Implementación de un portal web de intercambio de objetos dentro de la comunidad universitaria UIS para lograr una disminución de la informalidad dentro del campus

Jorge Andrés Triana Mojica,

Jhon Andrés Parra Rodríguez

Trabajo de grado para optar al título de

Ingeniero de Sistemas

Director

Carlos Adolfo Beltrán Castro

Ingeniero de Sistemas

Universidad Industrial de Santander

Facultad de ingenierías

Fisicomecánicas

Escuela de ingeniería de sistemas e informática

Bucaramanga

2022

Agradecimientos

Deseo dedicar el esfuerzo impreso en este proyecto de grado principalmente a mis padres, Jorge Triana y Ana María Mojica, quienes, sin su esfuerzo, comprensión y cercanía no hubiera sido posible la realización de esta meta.

A mi abuela, Amanda Osorio, quien me acompaño durante todos estos años de estudio y me acogió como un hijo más.

Al profesor Calos Adolfo Beltrán quien nos tendió una mano y creyó en nosotros en el nacimiento de esta iniciativa.

A mis compañeros de ingeniería de sistemas, en especial a Elkin Darío Fernández, quien fue pieza fundamental en este proyecto y sin su ayuda no hubiera sido posible la realización del mismo.

A mis amigos cercanos, que fueron mi soporte moral durante los momentos de dificultad en este proceso académico.

Agradecimientos

Primeramente, a Dios por permitirme pasar por esta etapa en la vida y poder cumplir a cabalidad con ella.

A mis padres Cruz Angelica Rodriguez y Freddy Javier Contreras, a mis hermanos Angel David Lopez y Mathias Alejandro Contreras por su comprensión y apoyo en todo el proceso de formación académica profesional y personal.

A mis abuelos Agustin Rodriguez y Cruz Delina Rangel por sus enseñanzas e instruirme disciplina y respeto.

Al profesor Carlos Adolfo Beltrán por su confianza e instrucción en la realización de este proyecto de grado.

A mis compañeros de Ingeniería de Sistemas y amigos por el apoyo brindado en el proceso académico.

Tabla de contenido

Pág.

Introducción **17**

1. Justificación y definición de la situación actual: **19**

2. Objetivos **20**

2.1 Objetivo general **20**

2.2 Objetivos específicos **20**

2. Marco teórico **22**

2.1 Marco conceptual **22**

2.1.1 Anuncio Clasificado 22

2.1.2 E-commerce 22

2.1.3 E-marketplace 23

2.1.5 Aplicativo Móvil 25

2.1.6 Modelo Vista Controlador 29

2.1.7 Desarrollo en cascada 30

2.1.8 Whatsapp 32

2.2 Marco tecnológico **33**

2.2.1 Flutter 33

2.2.2 NodeJS 34

2.2.3 GitHub 36

2.2.4 JSON 36

2.2.5 MongoDB 36

2.3 Estado del arte 38

2.3.1 Mercado libre 38

2.3.2 eBay 39

2.3.3 Facebook Marketplace 40

2.3.4 Wallapop 42

3. Metodología 44

3.1. Determinación de requerimientos. 44

3.2. Diseño del Software. 44

3.3. Desarrollo del software. 44

3.4. Pruebas 45

3.5. Implementación y verificación del programa. 45

5.6. Documentación. 45

5.7. Ajustes finales. 46

4. Desarrollo del proyecto 47

4.1 Determinación de requerimientos 47

4.1.1 Requerimientos funcionales 47

4.1.2 Requerimientos no funcionales 59

4.2 Diseño del software 62

4.2.1 Casos de uso 62

4.2.2 Tarjetas CRC 64

4.2.3 Modelo base de datos 67

4.2.4 Diagrama de clases 70

4.2.5 Diagrama de actividades 71

4.3 Diseño de interfaz grafica 74

4.3.1 Generalidades 74

4.3.2 Vistas diseño uno 78

4.3.3 Vistas diseño dos 93

4.4 Desarrollo 111

4.4.1 Arquitectura 111

4.4.2 Backend 113

4.4.3 Frontend 122

4.4.4 Despliegue del prototipo 136

4.4.5 Trazabilidad 140

5. Pruebas 142

5.1 Pruebas de integridad 142

5.2 Verificación de requerimientos 149

5.2.1 Verificación de requerimientos funcionales 149

5.2.2 Verificación requerimientos no funcionales 152

5. Observaciones y restricciones 154

6. Encuesta 154

6.1 Preguntas 154

6.2 Resultados 155

6.2.1 Resultados pregunta 1 156

6.2.2 Resultados pregunta 2 156

6.2.3 Resultados pregunta 3 157

6.2.4 Resultados pregunta 4 157

6.2.5 Resultados pregunta 5 158

6.2.6 Resultados pregunta 6 158

6.2.7 Resultados pregunta 7 159

6.2.8 Resultados pregunta 8 159

6.2.9 Resultados pregunta 9 160

6.2.10 Resultados pregunta 10 160

7. Recomendaciones 161

8. Conclusiones 163

Referencias bibliográficas 165

Lista de tablas

Pag.

Tabla 1. Requerimiento funcional: Registrar usuario 47

Tabla 2. Requerimiento funcional: Iniciar sesión 48

Tabla 3. Requerimiento funcional: Recuperar contraseña 48

Tabla 4. Requerimiento funcional: Cambiar contraseña 49

Tabla 5. Requerimiento funcional: Modificar datos personales 49

Tabla 6. Requerimiento funcional: Cerrar sesión 49

Tabla 7. Requerimiento funcional: Crear anuncios 50

Tabla 8. Requerimiento funcional: Visualizar anuncios 50

Tabla 9. Requerimiento funcional: Valorar anuncio 51

Tabla 10. Requerimiento funcional: Filtrar anuncios por categoría 51

Tabla 11. Requerimiento funcional: Filtrar anuncios por relevancia 52

Tabla 12. Requerimiento funcional: Filtrar anuncios por antigüedad 52

Tabla 13. Requerimiento funcional: Editar información del anuncio 53

Tabla 14. Requerimiento funcional: Eliminar anuncio 53

Tabla 15. Requerimiento funcional: Previsualizaciones de anuncios 54

Tabla 16. Requerimiento funcional: Visualización previsualizaciones 54

Tabla 17. Requerimiento funcional: Contactar con el anunciante 55

Tabla 18. Requerimiento funcional: Perfil personal 55

Tabla 19. Requerimiento funcional: Editar descripción del perfil de usuario 56

Tabla 20. Requerimiento funcional: Cambiar imagen de perfil 56

Tabla 21. Requerimiento funcional: Visualizar perfil 56

Tabla 22. Requerimiento funcional: Retornar a la vista principal 57

Tabla 23. Requerimiento funcional: Retornar a la pantalla anterior 57

Tabla 24. Requerimiento funcional: Búsqueda de anuncios 58

Tabla 25. Requerimiento funcional: Búsqueda de anuncios por perfil 58

Tabla 26. Requerimiento funcional: Visualizar datos de la app 59

Tabla 27. Requerimiento no funcional: Limitar el alcance la app 59

Tabla 28. Requerimiento no funcional: Interface amigable 60

Tabla 29. Requerimiento no funcional: Disponibilidad del aplicativo 60

Tabla 30. Requerimiento no funcional: Integridad de datos 61

Tabla 31. Requerimiento no funcional: Compatibilidad con Android 61

Tabla 32. Requerimiento no funcional: Capacidad del aplicativo 62

Tabla 33. Tarjeta CRC Usuario 65

Tabla 34. Tarjeta CRC Anuncio 65

Tabla 35. Tarjeta CRC Categoría 66

Tabla 36. Tarjeta CRC Perfil 66

Tabla 37. Tarjeta CRC Adjunto 66

Tabla 38. Tarjeta CRC Voto 67

Tabla 39. Colores aplicativo móvil 75

Tabla 40. Categorías de anuncios aplicativo móvil 76

Tabla 41. Rutas de anuncio 120

Tabla 42. Rutas de autenticación 120

Tabla 43. Rutas de perfil 121

Tabla 44. Rutas de categorías 121

Tabla 45. Rutas de ciudad 121

Tabla 46. Prueba registrar usuario 142

Tabla 47. Prueba iniciar sesión 142

Tabla 48. Prueba inicio de sesión incorrecto y reiterado 143

Tabla 49. Prueba recuperar contraseña 143

Tabla 50. Prueba cerrar sesión 143

Tabla 51. Prueba crear anuncios 144

Tabla 52. Prueba valorar anuncio 144

Tabla 53. Prueba eliminar anuncio 145

Tabla 54. Prueba confirmar eliminar anuncio 145

Tabla 55. Prueba contactar con anunciante 145

Tabla 56. Prueba búsqueda de anuncios 146

Tabla 57. Prueba navegación por categorías 146

Tabla 58. Prueba búsqueda filtrada por relevancia 147

Tabla 59. Prueba búsqueda filtrada por antigüedad de publicación 147

Tabla 60. Prueba cambio de contraseña 148

Tabla 61. Prueba edición descripción perfil personal 148

Tabla 62. Prueba cambiar imagen de perfil 149

Tabla 63. Prueba visualizar datos de la app 149

Lista de ilustraciones

Pag.

Ilustración 1. iOS vs. Android 26

Ilustración 2. Ejemplo de implementación del Modelo Vista Controlador 30

Ilustración 3. Modelo cascada original de Winston W. Royce 31

Ilustración 4. Comparación stateless widget vs stateful widget 34

Ilustración 5. Ejemplo de la estructura de un documento en MongoDB 37

Ilustración 6. Casos de uso del aplicativo, autenticación de usuarios 63

Ilustración 7. Casos de uso del aplicativo, gestión del perfil 63

Ilustración 8. Casos de uso del aplicativo, gestión de anuncios 64

Ilustración 9. Modelo base de datos 68

Ilustración 10. Diagrama de clases 70

Ilustración 11. Diagrama de actividades, Crear anuncio 71

Ilustración 12. Diagrama de actividades, mostar anuncio por categoría 72

Ilustración 13. Diagrama de actividades, calificar un anuncio 73

Ilustración 14. Logo Aplicativo móvil a color y blanco y negro 74

Ilustración 15. Nombre aplicativo móvil a color y blanco y negro 75

Ilustración 16. Vista pantalla de inicio 78

Ilustración 17. Vista crear cuenta 79

Ilustración 18. Vista inicio de sesión 80

Ilustración 19. Vista categoría 1 81

Ilustración 20. Vista categoría 2 83

Ilustración 21. Vista búsqueda 84

Ilustración 22. Vista filtros de búsqueda 85

Ilustración 23. Vista menú lateral 86

Ilustración 24. Vista perfil de usuario 87

Ilustración 25. Vista edición datos personales 88

Ilustración 26. Vista publicación de anuncio 89

Ilustración 27. Vista anuncio 90

Ilustración 28. Vista perfil anunciante 91

Ilustración 29. Vista anuncio propio 92

Ilustración 30. Vista pantalla de inicio V2 93

Ilustración 31. Vista crear cuenta V2 94

Ilustración 32. Vista inicio de sesión V2 95

Ilustración 33. Vista recuperar contraseña 96

Ilustración 34. Vista código de confirmación 97

Ilustración 35. Vista cambia tu contraseña 98

Ilustración 36. Vista categoría 1 V2 99

Ilustración 37. Vista categoría 2 V2 100

Ilustración 38. Vista búsqueda V2 101

Ilustración 39. Vista filtros de búsqueda V2 102

Ilustración 40. Vista menú lateral V2 103

Ilustración 41. Vista Acerca de la app 104

Ilustración 42. Vista perfil de usuario V2 105

Ilustración 43. Vista edición datos personales V2 106

Ilustración 44. Edición perfil cambio de contraseña 107

Ilustración 45. Vista publicación de anuncio V2 108

Ilustración 46. Vista anuncio V2 109

Ilustración 47. Vista perfil anunciante V2 110

Ilustración 48. Arquitectura del aplicativo 112

Ilustración 49. Codificación del modelo de datos backend 113

Ilustración 50. Codificación de una ruta 114

Ilustración 51. Codificación middlewares 116

Ilustración 52. Codificación de un controlador 117

Ilustración 53. Codificación de un helper 119

Ilustración 54. Codificación del modelo de datos backend 122

Ilustración 55. Codificación servicio de almacenamiento 125

Ilustración 56. Codificación multiprovider 127

Ilustración 57. Codificación provider en inicio de sesión 128

Ilustración 58. Codificación de una vista 130

Ilustración 59. Ejemplo estructura de una vista 131

Ilustración 60. Codificación formulario cambio de contraseña 132

Ilustración 61. Codificación contenedor del formulario 133

Ilustración 62. Codificación widget "_FormRecoveryChangePassword" 134

Ilustración 63. Codificación input nueva contraseña 135

Ilustración 64. Pantalla final registro de usuario e inicio de sesión 136

Ilustración 65. Pantalla final inicio de sesión 136

Ilustración 66. Pantalla final creación de anuncio 137

Ilustración 67. Pantalla final visualización de anuncio 137

Ilustración 68. Pantalla final listado de previsualizaciones 138

Ilustración 69. Pantalla final perfil de usuario 139

Ilustración 70. Repositorio Github backend 140

Ilustración 71. Repositorio Github frontend 141

Ilustración 72. Resultados encuesta, pregunta 1 156

Ilustración 73. Resultados encuesta, pregunta 2 156

Ilustración 74. Resultados encuesta, pregunta 3 157

Ilustración 75. Resultados encuesta, pregunta 4 157

Ilustración 76. Resultados encuesta, pregunta 5 158

Ilustración 77. Resultados encuesta, pregunta 6 158

Ilustración 78. Resultados encuesta, pregunta 7 159

Ilustración 79. Resultados encuesta, pregunta 8 159

Ilustración 80. Resultados encuesta, pregunta 9 160

Ilustración 81. Resultados encuesta, pregunta 10 160

Resumen

TÍTULO: Prototipo de plataforma de anuncios para la comunidad universitaria UIS.

AUTORES: Jorge Andrés Triana Mojica, Jhon Andrés Parra Rodríguez.

PALABRAS CLAVE: Pregrado, Aplicativo, Móvil, Anuncios, Comunidad.

Descripción

En el último par de años la pandemia del Covid-19 afectó no solo la salud de muchas familias sino también sus finanzas y economías, esto trajo consigo que muchas personas en condición de vulnerabilidad económica se vieran forzadas a buscar ingresos extras para su sostenimiento a través de métodos informales, problemática que ha afectado especialmente a estudiantes universitarios que necesitan ingresos para costear sus estudios y su sostenimiento. A raíz de esto en la actualidad la Universidad Industrial de Santander presenta una problemática de ventas ambulantes e informales y por ello se percibe una necesidad cada vez más creciente de un espacio donde los estudiantes puedan anunciar y promocionar sus artículos y servicios, sin ocupación de espacios al interior de la universidad que no están hechos para este fin, y a su vez, sin ir en contra del reglamento de la institución, por estos motivos nace la iniciativa del presente proyecto, la cual busca aportar una solución a estas problemáticas y contribuir a la reactivación económica de la comunidad, esto mediante el desarrollo de una plataforma de publicación de anuncios soportada por un aplicativo móvil, donde diferentes usuarios de la comunidad universitaria puedan hacer sus publicaciones de anuncios y dar a conocer sus productos y servicios.

---

* Trabajo de grado

** Facultad de Ingenierías Físico-mecánicas. Escuela de Ingeniería de Sistemas e Informática. Director: Carlos Adolfo Beltrán Castro

Abstract

TITLE: Prototype of advertising platform for the UIS university community.

AUTHORS: Jorge Andrés Triana Mojica, Jhon Andrés Parra Rodríguez.

KEY WORDS: Undergraduate, Application, Mobile, Advertisements, Community.

Description

In the past couple of years, the Covid-19 pandemic has affected not only the health of many families also their finances and economies, this has meant that many people in a condition of economic vulnerability were forced to seek extra income for their support through of informal sales, a problem that has especially affected university students who need income to pay for their studies and their support. As a result of this, the Universidad Industrial de Santander currently presents a problem of peddling and street sales, and for this reason there is an increasingly growing need for a space where students can advertise and promote their goods and services, without occupying spaces within the university that are not made for this purpose, and in turn, without going against the regulations of the institution, for these reasons the initiative of this project was born, which seeks to provide a solution to these problems and contribute to the economic reactivation of the community, through the development of an advertisement publication platform supported by a mobile application, where different people of the university community can make their advertising publications and publicize their goods and services.

---

* Trabajo de grado

** Facultad de Ingenierías Físico-mecánicas. Escuela de Ingeniería de Sistemas e Informática. Director: Carlos Adolfo Beltrán Castro

Introducción

En la actualidad nuestra institución se enfrenta a una problemática, una problemática social que se evidencia entre los pasillos, salones de clase y en los diferentes espacios de esparcimiento del campus, jóvenes que ante dificultades económicas deben recurrir al llamado coloquialmente “rebusque”, vendiendo diferentes artículos desde comestibles hasta el ofrecimiento de tutorías para obtener esos recursos que necesitan, yendo en muchos casos en contra de las mismas normas de la institución, invadiendo y obstaculizando espacios o simplemente incomodando a personas que no tienen interés en sus productos, todo esto a causa de no tener un espacio permitido para estas actividades.

Nuestra iniciativa es la de proponer un espacio que, desde la virtualidad, pueda suplir esta necesidad de la comunidad universitaria, sin ir en contra de las normas de la institución, buscando que estos miembros de la comunidad dejen sus espacios físicos dentro de la universidad y migren a un espacio virtual donde puedan dar a conocer sus productos, servicios y demás iniciativas que desean exponer a la comunidad universitaria.

Son también muchos los miembros de la comunidad universitaria que tienen la necesidad de poseer un espacio donde puedan encontrar con facilidad, orden y rapidez cosas tan necesarias para muchos estudiantes como lo es una tutoría para una materia difícil, una habitación en arriendo donde poder vivir o alguien a quien comprar su alimentación diaria. Actualmente, suplen la necesidad de este espacio mediante el uso de plataformas de redes sociales tales como Facebook, donde el acceso puede ser limitado, restringido, difícil, muy desordenado y caótico, ya que son plataformas y espacios que no están hechos ni fueron pensados para este fin.

Para buscar dar una solución a esta problemática se realiza el presente proyecto, el cual, mediante una plataforma de anuncios, dirigida a la comunidad universitaria y accesible mediante un aplicativo móvil, busca aportar a los estudiantes este espacio dedicado para este fin.

En el presente documento se relatan las diferentes etapas del proyecto, fijando los objetivos generales y específicos del mismo; precisando su contexto, marco teórico y marco tecnológico; definiendo sus requerimientos y requisitos funcionales y no funcionales; razonando su diseño de software, a través de diagramas UML y diferentes parámetros de la ingeniería del software; delimitando las pautas de diseño de la interfaz a seguir en el aplicativo, a través de diferentes mockups o prototipos de baja fidelidad, trazados con ayuda de herramientas profesionales, velando en ellos el cumplimiento de los requerimientos funcionales y no funcionales del proyecto; ilustrando la programación empleada en el aplicativo incluyendo su parte front-end y back-end, además de la descripción del framework usado, técnicas, métodos y patrones de diseño en los cuales fue elaborado y esta soportado el aplicativo; exponiendo la verificación y pruebas de la plataforma, mostrando las pruebas realizadas, el correcto cumplimiento de los objetivos y requerimientos propuestos, además de los datos de uso recolectados y los posibles puntos de mejora del proyecto.

1. Justificación y definición de la situación actual:

En vista de la complejidad a la cual se puede enfrentar una persona dentro de la comunidad universitaria de la UIS al querer obtener, donar o intercambiar algún elemento que requiera en su día a día o que haya entrado en desuso, y además a la necesidad que hay dentro de la comunidad universitaria de dar a conocer servicios propios tales como tutorías, asesorías, traducciones, préstamos de utensilios, o hasta el hecho de dar inmuebles en arriendo en los barrios aledaños al campus, y como reflejo de estas necesidades, se evidencia una cantidad sustancial de ventas informales dentro del campus universitario, conllevando a que sea muy común encontrar personas queriendo vender objetos o promocionarse, deambulando en el campus buscando potenciales clientes. Para contribuir a solucionar esta problemática, se ha optado por desarrollar un aplicativo móvil dirigido a la comunidad que presenta está necesidad, en la cual puedan dar a conocer sus avisos clasificados y se logren contactar ambas partes de una forma organizada para otorgar una solución fácil y rápida a la hora de requerir o dar algún elemento. Además, este desarrollo está orientado a brindar un apoyo importante a muchos estudiantes, diferentes miembros de la comunidad universitaria y sus alrededores, los cuales para ayudarse a sustentar sus gastos recurren a ventas informales ante la gran problemática de la no existencia de un medio organizado y confiable para dar a conocer sus productos a otras personas, acudiendo por esta razón a medios indirectos y muchas veces incómodos, o a plataformas no diseñadas para realizar estos procesos.

2. Objetivos

2.1 Objetivo general

- Desarrollar un aplicativo móvil que facilite la interacción entre los integrantes de la comunidad UIS interesados en anunciar y/o demandar productos usados o nuevos, bienes o servicios.

2.2 Objetivos específicos

- Facilitar el contacto de una forma sencilla entre los miembros de la comunidad UIS interesados en un anuncio, con el anunciante del mismo, siendo un puente para dar un canal de comunicación entre ambas partes.

- Desarrollar un sistema de clasificación y búsqueda de anuncios por categorías y palabras clave, que sea sencillo de usar y posea la capacidad de filtrar, según las instrucciones del usuario, para luego listar los anuncios mostrados a él.

- Hacer que el sistema sea fácil de usar, brindando al usuario de manera cómoda la posibilidad de publicar un anuncio.

- Diseñar un prototipo software que cumpla los requerimientos del aplicativo móvil, requerimientos basados en las necesidades de los usuarios que estarán presentes en el sistema, tales como:
    - Permitir la publicación de anuncios.
    - Permitir la clasificación de los anuncios publicados.
    - Permitir la visualización de anuncios de todos los usuarios, facilitando el

contacto entre los interesados.

- Permitir a cada usuario la posibilidad de evaluar una publicación dándole un punto positivo o negativo según sea el caso.
- Poseer un registro de un perfil de usuario en la plataforma, en donde este pueda visualizar sus anuncios y datos personales, en el cual el usuario cuente con un rango positivo o negativo con base en los puntos positivos o negativos recibidos en sus publicaciones.

- Validar las versiones del software desarrollado, verificando el cumplimiento de los objetivos y requerimientos del mismo.

2. Marco teórico

2.1 Marco conceptual

2.1.1 Anuncio Clasificado

Comúnmente vistos como parte de la publicidad en periódicos, programas de radio, sitios web, diarios o revistas (*Wells; Moriarty; Burnett, 2006*), los anuncios clasificados, también conocidos en algunos países como anuncios de búsqueda (*Danna, 2002*), tienen la función de, por una tarifa, ofrecer información al consumidor acerca de los bienes y servicios que se pueden comprar, vender o intercambiar por parte de un anunciante, los más comunes ofrecen inmuebles en venta, autos usados u ofertas de empleo, entre otras cosas. Reciben su nombre ya que suelen agruparse por categorías o temas en común, como, por ejemplo, ya sea por ser ventas, buscados o servicios ofrecidos.

Estos se diferencian de la publicidad estándar al permitir que individuos privados puedan ofrecer sus productos a un relativo bajo coste, además, de ser generalmente mejor recibidos que la publicidad en general (*Lorimor, 1977*). Normalmente están basados en textos y pueden consistir en algo muy resumido, conteniendo elementos como el tipo de producto que está siendo promocionado, el número de teléfono de contacto o una serie de imágenes, videos o demás contenido multimedia del producto en el caso del formato web.

2.1.2 E-commerce

En la actualidad se ha hecho frecuente el uso de este término junto con el de comercio electrónico (*RAE, 2020*), este está presente cuando existe una compra y venta de productos o de servicios a través de internet, esto empleando algún medio digital, tal como lo son redes sociales, aplicaciones móviles o páginas diseñadas para este fin (*Way2ecommerce, 2015*), usando como

forma de pago medios electrónicos a través de internet; siendo popular el uso de tarjetas de crédito y plataformas de pago como Paypal o Google Pay.

Internet es la clave de los comercios electrónicos ya que este brinda demasiadas ventajas a la hora de establecer un comercio en comparación con un comercio físico (*Hindle, 2009*), su ventaja más evidente es la inmensa reducción de costes de instalaciones y mantenimiento, como el alquiler o la necesidad de adquirir un local para exponer sus productos a un público, la posibilidad de poder tener abierto el comercio las 24 horas del día y los 365 días del año, posibilitando la realización de compras o demás actividades comerciales a cualquier hora, gracias a la disponibilidad constante que brinda estar localizada en la web, y esta disponibilidad no se encuentra solo a nivel temporal sino también a nivel espacial, ya que el alcance de un comercio electrónico puede llegar a ser de nivel mundial y no se limita al área de una localidad física; además de todo esto la facilidad para obtener y registrar la información, preferencias y comportamientos de los clientes dentro del comercio permite hacer labores de análisis de datos, big data, el trazo de patrones de comportamiento y la mejora de estrategias de marketing, permitiendo una mejor segmentación de clientes, una publicidad e información mejor dirigida y a su vez una publicidad mejor recibida por potenciales clientes.

2.1.3 E-marketplace

Muchos estudios apuntan cada vez más a la relevancia de las ventas en línea, 7 de cada 10 internautas declaran comprar online; (*IAB Spain, 2017*), (*Elogia, 2017*) Y no es para menos, cada vez la virtualidad está más presente en el día a día, viendo expandir su crecimiento exponencialmente durante es el aislamiento del año 2020, contribuyendo al éxito de los servicios de mercado online los cuales ya iban en aumento desde el año 2014 (*Forbes, 2014*).

Se define un e-marketplace como una plataforma online creada por una empresa que actúa como un tercero neutral donde proporciona información sobre productos o servicios para poner en contacto compradores y vendedores (*In Lee, 2016*), desde esta definición y tomando un punto de vista más general, un e-marketplace es un conjunto de e-commerce, donde el marketplace actúa como intermediario, siendo el canal que comunica el vendedor con sus potenciales clientes.

Su principal ventaja es la de poseer gran variedad de proveedores y permitir que minoristas puedan acceder al mercado y ofrecer sus productos, además de disponer de una variedad y selección más amplia de productos y servicios con una disponibilidad mejor que en la de un comercio electrónico especifico (*Bloomberg, 2008*).

2.1.3.1 Economía Colaborativa

Alex Stephany, fundador del mercado en línea JustPark define la economía colaborativa como el valor económico que surge de hacer que los activos infrautilizados estén disponibles en línea (*Arun Sundararajan, 2016*), siendo esta la búsqueda de obtener el máximo provecho de un activo buscando desperdiciar al mínimo su utilidad aprovechando las nuevas tecnologías para lograr ello (*Santander, 2021*) basándose en el principio del acceso a bienes y servicios sin que sea necesaria su propiedad (*Finanzasparamortales, 2017*) y siendo esta una de las principales razones de ser de un Marketplace.

La economía colaborativa tiene una fuerte inspiración en la filosofía de código abierto (*Hamari; Sjöklint; Ukkonen, 2016*), y se centra principalmente en sectores como el del alojamiento, el del transporte compartido, la venta o intercambio de artículos de segunda mano y la restauración o remate de objetos caídos en desuso o cercanos a ser desechados que pueden ser aprovechados.

La necesidad de búsqueda de eficiencia y ahorro por temas medioambientales o económicos se ha convertido en un tema prioritario en la población de la actualidad, en especial entre los más jóvenes, muchos de los cuales buscando esto se unen o fundan iniciativas bajo el principio de economía colaborativa, apuntando a obtener, en relación con productos más convencionales, una mejor sostenibilidad y cuidado del medio ambiente, sacando el máximo provecho a la vida útil de sus productos y hasta en algunos emprendimientos dándoles una segunda vida útil, reutilizándolos y evitando la necesidad de consumir productos nuevos.

Existen diferentes plataformas de intercambio o venta de artículos que ya no se necesitan pero que pueden servir para otros como la plataforma Española de artículos de segunda mano de Wallapop (*Wallapop, 2013*) o diversos ejemplos de proyectos de código abierto que apuntan a marketplaces entre pares, como el caso de Sharetribe (*Sharetribe, 2022*) y Cocorico (*Cocolabs SAS, 2021*) y como ejemplos de economía colaborativa fuera de un Marketplace tenemos a CouchSurfing, servicio de alojamiento y viajes, que, en 2010, logró constituirse como una corporación con fines de lucro y en 2014, servicios que basan su economía colaborativa en el principio de compartir la propiedad, como Uber y Airbnb tuvieron éxito en el mercado (*Codagnone; Karatzogianni; Matthews, 2018*) y en el caso de Uber este ha llegado a tener un valor de 213 millones de dólares.

2.1.5 Aplicativo Móvil

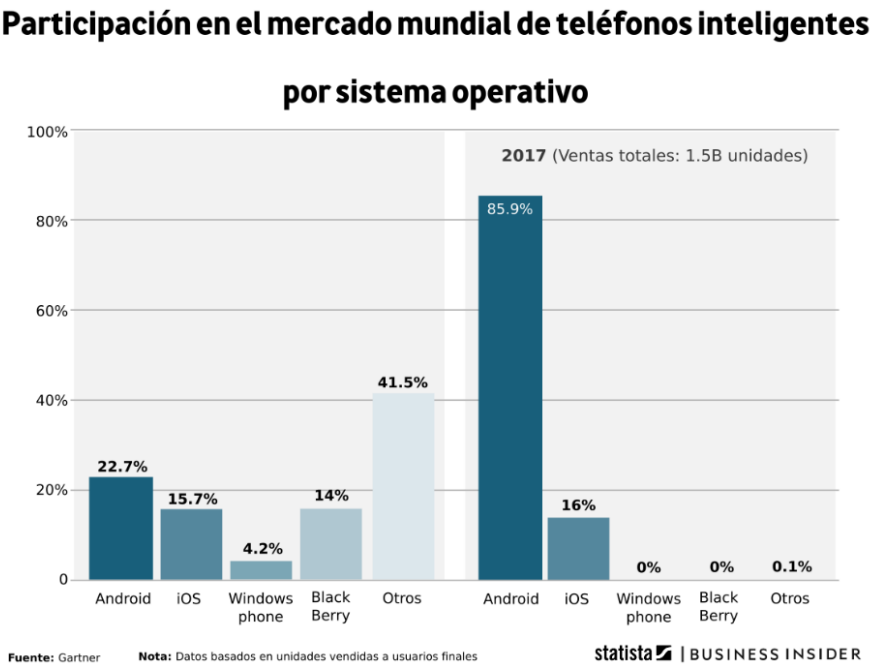
Desde aquella presentación del primer Iphone en el año de 2007 los smartphones han llegado a las manos de todos nosotros y con ellos sus incontables aplicaciones, potenciadas con la masificación de la red, brindándonos una innumerable cantidad de herramientas y soluciones a muchos de nuestros problemas, pero ¿qué es un aplicativo móvil? Un aplicativo móvil, o como

muchos lo conocemos una app, es el tipo de software diseñado para ejecutarse dentro de un dispositivo móvil, ya sea un smartphone o una tableta.

Existen diferentes sistemas operativos para móviles, compitiendo entre si desde hace años, y por ello la construcción y distribución de sus aplicaciones varían según a cuál sistema operativo se oriente la app, como se puede ver en la ilustración 1, con el paso de los años son 2 los sistemas operativos que dominan el mercado a día de hoy, la plataforma iOS de Apple Inc y el sistema operativo Android perteneciente a Google , los cuales definen, cada uno para sí mismos, como es la distribución, términos de uso y software de desarrollo para cada una de sus apps, estando así repartida la distribución de los aplicativos móviles según cual sea su sistema operativo, la Google Play Store en el caso de Android y la App Store en el caso de Iphone.

Ilustración 1. *iOS vs. Android*

(Computerhoy, 2018)



De forma general, a la hora de construir una aplicación móvil existen tres tipos de aplicativos, las aplicaciones nativas, las aplicaciones hibridas y las aplicaciones basadas en la web, y esta última puede poseer dos soluciones diferentes, haciendo esto que existan en total cuatro enfoques posibles de desarrollo; aplicaciones nativas, diseñadas específicamente para un sistema operativo en cuestión; aplicaciones multiplataforma, quienes al igual que las aplicaciones nativas se compilan en una aplicación que se ejecuta directamente en el sistema operativo del dispositivo pero a diferencia de estas pueden usar un framework o lenguaje de programación diferente al de este; aplicaciones web hibridas, las cuales son creadas con tecnologías web estándar como Javascript, CSS y HTML5 y se incluyen como paquetes de instalación de aplicaciones, funcionando como una aplicación web, pero con un contenedor nativo; aplicaciones web progresivas, también conocidas como PWA, ofrecen un enfoque alternativo al desarrollo tradicional de aplicaciones móviles al omitir la instalación de aplicaciones y la tienda de aplicaciones al ejecutarse a través de un navegador web, buscando ofrecer una experiencia similar a la de una aplicación móvil tradicional (*Amazon Web Services, Inc.).*

2.1.5.1 Aplicaciones Nativas

Son aplicaciones dirigidas y diseñadas para un sistema operativo particular, son propios de una plataforma específica, por lo que, por ejemplo, una aplicación diseñada para dispositivos Apple no se ejecutará en dispositivos Android, y viceversa.

Contrario a lo que pudiera parecer las aplicaciones nativas poseen grandes ventajas, ya que suelen poseer grandes mejoras en el rendimiento, su consistencia y estabilidad, además pueden brindar por ello una mejor experiencia de usuario, debido a que se aprovecha la interfaz de usuario del dispositivo nativo y se accede a una gama de opciones proporcionada por la API que permite acelerar el desarrollo y construcción de esta, además de ampliar los límites para el uso de la app.

Su principal inconveniente es su costo de construcción, debido a que, como ya se señaló en su definición, se enfoca a un sólo sistema operativo móvil, ejecutándose sólo en esa plataforma, excluyendo a las demás, por ejemplo, como una aplicación para Android no se puede ejecutar en iOS, si se está desarrollando una aplicación para Android y se desea también liberar esta misma aplicación para iOS se deberían tener dos equipos de desarrollo trabajando en paralelo.

2.1.5.2 Aplicaciones basadas en Web

Este es un tipo particular de software permite a los usuarios interactuar con un servidor remoto a través de una interfaz web (*Lvivity, 2018*), este tipo de aplicaciones poseen muchas ventajas en comparación con las aplicaciones nativas, sobre todo en el ámbito de la portabilidad, ya que estas liberan a los usuarios de la necesidad de instalar software adicional y además permiten que los desarrolladores no tengan que escribir una nueva versión del aplicativo para una plataforma distinta a la cual fue creada originalmente, posibilitando así su uso en diferentes sistemas operativos, siendo una solución multiplataforma, esto se logra a través de un navegador web compatible y una conexión a internet activa.

Como ventaja adicional este tipo de aplicaciones pueden ser desarrolladas empleando diferentes lenguajes de programación, tecnologías y frameworks, facilitando su desarrollo además de brindar una mayor seguridad en el almacenamiento de los datos del usuario al guardar estos datos en la nube y en servidores dedicados para este fin que son monitoreados contantemente para evitar ataques.

Uno de los mayores inconvenientes de en este tipo de aplicativos es su bajo rendimiento en comparación con aplicaciones nativas, el cual depende de la capacidad del servidor donde este alojado, además este tipo de aplicaciones suelen tener una fuerte dependencia a la conexión de red del dispositivo, haciendo que su rendimiento y velocidad dependa mucho de la estabilidad y

velocidad de la conexión de internet del usuario haciendo que estas aplicaciones en algunas ocasiones no puedan funcionar por completo si el usuario pierde su conexión a internet.

2.1.5.3 Aplicaciones Híbridas

Este tipo de aplicaciones buscan aprovechar las ventajas multiplataforma de las aplicaciones web sin tener que abandonar por completo el uso de las aplicaciones nativas, esto lo logran insertando dentro de una aplicación nativa, que actúa como contenedor, un aplicativo web, lo cual las permite ser usadas en varias plataformas con una misma base de código y logran acceder a diferentes funcionalidades de los dispositivos tales como cámara, contactos, GPS, entre otros utilizando código nativo (*Viallon, 2017*).

