Universidad de Valladolid

ESCUELA DE INGENIERÍA INFORMÁTICA

GRADO EN INGENIERÍA INFORMÁTICA
MENCIÓN EN TECNOLOGÍAS DE LA INFORMACIÓN

TRABAJO DE FIN DE GRADO

Estimación de la ocupación de un espacio físico a través de la captura de paquetes WiFi

---

Alumno: D. Aitor Ojeda Bilbao

Tutor: D. Jesús M. Vegas Hernández

Agradecimientos

Dedicado a mi familia, que siempre ha sido y será un pilar fundamental en mi vida, y en especial a mi abuela, que esto nunca lo habría conseguido sin ella.

A María, que siempre ha estado ahí para apoyarme y ayudarme en mis peores momentos.

A mis amigos y compañeros, con los que siempre he podido contar en las buenas y en las malas, ayudandándome en todo lo posible y compartiendo momentos memorables.

A mi tutor Jesús, que me ha ayudado en todo lo posible en este proyecto, siendo muy profesional y un gran tutor.

Resumen

El enorme crecimiento de los dispositivos personales, equipados con la última tecnología y con capacidades de comunicación inalámbricas, ha supuesto el surgimiento de diferentes servicios de alto nivel que se basan en la captura y en el procesamiento de los paquetes emitidos por dichos dispositivos. En este trabajo, tratamos de explotar dicha metodología para realizar un sistema que sea capaz de realizar una estimación de la ocupación, más o meos precisa, de las personas que están ocupando un lugar específico, una información que puede ser muy valiosa para diferentes escenarios que suceden hoy en día. Los sistemas tradicionales que existen actualmente y realizan un modelo de estimación de la ocupación, suelen ser costosos de implementar, o que no presentan un gran nivel de precisión en los resultados obtenidos. En este proyecto partimos de estos diferentes métodos tradicionales y proponemos un sistema con una mayor eficacia de resultados y un menor coste tanto de implementación cómo económico basado en la captura de tramas de solicitud Wifi emitidas por los dispositivos inalámbricos. El sistema implementa métodos de almacenamiento de los datos capturados que posteriormente serán transmitidos a una interfaz web, para que el usuario tenga constancia de lo que el sistema está realizando.

Abstract

The enormous growth of personal devices, equipped with the latest technology and with wireless communication capabilities, has led to the emergence of different high-level services that are based on the capture and processing of packets emitted by these devices. In this work, we try to exploit this methodology to create a system that is capable of estimating the occupation, more or less precise, of the people who are occupying a specific place, information that can be very valuable for different scenarios that happen today. The traditional systems that currently exist and perform an occupancy estimation model are usually expensive to implement, or do not present a high level of precision in the results obtained. In this project, we start from these different traditional methods and propose a system with greater efficiency of results and a lower cost both of implementation and economic based on the capture of Wifi request frames emitted by wireless devices. The system implements storage methods for the captured data that will later be transmitted to a web interface, so that the user is aware of what the system is doing.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

En este capítulo se hará una breve introducción de por qué ha surgido la idea de este proyecto, cuales son los objetivos que abordaremos, y una sencilla explicación del contenido del documento.

1.1. Motivación

Actualmente, estudios recientes llevados a cabo por CISCO, estiman que alrededor de 2021 habrá una cantidad de 11.6 billones de dispositivos inteligentes. Esto no engloba solamente a smartphones, si no también a todo tipo de dispositivos que sean capaces de conectarse a una red inalámbrica, tanto Wifi como Bluetooth. En paralelo a esta extensión de los dispositivos móviles, existen una gran diversidad de usos aprovechando los paquetes de mantenimiento de estos mismos dispositivo, que pueden ser capturados gracias a un mínimo elemento hardware que no requiere demasiada dificultad. Pueden destacar la localizacion de usuarios a traves de sus dispositivos, la estimación del comportamiento, la clasificación y la anonimación de los dispositivos, el análisis de mercado y muchos más.

Por ello, el problema que vamos a intentar resolver en este trabajo, es conseguir realizar una estimación de la ocupación en un espacio físico cerrado, a través de la captura de los paquetes de mantenimiento de estos dispositivos. El proyecto, además de las implicaciones descritas anteriormente, también puede destacar por la optimización y el mantenimiento de los edificios: como ejemplo, en los edificios de trabajo existen diferentes tipos sistemas que pueden ser optimizadas, como la luz, la calefacción, los conductos de ventilación o el aire acondicionado, entre otros. Un conocimiento del número de ocupantes de las diferentes salas puede reducir el consumo de estos sistemas entre un 20 % y un 80 %, según varios estudios realizados por diferentes investigaciones.

Esta optimización no solo se reduce al ámbito energético, si no que también puede utilizarse para mejorar la calidad de vida de las personas que ocupan un ligar determinado.En otro escenario, la información de la ocupación puede permitir a administradores del edificio y jefes de proyecto saber qué salas están disponibles para poder organizar encuentros o reuniones. En el caso de los administradores, conocer esta información en tiempo real, permite a los sistemas de monitoreo del edificio conocer cual es el lugar mas concurrido y poblado para poder organizar una de los ocupantes de manera organizada en caso de que se produzca una anomalía.

Por otro lado, también sirve para los empleados, ya que siendo conscientes de la información de la ocupación, se pueden evitar largas colas en la zona de descanso a la hora de comer, debido a algún servicio como la máquina de café o el microondas.

Además del ámbito laboral y energético, el problema que intentamos resolver también sirve para el ámbito educativo, por ejemplo, un conocimiento sobre la ocupación, podría permitir a los profesores determinar en qué salas se podría dar clases, organizar seminarios, e incluso controlar la asistencia y la presencialidad de sus alumnos, sin necesidad de hacer un control manual. Para los alumnos, puede ser útil a la hora de buscar una sala que esté disponible para realizar un proyecto con sus compañeros, o estudiar sin que nadie le pueda molestar.

Otro tipo de soluciones para este problema, como los sensores anbientales (C02, temperatura, humedad, etc) o las cámaras de vídeo, son enfoques que presentan ventajas y desventajas. Respecto a coste y precisión, los sensores ambientales no son demasiado caros para desarrollar, el problema que presnetan es que no son del todo precisos, y podrían dar una información falsa de lo que está ocurriendo. Por otro lado, las cámaras de vídeo es un sistema muy preciso, pero a la vez de un coste demasiado elevado, ya que es necesario que una persona esté controlando en todo momento qué está pasando y no se respetaría la privacidad de los ocupantes.

Esta solucón que estamos proponiendo, capturar la información a través de los paquetes wifi que envían los dispositivos, es una solución sencilla y que puede realizarse sin demasiado coste, solo es necesario un pequeño hardware para su implementación que servirá de dispositivo rastreador para estos paquetes. Los dispositivos wifi transmiten periódicamente solicitudes de sonda que sirven para recopilar información acerca del entorno en el que se encuentran.En este proyecto, desarrollaremos un sistema que sea capaz de recopilar las solicitudes de sonda de los dispositivos que se encuentran dentro de un espacio cerrado y utilizando un rastreador de bajo costo. El documento está estructurado en secciones.

1.2. Objetivos

En este apartado se realizará un análisis de cuales derán los objetivos que debe tener el proyecto incorporados para un completo funcionamiento y que srán de ayuda para posibles implementaciones futuras.

1. Objetivo principal del proyecto: Poder conseguir la capacidad de calcular de la forma más precisa posible el número de personas que se encuentran localizadas en un área geofráfica reducida, como una clase, un laboratorio, una sala, etc.
2. Realizar el objetivo principal de la forma más barata posible, para que el coste principal del proyecto no suponga un problema.
3. Mostrar la información que el sistema ha almacenado al usuario de la forma más clara y fácil de entender posible.
4. Utilizar información relevante para el sistema pero que no suponga un peligro para la privacidad del usuario del dispositivo que se ha extraído dicha información.
5. Mantener una clara transparecia con el usuario en todo momento a la hora del tratamiento de información relevante para el sistema.
6. Permitir que el sistema sea capaz de almacenar todos la información recopilada existiendo un límite muy alto de capacidad máxima.
7. Mostrar al usuario de forma clara los cambios que se van produciendo a lo largo de la ejecución en tiempo real.
8. Diseñar un sistema que sea capaz de poder implantarse en diferentes entornos sin que haya ningún tipo de restricción.

1.3. Estructura de la memoria

La memoria del proyecto se va a dividir en diferentes capítulos donde se hablarán de todos los aspectos necesarios para la comprensión y el entendimiento del sistema propuesto.

- ***Capítulo 2: Estado del arte.*** En este capítulo se hablará de los diferentes modelos de estimación que existen actualmente, además del modelo propuesto para el proyecto y la tecnología necesaria para la realización del mismo.
- ***Capítulo 3: Desarrollo.***En este capítulo se llevará a cabo un desarrollo completo del proyecto, desde su planificación hasta las pruebas necesarias para la comprobación de su correcto funcionamiento. Se abordarán todos los aspectos necesarios para la comprensión del proyecto así cómo los componentes que lo forman y los métodos que se han tomado para el diseño del mismo.
- ***Capítulo 4:Líneas futuras.*** Hablaremos sobre qué ha supuesto el proyecto de cara a fines educativos, al igual de los conocimientos adquiridos a lo largo del desarrollo del sistema.

Este documento también posee otra serie de secciones que no por ello son menos importantes:

- Bibliografía.
- Anexo A. Configuraciones. Apartado en el que se explican las configuracions del proyecto
- Anexo B. Manual de usuario. Manual de cómo debe utilizar el usuario el sistema
- Anexo C. Manual de instalación. Pasos necesarios para poder poner en marcha el sistema.
- Anexo D. Contenidos del CD-ROM

Capítulo 2

Estado del arte

En este capítulo se hablarán de todas las técnicas, métodos y bases fundamentales del proyecto, además de las herramientas utilizadas para la creación del mismo.

2.1. Trabajo relacionado

Como ya hemos descrito anteriormente, nuestro proyecto es una alternativa viable y asequible de diferentes enfoques y técnicas que han surgido a lo largo del tiempo para realizar el objetivo principal del proyecto: la estimación de personas en un lugar concreto. Existen varias técnicas de las que hablaremos a continuación.

2.1.1. Estimación de ocupación basada en sensores ambientales

Existen varios sensores ambientales que pueden determinar si una persona está presente en un lugar o no, o realizar la estimación de cuantas personas se encuentran en un área geográfica. En los espacios cerrados, la técnica más fiable actualmente de los sensores es el nivel de CO2, ya que permite realizar una estimación de las personas que se encuentran en una sala y a la vez respeta la privacidad de las mismas [3]. Esto se consigue analizando el gradiente del nivel de CO2 o resolviendo la ecuación de balance de masa de aire [4].
Pero este método sufre con problemas bastante comunes, como abrir una ventana o una puerta, que haría que el C02 en la sala aumentase o disminuyese, cayendo el nivel de precisión en la estimación a un nivel demasiado bajo. Por ello, es necesario que estos dos problemas, tanto el control de presencia como la estimación de ocupación, sean resueltos con técnicas de machine learning supervisadas, aun que aun así, los resultados de la estimación de la ocupación serían bastante pobres.

2.1.2. Estimación de ocupación basada en vídeo

Este segundo enfoque consiste en realizar la estimación de personas de un lugar determinado mediante cámaras de vídeo y técnicas d procesamiento de imágenes. El principal problema que plantea es la privacidad de los usuarios grabados, por ello, se utilizan cámaras con una resolución muy baja [6] o se instalan en lugares en los que no se identifique el rostro, como por ejemplo el techo. Sin embargo, otros enfoques grarantizan la privacidad usando sensores de infrarrojos pasivos (PIR)[7] o cámaras de profundidad [8] . En comparación con los sensores ambientales, esta técnica resulta mucho más fiable y precisa, pero es mucho más costosa de mantener e implementar, es por ello que algunos trabajos proponen enfoques híbridos que consisten en realizar ambas técnicas para una mejor fiabilidad y precisión .

2.2. Sistema propuesto

El sistema que vamos a desarrollar será un modelo de estimación que utilizará los paquetes Wifi que envían los dispositivos inalámbricos, rastreados por un hardware específico (Raspberry PI 3B+). Gracias a esto, podemos decir que es una solución sencilla y barata, ya que el coste del hardware necesario no supera los 50€, además de bastante efectiva, sin que existan factores externos que supongan un problema para la estimación. A continuación haremos una explicación más detallada sobre las técnicas de rastreo wifi que se utilizarán en el proyecto.

2.2.1. Rastreo wifi

Los puntos de acceso Wifi (APs) tienen un mecanismo para anunciar su presencia, cuyo funcionamiento se define cómo el envío de tramas de gestión de balizas (beacons) que contienen parámetros de configuración de red, así como el identificador de conjunto de servicios (SSID), que se refiere al nombre técnico de la identificación de la red, y también otros parámetros como las velocidades de datos compatibles. Estos beacons se transmiten por los canales 802.11b/g/n, y detectan de forma pasiva los puntos de acceso cercanos. La velocidad en la que se transmiten estas tramas depende de la velocidad de descubrimiento que se desee, además del ancho de banda permitido, es decir, sin sobrecargarse.

Tomando como base el procedimiento anteriormente mencionado, podemos explicar de manera similar cuál es el procedimiento que utilizan los dispositivos inálambricos. Éstos dispositivos tienen la misma forma de descubrir los puntos de acceso cercanos. En vez de mandar beacons, éstos dipositivos lo que hacen es mandar paquetes denominados "probe requests", en los que encapsulan dentro de ellos diferentes tipos de información. Esta información se manda en plano, es decir, la información no se encripta ni encapsula de ningún modo, ya que al ser tramas de descubrimiento puede que no existan APS cercanos. Los puntos de acceso responden a estas solicitudes con una respuesta de sonda, indicando donde se encuentran. Finalmente, estas solicitudes también sirven para conectarse a puntos de acceso ocultos, es decir, APs que no envían beacons.

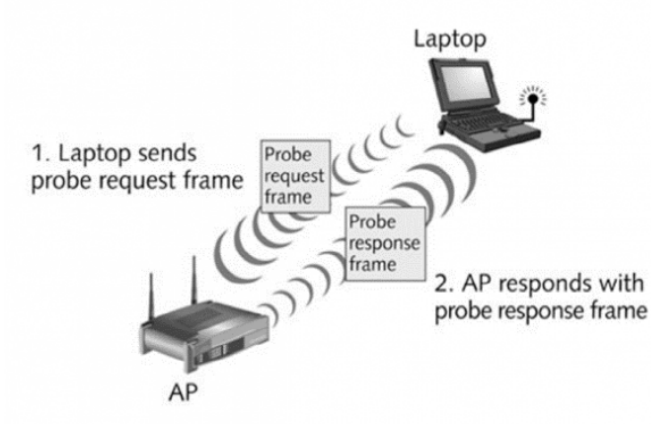

Figura 2.1: Ilustración de funcionamiento de probe requests
https://www.cwnp.com/802.11-mac-series-ndash-basics-mac-architecture-ndash-part-3/

En la figura 2.1 podemos observar cómo es el procedimiento de conexión a una red por parte de un dispositivo inalámbrico. Éste manda paquetes de solicitud de conexión, los probe requests, y el punto de acceso responde con otro paquete probe request indicando su posición geográfica para poder iniciar la conexión por parte del dispositivo. A continuación, haremos una breve explicación del contenido de un paquete probe request.

Frame Ctrl	Duration	Destination	Source	BSSID	SEQ	SSID	FCS
...	...	ff:ff:ff:ff:ff:ff	14:10:9F:d5:04:01	ff:ff:ff:ff:ff:ff	...	null	...
...	...	ff:ff:ff:ff:ff:ff	88:30:8a:49:db:0d	ff:ff:ff:ff:ff:ff	...	"PARC Visitor"	...

Figura 2.2: Estructura de un paquete probe request [2]

En la figura anterior observamos un ejemplo sencillo de un paquete probe request. El campo *Source* es la fuente desde donde se transmite el paquete, asociado a la dirección MAC. El campo SSID pertenece al nombre del punto de acceso al que se ha enviado la solicitud. SEQ es el número de secuencia del paquete, y el campo FCS es un código de verificación de redundancia. Destacar que el campo Frame Control indica el tipo de marco que es. Si es un paquete probe request, son tramas de gestión, por tanto su código será 0x00, con el subtipo

asociado 0x04. Gracias a este código podremos determinar que paquetes son probe request.

Dirección MAC

Es la principal información que contienen los probe requests. La dirección MAC de un dispositivo es un identificador de 48 bits, asociado de forma única a una tarjeta o dispositivo de red. También se la conoce como dirección física, y es única para cada dispositivo. Posee una estructura sencilla, cómo podemos observar en la imagen 2.3, donde podemos destacar múltiples aspectos que nos servirán de base para poder realizar el proyecto.

