DESARROLLO DE UNA APLICACIÓN MÓVIL QUE AYUDE A COORDINAR LA COMUNICACIÓN DE LA COMUNIDAD DE TRANSPORTE DE LA UPBBGA UTILIZANDO BASES DE DATOS NOSQL PARA MOSTRAR INFORMACIÓN EN LÍNEA

Fabian Humberto Castillo Pineda

Director del proyecto
Ing. René Rodriguez Clavijo

UNIVERSIDAD PONTIFICIA BOLIVARIANA SECCIONAL BUCARAMANGA
ESCUELA DE INGENIERÍAS
FACULTAD DE INGENIERIA DE SISTEMAS E INFORMATICA
BUCARAMANGA
2018

TABLA DE CONTENIDO

Contenido

1. INTRODUCCIÓN 10
2. RESUMEN DEL PROYECTO 10
   2.1 DEFINICIÓN DEL PROBLEMA 10
   2.2 JUSTIFICACIÓN 12
3. OBJETIVOS 12
   3.1 Objetivo General. 12
   3.2 Objetivos Específicos. 12
4. ANTECEDENTES 14
5. MARCO TEORICO 16
   5.1 METODOLOGÍA PARA DESARROLLO DE SOFWTARE 16
   5.2 EXPERIENCIA DE USUARIO (UX) EN EL DESARROLLO DEL APLIVATIVO MOVIL UBICAR 17
      5.2.1 Factores que se tienen en cuenta para mejorar la experiencia de usuario 18
   5.3 PROTOTIPO FUNCIONAL Y DISEÑO DE VISTAS DEL APLICATIVO MÓVIL CON MARVEL APP. 19
      5.3.1 Interfaz gráfica Ubicar 21
   5.4 ENCUESTAS DE USABILIDAD Y AMIGABILIDAD DEL SOFTWARE UBICAR EN LA UNIVERSIDAD PONTIFICIA BOLIVARIANA DE BUCARAMANGA. 25
   5.5 TECNOLOGIAS 29
      5.5.1 Frontend 29
      5.5.2 Base de datos 33
   5.6 Estructura de la base de datos Nosql-Firebase 36
   5.7 ARQUITECTURA UBICAR 43
   5.8 BACKEND 44
6. METODOLOGÍA 48
   6.1 Especificación de requerimientos funcionales y no funcionales según el estándar IEEE 830 48

6.2 Metodología Ubicar 49
6.3 Tipos de usuario aplicación móvil Ubicar: 50
6.4 Control de versiones de la interfaz de usuario Ubicar: 51
6.5 Control de versiones del código 53
6.6 Control de errores y pruebas con Firebase Crashlytics y Fabric.io 56
6.7 Pruebas de funcionalidad en la Universidad Pontificia Bolivariana de Bucaramanga 58
7. ACTA DE PROPIEDAD INTELECTUAL 60
8. RESULTADOS 60
8.1 Acceso al sistema 60
8.2 Menú del sistema 61
8.3 Ser un conductor 62
8.4 Agregar una ruta 63
8.5 Rutas en línea / Mis pasajeros 64
8.6 Automóviles en mapa 65
8.7 Rutas que salen en una hora próxima (rutas programadas) 66
8.8 Solicitudes 67
8.9 Mi conductor 68
9. CONCLUSIONES 72
10. REFERENCIA BIBLIOGRÁFICAS 74
11. ANEXOS 75
11.1 ANEXO A – Requerimientos funcionales y no funcionales IEEE 830. 75
11.2 ANEXO B – PROTOTIPO UBICAR EN MARVEL APP. 75
11.3 ANEXO C – DIAGRAMA NOSQL UBICAR. 75

TABLA DE FIGURAS

Figura 1. Capturas de pantalla de la aplicación Mi ruta 14
Figura 2. Captura de pantalla de la aplicación Los Móviles 15
Figura 3. Pantalla principal de Uber 16
Figura 4.Panel Marvel App, proyecto Ubicar. 20
Figura 5. Canvas Marvel App, del lado izquierdo los wireframes, centro layout, derecha propiedades del layout. 21
Figura 6. Login Ubicar 22
Figura 7. Interfaz principal Ubicar 23
Figura 8. Características de una ruta programada 23
Figura 9. Características de una rutaGPS en mapa geográfico. 24
Figura 10. Resultados encuesta, pregunta 1 25
Figura 11. Resultados encuesta, pregunta 2 26
Figura 12. Resultados encuesta, pregunta 3 26
Figura 13. Resultados encuesta, pregunta 4 27
Figura 14. Resultados encuesta, pregunta 5 27
Figura 15. LinearLayoutManager en Recyclerview 31
Figura 16. LinearLayoutManager en Recyclerview 32
Figura 17. LinearLayoutManager en Recyclerview 32
Figura 18. Servicios que provee la base de datos Firebase 35
Figura 19. JSON usuario 37
Figura 20. JSON Conductores 37
Figura 21. JSON servicio_peticion 39
Figura 22. JSON amigos 40
Figura 23. Esquema A - Ubicar 41
Figura 24. Esquema B - Ubicar 42
Figura 25. Dispositivo A - Ubicar actualizando datos 43
Figura 26. Firebase interactuando con un servidor aparte y utilizando Rest Api. ..44
Figura 27. Metodo onLocationChanged de LocationListener 46
Figura 28. Objeto conductor_con_gps leer y escribir con Firebase 47
Figura 29. Verifica si tiene conexión a internet, si es así, ejecuta el método signInWithEmailandPassword() 48
Figura 30. Versión 1,2 y 3 de layout ingresar al sistema 51
Figura 31. Versiones de diseño del menú durante el desarrollo del proyecto 51
Figura 32. Ubicar mostrando automóviles en tiempo real en Google Maps y sus actualizacion de UI durante el desarrollo 53
Figura 33.Ubicar alojado en Bitbucket y sincronizado con Git 54

Figura 34. Commits proyecto Ubicar....55
Figura 35. historial bloqueos en la aplicación....56
Figura 36. informe de errores por Firebase Crashlytics....57
Figura 37. comportamiento de vistas por la consola de Firebase....58
Figura 38. login y registro Ubicar....61
Figura 39. Menú Ubicar....62
Figura 40.Soy conductor Ubicar....63
Figura 41. Agregar ruta....64
Figura 42. Ruta en línea / Pasajeros....65
Figura 43. Google Maps con automóviles en línea y solicitar una ruta....66
Figura 44. Automóviles que saldrán en una hora próxima....67
Figura 45. Solicitudes Ubicar....68
Figura 46. mi conductor Ubicar....69
Figura 47. Contacto con Google-support....70
Figura 48. Comunidad Firebase en Slack....71

Nota de Aceptación:

_____
_____
_____
_____
_____
_____
_____
_____
_____

_____
Firma del director

_____
Firma del calificador

_____
Firma del calificador

RESUMEN GENERAL DE TRABAJO DE GRADO

TITULO: Desarrollo de una aplicación móvil que ayude a coordinar la comunicación de la comunidad de transporte de la UPBBGA utilizando bases de datos Nosql para mostrar información en línea.

AUTOR(ES): Fabian Humberto Castillo Pineda

PROGRAMA: Facultad de Ingeniería de Sistemas e Informática

DIRECTOR(A): René Rodriguez Clavijo

RESUMEN

Ubicar es una aplicación móvil para la UPB de Bucaramanga que coordina la comunicación entre las personas de la de comunidad de transporte de la UPBBGA que necesitan movilizarse desde sus casas a la Universidad Pontificia Bolivariana o viceversa, la aplicación permite que cualquier usuario sea conductor o pasajero. Cuando el usuario activa su rol de conductor, puede guardar y publicar sus rutas, controlar pasajeros, cancelar viajes y programar rutas. A la misma vez los usuarios como pasajeros que necesitan el servicio, pueden ver información de las rutas publicadas en línea que circulan en la ciudad de Bucaramanga en un mapa de geolocalización y rutas que saldrán pronto. Dichos usuarios podrán solicitar el servicio que les sirva para llegar a sus destinos. El proyecto fue realizado siguiendo la metodología ágil Scrum y utilizando el lenguaje de programación Java, se usó una base de datos Nosql en la nube integrando el sdk de Firebase desarrollado por Google e implementando técnicas de experiencia de usuario y el estilo de material design. Se utilizaron los espacios brindados por la facultad de Ingeniería de Sistemas e Informática de la Universidad Pontificia Bolivariana seccional Bucaramanga para el desarrollo del proyecto.

PALABRAS CLAVE: Scrum, Java, Nosql, Firebase SDK, Experiencia de usuario, Material design.

V° B° DIRECTOR DE TRABAJO DE GRADO

GENERAL SUMMARY OF WORK OF GRADE

TITLE: Development of a mobile application that helps coordinate the communication of the UPBBGA transport community using the Nosql databases to display information online.

AUTHOR(S): Fabian Humberto Castillo Pineda

FACULTY: Facultad de Ingeniería de Sistemas e Informática

DIRECTOR: René Rodriguez Clavijo

ABSTRACT

Ubicar is a mobile application for the UPB of Bucaramanga that coordinates the communication between the people of the transport community of the UPBBGA who need to move from their homes to the Universidad Pontificia Bolivariana or vice versa, the application allows any user to be a driver or passenger. When the user activates his role as a driver, he can save and publish his routes, control passengers, cancel trips and schedule routes. At the same time users such as passengers who need the service, can see information of the routes posted online that circulate in the city of Bucaramanga on a geolocation map and routes that will come soon. These users may request the service that will help them to reach their destinations. The project was made following the Agile Scrum methodology and using the Java programming language, a Nosql database was used in the cloud integrating the Firebase sdk developed by Google and implementing user experience techniques and material design style. The spaces provided by the Faculty of Systems and Information Engineering of the Universidad Pontificia Bolivariana Bucaramanga section were used for the development of the project.

KEYWORDS:

Scrum, Java, Nosql, Firebase SDK,User experience , Material design.

V° B° DIRECTOR OF GRADUATE WORK

1. INTRODUCCIÓN

Las aplicaciones móviles en la actualidad son una herramienta muy importante para las personas y sus tareas cotidianas. Las aplicaciones enfocadas para la Movilidad tienen una alta demanda debido a que la mayoría de las personas utilizan servicios de transporte público o privados y requieren de información en línea sobre la movilidad y control de estos servicios en sus dispositivos móviles. Aplicaciones como Easytaxi y Uber enfocadas al sistema de transporte privado cuentan con más de 500.000 usuarios registrados en el sistema, facilitando la relación entre el cliente y conductor para solicitar servicios de transporte privado que requiere a diario las personas para movilizarse hacia sus destinos, facilitando la información de los automóviles prestadores de servicio disponibles en línea.

En la Universidad Pontificia Bolivariana de Bucaramanga se creó una comunidad entre los mismos estudiantes para movilizarse hacia la universidad, por medio de la red social Facebook, llamado Ucar. Dicha comunidad nació para disminuir los problemas de demoras, imprevistos e incomodidades que experimentan a diario la comunidad UPB en el transporte público masivo del área metropolitana de Bucaramanga.

Esta tesis busca integrar diferentes tecnologías como bases de datos NOSQL en la nube con Firebase utilizando métodos y técnicas de experiencias de usuario(UX) para desarrollar una aplicación móvil que permita obtener ubicaciones del servicio de transporte “Ucar” en línea para disminuir los problemas de comunicación y demoras al momento de solicitar el servicio y de esta manera facilitando la relación entre estudiantes (conductores) y estudiantes que requiere el servicio.

2. RESUMEN DEL PROYECTO

2.1 DEFINICIÓN DEL PROBLEMA

La comunidad de la Universidad Pontificia Bolivariana Seccional Bucaramanga ha presentado dificultades a la hora de llegar a las instalaciones que se encuentran en el kilómetro 7 vía Piedecuesta, afectando a los estudiantes, egresados, administrativos, visitantes y docentes que viven en los diferentes municipios del área

metropolitana de Bucaramanga como Floridablanca, Girón, Piedecuesta y Bucaramanga.

Las dificultades de la familia UPB, comienzan desde el momento que una persona desea llegar al campus universitario sin saber exactamente las rutas y las posiciones de los medios de trasporte que tienen a disposición como: Buses públicos, taxis o transporte público informal, lo que conlleva, a que la comunidad pierda tiempo en estaciones o paradas, debido a la difícil tarea de predecir o saber en qué preciso momento pasará un servicio de transporte público cerca de sus hogares o en una parada especifica.

Las personas que viven en la zona centro de Bucaramanga no acuden a tomar buses como TransPiedecuesta, Viila San Carlos o Transgiron que se dirigen hacia Piedecuesta y pasan por la UPB, debido a que los buses deben tomar la carretera antigua o el anillo vial como ruta principal, haciendo muy largo el recorrido y por ende hacer que las personas en ocasiones lleguen tarde a sus respectivos compromisos dentro del campus universitario. Esto hace que la comunidad use servicios como Metrolinea, transportes informales y transportes privados para movilizarse hacia la universidad, teniendo de igual manera dificultades como sobrecupo, pérdida de tiempo en transbordos, inseguridad y dependencia de una tarjeta inteligente para bordar los buses de Metrolinea.

Aplicaciones como UBER o Easy Taxi que se enfocan en un cierto grupo de usuarios para prestar servicios de transporte privado “puerta a puerta”, donde integran de una forma agradable al conductor y las persona que utiliza el servicio controlando la información del automóvil en línea. De todas maneras, el grupo de usuarios de dichas aplicaciones deben contar con suficiente presupuesto para tomar estos servicios, por lo que no es una opción accesible para toda la comunidad en su día a día.

Además de todos los problemas mencionados anteriormente, los servicios de transporte público en Bucaramanga y su área metropolitana no cuentan con aplicaciones móviles que informen a sus usuarios aspectos relacionados con sus rutas, recorridos, destinos y demás información actualizada para tomarlos oportunamente de manera clara, ordenada y comprensible.

2.2 JUSTIFICACIÓN

El proyecto busca ayudar a la comunidad de la UPB facilitando un canal de comunicación entre las personas que cuentan con un automóvil (Conductores), y las personas que utilizan el transporte público o aplicaciones de transporte privado, para movilizarse desde sus hogares hacia el campus universitario y viceversa.

El canal de comunicación será una aplicación móvil, donde los conductores podrán gestionar de forma ordenada sus diferentes rutas, horarios, puntos de partida/llegada y cupos disponibles, además de poder conocer la lista de pasajeros confirmados y sus respectivos puntos de paradas. Por otro lado, los pasajeros podrán listar las rutas programadas por los conductores, reservar un cupo en las mismas y observar las posiciones de los automóviles activos en línea en un mapa geográfico.

Además de facilitar un canal de comunicación, el proyecto busca una mayor integración e inclusión entre sus miembros, proponiendo un medio de transporte comunitario privado para la familia de la Universidad Pontificia Bolivariana seccional Bucaramanga.

3. OBJETIVOS

3.1 Objetivo General.

Desarrollar una aplicación móvil centrada en la usabilidad, utilizando bases de datos NoSQL en la nube para apoyar el proceso de coordinación de movilidad en línea de la Universidad Pontificia Bolivariana Seccional Bucaramanga.

3.2 Objetivos Específicos.

- Establecer los requerimientos funcionales, no funcionales y de usabilidad utilizando técnicas de UX (Experiencia de Usuario).
- Estructurar los datos NoSql, para almacenar y recuperar la información ubicaciones, usuarios, rutas y horarios.
- Integrar la base de datos NoSQL Firebase con la aplicación móvil desarrollada en el lenguaje de programación Java, para sincronizar los datos en línea.

