UNIVERSIDAD AUTÓNOMA DE CIUDAD JUÁREZ

Instituto de Ingeniería y Tecnología

Departamento de Ingeniería Eléctrica y Computación

DESARROLLO DE PROTOTIPO DE UNA APLICACIÓN MÓVIL PARA MAPEO DE BACHES Y PROBLEMAS VIALES EN LA CIUDAD

Reporte Técnico de Investigación presentado por:

Antonio González López 131704

Requisito para la obtención del título de:

INGENIERO EN SISTEMAS COMPUTACIONALES

M. I. Arnulfo Castro Vazquez

Ciudad Juárez, Chihuahua 21 de noviembre de 2019

Ciudad Juárez, Chihuahua, a 8 de Noviembre de 2019

Asunto: Liberación de Asesoría

Mtro. Ismael Canales Valdiviezo
Jefe del Departamento de Ingeniería
Eléctrica y Computación
Presente.-

Por medio de la presente me permito comunicarle que, después de haber realizado las asesorías correspondientes al reporte técnico Desarrollo de prototipo de una aplicación móvil para el mapeo de baches y problemas viales en la ciudad, del alumno Antonio González López, de la Licenciatura en Ingeniería en Sistemas Computacionales, considero que lo ha concluido satisfactoriamente, por lo que puede continuar con los trámites de titulación intracurricular.

Sin más por el momento, reciba un cordial saludo.

Atentamente

Arnulfo Castro Vázquez
Profesor Investigador IIT

UACJ
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA Y COMPUTACIÓN

Ccp:
Coordinador del Programa de Sistemas Computacionales
Antonio González López
Archivo

II

Ciudad Juárez, Chihuahua, a 8 de Noviembre de 2019

Asunto: Autorización de publicación

C. Antonio González López

Presente.-

En virtud de que cumple satisfactoriamente los requisitos solicitados, informo a usted que se autoriza la impresión del documento de Desarrollo de prototipo de una aplicación móvil para el mapeo de baches y problemas viales en la ciudad, para presentar los resultados del proyecto de titulación con el propósito de obtener el título de Licenciado en Ingeniería en Sistemas Computacionales.

Sin otro particular, reciba un cordial saludo.

Mtra. Ivonne Haydee Robledo Portillo

Profesor Titular de Seminario de Titulación II

Declaración de Originalidad

Nosotros/Yo, Antonio González López, , declaramos que el material contenido en esta publicación fue elaborado con la revisión de los documentos que se mencionan en el capítulo de Bibliografía, y que la solución obtenida es original y no ha sido copiada de ninguna otra fuente, ni ha sido usada para obtener otro título o reconocimiento en otra institución de educación superior.

Antonio González López

Agradecimientos

Le agradezco a las personas que me brindaron ayuda cuando las dudas y preguntas me invadían. Le agradezco a la maestra Ivonne Robledo Portillo por la ayuda que me brindó y las palabras de aliento que compartió conmigo, ella fue la primer maestra que me dio clase en la carrera y también fue la última. Agradezco al profesor Arnulfo Castro por ser mi asesor y guiarme en la elaboración de este proyecto.

Dedicatoria

A mi familia y a Dios.

Índice general

Índice de figuras

Resumen

Este proyecto tiene como objetivo proponer el diseño de un sistema en una aplicación para el registro de baches y problemas viales nutrida por la información que registran los usuarios y conductores que transitan en las calles y es de su conocimiento la ubicación de los desperfectos, con el fin de proporcionar ayuda tanto a los conductores como a los peatones y ciclistas para que ellos puedan evitar o disminuir las probabilidades de tener un percance por algún motivo relacionado con daños en el pavimento, ya sea por la oscuridad que se genera en ciertas zonas de la ciudad cuando cae la noche o los obstáculos que inesperados como los trabajos de mantenimiento que se realizan o fugas de agua tanto negras como potables, así como el conocimiento de la falta de conciencia por el mal diseño de áreas necesarias para personas con problemas de movilidad. Estas razones exigen que sea importante recopilar la información sobre las malas condiciones de las calles y sectores de la ciudad. Mediante la utilización de la localización del dispositivo móvil (donde se instalará la aplicación) y la captura de una imagen del desperfecto para corroborar y validar la existencia de este. Cuando el usuario se encuentre en la zona que desea registrar y defina el tipo de problema con el que tiene contacto la información que genere se almacenará en la base de datos junto con la ubicación por medio de la longitud y latitud donde se encuentre el usuario con su dispositivo, esta localización con el problema será compartida a los demás usuarios que se encuentren registrados mediante el mapa que contendrá de manera general las agrupaciones de los daños, brindando así una vista general de las secciones de la ciudad que cuentan con más problemas.

Palabras clave: Baches, registro, pavimento, ubicación, aplicación.

Abstract:

This project aims to propose the design of a system in an application for the registration of potholes and road problems nourished by the information registered by users and drivers who travel on the streets and it is known to them the location of the damage, with the in order to provide assistance to both drivers and pedestrians and cyclists so that they can avoid or decrease the chances of having a mishap for some reason related to pavement damage, either because of the darkness that is generated in certain areas of the city when night falls or obstacles that unexpected as maintenance work is performed or water leaks both black and drinkable, as well as the knowledge of the lack of awareness due to the poor design of areas necessary for people with mobility problems. These reasons require that it is important to gather information about the poor conditions of the streets and sectors of the city. By using the location of the mobile device (where the application will be installed) and capturing an image of the damage to corroborate and validate its existence. When the user is in the area that he wishes to register and defines the type of problem with which he has contact, the information he generates will be stored in the database along with the location by means of the longitude and latitude where the user is located with your device, this location with the problem will be shared to the other users who are registered through the map that will generally contain the groupings of the damages, thus providing an overview of the sections of the city that have more problems.

Keywords: Potholes, registration, pavement, location, application.

Introducción

Las grietas en el pavimento, las tapas de registros de drenaje removidas, los trabajos de reparación de tuberías de agua potable, las deformidades en el pavimento generadas por la circulación de vehículos de carga pesada que generan vibraciones al transitar e incluso cúmulos de tierra, basura y suciedad son algunos de los elementos que alteran el flujo vehicular de las calles que funcionan como sistemas de comunicación vial entre las ciudades, generando confusión, frustración, irritación y enojo a los conductores. Bajo esta situación el automovilista puede llegar a tomar decisiones que podría ser perjudicial para él mismo y los demás. En la Figura 1.1 se muestran las vibraciones generadas por las deformidades en un vehículo de la marca Nissan Modelo Sentra 2010.

En países del primer mundo que cuentan con un desarrollo vial mas confiable y estructurado existe una gran utilización de tecnologías para la supervisión, control y administración de las vías de pavimentación y transporte terrestre. Distribuidas en la extensión del camino se utilizan cámaras de vigilancia con las que es viable examinar los elementos que puedan alterar la movilidad y circulación de los coches, de igual forma existe el uso de sensores con los que es posible identificar de manera mas precisa e incluso predecir la consecuencia y el impacto de las condiciones ambientales sobre los caminos y las carreteras [1].

En países con un desarrollo mas lento o tardío en el desarrollo vial, la supervisión y control se descartan de la organización ya que el mantenimiento y administración de la infraestructura del pavimento se convierten en un desafío a tal grado que la utilización de

tecnología es mínima o casi nula, de modo que se minimizan las opciones y oportunidades de contar con una atención pronta y solución rápida a los problemas que perjudican el flujo vehicular.

El problema principal que resulta de la mínima supervisión y control en la estructura del pavimento y las condiciones del entorno de este mismo son la falta de conocimiento de los daños por parte de los conductores que transitan por estos caminos. Basado en esto se busca brindar una experiencia de conducción preventiva a través de avisos oportunos sobre cualquier defecto estructural que exista en el trayecto vial en la cual el usuario se encuentre transitando mediante datos adquiridos anteriormente y siendo mostrados en tiempo real.

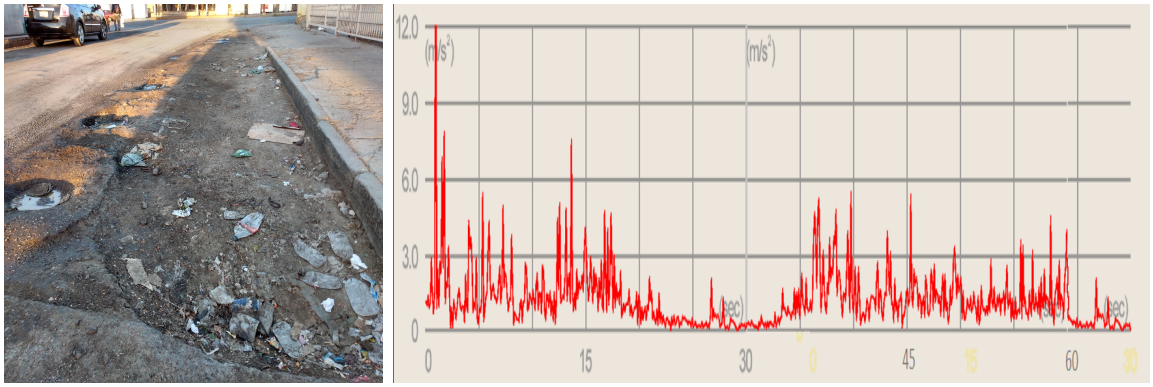

Figura 1: Ejemplo de pavimento en mal estado donde al transitar se genera una vibración en el vehículo.

Como inicio en el primer capítulo se describe el entorno del problema donde se busca plantear de manera rápida y concisa la investigación previa que se realizo para generar los datos necesarios para el desarrollo del proyecto.

En el segundo capítulo se muestra la información que fue necesaria investigar creando bases solidas para el desarrollo de todo el prototipo de la aplicación, donde se puede observar que la búsqueda de la información también auxilia para entender conceptos que se necesitaran a lo largo de los demás capítulos.

El tercer capítulo trata del desarrollo del proyecto como tal. Paso a paso el seguimiento que se tuvo que realizar para la creación y alcance del prototipo.

El cuarto capítulo abarca el resultado que se obtuvo del proyecto, siendo este bueno o malo los errores que se tuvieron que afrontar y resolver para cumplir con el desarrollo y como estos resultados se interpretaron.

El quinto capítulo comprende de la conclusión que se da al proyecto, si este cumple con la meta, y los trabajos futuros en los que se estará desarrollando.

Capítulo 1
Planteamiento del Problema

En este primer capítulo se busca plantear de manera concisa y detallada la investigación que se realizó para la obtención de los datos necesarios para el desarrollo de este proyecto. Se abordarán temas que hacen referencia al "¿Por qué?, ¿Quién?, ¿Cómo?" se concibió el reconocimiento del problema y cuales fueron las ideas generales que llevaron a la búsqueda de este y también una inspección de los antecedentes más cercanos o proyectos que tuvieron gran relevancia para el desarrollo del proyecto, así como la definición del problema y el planteamiento del objetivo central en el que se desarrolló este proyecto y su justificación.

1.1. Antecedentes

El desarrollo de programas auxiliares para los conductores en la actualidad se basan en hardware y software que llega a ser complicado para el usuario, difícil de obtener y de utilizar tales como lo son los acelerómetros, sensores, cámaras, Sistema de Posicionamiento Global (en inglés, GPS; *Global Positioning System*), Inteligencia Artificial (en inglés, AI; *Artificial Intelligence*).

Algunos vehículos como el Tesla modelo 3, modelo S y modelo X cuentan con una conducción autónoma, esta funciona por medio de sensores avanzados que tienen 8 cámaras que ofrecen una cobertura y visión de 360 grados, doce sensores ultrasónicos que complementan

esta visión, lo que permite la detección de objetos sólidos y blandos, y un radar delantero que le permite ver a través de lluvia intensa, neblina, polvo como se muestra en la Figura 1.2 [2].

Figura 1.1: Vehículo de la compañía Tesla.

El vehículo autónomo de la compañía Google (*Waymo*) tiene incorporados sensores, cámaras y sistemas de cómputo que en conjunto hacen posible que el auto no necesite conductor ya que es suficientemente autónomo para poder tomar decisiones [3]. Pero estos dos vehículos basan su integridad en la respuesta rápida que del auto y aunque sean completamente seguros sin la intervención interior y exterior del hombre y el ambiente, esta llega a ser su gran desventaja, ya que la coexistencia con el hombre y su entorno es cambiante e impredecible. La utilización de equipos hardware y software como los sensores, cámaras y acelerómetros llegan a ser costosos y pueden estar restringidos a un solo tipo de vehículo al cual fue diseñado [4].

Por otro lado específicamente en el área de la inteligencia artificial se trabaja con redes neuronales para lo cual es necesario extraer información de los conductores que van recopilando cuando conducen como lo explica George Hotz en la empresa *comma.ai* donde obtienen la información de conducción de los usuarios que utilizan la aplicación (*chffr - dash cam by comma.ai*) y el dispositivo que fabrican (*EON Dashcam DevKit*) obtienen un modelo por medio de la red neuronal que es alimentada por la comunidad [5]. Con ello se construye un

análisis sobre el estilo y comportamiento de conducción de los usuarios que posteriormente es utilizada para entrenar vehículos autónomos [6].

La utilización de sensores puede ser altamente beneficioso si estos son pequeños dispositivos con una batería de larga duración que puedan llevarse al interior del vehículo para realizar el monitoreo necesario, pero esto no sería en tiempo real, como se explica en [7].

La integridad del pasajero y su vehículo se basa en la respuesta rápida del conductor que tiene el control total. Si este mismo prevé con anticipación los cambios bruscos en el entorno del suelo obtendrá una mejor respuesta al transitar por la zona

1.2. Definición del problema

La ignorancia de las condiciones viales en el entorno del pavimento que tienen los conductores puede llegar al punto de ser peligroso como se muestra en la Figura 1.3. Ya sea los conductores que se ven forzados a manejar de manera imprevista por esas vías, como para los que ese entorno ya es algo común y cotidiano [8]. El desconocimiento de la condición del pavimento sobre lugares inexplorados hace que tiendan a aumentar las probabilidades para el conductor y pasajeros de tener un accidente.

Figura 1.2: Lluvias causaron daños mayores en una camioneta propiedad de una ciudadana que transitaba por la calle.

1.3. Objetivos

Desarrollar un prototipo de una aplicación móvil que permita al usuario registrar y compartir a la sociedad los daños y desperfectos que se encuentran en las calles y en el pavimento de la ciudad, por medio de imágenes y clasificaciones de los problemas indicando los lapsos de tiempo que han acumulado estas fallas en las calles a partir de que se realiza el registro.

- Desarrollar un módulo que permita que los usuarios puedan registrarse para reportar por ellos mismos posibles defectos en sus recorridos habituales y visualizar el estatus generado a partir de la recopilación de todos los usuarios.
- Desarrollar un módulo de administración para dar mantenimiento a los registros que han realizado los usuarios, verificando su autenticidad.

1.4. Justificación

La comunicación vial se ve afectada en gran medida por el deterioro de la infraestructura del pavimento, sin una herramienta informativa que permita registrar los daños y compartirlos a la comunidad que emplea estas rutas de comunicación, originan una presión hacia los conductores que lo que provoca que circulen por lugares nuevos o en dado caso pretendan intentar adivinar donde se encuentra la zona del desperfecto.

Es extremadamente difícil que el usuario por primera vez sea capaz de no sufrir algún inconveniente por las calles que están dañadas y que conozca la condición del pavimento en toda la ciudad. Sin embargo, se vuelve mucho mas probable obtener la información actualizada de los daños del camino si se reúnen los conocimientos de más usuarios con experiencia en transitar dichas calles que son desconocidas para otros usuarios.

Con base a los problemas encontrados, se desarrollará este proyecto como el prototipo

de aplicación móvil, este se enfocará en beneficiar a la comunidad automovilista y peatonal con avisos oportunos sobre el pavimento mediante la información que será recabada por otros usuarios que tienen conocimiento más preciso de los daños en el camino. Dicho esto, la aplicación permitirá auxiliar al conductor facilitándole la conducción por el trayecto sea cual sea el que se tome así disminuyendo las probabilidades de tener un percance y previendo grandes pérdidas de tiempo, dinero y también en la pérdida de la vida del automóvil.

