FACULTAD DE INGENIERÍAS Y CIENCIAS AGROPECUARIAS

DISEÑO Y DESARROLLO DE UN PROTOTIPO DE APLICACIÓN WEB Y MÓVIL NATIVA ANDROID PARA EL MONITOREO DE SEGURIDAD EN EL DISTRITO METROPOLITANO DE QUITO A TRAVÉS DE LA RED SOCIAL TWITTER.

Trabajo de Titulación presentado en conformidad con los requisitos establecidos para optar por el título de Ingenieros en Sistemas de Computación e Informática

Profesor Guía

Ing. Anita Elizabeth Yánez Torres

Autores

Adriana Abigail Amores Bassante
Cristian Aldheir Suárez Loaiza

Año

2016

DECLARACIÓN DEL PROFESOR GUÍA

“Declaro haber dirigido este trabajo a través de reuniones periódicas con los estudiantes, orientando sus conocimientos y competencias para un eficiente desarrollo del tema escogido y dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación”

……

Anita Elizabeth Yánez Torres

Ingeniera en Sistemas

C.I 1802462216

DECLARACIÓN DE AUTORÍA DE LOS ESTUDIANTES

“Declaramos que este trabajo es original, de nuestra autoría, que se han citado las fuentes correspondientes y que en su ejecución se respetaron las disposiciones legales que protegen los derechos de autor vigentes”

……	……
Adriana Abigail Amores Bassante	Cristian Aldheir Suárez Loaiza
C.I 0502896459	C.I 1721745170

AGRADECIMIENTOS

“Damos gracias a nuestros padres y hermanos por el apoyo infinito brindado y, a nuestros amigos por la ayuda otorgada que impulsó la culminación de nuestra carrera profesional.

Agradecemos de igual manera a la Ing, Anita Yánez por guiarnos y por brindarnos sus conocimientos en todo este proceso de desarrollo del presente trabajo de titulación.”

DEDICATORIA

“Dedicamos el presente trabajo de titulación a nuestros padres por su infinito esfuerzo y amor, a nuestros hermanos por sus enseñanzas de vida y a nuestros amigos que son la fuerza para alcanzar metas futuras.

RESUMEN

En el presente proyecto se desarrolló un prototipo de sistema web y un prototipo de aplicación móvil, los prototipos realizados permiten dar aviso de delitos cometidos en la ciudad de Quito por medio de la red social Twitter. Para reportar delitos los usuarios redactarán tweets con el hashtag (#) QuitoSeguridad, seguido de la dirección, usando hashtags, donde hayan sido víctimas de un delito o presenciado uno. Se recopilan los datos tomados de publicaciones de usuarios en la red social por medio de hashtags (#) para colocar toda la información en un mapa de calor y de estar forma se muestran las zonas de mayor peligro en la ciudad de Quito.

Este proyecto puede ser usado como una herramienta para control de seguridad en la ciudad de Quito que debe ser utilizada con responsabilidad por cada ciudadano y de esta manera poder construir una ciudad más segura y alertar de una manera eficaz los delitos cometidos en la ciudad. Además puede contribuir a tener una reacción rápida por parte de las autoridades para enfrentar de mejor manera los delitos que ocurran en el Distrito Metropolitano de Quito.

Actualmente en un mundo dominado por la tecnología es necesario innovar en la creación de herramientas que ayuden a las personas en el día a día, previniendo desastres para promover una cultura de ayuda mutua y permanente. El reto para nosotros, los involucrados en el área de tecnología es construir productos de calidad y eficiencia que sean mantenibles a lo largo del tiempo y que permitan construir un mundo con mejor calidad de vida.

.

ABSTRACT

In this project a prototype web system and a prototype mobile application has been developed, prototypes allow give notice of crimes committed in the city of Quito through Twitter social network. To report crimes users will write tweets with the hashtag (#) QuitoSeguridad, followed by the address, using hashtags, which have been victims of a crime or witnessed one. Data is collected from users publications of the social network through hashtags (#) to put all information on a heat map and highlight areas of greatest risk in the city of Quito.

This project can be used as a tool to control security in the city of Quito which must be used with responsibility by every citizen and thus can build a safer city and alert crimes effectively committed in the city. It can also help to have a quick reaction from the authorities to fight crimes that occur in the Metropolitan District of Quito.

Currently in a world dominated by technology it is necessary to innovate in the creation of tools that help people in everyday life, preventing disasters and promoting a culture of mutual and permanent support. The challenge for us, those involved in the technology area is to build products of quality and efficiency that are maintainable over time and allow building a world with better quality of life.

Índice

Índice de figuras

Índice de tablas

Introducción

Alcance

El alcance de este trabajo de titulación es desarrollar un prototipo de aplicación web que permitirá apreciar los niveles de inseguridad en el Distrito Metropolitano de Quito, a través de un mapa de calor, basado en una recopilación de datos, tomados de publicaciones de usuarios de la red social Twitter mediante el uso de *hashtags* (#) y la ubicación proporcionada por un persona afectada cuando haya sido víctima de un ataque o presenciado uno. Se utilizará una interfaz de programación de Twitter, para la toma de información en los *tweets* con los *hashtags* de la dirección del lugar del incidente.

Además desarrollar un prototipo de aplicación móvil Android que muestre dicha información, basada en la posición del usuario usando geolocalización, basado en el sistema GPS. Cada dato será tomado en cuenta y actualizado paulatinamente con la llegada de nuevos datos de la red social Twitter. Se creará una cuenta propia dentro de la red social para la promoción de esta iniciativa y para dar aviso de nuevos incidentes, así también se mostrarán los *tweets* originales de los informantes y un mapa en el cual se muestre las zonas de mayor inseguridad, reportados por los *hashtags* de cada usuario dentro de cualquier dispositivo con conexión a Internet.

Justificación

Seguridad en el Distrito Metropolitano de Quito

En el Ecuador entre los periodos enero-abril del 2010 y enero-abril del 2011, se realizó una comparación en la cual se evidenció que la ciudad de Quito se registraba como la ciudad más peligrosa del país como indica la Figura 1.

Incidencia de delitos

Delitos	Guayaquil	Quito
robo a personas	1,6% ↑	54% ↑
robo a domicilios	-29% ↓	14% ↑
robo a locales comerciales	-45% ↓	7,1% ↑
robo de automóviles	2,4% ↑	-3,1% ↓
robo de motocicletas	-6,7% ↓	-46% ↓
asaltos en carreteras	-48,9% ↓	43% ↑
homicidios	-7,6% ↓	-14% ↓

Comparativo entre enero - abril de 2010 y enero - abril de 2011

Figura 1. Incidencia de delitos.

Tomado de (Inseguridadenquito, s.f).

Se hicieron 234 mil denuncias por delitos en Ecuador hasta marzo del 2011. De esa cifra, unas 12.700 denuncias se refieren a crímenes contra la vida, que incluyen más de 4.000 asesinatos y homicidios, y unas 8.000 lesiones, mientras que las relacionadas con el narcotráfico sumaron unas 5.800. En su mayoría, las denuncias son de faltas contra la propiedad, que ascienden a más de 92.000 robos y hurtos (Torres, P. 2011).

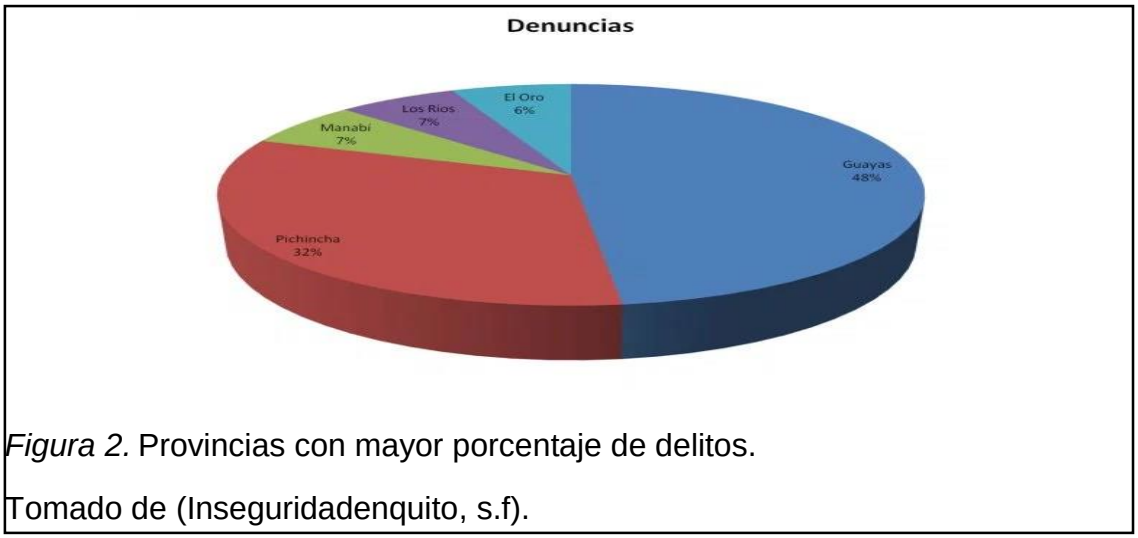


Figura 2. Provincias con mayor porcentaje de delitos.

Tomado de (Inseguridadenquito, s.f).

Un informe de monitoreo del Observatorio de Seguridad Ciudadana del DMQ, en el año 2013, refleja una reducción en la incidencia de delitos como homicidios, robo a personas, robo de autos, robo a comercios. Durante el primer trimestre del 2013 se mostró una reducción en las estadísticas de delitos

registrados en el Distrito Metropolitano de Quito (DMQ), con los datos comparados en el mismo período del 2012. Los delitos de homicidio se redujeron en un 38% y robo a personas en un 40%, además otros delitos presentan considerables reducciones, como robo de autos se reduce en un 27% y robo a comercios en un 24%. De este modo el robo a personas representa el 45% de los delitos, a continuación el robo a domicilios con un 21%, robo a comerciales 15% y el robo de autos con 11% (Observatorio Metropolitano de Seguridad, 2013).

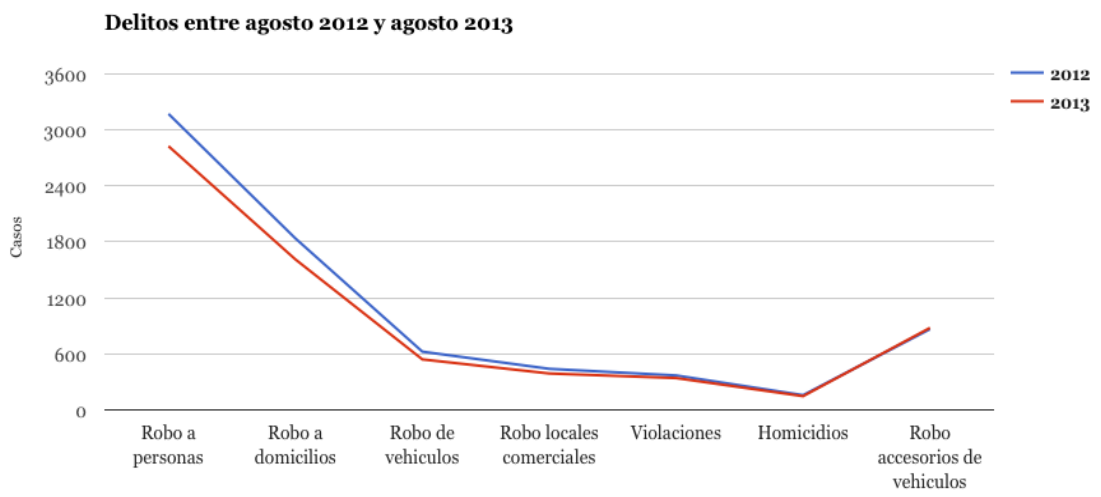


Figura 3. Estadísticas de delitos registrados entre el periodo agosto 2012 - agosto 2013.

a. Gráfico realizado a partir de las estadísticas mencionadas a continuación.

Según las estadísticas de la Comisión de Seguridad Integral, basadas en datos de la fiscalía, y representadas en la Figura 3, se presenta una reducción en el mes de agosto del 2012 a comparación de agosto del 2013. En agosto del 2012 el robo a personas se registró 3.171 casos y en agosto del 2013 fueron 2.825 casos. En robo a domicilios se registraron 1.837 casos en agosto del 2012, para el mismo mes del 2013 hubo 1.612 casos. En robo de vehículos a agosto del 2012 se denunciaron 624 casos y a agosto del 2013 bajó a 541 casos. El robo a locales comerciales en 2012 fue de 440 casos y en el 2013 llegó a 390. Entre los casos de violaciones se registraron 369 casos en el 2012,

mientras que en el 2013 fueron de 341. Las cifras de homicidios y asesinatos en agosto de 2012 fueron de 158 y para agosto del 2013 fue de 147. Sin embargo, el robo de accesorios a vehículos se incrementó en 1,62%, en agosto de 2012 hubo 865 casos y para agosto del 2013 ascendió a 879 casos (La Hora, 2013).

El sistema de seguridad ECU-911 implementado por el gobierno ecuatoriano está compuesto por centros provinciales y regionales. El centro ubicado en Quito atiende también las peticiones de las provincias de Napo y Orellana. El sistema que integra entidades como Policía Nacional, Bomberos, Cruz Roja, entre otros, informó que a finales del 2013, 505.102 diferentes tipos de emergencias fueron atendidas (Agencia Pública de Noticias del Ecuador y Suramérica (ANDES), 2014).

De acuerdo al diario El Comercio (2014), entre enero y mayo del 2014 un 61.59% de los delitos se registraron en la vía pública. El Observatorio Metropolitano de Seguridad Ciudadana (OMSC) afirma que la mayoría de los robos se suscitan en horas de la noche, además los jueves y viernes son los días en que más se reportan estos delitos en la semana. El último reporte de esta entidad detalla que, entre enero y mayo del 2014, existieron 2.773 casos.

La intimidación, amenazas y armas cortopunzantes son los medios que estas personas utilizan para realizar esta clase de delitos, suelen aprovecharse también de la congestión vehicular y el flujo de peatones en zonas concurridas. La Policía Nacional con base en la zona 9, D.M.Q., está desplegando estrategias de control, prevención y reacción antidelincuencial, con la permanente generación de operativos las 24 horas del día a lo largo y ancho de la ciudad capital, sus valles y comunidades rurales (Ministerio del Interior, 2014).

En los cuatro primeros meses del 2015 se ha registrado un decrecimiento del 23,6% en homicidios/asesinatos en comparación al mismo período del 2014. Según las estadísticas de seguridad ciudadana, otorgadas por la Coordinación

de Investigación y Estadísticas del Ministerio del Turismo la Zona 2 integrada por Napo, Pichincha y Orellana registran una reducción del 35,3% en homicidios durante el período de enero - abril del 2015, siendo esta la segunda Zona del país en registrar la mayor reducción de incidentes de este tipo. La primera zona de mayor reducción de homicidios es la zona 6 conformada por Azuay, Cañar y Morona Santiago con el 44% durante los cuatro primeros meses del 2015 (Ministerio de Turismo, 2015).

Según un artículo publicado en el diario El Comercio, la Policía Nacional reportó 1.456 denuncias de robos contra personas, es decir, un promedio de 20 delitos de este tipo diariamente. Sin embargo, esta cifra se ha reducido a comparación del primer trimestre del año 2015, donde se registraron 1.696 robos contra personas en Quito, de acuerdo al Ministerio de Interior (El Comercio, 2016).

Se ha registrado un significativo decrecimiento en los delitos en los últimos años lo que puede deberse a la creación de 31 unidades de Policía Comunitaria (UPC), y 1.889 operativos antidelincuenciales realizados en el año 2013 tanto en carreteras, control de armas y explosivos, control de bares, cantinas y centros de tolerancia (Agencia Pública de Noticias del Ecuador y Suramérica (ANDES), 2014).

En base a una encuesta realizada en el 2014 por Cedatos a 2.200 personas en 23 ciudades del Ecuador, el 29% de las mismas cree que uno de los problemas más graves del país es la delincuencia. Ciertamente, es un fenómeno que llama la atención ya que turistas también son vulnerables a hurtos y asaltos dentro del país.

Según la rendición de cuentas del Municipio del Distrito Metropolitano de Quito, durante el 2014 se ejecutaron 350 operativos de control de seguridad: 150 en eventos públicos, 150 en sitios generadores de inseguridad, y 50 en el Centro Histórico para el control de delincuencia y consumo de bebidas alcohólicas en espacios públicos; 900 operativos adicionales por parte de la Policía

Metropolitana en diversos lugares; migración del sistema “Ojos de águila” al sistema de video vigilancia del ECU911 y fortalecimiento de alarmas comunitarias; creación de cabinas exclusivas para atención de víctimas de violencia y acoso sexual en el sistema de transporte metropolitano. Todas estas acciones e iniciativas están destinadas a hacer un entorno más seguro y ofrecer condiciones para una vida normal y segura de sus habitantes (Municipio del Distrito Metropolitano de Quito, 2015).

El alarmante crecimiento de la delincuencia e inseguridad en las calles de Quito, ha sido el incentivo para la realización de este trabajo de titulación, incluyendo la influencia y constante uso de las redes sociales en el día a día de la ciudadanía. Como se sabe, el fenómeno de la inseguridad es un factor que aqueja a más personas con el pasar del tiempo.

Según estadísticas de los años 2011 al 2015, se encontró que los delitos en la ciudad de Quito han ido decreciendo paulatinamente. Sin embargo, en los primeros meses del año 2016, estas cifras no presentan decrecimiento y tienen tendencia a aumentar de forma alarmante debido al creciente uso de dispositivos móviles en la vida diaria

Es por eso que el desarrollo de una herramienta, que otorgue la posibilidad de reconocer de manera visual las zonas con más peligro en la ciudad, permitirá tomar mejores decisiones en cuanto a rutas adecuadas para realizar las actividades necesarias de la ciudadanía, disminuyendo el impacto de riesgo, esta herramienta también aportará información útil para ciudadanos, equipos de vigilancia y autoridades para, de esta manera, ser un punto de referencia y reforzar medidas de seguridad en las zonas más críticas.

De esta forma no solo se construye un mejor ambiente para todos sino que, al usar herramientas informáticas para el bien común, se reduce la brecha tecnológica en nuestra ciudad y lograr una mayor inclusión a la era digital.

Objetivo General

Diseñar y desarrollar un prototipo de aplicación web y móvil nativa Android para el monitoreo del Distrito Metropolitano de Quito, que permita mostrar las zonas de mayor peligro, para prevenir la exposición de la ciudadanía, por medio de la red social Twitter, recopilando datos enviados por usuarios afectados en las distintas zonas de la ciudad.

Objetivos específicos

- Analizar las metodologías y herramientas a utilizarse para el desarrollo de los prototipos.

- Desarrollar un prototipo de aplicación web que guarde y organice en una base de datos alojada en un servidor, la información recibida por la red social y aplicativo móvil la cual permita reconocer las zonas de mayor peligro mediante un mapa de calor.

- Desarrollar un prototipo de aplicación móvil nativa Android que muestre el mapa de calor, basado en la información obtenida de la red social Twitter, a partir de la posición del usuario usando geolocalización en base al sistema GPS integrado en cada dispositivo.

- Realizar pruebas a la aplicación para evidenciar su correcta funcionalidad las cuales reflejen la información obtenida de la red social a utilizar.

1. Capítulo I. Marco Teórico

1.1. Metodología de desarrollo

Una metodología de desarrollo se refiere a la forma de planificar y controlar el proceso de creación y desarrollo de un sistema. La metodología de ciclo de vida de desarrollo de software moderno se puede dividir en dos tipos, el proceso tradicional y el proceso ágil. El proceso para desarrollar software necesita una fuerte atención a todos los detalles requeridos para el sistema y una secuencia de pasos a seguir dependiendo cada metodología. Se debe recalcar que una determinada metodología no es necesariamente aplicada a todo desarrollo, la metodología a usar se debe ajustar a los requerimientos del desarrollo y de igual manera definir las herramientas a utilizar.

1.1.1. Metodología Tradicional

Las metodologías tradicionales se caracterizan por ser una serie secuencial de pasos como definición de requerimientos, planificación, construcción, pruebas y despliegue. Los requerimientos del cliente o del sistema son cuidadosamente documentados en su mayoría. Se visualiza de forma general la arquitectura del software y se inicia la codificación o programación. Se realizan distintos tipos de pruebas y se despliega finalmente en producción. La idea básica de esta metodología es ver de una forma detallada el proyecto terminado antes de iniciar su construcción y de esta manera trabajar visualizando el software ya finalizado. (Penadés, Canós & Letelier, 2003)

Entre las metodologías tradicionales se puede citar:

- **RUP (Rational Unified Process)**: Este proceso unificado de desarrollo fue creado en el año 1998 con el objetivo de producir software de alta calidad. Desde sus inicios se estableció a Unified Modeling Language (UML) como su lenguaje de modelado. Está dirigido por casos de uso, y

a partir de estos se sigue un flujo de trabajo: requisitos, análisis, diseño, implementación y prueba (Brito, K. 2009).

- **Microsoft Solution Framework (MSF):** Es una metodología desarrollada por Microsoft Consulting Services que provee principios, modelos y disciplinas para un correcto desarrollo de proyectos en cualquier plataforma (Microsoft, Linux, Unix). Se centra en modelos de proceso y de equipo dejando en segundo plano las elecciones tecnológicas (Brito, K. 2009).

- **Iconix:** Es un proceso que presenta las actividades de cada fase en forma clara y exhibe una secuencia de pasos a seguir. Ofrece también uso dinámico de UML como diagramas de casos de uso, diagramas de secuencia y de colaboración (Brito, K. 2009).

1.1.2. Metodología Ágil

Como su nombre lo indica, es un método ágil para el desarrollo del software, por lo tanto es mucho menos riguroso que un método tradicional. El punto principal de esta metodología es realizar un desarrollo incremental e iterativo, donde las fases del proceso de desarrollo son revisadas periódicamente. Los desarrolladores ágiles reconocen que el software no es un bloque de estructura, sino es una entidad que puede cambiar en el tiempo con partes complejas que interactúan entre sí. De esta manera, los desarrolladores, le dan más importancia a la capacidad de adaptación y pruebas de compatibilidad constantes (Optimus Information 2016).

Dentro de las metodologías ágiles se puede citar:

- Scrum.
- XP.
- Kanban.

1.1.2.1. Scrum

La metodología Scrum es una metodología ágil que se basa en la premisa de que el desarrollo de software es demasiado complejo e impredecible como para ser planeado en su totalidad desde un principio (Mahnic, V., Drnovscek, S. 2005).

Entre las características que identifican a esta metodología se encuentra la presencia de fases de desarrollo e iteraciones cortas (de 2 a 4 semanas) llamadas sprints las cuales producen como salida un avance sólido y potencialmente entregable del producto.

Se define como potencialmente entregable ya que al final de cada sprint se puede verificar si el código entregado es una parte totalmente funcional del sistema o se valora como una parte que está a punto de ser funcional, es decir que pocos detalles son requeridos para completarlo, de modo que pueda ser revisada por el cliente.

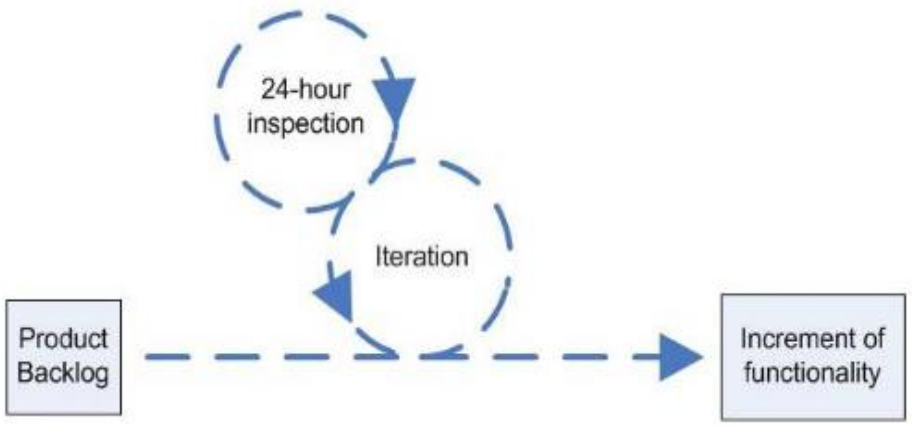


Figura 4. Flujo de trabajo metodologia Scrum.

Tomado de (Schwaber, s.f).

La Figura 4 muestra el esqueleto básico del funcionamiento de una iteración o sprint, el cual se basa, en un principio, en un conjunto definido de requerimientos llamado Product Backlog con las tareas agrupadas

anteriormente por prioridad. Se realizan reuniones diarias con el equipo para evaluar el progreso, los obstáculos presentados en el proceso y las posibles soluciones para que no se repitan en la siguiente iteración. Finalmente, el resultado esperado de cada iteración es un incremento en la funcionalidad del producto final.

Dentro de Scrum existen diferentes roles definidos para cada miembro del equipo:

- **Product Owner:** es el encargado de representar los intereses de todos aquellos involucrados en el proyecto y en el sistema resultante. Prioriza los requerimientos del sistema.

- **Equipo (Team):** Sus funciones incluyen el desarrollo del proyecto, es decir, convertir el Product Backlog en un producto funcional incremental después de cada iteración. Los miembros del equipo se organizan y administran por sí solos, además tienen la responsabilidad colectiva si el proyecto y el sistema es un fracaso o un éxito.

- **Scrum Master:** Ocupa la posición de Project Manager con algunas diferencias, es responsable de administrar el flujo del proceso Scrum y asegurarse que se cumplan sus buenas prácticas.

1.1.2.2. XP (Extreme Programming)

Una metodología que se centra en gran medida en garantizar la calidad del software entregado y que prescribe soluciones de ingeniería para lograr este fin. Un equipo XP, está compuesto por todos los que contribuyen al proyecto, participan en la planificación de lanzamiento y la planificación de iteración. Trabajan en ciclos de desarrollo muy cortos para que los cambios solicitados por cliente, que también es parte del equipo, se puedan integrar con frecuencia. XP trabaja para obtener un producto de mejora continua y de alta calidad que puede responder a los cambios o necesidades del cliente por medio del uso de varias prácticas, entre las más importantes se pueden mencionar:

- Desarrollo Dirigido por Pruebas (Test Driven Development)
- Pruebas del Cliente (Customer Testing)
- Integración Continua (Continuous Integration)
- Pequeños Entregables (Small Releases)
- Programación por Parejas (Pair Programing)

XP se basa en los siguientes valores:

- Simplicidad, se trata de simplificar todo lo que se pueda y de esta forma agilizar el desarrollo y facilitar el mantenimiento.
- Comunicación, se puede entender de mejor manera el código cuando más simple sea. Comentar dentro del código es buena práctica de comunicación y programación.
- Retroalimentación o reutilización del código, el cliente es parte fundamental del equipo y su opinión se da a conocer en tiempo real, esto ayuda a los desarrolladores a centrarse en las partes más importantes del código y también minimiza el tener que rehacer partes del sistema que no cumplan con los requisitos. Realizar pruebas unitarias frecuentes permite detectar fallos donde se hicieron cambios recientes en código.
- Necesidades actuales, se debe diseñar para el presente requerimiento del cliente y no pensar en diseños futuros. De esta forma se obtienen requerimientos de tiempos menores y simplifica el código, se debe tener en cuenta que se tendrá que reconstruir el código de ser necesario.