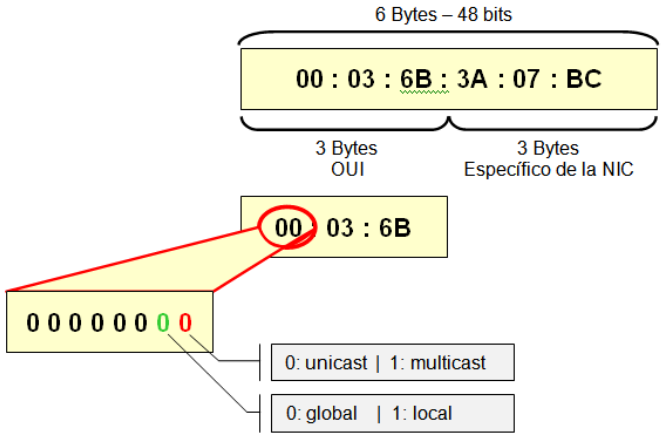

Figura 2.3: Estructura de una dirección MAC.
https://sistemas.tecnoderecho.com/direccion-mac-que-es/

Aparte de la dirección MAC, los probe requests también suelen contener los estándares que soporta el dispositivo y el SSID de la red a la que desean conectarse, además de la intensidad de la señal desde el dispositivo espía hacia el dispositivo del usuario, más conocido como RSSI (Received Signal Strength Indicator.

RSSI

En un sistema IEEE 802.11, RSSI es la fuerza relativa de la señal recibida en un entorno inalámbrico , en unidades arbitrarias. RSSI es una indicación del nivel de potencia que recibe el dispositivo receptor después de la antena. Por lo tanto, cuanto mayor sea el valor de RSSI, más fuerte será la señal. Si un valor RSSI se representa en forma negativa (por ejemplo, 100), cuanto más cerca esté el valor de 0, más fuerte ha sido la señal recibida. Este campo es muy importante, ya que gracias a él, podremos determinar cuando un dispositivo se encuentra dentro de una localización.

Una vez definido y explicado cuales son los campos más importantes que se encuentran dentro de la información de los probe requests, debemos plantear los 2 problemas fundamentales que debemos tener en cuenta a la hora de realizar el proyecto:

- **Direcciones MAC aleatorias**. Partiendo del principio anteriormente mencionado en el apartado de la dirección MAC, dado que los paquetes probe requests mandan la información de forma clara y en plano, sin cifrar, permiten realizar un seguimiento claro de los dispositivos por parte de un supuesto atacante. Para evitar este problema, los fabricantes transmiten direcciones MAC aleatorias, es decir, falsas, que a menudo se van actualizando. Por suerte, esta aleatorización es fácil de detectar, y basta simplemente con mirar la parte OUI de la dirección MAC. El campo OUI (Organization Unique Identification) identifica de forma única a un fabricante de tarjetas de red registrado, que se puede encontrar fácilmente en la tabla IEEE de fabricantes registrados. Si ésta dirección no aparece en la tabla, lo más probable es que se esté tratando de una dirección MAC aleatoria. Además, como podíamos observar en la imagen 2.3, el penúltimo bit del byte más significativo del OUI se conoce como bit local: si éste bit está establecido, la dirección MAC se asigna localmente y no necesita ser única. Un estudio reciente expuso que más del 99 % de las direcciones MAC asignadas localmente son de hecho aleatorias. Por lo tanto, la presencia del bit Local en el OUI es una fuerte indicación de aleatorización.

- **Umbral RSSI**. Idealmente, en este proyecto nos gustaría poder capturar los dispositivos que se encuentran dentro de una localización. Sin embargo, un hardware cuya tarea sea la de rastreador, puede localizar dispositivos que se encuentren en su rango de comunicación. Por tanto, debemos reducir el rango de comunicación de nuestro dispositivo para que ocupe sólo el rango que queremos analizar.Esto lo podremos hacer mediante un umbral RSS, considerando sólo los probe requests mayores a ese umbral, considerando que debe adaptarse de acuerdo al área que estemos analizando. Esto lo podremos ver más adelante en el capítulo 3.

2.3. Tecnología utilizada

En este apartado haremos una descripción de la tecnología, tanto métodos como herramientas utilizadas para el posterior desarrollo del proyecto.

2.3.1. Raspberry Pi B+

En primer lugar, el proyecto se ha realizado en un hardware especial, una raspberry pi, cuyo modelo es el 3 B+. Gracias a este modelo, podemos realizar el proyecto ya que los modelos a partir del 3, incluyen chip de red inalámbrico, para que el hardware sea capaz de establecer una conexión con cualquier punto de acceso inalámbrico. A continuación, haremos una breve descripción del modelo y de las características del mismo.

El modelo de la raspberry B+ surgió en el año 2018, siendo el sucesor del modelo B, que apareció por primera vez en el año 2015. Este modelo presenta una serie de mejoras respecto al modelo anterior. Lo más destacable de sus mejoras es su potencia y su conectividad, pasando de un procesador de 1,2 Ghz a uno mayor de 1.4 Ghz, y, en cuanto a la conectividad, el modelo B solo podía operar en el ámbito inalámbrico en la banda de 2.4 Ghz, mientras que el modelo B+ es capaz de operar en las dos bandas actuales, a 2.4 Ghz y a 5 Ghz. Además de esto, la potencia de su nuevo puerto Ethernet se triplica, pasando de 100 Mbits/s en el modelo B a 300 Mbits/s. También cuenta con Bluetooth 4.2 (Low Energy).

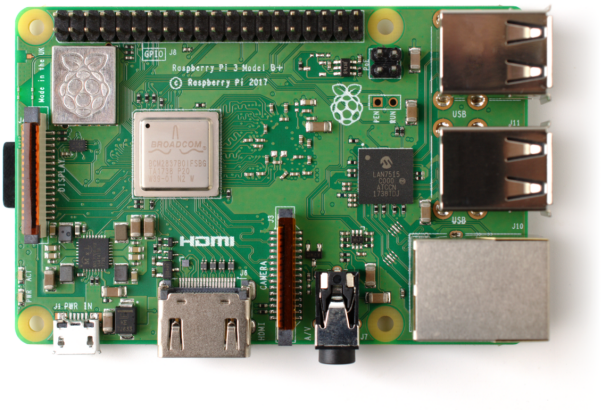

Figura 2.4: Aspecto de la Raspberry Pi 3 B+
https://www.amazon.com/-/es/Raspberry-Pi-Model-Tabla-3B/dp/B07P4LSDYV

2.3.2. Software Kali-Linux

Kali Linux es una distribución basada en Linux/Debian GNU que ha sido diseñada específicamente para realizar tareas de auditoría y seguridad informática, de manera general. Esta distribución es gratuita para cualquier usuario, y tiene instalados más de 600 programas con los que podemos realizar diferentes tareas relacionadas con la seguridad informática. Es muy popular entre la comunidad informática ya que permite una gran variedad de servicios de hacking, sniffing, tracking, etc, con los que el usuario puede probar y hacer tareas inocentes hasta tareas muy comprometidas que pueden suponer un problema para el mismo. Podemos destacra que, para el proyecto que vamos a realizar, permite una alta compatibilidad con dispositivos inalámbricos de largo alcance, admitiendo tantos dispositivos inalámbricos como sea posible y permitiéndolo funcionar correctamente en una amplia variedad de hardware. Para hacer una descripción más completa, destacaremos las principales herramientas que tiene integradas:

- **Nmap**. Nmap ("Network Mapper") es una herramienta de código abierto que sirve para la exploración de redes y la auditoría de seguridad. Esta herramienta ha sido diseñada para escanear rápidamente redes grandes, aunque también funciona bien contra hosts (**REFERENCIA**) únicos. Nmap utiliza paquetes IP sin procesar de de diferentes formas, para determinar qué hosts están disponibles en la red, qué servicios ofrecen esos hosts, qué sistemas operativos (y sus versiones) están ejecutando, qué tipo de filtros de paquetes / firewalls están en uso, y muchas más características, entre otras. Aun que Nmap se use comúnmente para auditorías de seguridad, muchos administradores de sistemas y redes lo encuentran útil para tareas rutinarias como el inventario de red, la administración de programas de actualización de servicios y el monitoreo del tiempo de actividad del host o del servicio.

- **Wireshark**. Wireshark es el analizador de protocolos de red más conocido y completo que se encuentra disponible actualmente. Esta herramienta permite ver lo que sucede en la red a un nivel microscópico y es el estándar real presente en muchas empresas comerciales y sin fines de lucro, agencias gubernamentales e instituciones educativas. El desarrollo de Wireshark prospera gracias a las contribuciones voluntarias de expertos en redes de todo el mundo.

- **John the Ripper**. John the Ripper es una herramienta de recuperación y auditoría de seguridad de contraseñas de código abierto disponible parauna gran variedad de sistemas operativos. Esta herramienta admite cientos de tipos de cifrado y hash, incluso para: contraseñas de usuario de versiones Unix (Linux, * BSD, Solaris, AIX, QNX, etc.), macOS, Windows, .ªplicaciones web"(como por ejemplo, WordPress), software colaborativo (por ejemplo, Notes / Domino) y servidores de bases de datos (SQL, LDAP, etc.); capturas de tráfico de red (autenticación de red de Windows, WiFi WPA-PSK, etc.); claves privadas cifradas (SSH, GnuPG, carteras de criptomonedas, etc.), sistemas de archivos y discos (archivos macOS .dmg y "paquetes dispersos", Windows BitLocker, etc.), archivos (ZIP, RAR, 7z) y archivos de documentos (PDF , Microsoft Office, etc.) Estos son solo algunos de los ejemplos que la herramienta posee, pero hay muchos más.

- **Aircrack-ng**. Aircrack-ng es un conjunto completo de herramientas para evaluar la seguridad de la red WiFi. Se enfoca en diferentes áreas de seguridad WiFi, principalmente 3: Supervisión: captura de paquetes y exportación de datos a archivos de texto para su posterior procesamiento por herramientas de terceros; Ataques: repetición de diferentes tipos de ataques, desautenticación, puntos de acceso falsos y otros muchos realizados mediante inyección de paquetes; Pruebas: comprobación de las tarjetas WiFi y las capacidades del controlador (captura e inyección) y ataque de contraseñas, tanto WEP como WPA PSK (WPA 1 y 2).

Esta es la herramienta que, a partir de un módulo que posee, utilizaremos para llevar a cabo el escaneo de los probe requests que mandan los dispositivos. Este módulo se llama *airmon-ng*. Es un script que sirve para activar el modo monitor de las tarjetas wireless, es decir, WiFi. Tambien puede usarse para parar las interfaces y salir del modo monitor. Su uso se realiza de la siguiente manera:

```
airmon-ng <start|stop> <interface> [canal]
```

Donde:

- start—stop. Órdenes que puedes realizar sobre la interfaz de red
- interface. Interfaz sobre la que quieres realizar la acción
- canal. Opcionalmente se puede añadir un número de canal

Posteriormente, se explicará de manera concreta el establecimiento de la Interfaz en modo monitor [Véase en el anexo]

2.3.3. Python

Para poder realizar toda la parte de codificación y desarrollo, se ha hecho uso del lenguaje de programación python. Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, dinámico y multiplataforma, ya que puede utilizarse sin nigún tipo de problema en muchos de los Sistemas Operativos que existen actualmente. En este proyecto utilizaremos la versión más reciente de Python, la versión Python 3.0, ya que es la versión más actual y con la que podremos desempeñar un mayor número de trabajos y tareas, contando con un mayor número de herramientas. Además de eso, lo hemos elegido por su facilidad a la hora de conectar con diferentes servicios, como un servidor web, ya que permite la conexión de manera rápida y eficaz.

Scapy

Scapy es el programa de Python que usaremos en este proyecto,y a que permite al usuario que lo utiliza enviar, escuchar, diseccionar y falsificar paquetes de red. Gracias a estas capacidades, el programa nos permite la creación de herramientas capaces de sondear, escanear, o incluso atacar las redes.En otras palabras, Scapy es un programa interactivo capaz de realizar tareas de manipulación de paquetes. Puede falsificar o decodificar paquetes de una amplia cantidad de protocolos, enviarlos por cable, capturarlos, hacer coincidir solicitudes y respuestas, entre muchas otras características. El programa realiza también la mayoría de tareas clásicas en una red, como escaneo, rastreo, sondeo, pruebas unitarias, ataques o descubrimiento de redes.

También realiza un buen desempeño en muchas otras tareas específicas que no son tan comunes, y que otras herramientas no son capaces de realizar, como enviar marcos no válidos, inyectar sus propios marcos 802.11, combinar técnicas (salto de VLAN + envenenamiento de caché ARP, decodificación VOIP en un canal cifrado WEP, ...), etc. Una vez explicadas las funcionalidades del programa, podemos concluir que la funcionalidad principal que se usará en este proyecto, será el rastreo de paquetes Wifi, más concretamente la de los paquetes denominados "probe requests", que envían los dispositivos cada vez que quieren conectarse a una red.

Flask

Flask es un framework de tamaño reducido, usado comúnmente en Python, que permite crear aplicaciones web rápidamente y con un mínimo número de líneas de código. Flask depende directamente de dos librerías específicas: El motor de plantillas Jinja y el kit de herramientas Werkzeug WSGI. La estructura que debe seguir un proyecto utilizando este módulo es muy estricta, a la par que sencilla, ya que las funciones y herramientas que utiliza están basadas en una estructura fija para que las operaciones sean correctas.

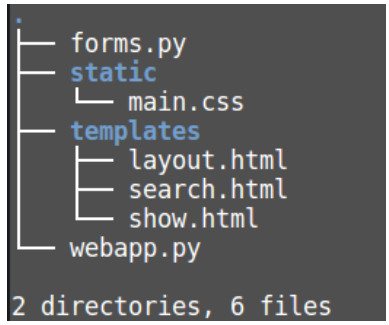

Figura 2.5: Estructura de un proyecto basado en Flask.
https://blog.nearsoftjobs.com/crear-un-api-y-una-aplicaci %C3 %B3n-web-con-flask-6a76b8bf5383

Gracias a este framework, en nuestro proyecto crearemos un servicio web basado en Flask, un servidor web al que podremos mandar peticiones básicas de HTML, como GET y POST, en el que el usuario podrá ver los resultados que ha producido el programa principal, así como las configuraciones necesarias que el usuario haya introducido en la web.

Sqlite3

SQLite es una biblioteca en lenguaje C que implementa un motor de base de datos SQL pequeño, rápido, autónomo, de alta confiabilidad y con todas las funciones necesarias para un almacenamiento de datos correcto. Es de los motores de base de datos más utilizado del mundo. A parte de ser de los más utilizados, SQLite está integrado en todos los teléfonos móviles y la mayoría de los ordenadores actuales, y viene incorporado dentro de muchas otras aplicaciones que las personas utilizamos día a dia.

En este proyecto, hemos elegido esta biblioteca ya que se adapta perfectamente a las herramientas que vamos a utilizar, y a parte de ser un motor de base de datos sencillo, nos provee de múltiples servicios, como el cumplimiento de las propiedades ACID (Atomicity, Consistency, Isolation, Durability) de las transacciones, su uso directamente, ya que no requiere ninguna configuración, su gran capacidad para el almacenamiento de datos, soportando tamaños de fila de hasta 1Gb, entre otros muchos.

Capítulo 3

Desarrollo

3.1. Metodología y planificación

Para poder afrontar de manera correcta el proyecto, se llevará a cabo primero una planificación del mismo, en la que podremos distinguir diferentes fases de la planificación con el fin de dejar lo más claro posible la realización del proyecto.

En primer lugar, el enfoque que vamos a tener en cuenta para la implementación será el de "Desarrollo en cascada". Es el modelo clásico de desarrollo del sistema y puede ser muy favorable para el proyecto. Al final de cada fase, este modelo crea hitos naturales, es decir, comprobaciones para verificar que se han cumplido los requisitos establecidos en el primer momento.Los jefes del proyecto al final de cada fase son los encargados de revisar el progreso del mismo e identificar si el caso de negocio para el modelo es válido, habiéndose cumplido los requisitos establecidos. Este modelo presenta diferentes fases que abordaremos a lo largo de este capítulo:

- *Estudio de la viabilidad*. En esta etapa, se hará un estudio sobre la viabilidad del proyecto,es decir, si el proyecto resulta viable y posible, además de los costes que suponga la realización de las tareas, el tiempo que tomará abordar las tareas, entre otras , centrándose principalmente en establecer los plazos y los posibles riesgos que puede encontrarse el proyecto durante su realización.
- *Requisitos de usuario*. Analizaremos de forma clara y específica cuales son los requisitos que debemos afrontar y los objetivos que debe tener el proyecto para que los usuarios puedan realizar un uso correcto del mismo. Esto se llevará a cabo con reuniones periódicas entre el desarrollador y el jefe de proyecto donde se expondrán de manera clara los requisitos y objetivos que debe cumplir el sistema.