Poseen un coste mucho menor en desarrollo que el de una app nativa, pero presentan como su problema principal el de no estar optimizadas para cada sistema operativo, lo que puede disminuir el rendimiento de estas.

Se puede diferenciar también dentro de las aplicaciones híbridas el tipo de tecnología que usan, como las tecnologías que traducen a código nativo tales como React Native, la cual, aunque se tarda más en escribir código se presenta una mejora en el rendimiento de estas, acercándose a un rendimiento nativo, o las que ejecutan una aplicación como una aplicación web dentro de un contenedor o web frame que se ejecuta dentro de una app nativa, que suelen tener un poco menos rendimiento y sus elementos visuales presentan diferentes formas a los de una aplicación nativa en general.

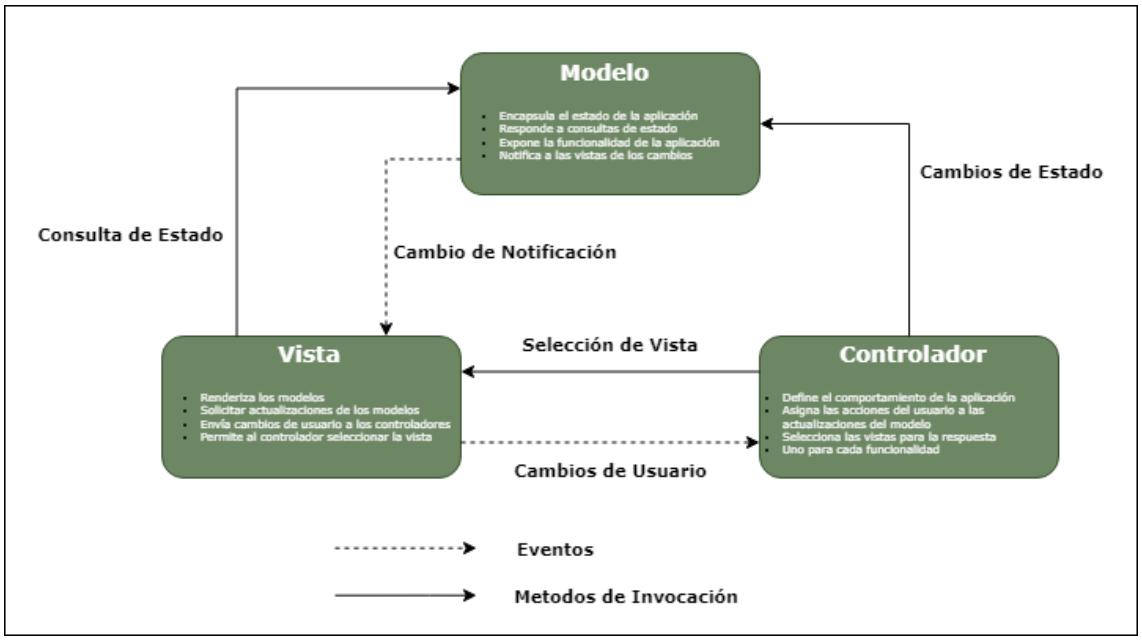
2.1.6 Modelo Vista Controlador

Es un patrón de arquitectura de software muy poderoso y conveniente al permitir el fácil desarrollo de grandes proyectos, se basa en dividir todo el sistema en tres elementos clave: modelo,

controlador y vista. La vista presenta el contenido del modelo, proporcionando una interfaz gráfica al usuario que permite ver la información proporcionada por el modelo de tal forma que si los datos del modelo se ven alterados la vista debe actualizarse, el controlador permite la interacción entre el modelo y la vista según las acciones del usuario con ella, traduciendo las interacciones del usuario con la vista en acciones que realizará el modelo, está gestiona los datos proporcionados por el modelo y los envía a la vista y viceversa, el modelo es un esquema en el cual se guardan los datos del sistema y sus reglas que rigen el acceso y actualización de los mismos, en la ilustración 2 podemos ver un ejemplo de una implementación de un modelo vista controlador.

Ilustración 2. *Ejemplo de implementación del Modelo Vista Controlador*

(R. Eckstein, 2007)

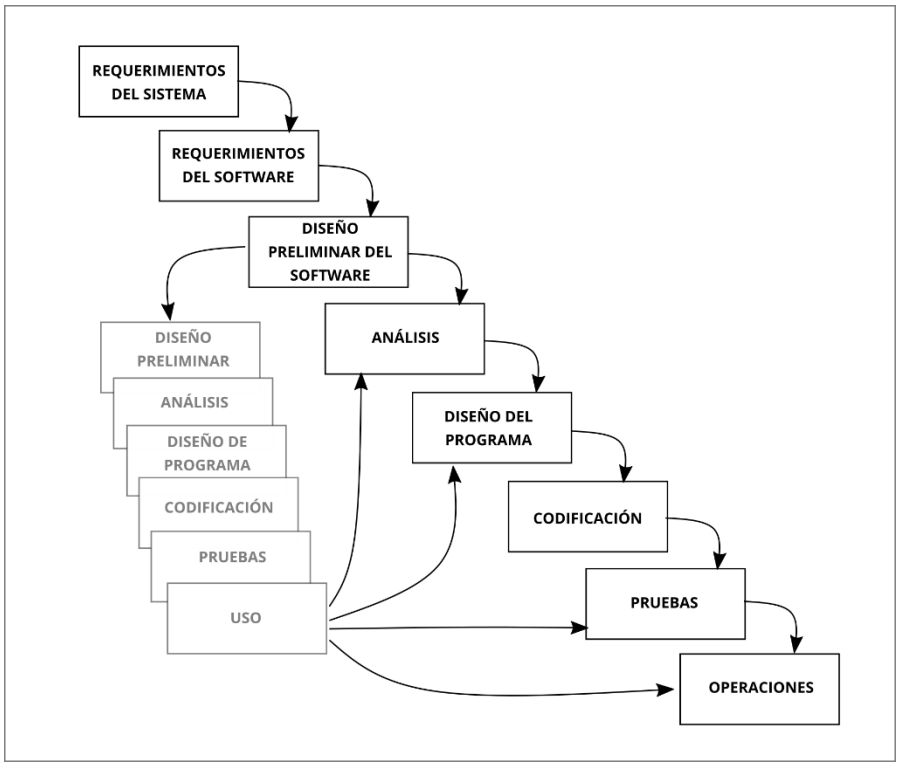


2.1.7 Desarrollo en cascada

El modelo de desarrollo en cascada se sustenta en el desglose de las actividades en fases estructuradas de forma secuencial, donde cada fase depende de los entregables de la anterior y corresponde a una especialización de tareas (*Petersen; Wohlin; Baca, 2009*). Este modelo atraviesa diferentes etapas de forma lineal, donde recorre pasos como la concepción del proyecto, su

inicialización, su análisis, su diseño, su construcción, las pruebas, su despliegue y mantenimiento (*Hughey, 2009*).

Ilustración 3. *Modelo cascada original de Winston W. Royce (Bremen University)*



Como puede verse en la ilustración 3, el modelo original de Royce sigue las siguientes fases en orden donde debe pasase a la siguiente fase solo cuando se revisa y verifica su fase anterior (*Bremen University*):

1. Requisitos del sistema y software: capturados en un documento de requisitos del producto.
2. Análisis: resultando en modelos, esquemas y reglas de negocio.
3. Diseño: dando como resultado la arquitectura de software.
4. Codificación: el desarrollo, prueba e integración de software.

5. Pruebas: el descubrimiento sistemático y la depuración de defectos.
6. Operaciones: la instalación, migración, soporte y mantenimiento de sistemas completos.

2.1.8 Whatsapp

Mundialmente conocida, Whatsapp es un aplicativo móvil de mensajería instantánea y de voz sobre IP (VoIP), que permite la comunicación entre pares a través de mensajes de texto, llamadas y mensajes de voz, y permite compartir imágenes, documentos, ubicaciones de usuarios y contenidos varios (*Orson, 2015*), fue desarrollado por WhatsApp Inc y a día de hoy es propiedad de la compañía Meta, se ha convertido en la aplicación más popular del mundo desde 2015 (*Metz, 2016*) y es la aplicación de mensajería instantánea número uno en cantidad de usuarios, alcanzando más de 2 mil millones de usuarios en febrero de 2020 (*WhatsApp.com, 2020*), además cuenta con un uso masivo y generalizado a nivel mundial, en especial en occidente. La función clave y distintiva de Whatsapp es la poseer un sistema de cifrado de extremo a extremo, con la cual blinda y protege a las comunicaciones dentro aplicación de ataques de tipo man-in-the-middle, cifrando los mensajes del aplicativo usando la biblioteca de código abierto Signal Protocol y encriptando sus llamadas utilizando el protocolo de transporte en tiempo real seguro SRTP (*WhatsApp Inc, 2016*).

2.2 Marco tecnológico

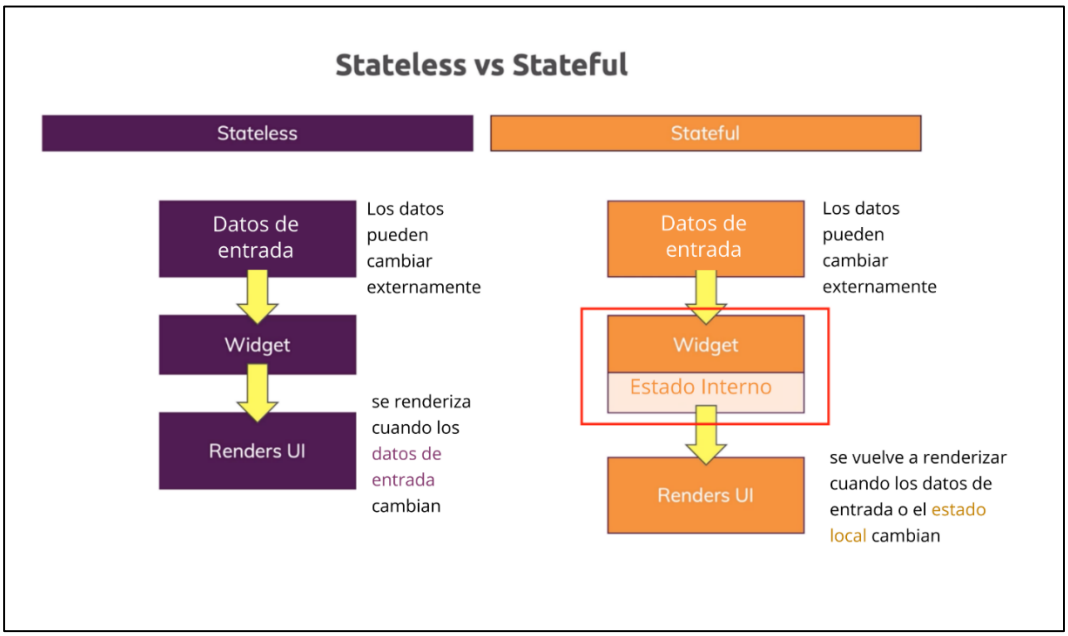
2.2.1 Flutter

Es un kit de herramientas de código abierto, creado por Google en 2016, permite la creación de aplicaciones multiplataforma compiladas de forma nativa a partir de una única base de código (*Flutter, 2022*). Flutter posee un rendimiento nativo, además de un sistema de widgets que facilitan la elaboración de aplicaciones, estos están clasificados en tres tipos generales, Stateful widgets, el cual a pesar de que todas las instancias de un widget son inmutables permite, mediante el método setState, la interacción entre el usuario y la aplicación actualizando el estado del widget; Stateless widgets, actuando como una constante, impidiendo que el contenido del Widget pueda ser actualizado; Inherited widgets, el cual a pesar de ser inmutable y stateless permite que otros widgets puedan adherirse a él, heredando su estado, y por tanto al reemplazarse este widget padre por una nuevo todos sus widgets hijos son redibujados.

Un stateful widget es usualmente usado como el controlador, mientras que el stateless widget actúa como la vista y el inherited widget será su archivo de configuración o el modelo del sistema (*K, Pradeep, 2020*), cumpliendo así el modelo vista controlador, en la ilustración 4 podemos ver la comparativa entre la estructura de un stateful widget contra la de un stateless widget.

Ilustración 4. *Comparación stateless widget vs stateful widget*

(K, Pradeep, 2020)



Las aplicaciones en Flutter están escritas en un lenguaje de programación desarrollado por Google, llamado Dart. Durante el año 2021 el framework sufrió importantes actualizaciones entre ellas la actualización de una nueva versión de Dart, Dart 2.0, cuyo principal cambio fue la incorporación sound null-safety, característica que provocó muchos cambios importantes e incompatibilidades con paquetes externos.

2.2.2 NodeJS

Lanzado en 2009, NodeJs es un entorno de ejecución multiplataforma que permite la creación de aplicaciones del lado del servidor fuera de un navegador, permite la creación de servidores web y herramientas de red escritas en JavaScript y una colección de módulos los cuales pueden ser usado para brindar distintas funcionalidades (*Teixeira, 2012*). Usa un modelo orientado a eventos asincrónicos enfocado en la entrada/salida de datos apuntando a la optimización del

rendimiento y una mejor estabilidad en aplicaciones web. Al ser un proyecto open-source posee una gran comunidad que al cabo de los años han desarrollado dependencias sólidas las cuales pueden ser usadas de manera gratuita mediante NPM, el sistema de gestión de paquetes propio de Node.js. Entre las corporaciones más importantes que hacen uso de este software están Netflix, Walmart y Amazon Web Services.

A pesar de que JavaScript es el único lenguaje de programación que permite Node.js de forma nativa, existen diversos transpiladores que traducen a JavaScript, como resultado de esto es posible escribir aplicaciones en lenguajes como Dart, TypeScript, CoffeeScript entre otros (*Jashkenas, 2022*).

Node.js lleva la programación basada en eventos a los servidores web, lo que permite el desarrollo de servidores web rápidos en JavaScript. Los desarrolladores pueden crear servidores escalables sin utilizar subprocesos mediante el uso de un modelo simplificado de programación basada en eventos que utiliza devoluciones de llamada para señalar la finalización de una tarea. Node.js conecta la facilidad de un lenguaje de secuencias de comandos como lo es JavaScript con el poder de la programación en red de Unix (*Teixeira, 2012*).

2.2.2.1 Express

Basado en el módulo http de Nodejs, es el framework más popular para la creación de aplicaciones del lado del servidor de NodeJS. Los componentes que conforman el framework se conocen como middlewares y son la columna principal del mismo. Permite una flexibilidad a la hora de personalizarlos y esto concede a los desarrolladores el poder configurar sus aplicaciones web de una manera más escalable, permitiendo su trazabilidad a lo largo del tiempo.

2.2.3 GitHub

Github es una plataforma de alojamiento de código abierto, desarrollo de software y control de versiones, la cual cuenta con características especialmente útiles para equipos que trabajan en proyectos de desarrollo de software. Posee un sistema de control de acceso y funciones de colaboración que facilitan el seguimiento de errores, la solicitud de nuevas características para proyectos, la gestión de tareas y la creación de wikis colaborativas gracias a estas se puede alcanzar más fácilmente una mayor productividad en los equipos de trabajo además de permitir centralizar un proyecto de software y mantener su seguimiento en cada una de las fases del mismo (*Williams, 2012*).

2.2.4 JSON

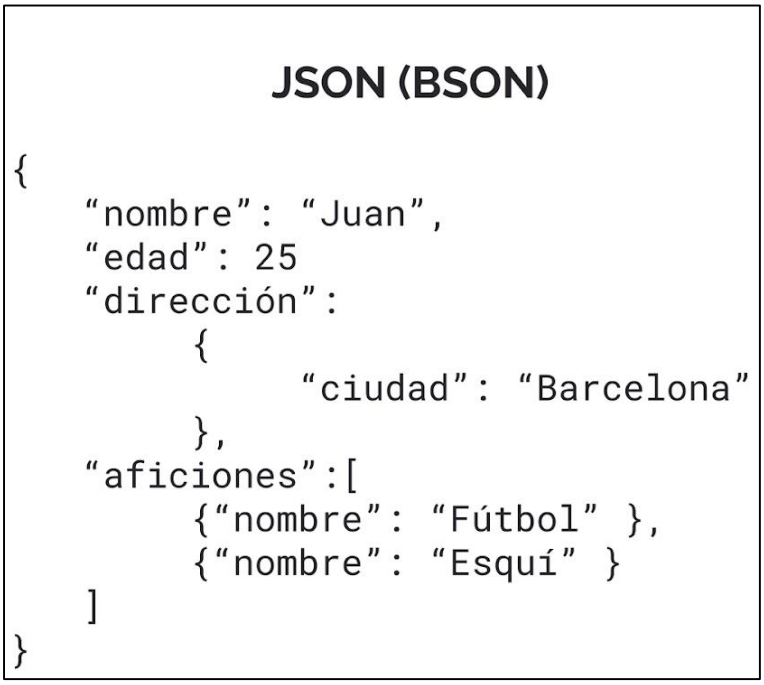
Usado ampliamente como una alternativa a XML, JSON es un sintaxis de texto, de código abierto, usable y entendible por cualquier persona, ya que emplea texto legible por personas de forma natural, es principalmente usado para la transmisión de arreglos y demás datos en pares atributo y valor, es especialmente útil el uso de este formato para la trasmisión de información en entornos donde el tamaño entre el flujo de datos entre el cliente y el servidor posee una gran importancia y además la fuente de estos datos es fiable. A pesar de que JSON proporciona un marco sintáctico para el intercambio de datos posee también muchos derivados con muchas semánticas posibles, donde el intercambio de datos sin ambigüedades requiere un acuerdo entre el productor y el consumidor sobre la semántica específica a emplear de la sintaxis JSON.

2.2.5 MongoDB

MongoDB es un sistema de base de datos cuya principal característica es la de no poseer un mecanismo de almacenamiento SQL, al producir bases de datos no relacionales, es de código abierto y es un sistema de base de datos enfocado a documentos, el cual consta de documentos

individuales con un identificador único, con una estructura tal como se ejemplifica en la ilustración 5, la cual organiza la información de una manera flexible en estructuras similares a JSON llamadas BSON (la cual es la representación binaria de JSON), no posee un esquema para los documentos lo que da la libertad de agregar datos dependiendo de las necesidades de cada registro. Además, posee un sistema de referenciación lo que permite realizar consulta entre diferentes documentos como lo haría la sentencia JOIN en una base de datos relacional (*Kerby, 2015*).

Ilustración 5. *Ejemplo de la estructura de un documento en MongoDB (Datademia, 2020)*



2.3 Estado del arte

2.3.1 Mercado libre

Mercadolibre es una compañía de comercio electrónico la cual proporciona a sus usuarios los servicios de compra, venta, anuncio y pago de diferentes bienes y servicios nuevos o usados que sus mismos usuarios (los cuales van desde particulares, productores, fabricantes, minoristas, mayoristas, hasta grandes empresas) se encargan de publicar en su plataforma.

Mercado libre cuenta con:

- Organización de sus productos según la categoría que correspondan.
- Un buscador que te permitirá llegar mucho más rápido al producto que deseas y con las características específicas que buscas.
- Un historial de productos vistos recientemente.
- Una sesión donde puedes encontrar las tiendas oficiales según su categoría.
- Sesión de ofertas y demandas.
- Cuenta con una herramienta de Ayuda o PQR.
- Permite la elección entre diversos medios de pago y también permite la opción de diferir los pagos hasta en 48 cuotas.
- Cuenta con servicio de envío gratuito a partir de un monto mínimo de compras.
- Cuenta con servicio de carrito de compras dónde podrás ver tus productos seleccionados, el coste de cada uno de ellos y el monto total de la compra.
- Tiene una sesión donde puedes ver tus compras y hacer un seguimiento de las mismas.

Para el 10 de agosto del 2020 Mercadolibre registra un aproximado de 11 millones de vendedores y 52 millones de compradores activos; 16 compras por segundo y 425 visitas por segundo; 17 millones de nuevos usuarios registrados durante Q2 2020; 289 millones productos listados (al cierre de Q2 2020), la cantidad de artículos enviados en el segundo trimestre del 2020 por MercaEnvíos fue un 124% más de los enviados en el año anterior, se registró que en el último trimestre del año un promedio de ventas por alrededor de los 178.50 millones de artículos vendidos (*Mercadolibre, 2020*). A la fecha la app mobile de Mercadolibre cuenta con más de 100 millones de descargas y aproximadamente 9 millones y medio de comentarios los cuales la dejan con un promedio de 4.8 estrellas de calificación (*Marketplacepulse, 2020*).

2.3.2 eBay

Así como Mercado libre, Ebay también es un sitio destinado al comercio electrónico, el cual también incluye la función de subasta de productos siendo esta una de las opciones más llamativas del sitio. Esta función de subasta funciona de la siguiente manera: el vendedor se encarga de colocar el producto a subastar, un precio de inicio y un tiempo de fin de dicha subasta; durante este tiempo los interesados en el producto subastado realizarán sus pujas por él y el que haya ofrecido la puja más alta al momento de acabar el tiempo es el que se llevará el producto subastado (*Ebay, 2020*).

Además de esto E-bay tiene a disposición de sus usuarios otras funcionalidades, tales como:

- Una buena organización de sus productos que va dependiendo de la categoría a la que pertenezcan.
- Posee una sesión de guardado en la cual se guardarán los productos que hayas seleccionado como favoritos al momento de estar navegando entre sus ofertas.

- Un botón de búsqueda avanzado donde puedes especificar alguna categoría o producto con ciertas especificaciones a buscar.

- Cuenta además con una sesión de “Por menos de $10” en la que podrás encontrar todo tipo de mercancía que se encuentre en ese rango de precio.

- Cuenta igualmente con notificaciones para que estés pendiente de si tus productos aún siguen disponibles o para saber cómo está el estado de tus pedidos.

- Posee un carrito de compras dónde podrás ver tus productos seleccionados, el coste de cada uno de ellos y el monto total de la compra.

- Una serie de “ofertas EBay” donde puedes encontrar descuentos destacados del momento.

- Una sesión de ayuda para resolver cualquier inquietud que se tenga acerca de la navegación en la página, compras, devoluciones, entre otros.

A la fecha registra unos 182 millones de usuarios registrados en su sistema, la categoría de productos más vendida fue la de “Electrónicos y accesorios” registrando un 16,4% de las ventas, se compraron y vendieron bienes por un valor de 22 millones en el último trimestre del 2019, hay alrededor 1.300 millones de anuncios publicados en ella, el 71% de sus ventas se envían gratuitamente, solo un 20% de los productos ofrecidos son de segunda mano. Además de esto su app mobile tiene más de 100 millones de descargas, un aproximado de 3 millones 400 mil comentarios los cuales la dejan con un promedio de 4.7 estrellas (*Oberlo, 2020*).

2.3.3 Facebook Marketplace

Facebook marketplace se ha propuesto como una reciente alternativa de compra, venta e intercambio de bienes nuevos o usados permitiendo la interacción entre usuarios de un modo

rápido y sencillo. Este servicio se encuentra disponible tanto desde el pc como desde la aplicación móvil de Facebook lo cual lo hace mucho más versátil y amigable.

Esta función de Facebook marketplace cuenta con propiedades como:

- Permite la acotación de búsquedas por medio de filtros específicos.
- Permite describir el producto a vender o intercambiar.
- En el área del usuario que desea comprar posee una gran variedad de categorías para mejorar la búsqueda de productos.
- También permite establecer topes mínimos y máximos para los precios del producto que se está buscando.
- En caso de inmuebles posee filtros más específicos como: cantidad de habitaciones, baños; costo que se desea buscar, zona en la que se desea buscar, entre otros.
- Facilita la conexión entre vendedor y comprador ya que permite el envío de mensajes directos entre ellos para que ellos mismos se encarguen de establecer las condiciones de la venta o intercambio.

Para mayo del 2018 se reportaron ventas alrededor de los 40 mil millones, aumentando así sus ganancias en más de la mitad. En ese momento, la plataforma Facebook era usada aproximadamente por 2 de cada 7 personas en el mundo todos los días (*Cnet, 2018*). Ahora bien, a día de hoy existen alrededor de 90 millones de páginas comerciales en Facebook y este servicio es utilizado por 140 millones de empresas; entre ellas se encuentran el 85% de las empresas estadounidenses las cuales lo han empezado a utilizar desde 2016, finalmente hay alrededor de 7

millones de personas que utilizan este servicio para colocar sus anuncios allí (*Businessofapps, 2020*).

2.3.4 Wallapop

Fundada en 2013, wallapop se ha convertido en la plataforma líder de compra y venta de productos de segunda mano en España a través de internet (*Rodríguez, 2018*), su característica más importante es el uso del sistema de geolocalización GPS propio de teléfonos inteligentes y tabletas dentro de la plataforma, con el cual usuarios puedan comprar y vender en función de su proximidad geográfica a través de la aplicación (*Galtes, 2017*).

Las principales funcionalidades de Wallapop se podrían listar en (*Funes, 2019*):

- Emplea un sistema geolocalización para los usuarios puedan encontrar ofertas cercanas.
- Mediante el uso de un chat, permite la comunicación directa entre comprador y vendedor.
- Permite a sus usuarios la negociación de precios con el vendedor.
- Indica el estado de disponibilidad sus usuarios, indicando si se encuentran o no en línea.
- La plataforma cuenta con sistema de entregas a domicilio con el cual esta se hace cargo del envió.
- La plataforma cuenta con filtros de búsqueda que permiten filtrar sus artículos por precio o según la categoría a la que pertenecen.
- Permite la construcción de listas de artículos favoritos para el seguimiento de sus precios.
- Permite a los vendedores, por un coste, destacar sus productos en la plataforma.

En 2015 la plataforma fue nombrada como la empresa emergente con mayores ingresos en España con mil millones de dólares en transacciones completadas utilizando la aplicación (*Elías, 2015*), en el año 2017 forma una alianza con la agencia de correos española para incorporar en la

plataforma el envio de artículos a domicilio (*Hdez, 2017*). En el último par de años ha experimentado un alto crecimiento en sus ventas, esto luego del año 2020, con el confinamiento por la pandemia del Covid19, además de esto durante el año 2021 recibió una fuerte inversión de capitales surcoreanos y franceses por valor de 157 millones de Euros, haciendo que el valor de la empresa alcanzara los 690 millones de Euros (*Servimedia, 2021*).

3. Metodología

Para correcta realización del proyecto, en su diseño e implementación se tomó la decisión de desarrollar el prototipo en una metodología divida en siete etapas, esto siguiendo los lineamientos del modelo de desarrollo en cascada, a continuación, están cada una de las etapas a elaborar en el desarrollo de la plataforma:

3.1. Determinación de requerimientos.

Esta fase corresponde a la formulación de requerimientos funcionales y no funcionales del software; en ella se hará la recolección de datos correspondientes para cada una de las partes del proyecto y se evaluará cada una de las posibles opciones, características y utilidades que serán incorporadas dentro de la plataforma.

3.2. Diseño del Software.

Durante el transcurso de esta etapa se hará el diseño lógico del software y así como también el diseño de las interfaces y diversas vistas que poseerá el aplicativo móvil, esto surgiendo luego del análisis de los requerimientos y la elaboración de diagramas de flujo, diagramas UML o diagramas entidad relación que los satisfagan, involucrando la conceptualización del proyecto.

Se modelarán diferentes prototipos de interfaz gráfica del aplicativo móvil que vayan cumpliendo con los objetivos del proyecto para luego pulir sus características, mejorar su diseño y adaptarlo a las posibilidades existentes en su programación para finalizar con un diseño final.

3.3. Desarrollo del software.

A lo largo del desarrollo de esta etapa se implementará toda la programación y codificación del proyecto, en síntesis, la ejecución del mismo, siguiendo los parámetros de diseño establecidos en la etapa anterior ejecutando la construcción del apartado frontend y backend del aplicativo

móvil, involucrando los elementos, plataformas, técnicas, frameworks y servidores de soporte más aptos encontrados a la hora de desarrollar el proyecto.

3.4. Pruebas

En la etapa de pruebas se verificará el apropiado funcionamiento de toda la programación en la plataforma, así como también el correcto funcionamiento de todas sus características y la correcta optimización de la misma, puliendo su programación, reparando y complementado los puntos que sean necesarios.

3.5. Implementación y verificación del programa.

En esta etapa el proyecto ya debe estar plenamente programado y desarrollado, aquí da inicio una fase de pruebas de software, haciendo una serie de ensayos, buscando posibles fallas de seguridad, errores de direccionamiento o errores de funcionamiento, esto con el objetivo de hacer la plataforma más sólida, robusta y poco vulnerable ante amenazas.

Se dará a conocer el aplicativo a un público de prueba el cual podrá, mediante una encuesta, manifestar su opinión o posibles errores encontrados por ellos, esto como respuesta a una serie de preguntas, comprobando la funcionalidad de la plataforma, comparándola según los requerimientos iniciales del proyecto, observando que falencias o inconsistencias posee para luego mejorarlas.

5.6. Documentación.

Esta etapa está compuesta por toda la documentación de las diferentes herramientas y páginas dentro de la plataforma, expresando y aclarando el funcionamiento de las secciones, para que si una persona ajena al proyecto, con su debida autorización, deseara analizar el código, pueda

este entenderlo sin problemas y a su vez, si se requiere realizar algún cambio o mejora en un futuro, se pueda hacer de una manera rápida y eficaz.

5.7. Ajustes finales.

En esta última etapa ha de realizarse una revisión general, donde tomando en consideración las opiniones del publico de prueba se analizarán su viabilidad y pertinencia, para contemplar su adaptación al proyecto como recomendaciones a una futura versión.

4. Desarrollo del proyecto

4.1 Determinación de requerimientos

En búsqueda de cumplir los objetivos del proyecto fue definido el listado de requerimientos funcionales y no funcionales que deben ser cumplidos por el aplicativo, llegando a estos luego del análisis de las necesidades de la comunidad universitaria expuestos en la justificación del presente documento, a continuación, se mencionan y describen estos requerimientos encontrados.

4.1.1 Requerimientos funcionales

Se describen a continuación las funcionalidades que debe poseer el aplicativo, haciendo mención a su identificador, nombre, prioridad y su respectiva descripción.

Tabla 1. *Requerimiento funcional: Registrar usuario*

Requerimiento funcional	
Requerimiento	RF01
Nombre	Registrar usuario
Prioridad	Alta
Descripción	Los usuarios deberán poder registrarse en el aplicativo suministrando su nombre, numero de celular, correo electrónico, ciudad de residencia y contraseña.

Tabla 2. *Requerimiento funcional: Iniciar sesión*

Requerimiento funcional	
Requerimiento	RF02
Nombre	Iniciar sesión
Prioridad	Alta
Descripción	Los usuarios ya registrados en el aplicativo deberán poder iniciar sesión en una pantalla de login, suministrando su correo electrónico y respectiva contraseña.

Tabla 3. *Requerimiento funcional: Recuperar contraseña*

Requerimiento funcional	
Requerimiento	RF03
Nombre	Recuperar contraseña
Prioridad	Alta
Descripción	Los usuarios ya registrados deberán poder recuperar su contraseña de inicio de sesión en caso de haberla olvidado, esto mediante el envío de un código de confirmación OTP a su correo electrónico.

Tabla 4. *Requerimiento funcional: Cambiar contraseña*

Requerimiento funcional	
Requerimiento	RF04
Nombre	Cambiar contraseña
Prioridad	Alta
Descripción	Los usuarios deberán poder modificar la contraseña de su cuenta proporcionando su contraseña actual junto con su nueva contraseña.

Tabla 5. *Requerimiento funcional: Modificar datos personales*

Requerimiento funcional	
Requerimiento	RF05
Nombre	Modificar datos personales
Prioridad	Media
Descripción	Los usuarios deberán poder modificar sus datos registrados en el aplicativo, su nombre, numero celular, correo electrónico y ciudad de residencia.

Tabla 6. *Requerimiento funcional: Cerrar sesión*

Requerimiento funcional	
Requerimiento	RF06
Nombre	Cerrar sesión
Prioridad	Alta
Descripción	Los usuarios deberán poder cerrar sesión en el aplicativo.

Tabla 7. *Requerimiento funcional: Crear anuncios*

Requerimiento funcional	
Requerimiento	RF07
Nombre	Crear anuncios
Prioridad	Alta
Descripción	Los usuarios deberán poder crear nuevos anuncios que serán almacenados en el aplicativo suministrando el título del anuncio, una descripción del anuncio, al menos una imagen referente al anuncio y la categoría a la cual este pertenece.

Tabla 8. *Requerimiento funcional: Visualizar anuncios*

Requerimiento funcional	
Requerimiento	RF08
Nombre	Visualizar anuncios
Prioridad	Alta
Descripción	Los usuarios deberán poder visualizar los diferentes anuncios publicados en el aplicativo junto con su información correspondiente.

Tabla 9. *Requerimiento funcional: Valorar anuncio*

Requerimiento funcional	
Requerimiento	RF09
Nombre	Valorar anuncio
Prioridad	Media
Descripción	Los usuarios deberán poder valorar los anuncios contenidos en el aplicativo de forma positiva, negativa o neutral.

Tabla 10. *Requerimiento funcional: Filtrar anuncios por categoría*

Requerimiento funcional	
Requerimiento	RF10
Nombre	Filtrar por categoría
Prioridad	Media
Descripción	Los usuarios deberán poder filtrar los anuncios contenidos en el aplicativo según a la categoría a la cual estos pertenecen.

Tabla 11. *Requerimiento funcional: Filtrar anuncios por relevancia*

Requerimiento funcional	
Requerimiento	RF11
Nombre	Filtrar por relevancia
Prioridad	Media
Descripción	Los usuarios deberán poder filtrar los anuncios contenidos en el aplicativo según la cantidad de votos positivos que estos posean.

Tabla 12. *Requerimiento funcional: Filtrar anuncios por antigüedad*

Requerimiento funcional	
Requerimiento	RF12
Nombre	Filtrar anuncios por antigüedad
Prioridad	Media
Descripción	Los usuarios deberán poder filtrar los anuncios contenidos en el aplicativo según la antigüedad respecto a su fecha de publicación.

Tabla 13. *Requerimiento funcional: Editar información del anuncio*

Requerimiento funcional	
Requerimiento	RF13
Nombre	Editar información del anuncio
Prioridad	Media
Descripción	Los usuarios deberán poder editar la información que compone los anuncios de su autoría.

Tabla 14. *Requerimiento funcional: Eliminar anuncio*

Requerimiento funcional	
Requerimiento	RF14
Nombre	Eliminar anuncio
Prioridad	Alta
Descripción	Los usuarios deberán poder eliminar los anuncios de su autoría.

Tabla 15. *Requerimiento funcional: Previsualizaciones de anuncios*

Requerimiento funcional	
Requerimiento	RF15
Nombre	Previsualizaciones de anuncios
Prioridad	Media
Descripción	Cada anuncio deberá tener una previsualización correspondiente a él, donde esta esté compuesta por el título del anuncio, su imagen principal, una marca de color que indique la categoría a la cual pertenece y el acceso a la visualización completa del anuncio.

Tabla 16. *Requerimiento funcional: Visualización previsualizaciones*

Requerimiento funcional	
Requerimiento	RF16
Nombre	Visualización previsualizaciones
Prioridad	Media
Descripción	Los usuarios deberán poder visualizar una pantalla principal compuesta por las diferentes previsualizaciones de los anuncios.

Tabla 17. *Requerimiento funcional: Contactar con el anunciante*

Requerimiento funcional	
Requerimiento	RF17
Nombre	Contactar con el anunciante
Prioridad	Alta
Descripción	Los usuarios deberán poder contactar con los autores de los diferentes anuncios del aplicativo, esto a través de un vínculo directo a un chat de la aplicación externa Whatsapp.

Tabla 18. *Requerimiento funcional: Perfil personal*

Requerimiento funcional	
Requerimiento	RF18
Nombre	Perfil personal
Prioridad	Alta
Descripción	Los usuarios deberán tener un perfil personal, compuesto por su nombre del usuario, imagen de perfil, una descripción ingresada por el usuario sobre su perfil y sus anuncios publicados.