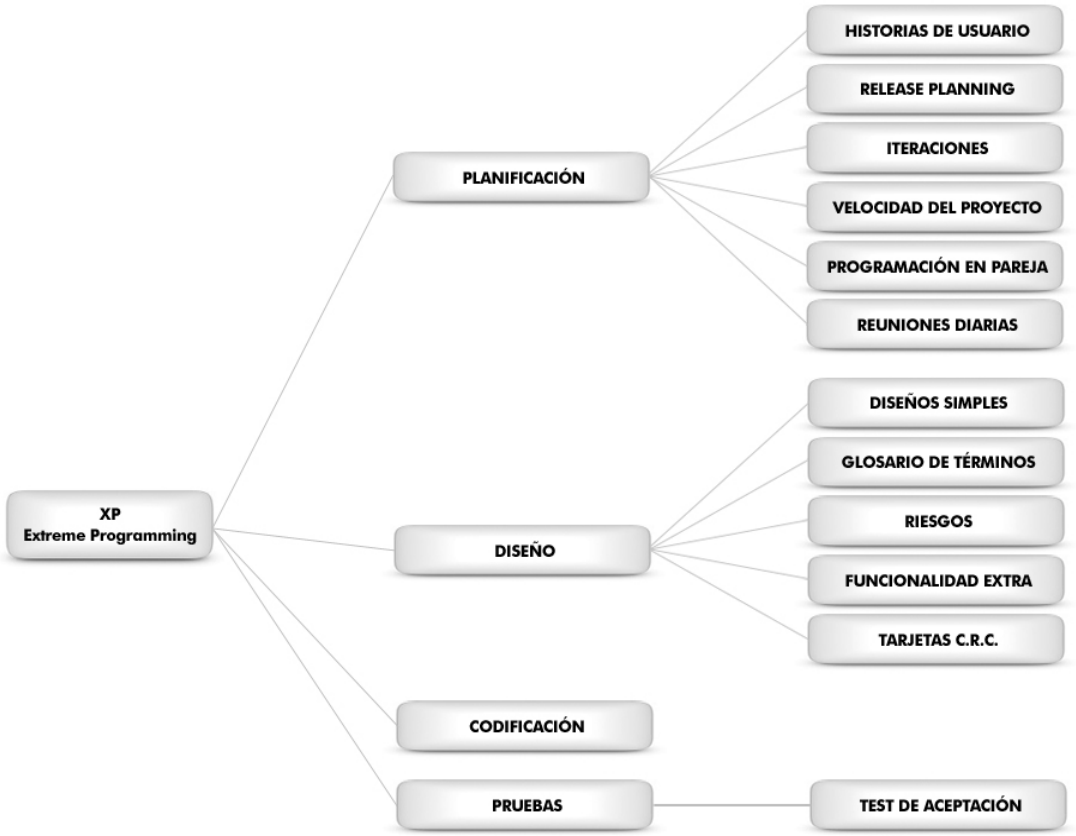


Figura 5. Fases de trabajo metodología XP.

Tomado de (Isla Visual, s.f).

Como se evidencia en la Figura 5, la metodología XP se compone de cuatro fases que son planificación, diseño, codificación y pruebas. En la fase de planificación se establece las historias de usuario, las iteraciones que se van a realizar, definir la programación en parejas, horarios de reuniones diarias, etc. Para la fase de diseño se trata de implementar diseños simples, glosario de términos, posibles riesgos, funcionalidad extra, tarjetas Clase-Responsabilidad-Colaboración (C.R.C, una herramienta para generar lluvia de ideas). La siguiente fase es la codificación para el desarrollo del proyecto. En la última fase se realizan las pruebas del proyecto.

1.1.2.3. Kanban

Es desarrollado como un subcomponente del Sistema de Producción Toyota y tiene su origen en procesos de manufactura justo a tiempo (JIT - just in time) en los que se utilizan tarjetas para describir las necesidades del material.

En Kanban el flujo del trabajo se visualiza dividiéndolo en pequeños y discretos objetos los cuales se van a escribir en una tarjeta que se le adjuntará en un pizarrón. El pizarrón se puede dividir en secciones como por ejemplo, la primera columna se puede llamar tareas por realizar, la siguiente columna tareas en progreso y seguido de la columna tareas para revisión, a continuación pruebas, tareas cumplidas, etc.

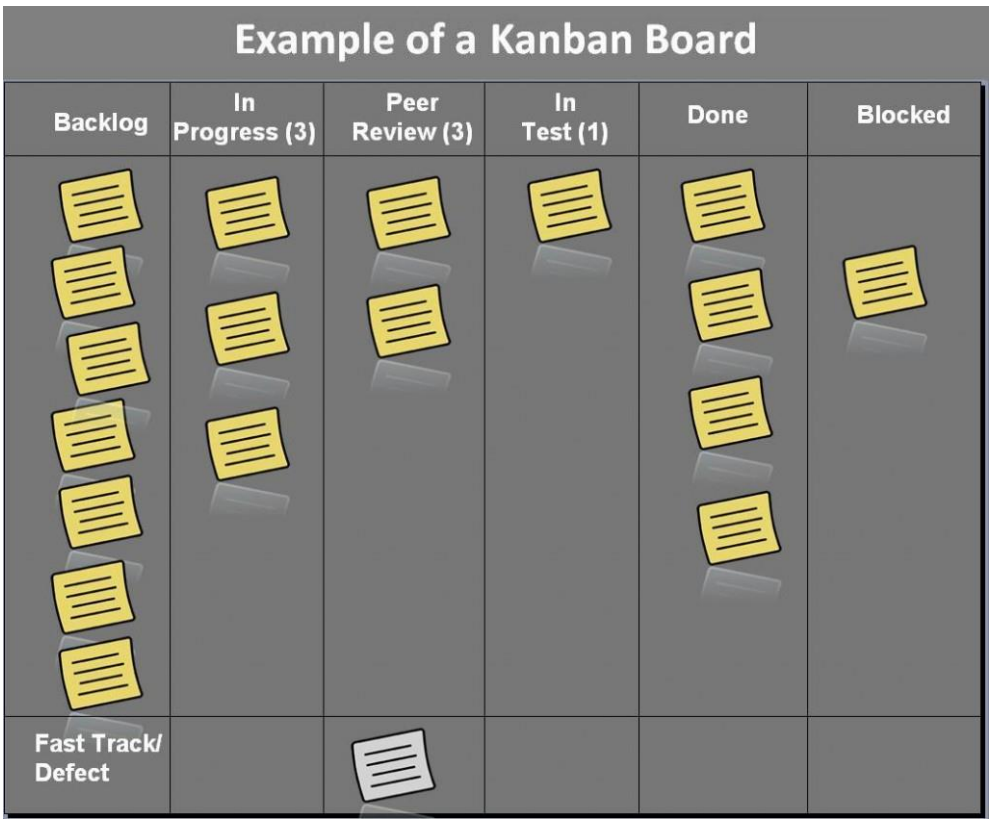


Figura 6. Ejemplo de pizarra de tareas Kanban.

Tomado de (Comunidad IEBS, s.f).

Como se muestra en la Figura 6, el pizarrón contiene varias columnas divididas en estados para establecer el progreso de cada una de las tareas. En este ejemplo se observan seis columnas: Backlog, que son las tareas a realizar; In Progress, que son las tareas que se están realizando en ese momento; Peer Review, para hacer la revisión de las tareas entre los integrantes del equipo; In Test, se colocan las tareas a las que se están realizando pruebas; Done, muestra las tareas finalizadas; y Blocked muestra las tareas bloqueadas que no pueden avanzar por alguna razón.

En Kanban el número de elementos o tareas que pueden estar en curso en ese momento es estrictamente limitado.

El tiempo promedio en que se tarda en completar una tarea o elemento, a veces llamado el tiempo de ciclo, se realiza un seguimiento y es optimizado de manera que el proceso se vuelve más eficiente y predecible como sea posible.

La metodología Kanban se basa en los siguientes principios:

- Calidad garantizada, todas las tareas realizadas deben salir bien y a la primera, no hay un margen de error. De este principio se deriva que en Kanban no se permite la rapidez, sino la calidad de las tareas cuando estas finalizan. Este punto se basa en el hecho de que en varias ocasiones cuesta más arreglar ciertos errores que realizar bien al primer intento.
- Reducción del desperdicio, se enfoca en hacer lo justo y necesario, pero hacerlo bien. Aplica el principio YAGNI, el cual supone la reducción de todo lo que es innecesario o superficial.
- Mejora continua, aparte de ser un método de gestión, también es un sistema de mejora en el desarrollo de proyectos, según los objetivos a alcanzar.
- Flexibilidad, las tareas a realizar se definen según las tareas pendientes por realizar (backlog), de esta manera se puede priorizar aquellas tareas según las necesidades que se puedan dar en un momento determinado y tener la capacidad de dar respuesta a tareas imprevistas.

1.1.3. Selección de metodología

Las principales diferencias entre estos dos grupos de metodologías se pueden resumir en la Tabla 1:

Tabla 1. *Cuadro comparativo entre metodología tradicional y ágil.*

Metodologías tradicionales	Metodologías ágiles
Énfasis en planificación, control del proyecto y especificación de requerimientos.	Énfasis en buenas prácticas de producción de código y adaptación a cambios
Necesita requerimientos detallados por parte del cliente. Se permiten implementar mejoras pequeñas una vez finalizado el producto.	Mayor flexibilidad de requerimientos y mejoras a medida que avanza el desarrollo. Más participación y satisfacción del cliente en cada fase.
El encargado de la comunicación con el cliente es, generalmente, una persona no involucrada en el desarrollo técnico como tal.	Alta interacción con el cliente por parte de los desarrolladores, generando gran flexibilidad funcional y de arquitectura.
Documentación extensa y detallada.	Documentación escasa, código auto explicativo.
Soporta sistemas de gran tamaño.	Ideal para sistemas medianos o pequeños debido a la compatibilidad al cambio de requerimientos.
Proceso mucho más controlado, con numerosas políticas y normas.	Proceso menos controlado, con pocos principios.

Para el presente proyecto se utilizará una metodología ágil debido al alto grado de adaptabilidad durante el avance y a su amplio uso en la actualidad, lo cual se acopla a la naturaleza de este proyecto. Además el uso de una metodología

tradicional se aplicaría a proyectos con requerimientos estrictamente definidos antes de iniciar el desarrollo.

1.1.3.1. Scrum vs. XP

En Scrum, los equipos y reuniones están establecidas y son muy difíciles de cambiar, además la manera de cómo trabajar se deja a decisión de los equipos. Por otro lado XP tiene un conjunto de prácticas que se podría ver abrumador frente a un desarrollador ágil que no posea experiencia en este campo.

Se puede decir que la metodología Scrum está más orientada a la productividad, mientras que XP está orientada a la ingeniería.

El valor de las prácticas de XP son indiscutibles y muchas organizaciones que usan Scrum adoptan principios de XP como la Programación en Parejas, Desarrollo Dirigido por Pruebas (Test Driven Development) y Refactorización como prácticas que mejoren la calidad, acelerar el proceso de liberación y reducir la necesitar de revisar el trabajo debido a la deuda técnica.

Los equipos XP trabajan sobre tareas bajo un orden de estricta prioridad, un equipo Scrum no podría hacer frente a las tareas bajo un orden de prioridad una vez iniciado el sprint.
Los equipos XP pueden agregar nuevas tareas de trabajo en un sprint y dejar de lado a tareas que tengan un tamaño equivalente siempre que estas no se hayan iniciado y si el cliente decide una nueva prioridad.

En términos similares, el rol de Customer (cliente) es muy similar al rol de Product Owner en Scrum; en ambos casos estos roles ayudan a definir las historias de usuario, a priorizar las mismas y están disponibles para los desarrolladores. En XP y Scrum es necesario preparar reuniones diarias.

1.1.3.2. Scrum vs Kanban

Scrum es más restrictivo que Kanban, lo que evita la definición de roles y equipos y no tiene una estructura formal para la preparación de reuniones. Kanban no define iteraciones (sprints), aunque se puede definir si se lo desea.

Las técnicas de visualización que usa Kanban son perfectas para equipos que se encuentran en una misma localidad y que trabajan con tareas que pueden cambiar constantemente.

El pizarrón Kanban es a menudo usado por los equipos Scrum como un tablero que describe las tareas a realizar y se utiliza para seguir el proceso de desarrollo a lo largo del sprint.

La regla del límite del trabajo en progreso en Kanban también lo hace adecuado para equipos que tienen recursos limitados o donde se requiere el aporte de cada miembro del equipo para realizar las tareas.

Scrum limita la cantidad de trabajo que se va a realizar en cada sprint, la carga o complejidad de trabajo está estimada por el tamaño o cantidad de trabajo que posea cada historia de usuario, valorada en puntos, y que está aprobada por el equipo en cada reunión de planificación.

El equipo Kanban monitorea el ciclo de vida para de esta forma optimizar el tiempo de ejecución para hacerlos tan cortos y predecibles como sea posible, un equipo Scrum tiene como objetivo mejorar su producción durante el sprint y la velocidad del equipo valorado según la cantidad de puntos completados por sprint. Esto sin duda hace a Scrum más conveniente para escalar el desarrollo, además lo hace más familiar y predecible que puede resultar tranquilizador para las grandes empresas.

Tabla 2*. Comparación de metodologías ágiles*.

Metodología			
Criterio	**Scrum**	**XP**	**Kanban**
Tipo de iteraciones	Iteraciones a plazo fijo	Iteraciones de plazo variable	Iteración de plazo fijo y variable
Roles - Facilitador	Scrum Master	Coach, Big Boss	N/A
Roles - Administrador de Requerimientos	Product Owner	Cliente	N/A
Roles - Equipo de desarrollo	Equipo (Team)	Programador	N/A
Equipo	Multifuncional, auto organizado	Especializado	Multifuncional y especializado
Limitación de trabajo en progreso	Limitación por iteración	Limitación por iteración	Limitación por estado.
Incorporación de tareas	No es posible, solo al terminar del sprint	No es posible, solo al terminar la iteración	Es posible, de estar en capacidad.
Estimación	Obligatoria	Obligatoria	Opcional

Tomado de (Academia.edu, s.f.).

Entre las metodologías ágiles se escogió a Scrum por su adaptabilidad al crecimiento y escalamiento en el desarrollo, además tiene la capacidad de tomar varias características de diferentes metodologías y adaptarlas a su conveniencia para un desarrollo más efectivo, se centra en la producción del software y mejorarlo continuamente durante cada iteración de desarrollo realizada.

1.2. Mapas de Calor

Un mapa de calor es una representación gráfica de datos estadísticos o colecciones de coordenadas cartesianas que permite resaltar zonas en base a un criterio específico, es este caso, zonas con mayor número de delitos. Los mapas de calor se basan en la termografía con el uso de un código de colores, se identifica con una gama de colores cálidos (rojo, amarillo, naranja) a las zonas más afectadas y colores fríos (azul, verde, turquesa) a las zonas menos afectadas.

Existen diferentes tipos de mapas de calor de acuerdo a su uso:

- Mapas de calor web: Es utilizado como herramienta analítica en sitios web donde se usan para mostrar áreas más concurridas o con mayor número de clicks dentro de una página.

Figura 7. Mapa de calor como herramienta de análisis web.

Tomado de (WebMaxFormance, s.f).

- Mapas de calor en biología: Son usados en biología molecular para representar el nivel de concentración de ciertos factores, por ejemplo: células en diferentes estados, muestras de diferentes pacientes.

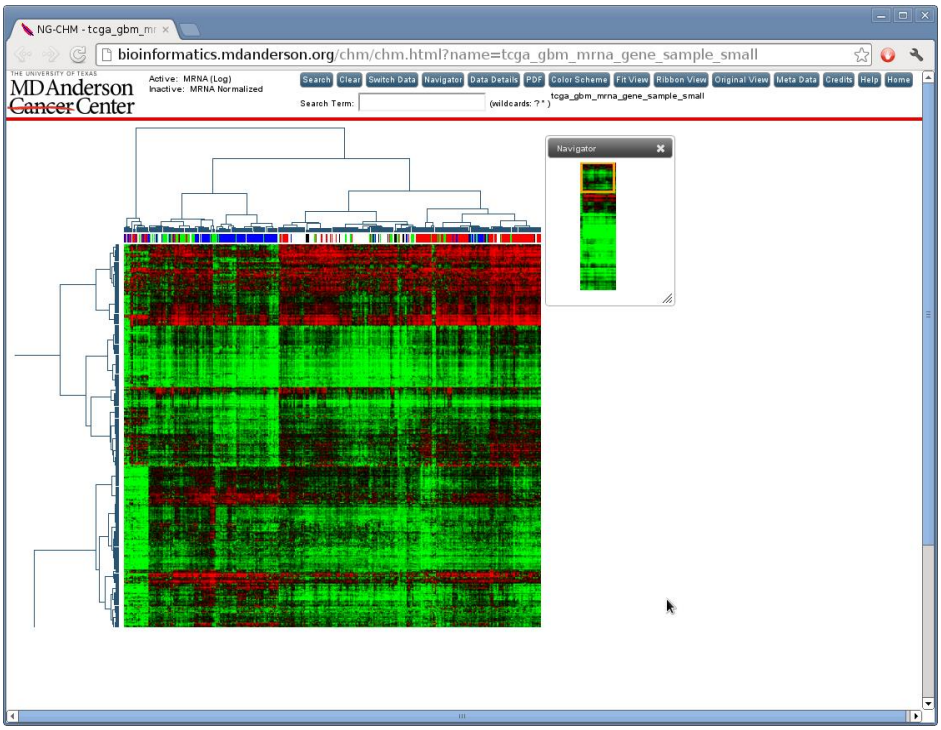


Figura 8. Mapa de calor usado en biología con microarrays de ADN.

Tomado de (Department of Bioinformatics and Computational Biology, s.f).

- Mapas de calor geográficos: Se usan para representar datos en puntos geográficos, usualmente se visualiza un mapa de calor como una capa adicional encima de un mapa geográfico bidimensional de cualquier área.

Figura 9. Mapa de calor geográfico generado con Google Maps.

Tomado de (Google Developers, s.f).

En definitiva, los mapas de calor son una herramienta gráfica muy útil y aplicable en varios casos y ramas de estudio. Los mapas de calor geográficos pueden ser usados en estudios de cualquier índole poblacional para expresar la presencia de un fenómeno, por ejemplo, tendencias políticas, proliferación de una enfermedad, entre otros. Son bastante accesibles dependiendo del uso que se le quiera dar, en este caso, en informática, existen varias áreas de aplicación como analítica web, en donde existen varias plataformas o plugins listos para usarse en la administración de páginas web. En el caso específico de desarrollo de software, existen plataformas que permiten manipular los datos representados en un mapa de calor así como otros parámetros visuales tales como la gama de colores a usarse, radio de despliegue, gradiente y opacidad, etc., los cuales dan la posibilidad de personalizar la apariencia para cada necesidad.

Para el desarrollo del presente proyecto se utilizó un mapa calor debido a su gran adaptabilidad visual la cual se requiere para representar las zonas de mayor peligro en la ciudad. A diferencia de otras herramientas visuales, tales como tablas o gráficos estadísticos (barras o sectores), un mapa de calor puede atraer más la atención del usuario por sus características interactivas, además de ser una herramienta informativa y amigable.

La personalización de un mapa de calor hace posible un desarrollo más detallado para la presentación al usuario final.

1.3. Herramientas de desarrollo

En esta sección se detallarán las herramientas técnicas específicas a utilizar en el desarrollo del presente proyecto, analizando sus ventajas frente a otras posibles herramientas disponibles en la actualidad. Se ha tomado en cuenta el grado de modernidad, la experiencia en el uso, y la presencia de soporte continuo como factores que incidieron en la elección de todas las herramientas en general.

1.3.1. Arquitectura - REST

Representational State Transfer, es un estilo de arquitectura para sistemas distribuidos de software. Se basa en un protocolo sin estado, cliente-servidor, usualmente las llamadas HTTP son las más utilizadas. La World Wide Web (WWW) puede ser vista como una arquitectura basada en REST.

REST se enfoca principalmente en recursos, cualquier porción de información puede ser considerada un recurso, por ejemplo, un archivo, un servicio temporal o un objeto tangible (ej. usuarios).

Las propiedades de esta arquitectura REST son:

- Desempeño, las interacciones entre los componentes pueden ser decisivos en cuanto al desempeño percibido por el usuario y la eficiencia de la red.
- Escalabilidad, soporta gran cantidad de componentes y sus interacciones.
- Simplicidad de interfaces.
- Modificabilidad de los componentes para adaptarse a los cambios.
- Portabilidad de componentes.

- Confiable, resistencia a fallos en el sistema sean de componentes, conectores o datos.

Una de las principales características de un servicio web o aplicación RESTful es el uso explícito de métodos HTTP para realizar las operaciones CRUD (Create/Read/Update/Delete) sobre datos, dentro de los métodos HTTP se encuentran: POST, para crear recursos en el servidor; GET, para devolver el valor de un recurso; PUT, para cambiar el estado o actualizar un recurso; DELETE, borrar un recurso. Cada recurso tiene un estado interno que no es accesible directamente, se utiliza representaciones de este recurso para poder acceder a su estado. Una representación es un formato de datos concreto usado para la transferencia de una copia del estado público de un recurso entre el cliente y el servidor, por ejemplo, XML, JSON, etc. Los recursos y representaciones REST son accedidos usando un URI (Identificador Único de Recurso) (Roy Fielding, 2000, cap. 5).

REST es usualmente comparada con Simple Object Access Protocol (SOAP) en el ámbito de Web Services. SOAP define un protocolo estándar de comunicación y especificación para intercambio de mensajes basados en XML, usa diferentes protocolos de transporte como HTTP o SMTP, en el ámbito de web services, SOAP se enfoca en exponer una parte de la lógica como un servicio, es decir, operaciones. En cambio, REST describe un conjunto de principios de arquitectura con los cuales se transmite información sobre una interfaz estandarizada como es HTTP, un web service REST se enfoca en exponer recursos (Dhingra S, 2013).

Las necesidades de cada proyecto pueden determinar cuál de estas dos tecnologías puede ser la más idónea. Para este proyecto, REST ha sido la elegida para el desarrollo de un web service tomando en cuenta las siguientes ventajas:

- REST permite el uso de varios formatos, por ejemplo JSON, mientras que SOAP solo permite XML. El uso de JSON también hace posible un mejor soporte para navegadores web.

- La implementación de REST es notablemente más simple que con SOAP.
- Un servicio REST se puede integrar fácilmente con servicios ya existentes desarrollados con otras tecnologías.

1.3.2. Framework

Un framework es un entorno de trabajo para el desarrollo de software, contiene un conjunto de librerías o clases mediante las cuales se pueden crear aplicaciones o sitios web de manera rápida, ordenada y segura.

1.3.2.1. Django

Es un framework web, gratuito y open-source, de alto nivel basado en Python que promueve un desarrollo y diseño rápido y limpio. Se encarga de todas las molestias del desarrollo web típico, como la conexión a una base de datos, sesiones y autenticación de usuarios, control de vulnerabilidades (SQL injection, cross-site scripting, clickjacking), administración de contenido etc., usando paquetes y librerías listas para ser usadas sin necesidad de grandes configuraciones.

Utiliza la arquitectura Modelo, Vista, Controlador (MVC), lo cual divide el desarrollo en tres grandes partes:

- Modelo: es la capa donde se encuentran los datos de aplicación y su reglamentación.
- Vista: es la capa en la cual se presentan los datos.
- Controlador: esta capa se encarga de controlar el flujo de ejecución del sistema y las peticiones de los usuarios.

Django fue diseñado para ayudar al desarrollo de aplicaciones escalables y flexibles lo más rápido posible, además ayuda a evitar los errores de seguridad

más comunes. Es un framework bastante versátil que permite crear desde sistemas manejadores de contenido hasta plataformas científicas de computación.

1.3.2.2. Flask

Flask es un micro framework de python que permite crear aplicaciones web de manera rápida y con un número mínimo de líneas de código, incluye un servidor web que permitirá probar la aplicación web que se esté construyendo. Se enfoca en que las aplicaciones o sitios web puedan funcionar el más rápido posible con el mínimo uso de recursos.

1.3.2.3. Comparación: Django vs. Flask

Flask junto con Django son los frameworks basados en Python más usados y conocidos. Flask fue creado cinco años después que Django, lo que desemboca en que Django tiene mayor extensión y además una comunidad de soporte más grande.
Django posee una interfaz de administración que tiene funcionalidades básicas listas para usarse, (out of the box). Con Flask, una tarea simple puede implicar un desarrollo desde cero (from scratch).

Tanto Flask como Django, pueden ahorrar tiempo en la construcción de un producto y manejar el crecimiento de usuarios con facilidad. Grandes empresas como Evenbrite, Prezi, Instagram y Bitbucket usan Django. Mientras Flask es usado por WakaTime and Twilio.
Aunque Flask es más sencillo que Django para empezar un desarrollo, no ofrece el suficiente apoyo durante todo el proceso de construcción de un producto, en cambio Django proporciona guías para todos los pasos.
Flask posee un mayor acople al momento de construir APIs mientras Django se enfoca más en el uso de REST.

Al contrario de Flask, Django hace muy fácil la construcción de páginas y sitios web HTML.
En el caso de usar una base de datos SQL, Django facilita las consultas complejas con su avanzado y potente ORM (Object-relational Mapping), Flask ofrece una librería adicional para consultas pero no de tan fácil uso como en Django.
A diferencia de Django, Flask es más flexible a la hora de admitir varios estilos en la programación como convenciones.

Se ha escogido el framework Django con el lenguaje de programación Python como la principal herramienta para el desarrollo de la página web de este proyecto, ya que por tratarse de un proyecto que integra la conexión con una red social y el uso de material gráfico avanzado, como un mapa de calor, se considera de gran utilidad el uso de este framework, tomando en cuenta las ventajas antes mencionadas, y entre otros factores, por su facilidad de acceso a paquetes de funcionalidad avanzada listos para ser usados y por su simplicidad de código.

1.3.3. Lenguaje de Programación

1.3.3.1. Python

Es un lenguaje de programación que posee una licencia de código abierto creado a finales de los años 80 por Guido van Rossum, soporta varios paradigmas tales como orientación a objetos, programación imperativa y funcional. Se enfoca en una sintaxis legible y transparente para el humano y en la facilidad de extensión o acople con otros lenguajes, por ejemplo se pueden escribir módulos en C o C++.
El lenguaje de programación Python, en el cual se basa Django framework, ofrece varias ventajas frente a otros lenguajes, como por ejemplo:

- Bloques de código organizados por indentación, es decir, no usa llaves como delimitadores de funciones lo que resulta en código más limpio y entendible para la persona.
- Gran cantidad de extras y librerías listas para su uso que incrementan la funcionalidad en tareas específicas, por ejemplo, manejo avanzado de fechas.
- Código de fácil entendimiento para principiantes, aun así poderoso para profesionales.
- Las variables no necesitan especificar su tipo antes de ser usadas, este aspecto también está presente en lenguajes populares como Javascript y PHP.
- Posee tipos de datos poderosos como listas polimórficas y diccionarios.
- Cantidad reducida y concisa de código a comparación con sus equivalentes en otros lenguajes.

1.3.3.2. Java

Java es un lenguaje de programación orientado a objetos y una plataforma informática que se comercializó por primera vez en 1995 por Sun Microsystems y que posteriormente fue adquirida por Oracle. Su principal objetivo es que se puedan desarrollar programas para que se puedan ejecutar en cualquier contexto, en cualquier ambiente lo que hace que tenga una característica importante de ser multiplataforma, es decir que se pueda ejecutar en Windows, Linux, Mac entre otros. Una característica importante de java es escribir el código una sola vez y ejecutarlo en varios dispositivos o plataformas.

1.3.4. IDE

1.3.4.1. PyCharm

Es un Entorno de Desarrollo Integrado (IDE), usado para programar en Python. Proporciona un analizador de código, un depurador gráfico, pruebas unitarias integradas, integración con control de versiones y soporta el desarrollo web con

Django. Desarrollado por la empresa JetBrains. Es multiplataforma, funciona en Windows, Mac OS y Linux. Tiene una edición profesional (Professional Edition) la cual es pagada, y una edición dedicada a la comunidad (Community Edition) que es posee menos funcionalidades que la profesional y es gratuita.

Para el presente proyecto es ideal usar este IDE ya que está hecho para programar en python y soporta el desarrollo de aplicaciones web en Django, el framework web para el desarrollo del presente proyecto, permite configurar los entornos virtuales necesarios para correr directamente el entorno virtual en el IDE. También soporta el control de versionamiento por lo que se puede integrar con git, en este proyecto se usará git con el uso de repositorios Bitbucket.

1.3.4.2. Android Studio

Es un Entorno de Desarrollo Integrado (IDE), basado en IntelliJ, el cual provee varias mejoras respecto al plugin *ADT* (Android Developer Tools) utilizado en Eclipse. Usa una licencia de software libre *Apache 2.0,* es multiplataforma y está programado en Java. Es el IDE oficial para el desarrollo de aplicaciones Android, su objetivo es crear un entorno dedicado exclusivamente al desarrollo de aplicaciones para dispositivos Android, el cual es proporcionado por Google para obtener un mayor control sobre el proceso de producción.

Principales características:

- Soporte, tiene un mayor soporte para dispositivos Android, como por ejemplo Android Wear el cual es un sistema operativo para dispositivos corporales como un reloj.
- Herramientas Lint, (permite detectar el código que no es compatible entre diferentes arquitecturas o identificar código confuso que el compilador no es capaz de controlar) para identificar problemas de rendimiento, usabilidad y compatibilidad de versiones.

- Integración con Gradle, una herramienta encargada de gestionar y automatizar la construcción de proyectos, como por ejemplo las tareas de compilación, testing y empaquetado.
- Nueva interfaz específica para el desarrollo de aplicaciones Android.
- Diseño del editor de código que puede soportar la edición de temas.
- Utiliza ProGuard que ayuda a optimizar y reducir el código del proyecto al momento de exportar a APK, lo cual resulta muy útil para dispositivos de gama baja con limitaciones de memoria interna.
- Permite realizar una vista previa en varios dispositivos y resoluciones.
- Permite la importación de proyectos realizados en Eclipse.
- Posee control de versionamiento accediendo a repositorios alojados en la nube para ser utilizados como Mercurial, Git, GitHub o Subversion.
- Alertas en tiempo real de errores sintácticos, rendimiento o compatibilidad antes de realizar la compilación de la aplicación.
- Integración con Google Cloud Platform, que permite el acceso a los diferentes servicios que proporciona google en la nube.

