Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación
Aplicaciones de Tecnología en Internet

Desarrollo de Solución Tecnológica para Centros Comerciales con Geolocalización como Funcionalidad Base

Trabajo Especial de Grado presentado ante la ilustre
Universidad Central de Venezuela
por los Bachilleres:

Johan J. Quintero A.
C.I.: 22.668.628
johandym50@gmail.com

Luinel J. Andrade G.
C.I.: 22.030.199
luinel1393@gmail.com

Para optar al título de Licenciado en Computación

Tutores:
Prof. Eric Gamess
Prof. Antonio Russoniello

Caracas, noviembre 2017

Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación
Laboratorio de Comunicación y Redes

ACTA DE VEREDICTO

Quienes suscriben, miembros del jurado designado por el Consejo de Escuela de Computación, para examinar el Trabajo Especial de Grado presentado por los Bachilleres Johan Quintero (C.I. V-22.668.628) y Luinel Andrade (C.I. V-22.030.199), con el título "**Desarrollo de Solución Tecnológica para Centros Comerciales con Geolocalización como Funcionalidad Base**", a los fines de cumplir con el requisito legal para optar al título de Licenciado en Computación, dejan constancia de lo siguiente:

Leído el nombrado trabajo por cada uno de los miembros del jurado, éste fijó el día miércoles 08 de noviembre de 2017, para que sus autores lo defendiera en forma pública, en la Sala de Internet II, Laboratorio LACORE, de la Escuela de Computación, mediante una exposición oral de su contenido, luego de la cual respondieron satisfactoriamente a las preguntas que les fueron formuladas por el jurado, todo ello conforme a lo dispuesto en la Ley de Universidades y demás normativas vigentes de la Universidad Central de Venezuela.

Finalizada la defensa pública del Trabajo Especial de Grado, el jurado decidió aprobarlo.

Es de aclarar que el Profesor Eric Gamess se encuentra de permiso fuera del país. Por esta razón, el Profesor Robinson Rivas, director de la Escuela de Computación, firma el presente documento en su lugar.

En fe de lo cual se levanta el presente acta en Caracas a los 08 días del mes de noviembre del año 2017, dejando constancia que el Profesor Antonio Russoniello actuó como coordinador del jurado.

por el Prof. Eric Gamess
Tutor

Prof. Antonio Russoniello
Tutor

Prof. Dedaniel Urribarri
Jurado Principal

Prof. Antonio Silva
Jurado Principal

REPUBLICA DE VENEZUELA
Facultad de Ciencias
Escuela de Computación
Universidad Central - Caracas

Agradecimientos

Todo la dedicación y el esfuerzo empleado en este Trabajo Especial de Grado y en toda mi carrera universitaria fue apoyado y acompañado por muchas personas a quienes quisiera agradecer por su apoyo incondicional, compañía, comprensión, inspiración y cariño para lograr cumplir mis metas y la culminación exitosa de este trabajo y de mi carrera universitaria.

Principalmente a Dios por bendecirme cada día en todos mis proyectos. Por darme la fortaleza para afrontar los momentos difíciles y seguir luchando por mis sueños.

A mi mamá Julia Abreu por su inmenso amor y por siempre apoyarme, acompañarme, aconsejarme y por todo el esfuerzo que ha empleado para inspirarme a cumplir mis metas. Por siempre preocuparse y estar pendiente de mí y por ser, más que mi mama, mi mejor amiga. Gracias por demostrarme siempre estarás para mí en todo momento ¡Te amo mamá!

A mi papá José Quintero por todo el apoyo que me ha brindado siempre, por sus consejos, su guía, su dedicación y esfuerzo, y por siempre darme el ejemplo a ser responsable y un hombre de bien, te quiero mucho papá.

A mis hermanos y hermana Adriana, Cheo, Gabriel y Miguel, por estar siempre presentes en todo momento, brindándome su compañía y muchos momentos de alegría y diversión con los que me han llenado de recuerdos inolvidables. Gracias por demostrarme el verdadero significado de la palabra “hermano” y “hermana” ¡Los quiero mucho!

A mi novia Oriana Fernández, por su compañía, apoyo, ayuda, comprensión y por el gran amor que me demuestra a diario. Gracias por ser mi fiel compañera a lo largo de estos casi 3 años juntos, por tus atenciones y por siempre alegrarme y levantarme el ánimo en los momentos de mayor estrés y frustración. Gracias por demostrarme que juntos podemos lograr cualquier meta. Por todo eso y mucho más ¡Gracias mi amor! ¡Te amo muchísimo!

A mis sobrinos Gabriel y Alejandro, por sus locuras que siempre me alegran, me hacen reír y hacen que disfrute cada momento a su lado. Espero siempre poder ser un ejemplo para ustedes, para que cumplas sus metas y todo lo que se propongan ¡Los quiero mucho loquillos!

A mi compañero Luinel Andrade, que más que un compañero ha sido un amigo, socio y hermano, a lo largo de nuestra carrera universitaria. Gracias por todo el apoyo que siempre me has brindado, por tu esfuerzo y dedicación en todos nuestros trabajos juntos y por tu amistad.

A la Ilustre Universidad Central de Venezuela, la mejor universidad del país, por ser mi segunda casa y por brindarme tantos momentos inolvidables y la oportunidad de aprender y crecer en lo personal y en lo profesional. Me siento sumamente orgulloso de ser un ucevista. A la Facultad de Ciencias y a todos los profesores que me han brindado sus conocimientos, consejos y experiencias que me han ayudado mucho. En especial, gracias a mis tutores Eric Gamess y Antonio Russoniello, por su guía, orientación y apoyo, durante toda la investigación.

Al personal del Centro Comercial Millennium Mall, por su colaboración, en especial al personal de la Gerencia de Mercadeo, por el apoyo que nos brindaron, por permitirnos el acceso a sus instalaciones, por creer en el proyecto y por darnos la oportunidad para realizar este proyecto.

A todos mis demás familiares, primos y primas, tíos y tías, abuelos y abuelas, etcétera, y a todos mis amigos y amigas. Que han sido importantes para poder ser la persona que hoy soy.

Johan Quintero.

Agradecimientos

A lo largo de mi carrera universitaria he recibido el apoyo y consejo de una gran cantidad de personas. Con el aporte de cada uno de ellos he podido llegar a dónde estoy ahora, he podido cumplir objetivos y alcanzar metas que han sido posibles gracias al granito de arena que cada uno ha dado. Dedico el presente Trabajo Especial de Grado a cada uno de ellos, les estoy profundamente agradecido.

Agradezco principalmente a Dios por todas sus bendiciones, por todas las oportunidades que ha puesto en mi camino y la fortaleza que me brinda para enfrentar los retos del día a día.

A mis padres Milagro Gascón y Nelson Andrade, por siempre haberme apoyado y por haberme entregado a lo largo de mi vida lo mejor de ellos sin esperar nada a cambio. Gracias por el amor, el esfuerzo y el respaldo que me han dado. En todos mis logros, en todos mis triunfos, siempre se verán reflejados, les dedico cada una de mis victorias.

A mi hermano Alejandro Andrade, a mi abuela Elena, a cada uno mis tíos, a cada uno de mis primos. Gracias por siempre estar atentos, preocupados por mí y por hacerme conocer el valor de lo que significa la palabra familia.

A Nicole Torres, gracias por su compañía, por todo el amor que me da, por todas las atenciones que tiene conmigo, por su comprensión, por todo su apoyo. Ella conoce muy bien cómo ha sido el camino recorrido y el que falta por recorrer, me siento feliz de que sea mi fiel compañera en este viaje.

A la Ilustre Universidad Central de Venezuela, que tanto le ha dado al país, orgulloso me siento de formar parte de ésta casa de estudios y de representarla. A la Facultad de Ciencias y a todo el personal que allí labora, incluyendo a todos los profesores de los que he tenido la oportunidad de recibir clases, por sus enseñanzas, por sus conocimientos y por su sabiduría. En especial, gracias mis tutores Eric Gamess y Antonio Russoniello, por la guía y orientación brindada a lo largo de la investigación.

A todos mis amigos y compañeros que conseguí a lo largo de la carrera, juntos hemos podido superar los obstáculos, gracias por las ayudas que en sus respectivos momentos me han brindado. Especial mención a mi compañero de tesis Johan Quintero que, desde el primer semestre, desde los primeros días de clases hasta ahora, siempre ha estado allí, gracias por creer en mí, por tu esmero, por tu esfuerzo.

A Rocco, que a pesar de que no entienda estas palabras, le agradezco su compañía en las noches de estudios y de meditación, en aquellos momentos en que todo me preocupaba y sólo sentía la necesidad de caminar y pensar, él siempre estuvo.

A todo el personal del Centro Comercial Millennium Mall, por su colaboración, en especial al personal de la Gerencia de Mercadeo, por el apoyo mostrado hacia mi compañero y hacia mi persona, por permitirnos el acceso a sus instalaciones en los momentos en los que lo necesitábamos, por creer en este proyecto, por darnos el espaldarazo para iniciar este sueño.

Luinel Andrade.

Resumen

Título:
Desarrollo de Solución Tecnológica para Centros Comerciales con Geolocalización como Funcionalidad Base.

Autores:
Johan J. Quintero A.
Luinel J. Andrade G.

Tutores:
Prof. Eric Gamess
Prof. Antonio Russoniello

En los últimos años los dispositivos móviles han evolucionado y producido cambios indudables en la sociedad y en la manera en que transcurre el día a día, en gran parte, gracias a distintas funcionalidades tecnológicas que incluyen y las diversas aplicaciones que están a disposición de los usuarios. En el caso de los centros comerciales, este tipo de tecnologías representan grandes oportunidades de negocio y una manera de mejorar la experiencia de sus usuarios dentro de las instalaciones. Ese tipo de edificaciones suelen ser de bastante amplitud, de varios niveles y con una gran cantidad de comercios, servicios, en donde los visitantes pueden verse con dificultades a la hora de querer saber cuáles tiendas son las que hay en el lugar, determinar cuáles tiendas y servicios son los que cumplen con las características o especificaciones que buscan, conocer la ubicación de cada comercio o cómo llegar a ellos.

En este Trabajo Especial de Grado se estudió y contempló diferentes tecnologías, herramientas y enfoques para el desarrollo de una solución tecnológica para centros comerciales que tenga como funcionalidad base un sistema de geolocalización en los espacios interiores del edificio. Dicha solución tecnológica incluye una aplicación móvil dirigida a dispositivos móviles con sistema operativo Android desarrollada usando el enfoque de desarrollo nativo y una aplicación web para la gestión de los datos, contenidos y configuraciones propias de la aplicación móvil que serán obtenidos siguiendo el modelo cliente-servidor a través de un API privado que se desarrolló también para el proyecto. Entre las distintas funcionalidades que ofrecen las aplicaciones están el poder visualizar el directorio de los comercios y servicios, descripción de los mismos y su ubicación en los mapas de la edificación, promociones propias del centro comercial o de los comercios y los eventos que se puedan realizar dentro del mismo, entre otras, destacando el ya mencionado sistema de geolocalización en interiores, el cual fue implementado haciendo uso de la tecnología WiFi y de los distintos Access Points instalados en el lugar, por medio del cual los usuarios se puedan ubicar dentro del centro comercial, localizar las tiendas y los servicios de su interés, y recibir indicaciones de cómo llegar a ellos.

Finalmente, como resultado final de la investigación y del proceso de desarrollo se puede decir que se realizaron de manera satisfactoria los prototipos funcionales de las aplicaciones planteadas en un comienzo, tanto a nivel móvil enfocada a los visitantes del centro comercial, como a nivel web enfocado al personal administrativo del mismo.

Palabras Claves: Aplicación Móvil, Aplicación Web, Geolocalización, Centro Comercial, WiFi, Access Points.

Tabla de Contenido

Índice de Figuras

Índice de Tablas

1. Introducción

A medida que pasa el tiempo, la tecnología va evolucionando rápidamente, aumentando así las soluciones tecnológicas que buscan facilitar el día a día de las personas. Es un hecho que tecnologías como la de los dispositivos móviles y las aplicaciones web forman parte de las actividades cotidianas de las personas tanto en el ámbito personal como en el laboral e influencian aspectos de la vida tan variados como las relaciones personales, el entretenimiento, la educación y las actividades económicas, entre otras.

Los dispositivos móviles evolucionan y producen cambios en la sociedad a través de las distintas funcionalidades tecnológicas que incluyen diversas aplicaciones que están a disposición de los usuarios, generando nuevas tendencias y modelos de comportamiento que cada vez se hacen más relevantes para las organizaciones y empresas de diversos sectores de actividad, ganando grandes oportunidades de negocio si se adaptan a ellos. Por eso los centros comerciales deben apostar a no sólo aumentar el número de sus ventas o visitantes, sino mejorar la experiencia de los visitantes dentro de sus instalaciones, brindar un servicio que permita que la experiencia sea más personal e inmediata, generar un recuerdo, posicionarse y estar más cerca de sus consumidores, incorporando los recursos tecnológicos que son cotidianos para ellos.

Los centros comerciales por lo general son estructuras de bastante amplitud, con mucho espacio, de varios niveles y tienen una gran cantidad de tiendas, servicios y numerosos puestos de estacionamiento, en donde los visitantes pueden verse con dificultades a la hora de querer saber cuáles tiendas son las que hay en el lugar, conocer la ubicación de cada local o cómo llegar a ellos, determinar cuáles tiendas y servicios son los que cumplen con las características o especificaciones que buscan y mucho más. El objetivo principal de esta investigación es contemplar y analizar diversas tecnologías que permitan el desarrollo de una aplicación móvil que permita al usuario visualizar información de un centro comercial, directorio de tiendas y servicios que posee, consultar mapas, contar con un sistema de geolocalización en interiores e incluso ver las diversas promociones y eventos que hay en el mismo, además del desarrollo de una aplicación web que le permita al personal administrativo del centro comercial gestionar los datos, contenidos y configuraciones que desean mostrar a sus visitantes en la aplicación móvil.

Este trabajo se encuentra estructurado de la siguiente manera:

Capítulo 2: Contiene el planteamiento formal del problema a estudiar, contemplando los objetivos que se desean cubrir, y el alcance del proyecto.

Capítulo 3: Describe las características de los dispositivos móviles y diversos sistemas operativos que pueden tener estos dispositivos.

Capítulo 4: Describe al sistema operativo Android y su arquitectura, así como los componentes de las aplicaciones de este sistema operativo y elementos propios de dicha plataforma.

Capítulo 5: Describe los distintos tipos de aplicaciones móviles y sus enfoques de desarrollo.

Capítulo 6: Presenta una serie de herramientas y tecnologías que son importantes y necesarias para el desarrollo de la aplicación móvil en la que se enfoca el estudio.

Capítulo 7: Describe cuatro aplicaciones que tienen relación directa con la investigación, las cuales dan soporte a conceptos y aspectos importantes a considerar en la creación de una aplicación móvil dirigida a centros comerciales.

Capítulo 8: Explica la metodología a utilizar y cómo influye en el desarrollo de un proyecto de software.

Capítulo 9: Explica en detalle las diferentes etapas del proceso de desarrollo de la solución tecnológica mediante la metodología utilizada.

Capítulo 10: Presenta conclusiones y recomendaciones obtenidas a partir del Trabajo Especial de Grado realizado.

2. El Problema

En el presente capítulo se describe y plantea el problema observado en los centros comerciales y en sus visitantes durante sus períodos de permanencia dentro de los mismos, junto con la justificación por la cual esos problemas pueden ser solucionados con la implementación de aplicaciones y uso de herramientas tecnológicas, los objetivos que se desean alcanzar, y el alcance del proyecto.

2.1. Planteamiento del Problema

Actualmente, son muchos los usuarios que frecuentan los centros comerciales en cualquier parte del mundo, ya que son lugares que proveen a las personas entretenimiento, tiendas en donde pueden comprar diferentes artículos, ferias de comidas, restaurantes, baños públicos, bancos, cines y otros servicios que hacen de estos espacios lugares cada vez más frecuentados por las personas.

Como ya se mencionó, los centros comerciales cuentan con muchas tiendas y muchas otras instalaciones y servicios, y por lo general, estos son bastante grandes, con mucho espacio y varios niveles, además de una gran cantidad de puestos de estacionamiento. Debido a la amplitud de los mismos y a su gran cantidad de locales, surge uno de los principales problemas que viven los usuarios al momento de visitar un centro comercial, que es, conocer la ubicación de cada local o servicio del mismo y cómo llegar a ellos. Además, tomando en cuenta el problema ya mencionado, también se presenta la situación en las que los usuarios buscan un tipo específico de artículos, servicios o promociones, y no conocen los locales dentro del centro comercial visitado que cuentan con las cualidades o especificaciones que buscan, por lo que se ven en la necesidad de recorrer el centro comercial buscando aquellos que son de su interés, lo que muchas veces conlleva a que realicen recorridos que podrían considerarse innecesarios o se pierdan dentro de las instalaciones sin tener referencias de dónde están ubicados, ocasionando así pérdida de tiempo y generando a su vez gran frustración en el usuario.

2.2. Justificación del Problema

Surge la necesidad de desarrollar una solución tecnológica que pueda resolver la problemática anteriormente planteada y que sea accesible por todos los usuarios que visitan un centro comercial. Es por eso, y apoyándose de que hoy en día existe un gran número de personas que tienen *smartphones* y *tablets*, que se propone una aplicación móvil que permita a los visitantes de un determinado centro comercial usar su dispositivo móvil para consultar el directorio de tiendas y servicios, ubicar los locales de su interés, consultar su posición en el edificio, así como también otras funciones que enriquecen su experiencia como visitante, como por ejemplo el poder consultar información propia del centro comercial, las promociones de las tiendas o los eventos que se realizarán en el mismo. Adicionalmente, se requiere desarrollar una aplicación web a ser usada por el personal administrativo del centro comercial para que puedan gestionar los datos, contenidos y configuraciones que se mostrará a los visitantes del centro comercial que posean la aplicación móvil.

2.3. Objetivos

En esta sección, se definen los objetivos que se desean alcanzar por medio del desarrollo del trabajo propuesto.

2.3.1. Objetivo General

Desarrollar una solución tecnológica para centros comerciales que permita visualizar y gestionar la información del mismo, datos de contacto, directorio de comercios y servicios que posee, mapas, promociones, eventos y un sistema de geolocalización como funcionalidad base mediante el desarrollo de dos aplicaciones.

2.3.2. Objetivos Específicos

- Definir los requerimientos funcionales y no funcionales en el desarrollo de una aplicación móvil para centros comerciales y de una aplicación web para el sistema de gestión de contenidos.
- Desarrollar una aplicación móvil nativa para el sistema operativo Android que permita al usuario visualizar la información, ubicación geográfica y contactos de un centro comercial, el directorio de tiendas y servicios que posee, las promociones y eventos, mapas con ubicación de las tiendas, además de un sistema de geolocalización básico que le permita al usuario ubicarse dentro del mismo y recibir indicaciones de cómo llegar a un determinado lugar.
- Desarrollar una aplicación web para la gestión de los datos, contenidos y configuraciones propias de la aplicación móvil.
- Implementar un mecanismo de comunicación que permita a la aplicación web gestionar los datos, contenidos y configuraciones en una base de datos centralizada para que sean accedidos desde la aplicación móvil.
- Realizar pruebas de depuración, aceptación, análisis y correcciones de la aplicación móvil y de la aplicación web.

2.4. Alcance

La aplicación móvil nativa a desarrollar para dispositivos móviles con sistema operativo Android debe permitir a los usuarios visualizar información propia del centro comercial, la ubicación geográfica del mismo y datos de contacto, así como también ver el directorio de los locales y servicios que posee, descripción de los mismos y su ubicación en los mapas de la edificación, promociones propias del centro comercial o de los locales y los eventos que se puedan realizar dentro del mismo. Además, la aplicación ofrecerá un sistema de geolocalización en interiores básico apoyándose en la tecnología WiFi, por medio del cual los usuarios se puedan ubicar dentro del centro comercial, localizar las tiendas o servicios de su interés y recibir indicaciones de cómo llegar a ellos.

El desarrollo de esta aplicación llegará a una fase beta, fase que generalmente representa la primera versión completa de la aplicación, la cual posiblemente sea inicialmente inestable pero que una vez agregadas las características del producto que fueron establecidas, procederá a realizarse pruebas en la misma para detectar y corregir errores.

Se estima que, en un principio, la aplicación móvil recibirá el nombre de “Millennium Maps”, debido a que se tomará como caso de estudio para su desarrollo al centro comercial Millennium Mall ubicado en la Av. Rómulo Gallegos, Urb. Los Dos Caminos, Caracas, Venezuela.

Además, también se deberá desarrollar una aplicación web de administración que se prevee que llevará por nombre “Millennium Maps Administrativo” que sirva para gestionar parámetros e información en la base de datos con la que se comunicará la aplicación móvil y de donde obtendrá la información mostrada en la misma.

3. Dispositivos Móviles

Una gran cantidad de dispositivos electrónicos se clasifican actualmente como dispositivos móviles, por ejemplo: teléfonos celulares, videoconsolas portátiles, agendas digitales, calculadoras, cámaras fotográficas digitales, entre otros. Con esta diversidad de dispositivos, resulta complicado determinar cuáles características deben poseer para que sean considerados como dispositivos móviles. Sin embargo, podrían mencionarse algunos aspectos esenciales que deben cumplir:

- Son aparatos pequeños. La mayoría de estos aparatos se pueden transportar en el bolsillo del propietario o en un pequeño bolso.
- Tienen capacidad de procesamiento.
- Tienen conexión permanente o intermitente a una red.
- Tienen memoria (tarjetas MicroSD, flash, etc.).
- Normalmente se asocian al uso individual de una persona.
- Tienen una alta capacidad de interacción mediante la pantalla o el teclado.

Existen cuatro factores que diferencian a los dispositivos móviles de otros dispositivos: movilidad, tamaño reducido, comunicación inalámbrica e interacción con las personas.

El presente trabajo se enfocará en dos tipos de dispositivos móviles que son los teléfonos móviles (específicamente los *smartphones*) y en las *tablets,* y se hará uso generalizado del término dispositivo móvil para referirse a los mismos.

3.1. Teléfonos Móviles

El teléfono móvil es un dispositivo inalámbrico electrónico basado en la tecnología de ondas de radio, que tiene la misma funcionalidad que cualquier teléfono de línea fija. Su principal característica es su portabilidad, ya que la realización de llamadas no es dependiente de ningún terminal fijo y no requiere ningún tipo de cableado para llevar a cabo la conexión a la red telefónica [1]. Su principal función es la comunicación de voz, pero su rápido desarrollo llevó a que se incorporarán funciones adicionales como SMS (*Short Message Service* o también conocido como mensajería instantánea), calculadora, agenda, juegos, reproducción de audio y video. Adicionalmente, han mejorado en muchos aspectos como las baterías más pequeñas y de mayor duración, y el soporte de pantallas más nítidas y de colores, entre otras cosas. Con el paso de los años y ante la introducción de los *smartphones*, los teléfonos móviles con las características mencionadas anteriormente pasaron a ser conocidos como *feature phones* o teléfonos básicos (Figura 3.1).

Figura 3.1: Feature Phones

Teniendo en cuenta la definición de teléfono móvil presentada anteriormente, en la última década ha surgido el concepto de *smartphones* o teléfonos inteligentes, término que se ha venido generalizando entre las personas para referirse a los teléfonos móviles. Un *Smartphone* es un dispositivo electrónico que funciona como un teléfono móvil básico con características similares a las de un computador personal y con una conectividad mayor [1], centrando gran parte de su operatividad en la conexión a Internet vía WiFi o redes 2G, 3G o 4G. Una característica importante es que casi todos estos dispositivos permiten al usuario instalar programas adicionales o aplicaciones que aumentan aún más la potencialidad y productividad de los mismos en tareas y actividades de diferentes índoles. Algunas funcionalidades comunes con los que cuentan son el acceso a Internet, cámara fotográfica, videocámara, reproductor de audio y video, agenda, administración de contactos, correo electrónico, acelerómetro, giroscopio, sistemas de geolocalización como GPS (Global Positioning System), capacidad de leer documentos, videojuegos, pantalla táctil y mucho más.

En la Figura 3.2 se puede apreciar uno de los *smartphones* más modernos y potentes hasta la fecha desarrollado por la compañía Samsung, el modelo Galaxy S7 edge.

Figura 3.2: Ejemplo de un Smartphone

Para tener un estimado de la cantidad de personas (en billones) que utilizan smartphones a nivel mundial, en la Figura 3.3 se muestra los resultados de un estudio disponible en el portal especializado en estadísticas *Statista*[1] acerca del tema.

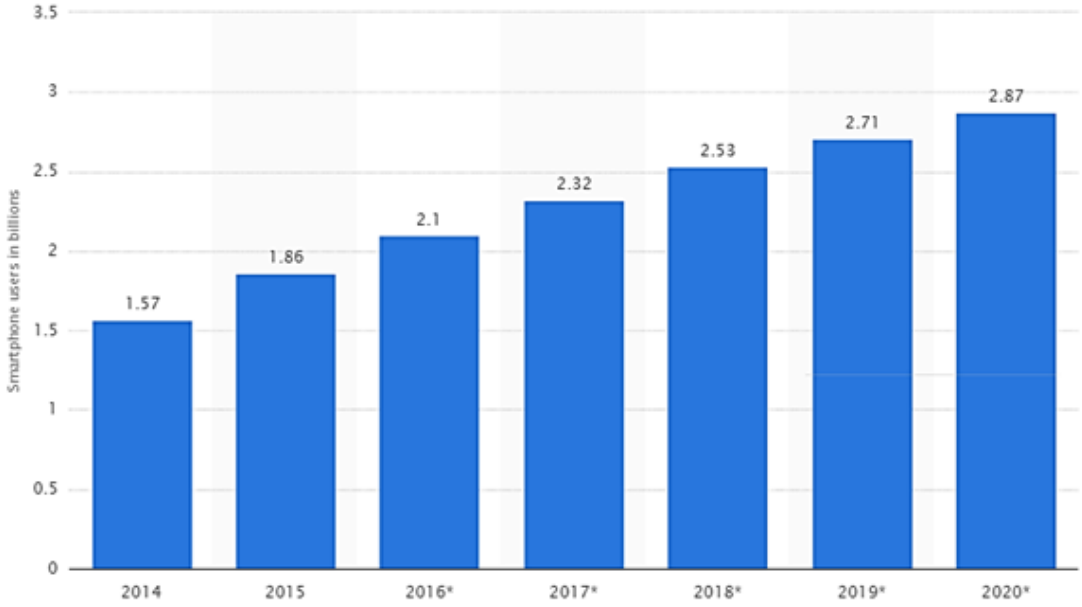

Figura 3.3: Número de Usuarios de Smartphones en todo el Mundo

3.2. Tablets

Las *Tablets* o Tabletas en español (Figura 3.4), son computadores portátiles que se asemejan en características y aspecto a los smartphones pero de mayor tamaño, tienen integrada una pantalla táctil para interactuar con el dispositivo sin necesidad de teclado físico ni de mouse, haciendo uso principalmente de los dedos mediante gestos e incluso se puede utilizar lápices especiales para este tipo de pantallas conocidos comúnmente como Lápiz Stylus.

Estos dispositivos tienen capacidades de procesamiento de información y navegación en Internet similares o ligeramente inferiores a la de un computador portátil. Algunas de sus principales características además de la ya mencionada pantalla táctil, es que cuentan con batería de larga duración (en el orden de las 8 horas), son de bajo peso (alrededor de los 500 gramos) y tienen un tamaño que va aproximadamente entre las 7 y 11 pulgadas, lo que facilita su movilidad. La gran mayoría de los modelos cuentan con los mismos sistemas operativos que tienen los *smartphones*, sus funcionalidades están íntimamente asociadas al perfil de usuario y permiten conectarse a Internet a través de WiFi y redes 3G [2].

[1] https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide

Figura 3.4: Ejemplo de Tablet

3.3. Sistemas Operativos de los Dispositivos Móviles

No existe una definición universal para sistemas operativos (SO), pero podría decirse que es un programa que administra el hardware de un computador. También proporciona las bases para los programas de aplicación y actúa como un intermediario entre el usuario y el hardware del computador [3].

Al igual que los computadores, cada uno de los dispositivos móviles tiene su respectivo sistema operativo, éste se ejecuta al encender dicho dispositivo encargándose de administrar los recursos del sistema, tanto de software (programas e instrucciones) como de hardware (memoria, pantalla, cámara, etc.) permitiendo así la comunicación entre el equipo y el usuario.

En el caso específico de los dispositivos móviles, teniendo en cuenta las características que los diferencian de otros sistemas computacionales, los sistemas operativos móviles están enfocados en la movilidad, la conectividad inalámbrica, formatos multimedia y en la administración de forma óptima del procesamiento, almacenamiento y el consumo de la energía [4]. Algunas características destacables de un sistema operativo móvil actual son:

- Kernel unificado.
- Construido por capas.
- Multiproceso y multitarea.
- Soporte a diferentes pantallas.
- Soporte multilenguaje.
- Multihilo.
- Conectividad inalámbrica.
- Administración del hardware.
- Administración de aplicaciones.
- Navegación web.
- Capacidad de adaptación.
- Reinvención y mejoramiento.
- Personalizable.
- Multiusuario.
- Inteligente.

Como se observa en una de las características, los sistemas operativos móviles se encuentran construidos por capas, las cuales son [5]:

- **Kernel:** Es la capa de software que permite el acceso a los diferentes elementos de hardware que conforman el dispositivo móvil. Además, se encarga de brindar diferentes servicios a las capas superiores como los controladores de hardware, gestión de procesos, sistemas de archivos, además del acceso y administración de la memoria del sistema.
- **Middleware:** Esta capa es el conjunto de módulos que permite que las aplicaciones diseñadas y escritas para tales plataformas puedan ser ejecutadas. Permite ejecutar servicios muy importantes para que otras aplicaciones, en capas superiores de la jerarquía, también puedan ejecutarse. Entre los servicios que presta esta capa se puede mencionar los motores de comunicaciones y mensajería, funciones de seguridad, servicios para la gestión de diferentes aspectos del móvil, intérpretes de páginas Web, entre otros.
- **Entorno de ejecución de aplicaciones:** Esta capa provee todos los elementos necesarios para la creación y desarrollo de software a los programadores, es decir contiene elementos que serán de gran ayuda a los mismos, en el momento de desarrollar aplicaciones compatibles con ese sistema operativo. Entre los servicios que los programadores pueden encontrar, se destacan un gestor de aplicaciones y una serie de APIs (Application Programming Interface) abiertas.
- **Interfaz de Usuario:** es el elemento del teléfono que usualmente se utiliza para interactuar con el equipo. Incluye todos los elementos gráficos que harán posible el uso del dispositivo: botones, menús, pantallas y listas, entre otros. En el dispositivo se encuentran una serie de aplicaciones que vienen incorporadas en el equipo, y que se encargan de tareas tales como menús, marcador de números de teléfono y demás.

A continuación, se presentan los sistemas operativos más comunes para dispositivos móviles.

3.3.1. iOS

Es un sistema operativo desarrollado por Apple Inc. para sus dispositivos como: iPhone, iPod Touch y iPad, y actualmente se encuentra en su undécima versión. Este sistema operativo no tiene permitido su instalación en dispositivos de otras marcas, solo en los dispositivos antes mencionados. La primera versión de iOS fue presentada en 2007 junto al primer iPhone. Este sistema operativo se deriva de MacOS, que también fue desarrollado por Apple Inc. para su gama de computadores Macintosh desde el año 2002. Es un sistema operativo tipo Unix que cuenta con cuatro (4) capas de abstracción, las cuales son:

- Capa del núcleo del sistema operativo.
- Capa de “Servicios Principales”.
- Capa de “Medios”.
- Capa de “Cocoa Touch”.

3.3.2. Android

Es un sistema operativo y una plataforma de software, basado en Linux para teléfonos móviles, aunque también es utilizado para otros dispositivos como *tablets* y aunque no muy habitual, *wearables devices*[2], reproductores de música e incluso *netbooks*. Android permite programar en

[2] Un *wearable device* es una prenda de vestir o complemento inteligente que incorpora elementos tecnológicos, electrónicos, etc. como por ejemplo los relojes inteligentes o pulseras de actividad.

un entorno de trabajo (framework) de Java, aplicaciones sobre una máquina virtual Dalvik que es una variación de la máquina virtual de Java con compilación en tiempo de ejecución [6].

Fue desarrollado inicialmente por Android Inc., empresa que fue adquirida posteriormente por Google en el año 2005. Está escrito en varios lenguajes, según diferentes partes del mismo, como: C (núcleo), Java (User Interface) y C++ (algunas bibliotecas de terceros).

3.3.3. Windows Phone

Windows Phone es un sistema operativo móvil desarrollado por Microsoft para sus teléfonos inteligentes que fue lanzado por primera vez en octubre de 2010. Fue presentado como el sucesor de Windows Mobile 5, 6, y Zune, y cuenta con una serie de cambios. A diferencia de Windows Mobile, Windows Phone está dirigido al mercado de consumo más que al mercado empresarial [7].

Debido a la fragmentación de sus sistemas operativos, Microsoft dio de baja a Windows Phone en el primer trimestre de 2015, para enfocarse en el sistema operativo móvil denominado Windows 10 Mobile el cual es una edición del sistema operativo Windows 10, disponible para *smartphones* y *tablets.*

3.3.4. Otros

Además de los ya mencionados, existen otros sistemas operativos que no tienen tanta popularidad y abarcan pocas ventas en el mercado actual. Algunos de estos son:

- **BlackBerry OS:** es un sistema operativo de código cerrado incluido en la gama de teléfonos móviles de la compañía canadiense BlackBerry antiguamente llamada Research In Motion (RIM) *y* que viene incorporado en los móviles fabricados por la empresa, que también lleva el apelativo BlackBerry, seguido por el modelo correspondiente[3]. BlackBerry OS cuenta con un núcleo basado en la máquina virtual de Java. Este sistema operativo se encuentra escrito en Java y C++.
- **Firefox OS:** es un sistema operativo móvil, basado en HTML5 con núcleo Linux, de código abierto para varias plataformas. Firefox OS está impulsado por la Fundación Mozilla, responsable del navegador Firefox y del cliente de correo electrónico Thunderbird. Está compuesto por 3 capas, las cuales son: Gonk (el núcleo), Gecko (el motor de renderizado Web) y Gaia (la interfaz de usuario).
  A finales de 2015, la Fundación Mozilla dio por concluido el desarrollo del sistema Firefox OS para teléfonos móviles y anunció el fin de su desarrollo. El sistema ya desarrollado será adaptado a otros tipos de dispositivos. Los principales obstáculos encontrados al desarrollo del sistema fueron de tipo comercial. El sistema no se vendió y los costos excedieron los beneficios[4].

[3] http://culturacion.com/blackberry-sistema-operativo-movil-de-rim

[4] http://www.technewsworld.com/story/Mozilla-Gives-Up-Firefox-Phone-Ambitions-82862.html

4. Sistema Operativo Android

Si bien ya se dio una breve descripción del sistema operativo Android, en esta sección se estudiará un poco más en profundidad debido a que será utilizado para el desarrollo de la aplicación móvil. Las razones por las cuales se eligió este sistema operativo se encontrará especificado más adelante.

Android es un sistema operativo además de una plataforma de software basada en el núcleo de Linux [8]. Ha sido diseñado para dispositivos móviles como, por ejemplo: *smartphones*, *tablets* y *wearables devices*. Uno de los principios con el que fue desarrollado es brindar a los desarrolladores la creación de aplicaciones móviles que aprovechan al máximo el uso de todas las herramientas que los dispositivos como los mencionados pueden ofrecer.

Es de código abierto, y además puede ser libremente ampliado para incorporar nuevas tecnologías de vanguardia que van surgiendo. La plataforma continuará evolucionando a medida que la comunidad de desarrolladores trabajando juntos puedan crear aplicaciones móviles innovadoras [9].

Fue desarrollado por Android Inc., empresa que fue comprada posteriormente por Google en el año 2005, pero no fue hasta el año 2008 cuando empezó a popularizarse, gracias a que el consorcio Open Handset Alliance formado por numerosas empresas de desarrollo de hardware, software y telecomunicaciones, decidió unirse al proyecto, promocionar el software libre y desarrollar estándares abiertos para dispositivos móviles. Google ha sido quien ha publicado la mayor parte del código fuente del sistema operativo, gracias a la ASF (Apache Software Foundation), que es una fundación que da soporte a proyectos de software de código abierto.

4.1. Arquitectura de Android

Android es una pila de software de código abierto basado en Linux creada para una variedad amplia de dispositivos. En la Figura 4.1 tomada de [10] se muestran los componentes principales de la arquitectura Android, que serán explicados consecutivamente partiendo de la información disponible en la documentación oficial que se encuentra en la página web de Android para desarrolladores.

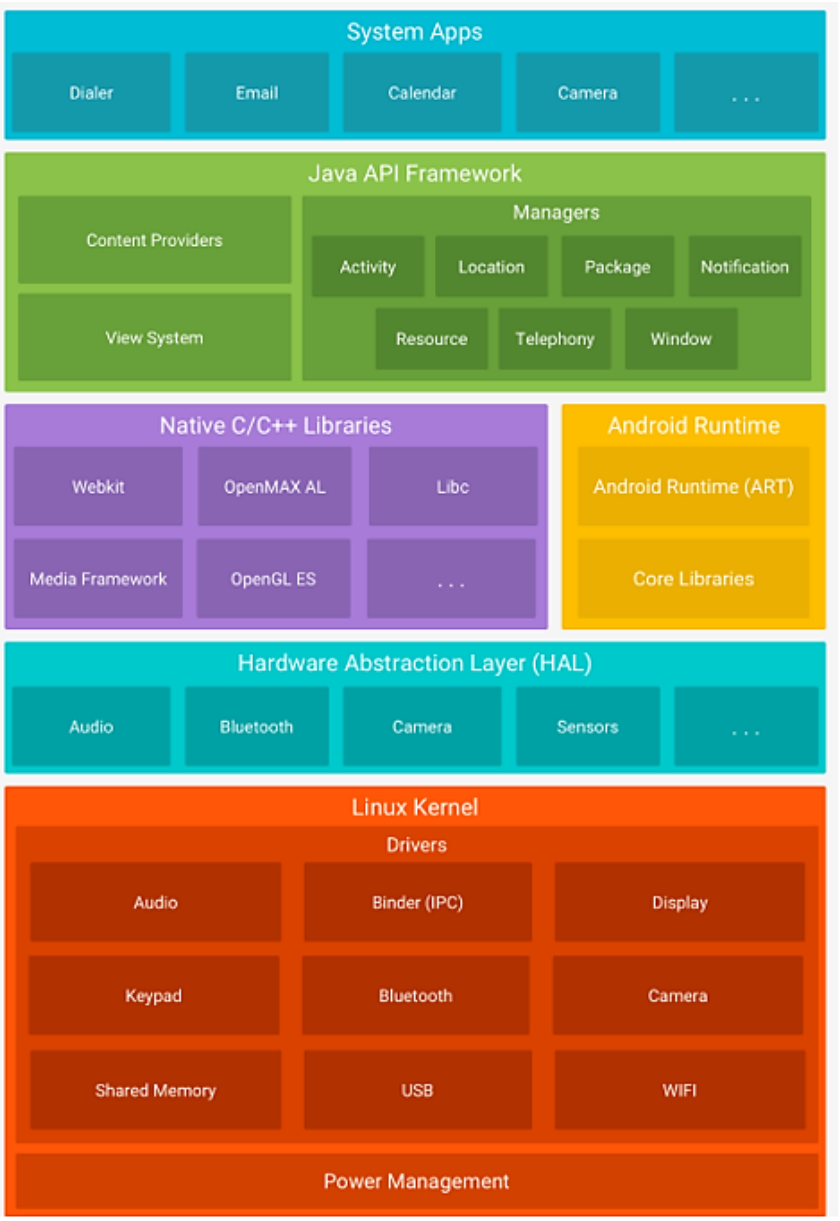

Figura 4.1: Arquitectura de Android

4.1.1. Kernel de Linux

El kernel o núcleo es la base de un sistema operativo. Provee servicios fundamentales del sistema como seguridad, gestión de memoria, gestión de procesos, funciones de red y manejo de *drivers*[5].

[5] *Device driver* o controlador de dispositivo es un programa informático que permite al SO interactuar con un periférico, haciendo abstracción del hardware y proporcionando una interfaz para utilizar el dispositivo.

La base de la plataforma Android es el kernel de Linux. Por ejemplo, el tiempo de ejecución de Android o ART (Android Runtime) se basa en el kernel de Linux para funcionalidades subyacentes, como la generación de subprocesos y la administración de memoria de bajo nivel. El uso del kernel de Linux permite que Android aproveche funciones de seguridad claves y, al mismo tiempo, permite a los fabricantes de dispositivos desarrollar controladores de hardware para un kernel conocido [10].

4.1.2. Hardware Abstraction Layer (HAL)

Brinda interfaces estándares del hardware del dispositivo al Java API Framework de nivel más alto. El HAL consiste en varios módulos de biblioteca y cada uno de estos implementa una interfaz para un tipo específico de componente de hardware, como el módulo de la cámara o de bluetooth. Cuando el framework de una API realiza una llamada para acceder a hardware del dispositivo, el sistema Android carga el módulo de biblioteca para el componente de hardware en cuestión [10].

4.1.3. Android Runtime (ART)

Es el tiempo de ejecución gestionado utilizado por las aplicaciones y algunos servicios donde cada uno ejecuta sus propios procesos con sus propias instancias.

ART sustituyó a la Máquina Virtual Dalvik (DVM por sus siglas en inglés) a partir de la versión 5.0 de Android, antes de esa versión, Dalvik era el runtime de Android. Ambos fueron específicamente creados para esta plataforma y son compatibles ya que ejecutan archivos DEX (Dalvik Executable) por lo que las aplicaciones desarrolladas para Dalvik deberían funcionar cuando se ejecuta con ART. Sin embargo, algunas técnicas que funcionan en Dalvik no funcionan en ART [11].

El ART está escrito para ejecutar varias máquinas virtuales en dispositivos de memoria baja ejecutando los DEX, que tienen formato de código de bytes diseñado para Android y que están optimizados para ocupar un espacio de memoria mínimo.

Algunas de las funciones principales del ART son:

- Compilación ahead-of-time (AOT) y just-in-time (JIT).
- Recolección de elementos no usados mediante la optimización de Garbage Collection (GC).
- Depuración mejorada, con un generador de excepciones de diagnóstico detalladas e informes de fallos, y la capacidad de establecer puntos de control para controlar campos específicos.

4.1.4. Bibliotecas Nativas

Muchos componentes y servicios centrales del sistema Android, como el ART y el HAL, se basan en código nativo que requiere bibliotecas nativas escritas en C y C++. La plataforma Android proporciona la capa Java API Framework para exponer la funcionalidad de algunas de estas bibliotecas nativas a las aplicaciones [10]. Por ejemplo, se puede acceder a la Open Graphics Library (OpenGL) a través de la API de Java OpenGL del framework de Android para agregar a la aplicación compatibilidad con gráficos 2D y 3D.

Si se desarrolla una aplicación que requiere de los lenguajes C o C++, se puedes usar el Android NDK (Native Development Kit) para acceder a algunas de esas bibliotecas nativas directamente desde el código.

4.1.5. Java API Framework

La capa de Java API Framework es un conjunto de API escritas en el lenguaje Java donde están disponibles todo el conjunto de funciones del SO Android. Estas API son los cimientos que se necesitan para crear aplicaciones de Android simplificando la reutilización de componentes del sistema y servicios centrales y modulares, como los siguientes [10]:

- Un View System extensible que se puede utilizar para compilar la interfaz de usuario de una aplicación; se incluyen elementos como listas, cuadros de texto, botones e incluso un navegador web integrable.
- Un Manager of Resources que brinda acceso a recursos sin código, como strings, imágenes y archivos de diseño. Manejar los recursos de forma externa permite mantenerlos de forma independiente al código y también permite proporcionar recursos alternativos que admiten configuraciones específicas de los dispositivos, como idiomas o tamaños de pantalla distintos, lo que cada vez es más importante, por la gran variedad de dispositivos Android que existen con configuraciones diferentes [12].
- Un Manager of Notifications que permite que todas las aplicaciones muestren alertas personalizadas en la barra de estado.
- Un Manager of Activities que administra el ciclo de vida de las aplicaciones y proporciona una pila de navegación.
- Content Providers que permiten que las aplicaciones accedan a datos desde otras aplicaciones, como por ejemplo la aplicación de Contactos, o que compartan sus propios datos.

4.1.6. Aplicaciones del Sistema

En Android se incluye un conjunto de aplicaciones para cámara, correo electrónico, mensajería SMS, calendario, contactos y navegación en Internet, entre otros elementos. Estas aplicaciones incluidas en la plataforma no tienen que ser las usadas obligatoriamente por el usuario (existen excepciones como por ejemplo la aplicación de configuración del sistema), ya que aplicaciones de terceros se pueden instalar y usarse en sustitución de las que vienen por defecto.

Las aplicaciones del sistema brindan funcionalidades claves a las cuales los desarrolladores pueden acceder desde sus propias aplicaciones. Por ejemplo, una aplicación externa que necesita acceder a la lista de contactos, invoca a la aplicación respectiva que ya está instalada por defecto en el sistema sin que el desarrollador tenga que implantar esa funcionalidad por sí mismo.

4.2. Aplicaciones de Android y sus Componentes

De manera sencilla, una aplicación móvil o también conocida simplemente como una *app*, puede definirse como una aplicación informática diseñada para ser ejecutada en dispositivos móviles para efectuar tareas, permitiendo al usuario satisfacer necesidades de diferentes tipos como educativo, profesional, entretenimiento, salud, etc.

Las aplicaciones del sistema incluidas por defecto en Android así como también aquellas que son añadidas posteriormente por el usuario (ya sean desarrolladas por el propio usuario o terceros), se encuentran y ejecutan en el último nivel de la arquitectura de Android y utilizan servicios, APIs y librerías de los niveles inferiores.