Tabla 19. *Requerimiento funcional: Editar descripción del perfil de usuario*

Requerimiento funcional	
Requerimiento	RF19
Nombre	Editar descripción del perfil de usuario
Prioridad	Media
Descripción	Los usuarios deberán poder editar la descripción de su perfil de usuario.

Tabla 20. *Requerimiento funcional: Cambiar imagen de perfil*

Requerimiento funcional	
Requerimiento	RF20
Nombre	Cambiar imagen de perfil
Prioridad	Media
Descripción	Los usuarios deberán poder cambiar su imagen de perfil de usuario.

Tabla 21. *Requerimiento funcional: Visualizar perfil*

Requerimiento funcional	
Requerimiento	RF21
Nombre	Visualizar perfil
Prioridad	Alta
Descripción	Los usuarios deberán poder visualizar su perfil personal así como también los perfiles de los diferentes anunciantes del aplicativo.

Tabla 22. *Requerimiento funcional: Retornar a la vista principal*

Requerimiento funcional	
Requerimiento	RF22
Nombre	Retornar a la vista principal
Prioridad	Baja
Descripción	Los usuarios deberán poder retornar a la vista principal de anuncios cuando estos lo deseen.

Tabla 23. *Requerimiento funcional: Retornar a la pantalla anterior*

Requerimiento funcional	
Requerimiento	RF23
Nombre	Retornar a la pantalla anterior
Prioridad	Alta
Descripción	Los usuarios deberán tener siempre a su disposición la posibilidad de retornar a la vista anterior de la cual se encuentren.

Tabla 24. *Requerimiento funcional: Búsqueda de anuncios*

Requerimiento funcional	
Requerimiento	RF24
Nombre	Búsqueda de anuncios
Prioridad	Media
Descripción	Los usuarios deberán poder buscar entre los diferentes anuncios contenidos en el aplicativo ingresando en un campo de búsqueda un texto a encontrar.

Tabla 25. *Requerimiento funcional: Búsqueda de anuncios por perfil*

Requerimiento funcional	
Requerimiento	RF25
Nombre	Búsqueda de anuncios por perfil
Prioridad	Baja
Descripción	Los usuarios deberán poder realizar búsquedas de anuncios de un perfil en particular, obteniendo resultados solo del perfil indicado.

Tabla 26. *Requerimiento funcional: Visualizar datos de la app*

Requerimiento funcional	
Requerimiento	RF26
Nombre	Visualizar datos de la app
Prioridad	Baja
Descripción	Los usuarios deberán poder visualizar datos de la app como su versión o el nombre de sus desarrolladores.

4.1.2 Requerimientos no funcionales

A continuación, se definen los parámetros no funcionales que debe seguir el aplicativo para el correcto cumplimiento de los objetivos propuestos.

Tabla 27. *Requerimiento no funcional: Limitar el alcance la app*

Requerimiento no funcional	
Requerimiento	RNF01
Nombre	Limitar el alcance de al app
Prioridad	Media
Descripción	La aplicación está pensada para la comunidad universitaria residente dentro del área metropolitana de Bucaramanga, por lo que el público objetivo debe ser los habitantes de los municipios de Bucaramanga, Floridablanca, Girón, Piedecuesta y Lebrija.

Tabla 28. *Requerimiento no funcional: Interface amigable*

Requerimiento no funcional	
Requerimiento	RNF02
Nombre	Interface amigable
Prioridad	Alta
Descripción	La aplicación está debe contener un diseño intuitivo y amigable con el usuario, debe ser muy visual y debe contar con redundancias en el acceso a algunas de sus funciones para comodidad del usuario.

Tabla 29. *Requerimiento no funcional: Disponibilidad del aplicativo*

Requerimiento no funcional	
Requerimiento	RNF03
Nombre	Disponibilidad del aplicativo
Prioridad	Alta
Descripción	El aplicativo debe tener una disponibilidad continua, siendo posible acceder a él en cualquier momento del día.

Tabla 30. *Requerimiento no funcional: Integridad de datos*

Requerimiento no funcional	
Requerimiento	RNF04
Nombre	Integridad de datos
Prioridad	Alta
Descripción	El aplicativo no debe admitir la adición de datos no validos dentro la base de datos, validando lo ingresado ya sea por su tipo o existencia, según sea el caso.

Tabla 31. *Requerimiento no funcional: Compatibilidad con Android*

Requerimiento no funcional	
Requerimiento	RNF05
Nombre	Compatibilidad con Android
Prioridad	Alta
Descripción	El aplicativo debe ser compatible y utilizable desde un dispositivo móvil Android con una versión de sistema operativo superior a la 5.0.

Tabla 32. *Requerimiento no funcional: Capacidad del aplicativo*

Requerimiento no funcional	
Requerimiento	RNF06
Nombre	Capacidad del aplicativo
Prioridad	Alta
Descripción	El aplicativo debe contar con la suficiente capacidad de almacenamiento y estabilidad para soportar una cierta cantidad de usuarios.

4.2 Diseño del software

El diseño de software corresponde a las diferentes etapas que anticipan y planifican el desarrollo del software, abarcando desde el diseño lógico y estructural del sistema del aplicativo, pasado por su interpretación y posibles soluciones en una interface gráfica, hasta las diferentes versiones del diseño gráfico que trazaran una hoja de ruta del desarrollo del aplicativo.

4.2.1 Casos de uso

Los casos de uso buscan especificar la comunicación, el comportamiento y las diferentes formas que tienen los distintos actores involucrados, a la hora de usar el sistema. Se puede visualizar en la ilustración 6 los casos de uso esperados en aplicativo móvil.

Ilustración 6. *Casos de uso del aplicativo, autenticación de usuarios*

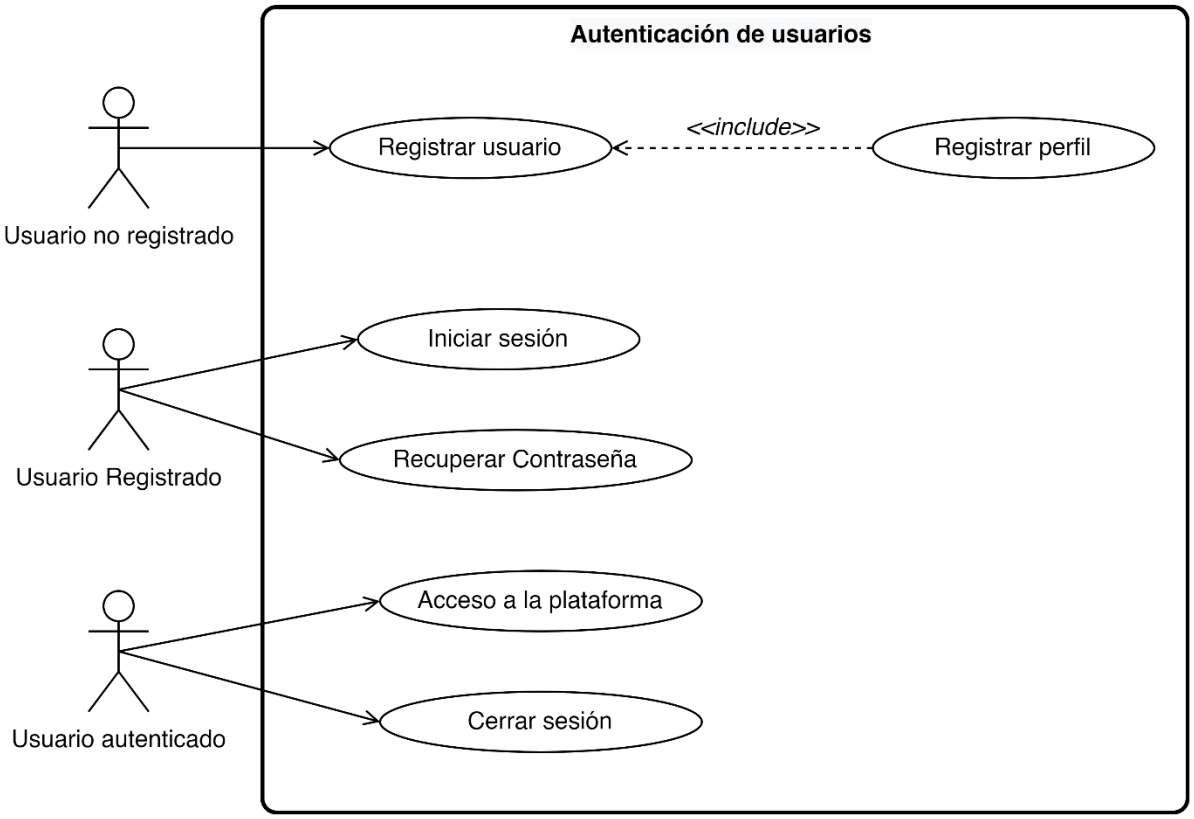


Ilustración 7. *Casos de uso del aplicativo, gestión del perfil*

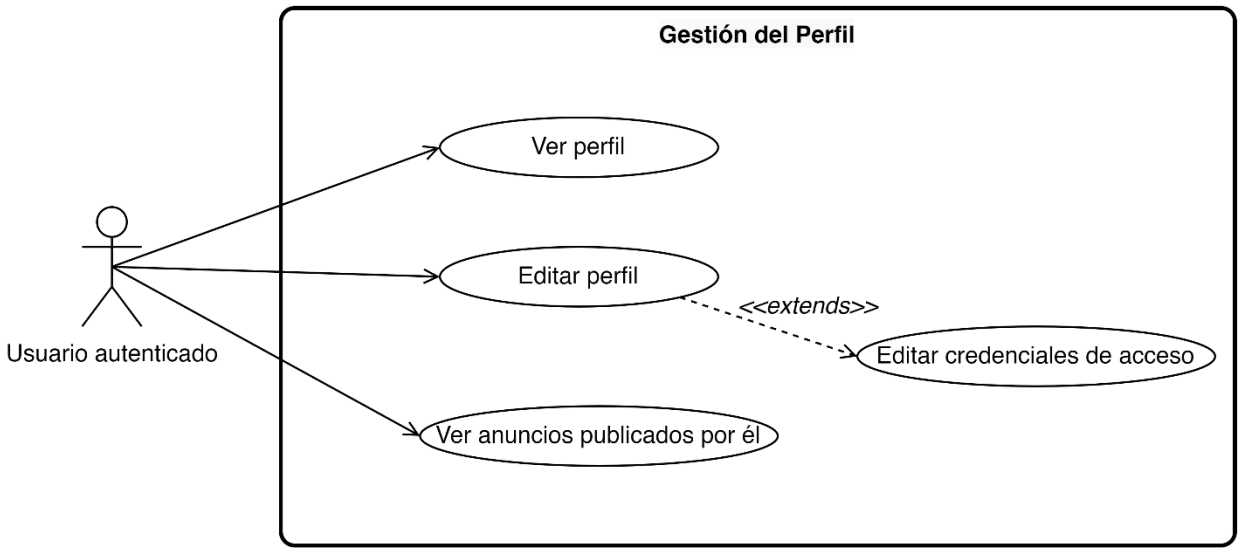
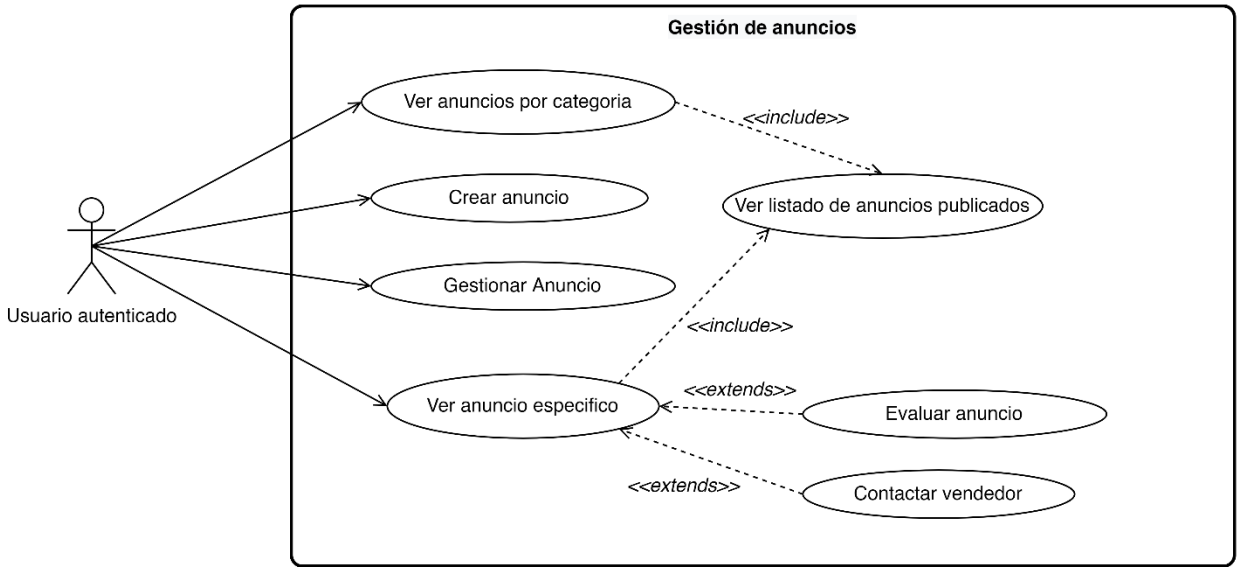


Ilustración 8. *Casos de uso del aplicativo, gestión de anuncios*



Serian tres el número de actores que conforman y pueden interactuar con el sistema, estos siendo el usuario que usa el aplicativo por primera vez o no está registrado, estando dentro de sus capacidades la posibilidad de registrar una cuenta de usuario, el segundo actor del sistema seria el usuario que ya teniendo una cuenta registrada aún no ha iniciado sesión permitiéndole el sistema a este actor la acción de iniciar sesión o recuperar su contraseña, y finalmente el tercer actor sería un usuario con una cuenta valida, registrada y con un inicio de sesión correctamente autenticado, donde este actor puede interactuar con las diferentes características y funcionalidades del sistema, expuestos en las ilustraciones 6, 7 y 8.

4.2.2 Tarjetas CRC

Particularmente útiles al momento de realizar lluvia de ideas, las tarjetas CRC permiten obtener una buena visión general del diseño del proyecto a partir de los casos de uso, su estructura está repartida en tres zonas, la superior con el nombre de la clase, la izquierda con las responsabilidades y objetivos a cargo de la clase y la derecha con las clases colaboradoras que

contribuyen a esta clase a cumplir sus responsabilidades, a continuación podemos observar en las tablas 33 a la 38 las tarjetas CRC correspondientes al presente proyecto.

Tabla 33. *Tarjeta CRC Usuario*

Clase: Usuario	
Responsabilidades	**Colaboradores**
● Registrar en la aplicación ● Ingresar en la aplicación ● Recuperar contraseña de ingreso ● Cambiar contraseña de ingreso ● Cerrar sesión dentro de la aplicación	

Tabla 34. *Tarjeta CRC Anuncio*

Clase: Anuncio	
Responsabilidades	**Colaboradores**
● Mostrar un listado de anuncios ● Mostrar un listado de anuncios filtrado por categorías ● Mostrar un listado de anuncios publicados por un perfil de usuario ● Mostrar un anuncio específico ● Buscar anuncios por palabras claves o características específicas ● Crear un anuncio ● Actualizar un anuncio específico ● Eliminar un anuncio específico ● Cargar imágenes del anuncio	● Categoría ● Perfil ● Voto ● Adjunto

Tabla 35. *Tarjeta CRC Categoría*

Clase: Categoría	
Responsabilidades	**Colaboradores**
● Mostrar las categorías	

Tabla 36. *Tarjeta CRC Perfil*

Clase: Perfil	
Responsabilidades	**Colaboradores**
● Crear un perfil de usuario ● Mostrar un perfil ● Actualizar un perfil ● Calcular calificación de un perfil basado en sus anuncios ● Cargar adjunto de la foto de perfil ● Registrar voto en un anuncio específico	● Usuario ● Anuncio ● Adjunto ● Voto

Tabla 37. *Tarjeta CRC Adjunto*

Clase: Adjunto	
Responsabilidades	**Colaboradores**
● Crear un adjunto ● Eliminar un adjunto ● Mostrar los adjuntos	● Perfil

Tabla 38. *Tarjeta CRC Voto*

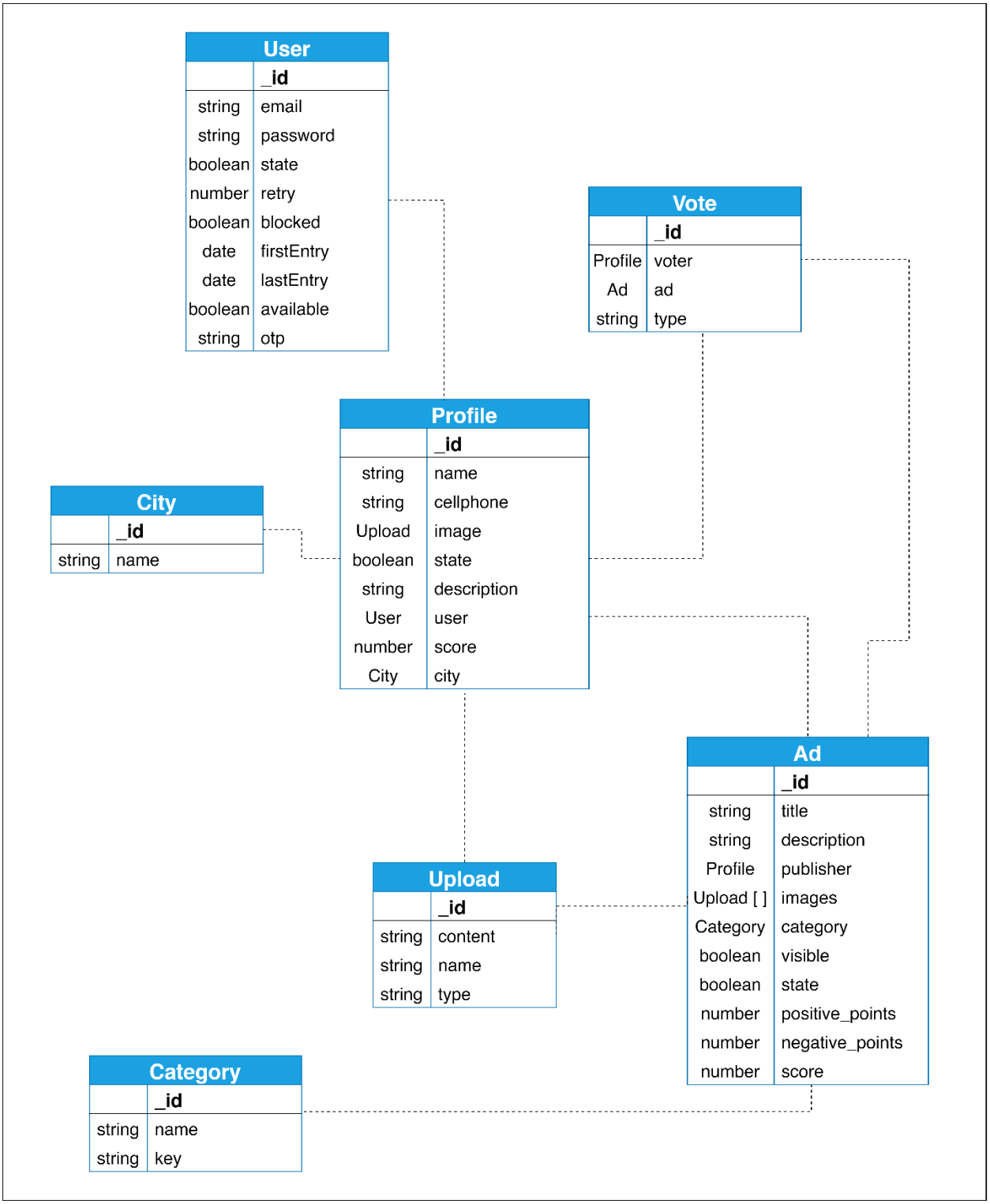
Clase: Voto	
Responsabilidades	**Colaboradores**
● Crear un voto positivo o negativo ● Eliminar un voto positivo o negativo	● Anuncio ● Perfil

4.2.3 Modelo base de datos

El presente es el modelo de base de datos del aplicativo el cual cuenta con un total de siete tablas y contiene los espacios para almacenar toda la información requerida para el funcionamiento del aplicativo.

4.2.3.1 Diagrama modelo base de datos

Ilustración 9. *Modelo base de datos*



4.2.3.2 Descripción del modelo

A continuación, se dará una descripción de cada una de las tablas del modelo de base de datos empleado en el sistema.

Tabla User. Esta tabla tiene como objetivo almacenar los datos de acceso del usuario, tales como su correo electrónico, contraseña u hora de ultimo acceso al sistema, datos con una alta importancia para el funcionamiento del sistema.

Tabla Profile. Esta tabla tiene como objetivo almacenar los datos propios del perfil de cada uno de los usuarios, conteniendo datos como el nombre del usuario, la referencia a la ciudad de residencia, su número de contacto a Whatsapp o la descripccion propia del perfil.

Tabla City. Esta tabla contiene el listado de las ciudades y referencias donde está disponible el aplicativo.

Tabla Ad. Esta tabla tiene como misión almacenar los datos relacionados a cada uno de los anuncios, estos incluyen elementos como el título del anuncio, los puntos positivos y negativos recibidos en él, la referencia al perfil del anunciante o la descripción del anuncio.

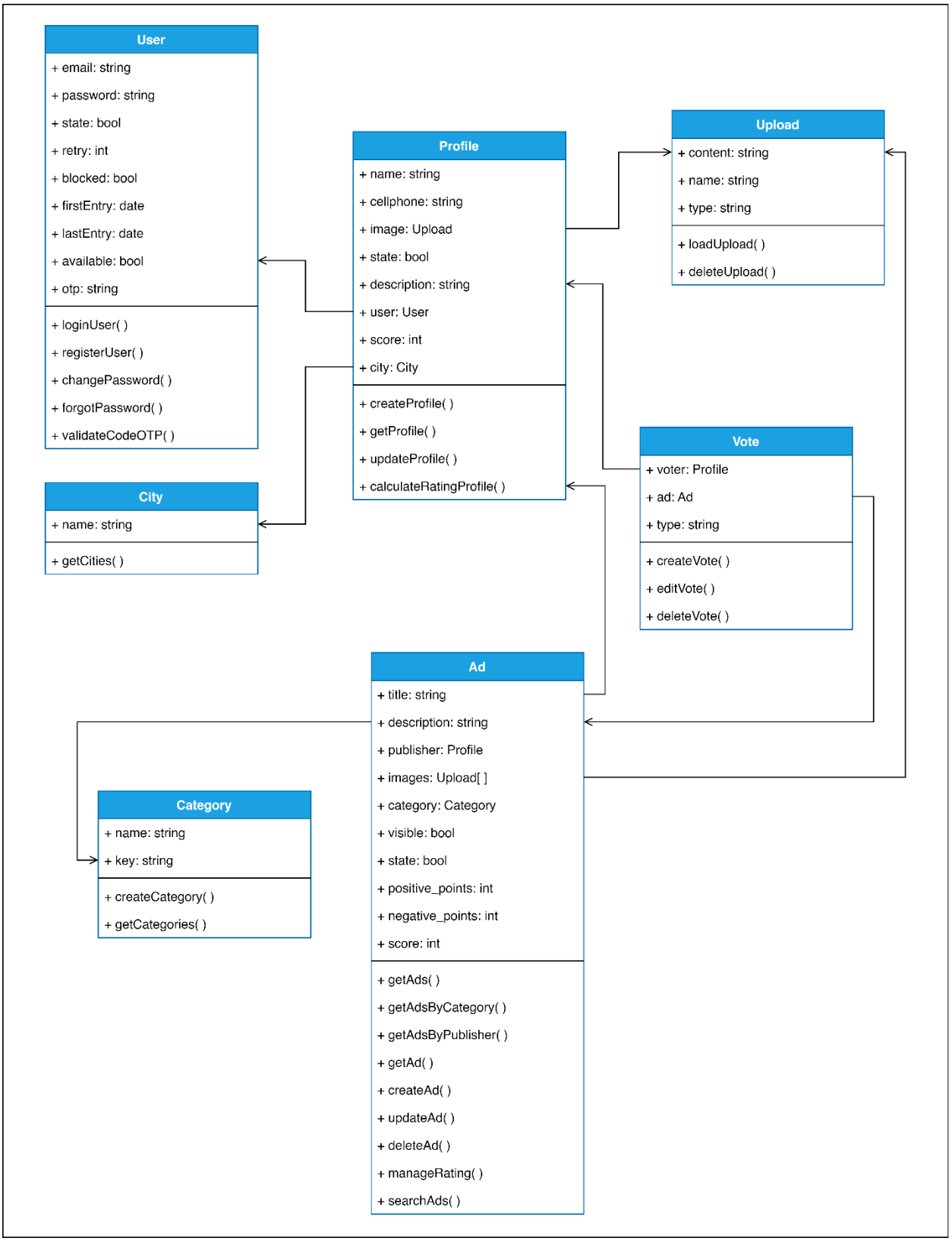
Tabla Vote. Con la ayuda de esta tabla es posible registrar los votos realizados por un perfil a un anuncio, almacenando las referencias del anuncio votado y el perfil que lo votó.

Tabla Upload. Esta tabla contiene el listado de imágenes que puede contener un anuncio, relacionando el perfil de un usuario con sus anuncios.

Tabla Category. Esta tabla contiene el listado de categorías disponibles dentro de la aplicación, otorgándole a un anuncio el acceso a las categorías a cuál pertenece.

4.2.4 Diagrama de clases

Ilustración 10. *Diagrama de clases*

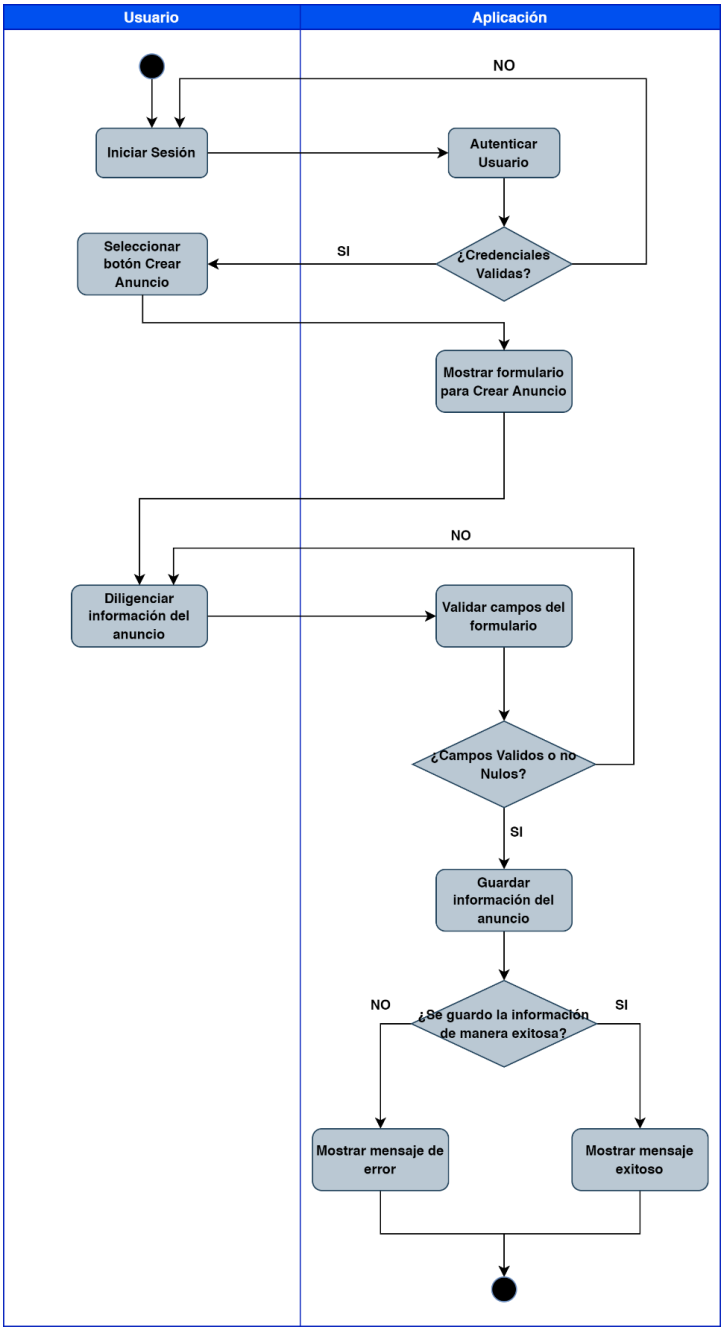


4.2.5 Diagrama de actividades

A continuación, se desarrollan los diagramas de actividades de las funcionalidades más relevantes del aplicativo y a su vez las acciones que podrán realizar actores en el sistema.

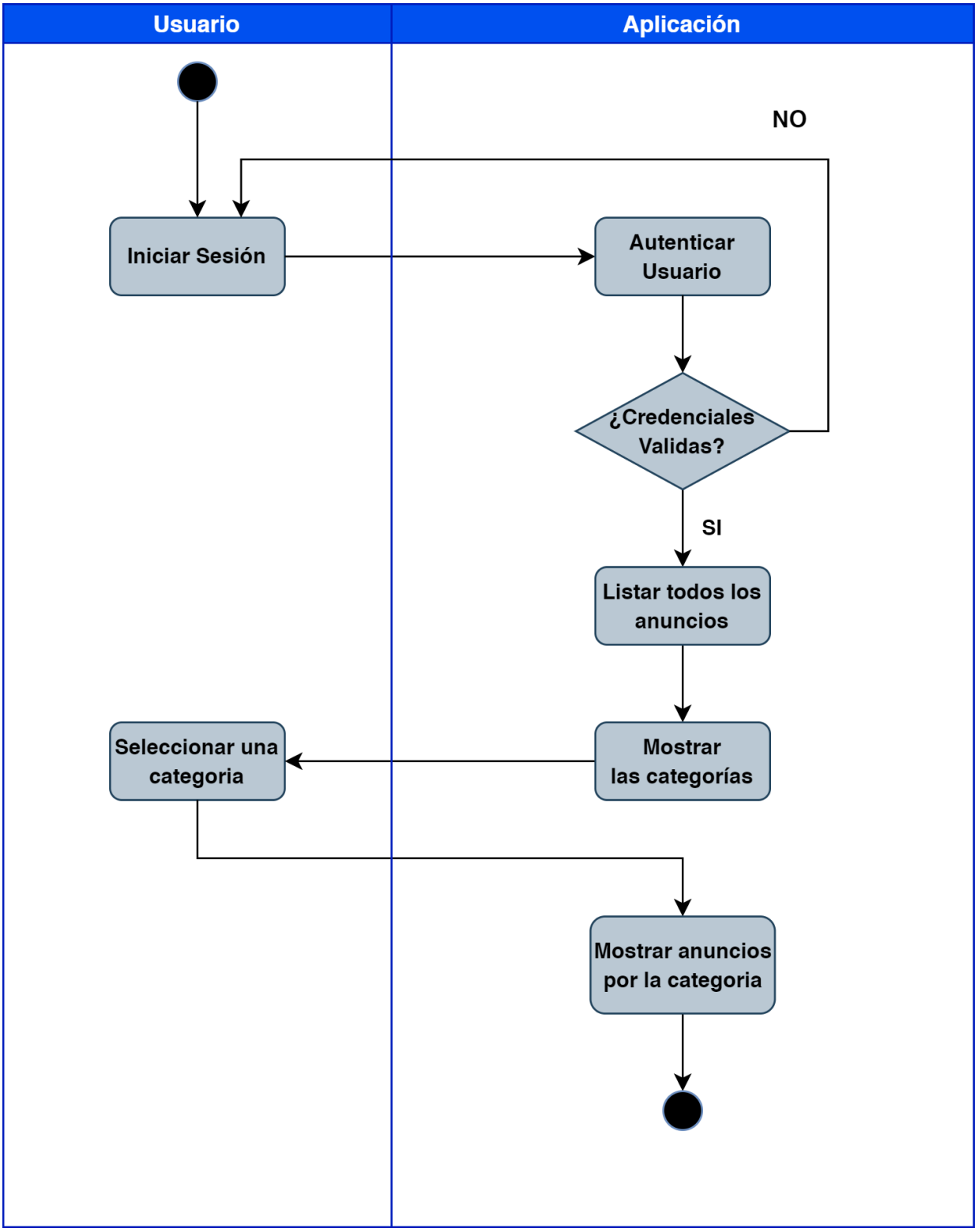
4.2.5.1 Diagrama de actividades, creación de anuncio

Ilustración 11. *Diagrama de actividades, Crear anuncio*



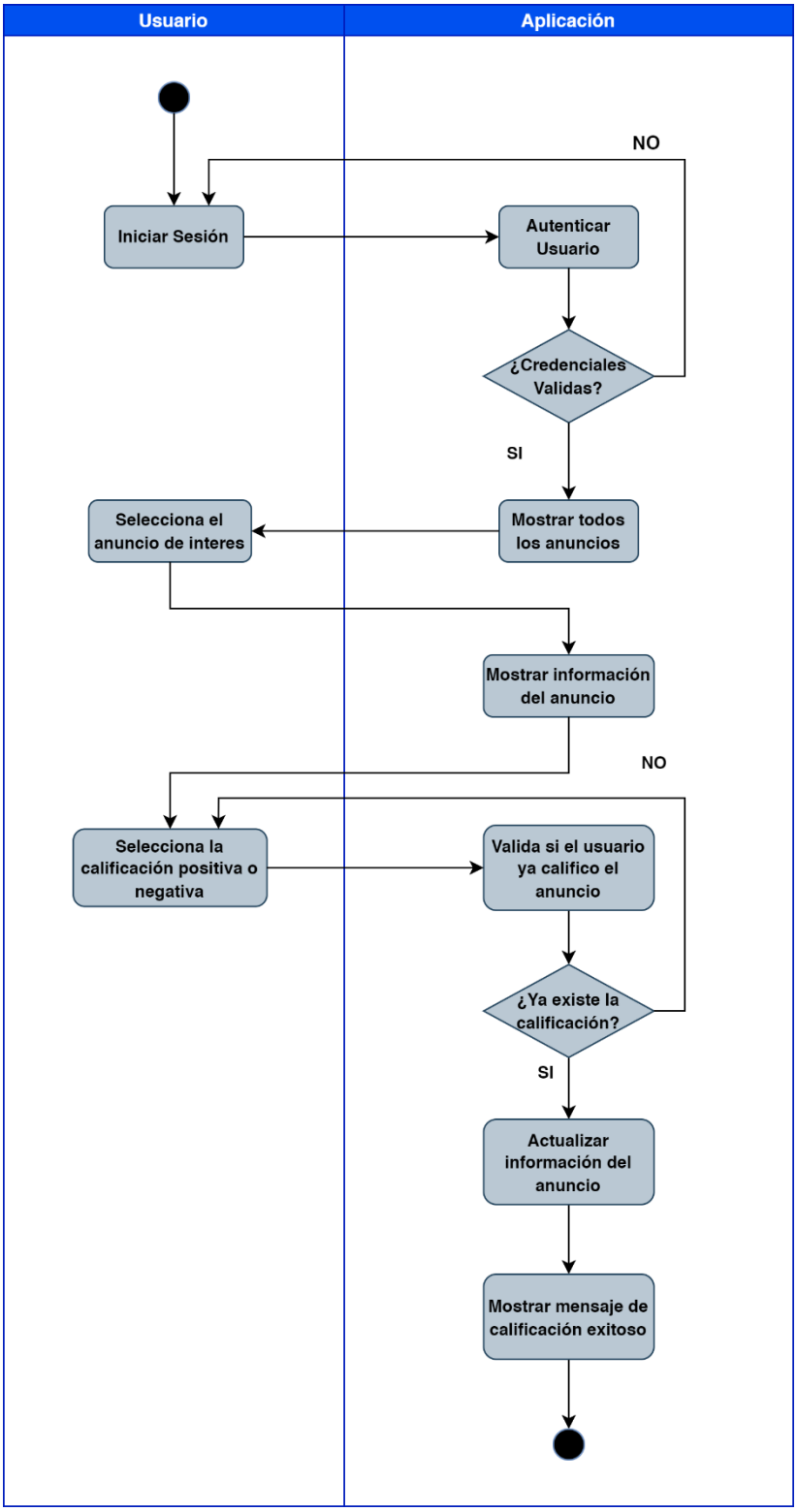
4.2.5.2 Diagrama de actividades, mostrar anuncio por categoría

Ilustración 12. *Diagrama de actividades, mostar anuncio por categoría*



4.2.5.3 Diagrama de actividades, calificar un anuncio

Ilustración 13. *Diagrama de actividades, calificar un anuncio*



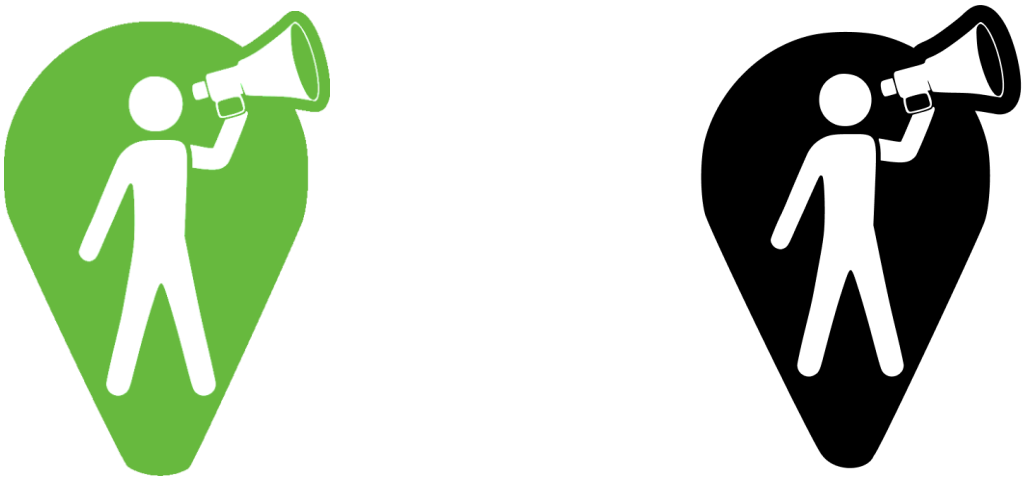
4.3 Diseño de interfaz grafica

4.3.1 Generalidades

Logo y nombre del aplicativo

Buscando plasmar y simbolizar la misión y el objetivo general del proyecto se elaboró el siguiente logo del aplicativo, siguiendo como pautas de diseño que este sea claro, fácilmente visible, y que pueda ser reconocido con facilidad incluso con escalas de tamaño reducidas.