1.5. Alcances y limitaciones

Alcances:

- El tiempo de las fallas que se manejará para el proyecto será desde que el usuario realiza el registro del daño.
- Se creará una agrupación de defectos entre las zonas que tienen registros.
- Para validar el daño que se desea registrar se deberá capturar una imagen del mismo.
- Se tomará la ubicación del usuario cuando este registre el daño.

Limitaciones:

- La aplicación no podrá definir automáticamente que es un daño y que si lo es por medio de la imagen.
- Las consultas realizadas simultáneamente por los usuarios deberá ser menor a 3.
- El número de consultas y registros no deberá exceder las 15000 solicitudes al mes, por motivos de restricciones para que estas continúen siendo gratuitas.
- El uso del sistema operativo como mínimo requerido deberá ser Android 4.4.

- El tiempo de respuesta por parte de la aplicación dependerá de la conexión de internet.

Capítulo 2
Marco teórico

Este capítulo se centrará en proporcionar de manera detallada el conocimiento básico acerca de los elementos, conceptos y terminología que se requieren para comprender y tener una visión más clara sobre los puntos esenciales de este proyecto con la finalidad de que el lector pueda congeniar con los datos que serán expuestos en capítulos posteriores.

El marco teórico se conformará primero por las definiciones de todos los tecnicismos, aplicaciones y métodos necesarios para realizar el software, así como datos importantes correspondientes al tópico que se solucionará.

Anteriormente se describirá más a detalle las herramientas, dispositivos, plataforma de desarrollo y funcionamiento básico de un móvil con lo cual se pretende mostrar el enfoque que se le dará al proyecto.

El presente trabajo analiza la construcción y desarrollo de una aplicación que auxilie a los conductores. Es necesario plantear algunos parámetros que sirvan como base para apoyar la lectura. Para empezar, se explicará el concepto de aplicación, aplicaciones móviles, sensores, acelerómetro, giroscopio, sistema de posicionamiento global (en inglés, GPS; *Global Positioning System*), base de datos, servidor, computación en la nube, sistema operativo android, android studio, entorno de trabajo (en inglés; *Framework*), HTML5, motor de aplicaciones de Google (en inglés; *Google App Engine*), inteligencia artificial, diagramas UML.

2.1. Aplicación

Una aplicación es sencillamente un programa informático diseñado con el objetivo de facilitar o llevar a cabo una tarea en un dispositivo informático. Es importante señalar que, aunque todas las aplicaciones son programas, no todos son aplicaciones. Esta definición solo aplica al software que fue diseñado con una finalidad concreta, lo que descarta como aplicación al software como sistema operativo, dado que su propósito es general.

2.1.1. Aplicaciones móviles

Las aplicaciones que también son conocidas como apps están presentes en los teléfonos celulares desde hace tiempo. Estas aplicaciones ya se encontraban integradas en los sistemas operativos de los dispositivos Nokia o Blackberry. Los teléfonos celulares o teléfonos móviles de ese momento contaban con aplicaciones que estaban enfocadas para mejorar la productividad personal, se trataban de alarmas, calendarios, calculadoras, algunos videojuegos, tonos de llamada y agendas.

En la actualidad se pueden encontrar una variedad de aplicaciones, que van desde lo profesional y educativas hasta de acceso rápido a servicios u ocio. Básicamente, una aplicación de este tipo es un programa o software desarrollado específicamente para teléfonos inteligentes de la actualidad (en ingles, *smartphones*).

Surgió un gran cambio con la introducción del iPhone en el mercado, ya que basándose en este mismo se crearon nuevos modelos que hicieron de las aplicaciones móviles algo mas beneficioso, para los desarrolladores y para las plataformas de aplicaciones, como App Store, Google Play y Windows Phone Store. En el mismo periodo, se mejoraron las herramientas de desarrollo de aplicaciones de las que disponían diseñadores y programadores, favoreciendo el desarrollo de crear una aplicación y exhibirla en cualquiera de las plataformas [9].

2.2. Sensores

Los sensores son componentes electrónicos que emulan el sistema sensorial de los seres humanos, de este modo, diferentes máquinas utilizan los sensores para interactuar con el medio a su alrededor. Existen diferentes tipos de sensores que son utilizados para interpretar la información que llega del exterior en un impulso eléctrico, esto para cualquier sensor es un estímulo que puede ser físico o químico que internamente se traduce en una magnitud eléctrica. Este impulso eléctrico usualmente pasa a una unidad de control en donde es examinado y modificado con el objetivo de generar una reacción o respuesta.

Los sensores pretenden de emular el sistema sensorial de los seres humanos, de este modo las máquinas y equipos electrónicos utilizan los sensores para interactuar con el medio que les rodea. Por ejemplo, un termómetro es un sensor de tipo térmico que proporciona información numérica sobre la temperatura del ambiente con base de una estimulación química [10].

2.3. Acelerómetro

El acelerómetro que se encuentra en un teléfono móvil es un elemento mecánico de tamaño pequeño por la nanotecnología que se aplica y este mismo se constituye de silicio.

El acelerómetro sirve para que el dispositivo (un teléfono celular) pueda percibir la orientación en la que está colocado, de esta forma el dispositivo puede saber cuándo se encuentra de manera horizontal, vertical, o colocado hacia abajo.

El acelerómetro del dispositivo móvil se constituye por dos partes, una de ellas se mueve si el dispositivo muestra una aceleración que se le este aplicando, y la otra parte se constituye de una parte fija que interpreta el voltaje que resulta del movimiento al que se vea sometido para determinar la velocidad a la que lo hace y la orientación a la que se vea sujeto.

En Figura 2.1 se puede observar como es que acelerómetros están compuestos de tres ejes

primordiales con los que pueden medir el movimiento en el espacio [11].

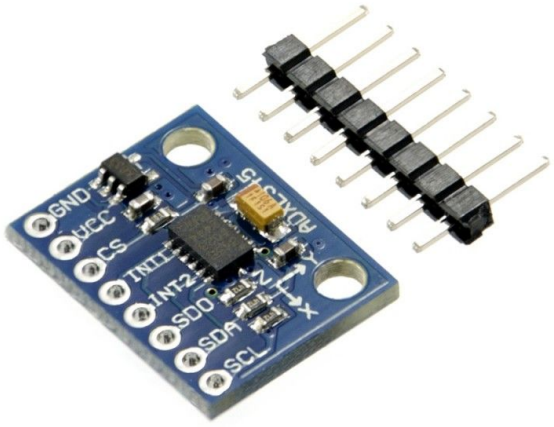

Figura 2.1: Sensor acelerómetro.

2.4. Giroscopio

El giroscopio, como el acelerómetro visto en el párrafo anterior, es también un sensor que ayuda a medir la aceleración no gravitacional, y el fin de este es el de complementar la información sobre la orientación del dispositivo que brinda el acelerómetro. Para lograr esto, se le suma una cuarta dimensión que mide la rotación o el giro. Como se muestra en la Figura 2.2 cuando alguna aplicación solicita la inclinación del dispositivo móvil, es el giroscopio el que mide los pequeños giros.

Al igual que el acelerómetro, el giroscopio está compuesto por sistemas microelectromecánicos, lo que significa que tiene nanotecnología. Se constituye de brazos en constante vibración, esta varía cuando un movimiento incurre en ella, y los cambios generados son interpretados por otro brazo [12].

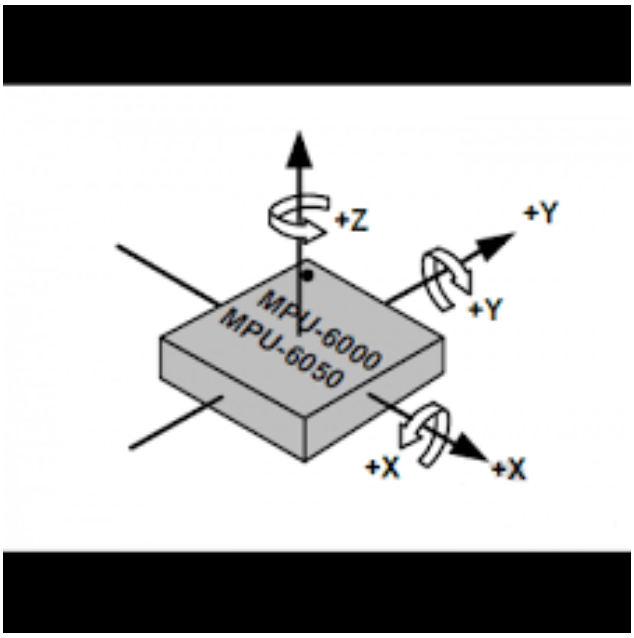

Figura 2.2: Sensor giroscopio.

2.5. Sistema de posicionamiento global

Los sensores GPS (en inglés, GPS; *Global Positioning System*) están diseñados para estar ininterrumpidamente analizando la señal que es transmitida por los satélites de GPS, emplean generalmente la señal de varios de los satélites para triangular una posición. La señal puede ser más débil en el interior de un edificio que en el exterior.

El GPS puede ser utilizado para diferentes aplicaciones, pueden ir desde indicar el camino mediante aplicaciones de navegación como para obtener la ubicación en el plano cuando se utiliza. Se trata de una función que se puede desactivar manualmente en el caso de que el usuario así lo prefiera para ahorrar batería, ya que de lo contrario estará situándose continuamente [13].

2.6. APIs de geolocalización de Google

Las API (en inglés, API; *Application Programming Interface*) permiten la comunicación e integración de los servicios que ofrece Google con otros servicios, estas permiten la comunicación, análisis o acceso a datos del usuario por ejemplo en las búsquedas, gmail, maps, etc. Las aplicaciones de terceros pueden usar estas API para hacer mas funcionales sus servicios [14].

2.7. Motor de aplicaciones de Google

Este es un servicio del tipo *Platform as a Service (PaaS)*, que permite publicar aplicaciones web en línea sin la preocupación de la infraestructura, lo cual permite un enfoque total en el desarrollo de la aplicación y un medio mas sencillo para poder correrla en la plataforma de Google.

App Engine facilita el desarrollar, mantener y mejorar la aplicación con forme sea necesario, lo que hace posible el desarrollo de aplicaciones escritas en lenguajes de programación como Java, Python, PHP y Go. Al utilizar el motor de Google la aplicación contará con un balanceador de carga y esta será atendida solo por las máquinas necesarias, para que esta tenga un buen comportamiento y sea óptima.

Además de que se cuenta con un almacenamiento continuo y rápido y es compatible con otros productos de la plataforma como para el procesamiento de archivos grandes como imágenes y videos, también se cuenta con un servicio de registros para el monitoreo en tiempo real del comportamiento de la aplicación [15].

2.8. Bases de datos

Una base de datos es un conjunto de datos que pertenecen a un mismo contexto y son almacenados con un orden sistemático para darles un uso posteriormente. Las bases de datos sirven para acceder los datos y archivos, esta no tiene función alguna si no se encuentra en un servidor. En este sentido, una biblioteca se puede considerar una base de datos compuesta en su mayoría por documentos y textos impresos en papel que están organizados y disponibles para una consulta [16].

2.9. Computación en la nube

Con el progreso en las tecnologías de información y comunicación (TIC), se han expuesto nuevas aplicaciones para Internet, en este caso, la computación en la nube o *Cloud Computing*. Esta es una herramienta automática que simplifica el realizar copias de seguridad ya que estas solo deben configurarse.

Es segura ya que utiliza encriptación de información, esto permite el acceso remoto a esta misma sea lo que la convierte en algo muy versátil, además de que ayuda a la disminución de inversiones en infraestructura, software y contratación de personal, la nube optimiza y brinda seguridad que permite la colaboración. Siguiendo esta idea, la computación en la nube es conveniente y rentable tanto para usuarios como para los proveedores [17].

2.10. Servidores

Un servidor es un ordenador que se encarga de suministrar información a una serie de clientes, sean personas con dispositivos u otros servidores que se conecten a él. Puede transmitir todo tipo de datos, desde programas informáticos, bases de datos, archivos, imágenes,

videos etc. Un servidor tiene un sistema operativo tipo server como por ejemplo Windows Server o GNU/Linux Server, este es un equipo informático que forma parte de una red y que brinda de servicios a otros equipos [18].

2.11. Apache

Apache es un servidor web http de código abierto para las plataformas Unix, Windows, Macintoch entre otras, que implementan el protocolo HTTP/1.1. Es el servidor web mas usado en el mundo con mas de 100 millones de sitios web para el año 2009. Es desarrollado y mantenido por la comunidad de usuarios (en inglés; *Apache Software Foundation*) [19].

2.12. IIS

Servidor IIS (en inglés, IIS; *Internet Information Server*) son los servicios que admiten la creación, configuración y administración de sitios web. Los servicios de IIS que provee Microsoft incluyen los protocolos NNTP (en inglés, *Network News Transport Protocol*) o el protocolo de transferencia de noticias a través de la red, FTP (en inglés, *File Transfer Protocol*). El servidor ISS es clave para gestionar la red [20].

2.13. WebHosting

El hospedaje web (en inglés, *Web Hosting*) es un servicio que se le proporciona a los usuarios de Internet para realizar almacenamiento de información de cualquier tipo ya sean imágenes, videos, o contenido que sea asequible vía web.

Definido como el espacio donde puedes almacenar tu información de tipo web, aunque esta definición sintetiza la idea el hecho de que el hospedaje o alojamiento web es realmente un

espacio en Internet para prácticamente cualquier tipo de información que el usuario almacene, sean archivos, sistemas, correos electrónicos, vídeos etc. Toda esta información esta disponible vía web [21].

2.14. Sistema operativo android

Este es un sistema operativo pensado para los teléfonos móviles, este esta basado en Linux, que es un núcleo de sistema operativo libre, gratuito y de muchas plataformas. Este sistema operativo permite programar aplicaciones en una versión de Java.

El sistema operativo brinda todas las interfaces que son necesarias para el desarrollo de aplicaciones y que estas puedan acceder a las funciones que tiene el teléfono móvil de una manera fácil y sencilla.

Debido a la sencillez y las herramientas de programación que son gratuitas se obtiene algo muy importante del sistema operativo y es el número de aplicaciones disponibles que hacen que el usuario no tenga límites [22].

2.15. Android Studio

Android Studio es la plataforma de desarrollo de android. Reemplazó a Eclipse como el IDE oficial para el desarrollo de aplicaciones para Android. Donde anteriormente se elaboraban las aplicaciones que se deseaba tener en la plataforma móvil [23].

2.16. Java

Java es un lenguaje de programación muy popular. Este esta orientado a objetos, es potente, versátil y de multiplataforma lo que permite que corra en cualquier sistema operativo. Este lenguaje se puede obtener con una gran cantidad de herramientas para trabajar de manera gratuita porque la mayor parte del código es libre.

Java tiene relación con Android porque este fue elegido como su entorno de desarrollo, es el sistema operativo móvil líder, por tanto Android es un sistema operativo y Java el lenguaje de programación utilizado para desarrollar aplicaciones en él [24].

2.17. HTML5

HTML5 es un lenguaje marcado o markup en el que se incorporan etiquetas al momento de codificar que contienen información de la estructura del texto o sobre su exhibición, este es usado para darle una estructura y presentación al contenido destinado a la web. Con esta versión se tienen nuevas posibilidades para realizar más usando menos recursos.

Es un sistema que ayuda configurar el layout o el esquema de distribución de los elementos de las páginas, este sistema no tiene mucha diferencia de su predecesor sin embargo la diferencia principal es que el código que se puede construir en HTML5 es más sofisticado en el que se notan los detalles al navegar, esto da pie a la creación de aplicaciones híbridas que consisten en desarrollar la aplicación en HTML5 y incrustarla en una ventana de navegador creada nativamente para ella [25].

