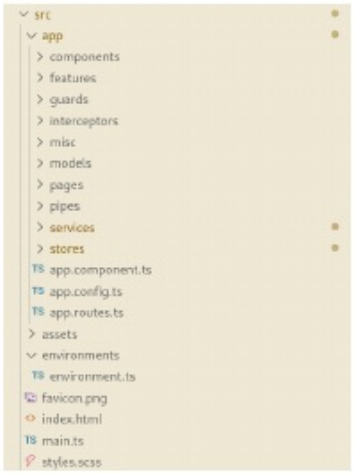

FIGURA 3.11. Estructura del *frontend*.

Respecto al uso de cada carpeta, se pueden describir las más relevantes de la aplicación:

- *pages*: corresponde a componentes que representan vistas enteras o secciones de la aplicación web.
- *components*: corresponde a componentes invocados dentro de las páginas, como son los *modals* o ventanas de carga de datos, el *layout* o estructura base de la interfaz y otros componentes.
- *stores*: hace referencia a clases que encapsulan datos y funcionan como una fuente de datos única para toda la aplicación [47].

30 Capítulo 3. Diseño e implementación

REST_FRAMEWORK = {
    "DEFAULT_SCHEMA_CLASS": "rest_framework.schemas.coreapi.AutoSchema",
    "DEFAULT_AUTHENTICATION_CLASSES": [
        "rest_framework_simplejwt.authentication.JWTAuthentication",
    ],
    "DEFAULT_PERMISSION_CLASSES": [
        "rest_framework.permissions.IsAuthenticated",
    ],
}

CÓDIGO 3.4. Configuración de *middlewares* para la autenticación de la API.

3.4. Desarrollo de módulo de *frontend*

El desarrollo de la aplicación web en *Angular* se estructuró siguiendo la convención estándar del framework [46], la cual se puede apreciar en la figura 3.11.

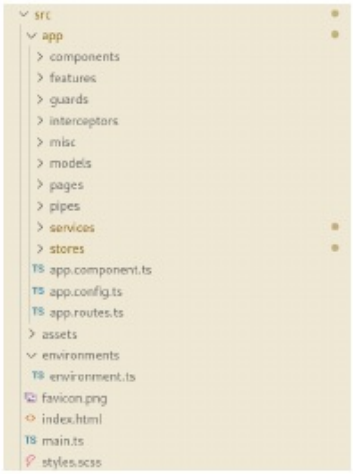

FIGURA 3.11. Estructura del *frontend*.

Respecto al uso de cada carpeta, se pueden describir las más relevantes de la aplicación:

- *pages*: corresponde a componentes que representan vistas enteras o secciones de la aplicación web.
- *components*: corresponde a componentes invocados dentro de las páginas, como son los *modals* o ventanas de carga de datos, el *layout* o estructura base de la interfaz y otros componentes.
- *stores*: hace referencia a clases que encapsulan datos y funcionan como una fuente de datos única para toda la aplicación [47].

Capítulo 5

Conclusiones

En este capítulo se presentan las conclusiones del trabajo realizado, así como también propuestas de continuidad para el futuro.

5.1. Conclusiones y resultados obtenidos

En este trabajo se ha culminado el desarrollo y las pruebas funcionales de un prototipo de botón antipánico y un sistema web para poder recibir y gestionar las alertas recibidas.

En relación a los objetivos planteados se considera que el único objetivo que no se alcanzó fue el de poder reducir drásticamente el consumo energético de tal forma que el dispositivo pueda durar varios días, y el hecho de querer plantear el desarrollo embebido como un proyecto que sea rápidamente extensible, como una biblioteca. Por otra parte, los demás objetivos sí se pudieron cumplir, y se considera importante destacarlo, para evidenciar el valor y aporte del desarrollo del trabajo. Debido a esto, se orientó la etapa final del desarrollo a poder completar de una forma más abarcativa los demás componentes involucrados para que puedan aprovecharse para proyectos futuros.

En base a las conclusiones y resultados generales, se puede analizar el grado de cumplimiento de todos los requerimientos en el anexo A. Se observa que la mayor cantidad de inconvenientes estuvo relacionada con las dificultades para cumplir con la optimización del consumo energético.

En relación a los objetivos no cumplidos, se debe al hecho de haber subestimado el trabajo a realizar sobre el sistema embebido y la poca experiencia en el desarrollo de sistemas embebidos de este tipo. Otro factor que influyó fue la dificultad para realizar pruebas exhaustivas sobre los componentes de hardware.

Se considera que la planificación original fue acorde al tiempo requerido, a excepción del módulo embebido cuya complejidad e inconvenientes detectados fueron considerables. Respecto a la organización del tiempo, se presentaron imprevistos laborales que obligaron a posponer la finalización del desarrollo.

A pesar de los desafíos atravesados y teniendo en cuenta tanto los entregables como los resultados, se considera que el resultado del trabajo fue muy positivo, destacando el aprendizaje, la experiencia adquirida y el desarrollo de un prototipo con la posibilidad de continuar el trabajo.

49

Capítulo 5

Conclusiones

En este capítulo se presentan las conclusiones del trabajo realizado, así como también propuestas de continuidad para el futuro.

5.1. Conclusiones y resultados obtenidos

En este trabajo se ha culminado el desarrollo y las pruebas funcionales de un prototipo de botón antipánico y un sistema web para poder recibir y gestionar las alertas recibidas.