Ilustración 14. *Logo Aplicativo móvil a color y blanco y negro*



Se buscó el desarrollo de un logo minimalista con el objetivo de que este sea amigable y fácilmente entendible por el usuario, además se apuntó a la incorporación de dos conceptos principales en la idea del logo del aplicativo, el concepto de dejar en claro a los usuarios la utilidad del aplicativo, con la que pueden dar a conocer sus iniciativas y demás anuncios a la comunidad, y inclusión del concepto de lugar, esto con el fin de dar a entender al usuario que el aplicativo tendrá en cuenta el lugar donde se encuentra el usuario, en este caso, el campus universitario de la universidad industrial de Santander, lugar hacia el cual está orientado el proyecto.

Ilustración 15. *Nombre aplicativo móvil a color y blanco y negro*

UISAds UISAds

De forma equivalente a como se diseñó el logo del aplicativo se definió también el nombre del mismo, buscando reflejar las mismas ideas del logo, dejando en claro el lugar y público objetivo del proyecto además de la capacidad del aplicativo de compartir anuncios a la comunidad.

Colores del aplicativo

En concordancia con las ideas plasmadas en el logo y nombre del aplicativo se hace uso de los colores institucionales de la universidad, delimitando la población y comunidad objetivo del presente proyecto.

Tabla 39. *Colores aplicativo móvil*

Color	RGB	HEX
	R: 103 G: 185 B: 62	# 67B93E
	R: 255 G: 185 B: 255	# FFFFFF

Categorías dentro del aplicativo

Como parte de la estructura dentro de la plataforma del aplicativo los anuncios serán organizados por los anunciantes en diferentes categorías según el tipo de información que este contenga, esto para darle mayor orden, estructura y visibilidad a cada una de las publicaciones hechas por ellos, en armonía con ello, cada categoría tendrá un icono y color característico, buscando que estas sean fácilmente identificables y amigables para el usuario.

Tabla 40. *Categorías de anuncios aplicativo móvil*

Nombre Categoría	Descripción	Color	Color RGB	Icono
Variados	Categoría principal y punto central del aplicativo, todos los anuncios publicados en la plataforma pertenecen a esta categoría		R: 103 G: 185 B: 62	
Alimentos	Categoría destinada para las publicaciones que quieran dar a conocer contenido que incluya productos alimenticios o servicios de alimentación.		R: 248 G: 106 B: 0	
Deportes	Categoría creada para mantener las publicaciones relacionadas al mundo del deporte y la actividad física.		R: 118 G: 113 B: 113	
Ofertas de empleo	Categoría hecha para ser un espacio donde los usuarios puedan publicar/buscar ofertas de empleo, o dar a conocer programas de prácticas laborales o pasantías empresariales.		R: 65 G: 113 B: 156	

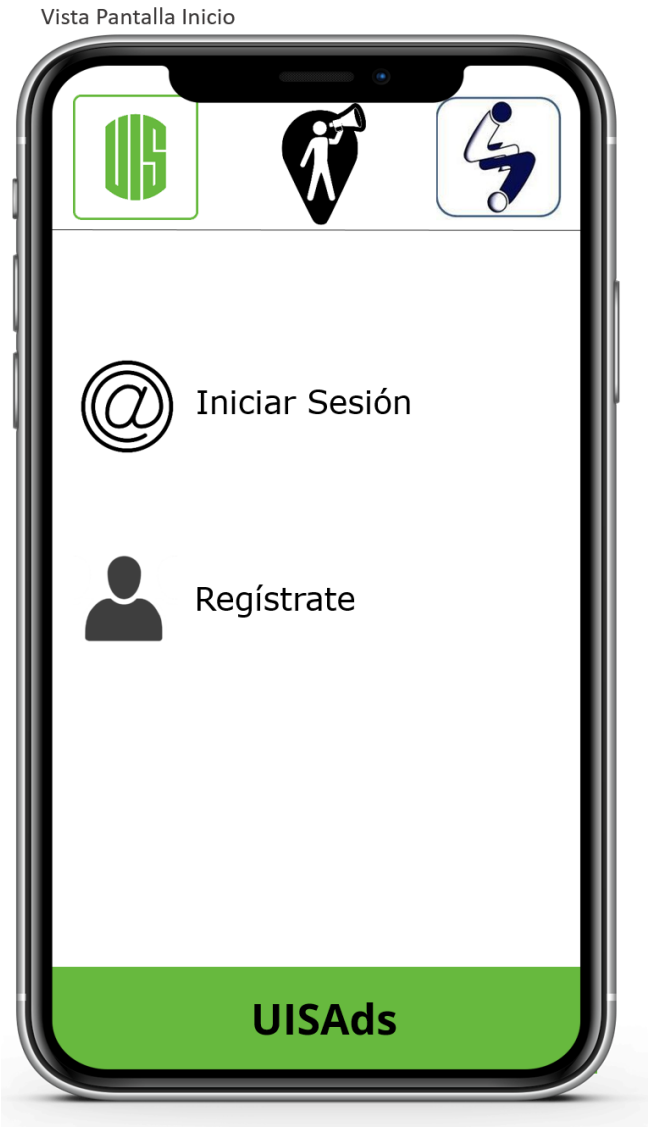
Servicios	Categoría creada para que los usuarios puedan publicar anuncios que estén destinados al ofrecimiento de servicios varios, tal y como lo puede ser el de reparación, traducciones, transporte, etc.		R: 128 G: 64 B: 0	
Vestuario	Categoría destinada para contener los anuncios que busquen dar a conocer artículos de ropa, calzado, moda y vestimenta en general.		R: 246 G: 8 B: 2	
Arte y literatura	Categoría para incluir anuncios cuyo propósito sea promocionar piezas artísticas o de literatura de propiedad del usuario.		R: 173 G: 49 G: 173	
Arrendamientos y alquiler	Categoría dispuesta para la publicación de anuncios que busquen dar a conocer espacios en arriendo o artículos en alquiler.		R: 255 G: 217 G: 102	
Tecnología	Categoría destinada para la publicación de artículos electrónicos, nuevos o de segunda mano.		R: 84 G: 130 G: 53	
Tutorías y aprendizaje	Categoría dispuesta para contener anuncios cuyo propósito sea el de ofrecer servicios de tutorías, clases y enseñanza en general.		R: 44 G: 178 G: 178	

Los iconos empleados en el aplicativo son de uso libre y de dominio público, cumpliendo la ley de uso justo DMCA y siendo obtenidos de la web freepng.es.

4.3.2 Vistas diseño uno

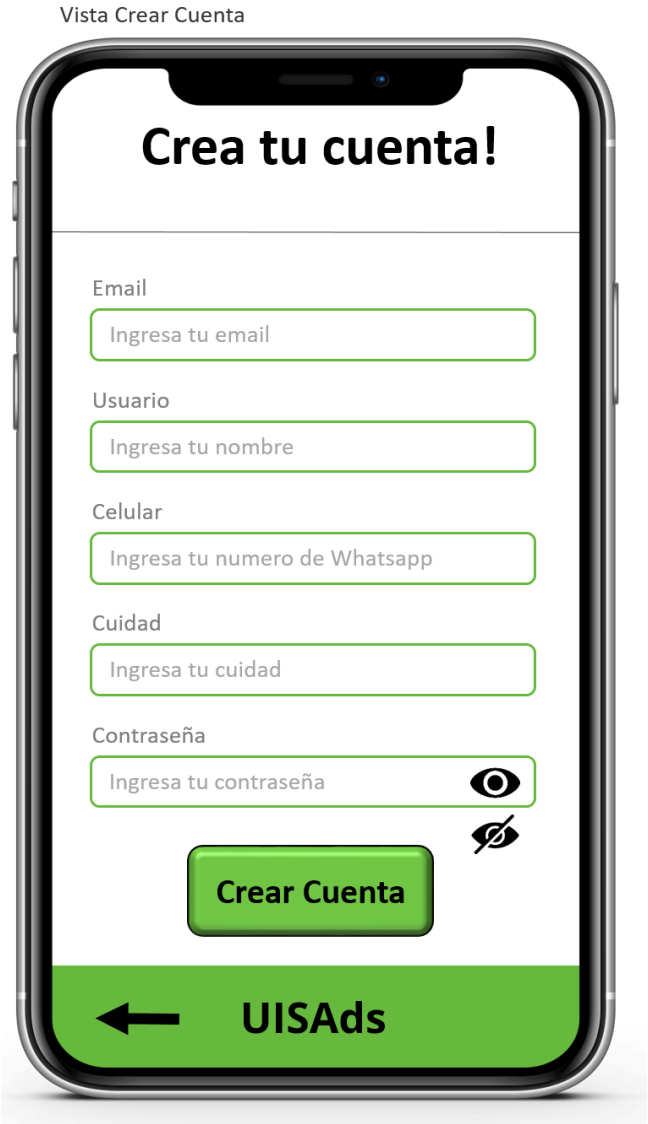
De forma inicial se elaboró un primer diseño, buscando el completo cumplimiento de todos los objetivos, requerimientos funcionales y no funcionales del proyecto, a su vez este primer diseño, busca marcar las pautas de navegación y estructura de la interface del aplicativo en el desarrollo final, en el marco de este proyecto este diseño será llamado diseño uno.

Ilustración 16. *Vista pantalla de inicio*



La vista presente en lustración 16, conformaría la pantalla principal, esta busca dar acceso al cumplimiento de los requerimientos funcionales RF01 y RF02, en ella el usuario debe poder elegir registrarse en el aplicativo si aún no lo ha realizado, o iniciar sesión en este si él ya ha creado una cuenta con anterioridad.

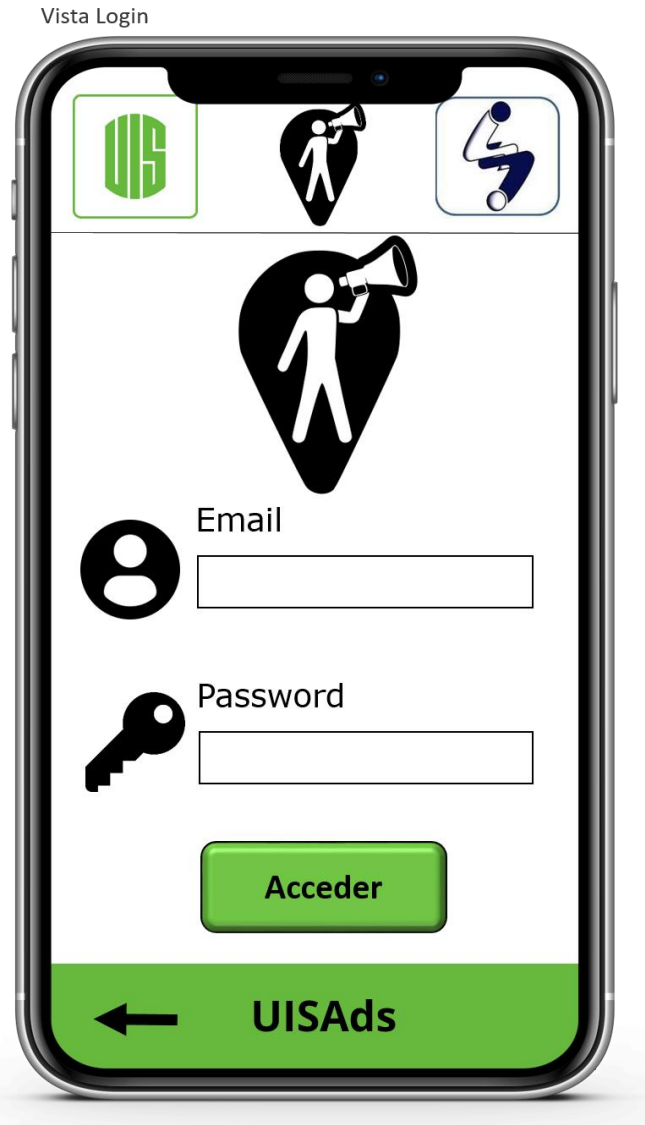
Ilustración 17. *Vista crear cuenta*



Si el usuario elige en la vista de inicio la opción de crear una cuenta, accedería a la presente vista expuesta en la ilustración 17, en la cual el usuario debe poder suministrar sus datos de

contacto, además de una contraseña de inicio de sesión, dando cumplimiento al requerimiento funcional RF01, en esta vista el usuario también deberá poder retornar a la vista anterior.

Ilustración 18. *Vista inicio de sesión*



Si el usuario accede a la opción de iniciar sesión en la vista de inicio, accedería a esta pantalla, ver ilustración 18, la cual busca dar cumplimiento al requerimiento funcional RF01, en donde, el usuario debe poder iniciar sesión, empleando su correo electrónico registrado junto con su respectiva contraseña.

Ilustración 19. *Vista categoría 1*

Al iniciar sesión en la anterior vista, ver ilustración 18, el usuario accederá a la categoría principal del aplicativo, presente en la ilustración 19, en la cual podrá previsualizar todos los anuncios publicados, dando cumplimiento al requisito funcional RF16, así como también deberá encontrar diferentes opciones de navegación, ya sea desplazándose verticalmente para ver más previsualizaciones, tocando alguna previsualización para ver la información completa del anuncio, accediendo a la opción de creación de un anuncio, donde el usuario podrá aportar a la plataforma

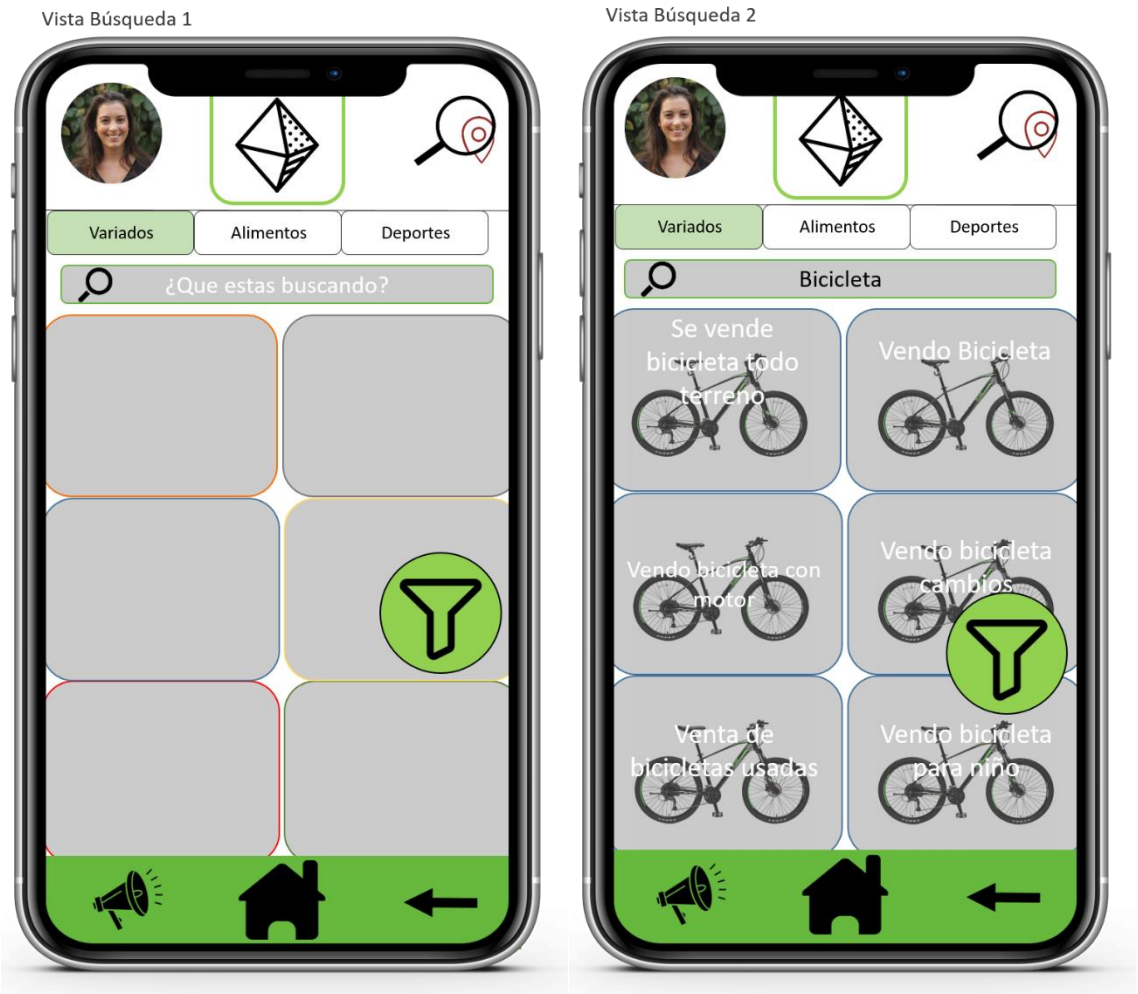
un anuncio nuevo, ingresando a la búsqueda de anuncios dentro de la plataforma, o desplazarse a un menú lateral deslizando hacia la derecha, además de poder visualizar una barra de navegación inferior donde el usuario tendrá un botón para retornar rápidamente a esta misma pantalla principal, dando cumplimiento al requisito funcional RF22 y un botón de retorno a la pantalla anterior según sea el caso, dando cumplimiento al requerimiento funcional RF23.

También en las vistas de categorías, las previsualizaciones deberán poseer un borde indicativo de color, que señalará y corresponderá a la categoría a la cual pertenece el anuncio.

Ilustración 20. *Vista categoría 2*

Al desplazarse entre las categorías de la aplicación los anuncios mostrados se deben ver filtrados por la categoría a la cual pertenecen, tal como lo muestra la ilustración 20, esto dando cumplimiento al requisito funcional RF10.

Ilustración 21. *Vista búsqueda*



Desde la vista de categorías se puede acceder a la presente vista, ver ilustración 21, conteniendo el apartado de búsqueda, la cual deberá mostrar una serie de anuncios según lo buscado, dando cumplimiento al requerimiento funcional RF24, esta vista deberá presentar un input, en el cual, el usuario pueda ingresar un texto que se buscará entre los diferentes anuncios almacenados en la plataforma, a su vez el usuario deberá contar con la posibilidad de señalar en cual categoría desea realizar la búsqueda, además de un botón con el cual podrá desplegar una ventana emergente para acceder a filtros adicionales.

Ilustración 22. *Vista filtros de búsqueda*

Los filtros de búsqueda disponibles deberán contener las opciones de filtrado por relevancia y por fecha de publicación, tal como se ve en la ilustración 22, dando cumplimiento a los requisitos funcionales RF11 y RF12 respectivamente, se plantea que de ser señalada la opción de por relevancia, los anuncios mostrados se deberán ordenar por la cantidad de valoraciones positivas que posean, en orden descendente, y si en su lugar es señalada la opción de por fecha de

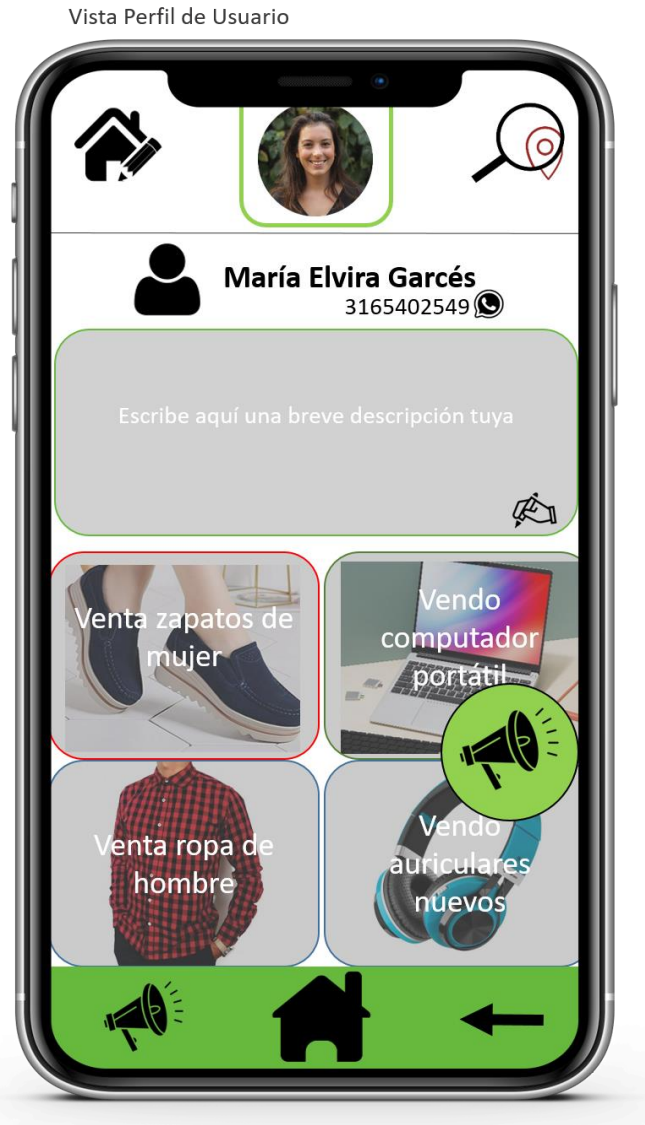
publicación del anuncio, el usuario deberá poder elegir entre un listado fijo de opciones la antigüedad máxima de los anuncios que desea visualizar.

Ilustración 23. *Vista menú lateral*

Desde la vista de categorías se deberá poder desplegar un menú lateral, acción con la se accede a la presente vista, ver ilustración 23, desde la cual se deberá poder cerrar sesión, dando cumplimiento al requisito funcional RF06, se podrá también obtener más detalles sobre la app,

dando cumplimiento al requisito funcional RF26, enviar un mensaje a los desarrolladores o acceder al propio perfil del usuario.

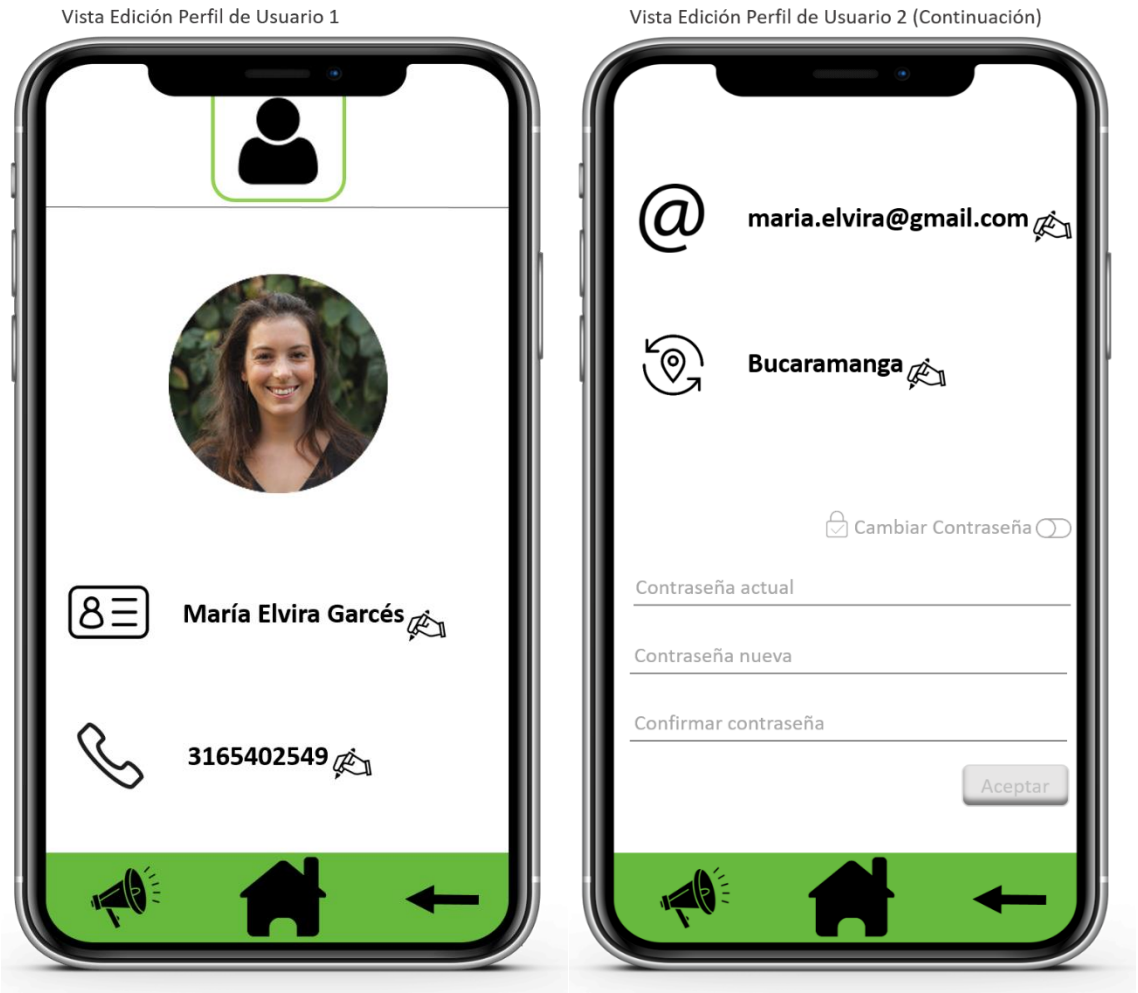
Ilustración 24. *Vista perfil de usuario*



Accediendo a la presente vista, ver ilustración 24, desde el menú lateral, se deberá poder ver el perfil del propio usuario, dando cumplimiento al requisito funcional RF21, este estando compuesto por su imagen de perfil previamente cargada por usuario, el nombre del usuario y su número de contacto, así mismo el usuario deberá disponer de las opciones de edición de su perfil,

y de edición de una descripción para su perfil, además de acceso a búsqueda de anuncios de su autoría, estas últimas, dando cumplimiento a los requisitos funcionales RF19 y RF25 respectivamente, igualmente, en esta vista el usuario deberá poder ver todos sus propios anuncios.

Ilustración 25. *Vista edición datos personales*



Accediendo a la opción de edición de perfil, en la vista perfil de usuario, ver ilustración 24, el usuario deberá encontrarse con esta pantalla, ver ilustración 25, la cual debe ofrecerle la posibilidad de modificar su nombre de usuario, número de teléfono, correo electrónico y ciudad de residencia, así mismo deberá poder acceder a una opción de cambio de contraseña de acceso, dando cumplimiento a los requisitos funcionales RF04 y RF05.

Ilustración 26. *Vista publicación de anuncio*

Si el usuario opta por acceder a la publicación de un anuncio, este deberá encontrarse con la presente vista, ver ilustración 26, dando cumplimiento al requisito funcional RF07, vista que deberá cumplir con el objetivo de desplegar un formulario para la creación de un nuevo anuncio, esta vista debe contener las opciones de selección de categoría, en la cual será publicado el anuncio, a su vez deberá poder cargar un máximo de imágenes desde su dispositivo móvil y un mínimo de una imagen para poder publicar el anuncio, estas imágenes se deberán poder visualizar en la

publicación final del anuncio y cuya primera imagen es obligatoria y hace parte de la previsualización del mismo, además, el usuario deberá definir un título para su anuncio, así como también una descripción del mismo. El usuario también deberá contar con la posibilidad de cancelar la publicación del anuncio y retornar a la pantalla anterior.

Ilustración 27. *Vista anuncio*

El usuario al tocar la previsualización de un anuncio deberá poder acceder a la visualización del anuncio como tal, ver ilustración 27, dando cumplimiento al requisito funcional RF08, desde el cual deberá poder visualizar, el nombre del anuncio, la descripción del anuncio y las imágenes que incorpora el anuncio, además de esto, deberá también contar con la posibilidad de valorar positiva o negativamente el anuncio, dando cumplimiento al requisito funcional RF09, así como también deberá poder acceder, por un botón, a un chat directo de Whastsapp con el anunciante, dando cumplimiento al requisito funcional RF17, e igualmente deberá poder acceder al perfil del anunciante mediante la imagen de perfil del mismo.

Ilustración 28. *Vista perfil anunciante*

La presente vista, ver ilustración 28, posee características muy similares a la vista de perfil de usuario, ver ilustración 24, ya que esta enseñará los datos propios de un perfil, junto con las publicaciones realizadas por este, pero se deberá diferenciar en que en esta vista se deberá encontrar la opción de contacto con el anunciante, para dar cumplimiento al requisito funcional RF17, botón con el cual el usuario que visualiza el perfil pueda abrir con facilidad un chat directo de Whatsapp con el anunciante.

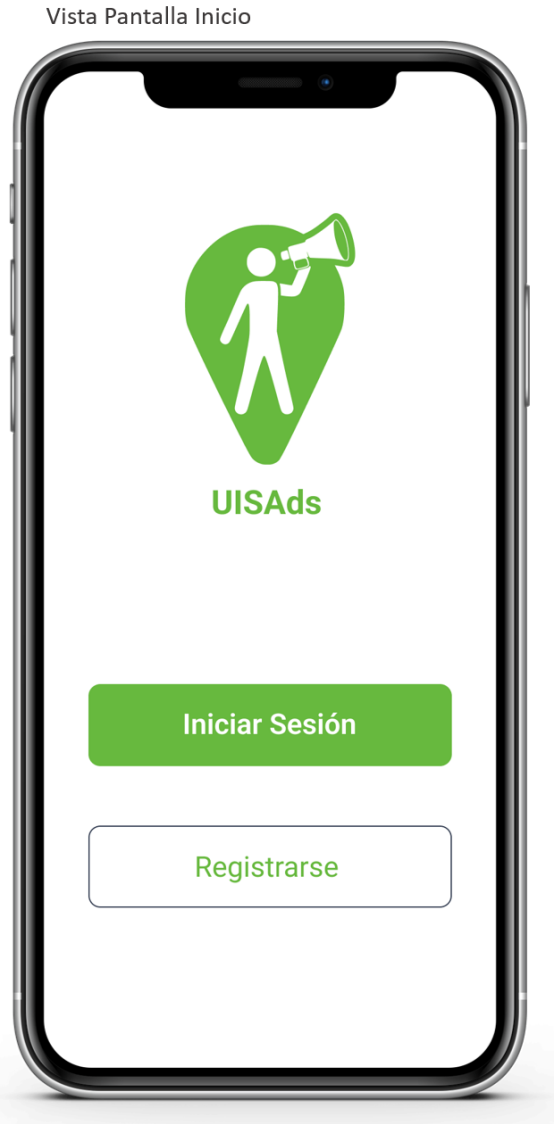
Ilustración 29. *Vista anuncio propio*

Si el usuario está visualizando un anuncio de su propiedad, ver ilustración 29, este deberá contar con la posibilidad de editar la información de su anuncio, esto para dar cumplimiento al requisito funcional RF13, u optar por la eliminación del mismo, dando cumplimiento al requisito funcional RF14, a su vez el usuario, al ser propietario del anuncio, no deberá poder asignar una valoración a su propio anuncio, limitando esta funcionalidad a solo usuarios ajenos a él.

4.3.3 Vistas diseño dos

Tomando como base las pautas definidas en el diseño uno e implementando la estructura de su diseño, se modeló una segunda versión, apuntando está a ser más acorde a la construcción final del aplicativo, para el uso de estas nuevas vistas se hizo uso de la herramienta web *figma.com*.

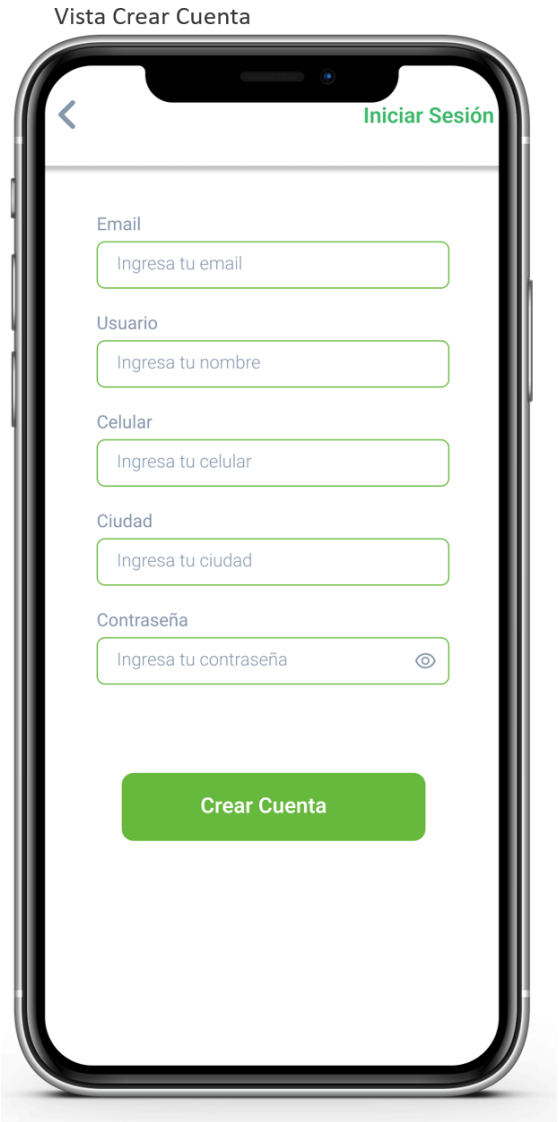
Ilustración 30. *Vista pantalla de inicio V2*



En este segundo diseño, ver ilustración 30, se busca abarcar mejor el cumplimiento de los requerimientos, y se opta por hacer mayor énfasis al requerimiento no funcional RNF02, añadiendo

cambios como un rediseño de interfaces o la modificación del tono del logo del aplicativo, esto buscando ser más consistente con los colores definidos para el aplicativo, se conservan las opciones, funcionalidades y cumplimiento de requerimientos del diseño uno.