2.18. Framework

En general se refiere a una estructura de software que se conforma de componentes que se pueden personalizar e intercambiar para desarrollar una aplicación. Es decir un framework se puede sopesar como una aplicación genérica que no esta completa y se puede configurar, a la que se le pueden añadir las partes finales para construir una aplicación concreta.

Los objetivos que busca un framework son el acelerar el proceso de desarrollo, la reutilización de código que ya existe y fomentar el uso de las buenas prácticas de desarrollo. Este es un conjunto de componentes como clases en Java o archivos XML que al final construyen un diseño que se puede reutilizar y que facilita de manera ágil el desarrollo de sistemas web [26].

2.19. Inteligencia artificial

La inteligencia artificial es una rama de la ciencia informática donde se busca que las máquinas realicen tareas como la mente del ser humano, con el poder de aprender y razonar. El campo de la inteligencia artificial tiene una historia larga con muchos avances, como el reconocimiento de objetos ópticos, resulta atractivo que las maquinas puedan desempeñar labores que eran exclusivas para el ser humano [27].

2.20. Diagramas UML

El diagrama UML (Lenguaje Unificado de Modelado), se compone por diferentes elementos gráficos, los cuales se combinan para crear diferentes diagramas. UML es considerado un lenguaje, así que este cuenta con reglas para combinar los elementos.

El fin que se consigue de estos diagramas es presentar diferentes perspectivas de un sistema

en particular, a las que al final se les conoce como modelo. Este modelo es una representación resumida de la realidad, este describe lo que hará el sistema sin embargo no dice como será implementado.

Existen diferentes diagramas UML, algunos serán listados a continuación:

- Diagrama de clases: Este diagrama describe la estructura de un sistema mostrando las clases del sistema, atributos, relaciones y operaciones. En la Figura 2.3 se muestra un ejemplo del diagrama de clases.

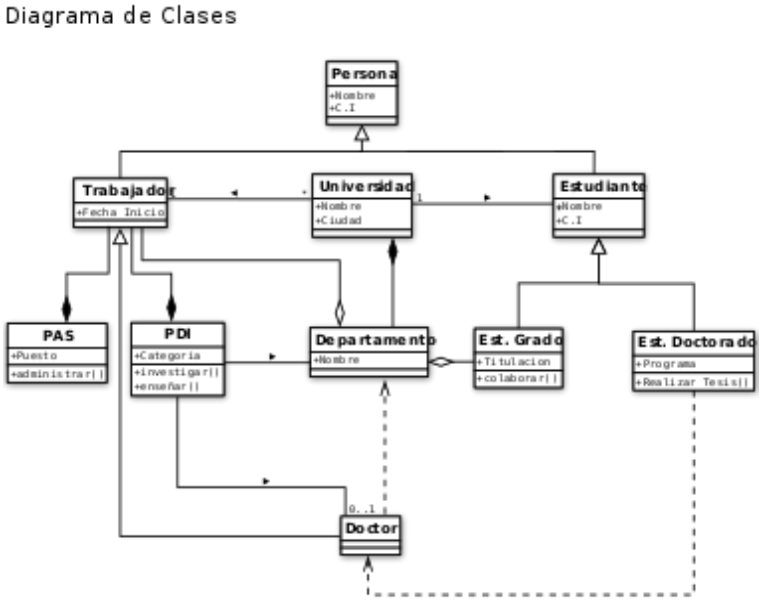

Figura 2.3: Diagrama de clases.

- Diagrama de objetos: Estos diagramas están vinculados a los diagramas de clases ya que un objeto es la instancia de una clase. Describen la estructura de un sistema en un momento preciso.

- Diagrama de casos de uso: El caso de uso es la descripción de los pasos que seguirá el sistema desde el punto de vista del usuario. En la Figura 2.4 se muestra un ejemplo del diagrama de casos de uso.

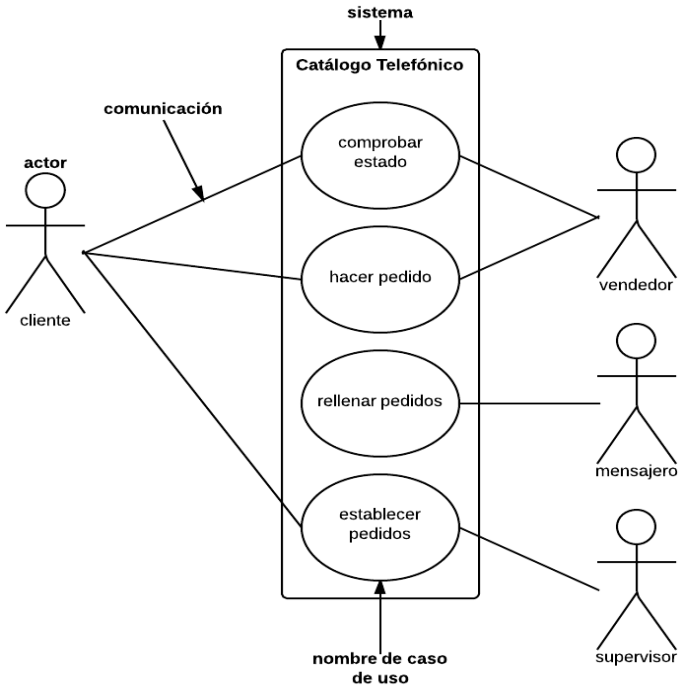

Figura 2.4: Diagrama de casos de uso.

- Diagrama de estados: El estado representa situaciones durante el lapso de aparición de un objeto, este tiene transiciones y un estado inicial y final. En la Figura 2.5 se muestra un ejemplo del diagrama de estados.

- Diagrama de secuencia: Cuando los objetos interactúan representan información y estas interacciones suceden en tiempo y se ven representadas en los cuadros de activación con e tiempo que se necesita para completar la tarea. En la Figura 2.6 se muestra un ejemplo del diagrama de secuencia.

- Diagrama de actividades: Una actividad es la representación de una operación en alguna de las clases que se encuentran en el sistema en lo que resulta como el cambio de un estado. Estos diagramas son utilizados para representar el flujo de trabajo que ocurre internamente en una operación [28]. En la Figura 2.7 se muestra un ejemplo del diagrama de actividades.

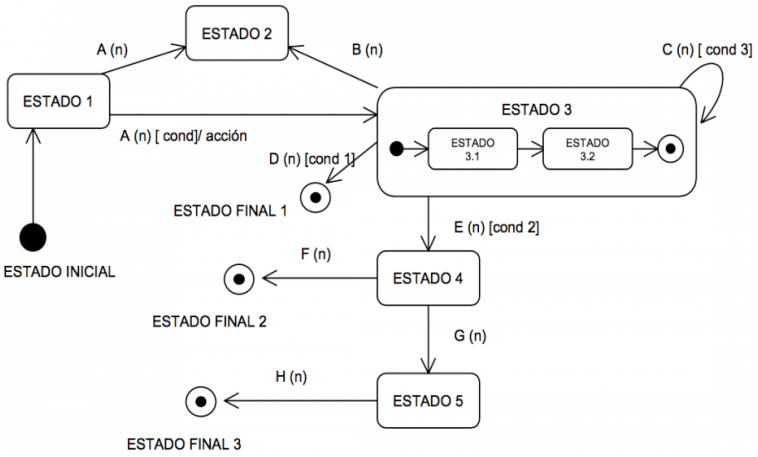

Figura 2.5: Diagrama de estados.

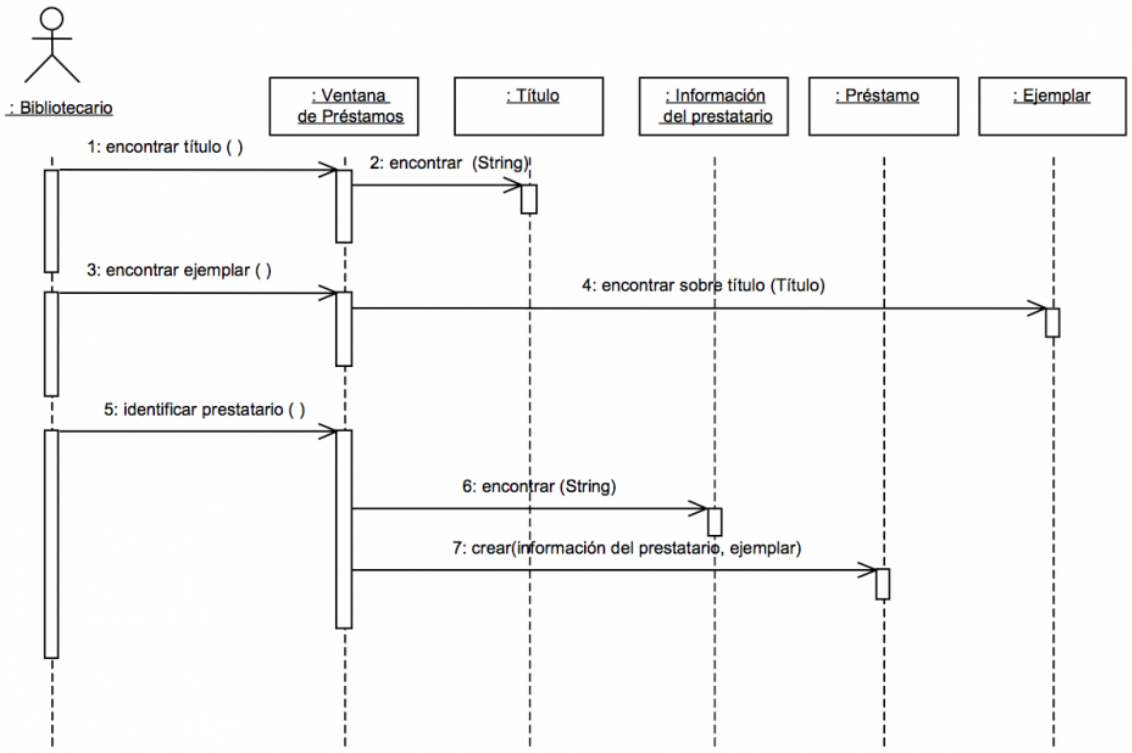

Figura 2.6: Diagrama de secuencia.

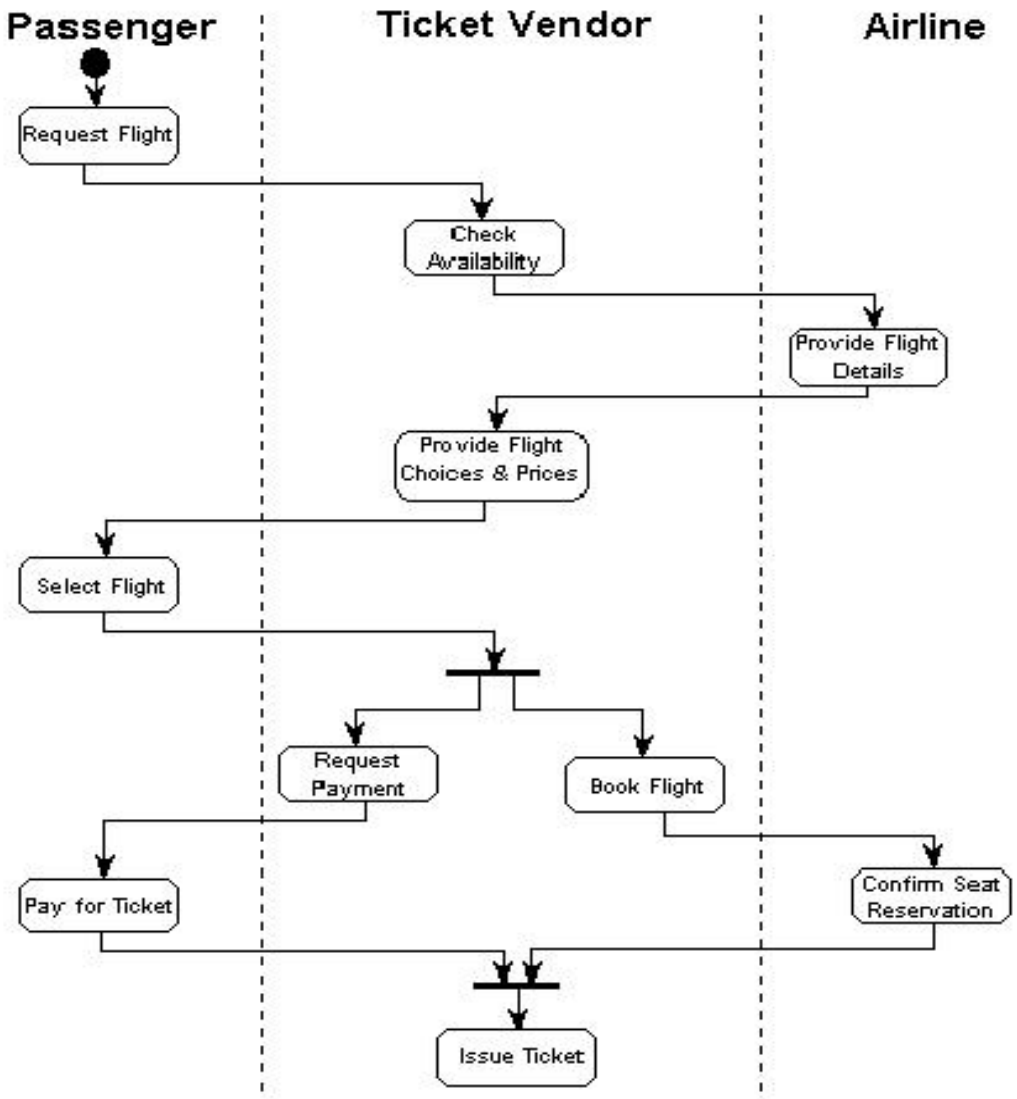

Figura 2.7: Ejemplo de un diagrama de actividades que expresa la secuencia que realiza un pasajero en una aerolínea.

Capítulo 3
Desarrollo del Proyecto

En este capítulo se mostrará de manera detallada y explicativa los pasos que se realizaron para obtener la solución previamente planteada para el problema presentado en capítulos anteriores. A partir de la investigación realizada mediante la recolección datos e información se procedió a analizar, llegando a una conclusión concisa y definitiva para obtener el resultado esperado, haciendo factible el uso de las herramientas y aplicaciones óptimas para el desarrollo de esta.

A continuación, para una mejor comprensión de la metodología propuesta, se plantea de manera visual, junto con la descripción escrita, el progreso obtenido a lo largo del proyecto como se muestra en la Figura 3.1.

Se considera que este proyecto es un desarrollo tecnológico perteneciente al área de la ingeniería de software. Contó con la necesidad de solucionar un problema que afecta de manera directa a la sociedad. Además, la innovación involucró la utilización de herramientas que son de uso común para los usuarios, tomando en cuenta los factores del proyecto se aplicaron metodologías y herramientas para ajustar de manera objetiva la solución del problema.

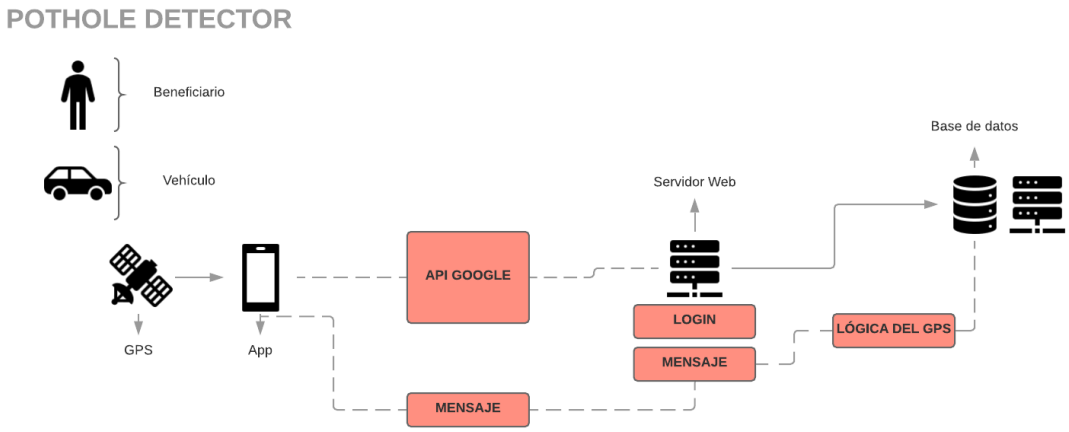

Figura 3.1: Explicación general del prototipo.