El motivo de elegir esta herramienta, a diferencia de otras, es que permite el desarrollo de una aplicación cien por ciento nativa para dispositivos Android, además es la herramienta oficial que provee Google para el desarrollo de aplicaciones móviles, lista para iniciar un proyecto sin necesidad de usar plugins externos. Al ser una de las herramientas más usadas y oficiales se puede encontrar un soporte mantenido en el tiempo y documentación oficial para desarrollar aplicaciones.

1.3.5. Base de Datos

1.3.5.1. PostgreSQL

Es un Sistema de Gestión de Base de Datos (SGBD) objeto-relacional, distribuida bajo licencia BSD y es de código abierto. Se puede decir que es el sistema gestor de base de datos de código abierto más potente del mercado, sus actualizaciones pueden compararse a una base de datos de alto nivel.

Utiliza un modelo cliente/servidor y en lugar de usar multihilos usa multiprocesos lo cual garantiza la estabilidad del sistema. Si se llega a dar un fallo en alguno de los procesos no afectará el resto del sistema y continuará con normalidad.

La última serie de versión en producción es la 9.5, sus características técnicas la hacen una de las bases de datos más robustas dentro del mercado. Durante su desarrollo se consideran características como estabilidad, potencia, robustez, facilidad de implementación y administración. Funciona de manera óptima con grandes cantidades de datos y alta concurrencia de usuarios que acceden al sistema.

Tiene características sofisticadas como Control de Concurrencia mediante Versiones Múltiples (Multi-Version Concurrency Control, MVCC), punto de recuperación en el tiempo, tablespace, replicación asíncrona, backups en caliente, un sofisticado planificador y optimizador de consultas. Soporta caracteres internacionales, codificación de varios bytes (multibyte), Unicode, sensible a mayúsculas y minúsculas. Es altamente escalable, tanto en la cantidad de datos que puede almacenar como en la cantidad de concurrencia de usuarios que puede soportar. Hay sistemas activos que usan PostgreSQL que manejan más de cuatro terabytes de datos. Algunos de los límites generales se incluyen en la siguiente tabla:

Tabla 3. *Límites generales de PostgreSQL*.

Límite	**Valor**
Tamaño máximo de base de datos	Ilimitado
Máximo tamaño de tabla	32 TB
Máximo tamaño de fila	1.6 TB
Máximo tamaño de campo	1 GB

Máximo número de filas por tabla	Ilimitado
Máximo número de columnas por tabla	250 - 1600 dependiendo del tipo de columna.
Máximos índices por tabla	Ilimitado

Tomado de (PostgreSQL, s.f).

1.3.5.2. MySQL

MySql es la base de datos de código abierto más popular del mundo, permite una entrega rentable de confiabilidad, alto rendimiento y escalabilidad en aplicaciones de bases de datos basadas en Web.

Es un sistema de gestión de base de datos relacional (RDBMS) basado en el lenguaje de consulta estructurado. Se ejecuta en casi todas las plataformas incluyendo Unix, Linux y Windows. Puede ser utilizada en una amplia gama de aplicaciones, sin embargo MySQL se asocia más con aplicaciones web.

Originalmente fue creada por la compañía sueca MySQL AB y en 2008 fue adquirida por Oracle. MySQL fue escrita en C y C++, se destaca por su gran adaptación a diferentes entornos de desarrollo y la interacción con varios lenguajes de programación como PHP, Java, Perl, Python entre otros.

1.3.5.3. Comparación: PostgreSQL vs. MySQL

Una de las características similares entre estos SGBD es que ambos utilizan el Lenguaje Estructurado de Consultas (SQL) y son de código abierto. Sin embargo, existen algunas diferencias que intervienen al momento de escoger uno, entre ellas se puede mencionar:

- PostgreSQL adhiere su código a los estándares SQL de ISO, lo que hace posible migrar o reusar este código en otros SGBD de manera sencilla, mientras que MySQL no sigue estos estándares.
- PostgreSQL es conocido por su escalabilidad y cumplimiento en principios de Atomicidad, Consistencia, Aislamiento, y Durabilidad (ACID) en transacciones.
- PostgreSQL está enfocado en la fiabilidad e integridad de sus datos, es bastante eficiente a la hora de realizar consultas que involucran gran número de tablas. MySQL se enfoca en la optimización de consultas sencillas.
- A diferencia de PostgreSQL, MySQL carece de soporte para rollbacks y no maneja integridad referencial.

Ciertamente, ambos pueden ser ideales de acuerdo a la función que se les pueda otorgar. Se debe tener en cuenta las necesidades de cada proyecto y equipo las cuales influirán al momento de decidir cuál de estos dos SGBD usar.

En el presente proyecto se ha decidido usar PostgreSQL por sus cualidades en manejo de gran volumen de información y escalabilidad, ya que se proyecta un rápido crecimiento de información en la base de datos. Además PostgreSQL puede ser visto como el equivalente a un gestor de bases de datos de alto nivel de código abierto (Tim Perdue, 2000).

1.3.6. Google Maps API

Google Maps es una aplicación tecnológica que brinda servicios relacionados con mapas web desarrollada por Google, ofrece varios servicios basados en localización como mapas de ciudades, calculadora de rutas en viajes a pie, auto o transporte público. Permite observar imágenes satelitales aunque no a tiempo real e imágenes a pie de calle con Google Street View. Google Earth es una aplicación de escritorio que ofrece servicios más completos que el sitio web de Google Maps, por ejemplo imágenes satelitales de todo el mundo incluyendo los polos norte y sur.

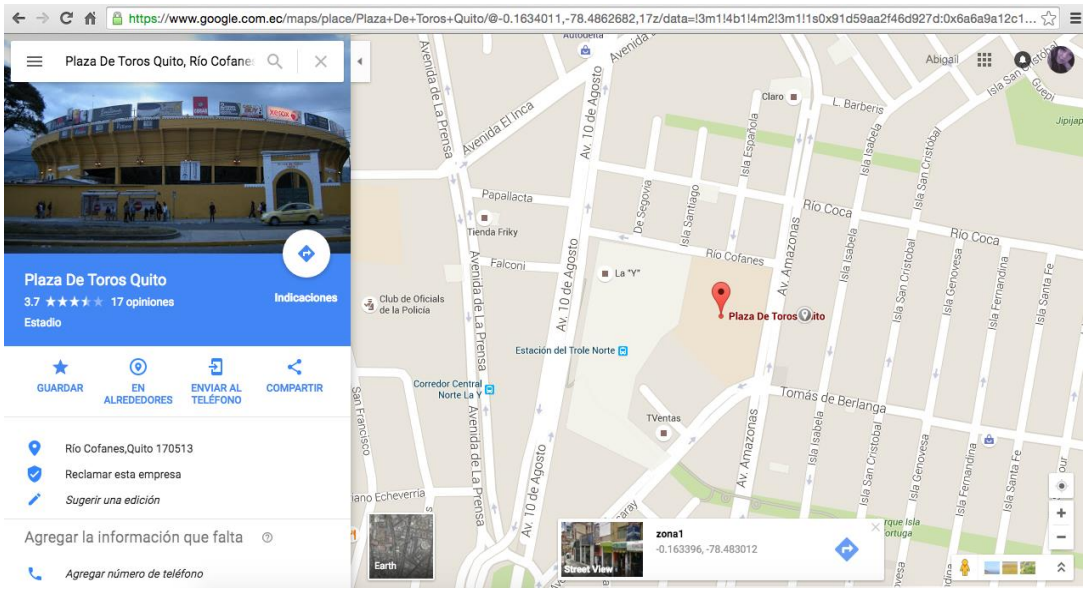


Figura 10. Sitio web de Google Maps con ubicación en la Plaza de Toros Quito con vista a nivel de calles.

Tomado de (Google Maps, s.f).

En la Figura 10 se puede apreciar las características básicas de Google Maps entre las cuales está la posibilidad de acercar o alejar un mapa con el mouse o mediante las teclas “+” y “-”, encontrar una dirección específica mediante la barra de búsqueda y encontrar información básica de establecimientos turísticos o restaurantes, calcular la mejor ruta entre dos ubicaciones, además es posible obtener las coordenadas geográficas de cualquier punto en el mapa mediante la barra de dirección en el navegador o con click derecho en el punto específico.

Google Maps API es una interfaz de programación formada por un grupo de librerías, funciones y procedimientos que permiten incrustar una instancia de un mapa con la posibilidad de manipular y personalizar parámetros provistos por Google Maps. Para poder acceder a estos procedimientos y librerías se debe registrar el uso de mapas en la página oficial de Google Developers, especificando el uso que se le dará y la dirección del sitio en caso de tratarse de una página web a continuación se obtendrá una llave de acceso para hacer uso de todas las funciones que ofrece esta API.

Google Maps API está disponible para plataformas web, Android, iOS y web services. Este proyecto hará uso de las plataformas web y Android.

Dentro de Google Maps APIs para plataformas web se puede encontrar varias opciones de acuerdo a cada necesidad, entre ellas se encuentran:

- Google Maps JavaScript API: Permite personalizar gran parte del contenido mediante las características disponibles como: tipo de mapa (mapa/satélite), colores, marcadores, nivel de zoom, etc., además hace posible la adición de capas a un mapa básico.

- Google Maps Embed API: Proporciona mapas embebidos en un sitio web sin la necesidad de escribir ni una línea de código JavaScript.

- Google Street View Image API: Provee una interfaz en donde se muestran imágenes panorámicas de un mapa en primera persona con la posibilidad de navegación.

- Google Places API JavaScript Library: Ofrece información actualizada de millones de ubicaciones como negocios o puntos de interés. Los datos de los lugares, tales como ubicación exacta, tipo de localidad y calificaciones otorgadas por usuarios, son obtenidos de una base de datos usada por Google Maps y Google+ Local, otras de las cualidades de esta librería son la barra de búsqueda que posiciona exactamente el punto encontrado en el mapa y la característica de autocompletado mientras se digita el lugar a buscar.

- Google Static Maps API: La más simple y básica de todas, proporciona una imagen estática de una porción de un mapa basado en parámetros enviados por una solicitud HTTP. La imagen puede ser de tipo mapa básico, mapa estilizado o satelital.

1.3.7. Here Geocoding REST API

La geocodificación (geocoding) es el proceso de convertir direcciones en formato “Av. Naciones Unidas, esq. y Av. 6 de Diciembre, Quito, Ecuador” a coordenadas geográficas, es decir, latitud y longitud (-0.176156, -78.480223), los cuales pueden usar usados para ubicar puntos en mapas. La geocodificación inversa (reverse geocoding) es el proceso de convertir coordenadas en una dirección legible para humanos.

“En general la geocodificación es el proceso que inicia con la entrada de una descripción (…), se identifica la semántica de la descripción para descomponerla por componentes definidos en el algoritmo de normalización, se estandariza estos componentes para unificar y homologar las estructura de los componentes y posteriormente se realiza la búsqueda en una fuente de datos para realizar la comparación por componentes y entrega el mejor candidato de la búsqueda para mostrarlo sobre un mapa.” (Vargas y Horfan, 2013, p.7)

La API de Here Geocoding Service provee una vía directa para acceder a estos servicios a través de una solicitud HTTP. El servicio web es usado en aplicaciones web y móviles para geocodificar direcciones o coordenadas ya conocidas.

En el presente proyecto se utilizará Google Maps JavaScript API ya que se necesita añadir una capa para el mapa de calor y en el mismo manipular los datos a desplegar así como otras opciones de interés, tales como: radio de los puntos visibles, gradiente de colores, entre otros.
Adicionalmente se usará Here Geocoding REST API, por su nivel de precisión en el proceso de geocodificación comparado con otras APIs como la de Google Maps.

El motivo para usar estas herramientas es la facilidad de integración a las aplicaciones web y móviles, además los servicios están almacenados en la

nube y se pueden acceder de una forma sencilla. Además, Google Maps JavaScript API, por ser un complemento desarrollado por Google posee documentación oficial y actualizada constantemente para llevar a cabo un desarrollo sustentable, también es un complemento que se puede usar libremente y que mejora con las experiencias de uso de cada uno de los usuarios.

2. Capítulo II: Desarrollo del prototipo

La arquitectura física del sistema son los recursos tangibles con los que se cuenta para realizar un sistema, como por ejemplo servidores, clientes, switchs, routers, etc. En el presente proyecto se utilizó dos servidores: un servidor de aplicaciones y un servidor de base de datos, servicio web y cron.

2.1. Arquitectura Lógica

La arquitectura lógica la conforman los componentes no tangibles que se crearán en el proyecto, como por ejemplo el servicio web o el cron, que son procesos que se ejecutan dentro de un mismo servidor físico pero que están separados dentro de la programación y que pueden comunicarse entre ellos.

La arquitectura lógica está conformada por 3 capas: aplicativo web/móvil, servicio web y base de datos.

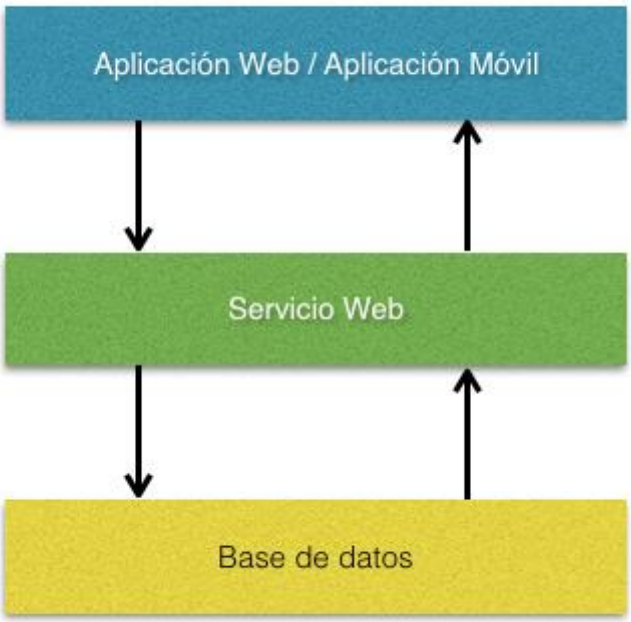


Figura 11. Arquitectura lógica.

En la Figura 11, el modelo de 3 capas indica que cada petición realizada desde la aplicación, será dirigida hacia el servicio web, y este será el encargado de establecer comunicación con la base de datos para acceder a la información necesaria y devolver una respuesta. Dicha respuesta tendrá que seguir el camino anterior en dirección opuesta hasta ser procesada.

Dentro de la arquitectura lógica podemos encontrar 3 capas las cuales poseen funciones propias y aisladas, y que mantienen comunicación entre sí para responder a peticiones externas. A continuación, se describen las funciones y objetivos de cada capa.

- **Aplicación Web:** Se encarga del despliegue del sitio web, manejando imágenes, estilos y texto HTML. Físicamente, se localiza en su propio servidor.
- **Aplicación Móvil:** Físicamente, se encuentra instalado en cada dispositivo. Maneja el conjunto de actividades que desempeñan una función específica.

- **Servicio Web:** Es la capa intermedia y posee la mayoría de la lógica del negocio, se encarga de construir y dar formato correcto a los objetos JSON que serán devueltos como respuesta hacia la capa superior. Se encuentra alojado en otro servidor junto con la base de datos.
- **Base de datos:** El lugar donde se encuentran almacenados los datos de interés los cuales son accedidos por medio de lenguaje de consultas SQL. Está ubicado, físicamente, en el mismo servidor que el servicio web.

2.2. Arquitectura Física

La arquitectura física del aplicativo consta de un servidor de aplicación, un servidor de base de datos y un servicio web, y los diferentes dispositivos (clientes) que realizarán las peticiones al servidor de aplicación.

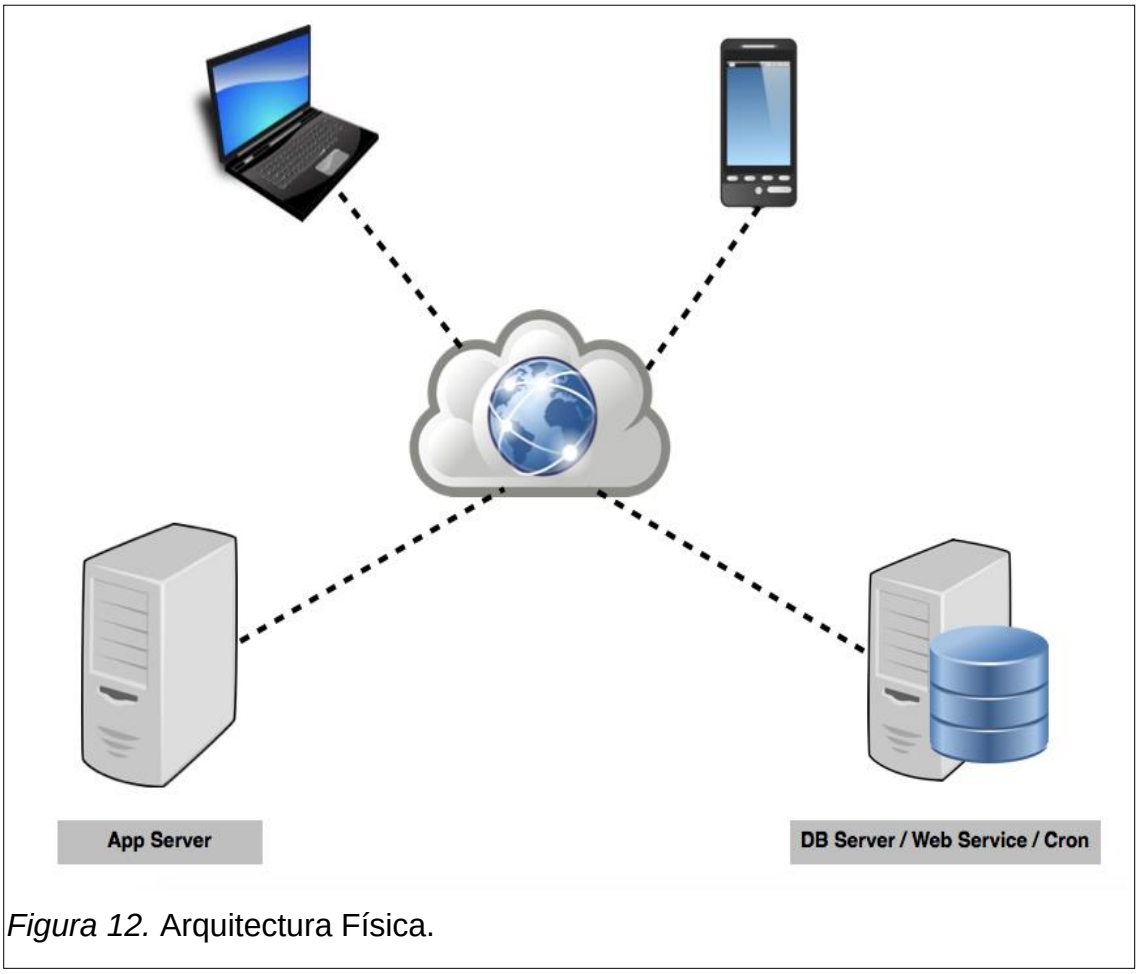


Figura 12. Arquitectura Física.

En la Figura 12 se describe la arquitectura física del proyecto, el cliente se conecta a través de internet al servidor de aplicaciones, aquí se encuentra el aplicativo web; este a su vez hace una llamada al servidor donde se encuentra alojada la base de datos y el web service. El web service construye la respuesta con la información solicitada por el cliente.

2.3. Funcionalidad del sistema

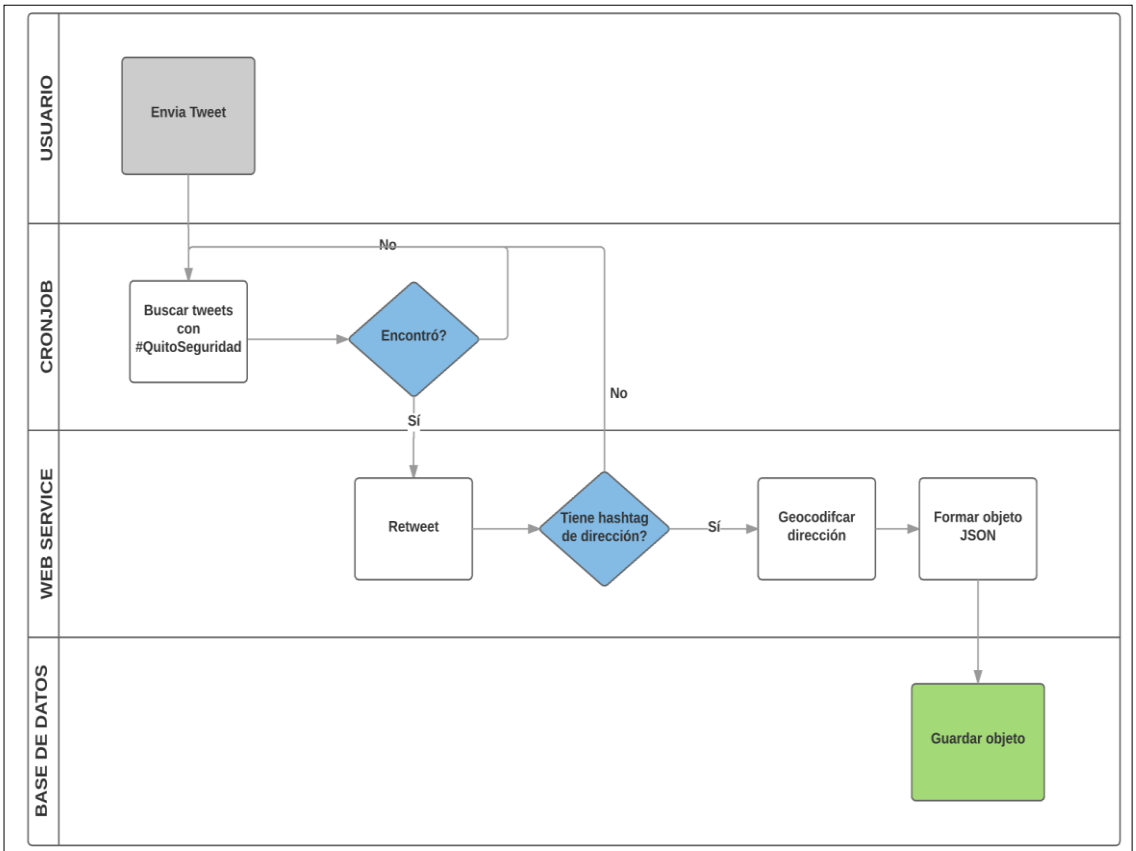


Figura 13. Proceso de búsqueda de tweets.

De acuerdo a la Figura 13, el proceso se inicia en la red social Twitter, cuando una persona es víctima o testigo de un asalto o delito en las calles del Distrito Metropolitano de Quito, puede alertar a los demás habitantes acerca del lugar donde ocurrió el hecho redactando un tweet con la dirección del hecho y el nombre de la cuenta creada para este proyecto (@QuitoSeguridad), antecedido por el símbolo # (hashtag). Por ejemplo: *"Me robaron el celular en la #GaspardeVillaroel y #6deDiciembre #QuitoSeguridad mucho cuidado!".*

El paso siguiente es tomar esta información a través de la función creada para buscar tweets periódicamente, las funciones de este tipo se denomina cron, (el tiempo de cada intervalo de búsqueda, así como la cantidad de tweets a buscar puede ser modificado desde una administración remota de parámetros), dicha función tomará una cantidad específica de tweets y los analizará uno por uno. La primera acción será retwittear, es decir, publicar este tweet desde la cuenta de este proyecto, después se toma la dirección contenida en el tweet y usando el servicio de Here Geocoding REST API, se transforma dicha dirección en coordenadas geográficas las cuales se almacenan de la base de datos junto con otra información de interés tal como el id del tweet, nombre del usuario autor y fecha y hora del mismo.

Por otro lado, la funcionalidad del sitio web ha sido diseñada para ofrecer una representación visual de los lugares más peligrosos en la ciudad de Quito, por lo cual, dentro del sitio web existirá un mapa de calor de gran tamaño para su fácil visualización, además el sitio desplegará una columna con tweets de la cuenta @QuitoSeguridad con la información recopilada para mostrar en el mapa. Junto a esta columna se encontrará un formulario para sugerencias y comentarios que puedan surgir entre los usuarios de la página. Además una última columna será colocada con una pequeña descripción y objetivo de este proyecto.

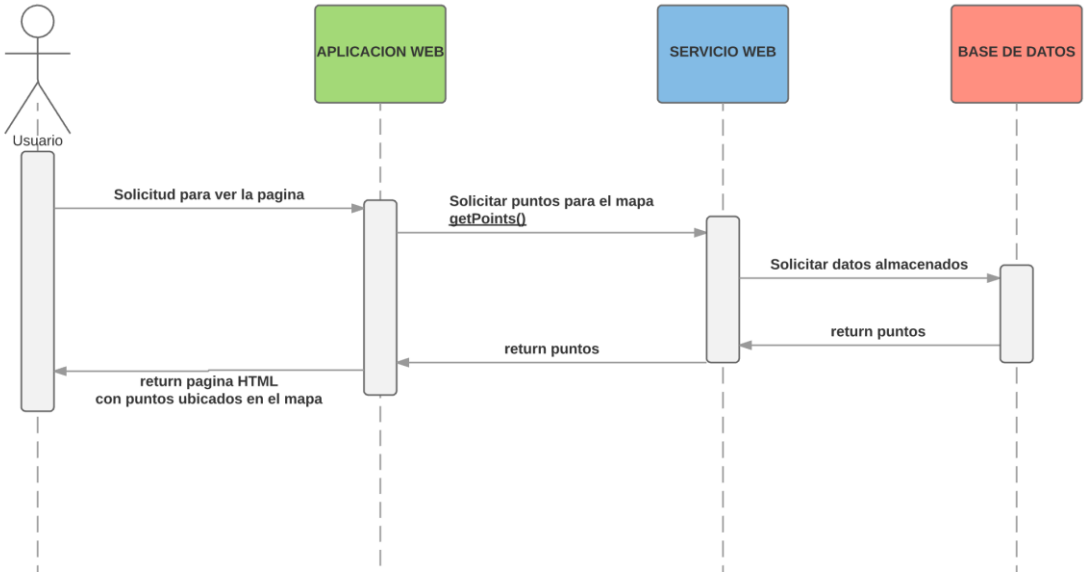


Figura 14. Proceso de despliegue del sitio web.

El proceso de despliegue del sitio web, mostrado en la Figura 14, empieza con la solicitud de un navegador hacia la dirección del sitio oficial **http://uioseguridad.herokuapp.com/**, esta petición llegará al servidor de aplicaciones, el cual solicitará los puntos guardados al servicio web y a su vez este realizará una consulta hacia la base datos para recopilar las coordenadas alojadas previamente, basados en tweets de usuarios. Estos puntos son enviados junto con la respuesta del servidor para mostrar el sitio, y en el mapa serán ubicados los puntos correspondientes.

En cuanto a la aplicación móvil el proceso es bastante similar. Al abrir la aplicación nativa Android, se realiza una llamada a la misma función que retorna los puntos para ubicar en el mapa. Existirá un menú el cual podrá mostrar el mapa observado inicialmente, los tweets de la cuenta @QuitoSeguridad y el logo oficial de este proyecto bajo el título “Sobre Nosotros”.