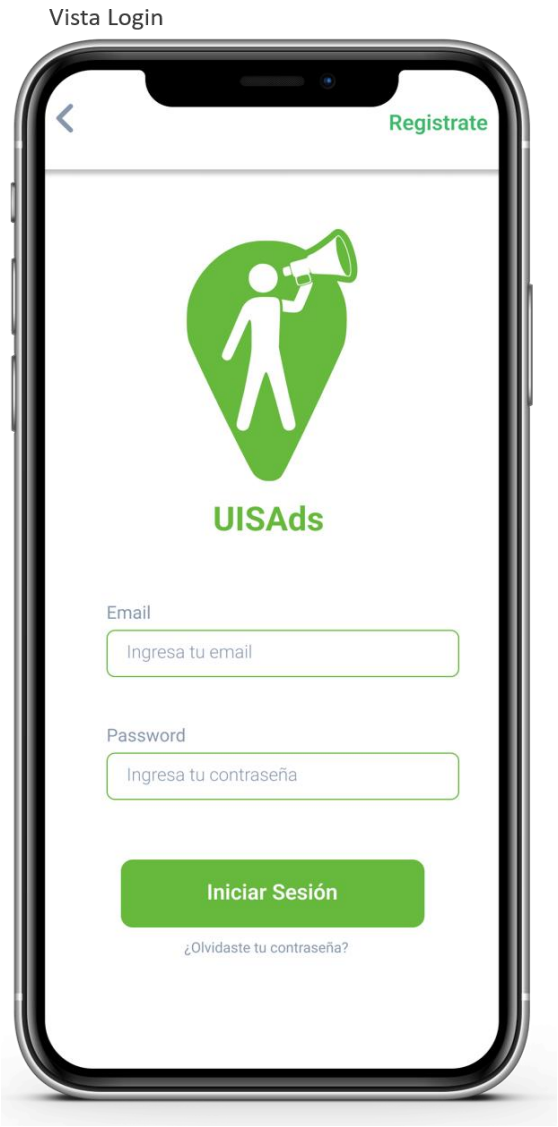
Ilustración 31. *Vista crear cuenta V2*



Con respecto al diseño uno, en la presente vista, ver ilustración 31, se conserva su estructura de diseño y los campos para el registro de un nuevo usuario, se modifica el posicionamiento del

botón de retorno a la vista anterior y se añade una opción para saltar directamente a la pantalla de inicio de sesión.

Ilustración 32. *Vista inicio de sesión V2*



En relación a su vista en el diseño uno, ver ilustración 18, en esta nueva vista, ver ilustración 32, se conserva la visualización del logo del aplicativo, esta vez con el tono verde propio del aplicativo y de nuestra institución, pero a diferencia de su primera versión, en esta se implementa un sistema de recuperación de la cuenta, en caso de que el usuario olvide su clave de

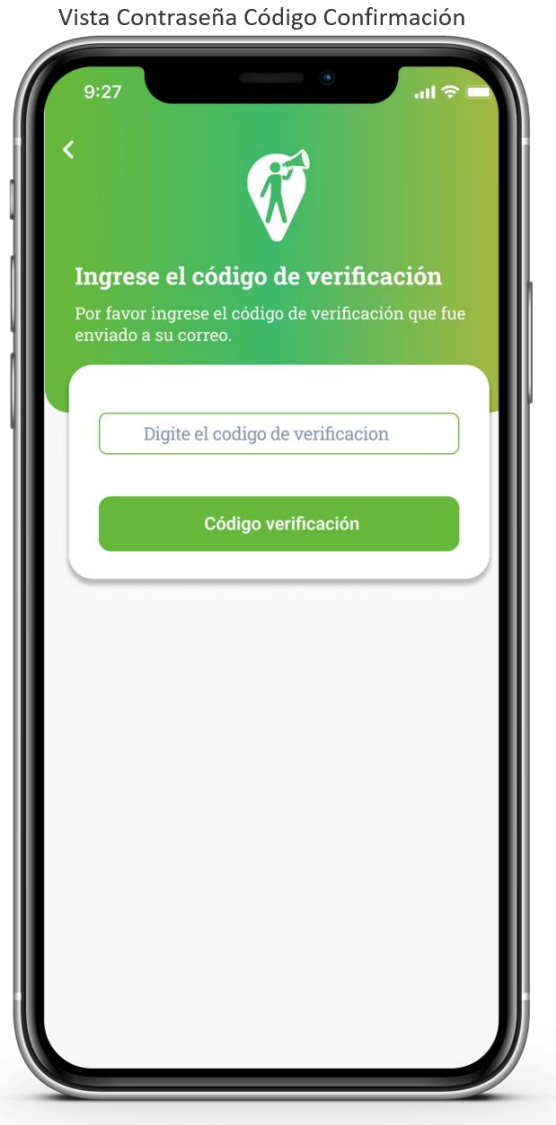
acceso, esto para dar cumplimiento al requerimiento funcional RF03, opción a la cual se puede acceder mediante el texto "¿Olvidaste tu contraseña?" ubicado en la parte inferior de la pantalla, bajo el botón de inicio de sesión, donde dicha opción desplazará al usuario a la pantalla de recuperación de contraseña.

Ilustración 33. *Vista recuperar contraseña*

En relación a la nueva opción de recuperación de contraseña incorporada en este diseño, se crea esta vista, ver ilustración 33, donde el usuario podrá ingresar su correo electrónico de

registro y al cual le será enviado un código de recuperación de la cuenta, donde este podrá recuperar el acceso, cumpliendo el requisito funcional RF03.

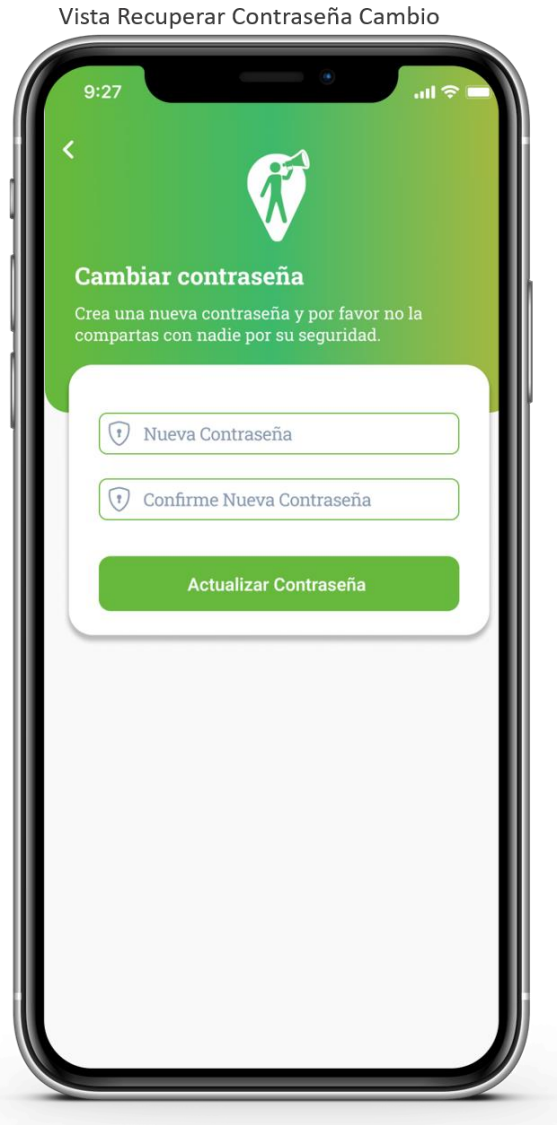
Ilustración 34. *Vista código de confirmación*



Luego de ingresar su dirección de correo electrónico en la vista de recuperación de contraseña el usuario será llevado a la presente vista, ver ilustración 34, y el sistema validará si el correo electrónico se encuentra en la base de datos del aplicativo, de estarlo, le será enviado un

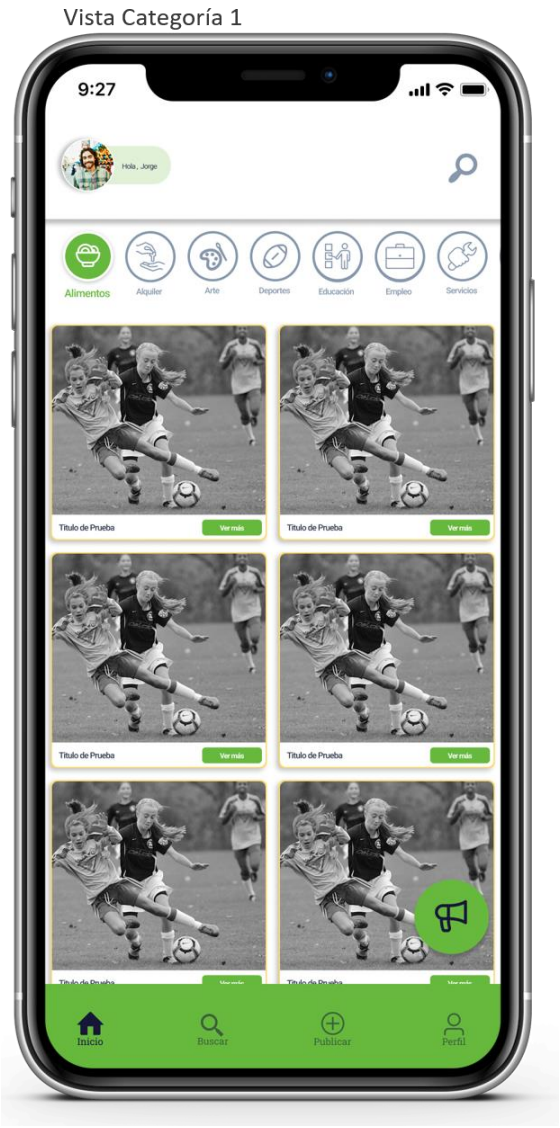
código de confirmación a su correo electrónico, código que el usuario deberá ingresar en la presente vista para continuar con el proceso de recuperación.

Ilustración 35. *Vista cambia tu contraseña*



Luego de que el usuario ingrese el código de verificación en la vista de código de confirmación, ver ilustración 34, y luego de que este código sea validado como correcto al ser el enviado al correo electrónico, el usuario podrá cambiar su clave en la presente vista, ver ilustración 35, permitiéndole el acceso a la plataforma.

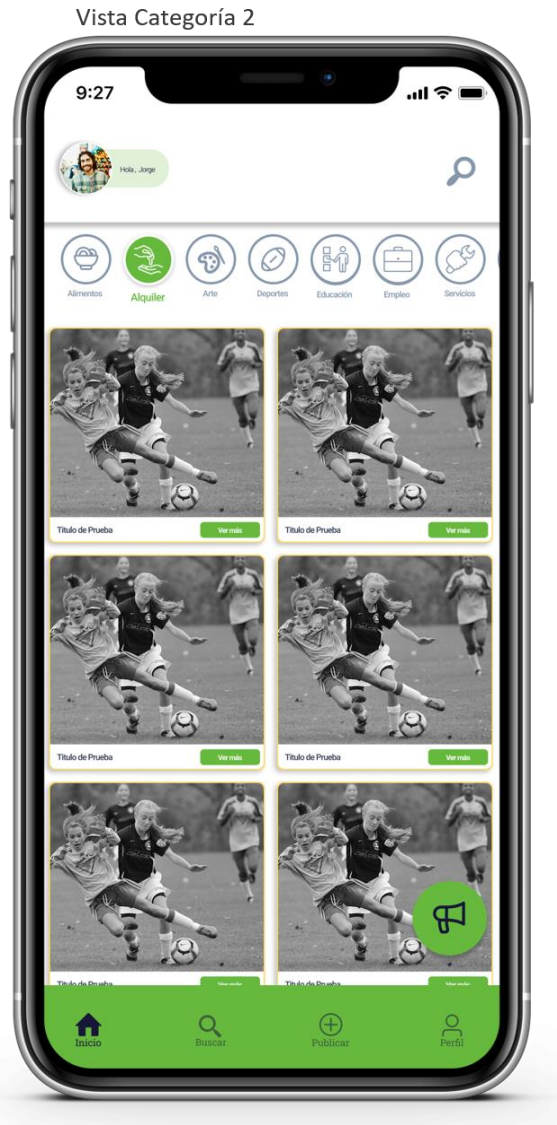
Ilustración 36. *Vista categoría 1 V2*



Al iniciar sesión en la plataforma el usuario se encontrará con esta vista, ver ilustración 36, esta segunda versión se diferencia del diseño uno al incorporar un diseño más amigable con el usuario y más opciones en la barra de navegación inferior, dando cumplimiento al requerimiento no funcional RNF02, siendo añadidas la opción de búsqueda de anuncios y un acceso directo al perfil del usuario, además de esto, en esta segunda versión de diseño se optó por identificar a las

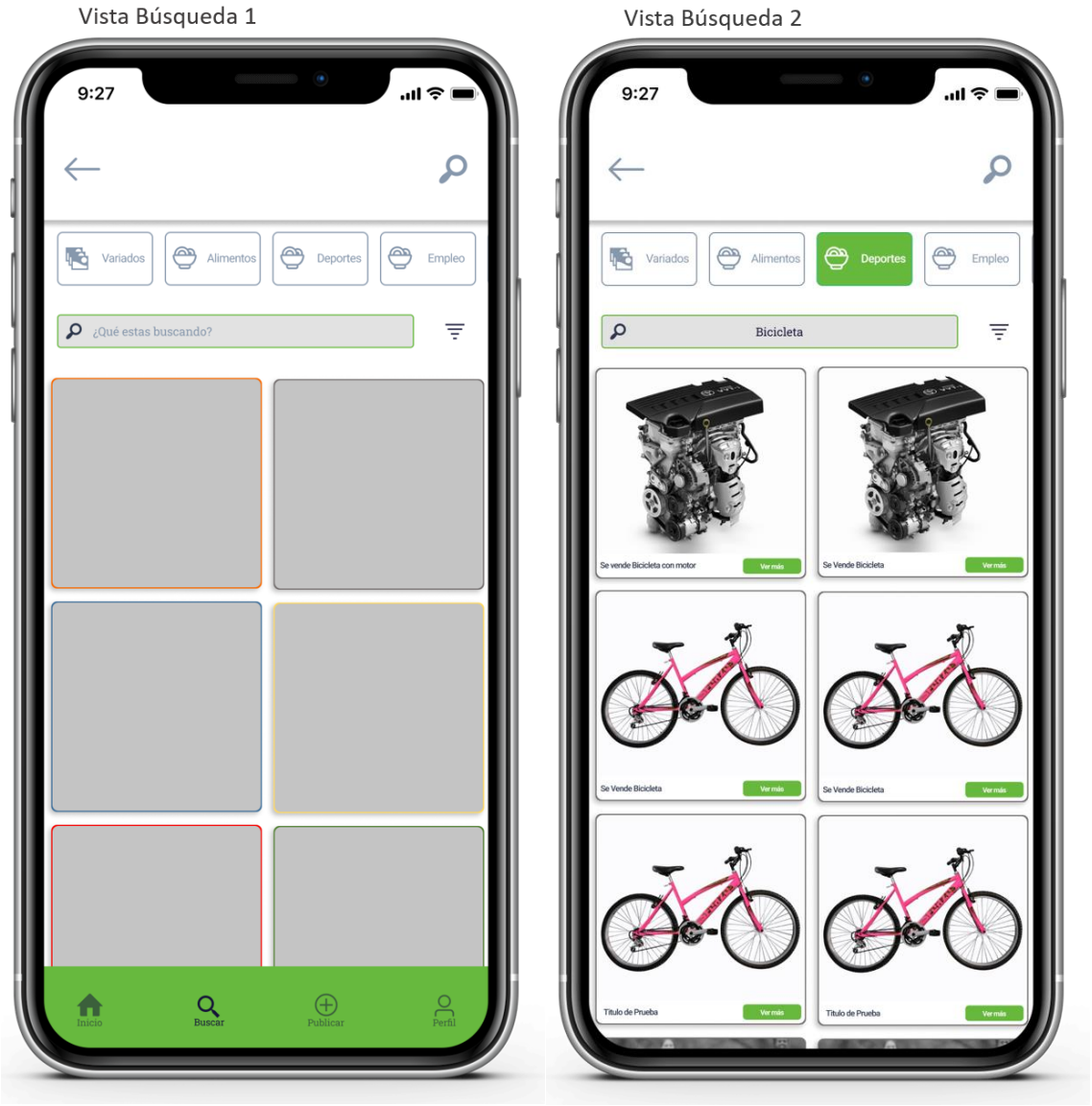
diferentes categorías de la plataforma, no solo por su nombre sino también por un logo distintivo que las identifique, más un saludo de bienvenida.

Ilustración 37. *Vista categoría 2 V2*



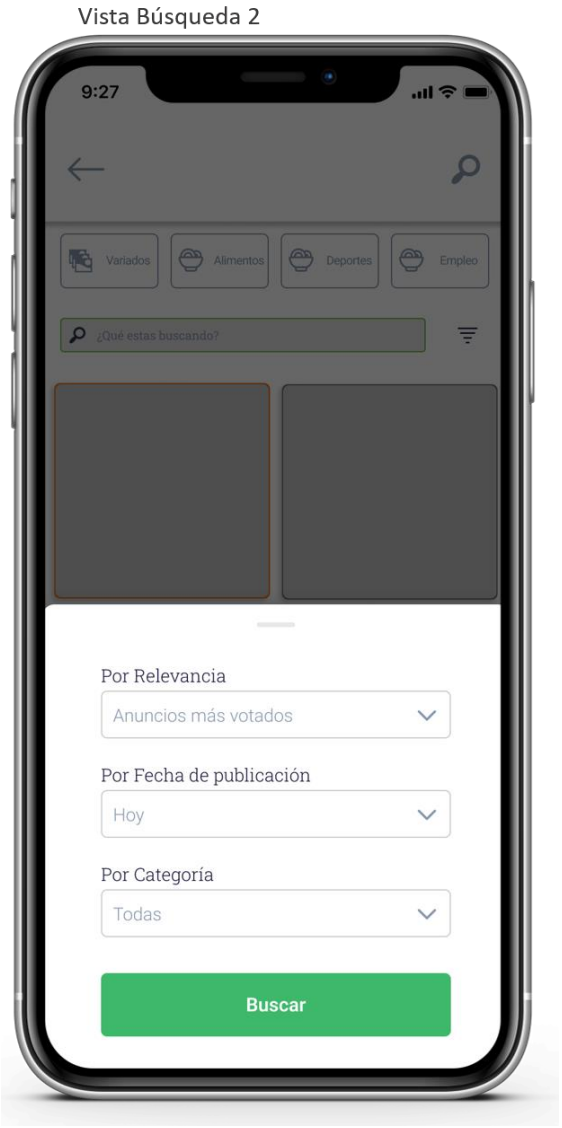
Con respecto al diseño inicial, en este segundo diseño, ver ilustración 37, se decidió mantener la característica que indica a cuál categoría pertenece el anuncio según el borde de la previsualización del anuncio, además, como puede observarse, el icono de la categoría se resalta junto con su nombre al estar dicha categoría seleccionada.

Ilustración 38. *Vista búsqueda V2*



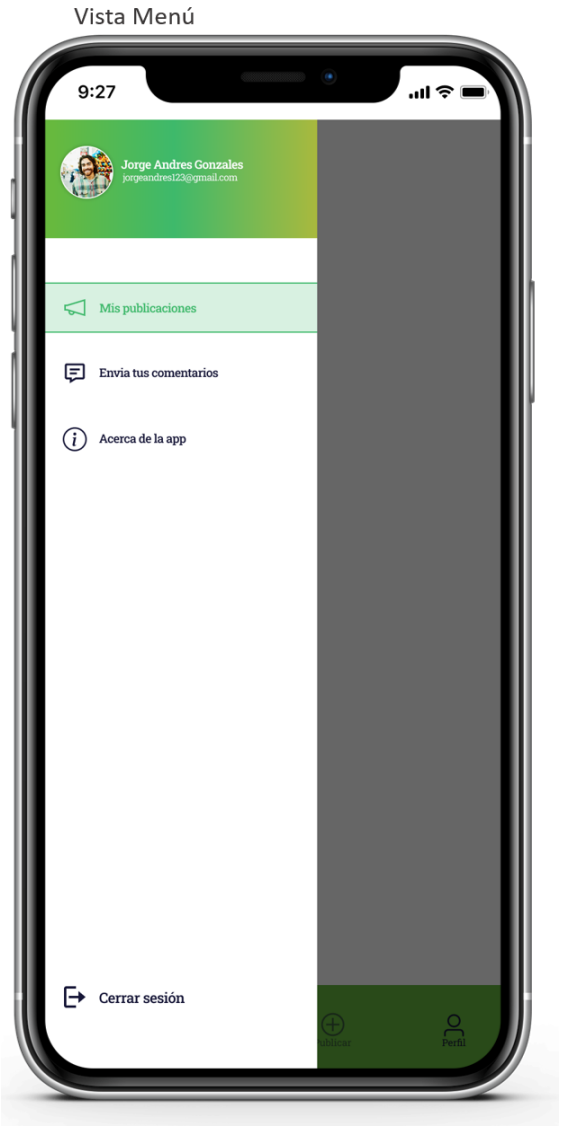
En esta segunda versión de diseño, ver ilustración 38, se optó por remover el botón flotante como acceso a los filtros de búsqueda y ser reemplazado por un botón de acceso a la derecha del campo de ingreso de texto, además de removerse el acceso al perfil de usuario ya que este ya cuenta con uno a través de la barra de navegación inferior, fueron incluidos los iconos de las categorías en el filtro por categoría para dar mayor claridad al usuario.

Ilustración 39. *Vista filtros de búsqueda V2*



En este diseño, ver ilustración 39, se buscó darle una mayor claridad al usuario sobre los filtros de búsqueda de los que este puede disponer, además fue añadida a la ventana emergente la opción de filtrado por categoría como método adicional para acceder a este filtro.

Ilustración 40. *Vista menú lateral V2*

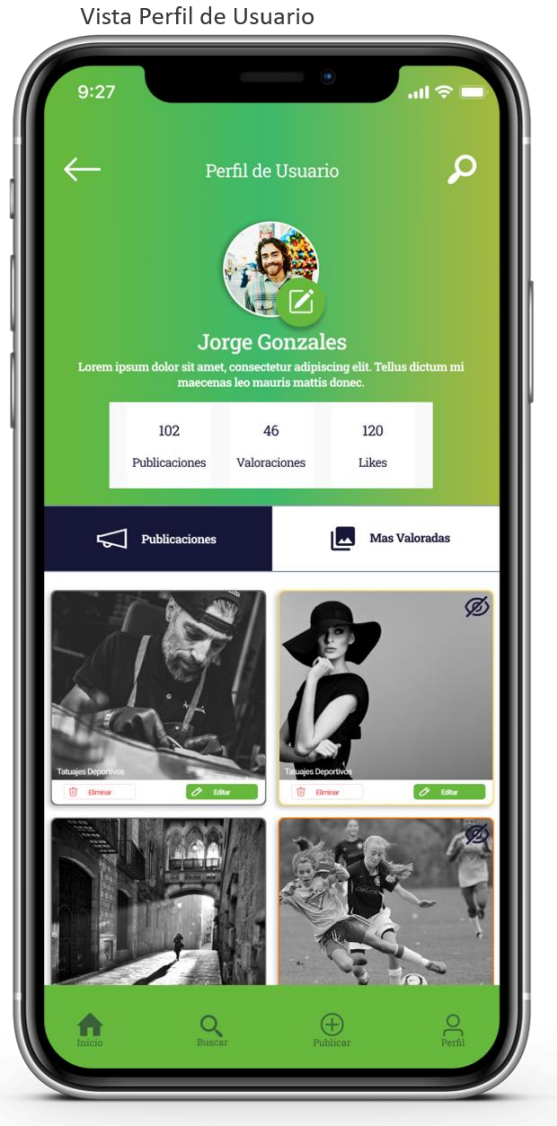


Con relación al diseño original, ver ilustración 23, en este diseño, ver ilustración 40, fueron mantenidas las mismas opciones, con la variación de que el acceso al perfil de usuario fue renombrado como el acceso a “Mis publicaciones”, esto con el fin de que el usuario tenga en claro que a través de esta opción podrá visualizar sus anuncios publicados.

Ilustración 41. *Vista Acerca de la app*

Fue incluido al diseño del aplicativo la información vista cuando el usuario acceda a la opción “Acerca de la app”, en la vista del menú lateral, ver ilustración 40, donde a través de una ventana emergente podrá visualizar una breve descripción de la motivación del proyecto y sus autores, ver ilustración 41.

Ilustración 42. *Vista perfil de usuario V2*



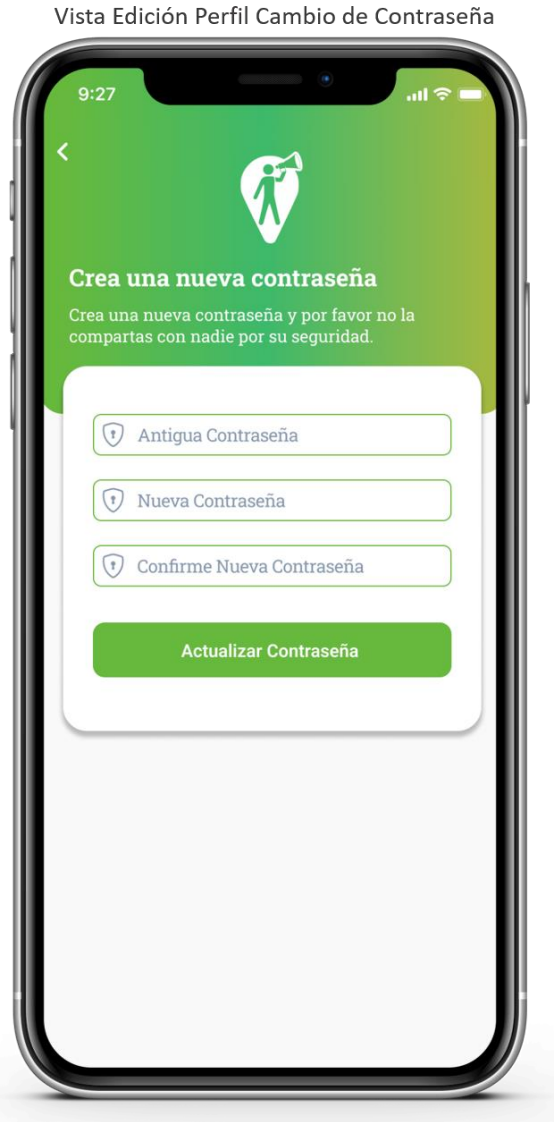
Con respecto al diseño uno, ver ilustración 24, fueron planteados varios cambios a la vista de perfil de usuario, ver ilustración 42, el botón de edición de perfil fue desplazado sobre la imagen del usuario para que este tenga mayor rapidez a la hora de entender la funcionalidad de este botón, además fue añadido a la vista la visualización de la cantidad de anuncios publicados, la cantidad de valoraciones totales recibidas en sus anuncios además del cómputo total de todas las valoraciones obtenidas en ellos, en suma a esto, el perfil de usuario contará con un botón con el

cual el usuario podrá elegir si desea visualizar sus publicaciones por las más recientes primero (visualización por defecto del perfil) o si desear verlas por las mejores valoradas primero (opción más valoradas), fue a su vez añadida la funcionalidad de publicar anuncios pero ocultando su visualización al público, opción que de estar activada en un anuncio se reflejaría en su previsualización con una marca para indicarlo.

Ilustración 43. *Vista edición datos personales V2*

En este segundo diseño de la vista de edición de datos personales, ver ilustración 43, fue añadida la posibilidad de que el usuario modifique su imagen de perfil, dando cumplimento al requerimiento funcional RF20, a su vez de incluir la edición de una descripción para su perfil en esta vista, se decidió también reubicar las opciones de cambio de contraseña en una nueva ventana a la cual se puede acceder mediante un botón en la zona inferior de la vista.

Ilustración 44. *Edición perfil cambio de contraseña*



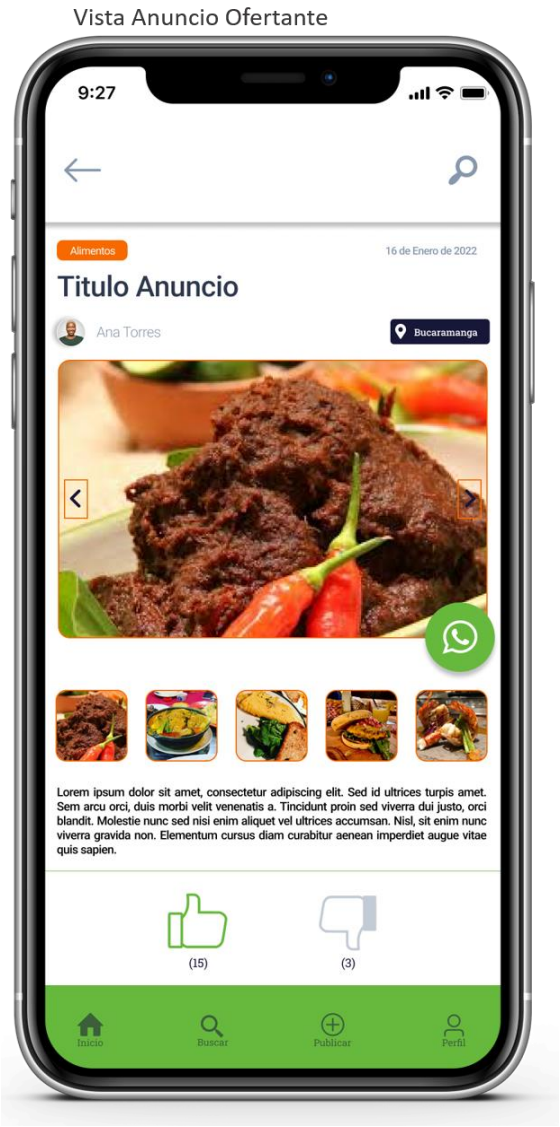
En esta nueva vista, ver ilustración 44, el usuario podrá realizar el cambio de contraseña, validando su contraseña actual e ingresando dos veces su nueva contraseña.

Ilustración 45. *Vista publicación de anuncio V2*

Cuando el usuario seleccione la opción de publicar un anuncio accederá a esta vista, ver ilustración 45, donde su mejora más destacable respecto al diseño uno, ver ilustración 26, es la incorporación de la opción de publicar un anuncio con la visibilidad desactivada, funcionalidad

con la que se busca que un usuario pueda guardar un anuncio como borrador para completar su información en un momento posterior.

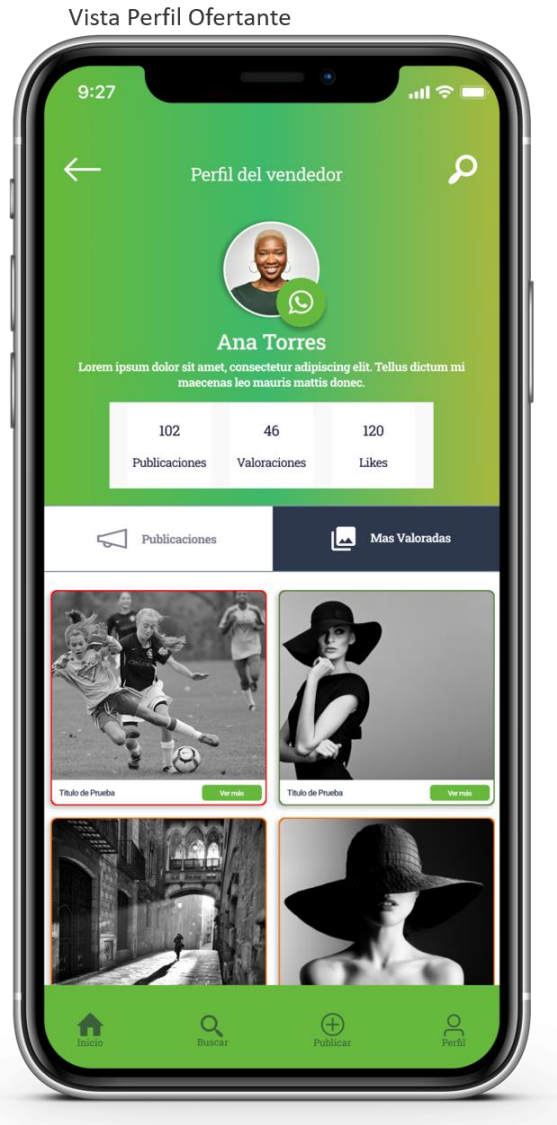
Ilustración 46. *Vista anuncio V2*



En esta nueva versión, ver ilustración 46, se mejora la distribución de los componentes del anuncio respecto al diseño original, ver ilustración 27, otorgándole más protagonismo a las imágenes del anuncio, además se añade nueva información al anuncio que puede ser muy útil para el usuario, como la ciudad de residencia del anunciante junto con su nombre, además de la fecha

de publicación del anuncio, brindando con esto, más herramientas al usuario para tomar una decisión.

Ilustración 47. *Vista perfil anunciante V2*



Empleando el nuevo diseño de perfil de usuario, se genera esta vista, ver ilustración 47, brindando la posibilidad de ver fácilmente los anuncios mejor valorados del anunciante, esto en pro de ayudar al usuario a encontrar más y mejor contenido de su interés, así mismo, se posiciona

el botón de abrir un chat directo de Whatsapp con el anunciante sobre su imagen de perfil, esto buscando hacer más intuitivo el acceso a esta opción.

4.4 Desarrollo

4.4.1 Arquitectura

En esta sección se describe de manera general la arquitectura de la cual está compuesta el prototipo, y se explica cada uno de los componentes que conforma dicho sistema.

Para el desarrollo del prototipo, se siguió el patrón de arquitectura Modelo Vista Controlador, en la ilustración 48, se observa las herramientas utilizadas en cada uno de los componentes que conforman el prototipo.

