

Facultad de Ingeniería

Escuela de Ingeniería en Computación e Informática

# “Safe Travel Map App”

## Autor:

#### Fabio Ismael Urrea Quezada

## Profesor Guía:

#### Sarita González

#### Viña del Mar, Chile

#### 2023

Índice

[“Safe Travel Map App” 1](#_Toc139019767)

[Autor: 1](#_Toc139019768)

[Profesor Guía: 1](#_Toc139019769)

[CAPITULO I INTRODUCCIÓN 4](#_Toc139019770)

[1.1 INTRODUCCIÓN 5](#_Toc139019771)

[1.2 DEFINICIÓN DEL PROBLEMA 5](#_Toc139019772)

[1.2.1 DIAGRAMA DE ISHIKAWA 6](#_Toc139019773)

[1.2.2 LEVANTAMIENTO DE INFORMACIÓN (FOCUS GROUP) 7](#_Toc139019774)

[1.2.3 REQUISITOS DE ALTO NIVEL 8](#_Toc139019775)

[1.2.4 REQUISITOS DE ALTO NIVEL CLASIFICACIÓN 9](#_Toc139019776)

[1.3 OBJETIVOS GENERAL 10](#_Toc139019777)

[1.4 OBJETIVOS ESPECÍFICOS 10](#_Toc139019778)

[1.5 MÉTRICAS 11](#_Toc139019779)

[1.7 PLAN DE PROYECTO 11](#_Toc139019780)

[1.8 JUSTIFICACIÓN DEL PLAN DE PROYECTO 12](#_Toc139019781)

[1.9 ALCANCE DEL PROYECTO 13](#_Toc139019782)

[1.10 LIMITACIONES DEL PROYECTO 13](#_Toc139019783)

[CAPITULO II FUNDAMENTACIÓN DEL TEMA 14](#_Toc139019784)

[2.1 ALTERNATIVAS DE SOLUCIÓN 15](#_Toc139019785)

[2.2 FACTIBILIDAD TÉCNICA 15](#_Toc139019786)

[2.3 FACTIBILIDAD LEGAL 16](#_Toc139019787)

[2.4 FACTIBILIDAD ECONÓMICA 16](#_Toc139019788)

[2.5 ESTUDIO DE FACTIBILIDAD DEL PROYECTO 18](#_Toc139019789)

[2.6 DIAGRAMA DE ALTO NIVEL 19](#_Toc139019790)

[2.7 DOCUMENTACIÓN DE DISEÑO 21](#_Toc139019791)

[2.8 VISTA DE ESCENARIOS 22](#_Toc139019792)

[2.9 VISTA LÓGICA 23](#_Toc139019793)

[2.10 VISTA DE DESPLIEGUE 24](#_Toc139019794)

[2.11 DOCUMENTACIÓN DE ANÁLISIS 24](#_Toc139019795)

[CAPITULO III PLANES Y METODOLOGÍAS 26](#_Toc139019796)

[3.1 METODOLOGÍA DE GESTIÓN 27](#_Toc139019797)

[3.2 METODOLOGÍA DE DESARROLLO 28](#_Toc139019798)

[3.3 PRODUCT BACKLOG 29](#_Toc139019799)

[3.3.1 HISTORIAS DE USUARIO 29](#_Toc139019800)

[3.3.2 PRIORIZACIÓN CON MOSCOW 30](#_Toc139019801)

[3.3.3 “PLANING POKER” ASIGNACIÓN DE PUNTOS DE ESFUERZO 31](#_Toc139019802)

[3.3.4 TABLA DE PRIORIZACIÓN 32](#_Toc139019803)

[3.4 DOCUMENTACIÓN DE SPRINTS 34](#_Toc139019804)

[3.4.1 Sprint uno 34](#_Toc139019805)

[3.4.2 Sprint dos 55](#_Toc139019806)

[3.5 PLAN DE CALIDAD 67](#_Toc139019807)

[3.6.1 Costos sprint uno 70](#_Toc139019808)

[3.6.1 Costos sprint dos 71](#_Toc139019809)

[3.7 PLAN DE PRUEBAS 72](#_Toc139019810)

[3.8 PLAN DE GESTIÓN DE LA CONFIGURACIÓN 73](#_Toc139019811)

[3.9 PLAN DE GESTIÓN DE CAMBIOS 74](#_Toc139019812)

[3.10 PLAN DE GESTIÓN DE RIESGOS 77](#_Toc139019813)

[3.11 AMBIENTE DE DESARROLLO, PRUEBAS Y PRUDUCCIÓN 78](#_Toc139019814)

[3.12 DESARROLLO DE LA APLICACIÓN MÓVIL 78](#_Toc139019815)

[3.12.1 Sprint uno 78](#_Toc139019816)

[3.12.2 Sprint dos 90](#_Toc139019817)

[3.13 POSTMORTEM 117](#_Toc139019818)

[3.13.1 Sprint uno 118](#_Toc139019819)

[3.12.2 Sprint dos 119](#_Toc139019820)

[3.14 PROBLEMAS ABIERTOS 119](#_Toc139019821)

[3.15 Evidencia ambiente de producción 120](#_Toc139019822)

[3.13.1 Sprint uno 120](#_Toc139019823)

[3.13.1 Sprint dos 124](#_Toc139019824)

# CAPITULO I INTRODUCCIÓN

## INTRODUCCIÓN

Este proyecto se llevó a cabo con un plan de trabajo y su diagnóstico, a través de lo cual se pudo obtener la información requerida para su desarrollo.

Actualmente ocurren muchos accidentes en la vía pública que pueden ser prevenidos si se contará con información actualizada y periódica, pero en estos momentos no se cuenta con dicha información siendo este el motivo que llevó al desarrollo de este proyecto.

Muchos peatones se ven afectados a diario por diferentes eventos que hay en la vía pública, no encuentran el espacio para hacer sus denuncias, ya que existe mucha burocracia para llegar a los organismos que solucionen estos temas. Este problema debe ser tratado, ya que esto evitará futuros accidentes y cada peatón será un actor importante en proporcionar dicha información para que las autoridades den respuesta rápida y efectiva a los peatones.

El desarrollo de este proyecto permitirá que a través de una aplicación los ciudadanos puedan hacer su denuncia a una plataforma que consistirá en una aplicación móvil, la cual se irá actualizando con la información proporcionada por ellos, la información recopilada por la aplicación móvil será vendida a los servicios públicos interesados en solucionar los desperfectos reportados en la vía pública para el bienestar de la comunidad.

El desarrollo de este proyecto tiene como finalidad aportar a la sociedad en disminuir las cifras de accidentes a través de la prevención.

## 1.2 DEFINICIÓN DEL PROBLEMA

Aplicando la metodología del árbol es posible llegar al problema de una forma más precisa indicando así sus ramas, tronco y raíces.

Ramas 🡪 efectos del problema

Tronco 🡪 problema

Raíces 🡪 causas del problema

Efectos: ocurrencia de accidentes personales o materiales, múltiples reclamos a servicios públicos de uno o más denunciantes, no se hace una mantención periódica a los servicios públicos.

Problema: Falta de información oportuna en un periodo de tiempo que no excedan los 60 minutos para peatones, sobre desperfectos en la vía pública que son de fácil solución como árboles que estén a punto de caer, tapas de alcantarilla rotas, cables eléctricos en el suelo, entre otros y que pueden causar accidentes.

Causas: proceso de validación de reclamos demasiado extenso, tiempo de respuesta extendido a reclamos registrados, desconocimiento de las atribuciones de cada servicio por parte de la comunidad para hacer un reclamo, la municipalidad no cuenta con personal suficiente para inspeccionar las vías.

### 1.2.1 DIAGRAMA DE ISHIKAWA

El siguiente diagrama de Ishikawa expuesto en la figura 1, se muestra el problema acotado e identificado y evidencia las causas que llevan al problema mediante la técnica de los “N ¿por qué?”, los cuales se encuentran debajo o arriba de cada jerarquía encerrada en los rectángulos correspondientes refiriéndose así a:

* Porque las noticias no cubren alertas de poca importancia.
  + Porque las denuncias a la municipalidad no tienen el impacto social para ser noticia.
    - Porque una noticia para ser noticia debe causar un daño grave a la comunidad.
* Porque la ocurrencia de sucesos espontáneos no genera alertas a tiempo.
  + Porque hay mucha burocracia al informar un evento.
    - Porque muchas de las noticias resultan ser falsas.
* Porque existen servicios públicos en mal estado.
  + Porque no se realizan mantenciones constantes de los servicios públicos.
    - Porque al ser tantos y diversos no hay personal suficiente para informar el deterioro de estos.
* Porque cuando hay mal tiempo la ciudad se ve afectada de diferentes formas.
  + Porque las entidades públicas no dan abasto con todos los problemas que se producen a raíz de esto.
    - Porque parte de los problemas que no fueron solucionados representan un riesgo para la integridad del peatón.

Diagrama

Descripción generada automáticamente

Figura 1: diagrama de Ishikawa sobre el problema de estudio.

### 1.2.2 LEVANTAMIENTO DE INFORMACIÓN (FOCUS GROUP)

Por medio de la aplicación WhatsApp se creó un grupo conformado por dieciséis personas las cuales accedieron a dar su opinión y aportar con requisitos que para ellos resultarían útiles al momento de usar la aplicación que se piensa desarrollar, enfocada en la prevención y alerta de denuncias.

Las personas comentaron que la aplicación debería permitir:

* Sacar una captura de imagen y dar breve descripción de esta captura.
* Que la aplicación por medio del GPS del celular de una geo referenciación desde donde se produjo dicha denuncia, que esta lleve el nombre y RUT de la persona que hizo dicha denuncia.
* Que otras personas puedan ver y calificar si la denuncia hecha es verídica o falsa.
* Poder calificar la denuncia como un riesgo para la vida de otras personas o una situación que aún no representa un peligro.
* Que las alertas no duren eternamente, sino que pasado un tiempo estas desaparezcan.
* Un sistema de puntos para penalizar a las personas que generen alertas falsas.
* Que las denuncias puestas en la aplicación sean de resolución inmediata.
* Si una alerta resulta falsa que los mismos usuarios puedan bajar la alerta.
* Si el sistema es usado para publicar cualquier otra cosa que no sea una denuncia por desperfecto se sancione la cuenta con el cierre.
* Que la aplicación indique los focos de delincuencia.
* Por cada incidente debería existir un tipo de historial que muestre estado, fecha de inicio, reincidencia de la denuncia, ver comentarios de personas que han comentado la denuncia, datos que podrían priorizar la denuncia para su pronta solución.
* Los tipos de denuncias acogidas por las aplicaciones deberían incluir tapas de alcantarillas rotas, arboles a punto de caer, cables eléctricos al alcance de peatones, semáforos apagados, arreglos hechos por los propios peatones, reportar mantención necesaria, carpas de indigentes, microbasurales, calles bloqueadas por comercio ambulante, fugas de gas, animales en carretera.
* Personas pueden interactuar con la denuncia de otro usuario a través de un menú.
* Al presentar el proyecto a la municipalidad agregar una clasificación para el estado de la alerta, ya sea un estado de revisión o uno de resolviendo.
* Denuncias acogidas por la aplicación deben ser definidas en categorías.
* Categorías definidas deben incluir solo las denuncias más comunes y que puedan suceder en la vía pública específicamente.
* La aplicación debe ser de color celeste pastel.
* Los botones deben ser de tamaño medio para no forzar la vista.

### 1.2.3 REQUISITOS DE ALTO NIVEL

1. Captura de imagen para evidenciar la denuncia y dar breve descripción de esta captura de forma opcional.
2. Que la aplicación móvil por medio del GPS del celular de una geo referenciación desde donde se produjo dicha denuncia, que esta lleve el nombre y RUT de la persona que hizo dicha denuncia.
3. Las personas deben estar registrada en la aplicación para hacer uso de esta.
4. Otros usuarios pueden ver y calificar si la denuncia hecha es verídica o falsa.
5. Usuarios pueden interactuar con la denuncia de otro usuario a través de un menú.
6. Usuario podrá calificar la denuncia como un riesgo para la vida de otras personas o una situación que aún no representa un peligro.
7. Denuncias puestas como alertas en el mapa deben ser de duración limitada y desaparecer en un tiempo.
8. Usuarios al interactuar con otras alertas deben ser capaces de reportar si la alerta es falsa y restarle tiempo de duración a la alerta en el mapa para su pronta eliminación.
9. Sistema de puntos para penalizar a los usuarios que generen alertas falsas, puntos negativos harán que las alertas duren menos tiempo.
10. En caso de reportar alertas que no sean exclusivamente denuncias se procederá a cerrar la cuenta del usuario.
11. Por cada incidente debería existir un tipo de historial que muestre el estado, fecha de inicio, reincidencia de la denuncia, ver comentarios de personas que han comentado la denuncia, datos que podrían priorizar la denuncia para su pronta solución.
12. Los tipos de denuncias acogidas por la aplicación deberían enfocarse en un tipo especifico de denuncias e ir desbloqueando el resto a modo de pago para no saturar el mapa.
13. Incluir un filtro de alertas y un radio especifico de metros alrededor del usuario para que este visualice.
14. Denuncias acogidas por la aplicación deben ser definidas en categorías.
15. Categorías definidas deben incluir solo las denuncias más comunes y que puedan suceder en la vía pública específicamente.
16. Color de la aplicación móvil debe ser color celeste pastel.
17. Botones en la aplicación móvil deben ser de tamaño medio.

### 1.2.4 REQUISITOS DE ALTO NIVEL CLASIFICACIÓN

|  |  |  |
| --- | --- | --- |
| Requisitos | Funcional | No funcional |
| Captura de imagen para evidenciar la denuncia y dar breve descripción de esta captura de forma opcional. | X |  |
| La aplicación móvil por medio del GPS del celular de una geo referenciación desde donde se produjo dicha denuncia, que esta lleve el nombre y RUT de la persona que hizo dicha denuncia. | X |  |
| Las personas deben estar registrada en la aplicación para hacer uso de esta. | X |  |
| Otros usuarios pueden ver y calificar si la denuncia hecha es verídica o falsa. | X |  |
| Usuarios pueden interactuar con la denuncia de otro usuario a través de un menú. | X |  |
| Usuario podrá calificar la denuncia como un riesgo para la vida de otras personas o una situación que aún no representa un peligro. | X |  |
| Denuncias puestas como alertas en el mapa deben ser de duración limitada y desaparecer en un tiempo. | X |  |
| Usuarios al interactuar con otras alertas deben ser capaces de reportar si la alerta es falsa y restarle tiempo de duración a la alerta en el mapa para su pronta eliminación. | X |  |
| Sistema de puntos para penalizar a los usuarios que generen alertas falsas, puntos negativos harán que las alertas duren menos tiempo. | X |  |
| En caso de reportar alertas que no sean exclusivamente denuncias se procederá a cerrar la cuenta del usuario. | X |  |
| Por cada incidente debería existir un tipo de historial que muestre el estado, fecha de inicio, reincidencia de la denuncia, ver comentarios de personas que han comentado la denuncia, datos que podrían priorizar la denuncia para su pronta solución. | X |  |
| Los tipos de denuncias acogidas por la aplicación deberían enfocarse en un tipo especifico de denuncias e ir desbloqueando el resto a modo de pago para no saturar el mapa. |  | X |
| Incluir un filtro de alertas y un radio especifico de metros alrededor del usuario para que este visualice. | X |  |
| Denuncias acogidas por la aplicación deben ser definidas en categorías. | X |  |
| Categorías definidas deben incluir solo las denuncias más comunes y que puedan suceder en la vía pública específicamente. |  | X |
| Color de la aplicación móvil debe ser color celeste pastel. |  | X |
| Botones en la aplicación móvil deben ser de tamaño medio. |  | X |

Tabla 1: requisitos clasificados por funcional y no funcional.

## 1.3 OBJETIVOS GENERAL

Desarrollar una aplicación móvil capaz de mantener informado al peatón sobre los accidentes e incidentes ocurridos en la vía pública alrededor de un área, los cuales serán reportados por otros usuarios retroalimentando así la información de la aplicación de una manera ágil, también permitirá al usuario interactuar con los reportes ya generados.

## 1.4 OBJETIVOS ESPECÍFICOS

* Las alertas que ya tengan cinco reincidencias y que cumplan con ser un riesgo grave serán enviadas automáticamente a correos que comuniquen este peligro a las redes sociales que ya existen mediante un protocolo para transferencias simples de correo (SMTP). Una vez en el despliegue se espera poder registrar al menos tres alertas de riesgo grave y que estas sean informadas correctamente en un tiempo estimado de un mes.

* Se implementarán variadas medidas para demostrar la veracidad de una alerta grave, esta va a ser informada a “Alerta Valparaíso” u otro stakeholder que quiera comprar la información reunida por la aplicación móvil. Se espera que las primeras tres alertas graves que cumplan con estos requisitos lleguen a una etapa donde sean comunicadas a los stakeholders interesados en un tiempo estimado de un mes.
* Se dejará un registro de las averías que ocurran y cuando fueron solucionadas, ya que en caso de que se detecten múltiples alertas en un plazo de cuatro meses de un punto del cual ya fue solucionado el desperfecto, se le indicara al stakeholder el lugar donde posiblemente hay un evento “x” que genera el desperfecto constantemente.

## 1.5 MÉTRICAS

Las métricas en este documento están enfocadas al desarrollo, control e implementación de el o los objetivos generales y específicos en una etapa previa a la implementación y después de está.

* Incrementar el número de alertas graves enviadas por correo en un 67% más durante el segundo mes de prueba de la aplicación móvil.
* Alertas graves deben cumplir con el 100% de las medidas que demuestren su veracidad para ser enviadas por correo a los stakeholders en un tiempo estimado de cinco días desde su primera reincidencia dentro del primer mes de prueba de la aplicación móvil.
* Incrementar la detección de eventos “x” en un 50% luego de encontrar el primero, en un tiempo estimado de 4 meses.

## 1.7 PLAN DE PROYECTO

**Planificación del proyecto**

A continuación, se describen las actividades planificadas para dar cumplimiento a los objetivos del proyecto en el transcurso del año 2023:

Carta Gantt

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Actividad | Mes 1 | Mes 2 | Mes 3 | Mes 4 | Mes 5 | Mes 6 | Mes 7 | Mes 8 | Mes 9 |
| Formulación y planificación del proyecto. |  |  |  |  |  |  |  |  |  |
| Identificación de los requisitos de alto nivel mediante Focus Group. |  |  |  |  |  |  |  |  |  |
| Revisión de fuentes secundarias sobre geolocalización. |  |  |  |  |  |  |  |  |  |
| Desarrollar un prototipo de implementación de geolocalización. |  |  |  |  |  |  |  |  |  |
| Trabajo en terreno sobre identificación de sitios estratégicos en el plano de Viña. |  |  |  |  |  |  |  |  |  |
| Desarrollo de diagramas de las vistas lógica, despliegue, procesos y física. |  |  |  |  |  |  |  |  |  |
| Programación de componentes. |  |  |  |  |  |  |  |  |  |
| Desarrollo de pruebas unitarias, de aceptación y de despliegue. |  |  |  |  |  |  |  |  |  |
| Redacción de la documentación de la aplicación y de la memoria de título. |  |  |  |  |  |  |  |  |  |
| Métrica para iincrementar el número de alertas graves enviadas por correo en un 67%. |  |  |  |  |  |  |  |  |  |
| Métrica para las alertas graves que deben cumplir con el 100% de las medidas que demuestren su veracidad para ser enviadas por correo. |  |  |  |  |  |  |  |  |  |
| Métrica para encontrar eventos “x” en los informes para stakeholders. |  |  |  |  |  |  |  |  |  |
| Entrega de memoria de título a la comisión |  |  |  |  |  |  |  |  |  |

## 1.8 JUSTIFICACIÓN DEL PLAN DE PROYECTO

Cada uno de los sprints de este proyecto esta pensado para durar un tiempo de cuatro semanas, es decir un mes debido a la cantidad de puntos asignada por el Planing Poker correspondiente a cuarenta puntos cada una. Hasta la fecha actual el sprint dos ha sido desarrollado e implementado satisfactoriamente antes de su termino programado en el mes de junio. Debido a la carga que representa cuarenta puntos no es posible acortar o disminuir el tiempo de desarrollo de estos sprints, se estima que serán seis y su duración esta prevista hasta el mes de noviembre en donde se desarrollara completamente la aplicación móvil Safe Travel Map.

## 1.9 ALCANCE DEL PROYECTO

Este proyecto abarcara a todos los peatones que se vean afectados a accidentes o resultados de estos dentro de la ciudad, estos pueden ser personas no deportistas, deportistas, minusválidos, lesionados entre otros.

Se creará una aplicación que les permita informarse e informar al resto de forma constante y actualizada de los riesgos que hay en la vía.

Se estima que en un plazo de 8 meses se podría implementar para la ciudad de viña del mar y Valparaíso, después de unos meses de prueba se extenderá a las demás ciudades, cuando ya se logre cubrir todas las ciudades se contactara a las municipalidades para hacerlos partícipes de forma activa en dicha aplicación.

## 1.10 LIMITACIONES DEL PROYECTO

La ejecución del proyecto se puede ver limitado por:

* La aplicación móvil depende de una conexión de red estable.
* El proyecto está diseñado para la ciudad de viña del mar y Valparaíso.
* Usuario debe estar registrado en la aplicación móvil.
* Aplicación móvil creada solamente para teléfonos celulares.
* Celulares de gama baja no corren la aplicación móvil.

# CAPITULO II FUNDAMENTACIÓN DEL TEMA

## 2.1 ALTERNATIVAS DE SOLUCIÓN

Se creará una aplicación la cual le permitirá a peatones subir información de problemas que se están generando en el minuto para aportar en la prevención de los diferentes desperfectos que se encuentran dentro de su ciudad, de esta manera se podrán prevenir a tiempo accidentes, esto ayudara a los ciudadanos a estar alerta ante cualquier peligro al que se puedan exponer.