- *Análisis*. En esta etapa se llevará a cabo un análisis sobre los requisitos funcionales y no funcionales que debe incorporar el sistema para que su funcionamiento sea correcto, cumpliendo con los objetivos especificados en el apartado anterior.
- *Diseño del sistema*. Se expondrá de manera clara una descomposición del sistema principal en tareas independientes para facilitar el desarrollo del proyecto, que al juntar todas puedan formar el sistema final.
- *Diseño del producto*. En este apartado se explicará la estructura que deberán llevar las tareas, al igual que los productos entregables generados por las mismas.
- *Codificación*. Apartado en el que se explicarán todas las técnicas, herramientas, métodos y funciones que se han utilizado para el desarrollo del sistema, al igual que los problemas que hayan surgido durante el periodo de desarrollo.
- *Pruebas*. Durante el periodo de codificación será necesario ir probando el sistema con cada tarea realizada para comprobar su correcto funcionamiento. Todo las pruebas quedarán recogidas en este apartado.
- *Funcionamiento y mantenimiento*. Es la puesta en marcha completa del proyecto finalizado, por parte del usuario, para comprobar que se han implementado todos los requisitos y objetivos propuestos al principio del desarrollo.

Roles del proyecto

Para poder comenzar a realizar el desarrollo completo del proyecto, primero de ello debemos identicar a los actores que serán los encargados de llevar a cabo la realización del proyecto. En este proyecto, podremos considerar 3 roles principales: Jefe de proyecto, Jefe de equipo, y Desarrollador.

Persona	**Rol**
Jesús María Vegas Hernández	Jefe de proyecto
Aitor Ojeda Bilbao	Desarrollador y jefe de equipo

Cuadro 3.1: Tabla de los roles del proyecto

El jefe de proyecto será el encargado de tomar las decisiones importantes del proyecto, además de establecer los objetivos principales del proyecto junto con los requisitos. Por otra parte, el jefe de equipo será el encargado de establecer los tiempos que deben de tomar las tareas, al igual que la asignación de las mismas a los desarrolladores.
Por último, el desarrollador será en el encargado de realizar todo el trabajo práctico del proyecto, es decir, la implementación completa, siempre teniendo en cuenta las decisiones del jefe de equipo y del jefe de proyecto, para seguir de acuerdo a las decisiones tomadas.

3.1.1. Planificación del proyecto

Para empezar a darle forma al proyecto, es necesario realizar una correcta estimación sobre el tiempo que va a tomar realizar cada fase. Por ello, debemos identificar y tener en cuenta los riesgos que pueden surgir durante la realización del mismo. Podemos asumir que los riesgos serán propiamente de software, ya que el proyecto no está basado en la unión de componentes hardware, si no en un programa que realice una función específica. Debido a esto, el riesgo más común que podremos tratar será el mal funcionamiento de los componentes instalados necesarios para el desarrollo del proyecto, lo cual deberemos preveer en los tiempos de estimación a la hora de realizar las diferentes fases.

Como el modelo de sistema que se utilizará en el proyecto es el "Desarrollo en cascada", a la hora de hacer una estimación de la duración de las fases, debemos tener en cuenta que un retraso en alguna de las fases, supondrá un retraso total en la finalización del proyecto, ya que para poder empezar una nueva fase, la fase anterior debe estar concluida.

Para realizar una correcta estimación de los tiempos de las fases que se abordarán, primero debemos establecer un tiempo coherente relacionado con los 3 tipos de enfoques que existen en el desarrollo de sistemas: optimista, pesimista, y más probable.

- Enfoque optimista. Se refiere a la estimación de las tareas suponiendo que el desarrollo ha fluido con normalidad, y no ha surgido ningún problema durante el mismo.
- Enfoque pesimista. Se refiere a la estimación de las tareas desde un punto pesimista, es decir, haciendo una suposición de que todos los posibles riesgos que puedan surgir durante el desarrollo se llevarán a cabo.
- Enfoque más probable. Es el enfoque más recomendado, una unión de los dos anteriores. Sirve para realizar una estimación mas precisa sin necesidad de llegar a los puntos más extremos, es decir, una duración de las tareas en circustancias normales, teniendo en cuenta los riesgos más probables dentro de la estimación.

El proyecto tendrá como fecha de inicio el 02 de Febrero de 2020, y como hemos visto en el apartado anterior, utilizando el enfoque más probable, la fecha de finalización sería el 6 de Julio de 2020. En la estimación de tiempos, se utilizarán las fases propias del "Desarrollo en cascada", estableciéndose los siguientes tiempos:

Fase del proyecto	**Comienzo**	**Fin**	**Duración**
Planificación	03/02/20	10/02/20	7 días
Análisis de requisitos	11/02/20	25/02/20	14 días
Diseño del sistema	26/02/20	11/03/20	14 días
Diseño del programa	12/03/20	26/03/20	14 días
Codificación	19/06/20	26/06/20	45 días
Pruebas	29/05/20	05/06/20	7 días
Funcionamiento y mantenimiento	08/06/20	15/06/20	7 días

Cuadro 3.2: Estimación de la duración de las fases del proyecto

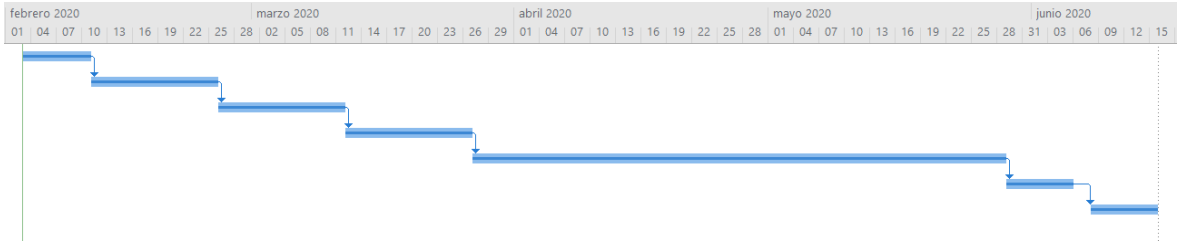

Figura 3.1: Diagrama de Gantt de las fases del proyecto

En el diagrama anterior podemos observar con claridad la tarea más significativa del desarrollo, la codificación, dónde es necesario realizar una estimación más elevada de los tiempos de la tarea ya que es la tarea principal dónde nos pueden ocurrir los riesgos más probables.Destacar que para que el desarrollo de las tareas avance, se realizarán reuniones con el jefe de proyecto al final de cada tarea para comprobar los resultados y discutir si los objetivos propuestos al principio del proyecto se van logrando.

Por último, debemos realizar una asignación de las fases con los participantes del proyecto, para así garantizar una mejor organización del mismo. La asignación debe hacerse en relación a qué personas son necesarias en el desarrollo de una fase, para que ésta no deje niguna tarea sin hacer u objetivos importantes sin contemplar. En la siguiente tabla se muestra la asignación de las fases con los roles mencionados anteriormente.

Fase del proyecto	**Rol asignado**
Planificación	Jefe de Proyecto y Jefe de equipo
Análisis	Jefe de Proyecto y Jefe de equipo
Diseño del sistema	Jefe de proyecto, Jefe de equipo y Desarrollador
Diseño del producto	Jefe de proyecto, Jefe de equipo y Desarrollador
Codificación	Jefe de equipo y Desarrollador
Pruebas	Desarrollador
Funcionamiento y mantenimiento	Jefe de equipo y Desarrollador

Cuadro 3.3: Asociación Fases - Roles del proyecto

3.1.2. Viabilidad y costes del proyecto

En este apartado nos centraremos en el análisis de la viabilidad del proyecto, es decir, un enfoque más económico que nos permitirá saber si el proyecto puede ser abordado o no, buscando en todo momento la rentabilidad del mismo.

Desde un enfoque económico, el proyecto no supone un coste muy elevado, ya que los elementos que describiremos a continuación son dispositivos corrientes y no hacen que el coste del proyecto se dispare. Los elementos principales y necesarios utilizados para el proyecto son:

Nombre del dispositivo	**Cantidad**	**Precio**
Raspberry Pi 3 B+	1	40 €
Tarjeta de memoria SanDisk Ultra microSDHC	1	7 €

Cuadro 3.4: Elementos necesarios para el proyecto

En la tabla anterior se muestran los precios de los dispositivos necesarios. Por tanto, el coste del proyecto sería aproximadamente de unos 47€. Con este precio total, podemos afirmar que el proyecto no supone un coste elevado, si no bastante asequible para cualquier desarrollador ya que necesita pocos componentes para ponerlo en funcionamiento.

Es necesario destacar, que para poder realizar el proyecto, se necesitan algunos componentes hardware. Estos componentes, si no se dispone de ellos, pueden ser comprados o alquilados, de forma que sería recomendable añadir su coste al coste total del proyecto. Los elementos que se necesitan son: un ratón, un teclado, y un monitor. Estos dispositivos, no se amortizan hasta pasados 4 años desde su compra, por consiguiente debemos hacer un cálculo relacionado entre su precio total, y su periodo de amortización en el proyecto. En la siguiente tabla se mostrarán los precios genéricos de los dispositivos utilizados, haciendo una pequeña media entre los diferentes modelos que existen en el mercado.

Dispositivo	**Precio**	**Periodo**	**Coste/Mes**	**Coste Total**
Monitor	130 €	6 meses	2.70 €	16.20 €
Ratón	10 €	6 meses	0.20 €	1.20 €
Teclado	15 €	6 meses	0.32 €	1.92 €

Cuadro 3.5: Costes del equipo necesario para el proyecto

Además de los costes mencionados en la tabla anterior, debemos tener en cuenta que para avanzar en el proyecto, se necesita personal. Por consiguiente, deberíamos recalcular el coste de la mano de obra necesaria para la realización del proyecto. El personal necesario es un desarrollador software para poder realizar las tareas sin ningún problema. Tomando como base el salario de un desarrollador software junior medio, cuya media anual es de 18000€brutos y como sólo desarrollaría su función en el proyecto durante 3 meses, podemos afirmar que su contratación costarían 4500€brutos al coste total del proyecto. Destacar que la jornada laboral sería de 8 horas de lunes a viernes.

Por último, tomando toda la información mostrada anteriormente, también habría que añadir al proyecto el coste total de amortización de los dispositivos utilizados. Hecho esto, podemos afirmar que el proyecto tendría un coste total de 4566.32€

Además, desde el enfoque económico, podemos plantear dos opciones: desarrollando el proyecto reutilizando material y realizando el desarrollo sin contratación o comprando todo nuevo y contratando personal. En nuestro caso, no ha sido necesaria la contratación de personal, y hemos podido reutilizar los siguientes elementos del material: Raspberry Pi B+, Ratón HP y la Tarjeta de memoria, ya que por motivos que se explicarán más adelante no hemos podido desarrollar el proyecto en un laboratorio. Asi que debido a esto, el coste real del proyecto ha sido de unos 19.42€

Retraso excepcional del desarrollo: COVID-19

Este año 2020, ha surgido un problema a la hora de continuar con la implementación del proyecto, y continuar con las fechas de planificación previstas: el Coronavirus o más técnicamente conocido cómo COVID-19. Debido a este contratiempo, el gobierno de España decretó el Estado de alarma en todo el territorio nacional el día 14 de marzo de 2020. El desarrollo del proyecto se vió afectado, y quedó paralizado hasta que la situación fuera diferente. Con el paso del tiempo, la situación no cambiaba, entonces se llegó a un acuerdo de continuar con el proyecto de la manera en la que fuera posible. El desarrollo del proyecto quedó paralizado en la etapa de Diseño del sistema, ya que aun no concordar con la fecha del Estado de alarma, tomó un poco más de tiempo del planificado. Por tanto, las nuevas fechas de desarrollo del proyecto ha sido necesario cambiarlas, dónde se pueden ver reflejadas en la siguiente tabla.

Fase del proyecto	Comienzo	Fin	Duración
Planificación	03/02/20	10/02/20	7 días
Análisis de requisitos	11/02/20	25/02/20	14 días
Diseño del sistema	1/06/20	8/06/20	7 días
Diseño del programa	9/06/20	23/06/20	14 días
Codificación	24/06/20	12/08/20	49 días
Pruebas	13/08/20	20/06/20	7 días
Funcionamiento y mantenimiento	21/08/20	28/08/20	7 días
Duración total del proyecto	105 días		

Cuadro 3.6: Estimación de la duración de las fases del proyecto después de la COVID-19

Recalculados los tiempos de planificación, a continuación se mostrará el diagrama de gant real en el que se indica el retraso excepcional debido a la COVID-19.

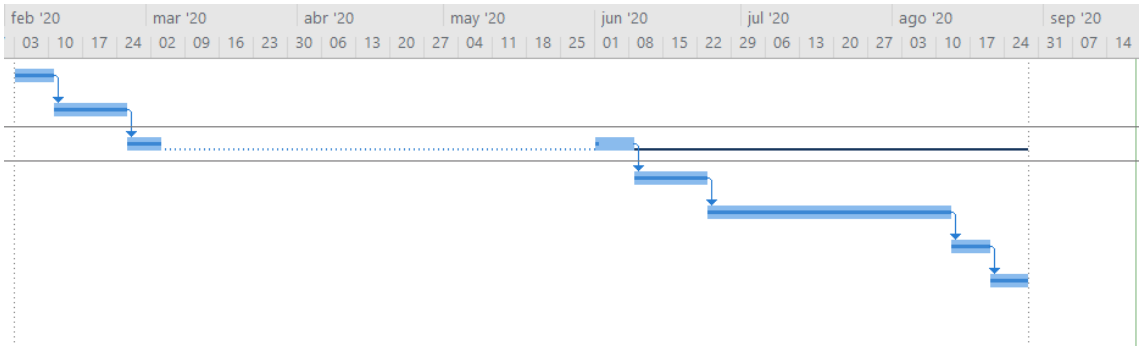

Figura 3.2: Diagrama de Gant con la situación excepcional

3.2. Análisis

En este apartado se realizará un análisis en profundidad de los requisitos que debe incorporar el proyecto para que su desarrollo sea completo. Teniendo en cuenta el objetivo principal del proyecto, que es conseguir realizar una estimación de las personas que se encuentran en un lugar a través de los paquetes wifi enviados por los dispositivos, se hará un estudio de los requisitos indispensables para lograr el objetivo.

3.2.1. Análisis de requisitos

Los requisitos son una propiedad que ha de exhibir el software a desarrollar para resolver un problema determinado. Además de ello, cumplen una doble función: son una declaración de un servicio que el sistema debe proporcionar, y además una definición formal específica del sistema. Podemos diferenciar 2 clases de requisitos en las que se divide nuestro proyecto:

- **Requisitos funcionales**. Describen el comportamiento del sistema, los servicios que el sistema debe proporcionar, cómo debe reaccionar frente a los cambios y ante situaciones particulares. Estos requisitos son los más importantes ya que definen los servicios que implementará el sistema.
- **Requisitos no funcionales**. Se refieren a las limitaciones que debe seguir el sistema para realizar correctamente los servicios que ofrece. Son aspectos tales como el tiempo de respuesta, el almacenamiento, la seguridad y la fiabilidad, etc.

A continuación, veremos cada uno de los requisitos propuestos para el sistema con un mayor grado de detalle, mostrando su descripción, su prioridad , el nombre del requisito, su tipo, si es funcional o no funcional, y además, una breve explicación sobre el motivo de elección del requisito propuesto.

Nombre del requisito	RF1 - Almacenamiento datos de usuario
Tipo de requisito	Funcional
Prioridad	Alta
Descripción	El sistema debe ser capaz de almacenar los datos introducidos por el usario desde la interfaz web
Razón de implementación	Es necesario el requisito ya que para que el sistema realice correctamente la funcionalidad, los datos introducidos por el usuario son indispensables, y deben ser guardados como tal

Cuadro 3.7: Requisito Nº1

Nombre del requisito	RF2 - Funcionalidad de usuarios
Tipo de requisito	Funcional
Prioridad	Alta
Descripción	El sistema debe ser capaz de permitir la funcionalidad a cualquier tipo de usuario que esté utilizando el sistema
Razón de implementación	Ya que en el sistema no se distinguen usuarios por su grado de seguridad al realizar las tareas, hemos creído consecuente permitir la funcionalidad a cualquier tipo de usuario

Cuadro 3.8: Requisito Nº2

Nombre del requisito	RF3 - Muestra de datos
Tipo de requisito	Funcional
Prioridad	Alta
Descripción	El sistema debe ser capaz de mostrar la información almacenada en la base de datos de una forma clara para el usuario
Razón de implementación	Para poder ver el trabajo que se está realizando por debajo, la interfaz web mostrará los datos que están almacenados en la base de datos para que el usuario sea consciente de ellos

Cuadro 3.9: Requisito Nº3

Nombre del requisito	RF4 - Rastreo de paquetes
Tipo de requisito	Funcional
Prioridad	Alta
Descripción	El sistema debe ser capaz de realizar un rastreo de paquetes para recoger únicamente los paquetes que se consideran necesarios para el proyecto.
Razón de implementación	Esta es la funcionalidad más esencial del proyecto, es decir, es el núcleo del mismo, y por ello es necesario su inclusión

Cuadro 3.10: Requisito Nº4

Nombre del requisito	RF5 - Filtrado de información
Tipo de requisito	Funcional
Prioridad	Alta
Descripción	El sistema debe ser capaz de realizar un filtrado con los datos recogidos para únicamente almacenar los datos esenciales
Razón de implementación	No podemos almacenar toda la información que el dispositivo rastreador vaya capturando, por tanto es necesario que se filtre dicha información y se almacene únicamente la imprescindible.