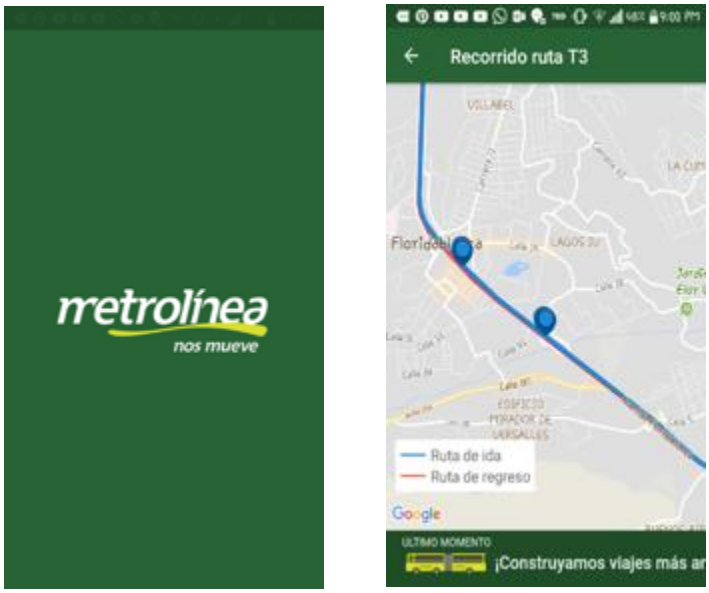
- Validar los requerimientos y la usabilidad de la aplicación, a través de pruebas de usuario.

4. ANTECEDENTES

- Aplicación móvil para el sistema de transporte masivo Metrolínea llamada “Mi ruta - Metrolínea APK” (Plataformas: IOS/ANDROID).

Aplicación móvil la cual sirve como herramienta para las personas de Bucaramanga que utilizan el servicio de la empresa Metrolinea. Su principal funcionamiento se basa en las consultas de rutas y paradas, en un mapa geográfico donde muestran todas las rutas del sistema. A continuación, se muestra en la figura 1 las capturas de pantalla de la aplicación nombrada

Figura 1. Capturas de pantalla de la aplicación Mi ruta



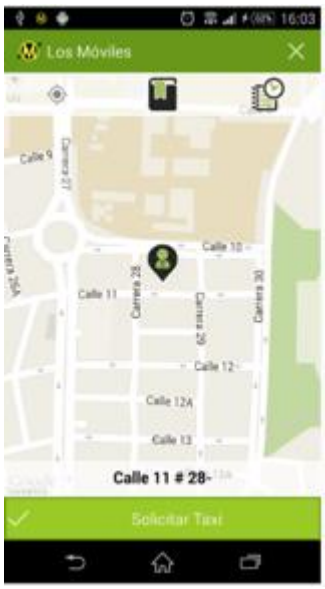
Fuente autor.

- Aplicación móvil para el sistema de taxis en Bucaramanga llamada “Los Móviles”

La aplicación móvil llamada “Los Móviles”, desarrollada en Bucaramanga, permite a los usuarios buscar un taxi confiable y seguro lo más cercano a su posición actual.

Para asegurar la confianza del taxi, la empresa valida que los conductores registrados en el sistema cuenten con la licencia de conducción al día y tengan su SOAT actualizado, además, la aplicación ofrece a quien toma el servicio de taxi, cierta tranquilidad ya que el pasajero puede ver la información del conductor e interactuar con él.

Figura 2. Captura de pantalla de la aplicación Los Móviles



Fuente autor.

- Aplicación móvil *internacional* para el sistema de *transporte privado* llamada "*Uber*"

"…Usando la aplicación, los usuarios que necesitan transporte consiguen fácilmente encontrar socios conductores que ofrecen este servicio. Uber ofrece una opción más para moverse por la ciudad, con más estilo, seguridad y comodidad que nunca. Presente en más de 310 ciudades alrededor del mundo." – Uber Chile. [2]

Figura 3. Pantalla principal de Uber

Fuente: https://goo.gl/nJhLxr

5. MARCO TEORICO

5.1 METODOLOGÍA PARA DESARROLLO DE SOFWTARE

Para realizar el proyecto en el tiempo indicado y a su vez obtener el resultado esperado es importante utilizar algunas metodologías que faciliten su dicha construcción, es por eso que se realiza la metodología Scrum el cual es un marco de trabajo para desarrollar, entregar y mantener productos complejos.

Scrum integra técnicas de gestión de producto y técnicas de trabajo de modo que se pueda mejorar continuamente las características del producto, el equipo de trabajo y el entorno donde se desarrolla el proyecto. [3]

El marco de trabajo Scrum consiste en los equipos Scrum y sus roles, eventos, artefactos y reglas asociadas. Cada uno de los componentes dentro del marco de trabajo sirve a un propósito especifico y es muy esencial para alcanzar el éxito del

proyecto y su buen uso. El quipo Scrum está integrado por el dueño del producto, el equipo de desarrollo y un Scrum Master. [3]

El proceso está fundamentado en Sprints, lo cual se caracteriza por cuatro eventos formales contenidos dentro del Sprint, tal y como se describen a continuación.

- Planificación del Sprint (*Sprint Planning*)
- Scrum Diario (*Daily Scrum*)
- Revisión del Sprint (*Sprint Review*)
- Retrospectiva del Sprint (Sprint Retrospective)

El trabajo a realizar durante un Sprint se incorpora en la planificación de Sprint. Este plan se crea mediante el trabajo colaborativo del equipo Scrum. La planificación del Sprint responde a las siguientes preguntas. ¿Qué puede entregarse en el incremento del Sprint que comienza? y ¿Cómo se conseguiría hacer el trabajo necesario para entregar el incremento? [3]

El Scrum diario es una reunión con un bloque de cierto tiempo ajustado en el calendario, donde se planea el tiempo de trabajo que se le dará al sprint; luego de verse un incremento del producto se hace la revisión del producto con sus posibles mejores para luego tomar la retrospectiva del producto y terminar con el objetivo del sprint.

5.2 EXPERIENCIA DE USUARIO (UX) EN EL DESARROLLO DEL APLIVATIVO MOVIL UBICAR

La experiencia de usuario es un conjunto de factores y elementos relativos a la interacción del usuario, cuando este interactúa con un dispositivo o entorno completo.

La experiencia de usuario no solo depende de los factores del diseño (hardware, software, usabilidad, diseño de interacción, diseño gráfico entre otras) sino busca también aspecto como las emociones que produce el usuario al utilizar el aplicativo móvil, sentimientos, construcción, confiabilidad del producto entre otros aspectos.

5.2.1 Factores que se tienen en cuenta para mejorar la experiencia de usuario

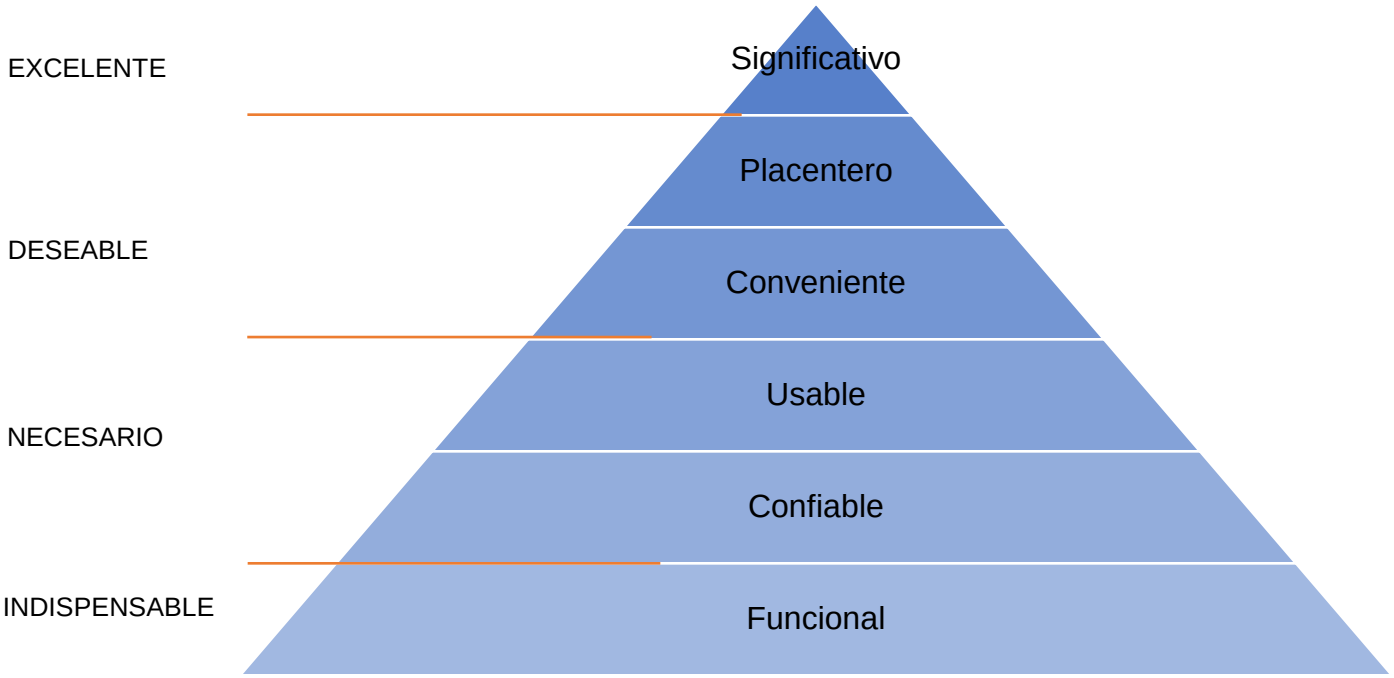
Para el desarrollo del aplicativo móvil Ubicar se tuvieron en cuenta los siguientes indicadores, para cumplir con una Experian de usuaria fluida.

- EL diseño visual del aplicativo móvil.
- El tiempo de carga del aplicativo móvil.
- El tiempo que el usuario pasa e interactúa en él.
- Satisfacción de usuario.
- Usabilidad del aplicativo móvil.

IPO (interacción, persona-ordenador)

La IPO es un área de estudio centrada en el fenómeno de interacción entre usuarios y sistemas informáticos. Cuyo objetivo es dar a conocer bases teóricas, metodologías y prácticas para el diseño y evaluación de productos interactivos el cual pueden ser utilizados de forma eficiente, eficaz, segura y satisfactoria. [4]

Las necesidades que debe cumplir según la IPO se encuentran planteadas a continuación, algunas más importantes que otras. En el grafico siguiente se expone por orden de importancia las necesidades que se deben cumplir según la IPO.



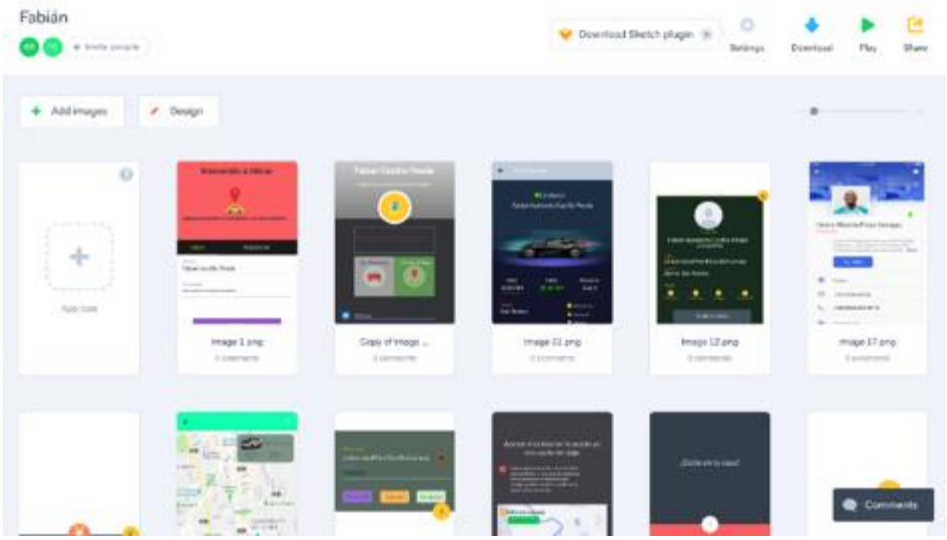
- Significativo: Integrado en la vida del usuario.
- Placentero: Genera emociones a uno mismo.
- Conveniente: Funcional tal como el usuario espera.
- Usable: Puede ser usado sin dificultad.
- Confiable: Funciona de manera consistente.
- Funcional: Cumple con su función.

5.3 PROTOTIPO FUNCIONAL Y DISEÑO DE VISTAS DEL APLICATIVO MÓVIL CON MARVEL APP.

Marvel App es una herramienta para crear prototipos interactivos que involucran plataformas digitales. Marvel ofrece una herramienta llamada Canvas para diseñar los prototipos funcionales que queramos crear, donde podremos declarar cada vista que compone el aplicativo móvil declarando funciones y diferentes comportamientos que quiera la app, Marvel ofrece funciones como:

- Swipe left: Deslizar a la izquierda
- Swipe rigth: Deslizar a la derecha
- Swipe up: Deslizar arriba
- Swipe down: Deslizar abajo
- Pinch: Juntar dedos.

Figura 4.Panel Marvel App, proyecto Ubicar.



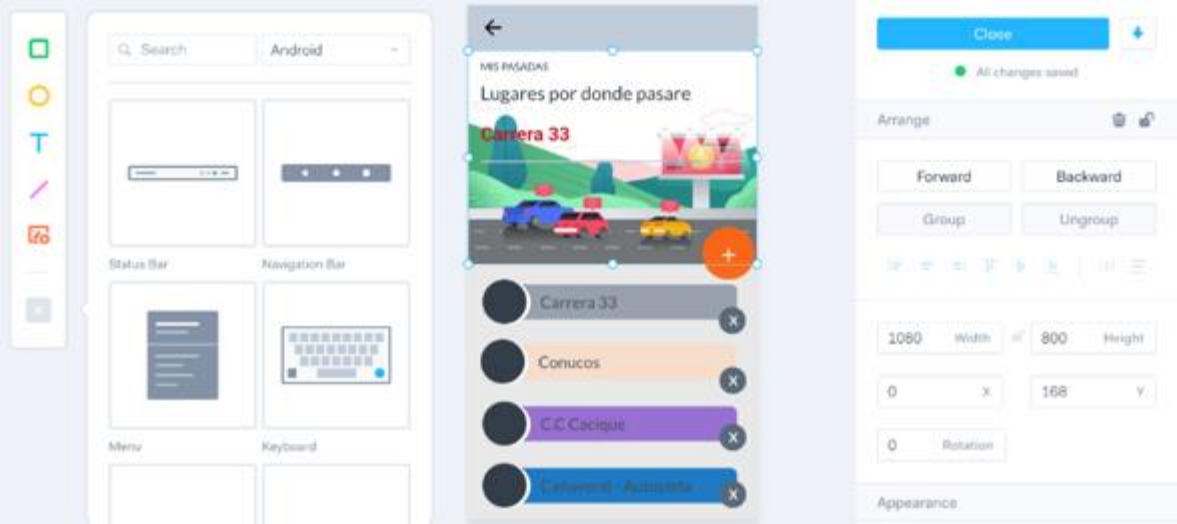
Fuente autor.

Wireframes de Marvel App

La herramienta nos ofrece un conjunto de componentes de diseño que hacen parte del sistema operativo Android, que se usaron para crear el prototipo y tener los layouts específicos que queremos utilizar en el proyecto. Para el prototipo funcional de Ubicar se utilizaron algunos wireframes como:

- Status Bar
- Navigation Bar
- Menu
- Floating Button
- Tabs
- Bottom Sheet
- Spinner
- Time Picker
- Card
- Chip

Figura 5. Canvas Marvel App, del lado izquierdo los wireframes, centro layout, derecha propiedades del layout.



Fuente autor.

5.3.1 Interfaz gráfica Ubicar

Con el prototipo de Marvel fue creado el diseño del aplicativo móvil Ubicar que fue cambiado y adaptado después de obtener el resultado de las encuestas hechas de usabilidad y experiencia de usuario en la Universidad Pontificia Bolivariana de la ciudad de Bucaramanga, a continuación, se presentan algunas vistas.

Como se muestra en la figura No 6. Se puede visualizar el *login* del aplicativo móvil que tiene dos secciones principales.