## 2.2 FACTIBILIDAD TÉCNICA

Recurso Humano

Es indispensable un desarrollador que sepa programar con Android Estudio, trabaje con la herramienta GCP para la base de datos de la aplicación móvil, demuestre un conocimiento básico para hacer versionamientos en Github y posea conocimiento en la herramienta de gestión Jira.

Hardware

|  |  |
| --- | --- |
| CPU | Intel Core i7-7700HQ (4 núcleos/8 hilos/2800 MHz – 3800 MHz) |
| RAM | 8 GB DDR4 (2400 MHz) |
| Pantalla | LED 17.3" (1920x1080) / 60 Hz |
| Almacenamiento | SSD 1TB |

Software

|  |  |
| --- | --- |
| Windows 10 pro | Herramientas de clase empresarial y una administración flexible, sistema operativo diseñado por Microsoft. |
| Android Studio | Android Studio es el entorno de desarrollo integrado oficial para la plataforma Android. Fue anunciado el 16 de mayo de 2013 en la conferencia Google I/O, y reemplazó a Eclipse como el IDE oficial para el desarrollo de aplicaciones para Android. La primera versión estable fue publicada en diciembre de 2014. |
| GCP | Google Cloud es una plataforma que ha reunido todas las aplicaciones de desarrollo web que Google estaba ofreciendo por separado. Provee servicios de Big Data, Networking, Machine Learning e informática. |
| GitHub | GitHub es una forja para alojar proyectos utilizando el sistema de control de versiones Git. Se utiliza principalmente para la creación de código fuente de programas de ordenador. El software que opera GitHub fue escrito en Ruby on Rails. |
| Jira | Jira es un producto de software propietario para la gestión de proyectos, seguimiento de errores e incidencias. La herramienta fue desarrollada por la empresa australiana Atlassian. Su herramienta de gestión ágil para proyectos de software es compatible con la metodología de SCRUM. |

## 2.3 FACTIBILIDAD LEGAL

Dentro del marco legal la única licencia necesaria para trabajar se paga una vez y corresponde a Windows 10 pro, ya que la herramienta de Android Studio es gratuita y GCP ofrece 12 meses gratis en su plan de prueba.

## 2.4 FACTIBILIDAD ECONÓMICA

Los recursos económicos y financieros necesarios para llevar a cabo el desarrollo de este software con sus actividades y procesos correspondientes serán obtenidos mediante un modelo de negocio que incluye:

* La venta de información para los stakeholders interesados en resolver las alertas que son informadas por los usuarios refiérase a stakeholders como la municipalidad, Esval, Chilquinta, entre otros. Cualquier servicio público que esté dispuesto a pagar por la información.
* Rentar la aplicación móvil a anuncios publicitarios que incluyan una forma de ingresos por un modelo de facturación que genere “costos por cualquier cosa” (CPX), así cualquier tipo de interacción por parte de los usuarios con los anuncios generara un ingreso económico.
* Para incentivar a nuestros usuarios a interactuar con los anuncios se aplicará un modelo de negocio llamado Freemium que contenga aparte de los servicios gratuitos otros más avanzados, estos se desbloquearan por un tiempo limitado luego de interactuar con algún anuncio de la aplicación. El objetivo de este último modelo es ser usado para ampliar los servicios ya existentes y apoyar las mejoras del desarrollo de la aplicación móvil, no va orientada directamente al cobro para los usuarios que son quienes nos proveen la información.

|  |  |
| --- | --- |
| CPX 2023 | Ingresos promedio Estimados |
| Costo por Clic (CPC) | 0.02 USD |
| Costo por Mil (CPM) | 20 USD |
| Costo por acción (CPA) | Variados USD |
| Anuncio banner | 0,1 USD \* visitante |
| Anuncio video | 0,16 USD |

Tabla 2.

La tabla dos contiene todos los tipos de interacción por las cuales se puede recibir un ingreso, ya sea un costo por darle Clic al anuncio, costo por mil interacciones, costo por acciones en el anuncio refiriéndose a descargar alguna aplicación recomendada o rellenar formularios, anuncios banner que puedes ser fácilmente ignorados o cuyas interacciones se den por equivocación y el anuncio de video que ocuparía la totalidad de la pantalla con una duración de segundos. Los ingresos estarían dados por la formula “Ingresos publicitarios = impresiones \* estimación recibida por impresiones cada mil anuncios” y estos son estimados porque una vez creada la aplicación el ingreso dependerá del tipo de anuncio y el tamaño.

|  |  |  |
| --- | --- | --- |
|  | Ingresos Estimados. | Costo evaluado |
| Venta de información | Aún por definir |  |
| Desarrollador (1) |  | 0,4 UF por HH |
| Windows 10 Pro |  | $9990 |
| Android Studio |  | $0 |
| GCP (Free) |  | $0 |
| SSD 1Tera |  | $50000 |
| Internet |  | $37000 |
| Luz |  | $20000 |
| RAM |  | $34000 |
| CPU |  | $259000 |
| Pantalla |  | $87000 |

Tabla 3.

La tabla tres muestra el ingreso estimado por venta de información a los servicios públicos y los gastos que tendrá el proyecto. Al ser una aplicación en desarrollo solo se gastará en programas.

La información mostrada en este documento para la factibilidad económica y sus tablas de ingresos estimados fue extraída del siguiente enlace:

https://www.movapps.com.ar/cuanto-se-paga-por-publicidad-en-una-app.html

## 2.5 ESTUDIO DE FACTIBILIDAD DEL PROYECTO

Entre las aplicaciones que se analizaron para llegar a este problema y a su solución se destacan las siguiente de las cuales se adjunta el enlace para ingresar a cada una de ellas:

* SOSAFE: una red ciudadana para solicitar ayuda en caso de robos, accidentes, extravíos de mascotas, solicitar ambulancias y llamar a bomberos. Creada para la zona oriente de Santiago que abarca las comunas de Las Condes, Vitacura y Lo Barnechea. Genera estadísticas de todas las denuncias en las zonas correspondientes para que el municipio tome acción.

**https://www.amszo.cl/la-zona-oriente-ya-cuenta-con-sosafe-seguridad-en-tu-celular/**

* VenApp: Es un software creado para celulares que les facilita a los usuarios hacer denuncias directamente a su municipio y abarca desperfectos en la vía como tuberías rotas, fallas en el alumbrado público, huecos entre otros. Esta aplicación móvil fue desarrollada para Venezuela y además del servicio de denuncias contiene secciones de grupos informativos para otras actividades que se dan en Venezuela, otra sección llamada VenMarket para ver los productos y servicios más buscados, comics, una red social similar a Instagram entre otros servicios.

**https://venapp.com/**

* Repara Ciudad: Aplicación móvil de celulares creada para el peatón cuya función es permitirle al usuario reportar desperfectos en la ciudad, clasificarlos según su peligro, ya sea leve o grave y el tipo de denuncia al cual corresponde, ya sea limpieza, faroles rotos entre otros. Genera un dialogo entre el municipio y el usuario para que este confirme que la denuncia a llegado y no se informó por informar.

**https://www.lavanguardia.com/tecnologia/20110726/54191469746/una-nueva-aplicacion- para-movil-permite-denunciar-los-desperfectos-de-la-calle.html**

* Waze: es un tipo de red ciudadana enfocada solo a los automovilistas que pueden reconocer desperfectos en la vía, tráfico, accidentes, controles policiales, vías bloqueadas, condiciones meteorológicas y otras cosas que dificulten o atrasen al conductor en su viaje. Waze se retroalimenta de todo lo que los usuarios comparten entre sí.

**https://support.google.com/waze/answer/6078702?hl=es**

Al profundizar en las redes sociales como Alerta Viña, Alerta noticias Valparaíso y RedGeoChile cuya finalidad es ser un medio de información para los usuarios quienes visitan estos sitios se pudo observar que las redes sociales con más visitas son “Instagram” por la entrega de contenido visual que puede generar, sin embargo existen usuarios que prefieren “Twitter”, en todas estas redes sociales es el municipio quien brinda información a la comunidad de posibles peligros para quienes transitan por las zonas afectadas, no existe un lugar donde el usuario pueda subir sus propias denuncias.

Se puede observar que en muchos casos usuarios comentan sus propias denuncias en post hechos por el municipio donde colocan la posición exacta del desperfecto en la ciudad con evidencia fotográfica y el tiempo que lleva en el lugar, pero no son tomados en cuenta porque en estas redes casi el 90% de las noticias no son para prevenir accidentes antes de que ocurran.

**https://twitter.com/vinadelmar/status/1428395140180324356**

Ejemplo: si un muro de contención está en mal estado con nuestra aplicación el usuario podría evidenciar que existe un problema, el cual representa un riesgo para la seguridad de los demás peatones o automovilistas y podría prevenir un accidente.

Hasta este punto aun no es una noticia de interés para las redes sociales, sin embargo, con nuestra aplicación esta alerta se le daría a todos los afectados en la zona para prevenir accidentes. En la escalabilidad del proyecto se espera poder vender la información recopilada por la aplicación móvil a todos los stakeholders interesados y que deseen darles una solución a las alertas emitidas por los usuarios.

## 2.6 DIAGRAMA DE ALTO NIVEL

La figura dos ubicada abajo muestra el diagrama de alto nivel que permite visualizar el flujo de interacción que tiene el usuario con el aplicativo. Primeramente, el usuario que en este caso sería el peatón abre el aplicativo y este se comunica con GCP una herramienta desarrollada por Google que provee servicios de Big Data, Networking, Machine Learning e informática. GCP nos permitirá manejar la base de datos necesaria para ir guardando las reincidencias de las alertas generadas por los usuarios e ir almacenando datos con respecto a estas mismas. Desde GCP se genera una retro alimentación al aplicativo y un informe de datos para el stakeholder.

Diagrama

Descripción generada automáticamente

Figura 2: diagrama de alto nivel.

## 2.7 DOCUMENTACIÓN DE DISEÑO

Con la finalidad de identificar, describir y comprender si el diseño para él sistema se está llevando a cabo de forma correcta se empleará el modelo 4+1 de las vistas de Kruchten pasando de esta forma por cada una de las vistas que se encuentran en la figura tres.

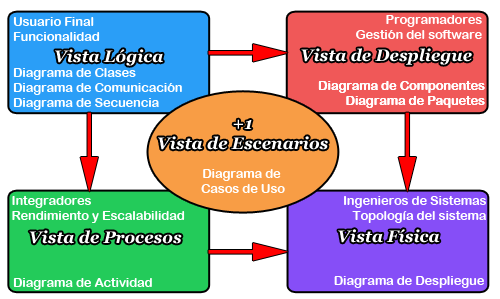


Figura 3: Metodología 4+1 modelo vistas.

## 2.8 VISTA DE ESCENARIOS

A continuación, la figura trece nos muestra las actividades del usuario y el administrador en la aplicación, el sistema está integrado por lo que no se muestra las actividades que realiza.



Figura 4: diagrama caso de uso Safe travel Map App.

## 2.9 VISTA LÓGICA

En la figura catorce se encuentra el diagrama de clases conceptual que muestra la estructura del sistema y con ello todas las clases que componen el sistema de la aplicación móvil Safe Travel Map con sus respectivos atributos, métodos, relaciones y herencias.

Diagrama, Esquemático

Descripción generada automáticamente

Figura 5: Diagrama de clases conceptual.

## 2.10 VISTA DE DESPLIEGUE

En la Figura se puede ver el diagrama de componentes que permite ver las funcionalidades del software en relación con los elementos de interconectividad, en el diagrama se puede apreciar los componentes existentes en el sistema, interfaces y relaciones de dependencias entre ellas.

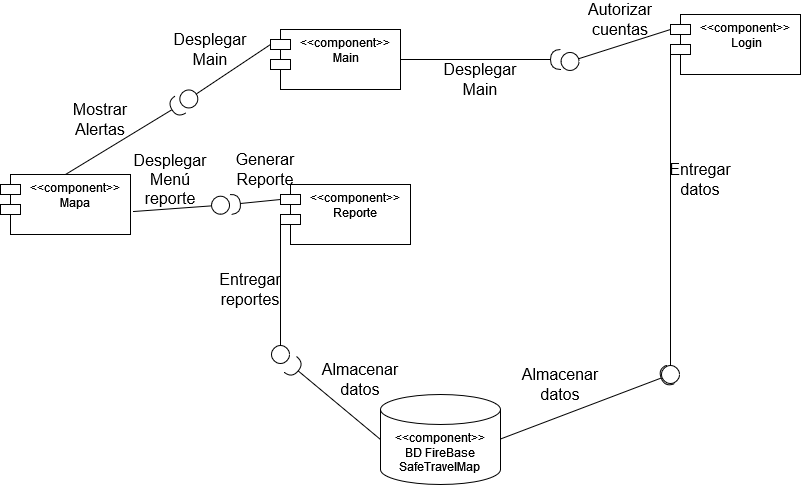


Figura 6: Diagrama de componentes.

## 2.11 DOCUMENTACIÓN DE ANÁLISIS

Por cada sprint se llevará a cabo un análisis dirigido a cada historia de usuario correspondiente del backlog las cuales serán desarrolladas e implementadas en el sprint que este en curso dentro de la herramienta de gestión Jira, al finalizar el sprint se realizara el sprint review para comprobar que todos los criterios de aceptación puestos en la matriz de trazabilidad para las historias de usuario se desarrollaron de manera correcta.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Enunciado | de | la | Historia | Criterios de | aceptación |  |
| Identificador  (ID)de la  historia | Rol | Característica/  Funcionalidad | Razón/Resultado | Número (#)  de Escenario | Criterios de aceptación título | contexto | evento |
|  |  |  |  | |  | | --- | | 1 | | 2 | | 3 | | 4 | | |  | | --- | |  | |  | |  | |  | | |  | | --- | |  | |  | |  | |  | | |  | | --- | |  | |  | |  | |  | |
|  |  |  |  |  |  |  |  |

Tabla 4: plantilla para el análisis de pruebas de aceptación.

# CAPITULO III PLANES Y METODOLOGÍAS

## 3.1 METODOLOGÍA DE GESTIÓN

Al ser un proyecto que en el trascurso del tiempo estará sujeto a cambios, será de larga duración pensado para un año donde se deberán hacer entregas semanales y tener reuniones con el profesor guía a cargo de ver el progreso del alumno se optara por elegir la metodología de SCRUM. Al elegir la metodología de SCRUM podremos trabajar con un conjunto de buenas prácticas las cuales le aportaran valor al producto final que se desarrollara en el transcurso de este año.

Se seguirá el proceso de SCRUM en sus diferentes etapas las que incluyen el levantamiento de los requerimientos del usuario, la creación del Product Backlog, organización de los Sprint Backlog, planificando los Sprint concluyendo con las revisiones de los entregables y su retrospectiva correspondiente.

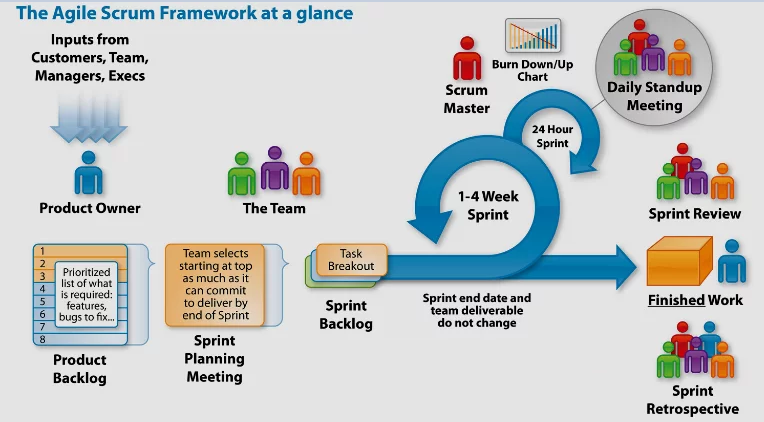


Figura 7: Proceso de la metodología Scrum en curso.

Jira por ser una herramienta ágil de gestión de proyectos compatible con metodologías agiles será elegida para llevar a cabo la gestión de este proyecto.

## 3.2 METODOLOGÍA DE DESARROLLO

Se empleará una metodología iterativa e incremental, ya que durante la ejecución de este proyecto se irán finalizando iteraciones que se controlaran mediante una inspección en la planificación del proyecto evaluando si va acorde al objetivo inicial planteado. Si luego de la inspección el desarrollo se ve comprometido se tomarán medidas correctivas, como aquellas vistas previamente en el plan de mitigación de riesgo y se aplicarán los cambios pertinentes al objetivo final del proyecto de ser necesario. Cualquier cambio que tenga un impacto en el desarrollo o que surja entre las inspecciones que resulte de forma inesperada podrá ser agregado.

Al finalizar cada iteración se le pregunta al usuario si lo desarrollado va cumpliendo con sus expectativas y en caso contrario se le van aplicando ajustes para acercarse a estas, lo que hace que las entregas sean incrementales al dar siempre un producto utilizable, así el usuario nos puede dar su feedback e ir reduciendo el riesgo para llevarlo de forma controlada.

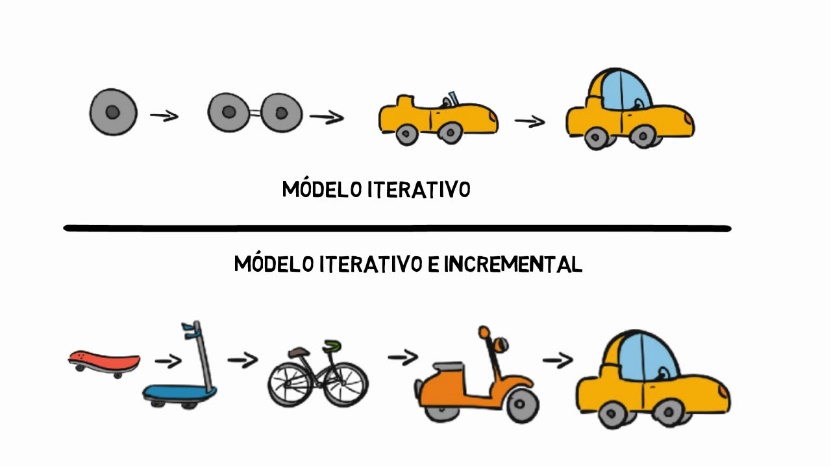
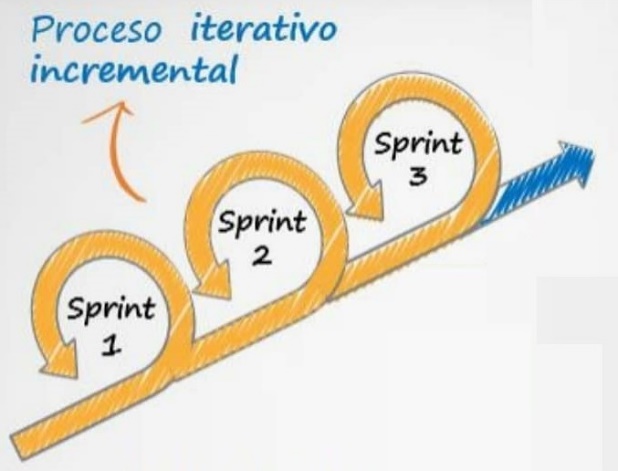


Figura 8: ejemplos de procesos iterativos e incrementales.

## 3.3 PRODUCT BACKLOG

### HISTORIAS DE USUARIO

|  |  |
| --- | --- |
| Identificador de la historia | Enunciado de la Historia |
| SAF2 | Yo como usuario quiero que todas las personas que usen la aplicación estén registradas para dar a conocer a quien resuelve las alertas que son personas reales las que se están siendo beneficiadas. |
| SAF3 | Yo como usuario quiero poder subir una captura desde el almacenamiento de mi celular o tomar una foto del desperfecto en la vía pública para dejar evidencia fotográfica del peligro encontrado. |
| SAF4 | Yo como usuario quiero dar una descripción opcional de lo que veo en la fotografía que estoy dejando en la alerta para dar más detalles del desperfecto en la vía pública. |
| SAF5 | Yo como usuario quiero calificar el desperfecto en la vía pública como una alerta grave o baja para que otros usuarios eviten pasar por el lugar si es grave. |
| SAF6 | Yo como usuario quiero que la aplicación móvil georreferencié desde donde se está emitiendo la alerta y guarde la ubicación para que la alerta quede subida en el lugar donde se reportó. |
| SAF7 | Yo como usuario quiero que luego de subir una alerta se genere un radio alrededor de está para que no se generen múltiples reportes de la misma alerta y se sature el mapa. |
| SAF8 | Yo como usuario quiero que exista un botón de generar informes que capten todas las alertas de una zona en específico para ir a evaluar la situación in situs y arreglarlas de ser posible. |
| SAF9 | Yo como usuario quiero un menú que me permita interactuar con las alertas de otros usuarios para dar de baja la alerta en el caso que sea falsa. |
| SAF10 | Yo como usuario quiero que las alertas duren un tiempo activas y luego se borren del mapa para poder reportar nuevas alertas en caso de que las anteriores se solucionaran. |
| SAF11 | Yo como usuario quiero ver si el estado de una alerta esta activo o en espera para volver a activarla en caso de que el desperfecto en la vía continué. |
| SAF12 | Yo como usuario quiero que existan categorías definidas en la aplicación para no estar tratando de darle con mis palabras una explicación al desperfecto encontrado. |
| SAF13 | Yo como usuario quiero que alrededor mío se muestren todas las alertas existentes para prevenir accidentes en la vía pública. |
| SAF14 | Yo como usuario quiero que exista un sistema de puntos para que aquellas personas que den alertas falsas pierdan credibilidad en sus alertas y duren menos tiempo activas. |
| SAF15 | Yo como usuario quiero que las alertas que no correspondan a denuncias den de baja la cuenta de origen del usuario para prevenir que la aplicación móvil sea usada con otro fin que no sea el de dar alertas preventivas. |
| SAF16 | Yo como usuario quiero que las alertas se suban con nombre y RUT del usuario para darle veracidad a la denuncia. |
| SAF17 | Yo como usuario quiero un filtro de alertas para ver solo las alertas que son graves o de mi interés. |
| SAF18 | Yo como usuario quiero que la aplicación móvil indique los focos de delincuencia que existen en la ciudad. |

Tabla 5.