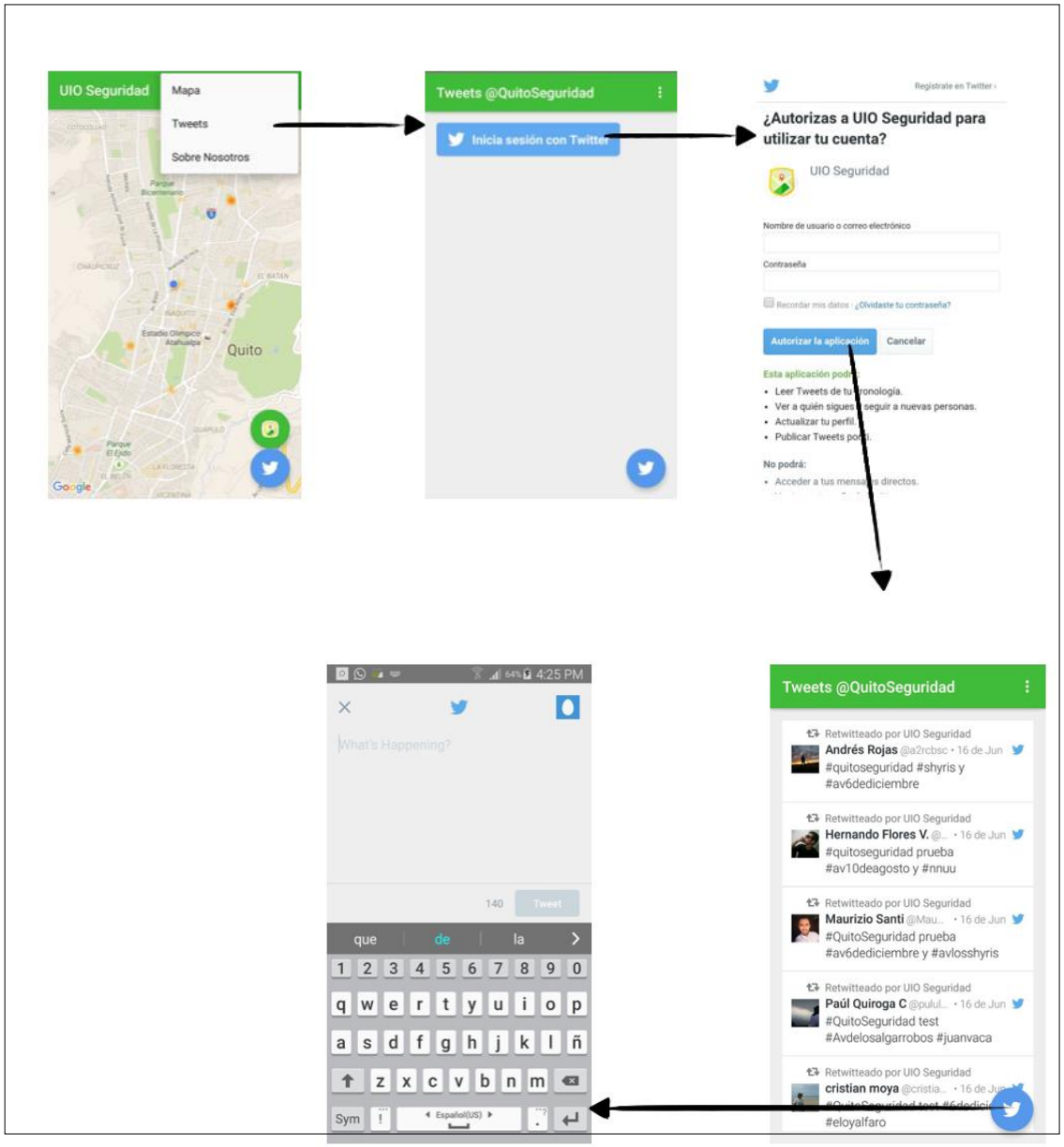


Figura 15. Proceso de autenticación y envío de tweets.

Como indica la Figura 15, además del menú, en las pantallas contenedoras del mapa y de la lista de tweets, se encontrará un botón flotante con el símbolo y color representativo de Twitter, el cual permitirá redactar tweets desde la misma aplicación.

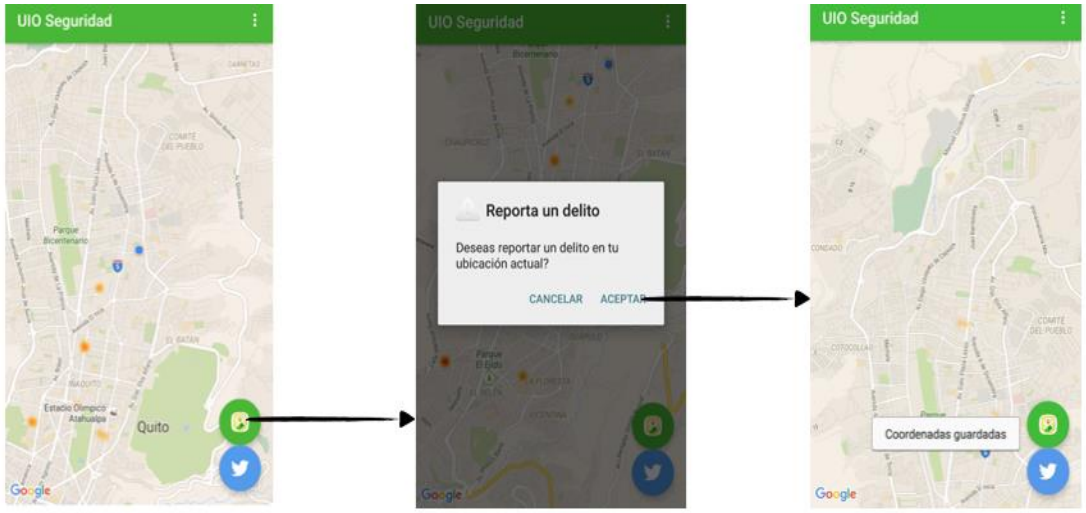


Figura 16. Proceso gráfico de toma de coordenadas con GPS.

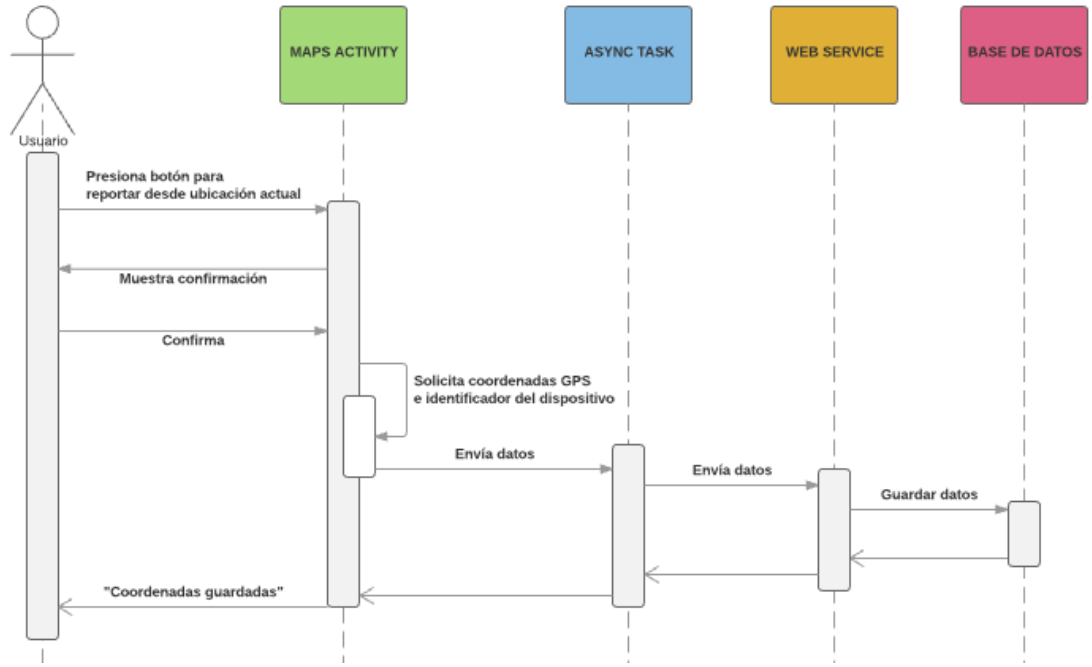


Figura 17. Proceso lógico para almacenado de coordenadas GPS.

Acorde con las Figuras 16 y 17, en la pantalla inicial, la que contiene el mapa, se encontrará un botón flotante adicional con el logo y color escogido para este proyecto, el cual permite reportar un delito usando las coordenadas provistas por el GPS del dispositivo móvil, es decir, se enviará la posición actual y se almacenará como un punto peligroso el cual se apreciará inmediatamente en la misma pantalla si el proceso es exitoso. Adicionalmente para poder identificar a

cada dispositivo que use esta funcionalidad, se tomará el número celular registrado en el dispositivo móvil, en el caso de que el mismo no se encuentre disponible, se tomará la dirección MAC del dispositivo, y dado el caso que esta dirección tampoco se encuentre disponible, (como en el caso de la actualización de Android 6.0 (Marshmallow) en la cual la lectura de estos y otros datos de la red fueron restringidos por motivos de seguridad) se almacenará el IMEI del dispositivo.

2.4. Product Backlog

Encapsula todos los requerimientos del sistema y de usuario en forma de historias de usuario.

Tabla 4. *Product backlog. Historias de usuario.*

ID	Historia	Esfuerzo	Prioridad	Estado
DMQ-1	Diseño de página web	2	Baja	Sin comenzar / iniciado / terminado.
DMQ-2	Diseño backend	3	Media	Sin comenzar / iniciado / terminado.
DMQ-3	Desarrollo Cron	5	Alta	Sin comenzar / iniciado / terminado.
DMQ-4	Desarrollo Web Service	4	Alta	Sin comenzar / iniciado / terminado.
DMQ-5	Integración Web y backend.	3	Alta	Sin comenzar / iniciado / terminado.
DMQ-6	Diseño de aplicación móvil: actividades	2	Baja	Sin comenzar / iniciado / terminado.

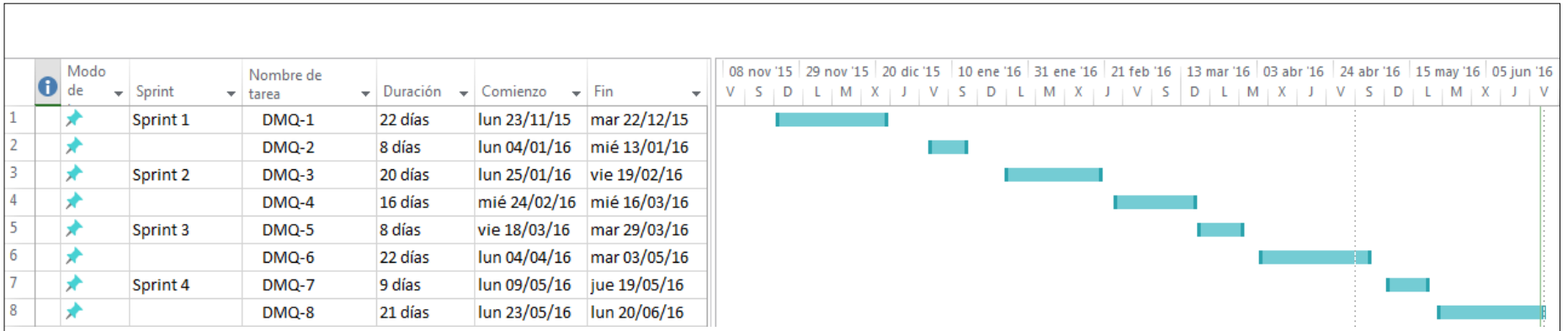
	estáticas.			
DMQ-7	Desarrollo de mapa de calor e integración con web service (Android.)	3	Media	Sin comenzar / iniciado / terminado.
DMQ-8	Integración Twitter Android.	4	Alta	Sin comenzar / iniciado / terminado.

En la Tabla 4 se detallan las historias de usuario que se identificaron en la fase inicial del proyecto, cada historia de usuario tiene:

- ID que es un identificador de cada historia de usuario.
- Historia, nombre que se le asigna a cada historia de usuario, tratando de que esta sea clara y concisa.
- Esfuerzo, valorado en puntos, siendo 5 el valor máximo de esfuerzo y 1 el mínimo.
- Prioridad, indica el grado de importancia que tiene la historia, es designada por el cliente, además este valor sirve para definir un orden en el desarrollo de las tareas.
- Estado, avance en el que se encuentra la historia de usuario, el equipo lo define según el avance del proyecto.

2.5. Sprint backlog

Se listan los requisitos de cada historia de usuario definidos en el product backlog, para convertirlos en tareas a realizar en cada sprint. Las tareas se dividirán entre el equipo y se estimarán los tiempos para realizar cada tarea. El tiempo de duración de cada sprint dependerá de las tareas asignadas y la complejidad de cada historia.



	Modo de	Sprint	Nombre de tarea	Duración	Comienzo	Fin
1		Sprint 1	DMQ-1	22 días	lun 23/11/15	mar 22/12/15
2			DMQ-2	8 días	lun 04/01/16	mié 13/01/16
3		Sprint 2	DMQ-3	20 días	lun 25/01/16	vie 19/02/16
4			DMQ-4	16 días	mié 24/02/16	mié 16/03/16
5		Sprint 3	DMQ-5	8 días	vie 18/03/16	mar 29/03/16
6			DMQ-6	22 días	lun 04/04/16	mar 03/05/16
7		Sprint 4	DMQ-7	9 días	lun 09/05/16	jue 19/05/16
8			DMQ-8	21 días	lun 23/05/16	lun 20/06/16

Figura 18. Cronograma desarrollo Sprints.

Los sprints se realizan en base al product backlog, la Figura 18 muestra el desarrollo general de los sprints obtenidos.

A continuación se describe el sprint 1 y sprint 4 como evidencia de la aplicación de la metodología para las historias de usuario DMQ-1 y DMQ-2. Los demás sprints generados pueden encontrarse en los anexos.

2.5.1 Sprint 1

Contendrá el diseño de la página web y el diseño de backend, correspondientes a las historias de usuario DMQ-1 y DMQ-2 respectivamente.

Tabla 5. *Sprint 1.*

ID	Descripción	Criterio de aceptación
DMQS1-1	Diseño de la arquitectura del proyecto.	Crear un esquema en el que se pueda evidenciar la arquitectura del proyecto, especificación de servidores a usar y cómo se conectarán.
DMQS1-2	Elaborar estructura HTML de página web.	Crear un esquema base de la página web, dividida en bloques para diferenciar cada área a diseñar.
DMQS1-3	Integrar sitio web con Google Maps API.	En el bloque designado, usando Google Maps, desplegar un mapa con ubicación en la ciudad de Quito.
DMQS1-4	Generar y agregar twitter	En el bloque designado

	timeline widget.	desplegar los tweets de la cuenta oficial por medio de un widget generado por Twitter.
DMQS1-5	Agregar formulario de sugerencias y comentarios.	En el bloque designado para formulario, agregar tres campos: nombre, email y comentario; los mismos que serán recibidos vía email. Verificar validez del formulario con Google recaptcha.
DMQS1-6	Agregar sección “Sobre Nosotros.”	En el bloque designado para la sección “Sobre nosotros”, agregar una descripción breve, el objetivo del sitio web y el logo oficial.
DMQS1-7	Diseño de tablas de base de datos.	Crear el diagrama entidad-relación de la base de datos con la que se trabajará en el proyecto.

Desarrollo del Sprint:

DMQS1-1

En esta tarea se construirá la arquitectura física para el proyecto, de tal forma que se pueda visualizar la estructura y flujo del sistema. Se construirá un esquema basado en la Figura 12 mencionada anteriormente.

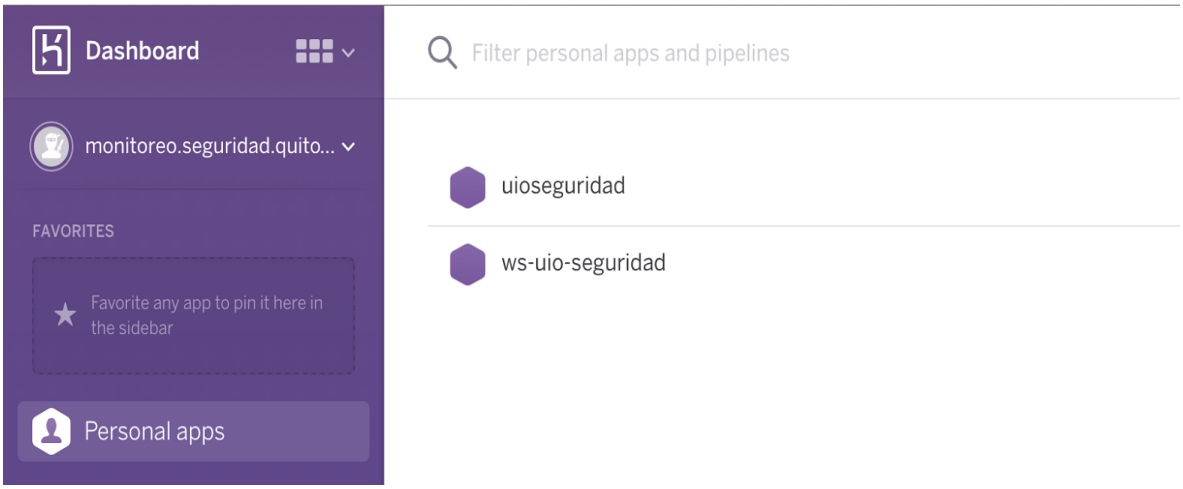


Figura 19. Aplicaciones creadas en Heroku.

La Figura 19 representa las aplicaciones que fueron creadas para alojar en los dos servidores como lo indica la Figura 12. En el servidor de aplicaciones (App Server) se alojará la aplicación uioseguridad y en el segundo servidor se alojará la aplicación ws-uio-seguridad, la cual contiene el servicio web, cronjob y la base de datos.

El motivo para usar una arquitectura compuesta por dos servidores, es que, tanto la aplicación móvil como el servidor de aplicaciones web se conectarán directamente al servidor que contiene el servicio web.

DMQS1-2

Durante esta tarea se realizó la estructura básica y estilo de la página web, se designaron espacios donde se ubicarán, más adelante, todos los componentes de la misma.

Figura 20. Mockup página web.

La Figura 20 muestra el diseño a rasgos generales de la página web, la cual contiene tres bloques principales:

- Header, contiene el título principal de la página web, el cual está designado como “Quito Seguridad”.
- Body, dividido a su vez en cuatro bloques en donde se visualizará el contenido de la página web. En el bloque Mapa, se mostrará el mapa de la ciudad de Quito. En el bloque Twitter Timeline se desplegará todos los tweets de la cuenta creada para este proyecto. En el bloque de Formulario Sugerencias se creará un formulario para que, por medio de los usuarios, se pueda mantener una mejora continua en el aplicativo. En el bloque Sobre Nosotros, una breve descripción del sitio web.
- Footer, derechos de autor.

DMQS1-3

Para la siguiente tarea se realizó la integración del API de Google Maps en la página web, la cual despliega dos tipos de mapas, un satelital y un estándar para la búsqueda de calles o direcciones en la ciudad de quito.

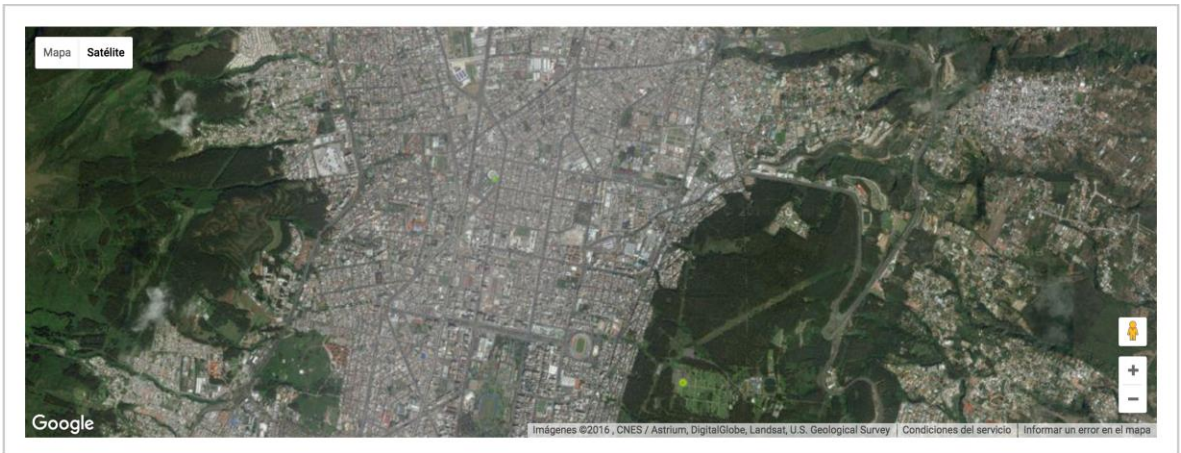


Figura 21. Mapa Satelital.

La Figura 21 muestra la integración de Google Maps en la página web, modificando la configuración básica, el mapa muestra la ciudad de Quito. El mapa tiene funcionalidades propias del API de Google Maps como acercar, alejar y cambiar el tipo de mapa (Satelital/Estándar).

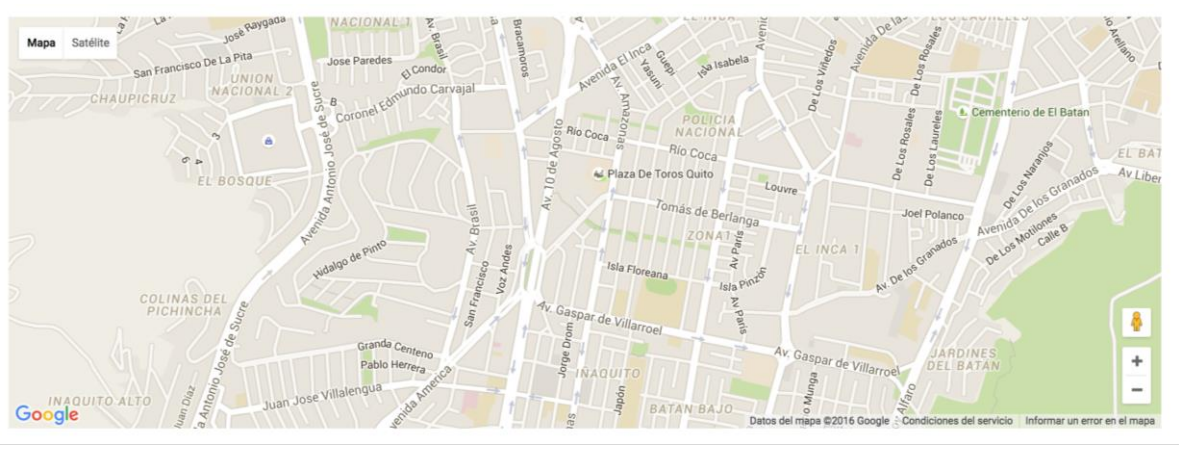


Figura 22. Mapa estándar Google API.

En la Figura 22 se aprecia el mapa con las calles de un sector de Quito, en la esquina superior izquierda se encuentran las opciones para cambiar el tipo de mapa.

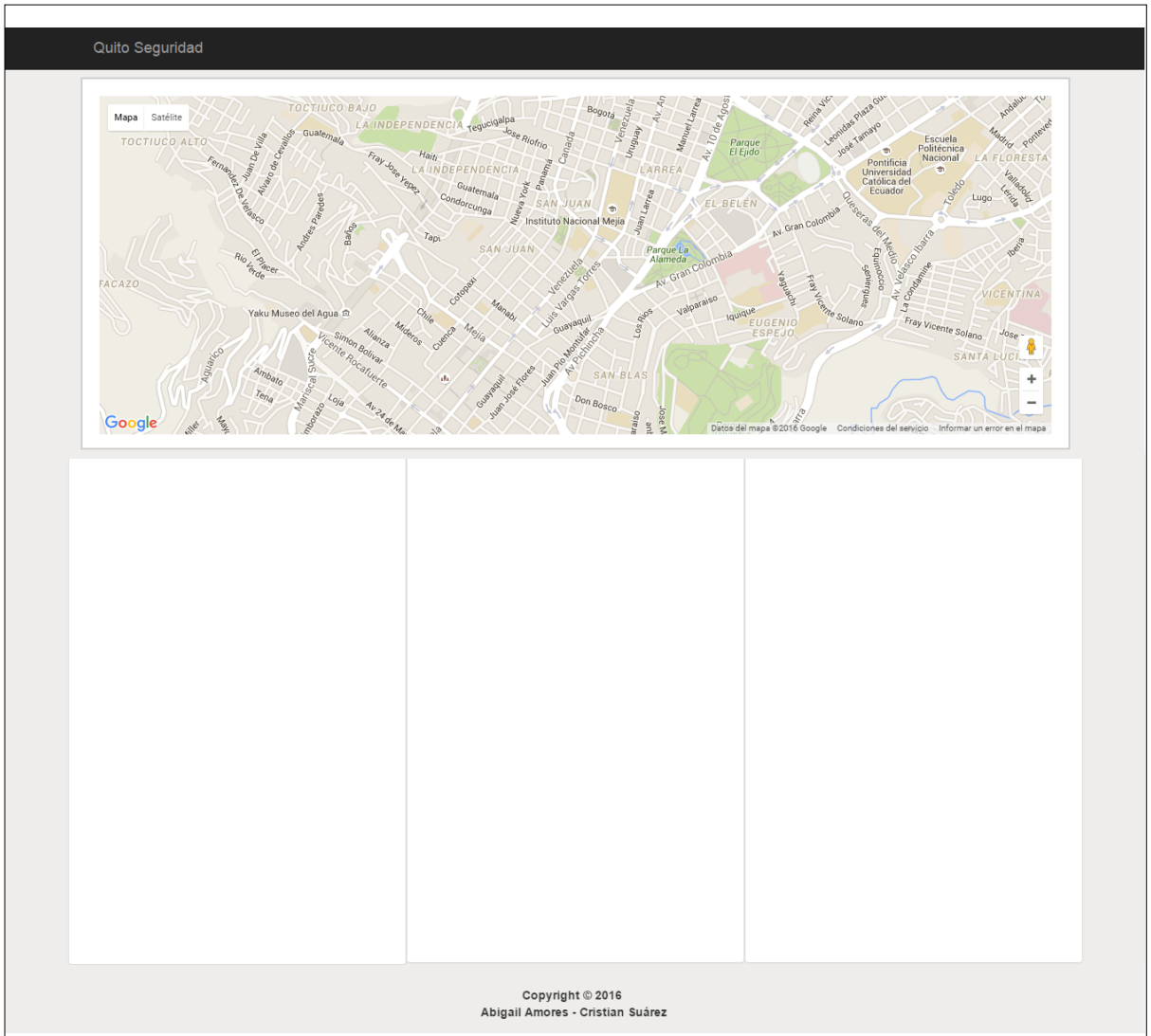


Figura 23. Página web con Google Maps.

En la Figura 23 se puede ver la página web con el resultado de la integración del mapa en el bloque designado con el uso de Google Maps API.

DMQS1-4

En la siguiente tarea se realiza la integración del widget de Twitter de la cuenta @QuitoSeguridad para que se pueda observar los acontecimientos registrados por esta cuenta.

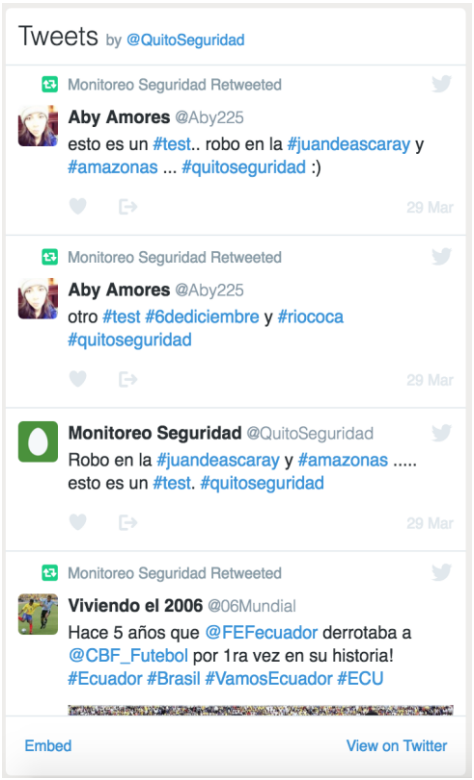


Figura 24. Timeline twitter de la cuenta @QuitoSeguridad.

La Figura 24 muestra la integración de Twitter en la página web, en este bloque se puede visualizar los tweets publicados por todos los usuarios que mencionaron a dicha cuenta para hacer discusiones sobre temas específicos propuestos o de manera informativa.