Login el cual ingresa con el usuario y su contraseña. El registro donde ingresa los campos que son de requisito para registrarse en el sistema. Se puede apreciar el uso de Tablayout, Appbarlayout, Viewpager que son componentes de Android.

Figura 6. *Login* Ubicar

Fuente autor.

Cuando el usuario ingresa en el aplicativo móvil, se encuentra con la interfaz principal de la plataforma como lo muestra en la figura No 7. Desde esta vista el usuario puede acceder a ser pasajeros, ver automóviles Ubicar que circulan en la ciudad de Bucaramanga en un mapa geográfico, Ver noticias UPB, estados, ajustes de usuarios.

En el BottomNavigationView podemos observar 3 ítems los cuales el primero pertenece a la lista de rutas programadas (rutas que saldrán en una hora próxima), el segundo hace referencia al Home del aplicativo móvil (pantalla principal) y como tercer ítem a notificaciones.

Figura 7. Interfaz principal Ubicar

Fuente autor.

Como se muestra en la figura No 8. El usuario ve las características de una ruta programada que saldrá en las próximas horas.

Figura 8. Características de una ruta programada

Fuente autor.

Terminando en la figura No 9. Se puede evidenciar en el mapa de Google maps los automóviles que se dirigen hacia la Universidad Pontificia Bolivariana, con su respectiva característica de ruta. El usuario puede filtrar automóviles, ver información de cada ruta como salida, llegada, hora de salida, cantidad de pasajeros entre otras características y solicitar un servicio que desee y se ajuste a su ruta.

Se utilizan componentes de Android como CoordinatorLayout y NestedScrollview para mostrar la información de cada ruta que contiene un automóvil dinámicamente y que no afecte el mapa y la visibilidad de el mismo.

Figura 9. Características de una rutaGPS en mapa geográfico.

Fuente autor.

Prototipo funcional echo en Marvel App de la aplicación Ubicar, en él se evidencian las vistas del aplicativo Móvil, dichas vistas fueron remodeladas y cambiadas después de los resultados de las encuestas de usabilidad y amigabilidad del software Ubicar.

Para ver el prototipo Marvel de la aplicación móvil Ubicar, Ver Anexo B

5.4 ENCUESTAS DE USABILIDAD Y AMIGABILIDAD DEL SOFTWARE UBICAR EN LA UNIVERSIDAD PONTIFICIA BOLIVARIANA DE BUCARAMANGA.

Se realizaron 50 encuestas a estudiantes de la universidad Pontifica Bolivariana con el fin de evidenciar su comportamiento y sentimiento que fue generado en los estudiantes al momento que utilizaron el prototipo del aplicativo móvil, esto con el fin de obtener su experiencia de usuario y sugerencia para ser tomados en cuenta y adaptarlos al producto final.

Antes de realizar las encuestas se les informo a los estudiantes que el estudio era de experiencia de usuario y se les informo un poco sobre el tema, los colores y diseño del aplicativo móvil cambiarían después de las sugerencias dadas por los estudiantes para ser puestos en práctica en el diseño final.
Las encuestas se realizaron en la fecha y lugar indicado a continuación.

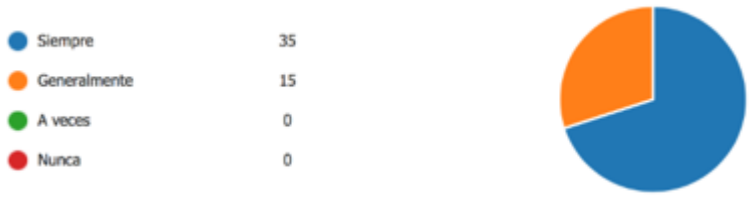
- Fecha: 15/Febrero/2018
- Lugar: Edificio I salón 203 – Universidad Pontificia Bolivariana / Laboratorios facultad de ingeniería de Sistemas e Informática.
- Encuestas realizadas en Google Forms.

La encuesta contiene 7 preguntas, se puede evidenciar las preguntas establecidas

1. ¿La distribución de los elementos gráficos como botones, listas, imágenes y flujo de la aplicación fue adecuada a sus necesidades?

    - Siempre
    - Generalmente
    - A veces
    - Nunca

Figura 10. Resultados encuesta, pregunta 1



Fuente autor

2. ¿Pudo interpretar el propósito del software sin dificultades después de la interacción?

    - Si
    - No

Figura 11. Resultados encuesta, pregunta 2

Fuente autor

3. ¿De 1 a 5 puedo identificar los roles como conductor y pasajero en la aplicación?

    - El promedio está por encima de 4.5 quiere decir que los estudiantes sintieron una buena fluidez y pudieron identificar los dos roles importantes del aplicativo móvil y como se comportaban.

Figura 12. Resultados encuesta, pregunta 3

Fuente autor

4. ¿Usted está cómodo sabiendo que usted como pasajero puede ver el nombre completo de los otros pasajeros?

    - Si
    - No

Figura 13. Resultados encuesta, pregunta 4

Fuente autor

5. ¿Usted estaría dispuesto a utilizar y recomendar la aplicación a otros miembros de la comunidad UPB?

    - El prototipo del aplicativo Ubicar fue muy bien aceptado por los estudiantes de la universidad Pontifica Bolivariana, se vieron motivados el momento de ver el producto y no tienen ningún problema en recomendar la aplicación con otros estudiantes de la universidad.

Figura 14. Resultados encuesta, pregunta 5

Fuente autor

6. ¿Sin tomar en cuenta los colores y diseños usted que elementos cambiaria?

    - En esta pregunta se recibieron comentarios de los estudiantes que realizaron la encuesta para tener en cuenta sus comentarios y ponerlos en práctica, en la siguiente tabla se muestra algunas respuestas.

ID	Nombre	Respuesta
1	Anónimo	No tanta información, puesto que dificulta entender, y el tamaño de algunos botones.
2	Anónimo	Me parece que la estructura está bien

		planeada, igual que el servicio que se piensa prestar, ya que a todos los de la universidad nos beneficia, de la parte del diseño está bien distribuida, no esta tan cargada visualmente, haciendo esto una aplicación amigable al usuario también es muy importante y me pareció interesante que muestre la capacidad, el nombre del conductor y de los acompañantes.
3	Anónimo	No cambiaria nada, nos da un muy buen manejo y se entiende muy bien.
4	Anónimo	La estructura de solicitud
5	Anónimo	Me parece que se ve con demasiada información, entre mas sencilla mejor.

7. ¿Omitiendo el color y diseño, si tu fueras un conductor te parece bien el proceso de agregar una ruta?

    - En esta pregunta se recibieron comentarios de los estudiantes que realizaron la encuesta para tener en cuenta sus comentarios y ponerlos en práctica, en la siguiente tabla se muestra algunas respuestas.

ID	Nombre	Respuesta
1	Anónimo	Si, porque facilita a los pasajeros más rutas.
2	Anónimo	El proceso de agregar la ruta está bien, ya que permite agregar de donde sale a donde llega, la hora en que sale, aparte también te

		deja agregar paradas para que los usuarios sepan por donde pasara el conductor.
3	Anónimo	De hecho, eso fue una de las cosas que más me gusto. El hecho de que el conductor pueda designar una ruta y no tener que agregarlas todos los días me parece una muy buena opción.
4	Anónimo	Un label que me indique el estado de la ruta, ya que la publicación de la ruta no se realiza hasta que no confirmo, dándole clic a la ruta y publicar.
5	Anónimo	Si, así de esta manera cada vez que vaya a prestar el servicio no voy a tener que estar creando la ruta.

Gracias a las encuestas de experiencia de usuario se observó el comportamiento del aplicativo móvil, como los usuarios interactuaban sobre el y si se lograba el objetivo de ser un aplicativo amigable. Según los resultados de las encuestas resulto ser amigable, practico y de buen uso para los estudiantes de la Universidad Pontifica Bolivariana. Sus comentarios y sugerencias fueron de gran utilidad y aplicados en el producto final.

5.5 TECNOLOGIAS

A continuación, se presentan las tecnologías seleccionadas para construir la aplicación móvil Ubicar.

5.5.1 Frontend

XML: Extensible Markup Languge (XML) es un formato que es universal para datos y documentos estructurados. Al igual que HTML, XML utiliza etiquetas para estructurar los datos del documento. [5]

Características de XML

- XML es un subconjunto de SGML en la cual incorpora tres características muy importantes que son extensibilidad, estructura, validación.
- Basado en texto.
- Orientado a contenidos y no a presentación.
- Las etiquetas son creadas para crear los documentos.
- No es sustituto de HTML.
- No tiene un visor genérico de XML.

El diseño de las interfaces del proyecto en Android está declaradas en XML que permite separar mejor los dos componentes que son la presentación de un aplicativo móvil y el código fuente que controla su comportamiento. [5]

Una característica importante de utilizar XML en el UI del aplicativo móvil es que cada descripción XML es externa al código fuente del software, lo que significa que pueden ser modificados o adaptados sin necesidad de modificar el código fuente o volverse a compilar.

A continuación, se puede evidenciar un cuerpo XML declarado en Android que contiene algunos componentes de layout que hacen parte del sistema operativo Android en este caso podemos evidencia un LinearLayout con una orientación vertical conteniendo dos etiquetas de diseño un Textview y un Botón.

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <LinearLayout xmlns:android="http://schemas.android.com/apk/res/and
   roid"android:layout_width="match_parent"
3. android:layout_height="match_parent"
4. android:orientation="vertical">
5.
6.       <TextView android:id="@+id/text"
7.       android:layout_width="wrap_content"
8.       android:layout_height="wrap_content"
9.       android:text="Hola, soy un textview" />
10.      <Button android:id="@+id/button"
11.      android:layout_width="wrap_content"
12.      android:layout_height="wrap_content"
13.      android:text="Hola, soy un boton" />
14.      </LinearLayout>
```

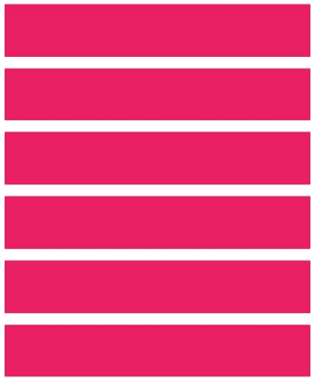
Recyclerview-v7: Un Recyclerview es un contenedor de elementos de Android en forma de lista, al igual que la clase Listview, estos dos componentes hacen prácticamente la misma función, pero Recyclerview permite reciclar los ítems que ya no son visibles por el usuario debido al scrolling.

Para utilizar este componente de Android se agrega la librería mediante gradle con la cita “com.android.support:recyclerview-v7:+” es recomendable utilizar la versión actualiza que ofrece la librería.

LayoutManager: Existen tres tipos de LayoutManager que son incluidos en la librería Recyclerview-v7 las cuales son LinearlayoutManager, GridLayoutManager y StaggeredGridLayoutManager los cuales ayudan indicando la forma en la que deberían ser mostrados los elementos.

En la figura No 15. Podemos notar la definición de un LinearLayoutManager como una colección de filas que se presentara en el Recyclerview.

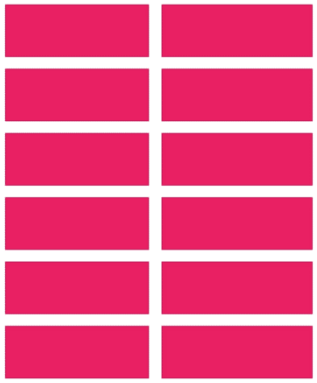
Figura 15. LinearLayoutManager en Recyclerview



Fuente autor

En la figura No 16. Se logra evidenciar un GridlayoutManager como una colección de filas separadas por dos columnas que se presentara en un Recyclerview.

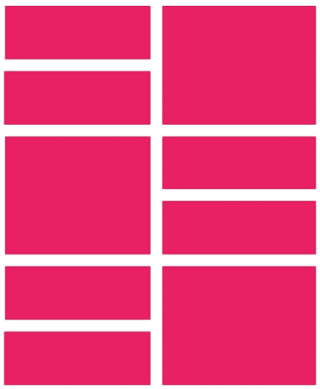
Figura 16. LinearLayoutManager en Recyclerview



Fuente autor

En la figura No 17. Se evidencia un StaggeredGridLayoutManager como una colección de filas igual al GridlayoutManager pero el tamaño de cada ítem puede variar dinámicamente en un Recyclerview.

Figura 17. LinearLayoutManager en Recyclerview



Fuente autor

FirebaseUI: Es una librería de código abierto que nos permite conectar componentes de diseño con el api de Firebase. Al momento de utilizar un recyclerview, listview y otros componentes de UI de Android, Firebase UI nos ayuda a conectar y sincronizar los datos que son alojados en la base de datos en tiempo real de Firebase y mostrarlos más fácilmente en la interfaz gráfica.

Picasso: Las imágenes en un aplicativo móvil ocupan mucho espacio por lo tanto esta librería nos ayuda a cargar imágenes sin problemas y sin ocupar mucho espacio. Picasso se encarga de corregir una gran cantidad de errores que se presentan al momento de cargar una imagen en nuestro dispositivo móvil.

Picasso cuenta con muy buenas características, algunas de las más importantes, esta librería nos permite administrar el reciclado de las imágenes y cancelar la descarga de un adaptador si algo sale mal, Picasso también transforma imágenes complejas (demasiada densidad de pixeles) con un uso de memoria mínimo del dispositivo y tiene un caching automático en la memoria y disco del dispositivo móvil.

CircularimageView: Librería que con un poco de código XML nos permite darle un efecto circular y agradable a las imágenes que conforman el aplicativo móvil.

5.5.2 Base de datos

Bases de datos NoSQL: También llamadas No Solo SQL, están enfocadas hacia la gestión de datos y el diseño de base de datos, que es útil para grandes conjuntos de datos distribuidos que necesitan ser accedidos remotamente. [6]

NoSQL, busca resolver los problemas que se encuentran al desarrollar software que contienen una gran cantidad de datos y que necesitan ser actualizados frecuentemente de manera rápida, teniendo un enfoque de escabilidad y rendimientos del big data de las cuales las bases de datos relacionales no fueron diseñadas para abordar. NoSQL es por lo general útil cuando empresas grandes necesitan acceder y analizar grandes cantidades de datos no estructurados o datos que se almacenan de forma remota en varios servidores virtuales en la nube. [6]

Ventajas de los sistemas NoSQL

- Se ejecutan en máquinas con pocos recursos ya que a diferencia de lo sistemas basados en SQL no requiere de mucho procesamiento.

- ∉ Escalabilidad horizontal: Para mejorar el rendimiento de estos sistemas simplemente se consigue añadiendo más nodos, con la única operación de indicar al sistema cuáles son los nodos que están disponibles. [7]
- ∉ Puede manejar una gran cantidad de datos debido a que utiliza una estructura distribuida mediante tablas Hash. [7]
- ∉ No genera cuellos de botella

Firebase-Google

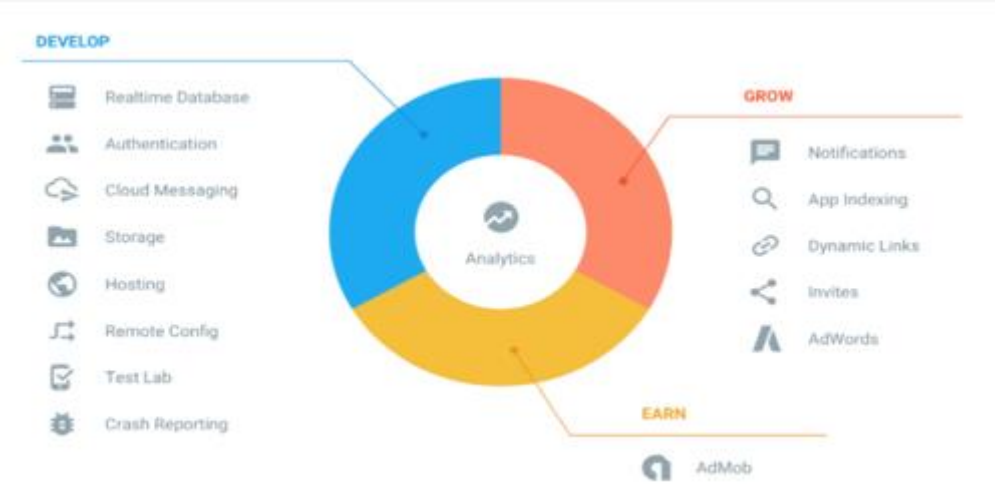
Plataforma de desarrollo móvil y web en la nube de google capaz de proveernos de un backend en la nube con una base de datos en tiempo real conteniendo librerías para acceder a ella desde aplicaciones Web, IOS y Android.

Según en el portal web oficial de Google define a Firebase como “una base de datos remota, alojada en la nube capaz de ser accedida desde apps para dispositivos móviles evitando muchas tareas engorrosas al momento de sincronizar datos, de manera que el desarrollador este más pendiente del usuario para satisfacer las necesidades del cliente”. -Equipo Firebase/Google [7]

Características Firebase:

- Base de datos sincronizada en tiempo real(BaaS)
- No-SQL JSON
- Referencia por URL
- Autenticación
- Auto-escalable
- Puede trabajar offline

Figura 18. Servicios que provee la base de datos Firebase



Tomado de:https://goo.gl/pYTxVA

Servicios Firebase

Son servicios orientados a la creación de aplicaciones de alta calidad que nos ofrece Firebase para hacer aplicaciones de una manera más ordenada, optimizada y un poco más sencilla. Las aplicaciones móviles se sincronizan con Firebase para utilizar dichos servicios descritos a continuación como lo son pruebas de usuarios, ejecutar funciones en la nube, guardar datos a la escala de Google, obtener estadísticas de rendimiento entre otras funciones.

RealtimeDatabase	Almacena y sincroniza datos de app en milisegundos
Authentication	Autentifica usuarios de forma simple y segura

Cloud Storage	Almacena y envía archivos a la escala de Google
Test Lab para Android	Prueba la app en dispositivos alojados en Google
Cloud functions	Ejecuta código de back-end para administrar móviles y servidores
Hosting	Entrega recursos de aplicaciones web con velocidad y seguridad
CrashReporting	Busca y prioriza errores para solucionarlos más rápido
Supervisión del rendimiento	Obtiene estadísticas sobre el rendimiento de tu app

5.6 Estructura de la base de datos Nosql-Firebase

Todos los datos de *Firebase Realtime Database* se almacenan como objetos JSON, la base de datos a utilizar puede conceptualizarse como un árbol JSON alojado en la nube.

A diferencia de una base de datos relacional, en la base de datos Nosql no existen tablas ni registros. Al momento de agregar un dato al objeto JSON, este simplemente se convierte en un nodo de la estructura JSON existente con una clave

asignada. A continuación, se muestran algunas colecciones del aplicativo móvil Ubicar.

En la figura No 19. La estructura del JSON esta conformada por el nombre de la colección “usuarios” que guardará el identificador de cada usuario “key_usuario” el cual contendrá la información de cada usuario en el sistema.

Figura 19. JSON usuario

