



**UNIVERSIDAD TÉCNICA DE AMBATO**

**FACULTAD DE TECNOLOGÍAS DE LA INFORMACIÓN,  
TELECOMUNICACIONES E INDUSTRIAL**

**CARRERA DE INGENIERÍA EN SISTEMAS COMPUTACIONALES E  
INFORMÁTICOS**

TEMA:

---

**APLICACIÓN HÍBRIDA PARA LA GESTIÓN DE DA-  
TOS GEORREFERENCIADOS OFFLINE UTILIZANDO  
SOFTWARE LIBRE.**

---

Proyecto de Trabajo de Graduación. Modalidad: Proyecto de investigación, presentado previo a la obtención del título de Ingeniero en Sistemas Computacionales e Informáticos.

**SUBLÍNEA DE INVESTIGACIÓN:**  
Desarrollo de Software

Autor: Víctor Hugo Bautista Salazar  
Tutor: Dr. Félix Oscar Fernández Peña

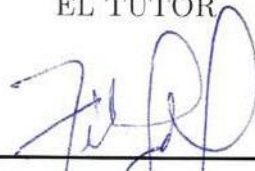
Ambato - Ecuador  
julio, 2019

## APROBACIÓN DEL TUTOR

En mi calidad de Tutor del Trabajo de Investigación sobre el Tema: "APLICACIÓN HÍBRIDA PARA LA GESTIÓN DE DATOS GEORREFERENCIADOS OFFLINE UTILIZANDO SOFTWARE LIBRE ", del señor Víctor Rugo Bautista Salazar, estudiante de la Carrera de Ingeniería en Sistemas Computacionales e Informáticos, de la Facultad de Tecnologías de la Información, Telecomunicaciones e Industrial, de la Universidad Técnica de Ambato, considero que el informe investigativo reúne los requisitos suficientes para que continúe con los trámites y consiguiente aprobación de conformidad con el numeral 7.2 de los Lineamientos Generales para la aplicación de Instructivos de las autoridades de Titulación de las Facultades de la Universidad Técnica de Ambato.

Ambato, julio de 2019

EL TUTOR



---

Dr. Félix Oscar Fernández Peña

## AUTORÍA

El presente proyecto de investigación titulado: "APLICACIÓN HÍBRIDA PARA LA GESTIÓN DE DATOS GEORREFERENCIADOS OFFLINE UTILIZANDO SOFTWARE LIBRE" es absolutamente original, auténtico y personal, en tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, julio de 2019

Víctor Rugo Bautista Salazar



---

CC: 180430713-8

## **DERECHOS DE AUTOR**

Autorizo a la Universidad Técnica de Ambato, para que haga uso de este Trabajo de Titulación como un documento disponible para la lectura, consulta y procesos de investigación.

Cedo los derechos de mi Trabajo de Titulación, con fines de difusión pública, además autorizo su reproducción dentro de las regulaciones de la Universidad.

Ambato, julio de 2019

Víctor Hugo Bautista Salazar



---

CC: 180430713-8

## APROBACIÓN COMISIÓN CALIFICADORA


La comisión calificadora del presente proyecto conformada por los señores docentes PhD. Julio Balarezo e Ing. Hernando Buenaño, revisó y aprobó el Informe Final del proyecto de graduación titulado "APLICACIÓN HÍBRIDA PARA LA GESTIÓN DE DATOS GEORREFERENCIADOS OFFLINE UTILIZANDO SOFTWARE LIBRE", presentado por el señor Víctor Hugo Bautista Salazar de acuerdo al numeral 1.9.1 de los Lineamientos Generales para la aplicación de Instructivos de las Modalidades de Titulación de las Facultades de la Universidad Técnica de Ambato.

Ambato, julio de 2019



---

Ing. Mg. Elsa Pilar Urrutia Urrutia  
PRESIDENTA DEL TRIBUNAL



---

PhD. Julio Balarezo  
DOCENTE CALIFICADOR



---

Ing. Hernando Buenaño  
DOCENTE CALIFICADOR

## **DEDICATORIA**

Se la dedico a mi padre celestial forjador de mi camino, y es quien siempre me acompaña y siempre me levanta de mis tropiezos, a mis padres y a las personas que más amo, con mi más sincero amor.

Víctor H. Bautista Salazar

## **AGRADECIMIENTO**

Gracias a mis padres por ser los promotores de mis sueños, por creer en mi y respaldarme cuando más necesité, la paciencia que me han brindado y sobre todo la comprensión que necesitaba para emprender mi propia motivación por esta vocación la cual siempre me ha gustado. Gracias a todas las personas que han estado ahí para consejos y palabras de respaldo, las largas noches de estudio para llegar al objetivo siempre valdrán la pena.

Víctor H. Bautista Salazar

# **APLICACIÓN HÍBRIDA PARA LA GESTIÓN DE DATOS GEORREFERENCIADOS OFFLINE UTILIZANDO SOFTWARE LIBRE.**

Víctor Hugo Bautista<sup>1</sup>

<sup>1</sup> *Universidad Técnica de Ambato, Facultad de Tecnologías de la información  
Telecomunicaciones e Industrial*

## **RESUMEN**

El presente trabajo investigativo aporta una solución a la gestión de información georreferenciada en equipos móviles utilizando software libre y mapas vectoriales. La propuesta pretende fomentar el uso del framework Ionic para el desarrollo de aplicaciones híbridas. Como resultado, se logró un aplicativo multiplataforma que optimiza recursos en tiempo y esfuerzo puesto que el código obtenido es compilable tanto para Android como para las plataformas IOS y Windows Phone. La gestión de datos georreferenciados tiene lugar utilizando mapas vectoriales disponibles de forma gratuita en Internet. De este modo, los usuarios no requieren de conexión para hacer anotaciones georreferenciadas y la aplicación se puede distribuir de forma gratuita. El estudio llevado a cabo evidencia que el aplicativo desarrollado tiene un alto nivel de usabilidad, lo que facilita su uso para la recuperación de información georreferenciada precisa y actualizada en diversos ámbitos de aplicación.

**Palabras Claves:** Móvil, Ionic, mapas vectoriales, Android, IOS, aplicación híbrida, software libre, datos georreferenciados.



# **HYBRID APPLICATION FOR THE MANAGEMENT OF GEOREFERENCED DATA OFFLINE USING SOFTWARE OPEN SOURCE.**

Víctor Hugo Bautista<sup>1</sup>

<sup>1</sup> *Universidad Técnica de Ambato, Facultad de Tecnologías de la información  
Telecomunicaciones e Industrial*

## **ABSTRACT**

This research work brings an open source solution to the management of georeferenced data in mobile devices by using vector maps. This proposal encourages the use of the Ionic framework for the development of hybrid solutions. As a result, a multi-platform app optimizing time and efforts was achieved. This is because the same source code can be seemly compiled for Android, IOS and Windows Phone. The georeferenced data management takes place using vector maps which were freely downloaded from Internet. This way, users do not need an online connection in order to make georeferenced annotation and the app can be distributed for free. This investigation proved that the app has a high level of usability, making easy to use it for retrieving accurate and updated georeferenced data in any scope of application.

**Keywords:** Mobile, Ionic, vector maps, Android, IOS, hybrid app, open source, georeferenced data.

## ÍNDICE

Aprobación del tutor .....	ii
Autoría .....	iii
Derechos de autor .....	iv
Aprobación comisión calificadora .....	v
Dedicatoria .....	vi
Agradecimiento .....	vii
Resumen .....	viii
Abstract .....	ix
<b>Introducción</b>	<b>xvi</b>
<b>CAPÍTULO 1 El Problema</b>	<b>1</b>
1.1 Tema . . . . .	1
1.2 Planteamiento del problema . . . . .	1
1.3 Delimitación . . . . .	2
1.4 Justificación . . . . .	2
1.5 Objetivos . . . . .	3
1.5.1 General . . . . .	3
1.5.2 Específicos . . . . .	3
<b>CAPÍTULO 2 Marco Teórico</b>	<b>4</b>
2.1 Antecedentes Investigativos . . . . .	4
2.2 Fundamentación teórica . . . . .	5
2.2.1 Georreferenciación y sistemas de coordenadas . . . . .	5
2.2.2 Base de datos móviles . . . . .	6
2.2.3 Almacenamiento offline de datos . . . . .	6
2.2.4 XAMPP . . . . .	7
2.2.5 PostgreSQL . . . . .	7
2.2.6 API . . . . .	8
2.2.7 Codeigniter . . . . .	9
2.2.8 OpenStreetMap .....	10
2.2.9 OpenMapTiles .....	10
2.2.10 MapBox.....	10
2.2.11 Bootstrap .....	11

2.2.12	AngularJS .....	12
2.2.13	Apache Cordova.....	13
2.2.14	Ionic Framework .....	13
2.2.15	Arquitectura Ionic.....	14
2.2.16	Mapas Raster .....	15
2.2.17	Mapas Vectoriales.....	15
2.2.18	Raster vs Vectorial .....	16
2.3	Propuesta de Solución.....	17
<b>CAPÍTULO 3 Metodología</b>		<b>18</b>
3.1	Modalidad de la investigación.....	18
3.2	Recolección de información.....	19
3.3	Procesamiento y análisis de datos.....	19
3.4	Desarrollo del proyecto.....	20
<b>CAPÍTULO 4 Desarrollo de la propuesta</b>		<b>22</b>
4.1	Análisis de requerimientos del sistema de información .....	22
4.1.1	Levantamiento de requerimientos funcionales.....	23
4.1.2	Levantamiento de requerimientos no funcionales .....	23
4.2	Diseño de arquitectura de la aplicación multiplataforma.....	24
4.2.1	Diseño de modelo físico de base de datos.....	24
4.2.2	Diseño del API-REST .....	25
4.2.3	Diagramas de actividades de la aplicación .....	37
4.3	Diagramas de secuencia.....	43
4.4	Desarrollo de la aplicación .....	48
4.4.1	Especificación de casos de uso .....	48
4.4.2	Producción de software.....	55
4.4.3	Requisitos no funcionales.....	66
4.4.4	Transición.....	72
<b>CAPÍTULO 5 Conclusiones y Recomendaciones</b>		<b>74</b>
5.1	Conclusiones .....	74
5.2	Recomendaciones .....	75
<b>Bibliografía</b>		<b>77</b>
<b>ANEXOS</b>		<b>82</b>

## ÍNDICE DE FIGURAS

1	Servicio Web Rest API [1] . . . . .	8
2	Arquitectura de los componentes de AngularJS Fuente: Medium[2].	12
3	Arquitectura aplicaciones Ionic. Fuente: Ionic Framework [3].	14
4	Entidad poligonal ráster. Fuente: SIG [4].	16
5	Imagen vectorial vs ráster Fuente: Mapas para OruxMaps[5] . . .	17
6	Arquitectura de la aplicación Fuente: Elaborado por el investigador	24
7	Modelo físico de la base de datos remota Fuente: Elaborado por el investigador	24
8	Modelo físico de la base de datos local Fuente: Elaborado por el investigador	25
9	Diagrama de actividades. Ingreso a la aplicación Fuente: Elaborado por el investigador	38
10	Diagrama de actividades. Iniciar sesión. Fuente: Elaborado por el investigador	39
11	Diagrama de actividades. Visualización de datos. Fuente: Elaborado por el investigador	40
12	Diagrama de actividades. Búsqueda de datos Fuente: Elaborado por el investigador	41
13	Diagrama de actividades. Agregar datos Fuente: Elaborado por el investigador	41
14	Diagrama de actividades. Modificar datos Fuente: Elaborado por el investigador.	42
15	Diagrama de actividades. Eliminar datos Fuente: Elaborado por el investigador	42
16	Diagrama de actividades. Sincronizar datos Fuente: Elaborado por el investigador	43
17	Diagrama de secuencia. Visualización de datos Fuente: Elaborado por el investigador	43
18	Diagrama de secuencia. Agregar puntos Fuente: Elaborado por el investigador	44
19	Diagrama de secuencia. Buscar de datos Fuente: Elaborado por el investigador	44

20	Diagrama de secuencia. Compartir puntos Fuente: Elaborado por el investigador .....	45
21	Diagrama de secuencia. Eliminar punto Fuente: Elaborado por el investigador.....	45
22	Diagrama de secuencia. Login Fuente: Elaborado por el investigador	46
23	Diagrama de secuencia. Modificar de datos Fuente: Elaborado por el investigador .....	46
24	Diagrama de secuencia. Sincronizar datos Fuente: Elaborado por el investigador .....	47
25	Diagrama de casos de uso. Fuente: Elaborado por el investigador.	48
26	Diagrama de clases. Programación del cliente Fuente: Elaborado por el investigador .....	55
27	Diagrama de clases. Programación del API-REST .....	56
28	Extracto de código. Estilo <i>klokantech-basic-gl-style</i> Fuente: Elaborado por el investigador.....	57
29	Extracto de código. Estilo <i>klokantech-basic-gl-style</i> modificado Fuente: Elaborado por el investigador.....	57
30	Vista de pantalla. Ingreso a la aplicación. Fuente: Elaborado por el investigador .....	58
31	Vista de pantalla. Login Fuente: Elaborado por el investigador . .	59
32	Vista general de la aplicación Fuente: Elaborado por el investigador	60
33	Extracto de código. Cliente - Sincronización de datos Fuente: Elaborado por el investigador.....	60
34	Vista de pantalla. <i>Lugares</i> de la aplicación. Fuente: Elaborado por el investigador. ....	61
35	Extracto de código. Cliente - Comparti.r Fuente: Elaborado por el investigador.....	62
36	Vista de pantalla. Cliente - Ver detalle. Fuente: Elaborado por el investigador.....	62
37	Extracto de código. Ver Detalle Fuente: Elaborado por el investigador. ....	63
38	Vista de pantalla. Sincronización Fuente: Elaborado por el investigador.....	63
39	Vista de pantalla. Compartir lugar en las redes sociales Fuente: Elaborado por el investigador.....	64
40	Extracto de código. Cliente - Realizar acción “compartir”. Fuente: Elaborado por el investigador.....	64

41	Vista pantalla. Seleccionar punto. Fuente: Elaborado por el investigador.....	65
42	Vista Pantalla. Cliente. Registrar un nuevo contenido Fuente: Elaborado por el investigador.....	66
43	Extracto de código. Registrar un nuevo contenido. Fuente: Elaborado por el investigador.....	66
44	Extracto de código. Cuadro de texto. Fuente: Elaborado por el investigador.....	66
45	Extracto de código. Filtro <i>pipe</i> <b>buscar</b> en la lista lugares. Fuente: Elaborado por el investigador.....	66
46	Extracto de código. <i>Pipe</i> <b>Buscar</b> . Fuente: Elaborado por el investigador.....	67
47	Vista de pantalla. Búsqueda de lugares. Fuente: Elaborado por el investigador.....	67
48	Extracto de código. Colores en <i>variables.scss</i> . Fuente: Elaborado por el investigador.....	68
49	Extracto de código. Estilo del header de la aplicación Fuente: Elaborado por el investigador.....	69
50	Extracto de código. Estilo del header de la aplicación para la plataforma Android. Fuente: Elaborado por el investigador.....	69
51	Extracto de código. Archivo de configuración de la aplicación. Fuente: Elaborado por el investigador.....	70
52	Vista de pantalla. Horizontal. Fuente: Elaborado por el investigador.	71
53	Carga de recursos en dispositivo móvil. Fuente: Elaborado por el investigador.....	71
54	Carga de recursos en entorno de producción. Fuente: Elaborado por el investigador.....	72

## ÍNDICE DE TABLAS

1	Recolección de información. Fuente: Elaborado por el investigador	19
2	Ejemplo. Definición de métodos en Codeigniter Fuente: Elaborado por el investigador .....	27
3	Caso de uso. Registrar “usuario” Fuente: Elaborado por el investigador.....	49
4	Caso de uso. Inicio de sesión Fuente: Elaborado por el investigador	50
5	Caso de uso. Visualizar datos Fuente: Elaborado por el investigador	51
6	Caso de uso. Registrar un nuevo punto Fuente: Elaborado por el investigador.....	52
7	Caso de uso. Modificar un punto Fuente: Elaborado por el investigador.....	53
8	Caso de uso. Buscar datos Fuente: Elaborado por el investigador .	54
9	Caso de uso. Compartir en las redes sociales un punto Fuente: Elaborado porel investigador .....	54

## INTRODUCCIÓN

En la actualidad, el aumento del uso de dispositivos móviles se ha convertido en una parte integral de la vida cotidiana de los usuarios, lo que conduce a la expansión del desarrollo de aplicaciones y el desafío de utilizarlas para las diferentes plataformas.

El desarrollo de aplicaciones nativas en entornos móviles implica que el código solo se puede desplegar en una plataforma específica, ya sea iOS, Android, Windows Phone u alguna otra existente en el mercado. Esto conlleva a la recodificación de un aplicativo en diferentes lenguajes por cuanto se quiere que las aplicaciones estén disponibles para la mayor cantidad de móviles posible. En este sentido, el desarrollo de aplicaciones híbridas permite que se obtenga un entorno óptimo por facilidad de uso (desarrollo de aplicaciones en un único conjunto de lenguajes del ámbito web), formas eficientes de utilizar recursos del dispositivo (se tiene acceso a los sensores y dispositivos integrados al teléfono, tal y como si fuera una aplicación nativa) y por ser posible la ejecución en la mayoría de dispositivos (el código es interpretable en la mayoría de plataformas). De esta forma, se reducen el costo y el tiempo de desarrollo y mantenimiento de los aplicativos.

Por su parte, las aplicaciones de gestión de datos georreferenciados han sido desarrolladas para cada una de las plataformas móviles específicas requiriendo, por lo general, la conexión a Internet para el acceso a servicio de mapas, enrutamiento, y geoposicionamiento. Una alternativa, menos explotada en el mercado, es la de gestionar datos en modo desconectado, habiendo descargado la información de los mapas a utilizar.

El presente trabajo está enmarcado en el desarrollo de una aplicación de este tipo utilizando software libre. Este documento consta de cinco capítulos; su contenido se detalla a continuación:

**Capítulo I.** “El Problema”, se identifica el problema que se suscita en un contexto de la realidad, para plantearlo de forma concreta, delimitando el alcance, con una respectiva justificación y el planteamiento de los objetivos que guiarán todo el proyecto.

**Capítulo II.** “Marco Teórico”, consta del fundamento teórico que ayuda a comprender de forma clara el problema gracias a los antecedentes investigativos, para luego plantear la propuesta de solución.

**Capítulo III.** “Metodología”, Se describe las metodologías de investigación que



se utilizarán, el enfoque, la modalidad de la investigación utilizada, el tipo de investigación realizada.

**Capítulo IV.** “Desarrollo de la Propuesta”, en este capítulo se describe todo el desarrollo de la solución, definiendo los requisitos necesarios, los criterios que se aplicaron dando como resultado el plan estratégico.

**Capítulo V.** “Conclusiones y Recomendaciones” estableciendo las conclusiones a las que llega el investigador luego del desarrollo del proyecto, así también las recomendaciones pertinentes.

Finalmente se incluye las referencias citadas en este documento, en los anexos se incluye los instrumentos utilizados para la recolección de la información correspondientes del presente proyecto.

# CAPÍTULO 1

## El Problema

### 1.1. Tema

“APLICACIÓN HÍBRIDA PARA LA GESTIÓN DE DATOS GEORREFERENCIADOS OFFLINE UTILIZANDO SOFTWARE LIBRE.”.

### 1.2. Planteamiento del problema

En la actualidad las aplicaciones móviles requieren ser ejecutadas en diferentes plataformas teniendo en cuenta la tecnología que el usuario disponga. Por esta razón, son desarrolladas en cada uno de los entornos que están dados por los sistemas operativos móviles más utilizados en la actualidad: Android (Android Studio), iOS (Xcode) y Windows Phone (Visual Studio). Hoy en día existen aplicaciones nativas con gestión de datos georreferenciales (Google Maps, Mapbox, Maps.me), cada una con APIs para funciones específicas. Entre ellas, las más significativas incluyen: búsqueda por dirección, cálculo de rutas y geoposicionamiento. Las limitaciones en cuanto a las aplicaciones nativas son que el código no es reutilizable para los diferentes entornos; por lo tanto, el esfuerzo, costo de desarrollo, actualización y distribución de versiones posteriores, son las dificultades más relevantes. La aceptación de una aplicación móvil se da en base a que dispone de la posibilidad de ser ejecutado en diferentes entornos, con lo que se optimiza el costo de desarrollo, esfuerzo y facilidad de actualización.

Las aplicaciones móviles, en su mayoría, se emplean para comunicación, con el fin de informar a muchos usuarios. Por su parte, las aplicaciones offline enfrentan el problema de que únicamente tienen acceso a recursos almacenados de forma local debido a las restricciones del operador o a la cobertura de red en ciertos lugares. Por ejemplo, en caso de que un usuario deba hacer una anotación sobre un lugar sin disponer de conexión, una opción es anotar en un lugar que solo se guarde de recordatorio para ese usuario, sea en un mensaje de texto o en un stick notes con la descripción y dirección del lugar, otra opción es tomar una foto del lugar para cuando se requiera se debe buscar la foto para su ubicación y descripción.

El desarrollo de una aplicación con un framework open source que está en constante crecimiento con el propósito de gestionar puntos georeferenciales como las

aplicaciones; Google Maps y Mapbox, con líneas de código reutilizables para compilar en las plataformas Android, iOS y Windows Phone, por optimización de recursos, y facilidad de uso, el cual contará con información disponible sincronizada, manejo de información y características de recursos web HTML5, CSS y componentes JS con acceso a recursos nativos del dispositivo, será la pauta para la inmersión en el desarrollo de aplicaciones móviles híbridas.

### **1.3. Delimitación**

#### **Área Académica:**

Software.

#### **Línea de Investigación:**

Ingeniería de Software

#### **Sublínea de Investigación:**

Aplicaciones para dispositivos móviles.

#### **Delimitación Espacial:**

La presente propuesta de investigación se llevará a cabo en Ecuador.

#### **Delimitación Temporal:**

La realización del proyecto está planificada para un período de 6 meses de desarrollo desde 17/09/2018 hasta 24/01/2019

### **1.4. Justificación**

La investigación de herramientas para el desarrollo en entornos móviles permitirá ser una guía para estudiantes y profesores en el área informática ampliando la calidad de conocimiento obteniendo un modelo adaptable ocasionando el uso de nuevas herramientas de desarrollo en las futuras aplicaciones con patrones innovadores de diseño.

La implementación de un gestor de datos almacenados offline que conste de un modelo híbrido a nivel de programación expuesto en el presente proyecto será básicamente una ayuda para la gestión de datos almacenados de forma offline

para su posterior sincronización en el propio proyecto y proyectos similares con requerimiento multiplataforma.

La optimización y reducción de costos para implementar un modelo de almacenamiento de datos geo referenciales offline de manera estándar para todos los entornos móviles conocidos con sincronización para utilización en un GIS (Sistema de Información Geográfica).

El desarrollo de aplicaciones multiplataforma, se diferencia de aplicaciones nativas, en cuanto a los lenguajes de programación y el mecanismo de despliegue de la aplicación. La implementación de aplicaciones web móviles de manera no nativa es una idea viable; por otro lado, las limitaciones que representan en cuanto a su alcance en el tema de plataformas, han motivado a la realización de un software que tenga las características de ser híbrida de manera nativa. Se pretende desarrollar un aplicativo completamente funcional para la anotación georreferenciada desconectada de destinos de interés y la sincronización de esta información con una base de datos en línea.

## **1.5. Objetivos**

### **1.5.1. General**

Construir una aplicación híbrida para el manejo offline de datos georreferenciados.

### **1.5.2. Específicos**

- Determinar la plataforma de desarrollo de aplicación híbrida a utilizar.
- Implementar aplicación híbrida para la anotación offline de datos georreferenciados.
- Sincronizar los datos de la aplicación con servidor de datos georreferenciados.
- Evaluar los resultados en el contexto de un caso de estudio.

## **CAPÍTULO 2**

### **Marco Teórico**

#### **2.1. Antecedentes Investigativos**

Como antecedente investigativo se puede mencionar:

Según el trabajo presentado por Lic. Lisandro Nahuel Delía en el año 2017 de título; “Desarrollo de Aplicaciones Móviles Multiplataforma” . Este menciona algunos aspectos a tener en cuenta en lo referente menciona: "Para maximizar su presencia en el mercado, un producto de software debe ejecutarse en la mayor cantidad de dispositivos posible. Una solución consiste en el desarrollo nativo de la aplicación en cada una de las plataformas existentes utilizando el entorno de desarrollo integrado..."[6].

Los usuarios según sus necesidades adquieren diferentes dispositivos para cada propósito sea para uso personal o para lo laboral, por cuanto una misma aplicación que funcione en todos los dispositivos sería de gran utilidad para el usuario y se optimizaría recursos.

Según el trabajo presentado por Cesar Stuardo Lucho Romero en el año 2017 de título; “Aplicación multiplataforma para la gestión de archivos” . Este menciona algunos aspectos a tener en cuenta en lo referente menciona: “... la experiencia y aprendizaje tanto en la investigación de nuevas tecnologías y herramientas, como en el desarrollo de aplicaciones móviles multiplataforma y la gestión de archivos en las plataformas utilizadas. Ha sido un trabajo muy fructífero para aumentar mis conocimientos en un campo que no se profundiza demasiado durante los cuatro cursos del grado, y que ha cogido mucha fuerza durante los últimos años al aumentar la potencia tanto en smartphones como en tablets...”[7].

El potencial sobre el desarrollo de aplicaciones híbridas aumenta en base a los resultados sobre la acogida de este tipo de aplicaciones, existe acogida en este tipo de aplicaciones pese a que los cambios entre versiones en ocasiones son significativos pero a la larga son beneficios en cuanto líneas de código y entendimiento del mismo.

