



**ESCUELA SUPERIOR POLITÉCNICA AGROPECUARIA DE MANABÍ
MANUEL FÉLIX LÓPEZ**

DIRECCIÓN DE POSGRADO Y FORMACIÓN CONTINUA

**INFORME DE TRABAJO DE TITULACIÓN
PREVIA LA OBTENCIÓN DEL TÍTULO DE MAGÍSTER EN
TECNOLOGÍAS DE LA INFORMACIÓN MENCIÓN REDES Y
SISTEMAS DISTRIBUIDOS**

MODALIDAD: PROYECTO DE INVESTIGACIÓN Y DESARROLLO

TEMA:

APLICACIÓN MÓVIL DE GEOLOCALIZACIÓN DE REDES WLAN

AUTORA:

MARIANA KAROLINA PINARGOTE CUSME

TUTOR:

ING. ALFONSO TOMÁS LOOR VERA, MG.

CALCETA, MAYO 2019

DERECHOS DE AUTORÍA

MARIANA KAROLINA PINARGOTE CUSME, declaro bajo juramento que el trabajo aquí descrito es de mi autoría, que no ha sido previamente presentado para ningún grado o calificación profesional, y que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo los derechos de propiedad intelectual a la Escuela Superior Politécnica Agropecuaria de Manabí Manuel Félix López, según lo establecido por la Ley de Propiedad Intelectual y su Reglamento.

MARIANA KAROLINA PINARGOTE CUSME

CERTIFICACIÓN DE TUTOR

ING. ALFONSO TOMÁS LOOR VERA, MG., certifica haber tutelado el trabajo de titulación **APLICACIÓN MÓVIL DE GEOLOCALIZACIÓN DE REDES WLAN**, que ha sido desarrollada por **MARIANA KAROLINA PINARGOTE CUSME**, previo a la obtención del título de Magister en Tecnologías de la Información mención Redes y Sistemas distribuidos, de acuerdo al **REGLAMENTO PARA LA ELABORACIÓN DE TRABAJO DE TITULACIÓN DE LA UNIDAD DE TITULACIÓN ESPECIAL** de la Escuela Superior Politécnica Agropecuaria de Manabí Manuel Félix López.

ING. ALFONSO TOMÁS LOOR VERA, MG.

APROBACIÓN DEL TRIBUNAL

Los suscritos integrantes del tribunal correspondiente, declaramos que hemos **APROBADO** el trabajo de titulación **APLICACIÓN MÓVIL DE GEOLOCALIZACIÓN DE REDES WLAN**, que ha sido propuesto, desarrollado por **MARIANA KAROLINA PINARGOTE CUSME**, previa la obtención del título de Magister en Tecnologías de la Información mención Redes y Sistemas distribuidos, de acuerdo al **REGLAMENTO PARA LA ELABORACIÓN DE TRABAJO DE TITULACIÓN** de la Escuela Superior Politécnica Agropecuaria de Manabí Manuel Félix López.

DR. INF. JORGE PÁRRAGA ÁLAVA
MIEMBRO

DR. INF. JORGE HERRERA TAPIA
MIEMBRO

ING. JÉSSICA MORALES CARRILLO, MG.SC.
PRESIDENTA

AGRADECIMIENTO

A la Escuela Superior Politécnica Agropecuaria de Manabí Manuel Félix López, institución que abrió sus puertas a muchos jóvenes, gracias a ella hoy somos profesionales,

A nuestros coordinadores, Dr. Inf. Marlon Navia Mendoza y al Mg. Joffre Moreira Pico, excelentes docentes y seres humanos, quienes fueron nuestros guías para que pudiéramos alcanzar nuestros objetivos,

A los facilitadores, que nos impartieron sus conocimientos día a día,

A mi tutor, por compartir conmigo sus conocimientos y experiencia, que fueron mi guía oportuna durante todo el proceso académico. Gracias por apoyarme y ayudarme de manera incondicional y desinteresada.

La autora

DEDICATORIA

A mis padres por su apoyo incondicional y por los cuales lucharé día a día,
A mis sobrinos, que son el motor de mis proyectos, quienes con sus risas me hacen crecer y sentirme muy afortunada de tenerles conmigo,
A mis amigos Miriam, Johanna, Gema, Rosannita, Ramón, por no dejarme desmayar en este largo camino.

La autora

CONTENIDO GENERAL

PORTADA	i
DERECHOS DE AUTORÍA	ii
CERTIFICACIÓN DE TUTOR	iii
APROBACIÓN DEL TRIBUNAL	iv
AGRADECIMIENTO	v
DEDICATORIA	vi
CONTENIDO GENERAL	vii
CONTENIDO DE TABLAS	ix
CONTENIDO DE GRÁFICOS	ix
CONTENIDO DE FIGURAS	x
RESUMEN	xii
ABSTRACT	xiii
CAPÍTULO I. ANTECEDENTES	1
1.1. PLANTEAMIENTO Y FORMULACIÓN DEL PROBLEMA	1
1.2. JUSTIFICACIÓN	2
1.3. OBJETIVOS	3
1.3.1. OBJETIVO GENERAL	3
1.3.2. OBJETIVOS ESPECÍFICOS	3
CAPÍTULO II. REVISIÓN BIBLIOGRÁFICA	4
CAPÍTULO III. DESARROLLO METODOLÓGICO	13
CAPÍTULO IV. RESULTADOS Y DISCUSIÓN	17
RESULTADOS	17
ETAPA 1: DEFINICIÓN DEL PRODUCT BACKLOG	18
ETAPA 2: PLANIFICACIÓN DEL SPRINT	20
ETAPA 3: SCRUM DIARIO	25

ETAPA 4: REVISIÓN DEL SPRINT	28
ETAPA 5: RETROSPECTIVA DEL SPRINT	39
DISCUSIÓN	48
CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES.....	51
CONCLUSIONES	51
RECOMENDACIONES	52
BIBLIOGRAFÍA	53
ANEXOS	55
ANEXO 1.	56
ESPECIFICACIÓN DE REQUISITOS DE SOFTWARE SEGÚN ESTÁNDAR IEEE 830.....	56

CONTENIDO DE TABLAS

Tabla 2. 1. Aplicaciones móviles usando realidad aumentada mediante geolocalización.	4
Tabla 2. 2. Aplicaciones móviles con geolocalización en el área de la salud.....	7
Tabla 2. 3. Aplicaciones móviles con geolocalización en rutas y transporte.	9
Tabla 2. 4. Aplicaciones móviles con geolocalización con envío de notificaciones	11
Tabla 3. 1. Planificación del sprint 1.....	20
Tabla 3. 2. Planificación del sprint 2.....	22
Tabla 3. 3. Planificación del sprint 3.....	24

CONTENIDO DE GRÁFICOS

Gráfico 3. 1. Sprint1.....	27
Gráfico 3. 2. Sprint 2.....	27
Gráfico 3. 3. Sprint 3.....	28

CONTENIDO DE FIGURAS

Figura 3. 1. Arquitectura de la aplicación.	17
Figura 3. 2. Esquema de funcionamiento de la aplicación.....	18
Figura 3. 3. Product Backlog de la aplicación.....	19
Figura 3. 4. Sprint 1.....	21
Figura 3. 5. Sprint 2.....	23
Figura 3. 6. Sprint 3.....	25
Figura 3. 7. Tablero de tareas en Trello.	26
Figura 3. 8. Ícono y Splash de la aplicación.	29
Figura 3. 9. Pantalla principal de la aplicación.....	30
Figura 3. 10. Inicio de sesión con Facebook y registro.	30
Figura 3. 11. Método loginFb().	31
Figura 3. 12. Inicio de sesión con usuario y contraseña.	32
Figura 3. 13. Perfil de usuario.	32
Figura 3. 14. Redes WiFi disponibles.	34
Figura 3. 15. Conectar una red disponible.....	34
Figura 3. 16. Listado de redes compartidas por un usuario y visualización en el mapa.....	35
Figura 3. 17. Función loadMap().	36
Figura 3. 18. Función addMarker().	37
Figura 3. 19. Obtener listado de redes WiFi de un usuario.....	37
Figura 3. 20. Monitoreo de dos redes inalámbricas.	38
Figura 3. 21. Listado de las APIs.....	40
Figura 3. 22. API para agregar un usuario.	40
Figura 3. 23. Prueba de funcionamiento API para obtener usuario.	41
Figura 3. 24. Prueba de funcionamiento API para comprobar login.....	41
Figura 3. 25. Prueba de funcionamiento del API agregar un usuario.....	41
Figura 3. 26. Prueba de funcionamiento API para obtener listado de usuario.	41
Figura 3. 27. Prueba de funcionamiento API para actualizar usuario.	42
Figura 3. 28. Prueba de funcionamiento API para cambiar foto de perfil.	42
Figura 3. 29. Actualización de datos.	42
Figura 3. 30. Comprobación de actualización de información del usuario.	43

Figura 3. 31. API para agregar red.....	43
Figura 3. 32. Registro correcto al agregar una red.	44
Figura 3. 33. Prueba de funcionamiento API para listar redes compartidas. ...	44
Figura 3. 34. Prueba de funcionamiento API para obtener red.....	44
Figura 3. 35. Prueba de funcionamiento API obtener redes de un usuario.....	44
Figura 3. 36. Registrando datos del usuario.....	45
Figura 3. 37. Petición al usuario para uso de información.	46
Figura 3. 38. Login correcto al iniciar con Facebook.	46
Figura 3. 39. Red guardada con éxito.	47
Figura 3. 40. Petición: Obtener redes del usuario.	47
Figura 3. 41. Petición: Obtener redes compartidas.	48

RESUMEN

El presente trabajo de titulación tuvo como finalidad el desarrollo de una aplicación móvil para el análisis, evaluación y socialización de redes WiFi. El propósito de la aplicación es brindar a los usuarios de este tipo de aplicaciones móviles una amplia gama de características que, en la mayoría de los casos, están disponibles únicamente para los usuarios Premium. Para el desarrollo se utilizó la metodología SCRUM, por su naturaleza, agilidad, flexibilidad, eficiencia y fundamento empírico que permite un desarrollo ordenado a través de sus cinco fases bien definidas. Partiendo desde la visión general del producto se pudo estructurar el Product Backlog y posteriormente los Sprint Backlogs, cuyas tareas se fueron cumpliendo una a una hasta conseguir un incremento funcional potencialmente entregable. Con la aplicación de esta metodología se obtuvo un resultado de cero días de retraso y se pudo cumplir con todos los entregables que contempló el proyecto, consiguiendo una aplicación funcional y estructurada. Las pruebas unitarias y de integración evidenciaron una satisfactoria capacidad de respuesta de la aplicación. Con la aplicación móvil implementada se pudo visualizar en el mapa las redes WiFi compartidas en un rango de hasta 300 metros desde la ubicación del usuario, permitiendo obtener datos como nombre de la red y contraseña.

PALABRAS CLAVES

Geolocalización, redes WiFi.

ABSTRACT

The objective of the present work was to develop a mobile application for the analysis, evaluation and socialization of WiFi networks. The purpose of the application is to provide users of this type of mobile applications with a wide range of features that in most cases, are only available for Premium users. For the development of this study, SCRUM methodology was used due to its agility, flexibility, efficiency and empirical background, which allow for an ordered development through its five well-defined phases. Having a general view of the product it was possible to structure the Backlog product and later the Sprint Backlog, of which the tasks were done one by one until getting a functional deliverable increase. By the application of this methodology, a result of zero days of delay was obtained and all the deliverables contemplated in the project were carried out. Therefore, the achievement of a well-designed and structured application was possible. The unit and integration tests showed a satisfactory response capacity of the application. With the mobile application implemented, WiFi networks shared in a range of up to 300 meters from the user's location could be displayed on the map, allowing data such as network name and password to be obtained.

KEYWORDS

Geolocation, WiFi networks.

CAPÍTULO I. ANTECEDENTES

1.1. PLANTEAMIENTO Y FORMULACIÓN DEL PROBLEMA

Desde su nacimiento, las redes inalámbricas le dieron otro rumbo a la forma en que la sociedad se comunicaba haciendo uso del internet. Gracias a su flexibilidad y ventajas, estas redes han llegado a convertirse en una de las favoritas por parte de quienes hacen uso de la tecnología para comunicarse constantemente; además, debido a su rápido crecimiento las redes inalámbricas han contribuido con el surgimiento de otros productos, como los dispositivos móviles, y servicios que han aportado significativamente a la sociedad en la era de la información.

Pimentel, Tummala y McEachen (2015) aseveran que la rápida expansión de redes inalámbricas y dispositivos móviles, han llevado al vertiginoso desarrollo de aplicaciones móviles. En este contexto, Tartan y Ciflikli (2018) afirman que hoy en día en gran parte de la población, los teléfonos inteligentes y las aplicaciones móviles se consideran parte indispensable de la vida diaria. Es tanto así, que en el mundo muchos lugares públicos disponen de una red inalámbrica a la que se pueden conectar sus visitantes y personas ubicadas a su alrededor.

Ecuador no se ha quedado atrás, en Guayaquil hasta el 2018 existían más de 6000 puntos de acceso inalámbrico distribuidos de tal manera que los habitantes o visitantes puedan conectarse sin problemas. (GAD Municipal de Guayaquil, 2019). Esto para satisfacer la necesidad que demuestra hoy en día la sociedad de contar con una conexión que le permitan acceder a internet para hacer uso de las diversas aplicaciones que existen, ya sea para realizar alguna tarea específica, estudiar, trabajar o simplemente entretenerse.

Este escenario ha sido la pauta para que se desarrollen aplicaciones móviles que permitan dotar a los usuarios de diversas herramientas que les ayuden en diversos ámbitos de la vida cotidiana (salud, educación, negocios, entre otros).

Desde el éxito del popular juego de Nokia “La serpiente”, las aplicaciones móviles han evolucionado rápidamente y mejorado su esencia, hoy en día hay aplicaciones que pueden ser usadas casi para cualquier tarea que el ser humano realice, inclusive a la hora de hacer turismo o visitar lugares desconocidos. A partir del surgimiento de la computación móvil, la integración de la informática con el entorno del usuario ha cobrado cada vez más fuerza. (Vera, 2012)

La necesidad de conocer la ubicación de recursos y/o personas resulta imprescindible para poner en práctica soluciones de computación ubicua. (Vera, 2012), de acuerdo con esto, se plantea desarrollar una aplicación que permita conocer si existen redes abiertas disponibles tomando como referencia la ubicación del usuario. La mayor parte de las aplicaciones móviles basadas en geolocalización están destinadas a determinar la ubicación geográfica de atractivos turísticos y, por ende, no satisfacen la necesidad de proporcionar la ubicación geográfica de puntos de acceso inalámbrico situados en un área cercana al usuario. Según lo mencionado se plantea:

¿De qué manera se puede detectar redes inalámbricas disponibles y visualizar su información relevante?

1.2. JUSTIFICACIÓN

Hoy en día, el desarrollo de aplicaciones móviles se ha convertido en una de las actividades de desarrollo de software más cotizadas y perseguidas, puesto que los dispositivos móviles son parte fundamental en la rutina diaria de los humanos. Los usuarios utilizan el acceso a internet para el desarrollo de múltiples tareas y/o actividades personales, laborales, de negocio y de ocio, en fin, cualquier ámbito.

En la actualidad, aún existen usuarios de Smartphone (teléfono inteligente) que no disponen de un servicio de internet fijo, ya sea a través de un plan de datos proveído por alguna operadora o de un ISP (Internet Service Provider –

Proveedor de servicio de internet). Gran parte de estos usuarios se conectan únicamente haciendo uso de redes inalámbricas gratuitas que están disponibles en lugares públicos.

El presente trabajo, cuyo resultado es una aplicación móvil, garantiza a los usuarios de dispositivos móviles tener la posibilidad de visualizar un listado de redes inalámbricas disponibles cerca de su ubicación y acceder a cualquiera de ellas. Para que una red sea visible a otros usuarios esta debe ser compartida por su titular previamente en el mapa, de este modo se garantiza el derecho a la propiedad privada.

La aplicación, permitirá a los usuarios ser descargada de manera gratuita, lo que contribuye significativamente a la población, puesto que incluye características que en la mayoría de aplicaciones similares sólo están disponibles en una versión pagada. De esta manera, más usuarios podrán acceder y hacer uso de esta herramienta.

1.3. OBJETIVOS

1.3.1. OBJETIVO GENERAL

Desarrollar una aplicación móvil para visualizar la ubicación geográfica de las redes inalámbricas disponibles e información relevante de las mismas.