```
“usuarios”:{
“key_usuario”:
{
“token”:”fbSxFr-DVCo:APA91bHfbU2U3JDLJhdksk789GERTN_fgd-grhbGTpz9”,
“nombre”:”Fabian Humberto Castillo Pineda”,
“donde”:”Provenza-Autopista”,
“celular”:”3105687495”,
“carrera”:”ingeniería de Sistemas e Informática”
}
}
```

Fuente autor

En la figura No 20. Se evidencia la colección “Conductores” que guardara la información de los usuarios que activaron rutas con GPS con su información de trayectoria. Una colección pasajeros que contiene la cantidad de pasajeros se estará actualizando automáticamente cada vez que el usuario con una ruta activa acepte pasajeros, la capacidad máxima de pasajeros es de 4 personas. El campo rutas guarda en un array los lugares por donde pasara el conductor.

Figura 20. JSON Conductores

```
“Conductores”:{
“key_usuario”:
{
“salida”:”Cabecera”,
“llegada”:”Universidad Pontificia Bolivariana”,
“usuario”:”Fabian Humberto Castillo Pineda”,
“h_salida”:”4:20 PM”
“latitud”:”4.567”,
“longitud”:”3.6785”,
“celular”:”3105601567”,
“rutas”:[“carrera-33” , “conucos” , “cacique” , “provenza”],
“pasajeros”:{
“key_pasajero”:{
“donde”:”cacique-conjunto villavel”,
“nombre”:”Alvaro Alexander”,
“carrera”:”Ingenieria Civil”
}}
}
}
```


Fuente autor

En la figura No 21. Se observa la colección “servicio_peticion” la cual guardara las solicitudes enviadas al momento de un pasajero solicitar un conductor y enviar una notificación, los elementos del objeto JSON son eliminados cada vez que el usuario acepta o no acepta una solicitud, la estructura del JSON tendrá el identificador del conductor(key_conductor) con sus pasajeros y el identificador del pasajero(key_pasajero) con su conductor al cual fue enviado la solicitud de servicio.

Figura 21. JSON servicio_peticion

```
"servicio_peticion":{
"key_conductor":
{
"key_pasajero1": {
"key":"ABMNO789J9Anksjj8",
"estado":"recibido",
"usuario";"Carlos Alfonso"
}
},
"key_pasajero1":
{
"key_conductor":
{
"key":"JSJnkdjfkljsdidsNJSO78",
"estado":"enviado",
"usuario":"Fabian Castillo"
}
}

}
```

Fuente autor

Terminando en la figura No 22. Se observa la colección "amigos" la cual es creada cuando un usuario conductor acepta un pasajero, la colección guardara los conductores con sus pasajeros y los pasajeros con sus respectivos conductores.

Figura 22. JSON amigos

```
“amigos”:{
“key_conductor”:
{
“key_pasajero1”: {
“key”:”ABMNO789J9Anksjj8”,
“fecha”:”2/Febrero/2018”,
“donde”:”Provenza-autopista”,
“soy”:”pasajero”,
“usuario”;”Carlos Alfonso”
}
},
“key_pasajero1”:
{
“key_conductor”:
{
“key”:”JSJnkdjfkljsdidsNJSO78”,
“fecha”:”2/Febrero/2018”,
“usuario”:”Fabian Castillo”,
“salida”:”Cabecera”,
“llegada”: “Universidad Pontifica Bolivariana”,
“h_salida”:”3:50 PM”,
“rutas”:[“carrera 33 , cacique, cañaveral,”]
“soy”:”conductor”
}
}

}
```

Fuente autor

Diagrama base de datos NoSql-Ubicar

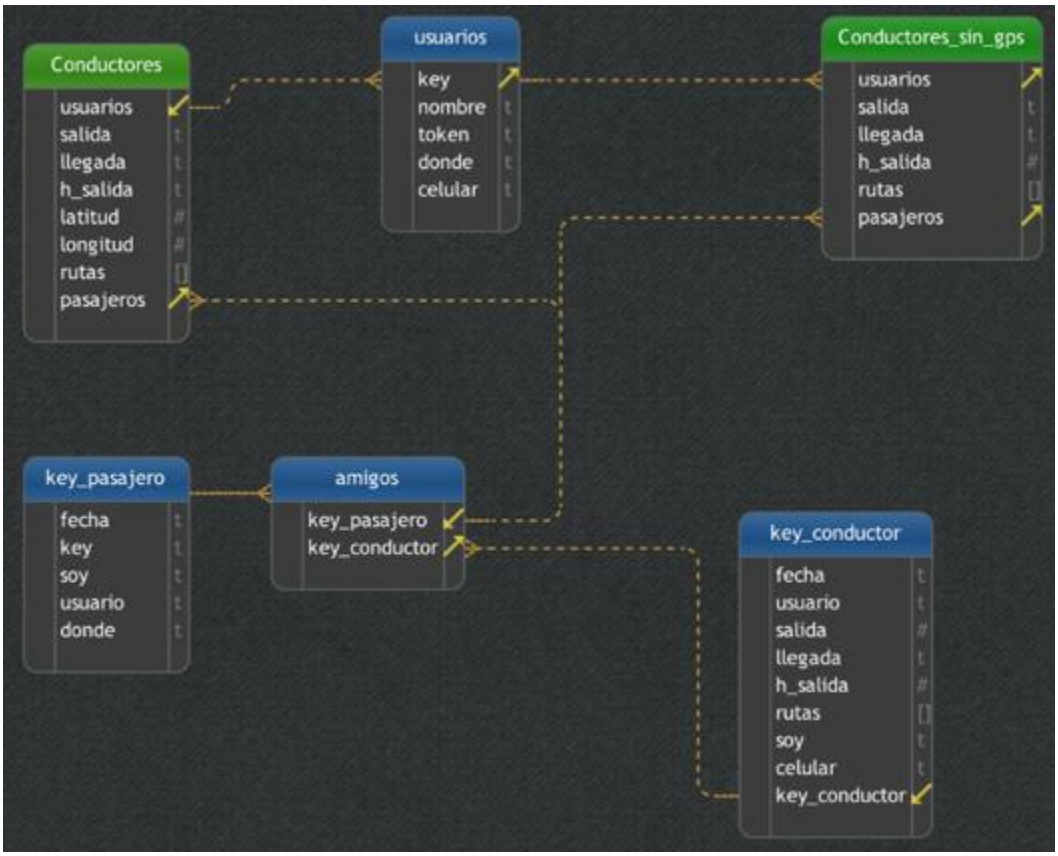
Los siguientes diagramas se encargan de mostrar las relaciones entre las colecciones de la base de datos Nosql. A continuación, se describen las relaciones de una forma entendible.

En la figura No 23. Se observa la colección “usuarios” con sus respectivos atributos. La colección conductores para rutas con gps obtiene la colección usuarios y añade atributos adicionales al objeto JSON como salida, llegada, latitud, longitud entre otros, de esta misma forma se implementa para conductores sin gps.

Al momento de un pasajero solicitar un conductor y ser aceptado, la colección “amigos” guardara las colecciones tanto del pasajero como la del conductor, los

usuarios que estén activos como conductores obtendrán la colección “key_pasajero” para ir actualizando sus respectivos pasajeros y cupos.

Figura 23. Esquema A - Ubicar

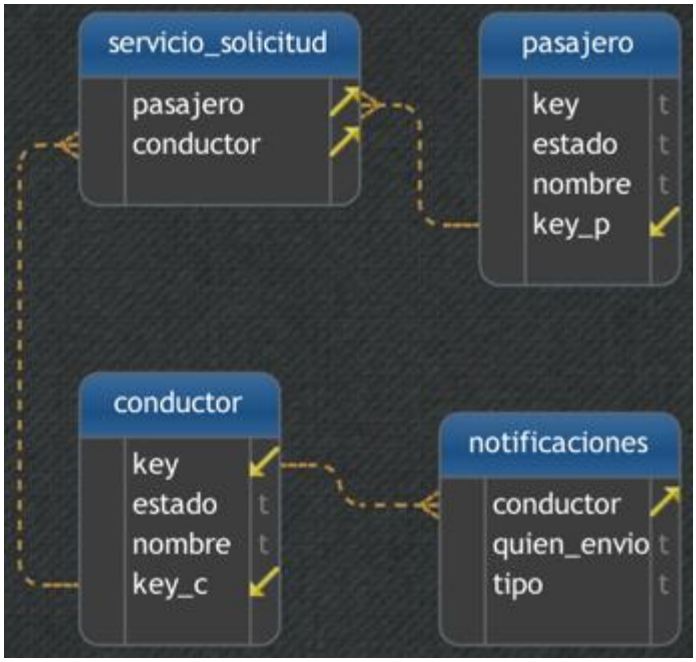


Fuente autor

En la figura No 24. Se encarga de mantener los datos cuando se envían solicitudes y notificaciones en el aplicativo móvil Ubicar. La colección “servicio_solicitud” mantiene la petición mientras el conductor lo acepta, el atributo estado indica si es enviado o recibido.

La colección “notificaciones” guardara la llave del pasajero que solicitud el servicio y enviara las notificaciones a los respectivos conductores con el servicio *Push notifications service*.

Figura 24. Esquema B - Ubicar



Fuente autor

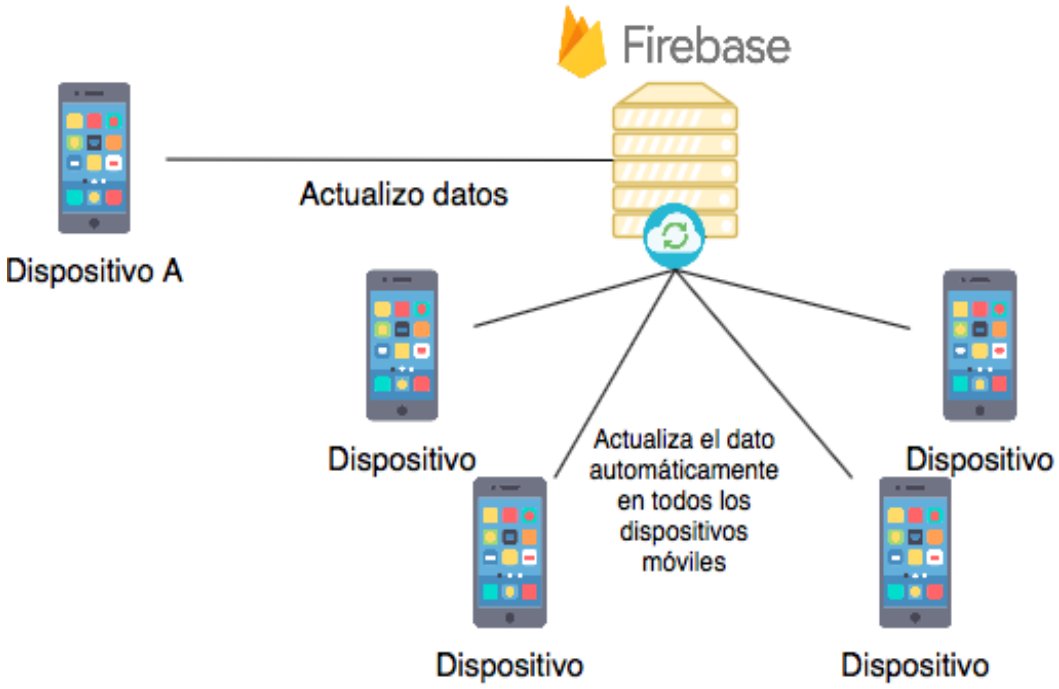
Diagrama Nosql Ubicar – Ver Anexo C

Se anexa el diagrama completo del aplicativo móvil Ubicar.

5.7 ARQUITECTURA UBICAR

En la arquitectura que se presenta en la figura No 25. Podemos evidenciar que un usuario Ubicar con un dispositivo A actualiza datos en el aplicativo, dicha actualización se envía a la base de datos en tiempo real de Firebase el cual provee el backend como servicio y este automáticamente envía la actualización a todos los dispositivos móviles registrados en el aplicativo móvil.

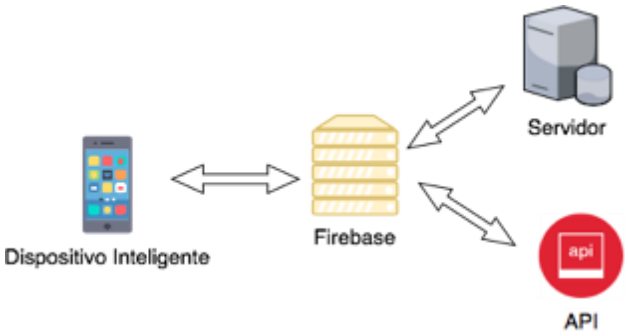
Figura 25. Dispositivo A - Ubicar actualizando datos



Fuente autor

En la figura No 26. Se logra evidenciar que Firebase también provee trabajar en conjunto con un servidor aparte e interactuar con él, se podría utilizar cualquier JSON de la base de datos Firebase en un RestApi, solo se anexa el JSON al final de la URL y se envía una solicitud HTTPS.

Figura 26. Firebase interactuando con un servidor aparte y utilizando Rest Api.



Fuente autor

5.8 BACKEND

Java: El lenguaje de programación Java se basa en el concepto de programación orientada a objetos (OOP) desarrollado por Sun Microsystems, que fue diseñado para tener pocas dependencias de implementación como fuera posible, Java es un lenguaje de programación principalmente para aplicaciones cliente-servidor.

Java contiene características muy importantes, entre ellas se puede describir como un lenguaje simple, orientado a objetos, distribuido, robusto, multihilo, dinámico, portable entre otras características que lo hacen un lenguaje competitivo.

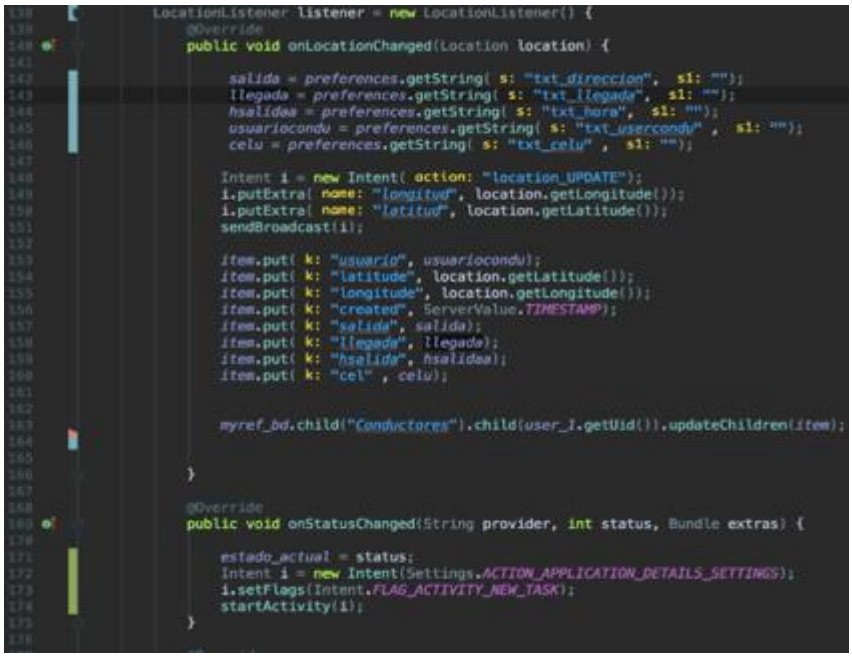
***Service location*:** El sistema operativo Android brinda en las aplicaciones un acceso a los servicios de geolocalización que son compatibles con la versión del Api del dispositivo a través de una clase declarada en el paquete android.location, la cual permite obtener la ubicación actual del dispositivo con coordenadas latitud/longitud, actualizaciones constantes en un determinado tiempo, servicio de localización en segundo plano y demás elementos que provee el servicio de localización.

Dentro del paquete de localización encontramos la clase *LocationManager Y LocationListener* la cual provee acceso al sistema de geolocalización para obtener actualizaciones contantes de ubicación del dispositivo móvil. Al momento de utilizar los servicios de localización en el aplicativo móvil Ubicar se importaron las siguientes clases del paquete *location*.