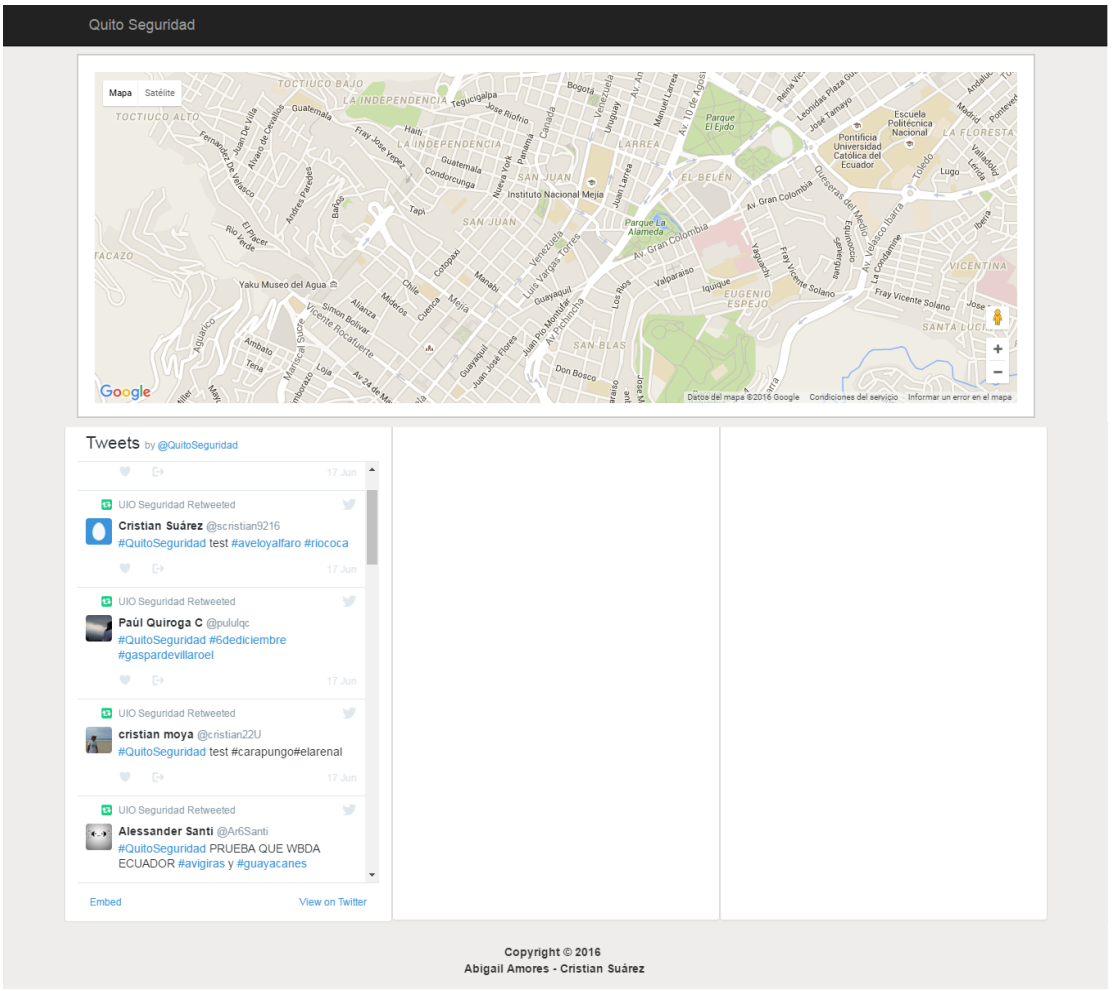


Figura 25. Diseño página web.

El avance de página web que incluye la integración de Google Maps y Twitter se muestra la Figura 25.

DMQS1-5

Durante la siguiente tarea se realizará un formulario para que los usuarios puedan dejar comentarios y sugerencias para una mejora continua de la página web

.

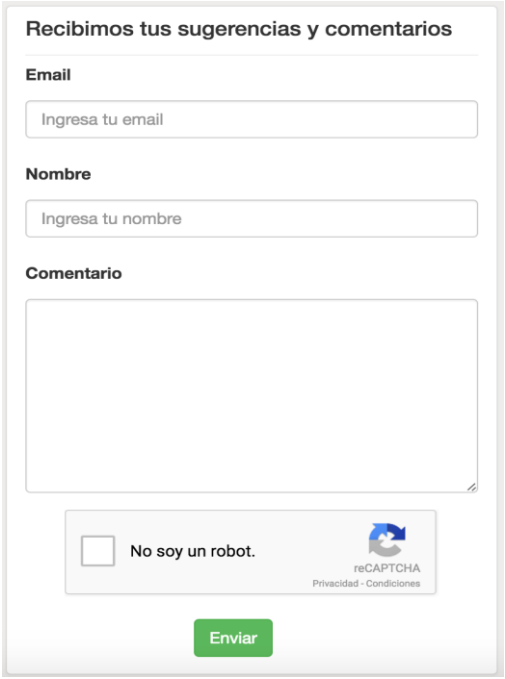


Figura 26. Formulario de Sugerencias y Comentarios.

En la Figura 26 se puede apreciar el formulario de sugerencias y comentarios, el cual se conforma de cuatro bloques, el primer bloque es donde el usuario debe ingresar el email, el segundo bloque es para el ingreso del nombre, el tercer bloque es para ingresar el comentario y por último se agregó un campo reCAPTCHA, el cual es un servicio gratuito para la protección de spam. Se enviará un correo electrónico a la cuenta de monitoreo (monitoreo.seguridad.quito@gmail.com) para administrar las sugerencias.

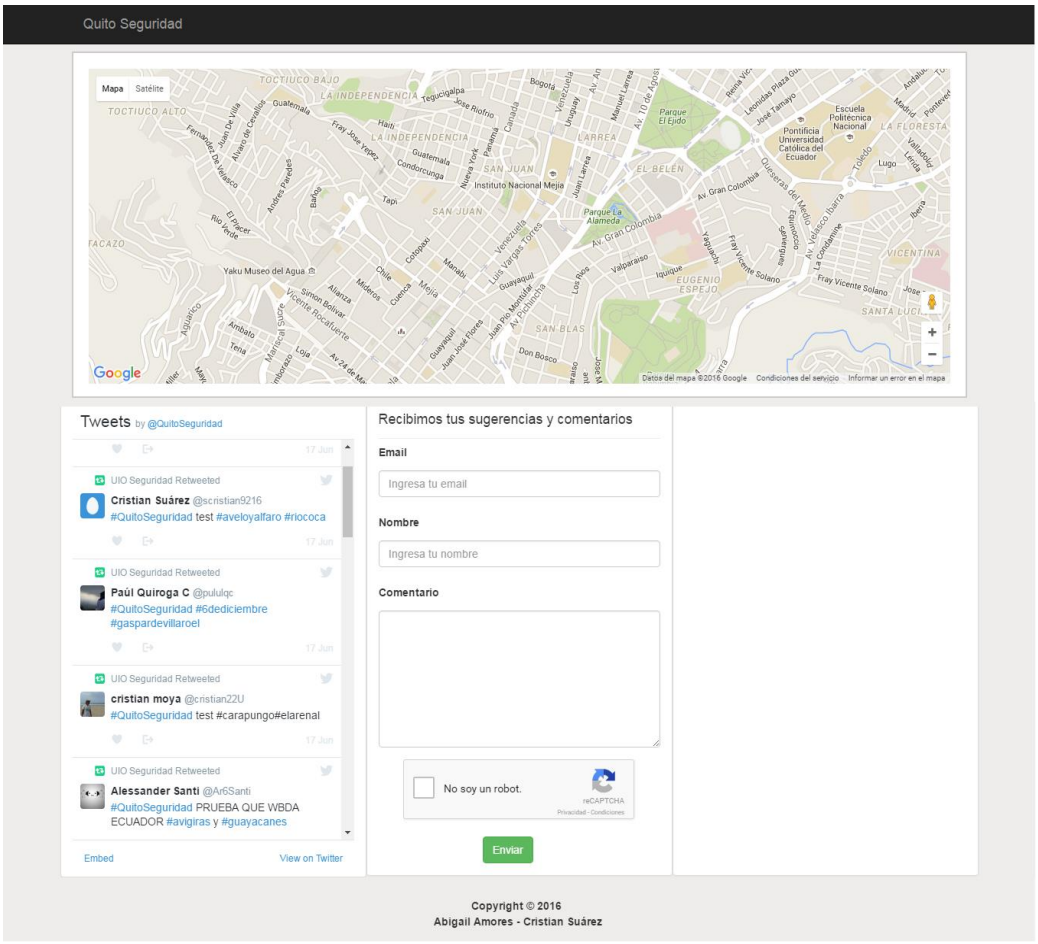


Figura 27. Diseño página web.

En la Figura 27 se muestra el progreso de integración hasta la presente tarea.

DMQS1-6

Figura 28. Sección “Sobre Nosotros”.

En esta tarea se realizó una pequeña descripción de la página web y una manera rápida de usar el aplicativo, incluye el logotipo oficial del sitio que además será usado en la aplicación móvil como se muestra en la Figura 28.

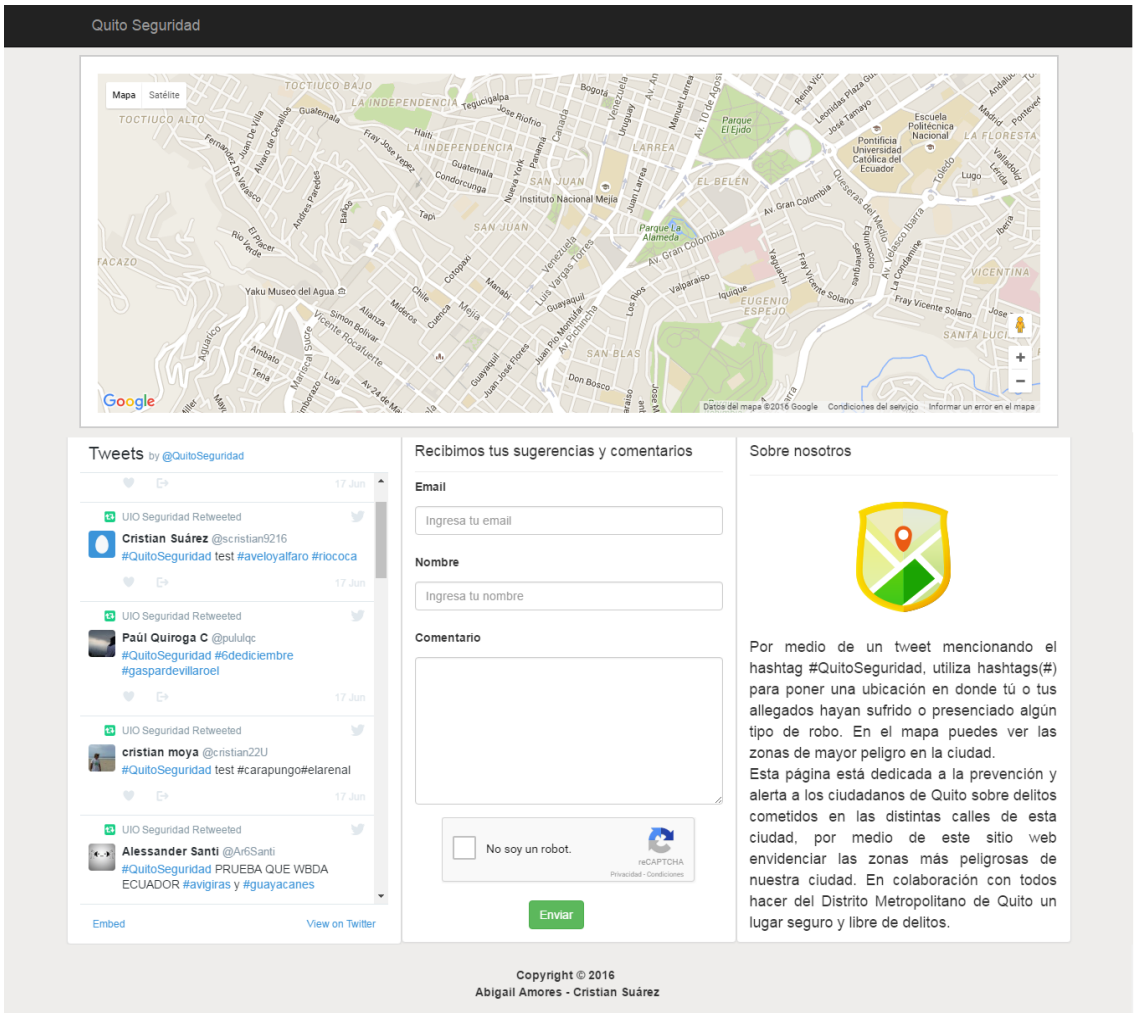


Figura 29. Diseño final de la página web.

Al finalizar las tareas relacionadas al diseño de la página web se puede visualizar el sitio completo como muestra la Figura 29.

DMQS1-7

En la siguiente tarea se realizará el esquema entidad-relación para definir la estructura de base de datos que será utilizada a lo largo del desarrollo.

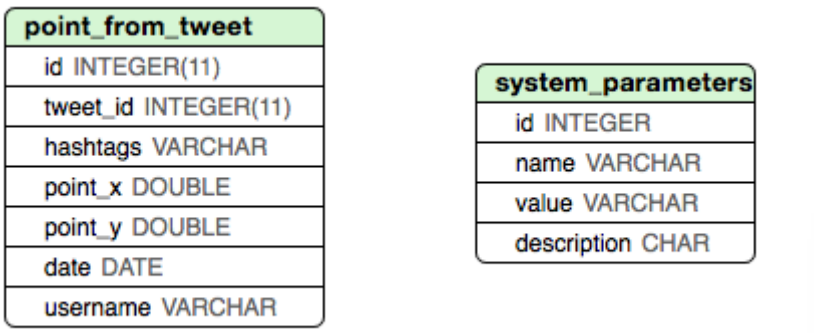


Figura 30. Diagrama entidad-relación.

En la Figura 30 se muestran dos tablas principales, la tabla “point_from_tweet” contiene los datos a guardar de los tweets que se buscarán referenciados a la cuenta @QuitoSeguridad. El campo tweet_id guardará el id propio de cada tweet, el campo hashtag guardará los valores encontrados antecedidos por un hashtag (#) que en este caso será una dirección, los campos point_x y point_y almacenarán las coordenadas de cada dirección, y finalmente el campo date guardará la fecha de creación del tweet.

En la tabla “system_parameter” se almacenará parámetros del sistema para aislar y administrar variables específicos del sistema, como por ejemplo el criterio de búsqueda de tweets, la cantidad de tweets a buscar o la dirección del web service a comunicarse.

El modelo de base de datos posee una sola tabla necesaria para cumplir la funcionalidad la cual contiene datos referentes a cada tweet ya que el sistema gira a entorno a la información provista por cada tweet. No es necesario almacenar ninguna información extra.

Por otro lado la tabla de parámetros del sistema es utilizada con fines administrativos en torno a valores clave usados en el sistema.

El uso de dos tablas es debido a que el flujo del sistema es en base a los tweets, es decir se recolecta información de los tweets enviados por los usuarios de modo que se guarda información enviada por twitter.

2.5.2 Sprint 4

En el sprint 4 se desarrollará el mapa de calor y se integrará la aplicación móvil con twitter. Las historias de usuario a tratar son DMQ-7 y DMQ-8.

Tabla 6. *Sprint 4.*

ID	Descripción	Criterio de aceptación
DMQS4-1	Integración de Google Maps y Android.	Desplegar el Mapa proporcionado por el API de Google Maps ubicado en la posición obtenida por el GPS de cada dispositivo.
DMQS4-2	Lectura de web service en la aplicación móvil.	Obtener los puntos almacenados en la base para desplegar en el mapa.
DMQS4-3	Integración Twitter y Android.	Generación de claves y permisos por medio de la plataforma Fabric para la comunicación con Twitter.
DMQS4-4	Despliegue de timeline de Twitter en activity.	Desplegar Twitter timeline de la cuenta @QuitoSeguridad.
DMQS4-5	Login con Twitter y redacción de tweets.	Login en Twitter con cuenta de cada usuario y

		creación de tweets por medio de la aplicación.
DMQS4-6	Reportar delito desde ubicación actual.	Reportar un delito desde la ubicación basada en el GPS de cada dispositivo.

Desarrollo del sprint:

DMQS4-1

Para esta tarea se realizará la integración de la API de Google Maps con Android.

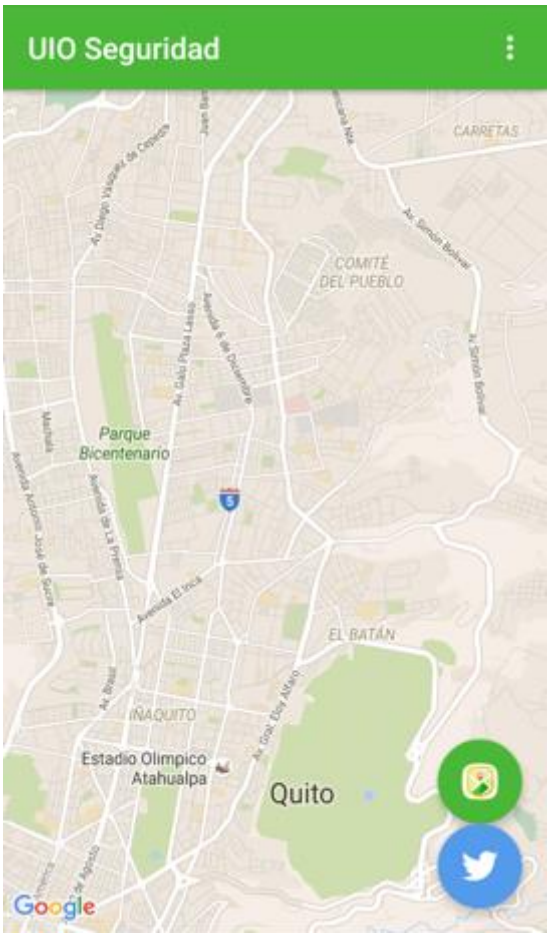


Figura 31. Integración Google Maps.

En la Figura 31 se muestra el mapa proporcionado por el API de Google Maps, además toma la posición actual por medio del GPS de cada dispositivo para abrir la

aplicación desde su ubicación actual. Además se creó un botón flotante que llevará a la pantalla de redacción de tweets.

DMQS4-2

Para esta tarea se establecerá la comunicación entre la aplicación móvil y el web service. Para solicitar los datos al web service se realizó una tarea asíncrona la cual hace la solicitud al servidor para tomar los registros.

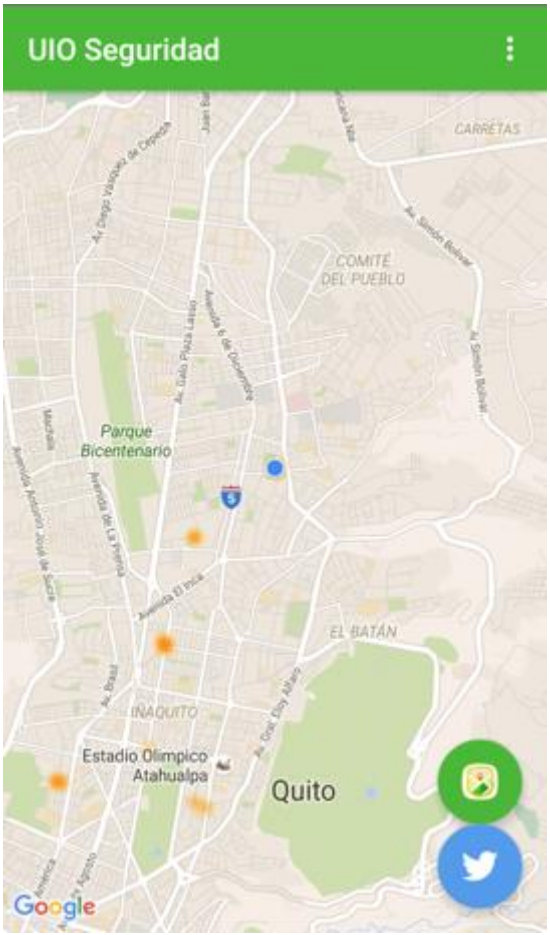


Figura 32. Mapa que muestra puntos recibidos de web service.

Como se muestra en la Figura 32, se desplegará el mapa con las zonas donde se reportaron delitos, es decir con la información almacenada en la base de datos.

DMQS4-3

Para esta tarea se realizará la integración con Twitter, se utilizará la plataforma

Fabric que provee un kit de desarrollo para implementar funcionalidades propias de Twitter como obtener el timeline de una cuenta específica, login de cuenta personal de Twitter, redactar tweet. Para la integración con Twitter se requiere generar tokens o claves para interactuar con twitter.

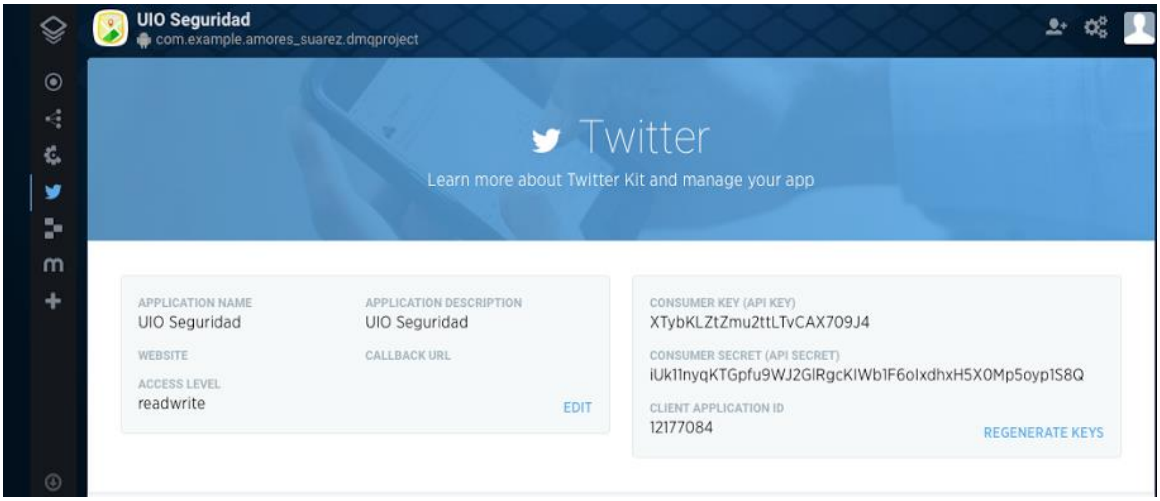


Figura 33. Generación de tokens por medio de Fabric para Twitter.

En la Figura 33 se puede observar los tokens y claves registradas para la aplicación móvil.

DMQS4-4

Para desplegar el timeline de la cuenta de Twitter @QuitoSeguridad, se crea una lista (ListView) para desplegar en la actividad de Android, en la misma que se carga el timeline correspondiente a la cuenta.

Figura 34. Twitter timeline.

En la Figura 34, se despliega la información de Twitter referente a la cuenta @QuitoSeguridad, sección en la que se recopila tweets para dar información al usuario. Esta sección se mostrará cuando el usuario inicie sesión con una cuenta de twitter.

DMQS4-5

Por medio de la plataforma Fabric es posible iniciar sesión con Twitter, de esta forma cada usuario tiene la posibilidad de visualizar el timeline de la cuenta @QuitoSeguridad y de redactar tweets.

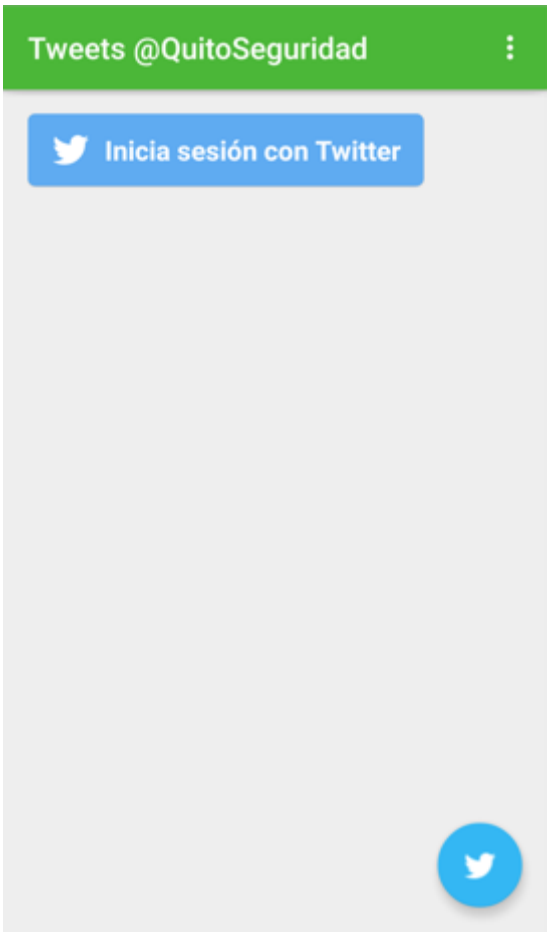


Figura 35. Botón inicio de sesión twitter.

Como se muestra en la Figura 35, con el botón de inicio de sesión de Twitter el usuario tiene la posibilidad de acceder a la funcionalidad de su cuenta como crear tweets y ver el timeline de la cuenta @QuitoSeguridad, una vez ingresadas las credenciales.

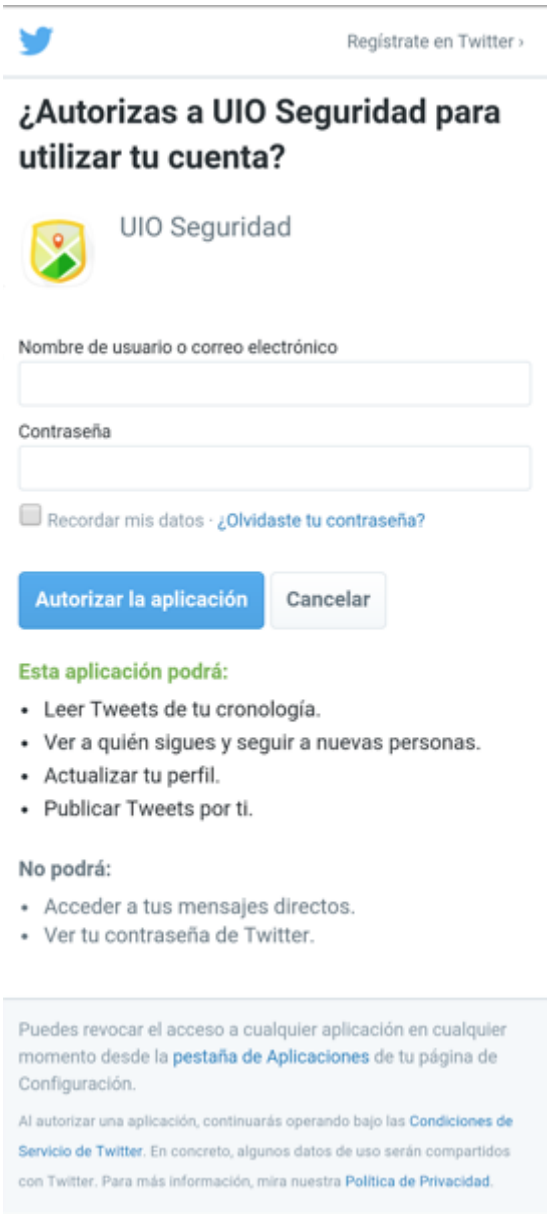


Figura 36. Login y permisos de acceso a cuenta Twitter.

La Figura 36 muestra la interfaz de permisos que se requieren para iniciar sesión en Twitter.

Figura 37. Redacción de tweets.

La Figura 37 muestra la interfaz para redactar un tweet una vez iniciada sesión sin necesidad de usar una aplicación externa o redireccionar a un navegador.

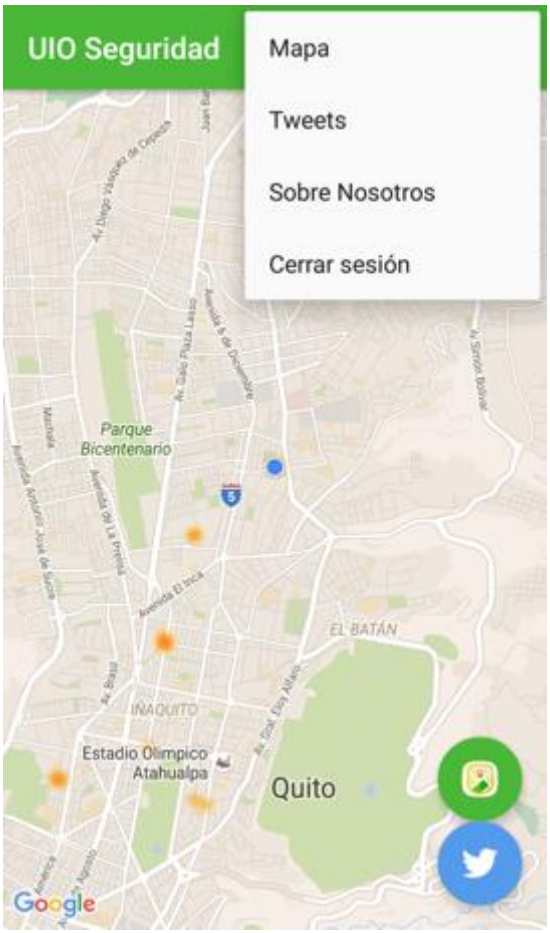


Figura 38. Cerrar sesión cuenta Twitter.

