Universidad de Alcalá
Escuela Politécnica Superior

GRADO EN INGENIERÍA INFORMÁTICA

Trabajo Fin de Grado

Desarrollo en Kotlin para la creación de una red social

Autor: Javier García Gallego

Tutor: Dr. Don José María Gutiérrez Martínez.

Co-Tutora: Dra. Doña Ana Castillo Martínez

2018

Escuela Politécnica Superior

Grado en Ingeniería Informática

Trabajo Fin de Grado

DESARROLLO EN KOTLIN PARA LA CREACION DE UNA RED SOCIAL

Javier García Gallego

Alcalá de Henares, junio 2018

UNIVERSIDAD DE ALCALÁ

Escuela Politécnica Superior

GRADO EN INGENIERÍA INFORMÁTICA

Trabajo Fin de Grado

Desarrollo en Kotlin para la creación de una red social

Autor: Javier García Gallego

Tutor: Dr. Don José María Gutiérrez Martínez.

Co-Tutora: Dra. Doña Ana Castillo Martínez

TRIBUNAL:

Presidente:

Vocal 1º:

Vocal 2º:

CALIFICACIÓN: .....

FECHA: .....

“Eres más valiente de lo que crees,
más fuerte de lo que aparentas,
y más listo de lo que piensas.”

Este trabajo ha sido fruto de innumerables horas de trabajo, lectura de documentación, de constantes pruebas y noches en vela. Agradecer a mis profesores de trabajo de fin de grado y de carrera la ayuda prestada en el transcurso esta bonita etapa.

Dedicado a todas las personas que han confiado en mi y me han dado fuerzas para la consecución de este logro.

Agradecer a mi padre Juan Pedro sus consejos, siempre cortos, potentes y eficientes. Te dan las dosis de frescura que a veces necesitas cuando más atascado puedas estar.
A mi madre Isabel, mi ejemplo de coraje, aguante y tesón. Siempre dispuesta a escuchar y a tender su mano, anteponiéndose a si misma si hiciese falta.
A mi hermano Oscar, porque tiene un mundo por descubrir y el corazón suficientemente grande para acaparar lo que desee, gracias por estar a mi lado aunque ni si quiera yo sea consciente de ello.
A mi novia Irene, porque ha sabido sufrir a un informático estresado, porque con muy poco haces mucho y porque tengo la certeza que de que cambiaras el mundo.
A mis amigos del barrio, que siempre están ahí para pasar un momento de desconexión, a mis amigos de la universidad, por las tardes de estudio compartidas y porque nunca me han dado por perdido.

ÍNDICE

Índice de Ilustraciones

Índice de Tablas

Resumen

Desarrollo de una aplicación Android en lenguaje Kotlin, para crear una red social en la que los usuarios registrados compartan información consistente sobre el estado de determinadas áreas de vegetación y, además, puedan llevar un control sobre la aparición y evolución de las plantas invasoras en dichas áreas gracias a esta aplicación.

Palabras clave.

Android. Kotlin. Red social. Especies Invasoras. Vegetación. Fauna. Desarrollo.

Abstract

Development of an Android application in Kotlin language, to create a social network in which registered users could share consistent information about the status of some vegetation areas and the appearance and evolution of invasive plants in those areas thanks to this application.

Key words.

Android. Kotlin. Social Network. Invasive Species. Vegetation. Wildlife. Development.

1. Introducción

En la actualidad vivimos en una sociedad que está en contacto constante través de nuestros dispositivos móviles, tanto es así, que gran parte de la información se obtiene de compartir esta misma en tiempo real.

De tal forma que, gracias a los usuarios de estas tecnologías, obtenemos información de primera mano y al instante, permitiendo acceder a gran cantidad de información de forma sencilla. Del mismo modo, el auge de la globalización y del acceso a todo tipo bienes materiales, junto con el uso irresponsable de los mismos, hace que determinados tipos de plantas pongan en peligro la supervivencia de las especies vegetales autóctonas de la península Ibérica, por ello, hay necesidad de proveer y alertar a la sociedad de este peligro.

Atendiendo a las posibilidades que ofrecen las redes sociales, como fuentes de generación de información colaborativa, y el problema planteado de conservación de los espacios naturales autóctonos, se pretende crear una solución que ayude a una mejor conservación de estos.

La conexión entre individuos de diferentes áreas geografías permite compartir experiencias y culturas, pero también provoca introducir especias nuevas en entornos autóctonos que provocan la desaparición de las especies locales con el consiguiente impacto medio ambiental.

Con el desarrollo de la aplicación base de este proyecto se consigue conectar dos campos, por un lado, hacer buen uso de la tecnología actual y mantener el ecosistema sostenible y libre de plantas invasoras que puedan deteriorar sus conservación y supervivencia.

Poniendo la mira en este problema, y teniendo en cuenta el empeoramiento de las condiciones climáticas, es importante poner remedio de forma contundente, de cara a tratar de subsanar los daños ocasionados por las plantas invasoras o al menos que no empeoren la situación.

Actualmente nos encontramos en un grado de desarrollo tecnológico óptimo y al alcance de todo el mundo, cada vez más cercano al usuario medio. Esta casuística hace que la tecnología no tenga el foco exclusivamente puesto en la potencia sino en la forma de optimizar la disponible. Así pues, empieza a ganar peso la optimización del software, esta que permite que nuestros dispositivos sean tremendamente fluidos y eficientes. Dicho esto hay una buena oportunidad para crear herramientas útiles de uso cotidiano y que pueden suponer un pequeño cambio en nuestra sociedad.

Con las herramientas y contenido adecuados, la motivación de los usuarios sobre su conocimiento del medio ambiente, tenemos una amplia base para el uso de la herramienta, además de ello, se ha puesto una especial atención en que la aplicación sea lo más sencilla posible y con una buena funcionalidad, ya que, en caso contario, no se conseguiría despertar la curiosidad y necesidad del uso de la aplicación en estos.

Es un hecho contrastado la creciente preocupación de la sociedad en general por el medio natural. Este hecho garantiza la aceptación de la herramienta que se pretende desarrollar. Al mismo tiempo, la posibilidad de compartir información a través de ella puede ayudar aun más a la sensibilización de los usuarios y al aumento del numero de estos.

2. Objetivo

El objetivo de este trabajo de final de grado es el desarrollo de una aplicación que permita el control de plantas invasoras a través de una red social, formada por los usuarios interesados en el problema planteado. Es decir, proporcionar una herramienta de fácil uso y amigable para que sea usada por asociaciones ecologistas y medioambientales, ya que son los colectivos más implicados en la sostenibilidad del ecosistema. También y a través de esta, queremos tener como público objetivo a usuarios amantes de la naturaleza, los cuales al encontrarse más aislados y no disponer de una fuente de información a la que poder acudir, se ven abocados a realizar la labor de forma individual.

La aplicación dispondrá de las siguientes funcionalidades:

- **Visión** de publicaciones de otros usuarios.
- **Búsqueda** de usuarios por nombre.
- **Visualización** del perfil del usuario de la aplicación.
- **Edición** de perfil de usuario.
- **Publicación** de hallazgos con foto y coordenadas GPS.
- **Mapa** que muestre los hallazgos de los usuarios.

3. Estado del arte

En este capítulo describiremos el estado de la tecnología disponible a la hora de realizar el proyecto, así como el de las soluciones existentes, para analizarlas de forma que tengamos una idea del estado de este.

3.1. Contexto Social

En este apartado explicaremos en que estado nos encontramos con el problema a tratar y las distintas vertientes de este.

Las especies invasoras son animales, plantas u otros organismos que se desarrollan fuera de su área de distribución natural, en hábitats que no le son propios o con una abundancia inusual, produciendo alteraciones en la riqueza y diversidad de los ecosistemas. Cuando son transportados e introducidos por el ser humano en lugares fuera de su área de distribución natural, consiguiendo establecerse y dispersarse en la nueva región se les denomina especies exóticas invasoras resultando normalmente muy dañinas.

En los últimos tiempos, el problema de las plantas invasoras comienza a poner en peligro el ecosistema de la península ibérica, tal es así, que la creciente desertización de esta, más la incipiente crecida de incendios forestales en los últimos años, hacen que las especies invasoras más fuertes y resistentes prosperen frente a las autóctonas debido a su dificultad por volver a germinar y restaurar la vegetación que se adueñaba anteriormente de la zona afectada.

Según el ministerio de agricultura y pesca, alimentación y medio ambiente, España es uno de los países con mayor diversidad biológica de la Unión Europea. Su posición geográfica, su rica diversidad geológica, la gran variabilidad climática, orográfica y edáfica, la historia paleobiogeográfica y la existencia de islas son algunos de los factores que han propiciado esta alta diversidad biológica además de una alta tasa de endemicidad.

La introducción de seres vivos fuera de su área natural es el segundo problema ambiental que afecta a la Biodiversidad a escala global de la biosfera,

después de la destrucción de los hábitats por la mano del hombre. Su impacto en el medio natural provoca cambios y genera alteraciones en los ciclos biogeoquímicos, incluso la extinción y rarificación de especies nativas.

Ilustración 1. Acacia dealbata, la mimosa

El número de plantas vasculares en España supera las 8.000 especies, de las que 1.500 son endemismos. Esto supone alrededor del 85% de las especies de plantas vasculares inventariadas en la Unión Europea y la mitad de los endemismos europeos.

Según datos del Ministerio de Agricultura, Alimentación y Medio Ambiente, en España las especies vegetales introducidas están en torno al 15 %, si bien la proporción aumenta hasta el 33 % en las islas Canarias. Además, hay más de 64 especies invasoras identificadas.

Ilustración 2. Ministerio de agricultura y pesca, alimentación y medio ambiente

Sólo en la provincia de Alicante, el número de estas especies se incrementó un 290% entre los años 1972 y 2003. En las dos últimas décadas han sido necesarios planes de actuación para el control de especies alóctonas invasoras en los Parques Nacionales de Garajona y, en la Caldera de Taburiente, Doñana, Timanfaya o el Parque Natural del Delta del Ebro.

Los problemas más notorios que producen este tipo de plantas son el deterioro del sotobosque, ya que sufre una reducción de luz y del consumo de agua, ya que, por ejemplo, la Acacia dealbata o mimosas, tienen un alto consumo de agua, lo que provoca que el resto de las especies no disponga de ella, además provoca una acidificación del suelo y alteraciones de las comunidades microbianas entre otras consecuencias.

Con esta problemática, hay que poner en contexto el uso de las redes sociales en la actualidad, ya que, con un buen uso de estas, pueden ser una forma positiva de compartir información.

Las redes sociales son una estructura social de nodos conectados por intereses similares, por los cuales se puede intercambiar información con distintos fines.

Estas están muy extendidas, estas las podríamos categorizar en varios tipos, de tipo general, como podría ser Facebook, de tipo mensajería, como podría ser WhatsApp, de tipo videos, como podría ser YouTube, de tipo Microblogging, como podría ser Twitter y por último de tipo profesional, como pudiera ser LinkedIn.

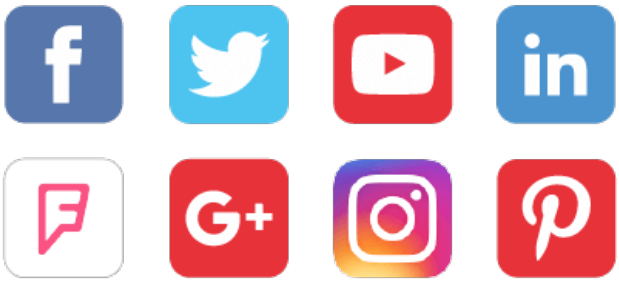


Ilustración 3. Redes sociales con más repercusión

El numero de usuarios de Internet se acerca a los 39,5millones, de los cuales 27 millones son usuarios activos de redes sociales. En cuanto al uso de móvil se sobrepasan los 37 millones de usuarios y 23 millones lo utilizan para redes sociales.

Ilustración 4. Estadísticas de usuarios en España

El dato más destacable del informe we are social and Hootsuite 2018 se han sobrepasado los 4 mil millones de usuarios de Internet y casi 3 millones doscientos mil utilizan las redes sociales en el mundo.

3.2. Soluciones existentes

En esta subsección vamos a explorar las distintas aplicaciones que hay en el mercado, así pues, se realizará una valoración un repaso de las características de estas.

3.2.1. Invasive Alien Species Europe:

Esta aplicación tiene varios puntos fuertes, como, por ejemplo, estar desarrollada por la Unión Europea. Este hecho nos aporta un nivel de fiabilidad y seguridad alto, ya que es una institución con gran repercusión. Al estar bajo su tutela sabemos que tendremos información de primera mano la información que recibimos por parte de la aplicación la consideramos veraz, exacta y completa.

La aplicación sirve para concienciar y alertar de ocurrencias de especies “alienígenas” (invasoras) en la unión europea vía GPS y cámaras de los dispositivos móviles, también provee de información con fotos de determinadas especies de las cuales queramos saber si son invasoras o no. Además de concienciarnos sobre el impacto en el medio que tienen las especies invasoras en el ecosistema, nos facilita ahondar en el problema. A continuación, se muestran algunas capturas de pantalla de la aplicación para mostrar su funcionamiento:

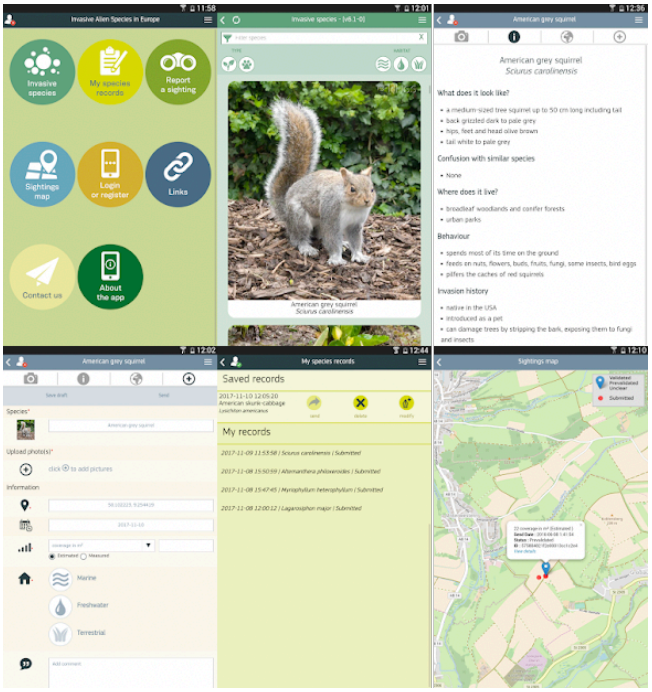


Ilustración 5. Funcionalidad de la aplicacion Invasive Alien Species in Europe

3.2.2. **Report Invasive Plants**

Esta aplicación es similar a la anteriormente descrita, pero poniendo el foco en las plantas, es una aplicación que podría considerarse similar a la que se ha realizado para la consecución de este trabajo, pero no tiene el componente social, al que consideramos importante debido a que motivara a los usuarios y creará conciencia entre ellos.

Las plantas invasoras representan el segundo mayor riesgo para la vida silvestre después de la destrucción del hábitat. La aplicación “Report Invasive Plants” permite informar avistamientos de plantas invasoras en su área. Una vez informado, esta ayudará a rastrear el alcance de estas plantas.

En la aplicación “Report Invasive Plants” tenemos tres vistas principales, la primera para que aprendamos a identificar a determinadas especies de plantas, necesario para poder identificarlas en la vida real, la segunda para reportar un avistamiento y la tercera para ver en un mapa los avistamientos realizados, como se menciona, tiene un parecido con nuestro proyecto, pero no tiene la componente social y cercana que ofrece nuestro producto.