El formato de archivo utilizado en Android para instalar aplicaciones es el .apk (Android Package) de manera análoga a la que por ejemplo los sistemas Windows utilizan los archivos .exe en los computadores para instalar software. Este formato .apk es una variante del formato .jar de Java y se empaqueta a través del AAPT (Android Asset Packaging Tool) que es la herramienta por

defecto de Android SDK (Software Development Kit) para empaquetar todas las clases y recursos en un archivo.

A continuación, se presentarán los distintos componentes relacionados con una aplicación Android.

4.2.1. Activity

Es un pilar fundamental de las aplicaciones en Android, sirve como punto de entrada para la interacción de un usuario con una aplicación, ya que es la ventana en la que la aplicación despliega su interfaz de usuario y también es fundamental para la forma en que un usuario navega dentro de una aplicación o entre aplicaciones. Esta ventana que se muestra normalmente cubre la pantalla, pero puede ser de menor tamaño y flotar en la parte superior de otras ventanas, siendo lo más común que un Activity se muestre en toda la pantalla en una aplicación. Este componente se implementa como una subclase de la clase Activity.

La mayoría de las aplicaciones contienen múltiples pantallas, lo que significa que comprenden múltiples actividades. Por lo general, se tiene un *main activity* que es la primera pantalla que aparece cuando el usuario inicia la aplicación. Cada actividad puede iniciar otra actividad con el fin de realizar diferentes acciones [13]. Por ejemplo, la actividad principal en una aplicación de correo electrónico sencilla puede ser la pantalla que muestra la bandeja de entrada de correo electrónico de un usuario. A partir de ahí, la actividad podría iniciar otras actividades que proporcionan pantallas para tareas más específicas como redactar un correo y abrir un correo de forma individual para leerlo en detalle.

Para que se puedan utilizar las actividades en una aplicación, se debe registrar la información acerca de ellas en el archivo manifiesto de la aplicación llamado AndroidManifest.xml (ver Sección 4.3), y además, para su correcto funcionamiento se debe entender cómo se gestionan los ciclos de vida de cada actividad y como se manejan dentro de una pila de actividades.

A medida que el usuario navega a través de una aplicación, hacia afuera de la misma y de vuelta a ella, las instancias de las actividades realizan transiciones a través de diferentes estados de su ciclo de vida. La clase Activity proporciona una serie de *callbacks*[6] que permiten a la actividad saber a qué estado ha cambiado, como por ejemplo para saber si ha sido creada, detenida, reanudada o destruida [14].

En la Figura 4.2, tomada de [14], se presenta el ciclo de vida de un Activity y posteriormente se describe de manera breve los *callbacks* utilizados para manejar las transiciones entre los estados:

[6] *Callback* en programación es una función “A” que se usa como argumento de otra función “B”.

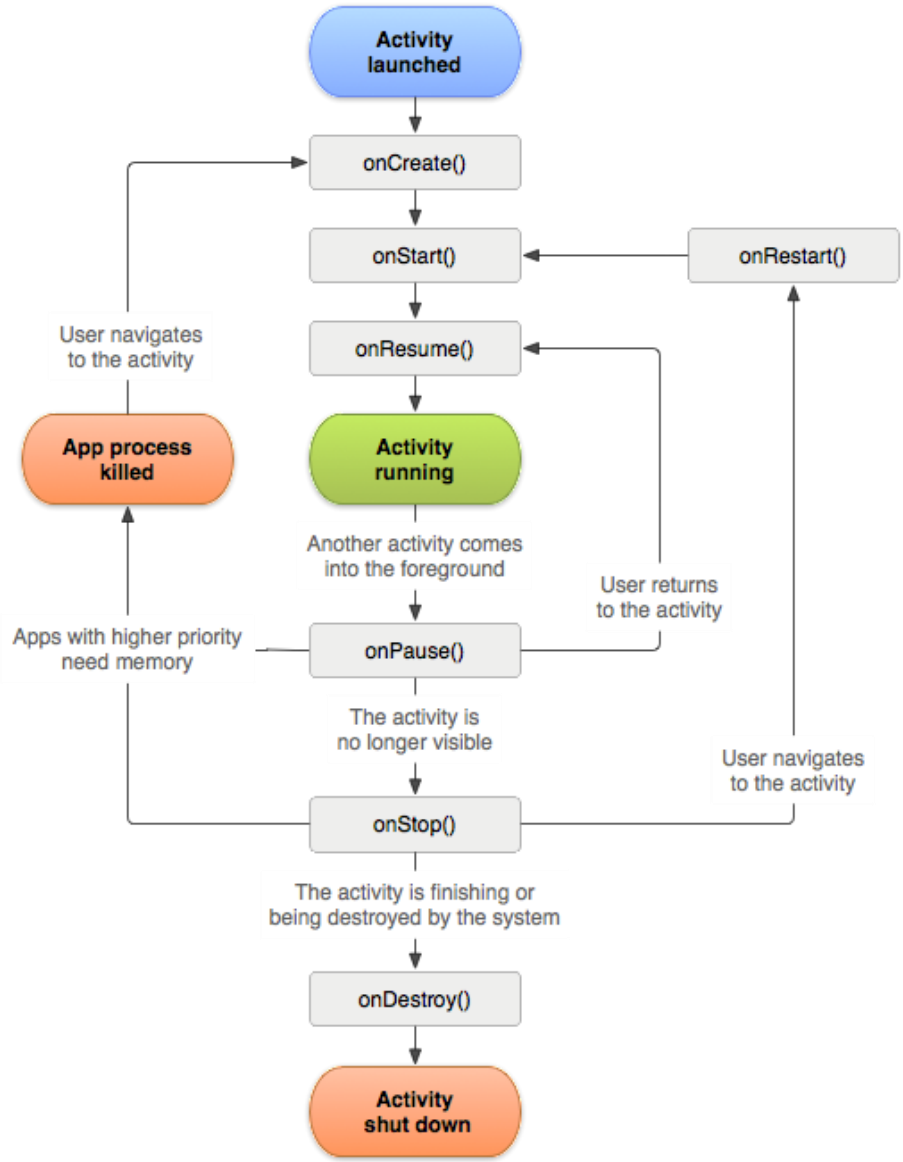

Figura 4.2: Ciclo de Vida de una Actividad

- **onCreate():** Se debe implementar este *callback* cuando el sistema crea su Activity y en este momento es que se inicializa los componentes esenciales de la actividad: Por ejemplo, en este estado la aplicación debe crear vistas y enlazar datos a las listas. Aquí es donde hay que llamar setContentView() para definir el diseño de la interfaz de usuario de la actividad. Cuando onCreate() termina, el siguiente *callback* es siempre onStart().
- **onStart():** Como el activity ya salió de onCreate(), entra en el estado Started o iniciado, y la actividad se hace visible para el usuario. Este *callback* contiene lo equivalente a unos preparativos finales para que la actividad pueda llegar a convertirse en el primer plano y empezar a interactuar.
- **onResume():** El sistema invoca este *callback* justo antes de que la actividad comience a interactuar con el usuario. En este punto, la actividad está en la parte superior de la pila de actividad, y captura toda la entrada del usuario. La mayor parte de la funcionalidad

básica de una aplicación se implementa en el método onResume(). El *callback* onPause() siempre sigue después de onResume().

- **onPause():** El sistema llama este método cuando la actividad pierde el foco y entra en un estado de pausa o Paused. Este estado se produce cuando, por ejemplo, el usuario pulsa el botón atrás en el dispositivo. Cuando el sistema llama onPause() es que el usuario abandona la actividad, y una vez que onPause() termina de ejecutarse, el próximo *callback* es onStop() o onResume(), dependiendo de lo que sucede después de que la actividad entra en el estado de pausa.
- **onStop():** El sistema llama este método cuando la actividad ya no es visible para el usuario. Esto puede suceder porque la actividad está siendo destruida, una nueva actividad está iniciando, o una actividad existente está entrando en un estado Resumed. En todos estos casos, la actividad detenida ya no es visible en absoluto. El próximo *callback* que llama el sistema es o bien onRestart(), si la actividad va a volver a interactuar con el usuario, o onDestroy() si esta actividad está terminando por completo.
- **onRestart():** El sistema invoca este *callback* cuando una actividad en el estado Stopped está a punto de reiniciar. onRestart() restaura el estado de la actividad desde el momento en que se detuvo. Este *callback* siempre es seguido por onStart().
- **onDestroy():** El sistema invoca este *callback* antes de que se destruya una actividad. onDestroy() por lo general se implementa para asegurar que todos los recursos de una actividad son liberados cuando la actividad, o el proceso que lo contiene, es destruido.

Como ya se ha mencionado, una aplicación puede contener múltiples actividades. Cada actividad debe diseñarse en torno a una clase específica de acción que el usuario puede realizar y a su vez puede iniciar otras actividades, incluso si existen en otras aplicaciones del dispositivo, por ejemplo, una determinada aplicación desea enviar un correo electrónico, en esa aplicación se puede incluir algunos datos como la dirección del email y el mensaje para enviarlos a través de un Intent, luego se abre una actividad de una aplicación de correo electrónico, maneja ese Intent y se muestra la interfaz de redacción de correo, se envía el correo electrónico y se reanuda a la aplicación original que se estaba usando, dando la impresión al usuario de que ambas actividades fueran parte de una misma aplicación. Si bien las actividades pueden ser de diferentes aplicaciones, Android mantiene esta experiencia del usuario ininterrumpida al mantener ambas actividades en la misma tarea.

Una tarea es una colección de actividades con la que los usuarios interactúan cuando realizan un trabajo determinado. Las actividades se organizan en una pila de actividades también conocida como *back stack*, en el orden en el que se abre cada actividad. La pantalla principal del dispositivo es el lugar donde se inician la mayoría de las tareas. Cuando el usuario toca un ícono de la aplicación para abrirla, la tarea de esa aplicación pasa a primer plano. Si no existe una tarea para la aplicación (la aplicación no se ha utilizado recientemente), se crea una nueva tarea y la actividad “principal” de esa aplicación se abre como la actividad raíz de la pila.

Como se explica en la documentación oficial [15], cuando la actividad actual inicia otra actividad, la actividad nueva pasa a la parte superior de la pila y obtiene el foco. La actividad anterior continúa en la pila, pero está detenida, en ese momento, el sistema retiene el estado actual de su interfaz de usuario. Cuando el usuario presiona el botón Atrás, se extrae la actividad actual de la parte superior de la pila (se destruye la actividad) y se reanuda la anterior (se restaura el estado anterior de su interfaz de usuario). En la pila, las actividades nunca son reorganizadas, se insertan en la pila cuando son iniciadas por la actividad actual y se sacan cuando el usuario la abandona utilizando el botón Atrás. Por lo tanto, la pila de actividades funciona como una estructura de “el último en entrar es el primero en salir”. En la Figura 4.3 se visualiza este comportamiento con una línea de tiempo que muestra el progreso entre actividades junto con la

pila de actividades actual en cada momento. Cuando el usuario presiona el botón Atrás, se destruye la actividad actual y se reanuda la anterior.

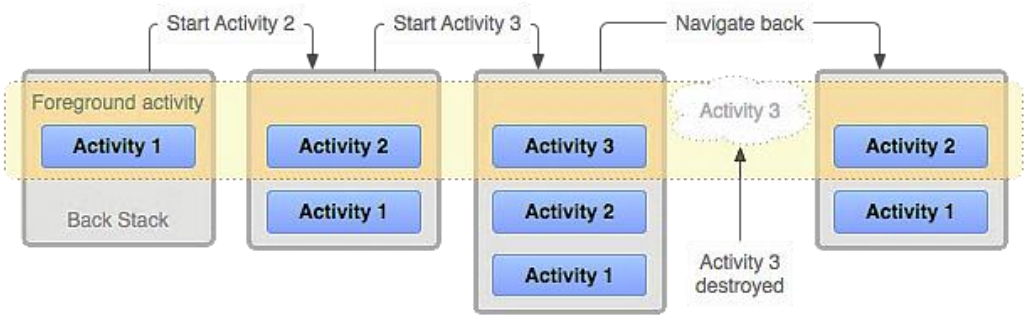

Figura 4.3: Representación de la Forma en que cada Actividad Nueva en una Tarea Agrega un Elemento a la Pila de Actividades

Si el usuario continúa presionando el botón Atrás, se saca cada actividad de la pila para mostrar la anterior, hasta que el usuario vuelve a la pantalla principal o a la actividad que se estaba ejecutando cuando se inició la tarea. Cuando se quitan todas las actividades de la pila, la tarea ya no existe.

4.2.2. Intent

Es un objeto que se puede usar para solicitar una acción de otro componente de la aplicación. Los intentos facilitan la comunicación entre los componentes y existen tres casos de uso fundamentales como se explica en [16]:

- **Para comenzar una actividad**: Se puede iniciar una nueva instancia de una actividad pasando un intento al método startActivity(). El intento describe la actividad que se debe iniciar y contiene los datos necesarios para ello. Si se desea recibir un resultado de la actividad cuando finalice, se llama al método startActivityForResult(). La actividad recibe el resultado como un objeto intento separado en el *callback* de onActivityResult() de la actividad.
- **Para iniciar un servicio:** Puede iniciar un servicio para realizar una operación única (como descargar un archivo) pasando un intento al método startService(). El intento describe el servicio que se debe iniciar y contiene los datos necesarios para ello. Si el servicio está diseñado con una interfaz cliente-servidor, se puede establecer un enlace con el servicio de otro componente pasando un intento al método bindService().
- **Para entregar una difusión:** Una difusión es un mensaje que cualquier aplicación puede recibir. El sistema entrega varias difusiones a los eventos del sistema, como cuando el sistema arranca o el dispositivo comienza a cargarse. Se puede enviar una difusión a otras aplicaciones pasando un intento a los métodos sendBroadcast(), sendOrderedBroadcast() o sendStickyBroadcast().

Además de conocer los casos fundamentales en los que se utiliza los Intents, también se debe saber que existen dos tipos, que son:

- **Intents explícitos:** Especifican qué componente se debe iniciar mediante el nombre completo de la clase. Por lo general, se utiliza este tipo de intents para iniciar un componente en la propia aplicación porque se conoce el nombre de clase de la actividad o el servicio que se desea iniciar. Un ejemplo de su uso es para iniciar una actividad

nueva en respuesta a una acción del usuario o iniciar un servicio para descargar un archivo en segundo plano.

- **Intents implícitos:** En este tipo de intent, no se nombra el componente de manera específica, sino que se declara una acción general a realizar, lo que permite que un componente de otra aplicación la maneje. Por ejemplo, se puede usar un intent implícito para solicitar que otra aplicación muestre una ubicación específica en un mapa en caso de querer mostrar al usuario su ubicación.

4.2.3. Service

Es un componente de una aplicación que puede realizar operaciones de larga ejecución en segundo plano y que no proporciona una interfaz de usuario. Puede ser iniciado por otro componente de la aplicación y ejecutarse en segundo plano, aún si el usuario cambia a otra aplicación. Además, un servicio y un componente pueden enlazarse para interactuar entre sí e incluso realizar una comunicación entre procesos [17]. Por ejemplo, un servicio puede en segundo plano manejar transacciones de red, reproducir música o interactuar con un proveedor de contenido.

Un servicio puede adoptar dos formas:

- **Servicio iniciado:** Un servicio está Iniciado o Started cuando un componente de aplicación lo inicia llamando al método startService(). Una vez iniciado, un servicio puede ejecutarse en segundo plano de manera indefinida, inclusive si el componente que lo inició es destruido. Generalmente, un servicio iniciado realiza una sola operación y no devuelve un resultado al emisor. Por ejemplo, puede descargar o cargar un archivo a través de la red y una vez terminada la operación, el servicio debe detenerse por sí mismo.
- **Servicio enlazado:** Un servicio está enlazado o Binded cuando un componente de la aplicación se vincula a él llamando al método bindService(). Un servicio enlazado ofrece una interfaz cliente-servidor que permite que los componentes interactúen con el servicio, envíen solicitudes, obtengan resultados e incluso lo hagan en distintos procesos con la comunicación entre procesos (IPC). Este tipo de servicios solamente se ejecuta mientras uno o varios componentes están enlazados con él, pero una vez que se desenlazan todos ellos, el servicio se destruye.

4.2.4. Content Providers

Administran el acceso a un conjunto estructurado de datos. Encapsulan los datos y proporcionan mecanismos para definir la seguridad de los datos. Los Content Providers son la interfaz estándar que conecta datos en un proceso con código que se ejecuta en otro proceso [18]. Cuando se desea acceder a datos en un proveedor de contenido, se usa el objeto ContentResolver en una interfaz con información global del entorno de la aplicación llamada Context para comunicarse con el proveedor como cliente. El objeto ContentResolver se comunica con el objeto del proveedor, una instancia de una clase que implementa ContentProvider. El objeto del proveedor recibe solicitudes de datos de clientes, realiza la acción solicitada y devuelve resultados.

4.3. Archivo Manifiesto de una Aplicación Android

Cada una de las aplicaciones en Android debe tener un archivo manifiesto que lleva exactamente el nombre de AndroidManifest.xml en el directorio raíz de la misma. Este archivo proporciona información esencial sobre la aplicación al sistema operativo para que éste pueda ejecutar el código de la aplicación.

El archivo permite entre otras cosas [19]:

- Nombrar la paquetería Java de la aplicación. Ese nombre sirve como identificador único de la aplicación.
- Describir los componentes de la aplicación como las actividades, servicios o proveedores de contenido que la integran. También nombra las clases que implementa cada uno de los componentes y publica sus capacidades, como los mensajes Intent con los que pueden funcionar. Estas declaraciones notifican al sistema Android los componentes y las condiciones para el despliegue de la aplicación.
- Determinar los procesos que alojan los componentes de la aplicación.
- Declarar los permisos que debe tener la aplicación para acceder a las partes protegidas de una API e interactuar con otras aplicaciones, y declarar los permisos que otros deben tener para interactuar con los componentes de la aplicación.
- Declarar el nivel mínimo del API de Android que requiere la aplicación.
- Enumerar las bibliotecas con las que debe estar vinculada la aplicación.

4.4. Razones para Desarrollar para Android

Al momento de desarrollar una aplicación para dispositivos móviles es importante definir para qué sistema operativo se desarrollará o cuáles en caso de que se decida desarrollar una aplicación que funcione en diferentes sistemas operativos. Cada uno de los sistemas antes mencionados poseen características propias, las cuales se deben tener presentes al momento de escoger el sistema operativo para el cual se desarrollarán las aplicaciones móviles.

Se escoge Android como la plataforma en la que se desarrollará la aplicación móvil por varias razones y características del SO, las cuales se explican a continuación:

- **Fácil de aprender:** el lenguaje usado para desarrollar generalmente aplicaciones en Android es Java, el cual es uno de los lenguajes de programación más usado a nivel mundial actualmente, por lo que, saber programar en Java, facilita mucho el aprendizaje del desarrollo para aplicaciones en Android. Además, se debe obtener el SDK (Software Development Kit) de Android, el cual provee un conjunto de herramientas de desarrollo que facilitan el trabajo.
- **Código abierto (Open-Source):** Android es de código abierto y está liberado con licencia Apache, que lo convierte en un sistema operativo totalmente libre para que cualquier desarrollador pueda modificar y mejorar su código. Además, Google ha publicado una gran cantidad de guías, tutoriales y ejemplos, que ayudan a los desarrolladores en el aprendizaje acerca del desarrollo de las aplicaciones para Android, así como manejar las APIs, entre otras.
- **Campo de pruebas más ágil**: Primero hay que saber que la Google Play Store es la plataforma de distribución digital de aplicaciones móviles para los dispositivos con SO Android. Ahora bien, si se tiene una cuenta de desarrollador en Android, y se crea una aplicación, subirla a la Play Store es más sencillo que hacerlo en la tienda de iOS, donde es necesario pasar por un proceso de aprobación más riguroso que incluso puede tardar semanas. Con lo que, si se quiere añadir una característica nueva a la aplicación o realizar pruebas a las funcionalidades de la misma, por poner algún ejemplo, el sistema facilitado por Google ayuda bastante más, pues las pruebas son mucho más ágiles.
- **Publicidad mejor segmentada:** Google es de las compañías que más provecho saca de la publicidad y de ahí se basa parte de su negocio, gracias al manejo de publicidad que hacen en la plataforma adaptándose a los usuarios y al contexto de las aplicaciones, motivo por lo cual atrae a una gran cantidad de desarrolladores que quieren obtener alguna remuneración de esa manera

- **Mercado mayor y de mayor crecimiento:** Android crece más y más rápido que ninguno de sus competidores. Para tener una idea, en la Figura 4.4 se muestra los resultados de un estudio disponible en el portal Statista[7] que refleja porcentualmente cómo se ha comportado el mercado móvil global de SO en ventas a usuarios finales desde el primer trimestre de 2009 al primer trimestre de 2016, lo que da una idea del número de usuarios que tiene la plataforma.
  Adicionalmente, en la Figura 4.5 tomada de otro estudio realizado por el mismo portal de estadísticas[8], se muestra el número acumulado de aplicaciones descargadas desde Google Play Store al mes de Mayo del año 2016 medido en billones.
  Se puede apreciar que el mercado de los dispositivos móviles en cuanto a número de usuarios se refiere está dominado por Android con un margen bastante amplio con respecto a su más cercano competidor, por lo que se considera que es la plataforma más apropiada para llegar a un público más extenso y para dar a conocer la aplicación en que se está centrando la presente investigación, siendo la tienda digital Google Play Store el lugar más idóneo para su distribución.

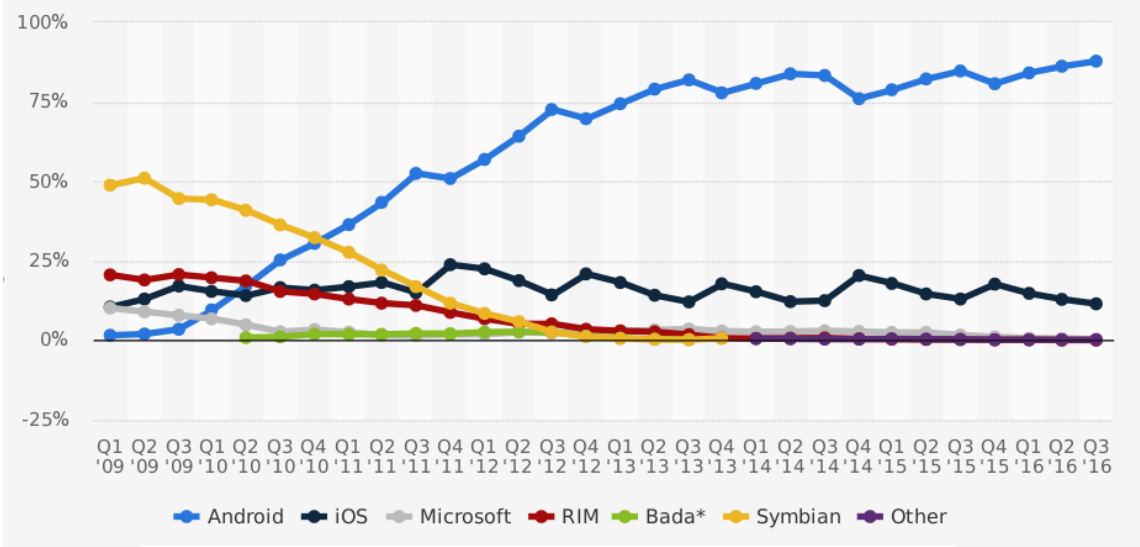

Figura 4.4: Mercado Móvil Global de SO en Ventas a Usuarios Finales desde el Primer Trimestre de 2009 al Primer Trimestre de 2016

[7] https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems
[8] https://www.statista.com/statistics/281106/number-of-android-app-downloads-from-google-play

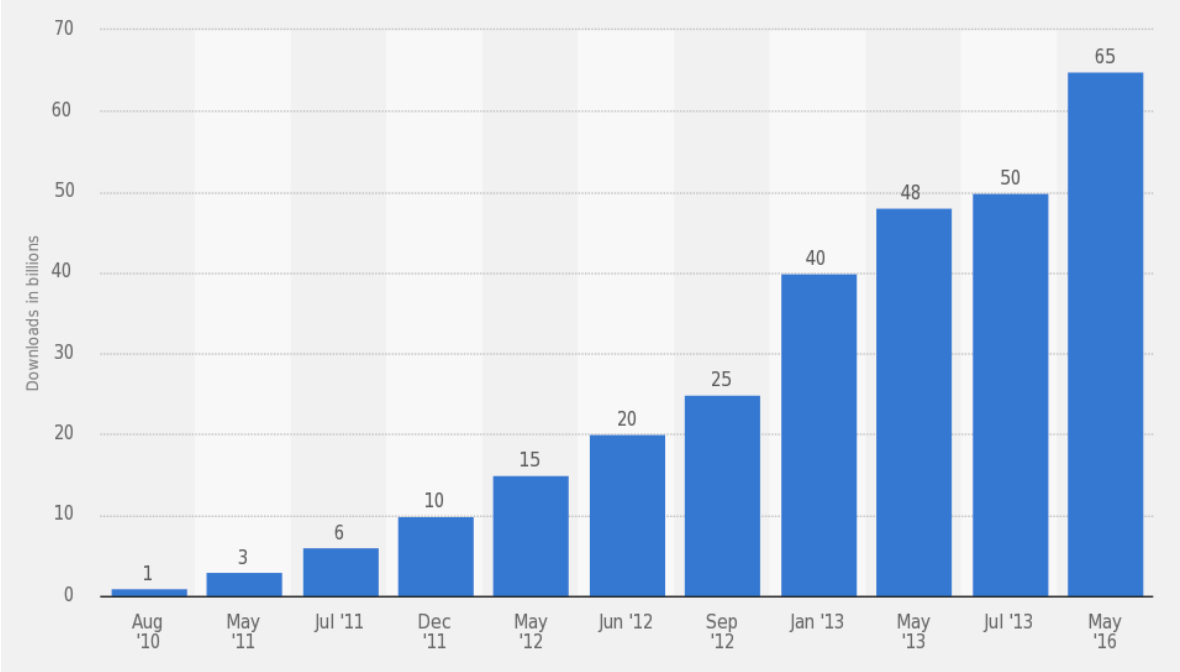

Figura 4.5: Número Acumulado de Aplicaciones Descargadas desde Google Play al Mes de Mayo del Año 2016 Medido en Billones

5. Aplicaciones Móviles y sus Enfoques de Desarrollo

Como ya se menciona en la Sección 4.2, una aplicación móvil o app, puede definirse como una aplicación informática diseñada para ser ejecutada en dispositivos móviles para efectuar determinadas tareas. Ahora bien, estas aplicaciones tienen que pasar por un proceso de análisis y desarrollo, en donde una de las primeras tareas es definir cuál enfoque de desarrollo es el más idóneo para la aplicación. Esta es una de las decisiones que más influyen en el resultado final del producto.

Escoger un enfoque de desarrollo para una aplicación móvil implica tomar en consideración muchos aspectos tales como presupuesto, plazos del proyecto, funcionalidad de la aplicación, tipo de usuario a la que va dirigida la app, entre otros. En las subsecciones siguientes se describe los distintos enfoques de desarrollo permitiendo así obtener una idea de los pros y contras de cada uno.

5.1. Aplicaciones Nativas

Es una aplicación móvil desarrollada para ser instalada y ejecutada en un sistema operativo en específico. Este tipo de aplicaciones puede acceder libremente a todas las APIs que el proveedor del SO ofrezca y, en muchos casos, tiene funciones y características únicas que son típicas de la plataforma en particular en la que se ejecuta.

Cuando se crea una aplicación nativa, los desarrolladores deben escribir el código fuente, crear recursos adicionales y declarar archivos específicos de la plataforma, como por ejemplo en el caso de Android debe haber un archivo de manifiesto (ver Sección 4.3). Utilizando herramientas provistas por el distribuidor del SO, se compila el código fuente para crear un ejecutable en formato binario que se pueda empaquetar junto con el resto de los recursos y estar listo para la distribución, como es el caso del archivo .apk en Android o el archivo .ipa en iOS.

El proceso de desarrollo suele ser similar para diferentes sistemas operativos, pero el SDK es específico de cada plataforma, y cada SO móvil cuenta con sus propias herramientas, lenguajes, formatos y canales de distribución [20]. Es ahí donde se observa que la mayor desventaja de este enfoque es que la aplicación móvil que se desarrolle para un sistema operativo determinado no se puede usar en otra plataforma, por lo que una misma aplicación llevaría un proceso de desarrollo y de mantenimiento distinto en cada SO.

A su vez, el hecho de que una aplicación sea desarrollada para un sistema operativo determinado es una ventaja, ya que permite sacar mejor provecho de las prestaciones de los dispositivos tales como GPS, acelerómetro, cámara, entre otros, porque sus implementaciones están normalmente desarrolladas en lenguajes de programación nativos para cada SO y se acceden a ellas a través de APIs propietarias de cada sistema, obteniendo muchas veces un mejor rendimiento en comparación a otros enfoques de desarrollo.

A través de las APIs la aplicación móvil puede interactuar con el hardware del dispositivo como por ejemplo para procesar audio recibido por el micrófono, reproducir sonido por el altavoz, interactuar con la pantalla táctil, recibir imágenes de la cámara, etc. Adicionalmente existen APIs ofrecidos por cada SO que ofrecen servicios de más alto nivel que mejoran la experiencia del usuario y enriquecen las funcionalidades de la aplicación como por ejemplo para hacer llamadas telefónicas, gestionar los contactos, acceder al calendario, visualizar las fotos del álbum, etc.

Otro conjunto importante de APIs que ofrecen los distintos SO son las de interfaz gráfica de usuario, proporcionando componentes como campos de texto, botones, menús, cuadros de

diálogo, barra de notificaciones, etc., que tienen características y funciones propias del SO específico.

5.2. Aplicaciones basadas en Web

Son aplicaciones que se ejecutan en un navegador web y que son optimizadas y adaptables a cualquier dispositivo móvil. En los últimos años, las tecnologías web han estado en constante evolución, ofreciendo así cada vez más posibilidades a la hora de desarrollar aplicaciones web dotadas de una gran funcionalidad, velocidad, rendimiento y experiencia de usuario, pudiendo expandirse a través de la web de una forma mucho más sencilla que por ejemplo una aplicación de escritorio en un PC. Esa evolución de las tecnologías permite hablar de una transición de "lenguajes para crear páginas web" a un potente estándar de desarrollo de complejas aplicaciones basadas en Navegador. Los dispositivos móviles modernos cuentan con poderosos navegadores que dan soporte a las nuevas funcionalidades de los lenguajes *HyperText Markup Language* versión 5 (HTML5), *Cascading Style Sheets* (CSS3) y *JavaScript* (JS), descritos en la Subsección 6.2.1.

Algunos de los componentes más destacables de HTML5 son: componentes de interfaz de usuario avanzados, soporte de gráficos y multimedia, servicios de geolocalización, acceso a archivos del sistema, *drag & drop*[9] y disponibilidad offline. Al hacer uso de los distintos componentes que brindan estos lenguajes se puede crear aplicaciones avanzadas basadas en Web.

Es importante diferenciar dos enfoques que existen al momento de desarrollar aplicaciones móviles basadas en web. El primer enfoque es el conocido como el Diseño Web Adaptivo o *Responsive Web Design* en inglés, el cual es una filosofía de diseño y desarrollo que pretende que un único diseño de una aplicación web tenga una adecuada visualización en los distintos dispositivos (*smartphones*, *tablets*, *PCs*, etc.) permitiendo así, brindar una buena experiencia al usuario incluso en pantallas táctiles pequeñas. En este enfoque, la aplicación se ejecuta accediendo a los distintos URL (*Uniform Resource Locator*) de la aplicación en el navegador. El segundo enfoque es crear aplicaciones web sólo con diseño dirigido a dispositivos móviles semejante al diseño nativo y que se ejecute como si fuera una aplicación nativa haciendo uso de un acceso directo en el dispositivo.

La principal ventaja de utilizar aplicaciones móviles basadas en web es que son multiplataforma, por lo que una vez desarrolladas, se distribuyen hacia distintos sistemas operativos, lo que ayuda a que los costos de desarrollo sean más bajos que cuando se desarrolla una aplicación nativa. Lo más probable es que se necesite realizar algunos ajustes para el correcto funcionamiento en los distintos dispositivos, pero eso no implicará un desarrollo para cada plataforma.

Otra ventaja es que no hay un control de las versiones de la aplicación que se publiquen ni de las posteriores actualizaciones porque al estar la aplicación en un servidor (ver Sección 6.1), llegan de manera automática a todos los usuarios independientemente del SO o dispositivo que tenga.

Entre las desventajas que tiene este enfoque de desarrollo [21], es que al no desarrollar de forma nativa con los SDKs y APIs disponibles para cada plataforma, se está inevitablemente perdiendo una parte importante de la funcionalidad del dispositivo, así como también en apariencia, siendo complicado que la interfaz se sienta tan ligada al dispositivo como en casos nativos. Pudiendo ser complicado lograr un solo diseño que se adapte perfectamente a todas las plataformas a la vez.

[9] *Drag & Drop* es la función de arrastrar y soltar elementos dentro de una aplicación.

Esa pérdida importante de la funcionalidad del dispositivo que se menciona en el párrafo anterior se debe a que, al ejecutarse este tipo de aplicaciones en el navegador, muy pocas APIs están disponibles para las aplicaciones que se ejecutan dentro del mismo a pesar de que el navegador de por sí es una aplicación nativa que tiene acceso directo a las APIs del SO, por lo que en las aplicaciones web muchas de las funcionalidades del dispositivo no se pueden utilizar o sólo de forma parcial.

5.3. Aplicaciones Híbridas

Las aplicaciones híbridas son aquellas que combinan desarrollo nativo con tecnología web, en donde, gran parte de la aplicación es desarrollada con tecnologías web como HTML, CSS y JS, para múltiples plataformas, manteniendo el acceso directo a APIs nativas en caso de necesitarlas; luego ésta es empaquetada como una aplicación nativa usando un empaquetador, el cual cumple la función de intermediario y traduce todas las instrucciones de forma que el sistema operativo del dispositivo las pueda entender al momento de ejecutar la aplicación en el mismo.

La porción nativa de la aplicación emplea APIs de sistemas operativos para crear un motor de búsqueda HTML incorporado que funcione como un puente entre el navegador y las APIs del dispositivo. Este puente permite que la aplicación híbrida aproveche todas las características que ofrecen los dispositivos modernos [20].

La principal ventaja de las aplicaciones híbridas es que son aplicaciones multiplataforma al igual que las basadas en web, por lo que pueden ser ejecutadas en diferentes sistemas operativos, además, debido a lo antes mencionado, permiten la reutilización de código, por lo que no habría que escribir una aplicación diferente para cada sistema. Otra ventaja importante es el acceso directo a APIs nativas, así como también a los recursos propios del sistema operativo.

Sin embargo, una de las principales desventajas es que no posee acceso completo a todos los recursos de hardware y APIs disponibles debido a la limitación del framework utilizado para desarrollar la app híbrida. Además, su rendimiento es menor al de las aplicaciones nativas dado que estas se ejecutan a un nivel más bajo que las híbridas, debido a que están desarrolladas directamente para un sistema operativo en específico, en cambio, las aplicaciones híbridas tienen que ser empaquetadas y al ejecutarse cada instrucción tiene que atravesar un proceso de traducción antes de ser ejecutada.

5.4. Comparación de los Enfoques de Desarrollo

A continuación, se establecen comparaciones entre los distintos enfoques de desarrollo, lo cual permiten obtener una idea de cuál es el más apropiado al momento de desarrollar una aplicación móvil según los requerimientos y necesidades que se tengan.

Las aplicaciones nativas sobresalen por su mejor rendimiento y por tener un mejor acceso a las APIs del SO y a las funcionalidades del dispositivo móvil, sin embargo, no son multiplataforma y tienden a tener procesos de desarrollo más costosos en esfuerzo, tiempo y recursos. Las aplicaciones basadas en web son multiplataforma, tienen un proceso de desarrollo más simple y menos costoso y, sus actualizaciones son más rápidas ya que la aplicación reside en un servidor y llega a todos los usuarios al momento de ser solicitado los recursos de la misma, aunque su mayor desventaja es que no aprovechan todas las funcionalidades de los dispositivos.

Las aplicaciones híbridas se encuentran a mitad de camino entre los dos enfoques ya mencionados, ofreciendo un poco de las ventajas que cada uno puede tener, en especial si se desea que la aplicación puede emplearse en diversos sistemas operativos, aunque muy probablemente su nivel de desempeño sea menor.

En la Tabla 5.1 [20] se muestran algunas de las características de los distintos enfoques de desarrollo, a modo de que se puede hacer una comparación entre ellos.

Característica	Aplicación Nativa	Aplicación Híbrida	Aplicación basada en Web
Lenguaje de desarrollo	Solo nativo	Nativo y web o solo nativo	Solo web
Portabilidad y optimización de código	Bajo	Alto	Alto
Características de acceso específicas del dispositivo	Alto	Mediano	Bajo
Uso de conocimiento existente	Bajo	Alto	Alto
Gráficos avanzados	Alto	Mediano	Mediano
Flexibilidad de actualizaciones	Bajo	Mediano	Alto
Experiencia de instalación	Alta	Alta	Mediana

Tabla 5.1: Características de los Enfoques de Desarrollo de las Aplicaciones Móviles

Se puede deducir de la Tabla 5.1 que no hay un único enfoque que brinde todos los beneficios al mismo tiempo, sino que al momento de desarrollar una aplicación móvil hay que tomar en consideración una serie de factores como por ejemplo son: las funcionalidades y características que se desea que tenga la aplicación, la experiencia del usuario, requerimientos a cumplir, presupuesto, recursos, mercado objetivo, plazos de entrega, entre otros.

El enfoque escogido por los autores para desarrollar la aplicación en la que se centra el presente estudio es el enfoque nativo. Entre algunas de los motivos de dicha elección se puede mencionar que la plataforma a la que va dirigida la aplicación es Android (razones de dicha elección en Sección 4.4), se tiene experiencia trabajando con el lenguaje de programación Java el cual es el lenguaje para desarrollos nativos en dicha plataforma, se desea tener una interfaz de usuario muy ligada a las que son las tendencias de diseño de ese SO y que se necesitará usar APIs nativas para utilizar ciertas funcionalidades de los dispositivos móviles.

6. Herramientas de Desarrollo

En los capítulos anteriores se ha hablado acerca de los dispositivos móviles, los sistemas operativos que se ejecutan en los mismos, más específicamente se habló del sistema operativo Android, las aplicaciones en esta plataforma y sus componentes, así como también más adelante se describió los distintos enfoques en el desarrollo que existen al momento de desarrollar las aplicaciones móviles. Ahora bien, en el presente capítulo se presenta una serie de herramientas y tecnologías que son igual de importantes y necesarias de considerar en el proyecto de desarrollo de la solución tecnológica en la que se enfoca el estudio.

Para comprender un poco más el por qué se describen algunas de las tecnologías y herramientas de desarrollo que siguen a continuación, es de relevancia mencionar que la aplicación móvil a desarrollar hará uso de un sistema de geolocalización en interiores para apoyar sus funcionalidades y lógica, así como también se piensa implementar lo que se conoce como un *Content Management System* (CMS) o Sistema de Gestión de Contenidos para tener una aplicación web que soporte la creación y administración de los contenidos que consumirá la aplicación por medio de una API.

6.1. Modelo Cliente-Servidor

El modelo cliente-servidor [22] se puede ver como un modelo en donde los clientes (o programas que representan entidades que necesitan servicios) y los servidores (o programas que proporcionan servicios) son objetos separados desde un punto de vista lógico y que se comunican a través de una red de comunicaciones para realizar una o varias tareas de forma conjunta.

El cliente realiza una petición de un recurso o servicio y recibe la respuesta a dicha petición; el servidor recibe y procesa esa petición para posteriormente retornar la respuesta (ver Figura 6.1). No necesariamente el cliente y el servidor tienen que estar en computadores distintos, puede darse el caso de que un mismo computador pueda ser ambos y la comunicación se realice en el mismo equipo. También puede darse el caso de que, en un mismo sistema distribuido, cada computador cumpla el rol del cliente para algunas tareas y el rol del servidor para otras.

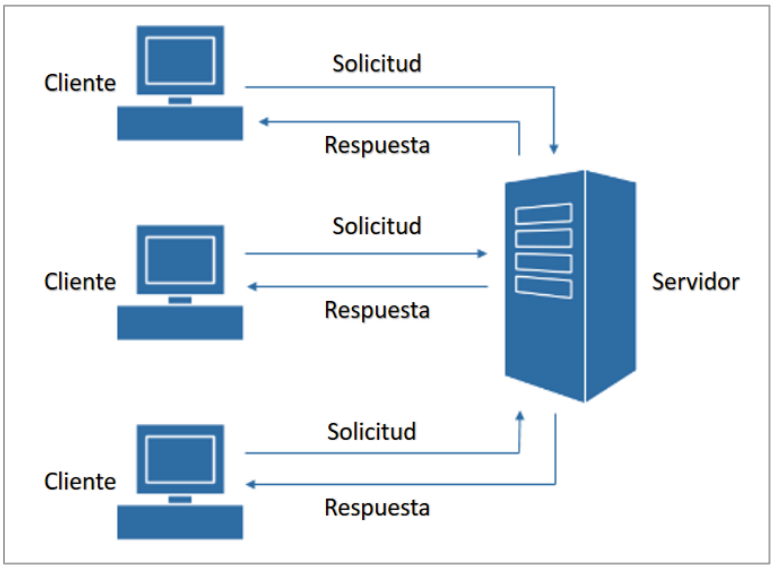

Figura 6.1: Modelo Cliente-Servidor

Las características del modelo cliente-servidor son [22]:

- **Protocolos asimétricos:** Los clientes inician la comunicación mediante la solicitud de un servicio o recurso. Los servidores esperan pasivamente por las solicitudes de los clientes. Existen una relación muchos a uno entre los clientes y un servidor.
- **Encapsulación de servicios:** El servidor está en la capacidad de determinar cómo proporcionar el servicio cuando se le hace entrega de una solicitud. Los servidores se pueden actualizar sin afectar a los clientes siempre y cuando el mecanismo como se comuniquen no se vea modificado.
- **Integridad:** El código y los datos de un servidor se mantienen centralizados, lo que ayuda a su mantenimiento e integridad. Al mismo tiempo, los clientes mantienen su independencia.
- **Transparencia de localización:** El servidor puede estar en el mismo computador que el cliente u en otro equipo de la red. En una comunicación cliente-servidor por lo general se oculta la localización del servidor a los clientes mediante la redirección de servicios.
- **Intercambios basados en mensajes:** Los clientes y los servidores son procesos débilmente acoplados que se comunican por pase de mensajes, utilizándolos para el intercambio de solicitudes de servicios y respuestas.
- **Modularidad:** El diseño extensible y modular de una aplicación que sigue el modelo cliente-servidor permite que la aplicación sea tolerante a fallos, por ejemplo, uno o más servidores pueden fallar sin parar el sistema en su totalidad mientras, otros servidores activos proporcionen y tengan disponibles los servicios que brindaban los servidores caídos.
- **Independencia de la plataforma:** Se busca que el modelo cliente-servidor cuente con independencia de hardware o de sistemas operativos, permitiendo así mezclar plataformas en los clientes y en los servidores.
- **Escalabilidad:** Los sistemas que siguen este modelo pueden ser escalados horizontal o verticalmente. Refiriéndose como escalado horizontal el añadir o eliminar estaciones clientes resultando en un ligero impacto en el rendimiento y, refiriéndose a escalado vertical como la capacidad de migrar a un servidor con mejores prestaciones y características o la incorporación de más servidores.
- **Separación de la funcionalidad cliente-servidor:** El modelo cliente-servidor es una relación entre procesos donde el proceso servidor es un proveedor de servicios y el cliente es un proceso que consume esos servicios, por lo que existe una clara separación de las funciones que debe desempeñar cada uno.
- **Recursos compartidos:** El servidor puede proporcionar servicios o recursos a muchos clientes al mismo tiempo, y regular el acceso a éstos a un conjunto específico de clientes.

6.2. Aplicación Web

Una aplicación web es cualquier aplicación que es accedida vía web por una red como Internet o una intranet, es decir, son aquellas aplicaciones que se ejecutan en el entorno de un navegador o codificadas con algún lenguaje soportado por el navegador, confiando en que la aplicación será ejecutada en dicho navegador.

Entre las ventajas más importantes de las aplicaciones web cargadas en Internet u otra red está que brinda facilidades para mantener y actualizar dichas aplicaciones sin que se necesite distribuir e instalar un software en los clientes. También se puede mencionar como ventaja la posibilidad de ser ejecutadas en múltiples plataformas.

Entre algunas características de las aplicaciones web, se tienen las siguientes:

- El usuario puede acceder fácilmente a estas aplicaciones haciendo uso de un navegador web (cliente).
- El usuario puede entrar desde cualquier lugar del mundo donde tenga un acceso a Internet.
- Como una única aplicación está instalada en un servidor, se puede actualizar y mantener la misma y todos sus usuarios verán los resultados inmediatamente.
- Emplean tecnologías como HTML, CSS, JavaScript, JQuery, Ajax, entre otras; que dan gran potencia a la interfaz de usuario.
- Emplean tecnologías que permiten una gran portabilidad entre diferentes plataformas, como por ejemplo computadores con SO Windows, Linux u otro sistema, dispositivos móviles, consolas de videojuegos, etc.

En cuanto al desarrollo de las aplicaciones web, se hace un tipo de abstracción para diferenciar y separar dos partes del sistema que son: Front-End y Back-End.

6.2.1. Front-End

Éste representa los procesos, funcionalidades y tecnologías de la aplicación web que corren de lado del cliente, es decir, los procesos y tecnologías que corren o se encuentran del lado del navegador web. Suele asociarse con la parte visual de la aplicación, como por ejemplo: la estructura, los estilos, colores, animaciones y efectos.