Según el trabajo presentado por Villacis Zúñiga Ángel Humberto y Barragán Averos Mercy Beatriz en el año 2016 de título; “LA TECNOLOGÍA ANDROID y su incidencia en el desarrollo de una aplicación móvil para la geo-localización de los centros asistenciales y farmacias de turnos para la DIRECCIÓN

PROVINCIAL DE SALUD LOS RÍOS ubicada en la ciudad de Babahoyo”.

Este menciona algunos aspectos a tener en cuenta en lo referente menciona:

“... Una aplicación nativa está programada en un lenguaje específico con APIs<sup>1</sup> propias de la plataforma. Se suele comprar, descargar y actualizar a través de la tienda de aplicaciones específica de la plataforma. Las aplicaciones nativas suelen ofrecer mejor rendimiento, integración más completa y la mejor experiencia de usuario en comparación con otras opciones; sin embargo, el desarrollo nativo suele ser también la opción de desarrollo más compleja.”

“... Una aplicación híbrida hace uso tanto de las tecnologías nativas como la web. Partes de ella se comportan como una aplicación nativa, mientras que otras se ejecutan sobre tecnologías web...”[8].

El desarrollo de aplicaciones híbridas facilita la conexión con las API's nativas ofrecidas por cada entorno. De esta manera, accedemos a funcionalidades propias de las aplicaciones nativas como son las notificaciones push, cámara, acceso a las compras de los stores, GPS<sup>2</sup>, sensores.

## **2.2. Fundamentación teórica**

### **2.2.1. Georreferenciación y sistemas de coordenadas**

La georreferenciación se basa en el uso de latitud y longitud de mapa para asignar una ubicación espacial a entidades. Cada elemento que forman parte de una capa de mapa tienen una ubicación geográfica y una extensión específicas que permiten situarlos en la superficie de la Tierra o cerca de ella. La localización precisa de las entidades geográficas es fundamental en un SIG<sup>3</sup>. [9]

Los mapas representan ubicaciones en la superficie de la Tierra que utilizan cuadrículas, graticulas y marcas de graduación con etiquetas de diversas ubicaciones terrestres (tanto en medidas de latitud-longitud como en sistemas de coordenadas proyectadas [como metros de UTM<sup>4</sup>]). Los elementos geográficos incluidos en diversas capas de mapa se trazan en un orden específico (uno sobre otro) para la extensión del mapa determinada.

Los orígenes de datos SIG incluyen ubicaciones de coordenadas disponen de un sistema de coordenadas cartesianas o globales para registrar ubicaciones y formas geográficas. De esta manera, es posible superponer capas de datos SIG sobre la superficie de la Tierra [9].

---

<sup>1</sup>Interfaces de programación de aplicaciones

<sup>2</sup>Sistema de Posicionamiento Global

<sup>3</sup>Sistema de información geográfica

<sup>4</sup>Universal Transverse Mercator (Sistema de coordenadas universal transversal de Mercator)

### **2.2.2. Base de datos móviles**

La arquitectura general de una plataforma móvil es un modelo distribuido formado por computadores fijos, estaciones base y unidades móviles. Los computadores fijos son computadores de uso general que no disponen de medios para comunicarse con las unidades móviles. Las estaciones base disponen de enlaces inalámbricos para conectar con las unidades móviles; son máquinas que actúan de intermediarios entre las unidades móviles y los computadores fijos. Los computadores fijos y las estaciones base están interconectados por medio de una red fija (cableada) de alta velocidad. Las unidades móviles se conectan a las estaciones base mediante enlaces inalámbricos; los enlaces más comunes son el estándar 802.11 (Wi-Fi), el servicio GPRS<sup>5</sup> y la tecnología Bluetooth[10].

Los dispositivos móviles tienen un dominio de movilidad geográfica el cual significa que se pueden desplazar libremente por un espacio conocido, cuyo alcance está determinado por la cobertura de los enlaces inalámbricos [10]. Este sistema se basa en subdominios llamados celdas. Cada subdominio es controlado por una estación base. Este sistema basado en celdas debe garantizar la disposición de información en el dominio de movilidad geográfica.

Guardar datos en una base de datos es ideal para los datos estructurados o que se repiten, como la información de contacto. Las bases de datos SQL<sup>6</sup> en general brindan ayuda para comenzar a utilizar las bases de datos SQLite en Android [11].

CouchDB: Cada documento tiene un nombre único en la base de datos, y CouchDB proporciona una API HTTP<sup>7</sup> RESTful<sup>8</sup> para leer y actualizar (agregar, editar, eliminar) documentos de la base de datos[12].

### **2.2.3. Almacenamiento offline de datos**

La localización de recursos mediante URL<sup>9</sup>, usa la Cache API (parte de los Service Workers). Para el resto de los datos, usa IndexedDB. Las APIs son asíncronas (IndexedDB está basada en eventos y Cache API). IndexedDB se encuentra disponible en todos los ámbitos. IndexedDB admite Observers, que posibilitan una sincronización simple entre pestañas. La File System API se encuentra disponible para Chrome, no es compatible con ningún otro navegador. El File API y Directory Entries API no es lo suficientemente madura o no está estandarizada para una adaptación de una aplicación a gran escala [13].

<sup>5</sup>General Packet Radio Service(Servicio general de paquetes vía radio)

<sup>6</sup>Structured Query Language(Lenguaje de consulta estructurada)

<sup>7</sup>Hypertext Transfer Protocol

<sup>8</sup>Transferencia de Estado Representacional

<sup>9</sup>Uniform Resource Locator

Android y iOS se distribuyen con una versión de SQLite en el sistema operativo del dispositivo, por lo que no es necesario hacer referencia a su propia versión de SQLite [14].

#### **2.2.4. XAMPP**

XAMPP<sup>10</sup> es un open source, dispone de intérpretes para lenguajes de script PHP y Perl , un sistema de gestión de bases de datos MySQL, un servidor web Apache, permite a los diseñadores web y programadores realizar pruebas de trabajo en sus propios ordenadores cuando no se tiene ningún acceso a Internet. XAMPP se utiliza en la actualidad como servidor de sitios web, son necesarias algunas modificaciones para realizar dicha función, es generalmente lo suficientemente seguro. Con el paquete se incluye una herramienta especial para proteger fácilmente las partes más importantes en una página [15].

#### **2.2.5. PostgreSQL**

PostgreSQL es un sistema de administración de base de datos relacional de objetos, es el sistema de base de datos de código abierto más avanzado. PostgreSQL fue desarrollado sobre la base de POSTGRES 4.2 en Berkeley Computer Science Department. Una característica fundamental de PostgreSQL es el control de concurrencias multiversión; o MVCC por sus siglas en inglés. Este método agrega una imagen del estado de la base de datos a cada transacción. Esto nos permite hacer transacciones eventualmente consistentes, ofreciendo ventajas en cuanto al rendimiento.

##### **2.2.5.1. PostGIS**

Para convertir PostgreSQL en una base de datos espacial es necesario agregar el módulo PostGIS para su utilización en Sistema de Información Geográfica [16]. Se publica bajo la Licencia Pública General de GNU<sup>11</sup>.

##### **2.2.5.2. pgAdmin**

pgAdmin es la herramienta de administración de las bases de datos en PostgreSQL. Permite realizar búsquedas SQL, en complemento permite el desarrollo toda base de datos de forma muy fácil e intuitiva; directamente desde la interfaz gráfica [17].

---

<sup>10</sup>X Any System Operative, Apache, MariaDB/MySQL, PHP, Perl

<sup>11</sup>GNU's Not Unix



En transacciones se tiene una mayor escalabilidad debido a que no se requiere realizar bloqueos para su ejecución.

### 2.2.6. API

Los servicios web son creados específicamente para proveer información a una aplicación o sitio web frente a una necesidad. Los programas clientes usan «interfaces de programas de aplicaciones» (API) para comunicarse con los servicios web. Generalmente una API expone un conjunto de datos y funciones para facilitar las interacciones entre los programas y permitir el intercambio de información [18].

#### 2.2.6.1. REST

Las comunicaciones mas ligeras entre servidor-cliente se las realiza con la arquitectura REST<sup>12</sup>, se tiene comunicaciones mantenibles y escalables, REST es un estilo de construcción popular para APIs basadas en la nube. Se denominan API RESTful (Interfaces de programación de aplicaciones) o API REST [19] cuando los servicios Web utilizan la arquitectura REST.

Es un protocolo sin estado, debido a que no se guarda la información en el servidor. Es decir, toda la información será enviada por el cliente en cada mensaje HTTP, consiguiendo un ahorro en variables de sesión y almacenamiento interno del servidor. Ver figura 1

Presenta un conjunto de operaciones bien definidas, siendo las más importantes GET, POST, PUT y DELETE, que se emplea en todos los recursos [18][19].

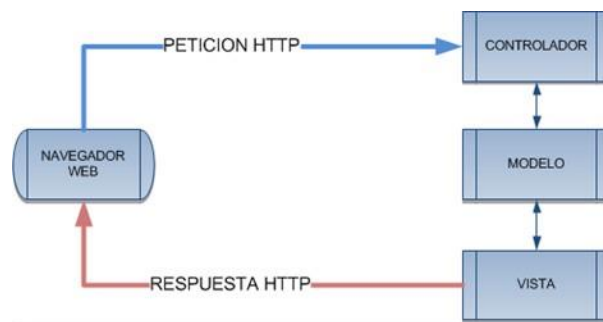


Figura 1: Servicio Web Rest API [1]

<sup>12</sup>Representational state transfer

### 2.2.7. Codeigniter

Codeigniter se basa en el patrón de desarrollo MVC, es un framework PHP<sup>13</sup> para el desarrollo rápido de aplicaciones web. CodeIgniter es rápido, relativamente sencillo y capaz de trabajar en la mayoría de entornos de hosting, es un producto de código libre. Una característica a destacar es su accesibilidad, debido a su uso en la amplia gama de entornos [20]. Base de datos: Codeigniter da un acceso a una clase de generador de consultas. Este patrón permite recuperar, insertar y actualizar información con pocas líneas de código, cada consulta generada por cada adaptador de base de datos es independiente.

- **Consulta** Se puede usar solo para recuperar todos los registros de una tabla

```
// Produces: SELECT * FROM mytable
$query = $this->db->get('mytable');
```

*/\*Consulta por campos específicos\*/*

```
$this->db->select('title, content, date');
$query = $this->db->get('mytable');
// Executes: SELECT title, content, date FROM mytable
```

- **Inserción:** Genera una cadena de inserción basada en los datos que suministra y ejecuta la consulta. Puede pasar una matriz o un objeto a la función.

```
$data = array(
    'title' => 'My title',
    'name' => 'My Name',
    'date' => 'My date'
);
$this->db->insert('mytable', $data);
// Produces: INSERT INTO mytable (title, name, date)
// VALUES ('My title', 'My name', 'My date')
```

- **Actualización:** Este método ejecuta una instrucción REPLACE, que es básicamente el estándar SQL para DELETE + INSERT (opcional), utilizando las teclas PRIMARIA y ÚNICA como factor determinante

---

<sup>13</sup>Hypertext Preprocessor

```

$data = array(
    'title' => 'My title',
    'name'  => 'My Name',
    'date'  => 'My date'
);
$this->db->replace('table', $data);
// Executes: REPLACE INTO mytable (title, name, date)
// VALUES ('My title', 'My name', 'My date')

```

- **Eliminación:** Genera una cadena de eliminación de SQL y ejecuta la consulta

```

$this->db->delete('mytable', array('id' => $id));
// Produces: // DELETE FROM mytable
// WHERE id = $id

```

### 2.2.8. OpenStreetMap

OpenStreetMap es un proyecto colaborativo para crear mapas editables y libres. La comunidad añaden datos sobre estaciones de ferrocarril, senderos, caminos, cafeterías, y muchas entidades a lo largo de todo el mundo. OpenStreetMap se puede usar libremente para cualquier propósito debido a que de código abierto, siempre y cuando se haga referencia a OpenStreetMap y a sus colaboradores. En caso de modificación de los datos y distribución se podrá hacerlo únicamente bajo la licencia establecida por OpenStreetMaps [21].

### 2.2.9. OpenMapTiles

OpenMaptiles proporciona mosaicos vectoriales a partir de datos de OpenStreetMap. El sitio web dispone de paquetes que contienen mosaicos vectoriales, por países individuales y ciudades importantes. Los mosaicos se pueden mostrar con visores de Javascript. El proyecto es de código abierto, bajo la licencia (BSD + CC-BY), reutiliza componentes de código abierto de OpenStreetMap & FOSS, Mapbox y terceros. El proyecto de OpenMapTiles es sucesor de OSM2VectorTiles [22].

### 2.2.10. MapBox

Este complemento de servicios geográficos permite que las aplicaciones accedan a los servicios basados en la ubicación de Mapbox mediante la API de ubicación.

El núcleo está basado en mapas vectoriales implementados en C++, los datos son enviados al dispositivo bajo petición y se interpretan en tiempo real, lo que se traduce en mapas muy rápidos y en visualizaciones mucho más ligeras[23].

Hay una serie de estilos disponibles por defecto (Dark, Light, Street) pero también tienes la posibilidad de diseñar/crear estilos por completo a través de la herramienta Mapbox Studio [24].

#### **2.2.10.1. Formato**

Los mosaicos vectoriales están codificados como Google Protobufs (PBF), que permiten serializar datos estructurados. Para mayor claridad, Mapbox Vector Tiles usa el *.mvt* sufijo de archivo. Los detalles de las especificaciones se estructuran en gran medida en torno a las reglas implementadas en el *.proto* archivo base[25].

#### **2.2.10.2. Relación los archivos PBF de OpenStreetMap con Mapbox Vector Tiles**

Los PBF son un formato, muy parecido a XML<sup>14</sup> y pueden tomar muchas formas. Mapbox Vector Tiles y OSM PBF son archivos protobuf, pero se ajustan a especificaciones completamente diferentes y se utilizan de diferentes manera.

#### **2.2.11. Bootstrap**

Es un conjunto de herramientas de código abierto para desarrollar con HTML<sup>15</sup>, CSS<sup>16</sup> y Javascript, cuyo fin es ayudar a la construcción y diseño de interfaces y componentes adaptativos. Trabaja con el acoplamiento de componentes tales que se apoyan entre sí. cuanto menos componentes dependan unos de otros, será más reutilizable y flexible [26] [27].

Bootstrap es modular una hoja de estilo llamada bootstrap.less incluye los componentes de las hojas de estilo consiste esencialmente en un encadenamiento de hojas de estilo LESS que implementan la multiplicidad de componentes de un objeto.

Algunos de los componentes que se encuentran ya contruidos y disponibles en Bootstrap son:

- Contenedores

---

<sup>14</sup>Extensible Markup Language

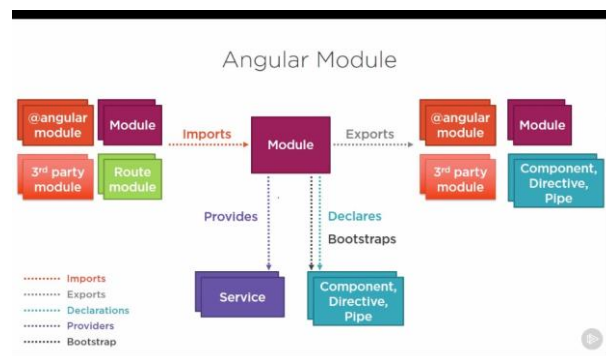
<sup>15</sup>HyperText Markup Language

<sup>16</sup>Hojas de estilo en cascada

- Menús desplegables
- Botones y grupos de botones
- Barra de Navegación
- Barra de progreso
- Etiquetas
- Alertas
- Barras de progreso
- Ventanas modales

### 2.2.12. AngularJS

AngularJS es un marco de desarrollo de JavaScript desarrollado y soportado por Google en conjunto con la comunidad [28][29]. Su objetivo es aumentar las aplicaciones basadas en navegador con capacidad de Modelo Vista Controlador (MVC). La forma de cómo AngularJS aplica el patrón MVC difiere del sentido tradicional siendo que más bien se acerca al concepto de MVVM (Modelo-Vista-Vista Modelo). Ver figura 2 [28][30].



**Figura 2:** Arquitectura de los componentes de AngularJS  
Fuente: Medium[2].

Anteriormente en la parte Front-End de las aplicaciones web sólo teníamos a jQuery para ayudarnos con el código JavaScript del cliente. Había la posibilidad de manipular el DOM de una forma más sencilla, pero no sin tener un patrón a seguir. AngularJS bajo directivas y atributos permite extender el vocabulario HTML, manteniendo la semántica y sin necesidad de usar bibliotecas jQuery [28].

### 2.2.12.1. Arquitectura de AngularJS

- **Vista:** Es una presentación de datos en un formato particular, desencadenada por la decisión del controlador de presentar los datos. Son sistemas de plantillas basadas en scripts, como JSP, ASP, PHP y fáciles de integrar con la tecnología AJAX.[31]
- **Controlador:** El controlador responde a las entradas del usuario y realiza interacciones en los objetos del modelo de datos. El controlador recibe la entrada, la valida y luego realiza operaciones comerciales que modifican el estado del modelo de datos[2].
- **Modelo** El modelo es responsable de gestionar los datos de la aplicación. Responde a la solicitud de vista y a las instrucciones del controlador para actualizarse[28].

### 2.2.13. Apache Cordova

Es un framework de desarrollo móvil open source. Permite usar JavaScript, HTML5, CSS3 que son tecnologías web destinadas para desarrollo web multiplataforma, omitiendo el desarrollo nativo de cada plataforma móvil. Las aplicaciones se ejecutan dentro de un WebView para cada plataforma y dependen de APIs para acceder características de cada dispositivo como sensores, datos y estado de la red [32].

La aplicación se implementa como una página web, que utiliza componentes javascript, css y otros recursos necesarios de forma predeterminada, la aplicación se ejecuta dentro de un envoltorio de aplicación nativa. Los componentes nativos a través de plugins en javascript se utilizan para el proceso, existe una amplia comunidad sobre librerías javascript que proporcionan funciones adicionales disponibles en NPMJS [33].

### 2.2.14. Ionic Framework

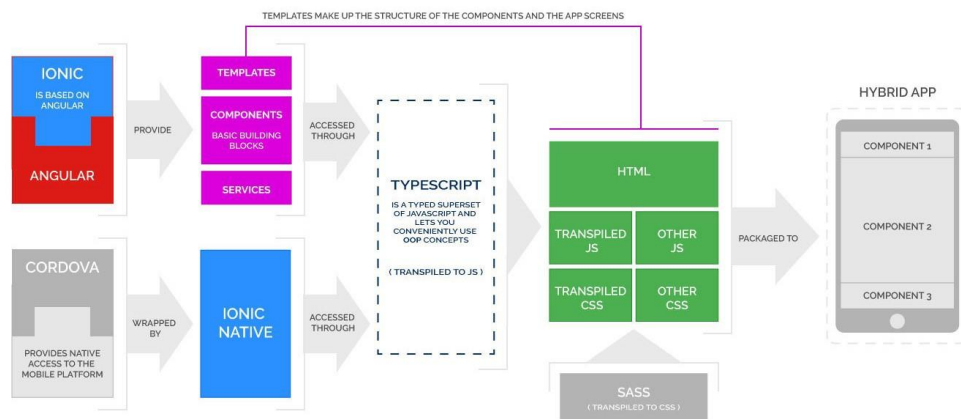
Ionic es un framework open source para el desarrollo de aplicaciones híbridas que permite crear aplicaciones multiplataforma que utiliza HTML5, CSS, componentes Javascript. Construido con Sass y optimizado para AngularJS que es un indicador de aplicaciones robustas, rápidas y escalables, creado por Max Lynch, Ben Sperry y Adam Bradley de Drifty Co. en 2013 [3].

Las aplicaciones pueden construirse con estas tecnologías web y luego distribuirse a través de tiendas de aplicaciones nativas para ser instalado en dispositivos aprovechando Cordova. Ver figura 3.

Ionic se basa en Angular el cual proporciona componentes personalizados y métodos para interactuar con ellos. Se basa en el marco de la interfaz de usuario SASS<sup>17</sup>, que está diseñado y optimizado específicamente para sistemas operativos móviles que también proporciona muchos componentes de interfaz de usuario para crear aplicaciones móviles robustas. Ionic aprovecha las transiciones y transformaciones de CSS para la animación como una forma de aprovechar la GPU<sup>18</sup> y maximizar el tiempo de procesador disponible.

Ionic permite a los desarrolladores de aplicaciones móviles construir una base de código única y usar en todas las plataformas principales. Ionic es bien conocido para el desarrollo de aplicaciones móviles de alto rendimiento, utiliza pautas de la interfaz de usuario de la aplicación nativa y los SDK<sup>19</sup> nativos.

### 2.2.15. Arquitectura Ionic



**Figura 3:** Arquitectura aplicaciones Ionic.  
Fuente: Ionic Framework [3].

#### 2.2.15.1. Estructura del proyecto Ionic

- **node modules:** La carpeta node modules se genera automáticamente al instalar las dependencias npm con “npm install”. Este comando explora el archivo package.json para todos los paquetes que necesitan ser instalados.
- **platforms:** En esta carpeta se genera los proyectos nativos para cada plataforma que se ha añadido anteriormente. Si se ha añadido la plataforma IOS y Android se creará una carpeta llamada ios y otra llamada android y

<sup>17</sup>Syntactically Awesome Style Sheets

<sup>18</sup>Graphics processing unit

<sup>19</sup>Kit de desarrollo de software

se generará un proyecto nativo con los archivos de su respectiva estructura, sea un proyecto de Android Studio o un proyecto XCode.

- **plugins:** Contiene la lista plugins con su versión que se han instalado en el proyecto
- **src:** Esta es la carpeta más importante y donde se realiza la mayor parte del trabajo. En ella se encuentran los archivos con el contenido de la aplicación, donde se define las pantallas, el estilo y el comportamiento que tendrá la aplicación.  
Dentro de la carpeta src se encuentra la carpeta que contiene a las páginas. El archivo **home.html** que contiene la plantilla html de la página. El archivo **home.scss** que contiene el archivo sass donde podremos modificar el estilo de los componentes de la página. El archivo **home.ts** que es el archivos typescript que contiene el controlador de la página, donde definiremos el comportamiento de la misma.
- **Ionic.config.json:** Contiene información básica sobre la configuración del proyecto
- **package.json::** Contiene paquetes y dependencias de nodeJS
- **tsconfig.json y tslint.json:** Son archivos que contienen información necesaria a la hora de compilar TypeScript

#### 2.2.16. Mapas Raster

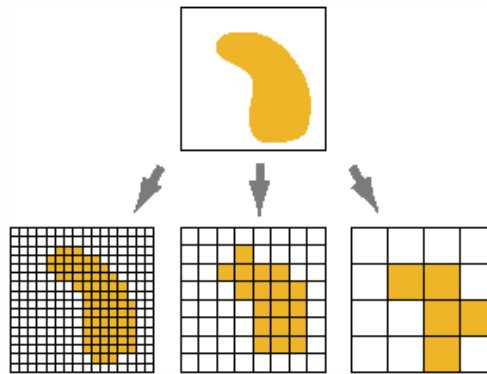
Los mapas raster constan de una matriz de celdas (o píxeles) cada celda (dadas por filas y columnas) contiene un valor que representa información, con el propósito de almacenar puntos, líneas, polígonos y superficies de manera uniforme. Para un área determinada, el cambio de celdas a la mitad del tamaño actual requerirá más espacio de almacenamiento, dependiendo del tipo de datos y de las técnicas de almacenamiento utilizadas. Ver figura 4.

La dimensión de las celdas puede ser tan grande o pequeña como sea necesario para representar la superficie dada por un origen de datos raster.

#### 2.2.17. Mapas Vectoriales

Los mapas vectoriales se tiene la posibilidad de ampliar el tamaño a voluntad sin sufrir la pérdida de calidad que sufren los mapas de bits. Como ventaja, permiten modificar (mover, estirar y retorcer) de manera relativamente sencilla [34].





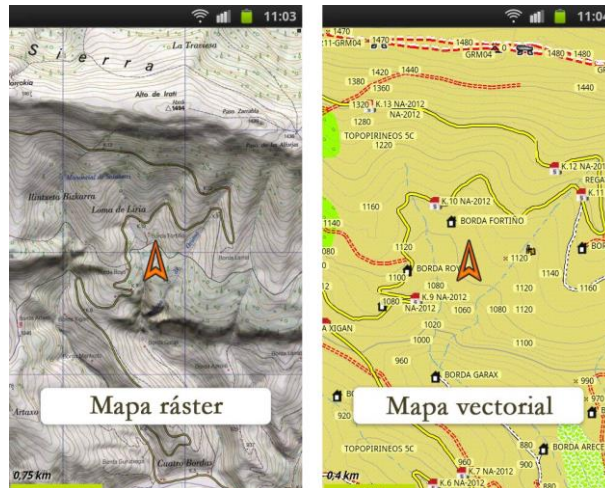
**Figura 4:** Entidad poligonal ráster.  
Fuente: SIG [4].

### 2.2.18. Raster vs Vectorial

En base del análisis de las propiedades topológicas son importantes para la consideración de implementación. El modelo de datos vectorial es la dominante opción, a pesar de su estructura precisa, es compleja y esto puede demorar el proceso. Por lo cual, si las propiedades topológicas juegan un papel importante, es mucho más rápido, sencillo y eficaz el uso del formato raster. Ver figura 5.

Las ventajas del modelo raster incluyen la facilidad y la velocidad en la ejecución de los visores, Por otra parte, la inexactitud que depende de la valor de los datos y la gran cantidad de espacio que requiere para el almacenamiento de los datos.