3.1. Producto propuesto

El prototipo se propuso con la finalidad de proporcionar el conocimiento a los usuarios de los problemas viales que se encuentran en las calles. La aplicación se instala en un dispositivo móvil desde donde se pueden registrar los daños.

La información de los daños del pavimento y el entorno de las mismas vías es recopilada por los usuarios que transitan por las calles día con día y que notan con gran intuición y exactitud donde se encuentran los desperfectos viales. Igualmente, el dispositivo presenta esquemas en un mapa de Google que indica qué calles son las que cuentan con más desperfectos.

La aplicación muestra cuales calles son transitables y cuales calles tienen un alto índice de daños a criterio del usuario, al igual de una clasificación de colores por agrupación y tipos de problemas para los diferentes estados de las vías de comunicación terrestre de la ciudad.

3.2. Descripción de metodología

Para el desarrollo de la aplicación se consideró la opción de crear el prototipo en el menor tiempo posible para realizar pruebas de estabilidad y gestión de datos, así como la recolección de la información para obtener la aprobación de los usuarios sin ver afectado la calidad del mismo proyecto.

Para el desarrollo del prototipo se utilizó la metodología de software en espiral, siguiendo los pasos de desarrollo ya establecidos para la recolección de información como se muestra en la Figura 3.2.

Figura 3.2: Descripción general del seguimiento utilizado en la metodología.

Fue necesario empezar creando un diseño superficial del prototipo para conocer sus principales fases. Una vez identificadas, se organizó en un diagrama de casos de uso, como el que se muestra en la Figura 3.3 donde se puede observar las opciones principales que tiene el

usuario para realizar el registro de los daños.

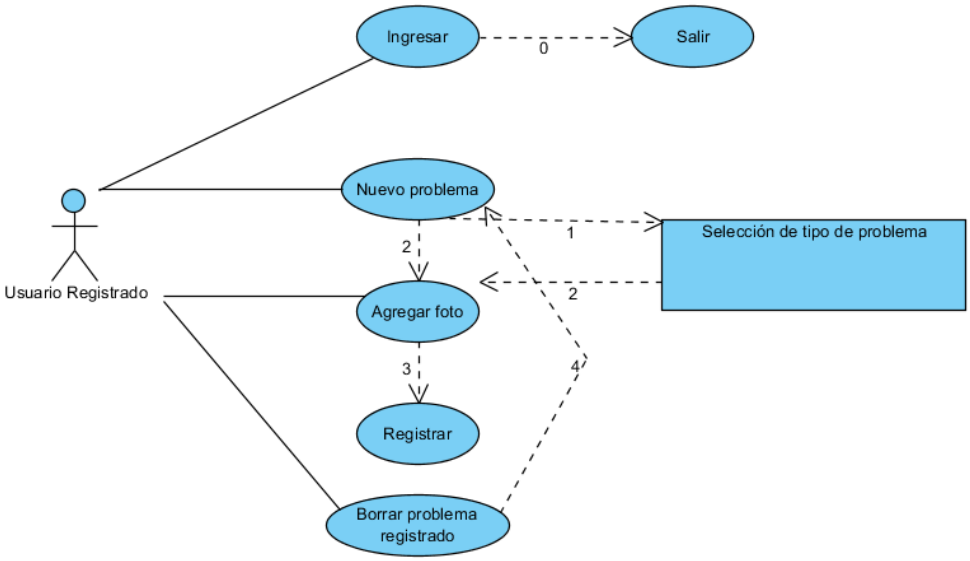

Figura 3.3: Diagrama de casos de uso inicial del usuario.

Para la etapa de desarrollo, se planeó una serie de iteraciones. Para cada iteración, se realizaron las siguientes actividades:

- Un análisis de riesgos para identificar las principales amenazas y ajustar el proyecto según las necesidades presentadas.
- Un escenario de uso para detallar el objetivo de la iteración y dar una claridad de lo que se quiere construir.
- Realizar un diagrama de clases que fue perfeccionando a la par del código que se estuvo generando para cumplir con la visión de cada iteración.
- Explicar el desarrollo realizado que brinda el servidor al usuario a través del prototipo.

3.3. Análisis y desarrollo del prototipo

3.3.1. Primera recolección de requisitos

Para el desarrollo del proyecto fue necesario planear los problemas que los usuarios pueden registrar y la forma en que lo harán. Se buscó principalmente la manera en que el usuario identifica cada problema, los factores que afectan mayormente al usuario. El principal método de recolección de datos fue directamente participando como usuario de los sistemas de comunicación vial y corroborando datos con otros usuarios.

Para esto fue necesario realizar diagramas de caso de uso para detallar los requerimientos que se tienen dentro de la aplicación. La generación de usuarios registrados para la recolección de datos y un administrador.

Dando como mejor resultado que el usuario registrado pueda ver las calles que otros usuarios registraron con señales de daños y también los marcadores grupales de colores que indican la cantidad de desperfectos que se registran en una misma vía, dando de alta nuevos daños o nuevas mejoras. Como se puede observar en la Figura 3.4.

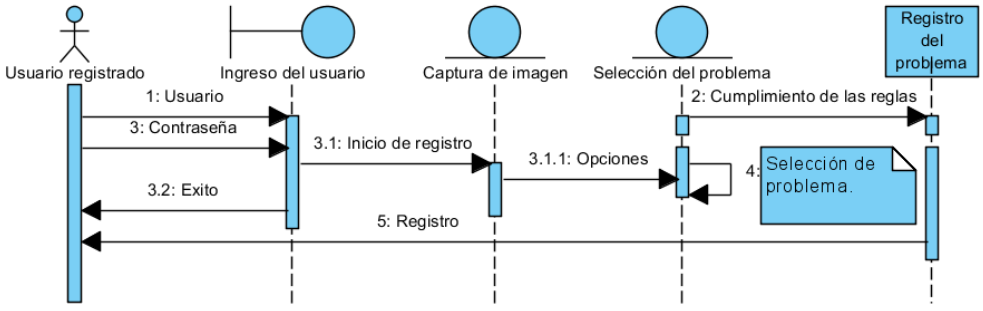

Figura 3.4: Diagrama de caso de uso con requerimientos.

3.3.2. Diseño inicial de la base de datos

Basado en estos aspectos, se diseñó el modelo inicial de la base de datos, que obedece con lo planteado para que el servidor guarde los datos que posteriormente podrán ser utilizados para que el usuario evalúe de manera personal el estado de las calles y pueda tomar una decisión al transitarla o no.

El diseño inicial se desarrolló utilizando la herramienta ER/Studio, como se puede observar en la Figura 3.5 la base de datos inicial se divide en cuatro tablas.

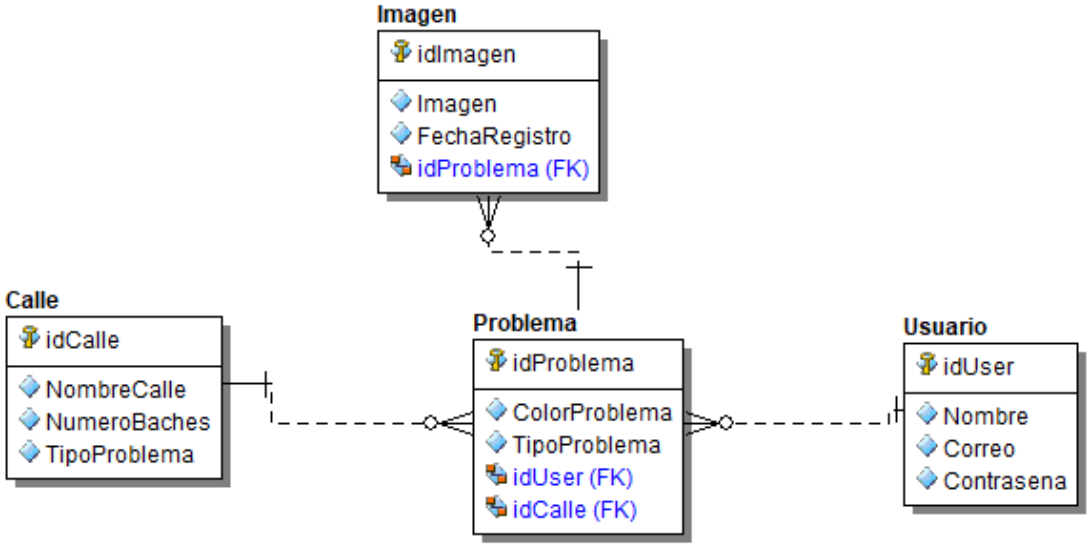

Figura 3.5: Diagrama inicial de la base de datos.

La tabla “Imagen” está asignada para corresponder a la información de la imagen que se capture y almacene por medio de un identificador único, la fecha en la que se realiza el registro y el tipo de problema que se le asigna para ser identificada. Esta directamente relacionada con la llave primaria del identificador del problema para obtener que tipo de daño fue registrado.

A la tabla “calle” está pensada para capturar el identificador que se le asignará a la calle

con el defecto, el mismo nombre de la calle, el número de baches que se han registrado y el tipo de problema al que corresponde a cada desperfecto registrado sobre la misma.

La tabla "problema" almacena los diferentes tipos de problemas, así como el color que se le asignarán en grupo, siendo dependientes del identificador de la calle con el problema y el identificador del usuario que registra el defecto. Utiliza la llave primaria del identificador del usuario y de la calle, todo esto con el fin de conocer qué calle tiene el tipo de problema y que usuario lo registro con la imagen.

La tabla "usuario" contiene los datos principales de los usuarios que realicen su registro, utilizando el nombre, el correo y una contraseña, así como un identificador único.

3.3.3. Diseño del sistema del usuario

Para la planificación de la arquitectura de la aplicación fue necesario identificar los problemas principales con los que tiene contacto el usuario y los datos a tomar en cuenta como se describe a continuación:

- Bache
- Alcantarilla sin tapa
- Zona con encharcamiento (pequeño o grande)
- Zona con encharcamiento y baches
- Pavimento en mal estado (vibraciones, desgastado)
- Trabajos de mantenimiento
- Calle sin pavimento
- Fuga de agua (potable, residuales)

- Basura
- Banqueta obstruída
- Banqueta en mal estado
- Rampa para discapacitados mal diseñada

Una vez identificados los problemas principales se prosiguió a desarrollar un diseño superficial del prototipo donde se pueda apreciar de manera visual el complemento que se agregó al modelo inicial, como se puede ver en la Figura 3.6.

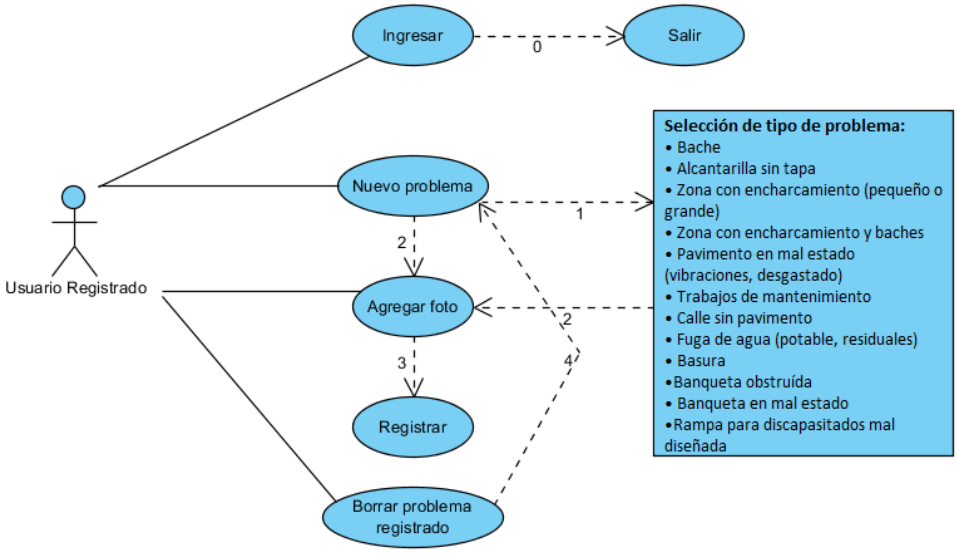

Figura 3.6: Modelo lógico inicial del usuario.

Al iniciar la aplicación se muestra una pantalla principal la cual requiere de una activación como usuario registrado o darse de alta en el registro.

En la opción de “Ingresar” es necesario cumplir con los campos previos de “Correo” y “Contraseña” como se muestra en la Figura 3.7.

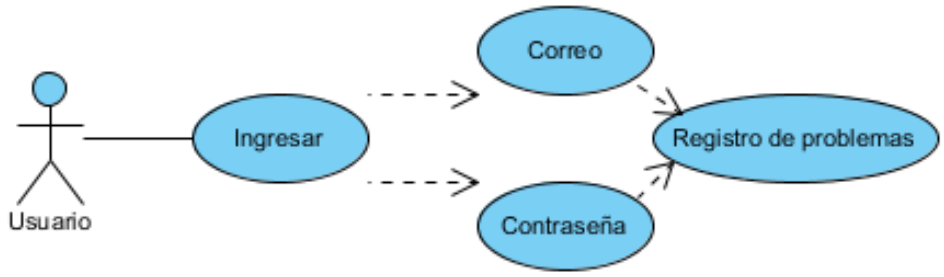

Figura 3.7: Modelo lógico inicial de permisos del usuario.

En la opción de "Registrarse" se brinda una segunda pantalla donde es necesario cumplir con los campos de "Nombre", "Correo" y "Contraseña" para poder tener acceso a registrar y ver los registros de los problemas, como se muestra en la Figura 3.8.

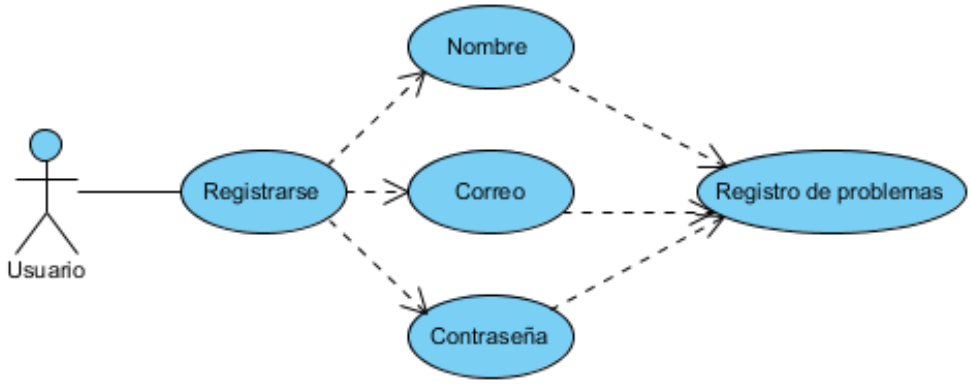

Figura 3.8: Modelo lógico inicial de validaciones del nuevo usuario.

La aplicación brinda como opción un registro de nuevos defectos y este será acompañado por medio de una imagen del desperfecto. Para prevenir datos erróneos por el usuario el registro de los daños toma la ubicación en el momento que el usuario comparte el defecto.

3.3.4. Diseño de las funciones del administrador

El administrador tiene la función de borrar un problema compartido si este no cumple con los parámetros requeridos según su consideración, así como el de borrar a un usuario (con todos los datos que este haya registrado) que se encuentre constantemente infringiendo las restricciones.

Como se muestra en la Figura 3.9 donde el administrador tiene permisos especiales, pero solo aplicables dentro de la página web en una tabla donde se organicen los problemas.