Cuadro 3.11: Requisito Nº5

Nombre del requisito	RF6 - Actualizar datos almacenados
Tipo de requisito	Funcional
Prioridad	Alta
Descripción	El sistema debe ser capaz de actualizar los datos almacenados con los datos recogidos para que la información sea la más reciente posible.
Razón de implementación	Para que los datos sean coherentes, el sistema debe actualizar los datos almacenados cada vez que se encuentre con el mismo principio bñasico de información, es decir, la dirección MAC.

Cuadro 3.12: Requisito Nº6

Nombre del requisito	RF7 - Comprobación de presencia
Tipo de requisito	Funcional
Prioridad	Alta
Descripción	El sistema debe ser capaz de comprobar si un dispositivo se ha movido de la sala o comprobar si permanece en ella
Razón de implementación	Este requisito es necesario ya que debemos determinar en todo momento las personas que se encuentran en un lugar, y actualizar lo más rápido posible los cambios que puedan surgir, es decir, entrar o salir

Cuadro 3.13: Requisito Nº7

Nombre del requisito	RF8 - Servicio Web
Tipo de requisito	Funcional
Prioridad	Alta
Descripción	El sistema debe ser capaz de implementar un servicio web para que exista una interacción entre el usuario y el sistema
Razón de implementación	Este requisito es fundamental ya que sin él, el sistema creado no dotaría de ningún tipo de interacción con el usuario final.

Cuadro 3.14: Requisito Nº8

Nombre del requisito	RNF1 - Conexión a internet
Tipo de requisito	No Funcional
Prioridad	Alta
Descripción	El sistema debe ser capaz de tener una conexión a internet configurada correctamente para realizar parte de la funcionalidad
Razón de implementación	Necesitamos una conexión a internet para poder realizar la muestra de datos al usuario

Cuadro 3.15: Requisito Nº9

Nombre del requisito	RNF2 - Almacenamiento de datos
Tipo de requisito	No Funcional
Prioridad	Alta
Descripción	El sistema debe ser capaz de almacenar todos los datos recogidos a lo largo de la ejecución en cualquier tipo de sistema de almacenamiento
Razón de implementación	Es necesario que todos los datos que se rastreen sean recogidos, y el sistema tenga capacidad suficiente como para almacenarlos

Cuadro 3.16: Requisito Nº10

Nombre del requisito	RNF3 - Actualización datos tiempo real
Tipo de requisito	No Funcional
Prioridad	Media
Descripción	El sistema debe ser capaz de reaccionar a los cambios que se producen en el almacenamiento de manera inmediata, de modo que el usuario no tenga que realizar ninguna acción para poder ver esos cambios
Razón de implementación	Para facilitar el uso del sistema al usuario, se implementará unmétodo para actualizar los datos en timepo real

Cuadro 3.17: Requisito Nº11

Nombre del requisito	RNF4 - Funcionamiento sin errores
Tipo de requisito	No Funcional
Prioridad	Alta
Descripción	El sistema debe ser capaz de funcionar durante todo el tiempo de la ejecución, sin que surja ningún tipo de error
Razón de implementación	Como cualquier sistema de software, la ejecución del programa no debe de dar nigún error a la hora de la ejecución

Cuadro 3.18: Requisito Nº12

Nombre del requisito	RNF5 - Usabilidad de usuario
Tipo de requisito	No Funcional
Prioridad	Media
Descripción	El sistema debe ser fácil de utilizar para cualquier tipo de usuario que quiera utilizarlo
Razón de implementación	Es conveniente que el sistema desarrollado no suponga ningún problema de aprendizaje para el usuario, y que entienda de manera sencilla cómo funciona el sistema

Cuadro 3.19: Requisito Nº13

Nombre del requisito	RF9 - Errores de usuario
Tipo de requisito	Funcional
Prioridad	Alta
Descripción	El sistema debe ser capaz de reaccionar a los errores que pueda provocar el usuario sin que se interrumpa la ejecución del mismo
Razón de implementación	Es necesario realizar siempre una comprobación de las acciones que pueda tomar el usuario en el sistema para que si ocurre un error, el sistema pueda recuperarse y continuar con la ejecución

Cuadro 3.20: Requisito Nº14

Nombre del requisito	RNF6 - Comprobación Aforo
Tipo de requisito	No Funcional
Prioridad	Baja
Descripción	El sistema debe ser capaz detectar cuando el aforo de la sala se ha completado y mostrarlo a través de la interfaz web
Razón de implementación	Para realizar un mejor desarrollo, hemos decidido que sería de ayuda poder mostrar al usuario cuando se ha completado el aforo de la sala

Cuadro 3.21: Requisito Nº15

3.2.2. Análisis de los casos de uso

Una vez realizado el análisis de los requisitos que debe cumplir el sistema, vamos a proceder a explicar los diferentes casos de uso que puede realizar el usuario en el mismo, es decir, la interacción que tendrá el usuario con el sistema de cara a los servicios que éste ofrece. Los casos de uso se definen como una secuencia de acciones realizadas por el sistema, que producen un resultado valioso para un actor en particular. Estas acciones son procedimientos atómicos [ref] y se invocan cuando el actor envía un estímulo al sistema.

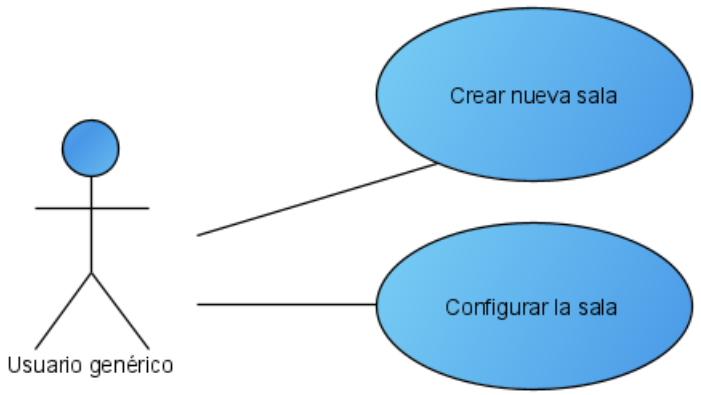

Figura 3.3: Casos de uso del sistema

En la imagen anterior podemos observar los casos de uso que puede realizar el usuario en el sistema propuesto, que son 2: Crear una nueva sala, y configurar una sala. Para poder dar una mejor visión de los casos de uso, a continuación se explicarán con más detalle en las siguientes tablas.

Identificador	**Crear nueva sala**
Descripción	El sistema deberá permitir al usuario genérico crear una sala nueva desde el escenario descrito a continuación
Precondición	El usuario debe de estar conectado al servidor de aplicación del sistema para poder realizar la acción
Secuencia normal de ejecución	1 El actor usuario genérico accede a la interfaz web del sistema tecleando la dirección principal del servidor 2 El sistema muestra la interfaz web propia de la aplicación, en la página de crear sala 3 El actor usuario genérico introduce los datos de la sala y pulsa el botón "Guardar sala" 4 El sistema muestra una vista donde aparece un mensaje de que se ha guardado correctamente la sala 5 El actor usuario genérico pulsa en el enlace *Ir a la página de inicio* 6 El sistema procesa la acción y carga los datos almacenados de la sala, así como la información rastreada, en la vista de la página de Inicio
Postcondiciones:	
Excepciones	3 El actor usuario genérico introduce datos no válidos 3.a El sistema muestra un mensaje de error indicando que los datos introducidos no son válidos 3.b El actor usuario genérico regresa al paso 2

Cuadro 3.23: Caso de uso Crear nueva sala

Identificador	**Configurar una sala**
Descripción	El sistema deberá permitir al usuario genérico configurar una sala previamente creada desde el escenario descrito a continuación
Precondición	El actor usuario genérico debe de haber realizado previamente el caso de uso Çrear nueva sala"
Secuencia normal de ejecución	1 El actor usuario genérico pulsa en el enlace Çonfigurar sala" 2 El sistema muestra la interfaz web propia de la aplicación, en la página de configurar sala 3 El actor usuario genérico introduce los nuevos datos de la sala y pulsa el botón "Guardar sala" 4 El sistema muestra una vista donde aparece un mensaje de que se ha guardado correctamente la sala 5 El actor usuario genérico pulsa en el enlace *Ir a la página de inicio* 6 El sistema procesa la acción y carga los datos almacenados de la nueva sala, así como la información rastreada, en la vista de la página de Inicio
Postcondiciones:	
Excepciones	3 El actor usuario genérico introduce datos no válidos 3.a El sistema muestra un mensaje de error indicando que los datos introducidos no son válidos 3.b El actor usuario genérico regresa al paso 2

Cuadro 3.25: Caso de uso Configurar una sala

3.3. Diseño

3.3.1. Diseño del sistema

En esta sección se describirá de una manera más detallada el diseño del sistema, las partes en las que se divide el sistema, su arquitectura, la estructuración de los datos almacenados, etc. El primer paso es definir la estructura del sistema que vamos a desarrollar, es decir, en que tareas se dividirá nuestro proyecto. Para ello, nos serviremos del "Workdown Breakdown Estructure". Este modelo sirve para realizar una estructura del sistema basado en las actividades que se realizarán en el proyecto, por tanto, es necesario identificar las tareas principales que incluirá el proyecto para completarlo y luego desglosar cada una de ellas en un conjunto de tareas de nivel inferior. Las actividades se añaden a una rama de la estructura si contribuyen directamente a la tarea inmediatamente superior; si no contribuyen a la tarea principal, no se deben añadir a esa rama. Para hacer una mejor explicación, en la imagen 3.4 se ilustrará el Work Breakdown Structure de nuestro proyecto.

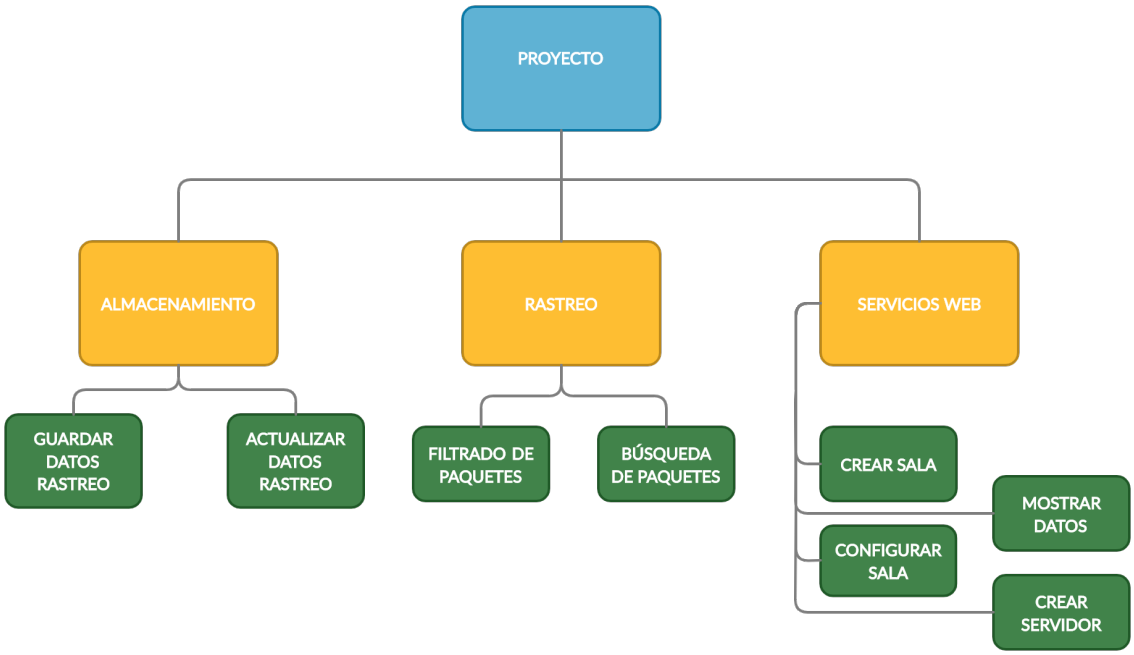

Figura 3.4: Work Breakdown Structure del proyecto

En la imagen anterior podemos observar como nuestro proyecto se divide en tres tareas principales, las cuales cada uno está formada por subtareas que harán posible su realización: Tareas de alamcenamiento, tareas de rastreo y tareas de servicios web.

- **Almacenamiento**. Dentro de este apartado se implementarán todas las subtareas relacionadas con el almacenamiento de datos que realizará el sistema, además no sólo el almacenamiento, sino también la actualización de los mismos en caso de repetición.
- **Rastreo**.Es el núcleo del proyecto, la tarea en la que está basada todo el sistema. Se implementarán las subtareas de la búsqueda de paquetes y , complementaria a esta, una subtarea que servirá como filtro de la información recogida para poder almacenarla.
- **Servicios Web**. Esta es la tarea principal para poder mostrar los resultados que presenta el sistema, mostrando los datos al usuario en una interfaz web, realizando tareas de configuración de sala, y para poder implementar todo esto, una subtarea que consistirá en levantar un servicio web, que hará de servidor para las peticiones de los clientes.

3.3.2. Diseño del producto

En esta sección se hará una explicación breve sobre el producto que va a generear la implementación del proyecto. En primero lugar, para poder realizar su descripción, se ha elegido hacerlo mediante el "Product Breakdown Structure", un enfoque similar al "Work Breakdon Structure.[ex]plicado anteriormente. El PBS consiste en indicar, para cada producto, que otros productos son requeridos como insumos. Por lo tanto, el PFD puede transformarse facilmente en una lista ordenada de actividades, identificando las transformaciones que convierten unos productos en otros. En la figura 3.5 podemos observar una ilustración de nuestro proyecto aplicando el PBS, donde el nivel 0 corresponde al proyecto; el nivel 1 corresponde a las partes software del mismo; y el nivel 3 refleja los componentes necesarios para realizar por completo el software mencionado.

Además, destacar que cada uno de los componentes que se van realizando, al igual que las partes completas del software, se irá haciendo una comprobación por parte del Jefe de equipo y el jefe del proyecto para asegurar que el componente sofware cumple con los requisitos necesarios y su funcionamiento es el adecuado.

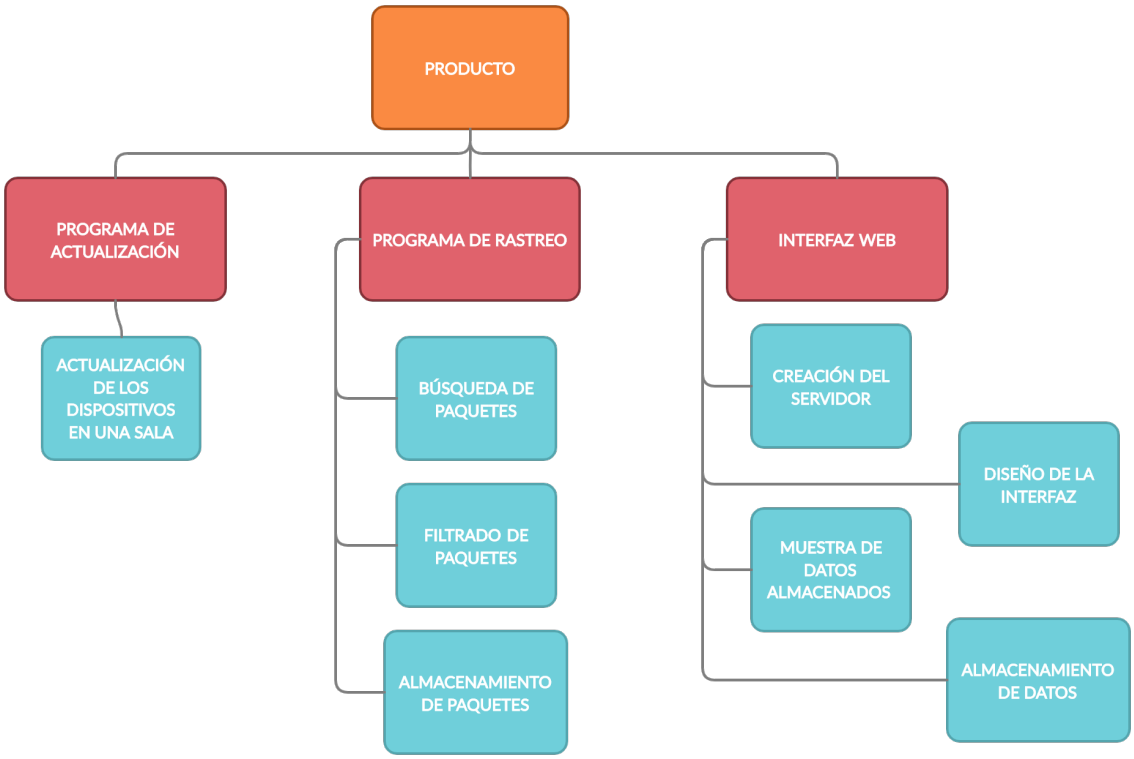

Figura 3.5: Product Breakdown Structure del proyecto

3.3.3. Modelo de dominio del sistema

El modelo de dominio del sistema es es un modelo conceptual de todos los temas relacionados con un problema específico. Es un diccionario visual de conceptos, es decir, no son componentes del sofware, es la lógica que mantendrá el sistema de cara al tratamiento de los datos. Es necesario identificar las clases pertenecientes, los atributos que contienen y las asociaciones/generalizaciones que existen entre ellas.