Si se hace en capas raster un zoom excesivo aparecerán los bordes de estas como advertencia de que la profundidad del zoom es excesiva, en el caso del formato vectorial no tenemos este mecanismo de advertencia y en muchos casos se fuerzan los zoom para obtener una precisión completamente nítida. Una distribución de datos vectorial se utiliza cuando hay que proyectar más de un atributo en un mismo espacio. El uso de la estructura raster se debe crear una capa por cada atributo.



**Figura 5:** Imagen vectorial vs ráster  
 Fuente: Mapas para OruxMaps[5]

### 2.3. Propuesta de Solución

El presente proyecto propone el desarrollo de un sistema de gestión de datos georreferenciales en el Ecuador, inspirado en la forma en que funcionan aplicaciones como Google Maps y basado en la biblioteca software libre para el manejo de mapas que provee Mapbox. Adicionalmente, con este trabajo se pretende el manejo de recursos disponibles de forma local (para que la aplicación pueda funcionar sin que el equipo móvil tenga conexión a Internet), por lo que se requiere la implementación de un mecanismo de sincronización con los datos del servidor que funcione cuando el aplicativo se encuentre en modo de conexión.

## CAPÍTULO 3

### Metodología

#### 3.1. Modalidad de la investigación

**Investigación aplicada:** Se realizará una investigación aplicada ya que se empleará y aplicará todo lo estudiado durante la carrera para alcanzar a satisfacer las necesidades del desarrollo del presente proyecto.

**Investigación Bibliográfica-Documental:** Se realizará una investigación bibliográfica - documental para poder recopilar información sobre el desarrollo basado en componentes que servirá de apoyo para la realización del proyecto.

La investigación tendrá la modalidad de campo porque se buscará obtener la información de la variable Almacenamiento de datos geo referenciales offline y de la variable con sincronización de datos almacenados en el lugar mismo en que ocurre.

**Investigación exploratoria:** Se encuentra dentro de esta categoría al no haberse realizado en la ciudad de Ambato trabajos de este tipo.

**Investigación descriptiva:** Por medio de la recolección, análisis y conclusiones por medio de encuestas se llegará a identificar la relación entre la variable independiente y la variable dependiente.

**Investigación explicativa:** Dentro de la investigación presente se intentará comprobar la siguiente afirmación: ¿El modelo para almacenar datos de manera offline para posterior sincronización es deficiente en la aplicación móvil multiplataforma a implementar?

### 3.2. Recolección de información

**Tabla 1:** Recolección de información.

Fuente: Elaborado por el investigador

Preguntas Básicas	Explicación
<b>¿Para qué</b> <b>¿De qué personas u objetos?</b> <b>¿Sobre qué aspectos?</b>	Para alcanzar los objetivos
	En la zona Ambato-Ecuador
	Indicadores <ul style="list-style-type: none"> <li>■ Información sobre almacenamiento de datos georreferenciales de interés</li> <li>■ Porcentaje de utilidad del modelo de almacenamiento y sincronización.</li> <li>■ Información sobre plataformas de consulta de datos georreferenciales</li> </ul>
<b>¿Quién, Quiénes?</b>	Investigador Víctor Bautista
<b>¿Cuándo?</b>	14/01/2019
<b>¿Dónde?</b>	Facultad de Ingeniería en Sistemas Computacionales e Informáticos
<b>¿Cuántas veces?</b>	Una
<b>¿Qué técnicas de recolección?</b>	Encuesta
<b>¿Con qué?</b>	Cuestionario
<b>¿En qué situación?</b>	En condiciones normales

### 3.3. Procesamiento y análisis de datos

#### Procesamiento de la información

1. Revisión crítica, de la información recogida; es decir, limpieza de la información defectuosa: contradictoria, incompleta, no pertinente, etc.
2. Tabulación o cuadros según variables: cuadros de una sola variable, cuadro con cruce de variables, etc.
3. Manejo de la información (reajuste de cuadros con casillas vacías o con datos tan aducidos cuantitativamente, que no influyen significativamente en los análisis).

#### Análisis de Resultados

1. Análisis de los resultados estadísticos.

2. Interpretación de los resultados, con apoyo del marco teórico, en el aspecto pertinente.
3. Establecimiento de conclusiones y recomendaciones.

### **3.4. Desarrollo del proyecto**

La metodología prototipo evolutivo fue aplicada para el desarrollo del aplicativo, se tomó como modelos principales aplicaciones y visores; MapboxGL y Google Maps. El desarrollo del proyecto estuvo previsto en cinco etapas.

- **Etapa 1.** Identificación de Requerimientos Conocidos
  - Análisis de requerimientos del Sistema de información.
    - Levantamiento de requerimientos funcionales.
    - Levantamiento de requerimientos no funcionales.
- **Etapa 2.** Desarrollo del Modelo
  - Diseño de arquitectura de la aplicación multiplataforma.
    - Diseño de modelo físico de base de datos.
    - Diseño del API-REST.
    - Diagramas de actividades de la aplicación.
    - Diagramas de secuencias de la aplicación.
- **Etapa 3.** Desarrollo de la aplicación.
  - Especificación de casos de uso.
  - Producción de software.
    - Elaboración de diagramas de clases
    - Diagrama de clases de la programación del cliente
    - Diagrama de clases de la programación del servidor
- **Etapa 4.** Iteración.
  - Requisitos no funcionales.
    - Usabilidad.
    - Estilos.
    - Diseño.
    - Mantenibilidad

- o* Seguridad.
  - o* Rendimiento.
- **Etapa 5.** Prototipado Terminado.
  - o* Transición.
    - o* Pruebas de funcionalidad.

## CAPÍTULO 4

### Desarrollo de la propuesta

Para concretar el desarrollo de la propuesta, se tomó como guía a partir del segundo objetivo específico planteado; por cada objetivo específico se llevaron a cabo pasos concretos que se detallan a continuación:

#### 4.1. Análisis de requerimientos del sistema de información

A partir de las pruebas realizadas en las aplicaciones de Mapbox y el uso del componente de Google Maps, se analizó las características con las que cuenta cada plataforma para la gestión de datos georreferenciados y se determinó que, en ambos casos, se utilizan mapas vectoriales, con los que se tienen mejores resultados que con los mapas raster. En el caso de Google Maps, se usan teselas vectoriales de su propio servidor, basado en el formato del estándar abierto creado por la empresa Mapbox, bajo licencia Creative Commons Attribution 3.0 US [35]. En el caso de Mapbox, como tal, utiliza teselas ráster de OpenStreetMap que luego renderiza y convierte a teselas vectoriales bajo su propio formato. Tanto Google Maps, como Mapbox, utilizan estilos aplicados a los datos de las teselas. Por ejemplo, en caso de ser una edificación, con el estilo aplicado se define el grosor de línea del polígono, color y el *zoom* en el cual debe aparecer el objeto. En el caso de avenidas, calles y subcalles, de la misma manera, se define el color, grosor de línea y el *zoom* en el cual debe aparecer el nombre del objeto.

Para ambas plataformas, se analizó la forma en que quedan disponibles las teselas vectoriales en modo *offline*. En el caso de Google Maps, el componente almacena las teselas en la caché del dispositivo, las que quedan disponibles hasta que el móvil se reinicia. Por su parte, al utilizar el SDK de Mapbox, el almacenamiento se da de la misma manera, solo que, en este caso, la caché queda disponible hasta al cierre de la aplicación. Ninguna de las dos garantiza la persistencia de los datos en el dispositivo.

Como resultado, se decidió enfocar el presente trabajo en el desarrollo de una aplicación móvil multiplataforma con gestión de datos georreferenciados utilizando teselas vectoriales en modo *offline* (sin depender de un GeoServer u otro componente con conexión a la red) pero aplicando un estilo sobre los mosaicos vectoriales del tipo que utiliza Google Maps. En este caso se decidió optar por uno de los estilos que provee MapBox y que también se distribuye con licencia

de software libre. De esta forma, este trabajo quedó enfocado en el desarrollo de aplicaciones móviles utilizando software libre, así como en la optimización de recursos para dejarlos disponibles en el dispositivo y no requerir una conexión a Internet para poder usar el aplicativo.

#### **4.1.1. Levantamiento de requerimientos funcionales**

La conexión a la red se requerirá exclusivamente para sincronizar la información recopilada con la que haya sido registrada en la aplicación pero utilizando otros dispositivos (ya sea por el mismo o por otros usuarios). A continuación se listan los requisitos funcionales que se definieron para la implementación de la aplicación. Estos son:

- Registrar usuario.
- Autenticar usuario del sistema.
- Registrar puntos de interés.
- Modificar o eliminar puntos de interés, siempre que estos sean añadidos por el usuario activo en el sistema.
- Mostrar los datos georreferenciados anotados utilizando la aplicación.
- Sincronizar los datos georreferenciados anotados en diferentes dispositivos donde esté instalada la aplicación.
- Buscar datos georreferenciados de interés para el usuario activo en la aplicación.
- Compartir datos georreferenciados anotados en la aplicación a través de Facebook, WhatsApp e Instagram.

#### **4.1.2. Levantamiento de requerimientos no funcionales**

Entre los requerimientos no funcionales, del aplicativo desarrollado, se tuvieron en cuenta:

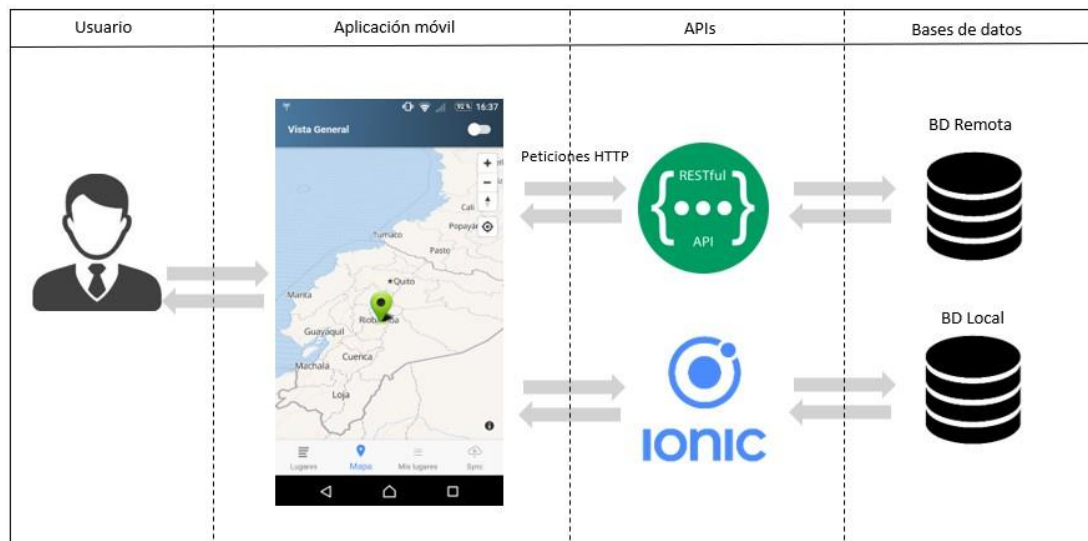
- Garantizar la gestión de tantos puntos como requiera el usuario, teniendo como límite la capacidad de memoria del dispositivo que utilice.
- Utilizar *Bootstrap* para garantizar que el diseño de interfaz del aplicativo sea homogéneo a través de las páginas de usuario.
- El aplicativo debe tener la capacidad de descargar la imagen de un punto.



- Garantizar la protección de las contraseñas de los usuarios del sistema.
- El aplicativo debe ser fácil de usar.

#### 4.2. Diseño de arquitectura de la aplicación multiplataforma

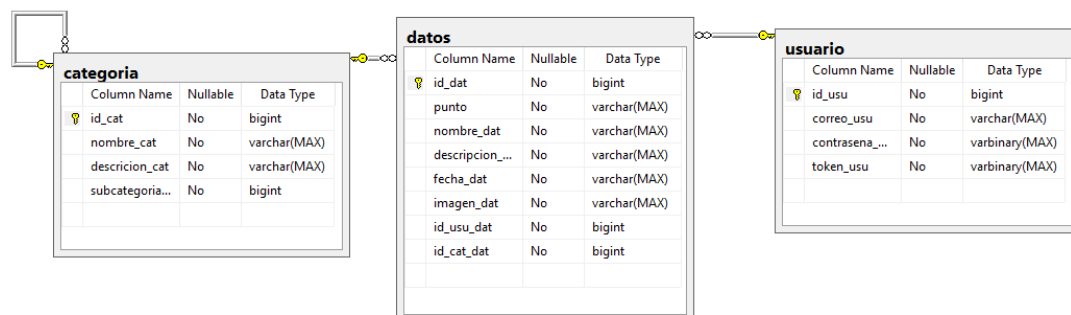
A continuación se describe el diseño de la arquitectura de la aplicación implementada, resaltando los componentes fundamentales que se tuvieron en cuenta, en concordancia con los patrones arquitectónicos fundamentales de la plataforma Ionic.



**Figura 6:** Arquitectura de la aplicación  
Fuente: Elaborado por el investigador

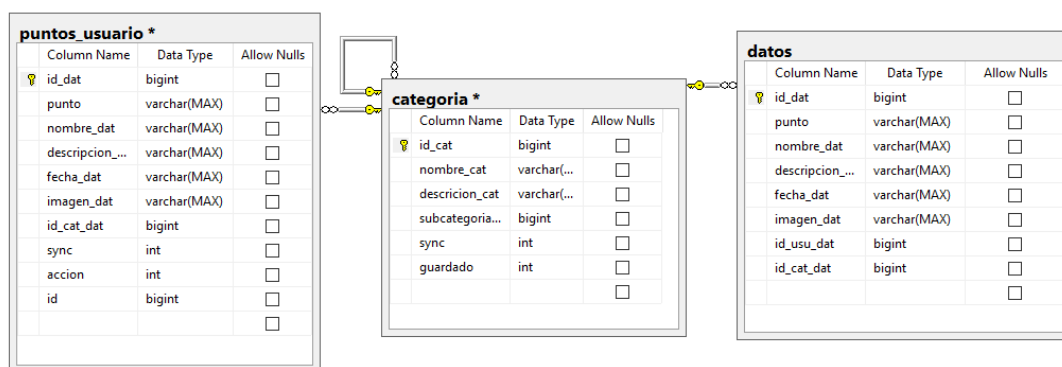
##### 4.2.1. Diseño de modelo físico de base de datos

El diseño físico de la base de datos remota tiene en cuenta la información que se guarda de los usuarios del sistema y de los datos que estos registran. Ver figura 7.



**Figura 7:** Modelo físico de la base de datos remota  
Fuente: Elaborado por el investigador

El diseño físico de la base de datos local tiene en cuenta la información proveniente de la base de datos remota, en la tabla datos se almacenan los puntos de todos los usuarios, mientras en la tabla puntos\_usuario almacena los puntos añadidos sin conexión, así como, actualización de puntos existentes y su eliminación, tal y como se muestra en la figura a continuación.



**Figura 8:** Modelo físico de la base de datos local  
Fuente: Elaborado por el investigador

#### 4.2.2. Diseño del API-REST

Para la implementación de la capa de servicios RESTful se utilizó la biblioteca de código diseñado por Phil Sturgeon para Codeigniter. Actualmente, Chris Kacerguis es quien le da soporte a esta biblioteca [36].

El archivo de configuración de esta biblioteca *application/config/database.php* permite configurar la conexión con la base de datos Postgresql, tal y como se muestra en el código que aparece a continuación:

```
$db['default'] = array(
    'dsn' => '',
    'hostname' => 'localhost',
    'username' => '*****',
    'password' => '*****',
    'database' => 'ambato_app_bd',
    'dbdriver' => 'postgre',
    'dbprefix' => '',
    'pconnect' => FALSE,
    'db_debug' => (ENVIRONMENT !== 'production'),
    'cache_on' => FALSE,
    'cachedir' => '',
    'char_set' => 'utf8',
```

```

'dbcollat' => 'utf8_general_ci',
'swap_pre' => '',
'encrypt' => FALSE,
'compress' => FALSE,
'stricton' => FALSE,
'failover' => array(),
'save_queries' => TRUE,
'port'=> 5432
);

```

Dentro de la carpeta ***application/controllers*** se añadieron los archivos con los métodos GET, POST, PUT para la persistencia de datos. Para cada clase que deba realizar acciones en la base de datos, se debe especificar que heredan de ***REST\_Controller***, y en el fichero correspondiente a cada clase que se define, se deben añadir encabezados sobre la petición HTTP que se va a realizar y habilitar el CORS, tal y como se muestr a continuación:

```

require_once( APPPATH.'/libraries/REST_Controller.php' );
use Restserver\libraries\REST_Controller;

```

```

class Login extends REST_Controller {
    public function construct() {
        /* Agregar a la cabecera de petición CORS */
        header("Access-Control-Allow-Methods:
        PUT,
        GET,
        POST,
        DELETE,
        OPTIONS");
        header("Access-Control-Allow-Headers:
        Content-Type,
        Content-Length,
        Accept-Encoding");
        header("Access-Control-Allow-Origin: *");
        /* Cargar el archivo de configuración */
        parent::construct();
        $this->load->database();
    }
}

```

```

    }
}

```

Para la especificación del tipo de petición a realizar en Codeigniter, al final de la función se debe detallar como se indica en la tabla 2.

**Tabla 2:** Ejemplo. Definición de métodos en Codeigniter  
Fuente: Elaborado por el investigador

Método	Ejemplo
GET	<i>obtener_todos_dat_get()</i>
POST	<i>registrar_usu_post()</i>
PUT	<i>actualizar_clave_put()</i>

En el archivo ***application/controllers/Login.php*** se han especificado los métodos para gestionar usuarios.

**Método registrar usuario:** El método verifica la existencia de las variables enviadas vía POST. Si existen, se forma la consulta sql apoyándose en la biblioteca propia de Codeigniter para realizar consultas. Si el usuario existe, se retorna un mensaje de error, detallando que el usuario ya se encuentra registrado; caso contrario, se cifra la contraseña mediante el algoritmo *PASSWORD\_BCRYPT* y se genera un token con el hash( 'ripemd16') a partir del correo ingresado. Posteriormente se forma una nueva sentencia sql para insertar un nuevo registro, el cual retorna un mensaje de éxito o error.

```

public function registrar_usu_post() {
    $data = $this->post();
    if( !isset( $data["correo_usu"] ) ||
        strlen( $data['correo_usu'] ) == 0 ||
        !isset( $data["contrasena_usu"] ) ||
        strlen( $data['contrasena_usu'] ) == 0 ) {
        $respuesta = array(
            'error' => TRUE,
            'mensaje' => "Correo y/o contraseña
                        no pueden estar vacíos",
            'correo' => $data["correo_usu"],
            'pass' => $data["contrasena_usu"]
        );
        $this->response($respuesta, REST_Controller::HTTP_BAD_REQUEST);
    }
}

```

```

        return;
    }
    // Verifica si existe
    $correo_usu=$data['correo_usu'];
    $condiciones = array('correo_usu' => $correo_usu);
    $this->db->where( $condiciones );
    $query = $this->db->get('usuarios');
    $existe = $query->row();
    if( $existe){
        $respuesta = array(
            'error' => TRUE,
            'mensaje'=> "El usuario ya existe"
        );
        $this->response( $respuesta );
        return;
    }
    // En este ámbito existe el usuario
    // y sus datos están correctos
    // Resetea el query
    $this->db->reset_query();
    $contrasena_encriptada=password_hash($data['contrasena_usu'],
                                           PASSWORD_BCRYPT);
    $token = hash( 'ripemd160', $data['correo_usu'] );
    $insertar= array('correo_usu' => $data["correo_usu"],
                    'contrasena_usu' => $contrasena_encriptada,
                    'token'=> $token );
    $this->db->insert( 'usuarios', $insertar );
    $id_usu=$this->db->insert_id();
    //Verificar si se registró
    $verificar=($this->db->affected_rows() != 1) ? false : true;
    if($verificar){
        $respuesta= array(
            'error'=>FALSE,
            'mensaje'=>'Registro exitoso',
            'token'=> $token,
            'id_usu'=>$id_usu
        );
    }
}

```

```

else{
    $respuesta= array(
        'error'=>TRUE,
        'mensaje'=>'Hubo un error intenta más tarde'
    );
}
$this->response ($respuesta);
}

```

**Método autentificar usuario:** El método verifica la existencia de las variables enviadas vía POST, si no existen las variables retorna un mensaje solicitando las variables requeridas, caso contrario arma una sentencia sql apoyandose de la librería para gestionar consultas de Codeigniter, envía el correo que es único en existencia en la tabla usuario, si existe retorna el contraseña, token y el Id de usuario, la contraseña será verificada con la variable almacenada a través de un hash, caso contrario retorna un mensaje de error.

```

public function obtener_usu_post(){
    $data = $this->post();
    //Verificar si las variables son enviadas
    if ( !isset($data['correo_usu']) OR
        !isset($data['contrasena_usu']) ) {

        $respuesta = array(
            'error' => TRUE,
            'mensaje'=> 'Los campos correo y/o
            contraseña no pueden estar vacíos'
        );
        $this->response( $respuesta,
            REST_Controller::HTTP_BAD_REQUEST );
        return;
    }
    // Verificar si el usuario existe
    $condiciones = array('correo_usu' => $data['correo_usu']);
    $this->db->where($condiciones);
    $query = $this->db->get('usuarios');
    $usuario = $query->row();
    if ( !isset($usuario) ) {
        $respuesta = array(

```

```

        'error' => TRUE,
        'mensaje'=> 'El usuario no existe'
    );
    $this->response( $respuesta );
    return;
}
//Verificar contraseña con un hash
$contrasena_verificada=password_verify($data['contrasena_usu'],
$usuario->contrasena_usu);
if($contrasena_verificada){
    $respuesta = array(
        'error' => FALSE,
        'mensaje'=> 'Login',
        'token' => $usuario->token,
        'id_usu'=>$usuario->id_usu
    );
}
else{
    $respuesta = array(
        'error' => TRUE,
        'mensaje'=>'Usuario y/o contraseña incorrectos'
    );
}
$this->response( $respuesta );
}

```

**Método para consultar todos los datos:** El método retorna un array en formato json con la información de la tabla “datos”.

```

public function obtener_todos_dat_get()
{
    $resultado = array();
    $punto=array();
    $query = $this->db->query('SELECT id_dat,
                                nombre_dat,
                                imagen_dat,
                                punto_dat,
                                descripcion_dat,
                                fecha_dat,

```

```

                id_usu_dat,
                id_cat_dat
            FROM datos
            ORDER BY fecha_dat desc ' );
foreach( $query->result() as $row ){
    $punto= explode( ',', str_replace('"', "",
        str_replace("(", "", $row->punto_dat )));
    $datos = array(
        'id_dat'=>$row->id_dat,
        'imagen_dat' => $row->imagen_dat,
        'nombre_dat'=> $row->nombre_dat,
        'lat'=> $punto[0],
        'lng'=> $punto[1],
        'descripcion_dat'=> $row->descripcion_dat,
        'fecha_dat'=>$row->fecha_dat,
        'id_usu_dat'=>$row->id_usu_dat,
        'id_cat_dat'=>$row->id_cat_dat
    );
    array_push( $resultado, $datos );
}
$response = array(
    'error' => FALSE,
    'datos'=> $resultado
);
$this->response( $respuesta );
}

```

**Método para agregar un punto:** El método recibe a través de la dirección URL el Id del usuario, y vía POST recibe una variable con los datos a ingresar. Para identificar registros se ha utilizado corchetes “[ ]”; en cuanto a separar campos se ha definido el apóstrofe “’”. El método verifica la existencia de las variables. En caso de no existir variables, retorna un mensaje de error y una variable de tipo *boolean* en *false*. Si pasa el control de existencia de la variable, procede a separar los registros enviados en forma de cadena, para luego iterar por cada registro. El proceso verifica qué tipo de imagen es la que se debe almacenar en la carpeta. En la base de datos se almacena el nombre del archivo que es generado en base a la fecha y hora en que se agregó. Así, el archivo no entra en conflicto con el nombre de otros archivos de imagen almacenados en la carpeta. Posteriormente es almacenado cada registro. Si el proceso termina con éxito, el método retorna



una variable con valor *true* y un mensaje; caso contrario, retorna un mensaje de error y la variable con valor *false*.

```
public function guardar_punto_post($id_usu) {
    $data = $this->post();
    //Verificar la existencia de las variables
    if( !isset($data['puntos']) || strlen($data['puntos']) == 0) {
        $respuesta = array(
            'error' => TRUE,
            'mensaje' => "Faltan datos"
        );
        $this->response($respuesta,
            REST_Controller::HTTP_BAD_REQUEST );
        return;
    }
    $puntos = explode('[]', $data['puntos']);
    foreach($puntos as &$punto_detalle){
        $variables = explode('"', $punto_detalle );
        $punto = "(".$variables[0].", ".$variables[1].")";
        $nombre_dat = $variables[2];
        $imagen_dat = $variables[3];
        $descripcion_dat = $variables[4];
        $fecha_dat = $variables[5];
        $id_cat_dat = $variables[6];
        $id_usu_dat = $id_usu;
        //Guardar la imagen en la carpeta
        if(strpos($imagen_dat, 'data:image/jpeg;base64') ||
            strpos($imagen_dat, 'data:image/jpg;base64')) {
            $imagen_dat = str_replace('data:image/jpeg;base64,',
                '', $imagen_dat);
            $imagen_dat = str_replace('data:image/jpg;base64,',
                '', $imagen_dat);
            $extension = 'jpg';
        }
        if(strpos($imagen_dat, 'data:image/png;base64')) {
            $imagen_dat = str_replace('data:image/png;base64,',
                '', $imagen_dat);
            $extension = 'png';
        }
    }
}
```

```

$imagen_dat = str_replace(' ', '+', $imagen_dat);
$image = base64_decode($imagen_dat);
$filename = $nombre_dat . date("d-m-Y-h-i-s") .
            $id_usu_dat . '.' .
            $extension;
            //Renombrar el nombre de la imagen
            //basado en la fecha y hora
//$filename = $nombre_dat . $id_usu_dat . '.' .
            $extension;
$path = 'uploads/img/';
$numeros=file_put_contents($path. $filename, $image);
//Restablecer el query
$this->db->reset_query();
//Insertar cada elemento
$insertar = array('punto_dat' => $punto,
'nombre_dat'=>$nombre_dat,
                'imagen_dat'=>$filename,
                'descripcion_dat' => $descripcion_dat,
                'fecha_dat' => $fecha_dat,
                'id_usu_dat' => $id_usu_dat,
                'id_cat_dat' => $id_cat_dat);
$this->db->set($insertar);
$this->db->insert('datos');
$verificar=($this->db->affected_rows() != 1) ? false : true;
if($verificar){
    $respuesta= array(
        'error'=>FALSE,
        'mensaje'=>'Registro exitoso'
    );
}
else{
    $respuesta= array(
        'error'=>TRUE,
        'mensaje'=>'Hubo un error intenta más tarde'
    );
}
$this->response($respuesta);
}}