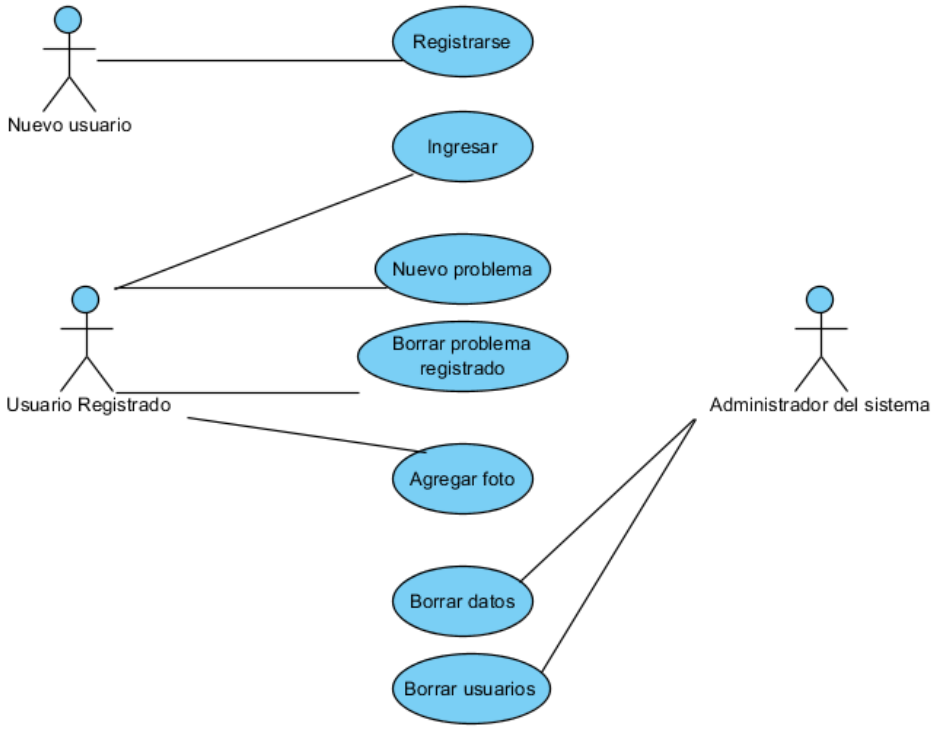

Figura 3.9: Modelo lógico inicial del administrador.

Las restricciones que se plantean son las siguientes:

- Capturar una foto con el problema y que la imagen sea acorde a la opción que señala.

- Registrar problemas inexistentes en zonas que no tienen daños.

3.3.5. Diseño de la interfaz gráfica de la aplicación para el usuario

Existe un tipo de usuario que tiene acceso a las herramientas para la consulta y creación de problemas, enviando señalizaciones en la ubicación en la que se encuentran. Cada problema puede ser eliminado por el usuario que lo creó, además de que puede ser revisado desde la pagina web desde donde se obtiene más información a detalle del problema registrado, que va desde la identificación del tipo de daño que existe hasta la fecha en la que fue creado.

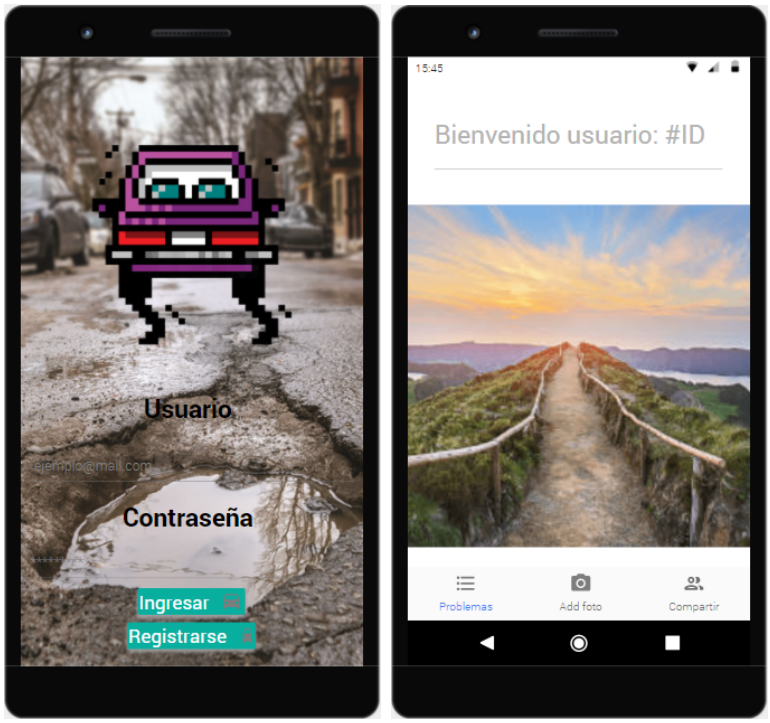

Figura 3.10: Diseño inicial de la interfaz gráfica.

Como se muestra en la Figura 3.10 donde se diseñó el primer modelo donde se cubren las necesidades reales de la aplicación.

3.3.6. Descripción de caso de uso de la interfaz gráfica del usuario

- Ingreso: El usuario registrado ingresa con sus datos necesarios “correo” y “contraseña”
- Nuevo registro: El prototipo permite crear un nuevo usuario desde la página web o desde la aplicación.
- Creación de problemas: El usuario registrado puede registrar tantos problemas encuentre en el camino mientras cumpla con las normas de seleccionar el tipo de problema, agregar una imagen del daño y proporcionar la ubicación de donde se encuentra.
- Tipos de problemas: Luego de capturar la imagen se seleccionará uno de los tipos de problemas que están predefinidos.
- Compartir: Al momento de compartir el problema se tomará la imagen como una validación de este, y se tomará la ubicación (como ya se había mencionado) para asignarle a ese punto en el mapa el problema con la imagen, como se puede ver en la Figura 3.11.

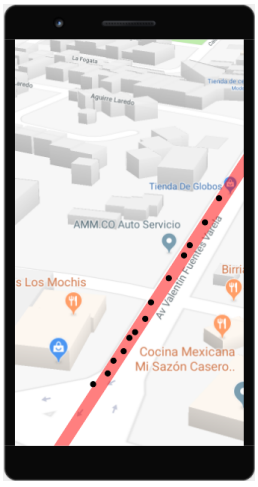

Figura 3.11: Diseño de los marcadores de la interfaz del usuario.

3.4. Segunda recolección de requisitos

Para la base de datos se tomó la decisión de que ésta fuera gestionada a través de MySQL. La comunicación de la base de datos se realizó a través de un servidor de Microsoft Azure descargado de los servicios que ofrece la universidad.

Basado en el modelo de la Figura 3.5 con las cuatro tablas iniciales, se tomó la decisión de agregar una tabla extra llamada "tipo problema", tal como se muestra en la Figura 3.12, se almacena el ID del problema y el nombre, todo esto con la finalidad de que la tabla "problemas" contenga la información de ésta misma.

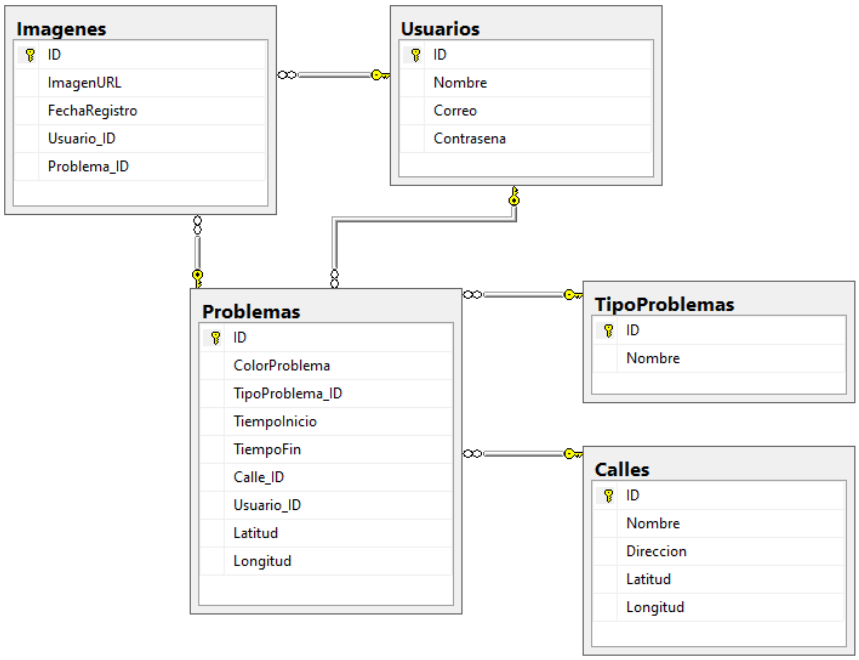

Figura 3.12: Diseño de la base de datos en su segunda versión.

Teniendo las tablas finales se procedió a crear archivo con ordenes o el guión (en inglés, *script*) para la generación de la base de datos como se muestra en la Figura 3.13. Véase el archivo completo en el Apéndice B.

```
USE [simplepc_DallasMeetup]
GO
/****** Object:  Table [simplepc_DallasMeetupUser].[Ca
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [simplepc_DallasMeetupUser].[Calles](
    [ID] [int] IDENTITY(1,1) NOT NULL,
    [Nombre] [varchar](500) NULL,
    [Direccion] [varchar](500) NULL,
    [Latitud] [varchar](500) NULL,
    [Longitud] [varchar](500) NULL,
 CONSTRAINT [PK_Calles] PRIMARY KEY CLUSTERED
(
    [ID] ASC
)WITH (PAD_INDEX = OFF, STATISTICS_NORECOMPUTE = OFF,
) ON [PRIMARY]

GO
SET ANSI_PADDING OFF
GO
/****** Object:  Table [simplepc_DallasMeetupUser].[Im
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
SET ANSI_PADDING ON
GO
CREATE TABLE [simplepc_DallasMeetupUser].[Imagenes](
    [ID] [int] IDENTITY(1,1) NOT NULL,
    [ImagenURL] [varchar](250) NULL,
    [FechaRegistro] [datetime] NULL,
    [Usuario_ID] [int] NOT NULL,
    [Problema_ID] [int] NOT NULL,
 CONSTRAINT [PK_Imagenes] PRIMARY KEY CLUSTERED
(
```

Figura 3.13: Código script para la generación de la base de datos.

Con la base de datos final se dispuso a realizar el primer módulo de la aplicación en donde se utilizó el software de *Android Studio* para programarla. Se inició con la pantalla principal codificada en el lenguaje XML que contendría dos botones.

Primero se realiza el registro de un nuevo usuario donde se despliega una segunda actividad donde se deben cubrir con los datos necesarios que son un nombre, correo y contraseña, como se muestra en la Figura 3.14.

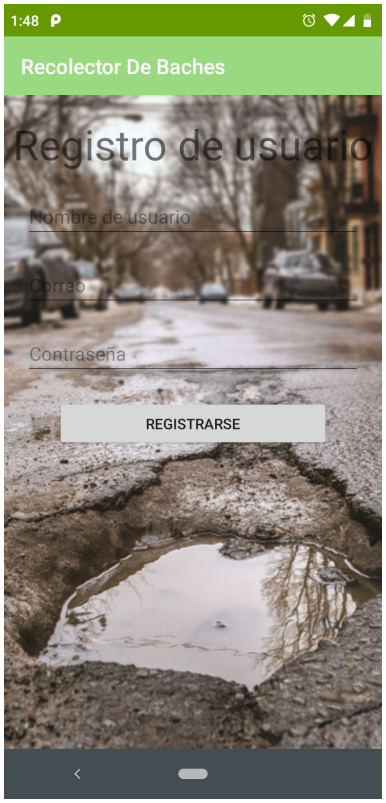

Figura 3.14: Interfaz de registro de usuario.

Como también otra actividad para ingresar como un usuario ya registrado donde es validado por dos campos iniciales en los que se debe ingresar el correo y contraseña, como se muestra en la Figura 3.15.

Después de esto se realizó la programación de la siguiente pantalla que muestra tres

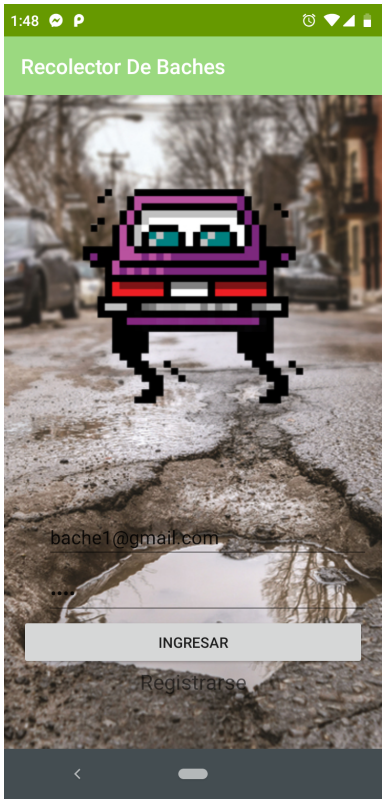

Figura 3.15: Interfaz de ingreso de usuario.

botones donde se realiza la captura de una imagen, la selección del tipo de problema que se desea registrar y el botón de compartir, como se muestra en la Figura 3.16.

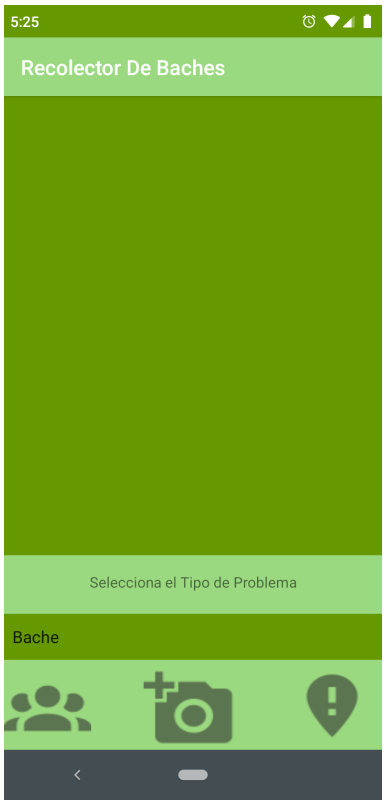

Figura 3.16: Pantalla de registro de problemas.

Se crearon las credenciales necesarias para obtener la *API* de *Google* con la que se obtendría el acceso al mapa, con el fin de realizar la tercera pantalla en la que el usuario podría ver todos los daños registrados por los usuarios. Para esto fue necesario ingresar a la página de desarrolladores de *Google* registrar el proyecto para así crear una llave con la que se podría tener acceso al mapa, como se muestra en la Figura 3.17.

La llave generada se agregó a la sección del proyecto de la aplicación llamado *AndroidManifiest.xml* de los archivos creados inicialmente al construir la aplicación, para mayor apreciación nótese el anexo del Apéndice A.

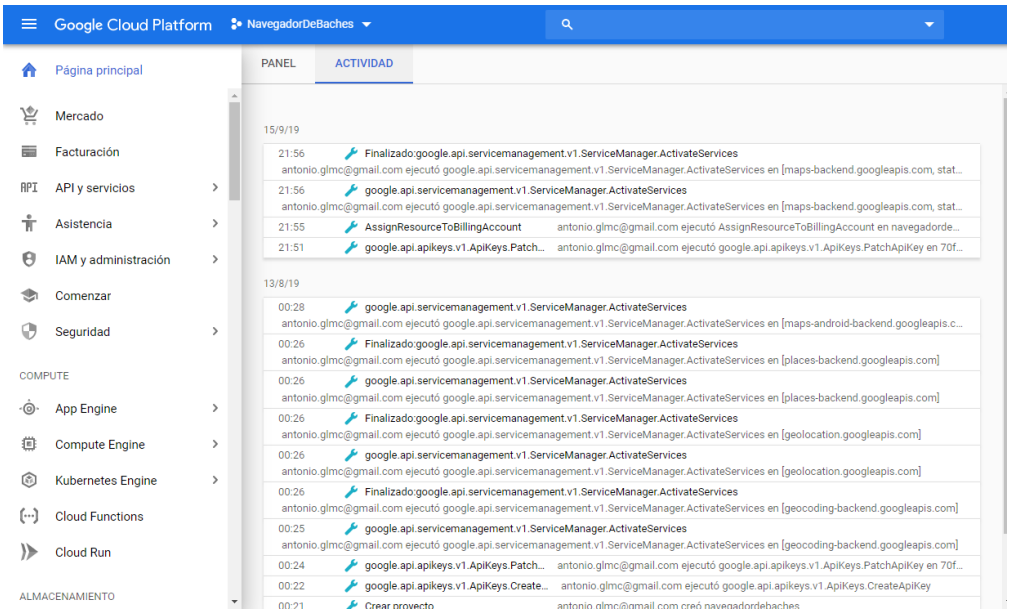

Figura 3.17: Pantalla de registro de proyecto en la plataforma de Google.