1.3.2. OBJETIVOS ESPECÍFICOS

- Determinar las historias de usuario a través del estándar IEEE 830.
- Diseñar la arquitectura, base de datos e interfaz de la aplicación móvil
- Ejecutar cada uno de los sprints de acuerdo a las prioridades.
- Efectuar pruebas de funcionamiento de la aplicación móvil.

CAPÍTULO II. REVISIÓN BIBLIOGRÁFICA

El presente trabajo tuvo como finalidad el desarrollo de una aplicación móvil que permita la detección y monitoreo de redes inalámbricas, además de poder compartir dichas redes con otros usuarios para que pueden conectarse con sus dispositivos móviles, todo esto en una misma versión gratuita. En este capítulo se presentan los trabajos de investigación relacionados, dentro del cual se evidencian las principales ventajas y desventajas, además de tecnologías utilizadas en el desarrollo de los mismos; sin embargo, es importante mencionar que no se encontró trabajo de similar característica al propuesto por la autora.

Para un mejor entendimiento se procedió a clasificar los trabajos de investigación en grupos, es decir, se los clasificó de acuerdo a su orientación o aplicación. En la tabla 2.1. se muestran aquellas investigaciones relacionadas a geolocalización con turismo y realidad aumentada. En la tabla 2.2. están investigaciones del área de la salud tanto humana como animal. En la tabla 2.3. lo referente a rutas y transporte. En la tabla 2.4. se muestran aquellas aplicaciones con envío de notificaciones.

Tabla 2. 1. Aplicaciones móviles usando realidad aumentada mediante geolocalización.

TEMA	AÑO DE PUBLICACIÓN	VENTAJAS	DESVENTAJAS
Desarrollo de una aplicación de geolocalización que facilite la ubicación de las dependencias en la Universidad Nacional de Loja con técnicas de realidad aumentada para dispositivos móviles.	2014	<ul style="list-style-type: none">▪ Brinda información en tiempo real de los bloques y dependencias a través de realidad aumentada y geolocalización de mapas.▪ Se utilizó un GPS GARMIN Map 60CSx para el levantamiento manual de información geográfica de los bloques y dependencias del campus universitario.▪ Enfoque de desarrollo nativo.	<ul style="list-style-type: none">▪ Se debe tener una conexión estable a internet.▪ Funciona correctamente cuando se está cerca de las instalaciones de la universidad.

(Gualotuña y Miranda, 2014)		<ul style="list-style-type: none"> ▪ Proporciona la ubicación mediante el uso de Google Maps y Wikitude (navegador de realidad aumentada). ▪ Permite la interacción con Facebook y Twitter. ▪ Posee un aplicativo web para la gestión de datos. ▪ Tecnología usada: Java, Wikitude, MySQL, servidor web Tomcat y librerías de Google Maps. ▪ Sistema operativo: Android versión 2.2 o superior. 	
<p>Mobile application to promote the Malecón 2000 tourism using augmented reality and geolocation. (Llerena, Andina y Grijalva, 2018)</p>	2018	<ul style="list-style-type: none"> ▪ Está dirigida a un lugar atractivo de la ciudad, como el Malecón 2000, y permite a visitantes locales, nacionales y extranjeros, a través de un dispositivo móvil y geolocalización, lograr la presentación de información digital con AR (Realidad Aumentada) del patrimonio cultural y las estructuras físicas que se encuentran en ese sector y que a su vez colaboran en la promoción turística de la ciudad. ▪ Tecnología usada: Android Studio, Vuforia SDK, Java, Visual Studio 2017, Google Maps, SketchUP 2018, Macromedia Fireworks 8. ▪ Sistema operativo: Android. 	<ul style="list-style-type: none"> ▪ Se debe tener una conexión estable a internet. ▪ Funciona en el Malecón 2000 de la ciudad de Guayaquil.
<p>UPV- MobARGuide Aplicación de Realidad Aumentada para</p>	2012	<ul style="list-style-type: none"> ▪ Ofrece una vista en realidad aumentada de la infraestructura de la universidad. 	<ul style="list-style-type: none"> ▪ Funciona solo en Android.

<p>guía interactiva de la UPV orientada a móviles. (Guillem, 2012)</p>		<ul style="list-style-type: none"> ▪ Dotar a los usuarios para que puedan crear contenido que pueda ser visualizado por la comunidad. ▪ Dispone de un servidor remoto. ▪ Tecnología usada: Android, MySQL y PHP, API de Google Maps, API de Wikitude. 	
<p>Desarrollo de una aplicación móvil para apoyar al turismo del centro histórico de Quito, utilizando realidad aumentada y geolocalización, para la empresa VLBS CIA.LTDA. (Vera, 2014)</p>	2014	<ul style="list-style-type: none"> ▪ La aplicación da al turista una referencia del siguiente punto de acuerdo a la posición que se encuentre en dicho momento. ▪ Tecnología usada: Unity, Vuforia SDK, framework iOS 5 de Apple. ▪ Sistema operativo: iOS y Android. 	<ul style="list-style-type: none"> ▪ La aplicación solo operará en el Centro Histórico de Quito. ▪ Solo muestra ubicación de 14 lugares. ▪ Se debe tener una conexión estable a internet. ▪ Se recomienda usar la aplicación en el día para que el software pueda reconocer las edificaciones.
<p>Aplicación móvil para mostrar sitios turísticos empleando realidad aumentada y geolocalización. (García, De la Rosa, Castillo y Cervantes, 2014)</p>	2014	<ul style="list-style-type: none"> ▪ La aplicación muestra realidad aumentada sobre determinadas áreas de la ciudad de Puebla. ▪ Tecnología usada: Java, Vuforia SDK, API de Google Maps, IDE Eclipse. ▪ Sistema operativo: Android. 	<ul style="list-style-type: none"> ▪ La aplicación solo operará en la ciudad de Puebla. ▪ El radio máximo de visualización es de 10 km.
<p>Integración de Foursquare y geolocalización en una aplicación móvil para la creación de rutas turísticas. (Sánchez, 2012)</p>	2012	<ul style="list-style-type: none"> ▪ Permite la creación de rutas turísticas. ▪ Permite añadir puntos a la ruta turística. ▪ Calcula la localización y los puntos cercanos al usuario. ▪ Admite descarga de rutas a través de un servidor Web. 	<ul style="list-style-type: none"> ▪ El usuario debe poseer una cuenta de Foursquare.

		<ul style="list-style-type: none"> ▪ Tecnología usada: Java, SQLite, API de Google Maps. ▪ Sistema operativo: Android. 	
<p>Aplicación de realidad aumentada para mostrar imágenes históricas de lugares turísticos de interés. (Fuster, 2015)</p>	2015	<ul style="list-style-type: none"> ▪ Muestra imágenes históricas del punto de interés turístico. ▪ Se enfoca con la cámara del dispositivo móvil y la aplicación proporciona imágenes del pasado. ▪ Muestra un listado con todos los puntos de interés, ordenado de acuerdo a la distancia del usuario. ▪ Tecnología usada: PhoneGap, Wikitude, jQuery Mobile, Sublime Text, HTML5, CSS3, ▪ Sistema operativo: Android versión 5 o superior. 	<ul style="list-style-type: none"> ▪ El dispositivo móvil debe tener conexión estable a internet, cámara y GPS. ▪ La distancia máxima de la cámara es de 150 metros.

Fuente: La autora.

De los trabajos de investigación abordados en la tabla 2.1., se puede observar que dos trabajos son de realidad aumentada por geolocalización para instituciones de educación superior, mientras que los restantes hacen referencia al turismo. Asimismo, se puede evidenciar que el sistema operativo más usado para el desarrollo de aplicaciones móviles es el Android.

Tabla 2. 2. Aplicaciones móviles con geolocalización en el área de la salud.

TEMA	AÑO DE PUBLICACIÓN	VENTAJAS	DESVENTAJAS
<ul style="list-style-type: none"> ▪ An Android application for geolocation based health monitoring, consultancy and alarm 	2018	<ul style="list-style-type: none"> ▪ Monitoreo de la frecuencia cardíaca basado en la geolocalización. ▪ La aplicación puede enviar mensajes de alarma a través de notificaciones, SMS, correo 	<ul style="list-style-type: none"> ▪ Se debe tener encendido Bluetooth en el dispositivo móvil y conexión estable de internet.

system. (Tartan y Ciflikli, 2018)		<p>y permite enviar mensajes con el experto en salud para consultas.</p> <ul style="list-style-type: none"> ▪ Uso del sensor de pulso plug-and-play de Arduino para medir la frecuencia cardiaca. ▪ Uso de HC-05 Bluetooth Shield para transmisión inalámbrica de datos del sensor al teléfono. ▪ Sistema operativo: Android. 	
Implementación de aplicación móvil para Android o iOS con realidad aumentada y geolocalización para asistencia y generación de citas en veterinarias del sur de Guayaquil sincronizado con gestor de contenido web publicitario. (Vernaza, 2015)	2015	<ul style="list-style-type: none"> ▪ Visualización de centros veterinarios más cercanos al usuario a través de un mapa. ▪ Tecnología usada: Eclipse, PhoneGap, SDK de Android ▪ Sistema operativo: iOS versión 4 o superior y Android 2.2 o superior. 	<ul style="list-style-type: none"> ▪ El dispositivo móvil debe tener conexión estable a internet, cámara y GPS. ▪ Funciona solo en la ciudad de Guayaquil.

Fuente: La autora.

En la tabla 2.2. se visualizan aplicaciones referentes al área de la salud, la primera relacionada al humano y la segunda al animal. La primera utiliza un sensor de pulso para medir la frecuencia cardiaca, misma que fue desarrollada para Android, mientras que la segunda usa realidad aumentada para localización de veterinarias en Guayaquil, funciona tanto en Android como iOS.

Tabla 2. 3. Aplicaciones móviles con geolocalización en rutas y transporte.

TEMA	AÑO DE PUBLICACIÓN	VENTAJAS	DESVENTAJAS
<p>Sistema de seguimiento mediante mapa online para los niños que viajan en transporte público. (Botella, 2018)</p>	2018	<ul style="list-style-type: none"> ▪ Monitoreo de transporte público a través de un mapa online. ▪ Tecnología usada: App Inventor, API de Google Maps, Google Fusion Tables y dispositivo Beacon. ▪ Sistema operativo: Android. 	<ul style="list-style-type: none"> ▪ Se debe tener encendido Bluetooth y GPS en el dispositivo móvil. ▪ El dispositivo Beacon tiene un alcance de 50 metros, puede trabajar hasta los 100 metros, pero presentan problemas.
<p>Automatización del proceso de ventas y distribución utilizando tecnología móvil y geolocalización para la empresa LÍDER SRL. (Ventura, 2014)</p>	2014	<ul style="list-style-type: none"> ▪ Se realiza la toma de pedidos por medio de la aplicación móvil tomando como referencia la ubicación geográfica exacta del cliente. ▪ La aplicación muestra la ruta más óptima para la entrega del pedido. ▪ Tecnología usada: C#.net, SQL Server 2005. ▪ Sistema operativo: Android. 	<ul style="list-style-type: none"> ▪ El dispositivo móvil debe tener conexión estable a internet y GPS.
<p>Desarrollo de una aplicación móvil Android para la búsqueda de plazas disponibles en un parqueadero. (Chinchay, 2015)</p>	2015	<ul style="list-style-type: none"> ▪ Facilita a los usuarios acceder a los parqueaderos, minimizando el tiempo de búsqueda de espacio disponible. ▪ Permite el acceso a los usuarios sin previo registro. ▪ Búsqueda de parqueadero cercano, desde la ubicación del usuario o alguna dirección alternativa. ▪ Tecnología usada: Eclipse ADT Bundle, Java, MySQL ▪ Sistema operativo: Android. 	<ul style="list-style-type: none"> ▪ El dispositivo móvil debe tener conexión estable a internet y GPS. ▪ Idioma español.

<p>Análisis y diseño de una aplicación móvil para la localización de rutas de transporte público. (Espinoza, 2015)</p>	<p>2015</p>	<ul style="list-style-type: none"> ▪ Permite al usuario mediante su posición GPS seleccionar un destino. ▪ Presenta de forma dinámica el recorrido del bus seleccionado de acuerdo al origen y destino del usuario. ▪ Tecnología usada: Android Studio 6.0, SQLite, ▪ Sistema operativo: Android versión 4.1.2 o superior. 	<ul style="list-style-type: none"> ▪ El dispositivo móvil debe tener conexión estable a internet y GPS. ▪ La aplicación funciona solo en la ciudad de Guayaquil.
<p>Aplicación Android para la búsqueda de precios económicos de combustible de E.E.S.S. (Jaén, 2017)</p>	<p>2017</p>	<ul style="list-style-type: none"> ▪ Permite buscar precios más económicos de diferentes tipos de combustible o la estación de servicio que cuente con el combustible deseado. ▪ Muestra las estaciones de servicio en un mapa. ▪ Tecnología usada: Android Studio, PHP, MySQL, servidor Apache. ▪ Cuenta con un proveedor de máquinas virtuales en la nube llamado DigitalOcean. ▪ Sistema operativo: Android versión 4.0.3 o superior. 	<ul style="list-style-type: none"> ▪ El dispositivo móvil debe tener conexión estable a internet y GPS.
<p>Aplicación Android: Red social de búsqueda de aparcamiento. (Company, 2015)</p>	<p>2015</p>	<ul style="list-style-type: none"> ▪ Permite a sus usuarios buscar la ubicación geográfica de espacios de aparcamiento libres, utilizando el sistema de geolocalización GPS y mapas de Google. ▪ Tecnología usada: Android Studio, Java, XML, JSON, PHP, API de Google Maps, MySQL. ▪ Sistema operativo: Android. 	<ul style="list-style-type: none"> ▪ El dispositivo móvil debe tener conexión estable a internet y GPS.

Fuente: La autora.

La tabla 2.3. muestra aplicaciones relacionadas con rutas y transporte, de las cuales hay dos referentes a búsqueda de aparcamiento; señalando que todas las aplicaciones fueron desarrolladas para que funcionen en el sistema operativo Android.

Tabla 2. 4. Aplicaciones móviles con geolocalización con envío de notificaciones

TEMA	AÑO DE PUBLICACIÓN	VENTAJAS	DESVENTAJAS
Aplicación de Android para geolocalización de tareas pendientes. Gestión e implementación de BD y aplicación móvil. (Carpintero, 2014)	2014	<ul style="list-style-type: none"> ▪ Permite personalizar la prioridad de las tareas pendientes, así como también las notificaciones. ▪ Sirve como recordatorio para la realización de tareas. ▪ Muestra la ruta más cercana. ▪ Tecnología usada: SQLite, PHP y RESTful, API de Google Maps y Google Task ▪ Sistema operativo: Android. 	<ul style="list-style-type: none"> ▪ Se debe tener encendido el GPS en el dispositivo móvil.
Geolocalización de centrales de impresión y fotocopiado en la Universidad Santiago de Chile. Una aplicación basada en Android. (Sepúlveda y Durán, 2014)	2014	<ul style="list-style-type: none"> ▪ Determina la central de impresión y fotocopiado más cercana a la ubicación del dispositivo a través de un mapa. ▪ Brinda la opción de calificar la central seleccionada. ▪ Permite que el usuario pueda realizar comentarios. ▪ Envía notificación si se crea una nueva central. ▪ Muestra el ranking de valoración. ▪ Tecnología usada: Android Studio, Sublime Text 3, Servidor Apache, MySQL, PHP, Java. ▪ Sistema operativo: Android versión 4.2 o superior. 	<ul style="list-style-type: none"> ▪ Se debe tener encendido el GPS en el dispositivo móvil y acceso a internet.

Fuente: La autora.

En la tabla 2.4. se detallan dos trabajos de investigación con envío de notificaciones, la primera que facilita la realización de tareas pendientes como la realización de compras, citas médicas, entre otras, funciona como recordatorio para la ejecución de las mismas, indicando de acuerdo a la posición del usuario el lugar o la ruta más cercana al lugar de destino. La segunda está basada en una central de impresión y fotocopiado en una universidad, dentro de la cual se puede realizar consultas sobre los servicios que ofrece. Ambas desarrolladas para Android.

Como se pudo observar en el contenido de las tablas anteriores, la mayor parte de las aplicaciones de geolocalización están enfocadas en ámbitos turísticos o en obtener ubicación de lugares específicos, algunas tienen limitado su funcionamiento dentro de una zona específica, otras requieren tener una cuenta paga o la distancia a la que trabajan es bastante limitada. AppGeo por su parte, está enfocada en brindarle a los usuarios la oportunidad de mantener una conexión a internet y compartir puntos de red inalámbrica con más personas sin tener que pagar y contando, además, con la función de monitoreo de red, que brindan aplicaciones de terceros, integrada. La aplicación es multiplataforma y con la configuración adecuada puede operar en varios sistemas operativos móviles y funciona en cualquier punto de ubicación en interiores y exteriores.