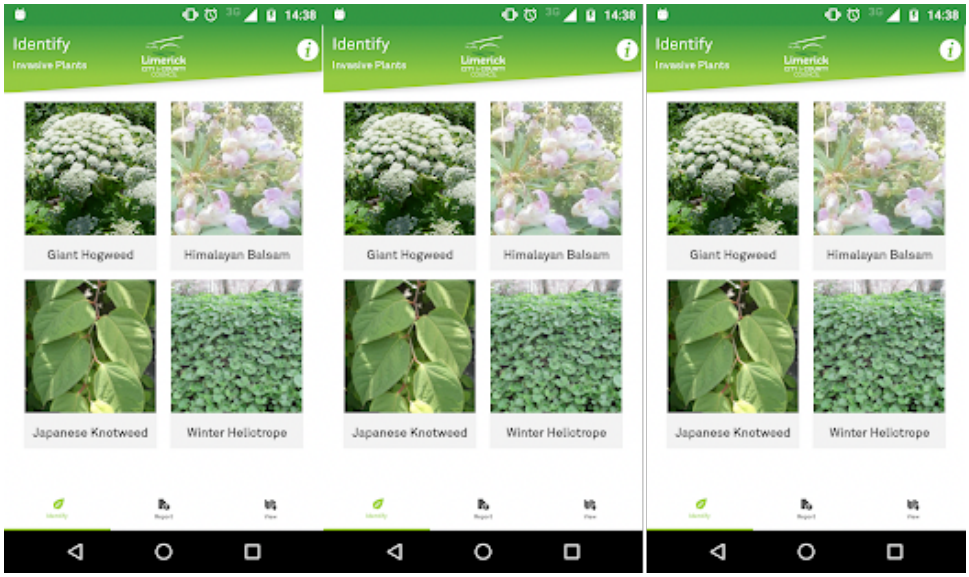


Ilustración 6. Report Invasive Plants

3.2.3. **PlantTracker**

Es una aplicación en la que las organizaciones: The Environment Agency, Scottish Natural Heritage, the Scottish Environment Protection Agency y Natural Resources Wales unen fuerzas para apaliar este hecho en el Reino Unido, ya que, según podemos leer en la descripción de la misma, el problema de las plantas invasoras es bastante grave por la facilidad que tienen para extenderse por toda la región, además sugieren que el coste económico que supone para el Reino Unido el problema de las plantas invasoras son de 2 mil millones de libras anualmente.

Finalmente, y como primer paso, requieren que se haga un registro de plantas para saber la situación en la que se encuentra su ecosistema, por lo cual piden ayuda a su población para poder hacerse un dibujo de cómo esta respecto a las plantas autóctonas y plantas invasivas.

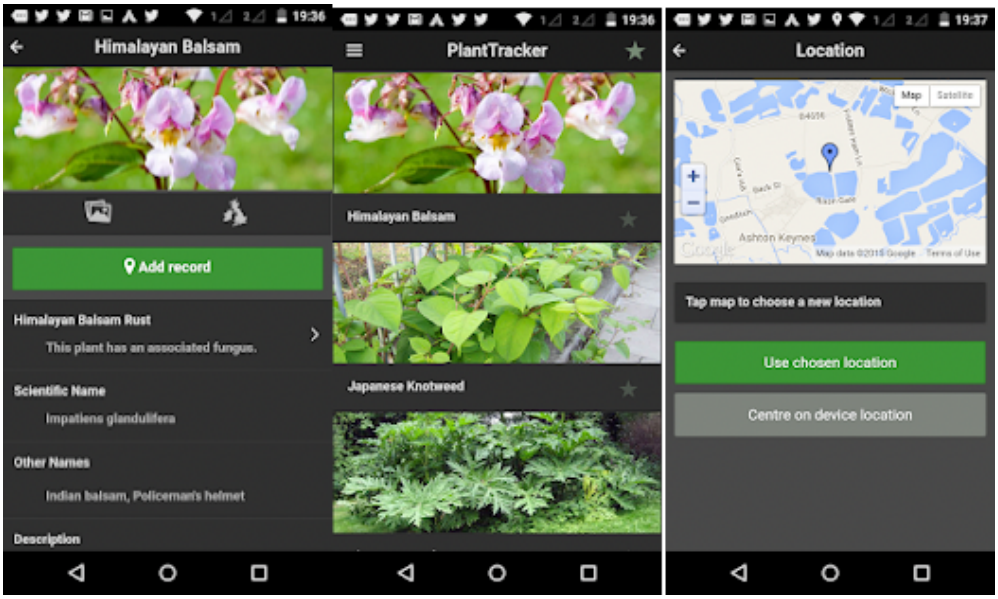


Ilustración 7. Funcionalidad de la aplicación PlantTracker

3.2.4. **Exóticas Murcia**

En esta aplicación, de origen nacional, nos hablan sobre las especies exóticas invasoras, las cuales, y como se ha mencionado antes, son una de las mayores amenazas para la biodiversidad, y para ello alegan que imprescindible recabar información para actuar cuanto antes en su adecuada gestión.

Como podemos ver, a través esta herramienta se pretenden fomentar la colaboración y participación ciudadana para la detección de especies de flora y fauna exóticas, principalmente para aquellas que tengan un marcado carácter invasor, sensibilizando de su impacto medioambiental, para que finalmente se puedan aplicar los programas de erradicación y control.

Esta aplicación permite la toma de datos georreferenciados que se incorporan, tras un proceso de validación, al Sistema de Gestión de Base de Datos Espaciales Biodiversidad de la Oficina de Impulso Socioeconómico del Medio Ambiente de la Comunidad Autónoma de la Región de Murcia. Además, incluye la opción de poder llamar al 112 en caso de incendio.

Esta aplicación es similar a la expuesta en el proyecto, la cual tiene características similares, pero no tiene componente social, ya que el usuario solo tiene acceso a sus hallazgos y no tiene en cuenta los de otros usuarios.

Ilustración 8. Exóticas Murcia, pantallas de la aplicación

3.2.5. **Nuestra aplicación propuesta - Invasive Plants**

Después de haber analizado las aplicaciones anteriores con la información obtenida podríamos llegar a las siguientes conclusiones que podríamos abordar en el proyecto a desarrollar.

Con el desarrollo de Invasive Plants podríamos dar a los usuarios una visión más extensa del compromiso y cuidado del medio ambiente en especial de la vegetación por medio de una aplicación más social y abierta. Que ayude al seguimiento, detección eliminación y control de las especies invasoras.

El usuario podrá realizar fotografías a los hallazgos que realice, podrá ponerle un titulo y una descripción aportando valiosa información a la publicación que vaya a hacer, además, podrá tener un perfil publico que ayudara a otros usuarios a tener un mejor conocimiento de la persona. Por otro lado los usuarios tendrán acceso a un mapa en el cual se podrán visualizar las diferentes localizaciones de las publicaciones de los usuarios.

En definitiva una aplicación que viene a cubrir las necesidades no cubiertas por el resto de las aplicaciones disponibles en este momento y que será una buena ayuda a gente decidida a mirar por la conservación de las especies autóctonas de España.

3.3. Tecnologías

En este punto vamos a hacer una valoración de las tecnologías existentes de cara a poder hacer uso a las que más nos favorezcan.

3.3.1. Tecnologías móviles

En este apartado hablaremos de las tecnologías que podemos encontrar en la actualidad para el desarrollo móvil en Android.

3.3.1.1. Android

Android es un sistema operativo pensado para teléfonos móviles, al igual que iOS, Symbian y Blackberry OS. Lo que lo hace diferente es que está basado en Linux, un núcleo de sistema operativo libre, gratuito y multiplataforma.

Ilustración 9. Primer Android frente a uno de ultima generación

El sistema permite programar aplicaciones en una variación de Java llamada Dalvik, aunque este dejo de usarse para pasar a usar ART en pos de una mayor optimización. El sistema operativo proporciona todas las interfaces necesarias para desarrollar aplicaciones que accedan a las funciones del teléfono (como el GPS, las llamadas, la agenda, etc.) de una forma muy sencilla en un lenguaje de programación muy conocido como puede ser Java o más actualmente, Kotlin.

La primera versión comercial de Android de Android salió al mercado en 2008, se llamaba Apple Pie y desde entonces se han ido sucediendo las distintas versiones, hasta la ultima de nuestros días, la versión de Android Oreo, la versión 8.1.

Android actualmente copa el 80% de ventas de dispositivos móviles (año 2017), el 17 de mayo de 2017, Google, actual propietaria de Android, revelo al mundo Android One, una versión más ligera de Android destinada a países en los que el acceso a tecnología es más complicado y llevar así tecnología de forma más sencilla a pases en vías de desarrollo.

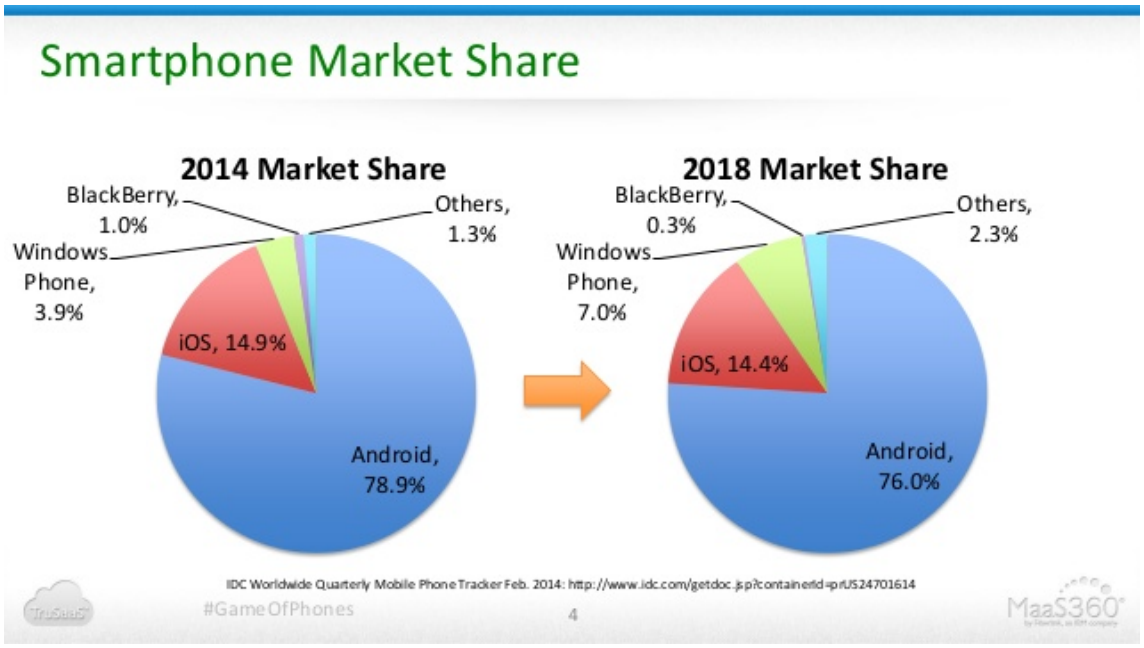


Ilustración 10. Reparto de usuarios según el sistema operativo

3.3.1.1.1. La arquitectura Android consta de las siguientes partes:

- **Aplicaciones**: las aplicaciones base incluyen el cliente de correo electrónico llamado Gmail, programa de SMS, Google calendar, Google Maps, contactos entre otros. Todas las aplicaciones están escritas en lenguaje de programación Java.

- **Marco de trabajo de aplicaciones**: los desarrolladores tienen acceso completo a los mismas API del entorno de trabajo usados por las aplicaciones base. La arquitectura está diseñada para simplificar la reutilización de componentes; cualquier aplicación puede publicar sus capacidades y cualquier otra aplicación puede luego hacer uso de esas capacidades (sujeto a reglas de seguridad del framework). Este mismo mecanismo permite que los componentes sean reemplazados por el usuario.

- **Bibliotecas**: Android incluye un conjunto de bibliotecas de C/C++ usadas por varios componentes del sistema. Estas características se exponen a los desarrolladores a través del marco de trabajo de aplicaciones de Android. Algunas son: System C library (implementación biblioteca C estándar), bibliotecas de medios, bibliotecas de gráficos, 3D y SQLite, entre otras.

- **Runtime** de Android: Android incluye un set de bibliotecas base que proporcionan la mayor parte de las funciones disponibles en las bibliotecas base del lenguaje Java. Cada aplicación Android corre su propio proceso, con su propia instancia de la máquina virtual Dalvik. Dalvik ha sido escrito de forma que un dispositivo puede correr múltiples máquinas virtuales de forma eficiente. Dalvik ejecutaba hasta la versión 5.0 archivos en el formato de ejecutable Dalvik (.dex), el cual está optimizado para memoria mínima. La Máquina Virtual está basada en registros y corre clases compiladas por el compilador de Java que han sido transformadas al formato .dex por la herramienta incluida dx. Desde la versión 5.0 utiliza el ART, que compila totalmente al momento de instalación de la aplicación.

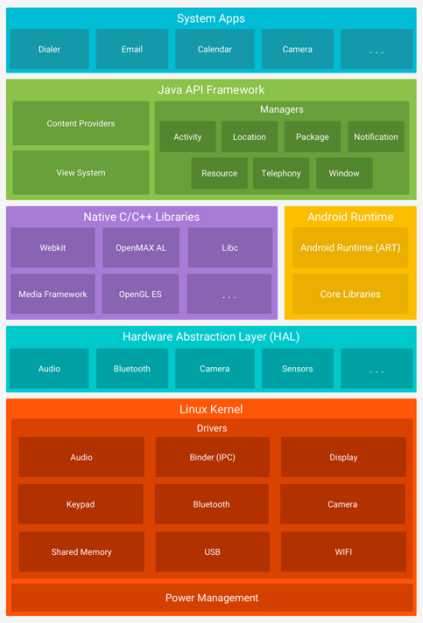


Ilustración 11. Arquitectura de la plataforma Android

- **Núcleo Linux**: Android depende de Linux para los servicios base del sistema como seguridad, gestión de memoria, gestión de procesos, pila de red

y modelo de controladores. El núcleo también actúa como una capa de abstracción entre el hardware y el resto de la pila de software.

3.3.1.1.2. **Versiones de Android**

ANDROID PLATFORM VERSION	API LEVEL	CUMULATIVE DISTRIBUTION
4.0 Ice Cream Sandwich	15	
4.1 Jelly Bean	16	99,2%
4.2 Jelly Bean	17	96,0%
4.3 Jelly Bean	18	91,4%
4.4 KitKat	19	90,1%
5.0 Lollipop	21	71,3%
5.1 Lollipop	22	62,6%
6.0 Marshmallow	23	39,3%
7.0 Nougat	24	8,1%
7.1 Nougat	25	1,5%

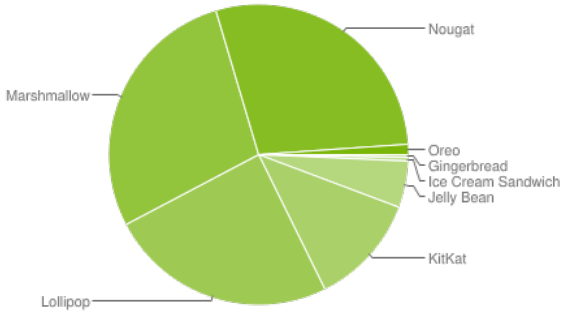


Ilustración 12. Versiones de Android y repartición de marcado

Como podemos observar en las imágenes anteriores, el mayor problema de Android es su fragmentación, por lo cual a la hora del desarrollo de aplicaciones móviles en este sistema operativo es conveniente tomar las precauciones oportunas para hacer nuestra aplicación compatible con la mayor parte de las versiones Android.

3.3.1.1.3. **Google Play Store**

Según Wikipedia, Google Play (previamente Android Market) es un servicio de distribución digital operado y desarrollado por Google. Funciona como la tienda de aplicaciones oficial para el sistema operativo Android, lo que permite a los usuarios navegar y descargar aplicaciones desarrolladas con el kit de desarrollo de software Android (SDK) y publicadas a través de Google. Google Play también sirve como una tienda de medios digitales, que ofrece música, revistas, libros, películas y programas de televisión. Anteriormente ofrecía dispositivos de hardware de Google para la compra hasta la introducción de un minorista de hardware en línea separado, Google Store, el 11 de marzo de 2015.

La tienda de Google Play tuvo más de 82 mil millones de descargas de aplicaciones en 2016 y ha alcanzado más de 3.5 millones de aplicaciones publicadas en 2017.