```
import android.location.Location
import android.location.LocationManager
import android.location.LocationListener
import android.app.Service
```

En la figura No 27. Podemos evidenciar el objeto de tipo LocationListener que contiene el método onLocationChanged el cual estará a la escucha cuando la posición del dispositivo cambie, las coordenadas latitud y longitud se obtienen de location.getLongitude() y location.getLatitude() para ser pasadas en un objeto clave-valor y por ende actualizarlas en Firebase.

Figura 27. Metodo onLocationChanged de LocationListener



```
LocationListener listener = new LocationListener() {
    @Override
    public void onLocationChanged(Location location) {

        salida = preferences.getString( s: "txt_direccion",  s1: "");
        llegada = preferences.getString( s: "txt_llegada",  s1: "");
        hsalidaa = preferences.getString( s: "txt_hora",  s1: "");
        usuariocondu = preferences.getString( s: "txt_usercondu" ,  s1: "");
        celu = preferences.getString( s: "txt_celu" ,  s1: "");

        Intent i = new Intent( action: "location_UPDATE");
        i.putExtra( name: "longitud", location.getLongitude());
        i.putExtra( name: "latitud", location.getLatitude());
        sendBroadcast(i);

        item.put( k: "usuario", usuariocondu);
        item.put( k: "latitude", location.getLatitude());
        item.put( k: "longitude", location.getLongitude());
        item.put( k: "created", ServerValue.TIMESTAMP);
        item.put( k: "salida", salida);
        item.put( k: "llegada", llegada);
        item.put( k: "hsalida", hsalidaa);
        item.put( k: "cel" , celu);


        myref_bd.child("Conductores").child(user_I.getUid()).updateChildren(item);


    }

    @Override
    public void onStatusChanged(String provider, int status, Bundle extras) {