CAPÍTULO III. DESARROLLO METODOLÓGICO

Tras un análisis comparativo entre tres de las principales metodologías de desarrollo de aplicaciones móviles: Xtreme Programming (XP), Mobile-D y SCRUM, se escogió la última, ya que esta metodología permite realizar pruebas unitarias, pruebas de integración y pruebas de adaptación, que se ajustan a la planificación de desarrollo establecida por la autora.

Para el desarrollo de la aplicación móvil se utilizó el framework Ionic, que es una plataforma híbrida de fácil instalación, sin importar el sistema operativo en el que se vaya a desarrollar. Entre sus principales ventajas, Ionic cuenta con un importante conjunto de mejoras enfocado en el desarrollo de aplicaciones híbridas; además, este framework permite a los desarrolladores acceder y controlar los recursos del dispositivo móvil mediante el uso de librerías de Cordova. Otro aspecto relevante es que Ionic posee basta documentación de la mayor parte de sus componentes. (Wilken y Bradley, 2016)

La arquitectura de la aplicación se definió dentro de un modelo de dos capas: cliente y servidor. La data, por su parte, estará alojada en una Base de Datos de tipo relacional, diseñada y estructurada mediante el servicio Cloud SQL (Postgres Plus Cloud Database).

De acuerdo a la metodología SCRUM se planteó el desarrollo del trabajo de acuerdo a sus etapas:

- Definición del Product Backlog
- Planificación del sprint
- SCRUM diario
- Revisión del sprint
- Retrospectiva del sprint

Es importante señalar que con base a que el proyecto para el desarrollo de la aplicación móvil fue propuesto por la autora, los roles de SCRUM fueron asumidos de la siguiente manera:

1. Dueño del Producto: Karolina Pinargote (autora).
2. SCRUM Máster: Tomás Loor (tutor).
3. Equipo de Desarrollo: Karolina Pinargote.

ETAPA 1: DEFINICIÓN DEL PRODUCT BACKLOG

En esta primera etapa se definió el listado de producto o Product Backlog, en el que se incluyeron todas las especificaciones relevantes para el desarrollo de la aplicación, detalladas a manera de historias de usuarios, mismas que se obtuvieron a través del estándar IEEE 830 mediante la Especificación de Requisitos de Software, mediante la que se recolectó toda la información necesaria que sirvió como punto de partida para la delimitación de los requisitos técnicos y tecnológicos (ver Anexo 1).

Toda vez que se enlistaron las historias de usuario en la matriz del Product Backlog, previamente analizadas por el dueño del producto y el SCRUM Máster, se asignó prioridad a cada una de ellas, de acuerdo a la funcionalidad prevista de la aplicación; también se asignó el esfuerzo (en horas) para el desarrollo de las historias de usuario y, posteriormente, se definió el sprint al cual pertenecerían.

ETAPA 2: PLANIFICACIÓN DEL SPRINT

Para esta etapa, las dos personas asignadas a los roles de SCRUM mantuvieron reuniones en la que se determinó el propósito que perseguía el proyecto y los recursos necesarios y disponibles para el desarrollo. Con el listado de producto ya definido, se determinó en primera instancia las historias de usuario o actividades que se incluyeron en el sprint 1, tomando como punto de partida aquellas que se determinaron en función de desarrollador.

Considerando que las tareas referentes al montaje del ecosistema de trabajo no tomarían más de dos días, se incluyeron a este sprint las actividades correspondientes a la gestión de usuarios dentro de la aplicación. Una vez asignadas las tareas, se estableció el objetivo del sprint y, en función de ello, el valor que se entregaría como resultado de su ejecución; además, se definió la fecha de revisión del incremento.

Para los sprints 2 y 3, se desarrollaron las mismas actividades de planificación que se mencionaron anteriormente. A diferencia del primer sprint, en el que la información de entrada fue únicamente el listado de producto, para el segundo y tercer sprint se requirió, además de este listado actualizado, el producto desarrollado hasta la fecha en los incrementos anteriores.

Al finalizar cada sprint, el listado de producto se actualizó con el propósito de determinar las tareas completadas y pendientes, así como también los posibles nuevos requerimientos que pudiesen haber surgido y que tendrían que ser considerados en el diseño de la pila del sprint que se desarrollaría a continuación. Al finalizar la reunión de planificación para cada sprint, se obtuvo la pila o listado de actividades, el objetivo y la fecha de revisión del sprint.

ETAPA 3: SCRUM DIARIO

Debido a que el desarrollo de la aplicación estuvo a cargo de una sola persona y que las tareas no demandaron mayor complejidad, es esta etapa se omitió la realización de la reunión diaria que establece esta metodología para otros proyectos. Sin embargo, para llevar un control del progreso obtenido cada día, se elaboró un gráfico de avance o “burndown”, de tal manera que se pudiese observar y controlar el ritmo de desarrollo. Cada día se actualizó la pila del sprint de acuerdo con las tareas completas, en desarrollo y pendientes.

ETAPA 4: REVISIÓN DEL SPRINT

Esta etapa tuvo lugar cada vez que se concluyó el desarrollo de cada una de las tareas de un sprint. Estas reuniones no duraron más de una hora y en ella participaron los dos involucrados en el equipo SCRUM; durante la reunión se presentó el incremento del producto en estado completo y funcional y en virtud esta presentación se emitieron observaciones y/o sugerencias para mejorar el valor del producto final.

ETAPA 5: RETROSPECTIVA DEL SPRINT

En la reunión de retrospectiva participaron la desarrolladora y el SCRUM Máster. El propósito de esta fue determinar la manera en la que se desarrollaría el producto y establecer las pautas necesarias para garantizar el cumplimiento de los objetivos propuestos. En esta reunión se definió las herramientas y programas que se utilizarían: Visual Studio Code, Git CMD.

CAPÍTULO IV. RESULTADOS Y DISCUSIÓN

RESULTADOS

A continuación, se muestran los resultados obtenidos durante la ejecución de cada una de las etapas de la metodología aplicada.

Inicialmente se diagramó el esquema de la arquitectura con la que funcionaría la aplicación, misma que se muestra a continuación.

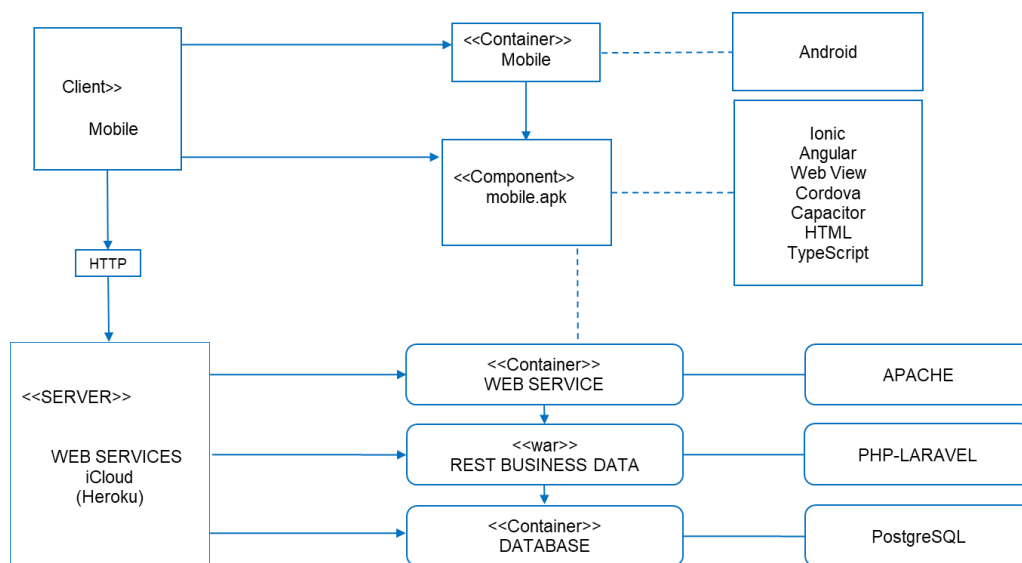


Figura 3. 1. Arquitectura de la aplicación.

Fuente: La autora

En la figura 3.1. se puede observar los dos niveles de los que consta la arquitectura de la aplicación, cliente y servidor. El cliente es un dispositivo móvil con sistema operativo Android. En esta capa, para la interfaz de usuario se utilizó Ionic, Angular, Web View, Cordova, Capacitor, HTML y TypeScript, el componente, en este caso el instalador, está disponible en extensión .apk. Desde la capa cliente, el usuario realizará peticiones HTTP al servidor iCloud que es Heroku; en el lado del servidor, además, se utilizó como ambiente de implantación a Apache para consumir los servicios web y una capa de negocio

estructurada con PHP y Laravel; también una base de datos diseñada en PostgreSQL, con el fin de aprovechar su integración con iCloud.

En la figura 3.2., se describe el esquema de funcionamiento de la interfaz de la aplicación.

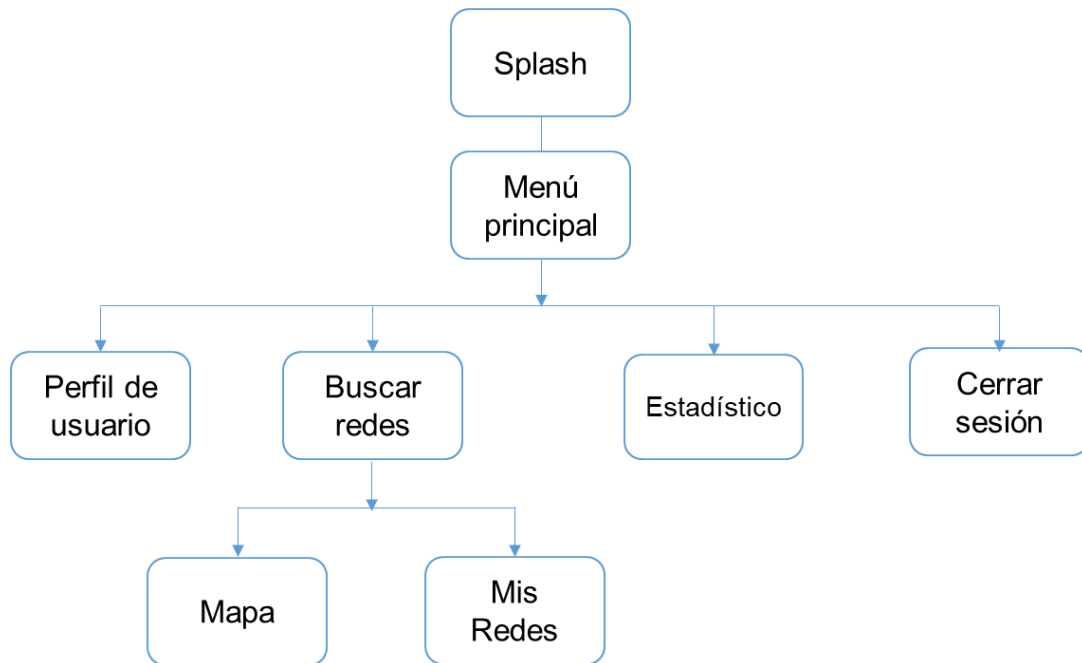


Figura 3. 2. Esquema de funcionamiento de la aplicación.

ETAPA 1: DEFINICIÓN DEL PRODUCT BACKLOG

Los requerimientos y pautas técnicas y tecnológicas para el desarrollo de la aplicación móvil se obtuvieron mediante la elaboración del documento de Especificación de Requisitos de Software (ERS) según el estándar IEEE 830 (Anexo 1 de este documento), describiendo cada uno de los apartados que sirvieron para recolectar los requerimientos de hardware, software (modelo entidad – relación, base de datos), herramientas y otros recursos necesarios para garantizar el óptimo desarrollo y funcionamiento de la aplicación.

Partiendo del ERS, se inició la estructuración del Product Backlog (figura 3.3.) enlistando las diferentes historias usuario para cada módulo. Finalmente, se

determinaron quince historias de usuario; a cada una de ellas se les asignó prioridad y tiempo estimado de ejecución, considerando que el tiempo diario máximo dedicado al desarrollo de la aplicación fue de aproximadamente cuatro horas diarias.

Identificador (ID) de la Historia	Alias	Enunciado de la Historia de Usuario	Estado	Dimensión / Esfuerzo	Prioridad / Importancia	Iteración (Sprint)	Comentarios
HU01	Montaje del ambiente de desarrollo	Como desarrollador necesito crear el ambiente de desarrollo adecuado, con la finalidad de contar con las herramientas necesarias	Realizada	16	Alta	1	...
HU02	Gestión de usuarios	Como usuario, necesito registrarme dentro de la aplicación, con la finalidad de obtener una cuenta de acceso	Realizada	4	Alta	1	...
HU03	Gestión de usuarios	Como usuario, necesito una cuenta de acceso, con la finalidad de acceder a las funcionalidades de la aplicación	Realizada	4	Alta	1	...
HU04	Gestión de usuarios	Como usuario, necesito acceder a la aplicación con la cuenta de Facebook, con la finalidad de probar sus funcionalidades	Realizada	8	Media	1	...
HU05	Gestión de usuarios	Como usuario, necesito tener mi perfil de usuario, con la finalidad de visualizar la información almacenada	Realizada	5	Media	1	...
HU06	Gestión de usuarios	Como usuario, necesito actualizar información personal, con la finalidad de mantener actualizada la cuenta	Realizada	4	Media	1	...
HU07	Gestión de redes	Como usuario, necesito escanear las redes WiFi disponibles, con la finalidad de visualizar las redes	Realizada	15	Alta	2	...
HU08	Gestión de redes	Como usuario, necesito conectarme a las redes disponibles, con la finalidad de tener acceso a internet	Realizada	8	Alta	2	...
HU09	Gestión de redes	Como usuario, necesito agregar redes WiFi ocultas, con la finalidad de conectarme a ellas	Realizada	8	Alta	2	...
HU10	Gestión de redes	Como usuario, necesito saber a qué red estoy conectado actualmente, con la finalidad de poder agregar la red al mapa	Realizada	5	Alta	2	...
HU11	Gestión de redes	Como usuario, necesito poder agregar la red al mapa, con la finalidad de tener disponible datos de la red en cualquier momento	Realizada	20	Alta	2	...
HU12	Gestión de redes	Como usuario, necesito visualizar las redes WiFi en el mapa, con la finalidad de observar la información de la red	Realizada	8	Alta	2	...
HU13	Gestión de redes	Como usuario, necesito dar clic en el marker del mapa, con la finalidad de visualizar la información disponible de la red	Realizada	6	Alta	2	...
HU14	Gestión de redes	Como usuario, necesito habilitar/eliminar redes, con la finalidad de tener solo redes disponibles	Realizada	3	Alta	2	...
HU15	Test	Como usuario, necesito poder generar/visualizar reporte estadístico de la red a la cual estoy conectado, con la finalidad de observar información en tiempo real.	Realizada	12	Alta	3	...

Figura 3. 3. Product Backlog de la aplicación.

ETAPA 2: PLANIFICACIÓN DEL SPRINT

Durante el desarrollo de la aplicación se efectuaron tres reuniones de planificación. En cada reunión, según correspondía, se definió el objetivo del sprint y se elaboró el Sprint Backlog.

SPRINT 1

Tabla 3. 1. Planificación del sprint 1.

PLANIFICACIÓN DE SPRINT 1	
Fecha	Participantes
17/12/2018	Karolina Pinargote Cusme Tomás Loor Vera
Lugar	Propósito
Instalaciones de la Unidad de Desarrollo Computacional de la Carrera de Computación de la ESPAM MFL.	Elaborar el Sprint Backlog para el primer sprint y definir un objetivo.
Entradas	Salidas
<ul style="list-style-type: none"> • Product Backlog • Recursos 	<ul style="list-style-type: none"> • Objetivo del sprint: Desarrollar funcionalidad para la administración de usuarios, con formulario de registro, opción de logueo con Facebook, actualizar y eliminar cuenta. • Sprint Backlog • Ecosistema
OBSERVACIONES:	

Fuente: La autora.

De la reunión de planificación se obtuvo el listado de las historias de usuario que conformarían el sprint 1, en primera instancia se consideraron aquellas historias referidas a la autenticación y gestión de usuarios, mismas que descompusieron en tareas y a las que se asignó tiempo estimado de desarrollo, adicional a estas tareas, en este sprint se consideraron las que correspondieron al montaje del ecosistema de trabajo (figura 3.4.). Para este primer sprint se estimó un tiempo de desarrollo de aproximadamente un mes.