Google Play se lanzó el 6 de marzo de 2012 y reunió a Android Market, Google Music y Google eBookstore en una sola marca, lo que marcó un cambio en la estrategia de distribución digital de Google. Los servicios que operan bajo el banner de Google Play son: Google Play Books, Google Play Games, Google Play Movies & TV, Google Play Music, Google Play Newsstand y Google Play Console. Tras su cambio de marca, Google ha ampliado gradualmente el soporte geográfico para cada uno de los servicios.

3.3.1.2. **iOS**

iOS es un sistema operativo móvil de la multinacional Apple Inc. Originalmente desarrollado para el iPhone (iPhone OS), después se ha usado en dispositivos como el iPod touch y el iPad. No permite la instalación de iOS en hardware de terceros.

Actualmente es el segundo sistema operativo móvil más utilizado del mundo, detrás de Android, con una cuota de mercado de entre 10-15% al año 2017. La última versión del sistema operativo es el iOS 11, aparecida en el mes de septiembre del 2017, disponible en dispositivos con procesadores 64-bits.

Ilustración 13. Distintas versiones de iOS

iOS se deriva de macOS, que a su vez está basado en Darwin BSD, y por lo tanto es un sistema operativo Tipo Unix. iOS cuenta con cuatro capas de abstracción: la capa del núcleo del sistema operativo, la capa de "Servicios Principales", la capa de "Medios" y la capa de "Cocoa Touch".

Como contrapartida, el sistema operativo de Apple es mucho as cerrado que Android de forma que el control y optimización del sistema esta mucho más desarrollada y el grado de fragmentación es muy inferior si lo comparamos con el otro competidor del mercado.

3.3.1.2.1. **App store**

Ilustración 14. Icono App Store

App Store es un servicio para el iPhone, el iPod Touch, el iPad y Mac OS X Snow Leopard o posterior, creado por Apple Inc, que permite a los usuarios buscar y descargar aplicaciones informáticas de iTunes Store o Mac App Store en el caso de Mac OS X, desarrolladas con el iPhone SDK y publicadas por Apple. Estas aplicaciones están disponibles para ser compradas o bien gratuitas, dependiendo de cada una. Las aplicaciones pueden ser descargadas directamente al iPhone o al iPod Touch por medio de una aplicación del mismo nombre, aunque App Store también está disponible en el interior del programa informático iTunes.

3.3.2. Lenguajes disponibles

A continuación analizaremos los dos lenguajes más usados para el desarrollo de Android.

3.3.2.1. Java

Ilustración 15. Logo de Java

Java es un lenguaje de programación de propósito general, concurrente, orientado a objetos, que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. Su intención es permitir que los desarrolladores de aplicaciones escriban el programa una vez y lo ejecuten en cualquier dispositivo, lo que quiere decir que el código que es ejecutado en una plataforma no tiene que ser recompilado para correr en otra. Java es, a partir de 2012, uno de los lenguajes de programación más populares en uso, particularmente para aplicaciones de cliente-servidor de web, con unos diez millones de usuarios reportados.

El lenguaje de programación Java fue originalmente desarrollado por James Gosling, de Sun Microsystems (la cual fue adquirida por la compañía Oracle), y publicado en 1995 como un componente fundamental de la plataforma Java de Sun Microsystems. Su sintaxis deriva en gran medida de C y C++, pero tiene menos utilidades de bajo nivel que cualquiera de ellos. Las aplicaciones de Java son generalmente compiladas a bytecode (clase Java), que puede ejecutarse en cualquier máquina virtual Java (JVM) sin importar la arquitectura de la computadora subyacente.

3.3.2.2. **Kotlin**

Ilustración 16. Logo del lenguaje Kotlin

Kotlin es un lenguaje de programación de tipado estático que corre sobre la Máquina Virtual de Java (JVM) y que también puede ser compilado a código fuente de JavaScript. Su desarrollo primario es de un equipo de programadores de JetBrains, una compañía conocida por crear IntelliJ IDEA, un entorno de desarrollo enfocado a desarrolladores java, esta compañía tiene base en San Petersburgo, Rusia (el nombre proviene de la Isla de Kotlin, cerca de San Petersburgo).

El líder de desarrollo Andrey Breslav enuncio que Kotlin está diseñado para ser un lenguaje de programación orientado a objetos de calidad industrial, y para ser un lenguaje mejor que Java, pero que todavía debe ser plenamente interoperable con código Java, permitiendo a las compañías hacer una migración gradual de Java a Kotlin, y de estar forma se consiga un trasvase de conocimiento efectivo y regulado, sin ello repercutir en el código de los programadores

Al igual que Pascal, PL/SQL y Scala —y a diferencia de C y sus derivados como C++ y Java — la declaración de variables y listas de parámetros en Kotlin tienen el tipo de dato después del identificador y un separador de dos puntos. Igual que en otros lenguajes modernos como Scala, los puntos y comas son opcionales como final de sentencia; en muchos casos un salto de línea es suficiente para que el compilador pueda deducir que la declaración terminó. En definitiva, es un lenguaje pensado para que los desarrolladores se encuentren en un ambiente familiar, ya que comparte muchas de las características de Java, y a unos leves cambios, que se aprenden con suma rapidez ya que no revisten de mucha complejidad, además de ellos tiene la capacidad de hacer uso de funciones Lambda, que permiten solventar problemas de forma más sencilla.

Otro punto a favor de Kotlin y de su fácil adaptación es que Android Studio tiene una completa integración con el, por lo que un proyecto desarrollado en

Java puede ser fácilmente convertido a Kotlin, simplemente con 3 clics de ratón, es un lenguaje que permite la interoperabilidad entre ambos lenguajes, gracias a este hecho podremos tener proyectos mixtos e incluso hacer uso de librerías escritas en java, como se ha hecho en este proyecto.

Una prueba de retrocompatibilidad que atesora Kotlin de cara a los recursos que disponen los desarrolladores es la gran fuente de información y de librerías escritas en java que pueden usarse sin complicaciones, en este proyecto, por ejemplo, se hace uso de una librería llamada “SmartImagePicker” la cual esta escrita en Java y es completamente funcional en nuestro proyecto escrito en Kotlin, más adelante se hará un repaso de la integración de esta funcionalidad en nuestro proyecto.

Ilustración 17. Compatibilidad de Kotlin y Java

Además de clases y métodos clásicos de la programación orientada a objetos, Kotlin también soporta programación por procedimientos y el uso de funciones. Igual que en C y C++, el punto de entrada a un programa Kotlin es una función llamada "main", que recibe un array que contiene los argumentos pasados desde la línea de comandos. La inferencia de tipos es otra de las características de Kotlin.

Kotlin es un lenguaje que reduce drásticamente la cantidad de código repetitivo, es un lenguaje muy expresivo, lo que se traduce en una menor necesidad de escribir código para realizar tareas que con Java serian escritas con más verborrea. Kotlin es, además, un lenguaje ligero , ya que, siguiendo una de las máximas de Android, se consigue crear APKs que se puedan ejecutar sin problemas en multitud de dispositivos.

A continuación, veremos una imagen en la que se puede apreciar la diferencia entre un lenguaje y otro en cuanto a la capacidad de reducir líneas de código:

```java
public class Person {
    private String name;
    private int age = 0;

    public Person(String name, int age) {
        this.name = name;
        this.age = age;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

    @Override
    public String toString() {
        return "Person{" +
                "name='" + name + '\'' +
                ", age='" + age + '\'' +
                '}';
    }
}
```

```kotlin
class Person(name:String, age:Int) {
  var name:String
  var age = 0
  init{
    this.name = name
    this.age = age
  }
  public override fun toString():String {
    return ("Person{" +
            "name='" + name + '\''.toString() +
            ", age='" + age + '\''.toString() +
            '}'.toString())
  }
}
```

Ilustración 18. Expresividad entre Java y Kotlin

Además, es un lenguaje seguro en cuanto a los valores “null”, ya que lidiamos con posibles situaciones en los que haya valores nulos en tiempo de compilación. Se debe especificar si una variable puede aceptar valores nulos o no, y más tarde comprobarlo, ya que hemos verificado su nulidad antes de compilar, esta característica nos ahorra mucho tiempo depurando y buscando las correcciones de código frente a errores de código relacionados con la nulidad de variables, en comparación con Java es un gran paso que hace producir más cantidad de código de forma robusta y sensible a este tipo de casuísticas. En estas dos capturas podemos observar un ejemplo de cómo podemos manejar este tipo de situaciones:

```kotlin
var b: String? = "abc"
b = null // ok
```

```kotlin
var a: String = "abc"
a = null // compilation error
```

Ilustración 19. Ejemplo de control de la casuística Null

Kotlin nos permite que dejemos de usar código duplicado con las funciones lambdas, ya que podremos pasar como parámetros funciones que nos facilitan la resolución de tareas sencillas y que necesitan de la elaboración de una clase abstracta de la misma para poder dotar de funcionalidad a esa parte de nuestro código. A continuación, muestro un ejemplo del propio proyecto de la aplicación de este hecho:

```
submitBtn.setOnClickListener { it: View!
    val name : String  = nameEt.text.toString()
    val mail : String  = mailEt.text.toString()
    val password : String  = passEt.text.toString()

    if (!TextUtils.isEmpty(name) && !TextUtils.isEmpty(title) && !TextUtils.isEmpty(password)) {
        sendPost(name, mail, password)
    }
    else {
        Toast.makeText(applicationContext,  text: "Please complete Name, Email and Password fields", Toast.LENGTH_SHORT).show()
    }
}
```

Ilustración 20. Uso de funciones Lambda

3.3.3. Tecnologías para la parte del servidor y base de datos

En este apartado vamos a realizar un análisis de las tecnologías usadas para la parte Back-End de nuestra aplicación

3.3.3.1. Nodejs

Node.js es un entorno en tiempo de ejecución multiplataforma, de código abierto, para la capa del servidor (pero no limitándose a ello) basado en el lenguaje de programación ECMAScript, asíncrono, con I/O de datos en una arquitectura orientada a eventos y basado en el motor V8 de Google. Fue creado con el enfoque de ser útil en la creación de programas de red altamente escalables, como, por ejemplo, servidores web. Fue creado por Ryan Dahl en 2009 y su evolución está apadrinada por la empresa Joyent.

Ilustración 21. Logo Node.js

- Concurrencia

Node.js funciona con un modelo de evaluación de un único hilo de ejecución, usando entradas y salidas asíncronas las cuales pueden ejecutarse concurrentemente en un número de hasta cientos de miles. Este diseño de compartir un único hilo de ejecución entre todas las solicitudes atiende a necesidades de aplicaciones altamente concurrentes, en el que toda operación que realice entradas y salidas debe tener una función callback. Un inconveniente de este enfoque de único hilo de ejecución es que Node.js requiere de módulos adicionales como cluster para escalar la aplicación con el número de núcleos de procesamiento de la máquina en la que se ejecuta.

- Módulos

Node.js incorpora varios "módulos básicos" compilados en el propio binario, como por ejemplo el módulo de red, que proporciona una capa para programación de red asíncrona y otros módulos fundamentales, como por ejemplo Path, FileSystem, Buffer, Timers y el de propósito más general Stream. Es posible utilizar módulos desarrollados por terceros, ya sea como archivos ".node" precompilados, o como archivos en JavaScript plano. Los módulos JavaScript se implementan siguiendo la especificación CommonJS para módulos, utilizando una variable de exportación para dar a estos scripts acceso a funciones y variables implementadas por los módulos.

Los módulos de terceros pueden extender node.js o añadir un nivel de abstracción, implementando varias utilidades middleware para utilizar en aplicaciones web, como por ejemplo los frameworks connect y **express**. A pesar de que los módulos pueden instalarse como archivos simples, normalmente se instalan utilizando el controlador de paquetes de Node, npm que nos facilitará esta tarea de compilación, instalación y actualización de módulos, así como la gestión de las dependencias.

- Desarrollo homogéneo entre cliente y servidor

Node.js puede ser combinado con una base de datos documental (por ejemplo, MongoDB o CouchDB) y JSON lo que permite desarrollar en un entorno de

desarrollo JavaScript unificado. Con la adaptación de los patrones para desarrollo del lado del servidor tales como MVC y sus variantes MVP, MVVM, etc. Node.js facilita la reutilización de código del mismo modelo de interfaz entre el lado del cliente y el lado del servidor.

- Bucle de eventos

Node.js se registra con el sistema operativo y cada vez que un cliente establece una conexión se ejecuta un callback. Dentro del entorno de ejecución de Node.js, cada conexión recibe una pequeña asignación de espacio de memoria dinámico, sin tener que crear un hilo de ejecución. A diferencia de otros servidores dirigidos por eventos, el bucle de gestión de eventos de Node.js no es llamado explícitamente, sino que se activa al final de cada ejecución de una función callback. El bucle de gestión de eventos se termina cuando ya no quedan eventos por atender.

3.3.4. **Mongoose**

Mongoose es un modulo de Node diseñada para trabajar en entornos asíncronos, mongoose es el ORM de MongoDB para Nodejs, es un API con el conseguimos ser capaces de usar un lenguaje orientado a objetos para interactuar con Mongo y realizar las operaciones de base de datos que necesitemos, como pueden ser las famosas CRUD, Create, Read, Update and Delete.

Además, nos permite definir los esquemas y modelos de datos para dar forma a los tipos de objetos que almacenaremos en nuestra base de datos de una forma programática, nos permite la creación de objetos con una gran facilidad y con una diverso numero de tipos de dato disponibles. Finalmente podemos crear la conexión a nuestra base de datos, definiendo un nombre para la misma y el puerto que esta va a usar en nuestro proyecto.

Ilustración 22. Logo de Mongoose

3.3.5. **Express**

Express es un web framework de Node.js, es una extensión del poderoso connect, Express tiene las características de ser rápido y sencillo y robusto, esta destinado a ser una ayuda para gestionar el servidor y las ruta s de nuestra aplicación.

Es una opción ideal ya que es asíncrono y de hilo único, lo que nos proporciona la certeza de que no nos encontraremos con situaciones de cuello de botella. Nos proporciona un control de las llamadas asíncronas que recibirá nuestro back-End, además de un control eficientes de las entradas y salidas.

Gracias a la robusta API que nos proporciona podemos configurar fácilmente

Ilustración 23. Logo Express

las rutas para el envío y la recepción de información entre la base de datos y nuestro Front-End.

3.3.6. **MongoDB**

MongoDB es una base de datos de tipo no relacional (NoSQL) que esta orientado a documentos y colecciones, y que es desarrollado en el concepto de código abierto.

MongoDB en vez de almacenar la información en tablas como hacen las bases de datos relacionales, este los almacena en forma de documentos. Los cuales, tienen formato similar al formato JSON, llamado BSON. De esta forma consigue que la introducción de datos sea más fácil y ágil.

El desarrollo de MongoDB empezó en octubre de 2007 por la compañía de software 10gen. Ahora MongoDB es una base de datos madura, preparada para

su uso comercial y con una gran variedad de funcionalidades, algunas de las empresas que usan esta base de datos son las siguientes: Forbes, Bosch o Cisco

3.3.7. **MySQL**

MySQL es un sistema de gestión de bases de datos de tipo relacional está considerada como la base datos de código abierto más popular del mundo. Es una de las más populares junto a Oracle y Microsoft SQL Server, en, sobre todo, desarrollos web.

MySQL fue inicialmente desarrollado por MySQL AB (empresa fundada por David Axmark, Allan Larsson y Michael Widenius). MySQL AB fue adquirida por Sun Microsystems en 2008, y ésta a su vez fue comprada por Oracle Corporation en 2010, la cual ya era dueña desde 2005 de Innobase Oy, empresa finlandesa desarrolladora del motor InnoDB para MySQL. Está desarrollado en su mayor parte en ANSI C y C++.

MySQL es usado por muchos sitios web grandes y populares, como Wikipedia, Google, Facebook, Twitter, Flickr y YouTube.

4. Desarrollo

En este apartado vamos a detallar el proceso de desarrollo realizado para la consecución de los objetivos impuestos en la realización de este proyecto. Una vez se han investigado los aspectos teóricos respecto a las tecnologías que hay disponibles aplicadas para la creación de conciencia sobre el medio ambiente y en especial sobre las plantas invasoras.