        estado_actual = status;
        Intent i = new Intent(Settings.ACTION_APPLICATION_DETAILS_SETTINGS);
        i.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
        startActivity(i);
    }
```

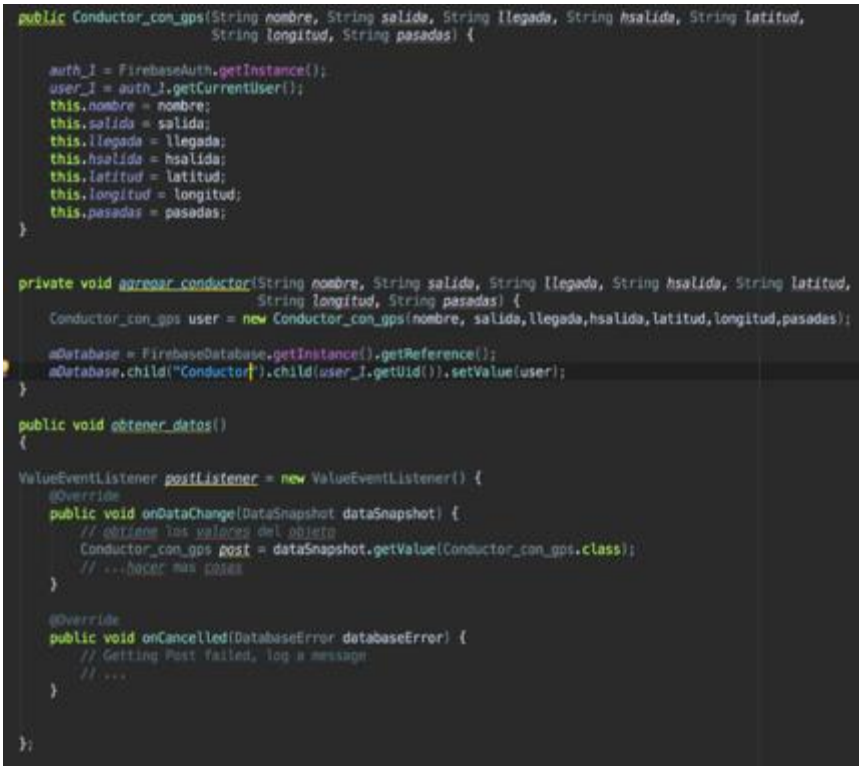
Fuente autor

Escribir y leer datos en Firebase/Java: Los datos de Firebase se escriben en una referencia de *FirebaseDatabase* la cual es una instancia a la base de datos en tiempo real. Para recuperar datos, se debe adjuntar un agente de escucha asíncrona a la referencia. El agente se activa cuando el estado del nodo inicial de datos y otra vez cuando los datos cambian.

En la figura No 28. Se puede evidenciar el objeto Conductor_con_gps el cual agrega al nodo “Conductor” con setValue el cual sobrescribe los datos en la ubicación específica, incluido los nodos secundarios.

En el método obtener_datos se evidencia un ValuEventListener el cual lee y detecta cambios en todo el contenido de la ruta definida, el objeto dataSnapshot obtiene los valores de la referencia.

Figura 28. Objeto conductor_con_gps leer y escribir con Firebase



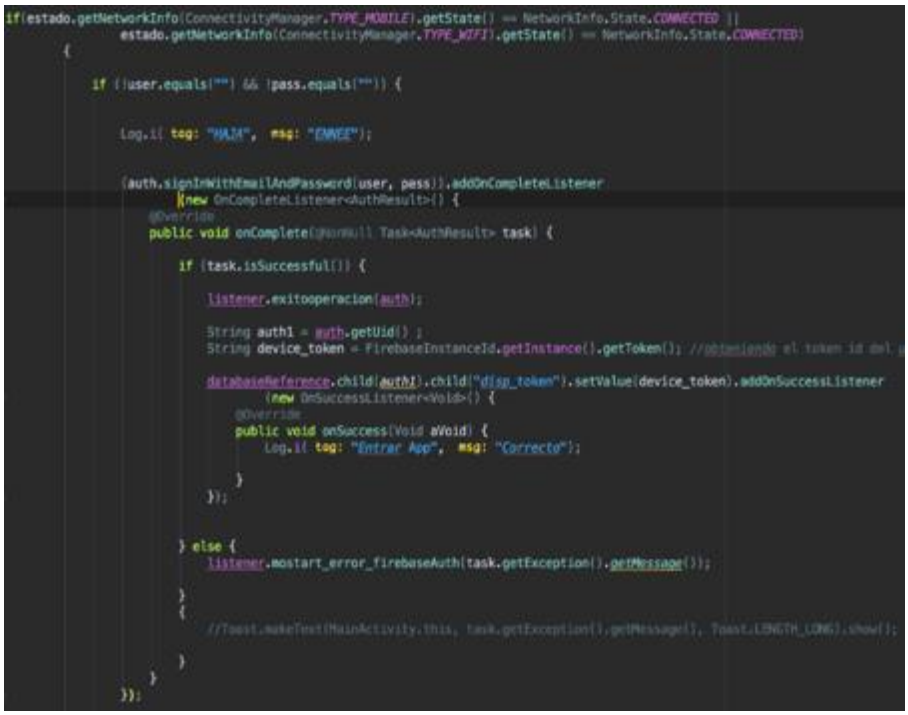
Fuente autor

Firebase Authentication: Firebase proporciona servicios de backend, SDK fáciles de usar y bibliotecas de IU ya elaboradas para identificar los usuarios en una aplicación. La autenticación de Firebase está disponible para contraseñas, números de teléfono, proveedores de identidad como Google, Facebook, Twitter y muchos más utilizando estándares de industria como OAuth 2.0 y OpenID Connect, para ser integrados en la aplicación.

En el aplicativo móvil Ubicar se autentican los usuarios con sus direcciones de correo electrónico y contraseñas. El sdk de FirebaseAuthentication proporciona métodos para crear y administrar los usuarios.

En la figura No 29. Se evidencia el objeto FirebaseAuth que tiene el método signInWithEmailAndPassword() el cual recibe dos parámetros, correo y contraseña para verificar si se encuentra en la base de datos de Firebase y poder ingresar a la plataforma.

Figura 29. Verifica si tiene conexión a internet, si es así, ejecuta el método signInWithEmailandPassword()



Fuente autor

6. METODOLOGÍA

6.1 Especificación de requerimientos funcionales y no funcionales según el estándar IEEE 830

Requerimientos funcionales y no funcionales– Ver Anexo A

Se anexa el documento que contiene los requerimientos funcionales y no funcionales del aplicativo móvil Ubicar, con el fin de comprender todas las tareas

que son requeridas para cumplir con la necesidad del software. El documento de requerimientos contiene el formato de especificación de requisitos de Software (ERS) con la última versión del estándar IEEE 830.

6.2 Metodología Ubicar

La metodología del proyecto se basa en Scrum, la cual se caracteriza por sus sprints que para el desarrollo del proyecto serán máximo de 20 días donde se espera crear un incremento del producto terminado.

El corazón del Scrum es un sprint por lo tanto se realiza una reunión de planificación máximo de 5 horas, donde se planteará con el Scrum máster que se requiere para el incremento del sprint que comienza y como se consigue cumplir con los objetivos del sprint en determinado tiempo para cumplir con el incremento del producto.

Los ítems del producto backlog y sprint backlog fueron declarados como requerimientos funcionales y no funcionales según el estándar IEEE 830 anexado en la entrega, los requerimientos fueron evaluados por el Scrum master, lo cual si se cumplían eran aceptados y puestos en el incremento del producto.

Dentro de la etapa de un sprint se realizan reuniones semanales de 3 a 5 horas y se mantiene contacto constante con el Scrum máster durante el desarrollo del incremento. Esto con el objetivo de mostrar los avances y llevar un control del trabajo realizado cumpliendo con los siguientes detalles:

- Que se realizó la semana pasada que ayudo y contribuyo a cumplir con el objetivo del sprint
- Que tareas se realizaran la semana siguiente para ayudar con el desarrollo y lograr el objetivo del sprint.
- Que inconvenientes se están presentando en el desarrollo del sprint y como se pueden combatir.

Al completar los 20 días establecidos para el sprint, se realiza una revisión general del cuerpo del sprint que toma máximo de 4 a 8 horas, con el fin de adaptarlo a la lista del producto final. En dicha reunión se prueba el funcionamiento del sprint para evaluar el desempeño y los aspectos a mejorar.

El equipo de Scrum está compuesto por:

Dueño del producto	Fabian Humberto Castillo Pineda

Scrum máster	Ing. René Rodríguez Clavijo
Equipo de desarrollo	Fabian Humberto Castillo Pineda

6.3 Tipos de usuario aplicación móvil Ubicar:

Conductor: Es el usuario que puede publicar rutas próximas o rutas que salen en esa misma hora, para prestar el servicio a los estudiantes de la Universidad Pontificia Bolivariana que necesitan movilizarse hacia ella o viceversa, el usuario conductor acepta las solicitudes de los pasajeros que harán parte de su recorrido.

Pasajero: Es el usuario que puede filtrar en el mapa y en las listas de rutas programadas para ver los conductores disponibles en tiempo real circulando por la ciudad de Bucaramanga con sus respectivas características de ruta y poder tomar el servicio.

Administrador Firebase: Encargado de utilizar la consola de Firebase, ver el crecimiento de la aplicación, utilización de Crashlytics, dashboard, performance, Google analytics para obtener análisis detallado de la forma en la que los usuarios interactúan con la aplicación móvil y demás componentes que proporciona la consola de Firebase.

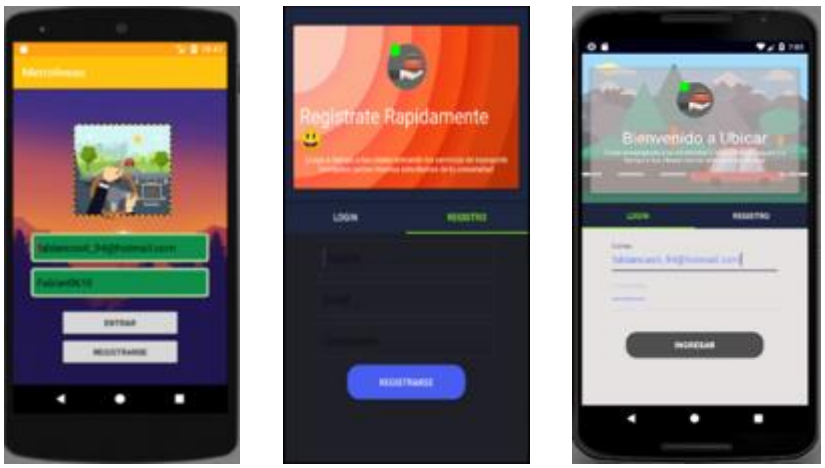
A continuación, se nombran algunas funcionalidades principales del aplicativo móvil Ubicar, todos los sprints y objetivos para cumplir con los requerimientos del software fueron declarados en el documento de especificación de requisitos de Software (ERS) con la última versión del estándar IEEE 830 anexados con la entrega del proyecto.

- Permitir el registro de usuarios de la comunidad UPB con el correo institucional u otro correo valido con una contraseña.
- Permitir el ingreso al aplicativo después de registrado.
- Permitir crear una ruta con características de salida, llegada, hora de salida, lugares por donde pasare, celular, nombre de usuario y guardarlas en la memoria del celular.
- Permitir publicar la ruta guardada como ruta que saldrá pronto o en ese mismo instante con localización, solo se podrá publicar de a una ruta.
- Pasajeros pueden solicitar una ruta que les sirva según su horario.
- Conductor acepta solicitudes.

6.4 Control de versiones de la interfaz de usuario Ubicar:

A medida del desarrollo de los sprints, las encuestas de experiencia de usuario y observaciones suministradas por los estudiantes de la Universidad Pontifica Bolivariana se fue adaptando y personalizando poco a poco el diseño del aplicativo según las expectativas de los estudiantes. A continuación, se muestran algunas versionas de las vistas que conforman el aplicativo, en la figura No 30. Se evidencia las etapas del diseño en ingreso al sistema Ubicar.

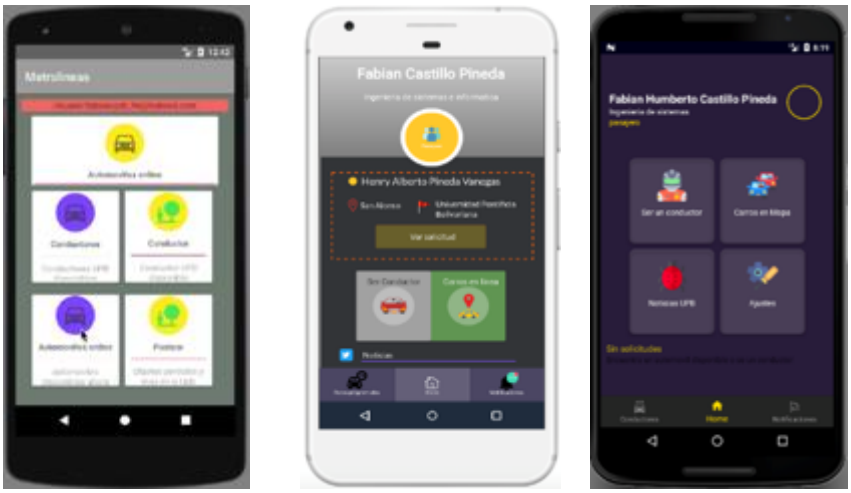
Figura 30. Versión 1,2 y 3 de layout ingresar al sistema



Fuente autor

En la figura No 31. Podemos evidenciar las etapas de diseño en la vista menú la cual, es una de las vistas más importantes y principales del aplicativo móvil, mostrando las diferentes opciones por las cuales se puede navegar en el aplicativo móvil Ubicar.

Figura 31. Versiones de diseño del menú durante el desarrollo del proyecto



Fuente autor

En la figura No 32. Se logra evidenciar el aplicativo móvil, obteniendo ya las credenciales para utilizar Google Maps, en esta etapa se logra utilizar la clase *location* y su configuración para visualizar y actualizar coordenadas de longitud y latitud a la base de datos de Firebase, de esta manera poder ser mostradas en tiempo real a los demás usuarios.

Figura 32. Ubicar mostrando automóviles en tiempo real en Google Maps y sus actualizacion de UI durante el desarrollo

Fuente autor

6.5 Control de versiones del código

Como buenas prácticas de programación y control de software por etapas, se creó un repositorio como lugar de almacenamiento y control de versiones mediante el software Git diseñado por Linus Torvalds, con el fin de pensar en la eficiencia y la confiabilidad del mantenimiento de versiones del aplicativo móvil Ubicar, ya que el aplicativo móvil es extenso y tiene un gran número de archivos de código fuente.

Gracias a los repositorios se logra tener un control total de los cambios efectuados en el software durante el desarrollo, que cambio se hacen en el aplicativo, en que archivo, cuantas líneas de código inserto, cuantas elimino y demás componentes que nos provee los comandos Git. A continuación, se muestra algunos comandos para actualizar los repositorios.