Tareas	Estado	Dimensión / Esfuerzo	Tiempo dedicado	Prioridad	Persona encargada	Iteración (Sprint)	Comentarios
Montar Backend	Desarrollado	16		<i>Alta</i>	Karolina	#1	...
Montar Frontend	Desarrollado			<i>Alta</i>	Karolina	#1	...
Diseñar formulario de registro	Pendiente	4		<i>Alta</i>	Karolina	#1	...
Crear el API REST para registro de usuario	Pendiente			<i>Alta</i>	Karolina	#1	...
Programar la funcionalidad del registro de usuario consumiendo el API REST	Pendiente			<i>Alta</i>	Karolina	#1	...
Diseñar formulario de login	Pendiente	4		<i>Alta</i>	Karolina	#1	...
Crear el API REST para iniciar sesión a través de usuario y contraseña	Pendiente			<i>Alta</i>	Karolina	#1	...
Programar la funcionalidad de inicio de sesión consumiendo el API REST	Pendiente			<i>Alta</i>	Karolina	#1	...
Diseñar formulario de inicio de sesión	Pendiente	8		<i>Alta</i>	Karolina	#1	...
Programar el inicio de sesión, consumiendo el API de Facebook	Pendiente			<i>Alta</i>	Karolina	#1	...
Diseñar formulario de perfil de usuario	Pendiente	5		<i>Alta</i>	Karolina	#1	...
Crear el API REST para mostrar información del usuario	Pendiente			<i>Alta</i>	Karolina	#1	...
Realizar la respectiva visualización de información de usuario, utilizando el API REST	Pendiente			<i>Alta</i>	Karolina	#1	...
Crear el API REST para editar datos del usuario	Pendiente	4		<i>Alta</i>	Karolina	#1	...
Habilitar campos del usuario a editar	Pendiente			<i>Alta</i>	Karolina	#1	...
Realizar la respectiva modificación de datos utilizando el API REST	Pendiente			<i>Alta</i>	Karolina	#1	...

Figura 3. 4. Sprint 1.

SPRINT 2

Para la planificación de este segundo sprint se requirió tener el Product Backlog actualizado y el producto desarrollado hasta la fecha en el incremento que generó el sprint 1.

Tabla 3. 2. Planificación del sprint 2.

PLANIFICACIÓN DE SPRINT 2	
Fecha 14/01/2019	Participantes Karolina Pinargote Cusme Tomás Loor Vera
Lugar Instalaciones de la Unidad de Desarrollo Computacional de la Carrera de Computación de la ESPAM MFL.	Propósito Elaborar el Sprint Backlog para el segundo sprint y definir un objetivo.
Entradas	Salidas
<ul style="list-style-type: none"> • Product Backlog actualizado • Incremento generado en el sprint 1 	<ul style="list-style-type: none"> • Objetivo del sprint: Desarrollar funcionalidad para la administración de redes WiFi, con funciones de escanear, conectar, compartir, consultar y eliminar redes. • Sprint Backlog • Fecha para Revisión del sprint: 11/01/2019 • Incremento

Fuente: La autora.

Las historias de usuario que se incluyeron en el sprint 2 fueron las que correspondían a la gestión de redes WiFi las que, al igual que en el sprint anterior, se descompusieron en tareas (figura 3.5.); el tiempo que estimó para la ejecución de este sprint fue también de un mes; sin embargo, en desarrollo se llevó un lapso de cinco semanas.

Tareas	Estado	Dimensión / Esfuerzo	Tiempo dedicado	Prioridad	Persona encargada	Iteración (Sprint)	Comentarios
Diseñar el formulario de escaneo de redes WiFi	Pendiente	15		Alta	Karolina	#2	...
Utilizar librería hotspot.scanWifi() de Cordova	Pendiente			Alta	Karolina	#2	...
Diseñar la ventana para conectar la red WiFi	Pendiente	8		Alta	Karolina	#2	...
Utilizar librería hotspot.connectToWifi() de Cordova	Pendiente			Alta	Karolina	#2	...
Diseñar el formulario para agregar redes WiFi ocultas	Pendiente	8		Alta	Karolina	#2	...
Utilizar librería hotspot.addWifiNetwork() de cordova	Pendiente			Alta	Karolina	#2	...
Realizar el respectivo diseño	Pendiente	5		Alta	Karolina	#2	...
Utilizar librería hotspot.getConnectionInfo() de cordova	Pendiente			Alta	Karolina	#2	...
Crear el API REST para agregar la red al mapa	Pendiente	20		Alta	Karolina	#2	...
Diseñar el formulario para agregar la red al mapa	Pendiente			Alta	Karolina	#2	...
Programar la funcionalidad para agregar la red al mapa consumiendo el API REST	Pendiente			Alta	Karolina	#2	...
Crear el API REST para visualizar las redes agregadas en el mapa	Pendiente	8		Alta	Karolina	#2	...
Diseñar el formulario para visualizar las redes agregadas al mapa	Pendiente			Alta	Karolina	#2	...
Programar la funcionalidad para la visualización de las redes en el mapa consumiendo el API REST	Pendiente			Alta	Karolina	#2	...
Crear el API REST para visualizar información de las redes WiFi agregadas al mapa	Pendiente	6		Alta	Karolina	#2	...
Diseñar el formulario para visualizar información de las redes WiFi agregadas al mapa	Pendiente			Alta	Karolina	#2	...
Programar la funcionalidad para la visualización de información de las redes WiFi consumiendo el API REST	Pendiente			Alta	Karolina	#2	...
Crear el API REST para la deshabilitación/eliminación de redes agregadas al	Pendiente	3		Media	Karolina	#2	...
Diseñar el formulario para deshabilitar o eliminar las redes agregadas al mapa	Pendiente			Media	Karolina	#2	...
Programar la funcionalidad para la eliminación o deshabilitación de las redes consumiendo el API REST	Pendiente			Media	Karolina	#2	...

Figura 3. 5. Sprint 2.

SPRINT 3

Para la planificación de este tercer sprint se requirió tener el Product Backlog actualizado y el producto desarrollado hasta la fecha en el incremento que generó el sprint 2.

Tabla 3. 3. Planificación del sprint 3.

PLANIFICACIÓN DE SPRINT 3	
Fecha 18/02/2019	Participantes Karolina Pinargote Cusme Tomás Loor Vera
Lugar Instalaciones de la Unidad de Desarrollo Computacional de la Carrera de Computación de la ESPAM MFL.	Propósito Elaborar el Sprint Backlog para el segundo Sprint y definir un objetivo.
Entradas	Salidas
<ul style="list-style-type: none"> • Product Backlog actualizado • Incremento generado en el sprint 2 	<ul style="list-style-type: none"> • Objetivo del sprint: Desarrollar funcionalidad para monitorear la red a la que se ha conectado. Se debe mostrar datos relevantes que permitan la toma de decisión. • Sprint Backlog • Fecha para Revisión del sprint: 23/02/2019 • Incremento

Fuente: La autora.

El sprint 3 se estructuró con las historias de usuario referentes al monitoreo de las redes WiFi, funcionalidad que permite al usuario valorar la red a que se ha conectado para tomar decisiones respecto al rendimiento de las misma. Las tareas que se establecieron para esta historia de usuario fueron el diseño del formulario y la integración de la librería chart.js con el método getConnectionInfo() para generar el test (figura 3.6.). El sprint se planificó para una semana.

Tareas	Estado	Dimensión / Esfuerzo	Tiempo dedicado	Prioridad	Persona encargada	Iteración (Sprint)	Comentarios
Diseñar el formulario para la generación del reporte	Pendiente	12		<i>Media</i>	Karolina	#3	...
Integrar librería chart.js y hotspot.getConnectionInfo() para generar el reporte estadístico de conectividad	Pendiente			<i>Media</i>	Karolina	#3	...

Figura 3. 6. Sprint 3.

ETAPA 3: SCRUM DIARIO

Este evento es equivalente al desarrollo de las actividades y tareas planificadas para cada sprint. Diariamente se evaluaba el porcentaje o estimación del avance, las dificultades o novedades halladas se determinaba, mediante el uso de un gráfico burndown, el cumplimiento de las tareas correspondientes a un determinado sprint dentro del tiempo establecido. Para esta etapa se hizo uso del software gratuito de administración de proyectos Trello, para organizar las tareas y garantizar dinamismo y objetividad en la gestión de las mismas.

En la figura 3.7. se evidencia el tablero de actividades o tareas elaborado en Trello, donde las tarjetas que corresponden los sprints son de color azul y las de las tareas de color amarillo; se utilizó la letra “S” para identificar los sprints y la “T” para las tareas.

SCRUM DIARIO ☆ App Redes Free Visible para el equipo ML Invitar

TAREAS PENDIENTES	EN DESARROLLO	COMPLETO
S1: GESTIÓN USUARIOS	S1.T14: Realizar la respectiva modificación de datos utilizando el API REST	S1.T1: Diseñar formulario de registro
S2: GESTIÓN DE REDES	+ Añada otra tarjeta	S1.T2: Crear el API REST para registro de usuario
S2.T1: Diseñar el formulario de escaneo de redes WiFi		S1.T3: Programar la funcionalidad del registro de usuario consumiendo el API REST
S2.T2: Utilizar librería hotspot.scanWifi() de Cordova		S1.T4: Diseñar formulario de login
S2.T3: Diseñar la ventana para conectar la red WiFi		S1.T5: Crear el API REST para iniciar sesión a través de usuario y contraseña
S2.T4: Utilizar librería hotspot.connectToWifi() de Cordova		S1.T6: Programar la funcionalidad de inicio de sesión consumiendo el API REST
S2.T5: Diseñar el formulario para agregar redes WiFi ocultas		S1.T7: Diseñar formulario de inicio de sesión
S2.T6: Utilizar librería hotspot.addWifiNetwork() de cordova		S1.T8: Programar el inicio de sesión, consumiendo el API de Facebook
S2.T7: Realizar el respectivo diseño		S1.T9: Diseñar formulario de perfil de usuario
S2.T8: Utilizar librería hotspot.getConnectionInfo() de cordova		S1.T10: Crear el API REST para mostrar información del usuario
S2.T9: Crear el API REST para agregar la red al mapa		S1.T11: Realizar la respectiva visualización de información de usuario, utilizando el API REST
S2.T10: Diseñar el formulario para agregar la red al mapa		S1.T12: Crear el API REST para editar datos del usuario
S2.T11: Programar la funcionalidad para agregar la red al mapa consumiendo el API REST		S1.T13: Habilitar campos del usuario a editar
S2.T12: Crear el API REST para visualizar las redes agregadas en el mapa		+ Añada otra tarjeta
S2.T13: Diseñar el formulario para visualizar las redes agregadas al mapa		
S2.T14: Programar la funcionalidad para la visualización de las redes en el mapa consumiendo el API REST		
S2.T15: Crear el API REST para visualizar información de las redes WiFi agregadas al mapa		
S2.T16: Diseñar el formulario para visualizar información de las redes WiFi agregadas al mapa		
+ Añada otra tarjeta		

Figura 3. 7. Tablero de tareas en Trello.

En los tres sprint se mantuvo un ritmo que permitió el desarrollo de cada tarea muy de cerca en los tiempos estimados para el fin. El cumplimiento de cada tarea se plasmó diariamente en el gráfico de burndown, tal como lo muestran los gráficos 3.1., 3.2. y 3.3.

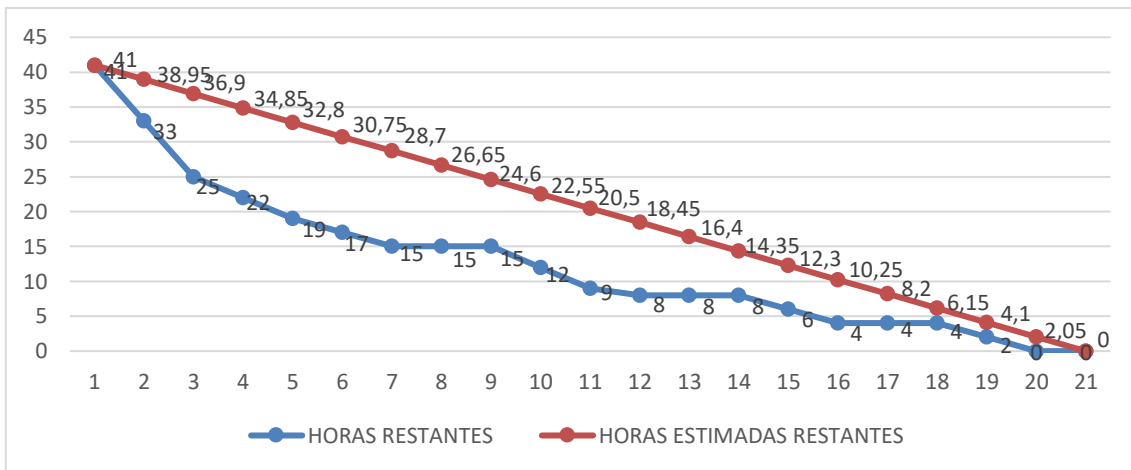


Gráfico 3. 1. Sprint1.

Fuente: La autora.

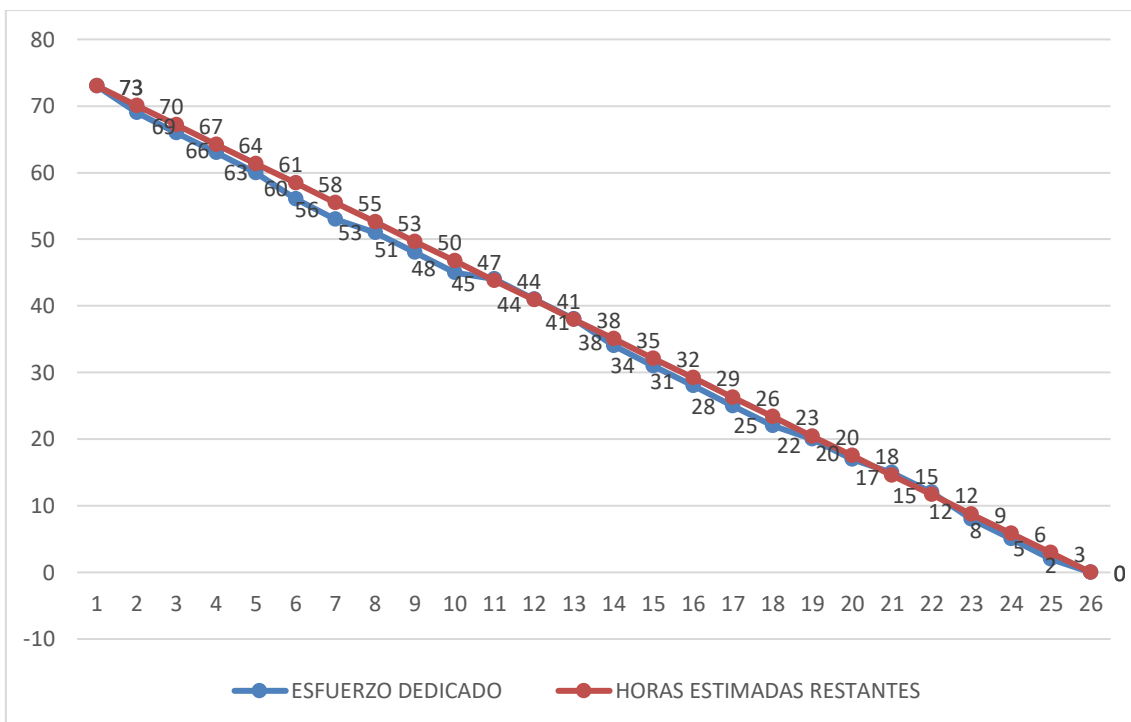


Gráfico 3. 2. Sprint 2.

Fuente: La autora.