Para desarrollar esta parte de la aplicación web se usan tecnologías web que permitan el desarrollo de los factores antes mencionados como los estilos, animación, efectos, entre otros. En cuanto a las tecnologías, se tienen HTML, CSS y JavaScript; de manera resumida, HTML es usado para definir la estructura, CSS para asignar los estilos y JavaScript para aplicarle el dinamismo a la aplicación web.

- **HyperText Markup Language (HTML):** Es un lenguaje de marcado, el cual permite desarrollar aplicaciones web, estableciendo una estructura básica y un código para la definición de contenido de una aplicación web, como textos, juegos, imágenes, videos, entre otros; mediante el uso de etiquetas. Este lenguaje posee muchas etiquetas con las cuales se pueden definir: hipervínculos, saltos de línea, estilos, scripts, tablas, imágenes, entre otros.
  HTML es un estándar creado por la W3C (World Wide Web Consortium), la cual es una comunidad internacional que se dedica al desarrollo de estándares que aseguran el crecimiento de la Web a largo plazo, debido a que se relacionan con casi todas las tecnologías ligadas a la web, sobre todo en lo referente a su escritura e interpretación. Actualmente se está usando la versión cinco de HTML, es decir, HTML5.
- **Cascading Style Sheets (CSS):** Es un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla, o cómo se va a imprimir, o incluso cómo va a ser pronunciada la información presente en ese documento a través de un dispositivo de lectura. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos [23].
  CSS describe la presentación de documentos HTML o XML (eXtensible Markup Language), además, de cómo debe ser renderizado el elemento estructurado en pantalla, en papel o en otros medios. CSS es uno de los lenguajes base de la *Open Web* y posee una especificación estandarizada por parte del W3C [24]. Actualmente se está usando la versión tres de CSS, es decir, CSS3.
- **JavaScript (JS):** Es un lenguaje de programación interpretado y orientado a objetos. Además, es un lenguaje script multiplataforma basado en prototipos, imperativo, débilmente tipado y dinámico.

JavaScript es usado principalmente para asignar efectos y procesos dinámicos a documentos HTML, para poder ser mostrados en una aplicación web. Existen muchos efectos o dinamismos que se le pueden asignar a una aplicación web usando JavaScript como el despliegue de un menú, aparecer o desaparecer elementos, cambiar textos o imágenes de manera dinámica, realizar cálculos y mostrar resultados, mostrar mensajes de alertas, diferentes efectos animados, entre otros.
La principal ventaja de JavaScript es que, permite dar respuestas rápidas a las acciones del usuario, ya que el código JavaScript se carga en el mismo momento que el código HTML en el navegador, y reside en el cliente, por lo que no es necesario establecer una comunicación con el servidor para enviar una petición y esperar por una respuesta. Además, JavaScript permite atender eventos (como el clic del mouse), permite modificar de manera dinámica el DOM (Document Object Model) y permite la transferencia de data entre el servidor y el navegador por medio de llamadas asíncronas.

6.2.2. Back-End

Esta parte de la aplicación representa los procesos que se ejecutan del lado del servidor, los cuales no son visibles para el usuario, como el procesamiento de los datos, las peticiones, etc.; además, permite el manejo y manipulación de los datos procesados para poder implementar la lógica de negocio de la aplicación web, es decir, la implementación de las funciones, conexión y consultas a la base de datos, manejo de sesiones, entre otras.

Entre algunos lenguajes de programación que permiten desarrollar esta parte de la aplicación se tienen PHP (PHP Hypertext Preprocessor), Python, .NET, Java, etc.; además se pueden usar frameworks (ver Sección 6.3), que no solo permitan el desarrollo del Back-End de la aplicación sino también el desarrollo del Front-End.

Hay que tener en cuenta que al desarrollar el Back-End de una aplicación, se tiene que tener presente la seguridad de la misma, ya que, como ésta es la parte de la aplicación que se encarga de la manipulación de los datos y la conexión y consultas con la base de datos, se debe procurar tener un sistema lo suficientemente seguro como para evitar que la información o datos manejados sean violados o corrompidos.

Actualmente se implementan varias medidas de seguridad, algunas de ellas son [25]:

- **Cifrado:** transforma los datos en algo que un atacante no puede entender. En otras palabras, el cifrado permite implementar la confidencialidad de los datos. Actualmente, una de las técnicas más usadas del cifrado es por medio del uso de tokens para la autenticación de usuarios por medio de una API.
- **Autenticación:** se utiliza para verificar la identidad pretendida de un usuario, cliente, servidor, anfitrión u otra entidad. Por lo general, los usuarios son autentificados por medio de contraseñas, aunque existen otras formas de hacerlo.
- **Autorización:** se emplea luego de que un cliente ha sido autentificado, y determina las acciones que se le permiten o no se le permiten hacer dentro del sistema.
- **Auditoría:** las herramientas de auditoría realmente no protegen contra amenazas de seguridad, pero permiten rastrear a qué tuvieron acceso los clientes y de qué forma lo hicieron, los registros pueden resultar extremadamente útiles para analizar una infracción de seguridad y tomar medidas subsecuentes contra intrusos.

6.3. Frameworks

Son estructuras de software conformadas por componentes personalizables e intercambiables para el desarrollo de una aplicación. Estos frameworks poseen un conjunto de herramientas,

librerías, convenciones y buenas prácticas que buscan encapsular las tareas repetitivas en módulos genéricos reutilizables.

Actualmente, los frameworks son usados más que todo en el desarrollo de aplicaciones web y móviles. Algunos ejemplos de frameworks para el desarrollo web son: Laravel, CakePHP, Symfony, Ruby on Rails, entre otros; y en cuanto a ejemplos de frameworks para el desarrollo móvil se tienen algunos como: Ionic, jQuery Mobile, PhoneGap y React-Native.

Los principales objetivos de los frameworks son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones. Entre algunas de las características que poseen los frameworks son [26]:

- **Abstracción de URLs y sesiones:** no es necesario manipular directamente las URLs (en caso de ser una aplicación web) ni las sesiones, el framework ya se encarga de hacerlo.
- **Acceso a datos:** incluyen las herramientas e interfaces necesarias para integrarse con herramientas de acceso a datos como por ejemplo XML.
- **Controladores:** la mayoría de frameworks implementa una serie de controladores para gestionar eventos, como la inserción de datos mediante un formulario o el acceso a una página.
- **Autentificación y control:** incluyen mecanismos para la identificación de usuarios.

Tanto los frameworks orientados al desarrollo web como los que están orientados al desarrollo móvil siguen algún patrón de arquitectura de software, siendo el más usado actualmente, el patrón MVC, de las siglas Modelo-Vista-Controlador.

El patrón MVC consiste en separar el código o la aplicación en tres capas, la cuales son:

- **Modelo:** esta parte de la aplicación define la lógica de negocio. Se encarga de la manipulación de los datos, el acceso y consultas a la base de datos, actualizaciones de datos en la misma, gestiona los permisos de acceso, entre otras. Además, es quien se encarga de enviar a la parte de la Vista la información que necesita ser mostrada por medio de alguna petición.
- **Vista:** esta parte se encarga de presentar o mostrar la información enviada por parte del Modelo, es decir, la Vista representa la interfaz de la aplicación y la parte de la aplicación con la que el usuario interactúa directamente.
- **Controlador:** se encarga de manejar eventos (generados usualmente por el usuario) e invoca las peticiones al Modelo cuando se realiza alguna solicitud sobre la información. Además, el Controlador es quien se encarga de enviar a la Vista cualquier información o data recibida por parte del Modelo. En otras palabras, el Controlador es una capa que sirve de enlace entre la Vista y el Modelo, es decir, que cumple la función de intermediario entre las otras dos partes de la aplicación. A menudo se dice que en esta capa se define la lógica de la aplicación.

En la Figura 6.2 tomada del portal oficial de la empresa de desarrollo web y móvil llamada Information Technology Consulting[10], se puede apreciar un flujo básico de cómo interactúan las distintas capas del patrón MVC ante una solicitud hecha por un usuario de algún dato en la aplicación.

[10] http://www.itc-software.com.ar/Newsletter/024/Newsletter_201410.html

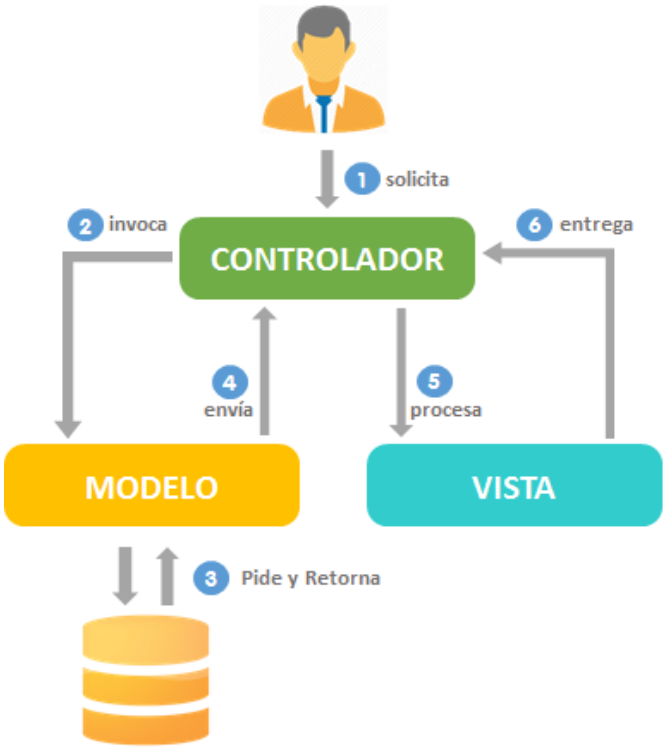

Figura 6.2: Interacción de las Distintas Capas del Patrón MVC

El framework a utilizar en el proyecto por parte de los autores para desarrollar la aplicación web relacionada al presente estudio es Laravel. Es un framework que fue lanzado oficialmente en el año 2011, código abierto, multiplataforma y que permite desarrollar aplicaciones y servicios web con el lenguaje de programación PHP.

Entre algunas de las ventajas que proporciona este framework y que ha motivado su elección está que tiene abundante documentación en la web y amplia comunidad y foros, es modular y con una amplia variedad de sistemas de paquetes y drivers con el que se puede extender su funcionalidad, facilita el manejo de rutas de la aplicación y la generación de las conocidas URLs amigables, brinda una herramienta de interfaces de líneas de comando llamada Artisan que permite realizar tareas como por ejemplo migraciones de datos, cuenta con un ORM (Object-Relational Mapping) llamado Eloquent que la interacción con las bases de datos sea totalmente orientada a objetos, y por último es adaptable al patrón MVC.

6.4. Sistema Manejador de Bases de Datos (SMBD)

Antes de adentrarse en lo que es un Sistema Manejador de Bases de Datos (SMBD), es importante precisar que una Base de Datos (BD) [27] es una colección o depósito de datos integrados, almacenados en soporte secundario (no volátil) y con redundancia controlada. Los datos, que han de ser compartidos por diferentes usuarios y aplicaciones, deben mantenerse independientes de ellos, y su estructura de datos almacenada se ha de apoyar en un modelo de datos, el cual ha de permitir captar las interrelaciones y restricciones existentes en el mundo real.

Ahora que ya se conoce que es una BD se puede definir que un SMBD [27] es un conjunto coordinado de programas, procedimientos, lenguajes, etc., que suministra a los distintos tipos de

usuarios los medios necesarios para describir y manipular los datos almacenados en la BD, garantizando su seguridad.

Las características de los SMBD son:

- Organizan los datos según las especificaciones dadas por los programas utilizados.
- Poseen metadatos, es decir, datos de naturaleza descriptiva para la BD.
- Soportan múltiples vistas de los datos y tienen la capacidad de compartirlos.
- Procesan transacciones multiusuario.
- Ayudan a reducir el tiempo de desarrollo de las aplicaciones.
- Tienen disponibilidad de la información actualizada.
- Gestionan BD de distintos tamaños según el SMBD elegido.
- Pueden manipular BD de diferentes tipos de complejidad.
- Poseen mecanismos de seguridad los cuales proporcionan acceso controlado a los datos.

Los SMBDs utilizan lenguajes para soportar funciones de manipulación, descripción y control de datos que se describen a continuación:

- **Lenguaje de manipulación de datos:** En inglés Data Manipulation Language (DML), es un lenguaje utilizado por los usuarios para consultar y manipular datos que se encuentran organizados de una manera estructurada. Este tipo de lenguajes a su vez se pueden clasificar en Lenguajes de Consulta Procedimentales en donde el usuario da instrucciones para que se desarrollen procedimientos y operaciones sobre la BD, y en Lenguajes de Consulta No Procedimentales en donde el usuario describe un procedimiento en específico como puede ser consultar, insertar, actualizar o eliminar datos.
- **Lenguaje de definición de datos:** En inglés Data Definition Language (DDL), es un lenguaje utilizado para definir estructuras que almacenarán los datos y los procedimientos o funciones para realizar las consultas. Con este lenguaje se especifica los elementos que integran la BD, su estructura, relación de los datos, reglas de validación, etc.
- **Lenguaje de control de datos:** En inglés Data Control Language (DCL), es un lenguaje utilizado para controlar el acceso a los datos en la BD. Proporciona una serie de instrumentos y procedimientos al administrador, como por ejemplo para conceder o eliminar permisos a usuarios o roles.

El SMBD del que harán uso los autores en el proyecto relacionado al presente estudio es MySQL, el mismo fue desarrollado originalmente por la empresa MySQL AB. Esta empresa fue comprada por Sun Microsystems en el año 2008 y a su vez, Sun Microsystems fue adquirida por la compañía Oracle Corporation en 2010. Se puede usar una versión *community* bajo la Licencia Pública General de GNU y en su versión empresarial bajo una licencia comercial que se debe obtener de Oracle.

Algunas de las razones por la que se decide trabajar con este SMBD son que es una base de datos *open source,* tiene un uso muy popularizado y respaldado a nivel mundial siendo utilizado con mucha frecuencia en aplicaciones web, es soportado y compatible con múltiples lenguajes de programación y es muy estable en aplicaciones donde hay baja concurrencia en la modificación de datos pero si un alto nivel de lectura de datos, lo cual se adapta a las necesidades de los autores para el manejo de datos tanto en el CMS como en la aplicación móvil.

6.5. Application Programming Interface (API)

Es un conjunto de funciones y procedimientos que proporcionan ciertas funciones o servicios para que puedan ser utilizados por otro software.

Cuando se hace uso de una API, se establece una comunicación entre componentes de software en donde se realiza una serie de llamadas a ciertas bibliotecas, funciones y procedimientos que ofrecen acceso a servicios, proporcionando una capa de abstracción al momento de programar, ya que, por ejemplo, los programadores pueden usar funciones predefinidas de cierto SO sin necesidad de saber cómo están implementadas o pueden construir sus aplicaciones sin necesidad de volver a programar funciones ya hechas por otros, reutilizando código.

Actualmente, las APIs juegan un factor clave en la integración de diversos y complejos sistemas informáticos para así enriquecer las características y alcance de los mismos. Están presentes en sistemas operativos, lenguajes de programación, redes y en la Web donde por el auge de las tecnologías y servicios Web ha hecho que el término API evolucione para generalmente hacer referencia a lo que de por sí es una Web API. Una Web API se caracteriza por estar apoyada totalmente en el protocolo HTTP (Hypertext Transfer Protocol) para brindar características y funciones.

Un ejemplo de las Web APIs sería una aplicación web de un cine por medio de la cual los usuarios pueden comprar las entradas para las funciones. Al momento de realizar una compra, se debe ingresar los datos de la tarjeta de crédito, luego la aplicación usa una Web API para enviar esos datos de forma remota al sistema de una entidad bancaria para que verifique y procese los datos. Una vez que se confirma el pago, la aplicación del banco envía la información a la aplicación del cine con el resultado de la operación y se emite los tickets de entrada. En todo ese proceso, el usuario solo ve la aplicación web del cine y para él es transparente todos los procesos que ocurren internamente y la comunicación entre las aplicaciones gracias a las APIs.

Existe una arquitectura para el intercambio y manipulación de datos llamada API REST que al día de hoy es la más usada por ser el estilo de arquitectura más lógico y eficiente en la creación de APIs para servicios de Internet.

6.5.1. API REST

REST (Representational State Transfer) es un estilo de arquitectura de software para sistemas hipermedias distribuidos tales como la Web [28]. El término fue introducido en la tesis doctoral de Roy Fielding en 2000, quien es uno de los principales autores de la especificación de HTTP.

REST permite crear aplicaciones y servicios que pueden ser usadas por cualquier cliente que entienda HTTP para obtener datos o generar operaciones sobre esos datos en diversos formatos como por ejemplo XML y JSON (JavaScript Object Notation), lo que hace que sea más simple y convencional que otras alternativas como el protocolo SOAP (Simple Object Access Protocol) o RPC (Remote Procedure Call).

Entre las razones que motiva a usar REST está que hace uso de una de las características de la Web que la ha hecho tan exitosa, se trata del esquema de direccionamiento global basado en URIs (Uniform Resource Identifier) los cuales identifican recursos u objetos conceptuales. La representación de tales objetos se distribuye por medio de mensajes a través de la Web.

Para desarrollar APIs REST hay que tener claro que los métodos HTTP más importantes son PUT, GET, POST y DELETE, los cuales son comparados con las operaciones CRUD asociadas a las bases de datos: CREATE, READ, UPDATE y DELETE. En la Tabla 6.1 [28], se refleja la analogía que existe entre los métodos HTTP, las operaciones CRUD y el lenguaje SQL (Structured Query Language).

Acción	HTTP	SQL
Create	POST	Insert
Read	GET	Select
Update	PUT	Update
Delete	DELETE	Delete

Tabla 6.1: Analogía entre Métodos HTTP, Operaciones CRUD y SQL

Ejemplos del uso de los métodos HTTP para un recurso de usuarios:

- *GET /usuarios* permite acceder al listado de usuarios.
- *POST /usuarios* permite crear un usuario nuevo.
- *GET /usuarios/9999* permite acceder al detalle del usuario 9999.
- *PUT /usuarios/9999* permite editar el usuario, sustituyendo todos los datos anteriores por los nuevos.
- *DELETE /usuarios/9999* permite eliminar al usuario 9999.

En la Figura 6.3 tomada del portal AWebFatory[11], se puede apreciar un diagrama de cómo sería la interacción entre el cliente, una API REST y los datos almacenados.

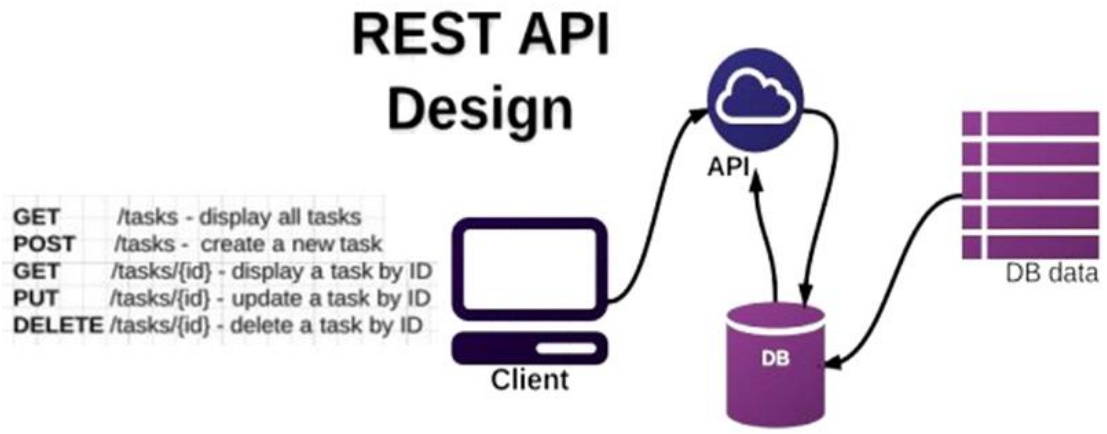

Figura 6.3: API REST

El API a desarrollar que permitirá que la solución móvil propuesta acceda a los datos que se gestionarán desde la aplicación web será visto como un módulo de ésta última mencionada, por lo que será implementado con la mista herramienta, es decir el framework Laravel.

6.6. Java

Se usará Java como lenguaje de programación para desarrollar la aplicación móvil en la que se centra el presente estudio, debido a que, como se explicó anteriormente, es el lenguaje usado para desarrollar aplicaciones móviles nativas en el sistema operativo Android.

Java es un lenguaje de programación [29]:

- **Sencillo:** se le conoce como sencillo ya que su estructura es similar a la de C++, el cual es un lenguaje popular a nivel mundial, pero además, Java es más sencillo que C++, puesto que sus características lo hacen más fácil de entender y programar.

[11] http://awebfactory.com/tecnoday/20150915-angular-restapi/#/5

- **Orientado a objetos:** porque permite al programador organizar su programa de tal forma que es muy similar al mundo real en estructura y en interacción entre sus componentes.
- **Distribuido:** ya que se desarrolló con la idea de redes y sus bibliotecas estándares, contienen unidades diseñadas para éste propósito lo que permiten desarrollar con cierta facilidad aplicaciones de Internet.
- **Interpretado:** ya que sus archivos ejecutables, llamados *bytecode*, son instrucciones o datos relativos a un computador virtual (Java Virtual Machine).
- **Robusto:** debido a características como la asignación automática de memoria, recolección de basura, etc., y poco susceptible a fallas.
- **Seguro:** ya que, los Applets (programas de Java), que corren en un navegador web, están sujetos a un número de restricciones que reducen las probabilidades de dañar un sistema o datos.
- **Neutro de arquitectura:** puesto que el bytecode de Java, está diseñado para ser leído e interpretado de la misma forma en cualquier hardware o sistema operativo que soporte Java.
- **Portátil:** debido a las características mencionadas anteriormente que hacen de Java un lenguaje de arquitectura neutral, ya que los programas de dicho lenguaje, no contienen dependencias de implementación.
- **Multi-hilos:** debido a que tiene facilidad de realizar varias tareas al mismo tiempo por medio de esta función, ya que por cada hilo que el programa ejecute, se realiza una tarea correspondiente a dicho hilo, por lo que, teniendo más de un hilo en ejecución al mismo tiempo se estarían realizando más de una tarea a la vez en tiempo real.
- **De alto rendimiento:** debido a su velocidad al momento de ejecutar los programas.

6.6.1. Java Virtual Machine (JVM)

Es el procesador o entorno virtual que se utiliza para interpretar el bytecode de los binarios de Java, ya que Java se hizo para correr en cualquier plataforma sin recompilar los binarios. De esta manera este bytecode se puede obtener para cualquier arquitectura y sistema operativo sin modificaciones al programa original. Entre las tareas principales de la JVM se tienen [30]:

- La reserva de espacio en memoria para los objetos creados.
- La liberación de la memoria no usada.
- La asignación de variables a registros y pilas.
- La invocación del sistema huésped para ciertas funciones, como los accesos a los dispositivos.
- La verificación del cumplimiento de las normas de seguridad de las aplicaciones Java.

6.7. Integrated Development Environment (IDE)

Es una aplicación que contiene un conjunto de herramientas de programación para facilitar el desarrollo de aplicaciones, es decir, está conformada por un editor de código fuente, un compilador, un depurador y un constructor de interfaz gráfica de usuario (GUI por sus siglas en inglés).

Por lo general, estos entornos de desarrollo se dedican exclusivamente a un solo lenguaje de programación, sin embargo, existen ciertos IDEs que pueden ser usados para varios lenguajes de programación, el cual se define en la instalación del IDE correspondiente.

Centrándose en el lenguaje de programación Java, el cual es el usado para desarrollar aplicaciones Android, se tienen algunos como: Eclipse, NetBeans, entre otros; con estos IDEs se puede desarrollar aplicaciones para diferentes entornos utilizando Java como lenguaje de

programación, desde aplicaciones de escritorio hasta aplicaciones para Android, sin embargo, existe un IDE que también usa Java como lenguaje de programación que a diferencia de los anteriores, éste está dirigido específicamente para el desarrollo de aplicaciones para Android, el cual es Android Studio.

6.7.1. Eclipse

Es una plataforma de desarrollo de código abierto basada en Java. Fue desarrollado por IBM, que puso su código fuente a la disposición de los usuarios. Eclipse es un marco y un conjunto de servicios para construir un entorno de desarrollo a partir de componentes conectados (plug-in). Hay plug-ins para el desarrollo de Java (JDT Java Development Tools) así como para el desarrollo en C/C++, Cobol, etc.

Eclipse dispone de un editor de texto con un analizador sintáctico. La compilación es en tiempo real. Posee pruebas unitarias con JUnit, control de versiones con CVS (Concurrent Versioning System), integración con Ant, clases, etc. Además, es importante destacar, que para desarrollar aplicaciones específicamente para Android con este IDE, se debe obtener e instalar un plugin llamado *Android Developer Tools (ADT)*, que incluye soporte integrado para proyectos de Android y herramientas. Este plugin incluye un conjunto de extensiones que facilitan la creación, ejecución y depuración de aplicaciones Android.

6.7.2. NetBeans

Es un IDE para Java usado ampliamente para el desarrollo de software que contiene muchas funcionalidades para distintos tipos de aplicaciones y así facilitar al máximo la programación, la prueba y la depuración de las aplicaciones que se desarrollan, e incorpora un editor propio. Además, NetBeans es libre, gratuito sin restricciones de uso y de código abierto.

NetBeans funciona en base a módulos, lo que permite que se pueda extender de manera sencilla. Además, se puede integrar fácilmente con controladores de versiones. Por medio de NetBeans se pueden crear diferentes tipos de aplicaciones como aplicaciones web, para computadores y aplicaciones móviles que funcionen en varias plataformas.

Al igual que Eclipse, para poder desarrollar aplicaciones para Android con NetBeans es necesario obtener e instalar un plugin, el cual es *NBAndroid*, que abarca diversos aspectos del desarrollo en Android en su funcionalidad, como: soporte al núcleo SDK de Android, soporte del proyecto, editores mejorados para archivos XML Android, previsualización del diseño GUI, entre otras.

6.7.3. Android Studio

Android Studio es el IDE oficial (reemplazando a Eclipse) para el desarrollo de aplicaciones para Android. Se encuentra basado en el software IntelliJ IDEA de JetBrains[12]. Android Studio proporciona las herramientas más rápidas para la creación de aplicaciones en todos los tipos de dispositivos Android.

Además del potente editor de códigos y las herramientas para desarrolladores de IntelliJ, Android Studio ofrece aún más funciones que aumentan la productividad durante la compilación de las aplicaciones para Android, como se explica a continuación según la documentación oficial disponible en [31]:

- Sistema de compilación flexible basado en Gradle.
- Un emulador rápido con distintas funciones.
- Un entorno unificado en el que se puede desarrollar para todos los dispositivos Android.

[12] https://www.jetbrains.com/idea

- Instant Run, para aplicar cambios mientras la aplicación se ejecuta sin la necesidad de compilar un nuevo APK.
- Integración de plantillas de código y GitHub, para ayudar a compilar funciones comunes de las aplicaciones e importar ejemplos de código.
- Gran cantidad de herramientas y frameworks de prueba.
- Herramientas Lint para detectar problemas de rendimiento, uso, compatibilidad de versión, etc.
- Compatibilidad con C++ y NDK.
- Soporte integrado para Google Cloud Platform, que facilita la integración de Google Cloud Messaging y App Engine.

En la Figura 6.4 [31], se puede apreciar cómo es la interfaz gráfica de Android Studio en su ventana principal y como consta de diversas áreas lógicas que se explican posteriormente.

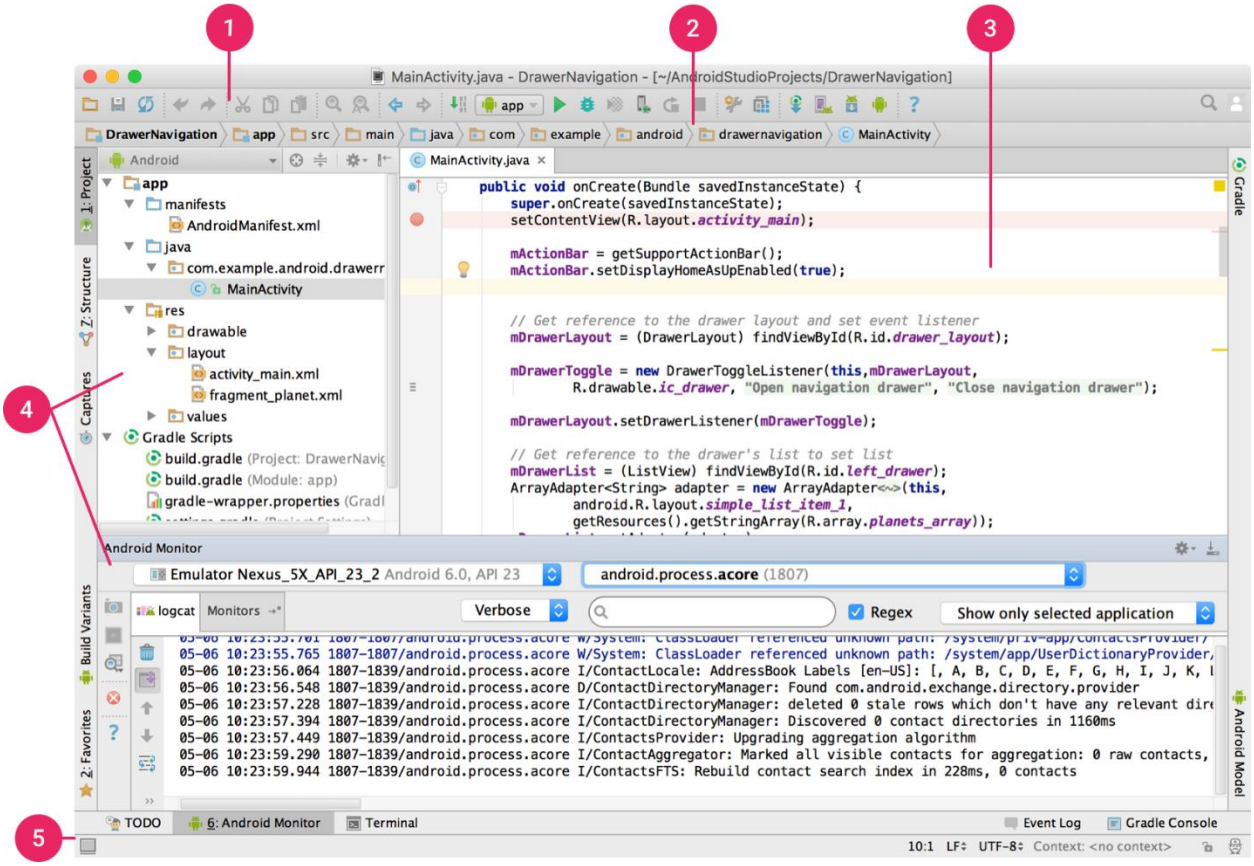

Figura 6.4: Ventana Principal de Android Studio

- (1) La barra de herramientas, permite realizar una gran variedad de acciones, como ejecutar la aplicación y el inicio de herramientas de Android.
- (2) La barra de navegación, ayuda a explorar el proyecto y abrir archivos para editar. Proporciona una vista más compacta de la estructura visible en la ventana Project.

- (3) La ventana del editor, es el área en la que se edita código. Dependiendo del tipo de archivo con el que se está trabajando, el editor puede cambiar. Por ejemplo, al visualizar un archivo de diseño el editor muestra el Editor de diseño.
- (4) Las ventanas de herramientas permiten acceder a tareas específicas, como la administración de proyectos, la búsqueda y los controles de versión, entre otras.
- (5) En la barra de estado se muestra el estado del proyecto y el IDE. También se muestra advertencias o mensajes.

Android Studio es el IDE por excelencia para el desarrollo de aplicaciones Android nativas, y por esta razón será usado para el presente proyecto, ya que se enfoca exclusivamente en el desarrollo de aplicaciones Android y provee muchas herramientas y funciones (mencionadas anteriormente) que facilitan el trabajo. Además, a diferencias de otros IDEs, Android Studio no requiere de la instalación de algún plugin externo para cumplir con su objetivo.

6.8. Geolocalización por WiFi

Antes de empezar con la geolocalización usando WiFi como tecnología o método, primero hay que tener en cuenta que la geolocalización puede definirse como la capacidad de calcular y obtener la ubicación geográfica de un objeto, como puede ser por ejemplo un smartphone.

6.8.1. Tecnologías para la Geolocalización

Se muestra a continuación algunas de las diferentes tecnologías que se pueden usar para poder llevar a cabo la geolocalización:

- **GPS:** el GPS (Global Positioning System) es un sistema de navegación basado en radiofrecuencia en donde los dispositivos de recepción trabajan en conjunto con satélites que orbitan alrededor de la tierra. Este sistema fue diseñado, desarrollado e implementado por Departamento de Defensa de los Estados Unidos y actualmente es utilizado de manera gubernamental, comercial y privada para la geolocalización en espacios abiertos. Esta tecnología se sirve de los satélites para calcular las distancias a los receptores, usando algoritmos de triangulación para determinar la posición del dispositivo (latitud y longitud) y poder así brindar información relevante.
  Hoy en día, todos los smartphones tienen implementada esta tecnología por lo que es una de las más accesibles con respecto a las otras tecnologías a mencionar. Sin embargo, para que funcione de manera óptima el dispositivo debe tener visión abierta a los satélites, lo que hace que no se tenga disponibilidad del servicio en ubicaciones interiores o lugares subterráneos, incluso, en espacios abiertos su disponibilidad puede verse reducida debido a que las señales de radio deben atravesar grandes distancias que no siempre se encuentran despejados, como por ejemplo puede haber muchas edificaciones que pueden obstaculizar la transmisión de señales.
- **RFID:** la tecnología RFID (Radio Frequency Identification) es un sistema de autoidentificación inalámbrico, el cual consiste en dispositivos pequeños denominados etiquetas o tags que almacenan información y lectores que pueden leer a estas etiquetas a distancia. Es una forma de comunicación a través de wireless (de forma inalámbrica) que usa las ondas de radios para identificar y hacer tracking (rastreo en español) de objetos. Está siendo adoptada cada vez por más industrias debido a que su costo es cada vez menor y sus capacidades crecen [32].
  Por lo general, las etiquetas RFID son dispositivos pequeños que pueden ser adheridas o incorporadas a objetos, personas o animales y que contienen antenas utilizadas para recibir y responder a peticiones hechas por radiofrecuencia desde un emisor-receptor RFID que viene siendo el lector.

En cuanto a su accesibilidad, la implementación de un sistema RFID requiere múltiples actores y muchos componentes diferentes, lo cual dificulta su uso. Además, su precisión varía según el tipo de etiqueta, la antena y lectores utilizados.

- **Bluetooh:** Bluetooth es una especificación tecnológica abierta del Bluetooth Special Interest Group (SIG) para redes privadas locales que permite transmitir voz y datos entre diferentes dispositivos, asegurando la compatibilidad entre una amplia gama de dispositivos de distintos fabricantes. Utiliza un enlace por radiofrecuencia en la banda ISM (Industrial, Scientific and Medical) de los 2.4 GHz reservada internacionalmente para el uso no comercial de radiofrecuencia en esas áreas.
  Con esta tecnología se busca facilitar la comunicación entre los dispositivos móviles, establecer comunicaciones inalámbricas y tener la posibilidad de crear pequeñas redes de área personal.
  Actualmente, una de las alternativas más modernas para la geolocalización en interiores mediante tecnología Bluetooth se hace haciendo uso de unos dispositivos capaces de emitir señales en esa especificación llamados Beacons. Son muy pequeños, de bajo consumo y emiten señales broadcast para transmitir mensajes a dispositivos móviles. Entre algunos de los puntos en contra que tienen estos dispositivos es que aún su uso no es muy extendido, aún la oferta y compatibilidad se da en dispositivos de gamas media y alta y que se necesita tener siempre activado el Bluetooth en los dispositivos, lo que hace que la batería se desgaste muy rápido.
- **WiFi:** Al día de hoy es una de las tecnologías de comunicación de mayor presencia en el mundo, basada en el estándar 802.11 del organismo IEEE (Institute of Electrical and Electronics Engineers), que ha conquistado el mercado desde el primer momento de aprobación de la especificación 802.11-b en 1999 [33]. Permite establecer un mecanismo de conexión entre dispositivos electrónicos de forma inalámbrica.
  Uno de los dispositivos principales en las redes WiFi son los Access Point (APs) o Punto de Acceso, los cuales permiten la interoperabilidad entre la red cableada y los dispositivos Wireless. Un AP recibe y emite datos, tanto a través de cables Ethernet como también en forma inalámbrica. Otras funciones asociadas a estos equipos es el control de acceso, seguridad y aislación del ruido ambiente o mejora de la razón señal/ruido [34].
  Todos los smartphones y tablets incorporan esta tecnología por la cual es totalmente accesible a todos los usuarios que posean un dispositivo de este tipo, además, de PCs, laptops, entre otros. El alcance de los dispositivos WiFi dependerá de muchos factores, como por ejemplo, factores que disminuyan el alcance por medio de interferencias en la señal por paredes, techos, muros, etc.; y pueden existir otros factores que en cambio, aumenten el alcance de la señal, como por ejemplo la instalación de repetidores de señal.

6.8.2. Razones para Usar Tecnología WiFi en la Geolocalización

Para la aplicación móvil a desarrollar por parte de los autores, se ha tomado la decisión de usar la tecnología WiFi para implementar las funciones relacionadas con la geolocalización.

WiFi tiene ventajas frente al sistema de geolocalización GPS, porque el segundo no presenta disponibilidad del servicio en espacios interiores por características propias de dicha tecnología.

Otra ventaja de WiFi es que es una tecnología de uso extendido y cotidiano presente en muchos lugares como hogares, oficinas, escuelas, bibliotecas, centros comerciales, plazas, etc, a diferencia de la tecnología Bluetooth y RFID cuyo uso no es extendido. En el caso de los centros comerciales, es muy común que exista una gran cantidad de Access Poinsts (APs) dentro del edificio, ya sean propios de las tiendas o propias de la infraestructura de red del edificio, lo que hace viable tener acceso a esta tecnología. Esa alta presencia de APs dentro de las instalaciones permite recolectar datos de redes WiFi y una vez que se establezca una relación entre las

respectivas redes WiFi con unos puntos geográficos determinados se puede determinar la ubicación de los usuarios para apoyar las funcionalidades de la aplicación móvil.

6.8.3. Métodos de Geolocalización por WiFi

La mayoría de los métodos de geolocalización mediante tecnología WiFi se basan en los valores de la potencia de la señal con los que los APs son detectados por los dispositivos móviles para así calcular la ubicación. Dependiendo del algoritmo que se utilice, la precisión y la complejidad de la aproximación varía.

Como se muestra en [8] y haciendo referencia a ellos, alguno de los métodos para llevar a cabo la geolocalización mediante redes WiFi son:

- **Vector de potencia:** Se debe almacenar en una base de datos la información de la señal que emiten los APs que se recoge en una fase de entrenamientos de los dispositivos. Esa base de datos puede verse como un vector donde cada celda contiene la potencia que le llega al usuario de cada AP a su posición en un instante determinado.
  Este método consta de tres fases: En la primera fase hay que conocer la información de los APs, en la segunda fase llamada fase de entrenamiento se construye la BD haciendo uso de información recolectada por el dispositivo, de manera que se guarden las potencias de cada AP para cada posición del mapa y en la tercera fase llamada fase de estimación, se obtiene para cada localización un vector con los niveles de señal recibida de cada uno de los APs. Este vector se compara con los datos almacenados en la BD para estimar la posición según aquella en la que los niveles de señal son más cercanos o parecidos.
- **Triangulación de potencia:** La palabra triangulación es un término geométrico que refiere al uso de la trigonometría de triángulos para determinar las posiciones de puntos, medidas de distancias o área de figuras. Se toman múltiples puntos de referencia para localizar una posición desconocida.
  El proceso de la triangulación es el siguiente: usando la potencia recibida de tres APs se construye un sistema de ecuaciones, que representa tres círculos. Luego, se procede a resolver ese sistema para obtener un conjunto de puntos que reciben el nombre de puntos de triangulación. Cada punto de triangulación pasa a considerarse el vértice de un triángulo. Posteriormente se forman todos los triángulos posibles y se calculan sus áreas para poder compararlas. El centro del triángulo con el área más pequeña se toma como la ubicación estimada del usuario.
- **Heurística de proximidad:** Se basa en ver que AP es el más cercano al dispositivo para determinar su posición, para ello, se usa la potencia recibida de cada AP, se descartan los valores mínimos y se dice que la mayor potencia corresponde al del AP más cercano al dispositivo. Por lo tanto, se asume que el cliente está en la posición de dicho AP o en esa zona cercana.
- **Método de los k-vecinos más cercanos:** Este método hace uso de un modelo de localización llamado *Fingerprinting* que se basa en medir la potencia de las señales recibidas de cada AP en un lugar en específico. Posteriormente a esas fuerzas se le aplica el algoritmo de los k-vecinos más cercanos para determinar la posición del usuario. En este algoritmo se debe buscar los datos dentro de la base de datos creada en el proceso de *Fingerprinting*, y seleccionar los puntos que más se ajustan a las señales recibidas en el lugar en el que se encuentra el usuario. Cuanto mayor sea el número de mediciones y comparaciones, mejor será el rendimiento de este método.
- **Métodos probabilísticos:** Estos métodos no se basan en las propiedades de propagación de las ondas electromagnéticas sino en teorías de probabilidades, entre esas la teoría de Bayes. Se trata de técnicas que mantienen una distribución de

probabilidad sobre todas las posibles ubicaciones del entorno. Las técnicas probabilísticas tienen un mayor coste computacional que otros métodos.

- **Localización usando RTT mediante llamadas Ping:** RTT (Round-Trip Time) es el tiempo de ida y vuelta que tarda un paquete de datos enviado desde un emisor en volver a él mismo habiendo pasado por el receptor del destino. Se mide el tiempo que tarda un paquete *ICMP* (*Internet Control Message Protocol*) enviado usando la utilidad *Ping*, desde el usuario cliente al AP y su retorno.
  Para localizar al usuario se necesitan al menos cuatro APs accesibles, para poder ejecutar un algoritmo de triangulación.

6.8.4. Método de los K-Vecinos Más Cercanos

Se escoge el método de los k-vecinos para desarrollar las funcionalidades de geolocalización de la aplicación móvil en la que se centra el presente estudio. Según una investigación hecha, se considera que es un método que se adapta a las necesidades del proyecto y a los requerimientos de precisión, siendo el más apropiado si se tiene en cuenta factores como implementación, coste de cálculo y precisión. A continuación se precisa más información acerca del método.

Este método se basa en la definición de una métrica en el espacio de celdas cubiertas electro-magnéticamente por los APs. A cada sector o área del espacio considerado para realizar la geolocalización se le asigna para cada AP un valor que representa la potencia recibida en dicho lugar. Cuando se mide la potencia en un punto arbitrario, se define como distancia la diferencia entre la potencia registrada inicialmente y la medida. El valor absoluto de esta magnitud será considerado como una distancia [34].

Una vez que se tiene una medición hecha en un lugar en específico, el algoritmo busca un punto tal que la suma de los cuadrados de la distancia sea mínima. Si no se produjeran errores de medición debería existir un punto para la cual la suma cuadrática de distancias fuera nula, sin embargo, como siempre existen errores, se considera que el punto para la cual la suma cuadrática de la distancia sea mínima es la mejor aproximación que se puede obtener.

Como fase previa a la implementación del algoritmo de los k-vecinos más cercanos se debe hacer el *Fingerprinting*. Se debe tener un mapa del lugar y dividirlo en sectores o áreas que tengan medidas uniformes para así registrar en él la ubicación de cada AP a utilizar y tener claro las ubicaciones donde se hará las mediciones. Luego se debe recorrer el edificio y almacenar los valores de la fuerza de la señal recibida de cada AP accesible, tomando distintas medidas desde la posición más centrada posible de cada una de los sectores y tomando una cantidad amplia de mediciones (por ejemplo 50) con respecto a cada AP y a cada orientación geográfica manteniendo constante la altura del dispositivo.

Según la distancia entre los puntos de medición que se tomen durante la fase de f*ingerprinting* variará la precisión en metros del algoritmo. Este proceso se realiza una vez y sirve para calcular la posición de todos los usuarios, salvo algunos casos en lo que es necesario repetir las mediciones como podría ser, cuando se produce cambios estructurales en el edificio o modificaciones considerables en la infraestructura de la red inalámbrica, por ejemplo, mueven todos los APs de lugar.

Ya una vez hecho el barrido de señales, teniendo todas las mediciones hechas, se le debe asignar a cada sector una medida característica que represente la señal de un AP en esa ubicación, para ello se promedia todas las intensidades recibidas de cada AP en cada sector. Como ejemplo, si se cuenta con tres APs, en cada sector se tendría tres valores de intensidad correspondientes al promedio de intensidades de cada AP como se refleja en la Tabla 6.2 [34].

	AP A	AP B	AP C
Sector 1	A1	B1	C1
Sector 2	A2	B2	C2
Sector 3	A3	B3	C3
Sector n	An	Bn	Cn

Tabla 6.2: Promedio de Intensidades Recibidas de cada AP en Método K-Vecinos

Se debe realizar la medición del dispositivo con respecto a cada AP para determinar la ubicación del mismo, obteniendo como resultado un vector $x_1, x_2, \ldots, x_N$, en donde N es la cantidad de APs. Se supone entonces que $P_{Aj,k}$ es la potencia que en el sector Aj se midió del AP k.

Si se efectúa la suma $S_{Aj} = \sqrt{\sum_{k=1}^{N}(P_{Aj,k} - x_k)^2}$ la estimación del punto donde se realizó la medición de señales será aquel sector para la cual S_{Aj} sea mínimo.

También debe determinarse la cantidad de k vecinos a considerar, es decir, los k valores más próximos al valor buscado, si recibirán alguna ponderación y algoritmos adicionales a aplicar en caso de que dos o más vecinos posean la misma distancia respecto al valor buscado e igual ponderación.