En la tabla cinco se encuentran todas las historias de usuario con su respectivo identificador de historia tanto el que se le asigno al momento de crear la historia el cual vendría a ser el “H.U”, como aquel que fue asignado por la herramienta de gestión de proyecto Jira al momento de ser subida en la plataforma y vendría a ser el identificador “SAF”.

### PRIORIZACIÓN CON MOSCOW

MoSCoW es un sistema que permite priorizar los requerimientos que se encuentran en el product backlog previamente definido, esta técnica plantea cuatro elementos.

* **Must have (Debe tener):** Son características no negociables que forman parte del producto mínimo viable, serán las absolutamente críticas para el proyecto, sin ellas el proyecto será un fracaso y debe ser la prioridad número uno.
* **Should have (Debería incluir):** Es importante, pero no es vital, sin estos requisitos el proyecto aún puede ser viable, son aspectos críticos también, pero no imprescindibles y su prioridad vendría siendo un dos.
* **Could have (Podría incluir):** Es deseable, pero no tan importante como un “Should have”, estas iniciativas son las que estaría bien tenerlas, ya que añadirían valor al proyecto, pero no son críticas y su prioridad seria de un tres.
* **Won’t have (No se van a hacer):**  Estas características no se merecen la inversión de tiempo, no aportan ningún beneficio en este momento y se podrían considerar más tarde, lo que pondría su prioridad en un cuatro.

A medida que el producto del proyecto cumpla con requerimientos que “deben ir” o que “deberían ir” esta ira adquiriendo más valor, se le considerara un producto ideal si cumple con los tres primeros elementos.

### “PLANING POKER” ASIGNACIÓN DE PUNTOS DE ESFUERZO

Apoyados por la página “planningpoker.com” asignaremos los puntos de esfuerzo a cada historia de usuario, con un estilo de cartas “Fibonacci modificado” para que la escala de esfuerzo sea del 0 al 100 siendo así el 100 una tarea imposible de realizar.



Figura 9: Muestra la actividad realizada en “planingpoker.com” donde se asignaron los puntajes correspondientes a cada historia de usuario.

### TABLA DE PRIORIZACIÓN

|  |  |  |  |
| --- | --- | --- | --- |
| Identificador de la historia por Jira | Enunciado de la Historia | Priorización MoSCoW | Asignación de puntos “Planing Poker” |
| SAF2 | Yo como usuario quiero que todas las personas que usen la aplicación estén registradas para dar a conocer a quien resuelve las alertas que son personas reales las que se están siendo beneficiadas. | Must Have | 40 |
| SAF3 | Yo como usuario quiero poder subir una captura desde el almacenamiento de mi celular o tomar una foto del desperfecto en la vía pública para dejar evidencia fotográfica del peligro encontrado. | Must Have | 20 |
| SAF4 | Yo como usuario quiero dar una descripción opcional de lo que veo en la fotografía que estoy dejando en la alerta para dar más detalles del desperfecto en la vía pública. | Must Have | 13 |
| SAF5 | Yo como usuario quiero calificar el desperfecto en la vía pública como una alerta grave o baja para que otros usuarios eviten pasar por el lugar si es grave. | Should have | 8 |
| SAF6 | Yo como usuario quiero que la aplicación móvil georreferencié desde donde se está emitiendo la alerta y guarde la ubicación para que la alerta quede subida en el lugar donde se reportó. | Must Have | 20 |
| SAF7 | Yo como usuario quiero que luego de subir una alerta se genere un radio alrededor de está para que no se generen múltiples reportes de la misma alerta y se sature el mapa. | Must Have | 13 |
| SAF8 | Yo como usuario quiero que exista un botón de generar informes que capten todas las alertas de una zona en específico para ir a evaluar la situación in situs y arreglarlas de ser posible. | Should have | 20 |
| SAF9 | Yo como usuario quiero un menú que me permita interactuar con las alertas de otros usuarios para dar de baja la alerta en el caso que sea falsa. | Could Have | 13 |
| SAF10 | Yo como usuario quiero que las alertas duren un tiempo activas y luego se borren del mapa para poder reportar nuevas alertas en caso de que las anteriores se solucionaran. | Could Have | 13 |
| SAF11 | Yo como usuario quiero ver si el estado de una alerta esta activo o en espera para volver a activarla en caso de que el desperfecto en la vía continué. | Could Have | 13 |
| SAF12 | Yo como usuario quiero que existan categorías definidas en la aplicación para no estar tratando de darle con mis palabras una explicación al desperfecto encontrado. | Should have | 8 |
| SAF13 | Yo como usuario quiero que alrededor mío se muestren todas las alertas existentes para prevenir accidentes en la vía pública. | Should have | 8 |
| SAF14 | Yo como usuario quiero que exista un sistema de puntos para que aquellas personas que den alertas falsas pierdan credibilidad en sus alertas y duren menos tiempo activas. | Could Have | 20 |
| SAF15 | Yo como usuario quiero que las alertas que no correspondan a denuncias den de baja la cuenta de origen del usuario para prevenir que la aplicación móvil sea usada con otro fin que no sea el de dar alertas preventivas. | Could Have | 13 |
| SAF16 | Yo como usuario quiero que las alertas se suban con nombre y RUT del usuario para darle veracidad a la denuncia. | Could Have | 5 |
| SAF17 | Yo como usuario quiero un filtro de alertas para ver solo las alertas que son graves o de mi interés. | Should have | 8 |
| SAF18 | Yo como usuario quiero que la aplicación móvil indique los focos de delincuencia que existen en la ciudad para evitar pasar por ahí. | Won’t have | 0 |

Tabla 6.

En la tabla seis se encuentran las historias de usuario ya mencionadas anteriormente con sus respectivos identificadores de historia, “SAF” para cuando fueron subidas a la plataforma Jira, también se le agrego su respectiva priorización con MoSCoW y la asignación de puntos luego de ejecutar el Planing Poker.

## 3.4 DOCUMENTACIÓN DE SPRINTS

## 3.4.1 Sprint uno

* En el primer sprint que se encuentra terminado dentro de la herramienta de gestión Jira está la historia de usuario con el identificador “SAF2” correspondiente a: “Yo como usuario quiero que todas las personas que usen la aplicación estén registradas para dar a conocer a quien resuelve las alertas que son personas reales las que se están siendo beneficiadas”, su priorización es Must Have lo que la ubica como parte del producto mínimo viable solicitado por el usuario con un puntaje de cuarenta debido al riesgo y la dificultad que está representa para el comienzo del desarrollo del proyecto, también esta agrupada en la épica correspondiente al “funcionamiento básico”.

###### 3.4.1.1 Tareas sprint uno Jira

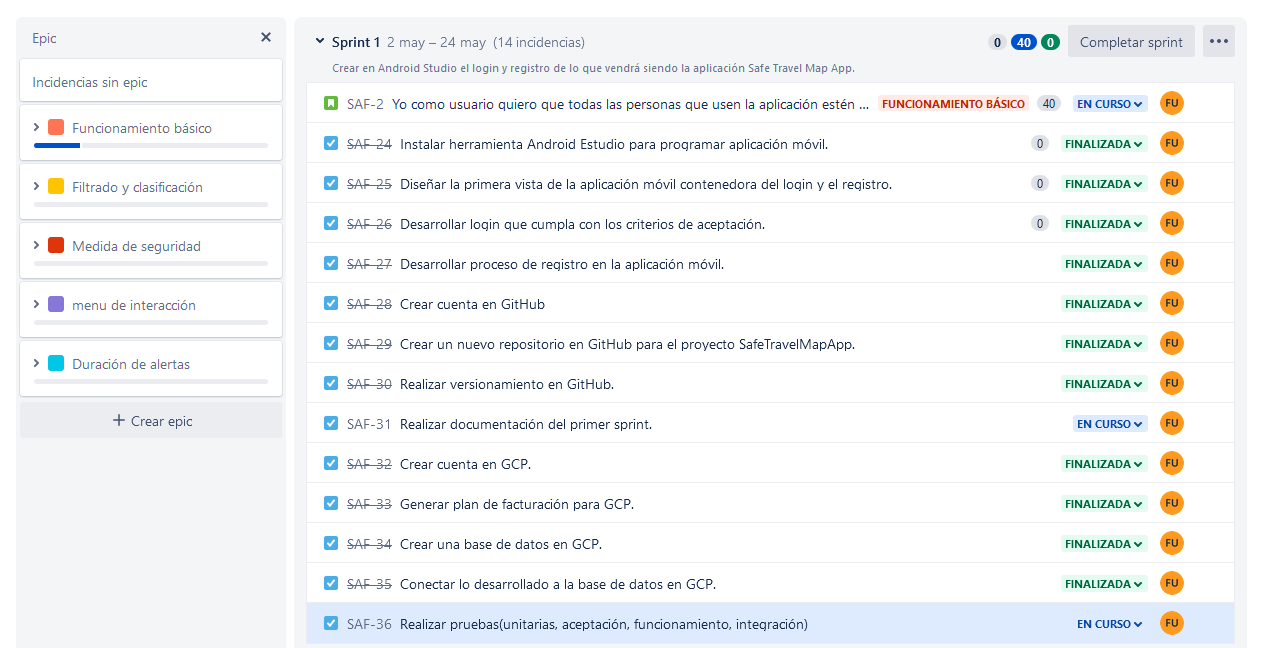


Figura 10: En esta figura se muestra la historia de usuario “SAF2” trabajada en Jira, todas las tareas por las cuales está compuesta la historia, se puede ver algunas tareas ya finalizadas y otras en curso, todas realizadas por el desarrollador Fabio Urrea. También las épicas que dieron origen a las historias de usuario y la duración del primer sprint que va desde el dos de mayo al veinticuatro de mayo.

###### 3.4.1.2 Versionamiento sprint uno GitHub

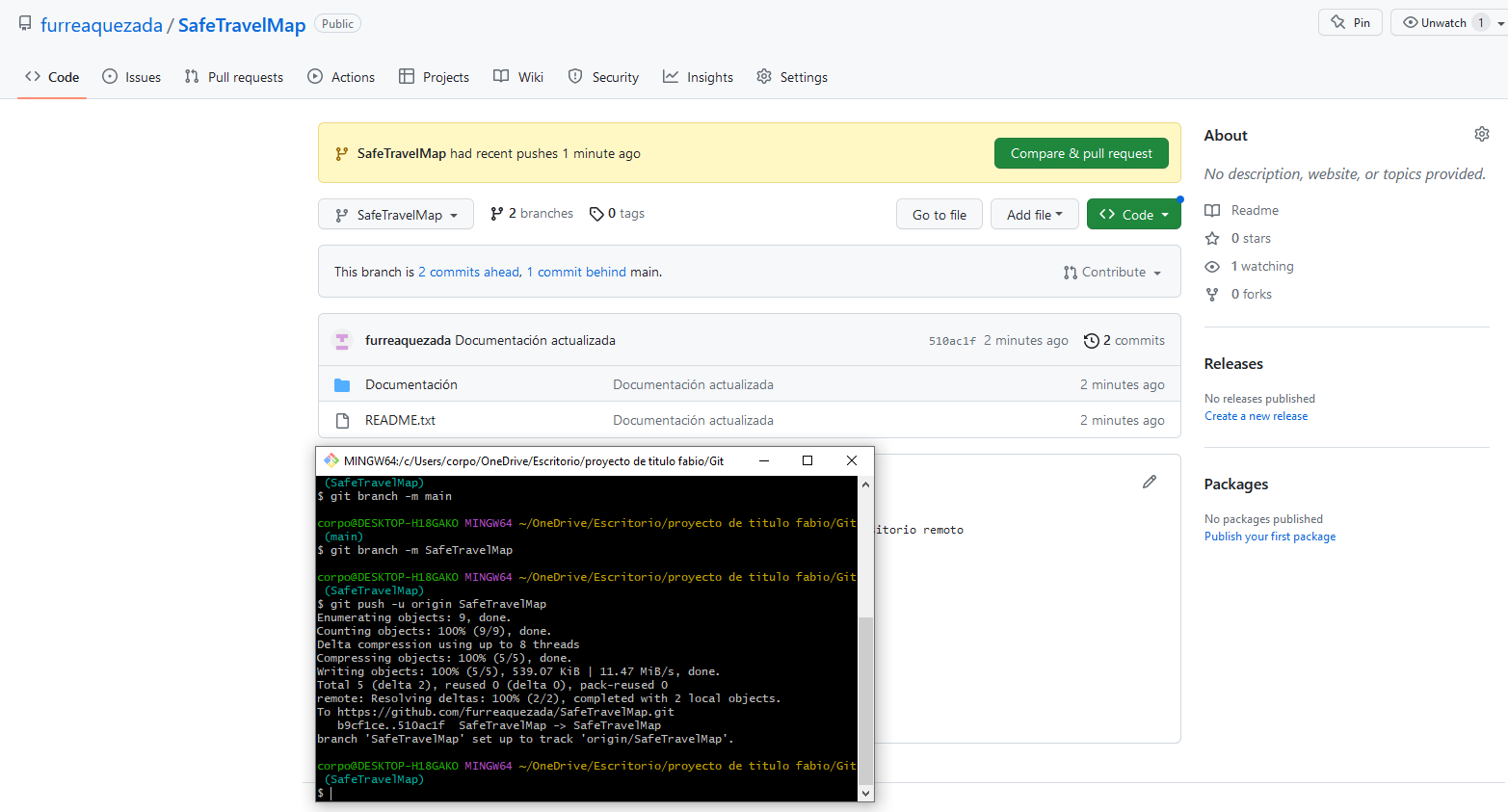


Figura 11: evidencia del versionamiento del sprint.

###### 3.4.1.3 Diseño sprint uno

Diagrama de clases conceptual

El diagrama de clases conceptual que comprende el sprint uno muestra en la figura 12. Todas las clases que se integran, correspondientes a las actuales funcionalidades implementadas.

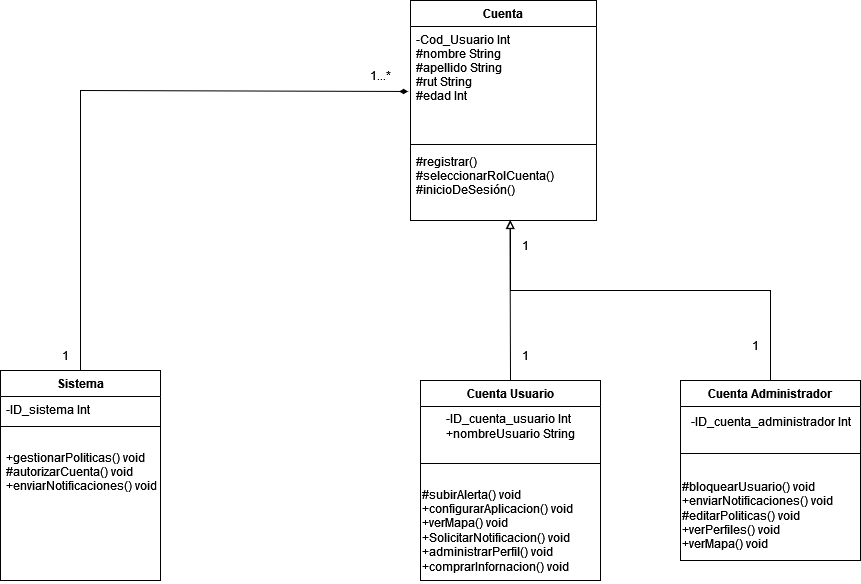


Figura 12: parte del diagrama de clases conceptual abarcado por el sprint uno.

Diagrama de componentes

1. Diagrama de componentes adaptado al proceso de “Login” y “Validación de usuarios” dentro de la base de datos.

Desarrollo:

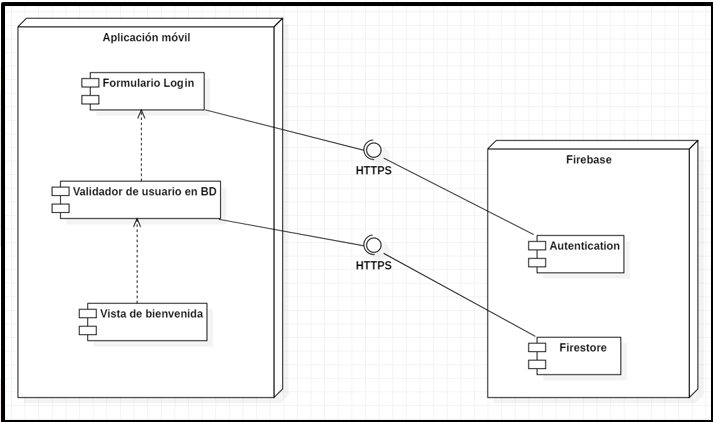


Figura 13: Parte del diagrama de componentes que abarca el sprint uno.

1. Diagrama de componentes adaptado al proceso de “Login” y “Validación de usuarios” dentro de la base de datos.

**Desarrollo:**

1. Aplicación móvil: Este es el punto de entrada principal de la aplicación. Aquí se solicita al usuario que introduzca sus credenciales para el inicio de sesión.

* Módulo de Autenticación: Este módulo interactúa con Firebase Authentication para validar las credenciales de los usuarios.
* Módulo de Manejo de Usuarios: Una vez que la autenticación es exitosa, este módulo interactúa con Firestore para recuperar la información del usuario.

1. Firebase: Plataforma en la nube que proporciona los servicios necesarios para la autenticación y el almacenamiento de datos.

* Authentication: Este servicio autentica las credenciales del usuario, proporcionadas por el módulo de autenticación de la aplicación móvil.
* Firestore: Este servicio almacena la información de los usuarios. Responde a las solicitudes del Módulo de Manejo de Usuarios en la aplicación móvil para recuperar la información de los usuarios.

1. Vistas de Bienvenida: Estas son las diferentes interfaces de usuario que se muestran al usuario después de un inicio de sesión exitoso.

* Vista de Bienvenida para Usuarios Normales: Esta interfaz se muestra a los usuarios normales después del inicio de sesión.
* Vista de Bienvenida para Administradores: Esta interfaz se muestra a los usuarios con privilegios de administrador después del inicio de sesión.

###### 3.4.1.4 Pruebas

Pruebas unitarias

Para llevar a cabo las pruebas unitarias el código fue alterado.

Dentro del sistema, correspondiente a una aplicación móvil, se han desarrollado varias clases que extienden de AppCompat Activity. El Activity que es de interés para las primeras pruebas de unidad corresponde a LoginActivity:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente con confianza media

Ahí se encuentra el método de login que es de interés monitorear.