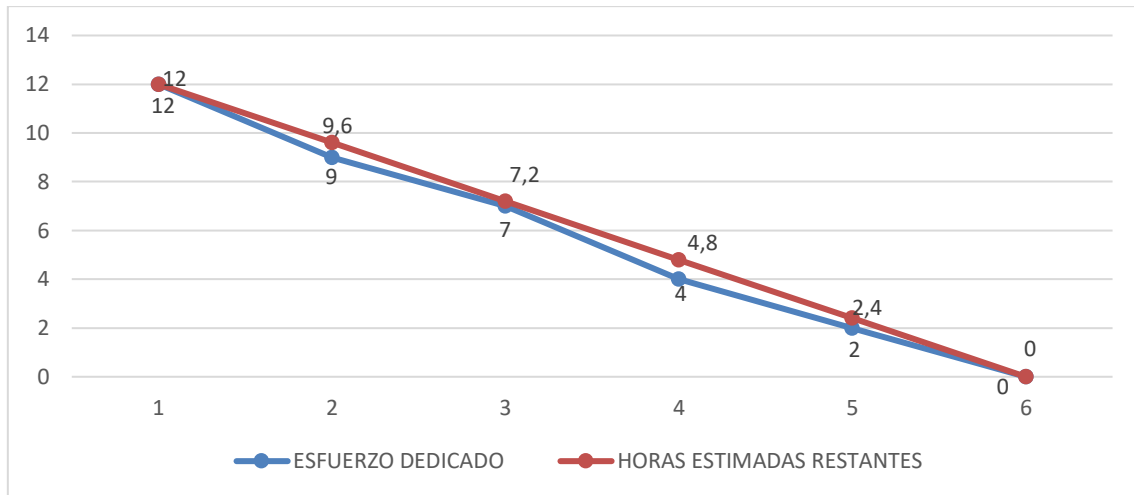


Gráfico 3.3. Sprint 3.

Fuente: La autora.

ETAPA 4: REVISIÓN DEL SPRINT

La reunión de Revisión del sprint se fundamentó en base a los objetivos y salidas determinados en la Planificación. Todos los sprints terminaron con un incremento del producto.

SPRINT 1

La primera salida o primer resultado de este sprint fue el ecosistema de trabajo preparado para el desarrollo. Esto correspondió a montar el backend y el frontend; en el primero se usó PHP con el framework de Laravel, en el segundo, se usaron framework y librerías tales como: Ionic Angular, TypeScript, CSS, y como intérprete de código HTML.

Como segunda salida, se tuvo el incremento del producto, evaluado de manera funcional y con las pruebas unitarias. En este sprint se desarrolló el módulo o las funcionalidades correspondientes a la gestión de los usuarios, con opción de crear una cuenta, actualizar su perfil, loguearse con credenciales de la aplicación o de Facebook. También se diseñó la interfaz principal de la aplicación. El detalle del incremento se muestra a continuación:

Al instalar la aplicación en el dispositivo móvil se establece el ícono en la pantalla (figura 3.8A.), y al iniciar la aplicación, lo primero que se muestra al usuario es el Splash (figura 3.8B.), conformado por una imagen que representa el propósito de la aplicación, el cual demora unos pocos segundos, luego aparecerá la pantalla principal de la aplicación.

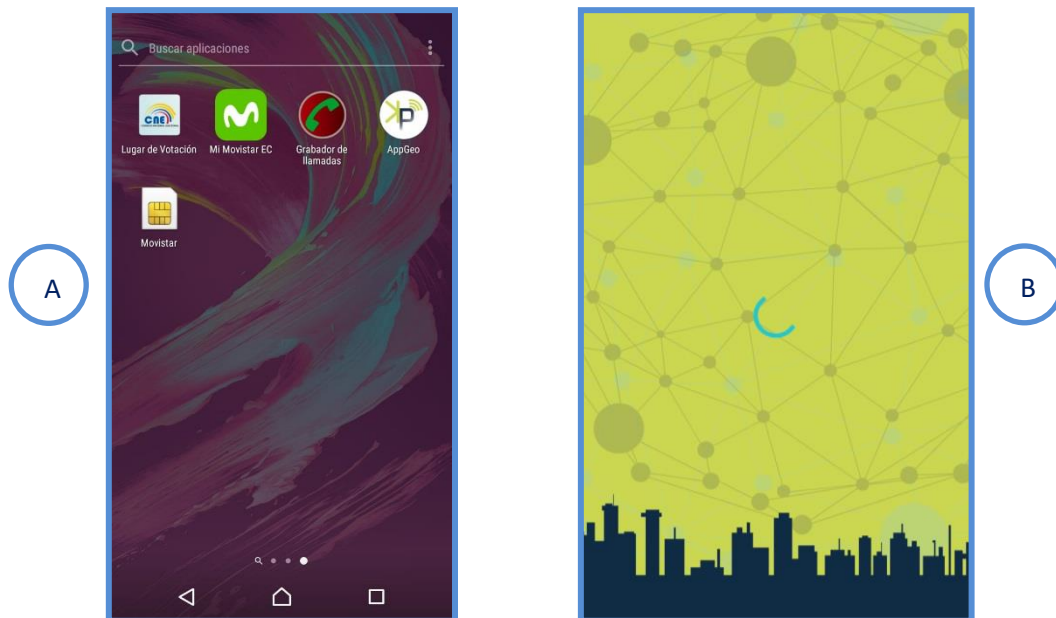


Figura 3. 8. Ícono y Splash de la aplicación.

La pantalla principal está compuesta por un botón para loguearse utilizando la cuenta de Facebook y otro botón para registrarse en la aplicación. Al iniciar sesión se puede observar la página principal con el menú, ubicado de manera lateral al lado izquierdo de la pantalla, con las opciones de que dispone la aplicación.



Figura 3. 9. Pantalla principal de la aplicación.

En caso de ingresar usando la cuenta de Facebook (ver figura 3.10A.) los datos necesarios se traen desde esta; por su parte, si el usuario quiere registrarse en la aplicación se muestra una pantalla en la que el usuario deberá ingresar la información solicitada (ver figura 3.10B.).

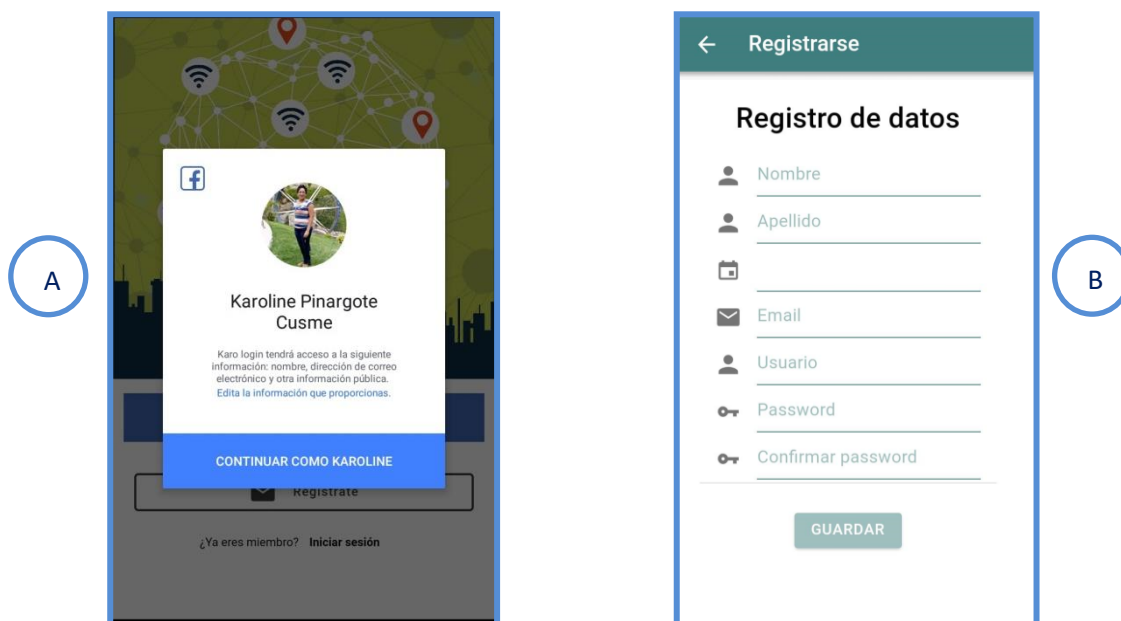


Figura 3. 10. Inicio de sesión con Facebook y registro.

Para el inicio de sesión con Facebook se utilizó uno de los servicios que este ofrece para el login con otras aplicaciones como es el API de Facebook, integrándolo en la aplicación para poder utilizar el método loginFb() y obtener datos del usuario del elemento access_token, devuelto en el objeto authResponse (ver figura 3.11.).

```
import { Facebook, FacebookLoginResponse } from '@ionic-native/facebook/ngx';
loginFb(){
  this.fb.login(['public_profile', 'email'])
  .then((res: FacebookLoginResponse) => {
    if (res.status==='connected'){
      this.user.img = 'https://graph.facebook.com/'+res.authResponse.userID+'/picture?type=square';
      this.getData(res.authResponse.accessToken);
    }else{
      alert('error al conectar');
    }
    console.log('Logueado a Facebook!', res)
  })
  .catch(e => console.log('Error al conectar con Facebook', e));
}
getData(access_token:string){
  let url = 'https://graph.facebook.com/me?fields=id,name,first_name,last_name,email&access_token=' + access_token;
  this.http.get(url).subscribe(data => {
    this.userdata = JSON.stringify(data);
    console.log(data);
  });
}
```

Figura 3. 11. Método loginFb().

Para el ingreso de datos se utilizaron cajas de texto HTML configurados con el tipo de datos que permitirían ingresar la información con el formato adecuado. El diseño es sencillo e intuitivo, básicamente en dos colores y los elementos como etiquetas, botones, iconos, se distribuyeron de tal manera que sea agradable a la vista del usuario. En la figura 3.12. se muestra el inicio de sesión mediante usuario y contraseña con previo registro.

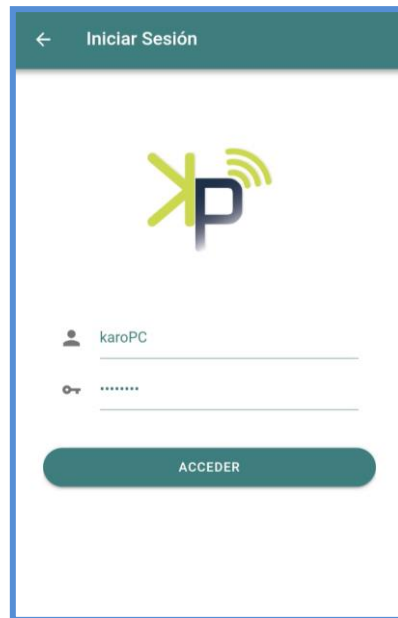


Figura 3. 12. Inicio de sesión con usuario y contraseña.

Los usuarios tienen la posibilidad de administrar su perfil, en este tienen la opción de editar los datos proporcionados en el registro.

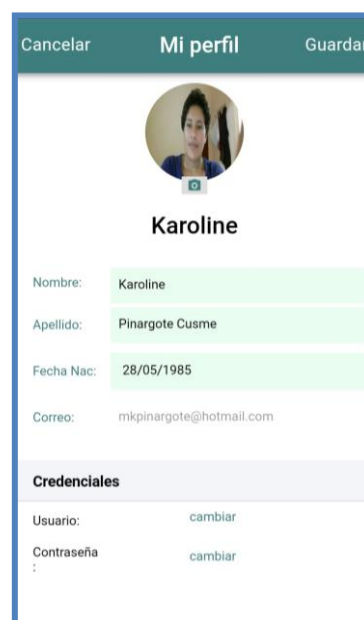



Figura 3. 13. Perfil de usuario.

SPRINT 2

Para este sprint se incluyeron todas las funciones referentes a la detección y administración de redes WiFi. En este apartado el usuario puede escanear, guardar, eliminar o establecer conexión con alguna de las señales detectadas. Para este fin, se utilizaron algunas librerías de Ionic; por ejemplo para el escaneo de las redes se usó la librería nativa de Ionic denominada Hotspot, este se pasó como constructor de la clase SearchWifiPage y ya con esto se hizo uso del método scanWifi() que devuelve un objeto de tipo Json con toda la información de las redes disponibles detectadas; este objeto lo almacenamos en un arreglo o array y como se utilizó también Angular, con el nombre del vector se pudo acceder a la información directamente desde la vista. Para mostrar atractivamente la información obtenida en el array, se utilizó la estructura de control ngFor de Angular, para poder acceder a los datos individuales de cada ítem o red escaneada y asignarlos a etiquetas HTML y nativas de Ionic utilizadas para este propósito.

Otra librería que se incluyó en la aplicación fue la LoadingController, que se utilizó para crear un método asíncrono llamado doRefresh(), con el que se actualiza la pantalla acorde aparezcan o desaparezcan redes dentro del radio de la ubicación actual. Adicionalmente, se creó un método para contar las redes escaneadas y el resultado se muestra en el footer de la página search-wifi.page.

Una vez que culmina la etapa de escaneo, la página muestra un listado de las redes encontradas, donde se visualiza el *SSID*, banda, dirección MAC y la intensidad de la señal. (figura 3.14.). Adicionalmente, se puede agregar redes ocultas mediante el botón  ingresando el nombre de la red y la contraseña.

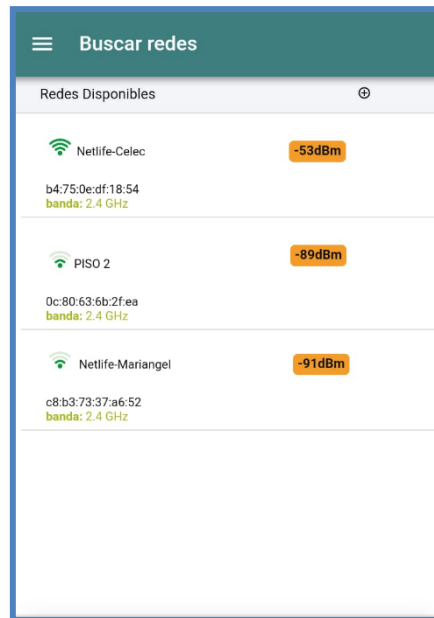


Figura 3. 14. Redes WiFi disponibles.

Para conectarse a la red, se selecciona la misma y aparece una ventana emergente, en la cual solita ingresar la contraseña.

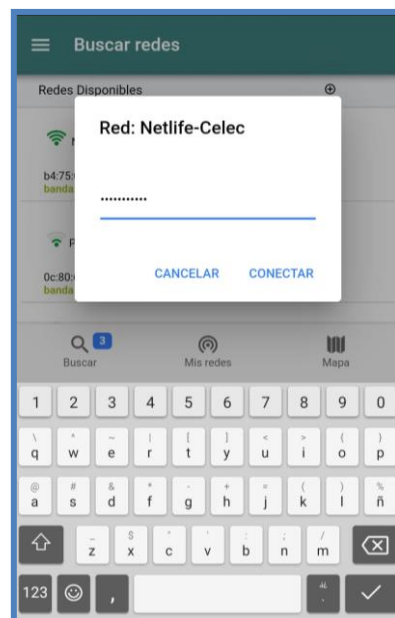


Figura 3. 15. Conectar una red disponible.

Adicionalmente, en el footer se agregaron dos botones denominados: *Mis Redes*, en el cual se pueden visualizar el listado de redes compartidas por el usuario (figura 3.16A.). El botón *Mapa*, que redirecciona a la ventana en la que se visualizan las redes de forma pública (figura 3.16B.).

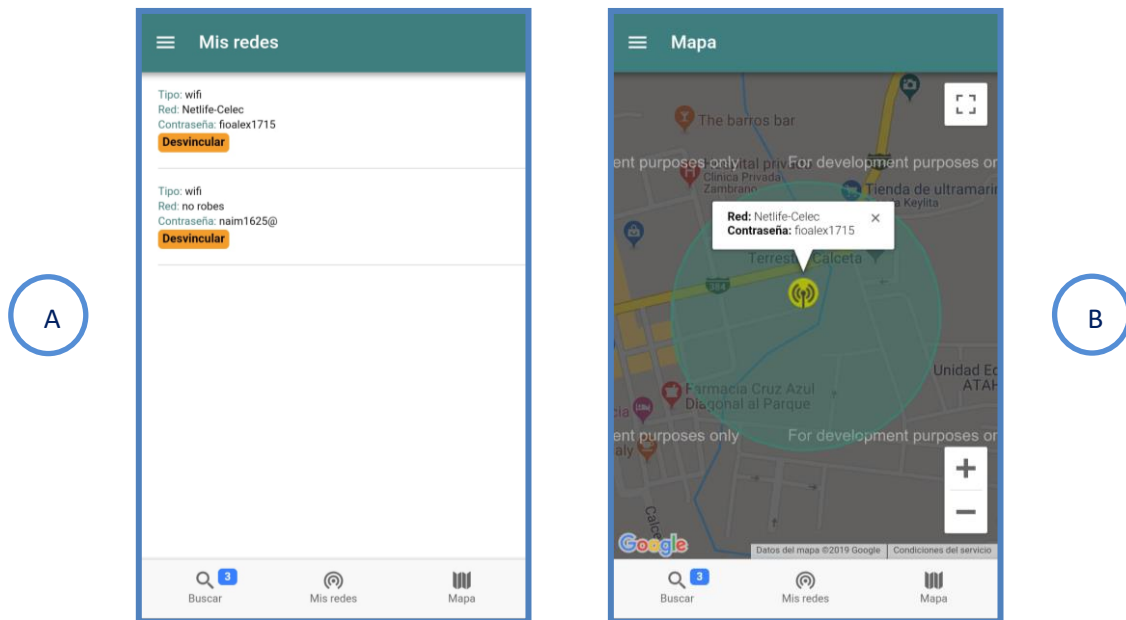


Figura 3. 16. Listado de redes compartidas por un usuario y visualización en el mapa.