Como desventajas de este método se tiene:

- Las fluctuaciones de la fuerza de la señal que se recibe de los APs en un lugar específico a lo largo del tiempo, debido a ondas electromagnéticas, paredes, columnas, otros dispositivos, etc., introducen una serie de saltos indeterminados al momento de calcular la ubicación del usuario. Estos saltos pueden producir por ejemplo que cuando el usuario esté detenido el sistema detecte que se está desplazando alrededor de un lugar, aunque el impacto que tiene este tipo de saltos podría minimizarse usando otros algoritmos o filtros como por ejemplo usar un detector de inercia.
- Puede ocurrir que en varias sectores o áreas del edificio se reciban valores similares de fuerzas de señal de los APs accesibles desde dichos puntos, lo que podría provocar que el algoritmo ubique al usuario en cualquiera de esas ubicaciones.
- Con este método por sí sólo no se puede trazar correctamente la ruta que sigue un usuario. Debido a problemas ya mencionados como los saltos en la fuerza de la señal si el usuario está en movimiento, es probable que se obtengan localizaciones físicamente imposibles si se toma en cuenta la velocidad de desplazamiento que tiene un usuario. Una manera de resolver este inconveniente es la de restringir los movimientos del usuario a los n puntos más cercanos a la localización que se había calculado anteriormente, así, no sería posible que el usuario se desplace en dos instantes de tiempo consecutivos a dos ubicaciones que no sean cercanas en el edificio.

Las ventajas que tiene este método son las siguientes:

- El algoritmo de localización no requiere de un costo computacional alto, la capacidad de cómputo que se requiere es asumible por los dispositivos móviles.
- A mayor número de APs con los que se trabaje y de mediciones hechas en la fase de f*ingerprinting*, mayor es la precisión del método. Se podría aumentar considerablemente el rendimiento del sistema incrementando la densidad de los APs en el edificio con los que se trabaja, aunque una densidad muy grande de APs aumentará levemente el tiempo de respuesta del algoritmo.
- Con este método no necesariamente se debe conocer la ubicación de los APs en el edificio, lo cual evita un estudio de la infraestructura de red inalámbrica. Está implementado de forma tal que se ajusta eficazmente a cualquier configuración de red

inalámbrica real que exista en el entorno, no siendo necesario que se modifique la misma para crear un entorno controlado. Sin embargo, lo más recomendable es que se realicé el estudio de infraestructura de red y si es posible antes de la fase de *fingerprinting* se ubiqué los APs en ubicaciones donde sus ondas tengan más áreas de cobertura y se saque mayor provecho de las fuerzas de las señales que emiten.

- No es necesario que el dispositivo se conecte a una red WiFi determinada para calcular su ubicación, con sólo tener el WiFi encendido el dispositivo obtiene de los diferentes APs la información que requiere como por ejemplo la dirección MAC (Media Access Control), el SSID (Service Set Identifier) y la fuerza de la señal recibida por cada AP, para crear la base de datos en la fase de fingerprintig y en el proceso de geolocalización en donde se estima la ubicación del usuario.

Como se explica en [35], también es importante tomar en cuenta algunas consideraciones acerca del algoritmo:

- Si al momento en que se escanea la red para empezar a calcular la ubicación se obtienen APs que no están en la base de datos creada durante el proceso de fingerprinting, se eliminan dichos APs y no se tienen en cuenta durante el resto del algoritmo de k-vecinos.
- En la función que halla la distancia del punto en donde se encuentra el usuario y los puntos de la base de datos de *fingerprinting*, se especifica que si el punto con el que se está intentando hallar la distancia no contiene el valor de fuerza de la señal para uno de los APs que se ha localizado en el escaneo, se debe a devolver una distancia máxima, de modo que, al elegir los mínimos, ese valor se ignorará, puesto que ese punto no puede ser de los buscados.
- Durante el proceso de *fingerprinting*, al tomar valores de señal en un punto, se toma varias medidas en el mismo lugar, con lo cual se obtienen varios valores de fuerza de la señal del mismo AP en el mismo punto, así que para que el algoritmo sea eficiente, se almacena la media de dichas fuerzas, y ese es el valor que se toma en cuenta al calcular las distancias.

7. Trabajos Relacionados

A continuación, se presentan trabajos relacionados con la presente investigación. El motivo para describir dichos trabajos es que tienen enfoques muy semejantes a la aplicación móvil en que se centra el estudio y cuentan con funcionalidades y características que se desean contemplar en el desarrollo de la misma. Además de que dan una idea a los autores sobre qué requerimientos se deben satisfacer para obtener una aplicación competitiva y realmente útil para los usuarios, por medio del análisis de las funciones implementadas y los resultados obtenidos de cada uno.

7.1. Larcomaps

Larcomaps es una aplicación móvil desarrollada para el centro comercial Larcomar, ubicado en el distrito de Miraflores en Lima Perú. Cuenta con una versión para Android, así como también con una versión para iOS, las cuales se pueden descargar directamente desde sus respectivos markets, como la Play Store para Android y la App Store para iOS. Esta aplicación se encarga de ofrecer el mapa de las instalaciones del centro comercial, con todas sus tiendas, el estacionamiento, etc.; además, ofrece un servicio de rutas que le muestra al usuario el camino que debe recorrer para llegar a un cierto destino, ya sea una tienda, un baño o cualquier otra ubicación interna del centro comercial. En la Figura 7.1 tomada de la descripción de la aplicación en el portal de la plataforma Google Play[13] se muestra un ejemplo de cómo la aplicación muestra la ruta hacia la tienda a la que el usuario desea llegar.

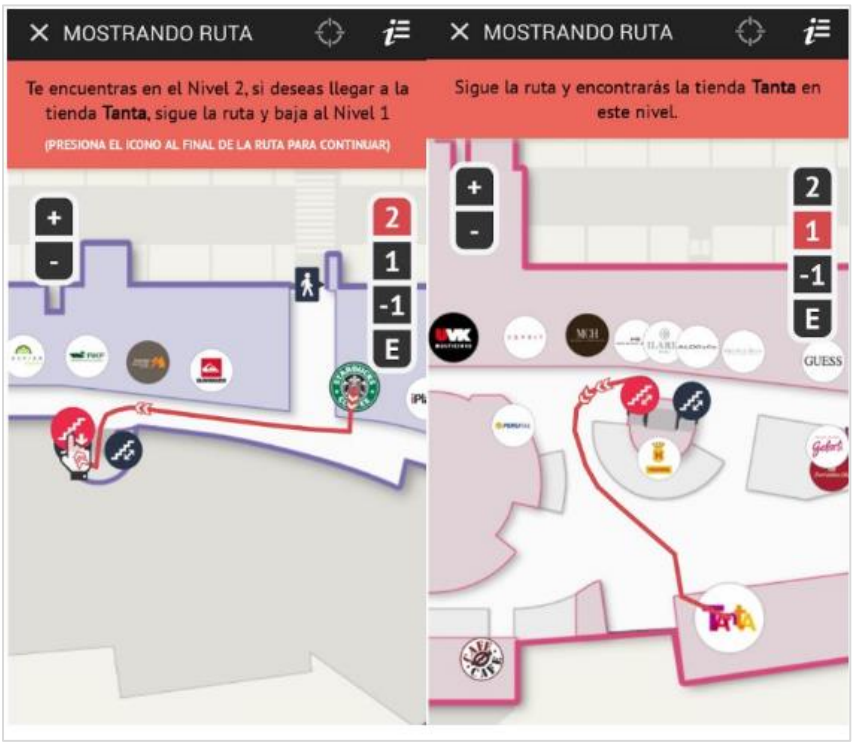

Figura 7.1: Pantalla de Mostrar Ruta para Llegar a una Tienda en Larcomaps

[13] https://play.google.com/store/apps/details?id=cl.acid.larcomar_mapps&hl=es_419

Esta aplicación le permite al usuario la actualización permanente de su ubicación dentro del centro comercial, lo que le permite recibir instrucciones que lo guiarán a lo largo del recorrido dentro del mismo. Además, permite mostrar un directorio de las tiendas y servicios ofrecidos dentro del centro comercial organizado por orden alfabético o por categorías, mostrado en la Figura 7.2 tomada también de la plataforma Google Play.

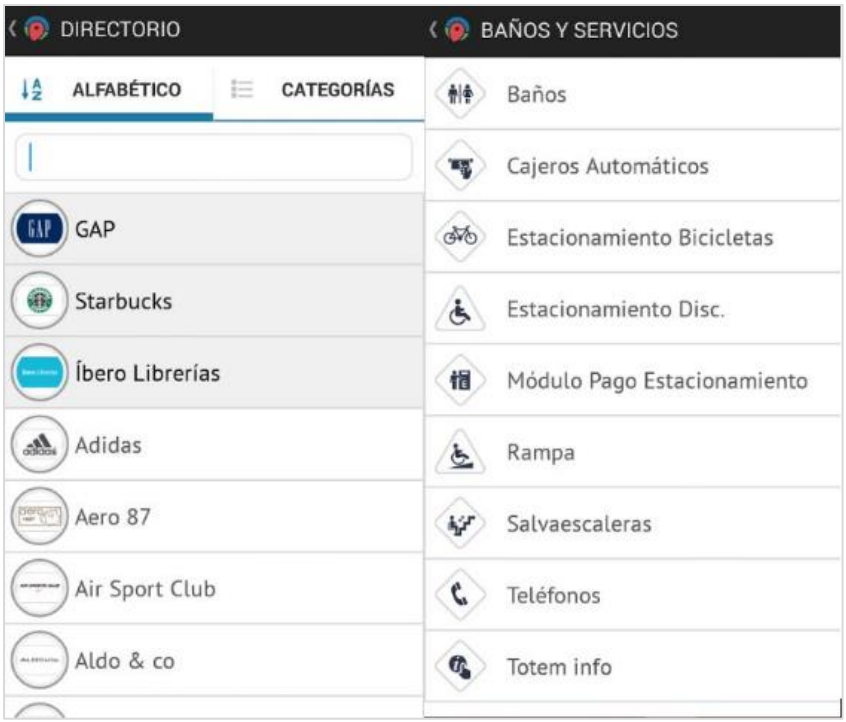

Figura 7.2: Pantallas de Directorio de Tiendas y Baños y Servicios en Larcomaps

Otro servicio que ofrece, el cual es muy importante para los usuarios del centro comercial, es aquel que le permite al usuario agregar un recordatorio del lugar en donde estaciona su auto, así como también le permite guiarlo mostrándole la ruta al mismo, al momento de dirigirse a su vehículo.

7.2. Plaza–Mall App

El centro comercial Plaza Las Américas ubicado en San Juan, Puerto Rico, cuenta con su propia aplicación móvil desarrollada para los sistemas operativos iOS y Android, disponible de forma gratuita en sus respectivas tiendas de aplicaciones y con la facilidad de poder usarse tanto en inglés como en español. Por de medio Plaza-Mall App, los usuarios podrán acceder al directorio de tiendas, información de los servicios del centro comercial, mapas, ofertas y promociones de las tiendas, entre otras funciones. Los usuarios pueden comprar tarjetas de regalo del centro comercial directamente desde la aplicación y también tienen la posibilidad de iniciar una sesión en Facebook o registrarse para recibir ofertas e información directamente en su dirección de correo electrónico. El usuario puede guardar la información de donde estacionó su vehículo en el estacionamiento del centro comercial ya sea tomando una foto, grabando un mensaje de voz o marcando su ubicación con un pin.

En la Figura 7.3 tomada de un reportaje acerca de la aplicación en la revista digital qiibo[14], se muestra el menú de la app, en donde se puede apreciar las funcionalidades principales con las que cuenta.

Figura 7.3: Pantalla de Menú en Plaza-Mall App

En la Figura 7.4 tomada de la descripción de la aplicación en el portal de la plataforma Google Play[15], se puede ver cómo está organizado el directorio de tiendas y como muestra el mapa del centro comercial para ubicar lugares específicos.

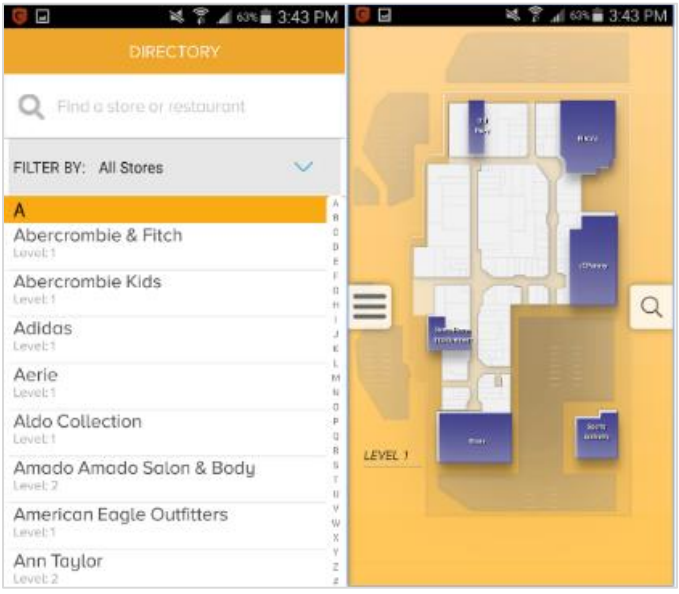

Figura 7.4: Pantallas de Directorio y Mapas de Plaza-Mall App

[14] http://www.qiibo.com/2015/08/25/plaza-las-americas-aplicacion-iphone-y-android
[15] https://play.google.com/store/apps/details?id=com.empresas.plazalasamericas.android

En la Figura 7.5, tomada también de la plataforma Google Play, se muestra la pantalla de promociones de la aplicación.

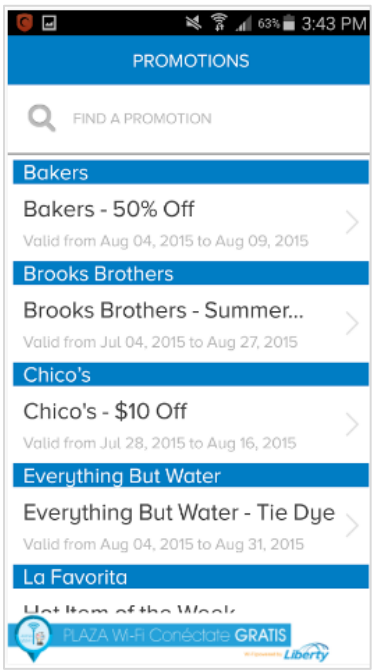

Figura 7.5: Pantalla de Promociones de Plaza-Mall App

7.3. Indoor Location

Es una aplicación desarrollada por un grupo de estudiantes de la Universidad Complutense de Madrid la cual permite geolocalizar dispositivos móviles en interiores usando redes WiFi. Esta aplicación permite realizar todo el proceso de *fingerprinting* así como también el visor de geolocalización de clientes. Indoor Location está desarrollada sobre Java y usa Swing y AWT (Abstract Window Toolkit) como librerías gráficas, lo que la hace una aplicación totalmente multiplataforma. En la Figura 7.6 [35], se muestra la pantalla de información de la aplicación.

Figura 7.6: Pantalla de Información de Indoor Location

Indoor Location permite realizar sus diferentes funciones de una manera sencilla. Para realizar el proceso de fingerprinting en la aplicación, se hace clic en la opción “Escanear” que se encuentra en la pestaña “Vista” del menú de opciones, y a su vez se selecciona en el mapa la posición de origen para el fingerprinting, luego se realizan tantos escaneos (cada 2 segundos) como el usuario haya indicado. En caso de querer cambiar de punto del mapa, simplemente se selecciona una nueva posición pulsando en el mapa o se usan los botones de desplazamiento.

Los datos obtenidos se mostrarán en la tabla superior, indicando la posición de escaneo, y los puntos de acceso encontrados, con sus fuerzas, dirección MAC y su SSID. Además, en el mapa de la izquierda, como se muestra en la Figura 7.7 [35], aparecen en azul los puntos donde se han realizado escaneos.

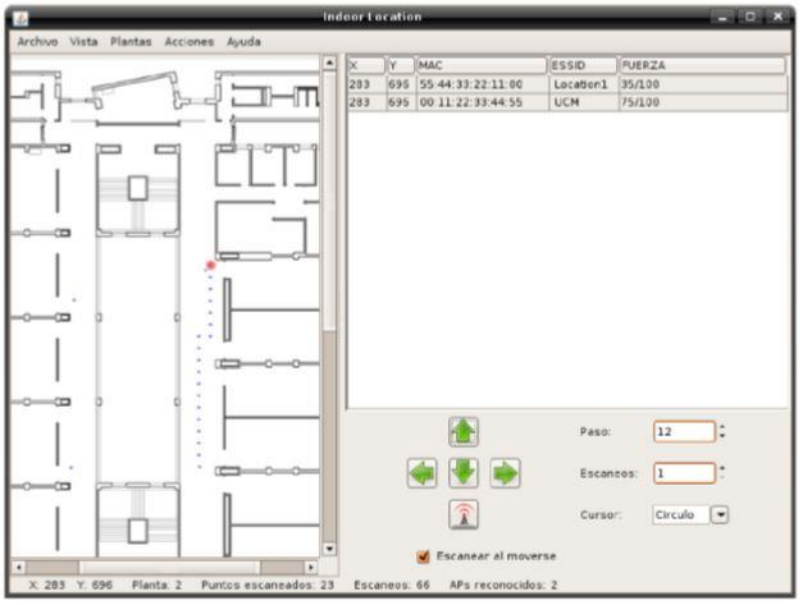

Figura 7.7: Pantalla de Escaneos de Indoor Location

Luego, para realizar el proceso de la geolocalización se dirige a la vista de “Localización”, para poder ver la posición del usuario dentro del lugar donde se estén realizando las pruebas. La aplicación realizará la geolocalización de manera automática, repitiéndose cada cierto tiempo, el cual es definido por el usuario, o también se puede forzar la geolocalización en un momento dando seleccionando la opción “Localizar ahora”.

Al pulsar el botón de “Localizar ahora”, se escanea la red y se pasan los resultados obtenidos, mostrados en la Figura 7.8 [35], al algoritmo de k-vecinos más cercanos, y se halla el baricentro para obtener la posición.

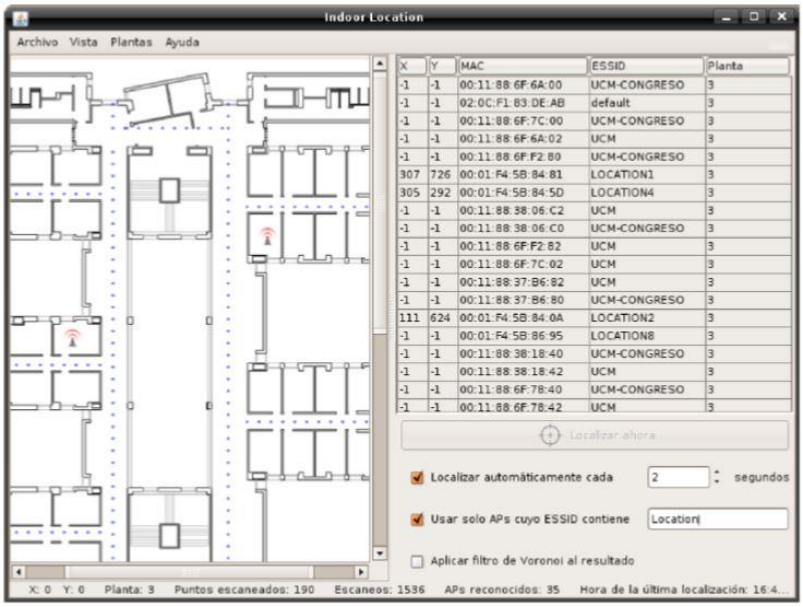

Figura 7.8: Pantalla de Localización de Indoor Location

7.4. ProMotion App

Es una aplicación desarrollada por Pedro Díaz y Edwin Alvarado, como Trabajo Especial de Grado para obtener la Licenciatura en Computación en la Universidad Central de Venezuela. En esta aplicación se encuentran todas las tiendas y comercios afiliados al sistema, en donde se podrán ver promociones de dichos comercios, los cuales se muestran al usuario por medio de una notificación al acercarse al local respectivo. Las notificaciones de promociones aparecen en la barra superior del dispositivo móvil al momento de estar en un rango de 0-30 metros de los comercios asociados, utilizando la tecnología *Bluetooth Low* Energy y dispositivos Beacons como método de localización, como se observa en la Figura 7.9 [36].

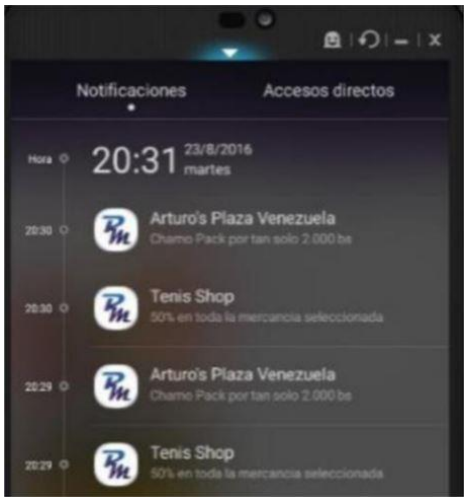

Figura 7.9: Pantalla de Barra de Notificaciones de ProMotion App

Al iniciar la aplicación se encuentra en primera plana un banner donde se observan las promociones más destacadas y un filtro de comercios por categorías, que al darle clic a una de estas promociones o comercios, la aplicación mostrará el detalle de la promoción o el comercio seleccionado. Además, la aplicación cuenta con una función muy útil, a la cual se puede acceder por medio del menú haciendo clic en la opción “Más cerca”, ésta consiste en mostrar en un mapa la ruta del comercio más cercano asociado al sistema, así como también la ubicación exacta de cada una de los comercios afiliados a la aplicación.

8. Marco Metodológico

Entendiendo que para lograr el desarrollo de una solución tecnológica de la manera más estructurada, planificada y controlada posible es necesario tener y seguir una metodología de desarrollo de software, se describe en este capítulo la metodología que fue implementada en la elaboración de este proyecto, así como el conjunto de herramientas y tecnologías que fueron utilizadas para alcanzar los objetivos planteados en el Capítulo 2.

8.1. Adaptación de la Metodología de Desarrollo

Para el proceso de desarrollo de las aplicaciones tanto móvil como web se implementó una metodología de desarrollo ágil llamada Scrum. Esta metodología centra su atención en las actividades de Gerencia y no especifica prácticas de Ingeniería. Fomenta el surgimiento de equipos auto dirigidos cooperativos y aplica inspecciones frecuentes como mecanismo de control [37].

El ciclo de vida definido por Scrum es incremental iterativo y se caracteriza por ser muy adaptable, siendo este último punto importante de recalcar, ya que existen aspectos definidos sobre esta metodología de desarrollo de software que no se adaptaron a las características y necesidades de los autores en éste proyecto por lo que fueron modificados u omitidos.

Para la gestión de todos los aspectos relacionados con la metodología, se utilizó de principio a fin una aplicación web llamada Teamwork[16], la cual es una herramienta flexible y con muchas características que permitieron adaptar la metodología a las funcionalidades que ofrece la plataforma. En la Figura 8.1, tomada por los autores de la aplicación Teamwork para gestionar el presente proyecto, se puede apreciar cómo es la pantalla principal al momento de iniciar sesión en un determinado proyecto, proporcionando un vistazo general de las últimas actividades que se han realizado haciendo uso de la aplicación.

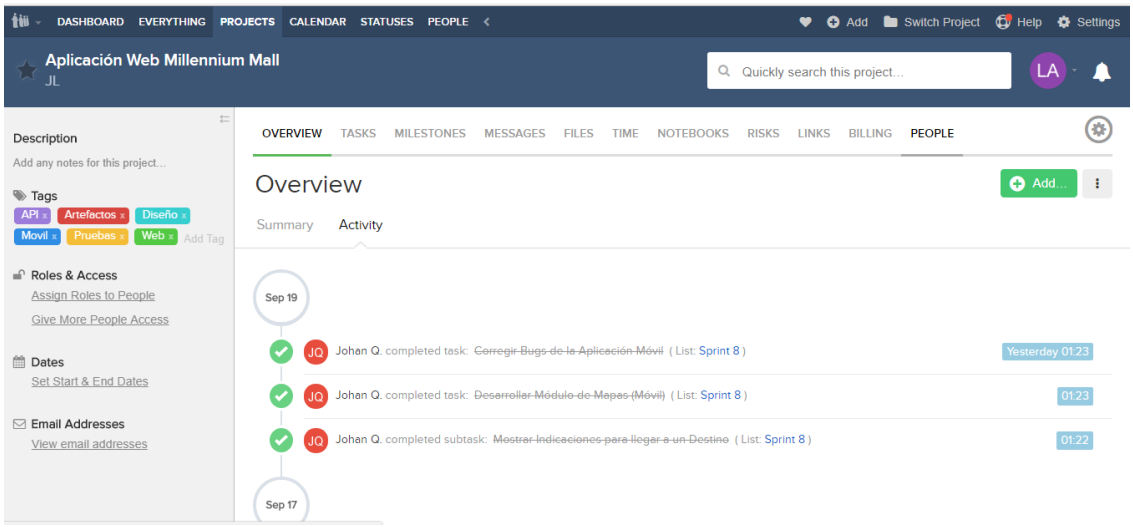

Figura 8.1: Pantalla Principal de un Proyecto en Teamwork

[16] https://www.teamwork.com/?ref=i_semanticlabs

El marco técnico de la metodología Scrum utilizada en el presente proyecto está formado por:

8.1.1. Roles

Todos los roles que siguen a continuación y que forman parte de la metodología fueron asumidos por los autores (Luinel Andrade, Johan Quintero). Se describe brevemente según [37] las responsabilidades de cada uno de ellos:

- **Productor Owner:** Es el responsable del proyecto, administra, controla y comunica la Product Backlog List, reflejando allí la visión del producto.
- **Scrum Master:** Es un rol de administración que debe asegurar que el proyecto se está llevando a cabo de acuerdo con las prácticas, valores y reglas de Scrum y que todo funciona según lo planeado. Su principal trabajo es remover impedimentos y reducir riesgos del producto. Generalmente es desempeñado por un Gerente de Proyecto o Líder de equipo.
- **Scrum Team:** Es el equipo del proyecto que tiene la potestad de decidir cómo organizarse para cumplir con los objetivos de un sprint. Entre las tareas que tienen están: estimar Esfuerzo, crear el Sprint Backlog, revisar la Product Backlog List y sugerir qué obstáculos que deberán ser removidos para cumplir con las tareas que aparecen.

8.1.2. Artefactos

Scrum especifica artefactos y herramientas de tipo gerencial que se aplican en sus distintas fases para organizar las actividades a realizar y que por sus características suelen reducir la gran cantidad de artefactos que requieren otras metodologías.

- **Product Backlog List:** Puede verse como una lista priorizada de los requerimientos sobre el producto y los cuales definen el trabajo que debe realizarse en el proyecto. Puede resultar complicado establecer desde un comienzo todos los requerimientos a cumplir, pero es suficiente con establecer los más importantes para llevar a cabo un sprint. Está lista puede modificarse en el transcurso del proyecto a medida que se conoce acerca del producto y las necesidades del cliente, lo único es que dicha lista solo puede cambiarse al culminar un sprint y antes de empezar otro.
  En el proyecto fue vista e implementada como un listado de requerimientos funcionales y no funcionales que se hicieron tanto para la aplicación web como para la aplicación móvil.
- **Sprint Backlog List:** En la metodología suele verse como una lista de ítems de la Product Backlog List que serán implementados en el siguiente sprint, aunque en el proyecto fue adaptada como una lista de tareas a realizar para cumplir con los requerimientos establecidos. Las tareas son establecidas y asignadas por el Scrum Team, el Scrum Master y el Product Owner a partir de la priorización de las tareas y los objetivos que se desean cumplir para ese sprint. El Scrum Team es el que se organiza y determina que tareas se deben desempeñar, pudiendo agregar o remover tareas en cualquier momento si se considera necesario.
  Como ejemplo, en la Figura 8.2 se puede apreciar cómo fue llevada a cabo en el proyecto los distintos Sprint Backlog List haciendo uso de la aplicación Teamwork. En el listado de tareas se refleja a que sprint están asociadas, quienes deben llevarlas a cabo, el tiempo estimado, etiquetas para categorizarlas y prioridades de las mismas.

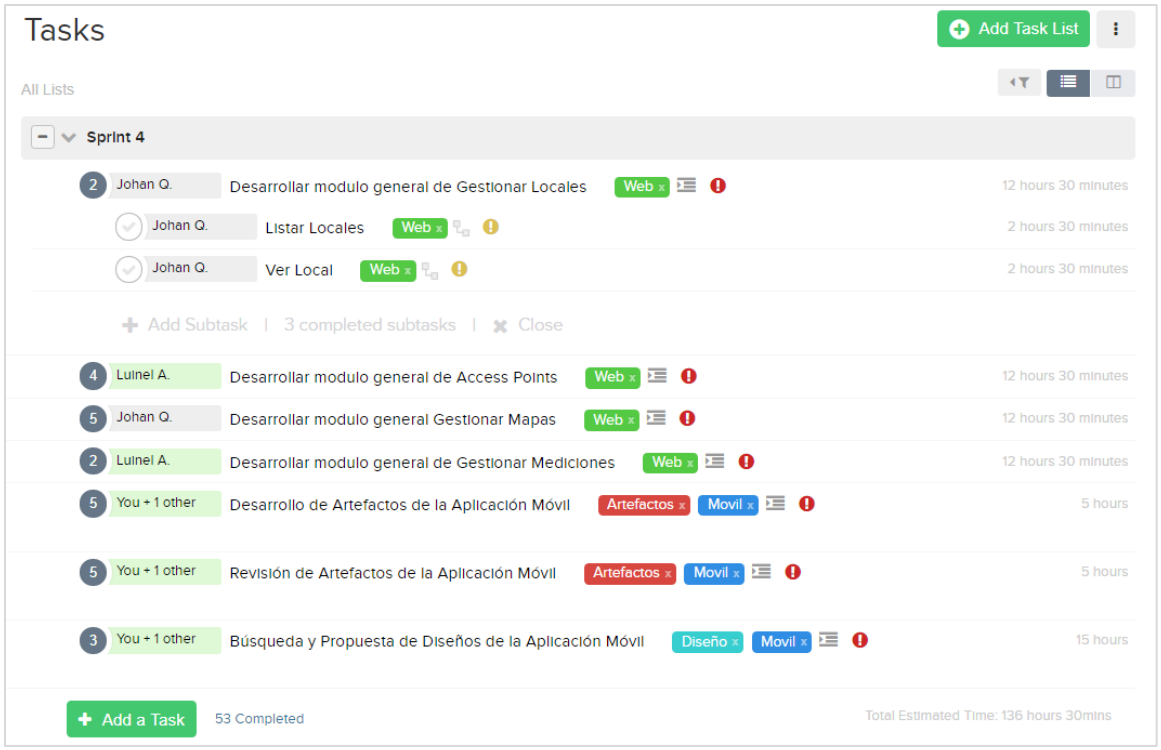

Figura 8.2: Pantalla que Refleja un Sprint Backlog List del Proyecto

8.1.3. Eventos

Los eventos de la metodología Scrum que fueron celebrados en el proyecto fueron:

- **Sprint:** Nombre que recibe cada iteración de desarrollo. La arquitectura, funcionalidad y diseño del producto evolucionan durante el desarrollo del sprint ya que mientras transcurre el mismo, el producto es diseñado, codificado y probado.
  Para el desarrollo del proyecto se estableció que la duración de los sprints sería de 3 semanas cada uno, entre los días lunes y viernes con un tiempo de trabajo estimado en 4 horas diarias mínimas por parte de cada uno de los autores.
- **Sprint Planning Meeting:** Reunión que se lleva a cabo antes del inicio de cada sprint en la cual se determina cuál es el objetivo del sprint y las tareas necesarias para conseguirlo. Se llevaron a cabo los días domingo, justo el día antes de comenzar un nuevo sprint por medio de herramientas informáticas que permiten la comunicación a distancia a través de internet como Whatsapp que es una aplicación de mensajería instantánea que se apoya también en otro tipo de servicios como la transmisión de archivos multimedia y Skype que es una aplicación que permite la comunicación por voz, texto y vídeo.
  En la Figura 8.3 se puede visualizar la planificación en la aplicación Teamwork de los objetivos de un sprint para cada uno de los autores que surge del Sprint Planning Meeting. En este caso particular se muestra como ejemplo la del sprint 2.

Figura 8.3: Pantalla de Objetivos de un Sprint que Surge de un Sprint Planning Meeting

- **Daily Scrum:** Reunión de corto tiempo entre los integrantes del equipo que se lleva a cabo diariamente para dar a conocer el estado del proyecto, básicamente en estas reuniones cada miembro responde a tres interrogantes: ¿Qué hiciste ayer?, ¿Qué harás hoy? Y ¿Has tenido algún problema que te haya impedido alcanzar tu objetivo?
  Durante el transcurso del proyecto los dailys fueron registrados en una libreta virtual que proporciona la aplicación Teamwork como se puede apreciar en la Figura 8.4.

Viernes 26/05

Luinel:

- **Ayer:** Desarrollo del Modificar Perfil.
- **Inconvenientes:** Sin mayor inconveniente. Errores propios de una actividad de desarrollo.
- **Hoy:** Desarrollo del Detalle de Perfil y de Eliminar Perfil.

Johan:

- **Ayer:** Desarrollo del Modificar Usuario.
- **Inconvenientes:** Ninguno.
- **Hoy:** Desarrollo del Detalle de Usuario y de Eliminar Usuario.

Lunes 29/05

Luinel:

- **Viernes:** Desarrollo del Detalle de Perfil y de Eliminar Perfil.
- **Inconvenientes:** Sin mayor inconveniente. Errores propios de una actividad de desarrollo.
- **Hoy:** Empezar con el desarrollo del modulo general para Gestionar Categorías.

Johan:

- **Viernes:** Desarrollo del Detalle de Usuario y de Eliminar Usuario.
- **Inconvenientes:** Ninguno.
- **Hoy:** Desarrollar Acerca de la Aplicación y Desarrollar Información del Centro Comercial

Martes 30/05

Figura 8.4: Libreta de Reporte de Dailys del Proyecto

- **Sprint Review Meeting:** Reunión llevada a cabo para analizar e inspeccionar la evolución del producto. En ese encuentro se revisa que trabajos fueron completados y cuáles no. Al igual que los Sprint Planning Meeting se llevaron a cabo los días domingo por medio de las ya mencionadas herramientas informáticas Whatsapp y Skype.

8.2. Tecnologías Utilizadas

En la presente sección y apoyándose en lo explicado a lo largo del Capítulo 6, se describe brevemente las tecnologías y herramientas de desarrollo utilizadas para llevar a cabo las aplicaciones en las que se centra la presente investigación:

- **Android SDK (Software Development Kit):** Conjunto de herramientas de desarrollo de software que permite a los desarrolladores crear aplicaciones Android nativas de una forma más sencilla.
- **Android Studio:** IDE por excelencia para el desarrollo de aplicaciones Android nativas. Utilizado para el presente proyecto, ya que se enfoca exclusivamente en el desarrollo de aplicaciones Android y provee muchas herramientas y funciones.
- **Git:** Software de control de versiones que permite el mantenimiento de aplicaciones cuando éstas tienen una gran cantidad de archivos de código fuente. Es ampliamente usado a nivel mundial por lo eficiente y confiable que es la herramienta. En el proyecto fue utilizado tanto para ambas aplicaciones.
- **Hostinger[17]:** Plataforma de alojamiento web que cuenta con servicios gratuitos y con planes pagos. Las características de los servicios gratuitos de esta plataforma fueron suficientes para desplegar la aplicación web durante el proceso de desarrollo del proyecto.

[17] https://www.hostinger.es/

- **Java:** Lenguaje de programación de propósito general, orientado a objetos y concurrente. Utilizado para el desarrollo de la aplicación móvil en la que se centra el presente estudio, debido a que es el lenguaje oficial usado para desarrollar aplicaciones móviles nativas en el sistema operativo Android.
- **JSON (JavaScript Object Notation):** Formato ligero de intercambio de datos. Si bien es un subconjunto de la notación literal de objetos de JavaScript, su formato de texto es independiente del lenguaje de programación que lo utiliza. Hoy en día es la principal alternativa ante XML, entre algunas de las razones está que es más ligero y más fácil más fácil de analizar sintácticamente. Fue el formato utilizado en las respuestas de la API implementada.
- **Justinmind Prototyper:** Herramienta para realizar prototipos de aplicaciones web y móviles y *wireframes* para sitios web de alta fidelidad. Fue utilizada para realizar los diseños de las interfaces de usuario de ambas aplicaciones que conforman la solución.
- **Laravel:** Framework lanzado oficialmente en el año 2011, de código abierto, multiplataforma y que permite desarrollar aplicaciones y servicios web con el lenguaje de programación PHP. Utilizado en el proyecto para desarrollar la aplicación web que funciona como página de administración. Además, para el Front-End de la aplicación web también permitió el uso de tecnologías como HTML, CSS y JS.
  La API a desarrollar que permite que la aplicación móvil propuesta acceda a los datos que se gestionarán desde la aplicación web también se realizará con este framework.
- **MySQL:** Es uno de los SMBD más populares a nivel mundial. Se puede usar una versión *community* bajo la Licencia Pública General de GNU o una versión empresarial bajo una licencia comercial que se debe obtener de Oracle. Para la persistencia de los datos se utilizó este SMBD, entre algunas razones para dicha elección está que es soportado por una gran cantidad de lenguajes de programación, es ampliamente usado en aplicaciones web y que es muy estable en aplicaciones donde hay baja concurrencia en la modificación de datos, pero si un alto nivel de lectura de datos, lo cual se adaptó a las necesidades del proyecto.
- **PHP (PHP Hypertext Preprocessor):** Lenguaje de programación que fue concebido para el desarrollo web de contenido dinámico. Este lenguaje se centra en código del lado del servidor y es de propósito general, es decir que puede ser usado para varios propósitos como acceso a bases de datos, comunicación entre computadores, captura de datos, cálculos matemáticos, diseño de páginas, etc.
- **Postman:** Cliente HTTP que permite probar servicios web fácilmente, basta con indicar la URL, el método HTTP y los parámetros de la petición. Fue utilizado para hacer probar y depurar la API desarrollada.
- **Sublime Text:** Editor de texto y editor de código fuente que fue escrito en C++ y Python para los plugins. Fue el IDE utilizado para el desarrollo de la aplicación web.
- **Teamwork:** Plataforma de gestión de proyectos que se utilizó en el todo el proceso de desarrollo del proyecto y que se caracteriza por ser una herramienta flexible y con muchas funcionalidades que permitieron adaptar la metodología a las necesidades de los autores.
- **WiFi:** Tecnología de comunicación basada en el estándar 802.11 del organismo IEEE (Institute of Electrical and Electronics Engineers) que permite establecer un mecanismo de conexión entre dispositivos electrónicos de forma inalámbrica.
  Fue la tecnología usada para implementar las funciones de geolocalización de la aplicación móvil junto al método de los k-vecinos más cercanos el cual procesa un modelo

de localización llamado Fingerprinting que se basa en medir la potencia de las señales recibidas de cada AP en un lugar en específico.

- **Xampp:** Servidor web de plataforma, software libre, que consiste principalmente en el SMBD MySQL, el servidor web Apache y los intérpretes para lenguajes PHP y Perl. Utilizado como servidor web local para la aplicación web durante los primeros sprints de desarrollo.

9. Marco Aplicativo

En este capítulo se explica el diseño de la solución a través de los sprints definidos en base a la adaptación de la metodología Scrum, descrita en el Capítulo 8. Las actividades de cada sprint se definieron en base a las necesidades y novedades que surgían a medida que se avanzaba en el desarrollo del proyecto y en base a un estimado de horas que llevaría realizar cada una de esas actividades hasta cubrir todos los días que conforman el sprint.

9.1. Arquitectura de la Solución

Antes de comenzar con la explicación de lo que fue el proceso desarrollo de cada uno de las aplicaciones, en la Figura 9.1 se muestra la arquitectura general de la solución tecnológica, para así conocer cómo es la integración de cada uno de los elementos que conforman dicha solución. Los elementos que conforman la arquitectura son:

- La Aplicación Web desarrollada para ser usada por el personal administrativo del centro comercial y que es accedida desde un navegador web. Fue desarrollada haciendo uso del lenguaje de programación PHP y el framework Laravel en su versión 5.2.
- La Persistencia de datos haciendo uso del SMBD MySQL.
- La Aplicación Móvil desarrollada para dispositivos con sistema operativo Android a partir de su versión 4.0.3.
- La API REST desarrollada también con el framework Laravel y que funciona como mecanismo de comunicación entre la aplicación web y la aplicación móvil.
- Los Access Points disponibles en los interiores del centro comercial que sirven para medir la potencia de las señales y así procesar el algoritmo de geolocalización. Es importante mencionar que no estuvo al alcance de los autores el poder trabajar con una infraestructura de red propia del centro comercial, por lo que se tuvo que trabajar con los APs pertenecientes a las tiendas y de los cuales no se conocía ninguna de sus especificaciones técnicas.
- El Servidor Web, en donde se aloja la aplicación web, la API y la BD. Se utilizó la plataforma de alojamiento web Hostinger, la cual cuenta con servicios gratuitos que fueron suficientes para desplegar la solución tecnológica durante el proceso de desarrollo.

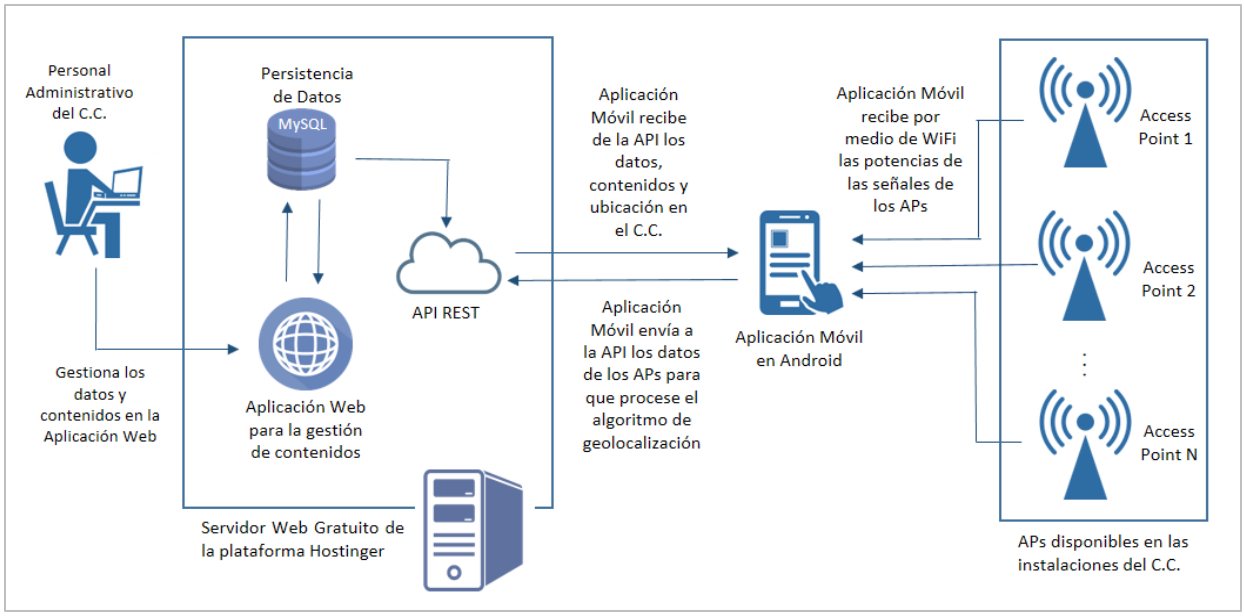

Figura 9.1: Arquitectura General de la Solución

9.2. Planificación de las Actividades del Proyecto

El desarrollo de la solución tecnológica planteada en el presente trabajo de investigación se llevó a cabo en un total de 8 sprints. Para entender con más claridad cómo fue todo el proceso de desarrollo y las tareas que se propusieron lograr en cada iteración, a continuación, se describen la planificación de cada sprint para cada autor que resultaba de las reuniones conocidas como Sprint Planning Meeting de la metodología Scrum que se llevaban a cabo antes de comenzar cada sprint. En la Tabla 9.1 y en la Tabla 9.2 se puede apreciar la planificación del sprint 0.

Sprint	Sprint 0
Responsable	Johan Quintero
Fecha	10 de abril al 28 de abril de 2017
Actividades	Entregar los siguientes artefactos de la aplicación web: • Lista de Requerimientos Funcionales y No Funcionales. • Guía de Estilo. • Modelo de Entidad-Relación. • Modelo Relacional. Además, se deben entregar los siguientes diseños: • Footer. • Inicio de Sesión. • Recuperar Contraseña. • Acerca de la Aplicación. • Módulo Comercios. • Módulo Categorías. • Módulo Promociones. • Módulo Sectores. • Módulo Mediciones. • Módulo Usuarios. • Agregar Perfil y Modificar Perfil.

Tabla 9.1: Sprint 0 – Johan Quintero

Sprint	Sprint 0
Responsable	Luinel Andrade
Fecha	10 de abril al 28 de abril de 2017
Actividades	Entregar los siguientes artefactos de la aplicación web: • Objetivo. • Alcance. • Diagrama de Casos de uso. • Descripción de los Casos de Uso. Además, se deben entregar los siguientes diseños: • Logo. • Menú Lateral. • Header. • "Mi Perfil". • Mensajes de Aplicación. • Módulo Información del Centro Comercial. • Módulo Servicios. • Módulo Eventos. • Módulo Locales. • Módulo Access Points. • Módulo Mapas. • Módulo Perfiles.

Tabla 9.2: Sprint 0 – Luinel Andrade

En la Tabla 9.3 y en la Tabla 9.4 se muestra la planificación del sprint 1.