En nuestro proyecto, existen 3 clases principales y una clase de tipo enum [23] : la clase Sala, la clase DatosRastreo, la clase Usuario y la clase Tamaño. La clase usuario es una clase representativa, de la que no guardaremos información, que nos sirve para ilustrar mejor el problema que estamos resolviendo.
La clase DatosRastreo se relaciona con una sala, pudiendo tener ésta 0 o múltiples datos. Sin embargo, unos datos de rastreo específicos pertenecen a una única sala. Los atributos que contiene son:

- **mac**. Dirección mac del paquete rastreado.
- **mac_info**. Nombre del proovedor del chip de red.
- **rssi**. Intensidad de la señal recibida por parte del dispositivo rastreador del paquete.
- **fechaHora**. Fecha y hora en la que se ha rastreado el paquete.
- **TTL**. Campo Tiempo de Vida (TTL) para determinar la presencia del dispositivo.

La clase Sala es la clase en la que se guardarán los datos de las salas creadas por parte del usuario. Por tanto, un usuario podrá crear como mínimo una sala para que funcione el sistema, o muchas más. Pero siguiendo la misma temática, una sala creada solo puede pertenecer a un usuario, es decir, el que la ha creado. Esta clase contiene los atributos:

- **nombre**. Nombre de la sala introducido por el usuario.
- **aforo**. Aforo de la sala, es decir, cantidad de personas que pueden ocupar dicha sala.
- **tamaño**. Tamaño de la sala, en la que el usuario especifica un tamaño, representado a través de la clase enum.

La clase Tamaño, es una clase explicativa, de tipo enum, en la que representamos un atributo que no sigue la terminilogía de los anteriores, es decir, es propio del usuario, ya que contiene una característica específica. Esta clase contiene 3 tipos de tamaño: pequeña, mediana y grande. En el apartado de implementación se explicará con más detalle este tipo de atributo.

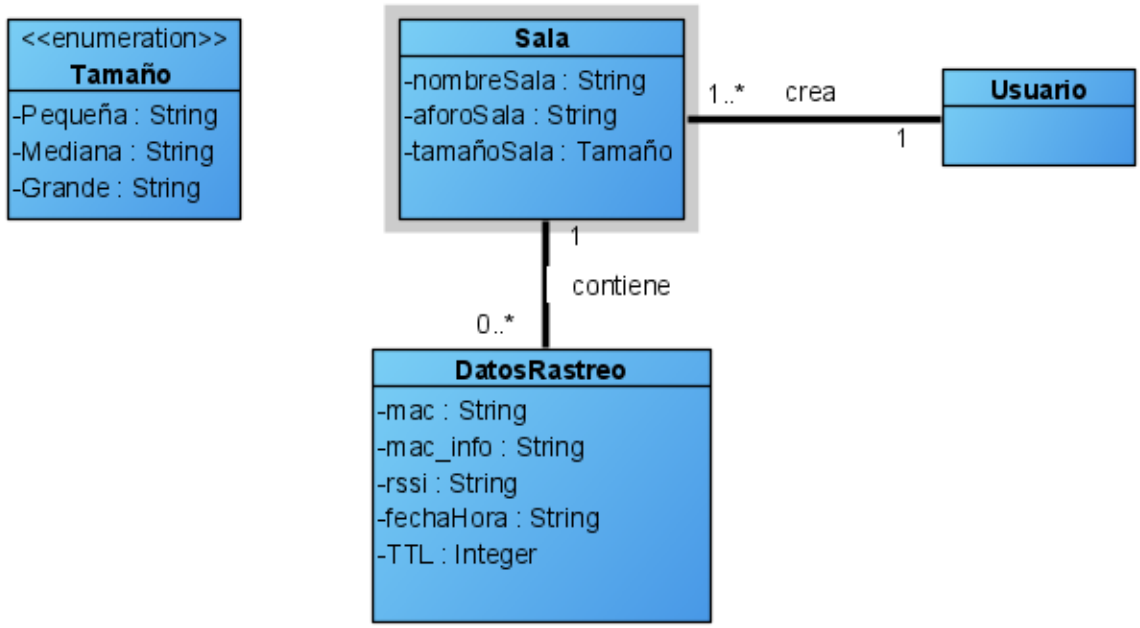

Figura 3.6: Modelo de dominio del sistema

3.3.4. Diseño del algoritmo del sistema

En esta sección describiremos de una forma más detallada el algoritmo que seguirá el sistema para realizar el objetivo principal del proyecto, hacer una estimación de las personas que se encuentran en un lugar específico. En esta sección podremos encontrar la hipótesis planteada de rastreo de paquetes, así como los dos algoritmos principales que utiliza el sistema.

Hipótesis de rastreo

La hipótesis que vamos a plantear es sencilla, y que podrá ser ampliada como veremos posteriormente en el capítulo 4. La hipótesis del sistema se basa en que, por cada dispositivo nuevo que encuentre el program rastreador, haremos la suposición de que el dispositivo encontrado es una persona. Es así, que en el escenario que estamos planteando, sería un escenario en el que cada persona lleve consigo sólo un dispositivo que sea capaz de realizar una conexión inalámbrica. Esta hipótesis no es una hipótesis compleja, pero es el primer paso para realizar el proyecto y poder lograr un prototipo bien asentado y que pueda servir de cara al futuro para proyectos que necesiten esta base.

Algoritmo de rastreo

Este es el algoritmo principal que seguirá el sistema. El primer paso que realiza es iniciar el rastreo. Si encuentra un paquete, la primera comprobacion es si es de tipo probe request. Si no cumple la característica, se descarta el paquete y se vuelve a iniciar el rastreo. Si es de tipo probe request, se extraen los datos necesarios del paquete y a continuación, se pasa a su análisis. El primer análisis se corresponde con el campo rssi, dependiendo de su valor, se le asignará un valor que corresponderá con el tamaño de la sala, pudiendo ser pequeña, mediana y grande.A continuación se analiza la dirección mac del paquete rastreado. Primero se realiza la comprobación de si se trata de una dirección mac aleatoria o no aleatoria, como ya se había explicado en el capítulo 2. Una vez determinado si es aleatoria o no, se comprueba si dicha direccion mac ya se encuentra en los registros de la base de datos. Si ésta ya está almacenada, se actualizan los campos correspondientes en la base de datos, pero si no está almacenada, y resulta que es una dirección nueva, se añade el registro a la base de datos. Una vez realizado esto, el último paso que realiza el program es guardar en el fichero log el paquete encontrado, para que en caso de que se produzcan errores, tener un registro de los paquetes rastreados. Cuando finaliza el trabajo del log, el programa vuelve a inicar el rastreo, continuando el proceso indefinidamente.

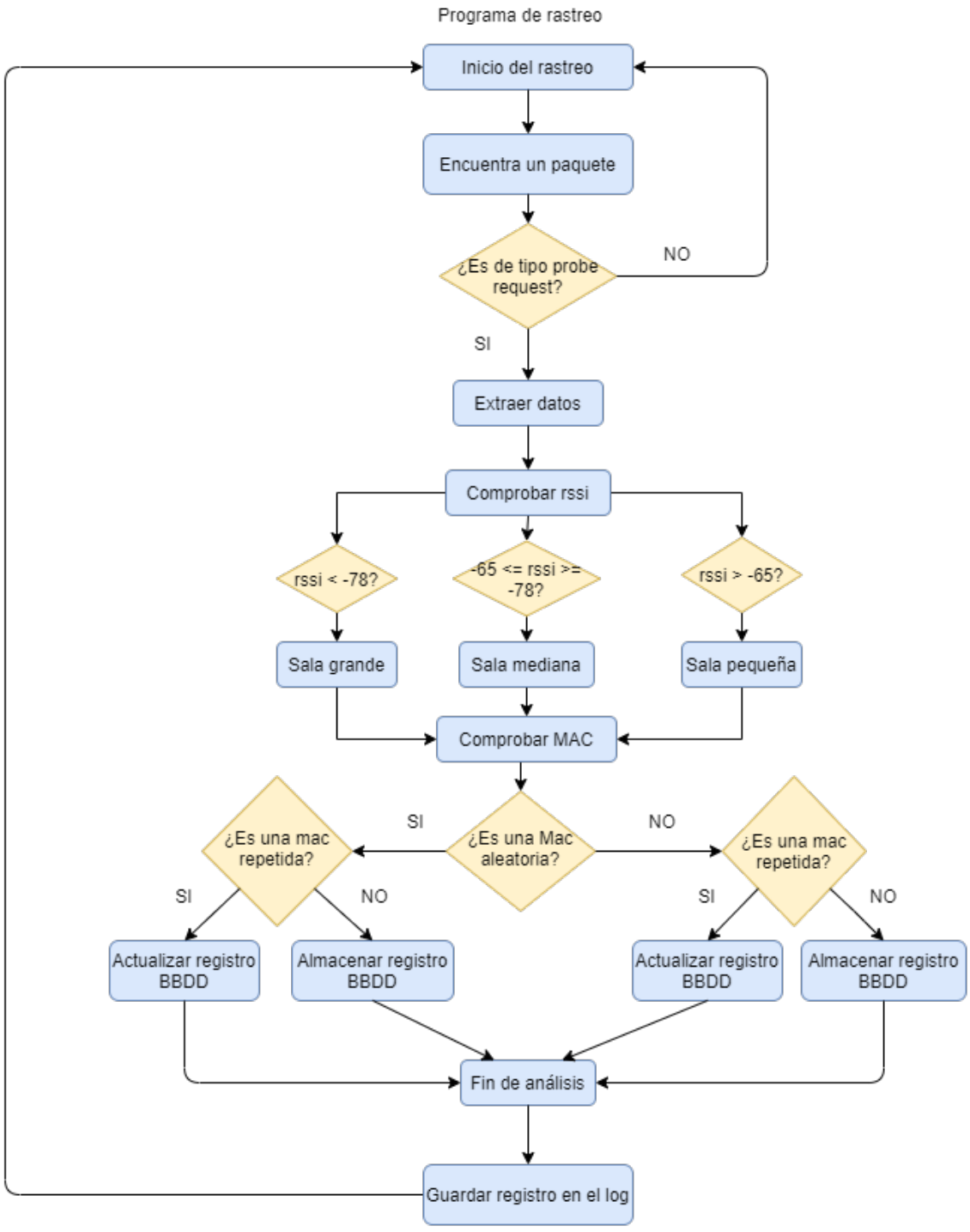

Figura 3.7: Algoritmo de rastreo del sistema

Algoritmo de control de presencia

Este algoritmo nos servirá para determinar si una persona permanece dentro de la sala o si se ha movido y ya no se encuentra presente en la misma. El primer paso que realiza es la espera de 1 minuto para que al programa principal le de tiempo de capturar dispositivos y almacenarlos. Una vez realizada la espera, se pasa a analizar el campo TTL que hemos establecido su valor en 5, coincidiendo así un control de presencia cada 5 min. Si el TTL no es 0, entonces el programa actualiza el registro de la base de datos del paquete y decrementa en uno su valor. Si el valor es 0, entonces el programa elimina todos los registros de la base de datos cuyo valor en el campo TTL sea 0. Una vez acabado este filtrado, el programa finaliza y se volvería al paso de esperar 1 minuto indefinidamente, es decir, un bucle infinito.

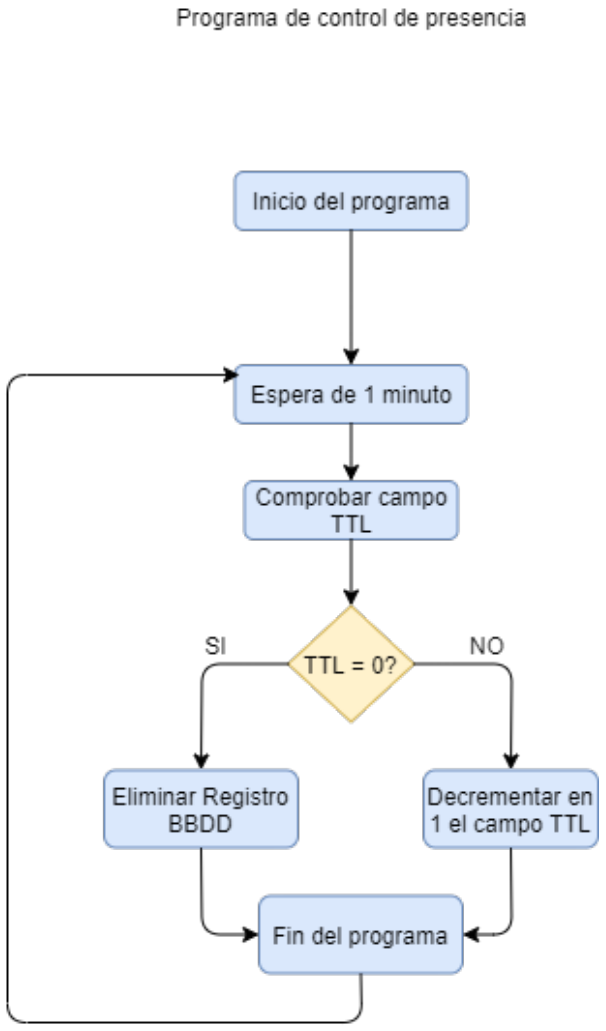

Figura 3.8: Algoritmo de control de presencia del sistema

3.3.5. Arquitectura del sistema

PAra poder hacer el diseño completamente correcto, debemos proponer una estructura válida que se ajuste al modelo de proyecto que estamos realizando. La arquitectura es una de las partes más importantes del diseño,es una representación abstracta de los componentes y del comportamiento de un sistema, definiendo de manera clara cual será la estructura que compondrá el sistema, cuales serán sus componentes físicos, que relación tendrán los componentes con el sistema, etc.

Existen actualmente 2 modelos de arquitectura: la arquitectura lógica y la arquitectura física, que explicaremos más en detalle separándolo en 2 secciones diferentes para realizar un mejor análisis.

Arquitectura Lógica

La arquitectura de sofware, también conocida como arquitectura lógica, es el diseño de más alto nivel de la estructura de un sistema. Consiste en un conjunto de patrones y abstracciones que proporcionan un escenario coherente y claro para interactuar con el código fuente del software. Esta arquitectura se diseña en base a los objetivos y restricciones impuestos por los responsables del proyecto al principio del mismo. Para poder proceder a diseñar la arquitectura lógica del proyecto, es necesario que se haya definido y explicado previamente la identificación inicial de los subsistemas, es decir, los casos de uso. Por ello, en el apartado de Análisis de este capítulo, se ha hecho una explicación concreta y detallada de los casos de uso, además del modelo de dominio que debe de seguir el sistema.

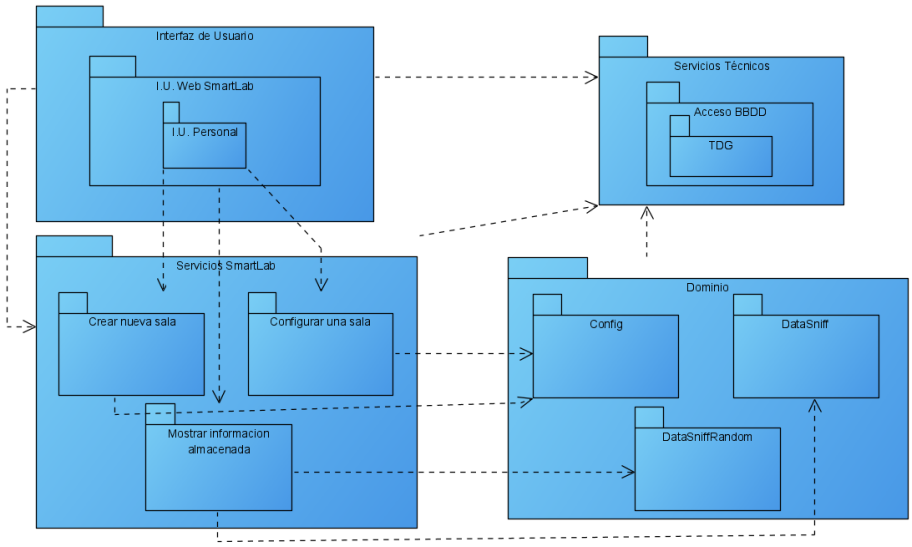

Figura 3.9: Arquitectura lógica del sistema

Como podemos observar en la imagen anterior, el modelo elegido para representar la arquitectura lógica del proyecto ha sido el patrón arquitectónico de capas. Este patrón sirve para el diseño de una aplicación compleja compuesta de un gran número de componentes a lo largo de múltiples niveles de abstracción.