- git add . (Agrega un proyecto al control de versiones Git)
- git commit -m “mensaje” (comando para agregar un mensaje al cambio efectuado)
- git push origin master (actualiza todos los cambios en la rama master del proyecto)

Bitbucket

Es un servicio de alojamiento basado en web, para proyectos que utilizan el sistema de control de versiones Git. Bitbucket ofrece planes comerciales y de forma gratuita, a diferencia de Github, Bitbucket permite crear repositorios privados con equipos de máximo 5 personas para la cuenta gratuita, cuenta con una interfaz de diseño amigable para que el equipo Scrum pueda ver el ciclo de vida del Software y como va avanzando.

En la figura No 33. Se logra evidenciar el software Ubicar alojado en la nube con Bitbucket ya sincronizado, de esta manera se mantiene el código en la nube y se evita perdida de información local o en un caso extremo perdida del proyecto.

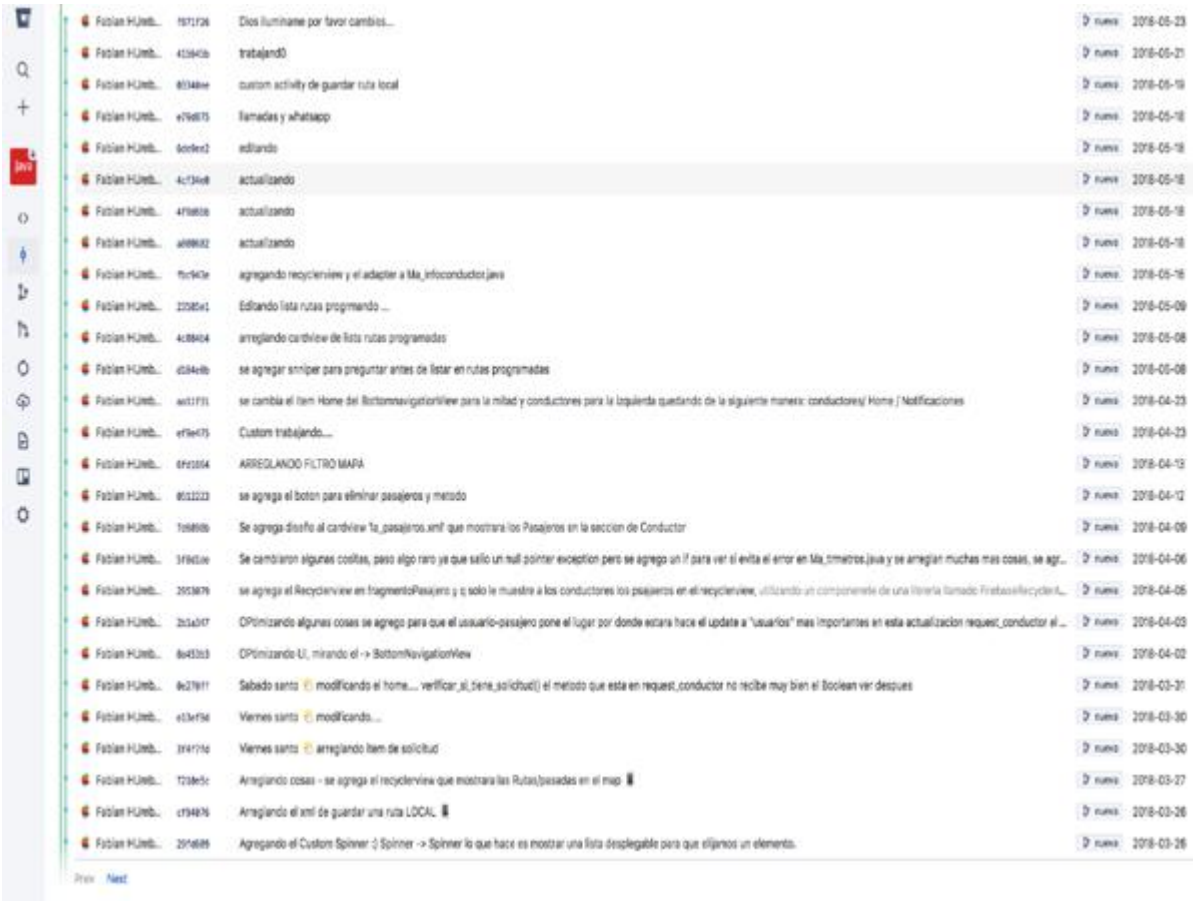
Figura 33.Ubicar alojado en Bitbucket y sincronizado con Git

Fuente autor

En la figura No 34. Se evidencian algunos *commits* hechos durante las etapas del desarrollo por el equipo Scrum, la etapa del desarrollo del software empezó unos

meses atrás y se empezó a actualizar en el repositorio unos meses después, de esta manera se tiene un control total de versiones y tiempo de desarrollo.

Figura 34. Commits proyecto Ubicar



Fuente autor

6.6 Control de errores y pruebas con Firebase Crashlytics y Fabric.io

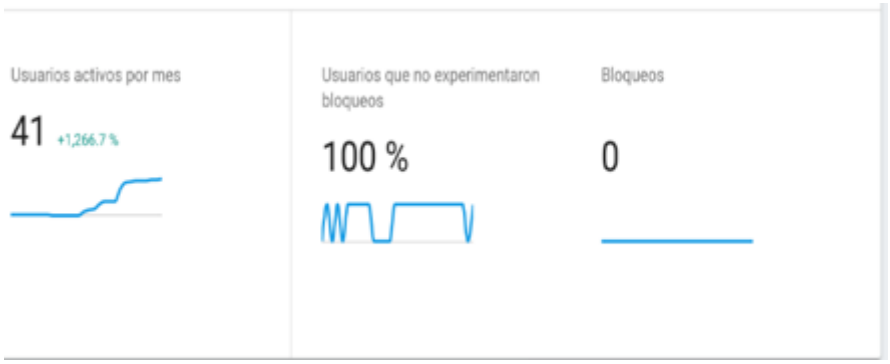
Firebase Crashlytics es una herramienta para informar fallos en tiempo real de un aplicativo móvil con el fin de ayudar y tener un seguimiento de los problemas de estabilidad y optimización que puedan afectar la calidad de la aplicación, priorizando dichos errores y buscando soluciones.

Crashlytics agrupa los fallos de forma inteligente y destaca las circunstancias en las que se produjeron, lo que permite ahorra tiempo en la solución de errores. Descubre si un fallo especifico afecta a muchos usuarios. Crashlytics envía alertas cuando la gravedad de un problema aumenta repentinamente y determina que líneas de código están provocando bloqueos.

Se requiere instalar Crashlytics junto con Fabric.io en el proyecto, por ende se agrega las dependencias en el archivo build.gradle del proyecto.

En la figura No 35. Se evidencia un reporte de Firebae Crashlytics el cual informa 0 bloqueos en los usuarios activos de la aplicación móvil Ubicar, cuando se presenta un error en un ciclo o proceso del Software, Crashlytics lo reporta como bloqueo y informa de donde es el error.

Figura 35. historial bloqueos en la aplicación

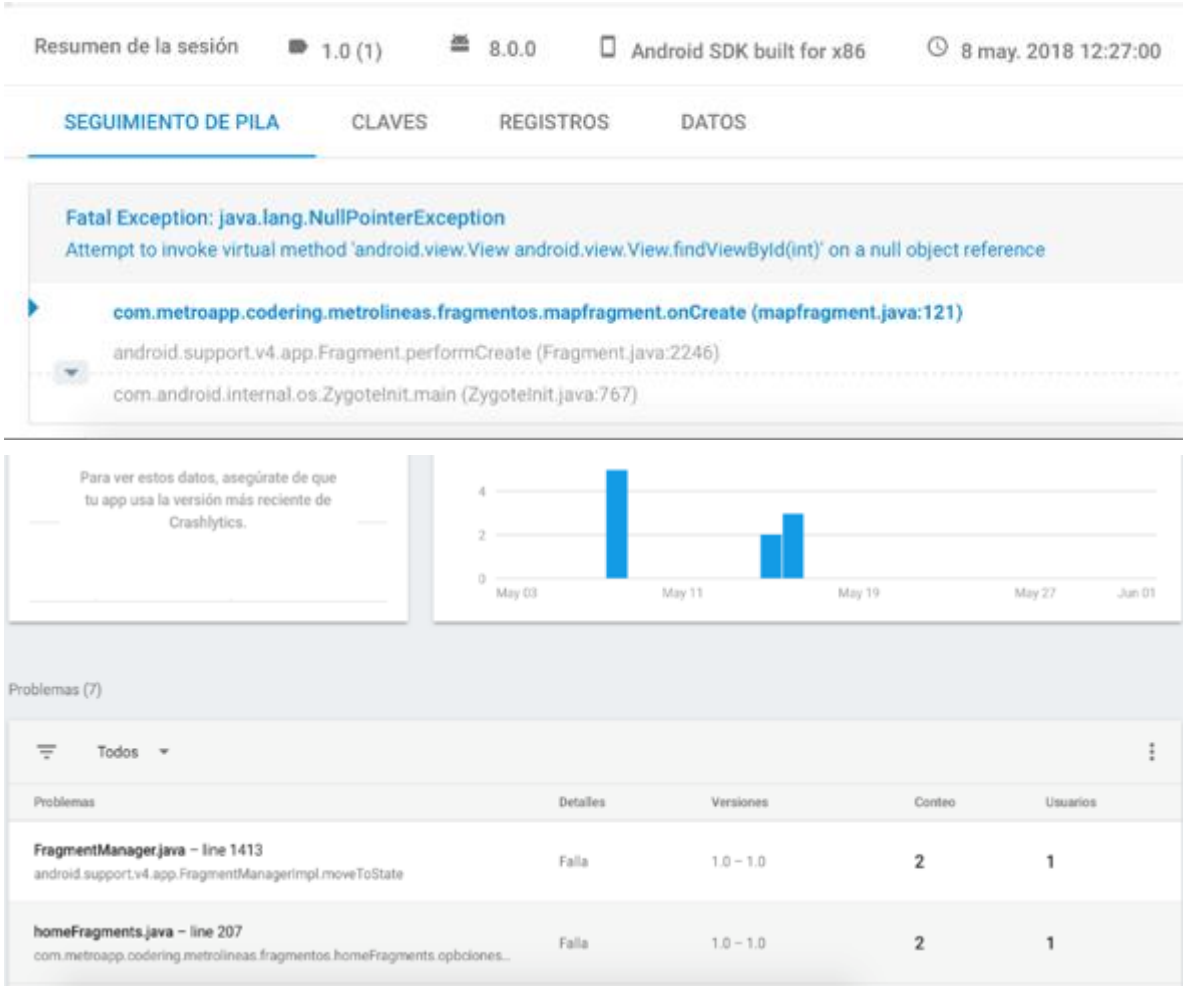


Fuente autor

Al momento de generarse un error en ejecución del software en un dispositivo móvil, Firebase Crashlytics informa dichos errores y de donde son generados, con una información detallada que ayudara al equipo de desarrollo Scrum a comprender mejor el problema y tener un control del aplicativo móvil ante fallos que se presenten

en tiempo de ejecución de la aplicación móvil Ubicar. En la figura No 36. Se evidencia errores generados por algunas clases, en el onCreate de la clase mapfragment se genera un Java.lang.NullPointerException, estos errores fueron solucionados durante el desarrollo del proyecto.

Figura 36. informe de errores por Firebase Crashlytics



Fuente autor

Firebase Crashlytics también captura eventos de UI y permite evidenciar cómo se comportan las vistas al momento de ejecución, cuanto es el tiempo que los usuarios interactúan con ella misma. En la figura No 37. Podemos evidenciar el comportamiento de las vistas declaradas en Ma_nmenu.java, Ma_infoRprograma.java y MainActivity.java.

Figura 37. comportamiento de vistas por la consola de Firebase

Fuente autor

6.7 Pruebas de funcionalidad en la Universidad Pontificia Bolivariana de Bucaramanga

Se hicieron pruebas del aplicativo móvil con una cierta cantidad de estudiantes, estas pruebas fueron de gran utilidad para ver la viabilidad del software, su funcionalidad y que emociones generaba en el estudiante al tener una versión beta del aplicativo móvil e interactuar con ella.

El aplicativo móvil no presento fallos críticos como cierres inesperados, tiempos lentos en la interfaz de usuario y demás fallos que se pueden presentar, los estudiantes dieron sugerencias y mejoras para el aplicativo móvil con el fin de tener una buena experiencia de usuario y poder competir en el mercado y ser usada por la comunidad UPB. A continuación, se presentan algunas sugerencias y comentarios generados por los estudiantes de la Universidad Pontificia Bolivariana seccional Bucaramanga el día 20 de mayo del 2018.

Al momento de realizar las pruebas, se presentaron algunas fallas de ejecución, estos fueron reportados por Firebase Crashlytics y solucionados previamente. Los estudiantes al final de la prueba dieron sus comentarios sobre el uso de la aplicación, algunos comentarios se ven reflejados en la siguiente tabla de respuestas.

Usuario	Comentario
Anónimo	Al momento de ver los automóviles en el mapa moviéndose y ver la información de la ruta al darle clic me parece genial.
Anónimo	Dar efecto de foco al momento que se selecciona un automóvil en el mapa.
Anónimo	Al momento de seleccionar una ruta y cambiar el modo de seleccionar si salgo o no de la universidad para ver la lista de conductores, me parece que se ve mejor como esta al ingresar al mapa.
Anónimo	El guardar las rutas y luego publicar me parece bien el proceso y de aceptar pasajeros también.
Anónimo	Que bien que muestre automóviles en un mapa y que uno pueda escoger el que le sirva.
Anónimo	La aplicación me gustó mucho deberían integrarle en el futuro un chat privado para hablar con el conductor, pero está bien lo de redirigir para llamar al

	conductor o escribirle por WhatsApp, buen trabajo.

7. ACTA DE PROPIEDAD INTELECTUAL

El desarrollo del proyecto de investigación realizado por el estudiante Fabian Humberto Castillo Pineda de la facultad de Ingeniería de Sistemas e Informática presenta la distribución de su propiedad intelectual de la siguiente forma:

- Los derechos morales corresponden al autor del proyecto: Fabián Humberto Castillo pineda.
- Los derechos patrimoniales serán conservados por el autor.

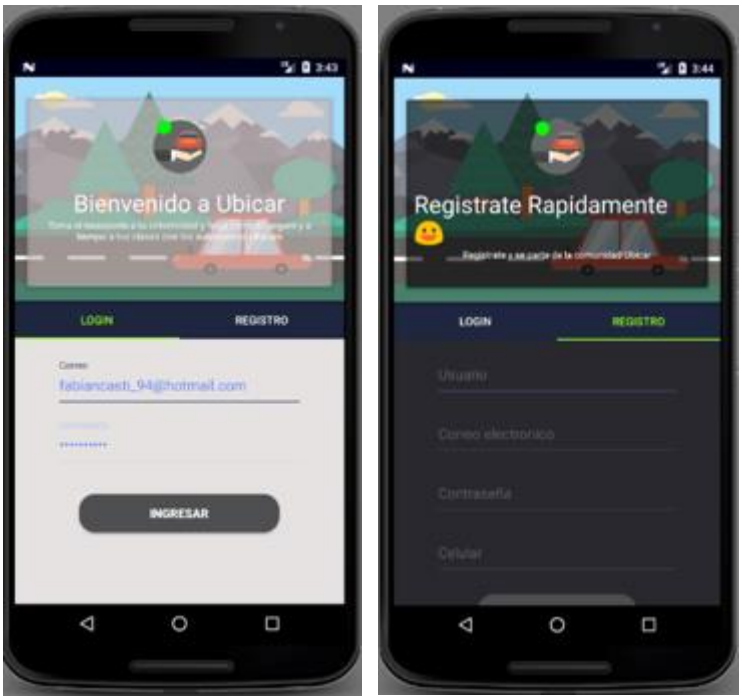
8. RESULTADOS

A continuación, se presentan los resultados del proyecto, por medio de figuras que ilustran las principales funcionalidades del aplicativo móvil Ubicar.

8.1 Acceso al sistema

En la figura No 38. Se evidencia si el usuario hace parte de Ubicar para poder ingresar al sistema de lo contrario deberá registrarse para poder ingresar.

Figura 38. login y registro Ubicar



Fuente autor

8.2 Menú del sistema

Después de acceder al sistema, el usuario se encontrará con el menú del sistema, como se muestra en la figura No 39. El usuario puede ser con un conductor, ver carros circulando en tiempo real, ver objetos perdidos UPB, perfil, ver lista de rutas que salen en una hora próxima, estado de notificaciones y ver notificaciones.

Figura 39. Menú Ubicar

Fuente autor

8.3 Ser un conductor

Si el usuario quiere prestar un servicio de transporte, ingresa a ser un conductor como se puede ver en la figura No 40. El usuario tiene una ruta guardada localmente sin ser publicada aun, puede agregar rutas, ver rutas que están publicadas y ver sus pasajeros.

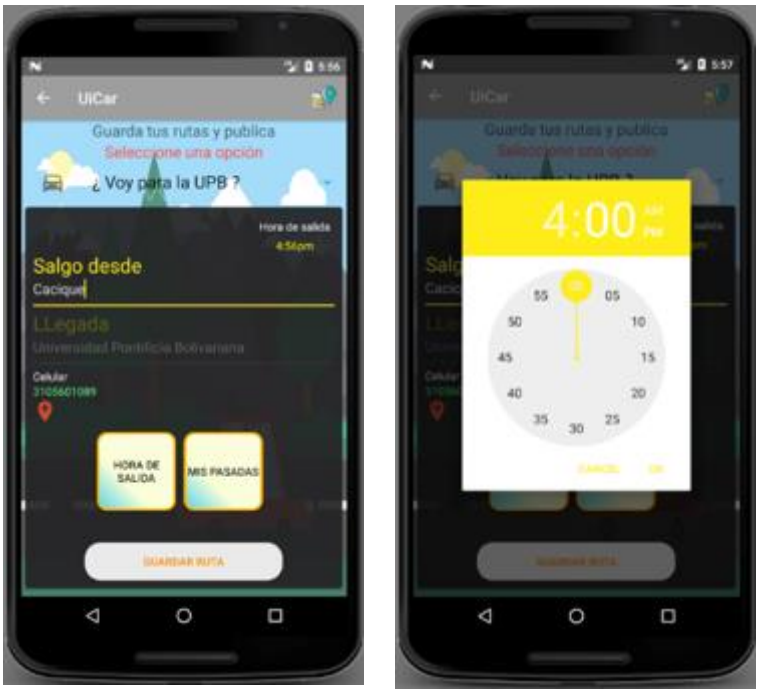
Figura 40.Soy conductor
Ubicar

Fuente autor

8.4 Agregar una ruta

El usuario selecciona primero si va para la Universidad Pontifica Bolivariana o si sale de ella, luego como se puede apreciar en la figura No 41. El usuario agrega la salida o llegada, horada de salida y los lugares por donde pasara, el celular y usuario son obtenidos de la base de datos.

Figura 41. Agregar ruta

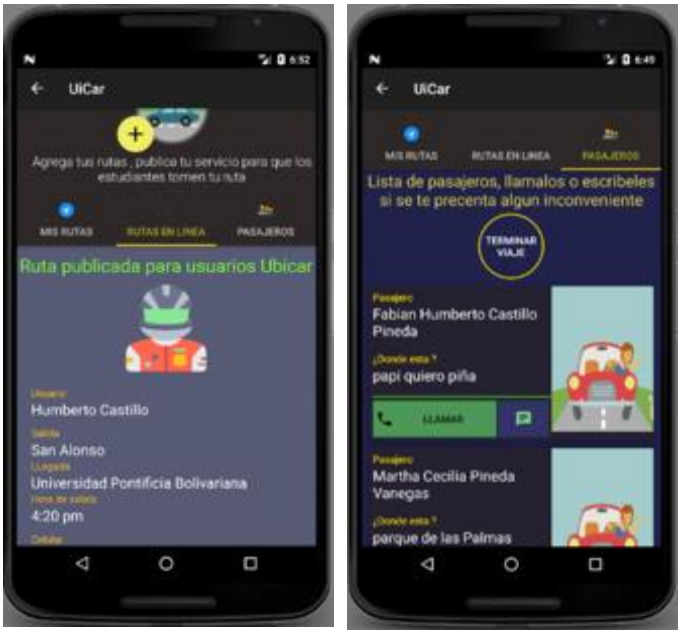


Fuente autor

8.5 Rutas en línea / Mis pasajeros

En la misma sección de soy conductor, al momento de un usuario publicar una ruta guardada localmente para los usuarios de Ubicar, en el tabmenu en rutas en línea se evidencia que el usuario puede ver si tiene una ruta publicada, todos los demás usuarios que le sirvan su ruta, podrán tomar el servicio, en este caso como se puede ver en la figura No 42. El usuario Humberto ha aceptado 2 pasajeros, él puede llamarlos directamente o enviarles un mensaje vía WhatsApp si sucede un problema, el usuario puede terminar el viaje cuando llegue a su destino o si se le presenta algo.

Figura 42. Ruta en línea / Pasajeros

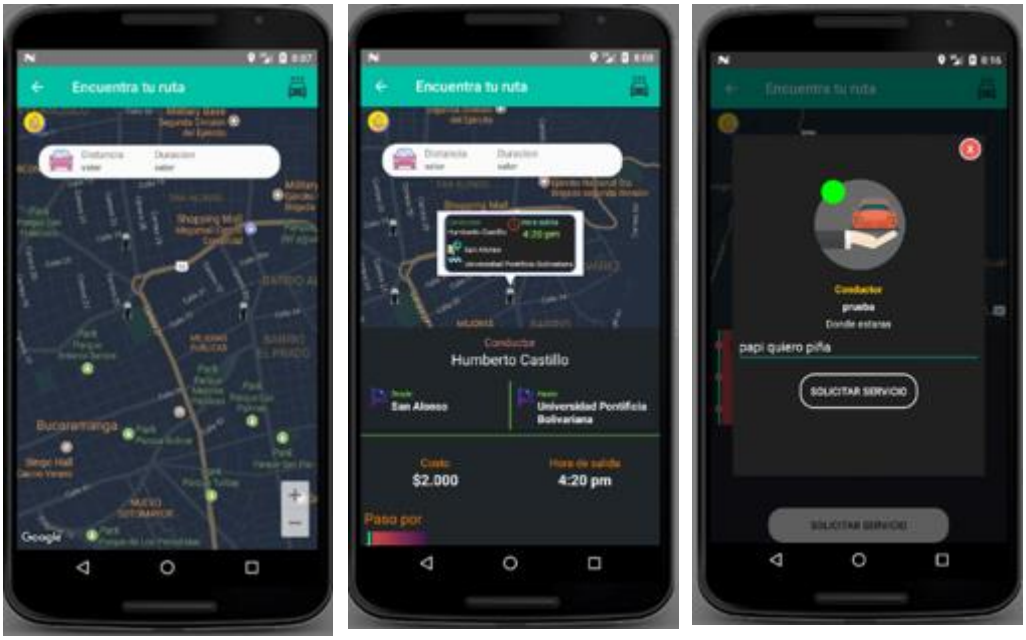


Fuente autor