Interfaz de usuario gráfica, Aplicación, Teams

Descripción generada automáticamente

Este método hace uso de las librerías de la plataforma de Firebase, con el fin de conectarse al servidor y desde él cual consumir los servicios de autenticación y storage.

|  |
| --- |
| Script para el login de usuarios y administradores |
| **if**(!**correo**.getText().toString().equals(**""**) && !**pass**.getText().toString().equals(**""**)){  **mAuth**.signInWithEmailAndPassword(**correo**.getText().toString().trim(), **pass**.getText().toString().trim()).addOnCompleteListener(**new** OnCompleteListener<AuthResult>() {  @Override  **public void** onComplete(@NonNull Task<AuthResult> task) {  **if**(task.isSuccessful()){  finish();  **if**(rol.equals(**"Usuario"**)){  consultar\_usuario();  }**else if**(rol.equals(**"Administrador"**)){  consultar\_administrador();  }  }**else**{  Toast.*makeText*(getApplicationContext(), **"Error en los datos de autenticación!!!"**, Toast.***LENGTH\_SHORT***).show();  }  }  });  **public void** consultar\_usuario(){  **mFirestore** = FirebaseFirestore.*getInstance*();  **mFirestore**.collection(**"usuarios"**).whereEqualTo(**"correo"**, **correo**.getText().toString().trim()).get().addOnSuccessListener(**new** OnSuccessListener<QuerySnapshot>() {  @Override  **public void** onSuccess(QuerySnapshot queryDocumentSnapshots) {  **if**(queryDocumentSnapshots.isEmpty()){  Toast.*makeText*(getApplicationContext(), **"No existe este usuario en base de datos."**, Toast.***LENGTH\_SHORT***).show();  **return**;  }**else**{  Intent intent = **new** Intent(getApplicationContext(), SistemaActivity.**class**);  startActivity(intent);  Toast.*makeText*(getApplicationContext(), **"Bienvenido"**, Toast.***LENGTH\_SHORT***).show();  }  }  });  }  **public void** consultar\_administrador(){  **mFirestore** = FirebaseFirestore.*getInstance*();  **mFirestore**.collection(**"administrador"**).whereEqualTo(**"correo"**, **correo**.getText().toString().trim()).get().addOnSuccessListener(**new** OnSuccessListener<QuerySnapshot>() {  @Override  **public void** onSuccess(QuerySnapshot queryDocumentSnapshots) {  **if**(queryDocumentSnapshots.isEmpty()){  Toast.*makeText*(getApplicationContext(), **"No existe este administrador en base de datos."**, Toast.***LENGTH\_SHORT***).show();  **return**;  }**else**{  Intent intent = **new** Intent(getApplicationContext(), SistemaActivity.**class**);  startActivity(intent);  Toast.*makeText*(getApplicationContext(), **"Bienvenido"**, Toast.***LENGTH\_SHORT***).show();  }  }  });  } |

El testeo desde la aplicación ha sido exitoso:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media Gráfico, Gráfico radial

Descripción generada automáticamente

Login de administradores:

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente Imagen que contiene Patrón de fondo

Descripción generada automáticamente

Sin embargo, para las pruebas unitarias, es necesario validar los métodos de logueo de las clases Cuenta\_usuario y Cuenta\_administrador:

Interfaz de usuario gráfica, Escala de tiempo

Descripción generada automáticamente

Como se puede apreciar en la imagen, cada clase tendrá sus propias entidades de pruebas unitarias.

|  |
| --- |
| Cuenta\_administradorTest |
| **package** com.example.safetravelmap.entities;  **import** junit.framework.TestCase;  **import** org.junit.Before;  **import** org.junit.Test;  **public class** Cuenta\_administradorTest **extends** TestCase {  **private** Cuenta\_administrador **cuenta\_administrador** = **new** Cuenta\_administrador();  @Before  **public void** setup(){  **cuenta\_administrador** = **new** Cuenta\_administrador();  }  @Test  **public void** testConsultar\_usuario() **throws** Exception{  *assertEquals*(**true**, **cuenta\_administrador**.consultar\_administrador(**"test@gmail.com"**, **true**));  }  } |
| Cuenta\_usuarioTest |
| **package** com.example.safetravelmap.entities;  **import** junit.framework.TestCase;  **import** org.junit.Before;  **import** org.junit.Test;  **import static** org.junit.Assert.*assertEquals*;  **public class** Cuenta\_usuarioTest **extends** TestCase {  **private** Cuenta\_usuario **cuenta\_usuario** = **new** Cuenta\_usuario();  @Before  **public void** setup(){  **cuenta\_usuario** = **new** Cuenta\_usuario();  }  @Test  **public void** testConsultar\_usuario() **throws** Exception{  *assertEquals*(**true**, **cuenta\_usuario**.consultar\_usuario(**"test@gmail.com"**, **true**));  }  } |

Realizando pruebas unitarias en el login de la cuenta de usuario:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

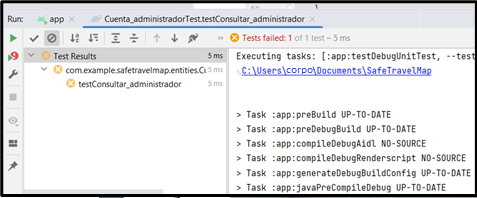
Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Realizando pruebas unitarias en el login de la cuenta de administrador (en este caso no obtendremos el resultado esperado, ya que el correo no corresponde a ninguno registrado en el storage de Firebase):

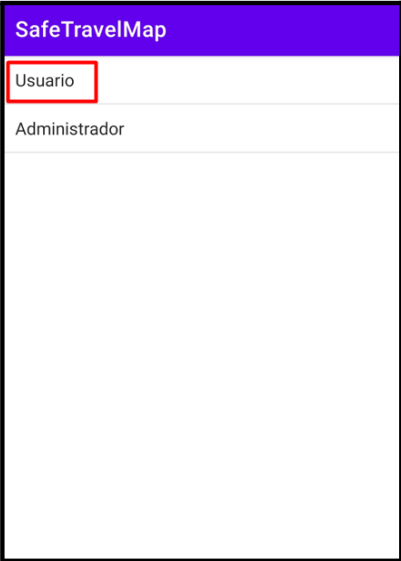
Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

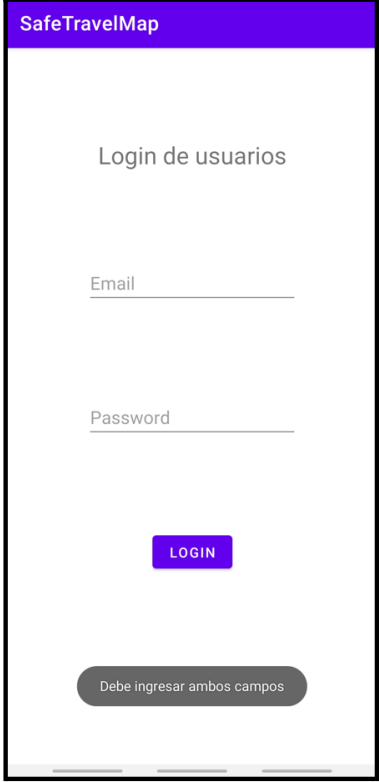


Pruebas de funcionalidad

1. Se selecciona la opción de usuario

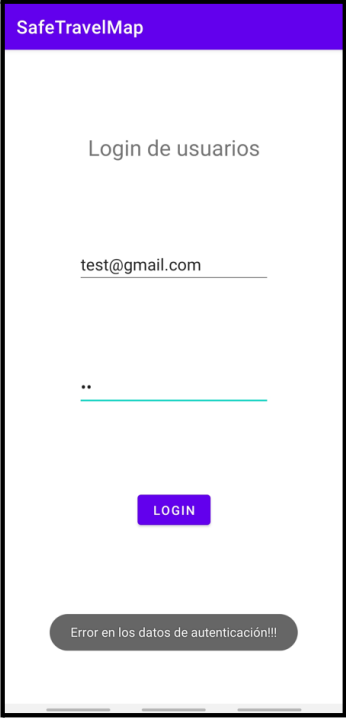


Se intenta loguear sin ingresar los datos.



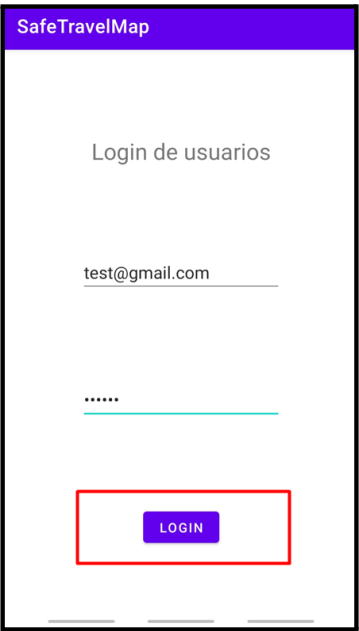
Despliega el mensaje “Debe ingresar ambos campos”

1. Se procede a ingresar mal la password



Desplegando así el mensaje “Error en los datos de autenticación”

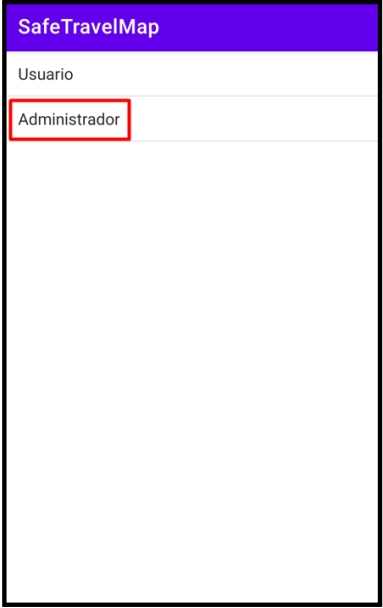
1. Ahora se ingresan los datos correctos y validamos la autenticación en FireBase:



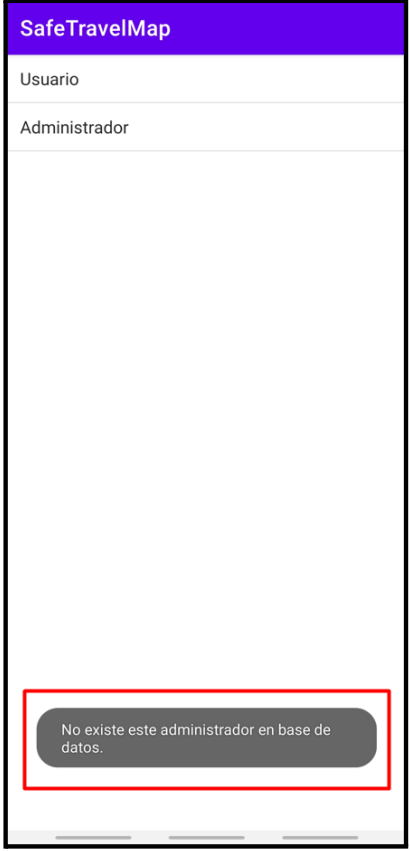
Ingresamos correctamente al sistema.



1. Intentamos ingresar con el rol de “Administrador”



Como no existe un administrador con estos datos, obtenemos el mensaje de error correspondiente.



###### 3.4.1.5 documentación pruebas de aceptación

Documentación de análisis

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Enunciado | de | la | Historia | Criterios de | aceptación |  |
| Identificador  (ID)de la  historia | Rol | Característica/  Funcionalidad | Razón/Resultado | Número (#)  de Escenario | Criterios de aceptación título | contexto | evento |
| SAF-2 | Yo como usuario/  administrador | quiero que todas las personas que usen la aplicación estén registradas | para dar a conocer a quien resuelve las alertas que son personas reales las que se están siendo beneficiadas | |  | | --- | | 1 |  |  | | --- | | 2 |  |  | | --- | | 3 |  |  | | --- | | 4 | | |  | | --- | | Solo un usuario o admin con cuenta registrada podrá acceder a la aplicación móvil. |  |  | | --- | | Al iniciar sesión el usuario o admin debe ingresar un correo electrónico valido. |  |  | | --- | | Usuario o admin debe ingresar una clave válida. |  |  | | --- | | Usuario o admin deberán ingresar correo electrónico y clave válida. | | |  | | --- | | Usuario o admin  no registrado intenta hacer uso de la aplicación móvil. |  |  | | --- | | Usuario o admin no completa o ingresa un correo electrónico incorrecto. |  |  | | --- | | Usuario o admin no ingresan clave o ingresan una clave incorrecta. |  |  | | --- | | Usuario o admin ingresan datos de forma correcta. | | |  | | --- | | La aplicación móvil le solicita al usuario o administrador iniciar sesión. |  |  | | --- | | Aplicación móvil solicita al usuario o admin ingresar correo electrónico valido. |  |  | | --- | | Aplicación móvil solicita ingresar una clave válida. |  |  | | --- | | Aplicación móvil despliega mensaje inicio de sesión correcto. | |
|  |  |  |  |  |  |  |  |

###### 3.4.1.6 Riesgo

Planificación de riesgos

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Identificador  de historias de usuario. | Cantidad de riesgos encontrados. | Identificación de riesgo. | Impacto. | Probabilidad de ocurrencia. | PxI | Valoración. | Color |
|  | 1 | a | 3 | 4 | 12 | Significativo y probable |  |
| SAF2 | 2 | b | 3 | 3 | 9 | Significativo y moderado |  |
|  | 3 | c | 5 | 5 | 25 | Severo y casi seguro |  |

Identificación de los riesgos.

Los riesgos encontrados en esta primera etapa consisten en:

1. La falta de experiencia por parte del desarrollador en las herramientas que se han elegido para el desarrollo y gestión del proyecto pueden retrasar la planificación, esto tiene una probabilidad de ocurrir de cuatro en la matriz de riesgos y se le considera de un riesgo significativo.
2. Existe un riesgo técnico relacionado directamente al estado del notebook, que no se encuentra en óptimas condiciones para llevar a cabo el desarrollo del proyecto, esto también podría atrasar la planificación su probabilidad de ocurrir es de tres y se le considera un riesgo significativo.
3. Problemas de salud del desarrollador pueden representar un riesgo en el avance del proyecto, su probabilidad de ocurrir es de cinco al igual que la severidad.

Mitigación del riesgo.

1. Se estima que el entendimiento para el correcto uso de cada de las herramientas y lenguajes usados tomara un tiempo mínimo de una semana por cada tema. Se comenzará a ver tutorías por internet donde se explique el uso de cada una para mitigar el riesgo.
2. Durante el desarrollo del proyecto se planea comprar un computador en caso de que el notebook deje de funcionar, esto tomara un tiempo de tres a cuatro meses.
3. El desarrollador sufre de agudos dolores de cabeza producto de secuelas que quedaron luego de un derrame cerebral, se estima que si el dolor es muy fuerte su recuperación podría tardar una semana.

## 3.4.2 Sprint dos

* El segundo sprint consta del desarrollo de la función que permite subir la información necesaria para una alerta, estas historias corresponden a la épica de: “Funcionamiento básico” y “Filtrado y comunicación ”, estas historias son: “SAF3, SAF4, SAF5”, para captura del riesgo, descripción y clasificación. Su priorización es Must Have y Shoul Have lo que ubica a este sprint como parte del producto mínimo viable solicitado por el usuario con un puntaje de 21.

###### 3.4.2.1 Diseño sprint dos

Diagrama de clases conceptual

El diagrama de clases conceptual que comprende el sprint dos muestra en la figura ¿?. Todas las clases que se integran, correspondientes a las actuales funcionalidades implementadas.

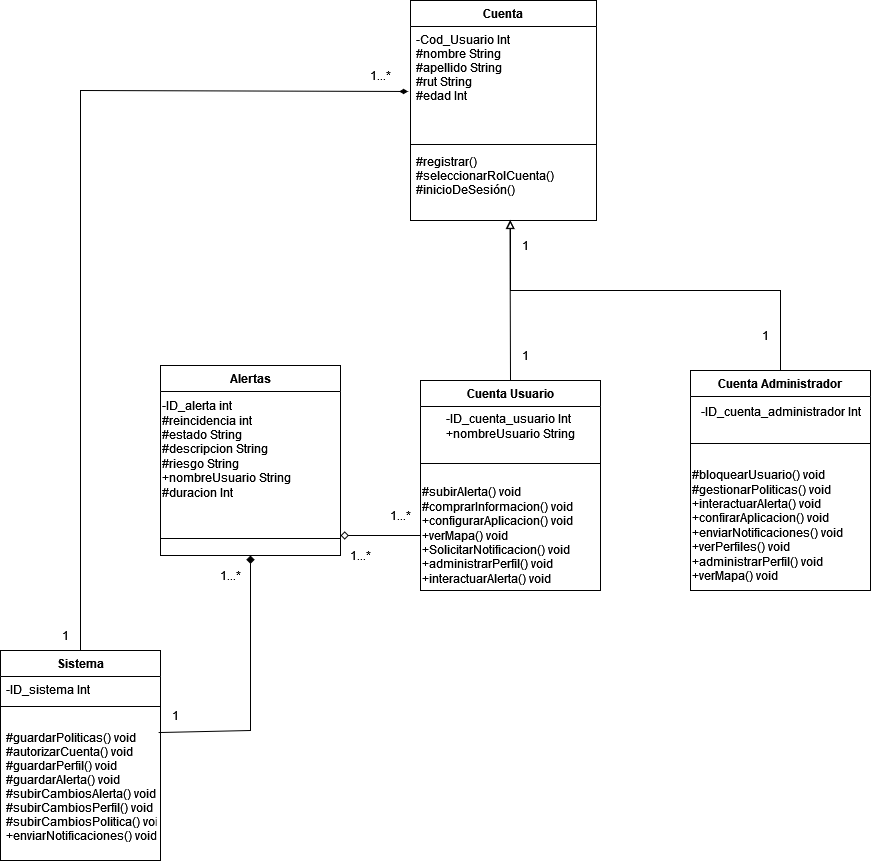
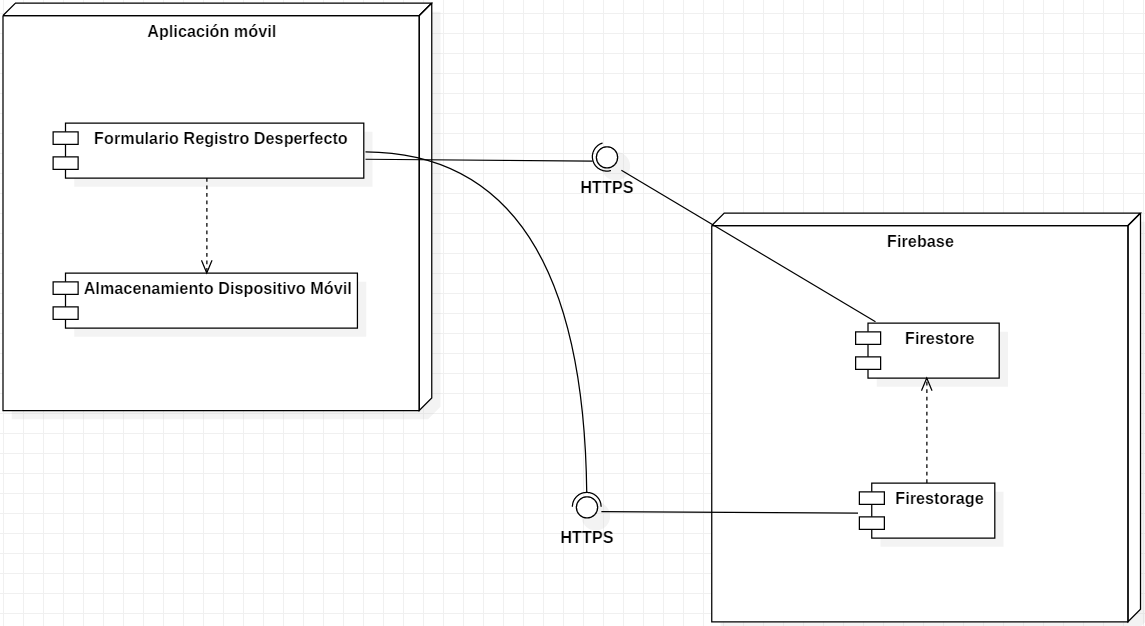


Diagrama de componentes



1. Aplicación móvil: Este es el punto de entrada principal de la aplicación. Aquí se solicita al usuario que introduzca sus credenciales para el inicio de sesión.

* Formulario de registro de desperfectos: Este formulario desempeña la función de recolectar información relevante acerca de diversos desperfectos. Entre los datos que recaba se incluye una descripción detallada del daño, una evaluación de su severidad (es decir, si es grave o no), y la opción de capturar una imagen del mismo. Una vez obtenida la fotografía, esta se almacena de manera temporal en la memoria interna del dispositivo móvil. Posteriormente, el formulario permite subir la información recopilada, incluida la fotografía, a Firebase. En este proceso, los datos descriptivos del desperfecto se registran en Firestore, mientras que la imagen se almacena de forma segura en Firebase Cloud Storage.
* Almacenamiento del dispositivo: La imagen capturada se guarda inicialmente en la memoria interna del dispositivo móvil. Posteriormente, se accede a ella desde esta ubicación y se procede a subirla a Firebase, haciendo uso específico del servicio Firebase Cloud Storage para su almacenamiento seguro y eficiente.

1. Firebase: Plataforma en la nube que proporciona los servicios necesarios para el almacenamiento de datos y de archivos como fotografías.

* Firestore: Este servicio no solo se encarga de almacenar la información detallada de los desperfectos, sino que también establece una vinculación precisa con la correspondiente imagen almacenada en Firebase Cloud Storage.
* Firebase Cloud Storage: Este servicio se emplea específicamente para el almacenamiento de las fotografías de los desperfectos. Además, ofrece la versatilidad de poder almacenar no solo imágenes, sino también vídeos, y brinda soporte para una amplia variedad de formatos de archivos.

###### 3.4.2.2 Tareas sprint dos Jira

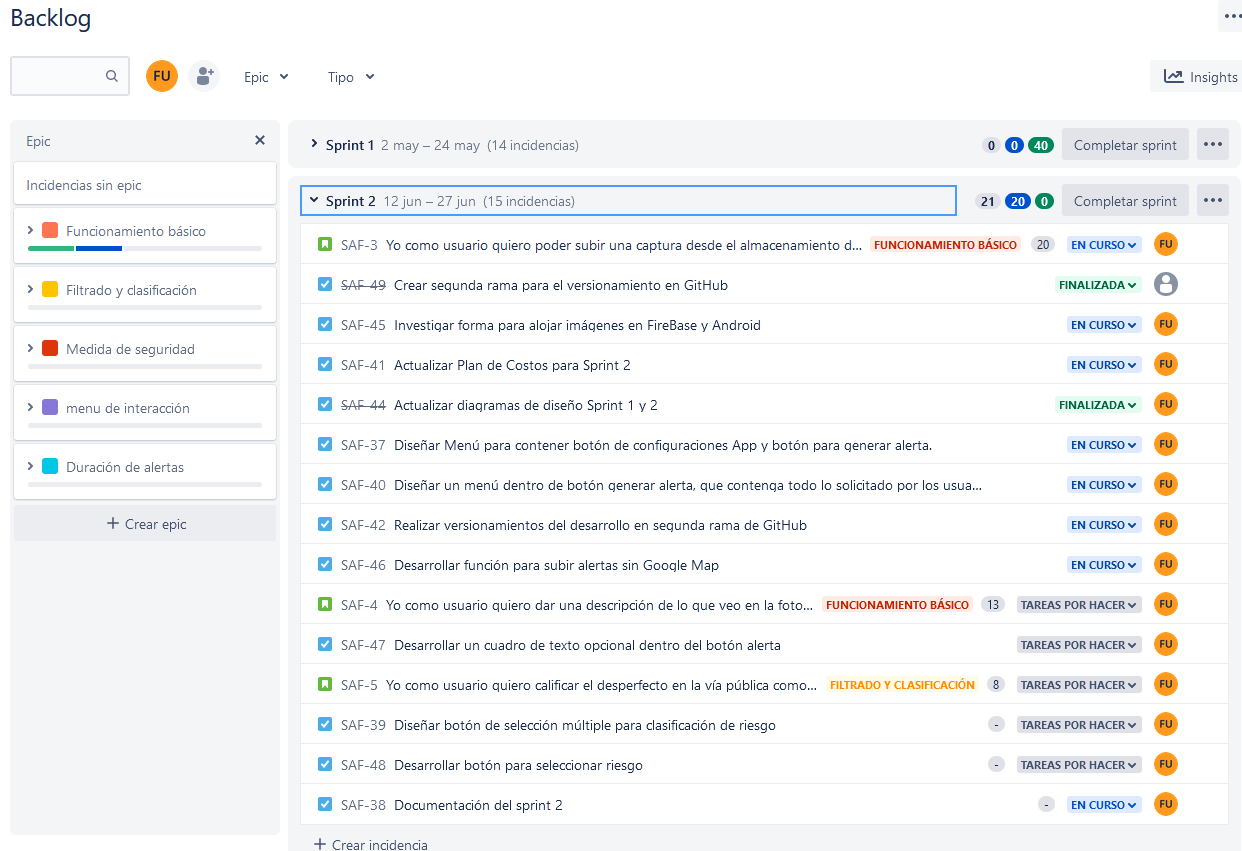


Figura ¿?: En esta figura se muestra la historia de usuario “SAF-3, SAF-4, SAF-5” trabajada en Jira, todas las tareas por las cuales está compuesta la historia, se puede ver algunas tareas ya finalizadas y otras en curso, todas realizadas por el desarrollador Fabio Urrea. También las épicas que dieron origen a las historias de usuario y la duración del segundo sprint que va desde el primero de junio al veintisiete de junio.