Sprint	Sprint 1
Responsable	Johan Quintero
Fecha	01 de mayo al 19 de mayo de 2017
Actividades	Revisar y arreglar de ser necesario siguientes diseños: • Logo. • Menú Lateral. • Header. • "Mi Perfil". • Mensajes de Aplicación. • Módulo Información del Centro Comercial. • Módulo Servicios. • Módulo Eventos. • Módulo Locales. • Módulo Access Points. • Módulo Mapas. • Módulo Perfiles. También se debe instalar y configurar el ambiente de trabajo, incluyendo: • Servidor Apache. • Servidor Apache • BD MySQL • PHP • Composer y Laravel • Git • IDE

Tabla 9.3: Sprint 1 – Johan Quintero

Sprint	Sprint 1
Responsable	Luinel Andrade
Fecha	01 de mayo al 19 de mayo de 2017
Actividades	Revisar y arreglar de ser necesario siguientes diseños: • Logo. • Menú Lateral. • Header. • "Mi Perfil". • Mensajes de Aplicación. • Módulo Información del Centro Comercial. • Módulo Servicios. • Módulo Eventos. • Módulo Locales. • Módulo Access Points. • Módulo Mapas. • Módulo Perfiles. También se debe instalar y configurar el ambiente de trabajo, incluyendo: • Servidor Apache. • BD MySQL • PHP • Composer y Laravel • Git • IDE Desarrollar: • Plantilla de la interfaz de la aplicación que contiene Menú, Header y Footer • Inicio de Sesión • Recuperar Contraseña. • Restablecer Contraseña.

Tabla 9.4: Sprint 1 – Luinel Andrade

En la Tabla 9.5 y en la Tabla 9.6 se puede visualizar la planificación del sprint 2.

Sprint	Sprint 2
Responsable	Johan Quintero
Fecha	22 de mayo al 09 de junio de 2017
Actividades	Desarrollar los siguientes módulos de la aplicación web: • Módulo de Acerca de la App. • Módulo de Usuarios • Módulo de Información C.C. • Módulo de Contenidos Destacados • Módulo de Servicios • Módulo de Eventos • Módulo de Mi Perfil

Tabla 9.5: Sprint 2 – Johan Quintero

Sprint	Sprint 2
Responsable	Luinel Andrade
Fecha	22 de mayo al 09 de junio de 2017
Actividades	Desarrollar los siguientes módulos de la aplicación web: • Módulo de Comercios. • Módulo de Categorías • Módulo de Perfiles • Módulo de Promociones

Tabla 9.6: Sprint 2 – Luinel Andrade

En la Tabla 9.7 y en la Tabla 9.8 se puede apreciar la planificación del sprint 3.

Sprint	Sprint 3
Responsable	Johan Quintero
Fecha	12 de junio al 30 de junio de 2017
Actividades	Desarrollar los siguientes módulos de la aplicación web: • Módulo de Mapas. • Módulo de Locales. • Módulo de Sectores (Listar y Ver solamente). Desarrollar las siguientes APIs: • Información C.C. • Contenidos Destacados. • Servicios. • Eventos. • Locales. • Sectores. • Mapas. Entregar los siguientes artefactos de la aplicación móvil: • Lista de Requerimientos Funcionales y No Funcionales. • Objetivo y Alcance. • Guía de Estilo Revisar y arreglar de ser necesario los artefactos de la aplicación móvil. Realizar búsqueda y propuesta de diseños para la Aplicación Móvil.

Tabla 9.7: Sprint 3 – Johan Quintero

Sprint	Sprint 3
Responsable	Luinel Andrade
Fecha	12 de junio al 30 de junio de 2017
Actividades	Desarrollar los siguientes módulos de la aplicación web: • Módulo de Access Points. • Módulo de Mediciones. • Módulo de Inicio. Desarrollar las siguientes APIs: • Comercios. • Categorías. • Promociones. • Access Points. • Mediciones. Entregar los siguientes artefactos de la aplicación móvil: • Diagrama de Casos de Uso. • Descripción del Diagrama de Casos de Uso. Revisar y arreglar de ser necesario los artefactos de la aplicación móvil. Realizar búsqueda y propuesta de diseños para la Aplicación Móvil.

Tabla 9.8: Sprint 3 – Luinel Andrade

En la Tabla 9.9 y en la Tabla 9.10 se muestra la planificación del sprint 4.

Sprint	Sprint 4
Responsable	Johan Quintero
Fecha	03 de julio al 21 de julio de 2017
Actividades	Desarrollar los siguientes diseños de las interfaces de la aplicación móvil: • Menú Principal. • Listado de Categorías. • Listado de Servicios. • Listado de Promociones. • Listado de Eventos. • Filtrar comercio por categoría. • Vista de Mapas. • Información de Contacto. • Acerca de la App. • Mi Puesto de Estacionamiento. Revisar y arreglar de ser necesario los diseños de la aplicación móvil. Instalar y configurar el ambiente de trabajo de la aplicación móvil, incluyendo: • Instalar Android Studio • Instalar Android SDK. • Integrar Android Studio con Gitlab.

Tabla 9.9: Sprint 4 – Johan Quintero

Sprint	Sprint 4
Responsable	Luinel Andrade
Fecha	03 de julio al 21 de julio de 2017
Actividades	Desarrollar los siguientes diseños de las interfaces de la aplicación móvil: • Splash Screen. • Listado de Comercios. • Detalle de Comercio. • Detalle de Seevicio. • Detalle de Promoción. • Detalle de Evento. • Comercios y Promociones de una Categoría. • Inicio. • Ubicación de comercio en el Mapa. • Indicaciones de ¿Cómo llegar a un comercio? • Ubicación de Servicio en el Mapa. • Indicaciones de ¿Cómo llegar a un servicio? • Leyenda de Servicios en el Mapa. • Mensajes de la Aplicación. Revisar y arreglar de ser necesario los diseños de la aplicación móvil. Instalar y configurar el ambiente de trabajo de la aplicación móvil, incluyendo: • Instalar Android Studio • Instalar Android SDK. • Integrar Android Studio con Gitlab.

Tabla 9.10: Sprint 4 – Luinel Andrade

En la Tabla 9.11 y en la Tabla 9.12 se puede visualizar la planificación del sprint 5.

Sprint	Sprint 5
Responsable	Johan Quintero
Fecha	24 de julio al 11 de agosto de 2017
Actividades	Desarrollar los siguientes módulos de la aplicación móvil: • Inicio. • Eventos. • Servicios. • Promociones. • Tu Puesto. • Contacto

Tabla 9.11: Sprint 5 – Johan Quintero

Sprint	Sprint 5
Responsable	Luinel Andrade
Fecha	24 de julio al 11 de agosto de 2017
Actividades	Migrar la Aplicación Web a un servidor remoto. Desarrollar los siguientes módulos de la aplicación móvil: • Splash Screen. • Menú. • Header. • Comercios. • Categorías. • Acerca de la App. • Detalle de Servicio. • Detalle de Comercio. • Detalle de Promoción.

Tabla 9.12: Sprint 5 – Luinel Andrade

En la Tabla 9.13 y en la Tabla 9.14 se puede apreciar la planificación del sprint 6.

Sprint	Sprint 6
Responsable	Johan Quintero
Fecha	14 de agosto al 01 de septiembre de 2017
Actividades	Desarrollar los siguientes módulos de la aplicación móvil: • Detalle de Categoría. • Detalle de Evento. • Ubicación de Servicios en Detalle de Servicio. • Módulo de Mapas (Solo la vista de los Mapas). Realizar el proceso de Fingerprinting del Nivel C1 del C.C. Diseñar los mapas relacionados al Nivel C1 del C.C. Realizar pruebas de la funcionalidad de Geolocalización en el C.C.

Tabla 9.13: Sprint 6 – Johan Quintero

Sprint	Sprint 6
Responsable	Luinel Andrade
Fecha	14 de agosto al 01 de septiembre de 2017
Actividades	Desarrollar los siguientes módulos de la aplicación web: • Módulo de Sectores (Completo). Realizar arreglos sobre los siguientes módulos de la aplicación web: • Módulo de Mapas. • Módulo de Sectores. Implementar algoritmo de geolocalización en la API Realizar el proceso de Fingerprinting del Nivel C1 del C.C. Diseñar los mapas relacionados al Nivel C1 del C.C. Procesar e Ingresar Data de las mediciones de los APs en la BD. Realizar pruebas de la funcionalidad de Geolocalización en el C.C.

Tabla 9.14: Sprint 6 – Luinel Andrade

En la Tabla 9.15 y en la Tabla 9.16 se muestra la planificación del sprint 7.

Sprint	Sprint 7
Responsable	Johan Quintero
Fecha	04 de septiembre al 22 de septiembre de 2017
Actividades	Realizar arreglos sobre los siguientes módulos de la aplicación web: • Módulo de Acces Points. • Módulos de Comercios. • Ubicación de Servicios en Detalle de Servicio. • Módulo de Mapas. Terminar el Módulo de Mapas de la Aplicación Móvil (Indicaciones, Ubicación de Comercios y Servicios). Realizar el proceso de Fingerprinting del Nivel C2 del C.C Realizar pruebas de la funcionalidad de Geolocalización en el C.C. Afinar y corregir algoritmo de Geolocalización. Corregir Bugs de la Aplicación Móvil.

Tabla 9.15: Sprint 7 – Johan Quintero

Sprint	Sprint 7
Responsable	Luinel Andrade
Fecha	04 de septiembre al 22 de septiembre de 2017
Actividades	Realizar el proceso de Fingerprinting del Nivel C2 del C.C. Diseñar los mapas relacionados al Nivel C2 del C.C. Procesar e Ingresar Data de las mediciones de los APs en la BD Realizar pruebas de la funcionalidad de Geolocalización en el C.C. Afinar y corregir algoritmo de Geolocalización. Corregir Bugs de la Aplicación Web

Tabla 9.16: Sprint 7 – Luinel Andrade

9.3. Análisis General de la Aplicación Web

La aplicación web para la gestión de los datos, contenidos y configuraciones de la API que consulta la aplicación móvil fue desarrollada haciendo uso del framework Laravel en su versión 5.2, el cual como ya se ha mencionado en ocasiones anteriores es un framework que permite desarrollar aplicaciones y servicios web con el lenguaje de programación PHP, que para este proyecto en particular fue usada la versión 5.6.30. Además, para el lado del Front-End permite el uso de tecnologías como HTML, CSS y JS. Para la persistencia de datos de la aplicación se usó el SMBD MySQL.

En un principio el desarrollo se llevaba de forma local en el computador de cada autor haciendo uso de la herramienta Git para el control de versiones y el trabajo colaborativo sobre el código de la aplicación y la herramienta Xampp en su versión 3.2.2 de su Control Panel. Posteriormente se efectuó la migración de la aplicación a un servidor web gratuito de la plataforma Hostinger.

De forma general, las funcionalidades que ofrece está aplicación son el poder gestionar los datos que luego serán consumidos por la aplicación móvil a través de la API de los módulos de información de contacto del centro comercial, los comercios, servicios, categorías, promociones, eventos, locales y los datos del sistema de geolocalización que incluye mapas, sectores de los niveles, access points, mediciones de los access points. Además, permite gestionar los usuarios y perfiles de usuario propios de esta aplicación web administrativa.

A continuación, se describe el proceso de desarrollo de la aplicación.

9.3.1. Artefactos y Prototipo General de Interfaz de la Aplicación Web

El sprint 0 marca el comienzo del desarrollo de la aplicación web. En dicho sprint se llevó a cabo un proceso de evaluación y análisis que dio como resultado una serie de artefactos que si bien no forman parte propiamente de la metodología Scrum si son muy comunes en otras metodologías de ingeniería de software, por lo que se decidió realizar algunos de ellos debido a la gran utilidad que tienen para el equipo de desarrollo involucrado durante la fase de desarrollo. También se realizaron los diseños de las interfaces de usuario que tendría la aplicación.

Primero se determinó de manera general los principales requerimientos funcionales y no funcionales para lograr la construcción de la solución web. Estos requerimientos fueron considerados por los autores como la Product Backlog List que toma en cuenta la metodología Scrum. A continuación, se muestra el listado de requerimientos:

- **Requerimientos Funcionales:**
  - Los distintos tipos de usuario podrán iniciar sesión con sus datos en el sistema.
  - Los usuarios Administradores podrán registrar y eliminar a otros usuarios de menor jerarquía en el sistema.
  - Los usuarios Administradores podrán crear y eliminar diferentes perfiles de usuario y modificar los permisos de los mismos sobre los módulos de la aplicación.
  - Los distintos tipos de usuario podrán modificar sus datos personales.
  - Los distintos usuarios con los perfiles de usuario adecuados podrán gestionar la información del centro comercial.
  - Los distintos tipos de usuario con los perfiles de usuario adecuados podrán gestionar contenido destacado en la aplicación móvil.
  - Los distintos usuarios con los perfiles de usuario adecuados podrán gestionar la información de todos los comercios, servicios, categorías, promociones, eventos o locales del centro comercial.
  - Los distintos usuarios con los perfiles de usuario adecuados podrán gestionar la información de los Access Points, Sectores y Mediciones con los que trabaja el sistema de geolocalización de la aplicación móvil.
  - La lógica de la aplicación permitirá poder diferenciar los comercios de los locales. Si bien cada tienda ocupa uno o más locales, puede haber locales vacíos o puede haber una tienda que ocupe varios locales.
  - La lógica de la aplicación permitirá poder identificar y gestionar los kioskos y stands del centro comercial. Los mismos no tienen locales asignados.
  - Los distintos usuarios con los perfiles de usuario adecuados podrán gestionar los mapas de los distintos niveles del centro comercial y gestionar la ubicación de los distintos comercios y servicios.
- **Requerimientos No Funcionales:**
  - Se desarrollará una aplicación fácil de cargar, que no sea tan pesada.

- La aplicación contará con variaciones de tonalidades de los colores: verde, azul y blanco.
- La navegación de la aplicación será sencilla y fácil de entender y manejar.
- La aplicación no contará con ningún tipo de información que no esté relacionada con el centro comercial.

Se diseñó un diagrama de casos de uso en donde para contemplar todos los posibles casos de uso de la solución, se tomó en cuenta un usuario con rol de “Administrador del Sistema” que tiene todos los posibles permisos sobre todos los módulos de la aplicación web. Habrá diferentes tipos de usuario con diferentes permisos, pero las actividades que podrían realizar sería un subconjunto de las que puede realizar un usuario “Administrador del Sistema” que tiene la más alta permisología, por lo que con realizar el diagrama con ese tipo de usuario ya se estaría considerando todos los casos de uso. En la Figura 9.2 se puede apreciar el diagrama de los casos de uso de primer nivel y segundo nivel de la aplicación web.

En la Figura 9.3 se visualiza que el caso de uso de segundo nivel “Gestionar Comercios” tiene asociado cuatro (4) casos de uso que lo que describen es que un usuario puede ver, agregar, modificar y eliminar registros dentro del módulo de comercios. De forma análoga sucede con cada uno de los casos de uso de segundo nivel descritos en el diagrama presentado en la Figura 9.2.

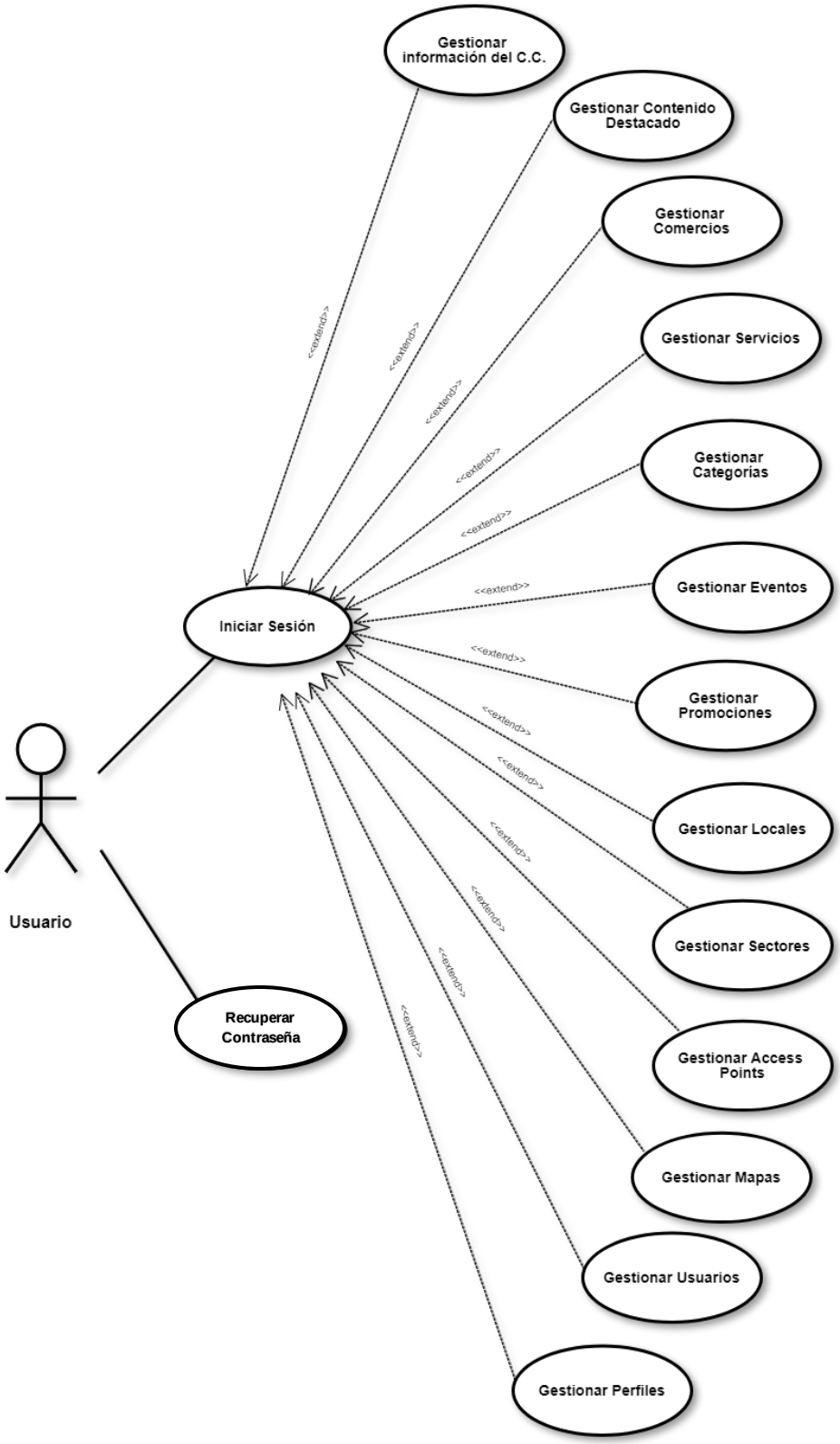

Figura 9.2: Diagrama de Casos de Uso de Primer y Segundo Nivel de la Aplicación Web

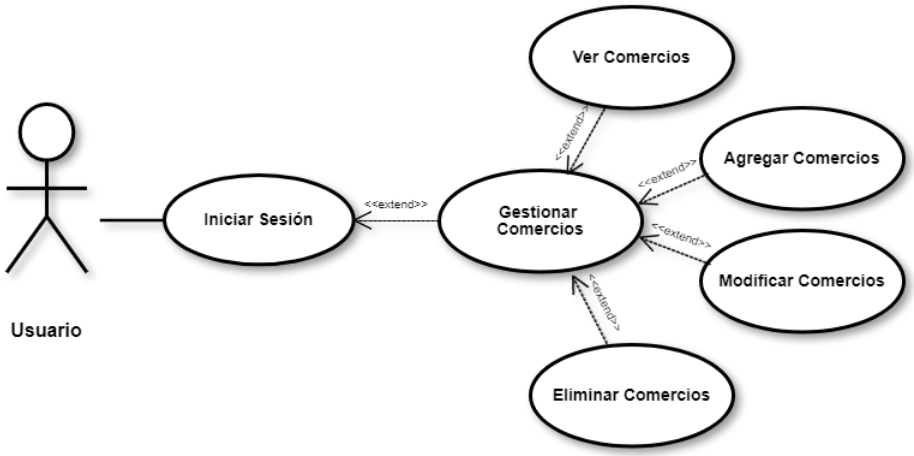

Figura 9.3: Ejemplo de Casos de Uso de Tercer Nivel de la Aplicación Web

En la Figura 9.4 se puede apreciar el diagrama entidad-relación de la aplicación web, dicho diagrama es un modelo de datos que permite obtener una representación visual de un conjunto de objetos denominados entidades y las relaciones entre ellas, que forman parte de un sistema de información.

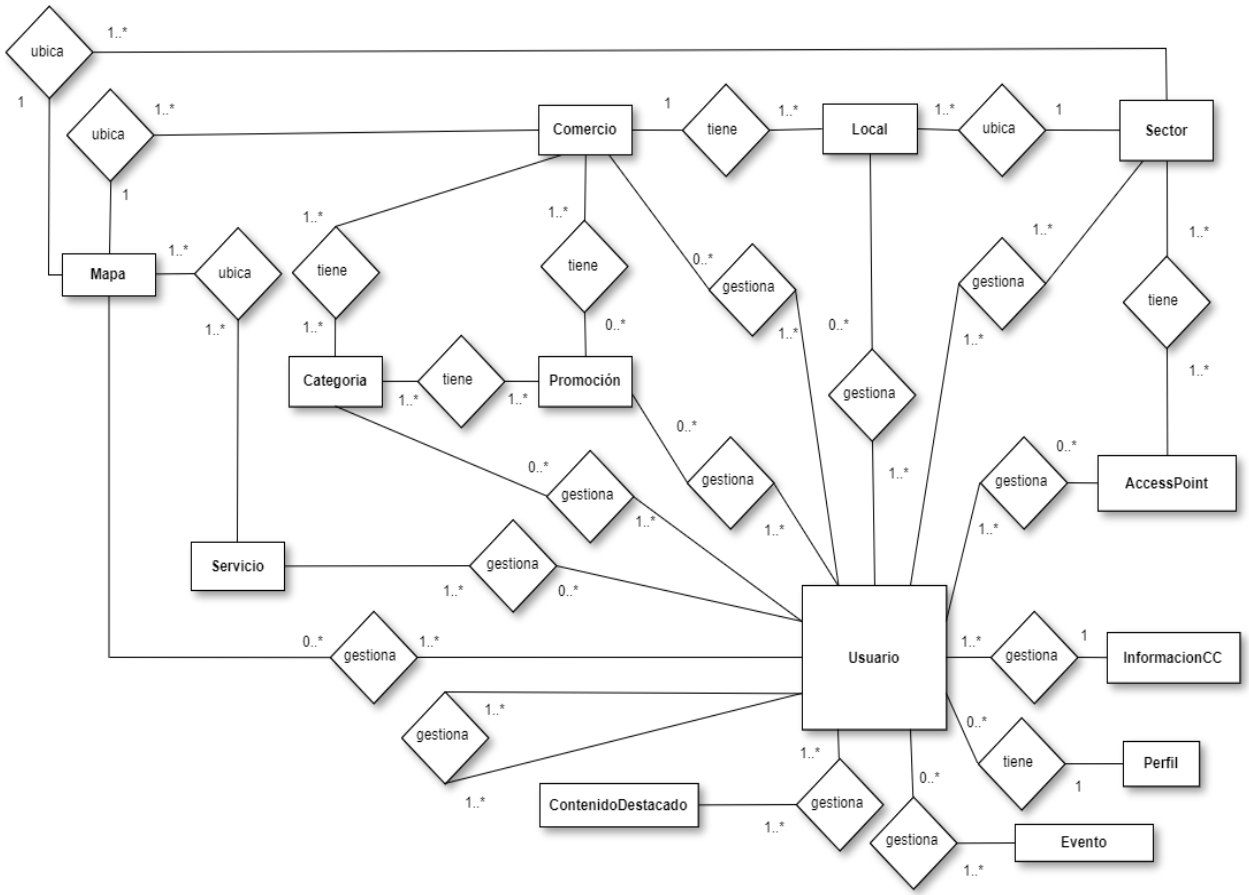

Figura 9.4: Diagrama Entidad-Relación de la Aplicación Web

En la Figura 9.5 se presenta el Modelo relacional de la aplicación, el cual es un modelo de datos en el que se muestra la estructura de la base de datos del sistema. Se reflejan las tablas que se definieron dentro de la base de datos, los campos que tienen, qué tipo de datos son y las relaciones entre esas diferentes tablas.

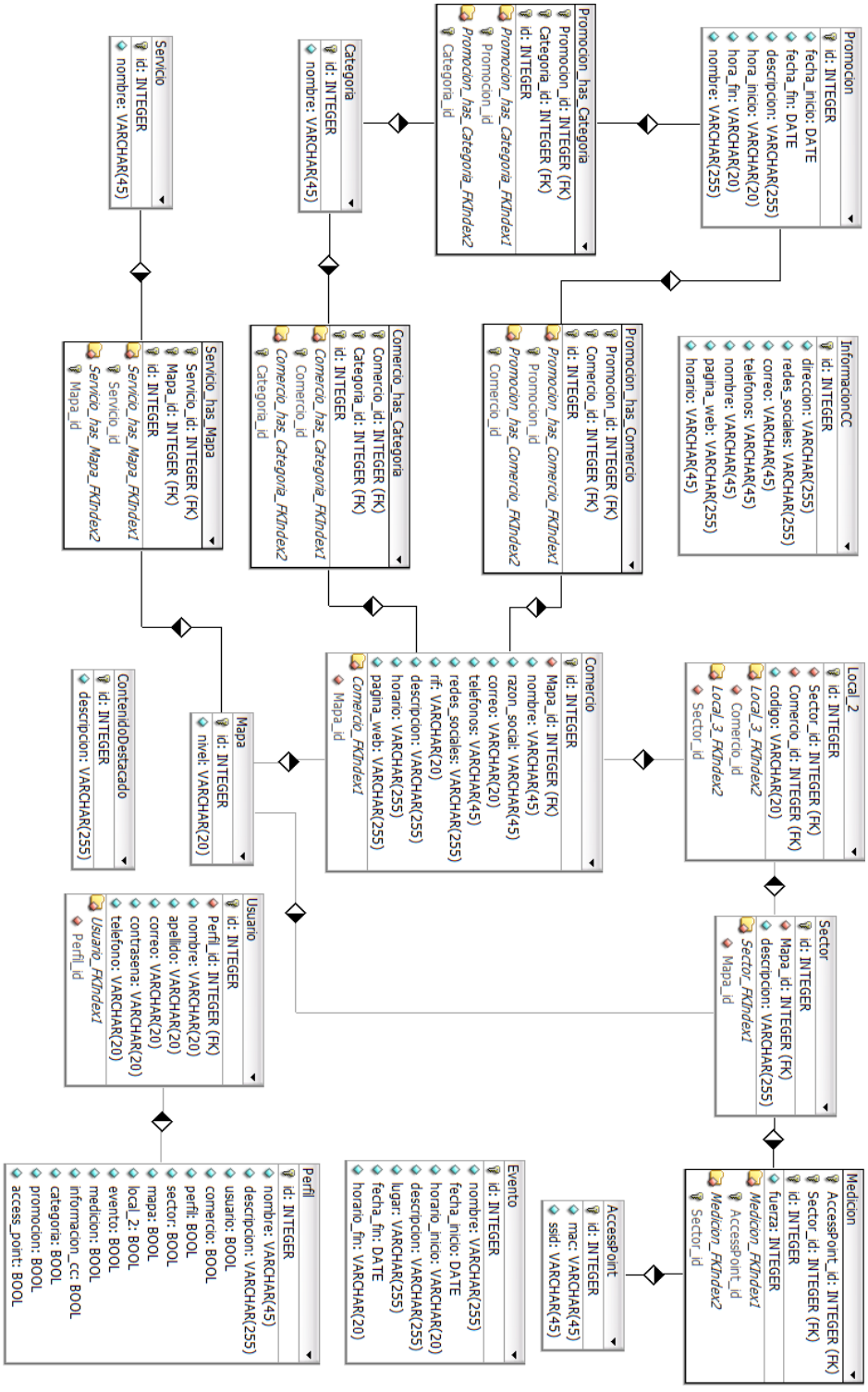

Figura 9.5: Modelo Relacional de la Aplicación Web

A continuación, se muestran algunos prototipos de interfaz de usuario para la página web de administración desarrollados con la herramienta Justinmind Prototyper.

En la Figura 9.6 se puede apreciar la pantalla de inicio de sesión de la aplicación web, la cual posee dos campos de texto para introducir el correo electrónico con el que fue registrado un usuario y la contraseña, además cuenta con un botón para iniciar la sesión y un enlace para la funcionalidad de recuperar contraseña.

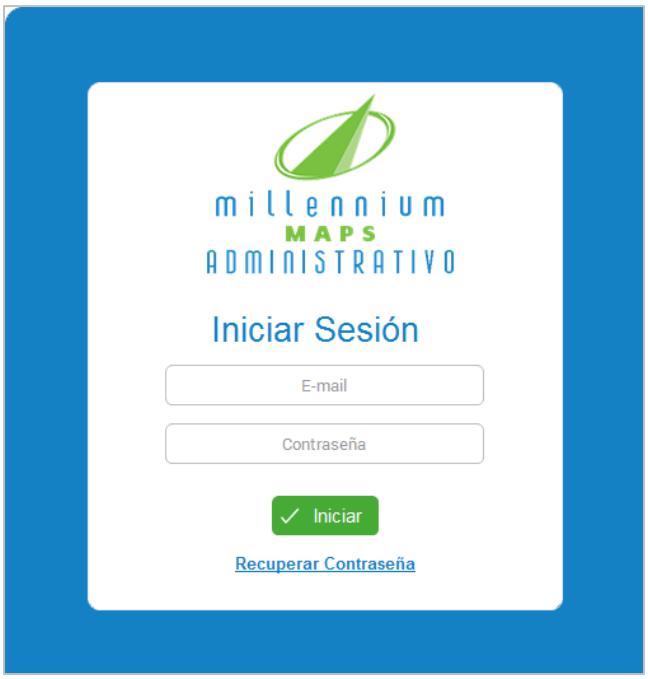

Figura 9.6: Diseño de Inicio de Sesión de la Aplicación Web

En la Figura 9.7 se visualiza el menú de la aplicación web tanto desplegado como en su forma compactada.

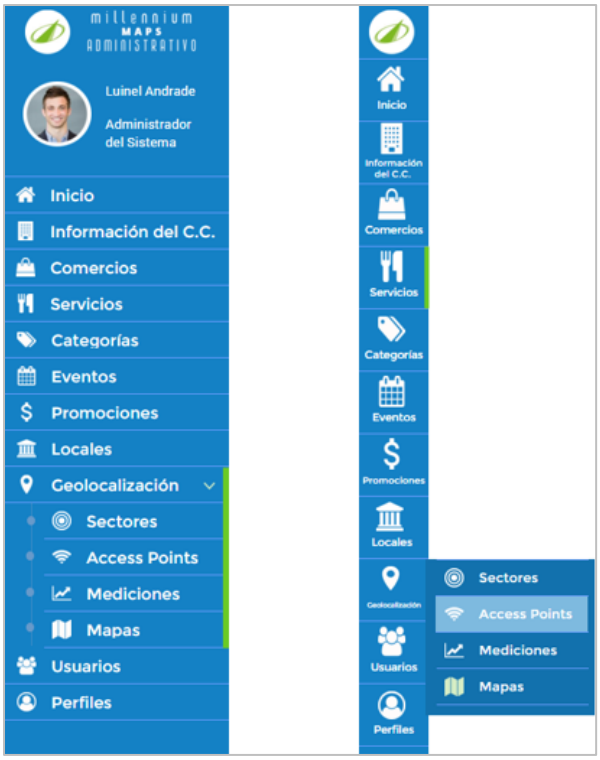

Figura 9.7: Diseño del Menú Lateral de la Aplicación Web

En la Figura 9.8 se aprecia el diseño de la pantalla de inicio de la aplicación web. Aquí se muestra la cantidad de registros que hay en base de datos por cada uno de los módulos de la aplicación.

Figura 9.8: Diseño de Pantalla de Inicio de la Aplicación Web

En la Figura 9.9 y en la Figura 9.10 se puede visualizar el listado de perfiles y de eventos respectivamente. Ambas ilustraciones sirven para tener noción de cómo es el listado de registros de un determinado módulo. Dichos listados varían un poco dependiendo de qué módulo sea y qué información se necesita mostrar, pero tienen elementos en común que son consistentes en todas las interfaces de listado, por ejemplo, que el nombre del módulo está en la parte superior izquierda, en la parte superior derecha se encuentra un buscador de registros, se tiene un botón de “Agregar” para crear un nuevo registro y en el listado de los elementos, del lado derecho de cada uno se tiene tres botones que son “Detalle”, “Editar” y “Eliminar”.

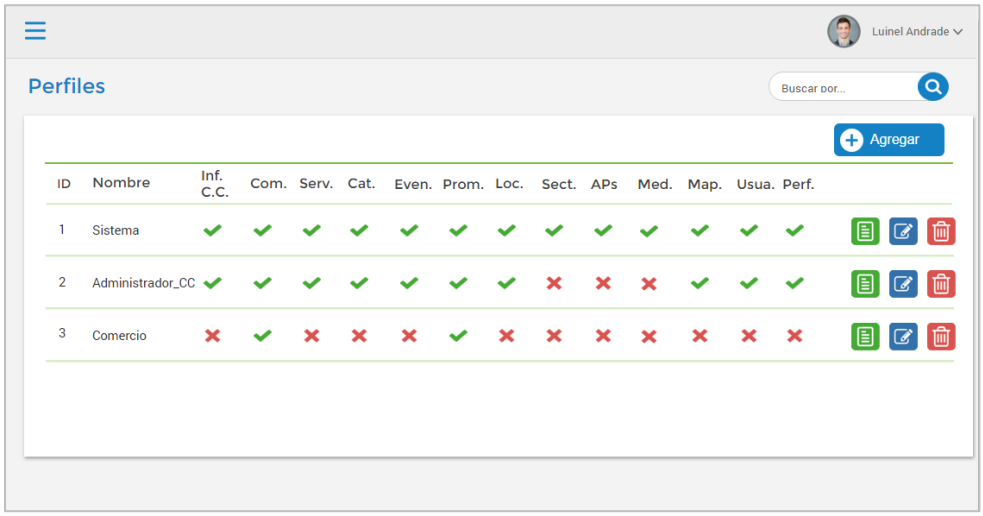

Figura 9.9: Diseño del Listado de Perfiles de l Aplicación Web

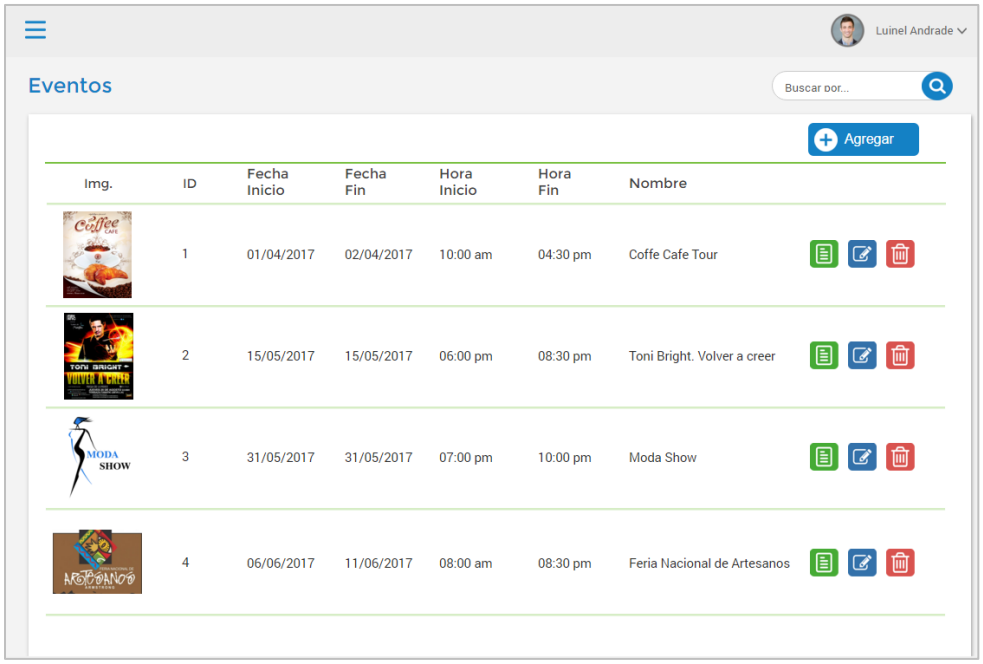

Figura 9.10: Diseño del Listado Eventos de la Aplicación Web

En la Figura 9.11 se puede apreciar un ejemplo de un formulario para agregar un registro en la aplicación web. En este caso se puede apreciar que es el formulario para agregar una promoción.

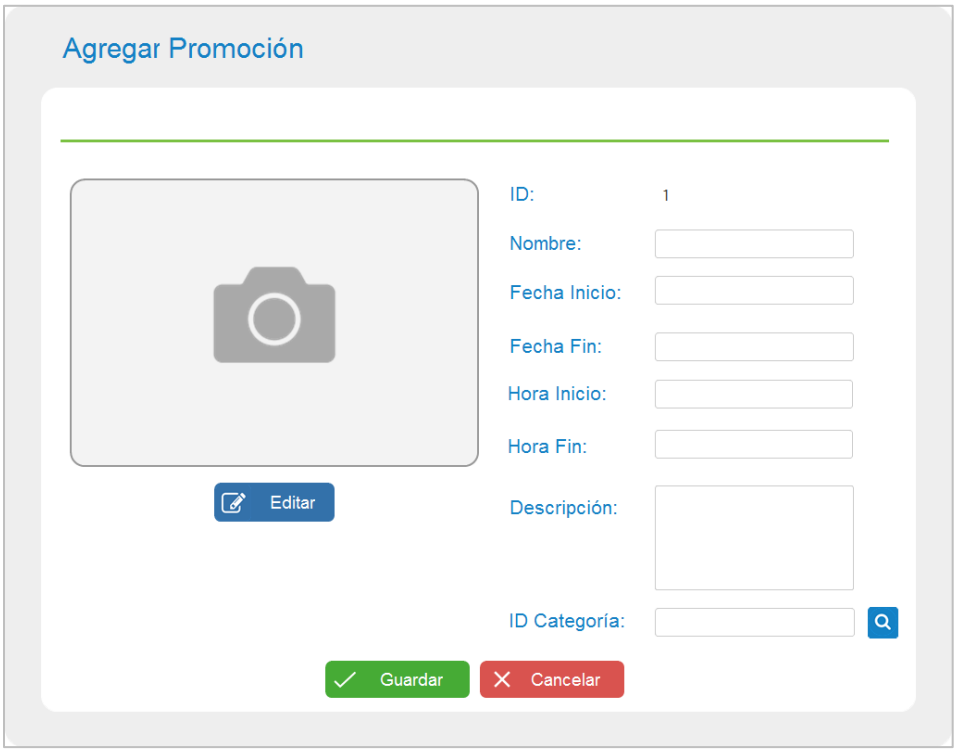

Figura 9.11: Diseño de Agregar Promoción de la Aplicación Web

En la Figura 9.12 se visualiza un ejemplo de un formulario para modificar un registro en la aplicación web. Tiene la misma estructura que el formulario de agregar del módulo al que pertenece, pero con la diferencia de que ya están los campos llenos con los datos que están almacenados del registro. En este caso se puede apreciar que es el formulario para modificar un comercio.

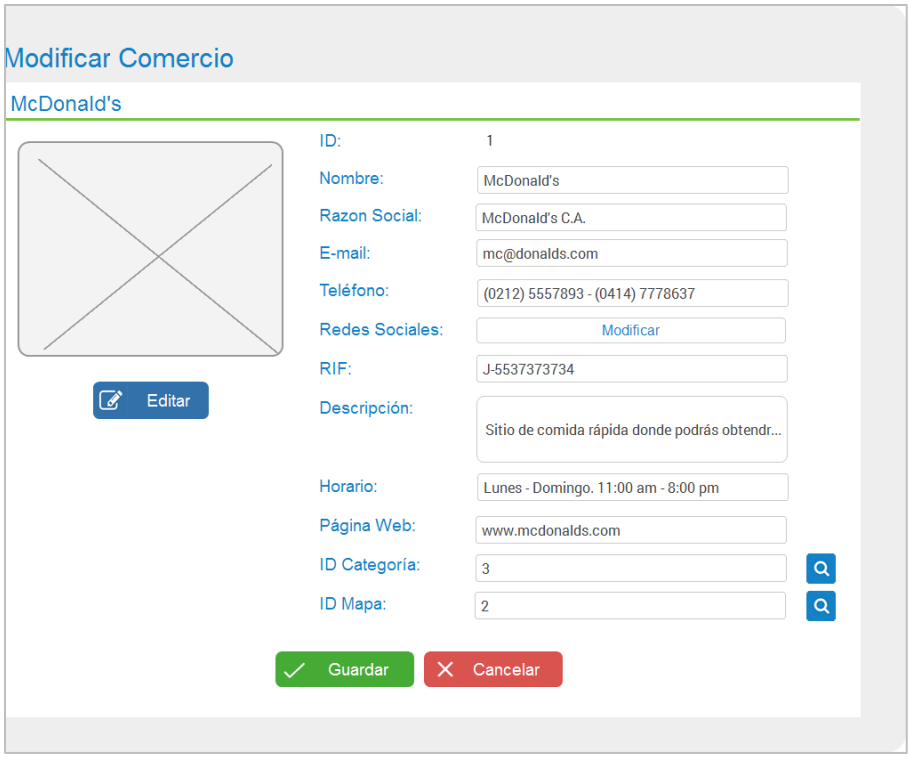

Figura 9.12: Diseño de Modificar Comercio de la Aplicación Web

En la Figura 9.13 se puede ver como es el detalle de un registro, tomando como ejemplo el detalle de un evento.

Figura 9.13: Diseño del Detalle de Evento de la Aplicación Web

En la Figura 9.14 se aprecia una ventana modal del sistema que se utiliza para seleccionar un registro de otro módulo que se relaciona con el registro que se está creando o modificando. Ejemplo, seleccionar que comercio se encuentra en un determinado local del centro comercial.

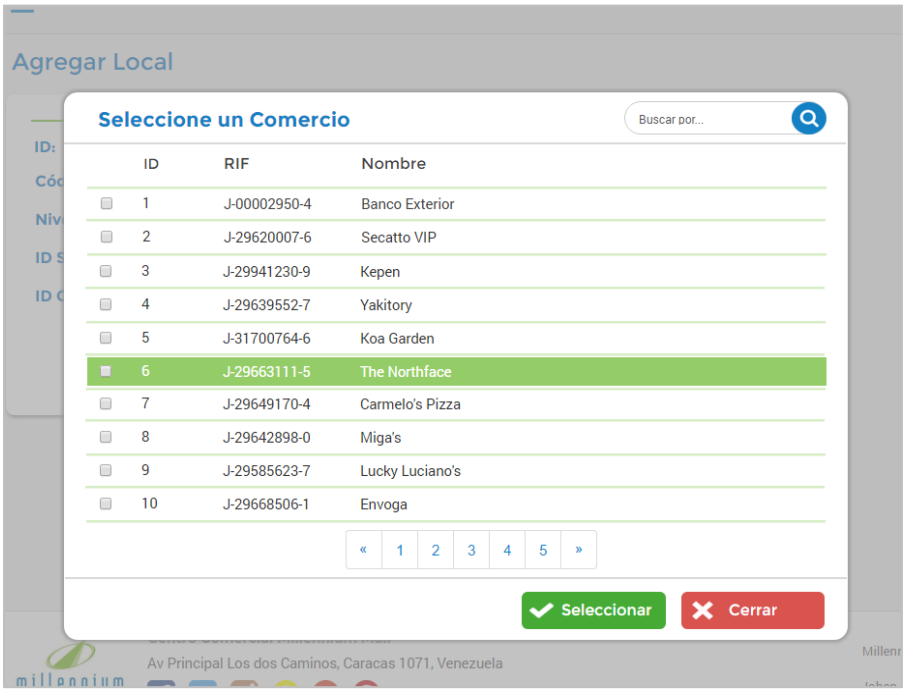

Figura 9.14: Diseño de Ventana Modal de Seleccionar Comercio de la Aplicación Web

En la Figura 9.15 se muestra el diseño de la vista que refleja la información del centro comercial y del contenido destacado.

Figura 9.15: Diseño de Información del C.C. de la Aplicación Web

9.3.2. Instalación y Configuración del Ambiente de Trabajo de la Aplicación Web

El sprint 1 comenzó con revisiones de los diseños de las interfaces de usuario de la aplicación web que se hicieron en el sprint anterior y haciendo arreglos generales sobre los mismos de ser necesario.

Una vez terminado los arreglos de los diseños, también en el sprint 1 se procedió con la instalación y configuración de las tecnologías y herramientas a utilizar para el desarrollo de la aplicación web administrativa, en donde se destacan: Sublime Text, Xampp que incluye PHP y MySQL, Git y Laravel.

Durante los primeros sprints de desarrollo, la aplicación web fue trabajada por cada autor en un servidor local en su computador personal y se utilizaba la herramienta Git para el control de versiones y trabajo colaborativo. Ambos computadores tienen el sistema operativo Windows 10 y fue haciendo uso de la herramienta Xampp que se instaló dicho servidor local, la cual tiene entre algunas ventajas el poder contar con MySQL y el lenguaje de programación PHP entre su plataforma por lo que se instalaban ambas herramientas de manera muy sencilla. En la Figura 9.16 se puede apreciar el panel de control de la aplicación Xampp corriendo en uno de los equipos donde se hizo el desarrollo de la solución con los servicios de Apache y MySQL iniciados.