```

**Método para modificar un punto:** El método recibe como parte de la cadena el Id del registro a actualizar y un número que permitirá verificar si se ha modificado la imagen del registro. En primera instancia se verifica los datos enviados vía POST; el mensaje de error se retorna en caso de no haber la variable. Si esta existe, se procede a separar los campos que están concatenados con apóstrofes. Si el número que se recibió por medio de la URL es *1* entonces el método realiza el proceso de verificar el tipo de imagen. Si el número que se recibió por medio de la URL es *0*, entonces se entiende que no ha habido modificación de la imagen y, por tanto, no es necesario actualizar la que ya existe en el servidor. Posteriormente se almacena en la carpeta de imágenes. Si el registro es exitoso, se retorna una variable de tipo *boolean* con el valor de **true** y un mensaje. En caso de haber un error un *boolean* con el valor de **false** y el correspondiente mensaje de error.

```
public function actualizar_punto_post($id,$numero){
    $data = $this->post();
    //Verificar la existencia de las variables
    if( !isset($data['puntos']) ||
        strlen($data['puntos'])==0){
        $respuesta = array(
            'error' => TRUE,
            'mensaje'=> "Faltan datos"
        );
        $this->response( $respuesta,
            REST_Controller::HTTP_BAD_REQUEST );
        return;
    }
    $puntos = explode( '[]', $data['puntos'] );
    foreach($puntos as &$punto_detalle){
        $variables=explode( "'", $punto_detalle );
        $punto="(".$variables[0].",".$variables[1].")";
        $nombre_dat=$variables[2];
        $imagen_dat=$variables[3];
        $descripcion_dat=$variables[4];
        $fecha_dat=$variables[5];
        $id_cat_dat=$variables[6];
        if($numero==1){
            //Guardar la imagen en la carpeta
            if(strpos($imagen_dat,'data:image/jpeg;base64') ||
```

```

    strpos($imagen_dat,'data:image/jpg;base64')){
$imagen_dat = str_replace('data:image/jpg;base64,',
'', $imagen_dat);
$imagen_dat = str_replace('data:image/jpeg;base64,',
'', $imagen_dat);
$extension='jpg';
    }
    if(strpos($imagen_dat,'data:image/png;base64')){
$imagen_dat = str_replace('data:image/png;base64,',
'', $imagen_dat);
$extension='png';
    }
    $imagen_dat = str_replace(' ', '+', $imagen_dat);
    $image = base64_decode($imagen_dat);
    $filename = $nombre_dat . date("d-m-Y-h-i-s") .
$id_usu_dat.'.' . $extension;
    //Renombrar el nombre del archivo
    // basado a la fecha y hora
    //$filename = $nombre_dat. $id_usu_dat.'.' .
$extension;
    $path='uploads/img/';
    $numero=file_put_contents($path. $filename, $image);
}

    //Restablecer el query
    $this->db->reset_query();
    //Insertar cada elemento
    if($numero==0){
$actualizar = array('punto_dat' => $punto,
'nombre_dat'=>$nombre_dat,
'descripcion_dat' => $descripcion_dat,
'fecha_dat' => $fecha_dat,
'id_cat_dat' => $id_cat_dat);
    }
    else{
$actualizar = array('punto_dat' => $punto,
'nombre_dat'=>$nombre_dat,
'imagen_dat'=>$filename,
'descripcion_dat' => $descripcion_dat,

```

```

'fecha_dat' => $fecha_dat,
'id_cat_dat' => $id_cat_dat);
}

$this->db->set($actualizar);
$this->db->where('id_dat',$id);
$this->db->update('datos');
$verificar=($this->db->affected_rows() != 1) ? false : true;
if($verificar){
    $respuesta= array(
        'error'=>FALSE,
        'mensaje'=>'Registro actualizado'
    );
}
else{
    $respuesta= array(
        'error'=>TRUE,
        'mensaje'=>'Hubo un error intenta más tarde'
    );
}
$this->response($respuesta);
}
}

```

**Método para eliminar un punto:** El método recibe, por URL, el Id del registro a eliminar de la base de datos.

```

public function eliminar_punto_post($id){
    $this->db->where('id_dat',$id);
    $this->db->delete('datos');
    $verificar=($this->db->affected_rows() != 1) ? false : true;
    if($verificar){
        $respuesta= array(
            'error'=>FALSE,
            'mensaje'=>'Registro eliminado'
        );
    }
    else{
        $respuesta= array(
            'error'=>TRUE,

```

```

        'mensaje'=>'Hubo un error intenta más tarde'
    );
}
$this->response($respuesta);
}

```

**Método para listar categorías:** El método devuelve la lista de categorías almacenadas.

```

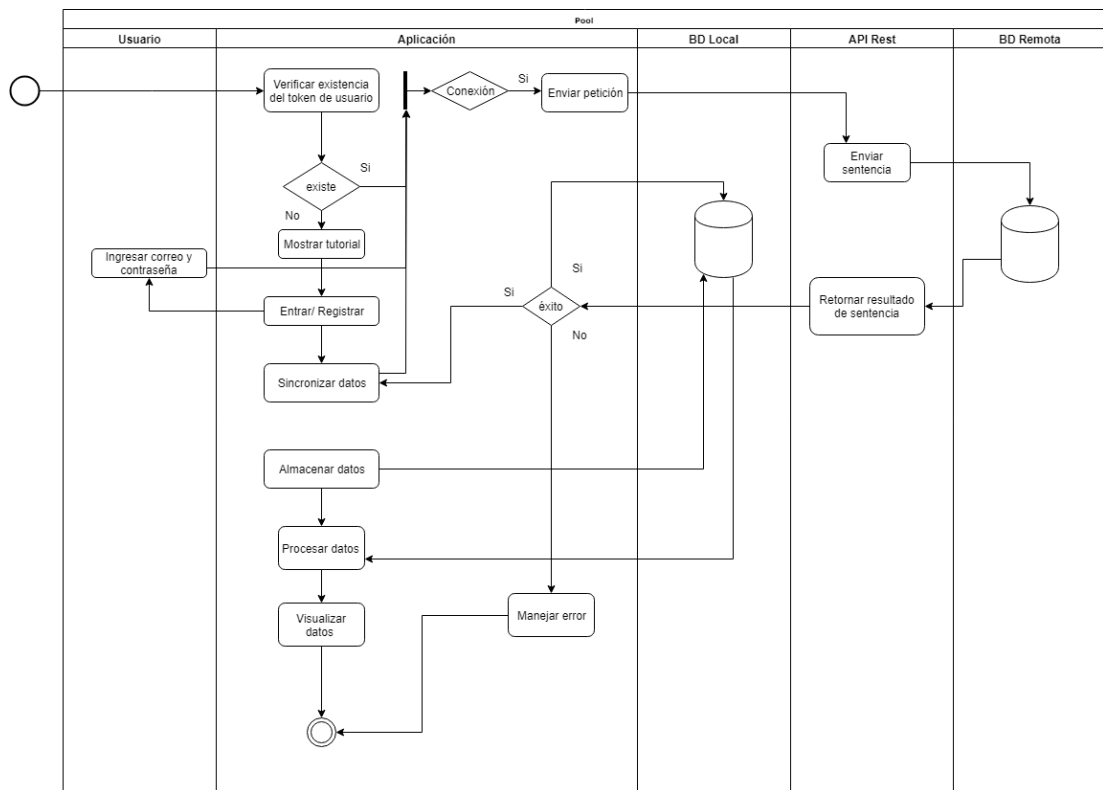
public function listar_categoria_get(){
    $resultado = array();
    $punto=array();
    $query = $this->db->query('SELECT id_cat,
                                nombre_cat,
                                descripcion_cat,
                                subcategoria_cat
                                FROM categoria
                                ORDER BY nombre_cat desc ');
    foreach( $query->result() as $row ){
        $datos = array(
            'id_cat'=>$row->id_cat,
            'nombre_dat'=> $row->nombre_cat,
            'descripcion_dat'=> $row->descripcion_cat,
            'subcategoria_dat'=> $row->categoria_cat
        );
        array_push($resultado, $datos);
    }
    $respuesta = array(
        'error' => FALSE,
        'datos'=> $resultado
    );
    $this->response( $respuesta );
}

```

### 4.2.3. Diagramas de actividades de la aplicación

El flujo de ingreso a la aplicación, en primera instancia, verifica si existe un token identificador. En caso de no existir, presenta un manual de usuario tipo tutorial de cómo funciona la aplicación. Posteriormente muestra una pantalla con las opciones de *Iniciar sesión*, *Registrar usuario* y la opción de *Omitir*.

Si el usuario acciona cualquiera de las tres opciones mencionadas anteriormente, el sistema se conecta con la API-REST que se encarga de retornar la información. Mientras tanto, la aplicación se encarga de almacenar localmente la información en el dispositivo móvil, procesarla y posteriormente visualizarla en el mapa. Si el usuario inicia sesión o se registra la aplicación, habilita funciones para agregar, modificar, borrar puntos en el mapa que pertenezcan al usuario, como se indica en la figura 9.



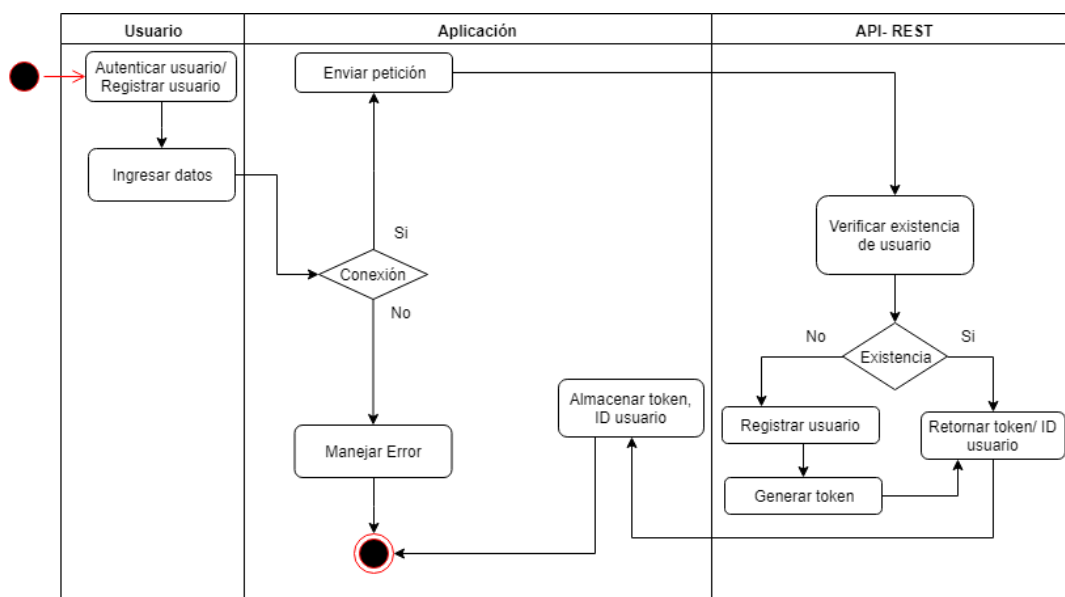
**Figura 9:** Diagrama de actividades. Ingreso a la aplicación  
Fuente: Elaborado por el investigador

El flujo para iniciar sesión es una interacción entre el usuario y el sistema. Cuando el usuario desea iniciar sesión en el sistema se comprueba la existencia del usuario, el fin del flujo es la visualización de datos sincronizados y la habilitación de gestión de datos georreferenciales tales como; agregar, modificar, eliminar, los cuales pertenezcan al usuario.

La aplicación recibe los datos del usuario, verifica si existe disponibilidad de red, en caso de verificarse se procesan los datos para posteriormente generar una petición POST vía HTTP, en este caso el API-REST se encarga de procesar la información para manejarla en el formato correcto, luego verifica si el usuario existe en la base de datos, en caso de éxito el API-REST genera un token y retorna el

token y el Id del usuario, en la aplicación las almacena en el storage del dispositivo, como se indica en la figura 10.

En caso de que el usuario no exista se genera un token, y se registra en la base de datos, luego que se ha registrado exitosamente el API-REST retorna el token y el Id del usuario, la aplicación se encarga de almacenar las variables, para mantener la sesión activa. Si la aplicación verifica la existencia del token almacenada en la aplicación procede a la recuperación de datos para visualizar.



**Figura 10:** Diagrama de actividades. Iniciar sesión.

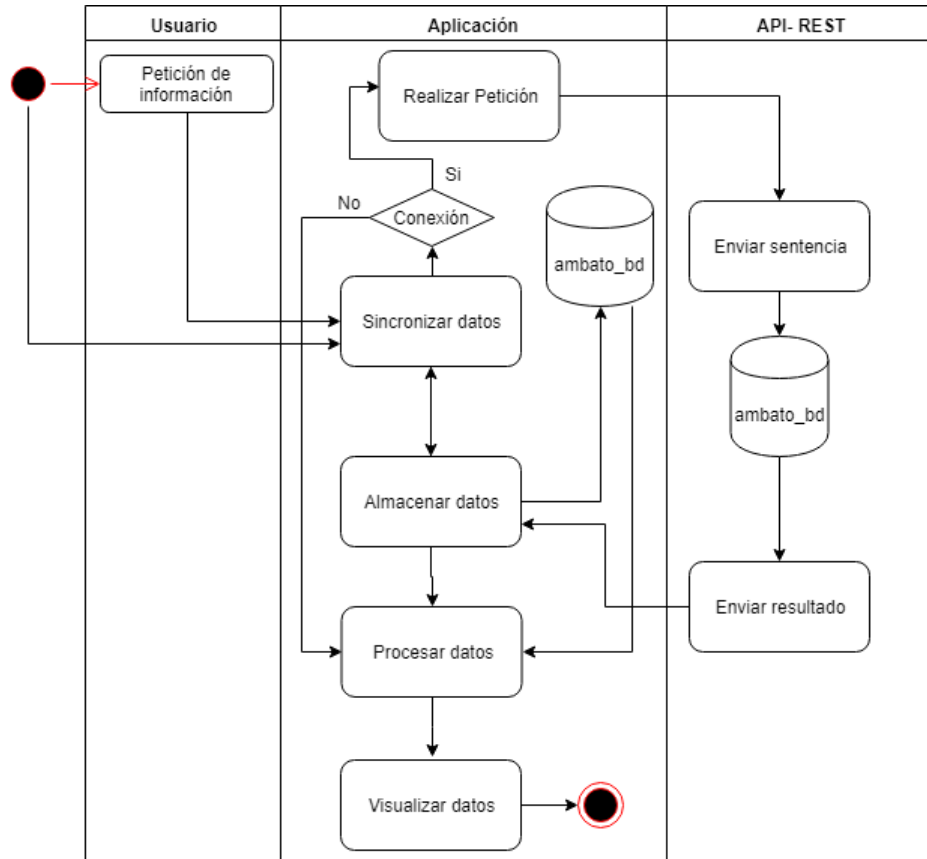
Fuente: Elaborado por el investigador.

El flujo para la visualización de los datos comienza con la petición de datos y termina con la visualización de los mismos en el formato correcto.

En la aplicación existen dos maneras de solicitar de la información actual que existe en la base de datos: bajo petición y por petición automática. En el caso de bajo petición, el usuario, con la acción de *actualizar*, genera una solicitud. Por su parte, la petición automática tiene lugar cada 7 segundos siempre y cuando el usuario esté en la vista de mapa. En ambos casos se realiza una solicitud al API-REST, el que retorna la información requerida.

La aplicación en el teléfono se encarga de procesar los datos recibidos para almacenarlos en la base de datos local. Luego, el sistema recupera los datos, ya almacenados en la base de datos local, para visualizarlos, como se indica en la figura 11.

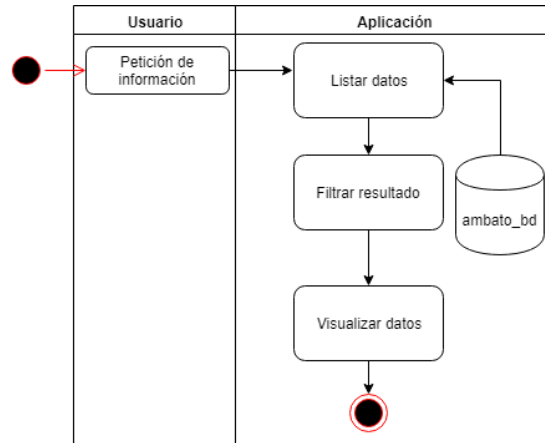




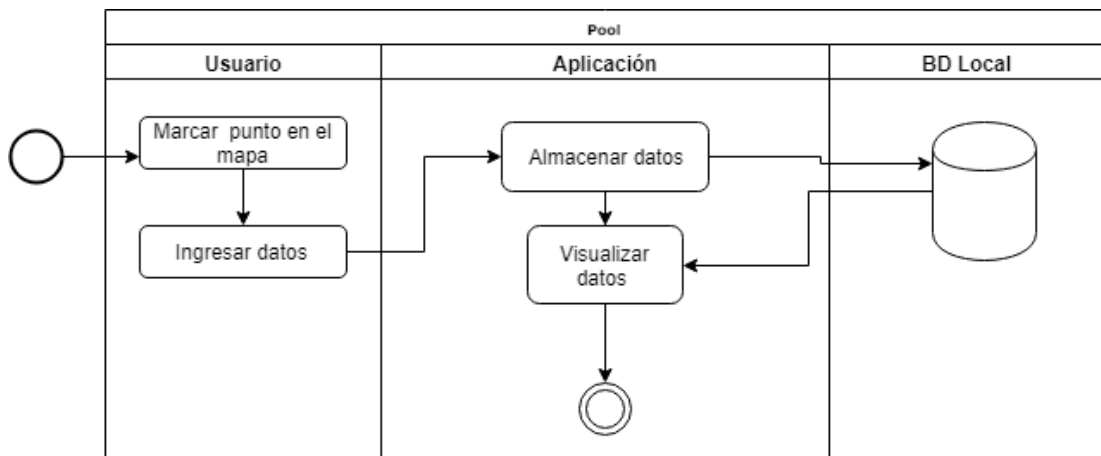
**Figura 11:** Diagrama de actividades. Visualización de datos.  
Fuente: Elaborado por el investigador.

La secuencia de la aplicación para filtrar información, comienza con la petición del usuario, la cual se receipta a través de una entrada de texto. El sistema se encarga de filtrar la lista proveniente de la base de datos local. El filtro se da a través del nombre de la ubicación. La aplicación retorna el resultado de datos que contengan la cadena de texto que el usuario ingresó. En caso que el usuario borre el contenido de la entrada de texto, el filtrado de los datos se inhabilita, como se indica en la figura 12.

El flujo para agregar datos comienza con la acción del usuario de marcar un punto en el mapa, ingresa datos y posteriormente se visualiza en el mapa. El usuario marca un punto en el mapa, el sistema detecta las coordenadas del punto en el mapa y las almacena en variables, luego el usuario ingresa datos como: nombre, descripción y tiene la posibilidad de elegir: tomar una foto o seleccionarla de la galería. El formulario de ingreso no habilitará el botón de guardar hasta que el usuario proporcione los datos requeridos. Al guardar la información en la base de datos local se agrega un registro con el campo de sincronización en estado falso, para posteriormente cambiar el estado cuando el usuario realice la petición de sincronizar, como se indica en la figura 13.



**Figura 12:** Diagrama de actividades. Búsqueda de datos  
Fuente: Elaborado por el investigador



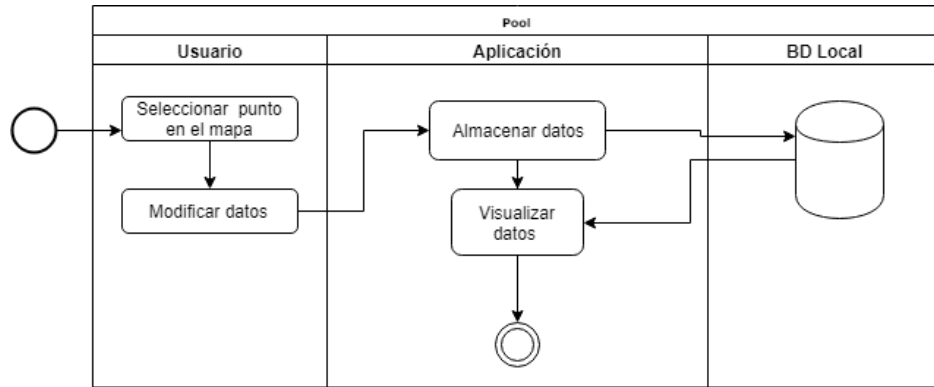
**Figura 13:** Diagrama de actividades. Agregar datos  
Fuente: Elaborado por el investigador

La secuencia para modificar datos comienza con la selección de un punto que pertenezca al usuario. El usuario modifica los datos del punto seleccionado, la aplicación lo almacena en la base de datos local y el flujo termina en la visualización de los datos. El registro modificado se almacena con un estado de modificado en la base de datos local. Luego, en el proceso de sincronización, el aplicativo incorpora dichos puntos en el listado de puntos a sincronizar, como se indica en la figura 14.

El flujo comienza con la selección del punto que pertenezca al usuario. El registro no se elimina; la aplicación cambia el estado del registro a eliminado, lo que se almacena en la base de datos local. El registro se eliminará en el proceso de sincronización.

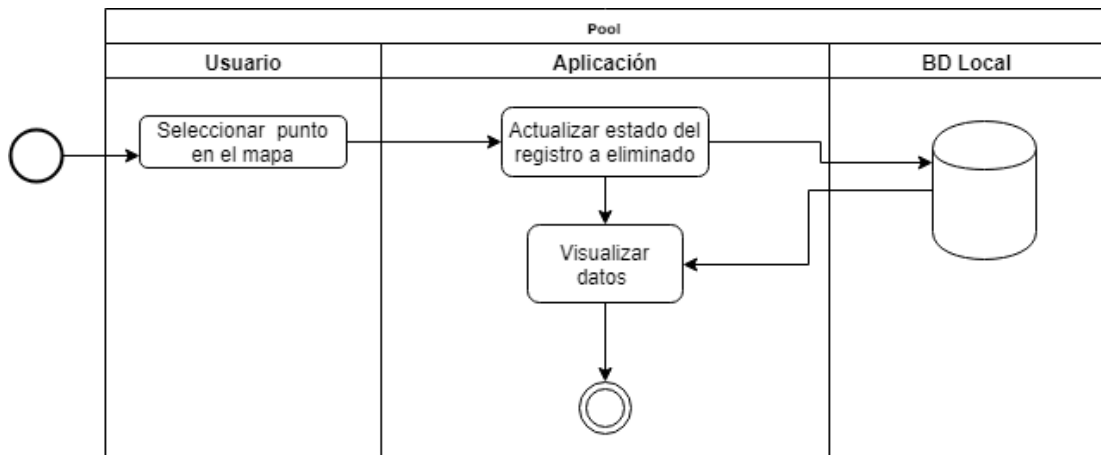
El registro se borra de manera exitosa en la base de datos remota y posteriormente se borra en la base de datos local, como se indica en la figura 15.

La secuencia para la sincronización de datos se inicia con la acción del usuario.



**Figura 14:** Diagrama de actividades. Modificar datos

Fuente: Elaborado por el investigador.



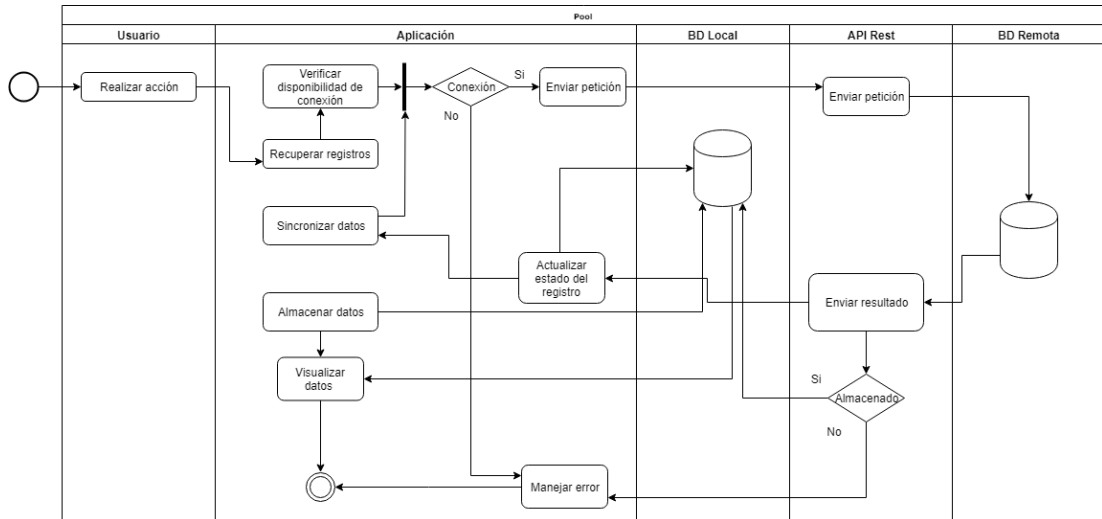
**Figura 15:** Diagrama de actividades. Eliminar datos

Fuente: Elaborado por el investigador

La aplicación lista los datos que tienen el estado *para sincronizar* y se verifica la disponibilidad de red. Si la red está disponible, se realiza la petición de inicialización del proceso de sincronización.