Se procedió a la programación de los botones de la segunda pantalla en la cual el botón de *tipo problema* se le agregó un *spinner* con todos los tipos de problemas por escoger. Al botón de capturar la imagen se le añadió una vista de imagen para que la fotografía pudiera ser observada por el usuario. Para el botón de compartir fue necesario la creación de un módulo en la página web para compartir los datos de forma adecuada aunado a la programación de este, ésto con la finalidad de que existiera congruencia con el registro de cada uno de los datos.

Para continuar fue necesario programar la página web que estaría alojada en el servicio web de *Microsoft Azure* para poder realizar la conexión a la base de datos como se muestra en la Figura 3.18. A esta se le agregaron dos pantallas: una para el registro, y otra para el ingreso del usuario validadas con restricciones en el correo y contraseñas.

Se creó una tercera pantalla en la que al ingresar el usuario podría acceder a los datos que ya ha registrado y visualizar las imágenes de los problemas junto con un mapa, donde se muestran las ubicaciones de los registros que se han realizado, como se puede ver en la

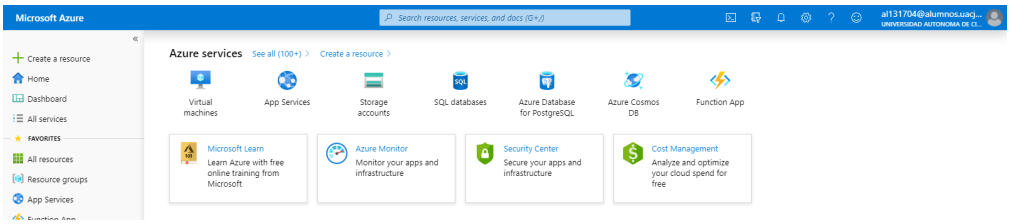

Figura 3.18: Pantalla de localización de la página web.

Figura 3.19.

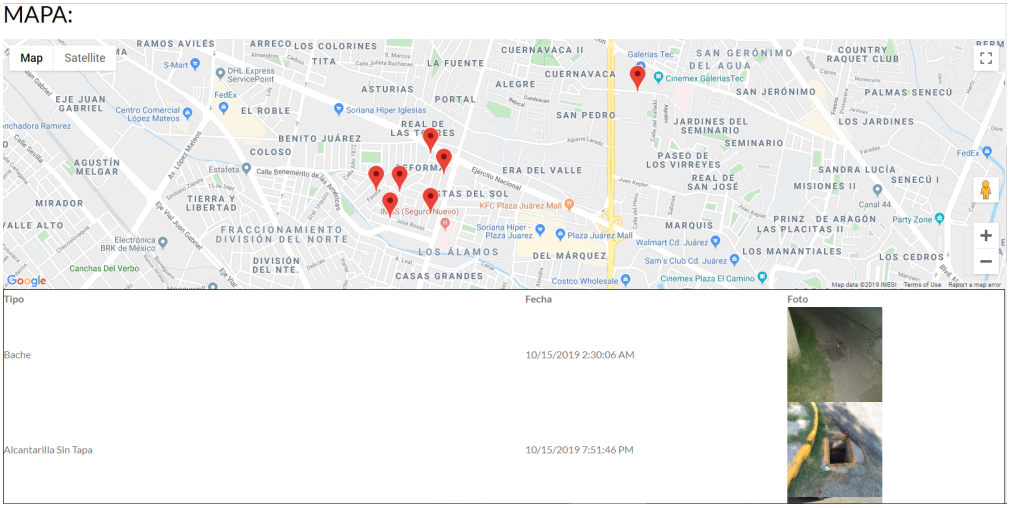

Figura 3.19: Pantalla de los problemas registrados por el usuario.

Para mantener una comunicación entre la página web y la aplicación se utilizó el envío y carga de datos consumiendo una *API* con *retrofit* para *Android*, que permite realizar peticiones *GET*, *POST* y *JSON rest api* de *JavaScript* con el objetivo de realizar un *request* y *response* obteniendo los argumentos de los parámetros registrados desde la base de datos a la *URL* de la página para ser consumida por la aplicación.

Este intercambio de datos se utilizó tanto en la estructura de la página web como en los archivos de la aplicación, como se muestra en la Figura 3.20 donde se puede observar que algunos de los nombres de los archivos tienen por nombre los atributos antes mencionados.

Para esto el primer paso es añadir las dependencias necesarias en el archivo del proyecto

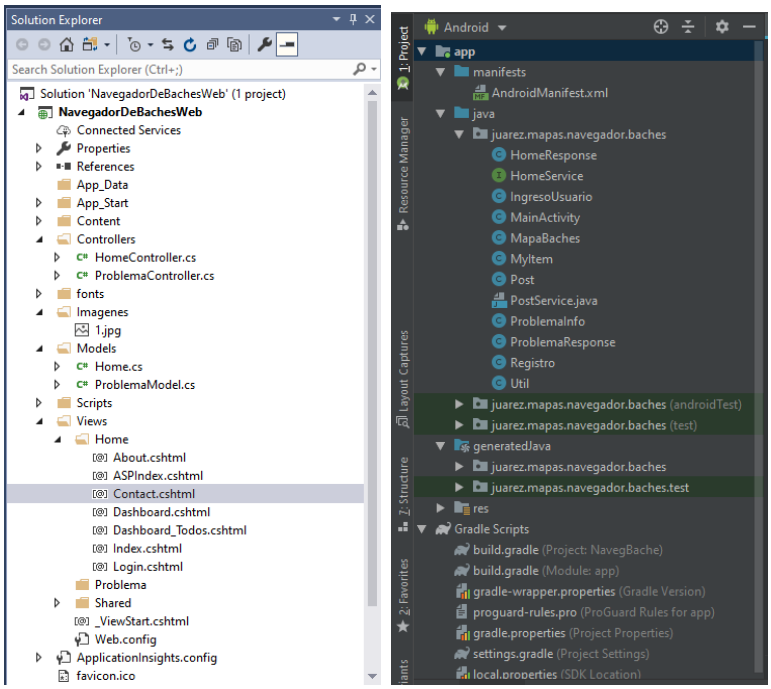

Figura 3.20: Archivos del proyecto.

llamado *Build.gradle(Module:app)* como se muestra en la Figura 3.21.

```
implementation 'com.squareup.retrofit2:retrofit:2.4.0'
implementation 'com.squareup.retrofit2:converter-gson:2.4.0'
```

Figura 3.21: Archivos del proyecto.

El archivo de la aplicación de *Android Studio* llamado *MainActivity* se encuentran las acciones que realizan los botones en la pantalla principal de la aplicación, así como la función privada que realiza la validación de los datos del usuario utilizando *retrofit* y una clase de Java llamada *HomeResponse* para el envío y/o carga de datos de una *URL* externa.

El archivo llamado *IngresoUsuario.java* (como se muestra en la Figura 3.21) contiene las opciones de selección de los diferentes tipos de problemas, así como los botones primordiales para la captura de la imagen, visualizar el mapa y el compartir el problema, aquí se define mediante la función *onCreate* que se permita acceder a la ubicación, así como el permiso para

escribir de manera externa en el almacenamiento para que la aplicación pueda compartir la ubicación a la base de datos al momento de registrar un problema.

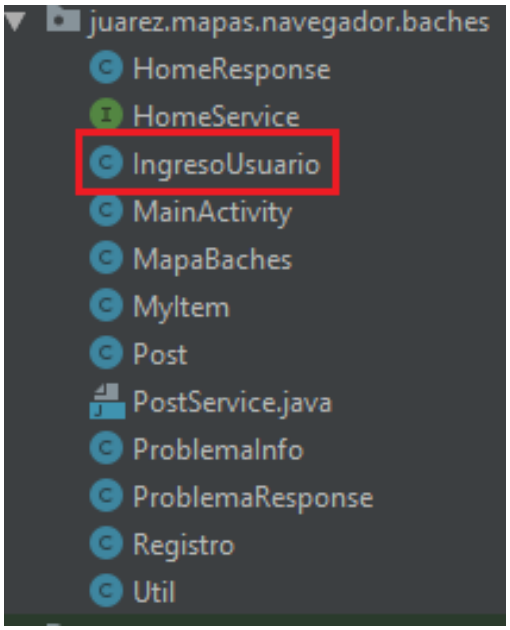

Figura 3.22: Archivos del proyecto.

Para el almacenamiento de las imágenes, y que estas no tengan conflicto al tener un nombre idéntico se optó por utilizar un *guid* para generar un número único y que éstas no produzcan un problema al ser registradas en la base de datos. Este también se agregó en el archivo *IngresoUsuario.java*.

Dentro de este archivo se encuentra la función *onActivityResult* donde se procesa la imagen como tipo *bitmap* y se crea un archivo temporal para poder ser editado por el usuario, después de esto se encuentra la función *SubirImagen* que se encarga de mandar un mensaje tipo *toast* al usuario pidiendo que capture una imagen del problema para que este pueda ser validado y donde luego de la captura de la imagen se crea un nuevo problema con el nombre único a la imagen y el identificador del usuario que registro el daño. Para más información obsérvese el Apéndice A .

En el archivo llamado *Registro.java* se encuentran tres funciones que se encargan de validar los datos del usuario al registrarse, los datos como el nombre, correo y contraseña de tipo *string*, apoyándose en *json* y *retrofit* para realizar el intercambio de datos. Además genera mensajes en la pantalla del usuario por medio del *toast* para avisar si se tuvo éxito o existió algún error al registrar al usuario. Para más información obsérvese el Apéndice A.

Capítulo 4
Resultados y Discusiones

El proyecto finaliza con el prototipo creado para el sistema operativo Android, que funciona como inicio para desarrollar más adelante una aplicación robusta y sólida para que pueda ser utilizada por los usuarios de manera práctica y sencilla. En este capítulo se verán reflejados los resultados obtenidos por el proyecto al ser puesto a prueba bajo las condiciones previamente señaladas y deseadas en los capítulos anteriores, con lo que se pretende demostrar la forma de utilización de la aplicación y la forma de recabar la información para compilar en un solo extracto los frutos obtenidos por el proyecto.

A pesar de que la idea inicial fue la creación de un prototipo de una aplicación para registrar los daños en las calles, así como llevar un control de tiempos para estos mismos. Un desarrollo importante fue la creación de la herramienta para la ubicación de los desperfectos y su validación para realizar el registro, brindando a todos los usuarios acceso a través de una plataforma móvil.

Un aspecto importante para este proyecto es el servidor donde se administra la base de datos y donde se realizan las solicitudes de los usuarios, debido a que ahí se almacena la información. Durante el desarrollo del prototipo se contó con un servicio limitado pero eficiente al momento de utilizar sus recursos.

Además, el tiempo de acción que se le da a las solicitudes de los usuarios esta totalmente ligado a las propiedades de procesamiento que ofrecen en el servidor. Actualmente, como

prototipo que no contiene usuarios reales, el número de peticiones que se realizan simultáneamente equivale a ser menor de tres. A pesar de ello, si la aplicación presentase usuarios reales que se encuentren activos y realicen solicitudes constantes de consulta y registro de datos, podría verse afectado el tiempo de respuesta que brinda el servidor a cada una de las peticiones, o incluso perder las últimas peticiones realizadas.

Tanto la herramienta de mapeo de baches como la de consulta en la página web que emplean mapas fueron desarrolladas utilizando el API de Google Maps. Esta API es gratuita, pero con restricciones de uso, se brinda un crédito que permite la solicitud del mapa 150000 veces al mes, sin embargo, superando esas vistas se cobrará dependiendo del número de solicitudes que se haya sobrepasado, es necesario ingresar una tarjeta de crédito.

La herramienta desarrollada para el mapeo de problemas viales cuenta con funciones que permite a cualquier usuario (mediante un dispositivo con sistema operativo Android 4.4 kitkat) crear y registrar problemas reales.

Dicho esto, al realizar las pruebas del prototipo se obtuvieron resultados prometedores. Principalmente al ingresar a la aplicación se encontró con un tiempo de espera para el ingreso a la aplicación, esto debido a la respuesta de la base de datos, ya que es gratuita se necesita de algunos segundos para tener acceso debido a la velocidad con la que se cuenta.

Esto se ve reflejado por el mensaje "toast" que se le envía al usuario avisándole que el usuario ingresó correctamente, como se muestra en la Figura 4.1.

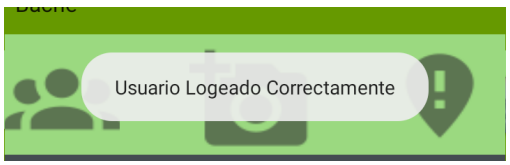

Figura 4.1: Mensaje de éxito que se le da al usuario al ingresar.

También se tuvo un buen resultado luego de probar al ingresar un usuario inexistente en la aplicación donde se observó el mensaje "toast" diciendo que existe un error al registrar al

usuario, como se muestra en la Figura 4.2.

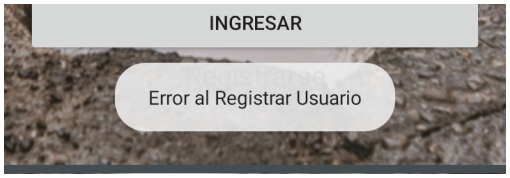

Figura 4.2: Mensaje de error que se le manda al usuario.

Se probó la actividad para crear un nuevo registro dentro de la aplicación y en el sitio web, ingresando datos idénticos y diferentes al mismo tiempo donde se pudo observar que el procesamiento para almacenar la información dentro de la aplicación y en la página web mantiene un lapso de espera de unos segundos para poder registrar ambos usuarios al mismo tiempo, siendo estos de datos diferentes.

Sin embargo al intentar registrar un usuario con los mismos datos se pudo visualizar que tanto el procesamiento de la aplicación como el del sitio web fue mas lento como se puede ver en la Figura 4.3.

Figura 4.3: Ingreso de los mismos datos en la aplicación y la página web.

Pero aun así dando una prioridad a la aplicación para realizar el registro y mandando dos mensajes al usuario desde la página web, uno de error y otro avisándole que el usuario ya esta registrado, como se puede apreciar en la Figura 4.4.

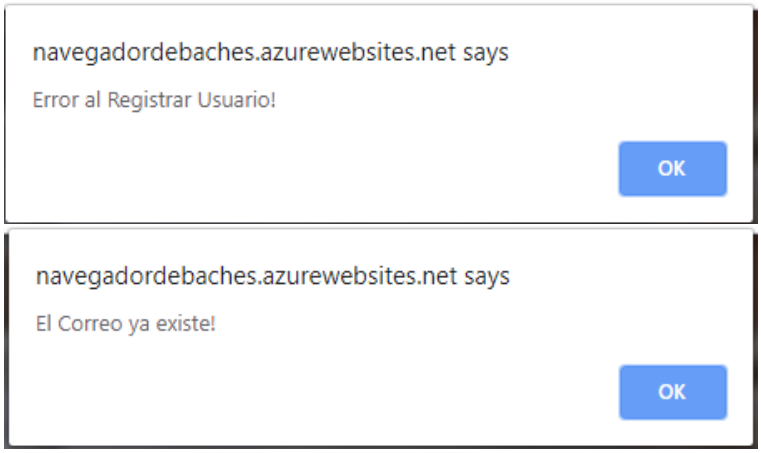

Figura 4.4: Mensajes que confirman que la aplicación tiene prioridad para el registro de usuarios.