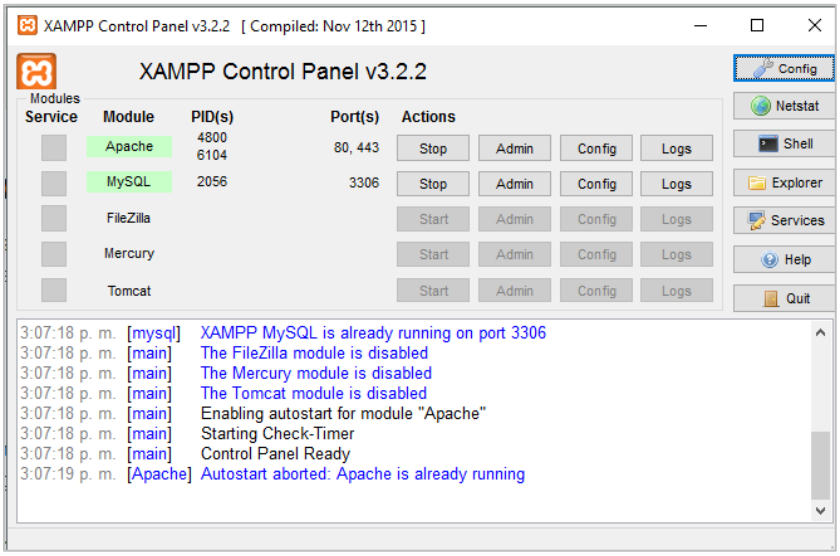

Figura 9.16: Panel de Control de la Aplicación Xampp

El framework Laravel fue instalado haciendo uso de un manejador de dependencias y paquetes PHP llamado Composer. Está herramienta fue descargada desde su página oficial[18] y una vez instalada permite crear una aplicación Laravel desde la línea de comandos del sistema una vez que se esté posicionado en la ruta donde se alojan los proyectos web. En la Figura 9.17 se aprecia la sintaxis del comando que se utiliza para crear un proyecto Laravel desde la línea de comandos del SO Windows. Una vez que se ejecuta el comando Composer comienza a descargar las librerías necesarias para el proyecto.

[18] https://getcomposer.org

```
C:\xampp\htdocs>composer create-project laravel/laravel nombre_proyecto --prefer-dist
```

Figura 9.17: Comando para Crear Proyecto Laravel con Composer

La herramienta Git se instaló descargando un instalador para SO Windows desde su página oficial [19] y se configuró el proyecto haciendo uso de la plataforma Gitlab[20], el cual es un servicio web de control de versiones y desarrollo de software colaborativo basado en Git. En la Figura 9.18 se muestra la pantalla de inicio del proyecto en la plataforma Gitlab.

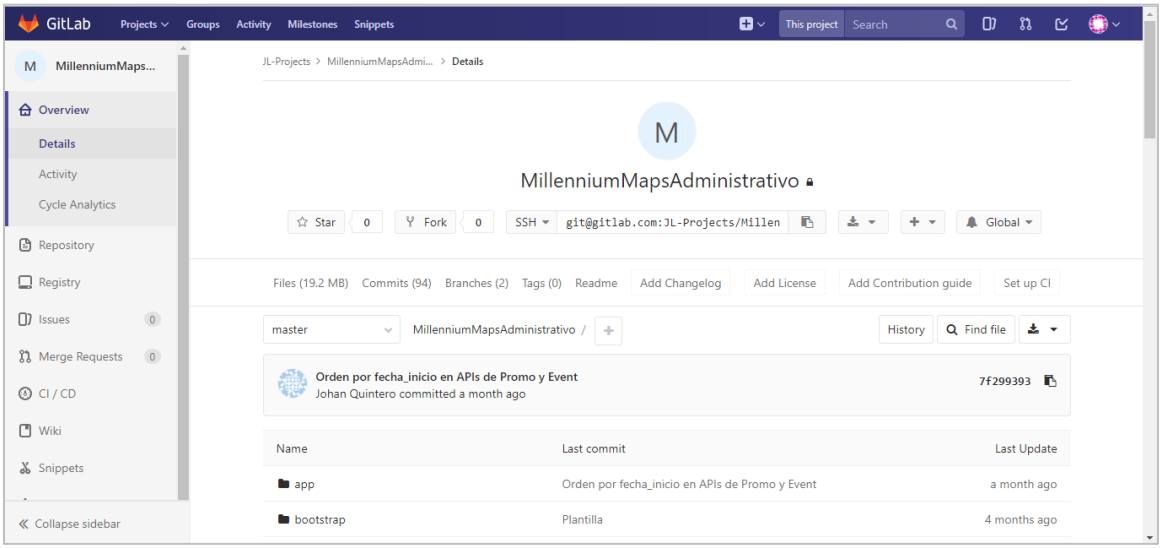

Figura 9.18: Proyecto de la Aplicación Web en la Plataforma Gitlab

Para establecer la comunicación entre la aplicación Laravel y la base de datos creada en MySQL se editó el archivo ".env" que se encuentra en la raíz del proyecto. Ese archivo es donde se guardan las variables de entorno del proyecto. En la Figura 9.19 se muestra cómo se editó el archivo mencionado en uno de los computadores de los autores para establecer la comunicación con la BD.

```
6  DB_CONNECTION=mysql
7  DB_HOST=127.0.0.1
8  DB_PORT=3306
9  DB_DATABASE=millennium_maps
10 DB_USERNAME=root
11 DB_PASSWORD=
```

Figura 9.19: Configuración del Archivo ".env" para la Conexión con la BD MySQL

[19] https://git-scm.com
[20] https://about.gitlab.com

9.3.3. Desarrollo de la Aplicación Web

El desarrollo de todos los módulos de la aplicación web salvo algunos submódulos del módulo de Sectores se llevó a cabo entre los sprints 1 y 4. Posterior a esos sprints se hicieron correcciones generales a algunas funcionalidades de la aplicación a lo largo de los sprints restantes según las pruebas que se realizaban y las necesidades que surgían durante el desarrollo de la aplicación móvil y su comunicación con la API.

Para entender cómo fue el proceso de desarrollo haciendo uso del framework Laravel se tomará como ejemplo para la explicación el módulo de "Comercios" de la aplicación web. El resto de los módulos tuvieron un desarrollo similar al que se describirá, por lo que con explicar uno, ya se entiende de forma general como se llevó a cabo la codificación de cada una de las funcionalidades.

Laravel incluye un ORM (Object-Relational Mapping) llamado Eloquent, el cual permite realizar de una forma fácil y sencilla los procesos correspondientes al manejo de bases de datos en el proyecto.

Una vez configurada la conexión con la BD MySQL, se puede crear los modelos desde la línea de comandos usando los comandos de Artisan, que es una herramienta de interfaces de líneas de comando de Laravel. En la Figura 9.20 se puede apreciar el comando empleado para crear el modelo desde la consola.

```
C:\xampp\htdocs\MillenniumMapsAdministrativo>php artisan make:model Comercio
```

Figura 9.20: Comando para Crear Modelo con Artisan

Por convención el nombre del modelo debe escribirse con la primera letra mayúscula y en singular. Con el comando anterior se crea un archivo dentro de la carpeta "/app" llamado "Comercio.php". Dentro de ese archivo se establece qué tabla de la BD va a usar ese modelo con la línea que se pueda apreciar en la Figura 9.21. En este caso es necesario establecer que tabla usará el modelo porque se está usando el idioma español para nombrar a los componentes del proyecto sino Laravel lo hace automáticamente al hacer uso de las convenciones en inglés.

```
10      protected $table = 'comercios';
```

Figura 9.21: Tabla de la BD que Usa el Modelo Comercio

Se procede a crear el archivo de migración. Las migraciones son archivos que se encuentran en la ruta "database/migrations" y que permitan crear la estructura de una tabla en BD. Este tipo archivo facilita entre otras cosas el poder tener un control de versiones sobre las tablas de la BD. En la Figura 9.22 se muestra el comando para crear un archivo de migración.

```
htdocs\MillenniumMapsAdministrativo>php artisan make:migration
crear_comercios_table --create=comercios
```

Figura 9.22: Comando para Crear un Archivo de Migración con Artisan

Se define los campos de la tabla en el archivo de migración creado con el comando anterior. En la Figura 9.23 se puede apreciar los campos definidos para la tabla comercios en el archivo de migración.

```
13      public function up()
14      {
15          Schema::create('comercios', function (Blueprint $table) {
16              $table->increments('id');
17              $table->string('nombre',45)->default('');
18              $table->string('razon_social',85)->default('');
19              $table->string('rif',20)->default('');
20              $table->string('descripcion',255)->default('');
21              $table->string('email',45)->default('')->unique();
22              $table->string('telefonos',45)->default('');
23              $table->string('horario',45)->default('');
24              $table->string('pagina_web',255)->default('');
25              $table->string('redes_sociales',500)->default('');
26              $table->integer('mapa_id')->unsigned()->nullable();
27              $table->timestamps();
28
29              $table->foreign('mapa_id')->references('id')->on('mapas');
30          });
31      }
```

Figura 9.23: Campos de la Tabla Comercios en el Archivo de Migración

Laravel permite asignar valores a cada uno de los atributos de una tabla de una manera muy sencilla con un *array* asociativo de elementos que se definen en el modelo y que pueden ser por ejemplo los campos de un formulario, para que de esa forma sea más fácil el proceso de almacenaje desde un *form*. Los campos pueden tratarse como protegidos o como campos que pueden ser llenados con asignación masiva. En la Figura 9.24 se puede apreciar el modelo "Comercio" y en la línea 10 de dicha figura se visualiza la declaración del arreglo asociativo donde se especifica los campos de la tabla que pueden ser llenados de la forma descrita, que es el caso cuando se envía un formulario creando un *array* asociativo para ser guardado.

```
7   class Comercio extends Model
8   {
9       protected $table = 'comercios';
10      protected $fillable = ['nombre','razon_social','rif','descripcion','email','telefonos',
11      'horario','pagina_web','redes_sociales','mapa_id'];
12
13      public function categorias(){
14          return $this->belongsToMany('\App\Categoria','comercios_categorias')
15                      ->withTimestamps();
16      }
17
18      public function promociones(){
19          return $this->belongsToMany('\App\Promocion','comercios_promociones')
20                      ->withTimestamps();
21      }
22  }
```

Figura 9.24: Porción de Código del Archivo "Comercio.php"

Además en la Figura 9.24 también se puede ver la definición de dos funciones que son "categorías()" y "promociones()". De esa forma se define en el modelo que existe una relación muchos a muchos entre la tabla "comercios" y "categorias" y en entre las tablas "comercios" y "promociones". Dichas relaciones muchos a muchos se puede apreciar en el modelo relacional de la Figura 9.5.

Ahora se debe crear el controlador de la entidad "Comercios" para poder manejar los eventos generados por el usuario e invocar las peticiones al modelo cuando se realiza alguna solicitud sobre la información. En la Figura 9.25 se aprecia el comando para crear un controlador usando los comandos de Artisan.

```
C:\xampp\htdocs\MillenniumMapsAdministrativo>php artisan make:controller
ComercioController
```

Figura 9.25: Comando para Crear un Controlador con Artisan

La ejecución de dicho comando genera un archivo llamado "ComercioController.php" ubicado en la ruta "/app/Http/Controllers". En la cabecera de ese archivo se debe importar el modelo Comercio, como se puede apreciar en la Figura 9.26, además también se puede apreciar la importación de otros modelos que se usan en ese controlador.

```
14  use App\Perfil;
15  use App\Comercio;
16  use App\Mapa;
17  use App\Promocion;
18  use App\Categoria;
```

Figura 9.26: Importación de Modelos en el Controlador

Una vez que se tiene el controlador, se debe asegurar que la aplicación tiene configuradas las rutas necesarias para usarlo. Las rutas de la aplicación se definen en el archivo "routes.php" ubicado en la ruta "/app/Http". Al momento de acceder a una ruta desde el navegador se realiza una petición HTTP que se envía al archivo mencionado, si la ruta no existe da un error y si la ruta existe, redirige al controlador que contiene la lógica de la aplicación. También es posible que en vez de definir totalmente la lógica de las peticiones en el archivo "routes.php", se organice ese comportamiento usando clases tipo Controller. Los Controladores trabajan con las peticiones GET, POST, PUT, DELETE y PATCH, la cuales se pueden agrupar y manipularse lógicamente en una clase. Los métodos de los controladores se asocian a las peticiones HTTP de la siguiente forma:

- **GET:** index, create, show, edit.
- **POST:** store.
- **PUT:** update.
- **DELETE:** destroy.
- **PATCH:** update.

En el archivo "routes.php" con usar un tipo de ruta llamada "resource" ya Laravel crea internamente las rutas de la aplicación relacionadas a un recurso específico y las asigna a los métodos del controlador. En la línea 43 del código que se muestra en la Figura 9.27 se puede apreciar la definición de las rutas "shop" para el recurso "Comercio". También se puede apreciar

la definición de otras rutas adicionales definidas por los autores. Además, en la Tabla 9.17 se muestra la lista de las rutas generadas con las líneas de código que en dicha figura se reflejan.

```
36  Route::get('/shop/search/map',['as' => 'shop.search.map', 'uses'=>'ComercioController@SearchMapa']);
37
38  Route::get('/shop/search/category',['as' => 'shop.search.category', 'uses'=>
39      'ComercioController@SearchCategoria']);
40
41  Route::get('/shop/search', ['as' => 'shop.search', 'uses'=>'ComercioController@search']);
42
43  Route::resource('shop', 'ComercioController');
```

Figura 9.27: Definición de Rutas para el Recurso "Comercio"

Método	URI	Controlador y Método Asociado
GET | HEAD	shop	App\Http\Controllers\ComercioController@index
POST	shop	App\Http\Controllers\ComercioController@store
GET | HEAD	shop/create	App\Http\Controllers\ComercioController@create
GET | HEAD	shop/search	App\Http\Controllers\ComercioController@search
GET | HEAD	shop/search/category	App\Http\Controllers\ComercioController@searchCategoria
GET | HEAD	shop/search/map	App\Http\Controllers\ComercioController@searchMapa
GET | HEAD	shop/{id}	App\Http\Controllers\ComercioController@show
PUT | PATCH	shop/{id}	App\Http\Controllers\ComercioController@update
DELETE	shop/{id}	App\Http\Controllers\ComercioController@destroy
GET | HEAD	shop/{id}/edit	App\Http\Controllers\ComercioController@edit

Tabla 9.17: Lista de Rutas para el Recurso Comercio

Ya en los diferentes métodos del controlador es que se codifica la lógica de la aplicación. Algunas de las diferentes acciones que se realizan en estos métodos son: procesar las peticiones generadas por los usuarios, interactuar con el modelo para la manipulación de los datos, hacer validaciones de los datos de los formularios, validar los permisos de los usuarios sobre los distintos módulos, enviar datos recibidos del modelo a las vistas, etc.

En el caso de las vistas, podría decirse que son la parte pública del sistema que el usuario va a poder ver, se implementan haciendo uso de HTML para definir la estructura, CSS para asignar los estilos y JS para aplicarle el dinamismo. Además, en el caso de Laravel también se hace uso de un motor de plantillas llamado Blade. Las vistas se encuentran ubicadas en la ruta "/resources/views/".

En el framework también se puede utilizar unos archivos que se llaman plantillas o *templates* que suelen ser los archivos de las vistas principales, que tienen los segmentos de código que se repiten en más de una vista, como por ejemplo el header, el menú principal y el footer, y que como están prácticamente presentes en todas las interfaces, no es cómodo para el desarrollador estar repitiéndolos en todas las vistas. Además de los *templates*, se encuentra también un tipo de archivos llamados *partials*, que son pequeños segmentos de código que suelen ser usados generalmente en partes del sistema específicas, como por ejemplo los formularios o secciones de mensajes.

En la Figura 9.28 se muestra una porción de código del archivo "templade.blade.php" ubicado en la ruta "/resources/views/template" el cual es el archivo que contiene la plantilla comprendida por el menú principal de la aplicación, el header, el footer y el menú de usuario desplegable de la aplicación web.

```
<body class="nav-md">

    <div class="container body">
        <div class="main_container">
            <div class="col-md-3 left_col">
                <div class="left_col scroll-view">
                    <div class="navbar nav_title" style="border: 0;">
                        <a href="{{ url('/home') }}" class="site_title">
                            <img class="logoSimple" src="{{ asset('images/logoSimple.png') }}"
                            style="margin-left:10px;">
                            <span>
                                <img class="letrasLogoSimple" src="{{ asset('images/LetrasLogoSimple.png') }}">
                            </span>
                        </a>
                    </div>
                    <div class="clearfix"></div>
                    <!-- menu prile quick info - MENU BIENVENIDA -->
                    <div class="profile">
                        <div class="profile_pic">
                            @if (file_exists(public_path('images/usuarios/'.Auth::user()->id.'.png')))
                                <img src="{{ asset('images/usuarios/'.Auth::user()->id.'.png') }}" alt="..."
                                class="img-circle profile_img">
                            @else
                                <img src="{{ asset('images/usuarios/generico.png') }}" alt="..."
                                class="img-circle profile_img">
                            @endif
                        </div>
                        <div class="profile_info">
```

Figura 9.28: Porción de Código del Archivo “template.blade.php”

A modo de ejemplo, a continuación, se muestra el código de algunos de los métodos definidos en el “ComercioController.php” y la vista que dicho controlador retorna al usuario.

En la Figura 9.29 se puede apreciar el código del método “index()” del “ComercioController.php”, que es el que se encarga de buscar en el modelo los datos de los comercios almacenados en BD y enviarlos a la vista correspondiente (Figura 9.30) para mostrarle al usuario el listado de los comercios.

```
    public function index(){

        $perfil_id = Auth::user()->perfil_id;
        $perfil_permisos = Perfil::find($perfil_id);

        if ($perfil_permisos->comercio == 1){

            $comercios = Comercio::paginate(15);
            $mapas = Mapa::all();

            return view('comercios.comercios',['perfil' => $perfil_permisos,
                'comercios' => $comercios, 'mapas' => $mapas]);

        }else return redirect('/home');
    }
```

Figura 9.29: Método “index()” del “ComercioController.php”

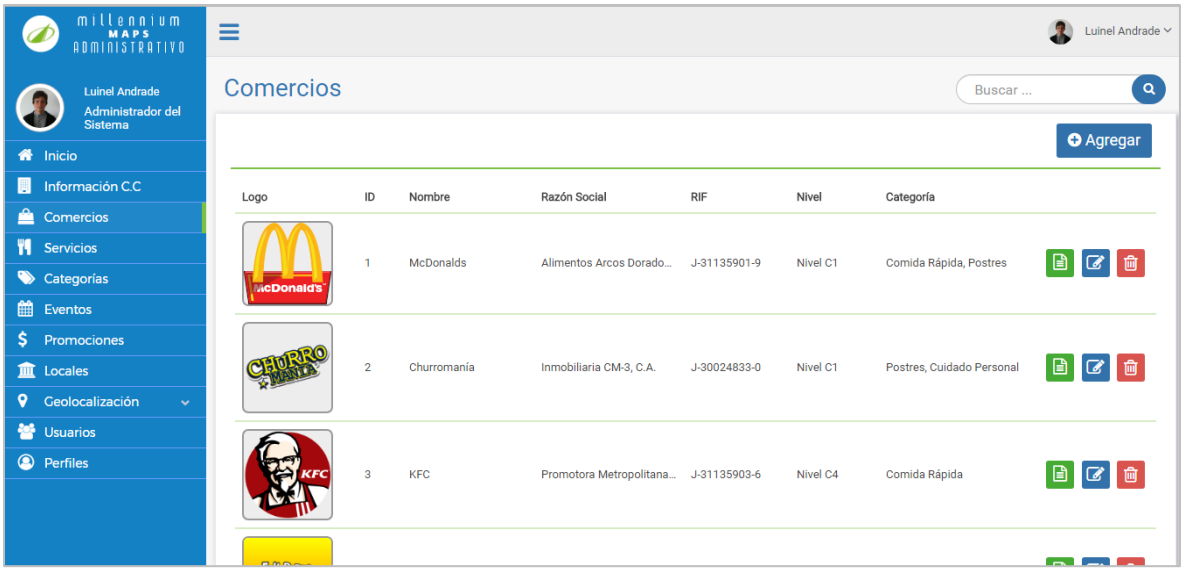

Figura 9.30: Vista del Listado de Comercios

En la Figura 9.31 se puede apreciar el código del método "show()" del "ComercioController.php", que es el que se encarga de mostrar buscar en el modelo los datos de un comercio en particular y enviarlos a la vista correspondiente para mostrarle al usuario el detalle del comercio (Figura 9.32).

```
201     public function show($id){
202         $perfil_id = Auth::user()->perfil_id;
203         $perfil_permisos = Perfil::find($perfil_id);
204 
205         $comercio = Comercio::find($id);
206         $mapa = Mapa::find($comercio->mapa_id);
207 
208         if ($perfil_permisos->comercio == 1) {
209             return view('comercios.detalle',['perfil' => $perfil_permisos,
210              'comercio' => $comercio, 'mapa' => $mapa]);
211         }else{
212             return redirect('/home');
213         }
214     }
```

Figura 9.31: Método "show()" del "ComercioController.php"

Figura 9.32: Vista de Detalle de Comercio

9.3.4. Desarrollo de la API

El proceso de desarrollo de la API que utiliza la aplicación móvil para consultar los datos y contenidos fue llevado cabo en el sprint 3 salvo la de la funcionalidad de geolocalización que se llevó a cabo en el sprint 6 cuando ya se inició el desarrollo del módulo de Mapas de la aplicación móvil. Ya teniendo desarrollado el resto de los módulos de la aplicación web resultó sencillo realizar la API ya que sólo hizo falta desarrollar unos nuevos controladores que validarán a través de un *middleware*[21] una clave denominada API Key necesaria para procesar la petición y retornar en los métodos las respuestas en formato JSON. También se añadió al archivo "routes.php" las rutas de la API.

En la API de este proyecto sólo fue necesario implementar métodos de consulta sobre los recursos de la base de datos y que se invocarán a través de peticiones GET del protocolo HTTP.

En la Figura 9.33 se muestra una porción de código del archivo "PromoAPIController.php" que se encuentra en la ruta "/app/Http/Controllers/API/" el cual es el controlador de los datos de las promociones que retorna la API.

[21] En Laravel, los *middlewares* son funciones que permiten agregar filtros a cada petición HTTP por un usuario de la aplicación, buscando disminuir la carga de los controladores y aplicar de forma más simple las restricciones de seguridad de la aplicación.

```php
    public function index(){
        return DB::table('comercios_promociones')
                        ->join('promociones', 'comercios_promociones.promocion_id', '=', 'promociones.id')
                        ->join('comercios', 'comercios_promociones.comercio_id', '=', 'comercios.id')
                        ->select('promociones.*', \DB::raw('GROUP_CONCAT(comercios.nombre) as comercio'))
                        ->orderBy('promociones.fecha_inicio','asc')
                        ->groupBy('promociones.id')
                        ->paginate(10);
    }

    public function show($id){
        return Promocion::find($id);
    }

    public function showCategory($id){
        $promocion = Promocion::find($id);
        return \Response::json($promocion->categorias);
    }

    public function showShop($id){
        $promocion = Promocion::find($id);
        return \Response::json($promocion->comercios);
    }
```

Figura 9.33: Porción de Código del Archivo "PromoAPIController.php"

En la Figura 9.34 se ve un ejemplo de la respuesta que retorna la API en formato JSON para la consulta de una promoción específica.


```json
{
  "id": 1,
  "nombre": "Tu Promo",
  "descripcion": "Llevate una promocion del 15% en lo que tu quieras",
  "fecha_inicio": "2017-08-01",
  "fecha_fin": "2017-10-15",
  "hora_inicio": "9:00 am",
  "hora_fin": "7:00 pm",
  "created_at": "2017-07-03 21:50:00",
  "updated_at": "2017-08-20 09:36:46"
}
```


Figura 9.34: Respuesta en formato JSON de la API

En la Tabla 9.18 se muestra la descripción completa de la API.

Método	URI	Descripción
GET	api/ap	Retorna la lista de access points
GET	api/ap/{id}	Retorna la información de un access point en específico
GET	api/category	Retorna la lista de categorías
GET	api/category/{id}	Retorna la información de una categoría en específico
GET	api/ category/{id}/promo	Retorna la lista de las promociones que tiene una categoría en específico
GET	api/ category/{id}/shop	Retorna la lista de los comercios que tiene una categoría en específico
GET	api/content	Retorna la lista de contenidos destacados
GET	api/content/{id}	Retorna la información de un contenido destacado en específico
GET	api/event	Retorna la lista de eventos
GET	api/event/{id}	Retorna la información de un evento en específico
GET	api/info	Retorna la información del centro comercial
GET	api/local	Retorna la lista de locales
GET	api/local/{id}	Retorna la información de un local en específico
GET	api/map	Retorna la lista de mapas del centro comercial
GET	api/map/{id}	Retorna la información de un mapa en específico
GET	api/ map/{id}/service	Retorna la lista de los servicios que tiene un mapa en específico
GET	api/med	Retorna la lista de las mediciones hechas de las potencias de los access points en cada sector del C.C.
GET	api/med/locate	Retorna el sector del centro comercial en el que está ubicado el usuario.
GET	api/med/{id}	Retorna la información de una medición específica realizada en un sector
GET	api/promo	Retorna la lista de promociones
GET	api/promo/{id}	Retorna la información de una promoción en específico
GET	api/ promo/{id}/category	Retorna la lista de las categorías que tiene una promoción en específico
GET	api/promo/{id}/shop	Retorna la lista de los comercios que tiene una promoción en específico
GET	api/sector	Retorna la lista de sectores del centro comercial
GET	api/sector/{id}	Retorna la información de un sector en específico
GET	api/service	Retorna la lista de servicios del centro comercial
GET	api/service/{id}	Retorna la información de un servicio en específico
GET	api/ service/{id}/map	Retorna la lista de los mapas que tiene un servicio en específico
GET	api/shop	Retorna la lista de los comercios
GET	api/shop/{id}	Retorna la información de un comercio en específico
GET	api/shop/{id}/category	Retorna la lista de las categorías que tiene un comercio en específico
GET	api/shop/{id}/promo	Retorna la lista de las promociones que tiene un comercio en específico

Tabla 9.18: Descripción de la API

Entre todas las rutas del API destaca la que implementa y procesa el método de geolocalización en los espacios interiores del centro comercial. Dicha funcionalidad retorna a la aplicación móvil el sector del centro comercial en el que se ubica el usuario y se encuentra disponible en la ruta

“api/med/locate”. Las peticiones a esa ruta son procesadas por la función “LocateSector()” del controlador “MedAPIController.php”. En dicha función se implementa el método de los k-vecinos (subsección 6.8.4) para procesar la ubicación del usuario según los access points que percibe el dispositivo móvil y sus respectivas fuerzas de las señales.

Para procesar el método de geolocalización, la aplicación móvil envía a la API en un hilo de ejecución distinto del hilo principal, los datos de todos los access points y su respectiva fuerza de señal que fueron captados en un instante de tiempo. El API recibe la petición, analiza y procesa la data recibida para obtener y guardar en arreglos los datos de las MAC (Media Access Control) y potencia de señales de los APs, filtra los APs que no fueron tomados en cuenta en el proceso de fingerprinting, procesa el algoritmo y retorna la ubicación donde, en teoría, se encuentra el usuario según los resultados.

La respuesta de la API de geolocalización tiene el siguiente formato:

```
{
  "sector": "<ID_Mapa>_s_<ID_Sector>"
}
```

En donde:

- **ID_Mapa:** Es el atributo ID del mapa del nivel del centro comercial en el que se encuentra el usuario.
- **ID_Sector:** Es el atributo ID del sector en el que se encuentra el usuario.

Por ejemplo, si se dice que un usuario está ubicado en el Piso 2 dentro del sector 15 del C.C., asumiendo que el ID de dicho piso también sea 2, es porque el API retornó la siguiente respuesta:

```
{
  "sector": "2_s_15"
}
```

En caso de que el dispositivo móvil no capte ningún access point registrado en la base de datos en el proceso de fingerprinting, se considera que el usuario no se encuentra dentro de las instalaciones del C.C. y por ende no se puede mostrar su ubicación. En ese caso, la API retorna la siguiente respuesta:

```
{
  "sector": "0"
}
```

9.4. Análisis General de la Aplicación Móvil

La aplicación móvil fue desarrollada usando Android Studio como IDE y Java como lenguaje de programación, debido a que, como ya se mencionó anteriormente, Java es el lenguaje oficial para desarrollar aplicaciones para Android.

Para el desarrollo de esta aplicación se usó la herramienta Git para el control de versiones y el trabajo colaborativo sobre el código de la aplicación y la plataforma GitLab, la cual está basada en Git y además permite tener repositorios privados de manera gratuita.

En términos generales, la aplicación posee un conjunto de funcionalidades las cuales le permiten al usuario ver el listado de los comercios del centro comercial, las promociones de dichos comercios, los eventos y servicios que posee el centro comercial, las categorías, así como también los mapas del mismo. Además, posee una función para poder anotar el puesto de estacionamiento en donde se encuentre estacionado el vehículo del usuario y un sistema de geolocalización para que el usuario se pueda ubicar dentro del centro comercial.

A continuación, se describe el proceso de desarrollo de la aplicación.

9.4.1. Artefactos y Prototipo General de Interfaz de Usuario de la Aplicación Móvil

Antes de iniciar con el desarrollo de la aplicación móvil, se realizó un análisis de las funciones que tendría la misma y se generaron un conjunto de artefactos, al igual que en el desarrollo de la aplicación web, como un listado de requerimientos funcionales y no funcionales y un diagrama de casos de uso, los cuales fueron realizados en el transcurso del Sprint 3. Además, en el Sprint 4, se realizaron los diseños de la aplicación móvil, los cuales posteriormente serían revisados por ambos autores y corregidos tomando en cuenta como factor principal la usabilidad de la aplicación.

En cuanto al listado de los requerimientos funcionales y no funcionales, se tienen los siguientes:

- **Requerimientos Funcionales:**
  - Los usuarios podrán ver la lista de comercios y el detalle de cada uno del centro comercial.
  - Los usuarios podrán ver la lista de promociones y el detalle de cada uno de los comercios centro comercial.
  - Los usuarios podrán ver la lista de eventos y el detalle de cada uno del centro comercial.
  - Los usuarios podrán ver la lista de contenidos destacados y el detalle de cada uno del centro comercial.
  - Los usuarios podrán ver la lista de servicios y el detalle de cada uno del centro comercial.
  - Los usuarios podrán ver la lista de categorías y el detalle de cada una que maneja la aplicación para clasificar los comercios, servicios y promociones del centro comercial.
  - Los usuarios podrán ver la lista de comercios, promociones y/o servicios agrupados por categorías.
  - Los usuarios podrán ver la información del centro comercial.
  - Los usuarios podrán ver los mapas del centro comercial y navegar en ellos.
  - Los usuarios podrán ver su ubicación actual en el centro comercial.
  - Los usuarios podrán recibir indicaciones de cómo llegar al destino de su interés.
  - Los usuarios podrán almacenar y ver posteriormente el puesto de estacionamiento donde estacionaron su vehículo.
  - Los usuarios podrán enviar un e-mail por medio de la aplicación de correo que tengan instalada en el dispositivo al correo de contacto tanto del centro comercial como de los comercios.

- Los usuarios podrán llamar al número de contacto del centro comercial o de los comercios por medio de la aplicación de llamadas del dispositivo.
- Los usuarios podrán ver las redes sociales del centro comercial de los comercios por medio del navegador o las aplicaciones de redes sociales respectivas que tenga instalada en el dispositivo.

- **Requerimientos No Funcionales:**
  - Se desarrollará una aplicación fácil de cargar, que no sea tan pesada.
  - La aplicación contará con variaciones de tonalidades de los colores: verde, azul y blanco.
  - La navegación de la aplicación será sencilla y fácil de entender y manejar.
  - La aplicación no contará con ningún tipo de información que no esté relacionada con el centro comercial.

En el diagrama de casos de uso presentado en la Figura 9.35, se tomó en cuenta como actor a un "Usuario General", el cual puede usar la aplicación móvil para probar todas las funcionalidades de la misma sin ninguna limitación de permisos o tener que autenticarse en el sistema, solo necesitando conexión a internet en su dispositivo móvil. Además, se muestra en el diagrama cada uno de los usos que el usuario le puede dar a la aplicación móvil, es decir, las funcionalidades que el mismo puede probar.

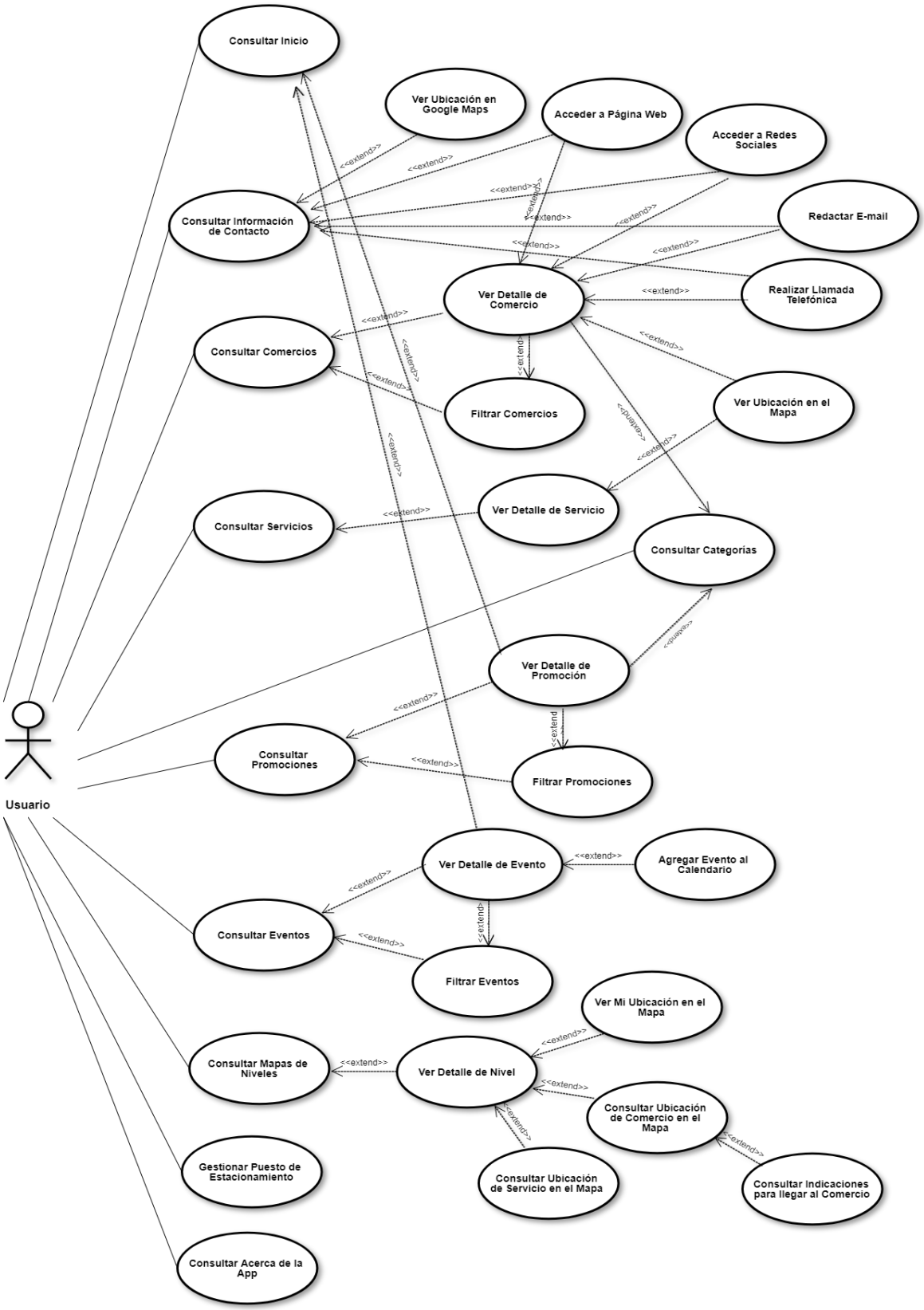

Figura 9.35: Diagrama de Casos de Uso de la Aplicación Móvil

A continuación, se muestran algunos prototipos de interfaz de usuario para la aplicación móvil desarrollados con la herramienta Justinmind Prototyper.

En la Figura 9.36 se puede apreciar el Splash Screen de la aplicación, la cual muestra el ícono de la misma con una imagen de fondo del centro comercial que se encuentra en constante movimiento hasta que la aplicación ejecuta el siguiente activity.

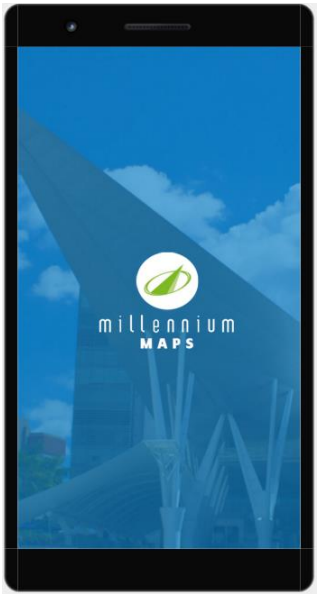

Figura 9.36: Diseño de Splash Screen de la Aplicación Móvil

En la Figura 9.37 se puede apreciar la pantalla de Inicio de la aplicación, la cual muestra un carrusel con los Contenidos Destacados, una lista con los cinco (5) eventos más pronto en realizarse y una lista con las cinco (5) promociones más recientes.

Figura 9.37: Diseño de la Pantalla de Inicio de la Aplicación Móvil

En la Figura 9.38 se puede apreciar el Menú de la aplicación con cada una de sus opciones las cuales son: Inicio, Comercios, Categorías, Promociones, Eventos, Servicios, Mapas, Tu Puesto, Contacto y Acerca de la Aplicación.

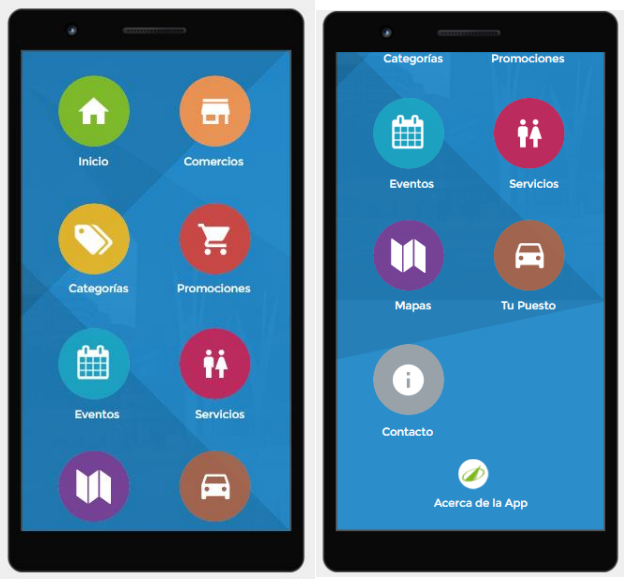

Figura 9.38: Diseño del Menú de la Aplicación Móvil

En la Figura 9.39 se puede apreciar la ventana emergente de Tu Puesto, en donde se puede gestionar el puesto de estacionamiento donde se encuentra estacionado el vehículo del usuario. Esta ventana posee dos opciones, el ícono de la papelera que permite borrar el texto y el ícono del lápiz que permite editar el texto.

Figura 9.39: Diseño de la Ventana Emergente de Tu Puesto de la Aplicación Móvil

En la Figura 9.40 se puede apreciar la pantalla de Contacto en donde se muestra toda la información del centro comercial como: una breve descripción, dirección, correo electrónico, página web, horario, números telefónicos y redes sociales. Además, posee una opción para poder ubicar el centro comercial en Google Maps.

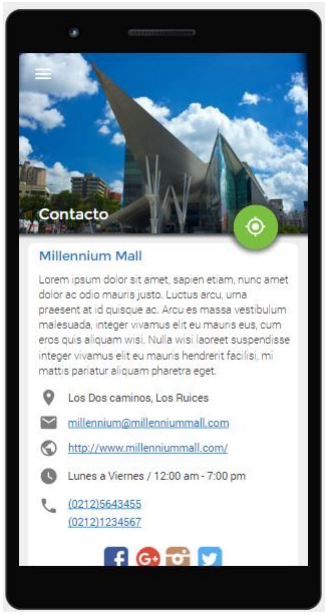

Figura 9.40: Diseño de la Pantalla de Contacto de la Aplicación Móvil

En la Figura 9.41 se puede apreciar la pantalla de Acerca de la Aplicación en donde se muestra la información de la aplicación y los datos del proyecto de los autores, como: una breve descripción de la aplicación, la foto de los autores con sus nombres y correos electrónicos y los datos del Trabajo Especial de Grado, como título, tutores, institución, entre otros.

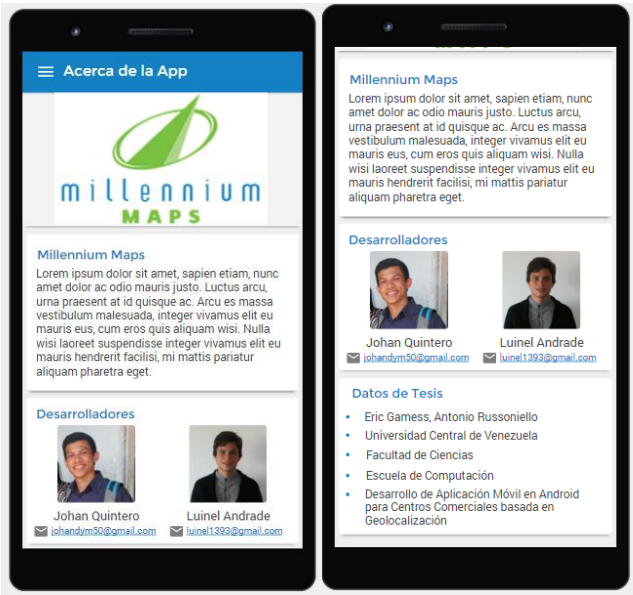

Figura 9.41: Diseño de la Pantalla de Acerca de la Aplicación de la Aplicación Móvil

En la Figura 9.42 se puede apreciar la pantalla de Comercios, en donde se muestra la lista de comercios que posee el centro comercial y una opción para poder filtrar comercios por categorías.

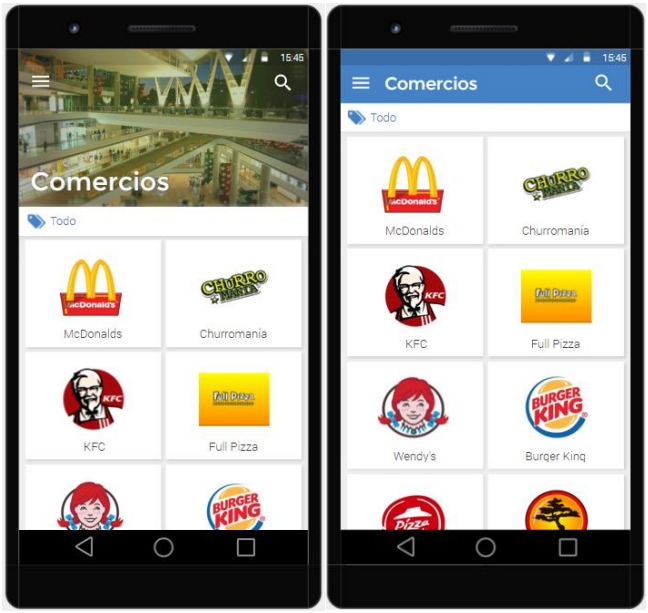

Figura 9.42: Diseño de la Pantalla de Comercios de la Aplicación Móvil

En la Figura 9.43 se puede apreciar la pantalla de Categorías, en donde se muestra la lista de categorías que posee el centro comercial.

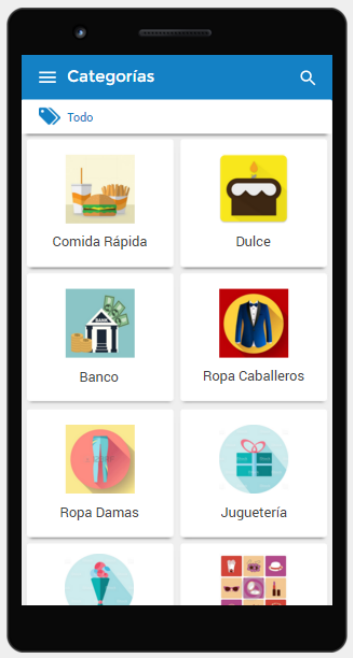

Figura 9.43: Diseño de la Pantalla de Categorías de la Aplicación Móvil

En la Figura 9.44 se puede apreciar la pantalla de Promociones, en donde se muestra la lista de promociones que poseen los comercios del centro comercial y una opción para poder filtrar promociones por categorías y por comercios.

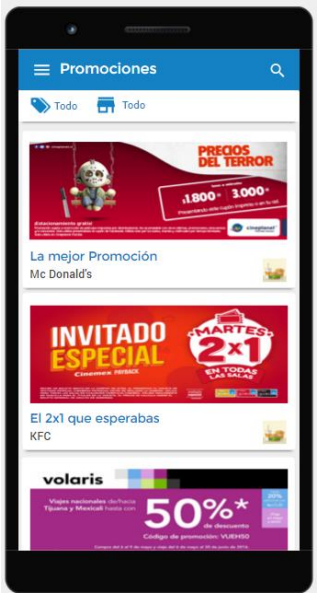

Figura 9.44: Diseño de la Pantalla de Promociones de la Aplicación Móvil

En la Figura 9.45 se puede apreciar la pantalla de Detalle de Evento en donde se muestra toda la información del respectivo evento como: una breve descripción, la fecha en la que se inició y en la que se termina, la hora en la que inicia y en la que termina y el lugar en donde se realizará el evento. Además, posee una opción para poder agregar el evento al calendario del dispositivo móvil.