###### 3.4.2.3 Versionamiento sprint dos GitHub

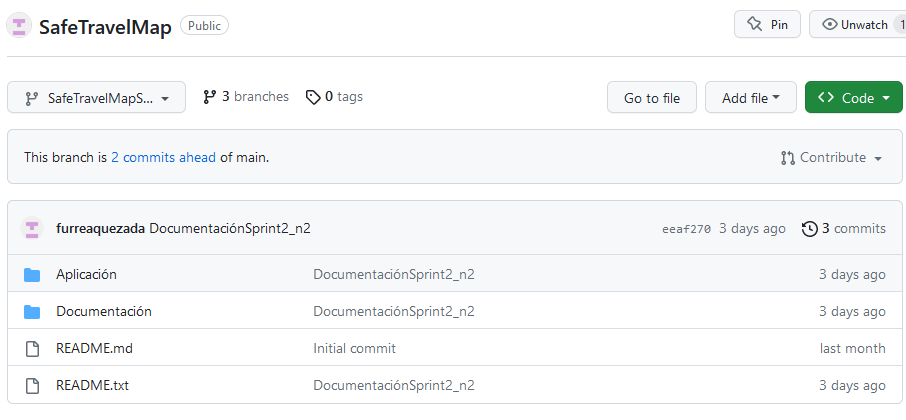


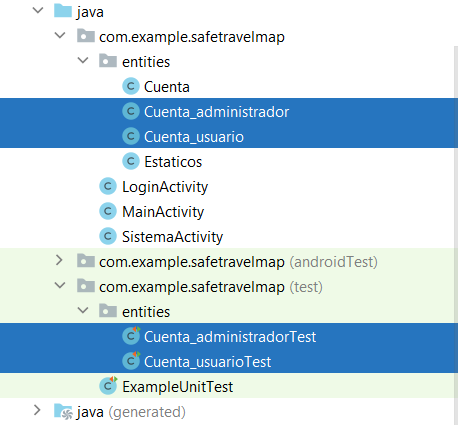
Figura 11: evidencia del versionamiento del sprint dos.

###### 3.4.2.4 Pruebas sprint dos

Pruebas unitarias.

Dentro del sistema, correspondiente a una aplicación móvil, se han desarrollado varias clases que extienden de AppCompat Activity. El Activity que es de interés para las primeras pruebas de unidad corresponde a SistemaActivity:

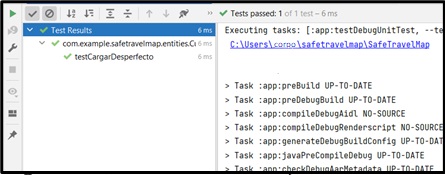
Para las pruebas unitarias, es necesario validar el registro del desperfecto.



Cada clase estará asociada a sus propias entidades de pruebas unitarias.

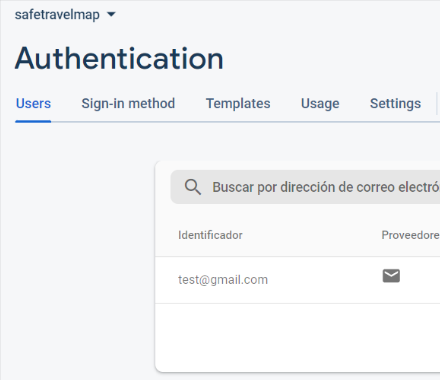
|  |
| --- |
| Cuenta\_administradorTest |
| **package** com.example.safetravelmap.entities;  **import** junit.framework.TestCase;  **import** org.junit.Before;  **import** org.junit.Test;  **public class** Cuenta\_administradorTest **extends** TestCase {  **private** Cuenta\_administrador **cuenta\_administrador** = **new** Cuenta\_administrador();  @Before  **public void** setup(){  **cuenta\_administrador** = **new** Cuenta\_administrador();  }  @Test  **public void** testConsultar\_administrador() **throws** Exception{  *assertEquals*(**true**, **cuenta\_administrador**.consultar\_administrador(**"test@gmail.com"**, **true**));  }  @Test  **public void** testCargarDesperfecto() **throws** Exception{  *assertEquals*(**true**, **cuenta\_administrador**.cargarDesperfecto());  }  } |
| Cuenta\_usuarioTest |
| **package** com.example.safetravelmap.entities;  **import** junit.framework.TestCase;  **import** org.junit.Before;  **import** org.junit.Test;  **import static** org.junit.Assert.*assertEquals*;  **public class** Cuenta\_usuarioTest **extends** TestCase {  **private** Cuenta\_usuario **cuenta\_usuario** = **new** Cuenta\_usuario();  @Before  **public void** setup(){  **cuenta\_usuario** = **new** Cuenta\_usuario();  }  @Test  **public void** testConsultar\_usuario() **throws** Exception{  *assertEquals*(**true**, **cuenta\_usuario**.consultar\_usuario(**"test@gmail.com"**, **true**));  }  @Test  **public void** testCargarDesperfecto() **throws** Exception{  *assertEquals*(**true**, **cuenta\_usuario**.cargarDesperfecto());  }  } |

Ejecutando pruebas unitarias:

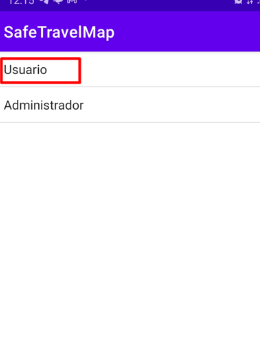


**Pruebas de funcionalidad**

Se realizan las pruebas funcionales por medio de la aplicación.

****

Seleccionamos la opción de login por el usuario.

****

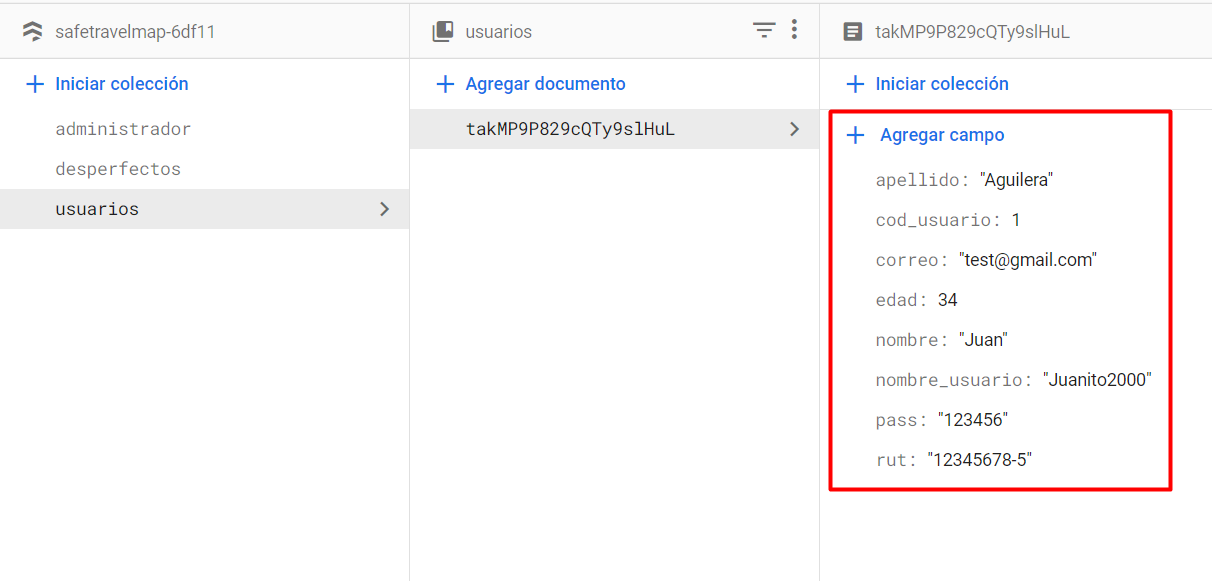
Ingresamos los datos de usuario.

****

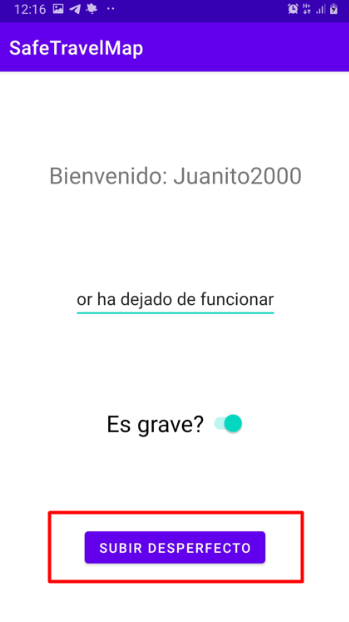
Se rellena los datos con el desperfecto.

****

Efectivamente este usuario esta registrado en la base de datos de FireStore.

****

Una vez dentro se describe el desperfecto, se indica si es o no grave y se captura una fotografía del mismo.



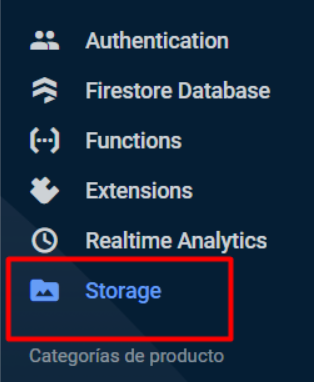
Tras un breve lapso de tiempo, que variará según la velocidad de la conexión a Internet, tanto la imagen como la información relativa al desperfecto serán subidas y almacenadas de manera permanente en la plataforma Firebase. Para lograr esto, se utiliza FireStorage para el almacenamiento del documento gráfico y FireBase como el sistema de gestión de la base de datos que contiene el registro del desperfecto.



Este es el código de la imagen registrada.



Se comprueba en FireStorage



Efectivamente, ahí está la imagen asociada al desperfecto, almacenada en la plataforma de FireBase.

###### 3.4.2.5 Documentación pruebas de aceptación

Documentación de análisis

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Enunciado | de | la | Historia | Criterios de | aceptación |  |
| Identificador  (ID)de la  historia | Rol | Característica/  Funcionalidad | Razón/Resultado | Número (#)  de Escenario | Criterios de aceptación título | contexto | evento |
| SAF-3 | Yo como usuario | quiero poder subir una captura desde el almacenamiento de mi celular o tomar una foto del desperfecto en la vía pública | para dejar evidencia fotográfica del peligro encontrado. | |  | | --- | | 1 |  |  | | --- | | 2 | | |  | | --- | | El usuario podrá tomar una foto para evidenciar un desperfecto en la vía pública. |  |  | | --- | | La foto del usuario debe almacenarse en la base de datos. | | |  | | --- | | Usuario abre la aplicación móvil, inicia sesión, sube un reporte y para ello toma una foto. |  |  | | --- | | Usuario toma la foto para generar el reporte. | | |  | | --- | | Al tomar la foto se genera el reporte |  |  | | --- | | La foto es almacenada en FireBase con un ID que la identifica y en FireStorage. | |
|  |  |  |  |  |  |  |  |
| SAF-4 | Yo como usuario | quiero dar una descripción de lo que veo en la fotografía que estoy dejando en la alerta | para dar más detalles del desperfecto en la vía pública. | |  | | --- | | 1 |  |  | | --- | | 2 | | |  | | --- | | Usuario podrá dar una descripción opcional de la fotografía sacada | | Descripción entregada por el usuario debe almacenarse en la base de datos. | | |  | | --- | | Usuario abre la aplicación móvil, inicia sesión, antes de generar el reporte da una descripción de lo que ve en la foto. | | Usuario da una descripción de lo que observa, | | |  | | --- | | La descripción entregada se guarda en el reporte subido | | Al generar el reporte esto se almacena junto a los demás datos del reporte em FireBase. | |
|  |  |  |  |  |  |  |  |
| SAF-5 | Yo como usuario | quiero calificar el desperfecto en la vía pública como una alerta grave o baja | para que otros usuarios eviten pasar por el lugar si es grave. | |  | | --- | | 1 |  |  | | --- | | 2 | | |  | | --- | | Usuario podrá seleccionar si la alerta es grave o no. | | La selección del usuario será almacenada en la base de datos junto a los demás datos. | | |  | | --- | | Usuario abre la aplicación móvil, inicia sesión, antes de generar el reporte indica si el reporte es grave o no. | | Usuario marca como grave el reporte. | | |  | | --- | | Al marcar el botón se dará por entendido que la alerta es grave y si esto no está seleccionado será alerta baja | | Al generarse el reporte la clasificación de este se almacena en FireBase. | |

###### 3.4.2.6 Riesgos

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Identificador  de historias de usuario. | Cantidad de riesgos encontrados. | Identificación de riesgo. | Impacto. | Probabilidad de ocurrencia. | PxI | Valoración. | Color |
| SAF-3 | 1 | a | 5 | 4 | 20 | Casi seguro y es severo |  |
| SAF-4 | 2 | b | 4 | 3 | 12 | Moderadamente mayor |  |
| SAF-5 | 3 | c | 4 | 2 | 8 | Poco probable mayor |  |
|  | 4 | d | 5 | 4 | 20 | Probable y severo |  |
|  | 5 | e | 5 | 2 | 10 | Poco probable y severo |  |

Identificación de los riesgos.

1. Los variados permisos que requiere la base de datos trabajada en FireBase para alojar imágenes puede atrasar el desarrollo de la planificación del proyecto.
2. La constante actualización de las reglas de seguridad para el uso de FireBase puede llegar rechazar las solicitudes de los clientes si estas no son implementadas correctamente o surge una actualización.
3. Si el desarrollador se enferma puede llegar a causar un atraso en alguna entrega programada.
4. La implementación de nuevas funciones que el programador desconoce como implementar o desarrollar puede llegar a atrasar el desarrollo de la aplicación móvil debido al tiempo de investigación requerido.
5. Pese a la planificación de costo el uso de herramientas nuevas que facilita GCP puede resultar en un gasto desmedido que escape de lo planificado y de como resultado el rechazo de las solicitudes de los clientes a la base de datos y la suspensión de los servicios de GCP.

Mitigación del riesgo.

1. Revisar periódicamente los permisos faltantes que muestren las alertas de FireBase.
2. Revisar periódicamente la actualización de las reglas de seguridad que saltan en las alertas de FireBase.
3. Se hablará con la Scrum Máster en caso de que alguna entrega se vea afectada.
4. En caso de atrasarse en el desarrollo se le informara al Scrum Máster para tomar las medidas correspondientes.
5. Evaluar los costos antes de usar una nueva herramienta que recomiende GCP.

Contingencia de riesgo

e) Si el costo del uso de las herramientas en GCP llega a exceder lo presupuestado se exportará la base de datos para evitar su perdida, se dará de baja la cuenta alumno con la cual fue creada, se creará una nueva cuenta alumno y se procederá a importar la base de datos a FireBase para evitar la pérdida de datos.

## 3.5 PLAN DE CALIDAD

Siguiendo el estándar ISO/IEC 9126: 2001 se definirá un marco conceptual para el modelo de calidad de este producto de software que deberá cumplir con las siguientes características:

* Funcionalidad
* Confiabilidad
* Usabilidad
* Eficiencia
* Mantenibilidad
* Portabilidad
* Satisfacción

#### Funcionalidad

Este proyecto se va a enfocar en cumplir e implementar poco a poco durante el desarrollo de cada sprint las características como la funcionalidad, medidos por el cumplimiento de los criterios de aceptación para cada una de las historias de usuario registradas en este documento y se modificarán algunas de ellas para no incumplir con la ley de protección de datos número 19.628. Es de suma importancia estar dentro de los parámetros de esta ley, ya que la aplicación Safe Travel Map se dedicará a la venta de información para aquellos stakeholders que estén interesados en solucionar las alertas que serán proporcionadas por nuestros usuarios.

Por lo dicho anteriormente historias de usuario como “SAF16” donde señala el usuario quiere que las alertas lleven el nombre y RUT de quien está publicando la alerta, serán adaptadas, en su defecto el usuario al momento de crear la cuenta para que esta sea única deberá completar los campos de Rut, contraseña y un nombre de usuario para proteger dichos datos el cual se podrá visualizar al momento que otros usuarios deseen evaluar si la alerta es verifica o es falsa.

#### Usabilidad

A lo largo del término de los sprint de este proyecto se le ira entregando partes funcionales a los usuarios del focus group para medir que tan intuitiva es la aplicación móvil que se está desarrollando y se ira modificando de acuerdo con la experiencia de usuario.

#### Satisfacción

Para llegar a cumplir esta característica de calidad será necesario cumplir con todos los requerimientos solicitados por las personas que participaron en el focus group, se medirá la efectividad de esta medida una vez finalice el proyecto y se demuestre que el objetivó inicial para el que fue creada la aplicación tuvo coherencia con el producto final desarrollado.3.6 PLAN DE COSTOS

El plan de costos fue subido a GitHub en conjunto con el desarrollo de la aplicación y la documentación del proyecto, el documento desarrollado será modificado cada vez que se realice un sprint para ir incluyendo los costos generados y haciéndole las modificaciones pertinentes.

## 3.6.1 Costos sprint uno

Interfaz de usuario gráfica, Aplicación, Tabla, Excel

Descripción generada automáticamente

Figura 14: Muestra el plan de costos del proyecto subido en GitHub.

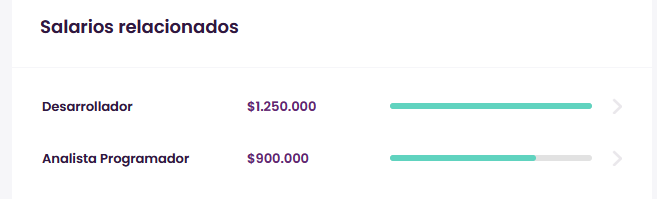
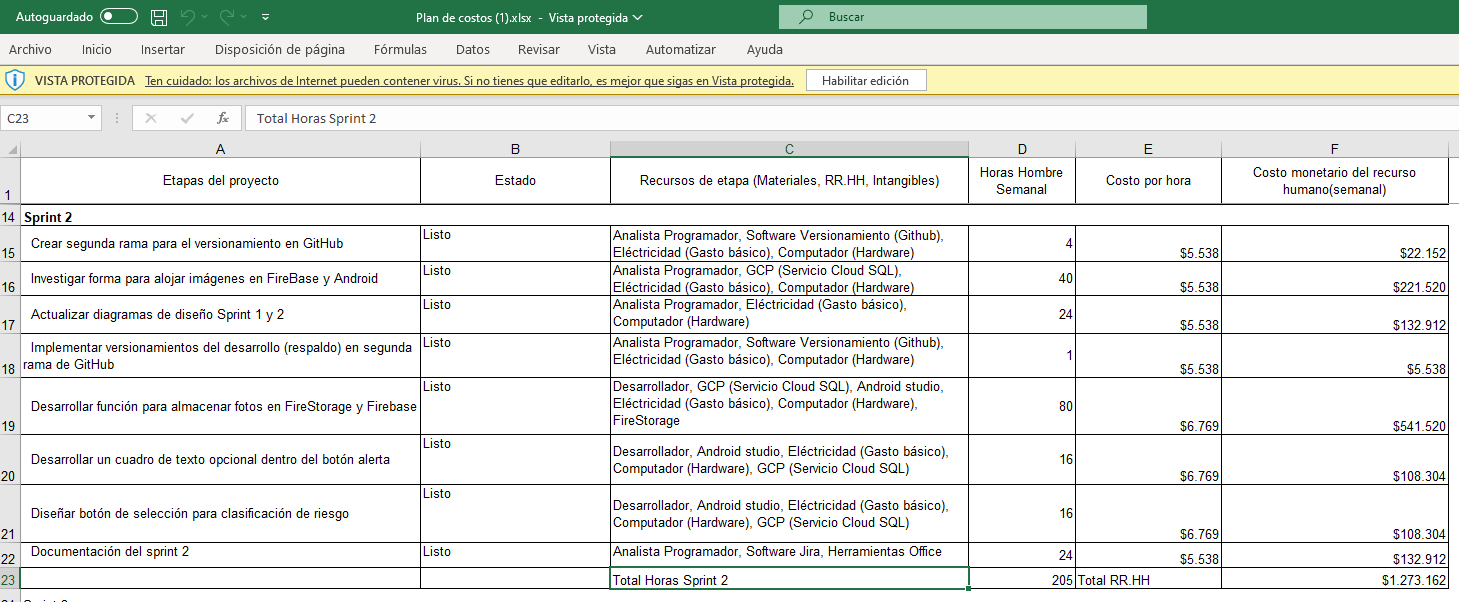
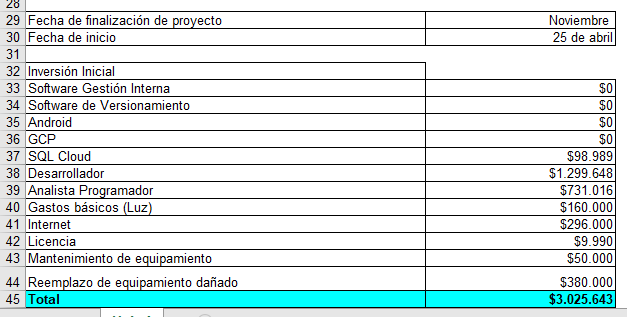


Figura 15: La información recopilada de los salarios para completar el plan de costos fue sacado de la página: “https://cl.talent.com/salary?job=desarrollador”