En la parte de Front-End usaremos el IDE de Android Studio y el desarrollo de la aplicación nativa lo haremos en lenguaje Kotlin en vez del más conocido Java. Kotlin es un lenguaje joven, con total interoperabilidad con Java, esto hace que los programadores seniors tengan una curva de aprendizaje corta y su adaptación sea mucho más sencilla de lo que en un primer momento pueda parecer. Es un lenguaje mucho más descriptivo que Java, de forma que Kotlin nos ayudaría a crear un código eficiente en términos de escritura, más que si lo hiciésemos en Java.

En la parte de almacenamiento de datos, vamos a hacer uso de la base de datos no relacional llamada MongoDB, este tipo de bases de datos son mucho más flexibles que las bases de datos SQL, ya que permiten una mejor escalabilidad y un mejor rendimiento a la hora de almacenar los datos, tratarlos y modificarlos.

Para gestionar la conexión entre nuestro cliente y nuestro servidor, vamos a hacer uso de la herramienta Nodejs, Nodejs es un entorno de desarrollo JavaScript, con el cual podremos gestionar fácilmente, los modelos de almacenamiento de datos, los controladores con los cuales haremos todo el proceso CRUD y las rutas, las cuales nos facilitaran la recepción y envío de peticiones desde nuestra aplicación del terminal.

4.1. Requisitos

4.1.1. **Funcionales**

ID	**Descripción**
RF1	El sistema permitirá logarse en la aplicación mediante un formulario concebido para ello.
RF2	El sistema notificará con un mensaje en la pantalla de log in de que se ha realizado un loggeo erróneo en caso de que así sea.
RF3	El sistema permitirá registrarse en el sistema. Mediante un formulario concebido para ello.
RF4	El sistema notificara en la pantalla de registro de una introducción de datos errónea.
RF5	El sistema notificara de si se ha realizado un login incorrecto.
RF6	El sistema mostrara una pantalla principal con las publicaciones de otros usuarios de forma que sean mostradas por fecha de publicación, las más recientes primero.
RF7	El sistema mostrara la opción de visualizar un mapa con los marcadores de las publicaciones de los usuarios.
RF8	El sistema mostrara una vista para buscar usuarios por nombre.
RF9	El sistema mostrara en una vista el perfil del usuario, mostrando su foto de perfil con su nombre y descripción y todas las publicaciones del usuario ordenadas por fecha.

RF10	El sistema permitirá realizar publicaciones al usuario, en las cuales debe facilitar un titulo una descripción, una foto y opcionalmente, su localización.
RF11	El sistema permitirá editar la descripción del usuario.
RF12	El sistema notificara mediante un cuadro de texto que no cumple con los requisitos mínimos para realizar una publicación.
RF13	Al inicio se muestra el login de acceso, tras iniciar sesión se muestra la pestaña de publicaciones.
RF14	La pestaña “Búsqueda”, permite realizar al usuario los perfiles de otros usuarios.
RF15	Los formularios de creación de publicaciones tienen validación previa de los datos antes del envió para su creación.
RF16	El formulario de edición de perfil a un usuario contará con la validación de los datos necesarios.
RF17	El sistema hará uso del posicionamiento GPS del terminal.
RF18	El sistema hará uso de la cámara de terminal.

Tabla 1. Requisitos funcionales

4.1.2. **No funcionales**

ID	**Descripción**	**Tipo**
RNF1	La aplicación debe poder utilizarse en cualquier dispositivo Android.	Producto
RNF2	El sistema dará un identificador a cada publicación hecha a través de la aplicación.	Producto
RNF3	La aplicación actualizará, creará y modificará los datos de las publicaciones, imágenes o usuarios mediante funciones asíncronas.	Producto
RNF4	La aplicación se adaptará a cualquier diagonal de pantalla del dispositivo móvil en el que se este utilizando.	Producto
RNF5	La aplicación guardara las contraseñas encriptadas para una mayor seguridad.	Producto
RNF6	Solo se permite una dirección de correo electrónico por usuario.	Organización
RNF7	El sistema identificara a cada usuario por un único email.	Organización
RNF8	Se permite el registro de cualquier usuario mediante el formulario disponible en el inicio de la aplicación.	Seguridad
RNF9	La aplicación podrá utilizarse desde cualquier país.	Externos

Tabla 2. Requisitos no funcionales

4.2. Casos de uso

En este apartado podemos observar los casos de uso de los que consta nuestra aplicación, en el se muestra las distintas acciones que puede hacer el usuario para interactuar con nuestra aplicación.

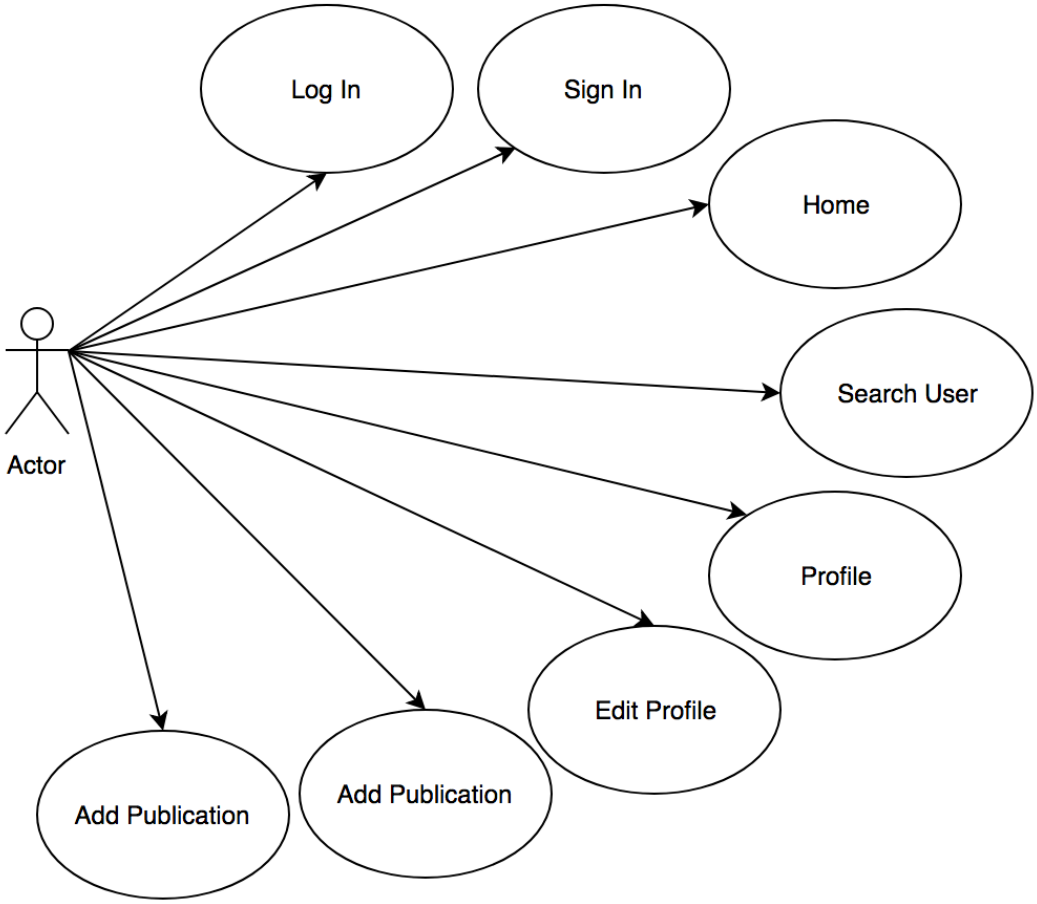


Ilustración 24. Casos de uso

4.3. Diagrama de clases

En este diagrama, podemos observar como estará organizado nuestro proyecto, en el encontraremos 3 carpetas contenedoras, en las cuales por un lado encontraremos la lógica de las aplicaciones, por otra parte, encontraremos los adapters, usados como parte de la estructura de los reciclerview y la carpeta utils, la cual contendrá la interfaz de retrofit para la conexión con la base de datos y los modelos para las respuestas del back-end

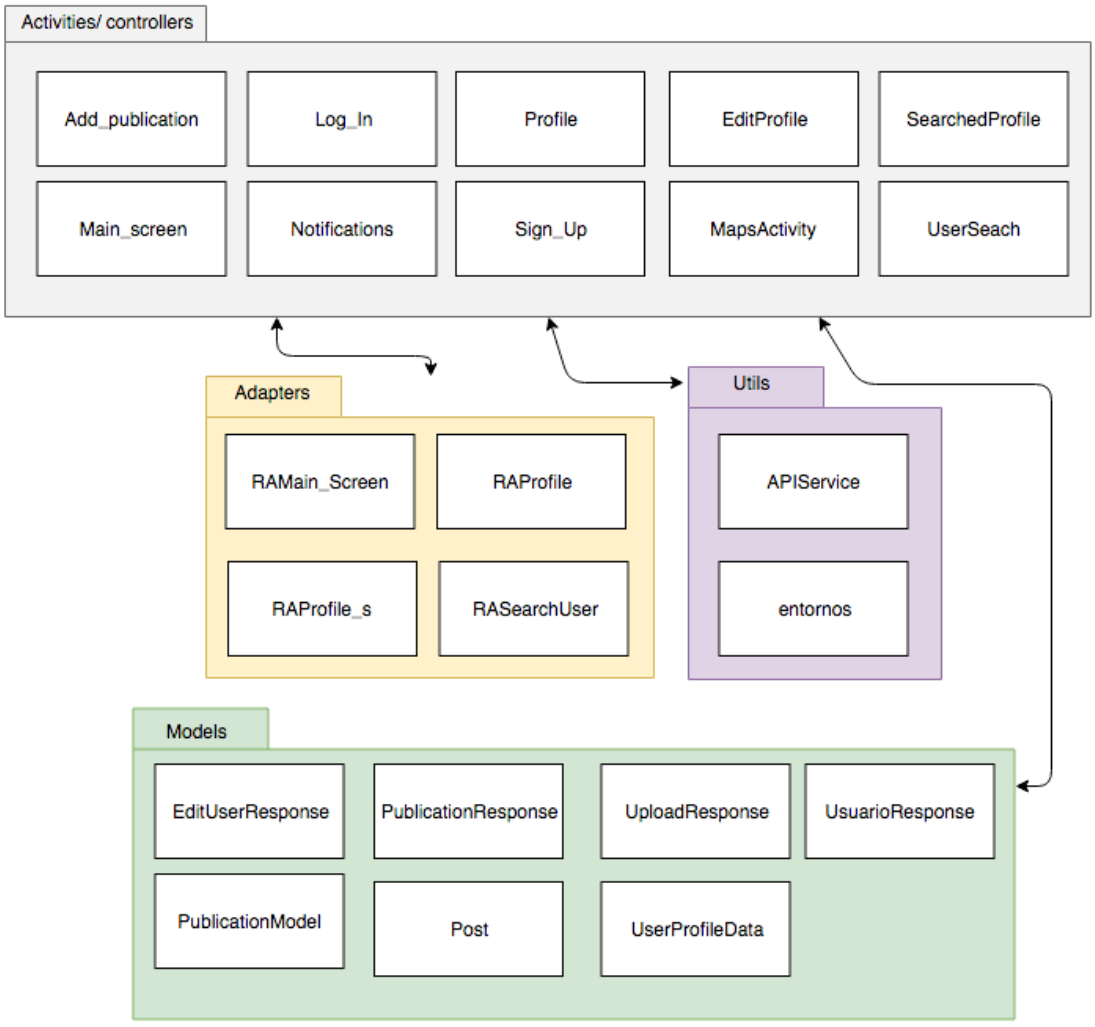


Ilustración 25. Diagrama de clases

4.4. Bocetos

4.4.1. Login

En una primera actividad nos encontraríamos con la ventana de login, en la cual nos permitirá, introduciendo una identificación valida, entrar en la aplicación, además en caso de no disponer de cuenta, podremos crearnos una cuenta

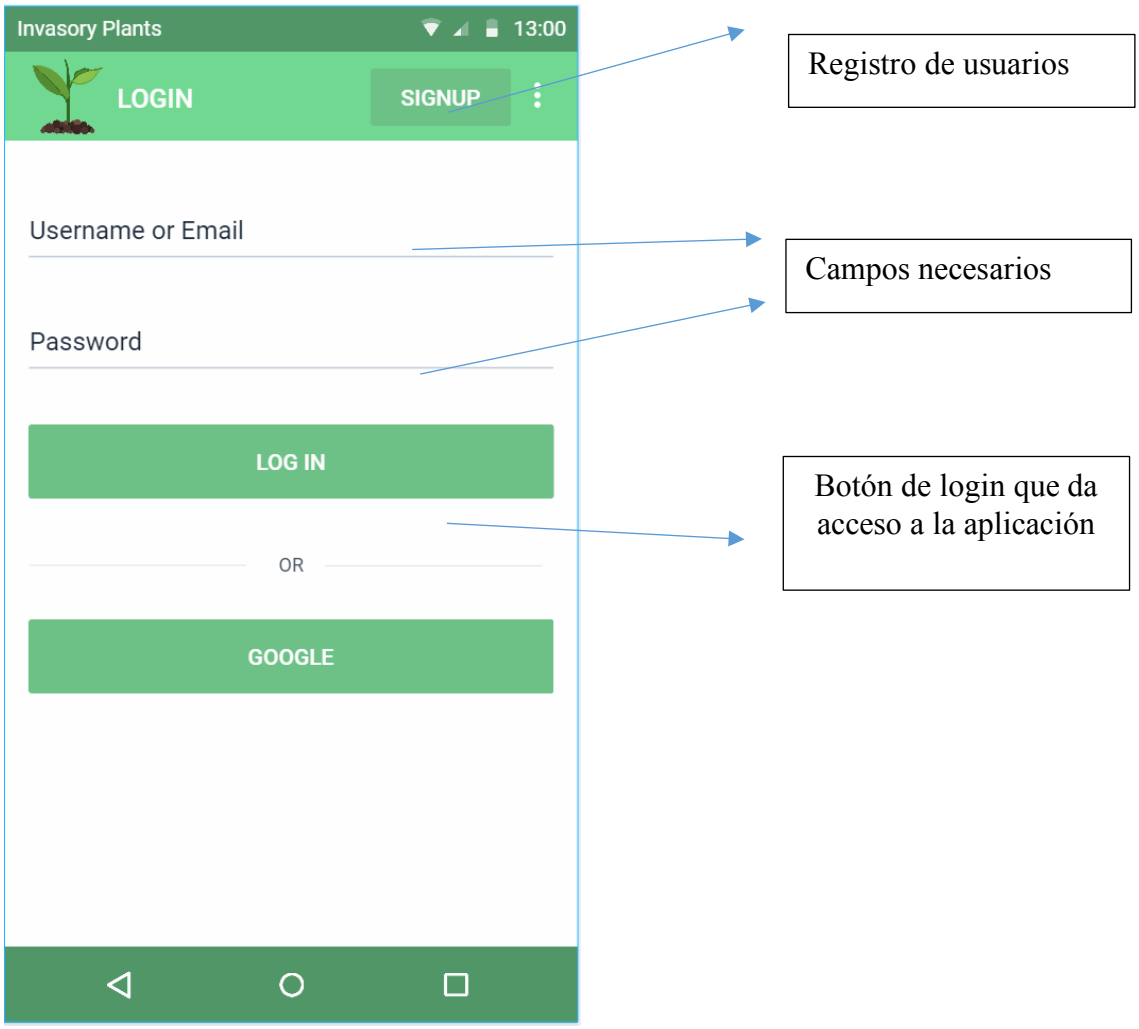


Ilustración 27. Boceto Login

4.4.2. **Ventana principal**

En ella encontramos las publicaciones de usuarios ordenadas por fecha y el botón de mapa para interactuar con la aplicación.