Una vez iniciada sesión con una cuenta de Twitter se habilita un botón dentro del menú principal que cierra la sesión de la cuenta Twitter como se observa en la Figura 38.

DMQS4-6

Para esta tarea se creará una opción para reportar delitos desde la ubicación proporcionada por el GPS, enviará las coordenadas proporcionadas por cada dispositivo para colocar dicha información en el mapa de calor.

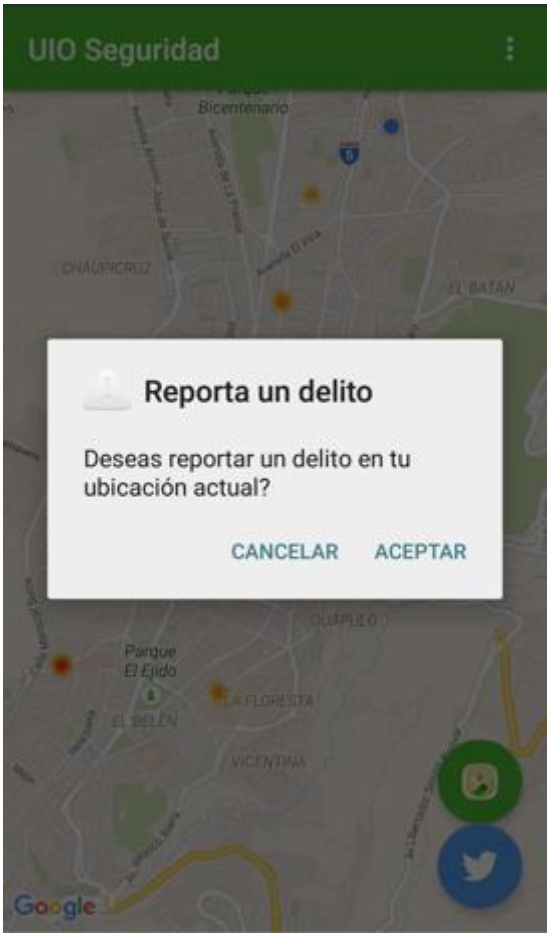


Figura 39. Confirmación para reportar delito.

La Figura 39 muestra la confirmación para reportar un delito de a partir de la ubicación obtenida por cada dispositivo por medio del GPS.

2.6. Sprint Planning

Es una reunión donde la principal función se compone de dos partes, la primera es donde el cliente presenta los requisitos para realizar el sistema de una manera priorizada, el equipo analiza los requisitos y plantea preguntas al cliente para tener una visión clara del proyecto a realizar. La segunda parte es una reunión del equipo en la que se creará una táctica que permita conseguir el mejor resultado con un esfuerzo mínimo. Se creará el listado de tareas en el sprint backlog y realizar la estimación de esfuerzo.

Para aplicar esta técnica en el proyecto se definió los requerimientos necesarios para el desarrollo del proyecto y se los organizó por prioridad como se detalla en la Tabla 4.

2.7. Scrum Daily Meeting

Reuniones que se realizan diariamente para confirmar los avances del proyecto, son reuniones cortas donde se trata de evidenciar el avance diario de las tareas asignadas a cada uno de los miembros del equipo. Se plantea hacer una revisión de que se realizó el día anterior, los problemas para avanzar con el desarrollo de las tareas y lo que se plantea realizar para el presente día.

Para aplicar esta técnica se realizaron revisiones periódicas de las tareas asignada a cada uno de los miembros del equipo.

2.8. Spring Retrospective

Una reunión cuyo objetivo es mantener una mejora continua en el desarrollo del proyecto y generar un trabajo de calidad, el equipo analiza cómo ha sido la manera de trabajar al final de cada iteración, qué cosas han funcionado bien, qué problemas existieron, qué cosas se pueden mejorar.

Para aplicar esta técnica en el presente proyecto se analizó con el equipo las tareas realizadas al final de cada sprint, donde se evaluaba la necesidad de incluir tareas para complementar el prototipo final y que este sea de mayor calidad, como se refleja en la Tabla 7.

Tabla 7*. Sprint retrospective.*

SPRINT	¿Qué cosas ha funcionado bien?	¿Qué es lo que se puede mejorar?
1	Integración de componentes a utilizar como Google Maps y Twitter.	El comportamiento de la página web haciéndolo responsive.
2	Creación de función cron para búsqueda de tweets.	Establecer un límite de registros a retornar para mostrarlos en un mapa de calor.
3	Envío de datos obtenidos por el cron hacia el web service.	Interfaces gráficas de la aplicación móvil.
4	Reporte de delitos desde la ubicación actual.	Manejo de errores generales para la aplicación móvil.

2.9. Despliegue de aplicaciones

Una vez finalizado el proceso de desarrollo se procedió a desplegar las aplicaciones distribuidas, como indica la Figura 19, es decir subir el prototipo de desarrollo a la web para que se pueda generar pruebas en el sistema. Los servidores se subieron en aplicaciones diferentes con lo que se puede evidenciar que se encuentran en servidores diferentes.

La plataforma que se usó para desplegar las aplicaciones es Heroku, un servicio de computación en la nube que soporta varios lenguajes de programación, es una plataforma como servicio (PaaS, platform as a service). Para acceder a cada una de las aplicaciones desplegadas, Heroku provee url y dominios con nombres aleatorios, los cuales pueden ser modificados. Dichos nombres de dominio contienen un segmento denominado herokuapp por defecto. Existe la posibilidad de cambiar por completo el dominio pero esta característica tiene un costo, se lo puede comprar en la misma plataforma de Heroku o en otro proveedor. Para este proyecto se conservó las urls provistas por defecto, **uioseguridad.herokuapp.com** y **ws-uio-seguridad.herokuapp.com**

3. Capítulo III: Pruebas

Para este capítulo se realizarán las pruebas de funcionalidad y de carga del sistema. Se comprobará que el cron realice la búsqueda de tweets cada cierto tiempo. Se analizará la posición de los puntos que se crearán en el mapa de calor en base a las direcciones que se encuentren en Twitter con el formato especificado.

3.1. Requisitos

Antes de iniciar con las pruebas se establecerán los requisitos necesarios para el buen funcionamiento de la página web y la aplicación móvil.

Requisitos para visualizar la página web:

Software

- Navegador Firefox, Safari, Chrome o Internet Explorer 7.0 o superior.

Requisitos para el buen uso de la aplicación móvil:

Software

- Sistema operativo Android 5.0 (Lollipop) o superior (Marshmallow)

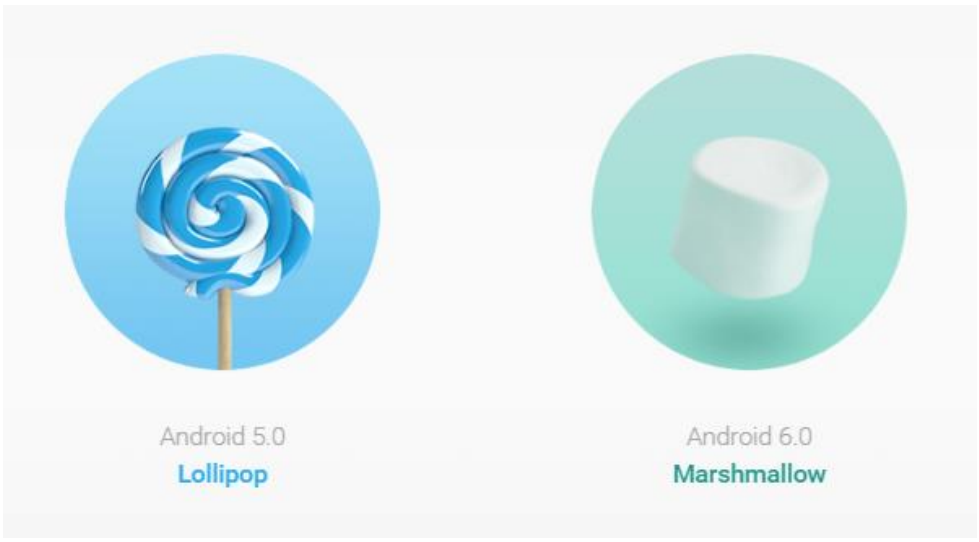


Figura 40. Sistemas Operativos Android Compatibles.

Tomado de (Android, s.f).

La Figura 40 muestra las imágenes de los sistemas operativos Android 5.0 (Lollipop) y Android 6.0 (Marshmallow) que son compatibles, hasta el momento, para la instalación de la aplicación desarrollada.

Version	Codename	API	Distribution
2.2	Froyo	8	0.1%
2.3.3 - 2.3.7	Gingerbread	10	2.0%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.9%
4.1.x	Jelly Bean	16	6.8%
4.2.x		17	9.4%
4.3		18	2.7%
4.4	KitKat	19	31.6%
5.0	Lollipop	21	15.4%
5.1		22	20.0%
6.0	Marshmallow	23	10.1%

Data collected during a 7-day period ending on June 6, 2016.
Any versions with less than 0.1% distribution are not shown.

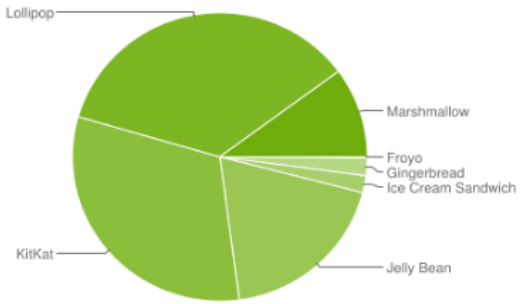


Figura 41. Estadísticas de uso de SO Android.

Tomado de (Android Developer, s.f).

Debido al uso de la plataforma Fabric para la integración de Twitter con Android, la cual permite únicamente su uso para versiones de Android desde 5.0 en adelante. Además como muestra la Figura 41 el sistema operativo Lollipop es el más usado dentro de las diferentes versiones de Android y junto al sistema operativo Marshmallow, que fue liberado en diciembre del 2015, ocupa el 45.5% de uso de todas los sistemas operativos Android. La información fue recolectada durante un período de 7 días que finalizó el 6 de junio del 2016.

En cuanto a la página web se comprobó su funcionalidad en computadoras en los navegadores Google Chrome, Safari, Mozilla Firefox, Internet Explorer. Además la página web se adapta a los navegadores de los diferentes dispositivos móviles.

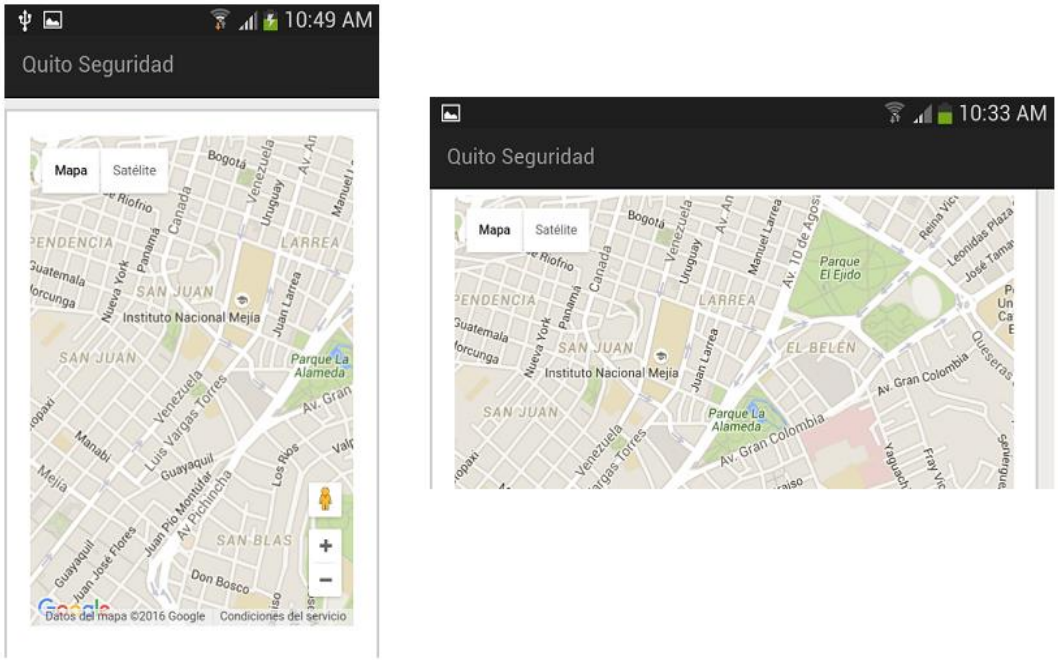


Figura 42. Página web en dispositivo Samsung Galaxy S3 Mini.

En la Figura 42 se muestra la página web en el dispositivo Samsung Galaxy S3 mini el cual tiene una pantalla de 4 pulgadas.

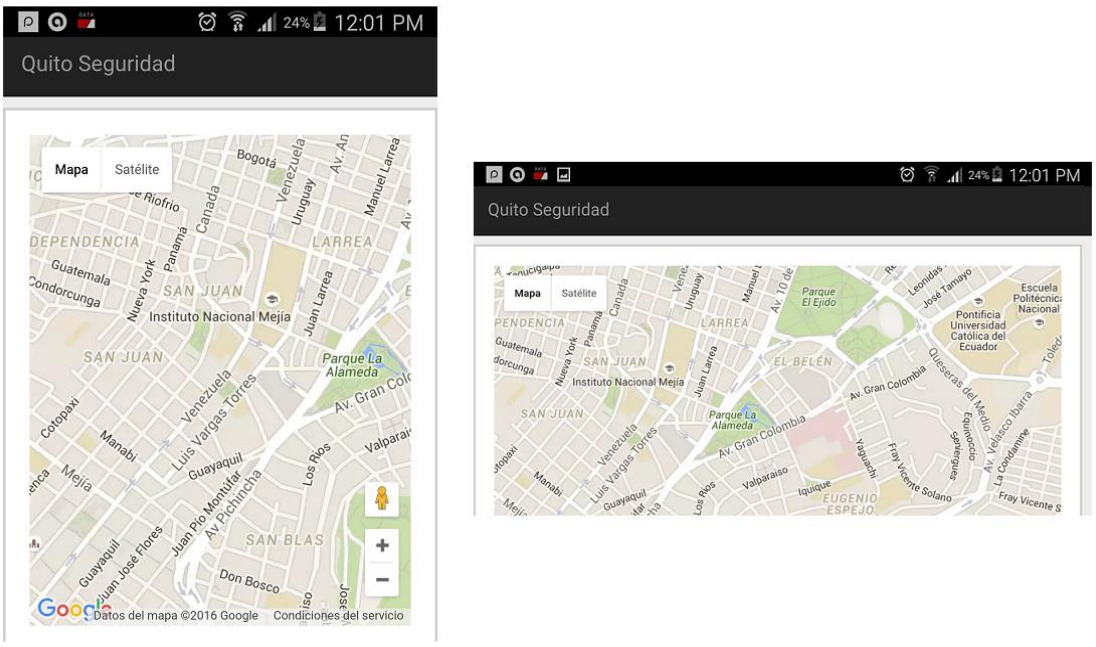


Figura 43. Página web en dispositivo Samsung Galaxy S4.

En la Figura 43 se muestra la página web en el dispositivo Samsung Galaxy S4 el cual tiene una pantalla de 5 pulgadas.

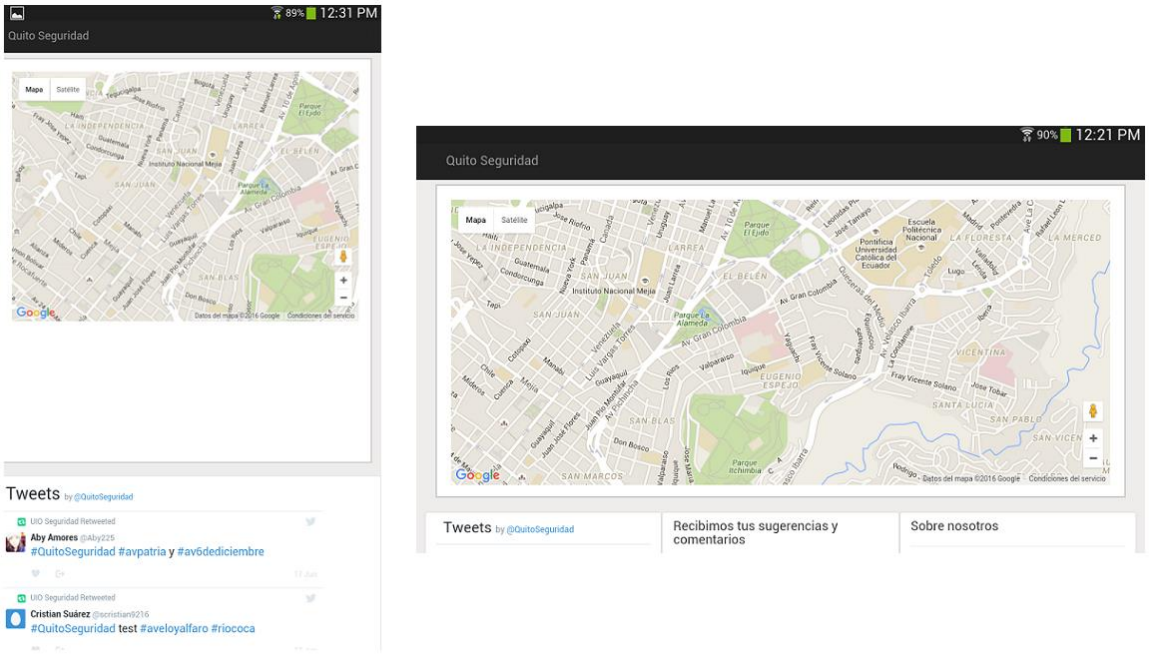


Figura 44. Página web en dispositivo Samsung Galaxy Tab 3 Lite.

En la Figura 44 se muestra la página web en el dispositivo Samsung Galaxy Tab 3 Lite el cual tiene una pantalla de 7 pulgadas.

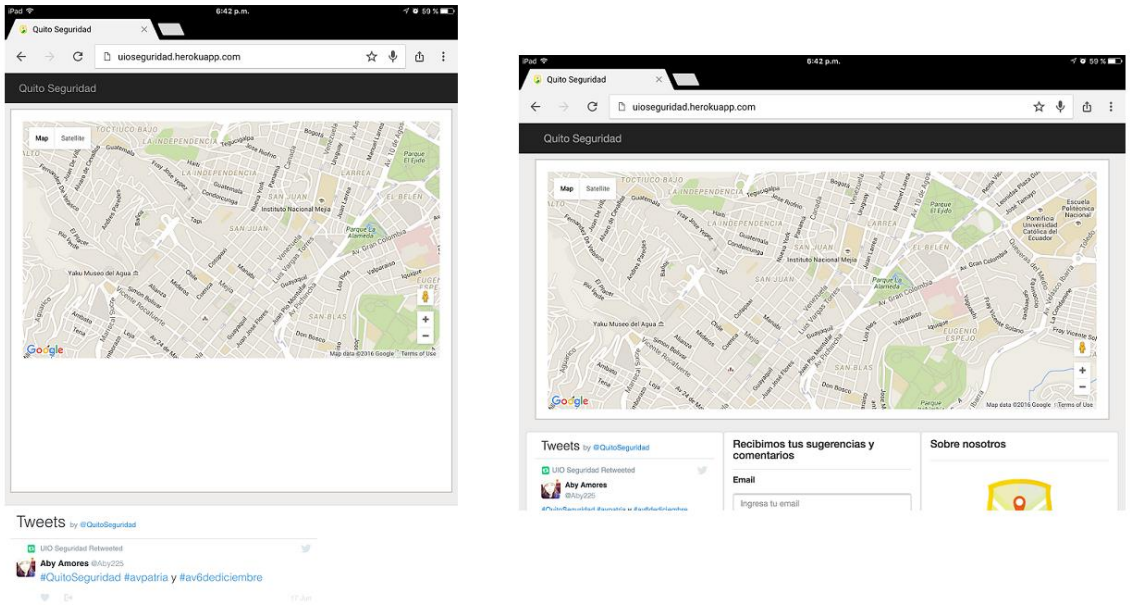


Figura 45. Página web en dispositivo Ipad 4ta Generación.

En la Figura 45 se muestra la página web en el dispositivo Ipad 4ta Generación, el cual tiene una pantalla de 9.7 pulgadas.

3.2. Pruebas de funcionalidad

Las pruebas de funcionalidad se realizaron en la Universidad de las Américas - UDLA con la colaboración de los estudiantes de la carrera de Ingeniería en Sistemas de Computación e Informática presentes en los paralelos 1 y 2 de la materia Introducción a los Sistemas de Computación - ACI110, junto con el docente Eddy Mauricio Armas Pallasco.

La primera característica en probar fue el uso de la red social Twitter para reportar delitos.

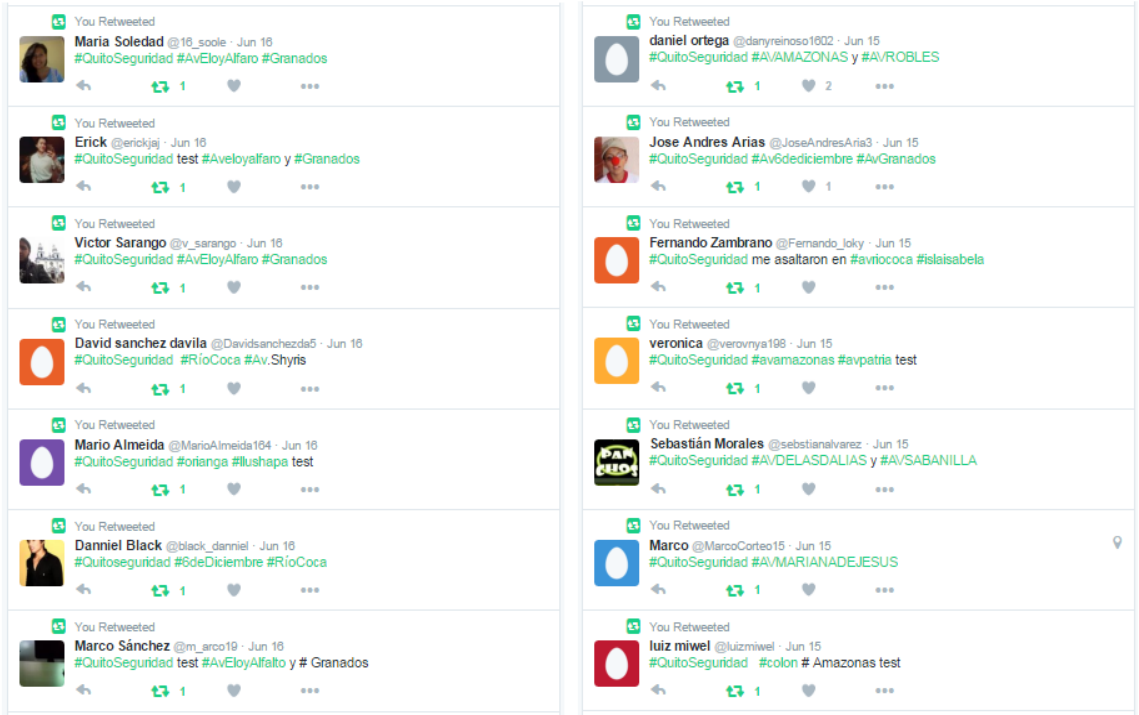


Figura 46. Lista de tweets.

En la Figura 46 se muestra una parte de los tweets encontrados y que fueron retwitteados por el sistema, los cuales marcaron puntos en el mapa.

Las pruebas con los estudiantes se realizaron durante 1 hora con 40 minutos aproximadamente por cada curso.

La función encargada de buscar tweets con el #QuitoSeguridad y retwittear dichos tweets funcionó en su totalidad como se describe en la Tabla 7.

Tabla 8. *Registro de Tweets.*

Cantidad de tweets enviados	Cantidad de tweets encontrados	Cantidad de tweets retwitteados
20	20	20

La Tabla 8 muestra que un 100% de los tweets fueron encontrados y posteriormente 100% de los tweets fueron retweeteados.

Durante el desarrollo de las pruebas se pudo apreciar que únicamente las direcciones redactadas en el formato específico mostrarán las coordenadas correctas. El mal uso de los hashtags (#), así como los errores ortográficos o el uso de caracteres especiales afectará las coordenadas resultantes.

Figura 47. Ejemplo de mal uso de hashtags.

Como muestra la Figura 47 los errores de escritura al momento de enviar un tweet pueden suceder, los cuales resultarán en coordenadas incorrectas.

Para comprobar la funcionalidad de geocodificación se realizó una comparación de las direcciones obtenidas a partir de los tweets y las coordenadas encontradas en Google Maps.

De los 20 tweets se tomó una muestra de los 10 tweets más significativos que cumplen con el formato especificado ya que tweets con errores en el formato produjeron errores representativos en las coordenadas obtenidas.

Tabla 9*. Margen de error de coordenadas obtenidas.*

	Dirección Obtenida	Punto X obtenido	Punto Y obtenido	Punto X Google Maps	Punto Y Google Maps	Diferencia (error)
1	AVRIOCOCA y ISLAISABELA	-0,16215	-78,4823	-0.162109	-78.482403	(0.000041, 0.000103)
2	AVAMAZONAS y AVPATRIA	-0,17155	-78,4847	-0.2077368	-78.497755	(0.03618, 0.013)

3	AVMARIANADEJESUS	-0,18729	-78,49347	-0.1888014	-78.493446	(0.00151, 0.00003)
4	AVDIEGODEVASQUEZ y GUALAQUIZA	-0,12258	-78,49163	-0.1186922	-78.493521	(0.0039, 0.0019)
5	AVDELOSALGARROBOS y JUANVACA	-0,1434699	-78,48095	-0.1428148	-78.483293	(0.0006, 0.0023)
6	AVELOYALFARO y GRANADOS	-0,16605	-78,46847	-0.1660628	-78,46847	(0.00001,0)
7	AVSHIRIS y REPUBLICADELSALVADOR	-0,1816599	-78,48022	-0.1856029	-78.484886	(0.0039, 0.0046)
8	6DEDICIEMBRE y GASPARDEVILLAROEL	-0,17834	-78,47775	-0.1709742	-78.478531	(0.00737, 0.00083)
9	REINAVICTORIA y COLON	-0,20308	-78,49136	-0.201091	-78.488744	(0.0019, 0.00262)
10	AVIGIRAS y GUAYACANES	-0,13734	-78,46692	-0.1373883	-78.469122	(0.00004, 0.0022)
					Promedio de error	**(0.0055451, 0.0027583)**

La Tabla 9 evidencia el margen de error resultante comparando las coordenadas obtenidas por el sistema y las coordenadas buscadas en Google Maps. Se tomaron diez direcciones redactadas en los tweets durante las pruebas y se observa que el margen de error es de 0.0055451 para la coordenada X y 0.0027583 para la coordenada Y.

Para la aplicación móvil se probó la característica de reportar un delito en base a la ubicación actual por medio del GPS de cada dispositivo. Para la instalación de la aplicación se utilizó dispositivos en las plataformas compatibles.

Figura 48. Reportar delito desde ubicación actual.

La Figura 48 muestra los puntos registrados que fueron reportados en la ubicación donde se efectuaron las pruebas por medio del sistema GPS.

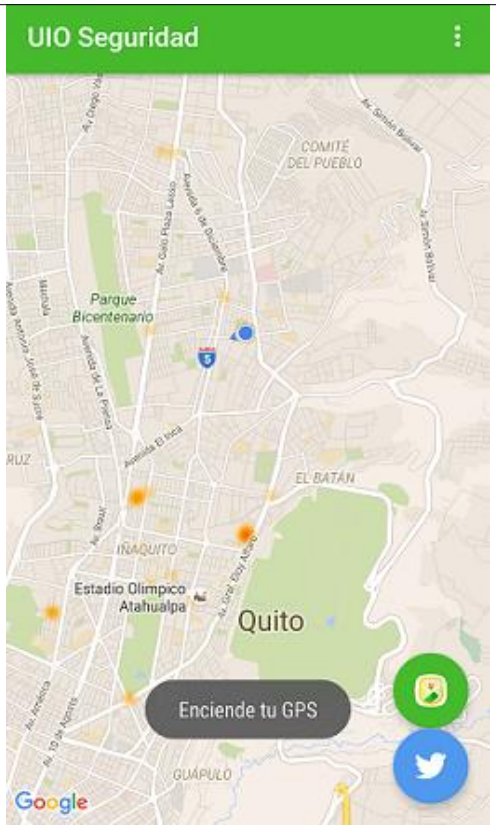


Figura 49. Mensaje de error GPS.

La Figura 49 muestra un mensaje de control si el GPS se encuentra apagado.

Se comprobó que la integración con Twitter funciona correctamente y no es necesario usar una segunda aplicación o un navegador para redactar tweets y tener la información de la línea de tiempo de la cuenta @QuitoSeguridad.