Aplicando un orden descendente, podemos observar en la parte superior las capas más cercanas a nivel de usuario, y a partir de ahí, las capas que van adquiriendo mayor nivel de abstracción. Las diferentes capas que compone el sistema son 4:

- **Capa de Interfaz de Usuario**. En esta capa se encuentran todas las interfaces con las que el usuario se va a relacionar. Sirve para separar la capa de presentación de la capa de dominio.
- **Capa de Servicios**.En esta capa se ven reflejados los servicios que puede realizar el usuario a través de la interfaz, es decir, las funciones que podrá realizar el usuario en el sistema.
- **Capa de Dominio**. Esta capa sirve para representar que recursos son necesarios para implementar los servicios de la capa anterior. Es la capa donde se encuentra la lógica de dominio del sistema.
- **Capa de Servicios Técnicos**. En esta capa se encuentran todos los servicios técnicos que utilizará el sistema, como las bases de datos. El servicio TDG (Table Data Gateway) es un objeto que actúa como Gateway (patrón básico) a una tabla de BBDD. Es el componente que nos permitirá acceder a la base de datos para extraer o introducir la información necesaria.

Además, en la ilustración de la arquitectura lógica también podemos observar las diferentes dependencias que mantienen las capas entre ellas. La capa de Interfaz de usuario mantiene dependencias directas con los servicios que ofrece el sistema, ya que es con lo que se relacionará el usuario a la hora de probar el sistema. La capa de Servicios tiene dependencia con la capa del dominio, debido a que es necesaria la información que contiene la misma para que los servicios puedan ser desarrollados correctamente. Por otro lado, la capa de Dominio tiene una dependencia muy importante con la capa de Servicios técnicos, ya que esta capa contiene la información de manera física para que la capa de dominio pueda realizar su correcta interpretación. También podemos observar como todas las capas mantienen una dependencia con la capa de servicios técnicos al ser fundamental para cualquier tipo de proceso que se produzca.

Arquitectura física

La arquitectura física de software consiste en definir de una manera más abstracta los componentes que llevan a cabo las tareas del sistema, sus interfaces y la comunicación entre ellos. Ademñas de esta información, también es necesario determinar la máquina que llevará a cabo cada tarea. Para una mejor ilustración, en la siguiente imagen se mostrará un ejemplo de la arquitectura físca del sistema propuesto.

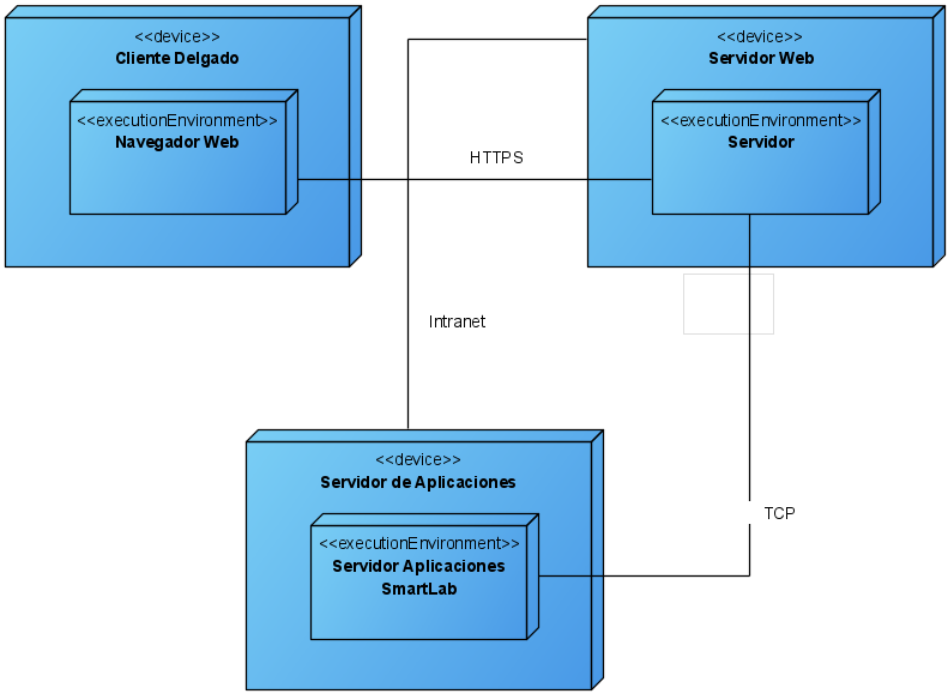

Figura 3.10: Arquitectura física del sistema

En la figura anterior podemos observar como el proyecto está separado en 3 componentes físicos: El usuario que utiliza el sistema, el servidor web del que obtiene la conexion a internet, y el servidor propio del sistema, el servidor de aplicación, donde se implementan los servicios que ofrece el sistema. Cada uno de estos elementos se relacionan entre ellos de la siguiente forma:

- El cliente inicia una nueva conexión a internet a través del navegador de su máquina, haciendo una petición de servicio al servidor que le provee los servicios de internet. Una vez establecida la conexión, el usuario introduce la dirección del servidor de aplicación del sistema, y el servidor propio, a través de una conexión tcp, proporciona el servicio al cliente para que se efectúe la conexión hacia el servidor de aplicación del sistema.

3.3.6. Diseño de la Base de Datos del sistema

Para poder realizar un correcto almacenamiento de los datos y un claro conocimiento de cómo serán tratados, es necesario realizar un diseño de la base de datos que utilizará el sistema. En primer lugar, el sistema usará la base de datos de Sqlite, anteriormente mencionada en el apartado de Tecnología utilizada. La base de datos del sistema estará compuesta por 3 tablas principales: DataSniff, DataSniffRandom y Config.

Figura 3.11: Esquema de base de datos del sistema

En la figura anterior podemos observar las diferentes tablas que compone nuestra base de datos. Cada una de ellas está compuesta por una clave primaria, para que no existan datos duplicados almacenados. En las tablas DataSniff y DatasniffRandom, se almacenan los datos provenientes de la operación de rastreo del sistema.

Antes de continuar, debemos hacer énfasis en un aspecto. Aun que las tablas tengan el mismo nombre de las filas, se almacena en cada una de ellas diferente información. En la tabla DataSniff, almacenamos los datos correspondientes de los paquetes que contienen en su dirección MAC, un campo OUI (Organization Unique Identification) conocido, es decir, que se encuentra registrado en la tabla de fabricantes IEEE, como ya se explicó en el apartado de Rastreo Wifi. Sin embargo, la tabla DataSniffRandom se corresponde con las direcciones MAC aleatorias que generan los dispositivos en diferentes ocasiones cuando quieren establecer una conexión con un punto de acceso.

Continuando con la explicación, podemos observar como la clave primaria de las 2 tablas mencionadas anteriormente, DataSniff y DataSniffRandom, se corresponde con la dirección mac, ya que ésta dirección es única para cada dispositivo rastreado y no cambia. En la tabla Config, se almacenan los datos que introduce el usuario a la hora de crear o configurar una sala. La clave primaria es el nombre de la sala, ya que éste debe ser único para cada una de las salas introducidas, es decir, para que no haya repetición de salas.

3.3.7. Diseño de la ejecución del sistema

Una vez explicado el diseño de los diferentes componentes que contendrá el sistema, en el siguiente diagrama de secuencia se hará una explicación de cómo se relacionarán esos componentes en el caso de uso genérico Crear sala.

En la figura que se mostrará a continuación, podemos observar como se realiza el caso de uso Çrear sala", para poder hacerlo previamente debe estar conectado al servidor de la aplicación web. Una vez dentro, el usuario introduce los datos de la sala y da al boton de guardar. La interfaz procesa la operación realizada por el usuario y manda una señal al controlador del sistema diciendo que acción se ha realizado. El controlador manda un mensaje a la capa de tratamiento de datos, el Table Data Gateway, indicando que se cree una sala nueva con los diferentes datos introducidos. El TDG, crea una sala con los datos, y manda un mensaje a la base de datos para que actualice su contenido. Una vez creado el registro de la sala, el controlador necesita los datos de la sala, por tanto manda un mensaje al TDG para recuperar dichos datos. El TDG recibe un mensaje, crea el objeto resultado de la consulta, y ejecuta la consulta en la BBDD. Una vez hecha la consulta, el controlador recoge el resultado del objeto creado por el TDG y posteriormente manda los datos a la capa de presentación para que ésta actualice la vista con los datos recibidos.

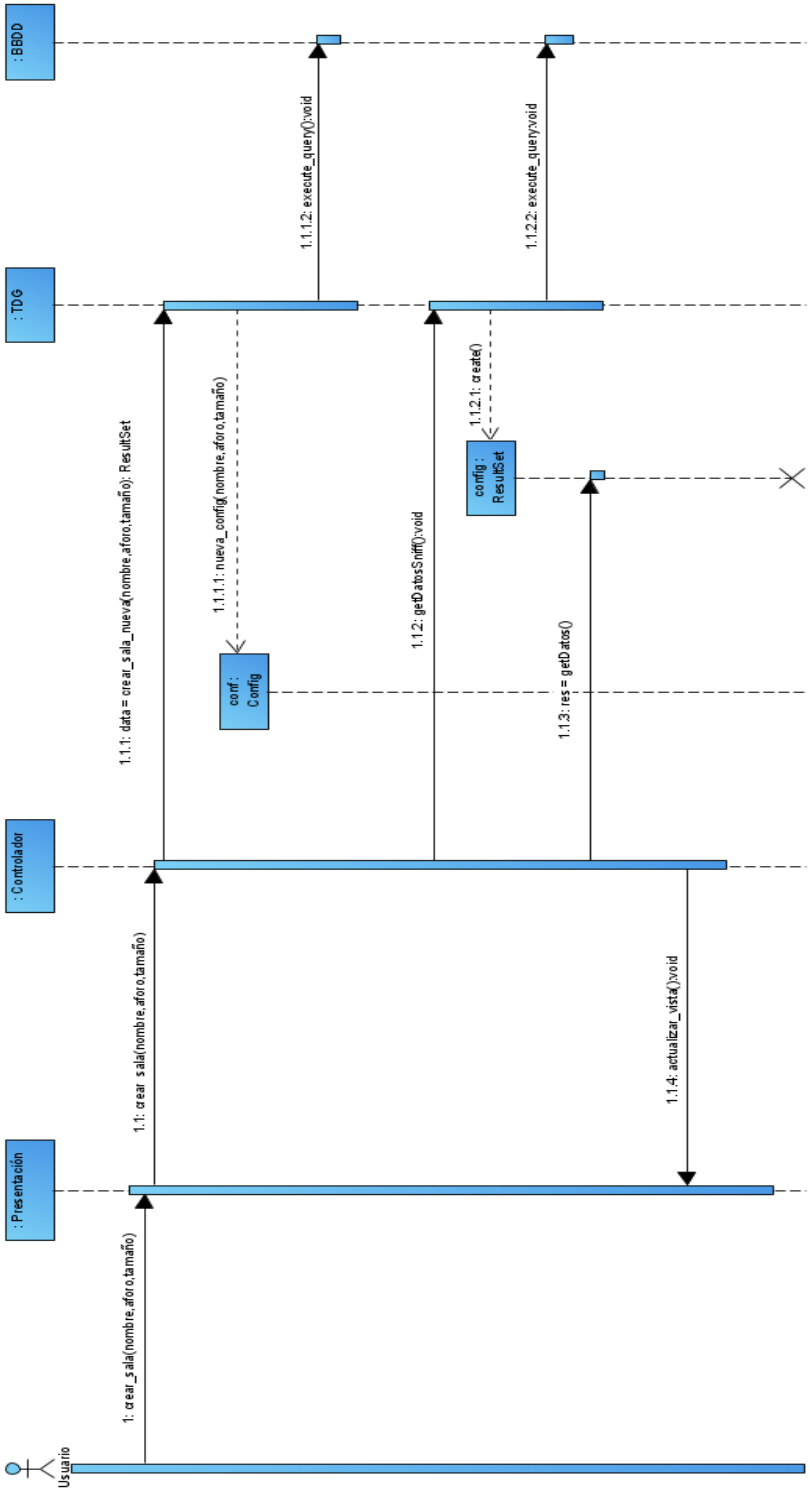

Figura 3.12: Diagrama de secuencia de Ejecución del caso de uso Crear Sala

3.3.8. Diseño de la interfaz web

Para completar el diseño del sistema, es necesario realizar un diseño de la interfaz web con la que interaccionrá el usuario y saber que información utilizará y que esquema de páginas seguirá.

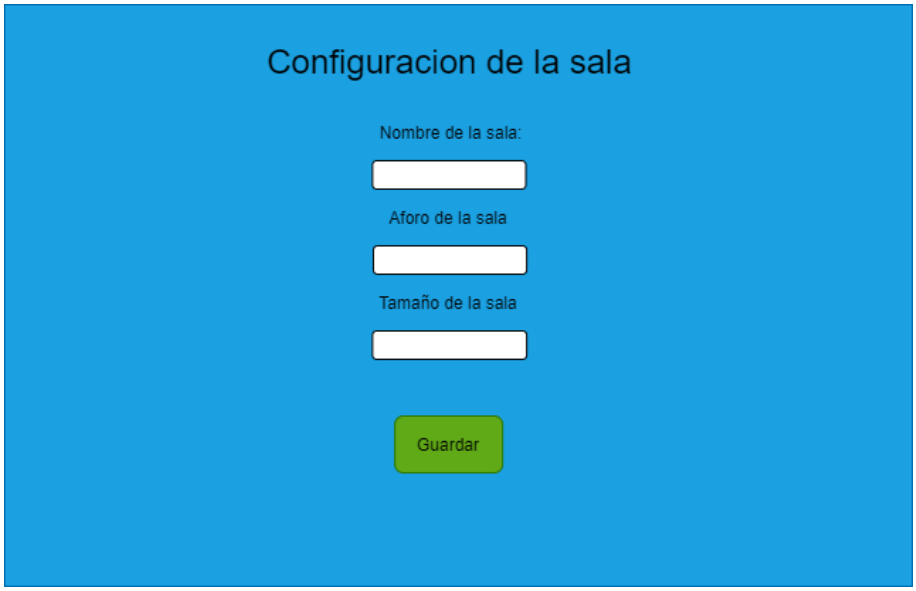

Figura 3.13: Página principal de acceso

En la imagen anterior, podemos observar el aspecto propuesto de la interfaz web, la página principal de acceso al introducir la dirección del servidor de aplicación del sistema. Esta vista incluye 3 campos en los que el usuario podrá introducir los datos: el nombre de la sala, su aforo, y el tamaño que tendrá la misma, pudiendo ser pequeña, median y grande. El botón guardar servirá para introducir los datos en la base de datos y redireccionará a una página de confirmación.

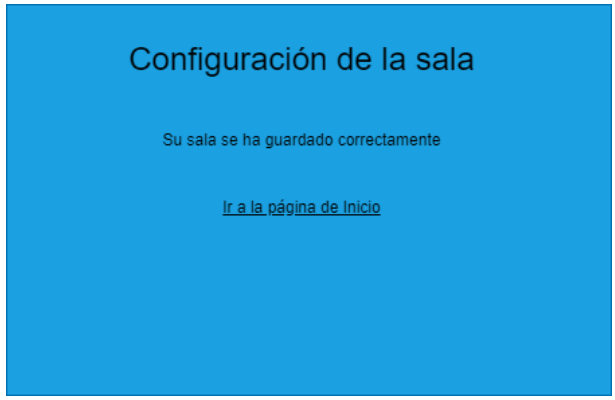

Figura 3.14: Página de confirmación de sala

Como podemos observar en la imagen mostrada, esta vista se corresponderá con el resultado de guardar una sala. Si el resultado es positivo, es decir, si se ha guardado correctamente, mostrará un mensaje diciendo que el resultado ha sido favorable, y haciendo click en el texto *Ir a la página de Inicio* nos llevará a la siguiente página. Por el contrario, si el resultado ha sido negativo, el mensaje mostrará que se ha producido un error y redireccionará a la página de crear una sala.

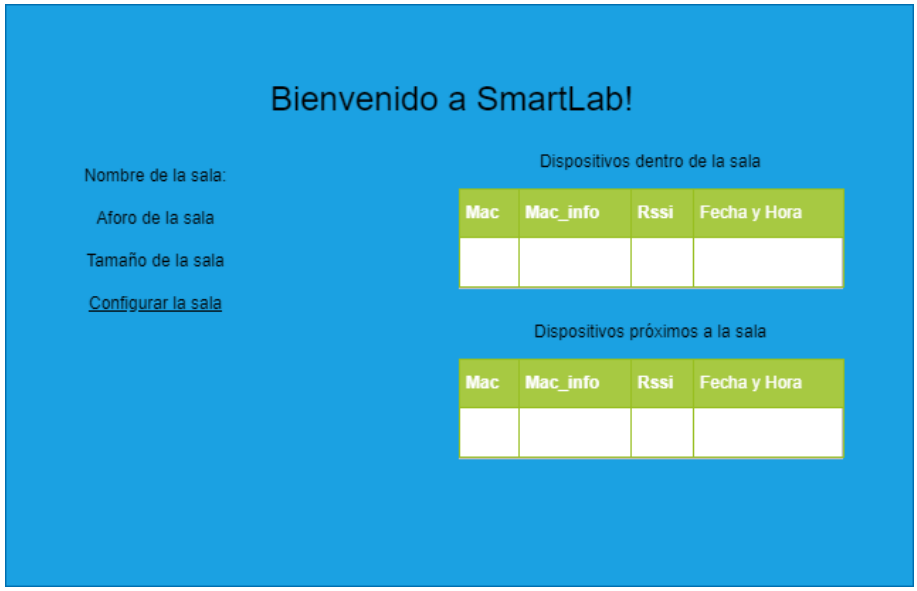

Figura 3.15: Página de inico de la interfaz web

En la página de inicio, podemos observar la vista de los diferentes elementos qu eha introducido el usuario, como el nombre, el aforo y el tamaño de la sala. Además, también se muestran en una tabla los dispositivos que se encuentran dentro de la sala, y cuáles están próximos a la misma, mostrando la dirección mac, el proveedor de chip de red (mac_info), la intensidad de señal recibida (rssi) y la fecha y la hora en la que se ha recibido el paquete para ambas tablas.
Por último, esta página cuenta con un enlace al que puedes acceder si se desea cambiar la información introducida acerca de la sala, redireccionando hacia la página de *Configurar una sala*.