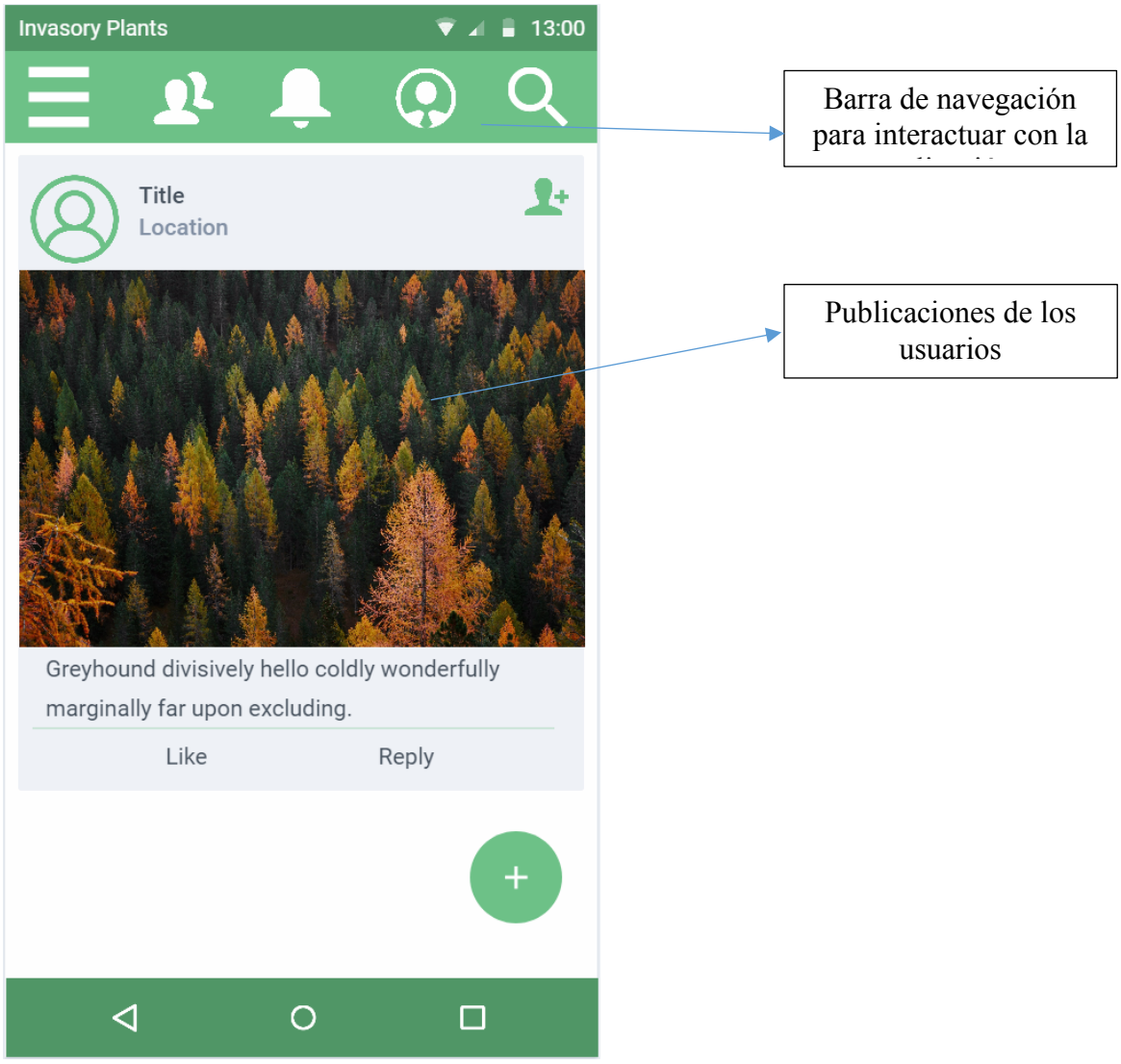


Ilustración 28. Boceto vista principal

4.4.3. Búsqueda de usuarios

En ella se permite la búsqueda por nombre de los usuarios de la aplicación, pudiendo ver los perfiles de estos haciendo clic en los resultados

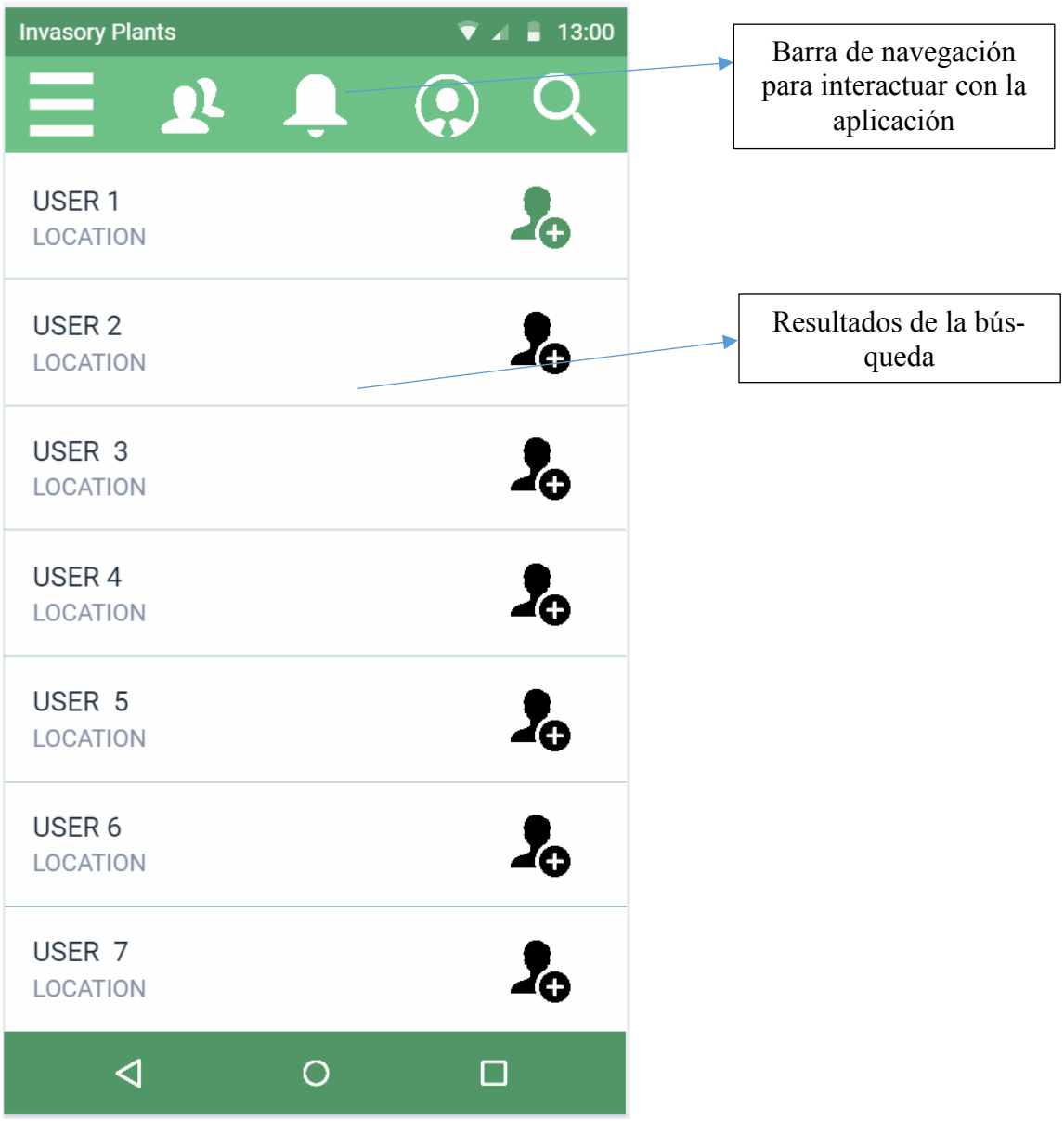


Ilustración 29. Boceto búsqueda de usuarios

4.4.4. Vista de pagina de perfil de usuario

En esta podremos ver datos como su foto de perfil, su nombre, descripción y publicaciones hechas por el usuario ordenadas por fecha

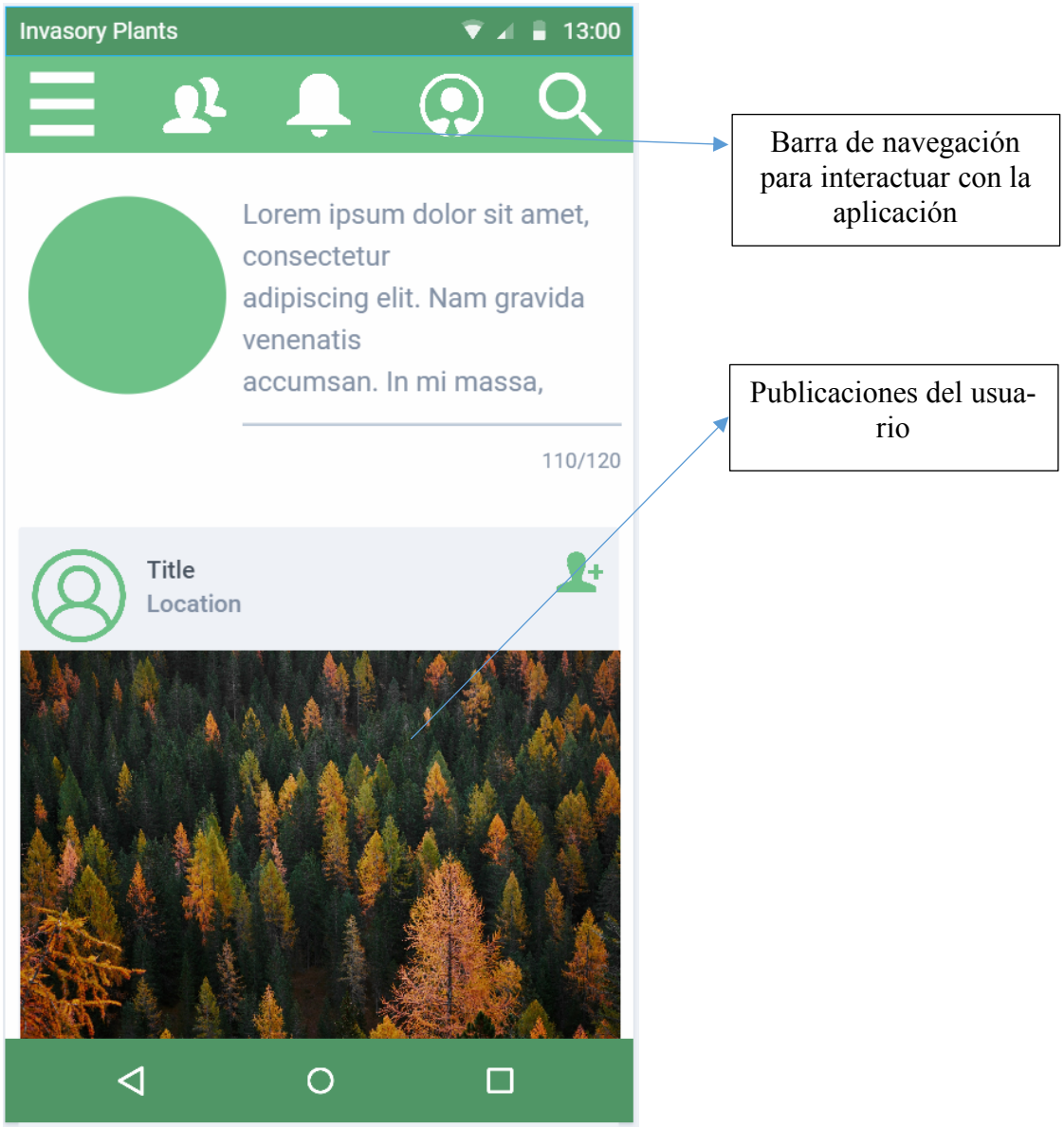


Ilustración 30. Boceto perfil de usuario

4.4.5. Vista de pagina de crear publicación

En esta actividad se podrán crear publicaciones, estas deben tener un titulo y una descripción, además de una fotografía y las coordenadas GPS en el momento de realizar la publicación.