Para la visualización de las redes en el mapa, en primera instancia se verifica si el dispositivo tiene una conexión a internet válida; si el resultado es falso, la función envía una alerta indicando que el dispositivo no está conectado a internet.

En la figura 3.17. se muestra la función `loadMap()`, carga el mapa con la ubicación del dispositivo y muestra el radio de 300 metros. Esto lo hace a través de la función `getLocation()` que devuelve la latitud y la longitud de ubicación.


```

async loadMap() {
  const loading = await this.loadingCtrl.create();
  loading.present();
  const myLatLng = await this.getLocation();
  //const myLatLng = new google.maps.LatLng(40.723333, -73.983438);
  const mapEle: HTMLElement = document.getElementById('map_canvas');
  this.mapRef = new google.maps.Map(mapEle, {
    center: myLatLng,
    zoom: 16,
    mapTypeControl: true,
    mapTypeControlOptions: { style: google.maps.MapTypeControlStyle.DROPDOWN_MENU },
    navigationControl: true,
    mapTypeId: google.maps.MapTypeId.ROADMAP
  });
  var cityCircle = new google.maps.Circle({
    strokeColor: '#348772',
    strokeOpacity: 0.8,
    strokeWeight: 2,
    fillColor: '#348772',
    fillOpacity: 0.35,
    map: this.mapRef,
    center: myLatLng,
    radius: 300
  });
  google.maps.event
    .addListenerOnce(this.mapRef, 'idle', () => {
      loading.dismiss();
      const marker = new google.maps.Marker({
        position: myLatLng,
        map: this.mapRef,
        animation: google.maps.Animation.DROP,
        title: 'Ubicación actual',
        icon: 'https://icon-icons.com/icons2/165/PNG/32/mapmarker_marker_outside_chartreuse_23006.png'
      });
      this.addMaker(myLatLng);
    });
}

```

Figura 3. 17. Función loadMap().

Los valores de latitud y la longitud devueltos por getLocation() se almacenan en una constante llamada myLatLng, la que se pasa como parámetro a la función addMaker(), que consume la función getRedes() del servicio web redesServices y almacena el listado devuelto por la función en un vector. Para cada ítem de este vector se crea una nueva coordenada de Google Maps asignándole la latitud y longitud de la red actual con la que se determina la distancia entre estas coordenadas y las del dispositivo. Si la distancia está dentro de los 300 metros, se añade el marcador de la red dentro del mapa y a este se le asigna un evento en la función dragend para mostrar el nombre de la red y la contraseña. (ver figura 3.18.)

```

private addMaker(posicionActual: any) {
  this.redesServices.getRedes()
    .then(data => {
      var location = [];
      location = data['redes'];
      var i;
      for (i = 0; i < location.length; i++) {
        var marker_lat_lng = new google.maps.LatLng(location[i].latitud, location[i].longitud);
        var distance_from_location = google.maps.geometry.spherical.computeDistanceBetween(posicionActual, marker_lat_lng);
        if (distance_from_location <= 300) {
          var infowindow = new google.maps.InfoWindow();
          const marker = new google.maps.Marker({
            position: marker_lat_lng,
            map: this.mapRef,
            animation: google.maps.Animation.DROP,
            title: 'wifi',
            icon: 'http://icons.iconarchive.com/icons/papirus-team/papirus-apps/32/fern-wifi-cracker-icon.png'
          });
          var geocoder = new google.maps.Geocoder();
          // le asignamos una funcion al eventos dragend del marcado
          google.maps.event.addListener(marker, 'click', (function (marker, i) {
            return function () {
              geocoder.geocode({ 'latLng': marker_lat_lng }, function (results, status) {
                if (status == google.maps.GeocoderStatus.OK) {
                  var address = results[0]['formatted_address'];
                  console.log(address);
                }
              });
              infowindow.setContent('<strong>Red:</strong> ' + location[i].nombreRed + '</br><strong>Contraseña: </strong>');
              infowindow.open(this.mapRef, marker);
            }
          }))(marker, i);
        } else {
          console.log('=> is NOT in searchArea');
        }
      }
    });
}

```

Figura 3. 18. Función addMarker().

La opción *Mis redes* muestra un listado de las redes a las que se ha conectado el usuario del dispositivo, este listado lo obtiene de la función `getRedesUser()` que recibe como parámetro el id del usuario.

```

async getMyredes(id:number) {
  const loading = await this.loadingCtrl.create();
  loading.present();
  this.redesServices.getRedesUser(id)
    .then(data => {
      loading.dismiss();
      this.redesUser = data;
      debugger
    }, (error) => {
      debugger
      loading.dismiss();
    })
}

```

Figura 3. 19. Obtener listado de redes WiFi de un usuario.

SPRINT 3

En este sprint se desarrolló la funcionalidad de test o monitoreo de la red WiFi a la que está conectado el dispositivo. Este test muestra información relevante como el nombre de la red, estado, intensidad de señal, velocidad de enlace, dirección MAC e IP (figura 3.20.); con estos datos el usuario podrá determinar si la red cumple con las características que necesita o, en su defecto, buscar otra señal que cumpla sus expectativas.

Para el diseño y codificación de este apartado se utilizó la librería chart.js, una librería de JavaScript que utiliza el canvas de HTML5 para mostrar impresionantes gráficos para la web gracias a las múltiples opciones de personalización que posee. Otro elemento utilizado fue el método `getConnectionInfo()`, que retorna información dinámica de la conexión WiFi actual, si hay alguna activa.

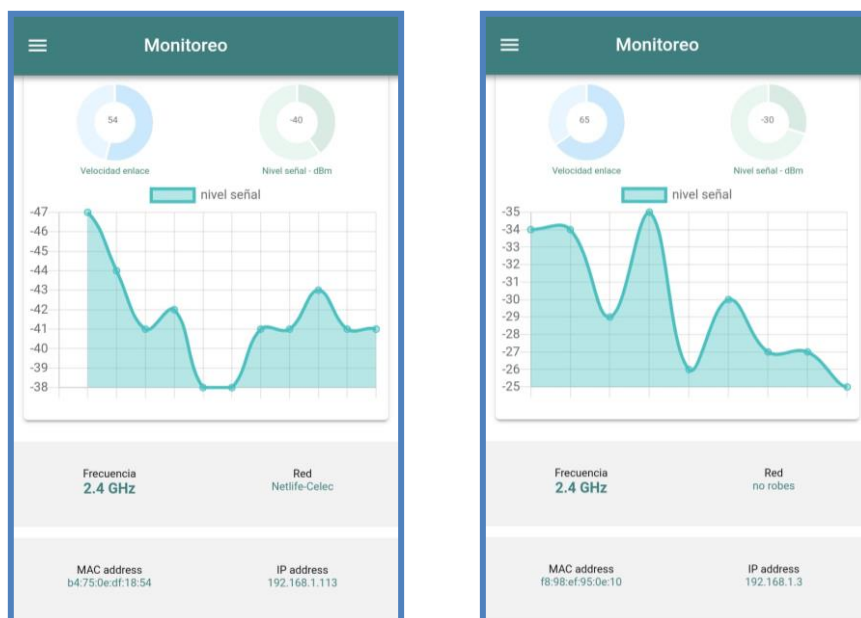


Figura 3. 20. Monitoreo de dos redes inalámbricas.

ETAPA 5: RETROSPECTIVA DEL SPRINT

Este último evento de cada sprint se ejecutó el mismo día en el que se llevó a cabo las reuniones de revisión del sprint, con el fin de valorar cómo se estaba desarrollando el trabajo, identificar problemas u otro tipo de incidencias que pudieran afectar los tiempos de desarrollo y la calidad de los incrementos y por ende del producto final.

En la retrospectiva del sprint 1 se determinó que las horas de dedicación iban de la mano con el tiempo planificado para el cumplimiento oportuno del mismo. En esta primera reunión la autora manifestó que las tareas le habían tomado más tiempo por cuanto tuvo que familiarizarse con el lenguaje de programación.

En el segundo sprint, el SCRUM Máster sugirió un cambio en el diseño del menú con el propósito de hacer más ágil e intuitiva la navegación para el usuario. Finalmente, en el último sprint se acordó realizar de manera inmediata la prueba de integración de los incrementos desarrollados para evaluar de manera conjunta e interrelacionada el desempeño de cada una de las funciones.

PRUEBAS UNITARIAS

Una vez terminado el incremento en cada uno de los sprint, se valoró el funcionamiento y capacidad de respuesta del mismo a través de la documentación de las APIs, en la figura 3.21. se muestra el listado de las APIs de la aplicación móvil.

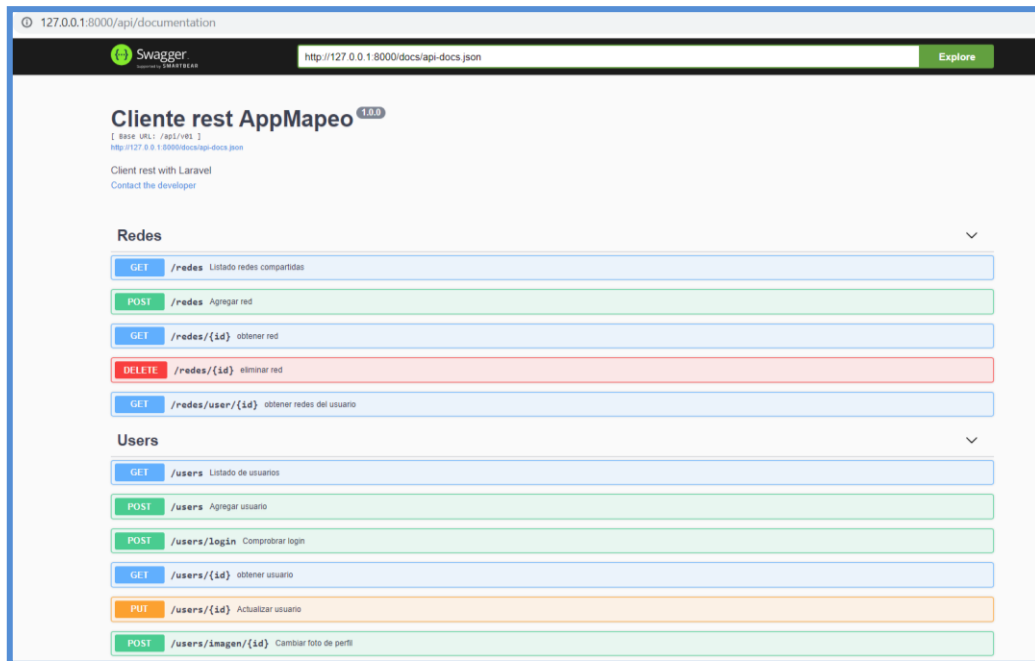


Figura 3. 21. Listado de las APIs.

En la figura 3.22. se muestra la API para agregar un usuario, se debe llenar los campos con información básica del usuario.

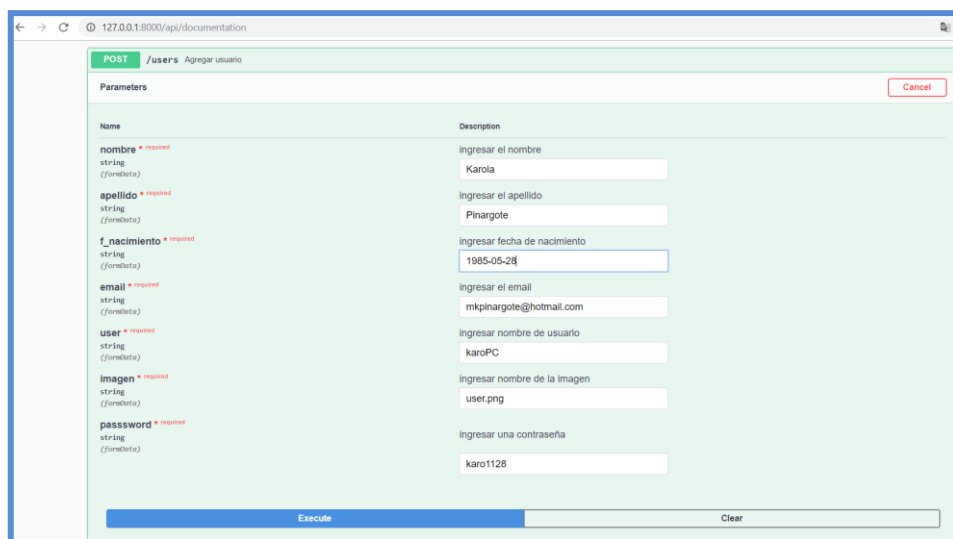


Figura 3. 22. API para agregar un usuario.

Una vez ingresado el usuario, se puede comprobar que el mismo se lo realizó de forma correcta, tal como se muestra en la figura 3.23.

```
{
  "message": "usuario guardado",
  "userId": 1,
  "user": "karoPC"
}
```

Figura 3. 23. Prueba de funcionamiento API para obtener usuario.

A continuación, se comprueba el inicio de sesión con el respectivo nombre de usuario y contraseña, previamente ingresados.

POST /users/login Comprobar login

Parameters

Name	Description
user * required string (formData)	ingresar el usuario karoPC
password * required string (formData)	ingresar la contraseña karo1128

Execute Clear

Figura 3. 24. Prueba de funcionamiento API para comprobar login.

Una vez hecha la comprobación del inicio de sesión, muestra que el inicio de sesión o login fue correcto, como se detalla en la figura 3.25.

```
{
  "message": "login correcto",
  "userId": 1,
  "user": "karoPC"
}
```

Figura 3. 25. Prueba de funcionamiento del API agregar un usuario.

En la figura 3.26., se muestra el listado de usuarios agregados.

```
{
  "user": {
    "id": 1,
    "nombre": "Karola",
    "apellido": "Pinargote",
    "f_nacimiento": "1985-05-28",
    "email": "mkpinargote@hotmail.com",
    "user": "karoPC",
    "imagen": "user.png"
  }
}
```

Figura 3. 26. Prueba de funcionamiento API para obtener listado de usuario.

En la figura 3.27., se realizó la actualización de datos del usuario.

Name	Description
id * required integer (path)	ingresar id del usuario 1
nombre * required string (formData)	ingresar el nombre Karolina
apellido * required string (formData)	ingresar el apellido Pinargote Cusme
f_nacimiento * required string (formData)	ingresar fecha de nacimiento 1985-05-28
email * required string (formData)	ingresar el email mkpinargote@gmail.com

Figura 3. 27. Prueba de funcionamiento API para actualizar usuario.

A continuación, se procedió a cambiar la foto de perfil del usuario.

Name	Description
id * required integer (path)	ingresar id del usuario 1
file * required file (formData)	subir imagen [Seleccionar archivo] KAROLA LOGOS-07.png

Figura 3. 28. Prueba de funcionamiento API para cambiar foto de perfil.

Una vez que se hizo la respectiva edición de datos del usuario, se puede comprobar que los cambios se realizaron con éxito, tal como se muestra en la figura 3.29.

```

{
  "user": {
    "id": 1,
    "nombre": "Karolina",
    "apellido": "Pinargote Cusme",
    "f_nacimiento": "1985-05-28",
    "email": "mkpinargote@gmail.com",
    "user": "karopC",
    "imagen": "KAROLA LOGOS-07.png"
  }
}

```

Figura 3. 29. Actualización de datos.

Para la obtención de información de un usuario, solo es necesario ingresar el id del usuario, tal como se observa en la figura 3.30.

Name	Description
id * required integer (path)	ingresar id del usuario

Figura 3. 30. Comprobación de actualización de información del usuario.

Para agregar una red, se hace necesario llenar los campos que se observan en la figura 3.31.

Name	Description
tipoRed * required string (formData)	ingresar el tipo de red wifi
nombreRed * required string (formData)	ingresar nombre de la red contrata tu plan
passwordRed * required string (formData)	ingresar la contraseña de la red naim1625
estadoRed * required string (formData)	ingresar el estado (1 o 0) de la red 0
latitud * required string (formData)	ingresar la latitud 084444
longitud * required string (formData)	ingresar la longitud 096666
idUser * required string (formData)	ingresar el id de usuario 1

Figura 3. 31. API para agregar red.