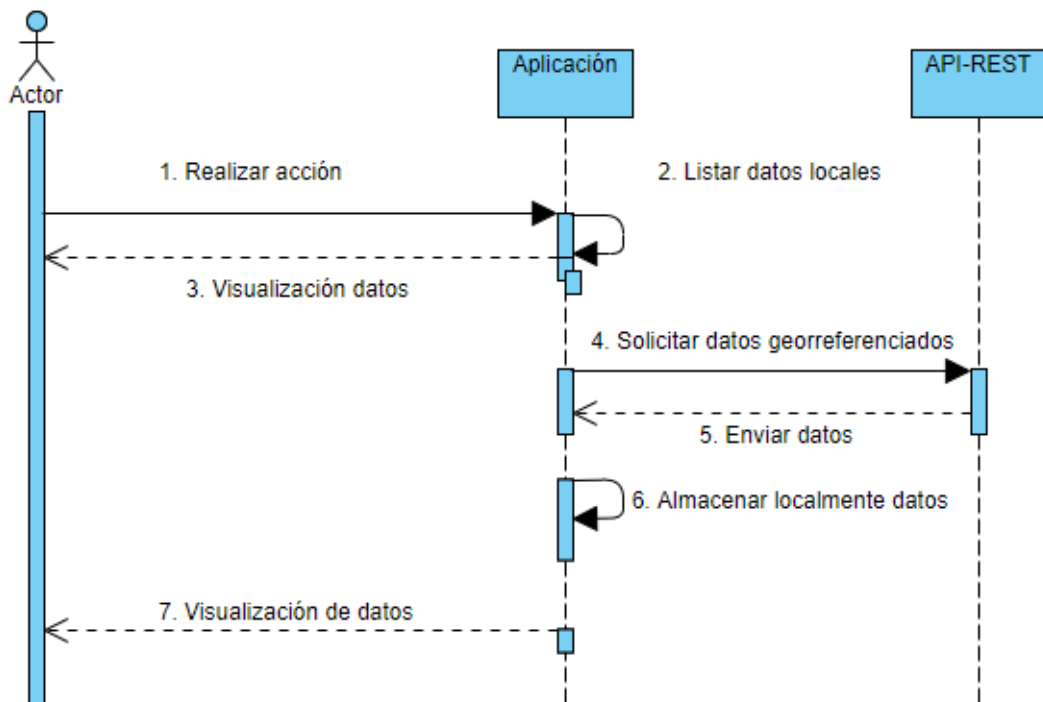
El API-REST procesa los datos enviados y luego envía la petición a la base de datos remota. Posteriormente el API retorna el resultado de la operación. En caso de éxito, el sistema inicia la recuperación de los datos disponibles en la base de datos remota. La aplicación realiza una nueva petición y el API-REST retorna los nuevos puntos que no constan en la base de datos local.

Posteriormente los almacena localmente y los visualiza. En caso de que el API-REST retorne algún error, se maneja el error y la aplicación presenta el mensaje correspondiente al usuario, como se indica en la figura 16.

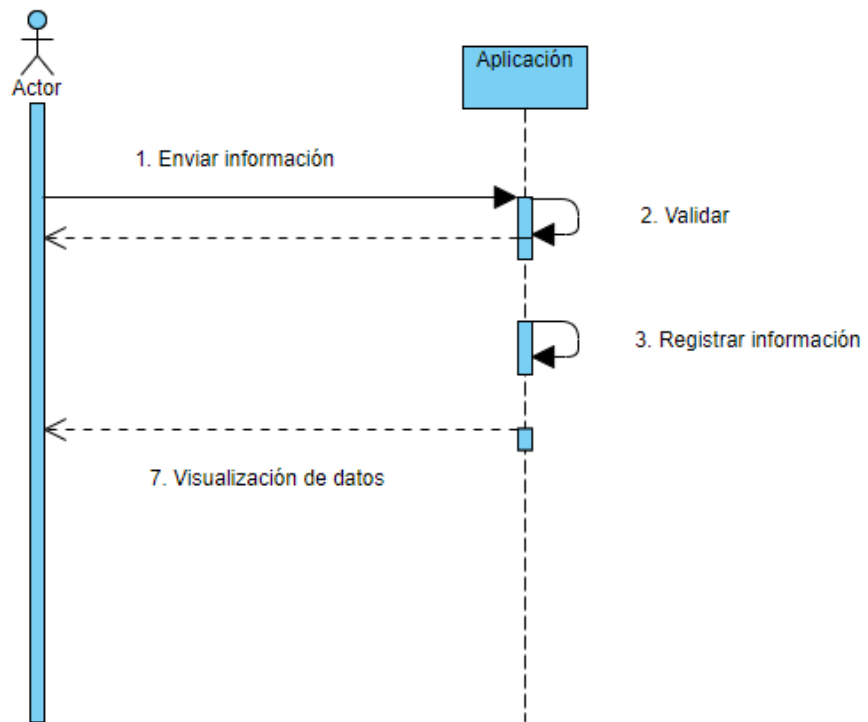


**Figura 16:** Diagrama de actividades. Sincronizar datos  
Fuente: Elaborado por el investigador

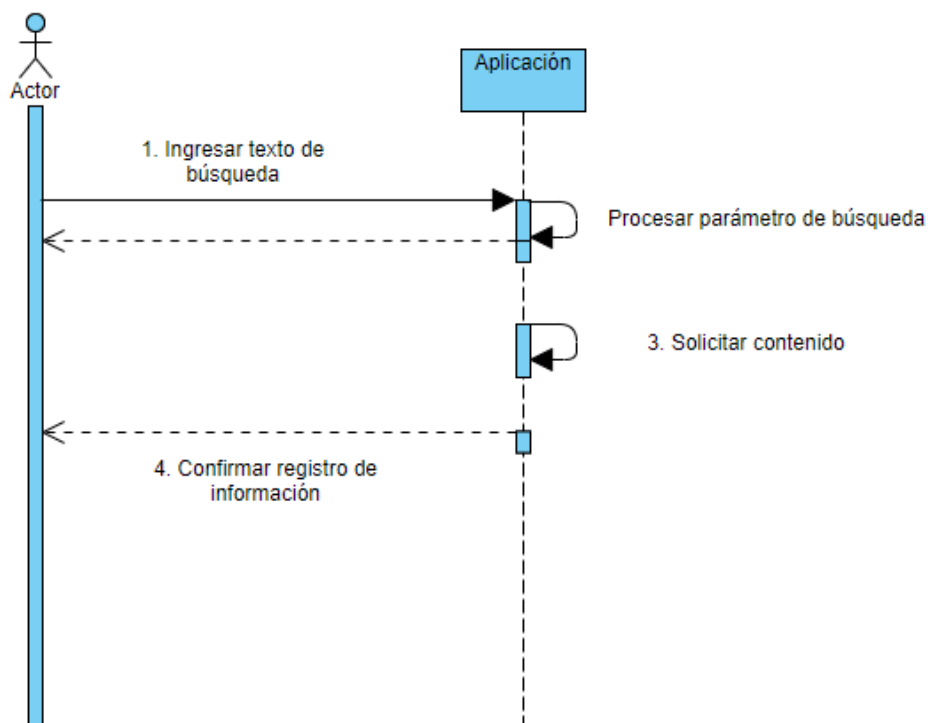
### 4.3. Diagramas de secuencia



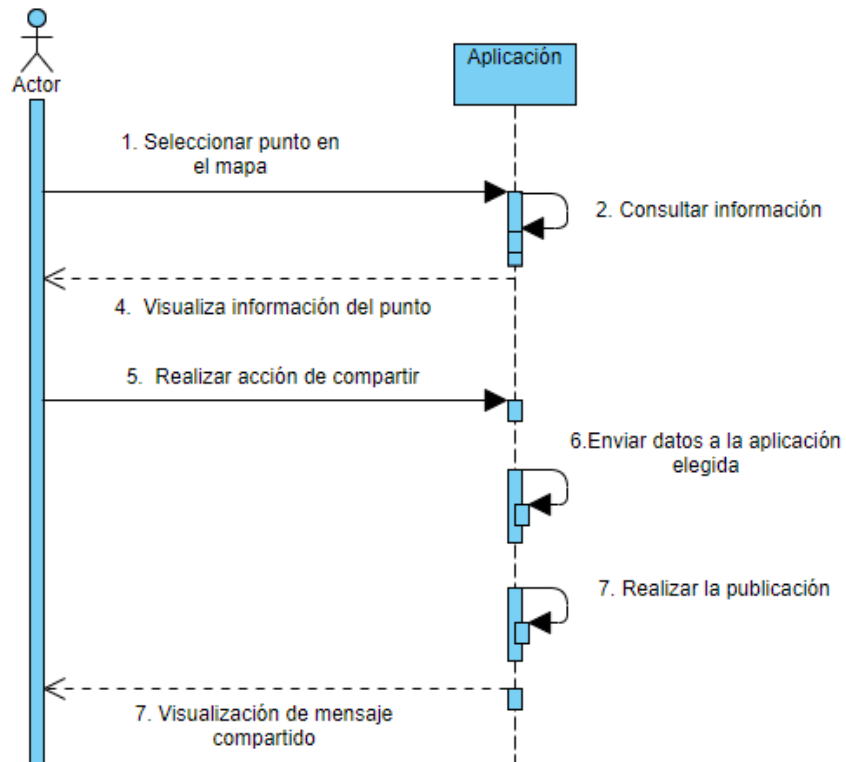
**Figura 17:** Diagrama de secuencia. Visualización de datos  
Fuente: Elaborado por el investigador



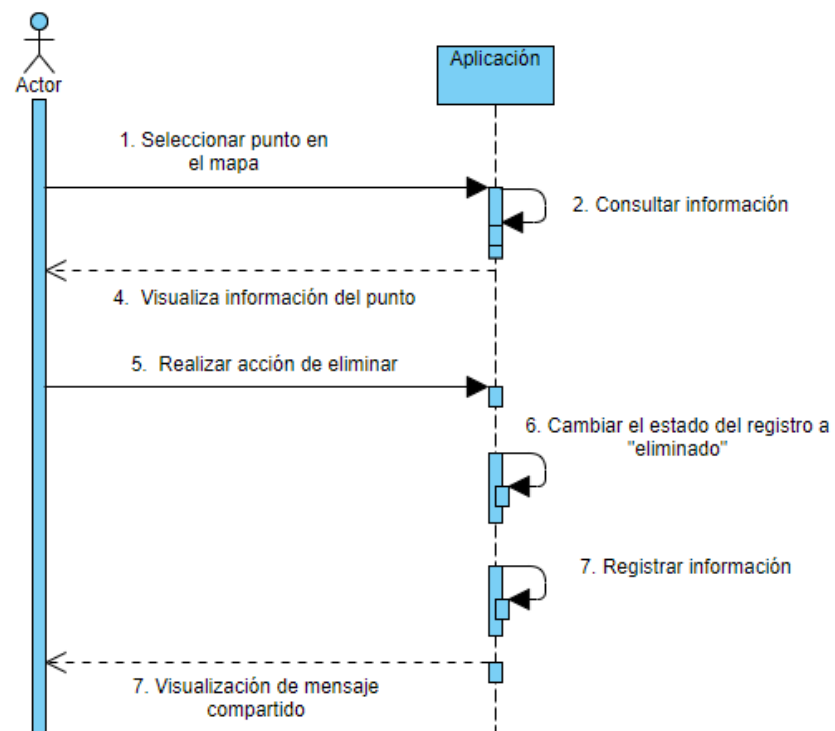
**Figura 18:** Diagrama de secuencia. Agregar puntos  
Fuente: Elaborado por el investigador



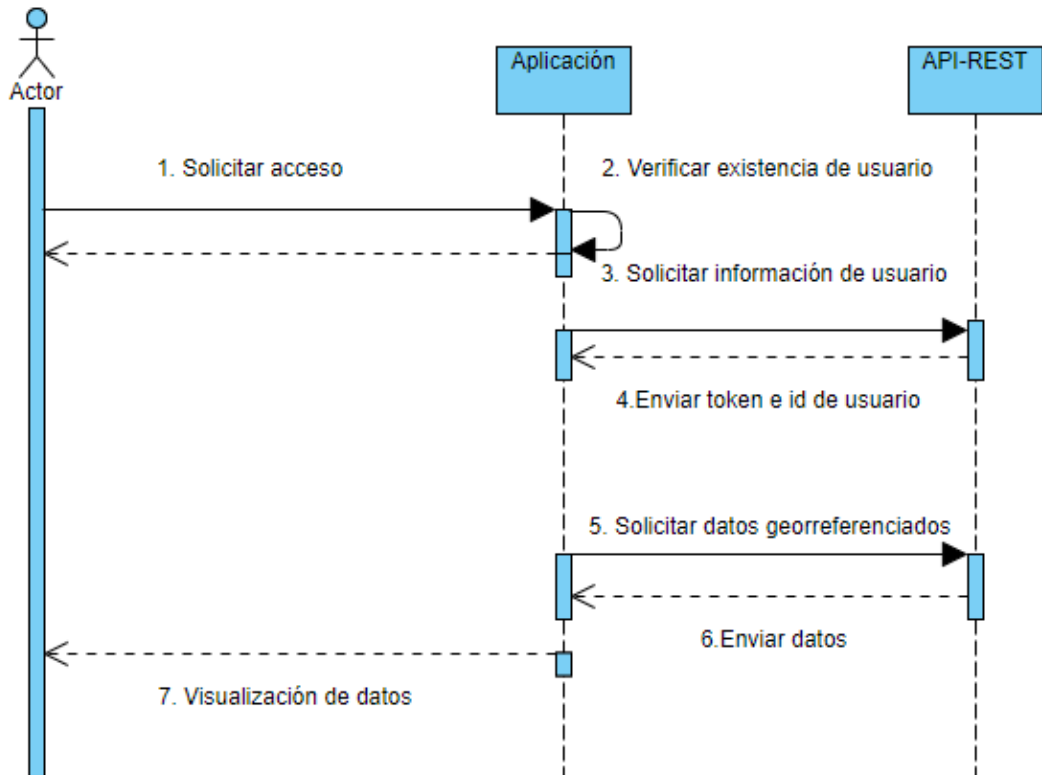
**Figura 19:** Diagrama de secuencia. Buscar de datos  
Fuente: Elaborado por el investigador



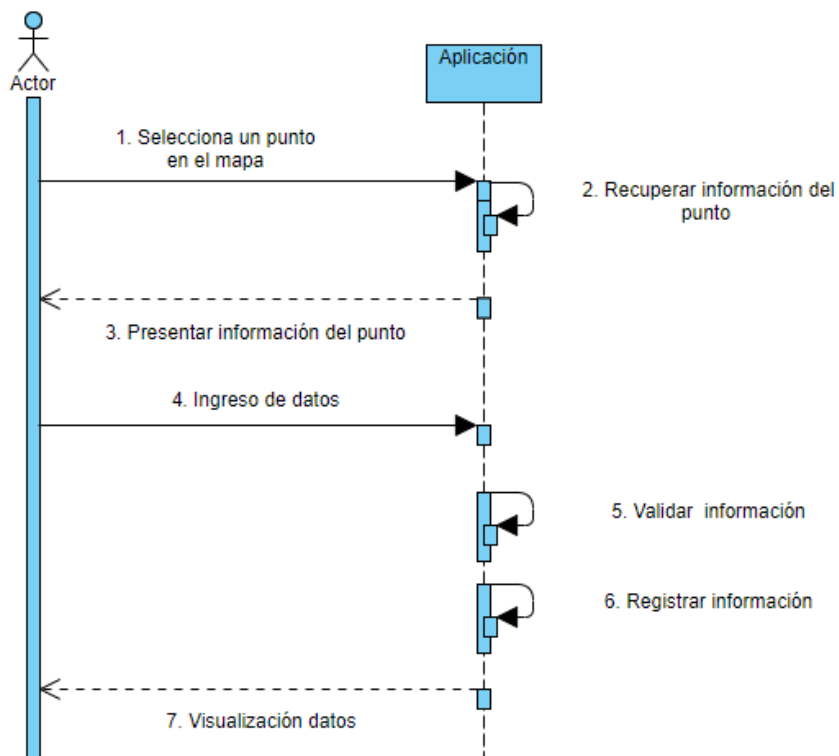
**Figura 20:** Diagrama de secuencia. Compartir puntos  
 Fuente: Elaborado por el investigador



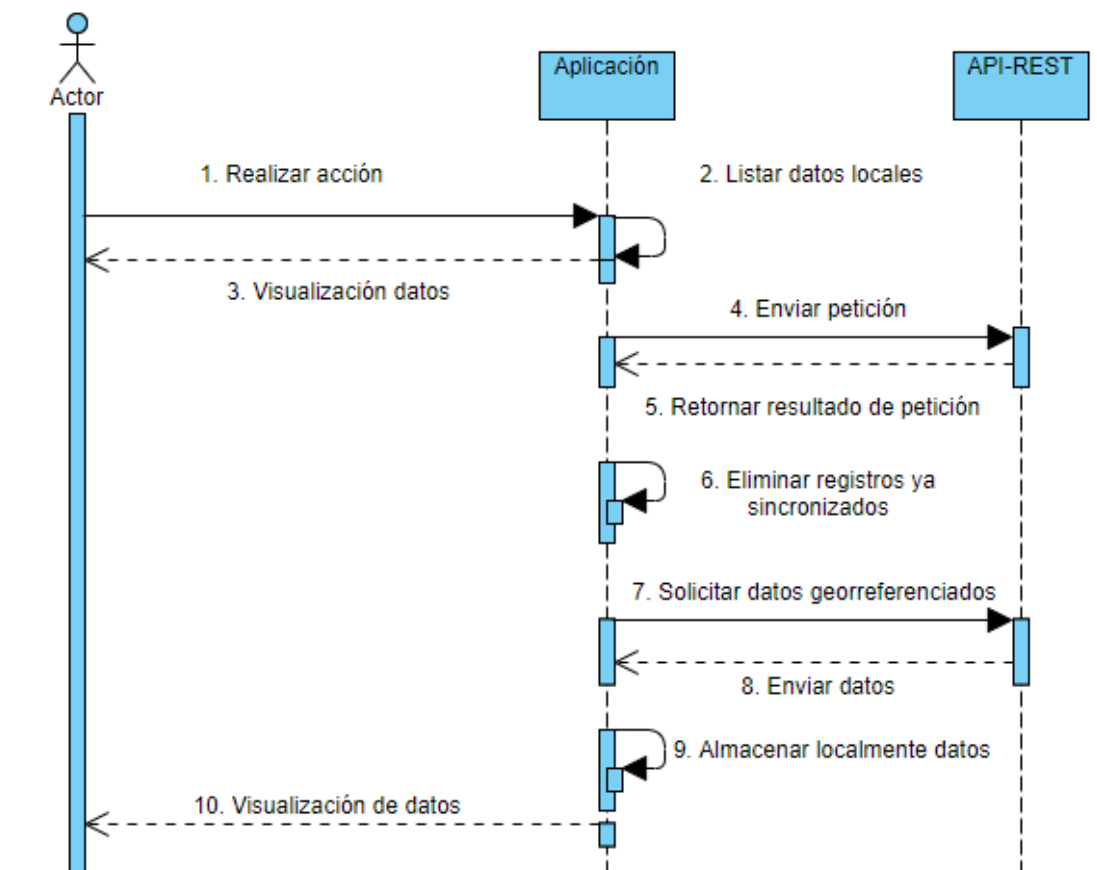
**Figura 21:** Diagrama de secuencia. Eliminar punto  
 Fuente: Elaborado por el investigador



**Figura 22:** Diagrama de secuencia. Login  
 Fuente: Elaborado por el investigador



**Figura 23:** Diagrama de secuencia. Modificar de datos  
 Fuente: Elaborado por el investigador



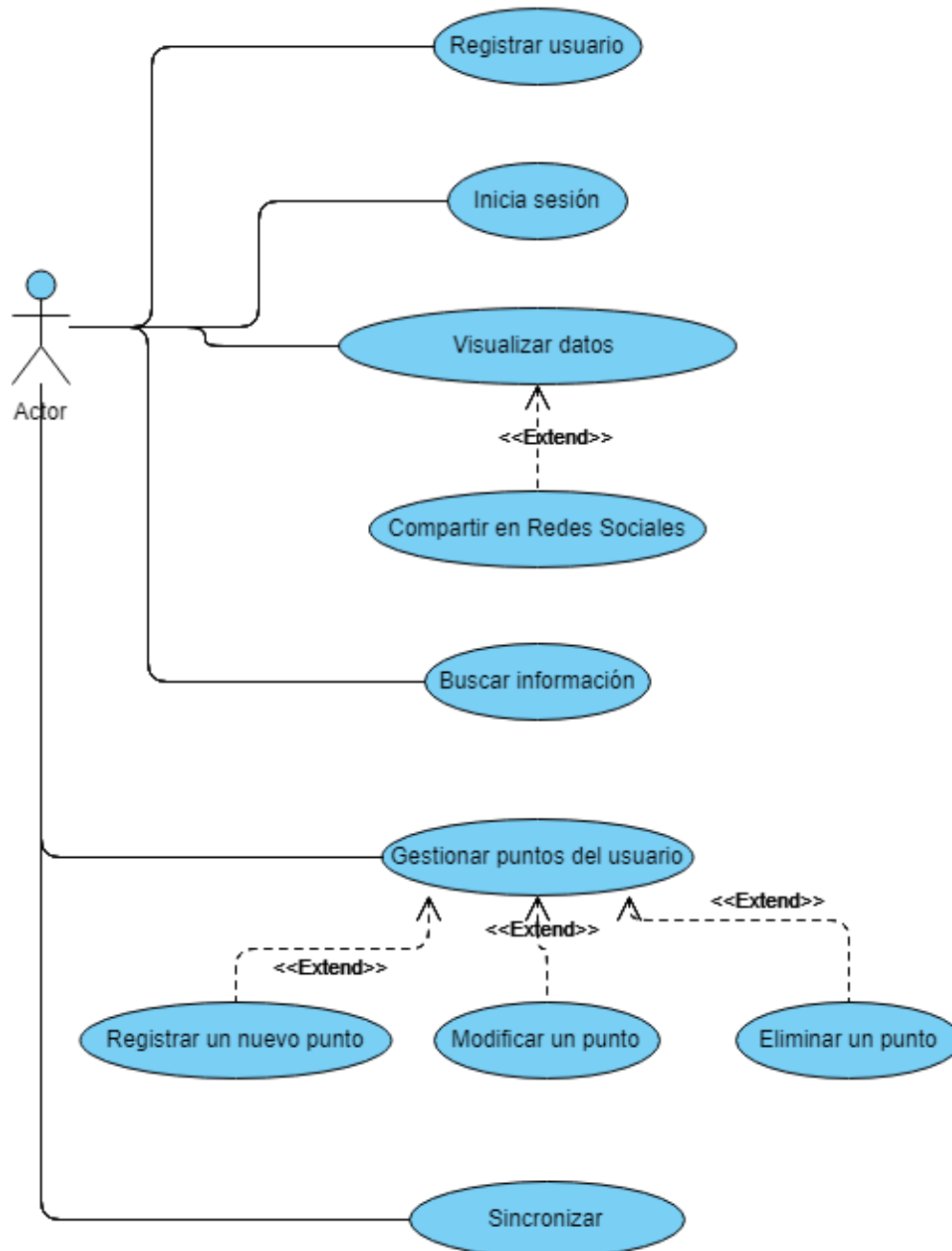
**Figura 24:** Diagrama de secuencia. Sincronizar datos  
 Fuente: Elaborado por el investigador



#### 4.4. Desarrollo de la aplicación

##### 4.4.1. Especificación de casos de uso

Se procede a explicar los diferentes casos de uso identificados, con el propósito de detallar los procesos y operaciones de los mismos. Ver figura 25.



**Figura 25:** Diagrama de casos de uso.  
Fuente: Elaborado por el investigador.

**Tabla 3:** Caso de uso. Registrar “usuario”  
 Fuente: Elaborado por el investigador

<b>Caso de Uso :</b>	Registrar “usuario”
<b>Resumen :</b>	Almacenar el usuario para cuando inicie sesión en el aplicativo pueda realizar acciones CRUD.
<b>Actor :</b>	Usuario
<b>Precondiciones :</b>	El actor debe tener un correo y proporcionar una clave de al menos 6 dígitos.
<b>Flujo Normal :</b>	<ul style="list-style-type: none"> <li>■ El actor abre la aplicación la cual le da las opciones de iniciar sesión y registrarse.</li> <li>■ El actor elige registrar.</li> <li>■ El actor ingresa los datos necesarios.</li> <li>■ El sistema realiza la conexión con el API-REST para registrar al usuario.</li> <li>■ El sistema retorna un mensaje el éxito o de error según el caso.</li> <li>■ El sistema habilita las opciones para agregar, modificar, eliminar datos del usuario.</li> <li>■ El sistema muestra la información proveniente de la base de datos externa, la cual se ha sincronizado.</li> </ul>
<b>Flujo Alternativo :</b>	<ul style="list-style-type: none"> <li>■ El actor dispuso omitir el registro.                             <ul style="list-style-type: none"> <li>• El sistema muestra contenido de manera general.</li> </ul> </li> </ul>
<b>Poscondiciones :</b>	Registros almacenados, actualizados, consultados.