8.6 Automóviles en mapa

Los usuarios pueden visualizar los automóviles en línea que están circulando en un mapa, ver sus características de rutas, si dicha ruta se adapta a sus necesidades de viaje solicita el servicio para ir a la Universidad Pontificia Bolivariana o viceversa, como se evidencia en la figura No 43.

Figura 43. Google Maps con automóviles en línea y solicitar una ruta

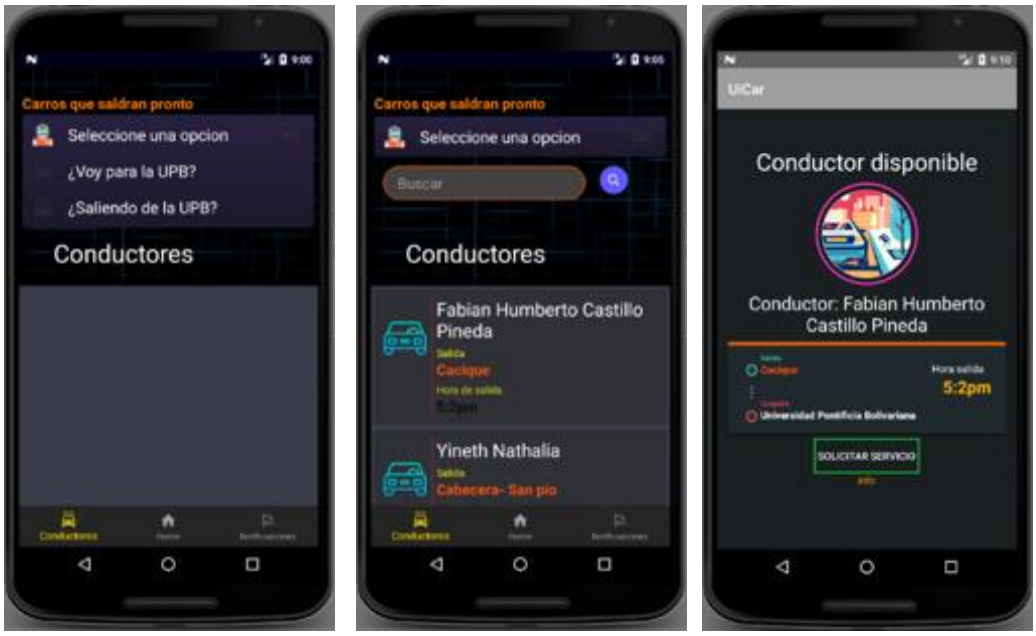


Fuente autor

8.7 Rutas que salen en una hora próxima (rutas programadas)

Los usuarios pueden listar los conductores que saldrán en una hora próxima y solicitar su servicio, estos automóviles no están con GPS. Como se muestra en la figura No 44. El usuario seleccionó “voy para la UPB” y muestra todos los automóviles que irán a la UPB en una hora próxima.

Figura 44. Automóviles que saldrán en una hora próxima

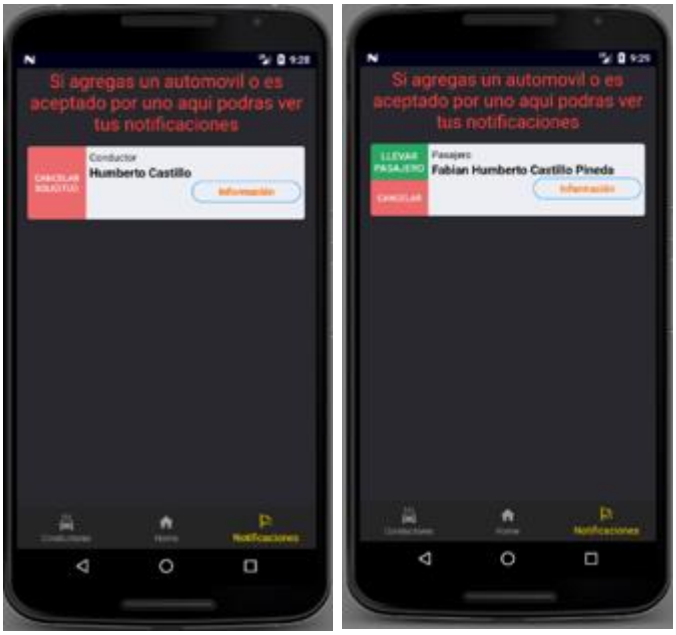


Fuente autor

8.8 Solicitudes

En la figura No 45. Se evidencia el *layout* que se encarga de las solicitudes en la pantalla de la izquierda un usuario envía una solicitud a un conductor para estar en su ruta, en la pantalla de la derecha el usuario como conductor.

Figura 45. Solicitudes Ubicar



Fuente autor

8.9 Mi conductor

Cuando un conductor acepta una solicitud y el pasajero ya hace parte de su ruta, el usuario como pasajero podrá ver la información de su conductor que lo llevará, podrá enviarle un mensaje directamente vía WhatsApp o llamarlo directamente. Mi conductor se logra evidenciar en la figura No 46.

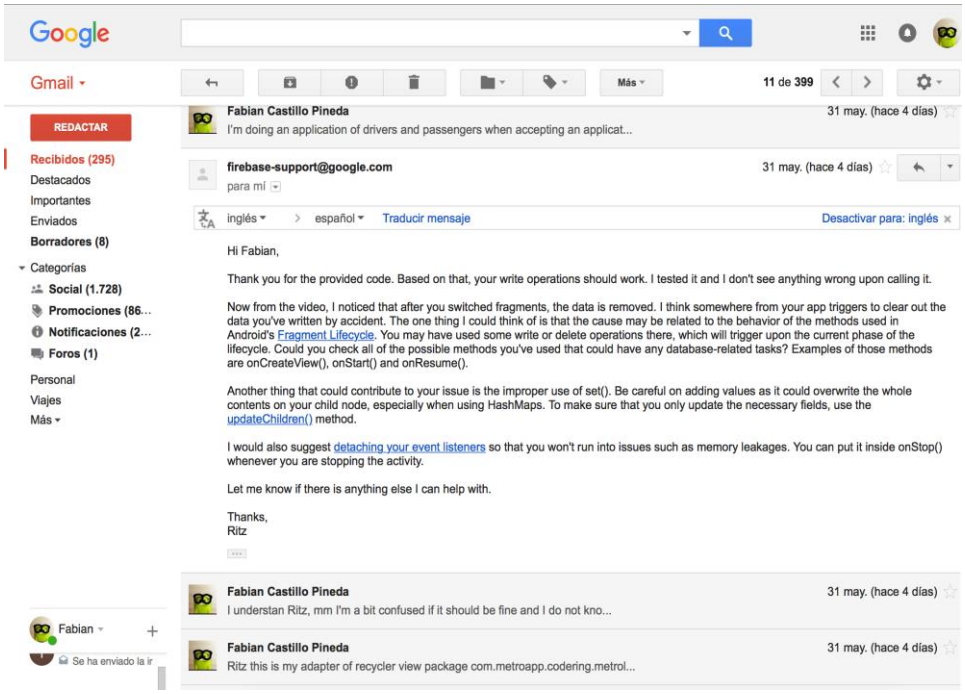
Figura 46. mi conductor
Ubicar

Fuente autor

Participación con la comunidad de Google y Firebase

Se publicaban foros sobre el comportamiento del aplicativo móvil o cuando surgía algún fallo que necesitaba alguna explicación para el equipo Scrum, se tubo contacto con la comunidad de soporte de Firebase-Google sobre el aplicativo móvil vía Gmail. Se utilizo el Software de herramienta de comunicación Slack con la comunidad de Firebase para estar actualizado constantemente y tener contacto con personas que utilizan el SDK de Google. En la figura No 47. Se evidencian correos electrónicos por parte de firebase-support@google.com.

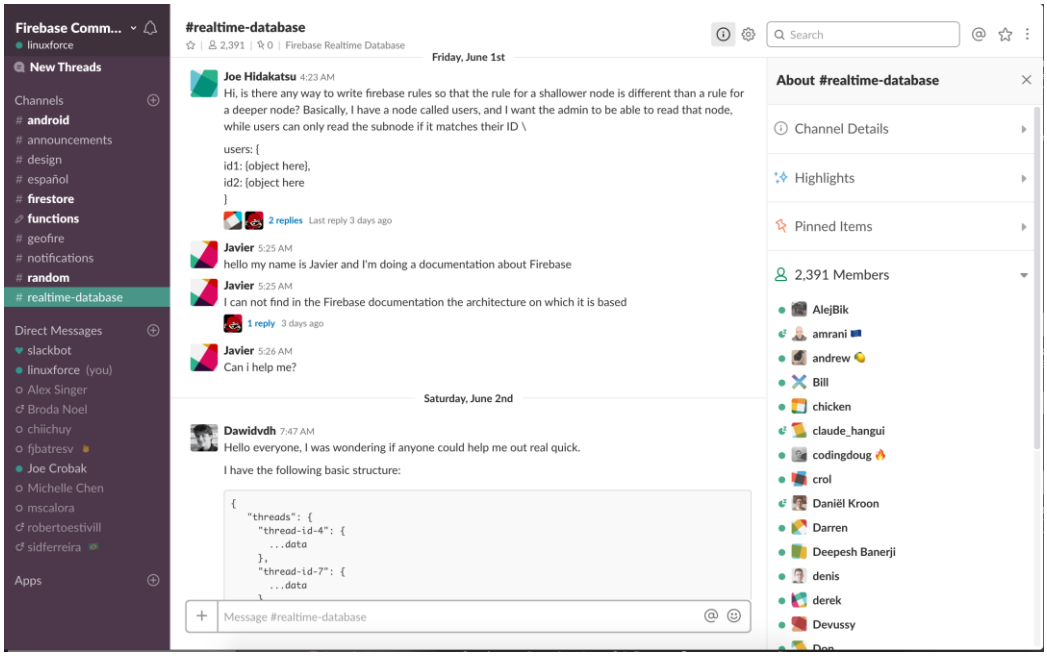
Figura 47. Contacto con Google-support



Fuente autor

En la Figura No 48. Se evidencia la comunidad de desarrolladores de Firebase privado en el software Slack, para acceder a la comunidad basta con enviar un correo a la comunidad Firebase para ser aceptado y poder ser parte de ella.

Figura 48. Comunidad Firebase en Slack



Fuente autor

9. CONCLUSIONES

- El uso de elementos de estudio visual de material design y las actividades elaboradas de experiencia de usuario en la Universidad Pontificia Bolivariana fueron muy útiles y necesarias para la creación del aplicativo móvil Ubicar, ya que aportaron un enfoque sobre como la experiencia de usuario y diseño puede afectar a las aplicaciones o generan emociones en las personas al momento de interactuar con ellas.

- La metodología Scrum permitió estructurar y organizar el proyecto con un objetivo claro del aplicativo móvil a desarrollar, tiempo desarrollo constante y reuniones periódicas con el Scrum master sobre la planeación y desarrollo de cada incremento para completar con los objetivos planteados.

- La indagación de bases de datos Nosql y utilización de ellas mismas en el desarrollo del proyecto dieron una visión sobre otras alternativas a las bases de datos relacionales y que ventajas o desventajas se pueden generar al momento de modelar una base de datos de este tipo.

- La indagación y uso del sdk de Firebase por Google ayudo a facilitar la gestión del aplicativo móvil, la interacción de los usuarios en ella, manteniendo los datos en la nube a partir de un servicio web que ofrece la posibilidad de programar aplicaciones con datos que son sincronizados en tiempo real a través de múltiples dispositivos, evitando muchas de las tareas engorrosas que se presentan al desarrollar un aplicativo con grandes dimensiones, sin embargo un cambio en el Sdk de Firebase o actualización por Google deberán ser actualizad en el aplicativo móvil también, para no generar conflictos.

- Uno de los factores que genero un desarrollo ágil fueron los frameworks y librerías como FirebaseUI utilizadas en el proyecto, debido a que tienen estructuras y modelos de código que permiten a un desarrollador adaptarlos según los requisitos de su aplicación.

- El aplicativo móvil cumplió con las técnicas de experiencia de usuario aplicadas en diferentes partes del desarrollo en la Universidad Pontificia Bolivariana, ayudando a mostrar la información de manera más organizada y en línea sobre los servicios de transporte que se prestan entre los mismos estudiantes para movilizarse desde sus casas hacia la Universidad Pontificia Bolivariana o viceversa.

- Firebase provee una versión gratuita para los desarrolladores que permite un máximo de 100 usuarios lo que es aceptable durante el desarrollo del producto, sin embargo, se requiere una inversión para los servidores por un agente tercero para el plan Flame comercial de Firebase y el equipo scrum estará haciendo actualizaciones constantes durante este año.

- Utilizar el software Marvel App para la creación del prototipo funcional de la aplicación móvil Ubicar y aprender a utilizar dicha herramienta fue muy importante durante la elaboración del proyecto, ya que dio a conocer cómo se elaboran prototipos funcionales con Marvel App para aplicaciones y en un futuro si se desarrollan aplicaciones móviles poder mostrar un prototipo funcional al cliente antes de empezar.

- La participación constante entre los desarrolladores y comunidad de Google Firebase fueron de gran importancia, para ampliar el conocimiento sobre el sdk de Google y estar en contacto con desarrolladores que saben sobre las tecnologías mencionadas.

10. REFERENCIA BIBLIOGRÁFICAS

[1] GALVIS RAMIREZ Y CIA. S.A, «Bucaramanga,» 5 Junio 2016. [En línea]. Available:https://www.bucaramanga.com/articulo/estas-seis-aplicaciones-te-ayudaran-a-movilizarte-por-bucaramanga.html. [Último acceso: 24 Septiembre 2017].

[2] U. Company, «Uber,» Uber, 2 Enero 2017. [En línea]. Available: https://www.uber.com/es-CL/blog/que-es-uber/ .[Último acceso: 1 Octubre 2017].

[3] K. S. y. J. Sutherland, «La Guía de Scrum,» Scrum tm, California, 2017.

[4] Y. H. Montero, «Elementos de la IPO: diseño, personas y tecnologiasa,» 3 Junio 2015. [En línea]. Available: https://www.exabyteinformatica.com/uoc/Informatica/Interaccion_persona_ordenador/Interaccion_persona_ordenador_(Modulo_2).pdf. [Último acceso: 10 Febrero 2018].

[5] IBM, «ibm,» [En línea]. Available: https://www.ibm.com/support/knowledgecenter/es/SSEPGG_8.2.0/com.ibm.db2.ii.doc/opt/c0007799.htm. [Último acceso: 1 Febrero 2018].

[6] C. Amate, «blogthinkbig,»blogthinkbig, 13 Septiembre 2014. [En línea]. Available: https://blogthinkbig.com/sistemas-operativos-moviles. [Último acceso: 12 Septiembre 2017].

[7] U. P. d. Valencia, «androidcurso,» androidcurso, 23 Enero 2016. [En línea]. Available: http://www.androidcurso.com/index.php/99. [Último acceso: 4 Octubre 2017].

[8] Oracle, «java,» Oracle, [En línea]. Available: https://www.java.com/es/download/faq/whatis_java.xml. [Último acceso: 11 Octubre 2017].

[9] Google, «firebase.google,» Google, 3 Febrero 2015. [En línea]. Available: https://firebase.google.com/?hl=es-419. [Último acceso: 23 Sptiembre 2017].

[10] M. Rouse, «searchdatacenter,»techtarge, 10 June 2015. [En línea]. Available: http://searchdatacenter.techtarget.com/es/definicion/NoSQL-No-Solo-SQL. [Último acceso: 18 Septiembre 2017].

[11] Guest, «Oracle,» 19 Abril 2016. [En línea]. Available: https://blogs.oracle.com/spain/qu-es-una-base-de-datos-nosql. [Último acceso: 14 Diciembre 2017].

11. ANEXOS

A continuación se presentan los anexos que pertenecen al proyecto.

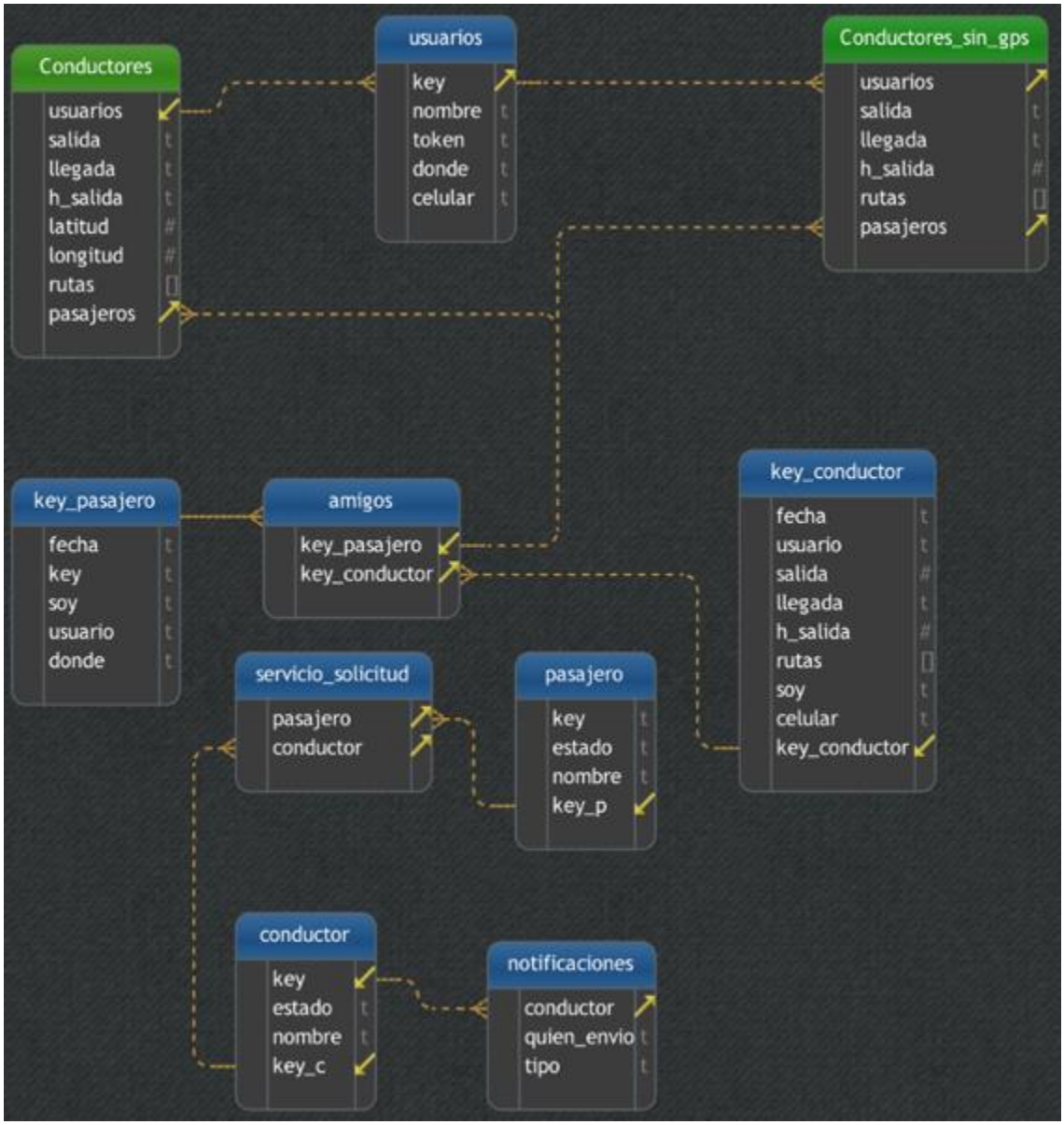
11.1 ANEXO A – Requerimientos funcionales y no funcionales IEEE 830.

Se adjunta el documento de requerimientos funcionales y no funcionales (.doc) en la carpeta de la entrega.

11.2 ANEXO B – PROTOTIPO UBICAR EN MARVEL APP.

PROTOTIPO MARVAL APP - LINK (Ctrl + click)

11.3 ANEXO C – DIAGRAMA NOSQL UBICAR.


Conductores
usuarios
salida
llegada
h_salida
latitud
longitud
rutas
pasajeros
usuarios
key
nombre
token
donde
celular
Conductores_sin_gps
usuarios
salida
llegada
h_salida
rutas
pasajeros
key_pasajero
fecha
key
soy
usuario
donde
amigos
key_pasajero
key_conductor
key_conductor
fecha
usuario
salida
llegada
h_salida
rutas
soy
celular
key_conductor
servicio_solicitud
pasajero
conductor
pasajero
key
estado
nombre
key_p
conductor
key
estado
nombre
key_c
notificaciones
conductor
quien_envio
tipo