Una vez agregada la red de forma correcta, muestra un mensaje donde se indica que el registro fue correcto.


```

{
  "message": "Registro correcto",
  "code": "201",
  "id": 1
}

```

Figura 3. 32. Registro correcto al agregar una red.

En la figura 3.33., se puede observar la API para listar redes compartidas.

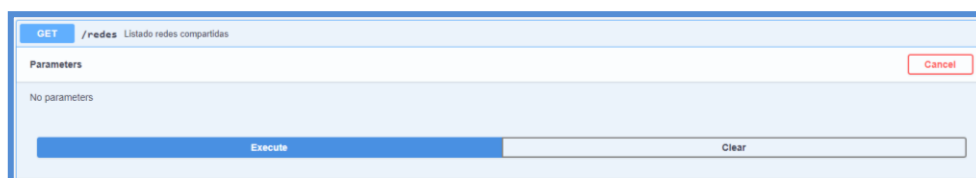


Figura 3. 33. Prueba de funcionamiento API para listar redes compartidas.

A continuación, se observa la API para obtener una red a través de un id.

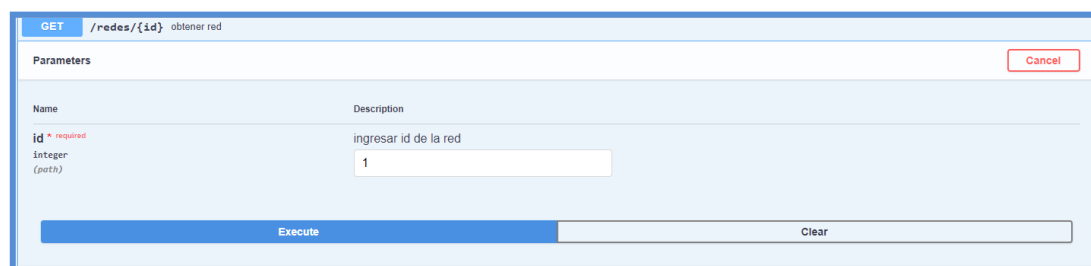


Figura 3. 34. Prueba de funcionamiento API para obtener red.

En la figura 3.35., se observan las redes que fueron ingresadas por un usuario específico.

```

{
  "redes": [
    {
      "id": 1,
      "tipoRed": "wifi",
      "nombreRed": "contrata tu plan",
      "passwordRed": "naim1625",
      "estadoRed": 0,
      "latitud": "084444",
      "longitud": "096666",
      "idUser": 1
    }
  ]
}

```

Figura 3. 35. Prueba de funcionamiento API obtener redes de un usuario.

PRUEBAS DE INTEGRACIÓN

Con los incrementos obtenidos del desarrollo de los sprint se realizó una prueba de integración en la cual se testeó el comportamiento de la aplicación. De esta primera prueba salió una modificación en la distribución de las diferentes opciones para la gestión de las redes; en una segunda prueba se ingresaron datos reales, obteniendo los resultados esperados en cuanto a funcionamiento ver figura 3.36. donde se observa el registro de usuario, figura 3.37. permiso por parte de la aplicación para usar datos del usuario dentro de la aplicación, figura 3.38. se muestra que el Logueo con Facebook se hizo de forma correcta y figura 3.39. donde se indica que la red fue guardada con éxito) y velocidad de respuesta.

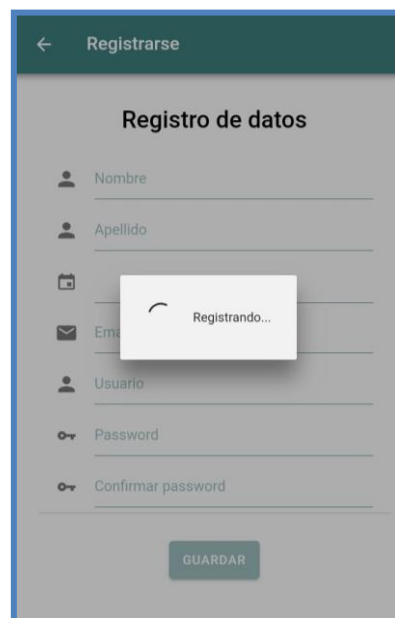


Figura 3. 36. Registrando datos del usuario.

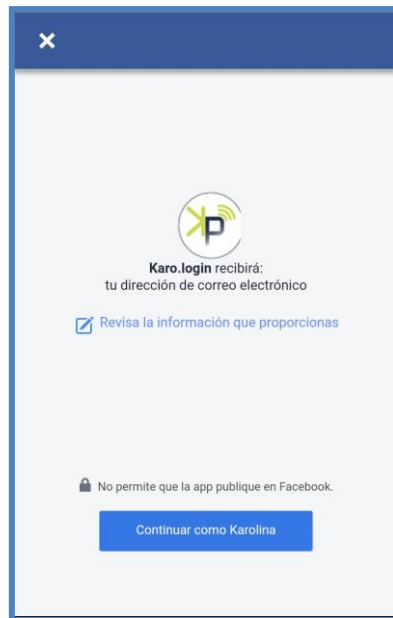


Figura 3. 37. Petición al usuario para uso de información.



Figura 3. 38. Login correcto al iniciar con Facebook.

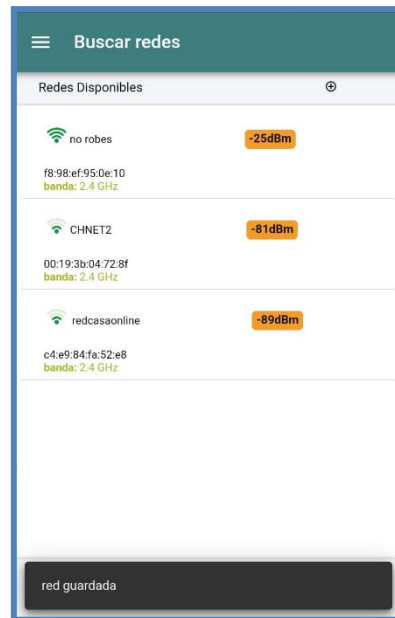


Figura 3. 39. Red guardada con éxito.

Con la ayuda de Apache JMeter (herramienta para realizar pruebas de carga y desempeño) se hizo la respectiva evaluación de rendimiento de la aplicación, simulando el envío de 1000 peticiones al servidor al mismo instante, obteniendo los siguientes resultados:

Para obtener las redes del usuario, el tiempo de respuesta para la primera petición fue de 0.27 segundos y de la última petición 14.85 segundos. Tal como se muestra en la figura 3.40.

Petición HTTP

Nombre: Usuarios

Comentarios

Basic Advanced

Servidor Web

Protocolo: Nombre de Servidor o IP: agile-scrubland-87518.herokuapp.com Puerto:

Petición HTTP

Método: GET Ruta: /api/v01/redes/user/7 Codificación del contenido:

Etiqueta	# Muestras	Media	Min	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
Usuarios	1000	7613	276	14854	4112,62	0,00%	63,1/sec	36,38	9,80	590,0
Total	1000	7613	276	14854	4112,62	0,00%	63,1/sec	36,38	9,80	590,0

Figura 3. 40. Petición: Obtener redes del usuario.

Para obtener las redes compartidas en un radio de 300 metros desde la ubicación actual del usuario, el tiempo de respuesta para la primera petición fue de 3.47 segundos y de la última petición 18.21 segundos. Tal como se indica la figura 3.41.

Petición HTTP

Nombre: Redes compartidas

Comentarios

Basic Advanced

Servidor Web

Protocolo: Nombre de Servidor o IP: agile-scrubland-87518.herokuapp.com Puerto:

Petición HTTP

Método: POST Ruta: /api/v0/redes/publicas Codificación del contenido:

Redirigir Automáticamente Seguir Redirecciones Utilizar KeepAlive Usar 'multipart/form-data' para HTTP POST Cabeceras compatibles con navegadores

Parameters Body Data Files Upload

Enviar Parámetros Con la Petición:

Nombre:	Valor	¿Codificar?	Content-Type	¿Incluir Equals?
latitud	-0.8485939	<input type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>
longitud	-80.1611082	<input type="checkbox"/>	text/plain	<input checked="" type="checkbox"/>

Etiqueta	# Muestras	Media	Min	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
Redes comparti...	1000	10449	3466	18208	4056.44	0.00%	52.6/sec	33.49	14.64	652.0
Total	1000	10449	3466	18208	4056.44	0.00%	52.6/sec	33.49	14.64	652.0

Figura 3. 41. Petición: Obtener redes compartidas.

DISCUSIÓN

En esta sección se describen las características que ofrece la aplicación a los usuarios de dispositivos móviles. Como se indicó en la revisión bibliográfica, no se encontró aplicación con similares características.

AppGeo permite detectar redes WiFi cercanas, conectarse a una red ingresando su contraseña, posteriormente queda a criterio del usuario compartir la red guardada de manera pública, para que otros usuarios que utilicen la aplicación puedan hacer uso de esta; indicando que, al compartir la red, aquellos usuarios que hagan uso de la aplicación y estén en un rango de 300 metros, puedan visualizar el nombre de la red con su respectiva contraseña y conectarse a la red. Otra funcionalidad que ofrece la aplicación es el poder monitorear la red a la cual se está conectado.

En el desarrollo de AppGeo se utilizó el framework open source (código abierto) Ionic que permite desarrollar aplicaciones móviles híbridas bajo un WebView de manera sencilla, ágil y con un buen rendimiento. Para poder loguearse con Facebook se utilizó el servicio API que esta ofrece, mediante la cual se hace uso de las credenciales del usuario para acceder a la aplicación o en su defecto los usuarios pueden registrarse mediante un formulario en el que deben ingresar información básica.

Para detectar las redes inalámbricas se utilizó la librería Hotspot que sirve administrar redes, con esta librería instalada se empleó el método scanWifi() que se configuró para el escaneo, la cual obtiene la información básica y geográfica de cada red. Para que las redes escaneadas se actualizarán en tiempo real se usó la librería LoadingController y gracias a framework Angular, la aplicación no requiere refrescar la página para actualizar el listado de las redes.

A través de la API de Google Maps, la aplicación dibuja las coordenadas de cada una de las redes en el mapa, donde se visualizan como marcadores, mostrando el nombre y clave de la red para que el usuario pueda conectarse. Para que las redes se muestren en el mapa, estas deben ser compartidas por el propietario o usuario de dicha red, indicando que se muestran las redes compartidas en el mapa, en una distancia de hasta 300 metros a la redonda.

Cuando el usuario se ha conectado a una red, tiene la opción de monitorearla gracias a la opción estadística de la aplicación, durante el test el usuario puede visualizar información como intensidad de la señal, frecuencia, velocidad de enlace, calidad de la red (dBm), dirección MAC e IP y el nombre de la red; con esta información el usuario puede tomar decisiones respecto a cada señal.

Todas estas características están disponibles en la aplicación en una misma versión free (libre). El objetivo de la aplicación es brindar al usuario una herramienta para la gestión de redes inalámbricas, que le permita estar conectado sin importar su ubicación.

La aplicación se desarrolló en un entorno multiplataforma; sin embargo, en esta primera versión, la compilación y las pruebas de AppGeo se realizaron para el sistema Android.

CAPÍTULO V. CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

Con base en los resultados obtenidos en el desarrollo de la aplicación móvil de geolocalización de redes WLAN se concluye lo siguiente:

1. Mediante la aplicación del ERS se recopiló un total de quince historias de usuario que permitieron una visión más precisa del propósito de la aplicación, lo que permitió realizar de manera satisfactoria cada una de los sprint obtenidos.
2. La arquitectura basada en Cloud Computing garantizó la permanente disponibilidad de la aplicación y de la data almacenada; asimismo, contribuyó a que la interacción entre los diferentes componentes fuese robusta y flexible.
3. La aplicación de la metodología SCRUM aseguró el desarrollo ordenado y oportuno de la aplicación, permitiendo cumplir con los entregables del producto en cada Sprint y finalmente con la aplicación integral de geolocalización de redes inalámbricas.
4. Las pruebas unitarias y de integración evidenciaron que la aplicación es capaz de procesar hasta 1000 solicitudes de manera simultánea, con una velocidad de respuesta de hasta 16 segundos.

RECOMENDACIONES

1. Que para el desarrollo de aplicaciones pequeñas, se utilice técnicas o estándares menos complejos, ya que el IEEE 830, es empleado en equipos de trabajo con mayor número de integrantes; por tanto, se recomienda se utilice un método o procedimiento sencillo que evite el consumo excesivo de tiempo y documentación innecesaria.
2. Que para este tipo de aplicaciones se utilice un entorno ágil basado en servicios de Cloud Computing que permita al proyecto crecer y evolucionar con gran flexibilidad sin perder su disponibilidad.
3. Que aún para el desarrollo de aplicaciones móviles pequeñas como AppGeo se apliquen metodologías de desarrollo ágil como SCRUM para garantizar el cumplimiento de las metas propuestas, ya que permite trabajar por sprint.
4. Que al momento de realizar las pruebas unitarias y de integración se consideren los parámetros y valores necesarios que permitan evaluar la aplicación ante posibles situaciones y poder tomar acciones de mejora temprana.

BIBLIOGRAFÍA

- Botella, I. (2018). Sistema de seguimiento mediante mapa online para los niños que viajan en transporte público. Universidad Politécnica de Valencia. España.
- Carpintero, J. (2014). Aplicación de Android para geo-localización de tareas pendientes. Gestión e implementación de BD y aplicación móvil. Universidad Politécnica de Valencia. España.
- Chinchay, M. (2015). Desarrollo de una aplicación móvil android para la búsqueda de plazas disponibles en un parqueadero. Universidad Nacional de Loja. Loja.
- Company, M. (2015). Aplicación Android: Red social de búsqueda de aparcamiento. Universidad Politécnica de Valencia. España.
- Espinoza, M. (2015). Análisis y diseño de una aplicación móvil para la localización de rutas de transporte público. Escuela Superior Politécnica del Litoral. Guayaquil.
- Fuster, F. (2015). Aplicación Android de realidad aumentada para mostrar imágenes históricas de lugares turísticos de interés. Universidad Politécnica de Valencia. España.
- GAD Municipal de Guayaquil. (2019). Red de conexión: ALCALDIA_GUAYAQUIL. Recuperado de: <https://guayaquil.gob.ec/Paginas/internet-gratis.aspx>
- García, J., De la Rosa, R., Castillo, H. y Cervantes, A. (2014). Aplicación móvil para mostrar sitios turísticos empleando realidad aumentada y geolocalización. Research in Computing Science, 88, 87-101.
- Gualotuña, D. y Miranda, S. (2014). Desarrollo de una aplicación de geolocalización que facilite la ubicación de las dependencias en la Universidad Nacional de Loja con técnicas de realidad aumentada para dispositivos móviles. Universidad Nacional de Loja. Loja.
- Guillem, F. (2012). UPV-MobARGuide: aplicación Android de realidad aumentada para guía interactiva de la UPV orientada a móviles (Proyecto fin de Carrera). Universidad Politécnica de Valencia. España.
- Jaén, J. (2017). Aplicación Android para la búsqueda de precios económicos de combustibles y geolocalización de las estaciones de servicio. Universidad Politécnica de Valencia. España.

- Llerena, J., Andina, M. y Grijalva, J. (2018). Mobile Application to Promote the Malecón 2000 Tourism Using Augmented Reality and Geolocation. 2018 International Conference on Information Systems and Computer Science (INCISCOS).doi:10.1109/inciscos.2018.00038
- Pimentel, B., Tummala, M., y McEachen, J. (2015). Passive geolocation of mobile subscribers using a single base station with third party capture. 2015 9th International Conference on Signal Processing and Communication Systems (ICSPCS).doi:10.1109/icspcs.2015.7391765
- Sánchez, M. (2012). Integración de Foursquare y geolocalización en una aplicación móvil para la creación de rutas turísticas. Universidad Politécnica de Valencia. España.
- Sepúlveda, M. y Durán, P. (2014). Geolocalización de centrales de impresión y fotocopiado en le Universidad de Santiago de Chile. Una aplicación móvil basada en Android. Universidad Santiago de Chile. Chile.
- Tartan, E. y Ciflikli, C. (2018). An Android Application for Geolocation Based Health Monitoring, Consultancy and Alarm System. 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC).doi:10.1109/compsac.2018.10254
- Ventura, L. (2014). Automatización del proceso de ventas y distribución utilizando tecnología móvil y geolocalización para la empresa líder SRL. Universidad Privada Antenor Orrego. Perú.
- Vera, D. (2014). Desarrollo de una aplicación móvil para apoyar al turismo del centro histórico de Quito, utilizando realidad aumentada y geolocalización, para la empresa VLBS CÍA. LTDA (Bachelor's thesis). Universidad de las Fuerzas Armadas ESPE. Quito.
- Vera, R. (2012). Un Modelo Predictivo para la localización de usuarios móviles en escenarios bajo techo. (Proyecto fin de Máster). Universidad de Chile, Chile.
- Vernaza, B. (2015). Implementación de aplicación móvil para Android o IOS con realidad aumentada y geolocalización para asistencia y generación de citas en veterinarias del sur de Guayaquil sincronizado con gestor de contenido web publicitario. Universidad de Guayaquil. Guayaquil.
- Wilken, J., y Bradley, A. (2016). Ionic in action: Hybrid mobile apps with Ionic and AngularJS. Manning Publications.