**Tabla 4:** Caso de uso. Inicio de sesión  
 Fuente: Elaborado por el investigador

<b>Caso de Uso :</b>	Inicio de Sesión
<b>Resumen :</b>	Permite ingresar al actor en el sistema.
<b>Actor :</b>	Usuario
<b>Precondiciones :</b>	El actor debe estar registrado en el sistema.
<b>Flujo Normal :</b>	<ul style="list-style-type: none"> <li>■ El actor solicita el ingreso al sistema.</li> <li>■ El sistema comprueba la existencia del actor dentro del registro de datos.</li> <li>■ El sistema habilita acciones de agregar, modificar, datos del usuario.</li> <li>■ Visualizar los datos.</li> </ul>
<b>Flujo Alternativo :</b>	<ul style="list-style-type: none"> <li>■ El usuario no existe.                             <ul style="list-style-type: none"> <li>• El sistema da la opción de registrar usuario.</li> <li>• El sistema redirige a la pagina para registrar usuario</li> <li>• Visualizar los datos.</li> </ul> </li> <li>■ Se mantiene la sesión iniciada.</li> </ul>
<b>Poscondiciones :</b>	Sesión de actor se encuentra inicializada y disponible

**Tabla 5:** Caso de uso. Visualizar datos  
Fuente: Elaborado por el investigador

<b>Caso de Uso :</b>	Visualizar datos
<b>Resumen :</b>	Permiten la visualización datos georreferenciados de todos los usuarios
<b>Actor :</b>	Usuario
<b>Precondiciones :</b>	El actor debe haber iniciado sesión.
<b>Flujo Normal :</b>	<ul style="list-style-type: none"> <li>■ El actor solicita la visualización de un contenido.</li> <li>■ Se visualiza los marcadores en el mapa.</li> <li>■ El actor selecciona un marcador en el mapa.</li> <li>■ El sistema muestra la información de ese punto, en una nueva página.</li> <li>■ Se habilita las opciones de descargar imagen y ubicar en el mapa.</li> <li>■ El actor tiene las opciones de buscar y compartir.</li> <li>■ El actor regresa a la pantalla principal.</li> </ul>
<b>Flujo Alternativo :</b>	<ul style="list-style-type: none"> <li>■ El sistema no tiene conexión. <ul style="list-style-type: none"> <li>• El actor realiza la petición de datos</li> <li>• El sistema retorna los datos almacenados localmente</li> <li>• El sistema muestra los datos en el mapa.</li> <li>• Se habilitan las opciones de compartir en redes Sociales y descargar la imagen del punto seleccionado</li> </ul> </li> <li>■ El actor tiene las opciones de buscar y compartir.</li> </ul>
<b>Poscondiciones :</b>	Ejecutar de visualización

**Tabla 6:** Caso de uso. Registrar un nuevo punto  
Fuente: Elaborado por el investigador

<b>Caso de Uso :</b>	Registrar un nuevo punto
<b>Resumen :</b>	Almacenar un nuevo registro referente a un punto seleccionado en el mapa
<b>Actor :</b>	Usuario
<b>Precondiciones :</b>	El actor debe tener una sesión inicializada y disponible.
<b>Flujo Normal :</b>	<ul style="list-style-type: none"> <li>■ El actor la selección de un punto en el mapa</li> <li>■ Se visualiza la página de registro.</li> <li>■ El actor ingresa la información necesaria.</li> <li>■ Acepta el guardar el contenido.</li> <li>■ El sistema valida la información.</li> <li>■ Almacenar registro localmente.</li> <li>■ Visualizar mensaje de éxito de registro.</li> </ul>
<b>Flujo Alternativo :</b>	<ul style="list-style-type: none"> <li>■ El actor ingreso datos incorrectamente <ul style="list-style-type: none"> <li>• Se muestra mensaje de error.</li> </ul> </li> </ul>
<b>Poscondiciones :</b>	Registro almacenados y consultados.

**Tabla 7:** Caso de uso. Modificar un punto  
 Fuente: Elaborado por el investigador

<b>Caso de Uso :</b>	Modificar un punto
<b>Resumen :</b>	Modificar un registro referente a un punto seleccionado en el mapa
<b>Actor :</b>	Usuario
<b>Precondiciones :</b>	El actor debe tener una sesión inicializada y disponible.
<b>Flujo Normal :</b>	<ul style="list-style-type: none"> <li>■ El actor la selección de un punto en el mapa.</li> <li>■ El actor selecciona modificar datos.</li> <li>■ Se visualiza la página de modificación.</li> <li>■ El actor modifica la información necesaria.</li> <li>■ Acepta el guardar el contenido.</li> <li>■ El sistema valida la información.</li> <li>■ Almacenar registro localmente.</li> <li>■ Visualizar mensaje de éxito de modificación.</li> <li>■</li> </ul>
<b>Flujo Alternativo :</b>	<ul style="list-style-type: none"> <li>■ El actor ingreso datos incorrectamente                             <ul style="list-style-type: none"> <li>• Se muestra mensaje de error.</li> </ul> </li> </ul>
<b>Poscondiciones :</b>	Registro modificados y consultados.

**Tabla 8:** Caso de uso. Buscar datos  
Fuente: Elaborado por el investigador

<b>Caso de Uso :</b>	Buscar datos
<b>Resumen :</b>	Consultar contenidos que cumpla con los parámetros de búsqueda.
<b>Actor :</b>	Usuario
<b>Precondiciones :</b>	El actor debe tener una sesión inicializada y disponible.
<b>Flujo Normal :</b>	<ul style="list-style-type: none"> <li>■ El actor ingresa el parámetro.</li> <li>■ El actor solicita la búsqueda.</li> <li>■ Visualiza el contenido que cumplan con el parámetros de búsqueda.</li> </ul>
<b>Flujo Alternativo :</b>	<ul style="list-style-type: none"> <li>■ El actor ingreso parámetros de búsqueda que no tiene contenido que lo cumplan. <ul style="list-style-type: none"> <li>• Se muestra una lista vacía .</li> <li>• Visualizar información recientes.</li> </ul> </li> </ul>
<b>Poscondiciones :</b>	Ninguna

**Tabla 9:** Caso de uso. Compartir en las redes sociales un punto  
Fuente: Elaborado por el investigador

<b>Caso de Uso :</b>	Compartir en las redes sociales un punto
<b>Resumen :</b>	Compartir en Facebook, Whatsapp, Instagram un punto seleccionado.
<b>Actor :</b>	Usuario
<b>Precondiciones :</b>	El actor debe seleccionar un ítem de la lista de puntos.
<b>Flujo Normal :</b>	<ul style="list-style-type: none"> <li>■ El actor dispone de la acción al pulsar el botón según el icono de la red social.</li> <li>■ Se envía información a cada aplicación de la red social; nombre, descripción, link de la imagen para ser compartido.</li> <li>■ Retorno de mensaje con éxito de compartido.</li> </ul>
<b>Flujo Alternativo :</b>	<ul style="list-style-type: none"> <li>■ No es posible realizar la acción social <ul style="list-style-type: none"> <li>• El sistema muestra un error de disponibilidad de red.</li> </ul> </li> </ul>
<b>Poscondiciones :</b>	Punto compartido en las Redes Sociales.





### 4.4.2.3. Diagrama de clases de la programación del servidor

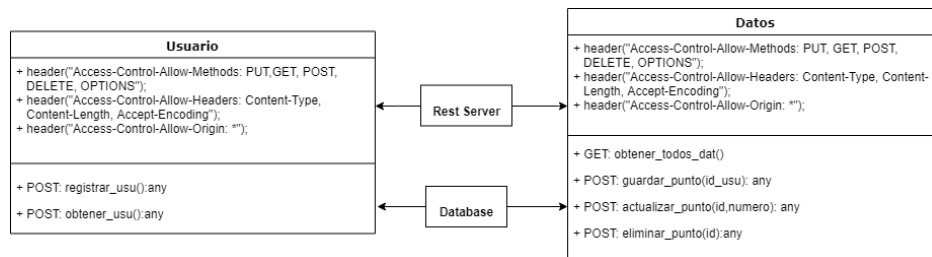


Figura 27: Diagrama de clases. Programación del API-REST

**Detalle de funcionalidad** La comunicación entre el sistema implementado y API-REST es una arquitectura que se da principalmente en el entorno móvil. Esto implica que lo primero que ocurre al querer usar la aplicación es un proceso de *instalación* a través del cual el usuario solicita registro, acceso u omisión. En la aplicación se tiene la posibilidad de cambiar de usuario.

Ese procedimiento garantiza que el sistema reciba el *token* de acceso a la aplicación para mantener una sesión activa y habilitar las opciones de gestión de datos pertenecientes al usuario en cuestión.

**“Mapa vectorial sin conexión”** Para obtener el origen de datos con teselas vectoriales se ha utilizado el API de SDK de MapboxGL JS modificado por Oscar Fonts, los cuales utilizan la petición bajo el formato de dirección *http://direccion\_ip/{x}/{y}/{z}.pbf*. Sobre el origen de datos se aplican *sprites*, que son archivos *png* que contienen los iconos de los lugares, los cuales están representados en el mapa; por ejemplo un hotel o un estadio.

Los archivos *glyphs* son archivos de extensión *.pbf* que contienen estilos de fuente de letra, los cuales se aplican en los diferentes niveles de zoom y dependen del objeto que se pretende identificar; por ejemplo, una calle principal va a tener el estilo de letra con negrilla y una calle secundaria va a tener el estilo de letra itálica.

MapboxGL JS modificado utiliza el origen de datos almacenados en un archivo *mbtiles*, el cual es una base de datos Sql que contiene las teselas vectoriales. El componente, en lugar de apuntar a un servidor que contiene las teselas como usualmente se usa, apunta a un archivo *mbtiles*. En lugar de extraer las teselas vectoriales del archivo, existe la posibilidad de leer el origen de datos mediante el uso del plugin *cordova-sqlite-ext*, el cual dispone de la característica de leer bases de datos prealmacenadas, como es el caso de los *mbtiles*. Esto último es lo que se implementó, diferenciando el comportamiento de la aplicación creada con

relación a lo que se hace en el caso, por ejemplo, de un GeoServer.

El fichero de estilo utilizado es un documento JSON que cumple con la especificación de estilo de Mapbox. La especificación de estilo está diseñada especialmente para que Mapbox GL JS y los SDK de Mapbox en móviles puedan leerlos y comprenderlos, de modo que el mapa se pueda representar satisfactoriamente en la página donde se despliega.

En el presente proyecto se utilizó *klokantech-basic-gl-style* [25], disponible en GitHub. Para adecuarlo a la forma en que opera la aplicación (de forma local), se hicieron algunas modificaciones en el archivo `style.json`, para que que apunten a directorios de la aplicación en lugar de un servidor remoto. De esta forma se logró un resultado satisfactorio. El estilo controla todo lo relacionado con la forma en que se muestra el mapa, con excepción de lo referente al manejo de los iconos y otras funcionalidades específicas que se gestionan directamente por el plugin de Mapbox.

```
"sources": {
  "openmaptiles": {
    "type": "vector",
    "url": "https://free.tilehosting.com/data/v3.json?key={key}"
  }
},
"sprite": "https://openmaptiles.github.io/klokantech-basic-gl-style/sprite",
"glyphs": "https://free.tilehosting.com/fonts/{fontstack}/{range}.pbf?key={key}"
```

**Figura 28:** Extracto de código. Estilo *klokantech-basic-gl-style*  
Fuente: Elaborado por el investigador

```
"sources": {
  "openmaptiles": {
    "type": "mbtiles",
    "path": "data/ecuador_mapa.mbtiles"
  }
},
"sprite": "styles/klokantech-basic/sprite",
"glyphs": "fonts/{fontstack}/{range}.pbf",
```

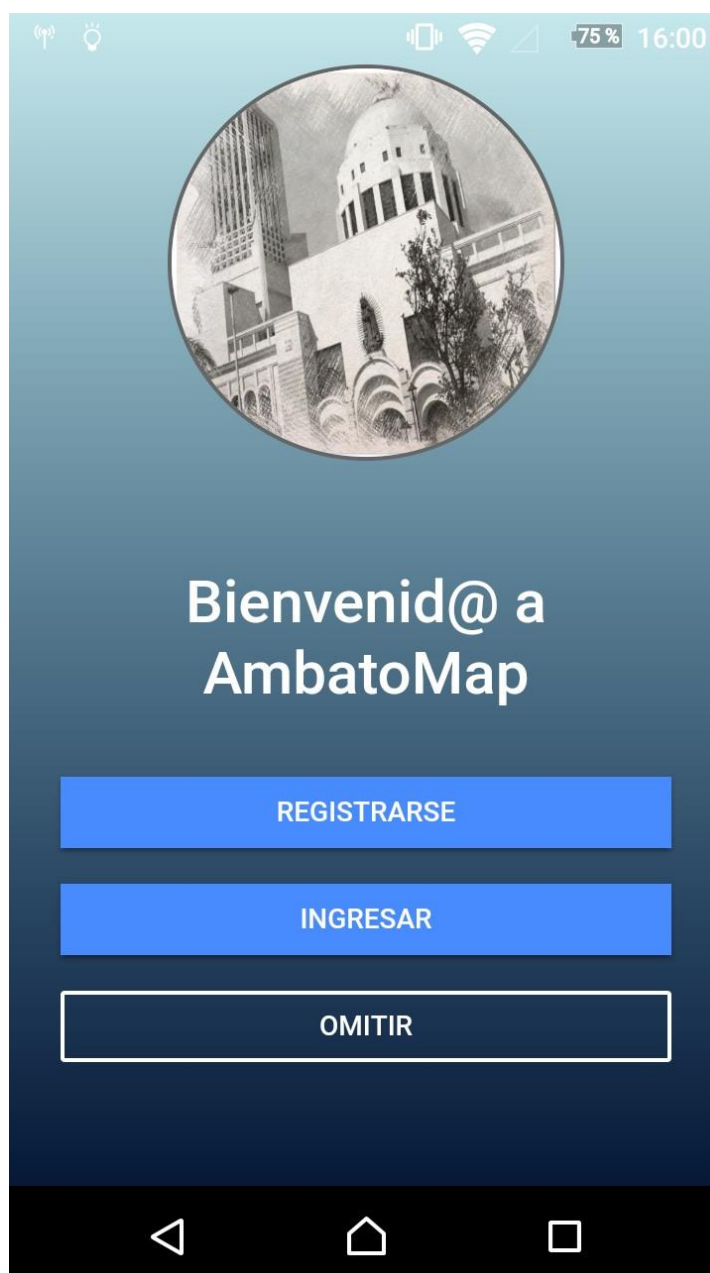
**Figura 29:** Extracto de código. Estilo *klokantech-basic-gl-style* modificado  
Fuente: Elaborado por el investigador

El archivo ha sido modificado en variables como el origen de las teselas vectoriales, la ruta de los iconos de los lugares del mapa como; estadios, hoteles, parques entre otros y la ruta de los archivos *pbf* con el la fuente de letra que utiliza el archivo de estilo *JSON*. De esta forma el plugin de MapboxGL JS modificado

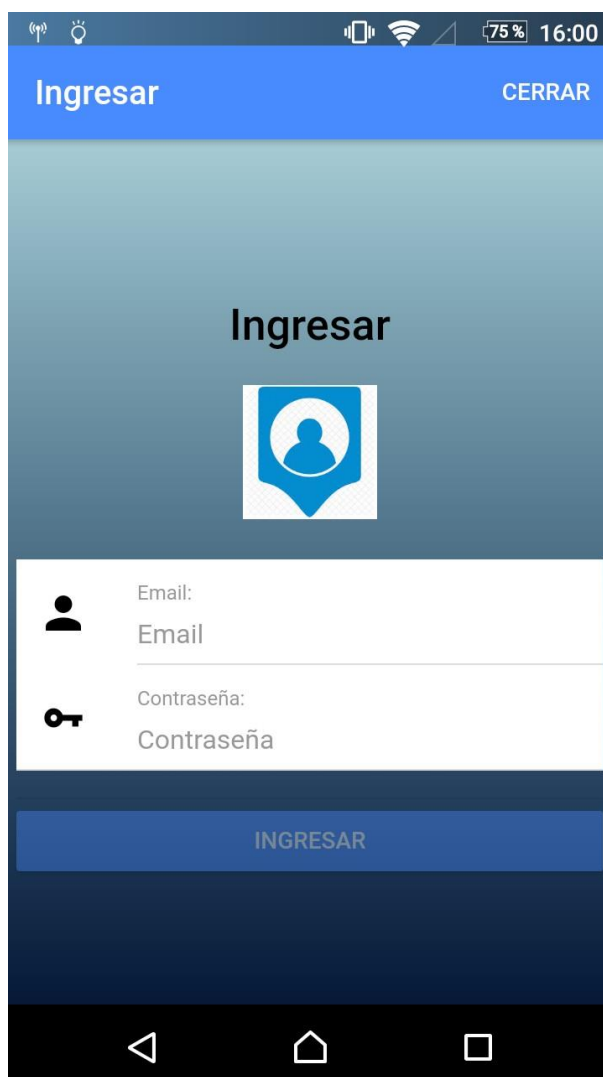
tiene los archivos y directorios necesarios para operar y brindar a la aplicación la disponibilidad de un mapa vectorial sin conexión.

**“Ingreso a la aplicación”** El aplicativo móvil muestra las pantallas de ingreso a la aplicación en caso de no existir un token de sesión como se muestra en la figura 30.

Se presenta el formulario para la autenticación del usuario, el formulario de registro consta de los mismos campos, por lo tanto tiene el mismo diseño con la pantalla de ingreso como se indica en la figura 31.



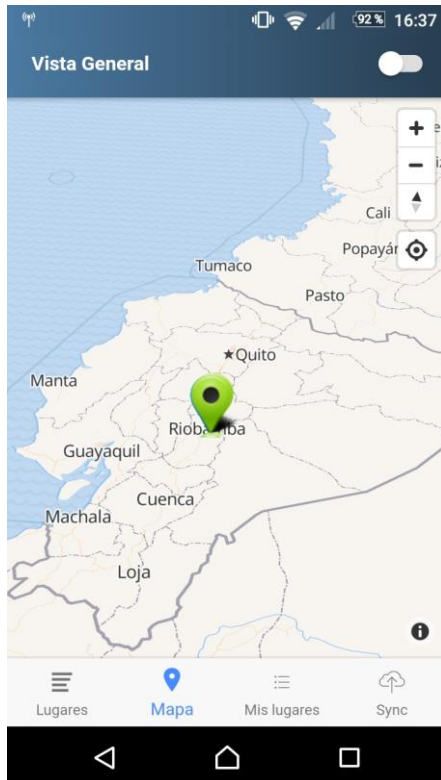
**Figura 30:** Vista de pantalla. Ingreso a la aplicación.  
Fuente: Elaborado por el investigador



**Figura 31:** Vista de pantalla. Login  
Fuente: Elaborado por el investigador

**“Visualización de contenido”** En la pantalla se listan todos los datos georreferenciados que se han sincronizado desde la base de datos, mediante la comunicación con el API-REST. La comunicación con el API-REST desde la aplicación obtiene una lista de datos. La aplicación se encarga de almacenarlos en el dispositivo móvil de forma local. Los datos serán usados sin conexión; en caso de requerir los datos, la aplicación mostrará los datos que han sido almacenados en el proceso de sincronización. Dicho proceso se ejecuta con un intervalo de 7 segundos.

Los items de la lista recuperada de los datos locales tienen los datos necesarios a excepción de la imagen del lugar. La imagen del lugar estará disponible sin conexión solo si el usuario realiza la acción de descargar imagen. Esta opción está disponible en la pantalla de *Detalle* del lugar.



**Figura 32:** Vista general de la aplicación  
Fuente: Elaborado por el investigador

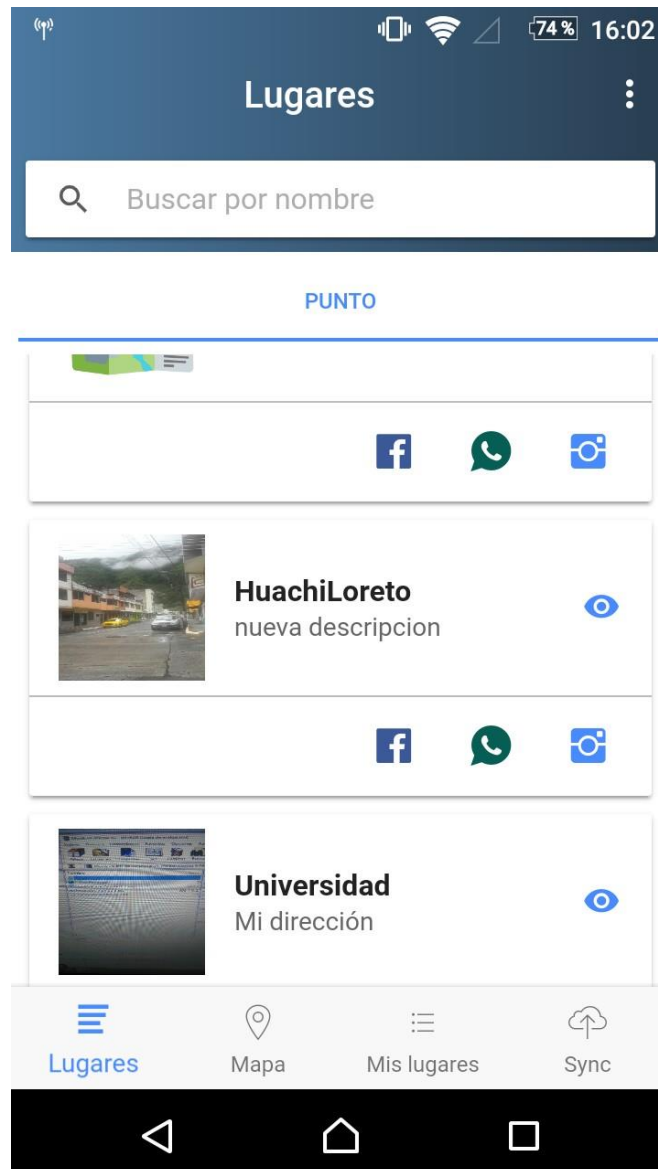
```

if (this.network.type!=='none') {
  this._ds.listar_puntos().then((datos) => { this.listaPuntos = datos });
  this._ls.listar_puntos()
    .then(() =>
      this._ss.insertarSync()
    )
    .then(() => {
      this._ds.listar_puntos().then((datos) => { this.listaPuntos = datos })
    })
  this._ds.listar_puntos().then((datos) => { this.listaPuntos = datos })
}
else {
  this.presentMensaje("Trabajando sin conexión...");
  this._ds.listar_puntos().then((datos) => { this.listaPuntos = datos })
  console.log(this.listaPuntos);
}
}

```

**Figura 33:** Extracto de código. Cliente - Sincronización de datos  
Fuente: Elaborado por el investigador

**“Lugares”** La pantalla de *Lugares* muestra los puntos agregados por todos los usuarios los cuales se han almacenado localmente, cada item tiene opciones de compartir en las redes sociales y la acción del usuario de ver a detalle el item al seleccionarlo como se indica en la figura 36.



**Figura 34:** Vista de pantalla. *Lugares* de la aplicación.  
Fuente: Elaborado por el investigador.

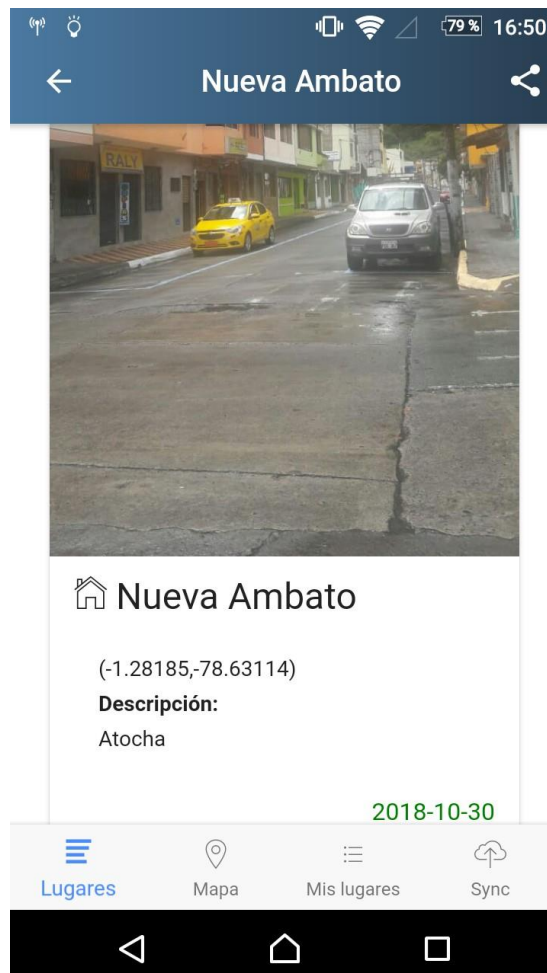
**“Sincronización”** La aplicación recupera la lista de datos almacenados localmente, el aplicativo verifica que exista disponibilidad de red para enviar los datos con la petición correcta, en caso de añadir, modificar o eliminar un punto del usuario.

El usuario con sesión activa en el dispositivo, tiene la posibilidad de gestionar los registros de la lista de sincronización.