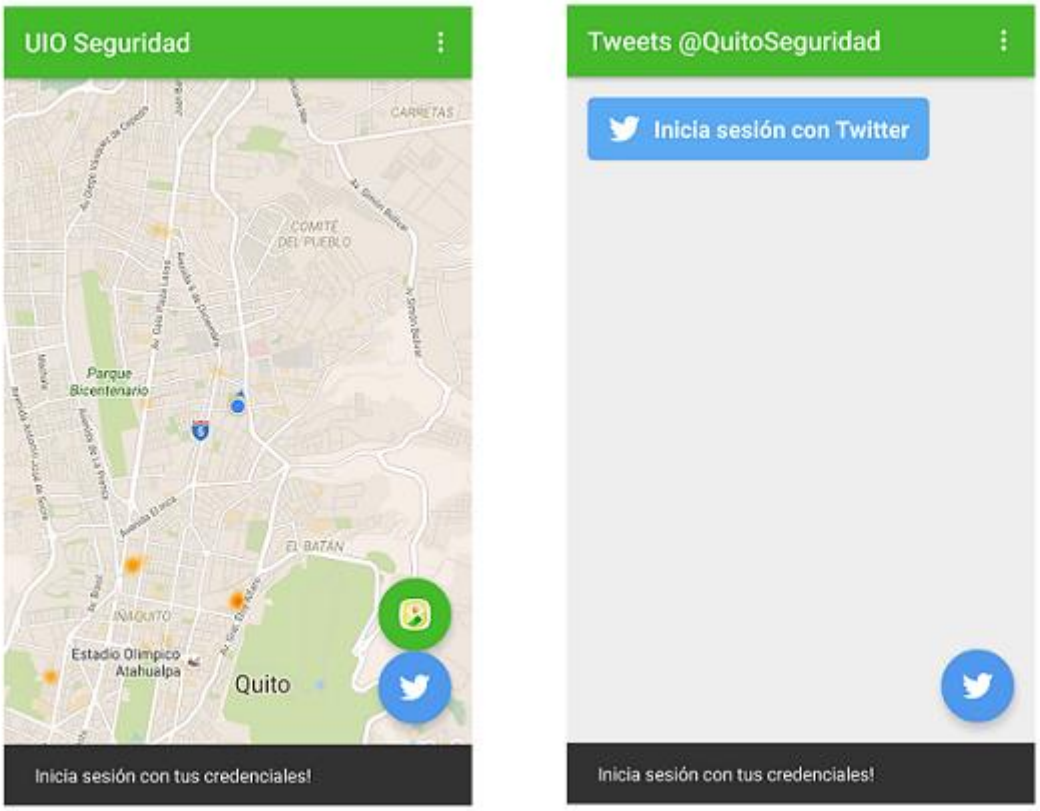


Figura 50. Mensaje de inicio de sesión.

La Figura 50 muestra el control de inicio de sesión con la cuenta personal de Twitter de cada usuario.

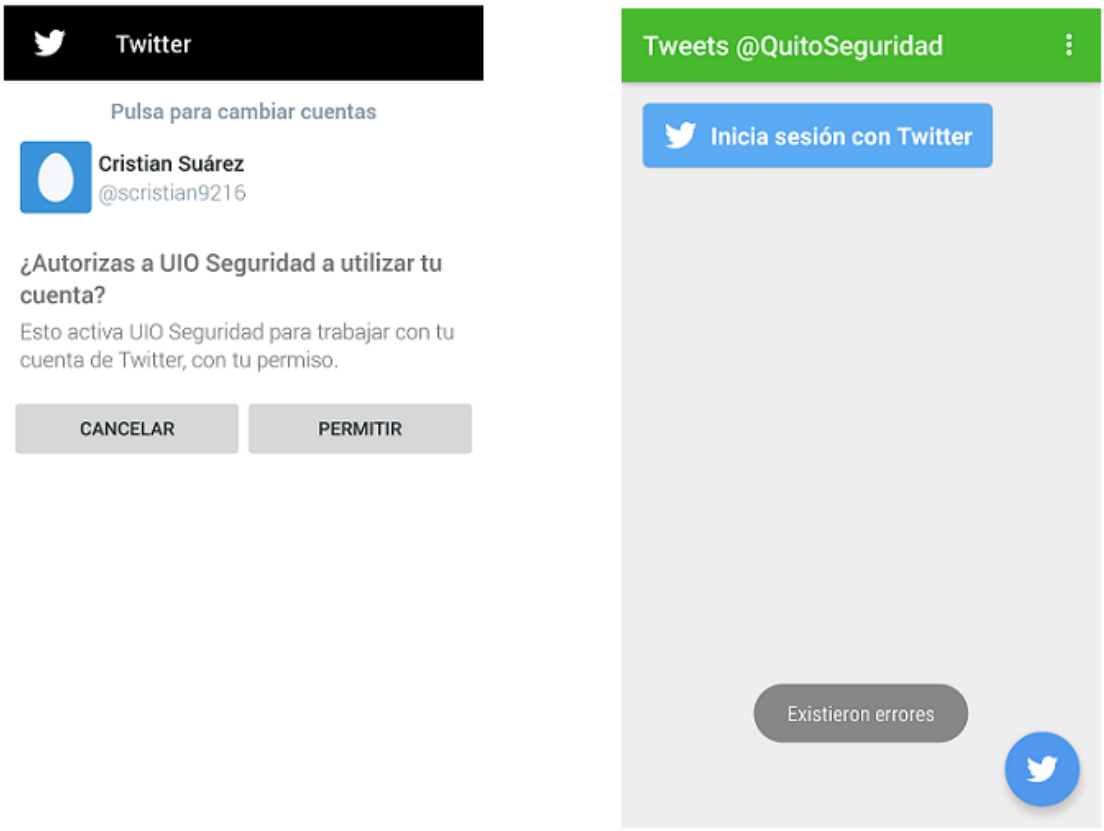


Figura 51. Mensaje de solicitud de permisos.

La Figura 51 muestra el control de inicio de sesión con Twitter si no se dan lo permisos necesarios para el uso de la aplicación.

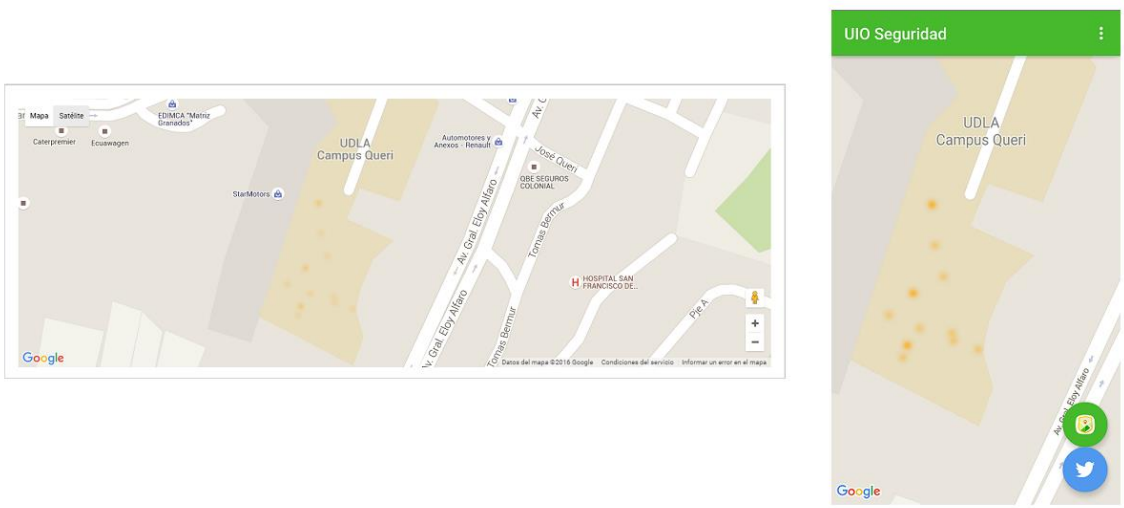


Figura 52. Comparación de puntos.

En la Figura 52 se evidencia que los puntos de calor que se encuentran en la página web, son los mismos en la aplicación móvil, de esta forma se mantiene la consistencia de información para la página web y la aplicación móvil.

3.3. Pruebas de Carga

El objetivo de realizar esta clase de pruebas es conocer la capacidad máxima de un servidor web para atender un conjunto de peticiones simultáneas simulando una carga de trabajo extrema, superior a la que podría recibir en un entorno real.

Existen varias herramientas disponibles para realizar esta clase de simulaciones, las cuales proveen algunos datos de interés como estadísticas de respuesta a las peticiones. Entre estas herramientas se ha escogido a las más simples en cuanto a su uso y más potentes en cuanto a desempeño, ApacheBench y Siege. Estas herramientas son bastante similares entre sí, el objetivo de usarlas es comparar los resultados para asegurar que estos se asemejen lo más posible a la realidad.

ApacheBench es una herramienta potente y flexible para generar una gran cantidad de peticiones HTTP y además brinda informes sobre los resultados, una versión estándar está incluida en la mayoría de ordenadores Mac y Linux. Una de sus principales ventajas es la facilidad y sencillez de utilizarla.

Siege es una herramienta similar a ApacheBench, una ventaja que posee esta herramienta es la aplicación de tiempos aleatorios entre las peticiones y así simular cargas de tráfico del mundo real, aunque una desventaja es la generación de reportes menos robustos.

Pruebas de carga usando la herramienta ApacheBench

A continuación se muestran simulaciones de usuarios y peticiones realizadas con la herramienta Apache Bench.
Se simuló 1000 peticiones y 100 conexiones concurrentes y otro escenario con 10000 peticiones y 200 conexiones concurrentes.

```
cristian@cristian-VirtualBox:~$ ab -n 1000 -c 100 -g grafica3.data -s 520 -r http://uioseguridad.herokuapp.com/
This is ApacheBench, Version 2.3 <$Revision: 1528965 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking uioseguridad.herokuapp.com (be patient)
Completed 100 requests
Completed 200 requests
Completed 300 requests
Completed 400 requests
Completed 500 requests
Completed 600 requests
Completed 700 requests
Completed 800 requests
Completed 900 requests
Completed 1000 requests
Finished 1000 requests


Server Software:        gunicorn/19.4.5
Server Hostname:        uioseguridad.herokuapp.com
Server Port:            80

Document Path:          /
Document Length:        17868 bytes

Concurrency Level:      100
Time taken for tests:   64.311 seconds
Complete requests:      1000
Failed requests:        0
Total transferred:      18188000 bytes
HTML transferred:       17868000 bytes
Requests per second:    15.55 [#/sec] (mean)
Time per request:       6431.077 [ms] (mean)
Time per request:       64.311 [ms] (mean, across all concurrent requests)
Transfer rate:          276.19 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:       95  100  32.1     98    1093
Processing:   375 5932 1264.3   5825    8786
Waiting:      277 5823 1265.7   5717    8689
Total:        471 6033 1264.3   5924    8883

Percentage of the requests served within a certain time (ms)
  50%   5924
  66%   6342
  75%   6613
  80%   6739
  90%   7560
  95%   8456
  98%   8725
  99%   8785
 100%   8883 (longest request)
```

Figura 53. Prueba 1 - Apache Bench.

La primera prueba realizada con la herramienta Apache Bench muestra que se completaron las 1000 solicitudes, es decir hay un 100% de efectividad y disponibilidad del servidor como muestra la Figura 53.

```
cristian@cristian-VirtualBox:~$ ab -n 10000 -c 200 http://uiosegurdad.herokuapp.com/
This is ApacheBench, Version 2.3 <$Revision: 1528965 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking uiosegurdad.herokuapp.com (be patient)
Completed 1000 requests
Completed 2000 requests
Completed 3000 requests
Completed 4000 requests
Completed 5000 requests
Completed 6000 requests
Completed 7000 requests
Completed 8000 requests
Completed 9000 requests
Completed 10000 requests
Finished 10000 requests


Server Software:        Cowboy
Server Hostname:        uiosegurdad.herokuapp.com
Server Port:            80

Document Path:          /
Document Length:        2960 bytes

Concurrency Level:      200
Time taken for tests:   23.384 seconds
Complete requests:      10000
Failed requests:        0
Non-2xx responses:      10000
Total transferred:      31360000 bytes
HTML transferred:       29600000 bytes
Requests per second:    427.64 [#/sec] (mean)
Time per request:       467.685 [ms] (mean)
Time per request:       2.338 [ms] (mean, across all concurrent requests)
Transfer rate:          1309.64 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:      100  221 221.8    165    2393
Processing:   104  239 260.1    170    3786
Waiting:      104  233 243.2    169    2887
Total:        221  460 427.7    319    4832

Percentage of the requests served within a certain time (ms)
  50%    319
  66%    348
  75%    382
  80%    414
  90%    508
  95%   1683
  98%   2045
  99%   2201
 100%   4832 (longest request)
```

Figura 54. Prueba 2 - Apache Bench.

La segunda prueba realizada con la herramienta Apache Bench muestra que se completaron las 10000 solicitudes, es decir hay un 100% de efectividad y disponibilidad del servidor como muestra la Figura 54.

Con los datos obtenidos se puede evidenciar que el aplicativo tiene una disponibilidad del 100%, es decir, responderá satisfactoriamente a todas las solicitudes de los usuarios en un entorno real.

Pruebas de carga usando la herramienta Siege

Para las pruebas realizadas con la herramienta Siege se establecieron dos escenarios, usando 100 conexiones concurrentes y un total de 1000 peticiones y las mismas 100 conexiones concurrentes con un total 10000 peticiones.

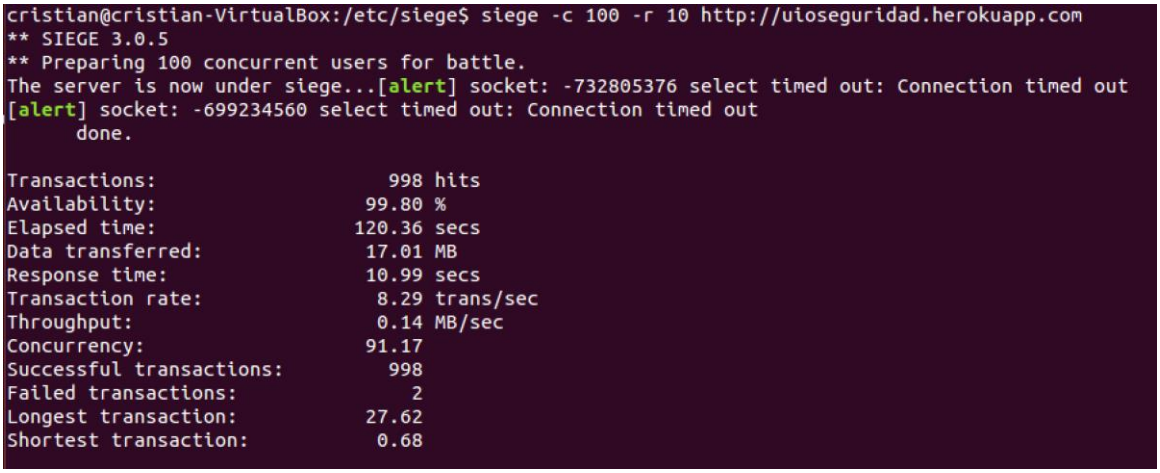


Figura 55. Prueba 1 - Herramienta Siege.

Con la primera prueba realizada se observa que 998 transacciones fueron satisfactorias y 2 transacciones perdidas, es decir una disponibilidad del 99,80%, evidenciado en la Figura 55.

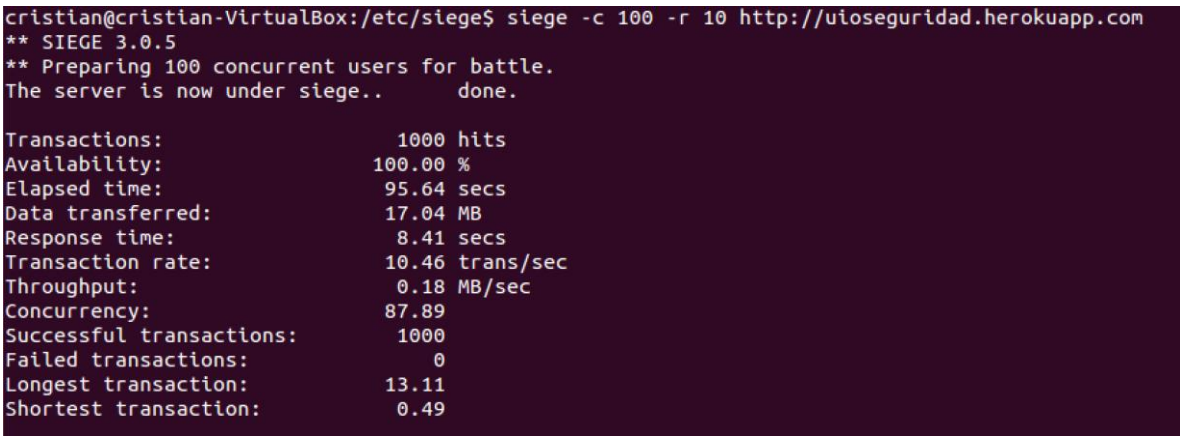


Figura 56. Prueba 2 - Herramienta Siege.

Con la segunda prueba realizada se observa que 1000 transacciones son satisfactorias y ninguna transacción perdida, es decir una disponibilidad del 100%, como muestra la Figura 56.

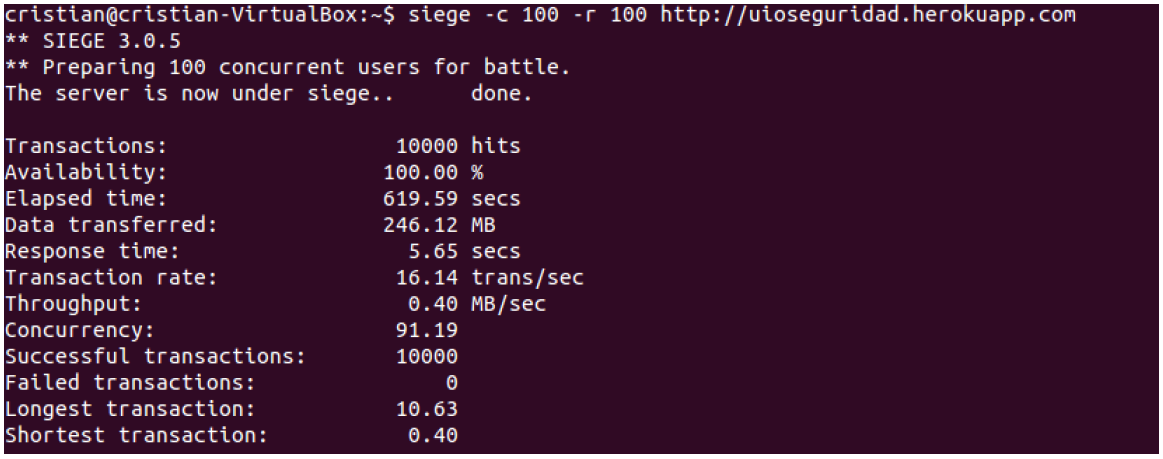


Figura 57. Prueba 3 - Herramienta Siege.

Con la tercera prueba realizada se observa que 10000 transacciones son satisfactorias y ninguna transacción perdida, es decir una disponibilidad del 100%, como muestra la Figura 57.

3.5. Resultados

A partir de las pruebas de funcionalidad y carga realizadas se obtuvo que:

- El sistema tiene un 100% de efectividad para retweetear las publicaciones de los usuarios que tienen el formato requerido.

- Las coordenadas obtenidas por el sistema dependerá del formato especificado y de la dirección escrita por los usuarios, ya que se observó que en algunos casos se encuentran tweets sin el formato necesario, direcciones ambiguas o mal escritas. El uso de caracteres especiales también puede ser causante de coordenadas incorrectas.

- El margen de error obtenido de las coordenadas por el sistema se puede considerar bajo, aunque al observar las coordenadas en el mapa es más notable esta diferencia.

- Los delitos reportados desde la aplicación móvil por medio de la posición actual fueron guardados en todos los casos y marcados en el mapa de calor en la ubicación mostrada por el GPS.

- La integración con Twitter en la aplicación móvil para redactar tweets y observar el timeline de la cuenta @QuitoSeguridad son efectivas y ayudan a optimizar el tiempo al momento de redactar tweets sin hacer uso de una tercera aplicación.

- Los errores que se presenten dentro de las funcionalidades están controlados para que se presenten mensajes y sirvan como guías para el usuario.

- En base a la comparación realizada entre las herramientas Apache Bench y Siege se obtiene como resultado un 100% de disponibilidad poniendo a prueba 1000 solicitudes con 100 conexiones concurrentes.

4. Capítulo IV: Conclusiones y Recomendaciones

4.1. Conclusiones

Por medio del análisis realizado entre metodologías tradicionales y ágiles se concluye que la metodología más adecuada para el desarrollo del presente proyecto es la metodología ágil SCRUM, debido a su adaptabilidad al cambio y ya que toma varias características de otras metodologías para adaptarlas a las necesidades del proyecto. Una de las ventajas más importantes de esta metodología es la revisión de los avances al final de cada sprint, en esta revisión se identifican cambios y nuevos requerimientos los cuales se pueden incluir en la siguiente iteración de desarrollo.

La herramienta de desarrollo web, Django, fue de gran utilidad al momento de realizar la administración de las tablas del modelo de la base de datos, ya que con solo pocos comandos se obtuvieron todas las funcionalidades CRUD.

Según las pruebas realizadas se concluyó que las direcciones obtenidas por el proceso de geocodificación son más exactas cuando los nombres de las calles son escritas correctamente y en su totalidad, ya que los algoritmos de geocodificación dependen de la semántica de la dirección, esto es dependiente de la API de geocodificación que se esté utilizando.

El correcto funcionamiento de este prototipo depende, en su mayoría, del buen uso de los usuarios, como por ejemplo, se encontraron casos durante las pruebas donde los tweets no cumplían con el formato especificado o direcciones mal escritas lo cual condujo a direcciones erradas.

En cuanto al desarrollo de la aplicación móvil, se observaron cambios significativos en el nuevo sistema operativo Android 6.0 relativos a la seguridad y permisos de acceso de aplicaciones, por lo que se aplicaron controles extra y se actualizó la funcionalidad referente a permisos para que este prototipo sea compatible con el nuevo sistema operativo.

4.2. Recomendaciones

La tendencia actual muestra un mayor uso de metodologías ágiles por lo cual es necesario hacer un análisis para elegir la metodología que sea más adecuada al proyecto que se desea desarrollar.

Existen varias herramientas para el desarrollo de aplicaciones móviles y sitios web, se recomienda examinar los requerimientos del sistema y la forma de desarrollo. Se debe analizar todas las características de las posibles herramientas y escoger las cuales se ajusten mejor a las propiedades del proyecto.

Es posible utilizar herramientas gratuitas o pagadas, dependiendo de los recursos asignados al proyecto y de los requisitos planteados por el cliente.

Se debe utilizar versiones estables de cada herramienta para mantener un código limpio y actualizado.

Es recomendable utilizar una herramienta de control de versionamiento para un desarrollo organizado y en especial cuando el equipo de desarrollo está conformado por varias personas.

Se recomienda desarrollar el sistema para plataforma iOS para tener un mayor alcance para los usuarios.

Referencias

Academia. (s.f). Comparativa Metodologías Ágiles. Recuperado 4 de Abril del 2016 de https://www.academia.edu/8058844/2.6ComparativaMetodologias_Agiles

Andes. (s.f). Estadísticas oficiales muestran reducción de delitos en Quito. *Recuperado el 13 de Enero del 2016 de http://www.andes.info.ec/es/noticias/estadisticas-oficiales-muestran-reduccion-delitos-quito.html*

Academia Android. (s.f.). *Android Studio v1.0: características y comparativa con Eclipse – Academia Android*. (2014). Recuperado 2 de Abril del 2016 de http://academiaandroid.com/android-studio-v1-caracteristicas-comparativa-eclipse/

Android. (s.f). *Android - History*. Recuperado 20 de Junio 2016 de https://www.android.com/history/#/marshmallow

Android Developers. (s.f.). Android Studio Overview. Recuperado el 9 de Febrero del 2016 de http://developer.android.com/intl/es/tools/studio/index.html

Android Developers. (2016). *Dashboards*. Recuperado 21 de Junio del 2016 de https://developer.android.com/about/dashboards/index.html

Arrs, A. (2014). *Flask: minimalismo para el desarrollo web en Python*. *Hipertextual*. Recuperado 2 de Abril del 2016 de http://hipertextual.com/archivo/2014/08/flask-python/

Bowes, J. (2015). *Kanban vs Scrum vs XP – an Agile comparison*. *Manifesto*. Recuperado el 26 de Enero del 2016 de https://manifesto.co.uk/kanban-vs-scrum-vs-xp-an-agile-comparison/

Bravo, D. (2014). Robos en Quito en la vía pública. *Recuperado el 15 de Enero del 2016 de http://www.elcomercio.com/actualidad/robos-quito-viapublica-delitos-estadisticas.html*

Brito Acuña, K. (2009). *Selección de metodologías de desarrollo para aplicaciones web en la facultad de informática de la universidad de Cienfuegos*. Cienfuegos, Cuba.

Dhingra, S. (2013). *REST vs. SOAP: How to choose the best Web service.* Recuperado el 8 Abril del 2016 de http://searchsoa.techtarget.com/tip/REST-vs-SOAP-How-to-choose-the-best-Web-service

Django overview | *Django* (s.f.). Recuperado el 2 de Febrero del 2016 de https://www.djangoproject.com/start/overview/

Ecuadorinmediato. (s.f). Índices delincuenciales. *Recuperado el 10 de Enero del 2016 de http://ecuadorinmediato.com/index.php?module=Noticias&func=news_user_view&id=194585&umt=indices_delincuenciales_bajaron_en_quito_segun_observatorio_metropolitano_seguridad*

El Comercio (s.f). Los asaltos y robos registrados el 2014. *Recuperado el 17 de Enero del 2016 de http://www.elcomercio.com/actualidad/ecuador-asaltos-robos-registrados-2014.html*

Esepestudio. (s.f.). *¿Qué es MySQL?*. Recuperado 2 de Abril del 2016, a partir de http://www.esepestudio.com/noticias/que-es-mysql

Fabric. (s.f). *Fabric Android documentation.* Recuperado el 4 de Mayo del 2016 de https://docs.fabric.io/android/index.html

Gilibets, L. (2013). *Qué es Kanban y cómo utilizarlo en el desarrollo de proyectos Blog de IEBSchool*. Recuperado el 27 de Enero del 2016 de http://comunidad.iebschool.com/iebs/general/metodologia-kanban/

Google Developers. (s.f.). *Google Maps JavaScript API*. Recuperado el 10 de Febrero del 2016 de https://developers.google.com/maps/documentation/javascript/?hl=es

Google Developers. *(s.f.) Heatmap Layer*. Recuperado 10 Febrero 2016 de https://developers.google.com/maps/documentation/javascript/heatmaplayer#overview

Google Developers. (s.f.). *Heatmap Layer*. Recuperado 10 Febrero 2016 de https://developers.google.com/maps/documentation/javascript/heatmaplayer#customize_a_heatmap_layer

Google Developers. (s.f.). *The Google Maps Geocoding API*. Recuperado el 11 de Febrero del 2016 de https://developers.google.com/maps/documentation/geocoding/intro#BYB

Holovaty, Adrian y Kaplan-Moss, Jacob. (2009). *The Definitive Guide to Django: Web Development Done Right.* New York, United States of America.