3.4. Implementación

En esta sección se hablará de una forma más detallada los pasos que se han seguido a la hora de empezar a hacer el código del proyecto, junto con las herramientas utilizadas para el desarrollo del mismo.

3.4.1. Preparación del entorno de trabajo.

Para poder empezar a codificar, es necedsario una preparación del entorno de trabajo, descargar e instalar todos los elementos necesarios que permitan el desarrollo del proyecto. En primer lugar, el elemento más importante que se debe descargar es el software de Kali-Linux. Para poder utilizarlo, es necesario tostar la imagen del SO, es decir de Kali-Linux, en la tarjeta SD, y esto lo realizaremos a través del programa *Balena Etcher*.

Balena Etcher es una aplicación que puede se puede utilizar para tostar fácilmente imágenes del sistema operativo en tarjetas SD y unidades USB. Si el usuario no opta por recopilar los datos analíticos, la aplicación rastreará su uso. Además de los datos descritos anteriormente, también se recopila información sobre el tipo de tarjetas SD y unidades USB.

Una vez tenemos en la tarjeta SD la imagen del SO tostada, podemos proceder a la instalación de Kali-Linux en la raspberry Pi. Esta instalación la podemos realizar de dos formas: por teclado, o de manera gráfica. En este proyecto hemos optado por la manera gráfica, ya que era un método más cómodo para la instalación.

A continuación, finalizado el proceso de instalación, debemos descargar los módulos y programas necesarios para empezar a escribir el código. Kali-Linux ya contiene el lenguaje que utilizaremos para el desarrollo, Python, pero no incluye la versión más reciente, que será necesaria para realizar las funcionalidades que requiere el sistema. Por tanto, debemos descargar la versión más actual y realizar su instalación.

Descargado el programa principal de desarrollo, ahora debemos instalar los diferentes módulos que se utilizarán para realizar el proyecto, es decir, los móduos que Python no tenga instalados por defecto. Las librerías que utilizaremos serán las siguientes:

- **Scapy**. Será el módulo necesario para realizar el rastreo y filtrado de paquetes.
- **Netaddr**. Módulo mediante el cuál podremos distinguir si una dirección MAC es aleatoria o no.
- **Flask**. Librería que nos permitrá levantar un servidor y posteriormente utilizarlo para mostrar la información necesaria en él en un esquema de páginas web.
- **Sqlite3**. Módulo que se utilizará para poder realizar el almacenamiento d elos datos de rastreo, al igual que los datos introducidos por el usuario en un esquema de bases de datos relacional como es SQL.

Para poder realizar la instalación de dichos módulos, simplemente debemos ejecutar el siguiente comando:

```
pip3 install <módulo deseado>
```

Así, con este comando, python ya descargaba e instalaba en su correspondiente directorio para su correcto uso.

Destacar también que, se hará uso de otra librerías para poder completar la funcionalidad del sistema, pero éstas ya vienen instaladas por defecto en Python. Algunas de estas librerías son *time*, *Datetime*, ...

3.4.2. Codificación

A continuación, vistos los módulos necesarios para la escritura del código, se hará una breve explicación de las versiones necesarias, las herramientas utilizadas, así como los componentes que forman el sistema completo. Respecto a las versiones de las librerías utilizadas, son las siguientes:

- Kali GNU/Linux 2020.2
- Python 3.8.2 (última versión)
- Scapy 2.4.4 (última versión)
- Flask 1.1.2 (última versión)
- Sqlite3 3.31.1

El entorno de programación en el que se escribirá el código será el programa preinstalado que contiene Kali-Linux: Mousepad. Este programa es un editor de texto sencillo en el que se pueden realizar las tareas de escritura de una manera cómoda y sencilla, permitiendo la mayoría (si no son todos) de los atajos de teclado conocidos actualmente, que facilitan la labor de escritura. Debemos tener en cuenta que para escribir en este programa, python sigue una estructura de código muy estricta, es decir, el código de las funciones y los métodos de control, deben ir en un nivel inferior a la definición de las mismas.

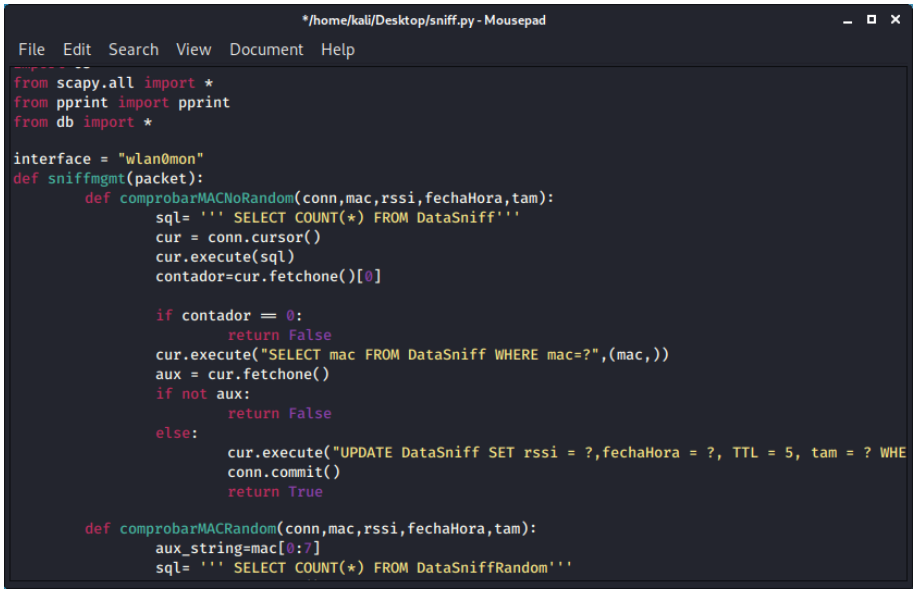

Figura 3.16: Fragmento de estructura básica de un programa del proyecto escrito en python

Los componentes que forman el sistema completo son varios, desde un script de ejecución básico, hasta un directorio completo donde se encuentran todos los ficheros necesarios para la correcta ejecución del servidor. Estos componentes son los siguientes:

- *scriptEjecucion*. Script básico en el que se ejecuta el programa de control de presencia en segundo plano y posteriormente el programa principal en una ejecución normal.
- *Sniff.py*. Programa principal en el que se realiza el rastreo de paquetes, su filtrado, y el almacenamiento y actualización de la información recogida de los mismos.
- *db.py*. Programa en el que están definidas todas las tareas de alamcenamiento, tales como la inserción de valores, la creación de una base de datos, la creación de tablas y la creación de una conexión a la base de datos.
- *tempo.py*. Programa secundario que se encarga del control de presencia de los dispositivos en la sala, previamente explicado en el apartado del diseño del algoritmo del sistema.

Ahora vamos a explicar todos los componentes que forman en conjunto la aplicación web. Dichos componentes son los siguientes:

- *server.py*. Programa python que permite el levantamiento del servidor web, además de recoger las tareas fundamentales que realizará el sistema en la interfaz web.
- *db.py*. Programa que nos permite la creación de las tablas en la base de datos, así como su conexión con la misma.
- *config.html*. Página html encargada de mostrar el formulario en el que el usuario introducirá los datos de una sala.
- *index.html*. Página principal en la que se mostrarán los datos almacenados correspondientes.
- *resultado.html*.Página de transición en la que muestra al usuario un mensaje de si los datos se han guardado o no.
- *style.css*. Fichero de estilo para todas las páginas web anteriores.

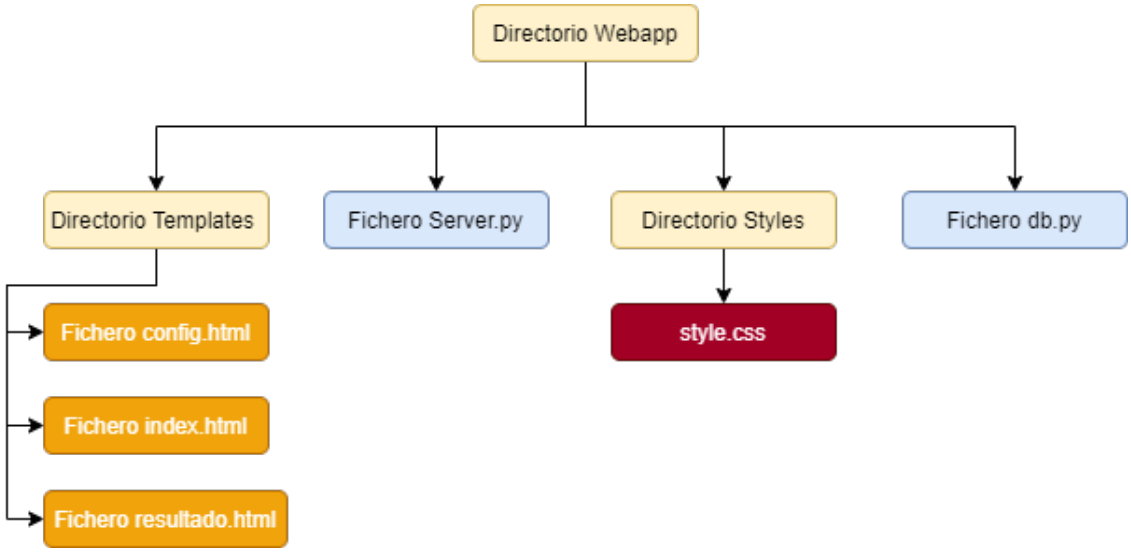

Figura 3.17: Esquema de directorios y archivos de la aplicación web

3.5. Pruebas

En este apartado se explicará de una manera breve cuales han sido los entornos en los que se han realizado las pruebas y cuando se han hecho las mismas.

Las pruebas se realizarán al final de cada tarea, para comprobar si la implementación de la misma es la correcta y si cumple con la funcionalidad esperada. Esta tarea la llevará a cabo el desarrollador del sistema, que es quien se encargrá de todo el procedimiento de las pruebas.

El primer paso es definir el entorno de ejecución, el cual en nuestro caso ha sido el terminal del propio Kali-Linux en vez de utilizar el iddle de python. Para poder realizar la ejecución del programa principal, es decir, del programa *sniff.py*, era necesario establecer la interfaz del terminal en modo monitor. Una vez realizado el proceso, simplemente debíamos ejecutar para cualquier programa que quisieramos probar el siguiente comando:

```
python3 <nombre-programa>.py
```

También destacar que, para probar el funcionamiento conjunto de los programas de rastreo y control de presencia, se ha realizado un script (scriptEjecucion) que al ejecutarlo, ejecutará ambos programas a la vez, para poder comprobar si el acoplamiento de los dos programas realizados funciona correctamente.

Por otro lado, en lo que refiere a la aplicación web, es necesario un terminal para el servidor, en el que se verán las solicitudes HTTP que manda los usuarios y si ocurre algún problema durante la transición de las páginas.

Por último, para comprobar el funcionamiento del programa completo, es necesario que el dispositivo hardware tenga una conexión a internet establecida por cable, ya que al establecer la interfaz de red en modo monitor, dicha interfaz se caerá y sólo realizará las acciones permitidas en este modo, y no permite la conexión a internet de manera inalámbrica. Una vez se ha establecido la conexión, basta con ejecutar en dos terminales independientes, el programa *server.py* y el script *scriptEjecucion*

Capítulo 4

Líneas futuras

4.1. Conclusiones

Las conclusiones que ha supuesto este proyecto han sido varias. Para empezar, he podido conocer de primera mano como es el desarrollo de software, los pasos necesarios a seguir, cómo se deben de analizar los datos, que métodos de diseño utilizar, cómo se deben de estructurar los componentes, etc.

Además del software, cabe destacar que al ser un proyecto de redes inalámbricas, he conocido la interacción en 1ª persona de los paquetes que envían los dispositivos entre sí, he comprendido su funcionamiento y su estructura interna, además de su identificación y análisis. Por otra parte, he practicado tareas propias de un administrador de bases de datos, ya que he realizado tareas como la gestión de la base de datos, el tratamiento de los datos almacenados, su conexión con las diferentes variables a lo largo del proyecto, cómo se relacionaban las tablas entre sí, etc.

También, he podido comprobar cómo es el proceso de planificación de un proyecto software. Las etapas que son necesarias, el orden de las mismas, las tareas que se van a desempeñar en el proyecto, cuáles con los componentes que forman el producto final, el estudio de la viabiliad del proyecto, cuánto supondrá en costes reales el desarrollo del mismo, entre otros aspectos.

Por otro lado, he sido capaz de montar una pequeña aplicación web, estudiar su funcionamiento, diseñar las diferentes interfaces que maneja, cambiar y configurar la transición de páginas de la misma, establecer un estilo fijo para cada una de las páginas que lo forman, etc.

Todos estos aspectos mostrados en el proyecto no habría sido capaz de acoplarlos sin los conocimientos que he adquirido durante mi estapa universitaria acerca de los campos de conocimiento mencionados anteriormente. Más concretamente, estos conocimientos me fueron enseñados en las asignaturas *Fundamentos de Ingeniería del Software*; *Diseño, implementación y adaptación de Software*; *Planificación y gestión de proyectos informáticos*; *Diseño, administración y seguridad de redes*; *Servicios y sistemas web*; *Adminisración de bases de datos*; y por último *Fundamentos de Redes de computadoras*.

Este proyecto ha sido una gran experiencia personal para mí, ya que he podido realizar todos los aspectos nombrados anteriormente por mi cuenta, siendo el principal responsable de todas las tareas que el proyecto suponía, así cómo de los errores provocados y los objetivos alcanzados. Además de los conocimientos ya adquiridos, he tenido que realizar un proceso de aprendizaje de herramientas que no conocía, al igual que programas y librerías, realizando las búsquedas necesarias por Internet y leyendo artículos cuyo contenido era relevante para el proyecto, cómo por ejemplo todos los módulos de Python utilizados.

Puedo afirmar que gracias al proyecto, mis conocimientos en el lenguaje de programación Python, han crecido significativamente, ya que era un lenguaje que no conocía muy bien y este proyecto me ha ayudado a entender su estructura y funcionamiento de una manera más clara y eficaz.

Centrándonos en el proyecto, gracias a la realización del mismo he podido conocer las ventajas que pueden suponer los diferentes tipos de sistemas de estimación de la ocupación que existen actualmente, en los que su fin sea facilitar acciones cotidianas e incluso situaciones de emergencia, mejorando la calidad de vida de las personas. Este proyecto estrá enfocado para cualquier usuario que quiera realizar una estimación de la ocupación de un lugar determinado, de una forma barata y sencilla de implementar.

4.2. Trabajo futuro

Este proyecto es un prototipo que puede servir como base de un sistema más complejo y completo que se centre en el mismo objetivo principal. Cómo implementaciones futuras y que no han podido ser incluídas en el proyecto debido a las consecuencias excepcionales en las que se ha realizado, podemos destacar las siguientes:

- Añadir junto con los paquetes wifi rastreados, la tecnología Bluetooth BLE (Bluetooth Low Energy), para poder realizar un rastreo de dispositivos más completo y eficaz, ampliando el abanico de información que podemos obtener.
- Realizar las pruebas necesarias en entrornos reales para comprobar la efectividad del sistema, ya que todas las pruebas que se han realizado en el proyecto han debido de ser desde mi domicilio particular, debido a la situación que ha golpeado a este año, la COVID-19, impidiendo así el desplazamiento hacia un área experimental más avanzada.

- Otra implementación importante que podría incluirse en el proyecto, sería añadir más funcionalidad en la interfaz web. Así, el usuario no sólo podría hacer tareas relacionadas con la creación de una sala, si no que también pueda reservar una sala para un fin determinado, cómo la realización de un proyecto o la impartición de una clase, almacenar las diferentes salas que puede tener un edificio, entre otras muchas impementaciones.
- Realizar tareas de análisis de información para determinar la cantidad de dispositivos que lleva una persona, sintetizando dicha información para que el escenario en que se realice el proyecto sea diferente y se pueda contemplar la opción de que un usuario lleve varios dispositivos consigo.
- Incorporar al proyecto un sistema de páginas más avanzado en el servidor de aplicación, para permitir que la funcionalidad que pueda realizar el usuario sea más sencilla y resulte un periodo de aprendizaje menor para el mismo, en caso de añadir más funcionalidades en la interfaz web.