El algoritmo utilizado para la sincronización, separa por registros a insertar, a modificar y a eliminar, se realizan las peticiones con retorno de variables *boolean* para confirmar su eliminación de la lista una vez sincronizado.

```
compartir(punto: puntoClase, opcion: number) {
  if (opcion == 1) {
    this.socialSharing.shareViaFacebook(punto.nombre_dat, URL_IMAGENES + punto.imagen_dat, URL_IMAGENES + punto.imagen_dat)
      .then(() => { })
      .catch((err) => { console.log(err) })
  }
  else if (opcion == 2) {
    this.socialSharing.shareViaWhatsApp(punto.nombre_dat, URL_IMAGENES + punto.imagen_dat, URL_IMAGENES + punto.imagen_dat)
      .then(() => { console.log(punto.nombre_dat) })
      .catch(() => { })
  }
  else {
    this.socialSharing.shareViaInstagram(punto.nombre_dat, URL_IMAGENES + punto.imagen_dat)
      .then(() => { console.log(punto.nombre_dat) })
      .catch(() => { })
  }
}
```

**Figura 35:** Extracto de código. Cliente - Comparti.r  
Fuente: Elaborado por el investigador.



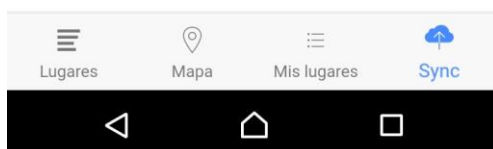
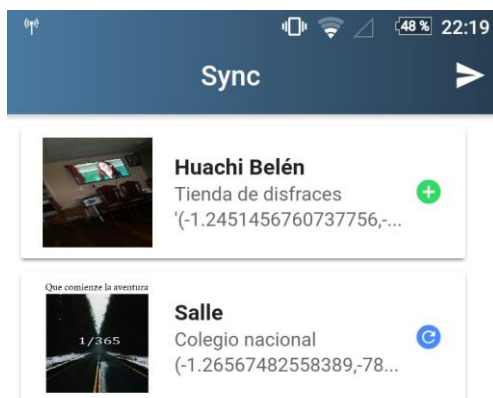
**Figura 36:** Vista de pantalla. Cliente - Ver detalle.  
Fuente: Elaborado por el investigador.

```

irPuntoDetalle(obj: any) {
  this.navCtrl.push(DetallePuntoPage, { 'punto': obj });
}

```

**Figura 37:** Extracto de código. Ver Detalle  
Fuente: Elaborado por el investigador.



**Figura 38:** Vista de pantalla. Sincronización  
Fuente: Elaborado por el investigador.

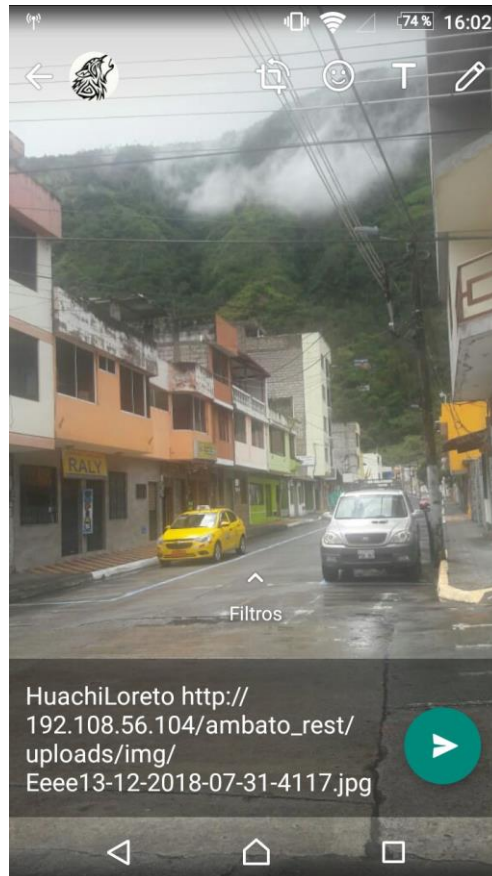
**“Compartir”** La acción dentro de la aplicación adquiere la información del objeto *Lugar*, que es un punto de usuario que esté registrado en el sistema, se visualiza en la lista general.

El componente instalado *cordova-plugin-x-socialsharing* tiene funciones para verificar si la aplicación está instalada en el dispositivo. El componente tiene la capacidad de compartir por correo electrónico y por aplicaciones sociales como; Facebook, Whatsapp e Instagram.

El sistema provee una página modal a partir de la información a compartir. En dicha página, el usuario podrá personalizar el contenido del mensaje de la publicación correspondiente.



La página modal se genera a través de las APIs de las redes sociales según la selección del usuario, por lo que su apariencia y funcionamiento es independiente del sistema.



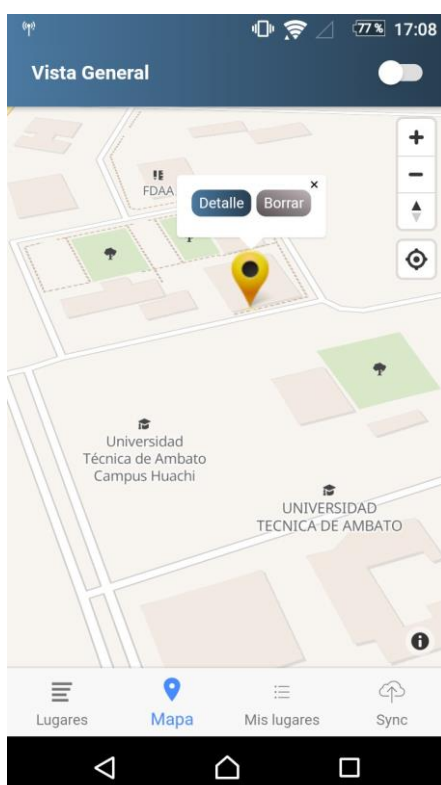
**Figura 39:** Vista de pantalla. Compartir lugar en las redes sociales  
Fuente: Elaborado por el investigador.

```
compartir(punto: puntoClase, opcion: number) {  
  if (opcion == 1) {  
    this.socialSharing.shareViaFacebook(punto.nombre_dat, URL_IMAGENES + punto.imagen_dat, URL_IMAGENES + punto.imagen_dat)  
      .then(() => { })  
      .catch((err) => { console.log(err) })  
  }  
  else if (opcion == 2) {  
    this.socialSharing.shareViaWhatsApp(punto.nombre_dat, URL_IMAGENES + punto.imagen_dat, URL_IMAGENES + punto.imagen_dat)  
      .then(() => { console.log(punto.nombre_dat) })  
      .catch(() => { })  
  }  
  else {  
    this.socialSharing.shareViaInstagram(punto.nombre_dat, URL_IMAGENES + punto.imagen_dat)  
      .then(() => { console.log(punto.nombre_dat) })  
      .catch(() => { })  
  }  
}
```

**Figura 40:** Extracto de código. Cliente - Realizar acción “compartir”.  
Fuente: Elaborado por el investigador.

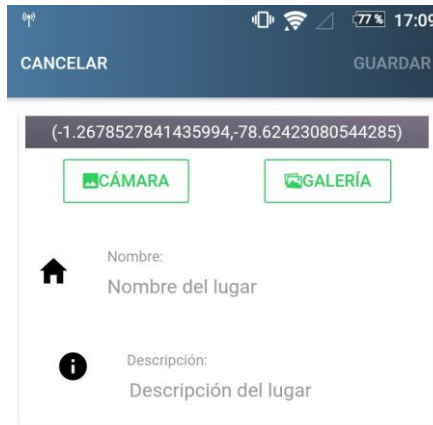
**“Registrar un nuevo lugar”** El sistema permite al usuario la creación de un nuevo lugar; esto se lo realiza mediante la selección de un punto en el mapa, posteriormente un formulario se carga en una pantalla del sistema, a través de la cual se ingresa la información necesaria para el registro de un nuevo lugar en la base de datos local del sistema.

El usuario ingresa la información, la que se valida en la capa cliente de la aplicación. Si los datos son válidos, se almacena un registro en la base de datos local, posteriormente se utilizara el registro para enviar a la API-REST y almacenar en la base de datos remota.



**Figura 41:** Vista pantalla. Seleccionar punto.  
Fuente: Elaborado por el investigador.

**“Búsqueda de lugares”** La búsqueda de contenidos a nivel del cliente se realiza mediante un cuadro de texto. El usuario introduce un texto de búsqueda, la aplicación se encarga de filtrar la lista y mostrar los items que cumplan con el parámetro ingresado, no se realiza consultas a la base de datos, optimizando el tiempo de respuesta, el filtro de lista utiliza un *Pipe* componente de Ionic que realiza dicho proceso, en caso que el usuario actualice la lista, el estado de búsqueda no se ve afectado, si algún item es agregado a la base datos durante el proceso de sincronización y cumple con el parámetro, se muestra en la lista filtrada.



**Figura 42:** Vista Pantalla. Cliente. Registrar un nuevo contenido  
Fuente: Elaborado por el investigador.

```
guardar() {
  this._ds.agregar_punto_nuevo(this.credentialsForm.value['nombre'], "(" + this.coordenadas.lat + "," + this.coordenadas.lng + ")",
    this.pathImage, this.credentialsForm.value['descripcion'], this._us.id_usuario, new Date().toISOString().substring(0, 10), false, 0, 0)
  .then(() => {
    this.presentLoadingText();
    this.loading.onDidDismiss(() => {
      this.presentMensaje("Punto: ${this.credentialsForm.value['nombre']} agregado");
      this.viewCtrl.dismiss(true);
    });
  })
  .catch((err) => {
    console.log(err);
  })
}
```

**Figura 43:** Extracto de código. Registrar un nuevo contenido.  
Fuente: Elaborado por el investigador.

```
<ion-searchbar placeholder="Buscar por nombre" animated="true" spellcheck="true" [(ngModel)]="nombre_dat">
  <button ion-button type="button" (click)="sort()">Sort</button>
</ion-searchbar>
```

**Figura 44:** Extracto de código. Cuadro de texto.  
Fuente: Elaborado por el investigador.

```
<ion-card *ngFor="let punto of listaPuntos | buscar : nombre_dat" >
  <ion-item (click)="irPuntoDetalle(punto)">
```

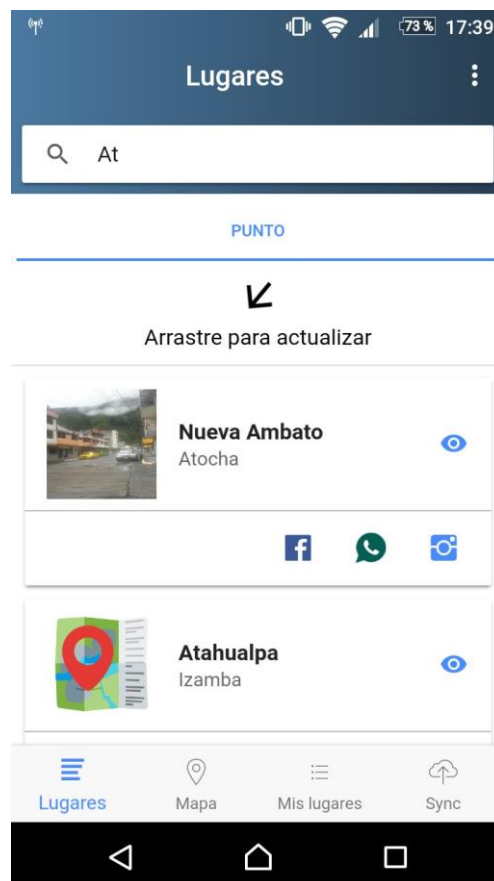
**Figura 45:** Extracto de código. Filtro *pipe* **buscar** en la lista lugares.  
Fuente: Elaborado por el investigador.

#### 4.4.3. Requisitos no funcionales

**Usabilidad** La aplicación da un valor de importancia a la usabilidad dado que es el grado en el cual un producto puede ser utilizado por sus usuarios para lograr

```
transform(items: any[], terms: string): any[] {
  if(!items) return [];
  if(!terms) return items;
  terms = terms.toLowerCase();
  return items.filter( it => {
    return it.nombre_dat.toLowerCase().includes(terms); // only filter name
  });
}
```

**Figura 46:** Extracto de código. *Pipe Buscar*.  
Fuente: Elaborado por el investigador.



**Figura 47:** Vista de pantalla. Búsqueda de lugares.  
Fuente: Elaborado por el investigador.

metas con efectividad, eficiencia y satisfacción en un determinado contexto de uso grado, lo cual es un factor de éxito considerable para un proyecto como para ser ignorado.

La usabilidad se encuentra relacionada íntimamente con la percepción del usuario respecto a la calidad del sistema; los algoritmos internos, procedimientos, velocidad de respuesta o la definición de la arquitectura puede ser de la manera mas adecuada y de un gran valor técnico, pero el usuario no tiene visibilidad de eso, sino que es la interfaz con la que interactúa y se relaciona.

**Estilos** Los estilos aplicados sobre la interfaz se pueden definir en múltiples archivos *scss*, la diferencia de cada uno es el alcance que tienen dentro de la aplicación. El archivo *variables.scss* contiene variables para definir colores y son accesibles a nivel de toda la aplicación.

```
$colors: (  
  primary:    #488aff,  
  secondary:  #32db64,  
  danger:     #f53d3d,  
  light:      #f4f4f4,  
  blanco:     #FFFFFF,  
  dark:       #222,  
  editar:     #E8B30F,  
  localizar:  #18d45c,  
  descargar:  #547d8b,  
  guardado:   #18A43C,  
  fila:       #D1D5D2,  
  whastapp:   #075E54,  
  facebook:   #3b5998,  
  opcion:     #c3c3c3  
);
```

**Figura 48:** Extracto de código. Colores en *variables.scss*.

Fuente: Elaborado por el investigador.

El archivo *app.scss* es un archivo de hoja de estilo, en el cual se define la apariencia de componentes de la aplicación. En la figura 49 se indica la forma en que el área de notificaciones se un estilo de degradado, el estilo se aplica a todas las pantallas en la aplicación.

```
ion-header {
  background: transparent;
  .toolbar-background {
    background: linear-gradient(to right, #4b79a1, #283e51);
  }
}
```

**Figura 49:** Extracto de código. Estilo del header de la aplicación  
Fuente: Elaborado por el investigador.

```
.platform-android {
  .ion-page {
    ion-header {
      padding-top: $cordova-md-statusbar-padding;
      background: linear-gradient(to right, #4b79a1, #283e51);
    }
  }
}
```

**Figura 50:** Extracto de código. Estilo del header de la aplicación para la plataforma Android.  
Fuente: Elaborado por el investigador.

**Diseño** El diseño la aplicación se basa en los componentes de Ionic, cada componente dispone la característica adaptativa.

En mayor medida la interfaz está compuesta por botones, íconos, listas, que tienen una apariencia visual diferente en cada uno de los sistemas operativos, porque Android, iOS y Windows Phone tienen su propia forma de entender el diseño, el cual el framework se encarga de mostrar el componente con el diseño indicado en cada una de las plataformas.

**Mantenibilidad** La facilidad de modificación va de la mano con la escalabilidad, pues se entiende que una aplicación de software es de fácil modificación si no se deben llevar a cabo muchos cambios en la estructura existente de tal aplicación, con la finalidad, ya sea de cambiar el funcionamiento actual o de introducir nuevas características. Una de las principales características de Ionic Framework es la facilidad para el versionamiento y distribución de la aplicación. El archivo *config.xml* contiene los parámetros para la configuración de la versión.

```

<?xml version="1.0" encoding="utf-8" ?>
<widget id="com.ionic.ecuadormap" version="1.0.0" xmlns="http://www.w3.org/ns/widgets" xmlns:cdv="http://cordova.apache.org/1.0" >
  <name>EcuadorMap</name>
  <description>Aplicación que contiene el mapa de todo el Ecuador, agrega puntos de interés de forma offline</description>
  <author email="victorbautista@gmail.com" href="victorbautista@gmail.com">Victor Bautista</author>
  <content src="index.html" />
  <access origin="*" />
</widget>

```

**Figura 51:** Extracto de código. Archivo de configuración de la aplicación.  
Fuente: Elaborado por el investigador.

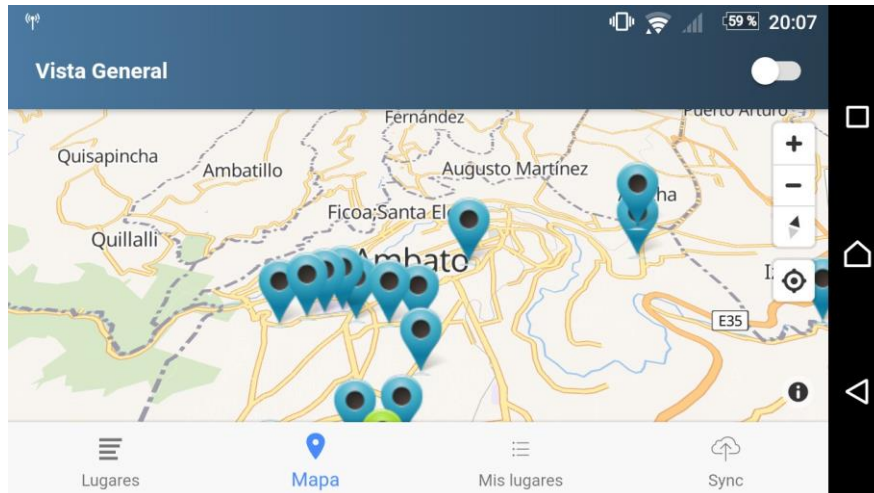
Explicación de la figura 51:

1. Nombre de la aplicación.
2. Identificador de la aplicación, se tiene que escribir de manera inversa el sitio web del aplicativo.
3. Descripción de la aplicación.
4. El número se debe cambiar cuando la aplicación se ha hecho cambios significativos.
5. El número se debe cambiar cuando se ha añadido una característica a la aplicación.
6. El número se debe cambiar cuando se realiza corrección de errores de la aplicación.

**Diseño de interfaz adaptativa** El diseño adaptativo es uno de los estándares que se caracteriza de gran valor en el desarrollo de aplicaciones y sitios web siendo por el valor de la visualización en diferentes dimensiones, adaptándose en el contexto que se encuentre sea un escritorio como aplicación web o en el entorno móvil.

El enfoque al diseño adaptativo se debe a las posibilidades de mejorar la experiencia e interacción del usuario [38].

La aplicación utiliza como base en el desarrollo de interfaces adaptativas a Bootstrap, dado que es un marco de desarrollo que viene integrado al framework Ionic que ya contempla varios de los aspectos y elementos necesarios en los procesos de diseño, permitiendo un ahorro en recursos e implementaciones.



**Figura 52:** Vista de pantalla. Horizontal.  
Fuente: Elaborado por el investigador.

**Seguridad** Para aplicación en caso de que un usuario inició sesión el API-REST genera un token basado en el correo de registro, el cual se utilizará para mantener la sesión activa en el dispositivo. Únicamente los usuarios que iniciaron sesión tienen habilitadas las funciones de gestión de datos y sincronización.

**Rendimiento** En el entorno móvil y las aplicaciones web llegan a tener niveles de retardo que pueden ser significativos en el lado del cliente dado por el tiempo de respuesta y los recursos que dispone el dispositivo. Las pruebas se realizaron en un dispositivo móvil de modelo *SONY M4 Aqua* procesador Qualcomm 615 64 bits (Octacore) de 2 núcleos a 1,7 GHz de velocidad, memoria RAM de 2GB, memoria interna de 8GB.

En la figura 53 se puede apreciar el tiempo de carga en el dispositivo mencionado, con un retraso en cuanto la carga de plugins de la aplicación, usualmente se ha visto este tipo de retraso en dispositivos móviles que tienen el almacenamiento interno mayor a 85%.

```

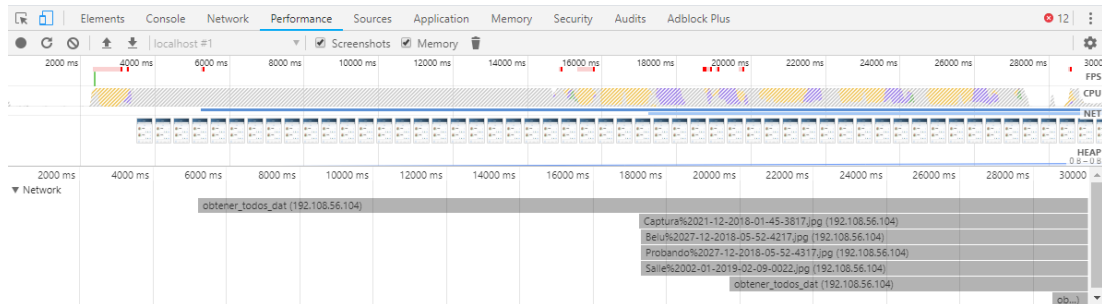
Angular is running in the development mode. Call enableProdMode() to enable the production mode.
deviceready has not fired after 5 seconds.
Ionic Native: deviceready did not fire within 5000ms. This can happen when plugins are in an inconsistent state. Try removing plugins from plugins/ and reinstalling them.
Ionic Native: deviceready event fired after 5579 ms
OPEN database: _ionicstorage
new transaction is queued, waiting for open operation to finish
OPEN database: _ionicstorage - OK
DB opened: _ionicstorage

```

**Figura 53:** Carga de recursos en dispositivo móvil.  
Fuente: Elaborado por el investigador.

Respecto a la petición de datos locales, es decir, sin conexión, como se indicó en el proceso de petición de datos, verifica la disponibilidad de red, posteriormente se realiza la petición, en la figura 54 se muestra el tiempo de respuesta de petición de datos.





**Figura 54:** Carga de recursos en entorno de producción.  
Fuente: Elaborado por el investigador.

Se utilizó *DevTools* de Chrome como herramienta para la depuración y medición de tiempos en la ejecución de la aplicación.

#### 4.4.4. Transición

##### 4.4.4.1. Diseño Pruebas

Las pruebas de funcionalidad tienen como fin validar que el sistema cumple los requisitos básicos de funcionamiento esperado y permitir que el usuario determine la aceptación del sistema.

Mediante gráficos estadísticos se muestra que la aceptación y valoración por parte de los usuarios está en un nivel admisible; a partir de esto se llevó a cabo un proceso de refinamiento de los procesos y componentes de la aplicación.

Para medir el tiempo de reacción y funcionamiento del sistema se utilizó herramientas de depuración, en este caso *DevTools* de Google Chrome, ya que cada dispositivo tienen diferentes características de funcionamiento, se ha optimizado la carga de plugins para mejorar la experiencia, así como implementación de componentes para grandes volúmenes de datos, en caso de vista de listas, se utilizó *Virtual-Scroll* diseñado especialmente para estos casos. A partir de ello, se determinó qué procesos se encontraban ralentizando la interfaz de la aplicación y la experiencia del usuario. Se tomaron medidas de mejora en aspectos como la usabilidad de las redes sociales y del proceso de carga plugins. Como pantalla principal se definió la vista general al mapa con los datos georreferenciales, en un principio la pantalla principal fue la lista de los lugares, se optó por la carga del mapa por optimización de recursos, a partir de esa carga solo se realizará una vez la carga de las teselas vectoriales a partir del archivo *MBTILES*, se agregó el condicional de si no resuelve en un intervalo de 3 segundos la ubicación del usuario, se toman parámetros de la última ubicación, dicho proceso no retrasará la carga del mapa.

Una vez mejorados los procesos, en base a las pruebas realizadas de la aplicación, se ha actualizado el sistema a la versión estable final.

<b>Casos de uso</b>	<b>Pantalla</b>	<b>Datos de entrada</b>	<b>Resultado</b>	<b>Cumplimiento</b>	<b>Observaciones</b>
Registrar usuario	Ver figura [22], Ver figura [23]	- Email - Password	Token de sesión, sesión activa	SI	
Inicio de sesión	Ver figura [22], Ver figura [23]	- Email - Password	Token de sesión, sesión activa	SI	
Visualizar datos	Ver figura [24], Ver figura [26], Ver figura [28]		Lista de puntos agregados registrados en la base de datos	SI	
Registrar un nuevo punto	Ver figura [33], Ver figura [34]	- Latitud, Longitud - Nombre - Descripción - Imagen	Registro local y remoto de la información del nuevo punto	SI	
Modificar un punto	Ver figura [33]	- Nombre - Descripción - Imagen	Modificación local y remota de la información del punto seleccionado	NO	La imagen en base64 se debe reemplazar en el servidor para evitar duplicidad de imágenes(corregido)
Buscar datos	Ver figura [26], Ver figura [22]	- Nombre	Lista de coincidencias según el criterio de búsqueda	SI	
Compartir en las redes sociales un punto	Ver figura [28], Ver figura [31]		La información de un punto seleccionado publicada en las redes sociales seleccionadas	NO	Verificar la aplicación móvil de red social deba estar instalada para realizar la acción de compartir(corregido)

## CAPÍTULO 5

### Conclusiones y Recomendaciones

#### 5.1. Conclusiones

La implementación de este proyecto permitió obtener una solución multiplataforma con un origen de datos del mapa del Ecuador con alta calidad visual. No se requiere de una conexión permanente a Internet; esto permite utilizar el aplicativo para gestionar datos georreferenciados en lugares sin cobertura de red. Las principales conclusiones a las que se determinó, después de realizar este trabajo, son:

- Se determinó como herramienta de desarrollo Ionic Framework sobre React debido a que dispone de componentes adaptativos por lo cual resultó más fácil el diseño de interfaces así como la utilización de menos plugins por ende mejor tiempo de respuesta, las características que nos brinda, esencialmente la aplicación toma un perfil *responsive* por sus componentes predeterminados, dichos componentes han sido diseñados en base a otras aplicaciones populares como Facebook, Whastapp, Instagram entre otras, el propósito de tomar como base ese tipo de aplicaciones, es por la experiencia de usuario y por la facilidad de implementación, ya que tiene una amplia documentación debido a su comunidad en crecimiento.