En relación a los objetivos planteados se considera que el único objetivo que no se alcanzó fue el de poder reducir drásticamente el consumo energético de tal forma que el dispositivo pueda durar varios días, y el hecho de querer plantear el desarrollo embebido como un proyecto que sea rápidamente extensible, como una biblioteca. Por otra parte, los demás objetivos sí se pudieron cumplir, y se considera importante destacarlo, para evidenciar el valor y aporte del desarrollo del trabajo. Debido a esto, se orientó la etapa final del desarrollo a poder completar de una forma más abarcativa los demás componentes involucrados para que puedan aprovecharse para proyectos futuros.

En base a las conclusiones y resultados generales, se puede analizar el grado de cumplimiento de todos los requerimientos en el anexo A. Se observa que la mayor cantidad de inconvenientes estuvo relacionada con las dificultades para cumplir con la optimización del consumo energético.

En relación a los objetivos no cumplidos, se debe a haber subestimado el trabajo a realizar sobre el sistema embebido y la poca experiencia en el desarrollo de sistemas embebidos de este tipo. Otro factor que influyó fue la dificultad para realizar pruebas exhaustivas sobre los componentes de hardware.

Se considera que la planificación original fue acorde al tiempo requerido, a excepción del módulo embebido cuya complejidad e inconvenientes detectados fueron considerables. Respecto a la organización del tiempo, se presentaron imprevistos laborales que obligaron a posponer la finalización del desarrollo.

A pesar de los desafíos atravesados y teniendo en cuenta tanto los entregables como los resultados, se considera que el resultado del trabajo fue muy positivo, destacando el aprendizaje, la experiencia adquirida y el desarrollo de un prototipo con la posibilidad de continuar el proyecto.

5.2. Trabajo futuro

En relación al trabajo futuro, se pueden tener en cuenta dos aristas para aprovechar el trabajo realizado y extenderlo hacia el futuro. A continuación, se detallan los principales puntos que se pueden desarrollar y trabajar hacia adelante.

- Tomar como referencia todo lo aplicado para el desarrollo con el framework ESP-IDF, y lo que se investigó pero no se pudo aplicar correctamente e implementarlo para otras placas similares, que tengan algunas características de hardware resueltas o ya incorporadas. Este el caso de la placa de desarrollo *LilyGo SIM7000G*, que integra módulos de telefonía móvil, GPS, batería, etc. en un mismo kit [66]. Además, incluye la posibilidad de utilizar redes de bajo consumo como NB-IoT [67] ya que posee un módulo GSM más moderno que el utilizado en el trabajo.
- Convertir el *Minimun Viable Product* o MVP del sistema web en un producto final o incorporar las funcionalidades consideradas más útiles en un producto *legacy* del empleo actual del autor con características muy similares, como es la parte de *WebSockets*, API REST e integración con números telefónicos virtuales.
- Extender la funcionalidad del *frontend* para contemplar múltiples usuarios de una misma organización o ambiente, teniendo ya la implementación en la API.
- Incorporar una estructura del *layout* de la aplicación web que sea adaptable a dispositivos móviles.
- Incorporar la posibilidad de recibir alertas no solamente mediante SMS, sino también mediante otras aplicaciones como *Whatsapp*, cuya integración puede hacerse vía APIs [68].
- Agregar de funcionalidades de seguimiento continuo en el sistema embebido para vehículos, sabiendo que hay trabajos realizados sobre el tema, pero considerando incorporar alimentación de energía continua.

50 Capítulo 5. Conclusiones

5.2. Trabajo futuro

En relación al trabajo futuro, se pueden tener en cuenta dos aristas para aprovechar el trabajo realizado y extenderlo hacia el futuro. A continuación, se detallan los principales puntos que se pueden desarrollar y trabajar para el proyecto hacia adelante.

- Tomar como referencia todo lo aplicado para el desarrollo con el framework ESP-IDF, y lo que se investigó pero no se pudo aplicar correctamente e implementarlo para otras placas similares, que tengan algunas características de hardware resueltas o ya incorporadas. Este el caso de la placa de desarrollo *LilyGo SIM7000G*, que integra módulos de telefonía móvil, GPS, batería, etc. en un mismo kit [66], incluyendo además la posibilidad de utilizar redes de bajo consumo como NB-IoT [67] ya que posee un módulo GSM más moderno que el utilizado en el trabajo.
- Convertir el *Minimun Viable Product* o MVP del sistema web en un producto final o incorporar las funcionalidades consideradas más útiles en un producto *legacy* del empleo actual del autor con características muy similares, como es la parte de *WebSockets*, API REST e integración con números telefónicos virtuales.
- Extender la funcionalidad del *frontend* para contemplar múltiples usuarios de una misma organización o ambiente, teniendo ya la implementación en la API.
- Incorporar una estructura del *layout* de la aplicación web que sea adaptable a dispositivos móviles.
- Incorporar la posibilidad de recibir alertas no solamente mediante SMS, sino también mediante otras aplicaciones como *Whatsapp*, cuya integración puede hacerse vía APIs [68].
- Agregar de funcionalidades de seguimiento continuo en el sistema embebido para vehículos, sabiendo que hay trabajos realizados sobre el tema, pero considerando incorporar alimentación de energía continua.