## 3.6.1 Costos sprint dos





## 3.7 PLAN DE PRUEBAS

Para elaborar el plan de pruebas se tomará una plantilla para proyectos agiles del libro “Agile Testing” cuyas autoras son Lisa Crispin y Janet Gregory.

|  |
| --- |
| Proyecto Safe Travel Map App |
| Plan de pruebas |
| Preparado por: Fabio Ismael Urrea Quezada |
|  |
| Introducción. |
| El plan de pruebas está diseñado como una línea de base para identificar lo que se considera dentro y fuera del alcance de las pruebas y cuáles son los riesgos y suposiciones. |
|  |
| Recursos. |
| |  |  | | --- | --- | | Tester | % Participación | | Fabio Urrea | 100% | |
|  |
| Alcance |
| Cada sprint tendrá sus propias pruebas unitarias, de aceptación, de funcionamiento e integración. Los sprints están pensados para durar entre una a cuatro semanas por lo cual por cada sprint se dejará la última semana antes de su entrega para efectuar todas las pruebas de software con el fin de evaluar y verificar que el producto que se está desarrollando cumple con lo que debería hacer, es decir que la mayor parte de las pruebas serán realizadas durante la iteración del sprint y luego en el reléase se le pasara una parte del software funcionando al usuario para que nos entregue su feedback y se planifiquen los cambios que se deben integrar o hacer a la aplicación móvil. |
|  |
| Fuera del alcance. |
| Pruebas que no se encuentren especificadas en el alcance no serán descritas en este documento debido a que muchas de ellas requieren que la aplicación este terminada en su totalidad para recién comenzar a hacer las y van más allá de la planificación de este proyecto. |
|  |
| Pruebas de aceptación. |
| Las siguientes pruebas serán ordenadas según la plantilla que se encuentra en la sección de documentación de análisis. Los usuarios que solicitarán estas pruebas se encuentran en el focus group que fue usado en un comienzo para todo el levantamiento de información. |
|  |
| Pruebas unitarias. |
| Se harán a nivel de código para probar los métodos dentro de cada función de la aplicación móvil, los resultados esperados pueden variar un poco debido a que la aplicación móvil será testeada sin la conexión a FireBase que corresponde a nuestra base de datos. |
|  |
| Pruebas de funcionamiento. |
| Serán validaciones que se harán en el código de la aplicación móvil con el fin de que el sistema haga lo que tenga que hacer y que no haga lo que no tenga que hacer en base a los requerimientos solicitados por los usuarios. |
|  |
| Pruebas de integración. |
| Comprueba que todas las pruebas unitarias pueden funcionar de forma correcta en conjunto. |
|  |
| Infraestructura |
| En la parte de hardware se necesitará un celular de gama baja, media o alta para emular la aplicación móvil.  El software requerido para estas pruebas será Junit, Android Studio, Jira, GitHub, FireBase, GCP, entre otros. |
|  |
| Suposiciones. |
| Todas las pruebas estarán desarrolladas y documentadas antes d finalizar el entregable. |
|  |

Riesgos.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Riesgos | Probabilidad | Impacto | PXI (severidad) | Plan de mitigación |
| 1 | Que las pruebas no se cumplan a tiempo. | 3 | 4 | 12 | Incluir las pruebas faltantes en la siguiente iteración, antes de llegar a la última semana de pruebas. Rellenar plantilla de gestión de cambios. |
| 2 | La emulación de la aplicación móvil puede desconfigurarse. | 2 | 5 | 10 | Solicitar ayuda para reconfigurar la emulación, buscar soluciones por internet, encontrar otro pc para emular la aplicación, emular aplicación directamente en el celular. |
| 3 | Retraso en la documentación del entregable. | 4 | 3 | 12 | Añadir documentación faltante en la siguiente iteración.  Rellenar plantilla de gestión de cambios. |
| 4 | Mala estimación de las tareas de desarrollo. | 2 | 4 | 8 | Revisar planificación de proyecto semanalmente, reportar problemas a Scrum master en caso de retraso. |

## 3.8 PLAN DE GESTIÓN DE LA CONFIGURACIÓN

Este plan se llevará a cabo mediante la utilización de GitHub donde se creará un repositorio, el manejo de este repositorio posee las siguientes características:

* Cada sprint tendrá su propia rama donde se irán subiendo los versionamientos correspondientes.
* Los versionamientos de la aplicación se irán haciendo semanalmente, debido al gran número de reajustes que se hacen durante la semana.
* En cada versionamiento se encontrará el plan costos, la aplicación, la documentación y otra información relevante para el desarrollo del proyecto.
* El repositorio será público y se podrá encontrar en la siguiente URL:

“https://github.com/furreaquezada/SafeTravelMap.git”

## 3.9 PLAN DE GESTIÓN DE CAMBIOS

Después de realizar el sprint review y antes de comenzar el siguiente sprint planing se realizará el sprint retrospective donde se analizará todo lo desarrollado en el sprint, identificando así que mejoras deben efectuarse o incluirse en el siguiente sprint junto a las nuevas historias de usuario que serán desarrolladas, cualquier cambio será agregado a la documentación del siguiente sprint.

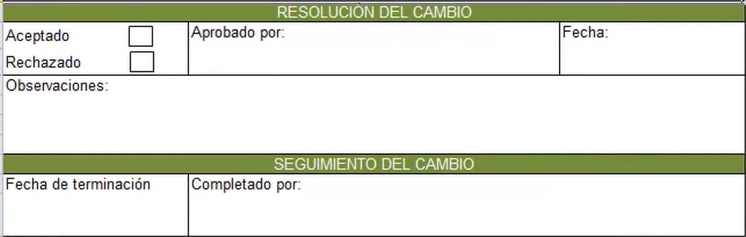
Cada cambio que se planee incluir en el siguiente sprint cumplirá con los siguientes “debe” de la norma ISO 9001:2015 :

1. El propósito de los cambios y sus consecuencias potenciales.
2. La integridad del sistema de gestión de la calidad.
3. La disponibilidad de recursos.
4. La asignación o reasignación de responsabilidades y autoridades.

La siguiente plantilla permitirá cumplir con los requisitos que la ISO 9001:2015 requiere para aprobar la solicitud de un cambio.







A lo largo de la plantilla se pueden ver cuadros de colores que indican donde se está cumpliendo con los puntos que la norma ISO 9001:2015 especifica, siendo así:

* Punto “a” color rojo.
* Punto “b” color naranjo.
* Punto “c” color verde.
* Punto “d” color celeste.

Al cumplir con la plantilla el cambio será incluido en Jira como una historia de usuario más con su debido identificador de historia, las tareas correspondientes y agregado a una épica llamada cambios.

## 3.10 PLAN DE GESTIÓN DE RIESGOS

La gestión de riesgo para este proyecto se creará poniendo de base un análisis obtenido del Instituto de gestión de proyectos (PMI) que se encuentra dentro de la sexta edición de la guía del PMBOK. Dicha gestión posee procesos que se subdividen en etapas como los son:

1. Etapa de planificación:

* Planificar la gestión de los riesgos.
* Identificar los riesgos.
* Realizar análisis cualitativo de riesgos.
* Realizar análisis cuantitativo de riesgos.
* Planificar la respuesta a los riesgos.

1. Ejecución:

* Implementar la respuesta a los riesgos.

1. Control:

* Controlar los riesgos.

Para el siguiente análisis de riesgos se aplicó la metodología de estructura de desglose de trabajo o EDT con el fin de entregar una gestión de riesgo por cada entregable.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Historias de usuario. | Cantidad de riesgos encontrados. | Identificación de riesgo. | Impacto. | Probabilidad de ocurrencia. | PxI | Valoración. | Color |
|  | 1 |  |  |  |  |  |  |
|  | 2 |  |  |  |  |  |  |
|  | 3 |  |  |  |  |  |  |

Tabla 7.

La tabla siete Ordena todos los datos de los riesgos correspondientes al sprint que se está desarrollando, junto a la identificación de él o los riesgos se entregará una mitigación de estos que actuará como un plan de respuesta a estos riesgos o un plan de contingencia. El análisis cualitativo incluye la probabilidad, el impacto y la multiplicación de ambos factores lo que indica que números grandes representarán un riesgo que debe ser atendido en su brevedad, ya que si se llega a materializar generará costos importantes.

Tabla

Descripción generada automáticamente

Figura 16: Matriz de riesgo que ubica en que casilla se encuentra el riesgo identificado según su impacto, la probabilidad de ocurrencia y la valoración.

## 3.11 AMBIENTE DE DESARROLLO, PRUEBAS Y PRUDUCCIÓN

Ambiente de desarrollo: software Android Studio versión Flamingo 2022.2.1.20-windows.

Ambiente de pruebas: software Junit 5 versión 5.8.2

Ambiente de producción: Celular marca Xiaomi, modelo POCO X4 Pro 5G, versión de Android 12.

## 3.12 DESARROLLO DE LA APLICACIÓN MÓVIL

## 3.12.1 Sprint uno

El proyecto consta de un paquete que almacena entidades tales como la clase padre “Cuenta”, de las cuales heredan “Cuenta\_usuario” y “Cuenta\_administrador”, las cuales serán de utilidad para gestionar el rol logueado:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Las demás clases corresponden a las “Activities” que gestionan cada una de las vistas con las que interactúa el usuario. Cada “Activitiy” posee una vista codificada en un archivo de extensión XML, para manipular de manera sencilla los controles necesarios.

Por otro lado, para la integración con Google Firebase, se ha integrado el JSON de Google que permite acceder a las librerías y APIs necesarias:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Creación de la aplicación con FireBase

Se creo la aplicación a través de la plataforma de Google FireBase.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Se utilizaron las herramientas de autentificación y base de datos no relacional.

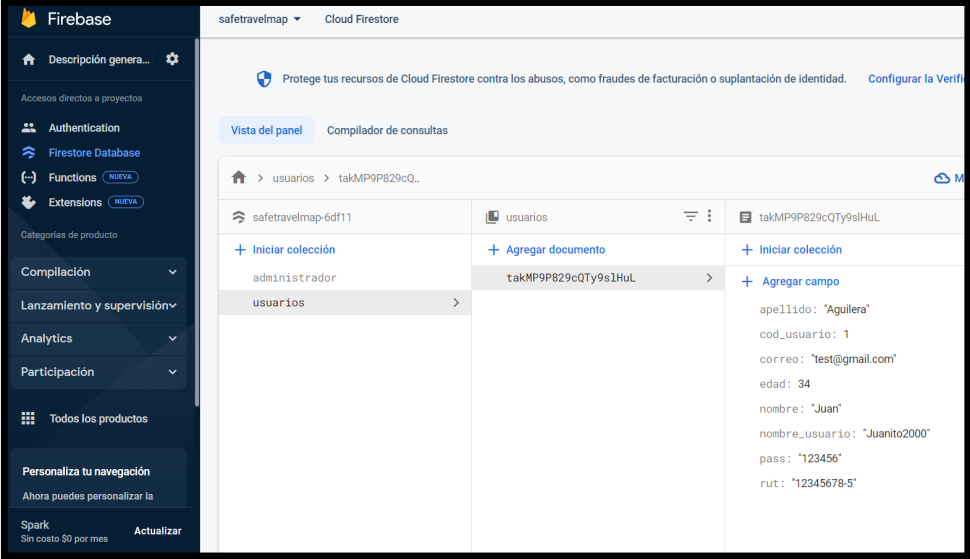
Interfaz de usuario gráfica, Aplicación, Teams

Descripción generada automáticamente

Dentro de la autenticación se ha utilizado el logueo por correo y password, se ha creado un correo de pruebas más una password de pruebas.

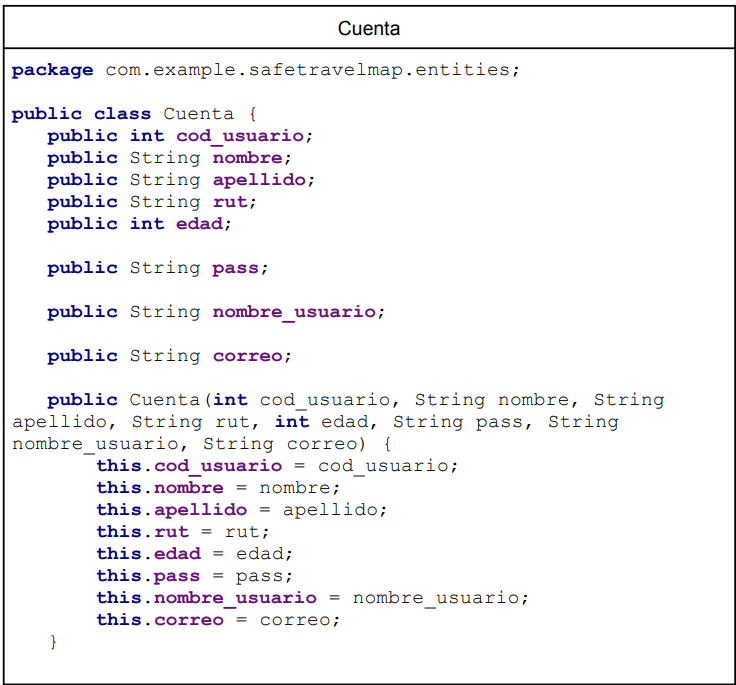


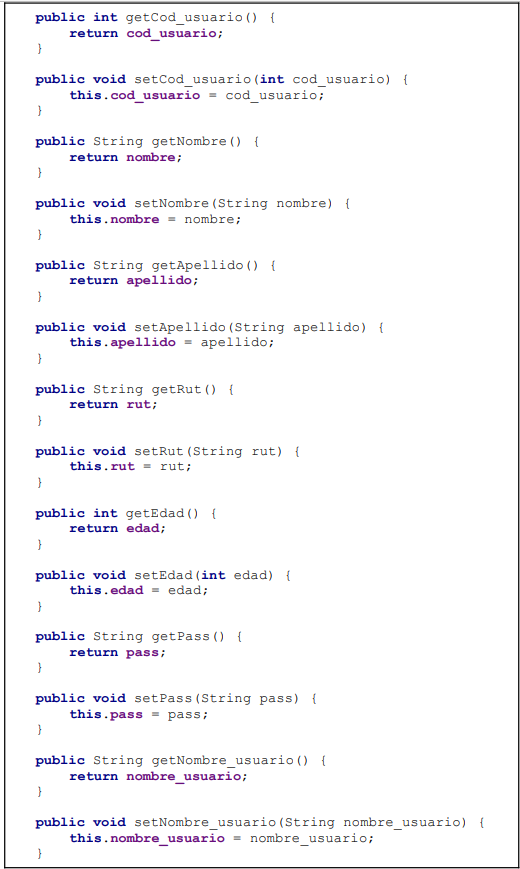
Y en la base de datos no relacional, se han creado dos colecciones donde una representa a los usuarios, mientras que la otra a los administradores:

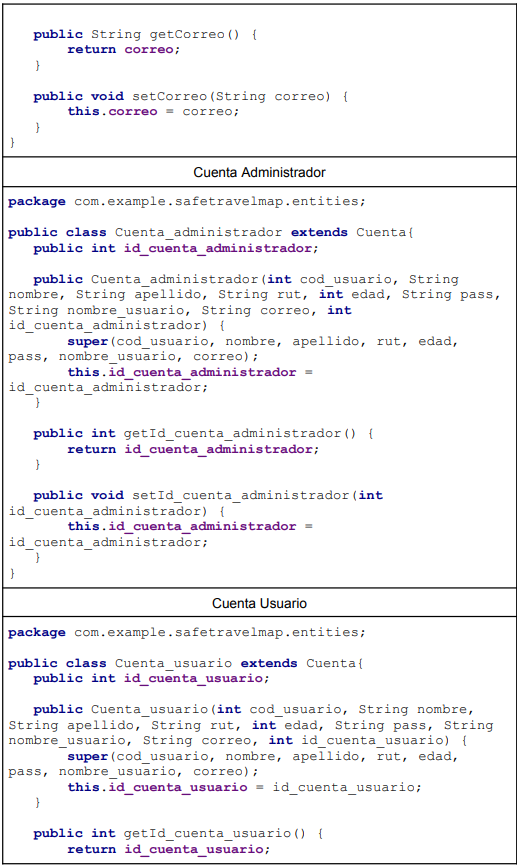


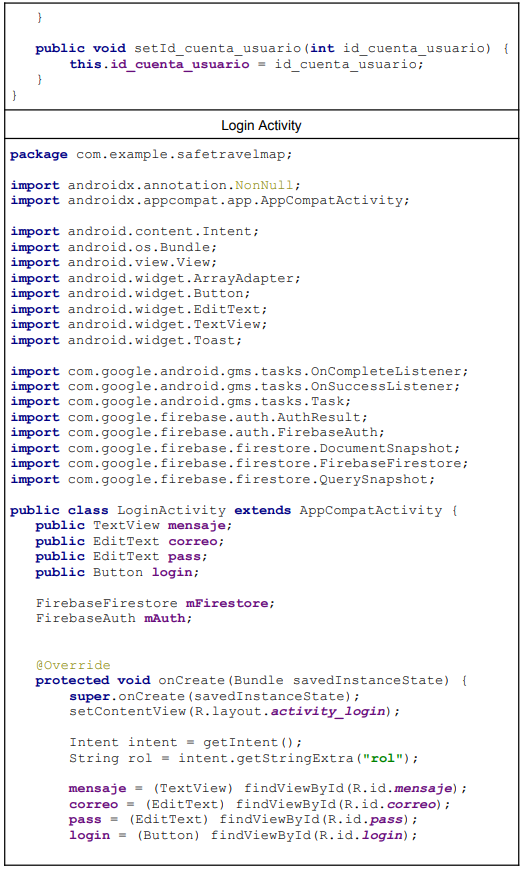
Código Fuente

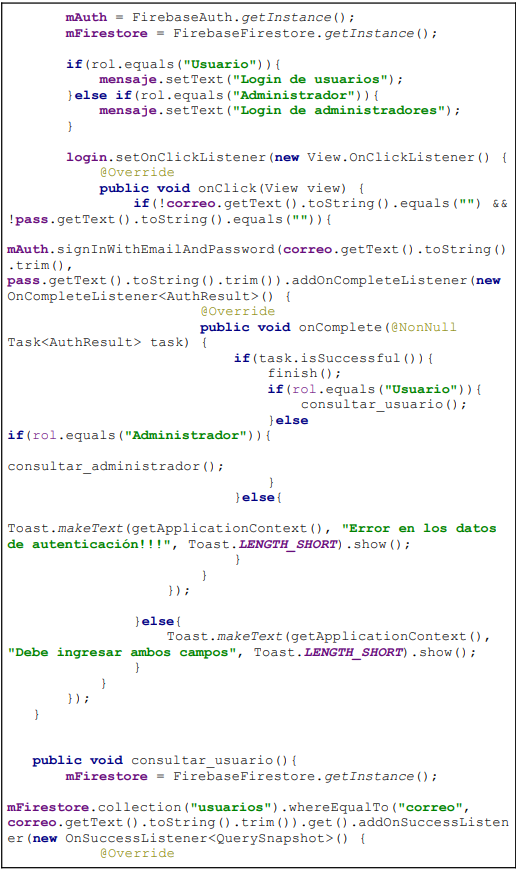
Desarrollo:





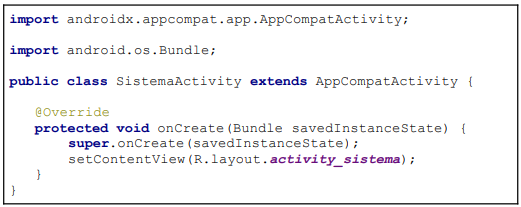








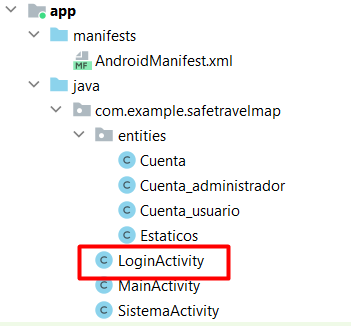




## 3.12.2 Sprint dos

1. Se ha trabajado sobre el activity **“LoginActivity”** con la intención de rescatar el tipo de usuario que ingresará al sistema, para así poder registrar desperfectos desde su perfil.