Otro motivo por la elección de la herramienta es por el plugin visor de mapas que se ha utilizado, debido a que la biblioteca usa código *Javascript*, la cual usa tecnologías web como CSS , HTML5 y Sass. En un principio la biblioteca utilizada fue diseñada para *Apache Cordova*, por lo que para ser utilizada en la aplicación no tuvo mayor inconveniente, debido que Ionic utiliza Cordova como motor.

- Se construyó la aplicación móvil con gestión de datos georreferenciados, usando el *storage del dispositivo* para mantener la sesión activa del usuario, una base de datos *sqlite* para el almacenamiento de datos provenientes de la base de datos remotas o anotaciones sin conexión, la cual se comunica con un API-REST, encargada de realizar las peticiones a la base de datos remota, en este caso, se utilizó la base de datos *Postgresql* por el tipo de dato *point* necesario para realizar anotaciones (latitud, longitud), los cuales en la recuperación de datos, la aplicación los receipta, procesa y ubica marcadores

en el mapa.

Es posible omitir el inicio de sesión, la aplicación muestra los datos de todos los usuarios que han registrado un punto, también está disponible la página para buscar un punto por su nombre y las opciones de compartir la ubicación por las redes sociales; Facebook, Whastapp e Instagram.

La aplicación es capaz de añadir un nuevo punto en el mapa, el cual usa un formulario para el ingreso de datos necesarios para el registro, la gestión del mismo está disponible únicamente si existe un usuario con una sesión activa.

## 5.2. Recomendaciones

- Para futuros trabajos se sugiere la investigación en lo que se refiere a la implementación de descarga de regiones a petición del usuario, el cual la aplicación tenga la posibilidad de tener varios orígenes de datos.
- Para mejora de la aplicación, reducir el tiempo de respuesta en cuanto a la carga del mapa vectorial, los plugins utilizados dejaron de tener soporte, por lo que se sugiere utilizar variantes para obtener un mejor rendimiento.
- Se aconseja aprovechar al máximo el componente *javascript* MapBoxGL JS por cuanto a mostrar edificaciones en 3D, el componente tiene la capacidad de usar el archivo base del estilo para generar edificaciones en 3D.
- Para determinar una mayor alcance de la aplicación se sugiere la implementación de funciones de enrutamiento para agregar valor a la aplicación, se ha visto durante la investigación del presente proyecto usar la biblioteca *Turf.js* es una excelente opción para dicho proceso.
- Como trabajo de mejora a posterior se recomienda optimizar el proceso de sincronización de datos para grandes cantidades de datos, así como peticiones concurrentes, debido al alcance que puede tener la aplicación
- Como característica adicional automatizar el proceso de sincronización de datos utilizando observables, con intervalo de tiempos y verificación de disponibilidad de red.
- Se sugiere la migración del proyecto a Ionic 4 debido a las mejoras en experiencia de usuario, los componentes, ampliación y mejora del motor.
- En futuros trabajos de investigación se recomienda el estudio del rendimiento y escalabilidad del entorno de programación Ionic .

- Como implementación posterior se recomienda analizar el nivel de carga de la concurrencia de las transacciones de datos, una estructura de base de datos distribuida para reducir carga de procesamiento y tener una mejor respuesta y ejecución.
- Se sugiere la migración al entorno de Ionic4 debido a la mejora de respuesta y velocidad en cuanto a componentes de nativos (sensores, cámara, estado de red, almacenamiento), en conjunto, los componentes predeterminados del entorno constan con diseños modernos y adaptativos.

## Bibliografía

- [1] Devjoker, "Rest api - patrón de diseño," 2019. [Online]. Available: <http://www.devjoker.com/contenidos/articulos/525/Patron-MVC-Modelo-Vista-Controlador.aspx>
- [2] Y. M. Jimenez, "Arquitectura de angular," 2019. [Online]. Available: <https://medium.com/@yonem9/angular-qu%C3%A9-son-los-m%C3%B3dulos-y-c%C3%B3mo-se-refactoriza-una-aplicaci%C3%B3n-9457550e8e9>
- [3] IonicFramework, "Ionic framework - instalation," 2019. [Online]. Available: <https://ionicframework.com/docs/intro/installation/>
- [4] Arcgis, "Imagen raster - definición," 2019. [Online]. Available: <http://desktop.arcgis.com/es/arcmap/10.3/manage-data/raster-and-images/what-is-raster-data.htm>
- [5] Vexels, "Datos vectoriales vs raster," 2019. [Online]. Available: <https://www.vexels.com/our-graphics/>
- [6] N. D. Lisandro, "Desarrollo de aplicaciones moviles multiplataforma," 2017. [Online]. Available: [http://sedici.unlp.edu.ar/bitstream/handle/10915/60497/Documento\\_completo\\_.pdf-PDFA.pdf?sequence=3](http://sedici.unlp.edu.ar/bitstream/handle/10915/60497/Documento_completo_.pdf-PDFA.pdf?sequence=3)
- [7] C. Romero, "Aplicacion multiplataforma para la gestion de archivos," 2016. [Online]. Available: <https://riunet.upv.es/handle/10251/48383>
- [8] A. Villacis, "La tecnologia android y su incidencia en el desarrollo de una aplicacion movil para la geo-localizacion de los centros asistenciales y farmacias de turnos para la direccion provincial de salud los rios ubicada en la ciudad de babahoyo," 2016. [Online]. Available: <http://dspace.utb.edu.ec/bitstream/49000/1206/1/T-UTB-FAFI-SIST-000113.pdf>
- [9] Argis, "Georreferenciación y sistemas de coordenadas," 2018. [Online]. Available: <http://resources.arcgis.com/es/help/getting-started/articles/026n000000s000000.htm>
- [10] N. Ortiz, "Arquitectura y diseño de bases de datos moviles," 2017. [Online]. Available: <https://revistas.udistrital.edu.co/ojs/index.php/tia/article/download/4296/6014>

- [11] Adatum, "Bases de datos android y sus características," 2018. [Online]. Available: <https://developer.android.com/training/basics/data-storage/databases.html?hl=es-419>
- [12] Couchdb, "Couchdb - definición - instalación, url = <http://docs.couchdb.org/en/stable/intro/overview.html>," 2019.
- [13] G. Gevelopers, "Almacenamiento offline en local storage," 2019. [Online]. Available: <https://developers.google.com/web/fundamentals/instant-and-offline/web-storage/offline-for-pwa?hl=es>
- [14] Microsoft, "App service movile," 2019. [Online]. Available: <https://docs.microsoft.com/es-es/azure/app-service-mobile/app-service-mobile-offline-data-sync>
- [15] K. O. Seidler, "Xampp - introducción," 2019. [Online]. Available: <https://www.apachefriends.org/es/about.html>
- [16] PostGIS, "Spatial and geographic objects for postgresql," 2019. [Online]. Available: <https://postgis.net/>
- [17] Community, "The pgadmin development," 2019. [Online]. Available: <https://www.pgadmin.org/>
- [18] M. Masse, *REST API design rulebook*. .o'Reilly Media, Inc.", 2011.
- [19] R. T. Fielding, "Architectural styles and the design of network-based software architectures," Ph.D. dissertation, University of California, Irvine, 2000.
- [20] Codeigniter, "Codeigniter - documentación," 2019. [Online]. Available: [https://www.codeigniter.com/user\\_guide/overview/features.html](https://www.codeigniter.com/user_guide/overview/features.html)
- [21] O. S. Map, "Openstreetmap," 2019. [Online]. Available: <https://www.openstreetmap.org/about>
- [22] O. M. Tiles, "Openmaptiles," 2019. [Online]. Available: <https://openmaptiles.org/about/>
- [23] O. Fonts, "Mapa vectorial sin conexión," 2019. [Online]. Available: <http://fonts.cat/siglibre2018/#14>
- [24] Mapbox, "Mapbox - introducción," 2019. [Online]. Available: <https://docs.mapbox.com/mapbox-gl-js/api/>

- [25] Klokantech, "Klokantech basic," 2019. [Online]. Available: <https://github.com/openmaptiles/klokantech-basic-gl-style>
- [26] T. Inc, "Bootstrap," [Web; accedida el 10-05-2016]. [Online]. Available: URL{<http://getbootstrap.com/>}
- [27] T. Firdaus, *Responsive Web Design by Example Beginner's Guide*. Packt Publishing Ltd, 2013.
- [28] N. Jain, P. Mangal, and D. Mehta, "AngularJS: A Modern MVC Framework in JavaScript," *Journal of Global Research in Computer Science*, vol. 5, no. 12, pp. 17–23, 2015.
- [29] G. Inc., "AngularJS," 2016, [Web; accedida el 11-08-2016]. [Online]. Available: <https://angularjs.org/>
- [30] M. García Salinero, "Grid de datos desarrollo con AngularJS utilizando el patrón Modelo Vista-Controlador," 2014.
- [31] G. Inc., "Documentation AngularJS," 2016, [Web; accedida el 11-08-2016]. [Online]. Available: <https://docs.angularjs.org>
- [32] Apache, "Apache cordova - instalación," 2019. [Online]. Available: <https://cordova.apache.org/docs/es/8.x/guide/overview/index.html>
- [33] Comunidad, "Repositorio node.js package manager," 2019. [Online]. Available: <https://www.npmjs.com/>
- [34] MapBox, "Mapa vectorial," 2019. [Online]. Available: <http://www.4rsoluciones.com/blog/bitmaps-y-graficos-vectoriales-cuales-son-las-diferencias-2/>
- [35] Mapbox, "Mapbox vector tiles," 2019. [Online]. Available: <https://docs.mapbox.com/vector-tiles/>
- [36] C. Kacerguis, "Rest server - codeigniter," 2019. [Online]. Available: <https://github.com/chriskacerguis/codeigniter-restserver>
- [37] B. Mallick and N. Das, "An Approach to Extended Class Diagram Model of UML for Object Oriented Software Design," *International Journal of Innovative Technology & Adaptive Management (IJITAM)*, vol. 1, no. 2, 2013.



- [38] M. Mazza, "Diseño web responsivo (responsive web design)," 2019. [Online]. Available: [http://xn--diseowebresponsivo-q0b.org/?utm\\_source=redirects&utm\\_medium=dise%25C3%25B1owebresponsivo.com.ar&utm\\_campaign=301\\_Redirects](http://xn--diseowebresponsivo-q0b.org/?utm_source=redirects&utm_medium=dise%25C3%25B1owebresponsivo.com.ar&utm_campaign=301_Redirects)

## **Anexos y Apéndices**

## Anexo A

### Encuesta para determinación de usabilidad de aplicaciones de datos georreferenciales offline

---

#### Objetivo:

Obtener información de aceptación de aplicaciones sobre anotación de datos georreferenciales, y determinar las posibles características a implementar para el desarrollo de la aplicación según la tendencia tecnológica.

#### Instrucciones:

- Lea detenidamente cada pregunta y marque con una X la respuesta que considere correcta.
  - Solo se permite marcar una opción por pregunta
- 

1. ¿Qué medio utiliza para encontrar ubicaciones en el mapa?

Sitios web de mapas       Aplicaciones Móviles       Personas

2. ¿Con qué frecuencia utiliza aplicaciones para ubicar un lugar no conocido en el mapa?

Siempre     Casi siempre     Ocasionalmente     Muy pocas veces

3. ¿Qué aplicación utiliza para hacer anotaciones sobre el lugar?

Mensajes de texto     Recordatorios     Stick notes     Fotografías

4. ¿Considera importante disponer de una aplicación con mapa offline en su dispositivo móvil?

Si       No

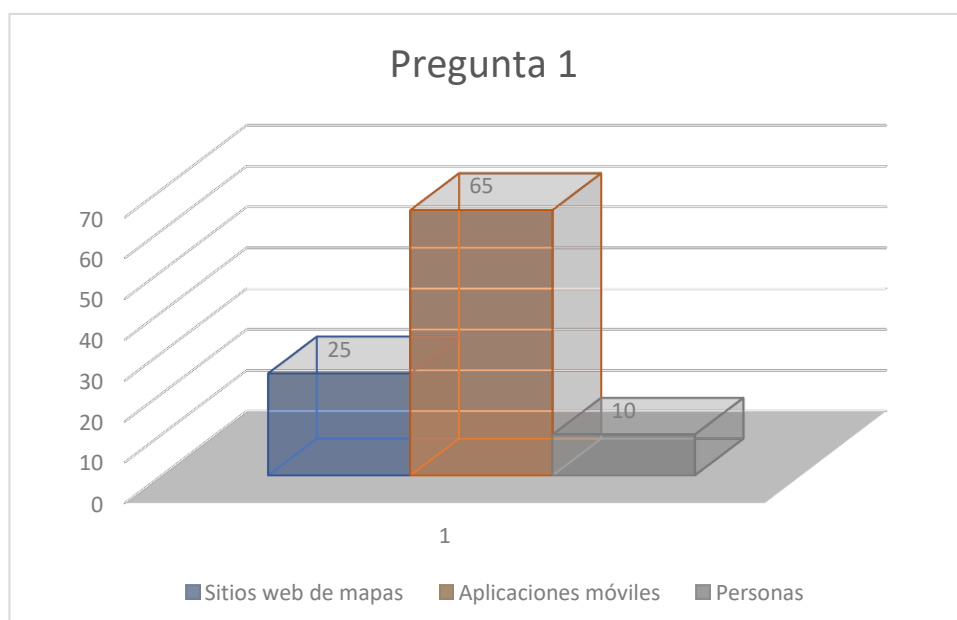
5. ¿Considera importante tener la capacidad de agregar sus propios puntos a un mapa en una aplicación móvil?

Si       No

## ANÁLISIS E INTERPRETACIÓN DE LOS RESULTADOS DE LA ENCUESTA PARA DETERMINACIÓN DE USABILIDAD DE APLICACIONES DE DATOS GEORREFERENCIALES OFFLINE

1. ¿Qué medio utiliza para encontrar ubicaciones en el mapa?

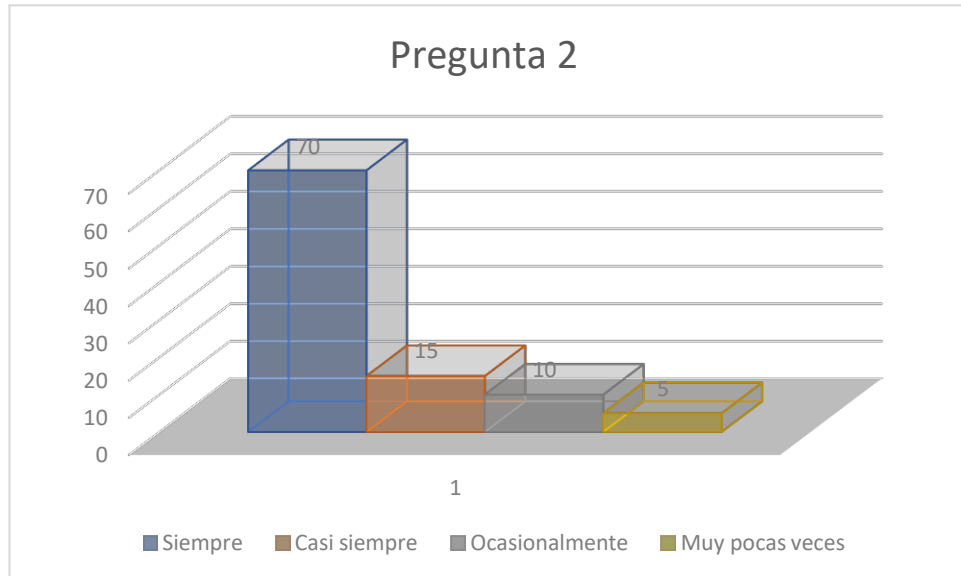
Opciones	Frecuencia	Porcentaje
Sitios web de mapas	25	25%
Aplicaciones móviles	65	65%
Personas	10	10%
TOTAL	100	100%



**Análisis:** Del total de la población, el 65% menciona que para encontrar ubicaciones en el mapa usa aplicaciones móviles, 25% acude a los sitios web para ubicar una dirección o lugar y el 10% pregunta directamente a personas que conozcan sobre la zona.

2. ¿Con qué frecuencia utiliza aplicaciones para ubicar un lugar no conocido en el mapa?

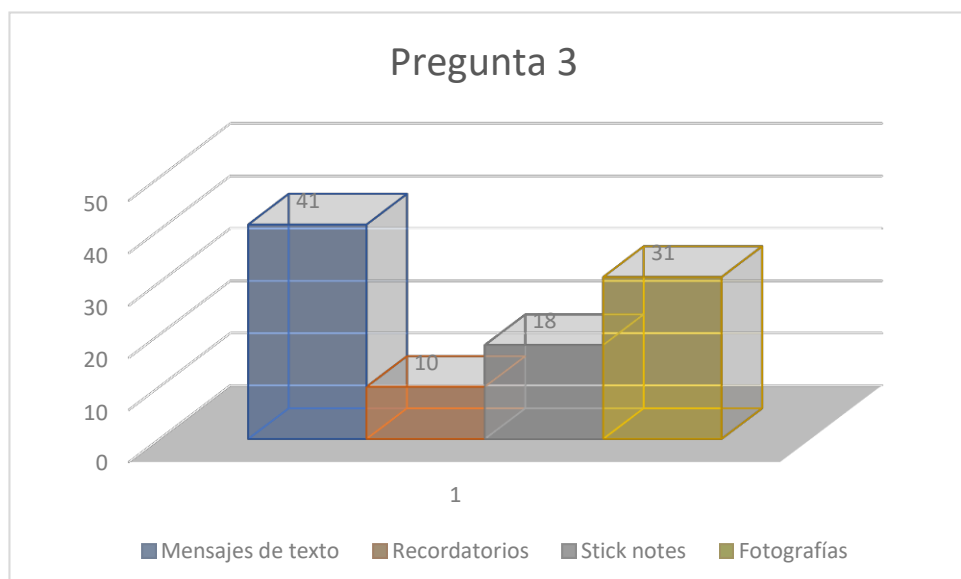
Opciones	Frecuencia	Porcentaje
Siempre	70	70%
Casi siempre	15	15%
Ocasionalmente	10	10%
Muy pocas veces	5	5%
TOTAL	100	100%



**Análisis:** Del total de los encuestados el 70% asegura que utilizan un dispositivo móvil para ubicar un punto no conocido en el mapa, 15% usa el móvil para dicha tarea, el 10% pertenece al grupo que en ocasiones ubican un punto no conocido y el 5% afirma que el dispositivo móvil no es una herramienta para realizar la ubicación del punto en el mapa.

3. ¿Qué aplicación utiliza para hacer anotaciones sobre el lugar?

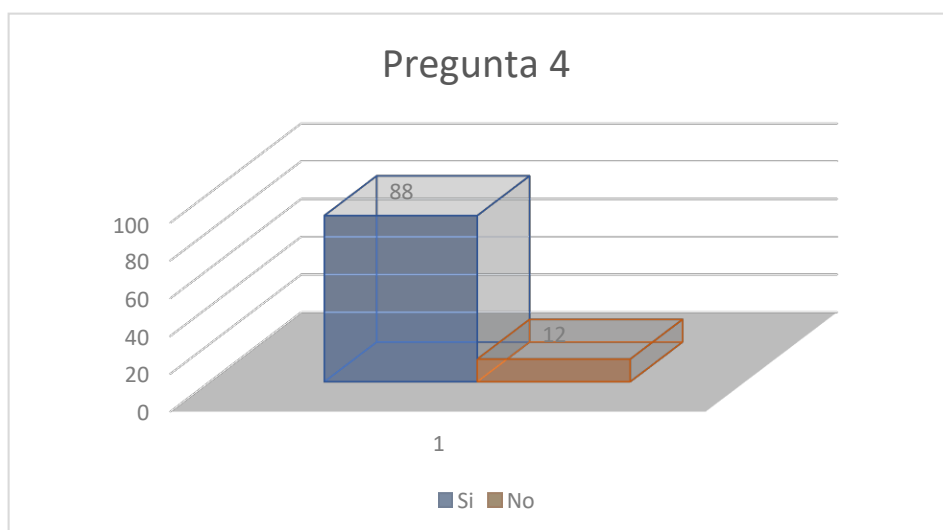
Opciones	Frecuencia	Porcentaje
Mensajes de texto	41	41%
Recordatorios	10	10%
Stick notes	18	18%
Fotografías	31	31%
TOTAL	100	100%



**Análisis:** Del total de las personas encuestadas el 41% realiza anotaciones sobre lugares a manera de recordatorio para su posterior utilidad por el usuario, el 10% crea un recordatorio en la aplicación de calendario el cual es una aplicación predeterminada de un dispositivo móvil, el 18% realiza anotaciones en un Stick Notes sea físico(papel) o aplicación que se debe descargar y el 31% toma una fotografía del lugar para su posterior utilidad.

4. ¿Considera importante disponer de una aplicación con mapa offline en su dispositivo móvil?

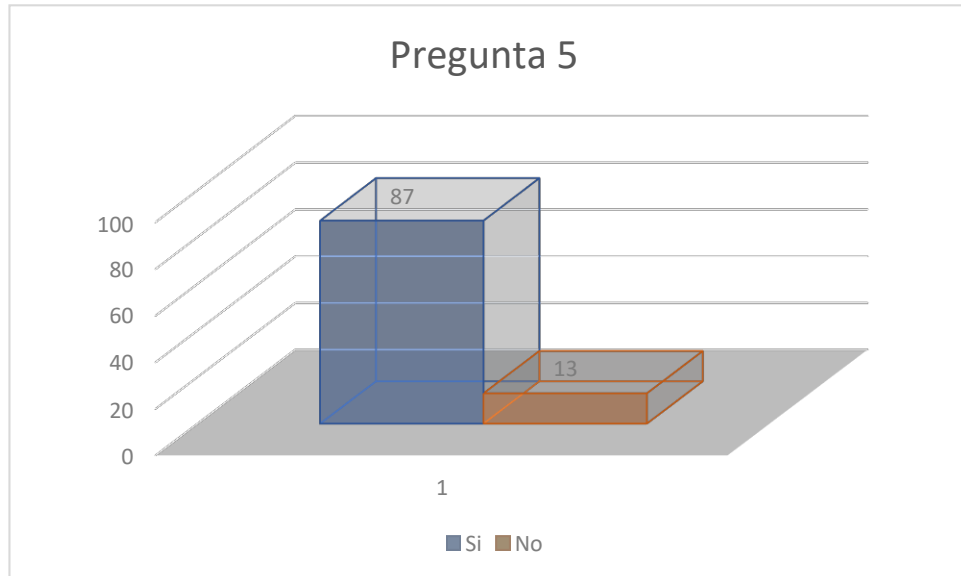
Opciones	Frecuencia	Porcentaje
Si	88	88%
No	12	12%
TOTAL	100	100%



**Análisis:** Del total de los encuestados el 88% está de acuerdo que es de utilidad disponer de una aplicación con un mapa sin conexión y el 12% no está de acuerdo con disponer de un mapa offline.

5. ¿Considera importante tener la capacidad de agregar sus propios puntos a un mapa en una aplicación móvil?

Opciones	Frecuencia	Porcentaje
Si	87	87%
No	13	13%
TOTAL	100	100%



**Análisis:** Del total de los encuestados el 87% considera el es de utilidad disponer de una aplicación en la cual se puedan realizar sus propias anotaciones y el 13% considera que no es importante disponer de dicha aplicación con las características de anotaciones y mapa offline.

## Anexo B

### Diccionario de Datos

#### Leyenda

Leyenda	
Término	Definición
PK	Primary key
NN	Not null
UN	Unique
AI	Autoincrement

#### Tabla “usuarios”

Column name	DataType	PK	NN	UN	AI	Default	Comment
id_usu	INT(10)						
correo_usu	VARCHAR(255)						
contrasena_usu	VARCHAR(255)						
token	VARCHAR(255)						

**Tabla 10:** Diccionario de datos. Tabla: **usuarios**

#### Tabla “datos”

Column name	DataType	PK	NN	UN	AI	Default	Comment
id_dat	INT						
punto_dat	VARCHAR(MAX)						
nombre_dat	VARCHAR(MAX)						
descripcion_dat	VARCHAR(MAX)						
fecha_dat	DATETIME						
imagen_dat	VARCHAR(MAX)						

**Tabla 11:** Diccionario de datos. Tabla **datos**



## GLOSARIO DE TÉRMINOS

**API** Interfaz de programación de aplicaciones (Application Programming Interface)

**BD** Base de Datos.

**CSS** Cascading Style Sheets es un lenguaje de diseño gráfico para definir y crear la presentación de un documento estructurado escrito en un lenguaje de marcado.

**DOM** Modelo de Objeto de Documento (Document Object Model). Es la estructura de objetos que genera el navegador cuando se carga un documento y se puede alterar mediante Javascript para cambiar dinámicamente los contenidos y aspecto de la página.

**GPS** Global Positioning System.

**HTML5** HyperText Markup Language, versión 5.

**HTTPS** Protocolo seguro de transferencia de hipertexto (Hypertext Transfer Protocol Secure)

**iOS** iPhone OS.

**JS** Javascript es un archivo de texto plano que contiene scripts.

**MBTiles** MapBox Tiles.

**MVC** Modelo-Vista-Controlador.

**NPM** Módulo de manejo de paquetes de NodeJS (Node Package Module).

**Observers** Para comunicar servicios y componentes.

**Open Source** Código abierto.

**OSM** Open Street Maps.

**PHP** Hypertext Preprocessor.

**Promises** Realización de tareas asíncronas.

**RESTful** Transferencia de Estado Representacional (Representational State Transfer).

**SIG** Sistema de Información Geográfica.

**URL** Uniform Resource Locator

**Usuario** Persona que usará el sistema para gestionar procesos.

**UTM** Universal Transversal de Mercator