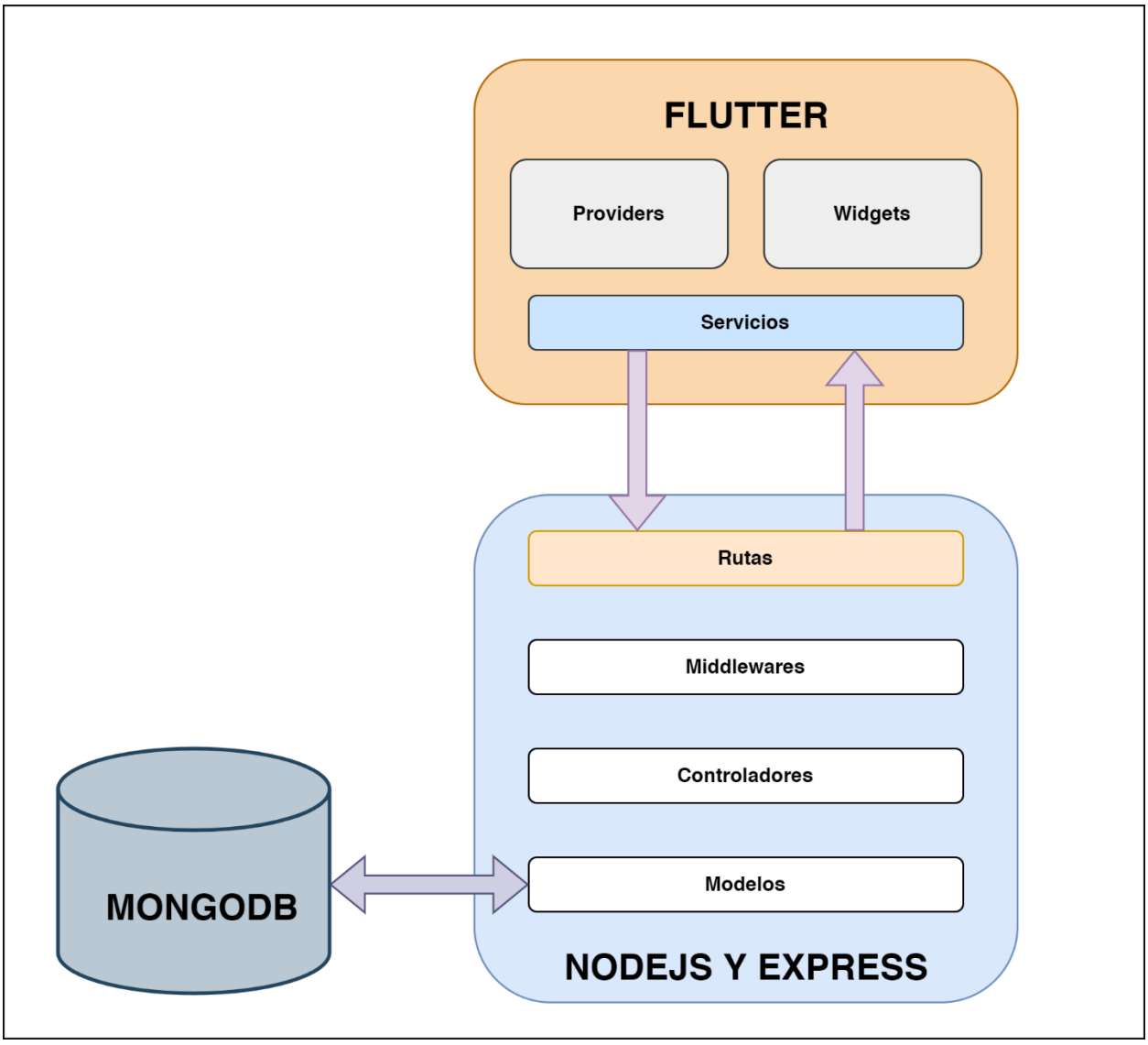
Para el componente de la vista se usó el framework Flutter, cumpliendo este con el propósito de crear y desplegar la aplicación, lugar donde se mostrarán los anuncios y perfiles de usuario. El aplicativo emplea el protocolo HTTP para interactuar con el servidor albergando la parte backend. Por otro lado, el componente controlador y modelo están desarrollados en NodeJS, Express y MongoDB, encontrándose este dividido en cuatro capas: rutas, middlewares, controladores y modelos.

Las rutas son las encargadas de establecer las conexiones entre el cliente y el servidor, estas poseen métodos para poder realizar la comunicación que son: GET, POST, PUT Y DELETE, y esto con el fin de realizar la correcta conexión mediante HTTP y poder visualizar la información del lado de la aplicación.

Los middlewares son funciones que se usan para validar o tratar la información que es enviada desde el lado de los clientes antes de llegar a los controladores, estos se usan para validar la autenticación y verificación de la información.

Los controladores son funciones que se emplean para la inserción, búsqueda, cambios y eliminación de información en la base de datos del prototipo. Los modelos son objetos que hacen referencia a las entidades que se encuentran en la base de datos de MongoDB.

Ilustración 48. *Arquitectura del aplicativo*



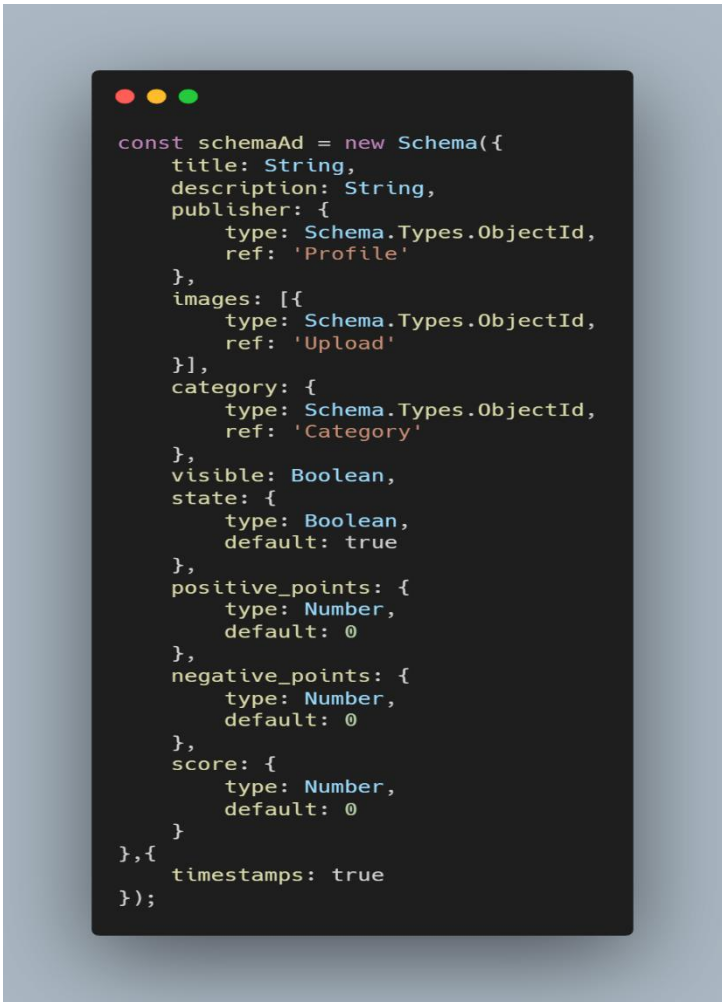
4.4.2 Backend

Abarcando toda la parte lógica del aplicativo, el backend conforma la estructura básica de la aplicación, esta sección del proyecto fue desarrollada usando el entorno de trabajo Express propio de Node.js.

4.4.2.1 Modelo

Cuando se habla de modelos, se hace referencia al objeto del cual tendrá la funcionalidad de esquematizar la información de la colección que se encuentra alojada en la base de datos. Estos objetos permiten, de un modo más simple, gestionar la información que es recibida del prototipo.

Ilustración 49. *Codificación del modelo de datos backend*



```
const schemaAd = new Schema({
    title: String,
    description: String,
    publisher: {
        type: Schema.Types.ObjectId,
        ref: 'Profile'
    },
    images: [{
        type: Schema.Types.ObjectId,
        ref: 'Upload'
    }],
    category: {
        type: Schema.Types.ObjectId,
        ref: 'Category'
    },
    visible: Boolean,
    state: {
        type: Boolean,
        default: true
    },
    positive_points: {
        type: Number,
        default: 0
    },
    negative_points: {
        type: Number,
        default: 0
    },
    score: {
        type: Number,
        default: 0
    }
},{
    timestamps: true
});
```

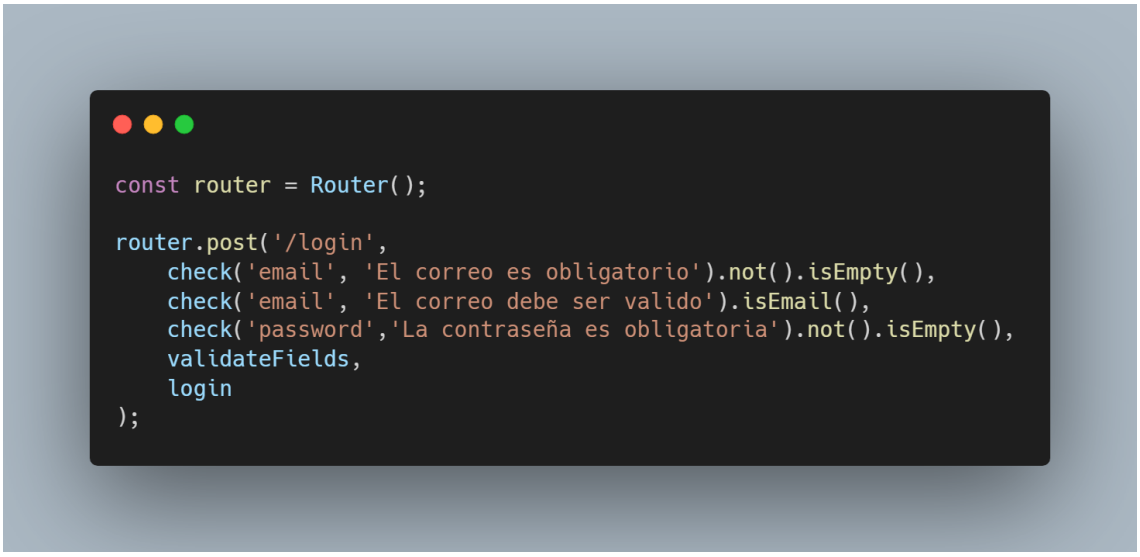
Como se observa en la ilustración 49, para la creación del modelo se usa la clase Schema importada de la dependencia de mongoose, dependencia la cual logra facilitar la conexión con la base de datos en MongoDB. Con esta clase, definimos las columnas que tendrá el objeto, donde en cada una se especifica el tipo de dato que tendrá. Además, existe la posibilidad de crear "relaciones" entre colecciones empleando el atributo ObjectId, esto nos permite referenciar el identificador de un registro que se encuentre en la tabla y poder traer los datos del mismo.

Por último, se usa la función model para definir el nombre de la colección y el esquema que se usa.

4.4.2.2 Rutas

Las rutas son las encargadas de conectar el componente backend con el cliente, mediante métodos HTTP (GET, POST, PUT, DELETE), esto permite crear un puente para la transferencia de datos dentro del sistema, logrando de esa forma realizar la correcta gestión de información en el servidor.

Ilustración 50. *Codificación de una ruta*



```
const router = Router();

router.post('/login',
    check('email', 'El correo es obligatorio').not().isEmpty(),
    check('email', 'El correo debe ser valido').isEmail(),
    check('password','La contraseña es obligatoria').not().isEmpty(),
    validateFields,
    login
);
```

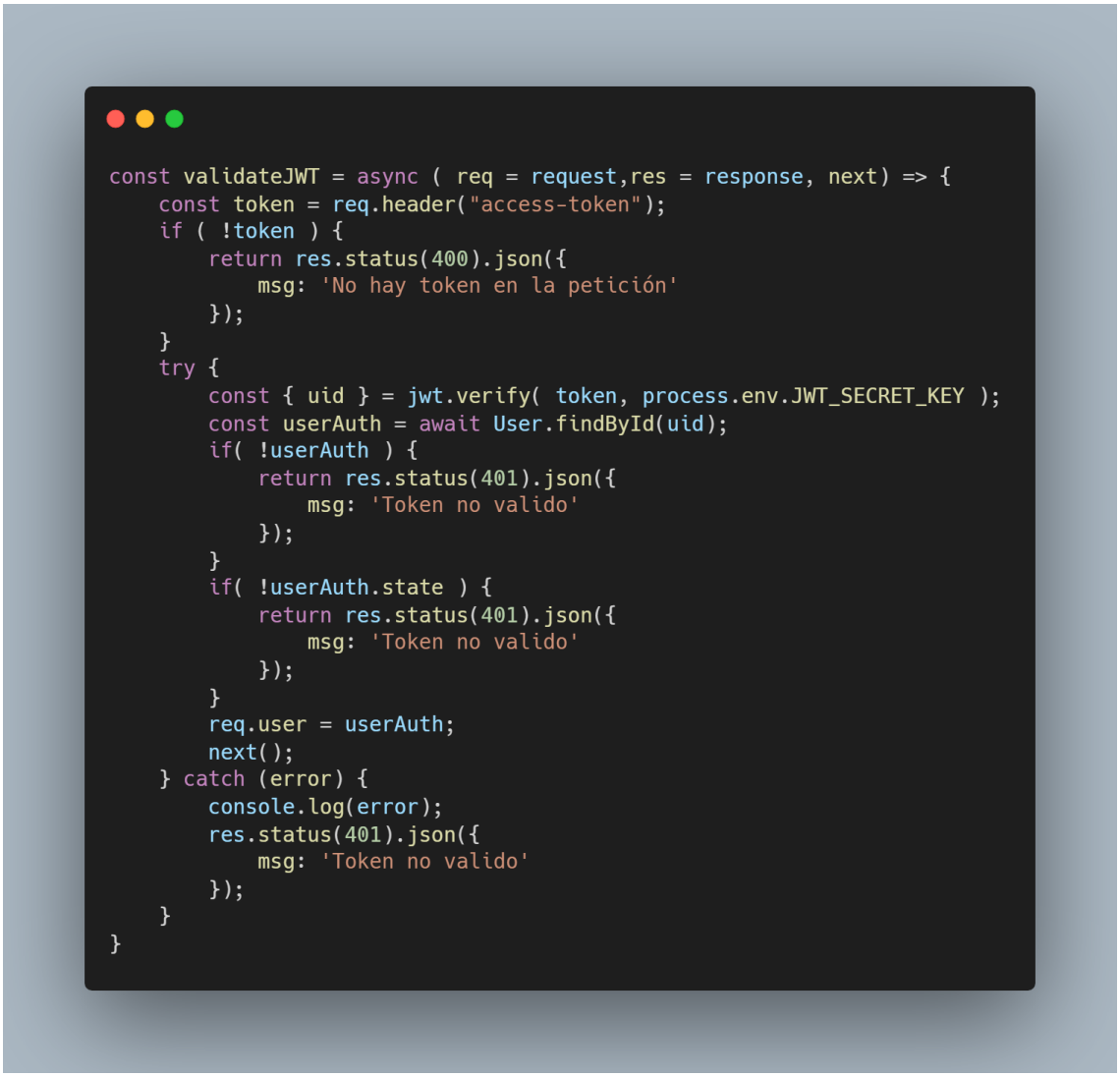
En la ilustración 50, se observa la creación de una instancia de la clase Router, esta clase es llamada desde la dependencia de express para realizar el manejo de las rutas dentro del servidor. Para la creación de una nueva ruta se usa la instancia ya inicializada, llamando uno de los protocolos HTTP, de esta forma se puede establecer un enlace con el aplicativo mediante los métodos que nos concede la clase.

Estos dos métodos reciben dos parámetros, el primer parámetro hace referencia al nombre del endpoint que se va a llamar desde la aplicación, para la transferencia de la información, como segundo argumento, tenemos las diferentes funciones por las cuales va a pasar el request del frontend, la mayoría de las funciones son middlewares que validan la información proveniente del aplicativo, por último, la ruta es entregada al controlador para manejar la información en la base de datos.

4.4.2.3 Middlewares

Los middlewares son métodos para validar y verificar información proveniente del aplicativo, estas funciones son usadas para el manejo de los objetos empleados por express en la gestión de los datos dentro del servidor, los métodos request y response. El funcionamiento de los middlewares puede cambiar dependiendo del modelo de negocio, estos pueden realizar cambios en la solicitud y en la respuesta del servidor, además de validar información y finalizar la petición. También son una capa muy importante para aprobar los datos que viajan hasta el controlador.

Ilustración 51. *Codificación middlewares*



```
const validateJWT = async ( req = request,res = response, next) => {
    const token = req.header("access-token");
    if ( !token ) {
        return res.status(400).json({
            msg: 'No hay token en la petición'
        });
    }
    try {
        const { uid } = jwt.verify( token, process.env.JWT_SECRET_KEY );
        const userAuth = await User.findById(uid);
        if( !userAuth ) {
            return res.status(401).json({
                msg: 'Token no valido'
            });
        }
        if( !userAuth.state ) {
            return res.status(401).json({
                msg: 'Token no valido'
            });
        }
        req.user = userAuth;
        next();
    } catch (error) {
        console.log(error);
        res.status(401).json({
            msg: 'Token no valido'
        });
    }
}
```

En la ilustración 51, se especifica la codificación de un middleware, se observa su funcionamiento en la validación de la existencia de un token registrado a un usuario, esto nos ayuda a validar si efectivamente este usuario existe en la base de datos y tiene permisos de acceder a los controladores, ayudándonos a proteger los controladores y así evitar la inserción de información fraudulenta a través de ellos y, por consiguiente, a la base de datos. Esta última recibe tres parámetros, el primero hace referencia a la solicitud del cliente, el segundo apunta a la respuesta del servidor, y por último, el tercero es una función next() usada para llamar al siguiente método en la pila de funciones estipuladas en la ruta.

4.4.2.4 Controlador

Los controladores son funciones cuyo objetivo es darnos una ayuda en la gestión de la información procesada proveniente del frontend, su función principal es insertar, editar o eliminar registros que se encuentran en la base de datos, esto mediante el uso del modelo. También es el encargado de retornar una respuesta válida al cliente, ya sea esta una respuesta exitosa o un error producido por la solicitud.

Ilustración 52. *Codificación de un controlador*

```
const getAds = async ( req = request, res = response ) => {
    try {
        const { pageValue, sortValue, sortDirection, filter, pageSize } = req.body;
        const page = {
            number: pageValue,
            size: pageSize
        };
        const sort = {
            value : sortValue,
            direction: sortDirection
        };
        const ads = await makePagination(page, sort, {} , filter );
        if ( !ads ) {
            errors = errorHandler('No se encontraron anuncios');
            return res.status(404).json({ errors });
        }
        res.status(200).json({ ads });
    } catch (error ) {
        errors = errorHandler('No se pueden visualizar los anuncios');
        return res.status(500).json({ errors });
    }
}
```

En la ilustración 52, se puede observar que el controlador recibe dos parámetros, el primero, el request, hace referencia a la solicitud del frontend, donde se encuentra toda la información ajustada con validaciones, el segundo parámetro es el objeto response, al cual hace alusión a la respuesta que se envía al cliente.

Dentro del cuerpo del controlador, se puede extraer información proveniente de sus argumentos para gestionar los modelos que se encargan de insertar, editar o eliminar los registros en la base de datos, igualmente, se invoca bloques de código que se encuentran globalizados y que ayudan a realizar las operaciones sin cargar o hacer uso del controlador.

Por última acción, el objeto response envía la información al cliente mediante los métodos status y json, el primer método define el código de respuesta que tendrá la petición HTTP, la cual podrá ser exitosa o fallida dependiendo de la lógica que posea el controlador, mientras tanto, el método json permite dirigir la información dentro de un objeto JSON.

4.4.2.5 Helper

Los helpers son bloques de códigos que permiten organizar, agregar, eliminar o procesar información que se usan en los controladores. Los helpers se encuentran centralizados y generalizados para poder ser utilizados en cualquier instancia del servidor. Además, cumplen una funcionalidad específica y permiten desacoplase de lógica al controlador.

Ilustración 53. *Codificación de un helper*

```
const createProfile = async ( req = request, res = response, user ) => {
    try {
        const { name, cellphone, city } = req.body;
        const newProfile = new Profile({
            name,
            cellphone,
            user: user._id,
            city
        });
        const savedProfile = await newProfile.save();
        if ( !savedProfile ) {
            errors = errorHandler( 'No se pudo crear el perfil' );
            return res.status(404).json({ errors });
        }
        return savedProfile;
    } catch (error) {
        errors = errorHandler( 'No se pudo crear el perfil' );
        return res.status(500).json({ errors })
    }
}
```

En la ilustración 53, se puede observar como el helper hace la creación del perfil, esto impide que el controlador que realiza el registro no se sobrecargue. Asimismo, se puede llamar una instancia del modelo para realizar cambios en la base de datos y validar errores que puedan llegar a ocurrir dentro del servidor.

4.4.2.6 Listado de peticiones HTTP

A continuación, son presentados los listados de peticiones HTTP propios de las rutas de los servicios creados por entidad en el aplicativo, junto con el método empleado y su respectiva descripción.

Tabla 41. *Rutas de anuncio*

URI	Método	Descripción
/api/ad/	GET	Obtiene todos los anuncios
/api/ad/:id	GET	Obtiene un anuncio en especifico
/api/ad/publisher/:id	GET	Obtiene los anuncios por anunciante
/api/ad/category/:id	GET	Obtiene los anuncios por categoría
/api/ad/	POST	Crea un nuevo anuncio
/api/ad/:id	PUT	Actualiza la información de un anuncio
/api/ad/:id	DELETE	Elimina un anuncio
/api/ad/rating/:id	POST	Gestiona la calificación de un anuncio en especifico
/api/ad/search/:query	GET	Busca anuncios por la palabra clave suministrada

Tabla 42. *Rutas de autenticación*

URI	Método	Descripción
/api/auth/login	POST	Inicia Sesión
/api/auth/register	POST	Registro de un nuevo usuario
/api/auth/change-password	POST	Cambio de contraseña
/api/auth/forget-password	POST	Petición nuevo cambio de contraseña
/api/auth/verify-otp	POST	Verifica el OTP enviado

Tabla 43. *Rutas de perfil*

URI	Método	Descripción
/api/profile/:id	GET	Obtiene el perfil de usuario
/api/profile/:id	PUT	Actualiza la información del perfil de usuario
/api/profile/calculate	POST	Calcula la calificación de perfil de usuario

Tabla 44. *Rutas de categorías*

URI	Método	Descripción
/api/category/	GET	Obtiene todas las categorías
/api/cateory/	POST	Crea una nueva categoría

Tabla 45. *Rutas de ciudad*

URI	Método	Descripción
/api/city/	GET	Obtiene todas las ciudades
/api/city/	POST	Crea una nueva ciudad
/api/city/:id	GET	Obtiene una ciudad en especifico

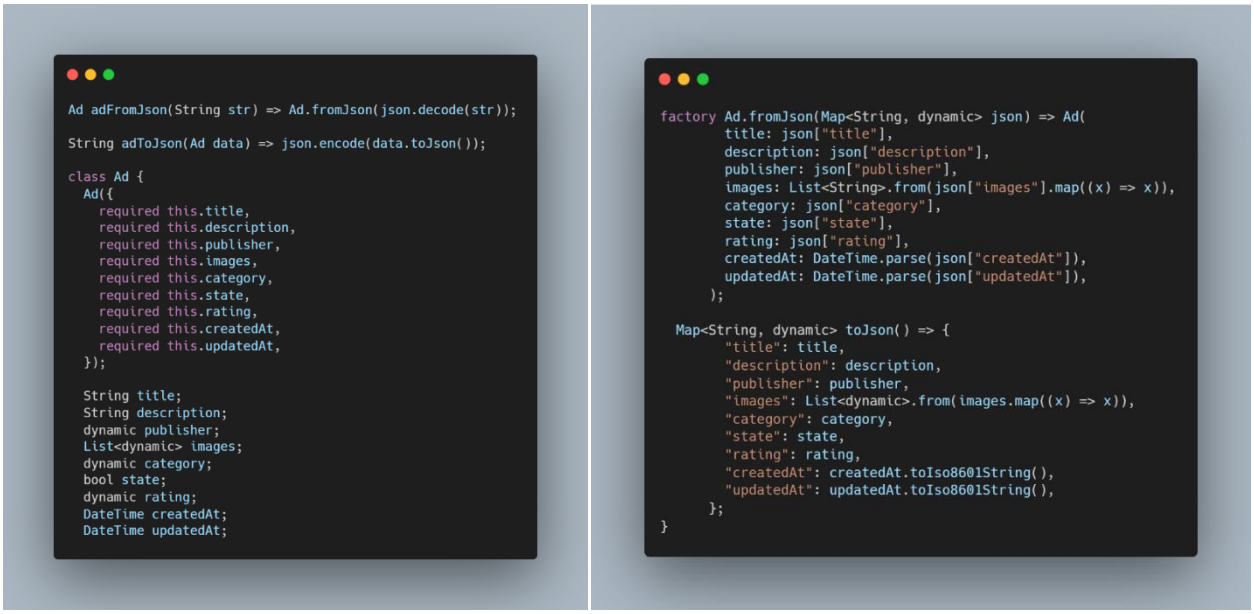
4.4.3 Frontend

La parte frontend del aplicativo abarca la programación de la parte estética y externa del aplicativo, comprendiendo la visualización final a la cual podrá acceder el usuario, encontrándose está desarrollada en el lenguaje Dart propio del SDK Flutter empleado en esta sección del proyecto.

4.4.3.1 Modelo

En el frontend del aplicativo, en el directorio de models, están contenidas las clases PODOs (Plain Old Dart Object) o clases modelos, concernientes a la respuesta JSON obtenida de parte del Backend. Los modelos empleados en el frontend apuntan a los datos puros que se presentan en la pantalla del aplicativo, todos estos datos son ingresados en las clases o plantillas generadas para este fin.

Ilustración 54. *Codificación del modelo de datos backend*



```dart
Ad adFromJson(String str) => Ad.fromJson(json.decode(str));

String adToJson(Ad data) => json.encode(data.toJson());

class Ad {
  Ad({
    required this.title,
    required this.description,
    required this.publisher,
    required this.images,
    required this.category,
    required this.state,
    required this.rating,
    required this.createdAt,
    required this.updatedAt,
  });

  String title;
  String description;
  dynamic publisher;
  List<dynamic> images;
  dynamic category;
  bool state;
  dynamic rating;
  DateTime createdAt;
  DateTime updatedAt;
```

```dart
factory Ad.fromJson(Map<String, dynamic> json) => Ad(
        title: json["title"],
        description: json["description"],
        publisher: json["publisher"],
        images: List<String>.from(json["images"].map((x) => x)),
        category: json["category"],
        state: json["state"],
        rating: json["rating"],
        createdAt: DateTime.parse(json["createdAt"]),
        updatedAt: DateTime.parse(json["updatedAt"]),
      );

  Map<String, dynamic> toJson() => {
        "title": title,
        "description": description,
        "publisher": publisher,
        "images": List<dynamic>.from(images.map((x) => x)),
        "category": category,
        "state": state,
        "rating": rating,
        "createdAt": createdAt.toIso8601String(),
        "updatedAt": updatedAt.toIso8601String(),
      };
}
```

En la ilustración 54, podemos ver ilustrado el modelo usado para representar un anuncio en la aplicación, el modelo es la base que soporta la vista para mostrar los datos recibidos desde la parte backend como objetos y sus respectivas propiedades. Los PODOs, actúan como una forma sencilla de tomar un JSON, decodificando y mostrando los datos deserializados en la aplicación, lugar donde transformamos las claves y valores del JSON en propiedades de una clase. Utilizando el método Factory, propio de su mismo patrón de diseño, tomamos los elementos de un Map u objeto JSON, y construimos un objeto base, donde son mapeados los datos recibidos del modelo en DART, logrando gestionar los datos almacenados en el aplicativo para su respectivo uso en las diferentes vistas y pantallas contenidas en el sistema.

4.4.3.2 Servicio

En el aplicativo móvil los servicios son clases que ofrecen una funcionalidad especifica. Un servicio en DART puede ser considerado una clase que usa métodos los cuales permiten proporcionar una característica especializada, ofreciendo una entrada distinta a la aplicación, ejemplos de algunos servicios que son usados en la aplicación pueden ser, obtener datos de un API, almacenar datos en el LocalStorage del dispositivo, gestionar la conectividad de la aplicación, manejar los temas de la aplicación, realizar cálculos complejos o en gran cantidad, trabajar con paquetes de terceros, entre otros. Los servicios buscan mantener la base de código lo más limpia posible y ayudan a reducir la cantidad de redundancias presentes.

Los servicios están contenidos dentro de la aplicación y no extendidos por la red como su nombre los puede dar a entender, pueden llamarse también como helpers o repositories. Su propósito principal es aislar una tarea y ocultar sus detalles de implementación del resto de la aplicación. Cuando la aplicación presenta un estrecho acoplamiento debido al uso de alguna función dispersa alrededor del código, lo hace difícil de mantener y propenso a errores.

Un ejemplo práctico de la importancia de los servicios, es por ejemplo el uso de las preferencias compartidas o Shared Preferences para guardar algunos datos del usuario, los cuales deben ser guardados y recuperados en varios lugares de la aplicación, implicando la posibilidad de presentar un problema en el almacenamiento debido a la necesidad de almacenar una gran cantidad de datos, más tarde puede suceder que se deja de actualizar el paquete de terceros usado para gestionar el almacenamiento, surgiendo la necesidad de usar otro paquete, provocando el cambio de referencias en el código y una gran posibilidad de que se presenten errores por el cambio de un parámetro o el cambio de paquete.

Aquí los servicios son una herramienta que puede dar solución al problema, con la cual se puede diseñar una clase StorageService que se encargue del almacenamiento de la data de la aplicación, librándonos de la necesidad de afectar las demás clases de la app, ya que simplemente realizan la llamada a los métodos en el servicio para guardar y recuperar datos. Esto hace que sea relativamente más sencillo cualquier cambio, por ejemplo, el paso de almacenar en preferencias compartidas a el almacenamiento en una base de datos usando peticiones a un backend, que pueden ser expresadas por otro servicio. O la actualización del código de un servicio, afectando automáticamente a todos los lugares donde se usa el servicio en la app.

Ilustración 55. *Codificación servicio de almacenamiento*

```dart
class StorageService {
  static const storage = FlutterSecureStorage(
      aOptions: AndroidOptions(encryptedSharedPreferences: true));

  static Future<void> saveData(String key, String value) async {
    await storage.write(key: key, value: value);
  }

  static Future<String?> readData(String key) async {
    return await storage.read(key: key);
  }

  static Future<Map<String, String>> readAllData() async {
    return await storage.readAll();
  }

  static Future<bool> containsData(String key) async {
    return await storage.containsKey(key: key);
  }

  static Future<void> deleteData(String key) async {
    await storage.delete(key: key);
  }

  static Future<void> deleteAllData() async {
    await storage.deleteAll();
  }
}
```

En la ilustración 55, se presenta un ejemplo del servicio de almacenamiento en la aplicación de forma segura, para almacenar datos tales como el token para la autenticación e inicio de sesión, este servicio fue implementado en el prototipo.

4.4.3.3 Provider

El manejo de estados dentro de una aplicación es un reto común al que se enfrentan todos los desarrolladores al momento de programar un aplicativo, siendo el control de esto obligatorio para el correcto funcionamiento del sistema. Para la gestión del estado global de una aplicación en Flutter existen diferentes maneras de realizarlo, puede ser usando paquetes que implementan el patron BLoC, Getx, Redux, Provider, entre otros. La forma más sencilla de gestionar los estados

de una app sin recurrir al uso de paquetes, es mediante el uso de StateFullWidgets y la función setState, funcionalidades ya incluidas por defecto en el interior de la API de Flutter. Esto puede ser útil en una aplicación con un alcance y tamaño pequeño, pero cuando se tiene que trabajar con una aplicación que contenga múltiples pantallas, gestione diferentes servicios o cargue información que se necesite mostrar entre diferentes widgets, el proceso se vuelve muy complejo de gestionar, llegando a ser imposible su control y la conexión de información entre los diferentes widgets hijos y padres, por ello surge la necesidad de implementar un patrón que cumpla una adecuada gestión del estado de la aplicación.

En nuestro caso del presente aplicativo UISAds, se implementó el patrón Provider para la administración de los estados en la aplicación, esto debido a la sencillez en su implementación y la posibilidad de comunicación que permite entre los componentes que conforman el árbol de widgets de la aplicación, el patrón Provider nos permite compartir información entre dos pantallas diferentes, por ejemplo, para el caso donde es necesario compartir de forma global la información del usuario actualmente autenticado en el aplicativo, el uso de un provider es muy útil, ya que ayuda a mantener esos datos entre pantallas, permitiendo ser usados aun cuando se translade de una pantalla a otra, para lograr esto se envolvió la aplicación en un Widget llamado MultiProvider, el cual permite contener diversos providers de forma global en la aplicación, tal como se muestra en la ilustración 56, posibilitando la conexión y comunicación de los widgets encontrados dentro de él.

Ilustración 56. *Codificación multiprovider*

```
class AppState extends StatelessWidget {
  const AppState({Key? key}) : super(key: key);


  @override
  Widget build(BuildContext context) {
    return MultiProvider(
      providers: [
        ChangeNotifierProvider(create: (_) => BottomNavigationBarProvider()),
        ChangeNotifierProvider(create: (_) => ProfileProvider()),
        ChangeNotifierProvider(create: (_) => RegisterFormProvider())
      ],
      child: const App(),
    );
  }
}
```

Al estar el “state” de Flutter contenido en el provider permite que los widgets “hijos” estén pendientes a los cambios de estado y puedan actualizarse tan pronto como se les notifique, obteniendo la ventaja de que en lugar de tener que hacer la reconstrucción de todo el árbol de widgets (como se hacía usando el StateFullWidget), solamente se reconstruirá el widget que haya sido afectado, lo cual permite reducir la carga de trabajo que se realizada en la aplicación, posibilitando una ejecución más eficiente y sin retrasos.

Ilustración 57. *Codificación provider en inicio de sesión*

```dart
class LoginFormProvider with ChangeNotifier {
  String _email = '';
  String _password = '';
  bool _isLoading = false;
  final formKey = GlobalKey<FormState>();
  final _regExpEmail = RegExp(
      r"^[a-zA-Z0-9.a-zA-Z0-9.!#$%&'*+-/=?^_`{|}~]+@[a-zA-Z0-9]+\.[a-zA-Z]+");

  String get email {
    return _email;
  }

  set email(String value) {
    _email = value;
    notifyListeners();
  }

  String get password {
    return _password;
  }

  set password(String value) {
    _password = value;
    notifyListeners();
  }

  bool get isLoading {
    return _isLoading;
  }

  set isLoading(bool value) {
    _isLoading = value;
    notifyListeners();
  }
}
```

En la ilustración 57, se muestra un ejemplo de un provider empleado para el inicio de sesión en el aplicativo, en donde el método notifyListeners() nos brinda la posibilidad de monitorear la actualización las propiedades y en caso de ocurrir una modificación, el provider pueda comunicar este cambio a todos los widgets conectados a la propiedad, para ejecutar los respectivos cambios en los widgets vinculados.

4.4.3.4 Vistas

Para el diseño de interfaces en flutter, se debe tener en cuenta que, en la pantalla visualizada por el usuario, a la unidad mínima o al bloque de construcción de una pantalla se le

conoce como widget. En flutter una pantalla o screen es un widget normal, pero que se carga de tal forma que llena toda o la mayoría de la pantalla del dispositivo. Para comprender que es un árbol de widgets debemos entender que un widget un elemento de una pantalla, la vista de una pantalla dependerá completamente de la elección de los widgets utilizados para crear la aplicación y la estructura del código de la aplicación es lo que conforma el árbol de widgets. Los widgets describen como deberá mostrarse la vista según su configuración y estado actuales, esto incluye la forma como se maneja los diferentes widgets básicos, los cuales pueden ser usados para estructurar parte de la aplicación, tales como Row widget, Column Widget, Container Widget, Stack Widget, Text Widget, entre otros.

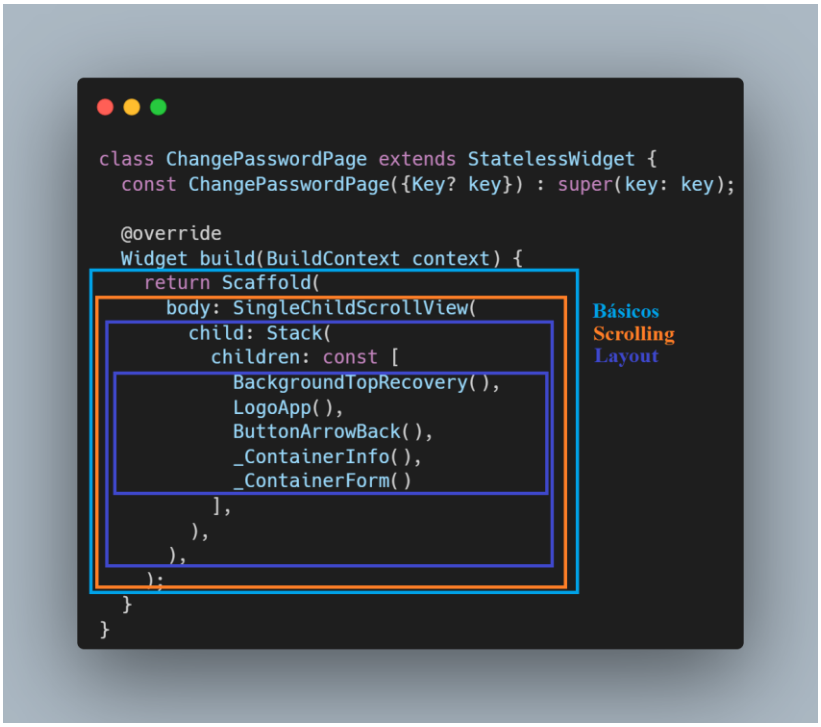
Flutter en su documentación divide los widgets en 14 categorías principales, de las cuales podemos decir que los más usados en el presente desarrollo fueron:

- Básicos
- Recursos, imágenes o iconos
- Input
- Layout
- Recursos, imágenes o iconos
- Componentes de material
- Texto
- Dibujo y efectos
- Estilos
- Scrolling
- Async

Para el caso de la construcción de la pantalla de cambio de contraseña en el aplicativo UISAds se tuvo en cuenta el montaje de la estructura con la ayuda de un widget básico conocido como Scaffold. El Scaffold implementa una estructura de diseño visual para contiene los elementos básicos en una pantalla, dentro del este scaffold podemos crear una estructura appBar, un body o una BottomNavigationBar, entre otros widgets. Esta estructura es muy necesaria para el desarrollo de cualquier aplicación básica en flutter.