Desarrollo:



|  |
| --- |
| **Login Activity** |
| **package** com.example.safetravelmap;  **import** androidx.annotation.NonNull;  **import** androidx.appcompat.app.AppCompatActivity;  **import** android.content.Intent;  **import** android.os.Bundle;  **import** android.provider.Settings;  **import** android.util.Log;  **import** android.view.View;  **import** android.widget.ArrayAdapter;  **import** android.widget.Button;  **import** android.widget.EditText;  **import** android.widget.TextView;  **import** android.widget.Toast;  **import** com.example.safetravelmap.entities.Estaticos;  **import** com.google.android.gms.tasks.OnCompleteListener;  **import** com.google.android.gms.tasks.OnFailureListener;  **import** com.google.android.gms.tasks.OnSuccessListener;  **import** com.google.android.gms.tasks.Task;  **import** com.google.firebase.auth.AuthResult;  **import** com.google.firebase.auth.FirebaseAuth;  **import** com.google.firebase.firestore.FirebaseFirestore;  **import** com.google.firebase.firestore.QueryDocumentSnapshot;  **import** com.google.firebase.firestore.QuerySnapshot;  **import static** android.content.ContentValues.***TAG***;  **public class** LoginActivity **extends** AppCompatActivity {  **public** TextView **mensaje**;  **public** EditText **correo**;  **public** EditText **pass**;  **public** Button **login**;  FirebaseFirestore **mFirestore**;  FirebaseAuth **mAuth**;  @Override  **protected void** onCreate(Bundle savedInstanceState) {  **super**.onCreate(savedInstanceState);  setContentView(R.layout.***activity\_login***);  Intent intent = getIntent();  String rol = intent.getStringExtra(**"rol"**);  **mensaje** = (TextView) findViewById(R.id.***mensaje***);  **correo** = (EditText) findViewById(R.id.***correo***);  **pass** = (EditText) findViewById(R.id.***pass***);  **login** = (Button) findViewById(R.id.***login***);  **mAuth** = FirebaseAuth.*getInstance*();  **mFirestore** = FirebaseFirestore.*getInstance*();  **if**(rol.equals(**"Usuario"**)){  **mensaje**.setText(**"Login de usuarios"**);  }**else if**(rol.equals(**"Administrador"**)){  **mensaje**.setText(**"Login de administradores"**);  }  **login**.setOnClickListener(**new** View.OnClickListener() {  @Override  **public void** onClick(View view) {  **if**(!**correo**.getText().toString().equals(**""**) && !**pass**.getText().toString().equals(**""**)){  **mAuth**.signInWithEmailAndPassword(**correo**.getText().toString().trim(), **pass**.getText().toString().trim()).addOnCompleteListener(**new** OnCompleteListener<AuthResult>() {  @Override  **public void** onComplete(@NonNull Task<AuthResult> task) {  **if**(task.isSuccessful()){  finish();  **if**(rol.equals(**"Usuario"**)){  consultar\_usuario();  }**else if**(rol.equals(**"Administrador"**)){  consultar\_administrador();  }  }**else**{  Toast.*makeText*(getApplicationContext(), **"Error en los datos de autenticación!!!"**, Toast.***LENGTH\_SHORT***).show();  }  }  });  }**else**{  Toast.*makeText*(getApplicationContext(), **"Debe ingresar ambos campos"**, Toast.***LENGTH\_SHORT***).show();  }  }  });  }  **public void** consultar\_usuario(){  Toast.*makeText*(getApplicationContext(), **"Consultando usuario en base de datos..."**, Toast.***LENGTH\_SHORT***).show();  **mFirestore** = FirebaseFirestore.*getInstance*();  **mFirestore**.collection(**"usuarios"**).whereEqualTo(**"correo"**, **correo**.getText().toString().trim()).get().addOnSuccessListener(**new** OnSuccessListener<QuerySnapshot>() {  @Override  **public void** onSuccess(QuerySnapshot queryDocumentSnapshots) {  **if**(queryDocumentSnapshots.isEmpty()){  Toast.*makeText*(getApplicationContext(), **"No existe este usuario en base de datos."**, Toast.***LENGTH\_SHORT***).show();  **return**;  }**else**{  **for** (QueryDocumentSnapshot document : queryDocumentSnapshots) {  Log.*d*(***TAG***, document.getId() + **" => "** + document.getData());  Toast.*makeText*(getApplicationContext(), **"Bienvenido: "** + document.getString(**"nombre"**) + **" "** + document.getString(**"apellido"**), Toast.***LENGTH\_SHORT***).show();  Estaticos.*cuenta\_usuario*.setApellido(document.getString(**"apellido"**));  Estaticos.*cuenta\_usuario*.setCod\_usuario(document.getLong(**"cod\_usuario"**).intValue());  Estaticos.*cuenta\_usuario*.setCorreo(document.getString(**"correo"**));  Estaticos.*cuenta\_usuario*.setEdad(document.getLong(**"edad"**).intValue());  Estaticos.*cuenta\_usuario*.setNombre(document.getString(**"nombre"**));  Estaticos.*cuenta\_usuario*.setNombre\_usuario(document.getString(**"nombre\_usuario"**));  Estaticos.*cuenta\_usuario*.setPass(document.getString(**"pass"**));  Estaticos.*cuenta\_usuario*.setRut(document.getString(**"rut"**));  }  Intent intent = **new** Intent(getApplicationContext(), SistemaActivity.**class**);  startActivity(intent);  Estaticos.*tipo\_usuario* = **true**;  *//Toast.makeText(getApplicationContext(), "Bienvenido", Toast.LENGTH\_SHORT).show();*  }  }  }).addOnFailureListener(**new** OnFailureListener() {  @Override  **public void** onFailure(@NonNull Exception e) {  Toast.*makeText*(getApplicationContext(), **"ERROR: "** + e.toString(), Toast.***LENGTH\_SHORT***).show();  }  });  }  **public void** consultar\_administrador(){  Toast.*makeText*(getApplicationContext(), **"Consultando administrador en base de datos..."**, Toast.***LENGTH\_SHORT***).show();  **mFirestore** = FirebaseFirestore.*getInstance*();  **mFirestore**.collection(**"administrador"**).whereEqualTo(**"correo"**, **correo**.getText().toString().trim()).get().addOnSuccessListener(**new** OnSuccessListener<QuerySnapshot>() {  @Override  **public void** onSuccess(QuerySnapshot queryDocumentSnapshots) {  **if**(queryDocumentSnapshots.isEmpty()){  Toast.*makeText*(getApplicationContext(), **"No existe este administrador en base de datos."**, Toast.***LENGTH\_SHORT***).show();  **return**;  }**else**{  **for** (QueryDocumentSnapshot document : queryDocumentSnapshots) {  Log.*d*(***TAG***, document.getId() + **" => "** + document.getData());  Toast.*makeText*(getApplicationContext(), **"Bienvenido: "** + document.getString(**"nombre"**) + **" "** + document.getString(**"apellido"**), Toast.***LENGTH\_SHORT***).show();  Estaticos.*cuenta\_administrador*.setApellido(document.getString(**"apellido"**));  Estaticos.*cuenta\_administrador*.setCod\_usuario(document.getLong(**"cod\_usuario"**).intValue());  Estaticos.*cuenta\_administrador*.setCorreo(document.getString(**"correo"**));  Estaticos.*cuenta\_administrador*.setEdad(document.getLong(**"edad"**).intValue());  Estaticos.*cuenta\_administrador*.setNombre(document.getString(**"nombre"**));  Estaticos.*cuenta\_administrador*.setNombre\_usuario(document.getString(**"nombre\_usuario"**));  Estaticos.*cuenta\_administrador*.setPass(document.getString(**"pass"**));  Estaticos.*cuenta\_administrador*.setRut(document.getString(**"rut"**));  }  Intent intent = **new** Intent(getApplicationContext(), SistemaActivity.**class**);  startActivity(intent);  Estaticos.*tipo\_usuario* = **false**;  *//Toast.makeText(getApplicationContext(), "Bienvenido", Toast.LENGTH\_SHORT).show();*  }  }  }).addOnFailureListener(**new** OnFailureListener() {  @Override  **public void** onFailure(@NonNull Exception e) {  Toast.*makeText*(getApplicationContext(), **"ERROR: "** + e.toString(), Toast.***LENGTH\_SHORT***).show();  }  });  }  } |

Para registrar de manera global el usuario que ha ingresado se han creado variables globales o estáticas dentro del proyecto, que ayudarán a acceder a los datos desde cualquier actividad.

|  |
| --- |
| Estáticos |
| **package** com.example.safetravelmap.entities;  **public class** Estaticos {  **public static** Cuenta\_usuario *cuenta\_usuario* = **new** Cuenta\_usuario();  **public static** Cuenta\_administrador *cuenta\_administrador* = **new** Cuenta\_administrador();  **public static boolean** *tipo\_usuario*;  } |

Una vez que se haya recuperado la información del tipo de usuario, junto con sus datos correspondientes, este será redireccionado a la pantalla principal de la aplicación. En esta pantalla, el usuario tendrá la opción de reportar cualquier desperfecto. Para ello, deberá proporcionar una descripción detallada, indicar la gravedad del problema, y tiene la opción de adjuntar una fotografía. Esta fotografía se almacenará inicialmente en el dispositivo del usuario. Posteriormente, se subirá a la plataforma Firebase, aprovechando su servicio de almacenamiento de archivos, Filestorage.

|  |
| --- |
| **Sistema Activity** |
| **package** com.example.safetravelmap;  **import** androidx.appcompat.app.AppCompatActivity;  **import** androidx.core.app.ActivityCompat;  **import** androidx.core.content.ContextCompat;  **import** androidx.core.content.FileProvider;  **import** android.content.Intent;  **import** android.content.pm.PackageManager;  **import** android.net.Uri;  **import** android.os.Bundle;  **import** android.os.Environment;  **import** android.provider.MediaStore;  **import** android.util.Log;  **import** android.view.View;  **import** android.widget.Button;  **import** android.widget.EditText;  **import** android.widget.Switch;  **import** android.widget.TextView;  **import** android.widget.Toast;  **import** com.example.safetravelmap.entities.Estaticos;  **import** com.google.android.gms.tasks.OnSuccessListener;  **import** com.google.firebase.firestore.DocumentReference;  **import** com.google.firebase.firestore.FirebaseFirestore;  **import** com.google.firebase.storage.FirebaseStorage;  **import** com.google.firebase.storage.StorageException;  **import** com.google.firebase.storage.StorageReference;  **import** com.google.firebase.storage.UploadTask;  **import** org.checkerframework.checker.nullness.qual.NonNull;  **import** org.checkerframework.checker.nullness.qual.Nullable;  **import** com.google.android.gms.tasks.OnFailureListener;  **import** java.io.File;  **import** java.io.IOException;  **import** java.text.SimpleDateFormat;  **import** java.util.Date;  **import** java.util.HashMap;  **import** java.util.Map;  **import** android.Manifest;  **import static** android.content.ContentValues.***TAG***;  **public class** SistemaActivity **extends** AppCompatActivity {  **public** Button **cargarImagen**;  **static final int *REQUEST\_IMAGE\_CAPTURE*** = 1;  **private static final int *REQUEST\_PERMISSIONS*** = 1;  String **currentPhotoPath**, **imageFileName**;  TextView **textView**;  EditText **desc**;  Switch **switch1**;  @Override  **protected void** onCreate(Bundle savedInstanceState) {  **super**.onCreate(savedInstanceState);  setContentView(R.layout.***activity\_sistema***);  **cargarImagen** = (Button) findViewById(R.id.***cargarImagen***);  **textView** = (TextView) findViewById(R.id.***textView***);  **desc** = (EditText) findViewById(R.id.***desc***);  **switch1** = findViewById(R.id.***switch1***);  **if**(Estaticos.*tipo\_usuario* == **true**){  **textView**.setText(**"Bienvenido: "** + Estaticos.*cuenta\_usuario*.getNombre\_usuario());  }**else**{  **textView**.setText(**"Bienvenido administrador: "** + Estaticos.*cuenta\_administrador*.getNombre\_usuario());  }  **cargarImagen**.setOnClickListener(**new** View.OnClickListener() {  @Override  **public void** onClick(View view) {  *// Solicita los permisos*  requestPermissions();  }  });  }  **private void** requestPermissions() {  String[] permissions = {  Manifest.permission.***CAMERA***,  Manifest.permission.***WRITE\_EXTERNAL\_STORAGE***,  Manifest.permission.***READ\_EXTERNAL\_STORAGE***  };  **if** (!hasPermissions(permissions)) {  ActivityCompat.*requestPermissions*(**this**, permissions, ***REQUEST\_PERMISSIONS***);  } **else** {  tomarFoto();  }  }  **private boolean** hasPermissions(String... permissions) {  **for** (String permission : permissions) {  **if** (ActivityCompat.*checkSelfPermission*(**this**, permission) != PackageManager.***PERMISSION\_GRANTED***) {  **return false**;  }  }  **return true**;  }  @Override  **public void** onRequestPermissionsResult(**int** requestCode, @NonNull String[] permissions, @NonNull **int**[] grantResults) {  **super**.onRequestPermissionsResult(requestCode, permissions, grantResults);  **if** (requestCode == ***REQUEST\_PERMISSIONS***) {  **if** (grantResults.**length** > 1 &&  grantResults[0] == PackageManager.***PERMISSION\_GRANTED*** &&  grantResults[1] == PackageManager.***PERMISSION\_GRANTED***) {  *// Los permisos fueron concedidos*  tomarFoto();  } **else** {  *// Los permisos fueron denegados*  Toast.*makeText*(**this**, **"Permiso de cámara y almacenamiento requerido"**, Toast.***LENGTH\_SHORT***).show();  }  }  }  **private void** tomarFoto() {  Intent takePictureIntent = **new** Intent(MediaStore.***ACTION\_IMAGE\_CAPTURE***);  **if** (takePictureIntent.resolveActivity(getPackageManager()) != **null**) {  File photoFile = **null**;  **try** {  photoFile = createImageFile();  } **catch** (IOException ex) {  Toast.*makeText*(**this**, **"ex: "** + ex.toString(),Toast.***LENGTH\_SHORT***).show();  }  **if** (photoFile != **null**) {  Uri photoURI = FileProvider.*getUriForFile*(**this**,  **"com.example.android.fileprovider"**,  photoFile);  takePictureIntent.putExtra(MediaStore.***EXTRA\_OUTPUT***, photoURI);  startActivityForResult(takePictureIntent, ***REQUEST\_IMAGE\_CAPTURE***);  **currentPhotoPath** = photoFile.getAbsolutePath();  }  }  }  @Override  **protected void** onActivityResult(**int** requestCode, **int** resultCode, @Nullable Intent data) {  **super**.onActivityResult(requestCode, resultCode, data);  *// Al terminar el proceso de captura de imagen*  **if** (requestCode == ***REQUEST\_IMAGE\_CAPTURE*** && resultCode == ***RESULT\_OK***) {  subirImagen();  }  }  **private** File createImageFile() **throws** IOException {  String timeStamp = **new** SimpleDateFormat(**"yyyyMMdd\_HHmmss"**).format(**new** Date());  **imageFileName** = timeStamp;  File storageDir = Environment.*getExternalStoragePublicDirectory*(Environment.*DIRECTORY\_PICTURES*);  File image = File.*createTempFile*(  **imageFileName**,  **".jpg"**,  storageDir  );  **currentPhotoPath** = image.getAbsolutePath();  **return** image;  }  **private void** subirImagen(){  FirebaseStorage storage = FirebaseStorage.*getInstance*();  StorageReference storageRef = storage.getReference();  StorageReference imagenRef = storageRef.child(**imageFileName** + **".jpg"**);  Uri file = Uri.*fromFile*(**new** File(**currentPhotoPath**));  Toast.*makeText*(SistemaActivity.**this**, **"Directorio: "** + file, Toast.***LENGTH\_SHORT***).show();  UploadTask uploadTask = imagenRef.putFile(file);  uploadTask.addOnFailureListener(**new** OnFailureListener() {  @Override  **public void** onFailure(@NonNull Exception exception) {  Toast.*makeText*(SistemaActivity.**this**, **"EL ERROR: "** + exception.getCause(), Toast.***LENGTH\_SHORT***).show();  System.***out***.println(**"EL ERROR: "** + exception.toString());  }  }).addOnSuccessListener(**new** OnSuccessListener<UploadTask.TaskSnapshot>() {  @Override  **public void** onSuccess(UploadTask.TaskSnapshot taskSnapshot) {  Toast.*makeText*(SistemaActivity.**this**, **"Bien"**, Toast.***LENGTH\_SHORT***).show();  almacenar();  }  });  }  **public void** almacenar(){  FirebaseFirestore mFirestore = FirebaseFirestore.*getInstance*();  *// Crear un nuevo objeto con los atributos que se quieren almacenar*  Map<String, Object> desperfecto = **new** HashMap<>();  **if**(Estaticos.*tipo\_usuario* == **true**) {  desperfecto.put(**"cod\_usuario"**, Estaticos.*cuenta\_usuario*.getCod\_usuario()); *// Aquí se debe poner el número que corresponda*  } **else** {  desperfecto.put(**"cod\_usuario"**, Estaticos.*cuenta\_administrador*.getCod\_usuario()); *// Aquí se debe poner el número que corresponda*  }  desperfecto.put(**"desc"**, **desc**.getText().toString()); *// Aquí se debe poner la descripción*  desperfecto.put(**"imagen"**, **imageFileName** + **".jpg"**); *// Aquí se debe poner la URL de la imagen*  **if**(**switch1**.isChecked()){  desperfecto.put(**"riesgo"**, 1);  }**else**{  desperfecto.put(**"riesgo"**, 0);  }  desperfecto.put(**"tipo\_usuario"**, Estaticos.*tipo\_usuario*); *// Aquí se debe poner el tipo de usuario*  *// Agregar un nuevo documento a la colección "desperfectos"*  mFirestore.collection(**"desperfectos"**)  .add(desperfecto)  .addOnSuccessListener(**new** OnSuccessListener<DocumentReference>() {  @Override  **public void** onSuccess(DocumentReference documentReference) {  *// Documento añadido con éxito*  Log.*d*(***TAG***, **"Documento añadido con ID: "** + documentReference.getId());  Toast.*makeText*(SistemaActivity.**this**, **"Documento añadido con ID: "** + documentReference.getId(), Toast.***LENGTH\_SHORT***).show();  }  })  .addOnFailureListener(**new** OnFailureListener() {  @Override  **public void** onFailure(@NonNull Exception e) {  *// Error al añadir documento*  Log.*w*(***TAG***, **"Error añadiendo documento"**, e);  }  });  }  } |

1. Permisos otorgados dentro del archivo manifest de Android.

**Desarrollo:**

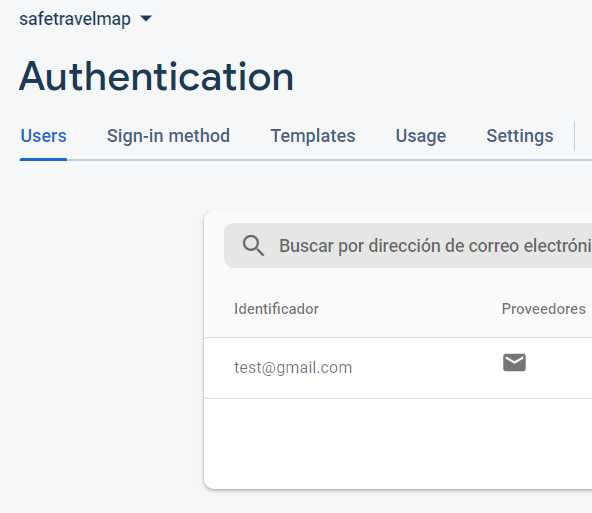
Para habilitar la conexión a Internet, el acceso a archivos almacenados en el dispositivo móvil, la capacidad de escribir archivos en el mismo, así como el uso de la cámara, se han configurado diversos permisos en el archivo AndroidManifest.xml.

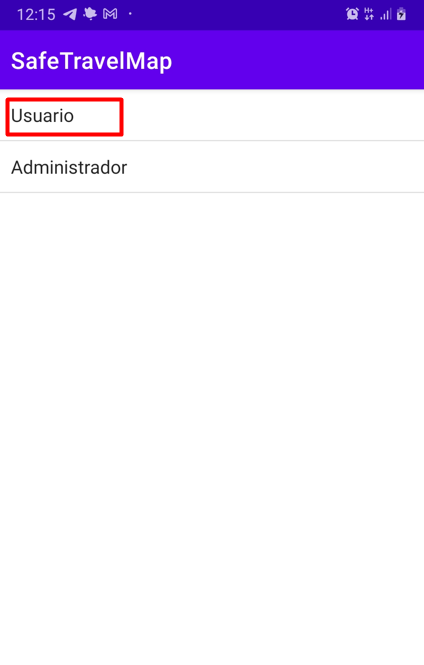
|  |
| --- |
| Permisos |
| **<uses-permission android:name="android.permission.INTERNET" />**  **<uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE" />**  **<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />**  **<uses-permission android:name="android.permission.CAMERA" />** |

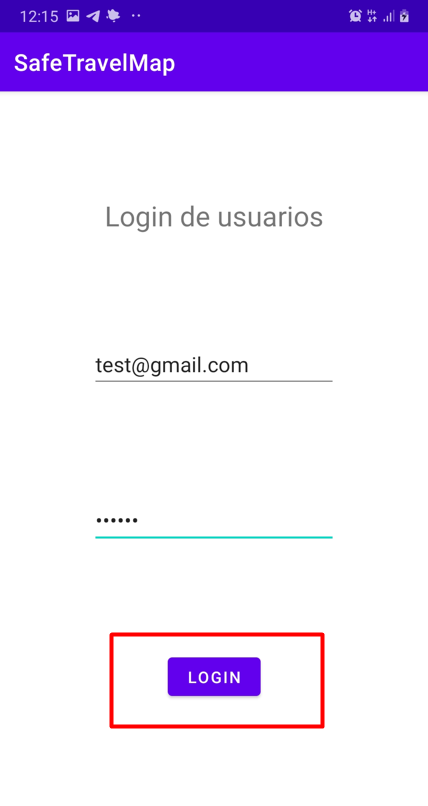
1. Demostración

Desarrollo:

Se utiliza el usuario con correo [test@gmail.com](mailto:test@gmail.com) para iniciar sesión dentro de la aplicación.