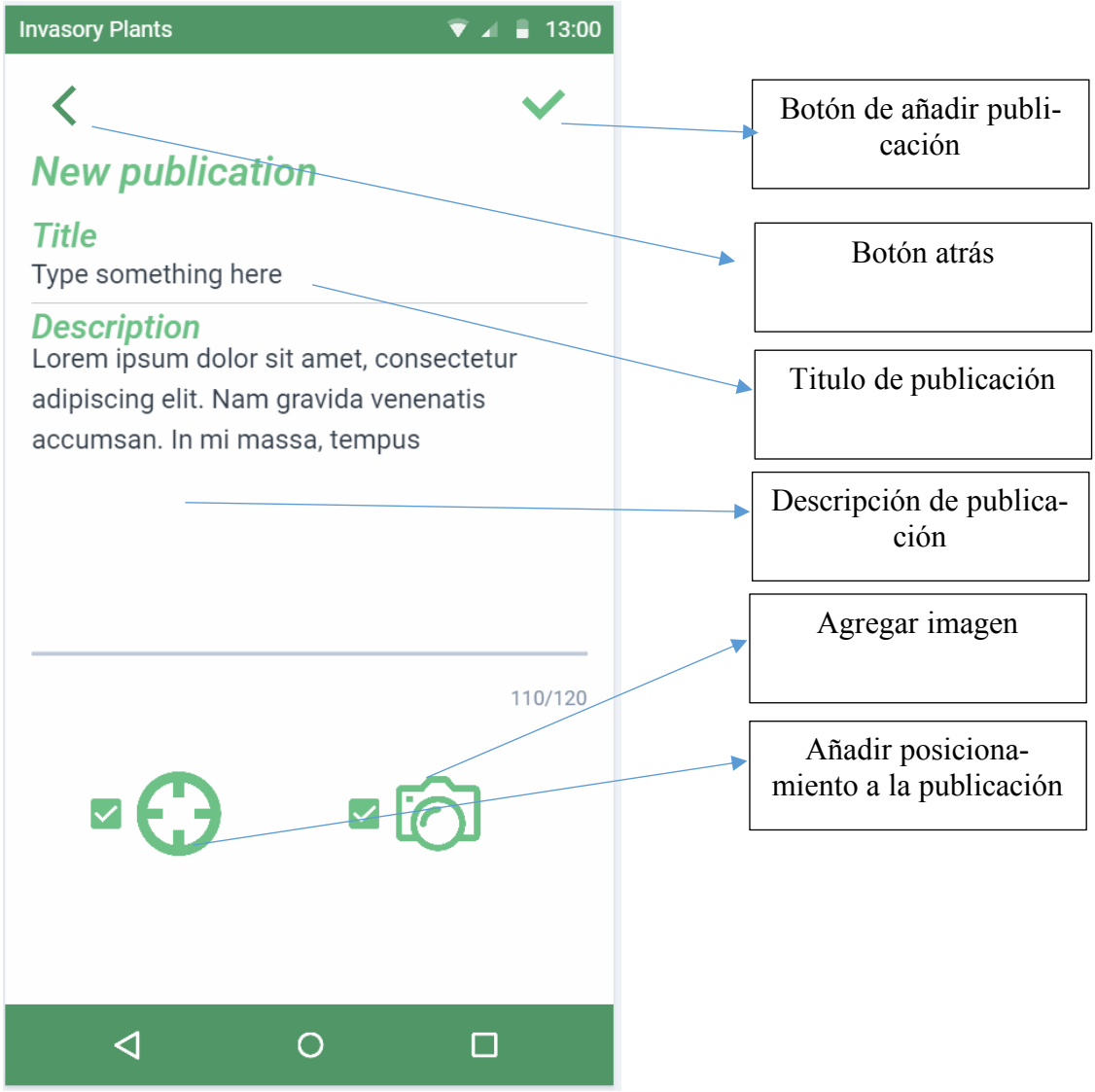


Ilustración 31. Boceto nueva publicación

4.4.6. **Vista de la actividad de mapa**

En esta vista podremos visualizar en un mapa las publicaciones de usuarios

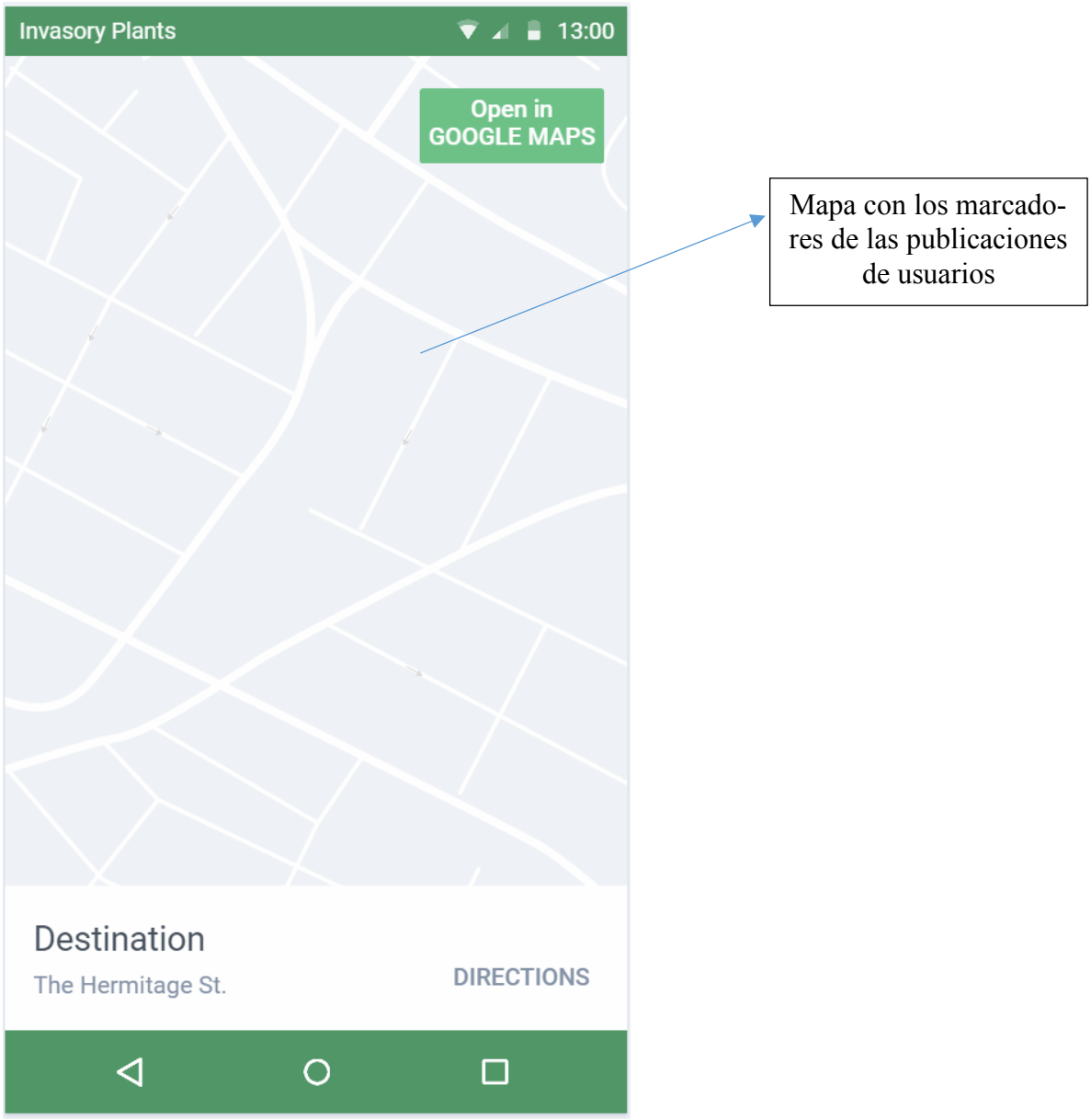


Ilustración 32. Boceto Mapa

4.5. Arquitectura del sistema

Nuestra aplicación podría definirse como una aplicación nativa, es una aplicación con conexión a Internet, con la consecuencia de que este tipo de aplicaciones tienen la información esta alojada en un back-end, el cual cuenta con una base de datos la cual contiene la información, en nuestro caso usamos MongoDB, que a través de un API Rest generada con Node.js, este comunica la base de datos con el front-end de la aplicación, que es donde se puede ver la información solicitada mediante las peticiones hechas desde Android.

Además, nuestra aplicación podría considerarse una aplicación móvil de información dinámica ya que toda nuestra información esta almacenada en la base de datos, la cual es accesible a través de Internet y que podremos estar actualizando constantemente a medida que los usuarios la añaden nuevas publicaciones o información.

Ventajas:

- Los cambios se realizan de manera fácil, solo contando con conocimientos básicos.
- Permite editar la información en la base de datos a través de una plataforma especifica.
- Cualquier persona con permisos necesarios, podría realizar tareas desde cualquier lugar.

Desventajas:

- El coste suele ser mayor, debido a que hay que contratar personal para el desarrollo web.
- Existe un coste también a la hora de contratar alguna empresa de hosting.
- El desarrollo de la aplicación es más complejo.

4.5.1. Modelo vista controlador

El Modelo-vista-controlador (MVC), según Wikipedia, es un patrón de arquitectura de software, que separa los datos y la lógica de negocio de una aplicación de su representación y el módulo encargado de gestionar los eventos y las comunicaciones. Para ello MVC propone la construcción de tres componentes distintos que son el modelo, la vista y el controlador.

Este patrón de arquitectura de software se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento.

- **Modelo**. Recoge la información de la lógica de la esta. En nuestro proyecto se correspondería con la base de datos. Es la parte más reutilizable, podemos portar fácilmente todo el modelo de una aplicación a otra.
- **Vista**. Se correspondería con la parte que el usuario ve e interactúa. En nuestra aplicación se corresponde con el XML del proyecto de Android Studio.
- **Controlador**. El controlador se corresponde con la parte de lógica de nuestra aplicación, se refiere a las acciones que realiza el usuario que tienen repercusión en nuestro modelo.

Gracias a esta arquitectura conseguimos las siguientes ventajas:

- El trabajo se divide entre programadores y diseñadores de forma fácil.
- El mantenimiento se vuelve más simple, al tener diferenciadas las partes.
- A la hora de realizar los test a la aplicación esta tarea resulta más sencilla.
- Permite una reutilización del código más eficiente.

Al utilizar el modelo, al igual que obtienes puntos positivos tiene su contrapartida y es que el conocimiento para el desarrollo de este tipo de proyectos requiere de conocimientos más avanzados.

4.5.2. Implementación de MVC

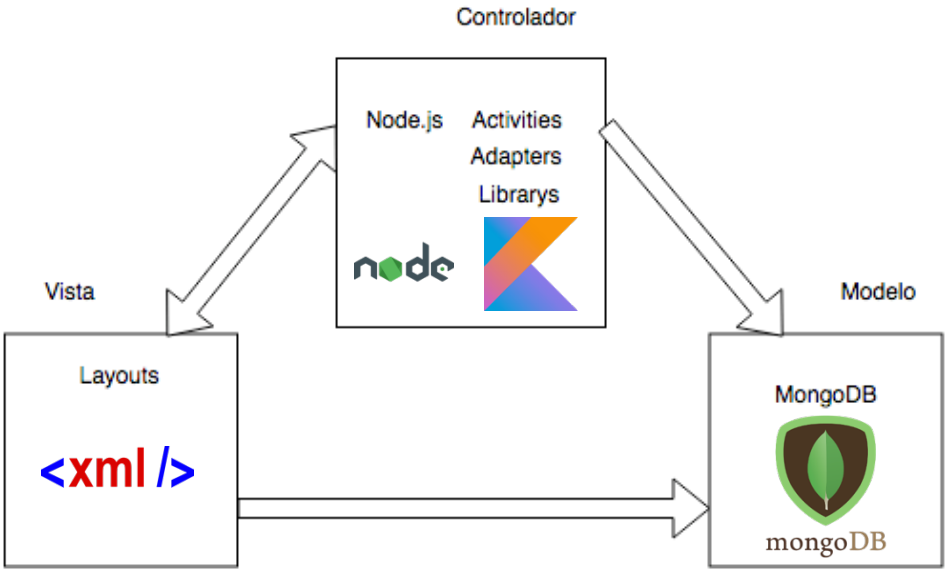


Ilustración 33. Implementación MVC

- **Modelos**: Serían las estructuras de datos que va a utilizar la aplicación, en este caso con los tres tipos de documentos de MongoDB. Escritos en el lenguaje JavaScript usando el ORM Mongoose.
- **Controladores**: Aquí entra en juego la parte de la aplicación, en la que manejamos las entrada y salida de información, proveniente tanto de las vistas como de la base de datos, usaremos Android (Kotlin) y los controladores de Node.js y su framework Express.
- **Vistas**: Todas las vistas que se mostrarán en la aplicación están almacenadas en esta carpeta. Tienen la extensión .XML, ya que es el lenguaje para la interfaz gráfica que utiliza Android.

4.6. Tecnologías usadas

4.6.1. Back-end

Como cualquier aplicación ya sea móvil o de ordenador, si queremos tener una persistencia de datos, se deberá disponer de una base de datos.

En las bases de datos actuales, podremos guardar desde imágenes, archivos de texto, código en XML o JSON, etc. En este proyecto hemos usado para almacenar información la dupla Node.js y MongoDB, la cual es ampliamente usada en desarrollos web, pero que para nuestra aplicación tiene mucho potencial para el desarrollo del proyecto.

MongoDB es una base de datos no relacional (NoSQL) la cual usando el ORM Mongoose a través de Node.js, podremos realizar las tareas de inserción modificación actualización y eliminación de manera sencilla, algunas de sus características son:

- Escalable. Lo que nos garantiza que si en algún momento la carga de usuarios aumenta la aplicación mantendrá unos buenos tiempos de respuesta.
- Ágil. MongoDB en distintas pruebas de rendimiento es entre 2 y 4 veces más rápido que MySQL en consultas sencillas, por ejemplo.
- Posee un esquema dinámico, de forma que si en un futuro la aplicación se sigue desarrollando y se le dota de más funcionalidades, podríamos añadir más campos a los documentos ya creados sin mayores problemas.
- Uso de JavaScript para realizar consultas y manejar datos.
- Usa documentos embebidos, que son documentos que se definen dentro de otro documento, permiten establecer relaciones entre distintas colecciones de documentos y de esta forma poder obtener y manipular datos en una sola petición.

Nuestros modelos de base de datos son los siguientes:

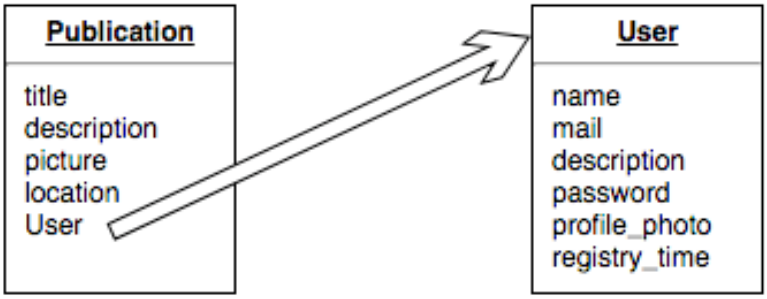


Ilustración 34. Modelo de datos

En cuanto al modelo Publication, encontramos partes como el titulo y descripción, imagen que tratamos con multipart para manejo de imágenes, localización, la cual estará formadas por coordenadas y el usuario, el cual esta referenciando al otro modelo, el modelo User.

Para el modelo User, encontramos los datos básicos de este además de una fecha de registro, hay que añadir que la contraseña esta cifrada de modo que al introducir el usuario esta se codifica y se hace opaca de cara a aumentar la seguridad de la aplicación.

Para el manejo de datos, hemos desarrollado un proyecto en Node.js el cual consta de directorios principales, las rutas, por las cuales realizaremos las llamadas Get, Post, Update y Delete. Por otro lado, tenemos los controladores, que aportan la parte lógica de obtención, recolección, tratamiento y localización de información. Todo ello lo realizaremos con ayuda del paquete Express el cual es la pieza fundamental de cara a poder realizar una gestión de los controladores y las rutas correcta.

Para, por ejemplo, realizar una petición de una publicación en particular, lo realizaríamos de la siguiente forma:

Para enviar una petición al back-end, debemos hacerlo con la siguiente ruta:

http://localhost:3700/api/publication/:id

En la parte lógica manejaríamos la petición de esta forma:

```
function getPublication(req, res) {
    var publicationId = req.params.id;
    Publication.findById(publicationId, function (err, publication) {
        if (err) {
            res.status(500).send({ messsage: 'Error al devolver la publicacion' });
        } else {
            if (!publicationId) {
                res.status(404).send({ messsage: 'La publicacion no existe' });
            } else {
                Album.populate(publication, { path: 'user' }, (err, publication) => {
                    if (err) {
                        res.status(500).send({ messsage: 'Error en la peticion' });
                    } else {
                        res.status(200).send({ publication });
                    }
                });
            }
        }
    });
}
```

Ilustración 35. Ejemplo lógica de obtención unitaria de publicación

De cara a la consecución de un dominio publico, se ha hecho uso de la herramienta localtunnel.me el cual nos provee de un dominio para poder acceder al back-end de forma remota. Hacemos uso de la herramienta de la siguiente forma.

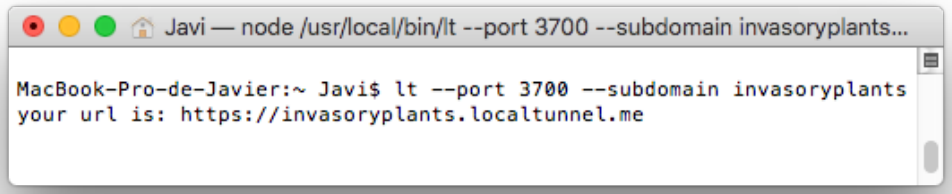


Ilustración 36. Herramienta para la obtención de dominio publico

Otro punto importante que merece la pena destacar en la parte back-end es la subida y almacenamiento de imágenes. Hacemos uso de la librería de Node.js llamada connect-multiparty, que nos permitirá realizar subidas de imágenes

nuestra base de datos. A continuación, mostramos el código de ejemplo haciendo uso de este paquete:

```
function uploadImage(req, res) {
    var albumId = req.params.id;
    var file_name = 'No subido...';
    if (req.files) {
        var file_path = req.files.image.path;
        var file_split = file_path.split('\\');
        var file_name = file_split[2];

        var ext_split = file_name.split('\.');
        var file_ext = ext_split[1];

        if (file_ext == 'png' || file_ext == 'jpg' || file_ext == 'gif') {

            Album.findByIdAndUpdate(albumId, { image: file_name }, (err, albumUpdated) => {
                if (!albumUpdated) {
                    res.status(404).send({ message: 'No se ha podido actualizar el usuario' });
                } else {
                    res.status(200).send({ album: albumUpdated });
                }
            });

        } else {
            res.status(200).send({ message: 'Extensión del archivo no valida' });
        }

    } else {
        res.status(200).send({ message: 'No has subido ninguna imagen...' });
    }
}
```

Ilustración 37. Ejemplo lógica de subida de imágenes

Debemos además debemos tener presente que a la hora de subir la imagen debemos usar la siguiente ruta con la llamada post:

http://localhost:3700/api/upload-image-p/:id

Dicho todo esto, cabe realizar hincapié en la parte en la que los dos modelos se relacionan entre si, ya que es parte importante para que el usuario tenga por ejemplo sus publicaciones localizadas y se pueda tener un control sobre ellas , esto lo realizamos en la declaración de la colección de Publicaciones, el cual tiene un atributo de tipo Shcema.ObjectId con un atributo que indique a que modelo se hace referencia. A continuación, vemos el código responsable de ello subrayado:

```
var PublicationSchema = Schema({
    title: { type: String, required: true, trim: true },
    description: { type: String, required: true, trim: true },
    picture: String,
    registry_time: { type: Date },
    lat: Number,
    lon: Number,
    user: { type: Schema.ObjectId, ref: 'User' }
});
```

Ilustración 38. Estructura del documento publicación

La importancia que le debemos a Mongoose reside en que gracias a este paquete podremos crear filtros, ordenar resultados y realizar distinto tipos en MongoDB de una forma mucho más suave de la que trae por defecto MongoDB.