Inseguridad en quito. (s.f). *Inseguridad en Quito*. Recuperado el 6 de Enero del 2016 de https://inseguridadenquito.wordpress.com/estadisticas/

Isla visual. (s.f). Diferencias Entre Scrum Y Xp. *Recuperado el 25 de Enero del 2016 de http://www.islavisual.com/articulos/desarrollo_web/diferencias-entre-scrum-y-xp.php*

Java*. (s.f.) ¿Qué es Java y para qué es necesario?*. Recuperado 3 de Abril del 2016 de https://www.java.com/es/download/faq/whatis_java.xml

JetBrains. (s.f.). *PyCharm*. Recuperado el 6 de Febrero del 2016 de https://www.jetbrains.com/pycharm/

La hora. (s.f). *Se identifican los 9 delitos con mayor incidencia en Ecuador*. Recuperado el 10 de Enero del 2016 de *http://lahora.com.ec/index.php/noticias/show/1101583434/-1/Se_identifican_los_9_delitos_con_mayor_incidencia_en_Ecuador.html#.Vu3Q3hIrKCS*

Levin, M. (2011). *Python Programming Language Advantages & Disadvantages – Mike Levin SEO Consultant NYC*. *Mike Levin SEO Consultant NYC*. Recuperado el 2 Febrero del 2016 de http://mikelev.in/2011/01/python-programming-language-advantages/

Macias, C. (2014). *¿Qué son los Frameworks?*. *Nubelo*. Recuperado el 2 de Abril del 2016 de http://www.nubelo.com/blog/que-son-los-frameworks/

Mahnic, V., & Drnovscek, S. (2006). *Agile Software Project Management with Scrum*. Recuperado a partir de http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.126.901&rep=rep1&type=pdf

MD Anderson Bioinformatics. (s.f.). *Department of Bioinformatics and Computational Biology*. Recuperado 3 de Abril 2016 de http://bioinformatics.mdanderson.org/main/File:NG-CHM-screenshot.png

Mendoza Vázquez, I. (2010). *Definición de un Framework para aplicaciones Web con navegación sensible a concerns* (1st ed.). República Argentina. Recuperado de http://sedici.unlp.edu.ar/bitstream/handle/10915/4192/Documento_completo.pdf?sequence=1

Ministerio de Turismo. (s.f). *Ecuador mejora sus índices de seguridad ciudadana.* Recuperado el 17 de Enero del 2016 de http://www.turismo.gob.ec/ecuador-mejorar-sus-indices-de-seguridad-ciudadana/

Ministerio del Interio.r (s.f). *Robos disminuyen hasta el 40%, en el Distrito Metropolitano de Quito*. Recuperado el 15 de Enero del 2016 de http://www.ministeriointerior.gob.ec/robos-disminuyen-hasta-el-40-en-el-distrito-metropolitano-de-quito/

Municipio del Distrito Metropolitano de Quito (s.f.). *Rendición de Cuentas 2014*. Recuperado el 2 de Abril de 2016 de http://www.quito.gob.ec/documents/rendicion_cuentas/Rendicion_de_cuentas_2014.pdf

Optimus Information Inc. (s.f). *Traditional vs Agile Software Development*. Recuperado el 20 de Enero del 2016 de http://www.optimusinfo.com/blog/traditional-vs-agile-software-development/

Oracle. (s.f.). *MySQL | The Most Popular Open-Source Database | Oracle*. Recuperado 2 de Abril del 2016 de http://www.oracle.com/us/products/mysql/overview/index.html

Ortega, J. (2016). *La Policía registra 20 robos en las calles de Quito, cada día.* Recuperado el 13 de Mayo del 2016 de http://www.elcomercio.com/actualidad/quito-robos-denuncias-policia.html

Penadés, M., Canós, J., & Letelier, P. (2003*). Metodologías Ágiles en el Desarrollo de Software.* Recuperado el 18 de Enero del 2016 de http://ima.udg.edu/Docencia/07-08/3105200728/TodoAgil.pdf

Perdue, T. (2000*). PHPBuilder - MySQL and PostgreSQL*. Recuperado el 11 de Mayo del 2016 de http://www.phpbuilder.com/columns/tim20000705.php3?page=5

Postgresql. (s.f.). *PostgreSQL*. Recuperado el 8 Febrero del 2016 de http://www.postgresql.org/about/

Postgresql (s.f.). *Sobre PostgreSQL*. (2016). Recuperado 9 de Febrero del 2016 de http://www.postgresql.org.es/sobre_postgresql

Proyectos Ágiles. (s.f). *Planificación de la iteración (Sprint Planning).* Recuperado el 16 Mayo del 2016 de https://proyectosagiles.org/planificacion-iteracion-sprint-planning/

Proyectos Ágiles. (s.f). *Retrospectiva (Sprint Retrospective).* Recuperado el 16 Abril del 2016 de https://proyectosagiles.org/retrospectiva-sprint-retrospective/

Python.org. (s.f.). Comparing Python to Other Languages. Recuperado el 4 de Febrero del 2016 de https://www.python.org/doc/essays/comparisons/

Rodríguez, A. (2015). *Aprende Flask – Introducción*. *Geeky Theory*. Recuperado 2 de Abril del 2016 de https://geekytheory.com/aprende-flask-introduccion-y-hola-mundo/

Rodriguez, A. (2008). *RESTful Web services: The basics*. Recuperado el 12 de Mayo del 2016 de http://www.ibm.com/developerworks/library/ws-restful/

Rouse, M. (2015). *¿Qué es MySQL?*. *SearchDataCenter en Español*. Recuperado 2 de Abril del 2016 de http://searchdatacenter.techtarget.com/es/definicion/MySQL

Roy Fielding (2000). *Fielding Dissertation: CHAPTER 5: Representational State Transfer (REST)*. *Ics.uci.edu*. Recuperado el 1 de Febrero del 2016 de http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm?cm_mc_uid=38763974456514590193998&cm_mc_sid_50200000=1459019399 "Architectural Styles and the Design of Network-based Software Architectures."

Steve Grossi. (s.f). *Load Testing Rails Apps with Apache Bench, Siege, and JMeter*. Recuperado 21 de Junio del 2016 de http://work.stevegrossi.com/2015/02/07/load-testing-rails-apps-with-apache-bench-siege-and-jmeter/

Vargas Benjumea, J. y Horfan Álvarez, D. (2013). *Proceso de geocodificación de direcciones en la ciudad de medellín, una técnica deterministica de georreferenciación de direcciones*. Medellín, Colombia.

WakaTime (s.f.). *Pirates use Flask, the Navy uses Django.* Recuperado el 5 deFebrero del 2016 de https://wakatime.com/blog/25-pirates-use-flask-the-navy-uses-django

WebMaxFormance. (s.f.). *Attention, Scroll and Click Heatmap Tracking Services*. Recuperado el 3 de Abril del 2016 de http://webmaxformance.com/attention-scroll-click-heatmap-tracking-services

ANEXOS

Anexo 1: Sprint 2

Se incluirán las historias de usuario DMQ-3 y DMQ-4, desarrollo de cron y desarrollo de web service.

Tabla 10. *Sprint 2.*

ID	Descripción	Criterio de aceptación
DMQS2-1	Integración Twython (twitter-python)	Utilizar un componente que permita la comunicación entre Twitter y código python.
DMQS2-2	Búsqueda y lectura de tweets.	Realizar búsqueda de todos los tweets referenciados a la cuenta @QuitoSeguridad.
DMQS2-3	Organizar información en arreglos.	Organizar la información en formato JSON.
DMQS2-4	Lectura de información almacenada en la base de datos.	Los registros almacenados en la base de datos retornan en formato JSON.
DMQS2-5	Almacenamiento de datos encontrados por el cron y guardados en la base de datos.	Aplicar geocodificación y almacenar en la base de datos.
DMQS2-6	Creación de páginas de error.	Crear páginas por defecto para errores HTTP 500, 404, 400.

Desarrollo del sprint:

DMQS2-1

En esta tarea se realizará la integración de Twitter con el proyecto, ya que el proyecto es desarrollado en python se utilizará un wrapper para la conexión con Twitter, denominado Twython. Un wrapper es un programa que controla el acceso a otro programa, un wrapper es un mecanismo de seguridad que cubre la identidad del segundo programa, en este caso Twitter, obteniendo de esta forma un nivel más alto de seguridad. Para la configuración se requiere generar claves o tokens de acceso, esto se obtiene a partir de registrar un proyecto para el desarrollo e integración con Twitter.

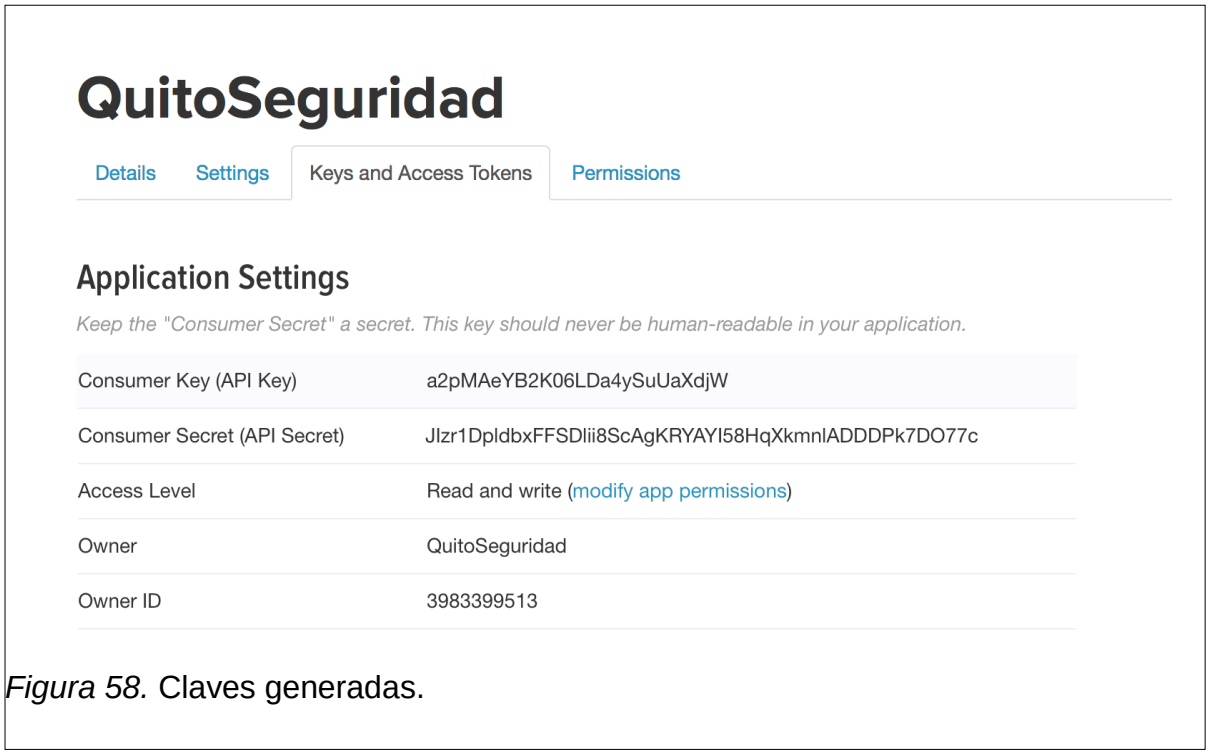


Figura 58. Claves generadas.

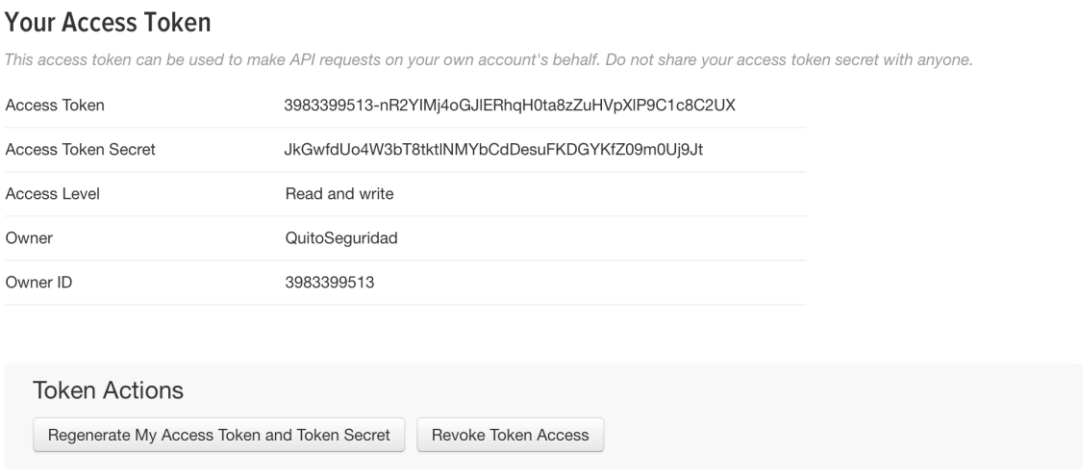


Figura 59. Token de conexión.

Para generar claves y tokens de acceso, se debe activar por medio de la cuenta de Twitter y el siguiente enlace: https://apps.twitter.com, en la Figura 58 se muestran las claves de la api, el nivel de acceso junto con el propietario y un id del mismo. Para Figura 59 se muestran los tokens para tener acceso a Twitter y a la cuenta que se quiere referenciar.

DMQS2-2

Para esta tarea se inicia el desarrollo del cron, es decir una función que se ejecuta en el servidor cada cierto tiempo, para la construcción de esta función primero se realizará la búsqueda y lectura de tweets. Para realizar estas operaciones se usarán funciones de Twython.

DMQS2-3

Para realizar la tarea DMQS2-3, se tomará la siguiente información: id del tweet, fecha de creación y hashtags que contenga el tweet, para organizar la información en formato JSON. Además se hará retweet de la información encontrada para tener un historial de información en el timeline de la cuenta @QuitoSeguridad.

```
def search_tweets():
    try:
        print '['+ctime()+'] Processing started'
        twitter_service = Twython(settings.TWITTER_CONSUMER_KEY,
                          settings.TWITTER_CONSUMER_SECRET,
                          settings.TWITTER_OAUTH_TOKEN,
                          settings.TWITTER_OAUTH_TOKEN_SECRET)

        search_word = get_search_word()
        number_of_tweets = get_number_of_tweets()
        last_tweet_date = get_last_tweet_date()
        results = twitter_service.search(q=search_word,count=number_of_tweets, since=last_tweet_date)
        contents = results.get('statuses')

        if contents:
            tweets = []
            for content in results['statuses']:
                try:
                    tweet_id = content['id_str']
                    twitter_service.retweet(id = tweet_id)
                    print 'Successfully retweeted'
                    tweet = create_tweet_from(content)
                    tweets.append(tweet)
                except Exception as e:
                    print 'This tweet has already been retweeted'
                    print e
            save(tweets)
        else:
            print 'No results found'
    except Exception as e:
        print e
    pass
```

Figura 60. Cron.

En la Figura 60 se muestra el código del cron, el cual usa los tokens generados para la aplicación para realizar la interacción con Twitter. También se usan funciones de twython para la búsqueda y retweet. Los resultados obtenidos se guardan en un archivo registro de eventos (log).

DMQS2-4

Obtener los registros almacenados en la base de datos, para esta tarea se construye el servicio web al cual se accederá a tomar la información para desplegar en el mapa, tanto el mapa de la aplicación móvil y el mapa del sitio web. La función que realiza el servicio web es retornar los registros ordenados descendentemente por fecha y con un límite determinado de registros. El número de registros a obtener es administrable.

Toda la información es retornada en formato JSON, para manejar un formato uniforme y así la aplicación web y móvil puedan consumir la información para desplegarla al usuario.

DMQS2-5

Almacenar los datos encontrados por el cron, se inicia con suprimir el criterio de búsqueda referenciado a la cuenta, los hashtags que se obtienen de la búsqueda de tweets son las direcciones a ubicar en el mapa, para esto se utiliza la geocodificación, es decir, transformar la dirección obtenidas a coordenadas de latitud y longitud para marcar los puntos en el mapa.

DMQS2-6

Para esta tarea se crearán las páginas por defecto para los errores HTTP 500, 404 y 400.

Figura 61. Página error 500.

La Figura 61 despliega el error HTTP 500, que describe un error interno del servidor.

Figura 62. Página error 404.

La Figura 62 despliega el error HTTP 404, que describe una página no encontrada.

Figura 63. Página error 400.

La Figura 63 despliega el error HTTP 400, que describe una solicitud incorrecta.

Anexo 2: Sprint 3

En el siguiente sprint se realizarán las tareas asignadas a las historias DMQ-5 y DMQ-6, que es la integración web y backend, y diseño de la aplicación móvil.

Tabla 11*. Sprint 3.*

ID	Descripción	Criterio de aceptación
DMQS3-1	Solicitud de puntos a web service.	Comprobar que la comunicación entre la página web y el servicio web sea exitoso.
DMQS3-2	Envío de datos desde cron a web service.	Obtener la información solicitada, comprobar que se lea correctamente.
DMQS3-3	Habilitar Interfaz de administración.	Crear página de administración y crear usuario administrador.
DMQS3-4	Desarrollo de interfaz gráfica de aplicación móvil (menú, toolbar, íconos, actividades).	Definir la arquitectura básica en la que se realizará la distribución de las pantallas en la aplicación móvil.

Desarrollo del sprint:

DMQS3-1

Para esta tarea se tiene que asegurar que los registros de las coordenadas almacenadas en la base de datos se lean correctamente en la página web, es decir, se debe asegurar la consistencia de datos que se despliegan en la página web con los datos almacenados, por medio de una solicitud HTTP.

DMQS3-2

Obtener la información recolectada por el cron, se debe consultar el parámetro con la URL que tiene definido el servicio web para construir la solicitud HTTP con la información a enviar, es decir, los datos recolectados por el cron. Una vez enviados los datos al servicio web, éste debe comprobar que se lean correctamente.

DMQS3-3

Habilitar interfaz de administración, registrar las tablas de base de datos de manera que resulte más sencillo la manipulación de información, además crear un usuario administrador.

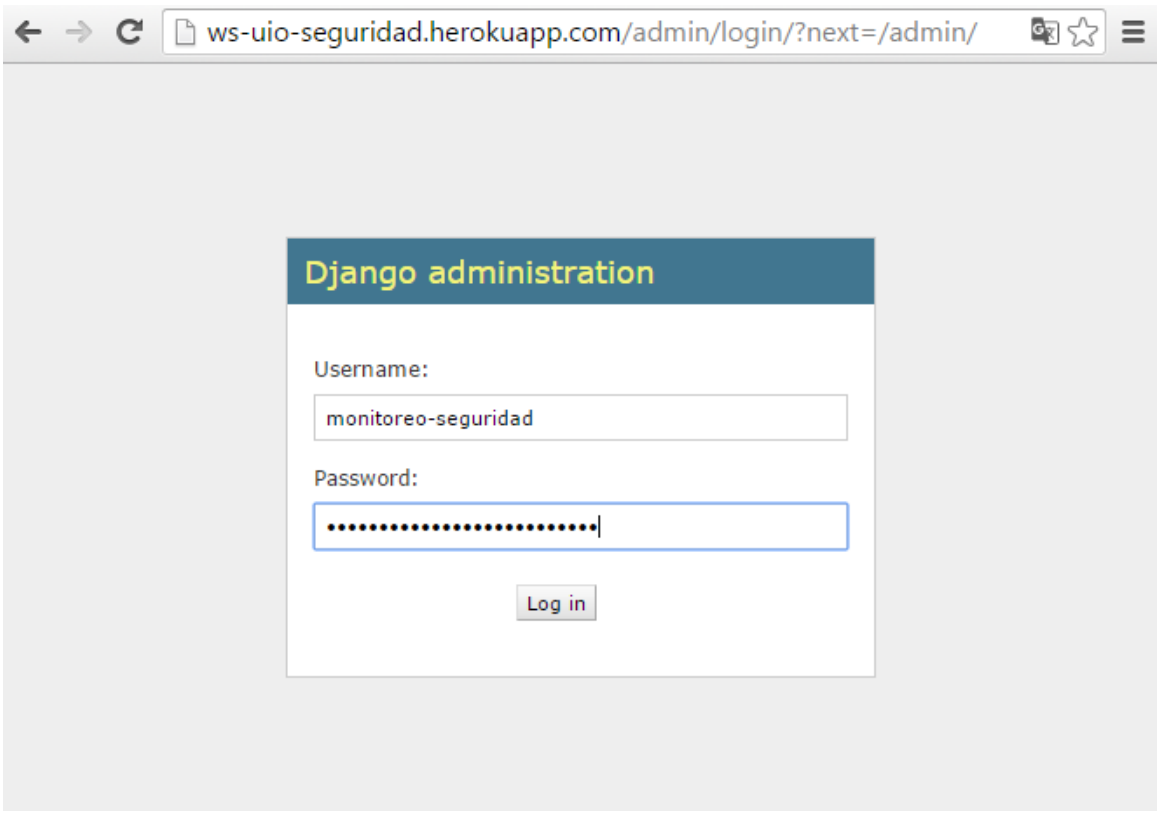


Figura 64. Login de Administración.

La Figura 64 muestra la pantalla de inicio de sesión para la administración del sistema.

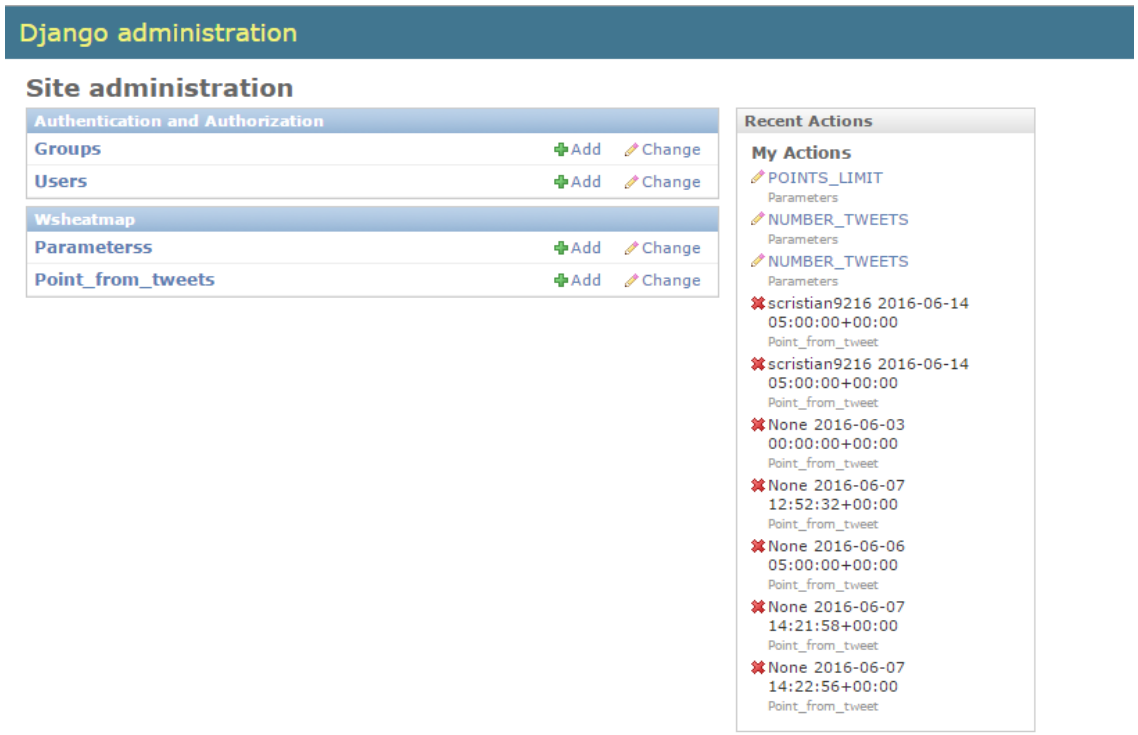


Figura 65. Interfaz de Administración.

La Figura 65 muestra la interfaz de administración.

DMQS3-4

Para la siguiente tarea se realizará los mockups para las pantallas de la aplicación móvil. Para las pantallas se definirá un título con letras blancas bajo un fondo verde, en esta sección se encontrará el menú por medio de un componente de toolbar.

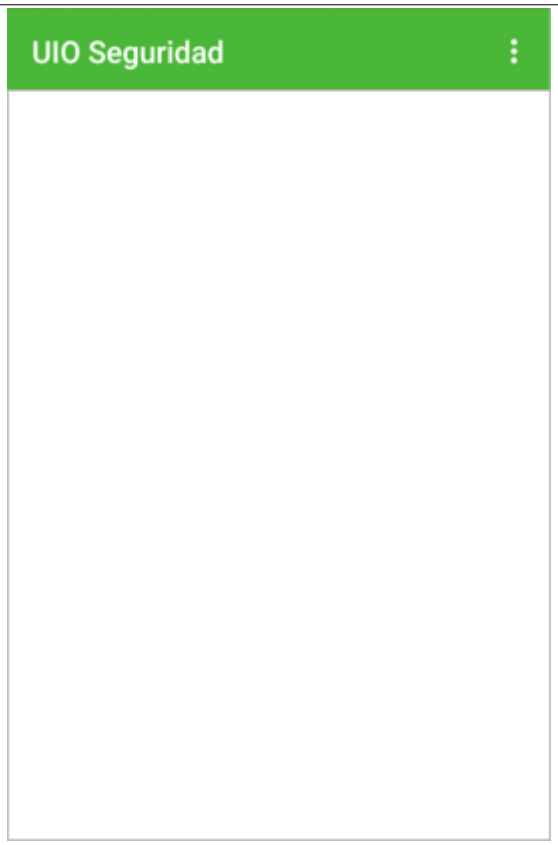


Figura 66. Pantalla Principal.

La pantalla principal o de inicio, mostrada en la Figura 66, es la pantalla donde se creará la integración con Google Maps y mostrará el mapa de calor. Tiene como título principal el nombre de la aplicación.

Figura 67. Página principal y menú (toolbar).

En la Figura 67 se muestra el menú con el que se navegará por la aplicación móvil el cual se compone de tres secciones: Mapa, Tweets y Sobre Nosotros, las cuales enviarán a la pantalla o actividad correspondiente.

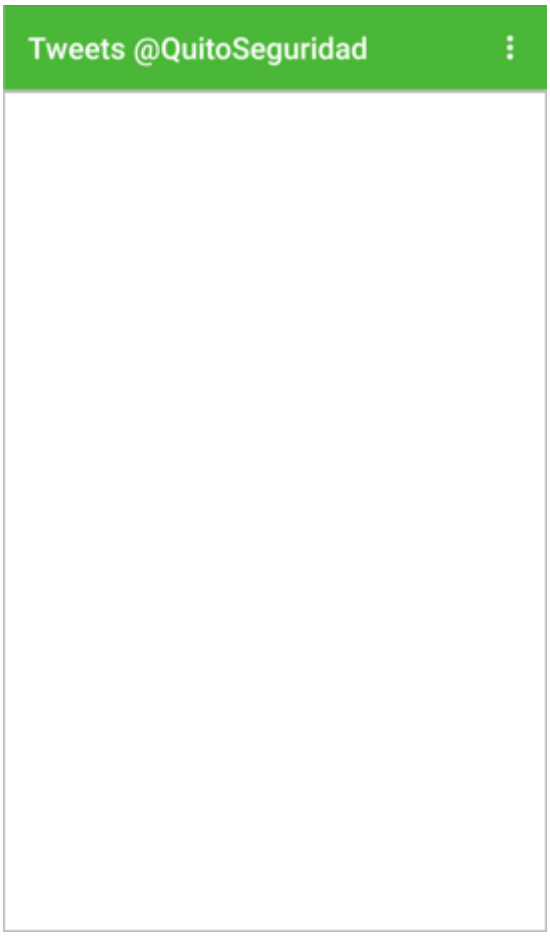


Figura 68. Página timeline de tweets.

En esta página, como muestra la Figura 68, se realizará la integración con Twitter, la cual mostrará el timeline de la cuenta @QuitoSeguridad.

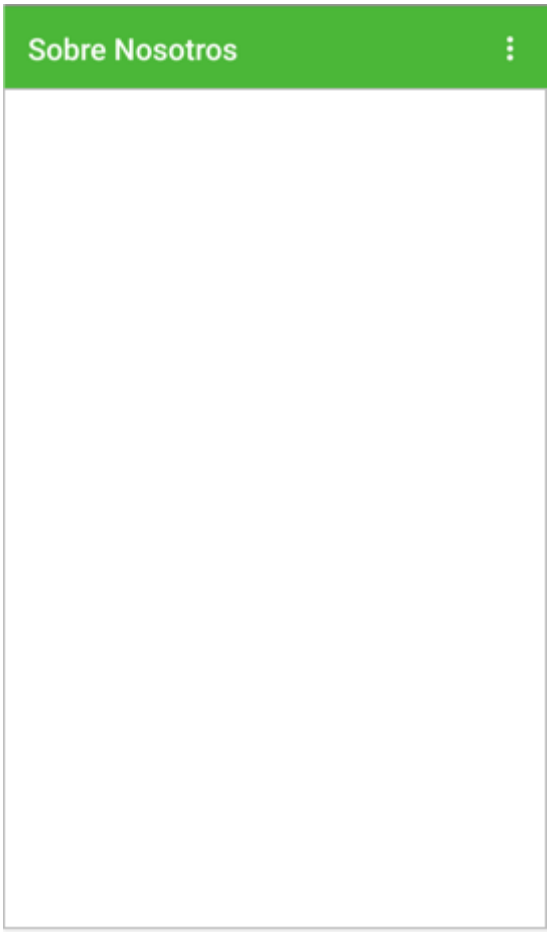


Figura 69. Página “Sobre Nosotros”

En la Figura 69, se aprecia la página que muestra el logo de la aplicación.

Anexo 3: Pruebas con alumnos

Figura 70. Pruebas Alumnos.

La Figura 68 muestra el día en el que se realizaron las pruebas de funcionalidad del sistema.