Figura 9.45: Diseño de la Pantalla de Detalle de Evento de la Aplicación Móvil

En la Figura 9.46 se puede apreciar la pantalla de Detalle de Comercio en donde se muestra toda la información del respectivo comercio como: una breve descripción, número telefónico, correo electrónico, página web, horario, redes sociales, categorías a las que pertenece, promociones que posee y el nivel en donde se encuentra. Además, posee una opción para poder ubicar el comercio en los mapas del centro comercial.

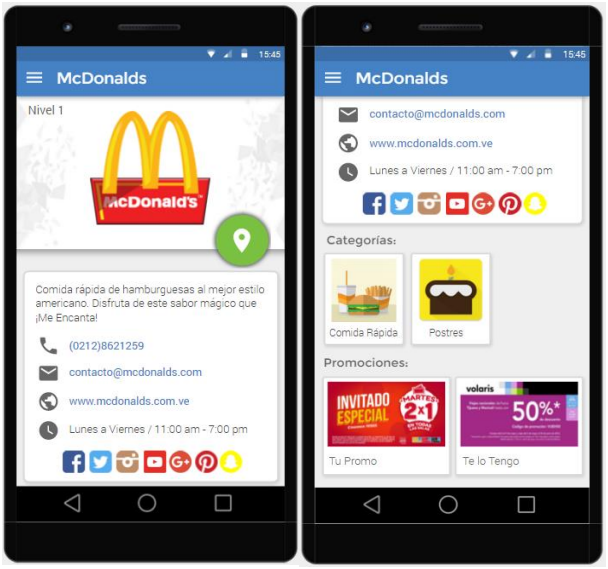

Figura 9.46: Diseño de la Pantalla de Detalle de Comercio de la Aplicación Móvil

En la Figura 9.47 se puede apreciar la pantalla de Mapas en donde se muestra la ubicación de un comercio en particular con un ícono de ubicación en color rojo. Además, se muestra las diferentes opciones de niveles del centro comercial, así como también una opción para poder ver la lista de servicios para ubicarlas en el mapa.

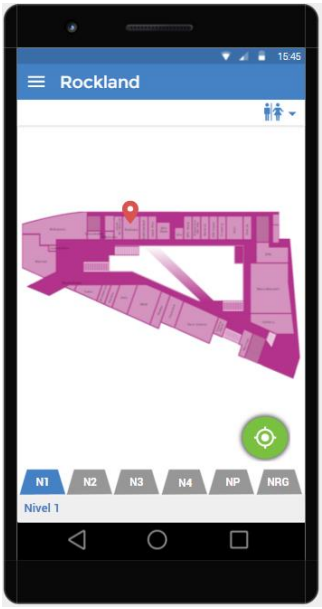

Figura 9.47: Diseño de la Pantalla de Mapas de la Aplicación Móvil Mostrando la Ubicación de Comercio

En la Figura 9.48 se puede apreciar la pantalla de Mapas en donde se muestra una indicación de cómo llegar a un determinado lugar dentro del centro comercial.

Figura 9.48: Diseño de la Pantalla de Mapas de la Aplicación Móvil Mostrando Indicación para Llegar a un Determinado Lugar

9.4.2. Instalación y Configuración del Ambiente de Trabajo de la Aplicación Móvil

Una vez concluido el proceso de análisis y de diseño de la solución móvil en donde se realizaron y revisaron los artefactos y prototipos de las interfaces de usuario, se procedió con la instalación y configuración de las tecnologías y herramientas a usar en el proceso de codificación y desarrollo. Un aspecto importante de mencionar es que Windows 10 es el sistema operativo instalado en cada uno de los computadores de los autores.

Uno de los requisitos para instalar y configurar Android Studio es tener instalado lo que se conoce como el JDK (Java Development Kit), el cual es un software que provee herramientas de desarrollo para la creación de programas en Java; el mismo fue descargado desde la página oficial[22] de Oracle. Posteriormente, el IDE Android Studio fue instalado descargando también desde su respectiva página oficial[23] un instalador para Windows. Cuando se instaló el mencionado IDE, durante el proceso de instalación pidió directamente descargar el Android Emulator, el Android SDK (Figura 9.49) y varios componentes del SDK adicionales. La versión de la API de Google instalada para desarrollar el proyecto es la número 25 correspondiente a la versión de Android 7.1.1.

[22] http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html
[23] https://developer.android.com/studio/index.html

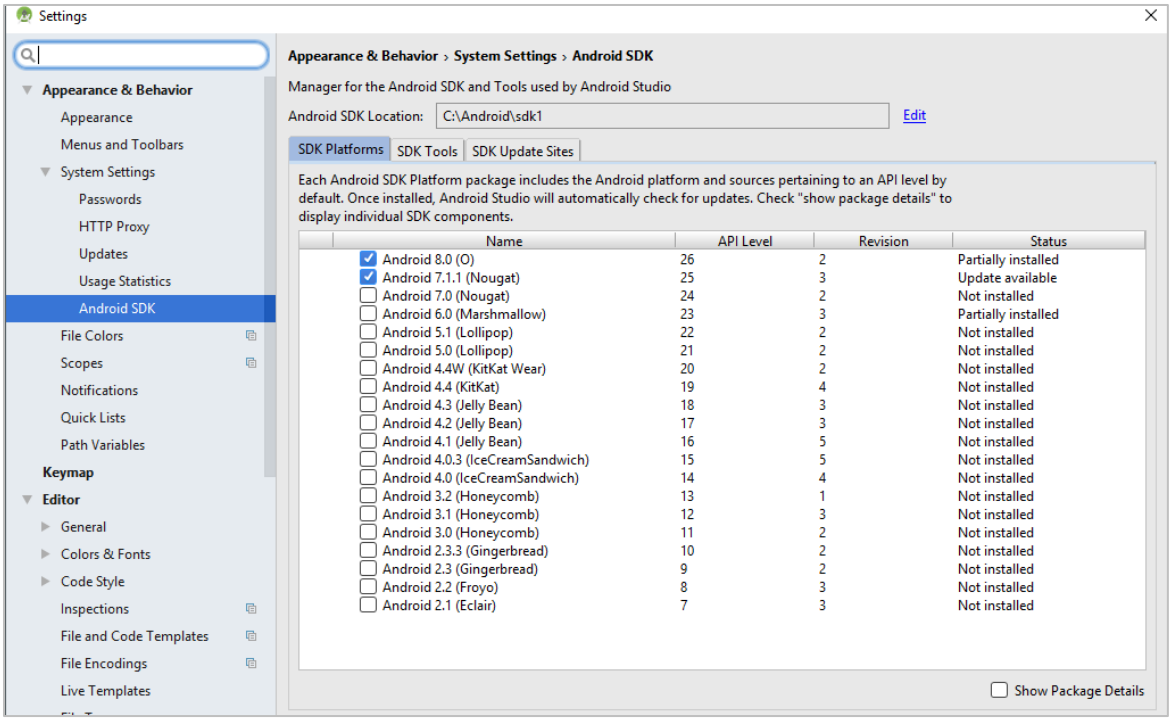

Figura 9.49: Android SDK

Para llevar el control de versiones sobre el código de la aplicación y el trabajo colaborativo se utilizó un repositorio de código Git configurado al igual que en la aplicación web con la plataforma GItlab. Esta herramienta se utilizó con el plug-in incluido en Android Studio para manejar repositorios Git.

9.4.3. Desarrollo de la Aplicación Móvil

El desarrollo de todos los módulos de la aplicación móvil se llevó a cabo entre los sprints 5 y 7, en conjunto con algunas correcciones generales a ciertas funcionalidades de la aplicación web.

Para el proceso de desarrollo se dividieron las clases java que contienen el código de la aplicación en diferentes paquetes, con sus respectivos subpaquetes, en donde, cada paquete representa un módulo diferente de la aplicación. En la Figura 9.50 se puede apreciar la distribución de dichos paquetes, los cuales son: About (Módulo de Acerca de la Aplicación), Category (Módulo de Categorías), Contact (Módulo de Contacto), Event (Módulo de Eventos), Main (Módulo de Inicio), Map (Módulo de Categorías), Promo (Módulo de Promociones), Service (Módulo de Servicios), Shop (Módulo de Comercios), Splash (Módulo de Splash Screen); y además se crearon otros dos (2) paquetes los cuales son: Model, que representa los modelos de los objetos que se manejan en el código de la aplicación, y RestClient, que representa la comunicación con la API para consultar la información que se mostrará en la aplicación.

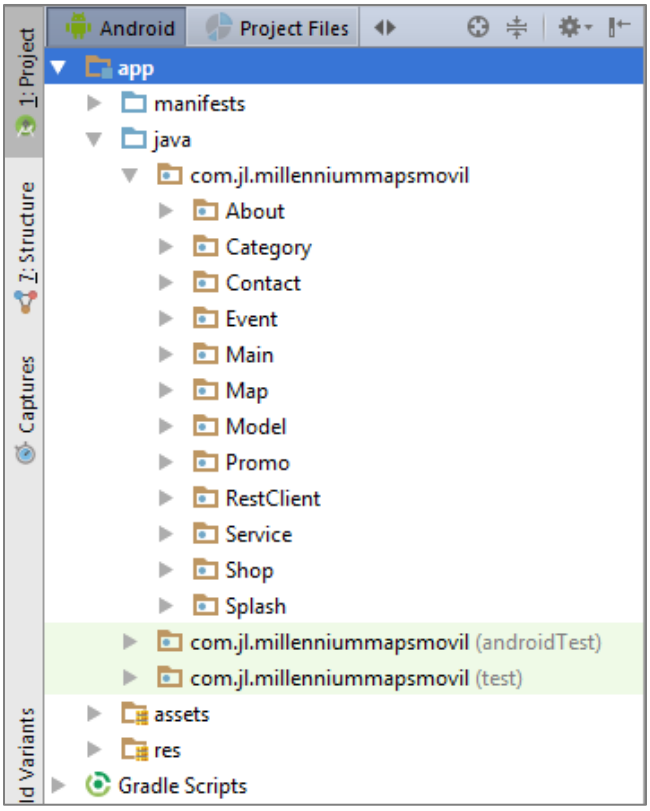

Figura 9.50: Estructura de los Paquetes del Proyecto en Android Studio

A su vez cada paquete se divide en diferentes subpaquetes que poseen las clases Java en donde se encuentra el código de la aplicación. En la Figura 9.51 se puede observar cómo se estructuran los paquetes en sus diferentes clases Java y subpaquetes.

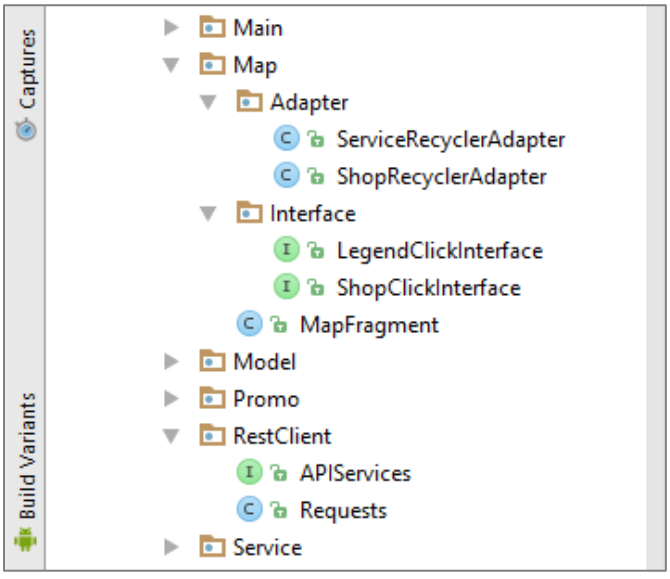

Figura 9.51: Estructura de los Paquetes con sus Clases Java y Subpaquetes del Proyecto en Android Studio

Estas clases Java que poseen el código de la aplicación manejan el comportamiento y las funcionalidades de la misma, sin embargo, también existen otros archivos muy importantes que son aquellos que representan las interfaces de usuario de la aplicación las cuales serán asociadas con las clases Java para que, desde el código, poder tener acceso a los elementos de cada vista. Éstos son archivos XML que se encuentran ubicados en el directorio *layout el* cual se encuentra en el directorio *res.* En la Figura 9.52 se puede apreciar el directorio layout con los archivos xml que representan las interfaces de usuario de la aplicación.

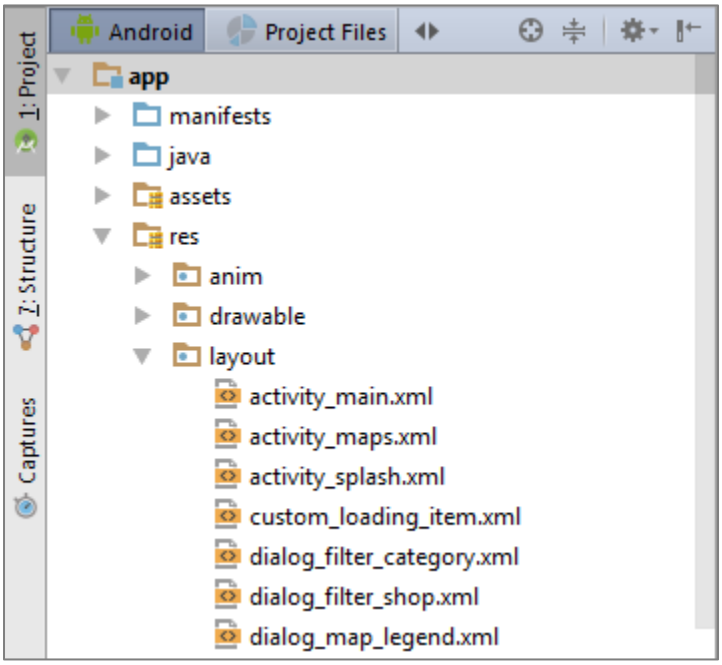

Figura 9.52: Directorio Layout con los Archivos XML de las Interfaces de Usuario

Además de los archivos ya explicados, también se hace uso de aquellos que representan ciertos recursos que puede usar la aplicación como: imágenes, strings, colores.

Para representar las imágenes se hace uso del directorio *drawable*, ubicado en el directorio *res*, en donde se alojarán las imágenes locales que usará la aplicación móvil. En la Figura 9.53 se muestra el directorio *drawable* con algunas de las imágenes que se encuentran en el proyecto las cuales son usadas en la aplicación.

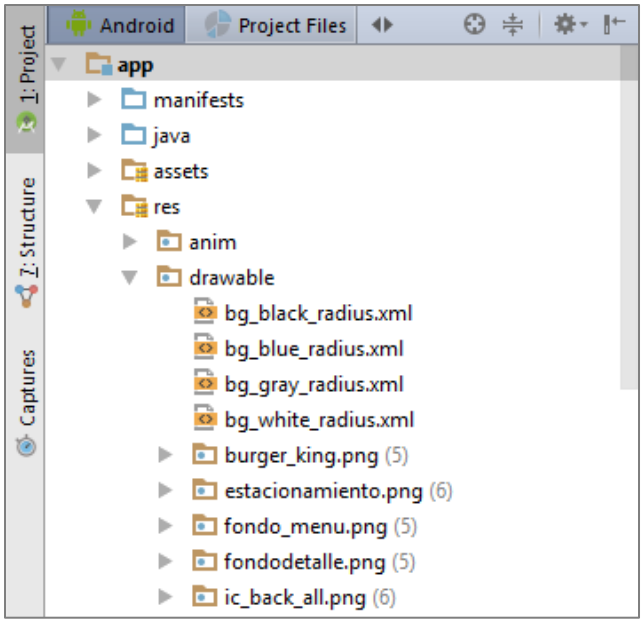

Figura 9.53: Directorio Drawable con Algunas Imágenes Locales en el Proyecto

Para el uso de ciertas cadenas de texto o strings definidas en el proyecto se hace uso de un archivo xml llamado *strings.xml* y éste se encuentra ubicado en el directorio *values* dentro del directorio *res.* Para establecer colores predfinidos, los cuales son los que usará la aplicación, se hace uso del archivo xml llamado *colors.xml* que se encuentra en el directorio *values* dentro del directorio *res.* En la Figura 9.54 se muestra el directorio *values* en donde se encuentran los archivos *strings.xml* y *colors.xml.*

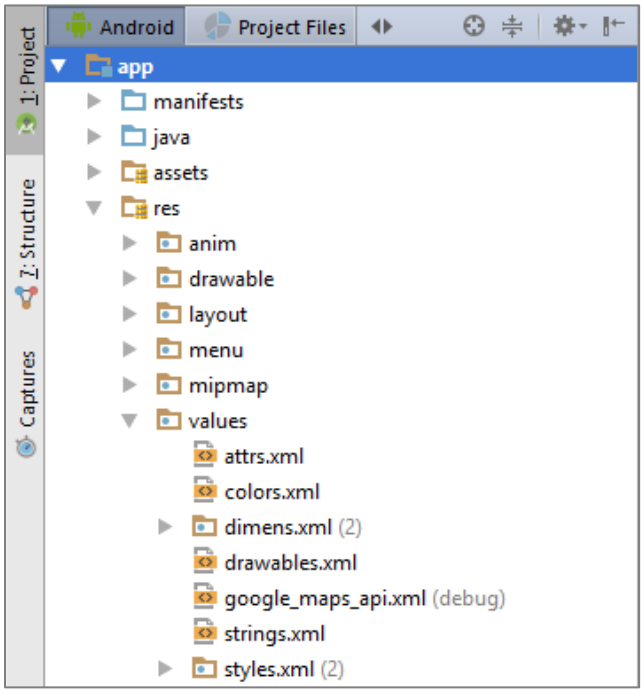

Figura 9.54: Directorio Values en Donde se Encuentran los Archivos colors.xml y strings.xml

Por último, también se hizo uso de los archivos *AndroidManifest.xml* y *gradle*, en donde, básicamente se realizan las configuraciones de la aplicación. El Android Manifest se explica anteriormente en la sección 4.3. Por otra parte, en cuanto a los archivos gradle, representan la configuración de la compilación de la aplicación, en donde se encuentra el SDK mínimo sobre el cual puede correr la aplicación, la versión del SDK con la que se compila la aplicación, las dependencias que posee, entre otras.

En la Figura 9.55 se muestran los archivos gradle y AndroidManifest.xml dentro de la estructura del proyecto de la aplicación.

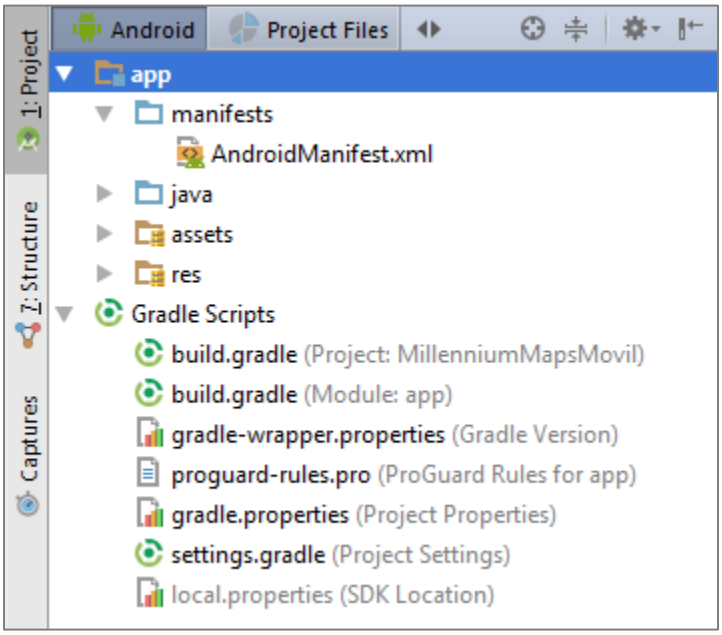

Figura 9.55: Archivos AndroidManifest.xml y Gradle en el Proyecto

Entrando como tal en el desarrollo de cada módulo de la aplicación, se tienen diferentes aspectos como: listado de elementos, detalle de un elemento en particular, entre otras. Para obtener toda la información que se muestra en cada módulo ya mencionado, se realizan peticiones a la API, usando Retrofit, el cual es un cliente REST para Android y Java.

Estas peticiones se realizan en el archivo *APIServices.java,* en donde se le especifica el tipo de petición (GET, POST, PUT, DELETE), la URI, el nombre de la función a la que se realizará la llamada desde el código, el objeto en el que recibirá la respuesta y los parámetros que se le pasarán a la petición. En la Figura 9.56 se muestra el archivo *APIServices.java* en donde se pueden observar lagunas de las peticiones que se realizan a la API.

APIServices.java

```
APIServices
30  public interface APIServices {
31      // INFO
32      @Headers({"apikey: $2y$10kPTKj3Ubl"})
33      @GET("info")
34      Call<ArrayList<Info>> getInfo();
35
36      // CONTENTS
37      @Headers({"apikey: $2y$10kPTKj3Ubl"})
38      @GET("content")
39      Call<PageContent> getContents();
40
41      // CATEGORIES
42      @Headers({"apikey: $2y$10kPTKj3Ubl"})
43      @GET("category")
44      Call<PageCategory> getCategories(@Query("page") int page);
45
46      @Headers({"apikey: $2y$10kPTKj3Ubl"})
47      @GET("category/{category}/shop")
48      Call<PageShop> getCategoriesShop(@Path("category") int category, @Query("page") int page);
49
50      @Headers({"apikey: $2y$10kPTKj3Ubl"})
51      @GET("category/{category}/promo")
52      Call<PagePromo> getCategoriesPromo(@Path("category") int category, @Query("page") int page);
53
54      // EVENTS
55      @Headers({"apikey: $2y$10kPTKj3Ubl"})
56      @GET("event")
57      Call<PageEvent> getEvents(@Query("page") int page);
```

Figura 9.56: Archivo APIServices.java con Algunas Peticiones a la API

Al recibir la respuesta de dichas peticiones, éstas se almacenan en un objeto según los modelos definidos en la aplicación, los cuales esta conformados por las propiedades y sus respectivos getters y setters. En la Figura 9.57 se muestra uno de los modelos como ejemplo, en donde se pueden observar sus propiedades con sus respectivos getters y setters.

Event.java ×

Event | setUpdated_at()

```
7   public class Event {
8       private int id;
9       private String nombre;
10      private String descripcion;
11      private String lugar;
12      private String fecha_inicio;
13      private String fecha_fin;
14      private String hora_inicio;
15      private String hora_fin;
16      private String created_at;
17      private String updated_at;
18
19      public int getId() { return id; }
22
23      public void setId(int id) { this.id = id; }
26
27      public String getHora_fin() { return hora_fin; }
30
31      public void setHora_fin(String hora_fin) { this.hora_fin = hora_fin; }
34
35      public String getHora_inicio() { return hora_inicio; }
38
39      public void setHora_inicio(String hora_inicio) { this.hora_inicio = hora_inicio; }
42
43      public String getFecha_fin() { return fecha_fin; }
46
47      public void setFecha_fin(String fecha_fin) { this.fecha_fin = fecha_fin; }
50
51      public String getFecha_inicio() { return fecha_inicio; }
```

Figura 9.57: Modelo de Evento

En cuanto a los módulos de la aplicación se tienen aquellos que listan una cierta cantidad de elementos, los cuales son: Comercios, Categorías, Promociones, Eventos y Servicios; y en cada uno de estos listados, al seleccionar un elemento específico se pasa al Detalle de dicho elemento, en donde se muestra la información necesaria del mismo.

En la Figura 9.58 se pueden apreciar las pantallas del listado de Comercios, Promociones y Eventos, en donde se muestran los elementos de cada uno, además de ciertas opciones de "filtrar" que se pueden usar para realizar un cierto filtrado de elementos, en donde, en Comercios se puede filtrar por categorías, en Promociones se puede filtrar tanto por categorías como por comercio y en Eventos se puede filtrar por rango de fechas.

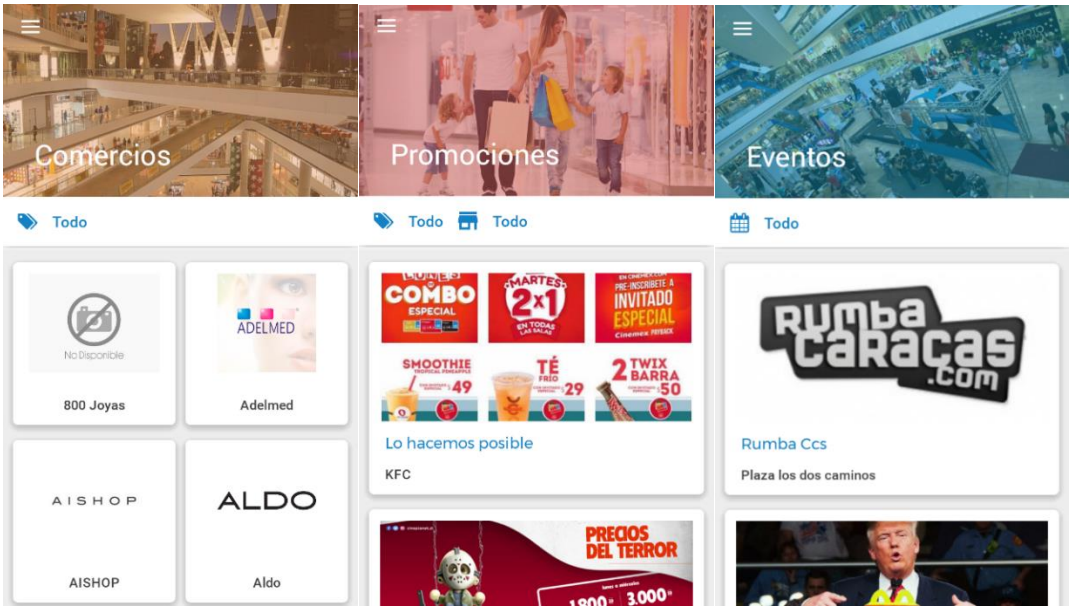

Figura 9.58: Pantallas del Listado de Comercios, Promociones y Eventos

En la Figura 9.59 se pueden apreciar las pantallas del detalle de Comercio, Promoción y Evento, en donde se muestra la información necesaria por cada uno de los elemento, como por ejemplo: en el detalle del comercio se muestra el nivel en el que se encuentra, una breve descripción, el correo electrónico, la página web, el horario, número telefónico, redes sociales, las categorías a las que pertenece y las promociones que posee; en el detalle de la promoción se muestra una breve descripción, la fecha de inicio y la fecha de fin, la hora de inicio y la hora de fin y las categorías a las que pertenece; y el detalle del evento muestra una breve descripción, la fecha de inicio y la fecha de fin, la hora de inicio y la hora de fin y el lugar en donde se llevará a cabo el evento.

Además, tanto el detalle del comercio como el del evento, poseen una opción adicional que permite realizar diferentes funciones, en donde, en el detalle del comercio se posee una opción para poder visualizar la ubicación de dicho comercio en los mapas del centro comercial y en el detalle del evento se tiene una opción para poder agregar dicho evento al calendario del dispositivo móvil.

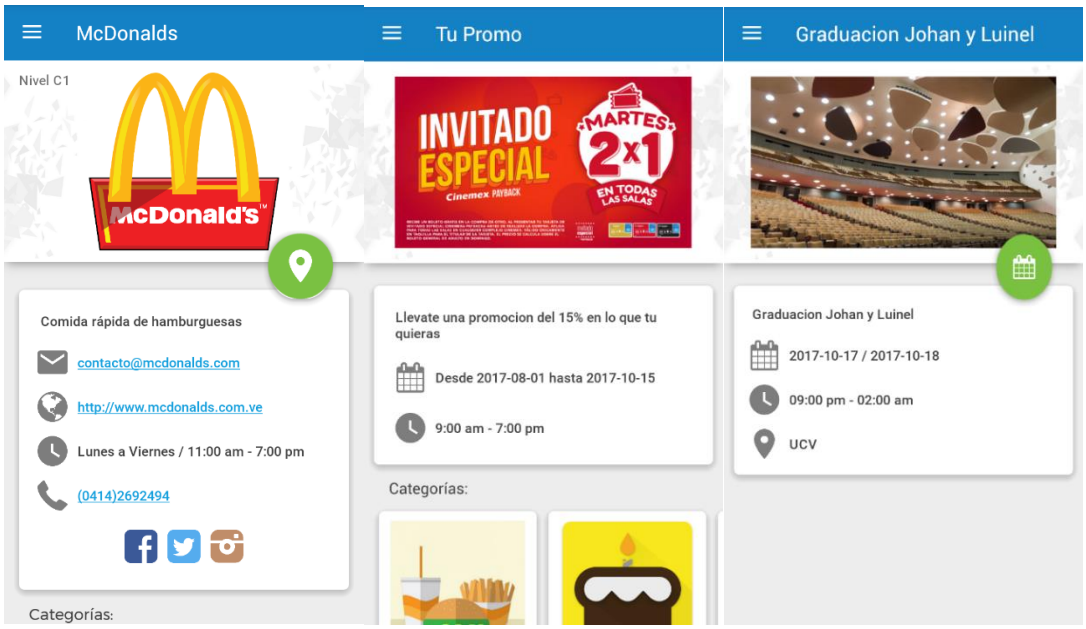

Figura 9.59: Pantallas del Detalle de Comercio, Promoción y Evento

Además, se desarrollaron módulos de “Contacto”, “Acerca de la Aplicación” y “Tu Puesto”, en donde, en el módulo de “Contacto” se muestra la información del centro comercial, en el módulo de “Acerca de la Aplicación” se muestra la información de la aplicación desarrollada, de los autores y del proyecto realizado, y en el módulo de “Tu Puesto” se muestra una ventana emergente con la posibilidad de anotar el puesto de estacionamiento donde el usuario estacionó su vehículo.

En la Figura 9.60 se muestra la pantalla de “Contacto”, en donde se puede observar toda la información del centro comercial como: una breve descripción, la dirección, correo electrónico, página web, horario, número telefónico y redes sociales. Además, posee una opción para poder ubicar el centro comercial en Google Maps.

Figura 9.60: Pantalla de Contacto

En la Figura 9.61 se puede apreciar la pantalla de "Acerca de la Aplicación", en donde se muestra una breve descripción de la aplicación, la información básica de los autores y los datos del proyecto, como título, tutores, institución, entre otras.

Figura 9.61: Pantalla de Acerca de la Aplicación

En la Figura 9.62 se muestra la ventana emergente para usar la funcionalidad de “Tu Puesto”, en donde se puede ver, eliminar y modificar el puesto de estacionamiento en donde el usuario estaciona su vehículo.

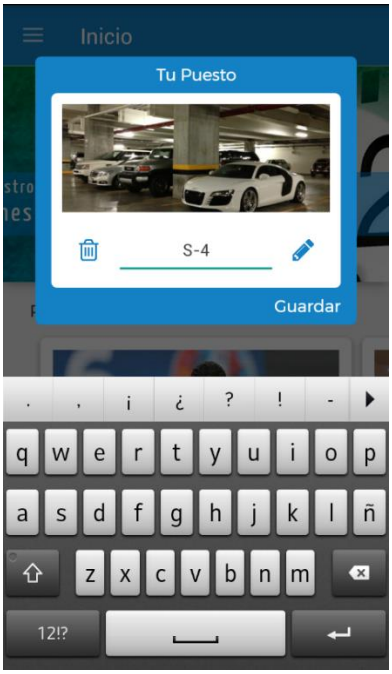

Figura 9.62: Ventana Emergente de Tu Puesto

Por último, se desarrolló el módulo de “Mapas”, el cual representa la parte más compleja de la aplicación. Este módulo tiene dos (2) variantes en su funcionamiento, las cuales se presencian cuando el usuario se encuentra con el dispositivo dentro de las instalaciones del centro comercial y cuando se encuentra fuera del mismo.

Cuando el usuario se encuentra con el dispositivo móvil fuera del centro comercial, al iniciar el módulo de mapas se muestra un mensaje emergente en donde se le indica al usuario que se encuentra fuera de las instalaciones del centro comercial y por lo tanto no se puede ubicar su posición actual dentro del mismo, sin embargo, aún puede hacer uso de las demás funcionalidades del módulo como, ver los mapas de cada piso del centro comercial, observar el nombre de cada piso y dos (2) opciones que representan las funciones de “cómo llegar a un comercio” y “leyenda de servicios”. Al iniciar el módulo de Mapas tomando en cuenta este caso se muestra el mapa del primer piso.

La función de “cómo llegar a un comercio”, consiste en mostrar la lista de los comercios del centro comercial y al seleccionar un comercio en específico se muestra al usuario un mensaje emergente con las indicaciones necesarias para saber cómo llegar al comercio requerido y la función de “leyenda de servicios”, consiste en mostrar la lista de los servicios del centro comercial y al seleccionar un servicio en específico se muestra al usuario un mapa por piso que posee solo el servicio requerido, es decir, se elimina de cada mapa los demás servicios que no son requeridos por el usuario.

En la Figura 9.63 se puede apreciar la pantalla de Mapas, en donde se muestran las indicaciones de cómo llegar a dos (2) comercios diferentes (Adelmed y Mercantil), además de la opción para ver la lista de comercios, la leyenda de los servicios, seis (6) botones para cambiar de pisos, una

opción específica para cuando se solicitan las indicaciones para ocultar o mostrar las indicaciones según cómo lo decida el usuario y, por último, una opción que se debe seleccionar en caso de que el usuario ya haya llegado al comercio requerido, que permite devolver los mapas a su estado base.

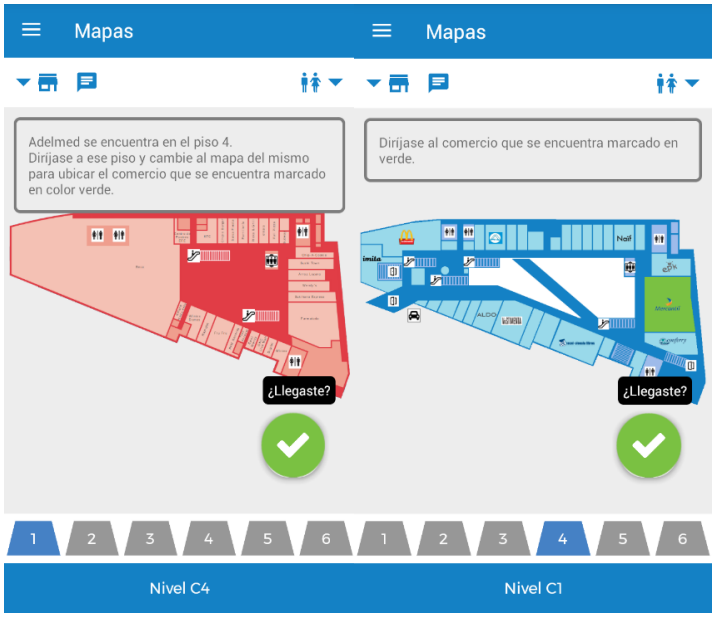

Figura 9.63: Pantallas del Módulo de Mapas con Indicaciones de Cómo Llegar a un Respectivo Comercio

Cuando el usuario se encuentra con el dispositivo móvil dentro de las instalaciones del centro comercial, el módulo de Mapas funciona prácticamente de la misma manera que el caso anterior, con la diferencia de que en este caso al iniciar el módulo de mapas se muestra directamente el mapa del piso en el que se encuentra el sector en donde está ubicado el usuario y dicho sector se marca en color verde en el mapa.

Una vez iniciado el módulo de Mapas, se realiza una captura de los access points del centro comercial cada ocho (8) segundos, en donde se obtienen de éstos las MAC, el SSID y la fuerza de la señal con la que es capturado cada access point al momento de la captura. Posteriormente, estos datos se envían a la API y se espera la respuesta de la petición, la cual retornaría el sector en donde se encuentra ubicado el usuario al momento de cada captura.

En cada captura, al momento de obtener el sector en donde se ubica el usuario, se carga la imagen del mapa con ese sector marcado, cuando vuelve a hacer la captura, si el nuevo sector que se obtiene como resultado de la API se encuentra en un piso diferente al del sector previo, se modifica la imagen del mapa a su diseño base sin el sector marcado en color verde y se muestra un mensaje emergente por un corto tiempo indicándole al usuario el piso en el que se encuentra ubicado, de manera que cuando cambie al mapa de ese piso, podrá ver el sector marcado en color verde en dicho mapa.

En la Figura 9.64 se puede apreciar el mapa del piso 4 (Nivel C1) con el sector en donde se encontraba el usuario al momento de la captura marcado en color verde.

Figura 9.64: Pantalla del Mapa del Piso 4 (Nivel C1) con el Sector en Donde se Ubica el Usuario Marcado en Color Verde

Además, es importante mencionar, que en este caso, cuando el usuario se encuentra dentro del centro comercial y se puede geolocalizar dentro del mismo, las indicaciones, a pesar de que funcionan de manera similar al caso anterior, el mensaje que se le muestra al usuario varía, debido a que, como en este caso se puede calcular la ubicación del usuario dentro de las instalaciones del centro comercial, las indicaciones pueden ser más específicas.

En la Figura 9.65 se puede apreciar el mapa del piso 4 (Nivel C1), en donde se muestran las indicaciones de cómo llegar al comercio requerido por el usuario (Back & Forth) de forma más detallada que en el caso anterior, además de la opción para ver la lista de comercios, la leyenda de los servicios, seis (6) botones para cambiar de pisos, una opción específica para cuando se solicitan las indicaciones para ocultar o mostrar las indicaciones según cómo lo decida el usuario y, por último, una opción que se debe seleccionar en caso de que el usuario ya haya llegado al comercio requerido, que permite devolver los mapas a su estado base.

Figura 9.65: Pantalla del Módulo Mapas en Donde se Muestra Indicaciones de Cómo Llegar un Respectivo Comercio Obteniendo la Ubicación del Usuario Dentro del Centro Comercial

9.4.4. Desarrollo de Herramientas para el Proceso de Fingerprinting

Adicional al desarrollo de la aplicación web y de la aplicación móvil en las que se centra el presente trabajo de investigación, para apoyar el proceso de *fingerprinting* que debe hacerse para alimentar la BD y poder procesar algoritmo de los k-vecinos más cercanos, se desarrolló una aplicación móvil para SO Android y un script en lenguaje C++. Es importante recordar que el proceso de *fingerprinting* consiste en medir la potencia de las señales recibidas de cada AP en un lugar en específico.

Entre los motivos que tuvieron los autores para desarrollar esta aplicación móvil es que de esa manera se facilita y agiliza la recolección de los datos de los APs por los distintos sectores del centro comercial, ya que anteriormente eso se hacía de forma manual en un cuaderno y usando aplicaciones de terceros para el análisis de las señales. Una ventaja que proporciona es que los datos recolectados los guarda en un archivo de texto en la memoria externa del dispositivo y el formato de mencionado archivo podía ser modificado por los autores antes de ir al centro comercial a realizar las capturas según las necesidades que se presentaban.

Su uso es muy sencillo, consiste en identificar el sector a capturar para tener así una referencia, seleccionar si se recolecta datos para realizar inserciones en BD de las mediciones tomadas o datos con el formato necesario para probar en la API la funcionalidad de geolocalización, y luego presionar el botón correspondiente para iniciar la captura.

Los datos que captura todos los APs que percibe en un instante de tiempo son: SSID, MAC y potencia de la señal.

En la Figura 9.66 se puede ver cómo es la pantalla de la aplicación.

Figura 9.66: Pantalla de Aplicación Móvil para el Fingerprinting

En la Figura 9.67 se muestra un ejemplo del archivo que genera la aplicación cuando se selecciona la opción de "Inserción" en donde destaca los datos de los APs por número de captura que se realizaron.

```
CAPTURA NUMERO: 1:
insert into medidas(sector,mac,ssid,fuerza) values ('s1','00:0d:67:44:3b:63','MOVISTAR WIFI',89);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','e8:de:27:74:df:4c','paos textil',71);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','00:19:0c:20:aa:e0','GTM',80);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','20:aa:4b:cd:4e:27','VIQIU',91);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','e8:de:27:75:25:c0','Xcess Millennium',97);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','c8:3a:35:1e:e8:a8','LANWF',92);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','e8:de:27:cb:10:66','PH12',92);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','00:0e:f4:d3:90:f9','KasdaD390F8',90);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','f8:d1:11:40:a3:4c','TP-LINK_40A34C',87);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','e8:de:27:d0:66:02','OMI',86);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','6c:19:8f:b9:e1:16','PHX',90);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','60:e3:27:36:9d:6e','aistiendas',87);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','6c:19:8f:0a:75:f2','Boticario',95);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','f4:ec:38:e4:44:e6','parrilleria',99);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','00:19:e0:a0:8b:c8','PI Millennium',75);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','74:ea:3a:f5:bc:66','Seccion 7121',99);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','00:0d:67:43:b6:46','MOVISTAR WIFI',90);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','00:0d:67:43:b6:49','MOVISTAR WIFI',92);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','64:66:b3:40:44:ea','ECOMIL',92);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','00:24:a5:c9:cc:ad','Salud y Bienestar',96);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','90:72:40:22:67:40','Red Wi-Fi de Oswaldo',99);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','64:66:b3:ab:89:0e','oswaldo',99);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','c8:3a:35:34:59:50','borghese',97);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','00:0d:67:43:b6:48','MOVISTAR WIFI',93);

CAPTURA NUMERO: 2:
insert into medidas(sector,mac,ssid,fuerza) values ('s1','00:0d:67:44:3b:63','MOVISTAR WIFI',90);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','e8:de:27:74:df:4c','paos textil',74);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','00:19:0c:20:aa:e0','GTM',85);
insert into medidas(sector,mac,ssid,fuerza) values ('s1','20:aa:4b:cd:4e:27','VIQIU',91);
```

Figura 9.67: Archivo de "Inserción" Generado por Aplicación de Fingerprintig

En la Figura 9.68 se visualiza un ejemplo del archivo que genera la aplicación cuando se selecciona la opción de “Localización”. El archivo muestra SSID, MAC y potencia de la señal de los APs por cada captura y debajo genera una línea con el formato de entrada necesario para usar la funcionalidad de geolocalización en la API, muy útil para cuando se estaba desarrollando y probando dicha funcionalidad haciendo uso de la herramienta Postman. La línea en cuestión es la número 72 que se aprecia en el archivo.

```
44  CAPTURA NUMERO: 2:
45
46  Localizacion 2:
47
48  MOVISTAR WIFI - 00:0d:67:44:3b:63,85||
49  paos textil - e8:de:27:74:df:4c,68||
50  VIQIU - 20:aa:4b:cd:4e:27,85||
51  TP-LINK_40A34C - f8:d1:11:40:a3:4c,84||
52  PHX - 6c:19:8f:b9:e1:16,84||
53  PI Millennium - 00:19:e0:a0:8b:c8,78||
54  MOVISTAR WIFI - 00:0d:67:43:b6:44,89||
55  MOVISTAR WIFI - 00:0d:67:43:b6:48,89||
56  OMI - e8:de:27:d0:66:02,86||
57  MOVISTAR WIFI - 00:0d:67:44:3b:5e,84||
58  MOVISTAR WIFI - 00:0d:67:43:b6:46,89||
59  MOVISTAR WIFI - 00:0d:67:44:3b:62,85||
60  MOVISTAR WIFI - 00:0d:67:43:b6:49,90||
61  MOVISTAR WIFI - 00:0d:67:43:b6:47,90||
62  KasdaD390F8 - 00:0e:f4:d3:90:f9,85||
63  MOVISTAR WIFI - 00:0d:67:43:b6:43,89||
64  aistiendas - 60:e3:27:36:9d:6e,91||
65  GTM - 00:19:0c:20:aa:e0,80||
66  PH12 - e8:de:27:cb:10:66,95||
67  MOVISTAR WIFI - 00:0d:67:43:b6:45,89||
68  OMAR - 00:1d:7e:0c:60:48,94||
69  SolucionesPopPrint - bc:f6:85:bf:03:86,97||
70  veloescalante - 98:de:d0:8b:71:dc,93||
71
72  00:0d:67:44:3b:63,85||e8:de:27:74:df:4c,68||20:aa:4b:cd:4e:27,85||f8:d1:11:40:a3:4c,84||6c:19:8f:b9:e1:16,84||00:19:e0:a0:8b:c8,78||00:0d:67:
    43:b6:48,89||e8:de:27:d0:66:02,86||00:0d:67:44:3b:5e,84||00:0d:67:43:b6:46,89||00:0d:67:44:3b:62,85||00:0d:67:43:b6:49,90||00:0d:67:43:b6:47,
    85||00:0d:67:43:b6:43,89||60:e3:27:36:9d:6e,91||00:19:0c:20:aa:e0,80||e8:de:27:cb:10:66,95||00:0d:67:43:b6:45,89||00:1d:7e:0c:60:48,94||bc:f6
    :d0:8b:71:dc,93
73
74  CAPTURA NUMERO: 3:
75
76  Localizacion 3:
77
78  MOVISTAR WIFI - 00:0d:67:44:3b:63,84||
79  paos textil - e8:de:27:74:df:4c,69||
80  VIQIU - 20:aa:4b:cd:4e:27,86||
```

Figura 9.68: Archivo de “Localización” Generado por Aplicación de FIngerprinting

Ahora se explicará el script realizado en lenguaje C++ para procesar el archivo de “Inserción” que genera la aplicación móvil. Lo que se busca es que con ejecutarlo indique mediante un archivo cuales son los APs estables que se pueden seleccionar para ingresar en BD y así hacer uso del algoritmo de los k-vecinos.