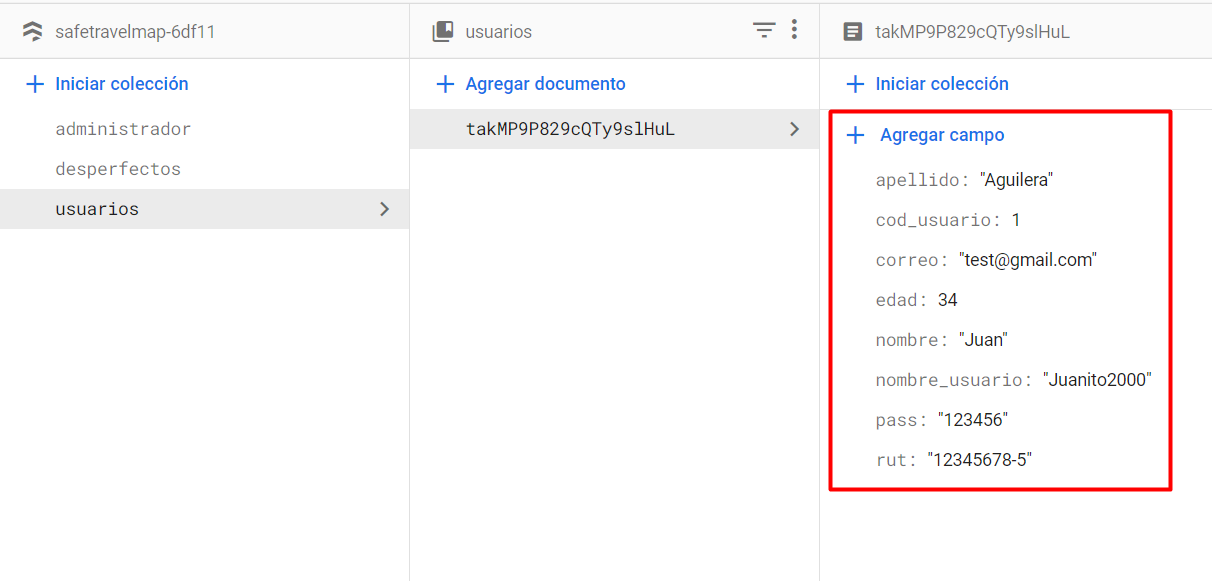
****

****

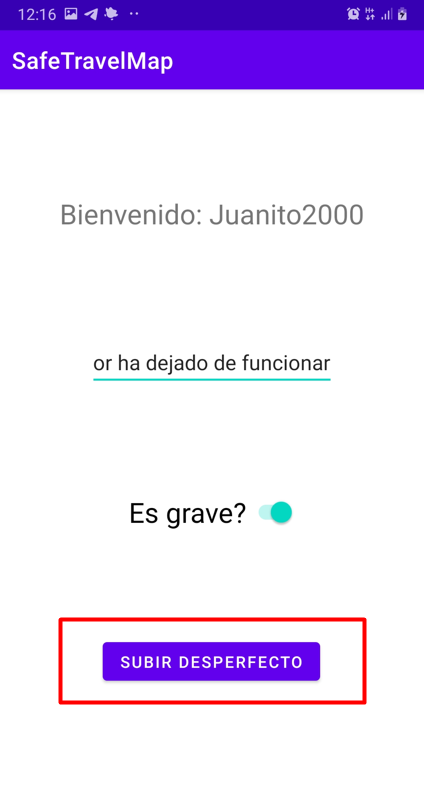
****

****

Efectivamente este usuario está registrado en la base de datos de Firestore.

****

Una vez dentro se describe el desperfecto, se indica si es o no grave y se captura una fotografía del mismo.



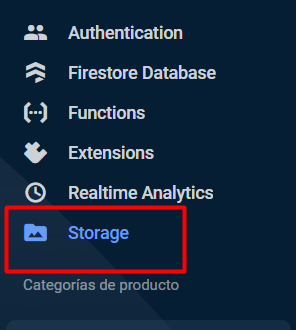
Tras un breve lapso de tiempo, que variará según la velocidad de la conexión a Internet, tanto la imagen como la información relativa al desperfecto serán subidas y almacenadas de manera permanente en la plataforma Firebase. Para lograr esto, se utiliza FireStorage para el almacenamiento del documento gráfico y Firestore como el sistema de gestión de la base de datos que contiene el registro del desperfecto.



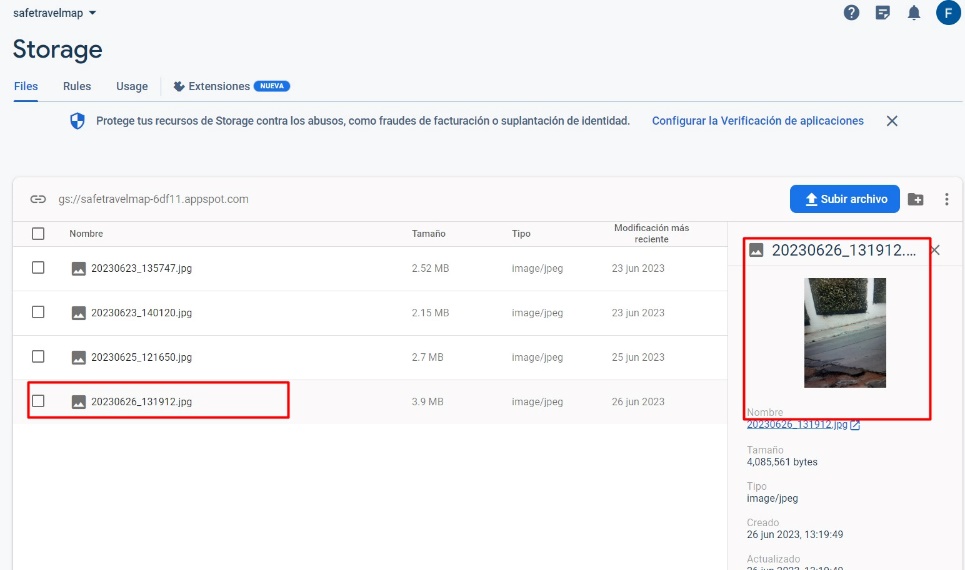
Este es el código de la imagen registrada.



Se comprueba en Firestorage.



Efectivamente, ahí está la imagen asociada al desperfecto, almacenada en la plataforma de Firebase.



## 3.13 POSTMORTEM

La metodología del velero será aplicada en la parte de retrospectiva después de la finalización del sprint, esto es una forma baja de presión que sirve para que los equipos reflexionen sobre como se ha manejado el proyecto.

Los aspectos para considerar o puntos a desarrollar son:

* Las rocas como riesgos.
* Las anclas como problemas que generaron retrasos.
* El viento como los equipos que ayudaron.
* La tierra como el objetivo.



## 3.13.1 Sprint uno

|  |
| --- |
| Personas que facilitaron el entendimiento del desarrollo, análisis y comprensión de temas. |
| Profesora Sarita González, Martin muñoz(analista), Nikza Flores (desarrolladora). |
|  |
| El equipo en el que se trabajaba causo varios contras tiempos por fallas mecánicas. |
| Los problemas de salud del desarrollador retraso el proyecto. |
|  |
| Las largas horas de capacitación para avanzar muchas veces genero retraso. |
| El ambiente de desarrollo dejo de emular la aplicación móvil por una semana. |
|  |
| El objetivo para cumplir era tener toda la documentación lista y se logró. |
| Que la aplicación móvil desarrollada funcionara antes de la entrega del sprint, también se logró. |
| Tener el sprint uno terminado en la fecha que se solicitaba, también fue logrado. |

## 3.12.2 Sprint dos

|  |
| --- |
| Personas que facilitaron el entendimiento del desarrollo, análisis y comprensión de temas. |
| Profesora Sarita González, Martin muñoz(analista), Nikza Flores (desarrolladora). |
|  |
| La documentación del sprint dos se dañó a días de la presentación. |
| Debido al daño en la carpeta raíz, también se dañó el versionamiento en GitHub. |
| FireBase necesito una gran cantidad de configuraciones para el desarrollo de este sprint lo que causo un retraso importante. |
| Se uso SQL cloud en GCP, esto disparo el costo a 400 dolares y se tuvieron que tomar las medidas indicadas en la mitigación de riesgo. |
|  |
| Las largas horas de capacitación para avanzar muchas veces genero retraso. |
| El desarrollador se vio obligado a tomar trabajos este mes, lo que dejo menos tiempo para el desarrollo. |
|  |
| El objetivo a cumplir era tener toda la documentación lista del sprint2 y se logró. |
| La aplicación móvil desarrollada funciona en un ambiente de producción. |
| Tener el sprint dos terminado en la fecha que se solicitaba fue logrado. |

## PROBLEMAS ABIERTOS

Luego del segundo entregable a los usuarios surgió una nueva historia de usuario, la cual se encuentra en la siguiente tabla de priorización.

|  |  |  |  |
| --- | --- | --- | --- |
| Identificador de la historia por Jira | Enunciado de la Historia | Priorización MoSCoW | Asignación de puntos “Planing Poker” |
| SAF19 | Yo como usuario quiero que exista una clasificación y filtrado nuevo para reportar animales muertos en la vía pública. | Won’t have | 0 |

## 3.15 TRABAJO FUTURO

Evaluando las historias de usuarios:

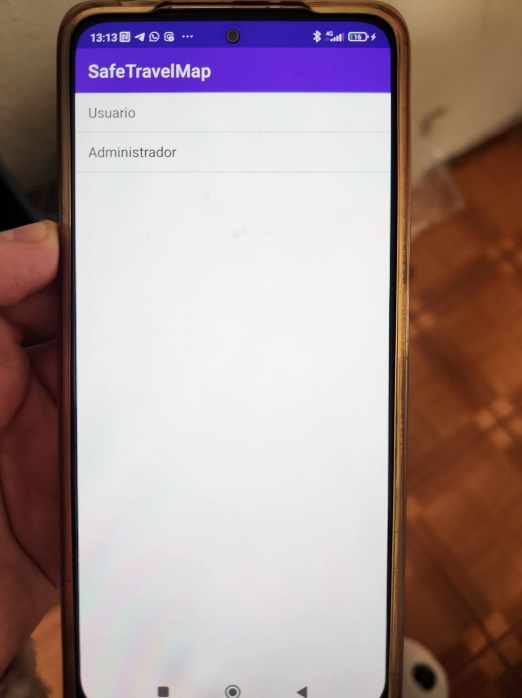
|  |  |  |  |
| --- | --- | --- | --- |
| SAF18 | Yo como usuario quiero que la aplicación móvil indique los focos de delincuencia que existen en la ciudad para evitar pasar por ahí. | Won’t have | Esta historia de usuario a futuro no se va a desarrollar, pese a que suena como una idea que le aportaría gran valor al proyecto sus puntos de estimación son de 100, debido a que se necesitaría trabajar con la base de datos de carabineros y en su mayoría a nivel nacional estas bases de datos no están actualizadas, los datos están incompletos o simplemente no se reporto el crimen por la persona a cargo. |
| SAF19 | Yo como usuario quiero que exista una clasificación y filtrado nuevo para reportar animales muertos en la vía pública. | Won’t have | Esta historia de usuario a futuro se debería re evaluar si se podría implementar, pese a no representar un puntaje superior a 8, la aplicación esta pensada para que personas de diferentes edades la usen, el ver imágenes de animales muertos podría herir la directamente sensibilidad de los demás usuarios. |

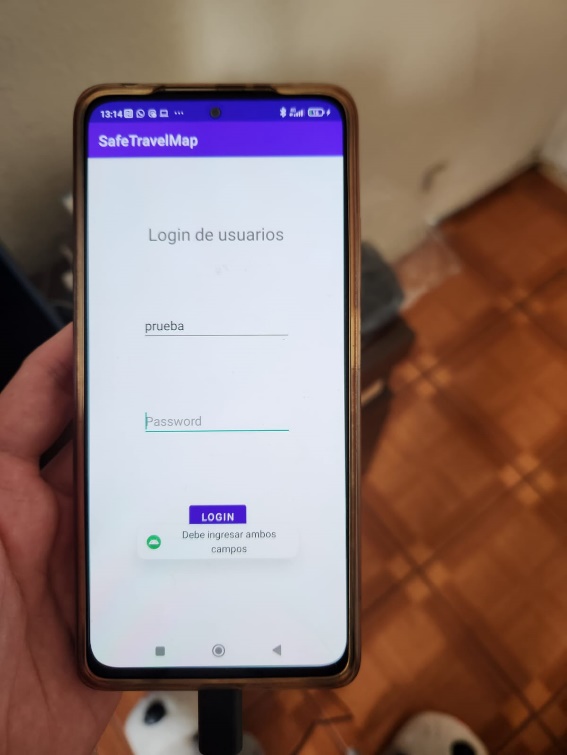
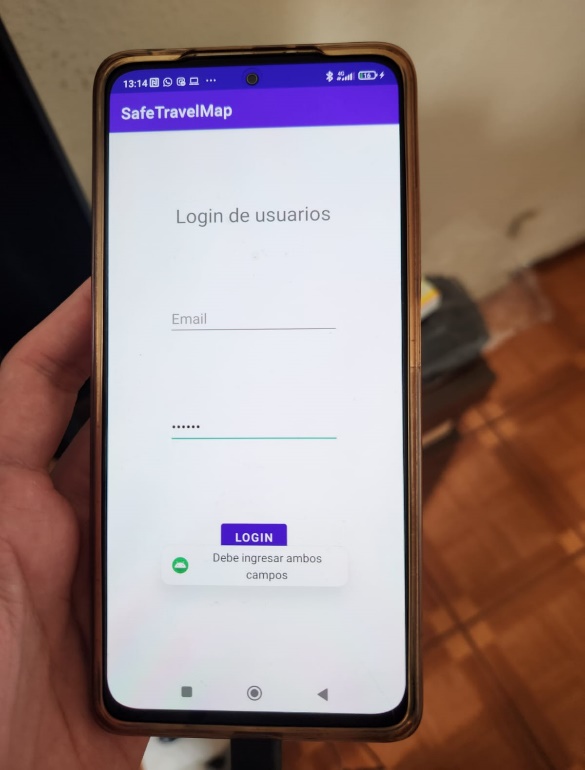
## 3.16 Evidencia ambiente de producción

## 3.13.1 Sprint uno

Aplicación móvil instalada en el celular. Aplicación móvil abierta

selección de roles usuario o administrador.

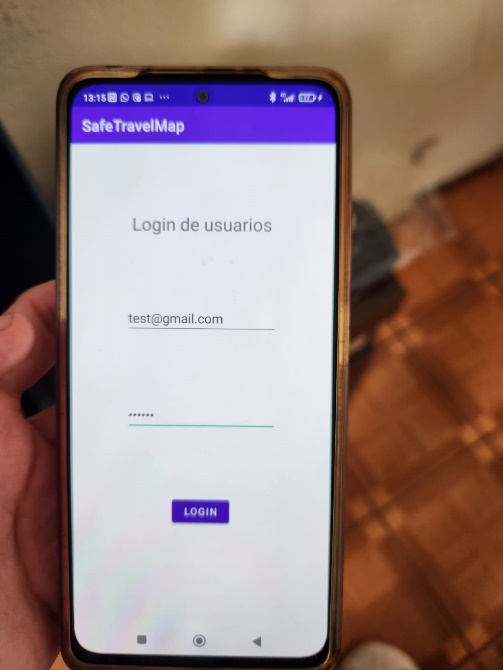
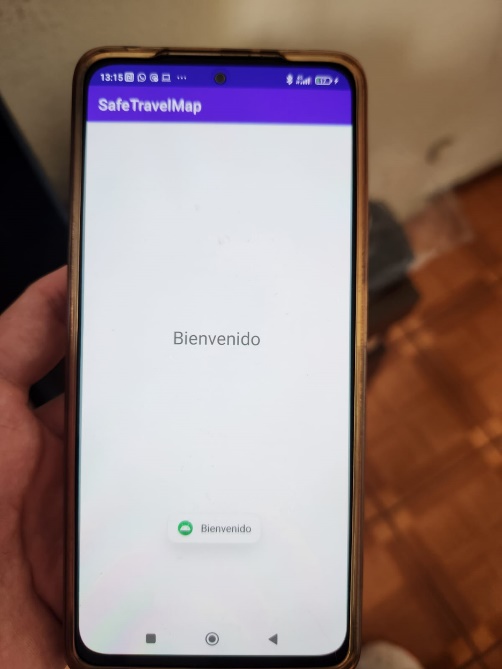
 

Rol usuario seleccionado, rellenando Rellenando solo campo de abajo.

solo el campo de arriba. Despliega Despliega mensaje de error: “Debe

mensaje de error : “Debe ingresar ingresar ambos campos”.

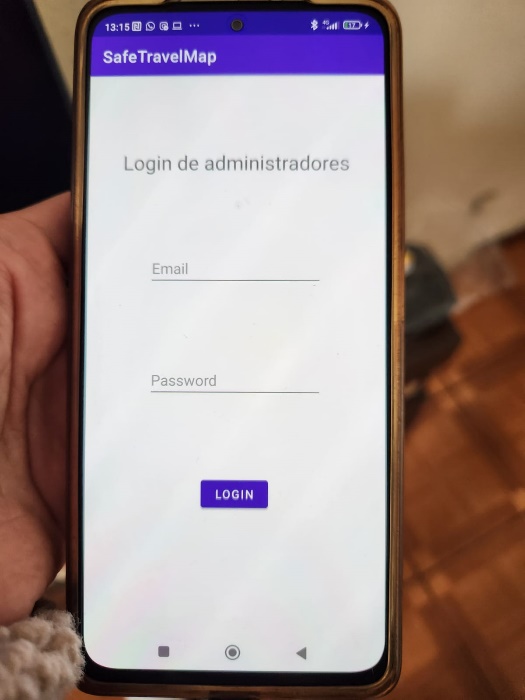
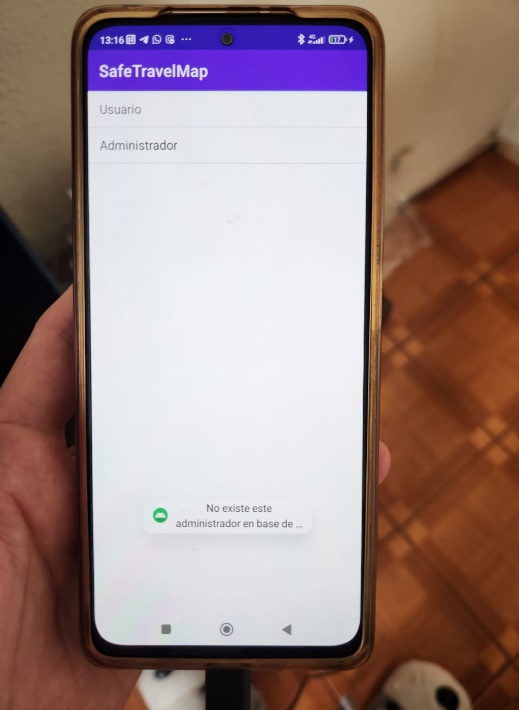
ambos campos”.

Se ingresa cuenta de prueba Al loguear despliega el main

ID:test@gmail.com debería aparecer en futuras iteraciones

Password:123456 el mapa y las alertas.

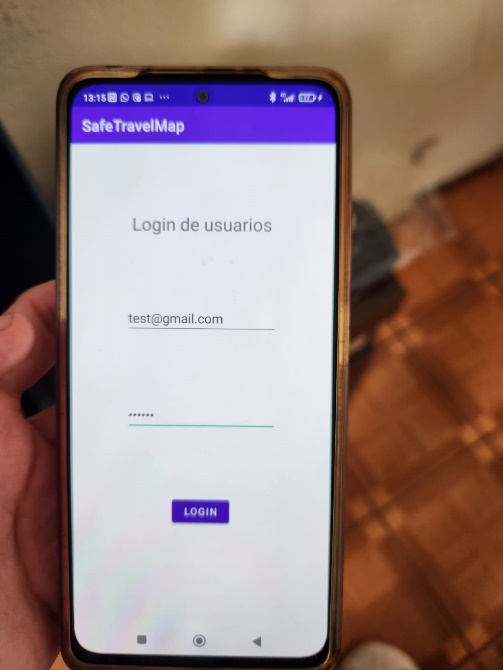
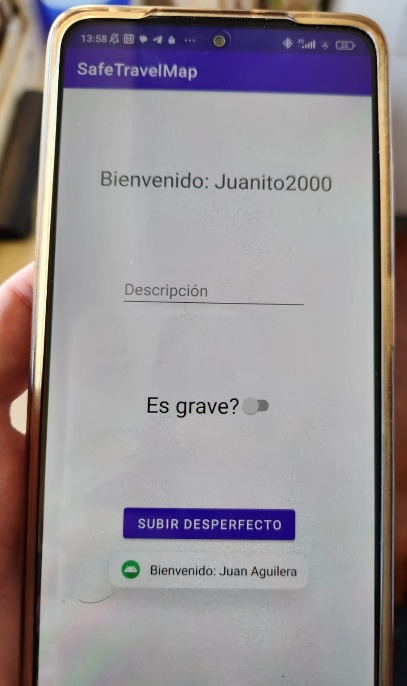
Login administradores, se prueba Despliega mensaje de error: “No

con la cuenta de usuario registrada existe este administrador en base

ID:test@gmail.com de datos”

Password:123456

## 3.13.1 Sprint dos

Se ingresa cuenta de prueba Al loguear despliega el main

ID:test@gmail.com donde se puede visualizar

Password:123456 el menú para subir el reporte

Luego de dar una descripción El reporte es enviado a la

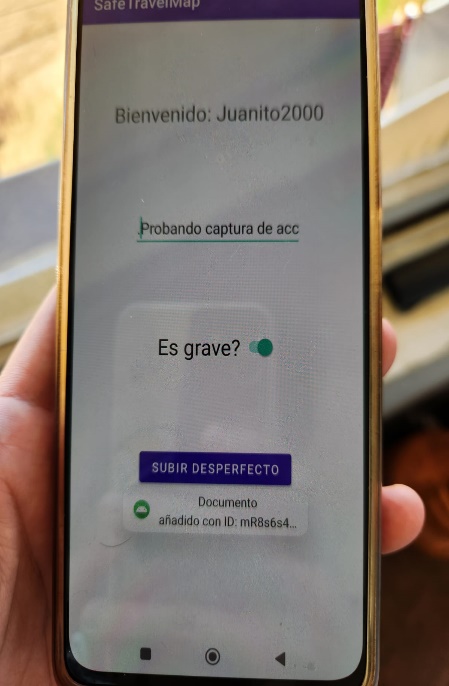
opcional, clasificar esto como base de datos de Firebase.

grave y oprimir subir desperfecto

la aplicación móvil nos solicita subir

una captura al desperfecto encontrado,

para hacer efectivo el reporte.

El reporte es almacenado en la El segundo mensaje que se despliega

plataforma de FireBase y despliega da el ID del almacenamiento

el mensaje “Bien” para confirmar grafico.

que los datos se han guardado de

forma permanente y exitosamente.