Es una capa de abstracción que nos permite trabajar con métodos y objetos antes de trabajar directamente con la base de datos. A continuación, mostramos algunos extractos de estas funcionalidades anteriormente descritas:

```
Publication.find({ user: userId }).sort('-title').exec…

Publication.findById(publicationId, function (err, publication) {…

publication.save((err, publicationStored) => {…

Publication.findByIdAndRemove(publicationId, function (err…

Album.findByIdAndUpdate(albumId, { image: file_name }…
```

Ilustración 39. Ejemplo de uso de Mongoose

4.6.2. **Front-end**

En cuando a la parte Kotlin hemos de hablar del uso de los siguientes componentes de cara a entender el funcionamiento de estos.

4.6.2.1. **RecyclerView**

Primeramente cabe hablar sobre el elemento RecyclerView, el cual es un contenedor de elementos en forma de lista. Este elemento permite “reciclar” los ítems que ya no son visibles por el usuario cuando este realiza sobre el contenido scrolling. Por lo que es ideal para proyectos que manejan grandes volúmenes de ítems que se actualizan constantemente, como el que estamos tratando, limitando la visibilidad de elementos.

Adicionalmente permite configurar una serie de animaciones para la eliminación, desplazamiento y creación de nuevos elementos en tiempo real. Se necesita una fuente de datos que provea la información lógica de cada elemento y un adaptador que los lea, interprete e “infle”. Pero también es necesario un nuevo elemento llamado LayoutManager.

El LayoutManager es el encargado de añadir y reusar los views en el recycler. Su función es calcular las posiciones fuera del foco del usuario y así reemplazar el contenido de un ítem fuera del volumen visual por el contenido de otro. Esto reduce los tiempos de ejecución, ya que el número de infladas es menor.

En nuestro proyecto hemos hecho uso de en total 3 reciclerview, para mostrar la actividad principal, para el perfil de usuario, y para mostrar los resultados de búsqueda de usuarios.

Además, una de las características que nos aporta el ReciclerView es que nos permite interactuar con las pulsaciones de los usuarios, así en nuestro caso nos permite realizar una búsqueda de usuarios y además que se pueda acceder al perfil con un toque en la pantalla.

4.6.2.2. **Librería “Android Better Image Picker Library”**

Una parte fundamental del Proyecto es la capacidad que el usuario pueda mostrar contenido por medio de imágenes, es por esto de la relevancia de esta librería, la cual nos habilita la posibilidad de accede a la cámara del teléfono o a algún contenedor de imágenes del mismo, como puede ser la galería para poder subir archivos a nuestra aplicación. A continuación se mostrará algunos puntos clave para su uso.

Según podemos observar en el código de abajo, tenemos la declaración de lo que sería nuestro imagePicker, en el especificamos si usamos la librería en un activity o en una fragment, recogemos su URI y además dejamos por defecto que el formato de imagen sea 4,3 como podemos apreciar.

```
imagePicker = ImagePicker(
        activity: this,
        fragment: null,
        { imageUri :Uri! ->
            //var uri: Any = imageUri
            this.iv_foto_selected.setImageURI(imageUri)
        })
        .setWithImageCrop(
                aspectRatioX: 4,
                aspectRatioY: 3)
```

Ilustración 40. Ejemplo de declaración del objeto de la librería

Además al llamar a la función podemos especificar si queremos que acceda a la cámara o solo podamos cagar la imagen, en este caso permitimos que el usuario tome una fotografía.

```
imagePicker.choosePicture( includeCamera: true)
```

Ilustración 41. Opciones de eventos de la librería

A la hora de coger el archivo para enviarlo a nuestra base de datos

Ilustración 42. Obtención del archivo de la imagen capturada

Para manejar la información que obtenemos de la acción de seleccionar la imagen, necesitamos tener esta capacidad, esta función nos habilita a gestionar la información obtenida del evento.

```
// Funcion manejadora del resultado de tomar la imagen
override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent) {
    super.onActivityResult(requestCode, resultCode, data)
    imagePicker.handleActivityResult(resultCode, requestCode, data)
}
```

Ilustración 43. Manejador de resultado del evento de captura de imagen

4.6.2.3. **Librería Picasso**

Para mostrar imágenes hemos hecho uso de una librería que pertenece a Square, esta es una potente herramienta que hemos usado para la muestra de imágenes a lo largo de la aplicación, nos provee de funcionalidades tan esenciales como el escalado de imágenes hasta la capacidad de colocación de estas en nuestra vista, otro punto muy interesante es que con una línea de código conseguimos realizar operaciones que requerirían bastantes líneas de código.

4.6.2.4. **Librerías para http de Square**

Para la conexión con el Back End, hemos hecho uso de las librerías de Square, como son Retrofit y Okhhtp, gracias a ellas se ha logrado la comunicación entre ambas partes de forma correcta.

En cuento a Retrofit, cuando vamos a hacer uso de ella, ha sido necesario, primero instanciar la interfaz de la API, a continuación crearnos un modelo de datos para que nuestro código pueda entender las respuestas que recibía y a continuación declararnos la función de llamada, la cual siempre tendrá dos funciones, una encargada de la realización de acciones en caso de respuesta positiva y otra encargada de actuar en caso de que la respuesta sea errónea.

Esta librería ha sido fundamentalmente usada para las llamadas post y put de nuestro proyecto. Cabe destacar el uso de la llamada multipart para la carga de imágenes a nuestros backend.

Por la parte de Okhhtp es una librería más sencilla de usar y es la que hemos usado para realizar las peticiones get con y sin filtros, las cuales nos sirven para poder rellenar de información los RecyclerViews y el mapa del, en concreto para poder poblarlo de marcadores.

4.6.2.1. **API Google Maps**

En este proyecto se ha hecho uso del api de Google Maps para poder mostrar un mapa y así mostrar los marcadores con las localizaciones de las publicaciones de los usuarios.

En se realizó una instalación del SDK de Google Play Services, para más tarde crear la actividad preconfigurada de un mapa. A continuación, se debe habilitar la API Key que te genera la propia actividad y comenzar a personalizar la vista, con las opciones que queramos, la localización de inicio y el zoom entre otros.

Para finalizar, mostramos un ejemplo de como esta configurado en el proyecto. A la hora de añadir los marcadores, se consigue a través de una llamada para traer los datos y a continuación agregarlos al mapa.

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_maps)
    val mapFragment : SupportMapFragment = supportFragmentManager
            .findFragmentById(R.id.map) as SupportMapFragment
    mapFragment.getMapAsync(this)
}

override fun onMapReady(googleMap: GoogleMap) {
    mMap = googleMap
    mMap.isMyLocationEnabled
    val spain = LatLng(40.463667, -3.74922)
    fetchJson(urlBase, googleMap)
    mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(spain, 4.2F))