ANEXOS

ANEXO 1.**ESPECIFICACIÓN DE REQUISITOS DE SOFTWARE SEGÚN ESTÁNDAR
IEEE 830**A rounded rectangular box with a dashed blue border containing the title text.

**ESPECIFICACIÓN DE
REQUISITOS DE SOFTWARE
SEGÚN ESTÁNDAR IEEE 830**

INTRODUCCIÓN

El presente documento detalla la Especificación de Requisitos de Software (ERS) para el desarrollo de la aplicación móvil de geolocalización de redes WLAN, dentro del cual se puntualiza cada uno de los requisitos o requerimientos, recursos necesarios, procedimientos y otros aspectos específicos de la aplicación. La información aquí descrita se ha obtenido mediante un análisis técnico de requerimientos, características, restricciones, entre otros, que son indispensables para el correcto funcionamiento de la aplicación.

OBJETIVO

Proveer de información técnica relevante sobre la aplicación móvil de geolocalización de redes WLAN (WiFi) que constituya una guía para los desarrolladores que requieran realizar mejoras o modificaciones en la aplicación.

ALCANCE

La aplicación móvil de geolocalización de redes WLAN (WiFi) permite a los usuarios escanear las señales de redes inalámbricas que existen alrededor en un radio determinado y luego poder ubicar estas redes en Google Maps, utilizando coordenadas de longitud y latitud; adicionalmente, los usuarios pueden registrarse y administrar sus credenciales. Además, la aplicación permite el inicio de sesión mediante la validación de la cuenta de Facebook.

PERSONAL INVOLUCRADO

Nombre	Karolina Pinargote Cusme
Rol	Analista, diseñadora y programadora
Responsabilidad	Análisis de requerimientos, diseño y programación de la aplicación móvil
Información de contacto	mkpinargote@hotmail.com

Nombre	Tomás Loor Vera
Rol	Scrum Máster
Responsabilidad	Velar por el óptimo y oportuno desarrollo del proyecto
Información de contacto	pipos_rgt@hotmail.com

DEFINICIONES, SIGLAS Y ABREVIATURAS

NOMBRE	DESCRIPCIÓN
Usuario	Persona que usará el sistema
AppGeo	Aplicación de geolocalización de redes WiFi
ERS	Especificación de Requisitos de Software
RF	Requisito Funcional
RNF	Requisito No Funcional

REFERENCIAS

TÍTULO DEL DOCUMENTO	REFERENCIA
Estándar IEEE 830 - 1998	IEEE

VISIÓN GENERAL DEL DOCUMENTO

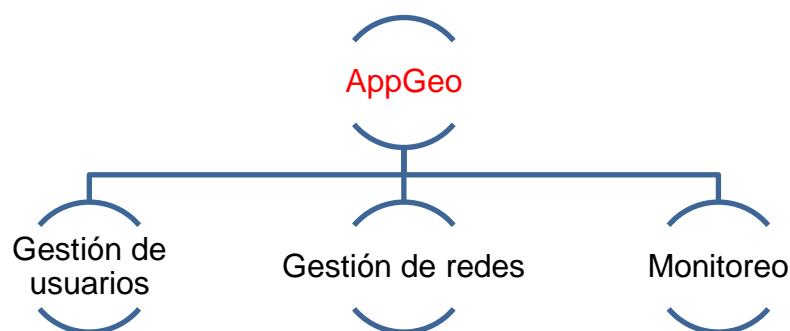
El presente documento contiene tres apartados: en el primero se presenta una breve introducción al mismo; en el segundo se realiza una descripción general de los procesos o funcionalidad de la aplicación; y en el tercero se describen los requisitos de forma detallada.

DESCRIPCIÓN GENERAL

PERSPECTIVA DEL PRODUCTO

La aplicación móvil AppGeo estará diseñada para funcionar en teléfonos inteligentes y otros dispositivos móviles con una función de conexión WiFi, para sistemas operativos Android.

FUNCIONALIDAD DEL PRODUCTO



- **Gestión de usuario**

En esta sección se podrá crear un nuevo usuario, además del inicio de sesión, el cual se lo realizará usando las credenciales de Facebook o ingresando el respectivo usuario y contraseña; además se podrá visualizar y actualizar información personal.

- **Gestión de redes**

En esta sección se listarán las redes WiFi disponibles, se las podrá guardar, y visualizar su geolocalización en un mapa de Google.

- **Monitoreo**

En esta sección se mostrará un reporte estadístico con todos los detalles de la red WiFi a la cual se está conectado.

CARACTERÍSTICAS DE LOS USUARIOS

Para acceder a la aplicación móvil se necesitará tener una cuenta de acceso, para lo cual existirá un registro previo para usuarios nuevos.

Tipo de usuario	Usuario
Detalle	Tendrá acceso a toda la aplicación móvil
Actividades	Uso de la aplicación.

LIMITACIONES

- La aplicación es de uso exclusivo en dispositivos móviles.
- Para que la aplicación funcione correctamente se necesitará contar con un plan de datos o estar conectados a una red WiFi.
- El Smartphone debe tener habilitado el servicio de ubicación (GPS).
- El dispositivo móvil debe tener instalada y corriendo la aplicación Google Play Services.

- El dispositivo debe tener instalada la aplicación Maps de Google.

SUPOSICIONES Y DEPENDENCIAS

La aplicación soportará el sistema operativo Android con versión 6.0.1 o superior.

RQUISITOS FUTUROS

- La aplicación puede ser compilada para que funcione en otros sistemas operativos, como requisito previo sería agregar la plataforma.

REQUISITOS ESPECÍFICOS

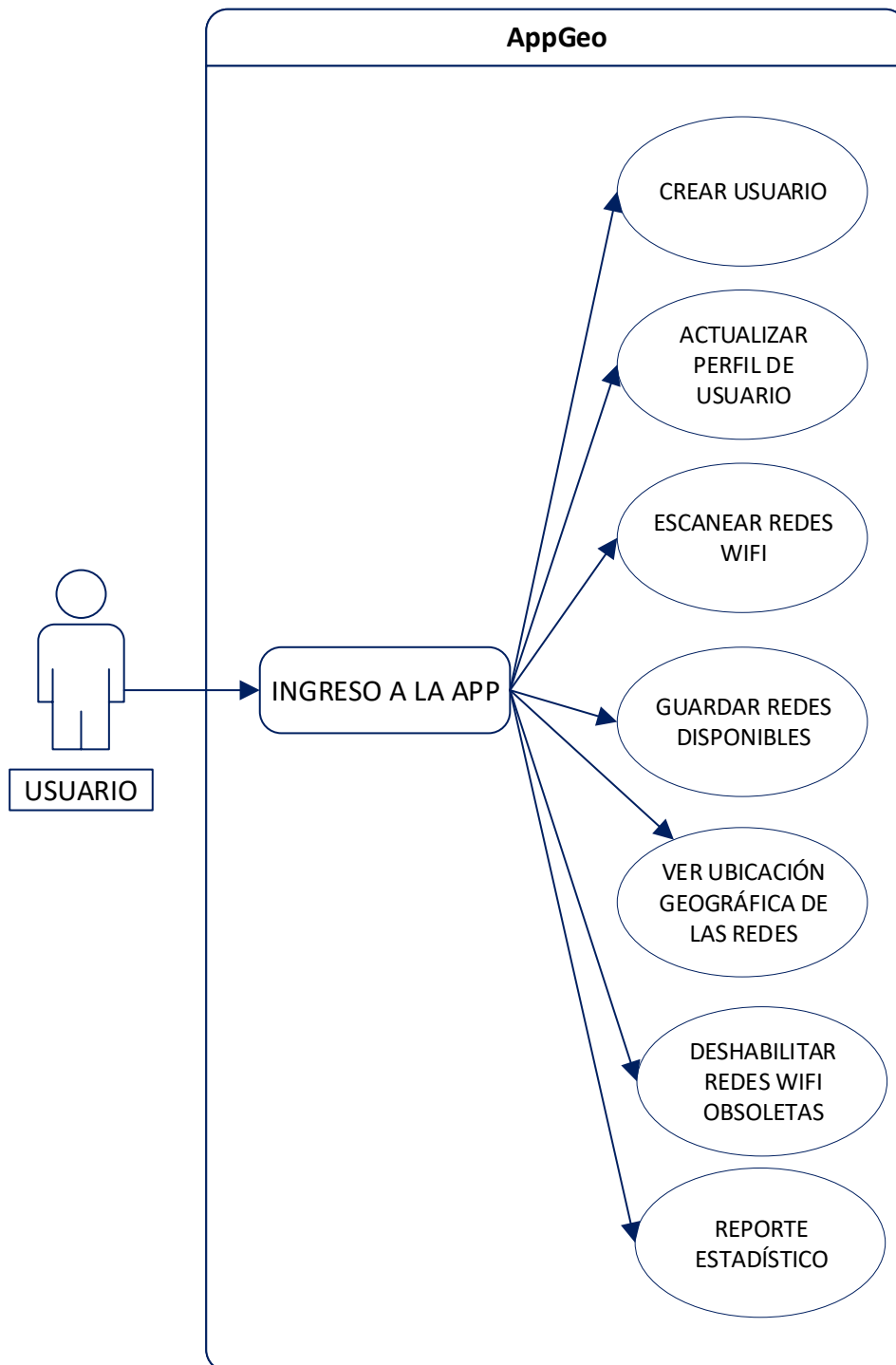
INTERFACES EXTERNAS

Interfaz de usuario: La interfaz de la aplicación móvil con el usuario estará constituida de botones, etiquetas, campos de textos, listas, entre otros. Métodos de validación de información en los formularios, lo cual deberá funcionar correctamente en Smartphone que cumplan con las características específicas.

Interfaz de software: La aplicación estará disponible para funcionar en dispositivos móviles con sistema operativo Android, que cumplan con las especificaciones físicas, técnicas y tecnológicas necesarias.

FUNCIONES

DIAGRAMA DE CASOS DE USO



REQUERIMIENTOS FUNCIONALES

Identificación del requisito:	RF01
Nombre del requerimiento:	Registro de usuario
Descripción del requerimiento:	Los usuarios deberán registrar información básica para poder iniciar sesión.
Prioridad del requerimiento:	Alta.

Identificación del requerimiento:	RF02
Nombre del requerimiento:	Autenticación de usuario
Descripción del requerimiento:	El usuario deberá autenticarse para poder acceder a la aplicación móvil, el mismo que lo podrán realizar de dos formas: autenticación por Facebook o ingresando su usuario y contraseña previamente registrados.
Prioridad del requerimiento:	Alta.

Identificación del requerimiento:	RF03
Nombre del requerimiento:	Actualización de información personal
Descripción del requerimiento:	El usuario podrá editar su información personal.
Prioridad del requerimiento:	Media.

Identificación del requerimiento:	RF04
Nombre del requerimiento:	Buscar redes WiFi disponibles
Descripción del requerimiento:	Los usuarios podrán realizar búsqueda de redes WiFi disponibles.
Prioridad del requerimiento:	Alta.

Identificación del requerimiento:	RF05
Nombre del requerimiento:	Agregar redes WiFi disponibles
Descripción del requerimiento:	Una vez realizada la búsqueda o escaneo de las redes disponibles, el usuario podrá conectarse a la red WiFi disponible.
Prioridad del requerimiento:	Alta.

Identificación del requerimiento:	RF06
Nombre del requerimiento:	Agregar redes WiFi ocultas
Descripción del requerimiento:	El usuario realizará el respectivo almacenamiento de datos de la red.
Prioridad del requerimiento:	Alta.

Identificación del requerimiento:	RF07
Nombre del requerimiento:	Vincular red al mapa
Descripción del requerimiento:	Una vez conectado a una red, automáticamente la información de la misma se guardará en la base de datos como una red privada que solo el usuario podrá visualizar, para hacer la red visible en el mapa se deberá compartir.

Prioridad del requerimiento:	Alta.
-------------------------------------	-------

Identificación del requerimiento:	RF08
Nombre del requerimiento:	Visualización de redes WiFi en el mapa
Descripción del requerimiento:	Una vez compartida la red, se la podrá visualizar en el mapa.
Prioridad del requerimiento:	Alta.

Identificación del requerimiento:	RF09
Nombre del requerimiento:	Gestionar reportes
Descripción del requerimiento:	La aplicación móvil permitirá generar reporte estadístico del monitoreo de la red.
Prioridad del requerimiento:	Alta.

REQUISITOS NO FUNCIONALES

Identificación del requerimiento:	RNF01
Nombre del requerimiento:	Interfaz de la aplicación móvil
Descripción del requerimiento:	La aplicación móvil debe tener una interfaz sencilla, ágil y sobretodo de fácil uso.
Prioridad del requerimiento:	Alta.

Identificación del requerimiento:	RNF02
Nombre del requerimiento:	Seguridad de la información
Descripción del requerimiento:	La aplicación móvil garantizará a los usuarios seguridad en cuanto a la información de la red que no estará compartida en el mapa.
Prioridad del requerimiento:	Alta.

REQUISITOS DE RENDIMIENTO

La aplicación deberá soportar un gran número de terminales con acceso simultáneo a la misma; asimismo, responder satisfactoriamente a un gran sinnúmero de peticiones, garantizando una respuesta oportuna a los usuarios.

RESTRICCIONES DE DISEÑO

El diseño de la aplicación consta de elementos básicos estrictamente necesarios, tomando en consideración que la aplicación podrá funcionar en dispositivos con características limitadas en cuanto a rendimiento y funcionalidad, por lo que la aplicación deberá ser lo suficientemente ligera sin comprometer su eficiencia.

ATRIBUTOS DEL SISTEMA

Fiabilidad: La aplicación móvil trabajará con un servidor web y base de datos basados en los principios del Cloud Computing, garantizando así su disponibilidad en todo momento.

Mantenibilidad: La aplicación móvil deberá estar documentada de manera adecuada atendiendo a cada uno de sus componentes y elementos; de igual manera procurar un código limpio en la programación.

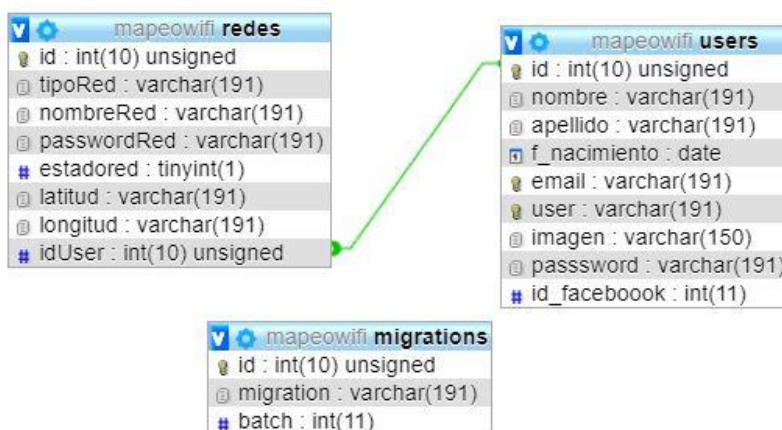
Portabilidad: La aplicación deberá ser desarrollada para funcionar sin problemas en el sistema operativo Android.

Seguridad: La aplicación móvil contará con un formulario de inicio de sesión en el que los usuarios se autenticarán mediante un nombre de usuario y contraseña.

APÉNDICES

BASE DE DATOS

Es de tipo relacional, estará compuesta por dos tablas; una tabla denominada *users*, que almacenará información de los usuarios que se registren o logueen en la aplicación; y, otra denominada *redes*, que contendrá la información de las distintas redes.



En la siguiente figura se muestra el modelo entidad-relación de la base de datos. La relación entre ambas tablas es de uno a varios y establecerá qué usuario comparte la red en la aplicación.