Capítulo 5
Conclusiones

En capítulos anteriores se pudo apreciar de manera detallada la búsqueda, selección y desarrollo del prototipo de aplicación, en este capítulo se verá de forma objetiva lo que el proyecto consiguió aportar y si este mismo alcanzó el objetivo principal, el cual será explicado paso a paso con el fin de enfatizar la manera en la que este se fue cumpliendo, llegando a si a la conclusión de las metas establecidas desde el inicio del proyecto. Se tratará de los tropiezos y fallas que se presentaron a la hora de desarrollar el prototipo y el cómo se resolvieron.

5.1. Con respecto al objetivo de la investigación

Dicho esto, el objetivo central de este proyecto fue encontrar una solución óptima para el registro e identificación al problema de baches y desperfectos en la comunicación vial de la ciudad, específicamente en el pavimento de las calles y los daños con los que cuentan.

La aportación principal del proyecto de un prototipo de aplicación para el mapeo de baches en la ciudad consiste un sistema de recopilación de información alimentada por usuarios que realizan la identificación de los problemas viales con descripciones específicas, acompañado de imágenes que validan la autenticidad del problema por medio de la ubicación en la que el registro es actualizado.

En este proyecto se demuestra que el contacto en la aplicación y los servicios web que se presentaron en el tercer capitulo funcionan correctamente. Ello permite que la información capturada tenga una comunicación adecuada y esto permita cumplir con el objetivo del proyecto.

5.2. Recomendaciones para futuras investigaciones

En el futuro se planea que la aplicación pueda ser utilizada por los conductores mientras manejan, así como agregar procesamiento de imágenes para reconocer que son daños reales y que no lo son. Así como también que el numero de usuarios que registran errores en las calles sea mayor.

Bibliografía

[1] F. Martínez, L. Carlos González, and M. Ricardo Carlos, "Identifying roadway surface disruptions based on accelerometer patterns", *IEEE Latin America Transactions*, vol. 12, no. 3, pp. 455–461, May 2014.

[2] Martin Eberhard and Marc Tarpenning, "The 21 st century electric car tesla motors", *Tesla Motors*, 2006.

[3] P. E. Ross, "Robot, you can drive my car", *IEEE Spectrum*, vol. 51, no. 6, pp. 60–90, June 2014.

[4] R. Bhoraskar, N. Vankadhara, B. Raman, and P. Kulkarni, "Wolverine: Traffic and road condition estimation using smartphone sensors", in *2012 Fourth International Conference on Communication Systems and Networks (COMSNETS 2012)*, Jan 2012, pp. 1–6.

[5] Eder Santana and George Hotz, "Learning a driving simulator", *CoRR*, vol. abs/1608.01230, 2016.

[6] Gys Albertus Marthinus Meiring and Hermanus Carel Myburgh, "A review of intelligent driving style analysis systems and related artificial intelligence algorithms", *Sensors*, vol. 15, no. 12, pp. 30653–30682, 2015.

[7] C. Lin, Y. Chen, T. Huang, T. Chiu, L. Ko *, S. Liang, H. Hsieh, S. Hsu, and J. Duann, "Development of wireless brain computer interface with embedded multitask schedu-

ling and its application on real-time driver's drowsiness detection and warning", *IEEE Transactions on Biomedical Engineering*, vol. 55, no. 5, pp. 1582–1591, May 2008.

[8] Z. Kim, "Robust lane detection and tracking in challenging scenarios", *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 1, pp. 16–26, March 2008.

[9] Lisandro Nahuel Delía, Nicolás Galdamez, Pablo Thomas, and Patricia Mabel Pesado, "Un análisis experimental de tipo de aplicaciones para dispositivos móviles", in *Congreso Argentino de Ciencias de la Computación (CACIC)*, 2013, vol. 18.

[10] Ramón Pallás Areny, *Sensores y acondicionadores de señal*, Marcombo, 2004.

[11] F. Huete, D. Esteban, J.B. da Silva, M. Skouri, Manuel A. González, D. Goudjami, W. Rochadel, and Miguel A. González, "Sensor mobile, aplicación android multilingüe con fines docentes para el acceso a sensores de smartphones", 2015.

[12] Antonio SABÁN, "Cómo funciona el acelerómetro y el giroscopio de los móviles", 2016.

[13] Bradford W Parkinson, Per Enge, Penina Axelrad, and James J Spilker Jr, *Global positioning system: Theory and applications, Volume II*, American Institute of Aeronautics and Astronautics, 1996.

[14] Bojan Pejić, Aleksandar Pejić, and Zlatko Čović, "Uses of w3c's geolocation api", in *2010 11th International Symposium on Computational Intelligence and Informatics (CINTI)*. IEEE, 2010, pp. 319–322.

[15] Alexander Zahariev, "Google app engine", *Helsinki University of Technology*, pp. 1–5, 2009.

[16] Chris J Date, *Introducción a los sistemas de bases de datos*, Pearson Educación, 2001.

[17] Nelly Lisbeth Hernandez and Anderson Smith Florez-Fuentes, "Computación en la nube", *Revista Mundo Fesc*, vol. 4, no. 8, pp. 46–51, 2014.

[18] Rafael Luis Granados La Paz, *Desarrollo de aplicaciones web en el entorno servidor. IFCD0210*, IC Editorial, 2015.

[19] Yixin Diao, Neha Gandhi, Joseph L Hellerstein, Sujay Parekh, and Dawn M Tilbury, "Using mimo feedback control to enforce policies for interrelated metrics with application to the apache web server", in *NOMS 2002. IEEE/IFIP Network Operations and Management Symposium.'Management Solutions for the New Communications World'(Cat. No. 02CH37327)*. IEEE, 2002, pp. 219–234.

[20] Scot Hillier and Dan Mezick, *Programming Active Server Pages*, Microsoft Press, 1997.

[21] 2D Comunicación Interactiva, "¿qué es un hosting o servidor web? ¿para qué sirve? – 2d comunicación interactiva".

[22] Kristel Malave Polanco and José Luis Beauperthuy Taibo, "Android el sistema operativo de google para dispositivos móviles", *Revista Negotium*, , no. 19, 2011.

[23] Android Studio, "Android studio", *The Official IDE for Android*, 2017.

[24] Deitel Paul J and Deitel Harvey M, *Java: como programar*, Pearson Educación,, 2008.

[25] Gauchat Juan Diego, *El gran libro de HTML5, CSS3 y Javascript*, Marcombo, 2012.

[26] Yingxu Wang, "The theoretical framework of cognitive informatics", *International Journal of Cognitive Informatics and Natural Intelligence (IJCINI)*, vol. 1, no. 1, pp. 1–27, 2007.

[27] Wladimir Rodríguez and Postgrado en Computación, "Inteligencia artificial", *Revista Electrónica de los Estudiantes*, p. 92, 2018.

[28] Kimmel Paul, *Manual de UML*, McGraw-Hill,, 2007.

Apéndice A
Código de la aplicación

<?xml version="1.0.^encoding=ütf-8¿><manifest xmlns:android="http://schemas.android.com/apk/res

<!– The ACCESS$_{C}OARSE/FINE_{L}OCATIONpermissionsarenotrequiredtouseGoogleMapsAndroi

– >

<uses-permission android:name=.^ndroid.permission.INTERNET/><uses-permission android:name=.^

$uses-permissionandroid : name = "android.permission.WRITE_{E}XTERNAL_{S}TORAGE" / >$

<uses-permission android:name=.^ndroid.permission.ACCESS$_{F}INE_{L}OCATION" / ><

$applicationandroid : allowBackup = "true"android : icon = "@mipmap/ic_{l}auncher"android :$
$label = "@string/app_{n}ame"android : roundIcon = "@mipmap/ic_{l}auncher_{r}ound"android :$
$supportsRtl = "true"android : networkSecurityConfig = "@xml/network_{s}ecurity_{c}onfig"android :$
$theme = "@style/AppTheme" >$

<!– The API key for Google Maps-based APIs is defined as a string resource. (See the file -
es/values/google$_{m}aps_{a}pi.xml").NotethattheAPIkeyislinkedtotheencryptionkeyusedtosigntheAPK.Youn$
$- >< meta - dataandroid : name = "com.google.android.geo.API_{K}EY"android : value =$
$"@string/google_{m}aps_{k}ey" / >$

<activity android:name=".MapaBaches.^ndroid:label="@string/title$_{a}ctivity_{m}apa_{b}aches" >

<!–<intent-filter>–><!–<action android:name=.^ndroid.intent.action.MAIN/>–><!–<action
android:name=.^ndroid.intent.action.VIEW/>–>

<!–<category android:name=.ªndroid.intent.category.LAUNCHER/>–><!–</intent-filter>–
>

</activity><activity android:name=".IngresoUsuario/><activity android:name=".Registro/><activi
android:name=".MainActivity» <intent-filter><action android:name=.ªndroid.intent.action.MAIN/><ac
android:name=.ªndroid.intent.action.VIEW/>

<category android:name=.ªndroid.intent.category.LAUNCHER/></intent-filter></activity><uses-
library android:name=.ºrg.apache.http.legacy.ªndroid:required="false/><provider android:name=.ªndroid
$exported = "false"android : grantUriPermissions = "true" >< meta - dataandroid :$
$name = "android.support.FILE_PROVIDER_PATHS"android : resource = "@xml/file_paths" ><$
$/meta - data >< /provider >< /application >$

</manifest>

package juarez.mapas.navegador.baches; import android.Manifest; import android.os.Environment;
import android.widget.ArrayAdapter;

import java.io.File; import java.io.FileOutputStream; import java.io.IOException; import
java.text.SimpleDateFormat; import java.util.Date; import java.util.HashMap;

import android.content.Intent; import android.graphics.Bitmap; import android.provider.MediaStore;
import android.support.annotation.Nullable; import android.support.v7.app.AppCompatActivity;
import android.os.Bundle; import android.view.View; import android.widget.Button; import
android.widget.ImageView; import android.widget.Spinner; import android.widget.Toast; im-
port android.support.v4.content.ContextCompat; import android.content.pm.PackageManager;
import android.support.v4.app.ActivityCompat;

import okhttp3.MediaType; import okhttp3.MultipartBody; import okhttp3.RequestBody;
import retrofit2.Call; import retrofit2.Callback; import retrofit2.Response; import retrofit2.Retrofit;
import retrofit2.converter.gson.GsonConverterFactory; import android.location.LocationManager;
import android.location.Location; public class IngresoUsuario extends AppCompatActivity

Spinner TipoProblemaSpinner; HashMap<Integer,String>spinnerMap; ImageView imageView; private int UsuarioLogeadoID; private String fotoPath; private Boolean fotoTaken =false; private int ProblemaNuevoID;

Button btnAbrirMapa;

@Override protected void onCreate(Bundle savedInstanceState) super.onCreate(savedInstanceState); setContentView(R.layout.activity$_{i}ngreso_{u}suario$);

//boton ingresar a mapa total

btnAbrirMapa = (Button) findViewById(R.id.BtncEntrarMapa); btnAbrirMapa.setOnClickListener(ne View.OnClickListener() @Override public void onClick(View v) startActivity(new Intent(IngresoUsuario.tl MapaBaches.class)););

//fin boton mapa total

TipoProblemaSpinner = (Spinner)findViewById(R.id.TipoProblemaSpinner);

Button TomarFotoButton=(Button)findViewById(R.id.BtnCaptImagen); imageView = (ImageView)findViewById(R.id.imageViewID);

TomarFotoButton.setOnClickListener(new View.OnClickListener() @Override public void onClick(View v)

Intent intent = new Intent(MediaStore.ACTION$_{I}MAGE_{C}APTURE$); $startActivityForResult(intent,$ $(Button)findViewById(R.id.BtncShareImagen)$;

btnShare.setOnClickListener(new View.OnClickListener() @Override public void onClick(View v) SubirProblema();); Bundle b = getIntent().getExtras(); UsuarioLogeadoID = -1; // or other values if(b != null) UsuarioLogeadoID = b.getInt(ÜsuarioLogeadoID");

// 1 Bache // 2 Calle Sin Pavimento // 3 Trabajos de Mantenimiento // 4 Basura // 5 Alcantarilla Abierta

String[] spinnerArray = new String[12]; spinnerMap = new HashMap<Integer, String>();

spinnerMap.put(0,"Bache"); spinnerArray[0] = "Bache"; spinnerMap.put(1,.ªlcantarilla Sin Tapa"); spinnerArray[1] = .ªlcantarilla Sin Tapa"; spinnerMap.put(2,"Zona Con Encharcamiento (Pequeño o Grande)"); spinnerArray[2] = "Zona Con Encharcamiento (Pequeño o Grande)"; spinnerMap.put(3,"Zona Con Encharcamiento y Baches"); spinnerArray[3] = "Zona Con Encharcamiento y Baches"; spinnerMap.put(4,"Pavimento en Mal Estado (Vibraciones, Desgastado)"); spinnerArray[4] = "Pavimento en Mal Estado (Vibraciones, Desgastado)"; spinnerMap.put(5,"Trabajos de Mantenimiento"); spinnerArray[5] = "Trabajos de Mantenimiento"; spinnerMap.put(6,Çalle Sin Pavimento"); spinnerArray[6] = Çalle Sin Pavimento"; spinnerMap.put(7,"Fuga de Agua (Potable, Residuales)"); spinnerArray[7] = "Fuga de Agua (Potable, Residuales)"; spinnerMap.put(8,"Basura"); spinnerArray[8] = "Basura"; spinnerMap.put(9,"Banqueta Obstruida"); spinnerArray[9] = "Banqueta Obstruida"; spinnerMap.put(10,"Banqueta En Mal Estado"); spinnerArray[10] = "Banqueta En Mal Estado"; spinnerMap.put(11,Rampa Para Discapacitados Mal Diseñada"); spinnerArray[11] = Rampa Para Discapacitados Mal Diseñada";

ArrayAdapter<String>adapter =new ArrayAdapter<String>(this,android.R.layout.simple$_s pinner_i tem$
$PackageManager.PERMISSION_G RANTEDContextCompat.checkSelfPermission(this, android.Ma$
$PackageManager.PERMISSION_G RANTEDContextCompat.checkSelfPermission(this, android.Ma$
$PackageManager.PERMISSION_G RANTED)//googleMap.setMyLocationEnabled(true); //googleMa$

// Create an image file name String timeStamp = new SimpleDateFormat(zyyyMMdd$_H Hmmss$").$form$
$"JPEG_"+timeStamp+"_"; //Thisisthedirectoryinwhichthefilewillbecreated.Thisisthedefaultlocationof$
$newFile(Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_D CIM),"Can$
$File.createTempFile(imageFileName, /* prefix */".jpg", /* suffix */storageDir/ *$
$directory*/); //Saveafile : pathforusingagainfotoPath = image.getAbsolutePath(); //fotoPath =$
$"file : //"+image.getAbsolutePath(); returnimage; @OverrideprotectedvoidonActivityResult(intreques$

Bitmap bitmap = (Bitmap)data.getExtras().get("data"); imageView.setImageBitmap(bitmap);
try File fileTemp = createImageFile();

catch (IOException e) e.printStackTrace();

try (FileOutputStream out = new FileOutputStream(fotoPath)) bitmap.compress(Bitmap.CompressFo
100, out); // bmp is your Bitmap instance // PNG is a lossless format, the compression factor (100) is ignored catch (IOException e) e.printStackTrace(); fotoTaken = true; private void SubirImagen() if(!fotoTaken) Toast toast = Toast.makeText(getApplicationContext(), "Favor de tomar una foto del problema", Toast.LENGTH$_{S}HORT$);

toast.show(); return; String url = ; Util util = new Util();