Para ejecutarlo, el script se tiene que ubicar en el directorio en donde están los archivos de inserción. Una vez iniciado solicitará el nombre con la extensión del archivo a procesar. Internamente al ejecutarse, lo que hace es agrupar todos los APs, calcula el promedio de las potencias de las señales de todas las inserciones por cada AP y suma la cantidad de veces que se repiten. Los resultados los guarda en un archivo con formato que se aprecia en la Figura 9.69 en donde los que están marcado con la frase “-- ESTE” son los que se deben seleccionar para generar la base de datos.

```
00:0d:67:44:3b:63, MOVISTAR WIFI, 89, 20 -- ESTE
e8:de:27:74:df:4c, paos textil, 73, 20 -- ESTE
00:19:0c:20:aa:e0, GTM, 82, 16
20:aa:4b:cd:4e:27, VIQIU, 87, 20 -- ESTE
e8:de:27:75:25:c0, Xcess Millennium, 92, 18  -- ESTE
c8:3a:35:1e:e8:a8, LANWF, 90, 19  -- ESTE
e8:de:27:cb:10:66, PH12, 92, 2
00:0e:f4:d3:90:f9, KasdaD390F8, 89, 19 -- ESTE
f8:d1:11:40:a3:4c, TP-LINK_40A34C, 80, 20 -- ESTE
e8:de:27:d0:66:02, OMI, 86, 18 -- ESTE
6c:19:8f:b9:e1:16, PHX, 92, 20  -- ESTE
60:e3:27:36:9d:6e, aistiendas, 92, 18  -- ESTE
6c:19:8f:0a:75:f2, Boticario, 91, 11
f4:ec:38:e4:44:e6, parrilleria, 93, 16
00:19:e0:a0:8b:c8, PI Millennium, 78, 20 -- ESTE
```

Figura 9.69: Formato de Archivo Generado por Script de C++

10. Pruebas y Análisis de Resultados

En este capítulo se presentan las pruebas realizadas a la aplicación móvil y los resultados obtenidos, lo cual permite evaluar entre otras cosas el nivel de usabilidad, el nivel de calidad y que el producto de software desarrollado cumpla con los objetivos y funciones específicas para los cuales ha sido diseñado.

10.1. Pruebas de Aceptación

Antes de empezar a utilizarse un software y pasarse a lo que se conoce como fase de producción es importante evaluar si las ventajas y soluciones que brinda ante una problemática planteada realmente justifican su uso. Dentro de la solución tecnológica en general desarrollada, la aplicación móvil es la que va dirigida a los usuarios finales, por lo que se sometió a una evaluación para probar y determinar si la misma cumple con los requerimientos establecidos al momento de diseñar la aplicación.

Para realizar las pruebas de aceptación se aplicó una encuesta técnica a un grupo de usuarios finales comprendido por veinte (20) personas de diferentes géneros, nivel social y nivel de instrucción con el fin de conocer la opinión general sobre la usabilidad y funcionamiento de un grupo variado de personas que represente al amplio número de visitantes que frecuentan un centro comercial.

A continuación, se muestra las veinte (20) preguntas que se formularon en la encuesta.

1. ¿Cómo califica la combinación de colores usados para la aplicación?
2. ¿Cómo califica la distribución de los elementos de la aplicación?
3. ¿Cómo califica los diseños de la aplicación?
4. ¿Cómo califica la navegación de la aplicación?
5. ¿Cómo califica la fluidez del funcionamiento de la aplicación?
6. ¿Considera que la aplicación es fácil e intuitiva de manejar?
7. ¿Considera agradable la Pantalla de Carga o Splash Screen al momento de abrir la aplicación?
8. ¿El menú de la aplicación muestra sus opciones de forma clara y legible?
9. ¿Considera que los elementos que se muestran en el Inicio o Home de la aplicación son los indicados?
10. ¿Considera importante para la aplicación la funcionalidad de “Tu Puesto”?
11. ¿Considera que es fácil e intuitivo manejar la funcionalidad de “Tu Puesto”?
12. ¿Considera que el listado de Comercios, Categorías, Promociones, Eventos y Servicios muestra con claridad cada elemento y a qué hacen referencia?
13. ¿Se entiende con claridad la opción de “filtrar” en el listado de Comercios, Promociones y Eventos?
14. ¿Considera que en el detalle de los elementos (Comercios, Promociones, Eventos, Servicios) se muestra toda la información necesaria del elemento respectivo?
15. ¿Se entiende con claridad la opción de “ubicación” en el Detalle de Comercio, Detalle de Servicio y Contacto?
16. ¿Se entiende con claridad la opción de “agregar al calendario” en el Detalle de Evento?
17. ¿Se entiende con claridad que los textos que representan correos, páginas web y números telefónicos, son clicables?
18. ¿Considera que es fácil e intuitivo el manejo de la funcionalidad de “Mapas” en la aplicación?

19. ¿Se entiende con claridad la opción de “servicios” y “comercios” de la funcionalidad de “Mapas”?
20. ¿Considera que las indicaciones acerca de cómo llegar a un determinado comercio de la funcionalidad de “Mapas” son claras?

La escala de opinión de los usuarios se mide según las siguientes cinco (5) posibles respuestas:

1. Muy mal
2. Mal
3. Regular
4. Bien
5. Muy Bien

En la Figura 10.1 y en la Figura 10.2 se observan los resultados obtenidos de la encuesta:

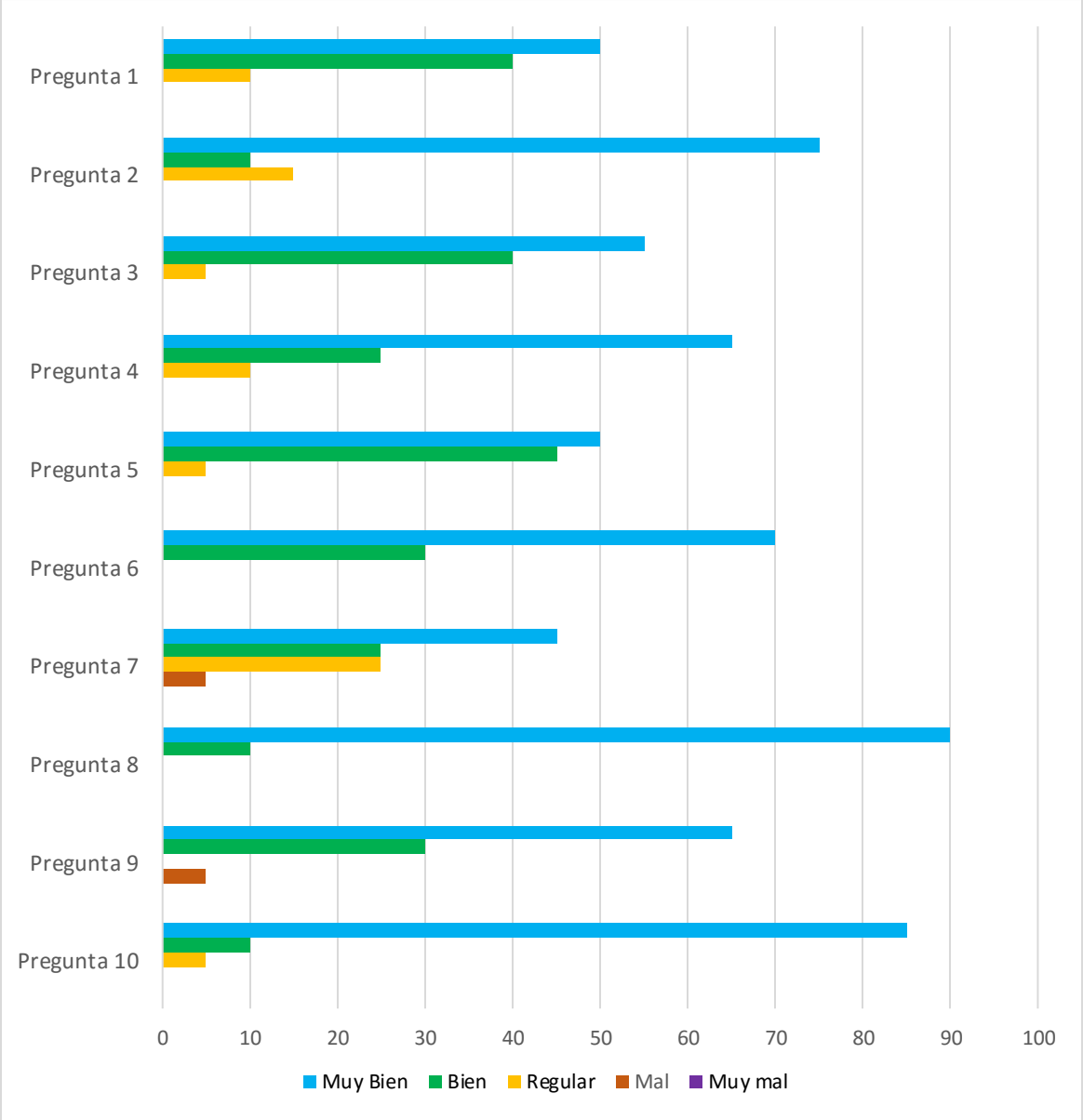

Figura 10.1: Resultados de la 1 a la 10 de las Preguntas de la Encuesta

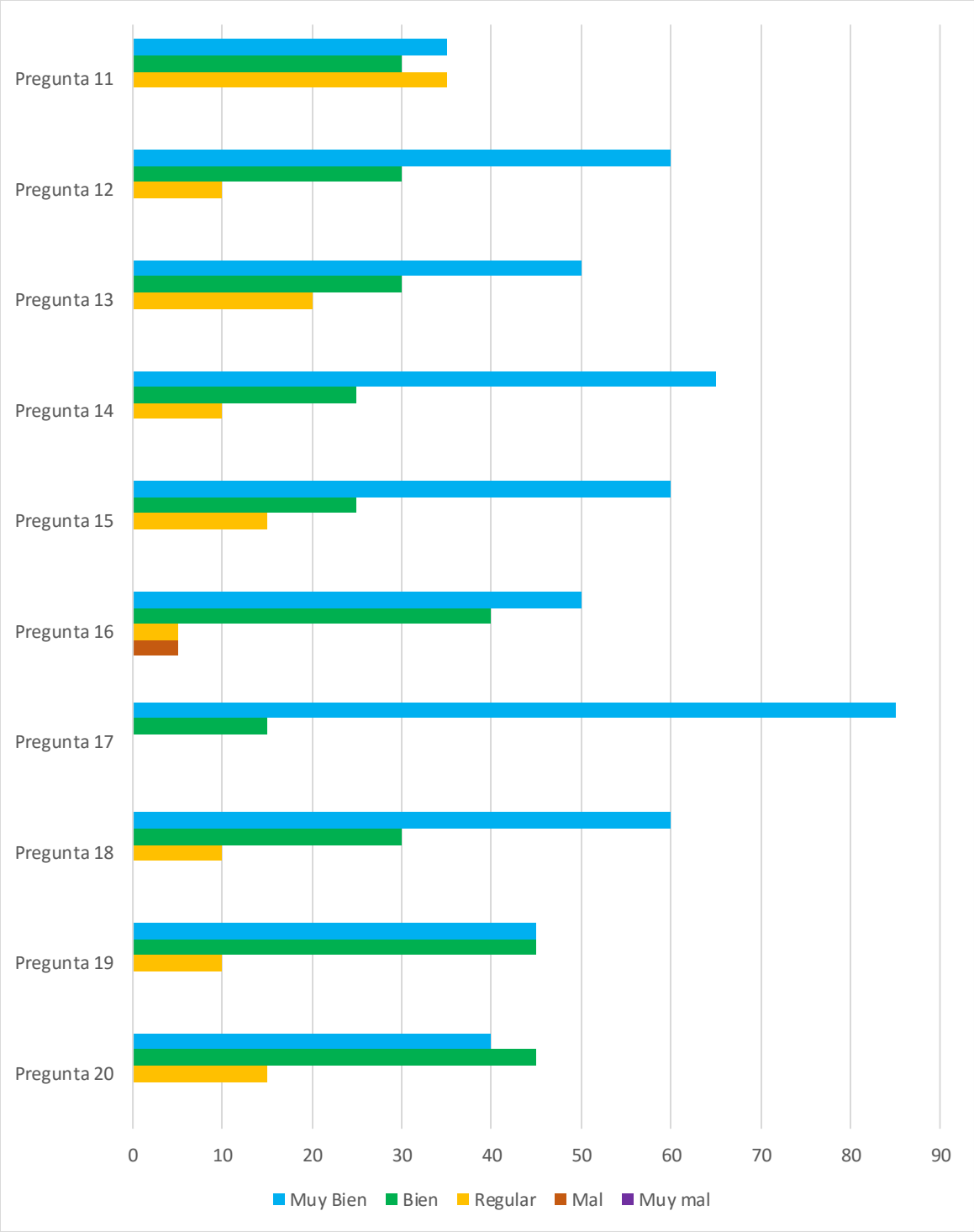

Figura 10.2: Resultados de la 11 a la 20 de las Preguntas de la Encuesta

De manera general, los gráficos evidencian que las respuestas dadas por las personas que usaron la aplicación fueron positivas, por lo que se puede decir que la misma cumple con los requerimientos funcionales establecidos y con el alcance definido en un principio. Sin embargo, se notó que en muchas de las preguntas formuladas se obtuvo un porcentaje de respuestas regulares que, si bien no representa la mayoría de las opiniones, es importante considerar para realizar correcciones, en trabajos futuros o antes de que la aplicación pase a una fase de producción, ya que indican que muchos aspectos de la misma si bien ya funcionan pueden mejorarse para aumentar el nivel de calidad, funcionamiento y usabilidad del producto.

10.2. Pruebas de la Funcionalidad de Geolocalización

Para probar el algoritmo de los k-vecinos como método de geolocalización por WiFi y para efectos del presente trabajo de investigación, se decidió utilizar como caso de estudio, únicamente dos (2) niveles del centro comercial Millennium Mall, que son Nivel C1 y Nivel C2, en donde, el Nivel C1 se dividió en ocho (8) sectores (ver Figura 10.3), del sector 1 hasta el sector 8, y el Nivel C2 se dividió en siete (7) sectores (ver Figura 10.4), del sector 9 hasta el sector 15.

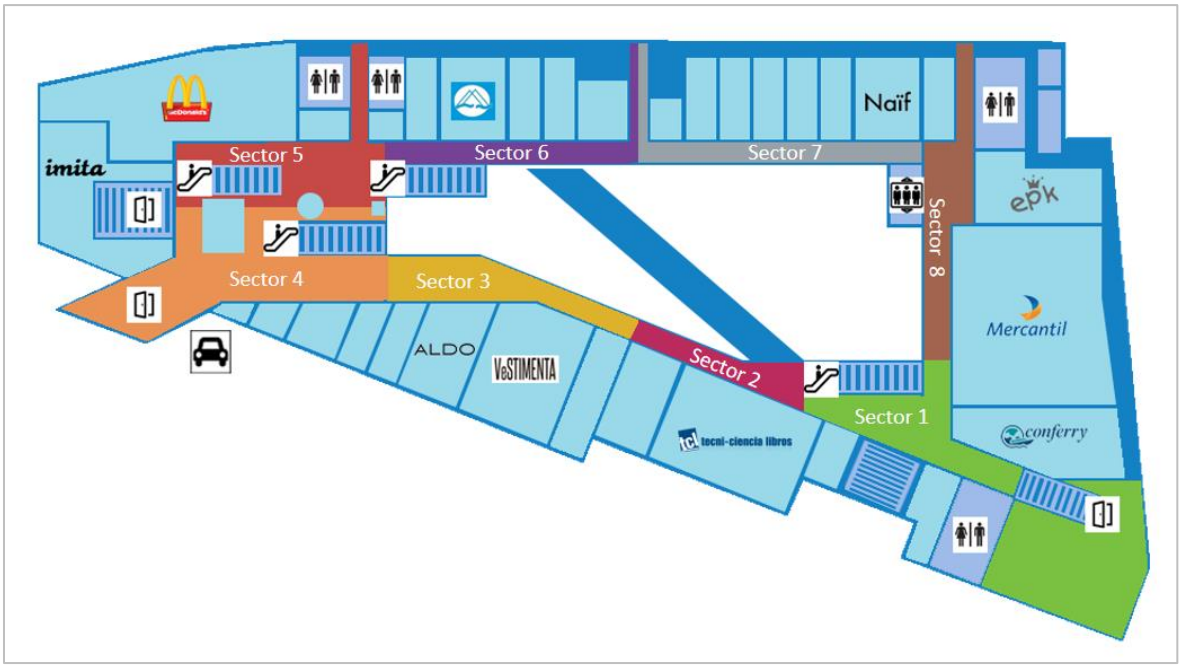

Figura 10.3: Sectores Identificados del Nivel C1

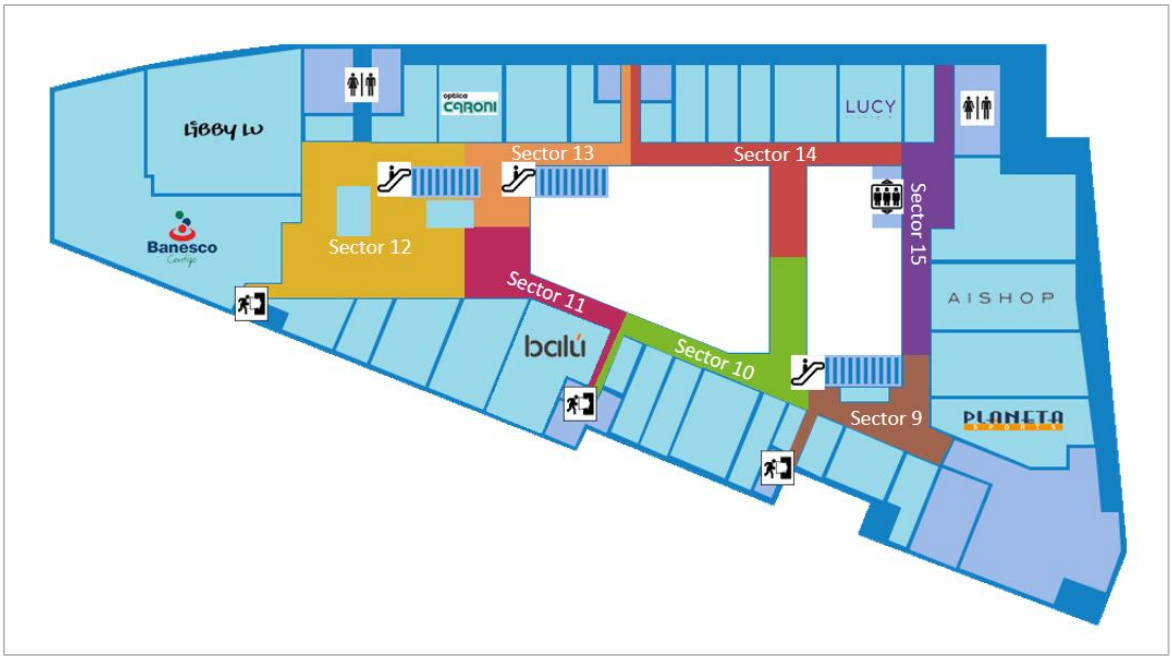

Figura 10.4: Sectores Identificados del Nivel C2

Cómo se puede apreciar en las figuras anteriores, los sectores en que se dividieron los niveles del mapa tienen formas de polígonos irregulares y fueron definidos por los autores sin usar alguna técnica formal, sino que, por practicidad se definían estimando sus tamaños según las características de los pasillos, buscando que todos tuviesen un tamaño parecido entre ellos y buscando adaptar dichos sectores a las necesidades de precisión del proyecto.

Para generar la base de datos a utilizar por el algoritmo mencionado en la funcionalidad de geolocalización de la aplicación móvil, se realizó el proceso de fingerprinting en un horario en el que el centro comercial restringe el acceso al público en general. Para ello se tomaba como muestra de las señales de los APs treinta (30) capturas en el centro de cada uno de los sectores definidos en intervalos de tres (3) segundos entre captura y captura. Cabe destacar que dicha posición centrada no se estableció calculando exactamente el centroide del polígono irregular que representa el sector sino una posición aproximada.

Se hacía un análisis de la data generada en las capturas y se seleccionaba los APs que fueron más estables para ser procesados e ingresados en la base de datos por medio de la aplicación web. Considerando como AP estable aquel que fue capturado en más del 90% del total de capturas realizadas en un determinado sector.

Es importante destacar que el fingerprinting fue realizado haciendo uso de los APs visibles dentro de las instalaciones del centro comercial, pero sin conocer datos como: tipo de routers que eran, ubicación de los mismos, cuando los apagan o encienden, etc. Todo eso motivado a que el centro comercial no posee instalada una infraestructura de red propia que pueda usarse para la geolocalización.

Una vez implementada la funcionalidad de geolocalización en la API y en la aplicación móvil, se realizaron varias pruebas dentro de las instalaciones del centro comercial para determinar el funcionamiento de la misma y los resultados del algoritmo implementado. En las primeras

pruebas se obtuvieron errores que fueron llevando a los autores a afinar con más detalle los datos del algoritmo.

A continuación, se muestra los resultados de las pruebas realizadas con la versión más estable de la API de geolocalización que se logró desarrollar. Estas pruebas se realizaron en dos (2) ámbitos diferentes, el primero cuando el centro comercial se encontraba cerrado, es decir, los pasillos estaban totalmente libres y no había personas en el mismo, y el segundo cuando el centro comercial ya había realizado su apertura, por lo que se encontraban en todos los pasillos los visitantes del centro comercial y a su vez usando sus dispositivos móviles, esto con el objetivo de verificar si se podrían producir alteraciones en las señales de los APs de los comercios debido a la gran cantidad de visitantes y empleados usando sus dispositivos móviles ya sea conectado a una red WiFi específica o no.

Para las pruebas se usó la funcionalidad de geolocalización en la aplicación en donde se hicieron veinte (20) capturas por cada sector, en donde, las primeras cinco (5) capturas se realizaron de manera estática en el centro del sector y desde la sexta hasta la última captura se realizaron movimientos a lo largo de todo el sector, para ir observando los resultados que se iban obteniendo.

A lo largo de las veinte (20) capturas en cada sector, se iban anotando cuántos errores se presentaban y qué sector erróneo mostraba. Con estas anotaciones, se obtuvo un porcentaje de resultados correctos e incorrectos por cada sector y posteriormente se calculó un porcentaje general de resultados correctos e incorrectos que incluye los resultados de las pruebas de todos los sectores.

En el proceso de pruebas, se consideraba un resultado como correcto en el caso de que la API retornara el sector donde exactamente se encontraba posicionado el usuario o en el caso de que el usuario estaba posicionado en una zona limítrofe o borde con otro sector. Dicha consideración se debe a que físicamente los sectores no están divididos entre sí y el “sector vecino” se empezaba a retornar cuando se estaba en un rango de 2 a 5 m del área donde limitan lógicamente según la división establecida por los autores.

En el proceso de pruebas, se consideraba un resultado como incorrecto en el caso de que la API retornaba un sector que se encuentra en otro nivel con respecto a aquel donde se ubica realmente el usuario y en el caso de que retornara un sector del mismo nivel pero que físicamente se encuentra muy alejado a donde realmente está posicionado el usuario.

En la Figura 10.5 se aprecia los resultados de hacer las pruebas de geolocalización en los espacios interiores del centro comercial cuando se encontraba cerrado al público y los comercios estaban cerrados.

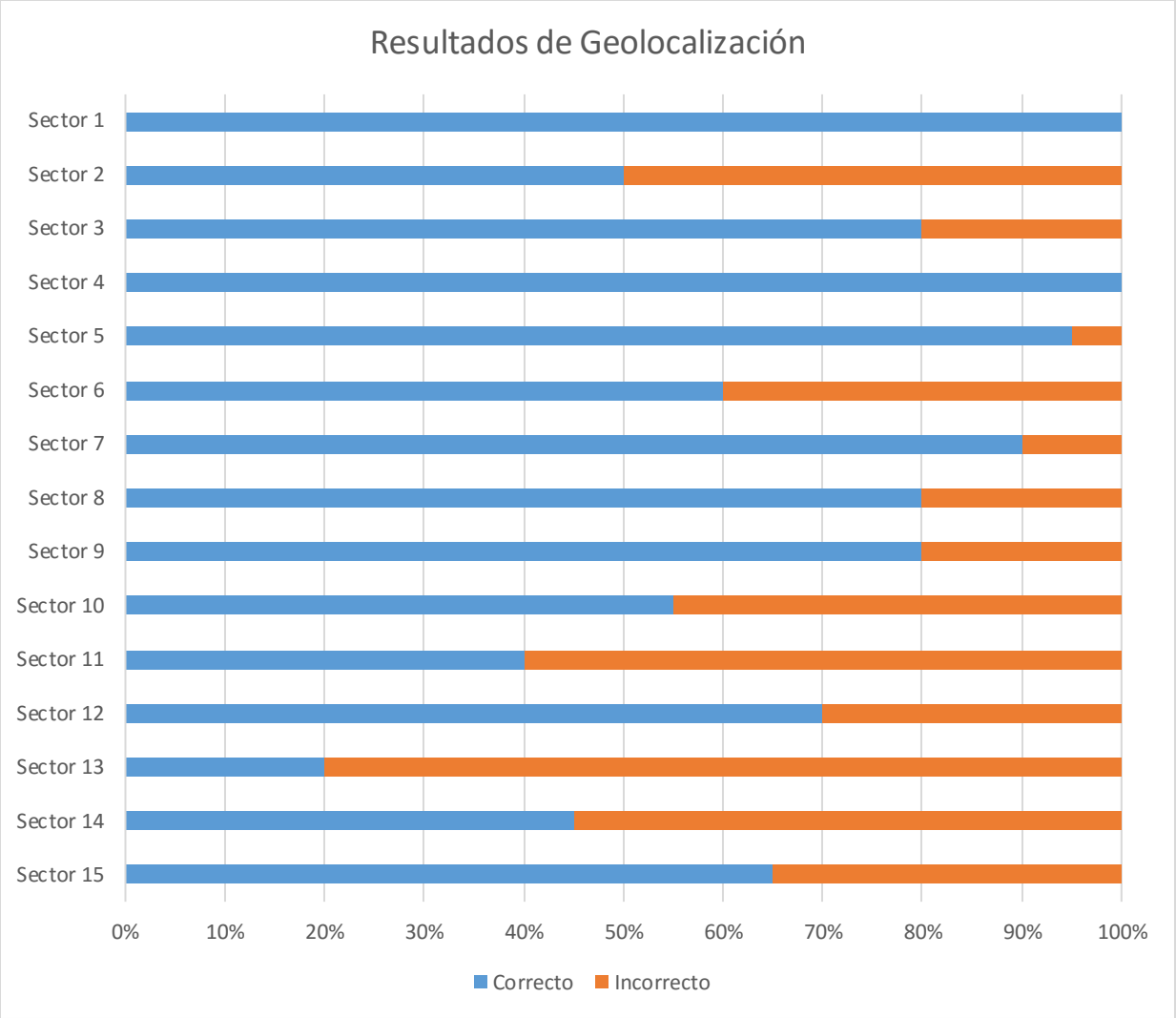

Figura 10.5: Resultados de Geolocalización con Centro Comercial Cerrado al Público

En la Figura 10.6 se muestra el porcentaje general de resultados correctos e incorrectos de las pruebas que se realizaron en los quince (15) sectores cuando el centro comercial se encontraba cerrado y sin visitantes en los pasillos.

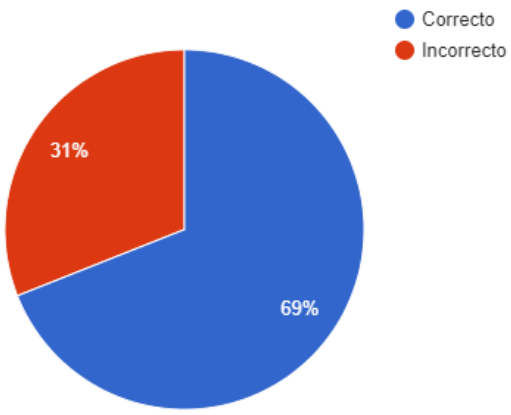

Figura 10.6: Resultado General de Geolocalización con C.C. Cerrado al Público

En la Figura 10.7 se muestra los resultados obtenidos al realizar las pruebas de geolocalización en los espacios interiores del centro comercial cuando ya había realizado su apertura al público en general.

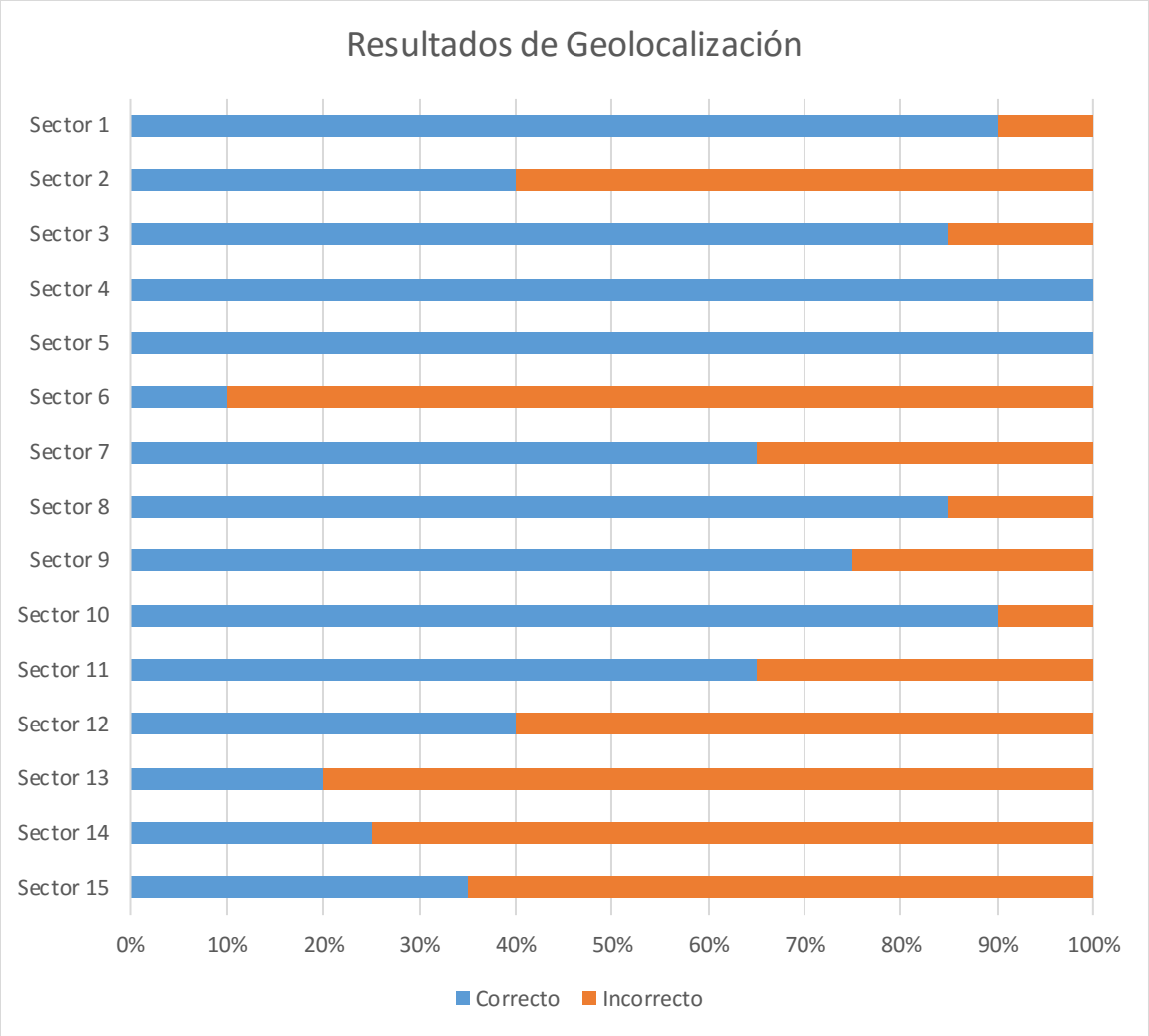

Figura 10.7: Resultados de Geolocalización con Centro Comercial Abierto al Público

En la Figura 10.8 se aprecia el porcentaje general de resultados correctos e incorrectos de las pruebas que se realizaron en los quince (15) sectores cuando el centro comercial ya había realizado la apertura al público.

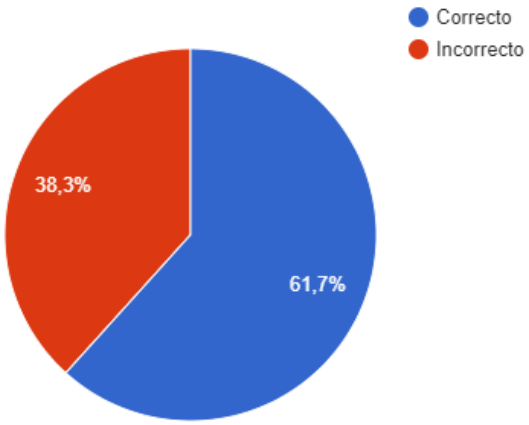

Figura 10.8: Resultado General de Geolocalización con C.C. Abierto al Público

Las pruebas arrojaron un porcentaje de acierto superior al 60% por lo que podría decirse que la implementación del método de los k-vecinos como método de geolocalización por WiFi es factible y podría funcionar en los espacios interiores de las edificaciones aunque es importante aclarar que según la experiencia que se obtuvo a través de toda la investigación y período de pruebas es que con este método usando la tecnología WiFi es muy complicado lograr una alta precisión y funciona mejor para cobertura de áreas más grandes, como lo fueron por ejemplo los sectores definidos por cada nivel del centro comercial. Se considera que los resultados podrían mejorar considerablemente si toda la funcionalidad de geolocalización se realizará con APs bien definidos y sabiendo exactamente su ubicación, incluso existen algunos modelos que se les puede modificar el *firmware* y así modificar la potencia con que envían las señales según las necesidades que se presenten por la estructura de la edificación, de esa manera sería posible tener un mayor control en la definición del fingerprinting.

11. Conclusiones y Trabajos Futuros

En los últimos años, los smartphones y tablets se han convertido en los dispositivos móviles de uso más extendido a nivel mundial y han revolucionado la forma de entender el consumo de la tecnología, lo cual ha generado un cambio en la vida de las personas y en la manera en que se desenvuelven en su día a día, en gran parte debido a las utilidades que brindan para realizar actividades de diferentes ámbitos.

Una característica importante de los dispositivos móviles, es que permiten al usuario instalar aplicaciones que aumentan aún más su potencialidad y productividad. De manera sencilla, una aplicación móvil puede definirse como una aplicación informática diseñada para ser ejecutada en dispositivos móviles para realizar diferentes tareas. Las aplicaciones móviles a su vez, utilizan generalmente servicios, librerías y APIs para su funcionamiento. Existe una gran cantidad de tecnologías que permiten el desarrollo de aplicaciones móviles, aplicaciones web que sirvan como sistemas de gestión de datos y contenidos a presentar y de APIs para la comunicación entre esos dos componentes. Según la tecnología a usar y los enfoques de desarrollo, se puede tener aplicaciones móviles nativas, basadas en web o híbridas, de lo cual se escogió el desarrollo nativo para lograr un producto más ligado al sistema operativo Android que es al que está dirigido, de mejor rendimiento, interfaz de usuario ligada a las tendencias de diseño de este SO y para poder utilizar APIs nativas que permitieron la incorporación dentro de la aplicación de ciertas funcionalidades de los dispositivos móviles.

Para las empresas y organizaciones, las aplicaciones móviles suponen un valor adicional para ellas y un salto de altura. En el caso de los centros comerciales, las ventajas que pueden brindar se refiere a un aumento de productividad y mayores oportunidades de negocio, representando un soporte muy importante en la relación entre el centro comercial y sus visitantes, ampliando la red de beneficios para ambas partes.

En el presente trabajo de investigación, se presentaron una serie de conceptos, herramientas y tecnologías que sirvieron de base sólida para la ejecución exitosa del proyecto que tenía como objetivo desarrollar una solución tecnológica para centros comerciales que permita visualizar y gestionar la información del mismo, datos de contacto, directorio de comercios y servicios que posee, mapas, promociones, eventos y un sistema de geolocalización como funcionalidad base.

La metodología usada para el proyecto y el proceso de desarrollo del mismo fue la metodología Scrum, la cual fue gestionada en el proyecto haciendo uso de la plataforma web llamada Teamwork. Scrum se caracteriza por ser muy adaptable, lo cual resultó ser un factor importante ya que existen aspectos definidos sobre esta metodología de desarrollo de software que no se adaptaron a las características y necesidades de los autores por lo que fueron modificados u omitidos. Además, cuando se llevó a cabo el proceso de evaluación y análisis de las aplicaciones a desarrollar, se utilizaron una serie de artefactos que si bien no forman parte propiamente de la metodología usada en el proyecto si son muy comunes en otras metodologías de ingeniería de software como el diagrama de casos de usos, descripción de los casos de uso, modelo relacional y el diagrama entidad-relación, los cuales fueron considerados por la gran utilidad que tienen para el equipo de desarrollo involucrado durante la fase de desarrollo. También se realizaron los diseños de las interfaces de usuario que tendría la aplicación.

El desarrollo de la solución comprende una aplicación móvil para dispositivos móviles con SO Android desarrollada en Android Studio, una aplicación web desarrollada haciendo uso del framework Laravel para la administración de datos y contenidos a ser consultados por la aplicación móvil y una API privada como mecanismo de comunicación entre ambas aplicaciones

desarrollada también con el mismo framework mencionado. Cabe destacar, que los access points que son captados dentro de las instalaciones del centro comercial también forman parte de la arquitectura general de la solución ya que con los datos que son obtenidos de ellos se procesa la funcionalidad de geolocalización en los espacios interiores por medio de la tecnología WiFi.

Para la funcionalidad de geolocalización se estudió, implemento y evaluó el algoritmo de los k-vecinos más cercanos, el cual para efectos del presente trabajo de investigación fue probado en dos (2) niveles del centro comercial Millennium Mall, en donde, un nivel se dividió en ocho (8) sectores, y el otro nivel se dividió en siete (7) sectores. Cabe destacar que los sectores en que se dividieron los niveles del mapa tienen formas de polígonos irregulares y fueron definidos por los autores sin usar alguna técnica formal, sino que, por practicidad se definían estimando sus tamaños según las características de los pasillos, buscando que todos tuviesen un tamaño parecido entre ellos y buscando adaptar dichos sectores a las necesidades de precisión del proyecto. Además, también es importante mencionar que en algunas fases del proyecto se necesitaba realizar mediciones desde el centro de los sectores, pero dicha posición centrada no se estableció calculando exactamente el centroide del polígono irregular que representa el sector sino una posición aproximada.

Adicional a la implementación de la solución, se realizaron pruebas de funcionalidad de la geolocalización en la aplicación, en donde, al analizar los resultados obtenidos, se concluye que el algoritmo de los k-vecinos más cercanos es una opción factible para implementar este tipo de funcionalidad, ya que se obtiene un mayor porcentaje de resultados correctos con respecto a los incorrectos, sin embargo, se podrían aumentar considerablemente los resultados correctos mejorando las limitaciones y condiciones en las que se realizaron las pruebas en el proyecto.

Los resultados del algoritmo se vieron afectados en gran parte por la ausencia de un sistema de red propio del centro comercial, por lo que se tuvo que trabajar con los Access Points pertenecientes a las tiendas, lo cual representó un elemento de perturbación en los resultados obtenidos.

En el desarrollo de este trabajo se comprobó cómo ciertos factores son vitales para garantizar una lectura de geolocalización precisa, con base a esta experiencia adquirida, se pueden dar unas recomendaciones para mejorar las predicciones: establecer previamente la ubicación de los AP evitando obstáculos que puedan interferir la señal de los mismos, contar con dispositivos AP de características similares en cuanto a potencia de transmisión y recepción y modificar posteriormente estos parámetros para afinar la huella de cobertura de cada dispositivo, realizar un monitoreo y control de los AP para tomar acciones ante posible fallas a desajustes de su potencia, evaluar el uso de repetidores inalámbricos de la señal en casos puntuales que impliquen un difícil acceso.

Como trabajos futuros, se propone repetir la experiencia contenida en este trabajo considerando adicionalmente, la posibilidad de que el sitio de implementación cuente con la infraestructura de red necesaria. Además, se propone extender la implementación a los espacios internos de estacionamientos y por otro lado considerar otras tecnologías de redes tipo Bluetooth y dispositivos Beacon de manera que pueda realizarse un estudio comparativo en cuanto a la precisión de la ubicación de geolocalización resultante.

Referencias

[1] A. Baz, I. Ferreira, M. Rodríguez, R. García, “Dispositivos Móviles”, Universidad de Oviedo, Ingeniería de Telecomunicación, Oviedo, España.

[2] L. Marés, “Tablets en Educación. Oportunidades y Desafíos en Políticas Uno a Uno”, Red Latinoamericana de Portales Educativos, Buenos Aires, Argentina, 2012.

[3] A. Silberschatz, P. Baer Galvin y G. Gagne, Fundamentos de Sistemas Operativos. Séptima Edición, McGraw-Hill, 2005.

[4] G. M. Ramírez Villegas, “Seguridad en Aplicaciones Móviles”, Unidad Educativa Nacional Abierta y a Distancia, Abril 2013, http://datateca.unad.edu.co/contenidos/233016/EXE_SAM/leccin_1_sistemas_operativos_moviles.html.

[5] G. Pedrozo, “Sistemas Operativos en Dispositivos Móviles”, Universidad Nacional del Nordeste, Provincia de Corrientes, Argentina, Abril 2012.

[6] M. Báez, Á. Borrego, J. Cordero et al. “Introducción a Android”, Universidad Complutense de Madrid, E.M.E. Editorial, Madrid, España.

[7] Windows Central, “Windows Phone”, http://www.windowscentral.com/windows-phone.

[8] A. Chico, “Diseño y Desarrollo de un Sistema de Posicionamiento en Interiores basado en WiFi con Tecnología Android”, Universidad Carlos III de Madrid, Madrid, España, Diciembre 2009.

[9] Open Handset Alliance, “Android”, http://www.openhandsetalliance.com/android_overview.html.

[10] Developer Android, “Platform Architecture”, https://developer.android.com/guide/platform/index.html.

[11] Source Android, “ART and Dalvik”, http://source.android.com/devices/tech/dalvik/index.html.

[12] Developer Android, “Resources of the App”, https://developer.android.com/guide/topics/resources/overview.html.

[13] Developer Android, “Introduction to Activities”, https://developer.android.com/guide/components/activities/intro-activities.html.

[14] Developer Android, “The Activity Lifecycle”, https://developer.android.com/guide/components/activities/activity-lifecycle.html.

[15] Developer Android, “Tasks and Back Stack”, https://developer.android.com/guide/components/tasks-and-back-stack.html.

[16] Developer Android, “Intents and Filterintets”, https://developer.android.com/guide/components/intents-filters.html.

[17] Developer Android, “Services”, https://developer.android.com/guide/components/services.html.

[18] Developer Android, “Content Providers”, https://developer.android.com/guide/topics/providers/content-providers.html.

[19] Developer Android, “App Manifest”, https://developer.android.com/guide/topics/manifest/manifest-intro.html.

[20] IBM Corporation, “El Desarrollo de Aplicaciones Móviles Nativas, Web o Híbridas”, Nueva York, Estados Unidos, Abril 2012.

[21] P. Rincón, “Aplicaciones Móviles Nativas con Consumo de APIs Online, Estudio comparado con Aplicaciones Web Móviles en iOS y Android y Caso Práctico ‘Native Client’ para Wordpress”, Universidad Carlos III de Madrid, Madrid, España, Julio 2012.

[22] F. Díaz, “El Modelo Cliente/Servidor”, Departamento de Informática, Universidad de Valladolid, Valladolid, España, Marzo 2006.

[23] W3C, “Guía Breve de CSS”, http://www.w3c.es/Divulgacion/GuiasBreves/HojasEstilo.

[24] Mozilla Developer Network, “CSS”, https://developer.mozilla.org/es/docs/Web/CSS.

[25] A. Tanenbaum y M. Steen, Sistemas Distribuidos Principios y Paradigmas. Segunda Edición, Pearson Prentice Hall.

[26] J. Gutiérrez, “¿Qué es un Framework Web?”, Departamento de Lenguajes y Sistemas Informáticos, Universidad de Sevilla, España, 2006.

[27] M. Piattini y A. Miguel, Fundamentos y Modelos de Base de Datos. Segunda Edición, Editorial RA-MA, Madrid, España, 1999.

[28] R. Navarro, “REST vs Web Services”, Departamento de Sistemas Informáticos y Computación, Universidad Politécnica de Valencia, Valencia, España, 2007.

[29] G. Torres, “Espacios Virtuales de Experimentación Cooperativa. Caso de Estudio: Laboratorio Virtual de Cinemática”, Centro de Investigación en Tecnologías de Información y Sistemas, Universidad Autónoma del Estado de Hidalgo, Pachuca, México, 2001.

[30] Anónimo, “Máquina Virtual Java (JVM)”, Escuela de Ciencias Básicas Tecnología e Ingeniería, Universidad Nacional Abierta y a Distancia UNAD, Colombia, 2016.

[31] Developer Android, “Android Studio”, https://developer.android.com/studio/intro/index.html.

[32] J. Alvarado, “Sistema de Control de Acceso con RFID”, Departamento de Ingeniería Eléctrica, Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, México D.F., México, 2008.

[33] M. El Yaagoubi, “Acceso a Internet vía WiFi-WiMax”, Departamento de Tecnología Electrónica, Universidad Carlos III de Madrid, Madrid, España, 2012.

[34] D. Cohen, L. Cohen, G. Faillace et al. “La Localización Utilizando WiFi (802.11b-g), Diferentes Algoritmos”, Facultad de Ingeniería, Universidad de Palermo, Buenos Aires, Argentina, 2005.

[35] P. Mulas, A. González, R. Rivera, “Localización de Dispositivos Móviles en Interiores usando Redes Wireless”, Facultad de Informática, Universidad Complutense de Madrid, Madrid, España, 2007.

[36] P. Díaz, E. Alvarado, “Desarrollo de Soluciones Web y Móvil para la Integración de Marketing de Proximidad y Gestión Publicitaria con Dispositivos Beacon como Tecnología Base”, Facultad de Ciencias, Universidad Central de Venezuela, Caracas, Venezuela, 2016.

[37] A. Peralta, “Metodología SCRUM”, Cátedra de Ingeniería de Software, Facultad de Ingeniería de la Universidad ORT Uruguay, Montevideo, Uruguay, 2003.