Éstas son algunas de las implementaciones futuras que puede incorporar el proyecto para conseguir un sistema más complejo y que pueda ser utilizado de una manera más profesional, y no sólo con fines educativos.

Bibliografía

[1] *Edoardo Longo, Alessandro E.C. Redondi ,Matteo Cesana, Accurate occupancy estimation with WiFi and bluetooth/BLE packet capture.*
De *Dipartimento di Elettronica, Informazione e Bioingegneria Politecnico di Milano. 2019*

[2] *Julien Freudiger, How talkative is your mobile device?: an experimental study of Wi-Fi probe requests.*
Recuperado de *Proceedings of the 8th ACM Conference on Security Privacy in Wireless and Mobile Networks,June 2015, Article No.8 Pages 1–6* https://doi.org/10.1145/2766498.2766517

[3] *M. Gruber, A. Trüschel, J.-O. Dalenbäck, CO2 sensors for occupancy estimations: potential in building automation applications, Energy Build. 84 (2014) 548–556*

[4] *] D. Cali, P. Matthes, K. Huchtemann, R. Streblow, D. Müller, CO2 Based occupancy detection algorithm: experimental analysis and validation for office and residential buildings, Build. Environ. 86 (2015) 39–49.*

[5] *SEBASTIAN SADOWSKI, PETROS SPACHOS ,RSSI-Based Indoor Localization With the Internet of Things. Received April 11, 2018, accepted May 23, 2018, date of publication June 4, 2018, date of current version June 20, 2018*

[6] *I. Amin, A.J. Taylor, F. Junejo, A. Al-Habaibeh, R.M. Parkin, Automated people–counting by using low-resolution infrared and visual cameras, Measurement 41 (6) (2008) 589–599*

[7] *F. Wahl, M. Milenkovic, O. Amft, A distributed pir-based approach for estimating people count in office environments, in: Computational Science and Engineering (CSE), 2012 IEEE 15th International Conference on, IEEE, 2012, pp. 640–647*

[8] *S. Munir, R.S. Arora, C. Hesling, J. Li, J. Francis, C. Shelton, C. Martin, A. Rowe, M. Berges, Real-time fine grained occupancy estimation using depth sensors on ARM embedded platforms, in: Real-Time and Embedded Technology and Applications Symposium (RTAS), 2017 IEEE, IEEE, 2017, pp. 295–306.*

[9] *F. Buschmann , R. Meunier , H. Rohnert, P. Sommerlad , M. Stal , y M. Stal , Pattern Oriented Software Architecture Volume 1: A System of Patterns, Volume 1. Wiley, 1996. Capítulo 2*

[10] *C. Larman , Applying UML and Patterns: An Introduction to Object Oriented Analysis and Design and Iterative Development , 3rd Edition. Prentice Hall, 2004. Captulos 26, 33, 34 y 35*

[11] *M. Fowler, Patterns of Enterprise Application Architecture, 1.ª ed. Addison Wesley Professional, 2002. Capítulos 1, 2, 3, 4, 9, 10 y 14.*

[12] *E. Gamma et al. "Patrones de Diseño . Elementos de software orientado al objeto reutilizable ". Addison Wesley, 2002.*

[13] *Sommerville, I. Ingeniería del software Pearson, 2011 (9ª ed.)*

[14] *IEEE 802.11.*
Recuperado de https://es.wikipedia.org/wiki/IEEE_802.11

[15] *Qué es Wi-Fi 6 y qué ventajas tiene con respecto a la versión anterior.*
Recuperado de https://www.xataka.com/basics/que-wi-fi-6-que-ventajas-tiene-respecto-a-version-anterior

[16] *Nmap Reference Guide.*
Recuperado de https://nmap.org/book/man.htmlman-description

[17] *About Wireshark* https://www.wireshark.org/

[18] *Aircrak-ng*
Recuperado de https://www.aircrack-ng.org/

[19] *Airmon-ng*
Recuperado de https://www.aircrack-ng.org/doku.php?id=es:airmon-ng

[20] *John the ripper password craker*
Recuperado de https://www.openwall.com/john/

[21] *Arquitectura de software*
Recuperado de https://es.wikipedia.org/wiki/Arquitectura_de_software

[22] *About SQLite*
Recuperado de https://www.sqlite.org/about.html

[23] *Propiedad enum value.* Recuperado de http://library.gxtechnical.com/gxdlsp/pub/GeneXus/DevEnv/Docum/ReleaseNotes/8.0/PropiedadEnumValue.htm

[24] *Python*
Recuperado dehttps://www.python.org/

[25] *Sqlite tutorial*
Recuperado de https://www.sqlitetutorial.net/

[26] *Direccion MAC*
Recuperado de https://es.wikipedia.org/wiki/Direcci%C3%B3n_MAC

[27] *Transmission power, Range and RSSI*
Recuperado de https://support.kontakt.io/hc/en-gb/articles/201621521-Transmission-power-Range-and-RSSI

Anexo A

Configuraciones

En este apartado se mostrán las configuraciones y código necesario para la elaboración del proyecto.

A.1. Establecimiento del modo monitor

Para poder establecer el modo monitor en el terminal de Kali-Linux, es necesario realizar los siguientes pasos:

1 Abrir un terminal en el directorio donde se desea trabajar.

2 Hacer login con la cuenta de administrador introduciendo el comando *sudo su* en el terminal.

3 A continuación, iniciado sesión como adminstrador, introducir el comando *iwconfig* y elegir el nombre de la interfaz con la que se desea trabajar. Una vez tecleado, aparecerá algo similar a la imagen.

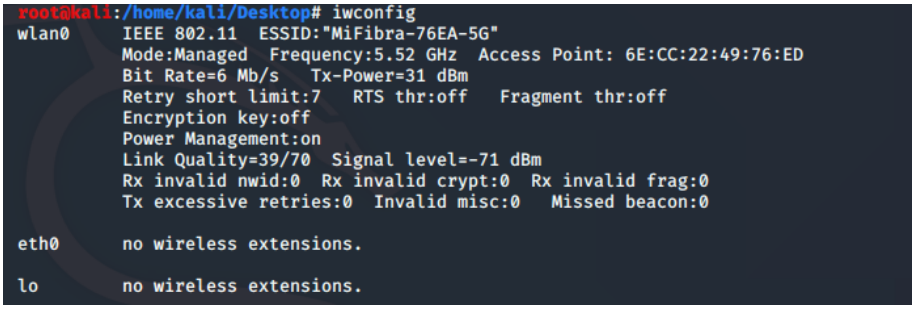

Figura A.1: Resultado de ejecutar el comando iwconfig

4 Una vez elegida la interfaz de red, basta con introducir *airmon-ng start (nombre-interfaz)* para establecer el modo monitor.

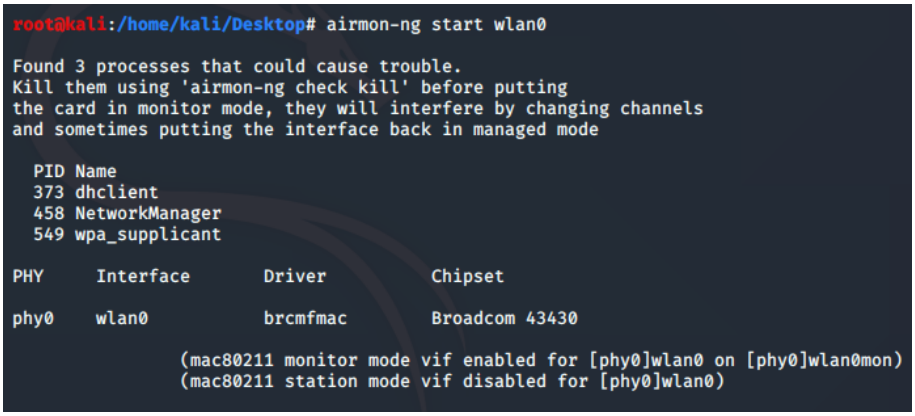

Figura A.2: Resultado de establecer la interfaz a modo monitor

Destacar que una vez establecido el modo monitor en la interfaz elegida, ésta interfaz pasará a llamarse (nombre-interfaz)mon. Por tanto, para parar el modo monitor, basta con ejecutar: *airmon-ng stop (nombre-interfaz)mon*

A.2. Script: Ejecución en segundo plano

```
#!/bin/bash
echo "Ejecutando programa 1 en segundo plano"
python3 tempo.py &
echo "Ejecutando programa principal"
python3 sniff.py
```

A.3. Inicio del servidor

Para poder utilizar la aplicación web, es necesario primero poner en marcha el servidor.

1 Abrir un terminal dentro del directorio donde se encuentra el programa del servidor, en nuestro caso el directorio es */home/kali/Desktop/webapp*

2 Una vez dentro del directoio, basta con ejecutar el comando *python3 server.py* para iniciar el programa

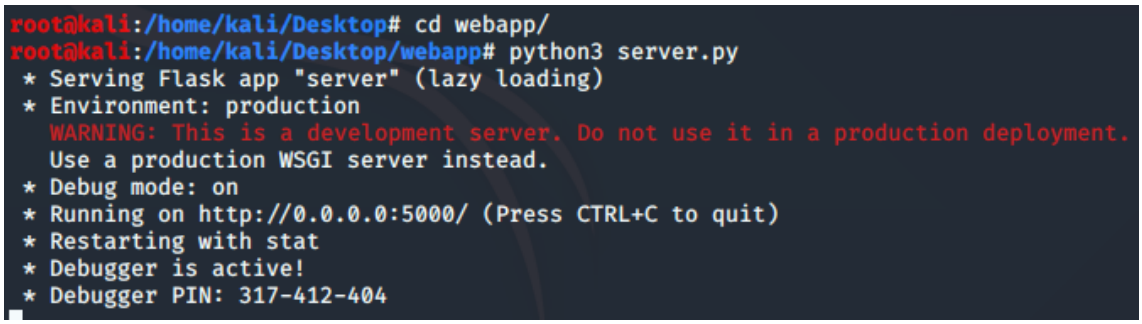

Figura A.3: Resultado de ejecutar el servidor

Anexo B

Manual de usuario

El usuario se relacionará directamente con la aplicación web creada. El objetivo principal de esta aplicación web es facilitar el uso del sistema por parte del usuario, además de la comprensión y la visualización de los datos que almacena el sistema para que el usuario esté pendiente en todo momento de qué operaciones realiza el mismo de una forma pasiva, es decir, sin que el usuario tenga constancia de ello. En esta aplicación web, el usuario podrá realizar los diferentes servicios:

- Creación de una nueva sala. Nada más inicializar el servidor, y establecer la conexión con el mismo, el usuario será capaz de crear una nueva sala sobre la que operará el sistema
- Configuración de la sala. El usuario será capaz de configurar una sala existente si lo desea, o si quiere cambiar algún parámetro en la configuración.

B.1. Vistas de la interfaz web

Figura B.1: Página principal de creacion de sala

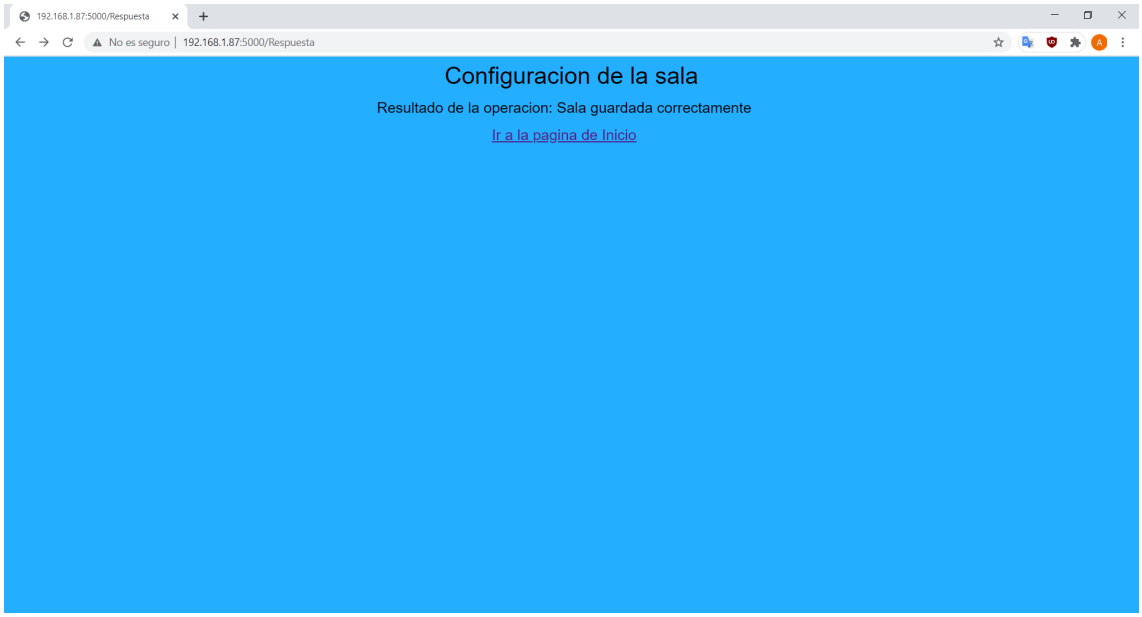

Figura B.2: Página de resultado de almacenamiento

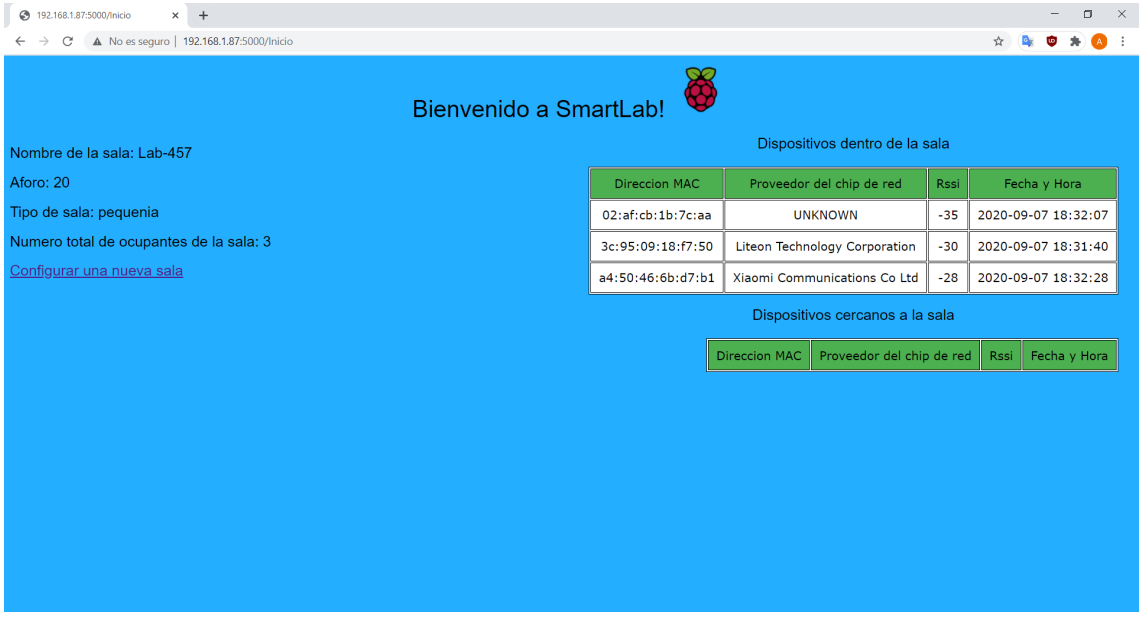

Figura B.3: Página de visualización de los datos

Anexo C

Manual de instalación

Para poder poner en marcha el sistema, son necesarios los siguientes pasos:

1 Dirigerse al repositorio de github *https://github.com/aitorojeda/TFG* y descargar todos los archivos necesarios del proyecto.

2 Descargar e instalar la imagen de Kali-Linux utilizada en el proyecto *kali-linux-2020.2b-rpi3-nexmon* desde la página oficial de descarga, adaptada al modelo de raspberry correspondiente: *https://www.offensive-security.com/kali-linux-arm-images/*.

3 Instalar Python3 con el comando *apt-get install python3*.

4 Instalar los modulos necesarios menicionados en el apartado de implementación.

5 Para poder iniciar la ejecución del sistema, el hardware utilizado debe tener establecida una conexión a internet de forma cableada, para que todos los componentes puedan funcionar a la vez.

6 Iniciar en primer lugar el servidor de la aplicación web.

7 Establecer la interfaz de red en modo monitor.

8 En un terminal separado, ejecutar el script *scriptEjecucion* para poner en marcha el sistema completo.

Anexo D

Contenidos del CD-ROM

El contenido del CD-ROM es el siguiente:

- Memoria. Se aportará una copia del documento en formato digital. A mayores de la información del documento, en los anexos, se aportará también el manual de usuario y el manual de instalación.
- Código fuente del proyecto. El directorio TFG contendrá todos los ficheros y documentos necesarios para la instalación y puesta en marcha del proyecto.
- Configuraciones. Se aportará un documento que contendrá las configuraciones necesarias para la realización del sistema. Estas configuraciones también están contenidas en el documento principal, en la sección de implementación, tanto como en los anexos.