En el body del Scaffold podemos añadir cualquier tipo de estructura básica o de layout para construir o anidar más widgets dentro de este, para este caso se construyen más widgets de tipo layout para estructurar los demás widgets internos de la pantalla, en este caso, es usado un stack wigdet para superponer varios elementos secundarios de forma simple uno sobre otro, de tal forma que un texto y un botón puedan colocarse encima de un degradado predefinido, como el código que se presenta en la ilustración 58.

Ilustración 58. *Codificación de una vista*



En el stack, los widgets internos forman la estructura para contener los demás componentes que conforman la vista, dentro del stack se encuentra el widget BackgrundTopRecovery siendo un componente que forma el degradado del fondo, superpuesto a él encontramos el Widget LogoApp, que contiene el Asset o recurso con el logo colocado en la parte superior del degradado, tal como se muestra en la ilustración 59.

Ilustración 59. *Ejemplo estructura de una vista*

El método ButtonArrowBack permite ubicar el IconButton permitiendo regresar a la pantalla anterior y superponiéndose a los dos anteriores, de tal forma que en un Stack entre más abajo se ubiquen los elementos en el código, más arriba se mostrarán en pantalla. El ContainerInfo

contiene la estructura de widgets que abarca los textos de título y descripción mostrados en la figura 59, llevando a cabo su implementación de la siguiente manera:

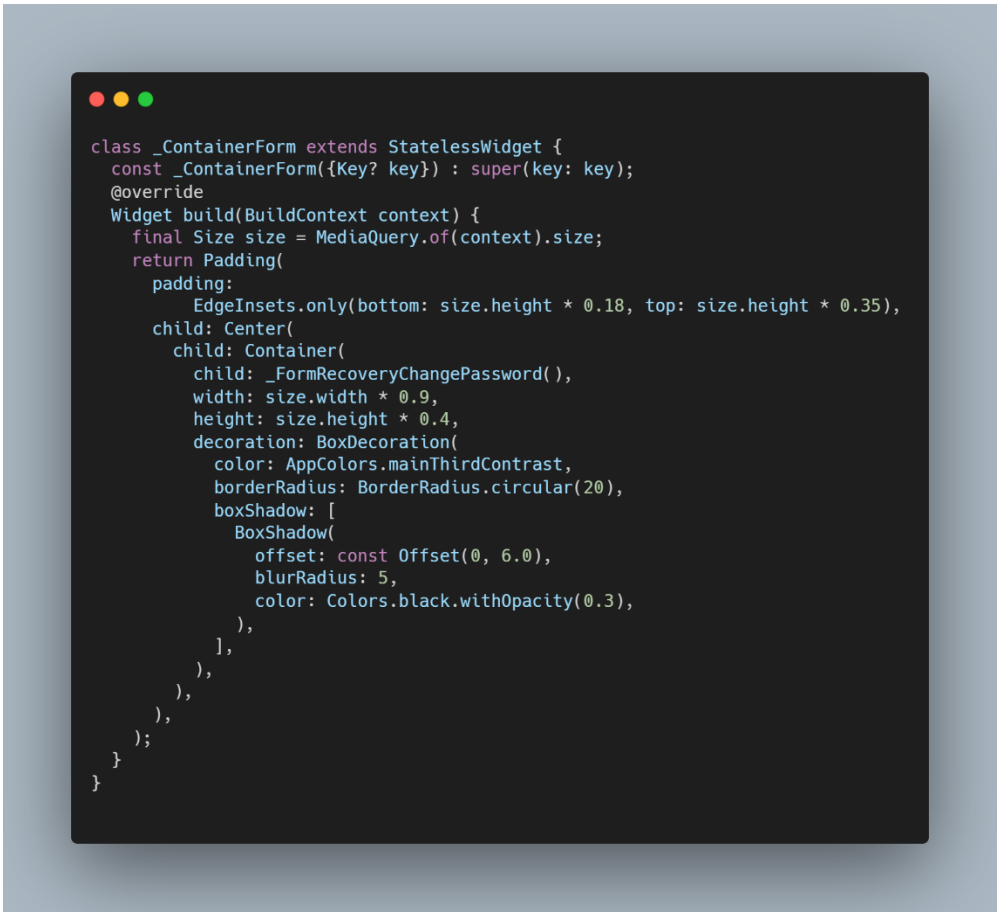
Ilustración 60. *Codificación formulario cambio de contraseña*

```dart
class _ContainerInfo extends StatelessWidget {
  const _ContainerInfo({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    String title = 'Cambiar Contraseña';
    String description =
        'Crea una nueva contraseña y por favor no la compartas con nadie por su seguridad';
    final Size size = MediaQuery.of(context).size;
    return Center(
      child: Column(
        children: [
          SizedBox(
            height: size.height * 0.17,
          ),
          TitleInfo(title: title),
          DescriptionInfo(description: description)
        ],
      ),
    );
  }
}
```

Ubicado en el código en el piso inferior dentro del Stack, ver ilustración 58, se encuentra el formulario, estando este estructurado en el interior de un elemento container actuando como layout, ver ilustración 61, dentro de este, se construye un widget llamado "_FormRecoveryChangePassword" que contendrá los elementos e inputs del formulario mostrados en la ilustración 60, se controla el tamaño de este contenedor con la ayuda de un widget MediaQuery, el cual nos permite obtener las dimensiones del dispositivo y en base a este, definir el tamaño que tendrá el componente especificado, dándole un comportamiento responsivo.

Ilustración 61. *Codificación contenedor del formulario*



```
class _ContainerForm extends StatelessWidget {
  const _ContainerForm({Key? key}) : super(key: key);
  @override
  Widget build(BuildContext context) {
    final Size size = MediaQuery.of(context).size;
    return Padding(
      padding:
          EdgeInsets.only(bottom: size.height * 0.18, top: size.height * 0.35),
      child: Center(
        child: Container(
          child: _FormRecoveryChangePassword(),
          width: size.width * 0.9,
          height: size.height * 0.4,
          decoration: BoxDecoration(
            color: AppColors.mainThirdContrast,
            borderRadius: BorderRadius.circular(20),
            boxShadow: [
              BoxShadow(
                offset: const Offset(0, 6.0),
                blurRadius: 5,
                color: Colors.black.withOpacity(0.3),
              ),
            ],
          ),
        ),
      ),
    );
  }
}
```

En su interior el widget "_FormRecoveryChangePassword" contiene elementos de layout que permiten posicionar y administrar el espacio que ocupan en pantalla los componentes, tal como el ButtonRecovery, el cual es un widget personalizado, construido para cumplir con la funcionalidad de ser el botón de recuperación de contraseña o el InputNewPassword y el InputNewConfirmPassword siendo widgets que componen los inputs de entrada de la nueva contraseña y confirmación de ella. El SizedBox es un widget de tipo layout que permite agregar un espaciado entre elementos implementado en el desarrollo para realizar la separación entre los inputs y el botón de "Actualizar Contraseña", estos elementos pueden ser observados en la ilustración 62.

Ilustración 62. *Codificación widget "_FormRecoveryChangePassword"*

```
class _FormRecoveryChangePassword extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    final Size size = MediaQuery.of(context).size;
    return SizedBox(
      width: size.width * 0.4,
      child: Column(
        children: [
          const _InputNewPassword(),
          const _InputNewConfirmPassword(),
          SizedBox(
            height: size.height * 0.03,
          ),
          ButtonRecovery(
              routeName: 'login',
              text: 'Actualizar Contraseña',
              navigator: 'until'
          )
        ],
      ),
    );
  }
}
```

Finalmente encontramos el widget "InputNewPassword", el cual en su interior contiene un widget de tipo TextFormField, siendo este un widget de tipo input, permitiéndole este al usuario introducir y manejar datos en la aplicación con ayuda del teclado en pantalla o del teclado en hardware según sea el caso.

El elemento input se customiza usando su propiedad decoration, donde se ingresan las diversas características que se requiere que contenga el campo de texto para ingresar una nueva contraseña, ver ilustración 59, su implementación en código puede ser visualizada en la ilustración 63.

Ilustración 63. *Codificación input nueva contraseña*

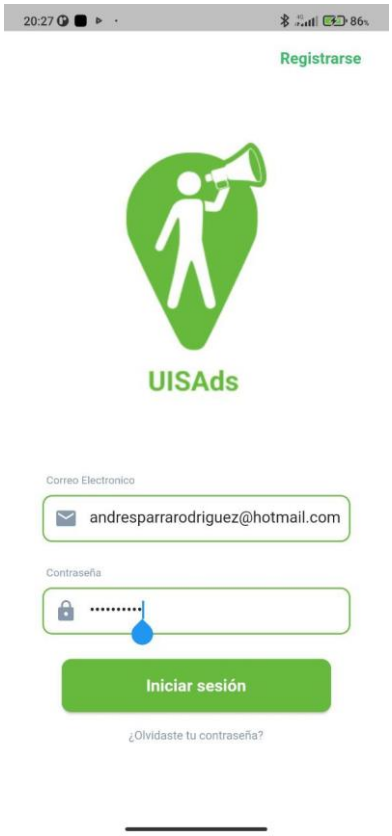
```
class _FormRecoveryChangePassword extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    final Size size = MediaQuery.of(context).size;
    return SizedBox(
      width: size.width * 0.4,
      child: Column(
        children: [
          const _InputNewPassword(),
          const _InputNewConfirmPassword(),
          SizedBox(
            height: size.height * 0.03,
          ),
          ButtonRecovery(
              routeName: 'login',
              text: 'Actualizar Contraseña',
              navigator: 'until'
          )
        ],
      ),
    );
  }
}
```

4.4.4 Despliegue del prototipo

En esta sección se muestran las vistas principales del desarrollo final dentro del aplicativo, con sus respectivas funcionalidades.

Ilustración 64. *Pantalla final registro de usuario e inicio de sesión*

Ilustración 65. *Pantalla final inicio de sesión*



En las ilustraciones 64 y 65, se hace referencia a las vistas de registro de usuario, autenticación e inicio de sesión respectivamente. A través de estas pantallas, los usuarios pueden acceder al aplicativo, esto mediante la creación de un perfil de usuario compuesto por sus datos personales con los cuales luego podrán iniciar sesión. De esa forma podrán tener acceso a las publicaciones de anuncios alojadas en la plataforma.

Ilustración 66. *Pantalla final creación de anuncio*

Ilustración 67. *Pantalla final visualización de anuncio*

En las ilustraciones 66 y 67, se observa la gestión de los anuncios dentro del aplicativo, en la ilustración 66 se observa la pantalla final del formulario para la creación de un anuncio, mientras tanto, en la ilustración 67, se contempla la visualización de un anuncio ya creado y publicado con éxito, presentando la información inherente al anuncio, junto con sus valoraciones recibidas.

Ilustración 68. *Pantalla final listado de previsualizaciones*

En la ilustración 68, se observan la pantalla de inicio con las diferentes previsualizaciones de los anuncios publicados y alojados dentro del aplicativo, desde esta sección general podemos ver todos los anuncios, y además de esto, desde esta vista es posible no solo filtrar los anuncios por las diferentes categorías definidas en el sistema si no también acceder a la opción de búsqueda por palabras claves.

Ilustración 69. *Pantalla final perfil de usuario*

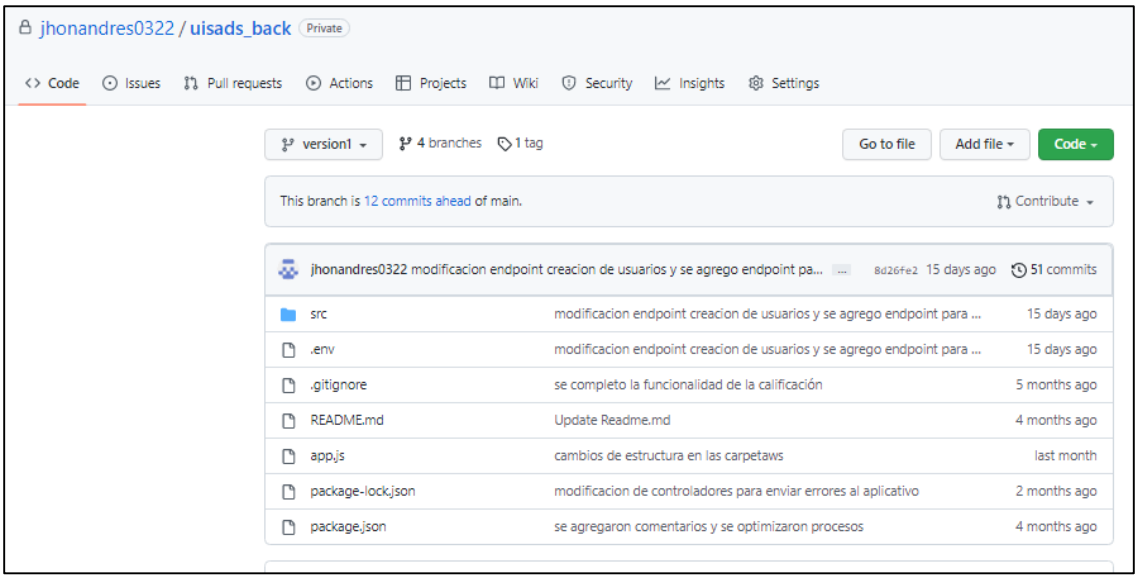
La pantalla presente en la ilustración 69 compone la visualización de un perfil de usuario, esta puede ser dividida en dos secciones principales, el área superior, compuesta por la información pública, mostrándonos correctamente el número de publicaciones que ha realizado, además de su puntaje como usuario, siendo este la suma total de las valoraciones positivas recibidas en sus anuncios menos el total de las valoraciones negativas, y junto con este valor, se encuentra la cifra que representa la cantidad de valoraciones totales que ha recibido el usuario en la totalidad de sus anuncios, y el área inferior, siendo conformada por el listado de las previsualizaciones de sus anuncios publicados, logrando verse por orden de publicación o por las mejor valoradas.

4.4.5 Trazabilidad

Se mantuvo la trazabilidad a través de la plataforma Github, desde la cual se mantuvieron dos repositorios, uno para el componente Backend, el cual puede ser visualizado en la ilustración 70 llamado “uisads_back” y otro para el componente Frontend, el cual puede ser visualizado en la ilustración 71 llamado “uisads_app”.

4.4.5.1 Componente backend

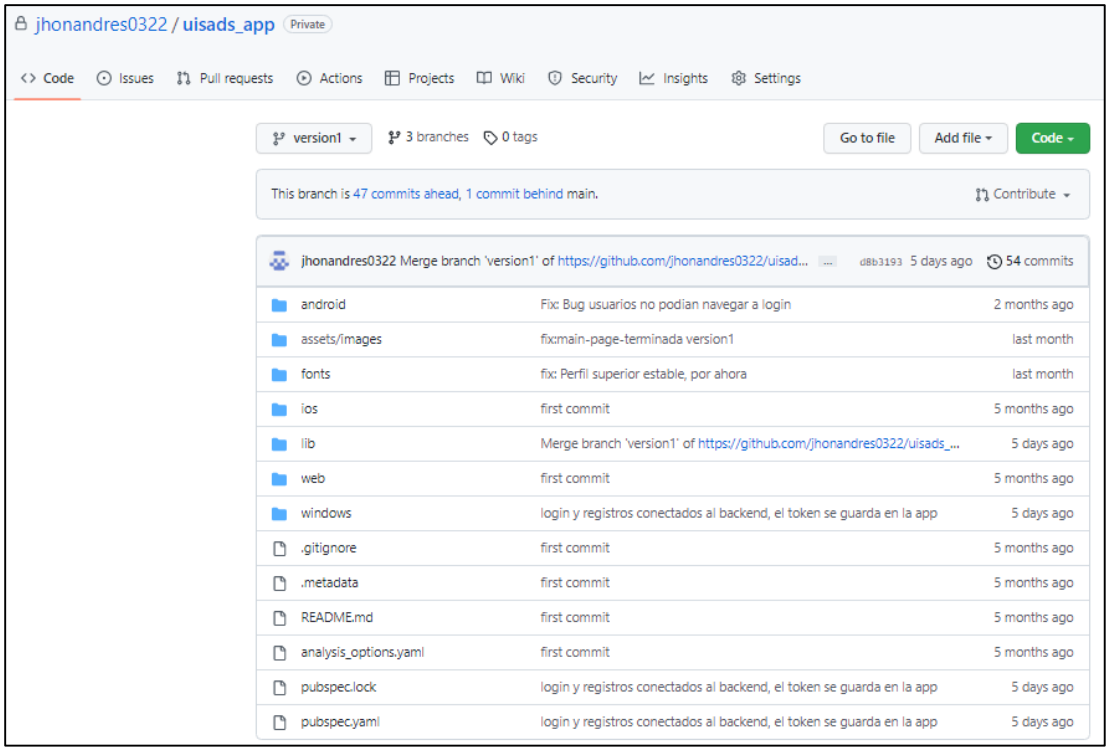
Ilustración 70. *Repositorio Github backend*



El componente backend del aplicativo se encuentra alojado en el repositorio “uisads_back” accesible mediante el siguiente enlace https://github.com/jhonandres0322/uisads_back.

4.4.5.2 Componente frontend

Ilustración 71. *Repositorio Github frontend*



El componente frontend del aplicativo se encuentra alojado en el repositorio “uisads_app” accesible mediante el siguiente enlace https://github.com/jhonandres0322/uisads_app.

5. Pruebas

Con el fin de validar el correcto cumplimiento de los requerimientos planteados se realizaron las siguientes pruebas en el aplicativo.

5.1 Pruebas de integridad

Tomando como base el requerimiento no funcional RNF04 se realizaron las siguientes pruebas buscando validar los requisitos funcionales del aplicativo.

Tabla 46. *Prueba registrar usuario*

Registrar usuario						
Email	correo1@gmail.com	-	-	-	-	-
Usuario	-	usuario1	-	-	-	-
Celular	-	-	3133333333	-	-	-
Ciudad	-	-	-	Bucaramanga	-	-
Contraseña	-	-	-	-	clave1	-
Respuesta	El campo usuario es requerido	El campo email es requerido	El campo email es requerido	El campo email es requerido	El campo email es requerido	El campo email es requerido
Resultado	Correcto	Correcto	Correcto	Correcto	Correcto	Correcto

Tabla 47. *Prueba iniciar sesión*

Iniciar sesión						
Email	-	correo1 @gmail.com	-	correoerroneo @gmail.com	correo1 @gmail.com	correo1 @gmail.com
Contraseña	clave1	-	-	clave1	claveerronea	clave1
Respuesta	El campo email es requerido	El campo contraseña es requerido	El campo email es requerido	No existe un cuenta registrada con este email	Contraseña incorrecta	Inicio de sesión correcto
Resultado	Correcto	Correcto	Correcto	Correcto	Correcto	Correcto

Tabla 48. *Prueba inicio de sesión incorrecto y reiterado*

Inicio de sesión	
Inicio de sesión	Inicio de sesión fallido en más de 5 ocasiones
Respuesta	El usuario se encuentra bloqueado
Resultado	Correcto

Tabla 49. *Prueba recuperar contraseña*

Recuperar contraseña			
Código de verificación	codigocorrecto	codigoerroneo	-
Respuesta	Código invalido	Código valido	Código invalido
Resultado	Correcto	Correcto	Correcto

Tabla 50. *Prueba cerrar sesión*

Cerrar sesión	
Opción "Cerrar sesión"	Seleccionar botón cerrar sesión
Respuesta	Destrucción de token de autenticación
Resultado	Correcto

Tabla 51. *Prueba crear anuncios*

Crear anuncios							
Titulo	Titulo anuncio	-	-	-	-	Titulo anuncio	Titulo anuncio
Descripción	-	Descripción anuncio	-	-	-	Descripción anuncio	Descripción anuncio
Imagen	-	-	Imagen 1	-	-	-	Imagen 1
Categoría	-	-	-	Categoría 1	-	Categoría 1	Categoría 1
Respuesta	El campo descripción es requerido	El campo título es requerido	El campo título es requerido	El campo título es requerido	El campo título es requerido	El anuncio debe tener una imagen o mas	Se ha guardado el anuncio con éxito
Resultado	Correcto	Correcto	Correcto	Correcto	Correcto	Correcto	Correcto

Tabla 52. *Prueba valorar anuncio*

Valorar anuncio			
Like	Seleccionado	-	-
Dislike	-	Seleccionado	-
Respuesta	*Valoración del anuncio ha aumentado en 1*	*Valoración del anuncio ha disminuido en 1*	*Ninguna valoración aportada al anuncio*
Resultado	Correcto	Correcto	Correcto

Tabla 53. *Prueba eliminar anuncio*

Eliminar anuncio	
Opción "Eliminar anuncio"	Seleccionar opción eliminar anuncio
Respuesta	¿Está seguro de eliminar este anuncio?
Resultado	Correcto

Tabla 54. *Prueba confirmar eliminar anuncio*

Eliminar anuncio	
Opción "Eliminar anuncio"	Seleccionar opción confirmar eliminar anuncio
Respuesta	*Anuncio eliminado*
Resultado	Correcto

Tabla 55. *Prueba contactar con anunciante*

Contactar con anunciante	
Botón "Whatsapp"	Presionar botón Whatsapp en anuncio o perfil del anunciante
Respuesta	*Redirección a chat de Whatsapp*
Resultado	Correcto

Tabla 56. *Prueba búsqueda de anuncios*

Búsqueda de anuncios			
Búsqueda	*Búsqueda sin resultados*	*Búsqueda menor a 4 caracteres*	*Búsqueda con resultados*
Respuesta	Publicaciones no encontradas	El texto de búsqueda debe ser mayor a 3 caracteres	*Los anuncios que cumplen con los resultados de la búsqueda son enseñados al usuario*
Resultado	Correcto	Correcto	Correcto

Tabla 57. *Prueba navegación por categorías*

Navegación por categorías		
Navegación	*No realizar navegación*	*Desplazarse a otra categoría*
Respuesta	*Categoría por defecto: Variados*	*Son mostrados anuncios solo pertenecientes a la categoría seleccionada*
Resultado	Correcto	Correcto

Tabla 58. *Prueba búsqueda filtrada por relevancia*

Búsqueda filtrada por relevancia		
Filtrado	Anuncios más recientes	Anuncios mejor votados
Respuesta	*Son mostrados anuncios organizados por fecha de publicación, opción por defecto*	*Son mostrados anuncios organizados por cantidad de valoraciones positivas de forma descendente*
Resultado	Correcto	Correcto

Tabla 59. *Prueba búsqueda filtrada por antigüedad de publicación*

Búsqueda filtrada por antigüedad de publicación					
Antigüedad	24 horas	Una semana	Un mes	Un año	Sin limite
Respuesta	*Solo son mostrados los anuncios publicados las últimas 24 horas con respecto al momento actual*	*Solo son mostrados los anuncios publicados las últimas 168 horas con respecto al momento actual*	*Solo son mostrados los anuncios publicados las últimas 720 horas con respecto al momento actual*	*Solo son mostrados los anuncios publicados las últimas 8760 horas con respecto al momento actual*	*Son mostrados todos los anuncios sin importar su fecha de publicación*
Resultado	Correcto	Correcto	Correcto	Correcto	Correcto

Tabla 60. *Prueba cambio de contraseña*

Cambio de contraseña						
Antigua contraseña	*Contraseña antigua correcta*	*Contraseña antigua incorrecta*	*Contraseña antigua correcta*	*Contraseña antigua correcta*	*Contraseña antigua incorrecta*	-
Nueva contraseña	*Contraseña nueva valida*	*Contraseña nueva valida*	*Contraseña nueva menor a 4 caracteres*	*Contraseña nueva valida*	*Contraseña nueva menor a 4 caracteres*	-
Confirmar contraseña	*Contraseña de confirmación igual a nueva contraseña*	*Contraseña de confirmación igual a nueva contraseña*	*Contraseña de confirmación igual a nueva contraseña*	*Contraseña de confirmación diferente a nueva contraseña*	*Contraseña de confirmación diferente a nueva contraseña*	-
Respuesta	Contraseña actualizada	Contraseña antigua incorrecta	Contraseña nueva muy corta, debe ser mayor a 3 caracteres	Las contraseñas no coinciden	Al menos dos de los campos ingresados no son validos	Al menos dos de los campos ingresados no son validos
Resultado	Correcto	Correcto	Correcto	Correcto	Correcto	Correcto

Tabla 61. *Prueba edición descripción perfil personal*

Edición descripción perfil personal		
Descripción perfil	Descripción de prueba.	-
Respuesta	*Descripción del perfil actualizada*	*Descripción del perfil actualizada*
Resultado	Correcto	Correcto

Tabla 62. *Prueba cambiar imagen de perfil*

Cambiar imagen de perfil		
Imagen de perfil	*Seleccionar nueva imagen de perfil*	*Cancelar cambio de imagen de perfil*
Respuesta	*Imagen de perfil actualizada*	*Imagen de perfil no es actualizada*
Resultado	Correcto	Correcto

Tabla 63. *Prueba visualizar datos de la app*

Visualizar datos de la app	
Botón "Visualizar datos de la app"	Presionar botón visualizar datos de la app
Respuesta	*Es mostrado un breve resumen de la misión del aplicativo junto con el nombre de sus desarrolladores*
Resultado	Correcto

5.2 Verificación de requerimientos

5.2.1 Verificación de requerimientos funcionales

Requerimiento	Implementación	Estado
RF1	Funcionalidad implementada correctamente en la pantalla de crear cuenta	OK
RF2	Funcionalidad implementada correctamente en la pantalla login	OK
RF3	Funcionalidad implementada correctamente en las pantallas de login y recuperación de contraseña	OK
RF4	Funcionalidad implementada correctamente en las pantallas de edición de perfil de usuario y edición de contraseña	OK

RF5	Funcionalidad implementada correctamente en la pantalla de edición de perfil de usuario	OK
RF6	Funcionalidad implementada correctamente en la pantalla de menú lateral	OK
RF7	Funcionalidad implementada correctamente en la pantalla de creación de anuncios, accesible desde la barra de navegación inferior de la aplicación	OK
RF8	Funcionalidad implementada correctamente al ingresar a las diferentes previsualizaciones	OK
RF9	Funcionalidad implementada correctamente en sus tres casos, en la pantalla de anuncio	OK
RF10	Funcionalidad implementada correctamente en la pantalla de búsqueda como uno de los filtros de búsqueda disponibles y a través de las diferentes categorías dentro del aplicativo	OK
RF11	Funcionalidad implementada correctamente como un filtro de búsqueda accesible en la pantalla de búsqueda y desplegada en una ventana emergente	OK
RF12	Funcionalidad implementada correctamente como un filtro de búsqueda accesible en la pantalla de búsqueda y desplegada en una ventana emergente	OK
RF13	Funcionalidad implementada correctamente permitiendo la edición de anuncios propios desde el propio perfil del usuario	OK

RF14	Funcionalidad implementada correctamente permitiendo la eliminación de anuncios desde el propio perfil del usuario	OK
RF15	Funcionalidad implementada correctamente ubicando las previsualizaciones en las diferentes categorías dentro del aplicativo, conteniendo todas las características propuestas	OK
RF16	Funcionalidad implementada correctamente existiendo una pantalla principal y general que contiene todos los anuncios publicados	OK
RF17	Funcionalidad implementada correctamente como un botón flotante en la vista de cada anuncio y en los perfiles de los anunciantes	OK
RF18	Funcionalidad implementada correctamente en la pantalla de perfil de usuario, compuesto con los datos propuestos en el requerimiento	OK
RF19	Funcionalidad implementada correctamente en la pantalla de edición perfil de usuario	OK
RF20	Funcionalidad implementada correctamente como un botón accesible desde la pantalla de edición perfil de usuario	OK
RF21	Funcionalidad implementada correctamente en la pantalla de perfil del usuario	OK
RF22	Funcionalidad implementada correctamente como un botón en la barra de navegación inferior	OK

RF23	Funcionalidad implementada correctamente como un botón superior que aparece en la pantalla según el caso	OK
RF24	Funcionalidad implementada correctamente en la pantalla de búsqueda la cual es accesible mediante un botón superior de búsqueda en la pantalla principal de anuncios	OK
RF25	Funcionalidad implementada correctamente en la pantalla de perfil de cada usuario, accesible mediante un botón superior, limitando la búsqueda a solo el perfil donde se encuentre el usuario	OK
RF26	Funcionalidad implementada correctamente en la pantalla del menú lateral como una opción del menú	OK

5.2.2 Verificación requerimientos no funcionales

Requerimiento	Implementación	Estado
RNF1	Funcionalidad implementada al limitar opciones de ciudad en la pantalla de creación de una cuenta	OK
RNF2	Funcionalidad implementada	OK
RNF3	Funcionalidad implementada al estar soportado el aplicativo en un servidor que garantiza el funcionamiento continuo	OK
RNF4	Funcionalidad implementada al corroborar que los tipos de datos ingresados sean validos	OK

RNF5	Funcionalidad implementada en el desarrollo del aplicativo otorgándole compatibilidad con dispositivos Android superiores a la versión 5.0	OK

5. Observaciones y restricciones

- El aplicativo está dirigido a personas cercanas o pertenecientes a la comunidad de la Universidad Industrial de Santander, por tal motivo el alcance del aplicativo está limitado a Bucaramanga y sus alrededores, connotando que su uso no está limitado a solo estudiantes UIS.
- La aplicación fue diseñada para ser usada en dispositivos Android con versiones mayores a Android 5.0 (Lollipop), por esta razón el aplicativo no está disponible para versiones inferiores de Android o dispositivos Apple.
- El aplicativo busca ser solo un mediador entre usuarios interesados en un bien y los anunciantes del mismo, por este motivo el aplicativo no permite pagos o compras dentro de la app.
- Los anuncios del aplicativo no permiten la carga de videos o demás material multimedia diferente a fotos e imágenes.

6. Encuesta

Para validar el funcionamiento del aplicativo se entregó la aplicación a un público de prueba de 15 personas, las cuales usaron el aplicativo por un periodo de una semana, luego de esto se les otorgo una encuesta cuestionando la validación de los principales objetivos de la aplicación, además de posibles puntos de mejora.

6.1 Preguntas

Las preguntas realizadas en la encuesta fueron las siguientes:

1. ¿Considera que el aplicativo facilita la difusión de anuncios entre los miembros de la comunidad universitaria? (Si/No)

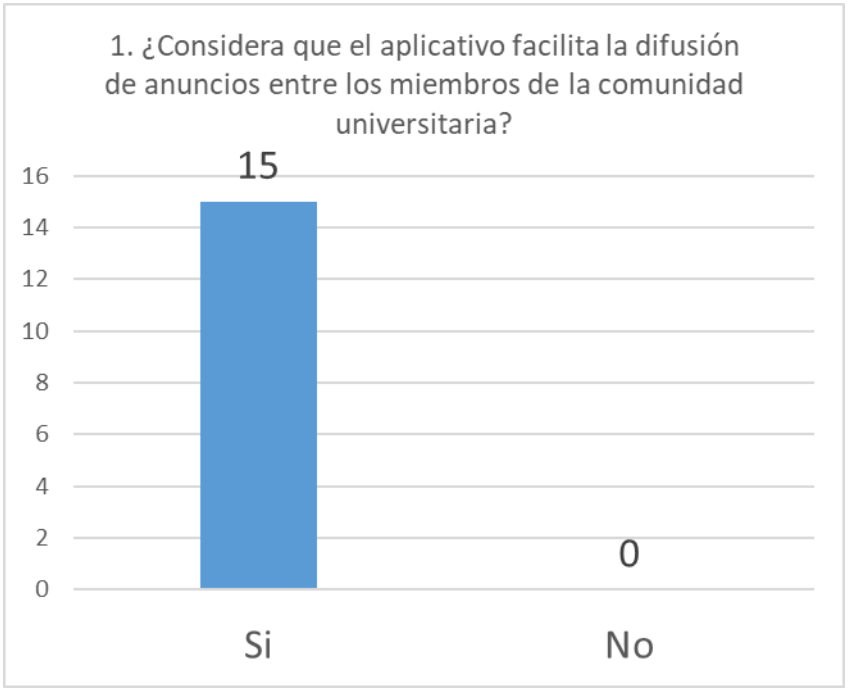
2. En una escala del 0 al 5, ¿Qué tan fácil le fue usar la opción de búsqueda incorporada dentro del aplicativo? (Donde 0 es le fue imposible de usar y 5 le fue muy fácil de usar)
3. ¿Considera útiles y relevantes los filtros de búsqueda incorporados en el aplicativo? (Si/No)
4. En una escala del 0 al 5, ¿Qué tan fácil le fue entender y usar el sistema del aplicativo? (Donde 0 es le fue imposible de usar y 5 le fue muy fácil de usar)
5. En una escala del 0 al 5, ¿Qué tan cómodo considera el registro e inicio de sesión en el aplicativo? (Donde 0 es le fue muy incómodo y 5 le fue muy cómodo)
6. En una escala del 0 al 5, ¿Qué tan útil vería en el aplicativo, un sistema de notificaciones basado en sus temas de interés?
7. En una escala del 0 al 5, ¿Qué tanto le gustaría que los usuarios al interior del aplicativo tuvieran una opción de reporte de anuncios no deseados o inadecuados?
8. ¿Vería útil incorporar al aplicativo métodos de acceso alternativos y más rápidos, como autenticación mediante cuenta de Google o Facebook? (Si/No)
9. ¿Vería útil incorporar al aplicativo la opción almacenar anuncios para visualizarlos en otro momento? (Si/No)
10. ¿Vería útil incorporar al aplicativo un historial de las publicaciones que ha visualizado? (Si/No)

6.2 Resultados

Fue presentada la anterior encuesta a un público de 15 personas, todos estos estudiantes UIS, los cuales luego de usar y examinar el aplicativo por un periodo de 5 días, dieron respuesta a cada una de las preguntas del sondeo, recolectando los siguientes resultados.