File file = new File(fotoPath); // Create a request body with file and image media type RequestBody fileReqBody = RequestBody.create(MediaType.parse(ïmage/*"), file); // Create MultipartBody.Part using file request-body,file name and part name MultipartBody.Part part = MultipartBody.Part.createFormData(üpload", file.getName(), fileReqBody);

RequestBody description = RequestBody.create(MediaType.parse("text/plain"), ïmagetype"); RequestBody ProblemaNuevoIDPart = RequestBody.create(MediaType.parse("text/plain"), Integer.toString(ProblemaNuevoID)); RequestBody UsuarioLogeadoIDPart = RequestBody.create(MediaT Integer.toString(UsuarioLogeadoID));

Retrofit retrofit = new Retrofit.Builder() .baseUrl(util.baseUrl) .addConverterFactory(GsonConverterFa .build(); HomeService postService = retrofit.create(HomeService.class);

Call<HomeResponse >call = postService.postImagen(part, description, ProblemaNuevoIDPart, UsuarioLogeadoIDPart);

call.enqueue(new Callback<HomeResponse>() @Override public void onResponse(Call<HomeResponse Response<HomeResponse>response)

if(response.body().getSuccess())

Toast toast = Toast.makeText(getApplicationContext(), Ïmagen subida correctamente", Toast.LENGTH$_{S}HORT$);$toast.show()$;

else Toast toast = Toast.makeText(getApplicationContext(), response.body().getResponseText(), Toast.LENGTH$_{S}HORT$);

toast.show(); //arrayAdapter.notifyDataSetChanged();

@Override public void onFailure(Call<HomeResponse>call, Throwable t) Toast toast = Toast.makeText(getApplicationContext(), .^{Er}ror al Registrar Usuario", Toast.LENGTH$_{S}HORT$);

toast.show(););

private void SubirProblema() if(!fotoTaken) Toast toast = Toast.makeText(getApplicationContext(), "Favor de tomar una foto del problema", Toast.LENGTH$_{S}HORT$);

toast.show(); return; String url = ; Util util = new Util(); if (ContextCompat.checkSelfPermission(this, android.Manifest.permission.ACCESS$_{F}INE_{L}OCATION$) == $PackageManager.PERMISSION_{G}RAN$ $PackageManager.PERMISSION_{G}RANTEDContextCompat.checkSelfPermission(this, android.Ma$ $PackageManager.PERMISSION_{G}RANTED)//googleMap.setMyLocationEnabled(true); //googleMa$

//String name = spinner.getSelectedItem().toString(); String ProblemaTipoID = spinnerMap.get(TipoProblemaSpinner.getSelectedItemPosition());

Retrofit retrofit = new Retrofit.Builder() .baseUrl(util.baseUrl) .addConverterFactory(GsonConverterFa .build(); HomeService postService = retrofit.create(HomeService.class); Call<HomeResponse >call = postService.postProblema("Valentin Fuentes", Double.toString(latitude), Double.toString(longitud TipoProblemaSpinner.getSelectedItemPosition()+1, UsuarioLogeadoID);

call.enqueue(new Callback<HomeResponse>() @Override public void onResponse(Call<HomeResponse Response<HomeResponse>response)

if(response.body().getSuccess())

Toast toast = Toast.makeText(getApplicationContext(), "Problema Reportado Correctamente, subiendo Imagen", Toast.LENGTH$_{S}HORT$); $toast.show(); ProblemaNuevoID = response.body().getReturnID(); SubirImagen(); elseToasttoast = Toast.makeText(getApplicationConte$

toast.show(); //arrayAdapter.notifyDataSetChanged();

@Override public void onFailure(Call<HomeResponse>call, Throwable t) Toast toast = Toast.makeText(getApplicationContext(), .ᴱʳror al Registrar Usuario", Toast.LENGTH$_{SHORT}$);

toast.show(););

<?xml version="1.0.ᵉⁿcoding=ütf-8¿><LinearLayout xmlns:android="http://schemas.android.com/ap auto"xmlns:tools="http://schemas.android.com/tools.ᵃndroid:layout$_{width}$ = "$wrap_{content}$"$android$: $layout_{height}$ = "$match_{parent}$"$android$: $background$ = "@$color/colorPrimary$"$android$: $orientation$ = "$vertical$"$tools$: $context$ = ".$IngresoUsuario$" >

<ImageView android:id="@+id/imageViewID.ᵃndroid:layout$_{width}$ = "$match_{parent}$"$android$: $layout_{height}$ = "300dp"$android$: $layout_{weight}$ = "15"$android$: $background$ = "@$color/colorPrimaryDar$ $textAlignment$ = "$center$" / >

<TextView android:id="@+id/textView.ᵃndroid:layout$_{width}$ = "$match_{parent}$"$android$: $layout_{height}$ = "$wrap_{content}$"$android$: $layout_{margin}Top$ = "16dp"$android$: $layout_{margin}Bottom$ = "10dp"$android$: $layout_{weight}$ = "1"$android$: $text$ = "$SeleccionaelTipodeProblema$"$android$: $textAlignment$ = "$center$" / >

<Spinner android:id="@+id/TipoProblemaSpinner.ᵃndroid:layout$_{width}$ = "363dp"$android$: $layout_{height}$ = "25dp"$android$: $layout_{margin}Top$ = "2dp"$android$: $layout_{margin}Bottom$ = "8dp"$android$: $layout_{weight}$ = "2"$android$: $background$ = "@$color/colorPrimaryDark$"$android$: $spinnerMode$ = "$dialog$" / >< $LinearLayoutandroid$: $layout_{width}$ = "$match_{parent}$"$android$: $layout_{height}$ = "$wrap_{content}$" >

<Button android:id="@+id/BtncShareImagen.ᵃndroid:layout$_{width}$ = "83dp"$android$: $layout_{height}$ = "72dp"$android$: $layout_{gravity}$ = "$left|end$"$android$: $layout_{margin}Left$ = "0dp"$android$: $background$ = "@$drawable/ic_{account}_{group}$" / >

<Button android:id="@+id/BtnCaptImagen.ᵃndroid:layout$_{width}$ = "$wrap_{content}$"$android$: $layout_{height}$ = "78dp"$android$: $layout_{gravity}$ = "$center$"$android$: $layout_{margin}Left$ =

"50dp"$android$: $background$ = "@$drawable/ic_camera_plus$"$android$: $clickable$ = "$true$" / >

<Button android:id="@+id/BtncEntrarMapa.ªndroid:layout$_width$ = "83dp"$android$: $layout_height$ = "72dp"$android$: $layout_gravity$ = "$right|end$"$android$: $layout_marginLeft$ = "50dp"$android$: $background$ = "@$drawable/ic_map_marker_alert$" / >< /$LinearLayout$ >

<!–<android.support.design.widget.FloatingActionButton android:id="@+id/BtnCaptImagen.ªndroid "$wrap_content$"$android$: $layout_height$ = "$match_parent$"$android$: $layout_gravity$ = "$center$"$android$: $clickable$ = "$true$"app : $backgroundTint$ = "@$android$: $color/holo_green_dark$"app : $srcCompat$ = "@$android$: $drawable/ic_menu_camera$" / >

<android.support.design.widget.FloatingActionButton android:id="@+id/BtncShareImagen.ªndroid:la "$wrap_content$"$android$: $layout_height$ = "$wrap_content$"$android$: $layout_weight$ = "15"$android$: $clickable$ = "$true$"app : $backgroundTint$ = "@$android$: $color/holo_green_dark$"app : $srcCompat$ = "@$android$: $drawable/ic_menu_share$" / > −− >

</LinearLayout>

package juarez.mapas.navegador.baches;

import android.content.Intent; import android.support.v7.app.AppCompatActivity; import android.os.Bundle; import android.view.View; import android.widget.Button; import android.widget.TextView; import android.widget.Toast; import java.util.List;

import retrofit2.Call; import com.google.gson.Gson; import com.google.gson.GsonBuilder; import java.util.ArrayList; import retrofit2.Call; import retrofit2.Callback; import retrofit2.Response; import retrofit2.Retrofit; import retrofit2.converter.gson.GsonConverterFactory;

public class Registro extends AppCompatActivity Button BotonIngresar; ArrayList<String>titles = new ArrayList<>(); TextView NombreRegistro; TextView CorreoRegistro; TextView PsswordRegistro;

@Override protected void onCreate(Bundle savedInstanceState) super.onCreate(savedInstanceState); setContentView(R.layout.activity$_registro$); $BotonIngresar = findViewById(R.id.BotonRegistro)$; Nom

$findViewById(R.id.CajaTextoNombreRegistro); CorreoRegistro = findViewById(R.id.CajaTextoC$
$findViewById(R.id.CajaTextoPassRegistro);$

BotonIngresar.setOnClickListener(new View.OnClickListener() @Override public void onClick(View v) RegistraUsuario(NombreRegistro.getText().toString(), CorreoRegistro.getText().toString PsswordRegistro.getText().toString()); //);

private void RegistraUsuario(String Nombre, String Correo, String Password) String url = ; Util util = new Util();

Retrofit retrofit = new Retrofit.Builder() .baseUrl(util.baseUrl) .addConverterFactory(GsonConverterFa .build(); HomeService postService = retrofit.create(HomeService.class); Call<HomeResponse >call = postService.postRegistro(Correo, Password, Nombre);

call.enqueue(new Callback<HomeResponse>() @Override public void onResponse(Call<HomeRespons Response<HomeResponse>response)

if(response.body().getSuccess())

Toast toast = Toast.makeText(getApplicationContext(), Üsuario Creado Correctamente", Toast.LENGTH$_SHORT$);

toast.show(); Intent intent = new Intent(Registro.this, IngresoUsuario.class); Bundle b = new Bundle(); b.putInt(ÜsuarioLogeadoID", response.body().getReturnID()); //Your id intent.putExtras(b); //Put your id to your next Intent startActivity(intent); finish(); else Toast toast = Toast.makeText(getApplicationContext(), response.body().getResponseText(), Toast.LENGTH$_SH$

toast.show();

//arrayAdapter.notifyDataSetChanged();

@Override public void onFailure(Call<HomeResponse>call, Throwable t) Toast toast = Toast.makeText(getApplicationContext(), .ErrorÖr al Registrar Usuario", Toast.LENGTH$_SHORT$);

toast.show(););

<?xml version="1.0."encoding=ütf-8¿><LinearLayout xmlns:android="http://schemas.android.com/ap
auto"xmlns:tools="http://schemas.android.com/tools.ªndroid:layout$_{w}idth$ = "$wrap_{c}ontent$"$android$:
$layout_{h}eight$ = "$match_{p}arent$"$android$: $background$ = "@$drawable/potholedos$"$android$:
$orientation$ = "$vertical$"$tools$: $context$ = "$.Registro$" >

<TextView android:id="@+id/CajaTextoTituloRegistro.ªndroid:layout$_{w}idth$ = "$match_{p}arent$"$androi$
$layout_{h}eight$ = "$wrap_{c}ontent$"$android$: $layout_{m}arginTop$ = "20dp"$android$: $text$ = "@$string/action_{t}itle$"
$textAlignment$ = "$center$"$android$: $textSize$ = "40sp"/ >

<EditText android:id="@+id/CajaTextoNombreRegistro.ªndroid:layout$_{w}idth$ = "$match_{p}arent$"$androi$
$layout_{h}eight$ = "$wrap_{c}ontent$"$android$: $layout_{m}arginStart$ = "20dp"$android$: $layout_{m}arginTop$ =
"20dp"$android$: $layout_{m}arginEnd$ = "20dp"$android$: ems = "10"$android$: $hint$ =
"@$string/action_{r}egister$"$android$: $inputType$ = "$textPersonName$"/ >

<EditText android:id="@+id/CajaTextoCorreoRegistro.ªndroid:layout$_{w}idth$ = "$match_{p}arent$"$androi$
$layout_{h}eight$ = "$wrap_{c}ontent$"$android$: $layout_{m}arginStart$ = "20dp"$android$: $layout_{m}arginTop$ =
"20dp"$android$: $layout_{m}arginEnd$ = "20dp"$android$: ems = "10"$android$: $hint$ =
"@$string/prompt_{e}mail$"$android$: $inputType$ = "$textEmailAddress$"/ >

<EditText android:id="@+id/CajaTextoPassRegistro.ªndroid:layout$_{w}idth$ = "$match_{p}arent$"$android$:
$layout_{h}eight$ = "$wrap_{c}ontent$"$android$: $layout_{m}arginStart$ = "20dp"$android$: $layout_{m}arginTop$ =
"20dp"$android$: $layout_{m}arginEnd$ = "20dp"$android$: ems = "10"$android$: $hint$ =
"@$string/prompt_{p}assword$"$android$: $inputType$ = "$textPassword$"/ >

<Button android:id="@+id/BotonRegistro.ªndroid:layout$_{w}idth$ = "$match_{p}arent$"$android$:
$layout_{h}eight$ = "$wrap_{c}ontent$"$android$: $layout_{m}arginStart$ = "50dp"$android$: $layout_{m}arginTop$ =
"20dp"$android$: $layout_{m}arginEnd$ = "50dp"$android$: $text$ = "@$string/action_{b}utton_{r}egister$"/ >< /$LinearLayout$ >

Apéndice B
Archivo generado por la base de datos

USE [simplepc$_D allas Meetup]GO/******Object : Table[simplepc$_D allas MeetupUser].[Calles]ScriptDa
10/29/201910 : 02 : 53PM******/SETANSI$_N ULLSONGOSETQUOTED$_I DENTIFIERONGOSET
$OFF, STATISTICS_N ORECOMPUTE = OFF, IGNORE_D UP_K EY = OFF, ALLOW_R OW_L OCKS =$
$ON, ALLOW_P AGE_L OCKS = ON)ON[PRIMARY])ON[PRIMARY]$

GO SET ANSI$_P ADDINGOFFGO/******Object : Table[simplepc_D allas MeetupUser].[Imagenes]Sc$
10/29/201910 : 02 : 54PM******/SETANSI$_N ULLSONGOSETQUOTED_I DENTIFIERONGOSET$
$OFF, STATISTICS_N ORECOMPUTE = OFF, IGNORE_D UP_K EY = OFF, ALLOW_R OW_L OCKS =$
$ON, ALLOW_P AGE_L OCKS = ON)ON[PRIMARY])ON[PRIMARY]$

GO SET ANSI$_P ADDINGOFFGO/******Object : Table[simplepc_D allas MeetupUser].[Problemas]S$
10/29/201910 : 02 : 54PM******/SETANSI$_N ULLSONGOSETQUOTED_I DENTIFIERONGOSET$
$OFF, STATISTICS_N ORECOMPUTE = OFF, IGNORE_D UP_K EY = OFF, ALLOW_R OW_L OCKS =$
$ON, ALLOW_P AGE_L OCKS = ON)ON[PRIMARY])ON[PRIMARY]$

GO SET ANSI$_P ADDINGOFFGO/******Object : Table[simplepc_D allas MeetupUser].[TipoProblem$
10/29/201910 : 02 : 54PM******/SETANSI$_N ULLSONGOSETQUOTED_I DENTIFIERONGOSET$
$OFF, STATISTICS_N ORECOMPUTE = OFF, IGNORE_D UP_K EY = OFF, ALLOW_R OW_L OCKS =$
$ON, ALLOW_P AGE_L OCKS = ON)ON[PRIMARY])ON[PRIMARY]$

GO SET ANSI$_P ADDINGOFFGO/******Object : Table[simplepc_D allas MeetupUser].[usuarios]Scr$

$10/29/201910 : 02 : 54PM******/SETANSI_NULLSONGOSETQUOTED_IDENTIFIERONGOSET$
$OFF, STATISTICS_NORECOMPUTE = OFF, IGNORE_DUP_KEY = OFF, ALLOW_ROW_LOCKS =$
$ON, ALLOW_PAGE_LOCKS = ON)ON[PRIMARY])ON[PRIMARY]$

GO SET ANSI$_P ADDING OFF GO ALTER TABLE [simplepc_D allas MeetupUser].[Imagenes] WITH$