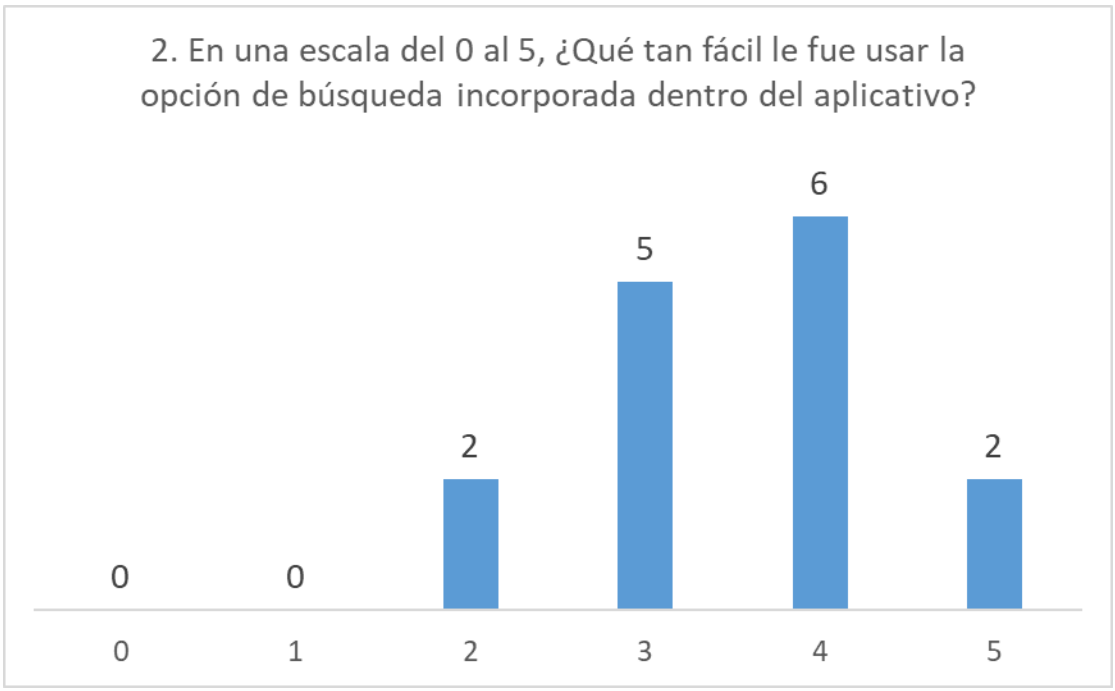
6.2.1 Resultados pregunta 1

Ilustración 72. *Resultados encuesta, pregunta 1*



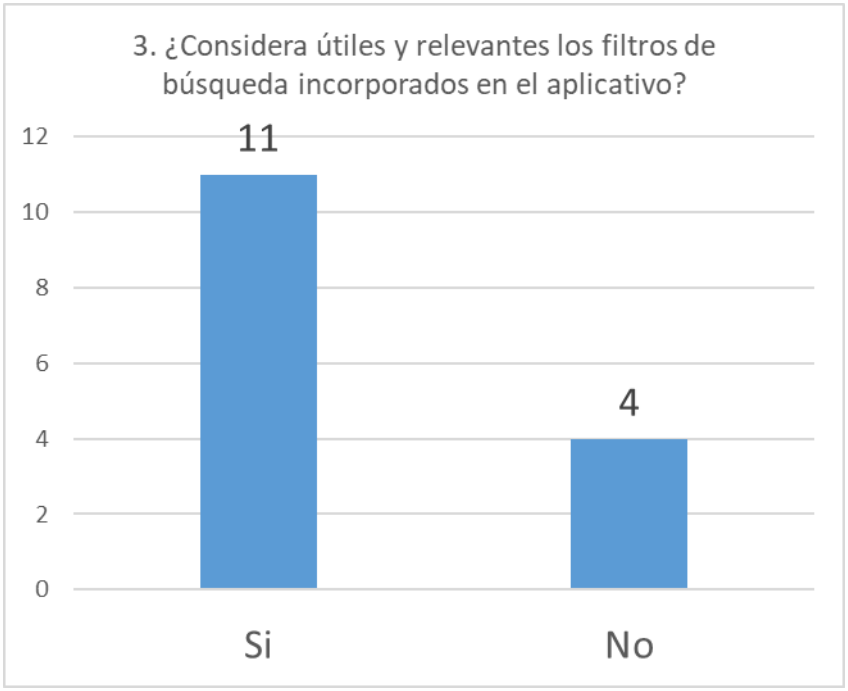
6.2.2 Resultados pregunta 2

Ilustración 73. *Resultados encuesta, pregunta 2*



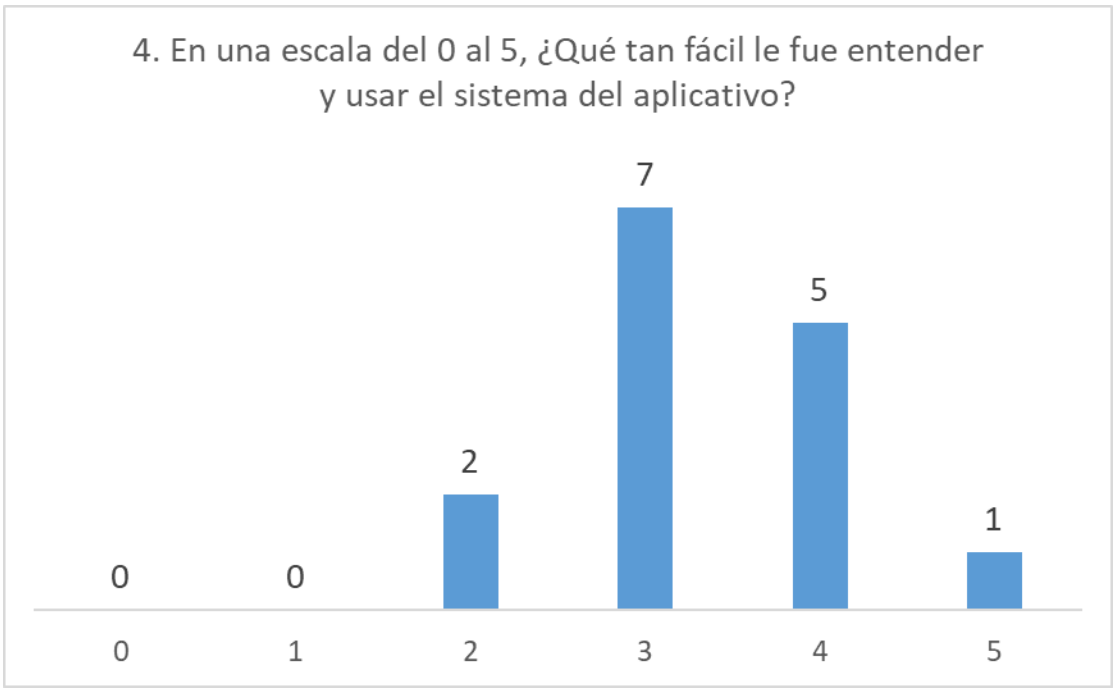
6.2.3 Resultados pregunta 3

Ilustración 74. *Resultados encuesta, pregunta 3*



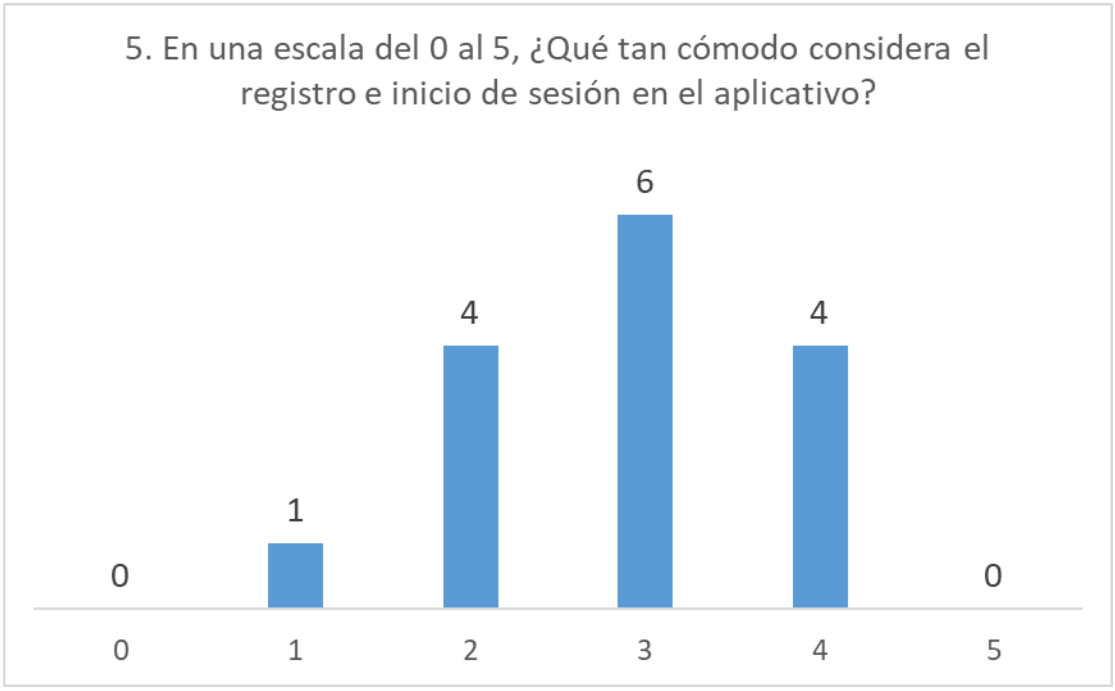
6.2.4 Resultados pregunta 4

Ilustración 75. *Resultados encuesta, pregunta 4*



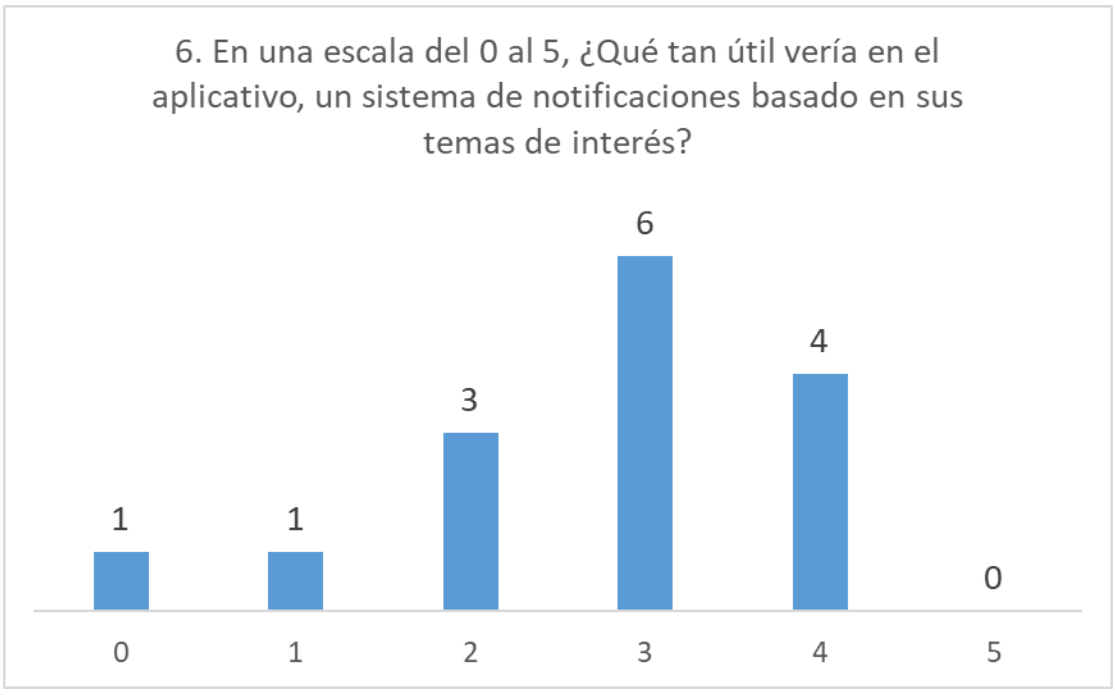
6.2.5 Resultados pregunta 5

Ilustración 76. *Resultados encuesta, pregunta 5*



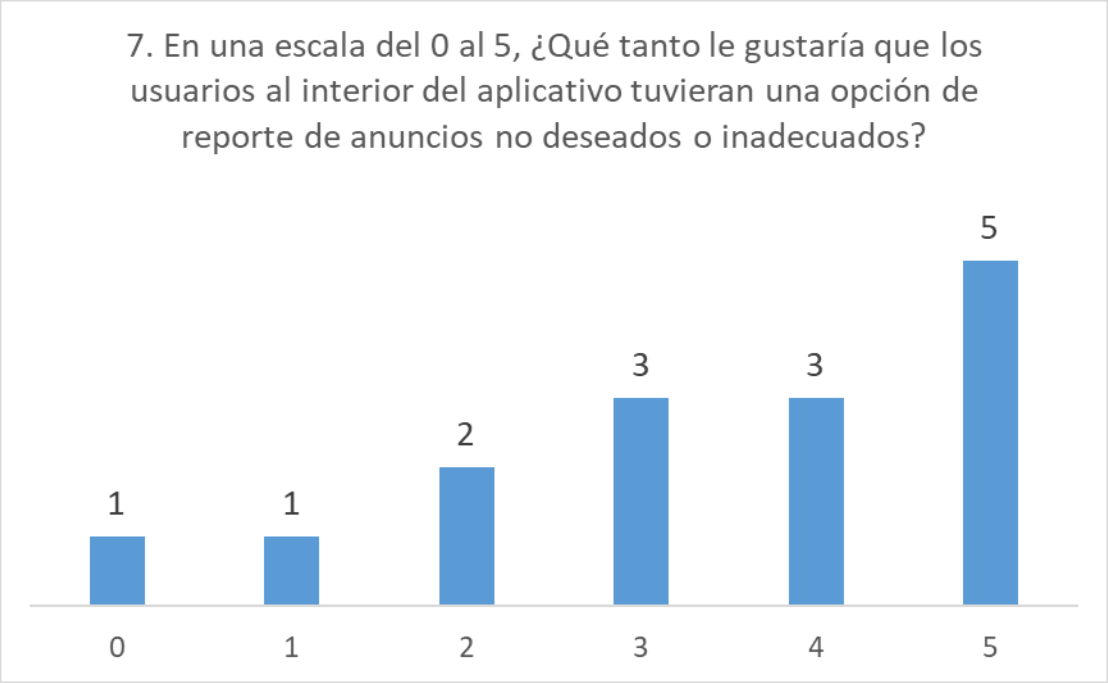
6.2.6 Resultados pregunta 6

Ilustración 77. *Resultados encuesta, pregunta 6*



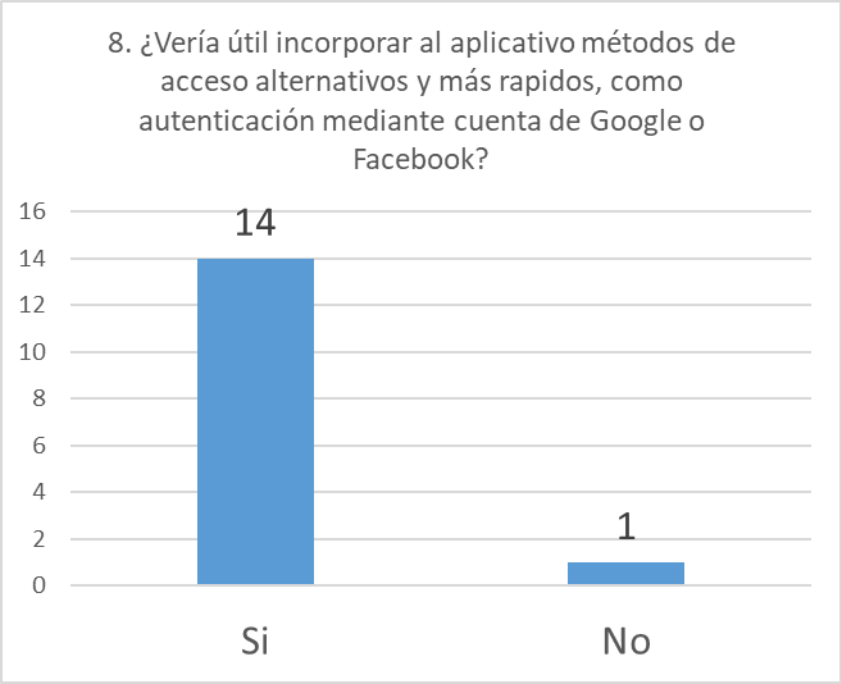
6.2.7 Resultados pregunta 7

Ilustración 78. *Resultados encuesta, pregunta 7*



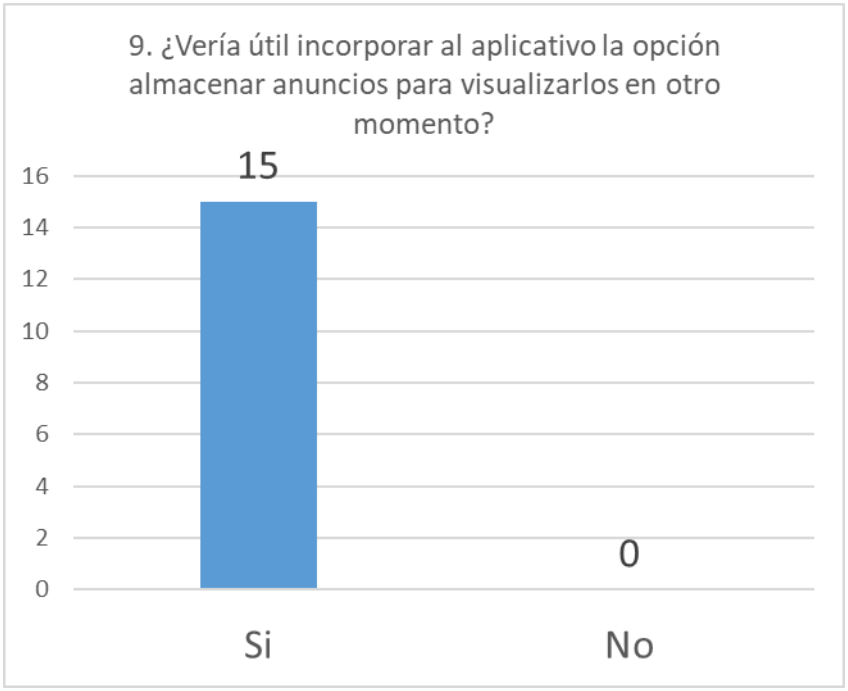
6.2.8 Resultados pregunta 8

Ilustración 79. *Resultados encuesta, pregunta 8*



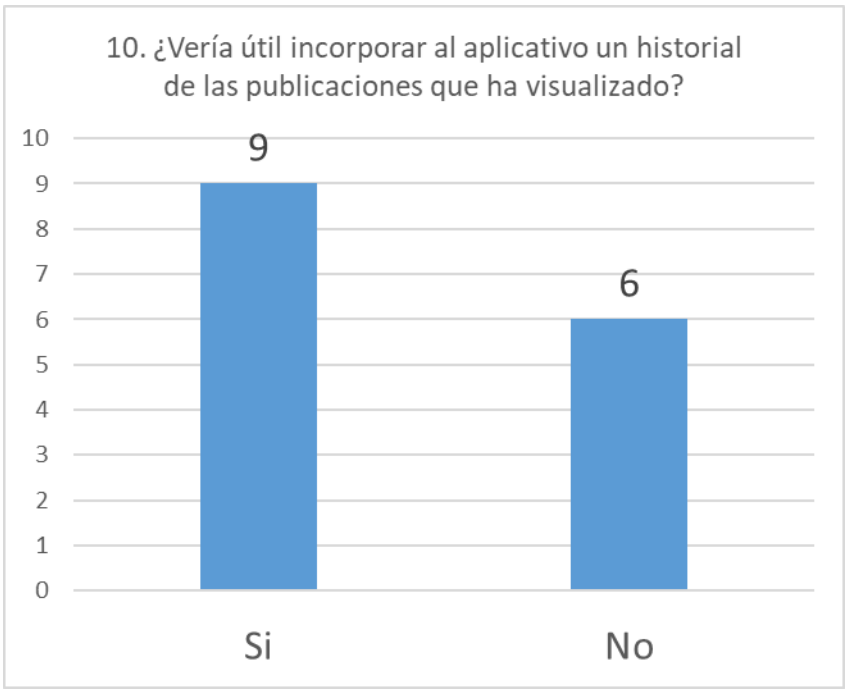
6.2.9 Resultados pregunta 9

Ilustración 80. *Resultados encuesta, pregunta 9*



6.2.10 Resultados pregunta 10

Ilustración 81. *Resultados encuesta, pregunta 10*



7. Recomendaciones

Con base a los resultados obtenidos en la encuesta y revisando los posibles puntos de mejora del proyecto se proyectaron posibles mejoras que pueden ser implementadas al aplicativo en un posible nuevo proyecto que apunte a futura nueva versión, se proponen las siguientes mejoras:

- Implementar nuevos métodos de registro de cuentas que sean más atractivos y rápidos para los usuarios, esto usando plataformas que actúen como mecanismo de autenticación como Facebook o Google.
- Implementar un acceso libre pero limitado a la plataforma, donde personas del común puedan acceder sin la necesidad de registrar una cuenta de usuario, limitando sus acciones en la plataforma, esto con miras a que una persona que conoce la plataforma por primera vez puedan familiarizarse y observar anuncios con rapidez.
- Extender el alcance del aplicativo, enfocando su uso no solo a miembros de la comunidad universitaria residentes del área metropolitana de Bucaramanga.
- Dotar al aplicativo de la capacidad de conocer la ubicación del dispositivo al momento de realizar la publicación de un anuncio.
- Añadir un nuevo filtro de búsqueda que permita la visualización de anuncios en un área cercana al usuario determinada por él, esto basándose en el lugar de publicación de los anuncios.
- Añadir al aplicativo un campo donde los usuarios puedan inscribir sus temas de interés, esto con miras a la creación de un sistema de alertas mediante notificaciones, los cuales mantengan al tanto al usuario de nuevas publicaciones que cumplan con sus intereses.

- Añadir al aplicativo la opción y funcionalidad de reportar y eliminar anuncios considerados como inadecuados.
- Otorgar al aplicativo la capacidad de contabilizar la cantidad de usuarios que ha visualizado un anuncio.
- Incluir en el aplicativo la capacidad de almacenar un historial de anuncios visualizados por cada uno de los usuarios.
- Añadir dentro de las opciones de un usuario en el aplicativo, la funcionalidad de guardar anuncios para su posterior visualización.
- Implementar un límite de anuncios que pueden publicados por un usuario en un periodo determinado de tiempo.

8. Conclusiones

Desde las bases del presente proyecto, se consideró de gran importancia conectar a la comunidad universitaria a través de una plataforma móvil, plataforma que les diera un espacio a los miembros de la comunidad, donde lograran, de una manera asertiva y simple, la publicación de anuncios, con los cuales puedan poner a disposición del público sus bienes o servicios, esto en un espacio que presente una estructura ordenada, accesible y amigable con el usuario. Siguiendo este enfoque, se logró cumplir correctamente los objetivos del presente proyecto, logrando establecer una plataforma de anuncios, enfocada a la comunidad UIS, que permitiera el registro de usuarios, la gestión y visualización de anuncios y el manejo de perfiles de usuario, logrando ser un medio facilitador para la interacción de la comunidad. Se consiguió establecer unas bases sólidas en el diseño, que apuntaran a ser amigables con el usuario, tanto en su estructura lógica de software como en sus diferentes interfaces expuestas en el aplicativo, bases que fueron vitales en el desarrollo del proyecto. Fue lograda la incorporación de una forma sencilla y directa de comunicación entre ambas partes dentro del aplicativo, esto con ayuda de la plataforma de mensajería Whatsapp, la cual, debido a su amplio uso en la sociedad, es la herramienta ideal para ser un canal de comunicación. Se logró establecer todo un abanico de categorías debidamente pensadas, según las necesidades detectadas en la comunidad universitaria, y además de esto, el aplicativo se dotó a cabalidad con herramientas de búsqueda, que permitieran al usuario navegar entre los anuncios con exactitud. Se logró establecer un sistema de publicación de anuncios que fuera realmente cómodo para un usuario, siendo amigable desde su localización en pantalla, donde es posible acceder a esta opción desde una barra de navegación inferior, presente tanto desde el primer momento de acceso a la plataforma, como desde un botón flotante que está presente en la navegación entre previsualizaciones, buscando ser intuitiva hasta en la forma creación de un nuevo

anuncio, donde con su diseño, es fácil de entender, para un usuario, los datos mínimos requeridos que conforman un anuncio. Se logró enfocar el proyecto en las diferentes necesidades detectadas en los miembros de la comunidad universitaria, manteniendo una filosofía democrática, donde es la misma comunidad quien puede administrar lo que ve, haciendo que esta pueda valorar cada uno de los anuncios presentes y dictaminar si un anuncio es relevante para la misma comunidad o no, a su vez, se logró establecer un espacio que los usuarios puedan sentir como propio, donde cada usuario pueda tener sus publicaciones y pueda ver su información individual, además, donde el usuario pueda darle su toque personal, a través de la redacción de una descripción de su perfil. Se logró validar el buen funcionamiento del sistema y la completa implementación de todos sus requerimientos, esto con la ejecución de diferentes pruebas, buscando posibles fallos en el sistema, siendo al final los resultados de estas pruebas exitosos, así mismo fue realizada una encuesta, donde las personas podían manifestar su opinión ante la plataforma, validando los objetivos de manera positiva.

La realización de este proyecto, fue posible gracias a las buenas prácticas en el desarrollo del software, donde es necesario resaltar la utilidad del modelo-vista-controlador, dotando al aplicativo con la posibilidad de escalar a una nueva versión en un futuro proyecto, también es importante resaltar el software de código abierto, en especial las herramientas Express y Flutter, las cuales fueron piezas importantes, en las partes backend y frontend de la realización del proyecto respectivamente, sin las cuales no hubiera sido posible el desarrollo del mismo. Finalmente es necesario resaltar la necesidad, por parte del campus, de impulsar y promover iniciativas o emprendimientos, al interior de proyectos de grado, que busquen el mejoramiento de la calidad de vida de la comunidad universitaria.

Referencias bibliográficas

Amazon Web Services, Inc. “What is Mobile Application Development?”. Amazon. Obtenido de https://aws.amazon.com/es/mobile/mobile-application-development/.

Arun Sundararajan. (2016). The Sharing Economy: The End of Employment and the Rise of Crowd-Based Capitalism. MIT Press. pp. 29–30. ISBN 9780262034579.

Bloomberg. (2008). “Determining where to sell online”. Bloomberg. Obtenido de https://www.bloomberg.com/news/articles/2008-11-07/determining-where-to-sell-onlinebusinessweek-business-news-stock-market-and-financial-advice.

Bremen University. “Waterfall”. Mathematics and Computer Science. Obtenido de http://www.informatik.uni-bremen.de/uniform/vm97/def/def_w/WATERFALL.htm.

Businessofapps. (2020). “Facebook statistics”. Business Of Apps. Obtenido de https://www.businessofapps.com/data/facebook-statistics/.

Cegos. (2021). “Wallapop, la historia de una startup Española de éxito”. Cegos Online University. Obtenido de https://www.cegosonlineuniversity.com/wallapop-la-historia-de-una-startup-espanola-de-exito/.

Cnet. (2018). “Facebook Marketplace is used in 70 countries, by 800 million people monthly”. Cnet. Obtenido de https://www.cnet.com/news/facebook-marketplace-is-used-in-70-countries-by-800-million-people-monthly/.

Cocolabs SAS. (2021). “Cocorico is an open source marketplace solution for services and rentals”. Github. Obtenido de https://github.com/Cocolabs-SAS/cocorico.

Codagnone, Cristiano & Karatzogianni, Athina & Matthews, Jacob (2018). Platform Economics: Rhetoric and Reality in the "Sharing Economy". Emerald Group Publishing. pp. 51–52. ISBN 9781787438101.

Computerhoy. (2018). “Android vs iPhone: la guerra de los smartphones en cifras”. computerhoy.com. Obtenido de https://computerhoy.com/reportajes/industria/android-vs-iphone-guerra-smartphones-cifras-271447.

Danna, Sanny (2002). “Classified Advertising”. The Advertising Age Encyclopedia of Advertising. New York: Routledge. p. 325. ISBN 9781135949068.

Datademia. (2020). “¿Qué es MongoDB?”. Datademia. Obtenido de https://datademia.es/blog/que-es-mongodb.

Easyappcode. (2020). “Patrón de diseño MVC. ¿Qué es y cómo puedo utilizarlo?”. Easyappcode. Obtenido de https://www.easyappcode.com/patron-de-diseno-mvc-que-es-y-como-puedo-utilizarlo.

eBay Inc. (2020). eBay. Obtenido de https://www.ebay.com/.

Elogia. (2017). Elogia. Obtenido de https://elogia.net/.

Elías Maria. (2015). “Wallapop, la startup española que pasó de una habitación 'cutre' a querer competir con eBay en EEUU”. Voz Pópuli. Obtenido de http://web.archive.org/web/20160716121148/http://vozpopuli.com:80/economia-y-finanzas/69281-wallapop-la-startup-espanola-que-paso-de-una-habitacion-cutre-a-querer-competir-con-ebay-en-eeuu.

Finanzasparamortales. (2017). “La economía colaborativa, un fenómeno imparable”. Finanzas para Mortales. Obtenido de https://finanzasparamortales.es/la-economia-colaborativa-un-fenomeno-imparable/.

Flutter. (2022). “Build apps for any screen”. Flutter.dev. Obtenido de https://flutter.dev/.

Forbes. (2014). “Why online marketplaces are booming”. Forbes. Obtenido de https://www.forbes.com/sites/groupthink/2014/08/20/why-online-marketplaces-are-booming/?sh=42b1f27f531d.

Funes Ana, (2019). “Cómo funciona Wallapop, la app para vender de forma fácil y rápida”. El Español. Obtenido de https://www.elespanol.com/como/funciona-wallapop-app-vender-forma-facil-rapida/435456727_0.html.

Galtés Mar. (2017). “Wallapop: el aterrizaje del unicornio”. La Vanguardia. Obtenido de https://www.lavanguardia.com/economia/emprendedores/20170625/423671015279/wallapop-agustin-gomez.html.

Hamari, Juho & Sjöklint, Mimmi & Ukkonen, Antti. (2016). “The Sharing Economy: Why People Participate in Collaborative Consumption”. Journal of the Association for Information Science and Technology. Obtenido de https://www.researchgate.net/publication/255698095.

Hdez Justo. (2017). “Tus compras de Wallapop ahora con envío mediante Correos”. El Español. Obtenido de https://www.elespanol.com/elandroidelibre/noticias-y-novedades/20170914/compras-wallapop-ahora-envio-mediante-correos/246726586_0.amp.html.

Hindle, T. (2009). “The Economist Guide to Management Ideas and Gurus”. Economist. Obtenido de https://www.economist.com/news/2009/10/08/e-commerce.

Hughey Douglas. (2009). “The Traditional Waterfall Approach”. University of Missouri–St. Louis. Obtenido de https://www.umsl.edu/~hugheyd/is6840/waterfall.html.

IAB Spain. (2017). IAB Spain. Obtenido de https://iabspain.es/.

In Lee. (2016). Encyclopedia of E-Commerce Development, Implementation, and Management. IGI Global. ISBN 9781466697881.

Jashkenas. (2022). “List of languages that compile to JS”. Github. Obtenido de https://github.com/jashkenas/coffeescript/wiki/list-of-languages-that-compile-to-js.

K, Pradeep. (2020). “What is the difference between stateless and stateful widgets?”. Devops School. Obtenido de https://www.devopsschool.com/blog/what-difference-between-stateless-and-stateful-widgets-flutter/.

Kerby Davis. (2015). “Why MongoDB is the Way to Go”. DZone. Obtenido de https://dzone.com/articles/why-mongodb-is-worth-choosing-find-reasons.

Lorimor, E. S. (1977). Classified advertising: a neglected medium. Journal of Advertising, 6(1), 17-25.

Lvivity. (2018). “Web-Based Application: What It Is, and Why You Should Use It”. Lvivity. Obtenido de https://lvivity.com/web-based-applications.

Marketplacepulse. (2020). “Mercado Libre Statistics”. Market Place Pulse. Obtenido de https://www.marketplacepulse.com/stats/mercadolibre.

Mercado libre. (2020). “Historia de mercado libre”. Mercadolibre. Obtenido de https://ideas.mercadolibre.com/ar/noticias/historia-de-mercado-libre/.

Metz, Cade. (2016). “Forget Apple vs. the FBI: WhatsApp Just Switched on Encryption for a Billion People”. Wired. Obtenido de https://www.wired.com/2016/04/forget-apple-vs-fbi-whatsapp-just-switched-encryption-billion-people/.

Oberlo. (2020). “10 eBay Statistics You Need to Know in 2020”. Oberlo. Obtenido de https://www.oberlo.com/blog/ebay-statistics.

Orson Parmy. (2015). “Facebook's Phone Company: WhatsApp Goes To The Next Level With Its Voice Calling Service”. Forbes. Obtenido de https://www.forbes.com/sites/parmyolson/2015/04/07/facebooks-whatsapp-voice-calling/?sh=430b43321388.

Petersen, Kai; Wohlin, Claes; Baca, Dejan. (2009). Bomarius, Frank; Oivo, Markku; Jaring, Päivi; Abrahamsson, Pekka (eds.). “The Waterfall Model in Large-Scale Development” Product-Focused Software Process Improvement. Lecture Notes in Business Information Processing. ISBN 978-3-642-02152-7. Obtenido de https://link.springer.com/chapter/10.1007/978-3-642-02152-7_29.

R. Eckstein. (2007). “Java SE Application Design With MVC”. Oracle. Obtenido de https://www.oracle.com/technical-resources/articles/javase/application-design-with-mvc.html.

RAE. (2020). “Definición de comercio electrónico - Diccionario panhispánico del español jurídico – RAE”. Diccionario panhispánico del español jurídico - Real Academia Española. Obtenido de https://dpej.rae.es/lema/comercio-electrónico.

RedHat. (2020). “What are microservices?”. RedHat. Obtenido de https://www.redhat.com/es/topics/cloud-native-apps/what-is-an-application-architecture.

Rodríguez. (2018). “Agustín Gómez, CEO de Wallapop: “Cuando empezamos creíamos que íbamos a ser unos losers””. Forbes. Obtenido de https://forbes.es/emprendedores/41613/agustin-gomez-ceo-de-wallapop-cuando-empezamos-creiamos-que-ibamos-a-ser-unos-losers/.

Santander. (2021). “La economia colaborativa: ¿qué es y qué nos puede aportar?”. Santander. Obtenido de https://www.santander.com/es/stories/la-economia-colaborativa-que-es-y-que-nos-puede-aportar.

Semrush. (2020). Semrush. Obtenido de https://es.semrush.com/blog/que-es-marketplace-ventajas-inconvenientes/.

Servimedia. (2021). “Wallapop logra 157 millones de euros de financiación y eleva su valor a 690 millones”. Voz Pópuli. Obtenido de http://webcache.googleusercontent.com/search?q=cache:https://www.vozpopuli.com/economia_y_finanzas/wallapop-beneficios-financiacion.html%3famp=1.

Sharetribe. (2022). “Sharetribe Go is a source available marketplace software”. Github. Obtenido de https://github.com/sharetribe/sharetribe.

Teixeira Pedro. (2012). “Professional Node.js: Building Javascript Based Scalable Software”. John Wiley & Sons. ISBN 9781118185469.

Viallon François Joseph. (2017). “Hybrid apps: an overview of advantages, limitations & consequences for your testing phases”. stardust-testing.com. Obtenido de https://www2.stardust-testing.com/en/blog-en/hybrid-apps.

Víctor Pérez Pérez. Universidad Politécnica de Valencia. Obtenido de https://riunet.upv.es/bitstream/handle/10251/10273/Memoria.pdf.

Wallapop. (2013). “Gana dinero y encuentra oportunidades cerca de ti en wallapop”. es.wallapop.com. Obtenido de https://es.wallapop.com/.

Way2ecommerce. (2015). “E-commerce ¿Que es el e-commerce?”. Way2ecommerce. Obtenido de https://way2ecommerce.com/e-commerce-que-es.

Wells, William & Moriarty, Sandra & Burnett, John (2006). Advertising: Principles and Practice (7th Ed.). New Jersey: Prentice Hall. p. 217. ISBN 9780131465602.

WhatsApp.com. (2020). “WhatsApp Blog”. WhatsApp.com. Obtenido de https://blog.whatsapp.com/10000666/Two-Billion-Users--Connecting-the-World-Privately.

WhatsApp Inc. (2016). “WhatsApp Encryption Overview – Technical white paper”. WhatsApp Inc. Obtenido de https://www.whatsapp.com/security/WhatsApp-Security-Whitepaper.pdf.

Williams Alex. (2012). “GitHub Pours Energies into Enterprise – Raises $100 Million From Power VC Andreessen Horowitz”. Tech Crunch. Obtenido de https://techcrunch.com/2012/07/09/github-pours-energies-into-enterprise-raises-100-million-from-power-vc-andreesen-horowitz/.