}
```

Ilustración 44. Lógica de actividad que muestra el mapa

5. Coste del proyecto

El presente apartado contiene la estimación presupuestaria realizada para el desarrollo de este proyecto. La estimación del presupuesto de este proyecto se realiza teniendo en cuenta dos pautas: por un lado, se considera el precio de los equipos necesarios para el desarrollo del proyecto, y por otra parte se tiene en cuenta el coste que supone el tiempo empleado en la ejecución de dicho trabajo.

5.1. Presupuesto de ejecución material

Se denomina coste total de la ejecución material, a la suma del coste de los equipos y del coste por tiempo de trabajo.

5.1.1. Coste de material

En este apartado se detallará el material que será utilizado para la realización del trabajo:

- Se contará con un ordenador para el desarrollo de la aplicación con la suficiente potencia como para llevar a cabo la compilación y depuración de la aplicación.
- Se contará con un terminal Android para probar la aplicación fuera del entorno de desarrollo, este terminal contará con la versión 6 de Android.
- Por último, se contará con una conexión a Internet para poder hacer uso de los servicios online de la aplicación.
- Se usará el IDE de Android Studio para la parte gráfica y de desarrollo.
- Como formación se ha seguido un curso de pago para la adquisición de conocimientos.

Tomando como referencia estos aspectos el coste de material de trabajo podría establecerse de la siguiente forma:

EQUIPO	PRECIO	DURACION	USO	TOTAL
Ordenador	2.500,00 €	4 años	9 meses	468,75 €
Conexión a internet	120,00 €	---	9 meses	1080 €
Oneplus X	200,00 €	3 años	9 meses	50,00 €
Oneplus 3t	479 €	1,5 años	1 mes	26,61 €
Formación	15 €	2 semanas	6 meses	15,00 €
		Coste total de equipos		1.625,36 €

Tabla 3. Coste total equipo

5.1.2. Coste de mano de obra.

CONCEPTO	Nº HORAS	EUROS/HORA	TOTAL
Ingeniería (jefe proyecto)	150	25	3.750,00 €
Analista	250	14	3.500,00 €
Programador	500	10	5.000,00 €
	Coste total tiempo trabajado		12.250,00 €

Tabla 4. Coste mano de obra

En el concepto de Ingeniería incluye:

- Análisis de la aplicación en función de los requerimientos:
  - Diseño y desarrollo de la aplicación.
  - Diseño de tablas, consultas y relaciones.
  - Desarrollo de la interfaz de usuario.
  - Implementación de los Informes.
  -
- Depuración del código y puesta en marcha.
- Redacción de la memoria.

5.1.3. Coste total de ejecución material

Se refiere a la suma de ambos presupuestos

CONCEPTO	COSTE
Coste de equipos	1.625,36 €
Coste de mano de obra	12.250,00 €
Coste de ejecución material	13.875,36 €

Tabla 5. Coste total ejecución material

5.2. Gastos generales y beneficio industrial

Se trata de los gastos relativos a las instalaciones en las que desarrollar el proyecto y otros gastos adicionales. En definitiva consiste en aplicar un recargo del 20% sobre el coste de ejecución material, obteniendo lo siguiente:

Gastos generales y beneficio industrial	2.775,07 €

Tabla 6. Gastos generales y beneficio industrial

5.3. Presupuesto de ejecución por contrata

Es el resultado del sumatorio entre presupuesto de coste total de ejecución material y los gastos generales y beneficio industrial.

Gastos generales y beneficio industrial	16.650,43 €

Tabla 7. Presupuesto de ejecución por contrata

5.4. Honorarios

Los honorarios facultativos por la ejecución de este proyecto se determinan de acuerdo a las tarifas de los honorarios de los ingenieros en trabajos particulares vigentes a partir del 1 de septiembre de 1997, dictadas por el Colegio Oficial de Ingenieros Técnicos de Telecomunicación.

Importe	Coeficiente reductor	Porcentaje
Hasta 5 millones	C = 1	7%
Desde 5 millones	C = 0,9	7%

Tabla 8. Honorarios

Los derechos del visado se calcularán aplicando la siguiente fórmula: 0,07 × P × c , donde P es el presupuesto de ejecución material y c es el coeficiente reductor.

Derechos de visado	Coste
0,07 x 16.650,43 x 1	1.165,53 €
Total honorarios	1.165,53 €

Tabla 9. Honorarios específico

5.5. Importe total del presupuesto

El importe total del presupuesto de este proyecto se calcula sumando el presupuesto de ejecución por contrata y los honorarios. A dicho valor se le aplica el 21 % de I.V.A.

Presupuesto de ejecución por contrata	16.650,43 €
Honorarios	1.165,53 €
SUBTOTAL	17.815,96 €
21% de I.V.A.	3.741,35 €
IMPORTE TOTAL	21.557,31 €

Tabla 10. Presupuesto total del proyecto

El importe total del proyecto asciende a la cantidad de VEINTIUN MIL QUINIENTOS CINCUENTA Y SIETE EUROS CON TREINTA Y UN CENTIMOS

6. Resumen, conclusiones y trabajos futuros

6.1. Resumen

El trabajo de fin de carrera pretende servir como sello a una carrera de aprendizaje, a la facilidad de adaptación con la que los alumnos debemos lidiar, ya que estamos expuestos constantemente a la actualización de nuevas tecnologías, motivo por el cual nunca dejamos de aprender y siempre debemos tener curiosidad por la consecución de objetivos y de descubrimientos, personalmente el trabajo de fin de carrera es un gran reto, personal y un sueño desde que estudiaba en el instituto, un sueño que poco a poco, con la elaboración de este proyecto se ha ido cumpliendo, la movilidad a día de hoy es una de las fuentes de movilización más importante, por ello debemos realizar contenido y aportar herramientas, fiables, amigables y respetuosas con los valores éticos y morales que tenemos en nuestra sociedad.

6.2. Conclusiones

La confección de una red social siempre ha sido muy interesante, ya que es una de las razones por las que las tecnologías móviles han conseguido entrar tan fuerte en nuestras vidas, permiten conectar con usuarios que anteriormente se consideraban lejanos físicamente, y no solo eso, nos permiten, además, mantener un seguimiento de la actividad de estos.

Una de las mayores trabas que se pueden encontrar los usuarios de hoy día es la dificultad para hacer uso de las aplicaciones, bien por su mala experiencia de usuario, que tengan fluidez o que no aporten la confianza necesaria para su uso. Con estas premisas, se ha tratado de crear un trabajo que pueda ser usado por cualquier persona con un mínimo conocimientos en el manejo de dispositivos electrónicos.

Además, el problema principal de este trabajo premiará a aquellos usuarios que tengan conocimiento relativo a la vegetación, ya que la aplicación servirá para realizar un seguimiento de sucesos, de cara a conseguir una sostenibilidad ambiental adecuada a cada tipo de clima, localización e historia.

En nuestras manos está que la situación mejore, con las herramientas adecuadas y la conciencia medioambiental generada será por seguro una apuesta de valor de cara a afrontar la problemática de a de este tema.

6.3. Trabajos futuros

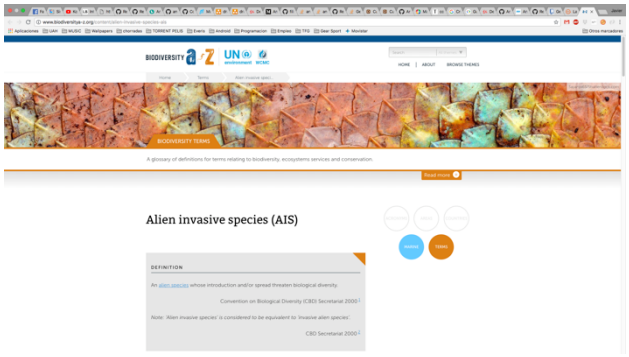
En un futuro este proyecto puede seguir introduciendo más funcionalidades, por ejemplo se podría trabajar en una mejor implementación a nivel de seguridad, se podría implementar algún tipo de gamificación que hagan que los usuarios quieran usar la aplicación, y en una versión más avanza se podría incluir la realidad aumentada, de forma que se pueda tener una experiencia más amplias de las zonas tratadas, para futuros brotes de plantas ya eliminados y para tener un concepto en definitiva más extenso de la evolución de la vegetación invasiva en la península Ibérica.

Otra alternativa de desarrollo de la aplicación en un futuro seria extenderla a las escuelas e institutos para formar a los alumnos en el impacto medio ambiental de las especies invasoras.

En referencia a todo lo anterior descrito, la motivación de este TFG siempre ha sido un sueño, un sueño que sentó sus bases aun estando en el instituto, el sueño de poder realizar mi propia aplicación en Android, un reto que se ha cumplido con la realización de esta tarea y que pone el broche a este maravilloso ciclo académico.

7. BIBLIOGRAFIA

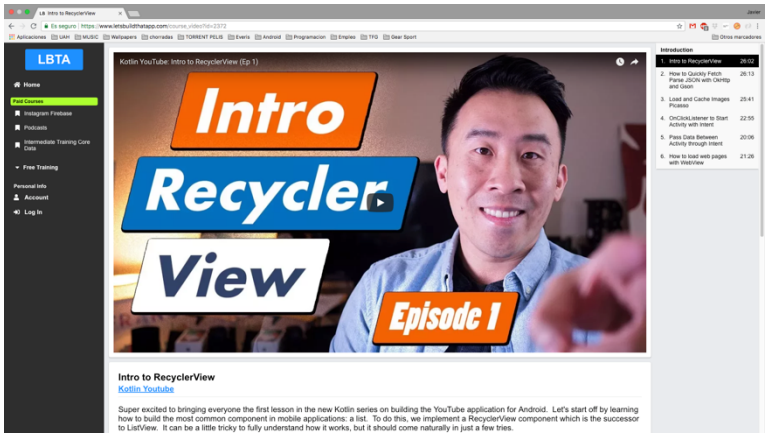
1. Biodiversitya-z.org. (2018). Alien invasive species (AIS) definition| Biodiversity A-Z. Disponible en: http://www.biodiversitya-z.org/content/alien-invasive-species-ais



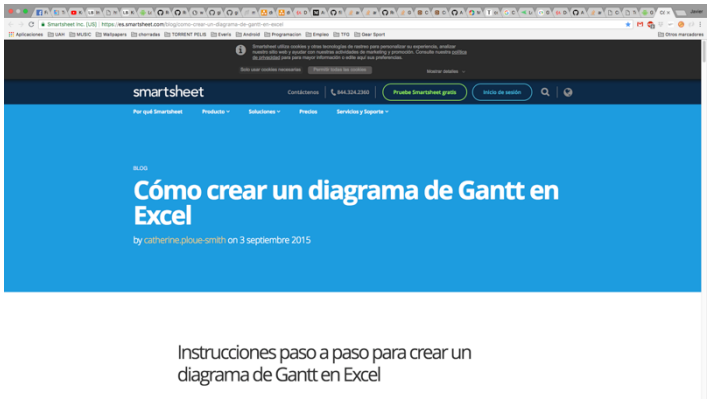
2. Webempresa20 - Internet orientado a resultados. (2018). Las 30 Redes Sociales más Utilizadas. Disponible en: https://www.webempresa20.com/blog/las-30-redes-sociales-mas-utilizadas.html

3. Antonio Leiva, ED (2016). *Kotlin for Android Developers: Learn Kotlin the easy way while developing an Android App.* 21 mar 2016.

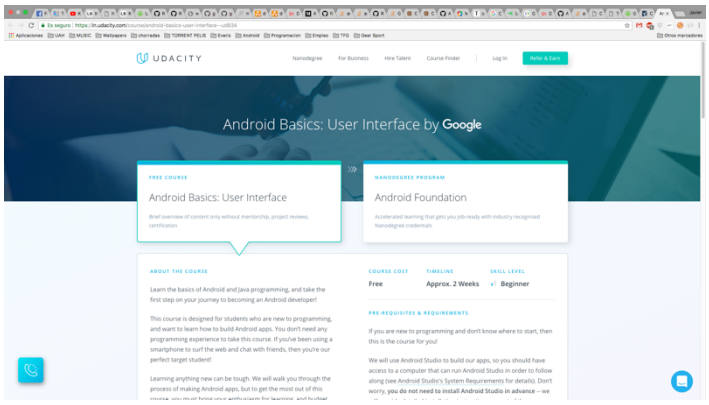
4. Let’s build that app. https://www.letsbuildthatapp.com/course_video?id=2372



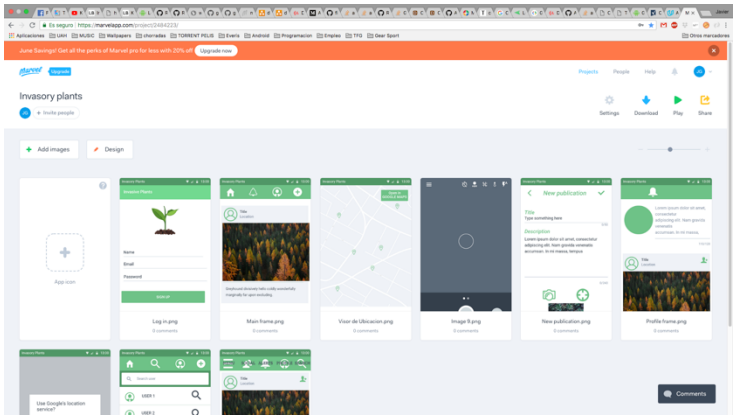
5. Como crear un diagrama Gantt en Excel. https://es.smartsheet.com/blog/como-crear-un-diagrama-de-gantt-en-excel (Consultado el 21 de noviembre de 2017).



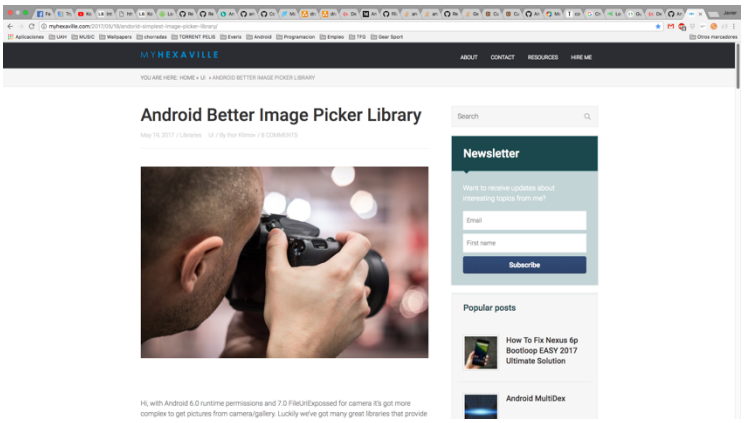
6. Katherine Kuan, Kunal Chawla. Google. *Android Basics: User Interface. Curso Online de Udacity* https://in.udacity.com/course/android-basics-user-interface--ud834



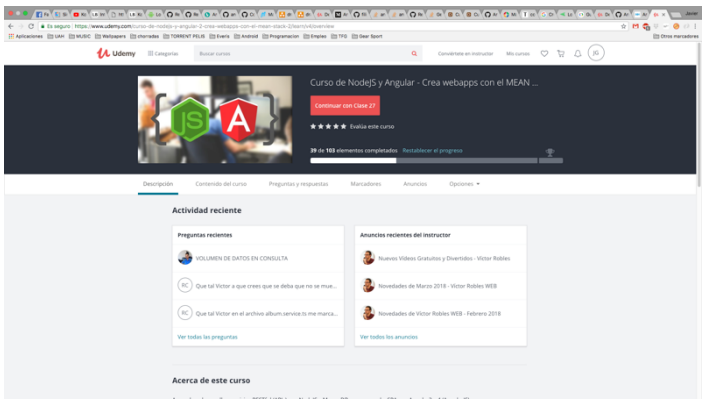
7. Marvelapp. Maquetación de interfaces. https://marvelapp.com/project/2484223/ (Consultado el 28/11/2017).



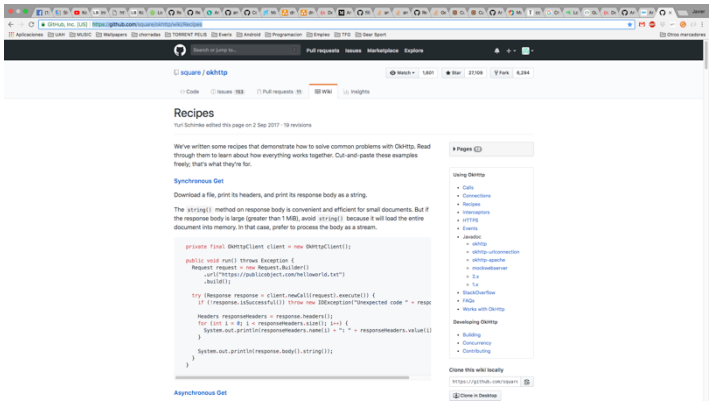
8. Android Better Image Picker Library *http://myhexaville.com/2018/02/05/android-smart-image-picker-library-update/*



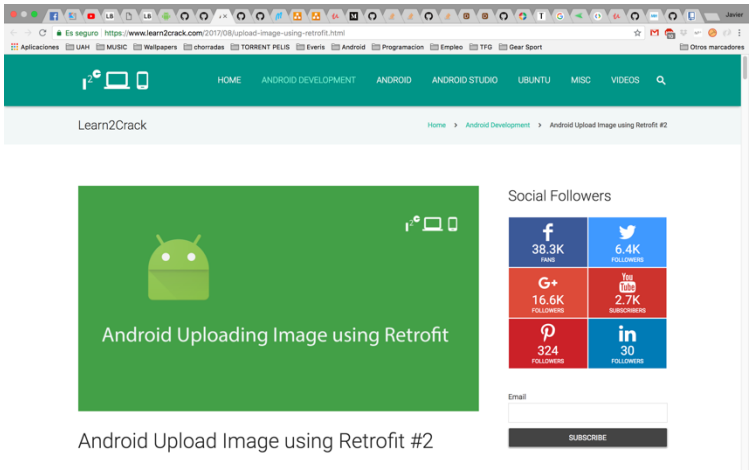
9. Curso de NodeJS y Angular - Crea webapps con el MEAN Stack - Víctor Robles https://www.udemy.com/curso-de-nodejs-y-angular-2-crea-webapps-con-el-mean-stack-2/learn/v4/overview



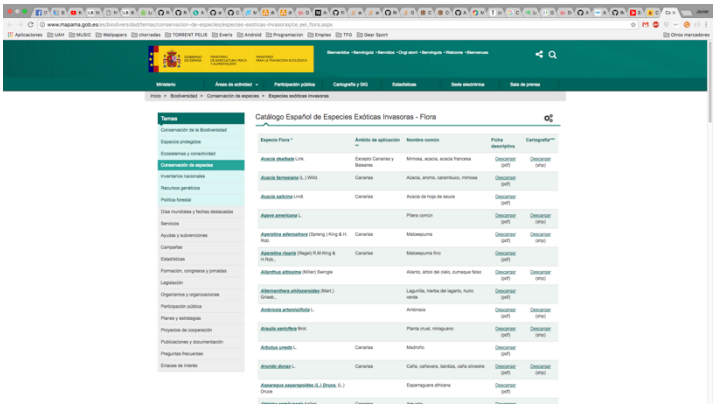
10. Documentación de la librería retrofit - https://github.com/square/okhttp/wiki/Recipes



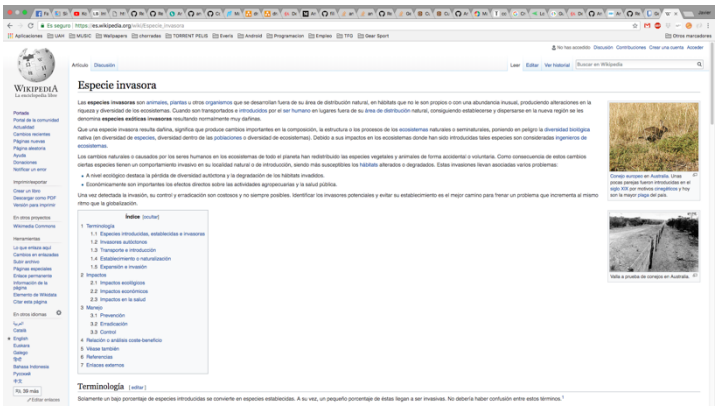
11. Learn2Crack - Step in to Learn. (2018). Android Upload Image using Retrofit #2. https://www.learn2crack.com/2017/08/upload-image-using-retrofit.html.



12. Mapama.gob.es. (2018). Catálogo Español de Especies Exóticas Invasoras - Flora. Disponible en: http://www.mapama.gob.es/es/biodiversidad/temas/conservacion-de-especies/especies-exoticas-invasoras/ce_eei_flora.aspx



13. Especie invasora. Wikipedia - https://es.wikipedia.org/wiki/Especie_invasora



14. Noticias Universia España. (2018). Estudio sobre las plantas invasoras en España. Disponible en: http://noticias.universia.es/ciencia-nn-tt/noticia/2008/01/31/585113/estudio-plantas-invasoras-espana.html

15. Sánchez, M. (2018). ¿Qué son las plantas vasculares?. Jardinería On. Disponible en: https://www.jardineriaon.com/que-son-las-plantas-vasculares.html

16. Resources MongoDB. (2018).] Disponible en: https://resources.mongodb.com/mongodb-architects/mongodb-architecture-guide?_ga=2.40435362.1619009320.1529528515-1508131472.1516901771

17. Resources Mysql.com. (2018). MySQL. Disponible en: https://www.mysql.com/

18. Db-engines.com. (2018). Couchbase vs. MongoDB Comparison. Disponible en: https://db-engines.com/en/system/Couchbase%3BMongoDB

19. TechCrunch. (2018). Sun Picks Up MySQL For $1 Billion; Open Source Is A Legitimate Business Model. [online] Disponible en: https://techcrunch.com/2008/01/16/sun-picks-up-mysql-for-1-billion-open-source-is-a-legitimate-business-model/.

8. Anexo

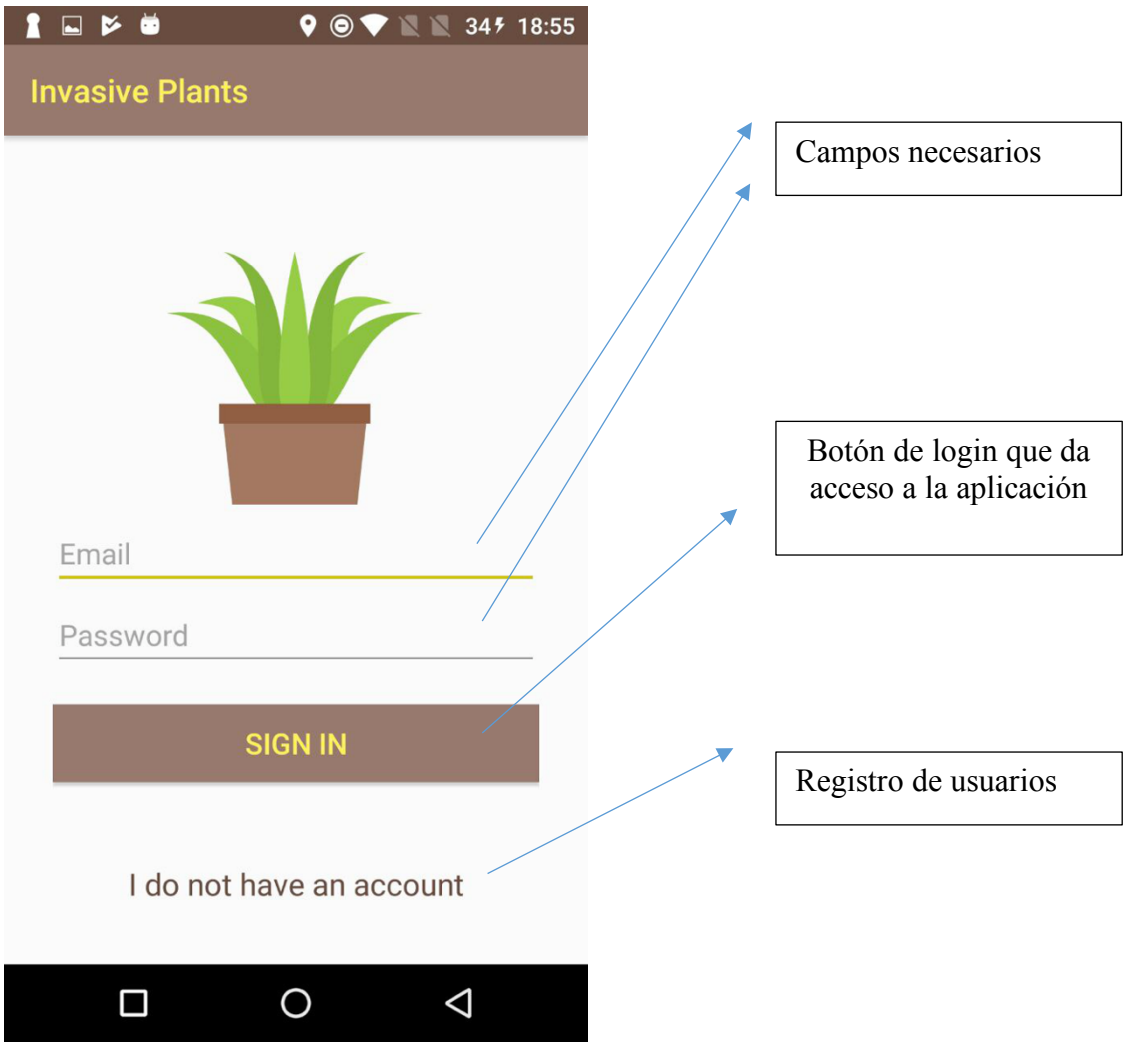
Documentación complementaria adicional.

8.1. Manual de usuario

En este apartado se procederá a la explicación de uso de nuestra aplicación y así poder hacer un buen uso de ella

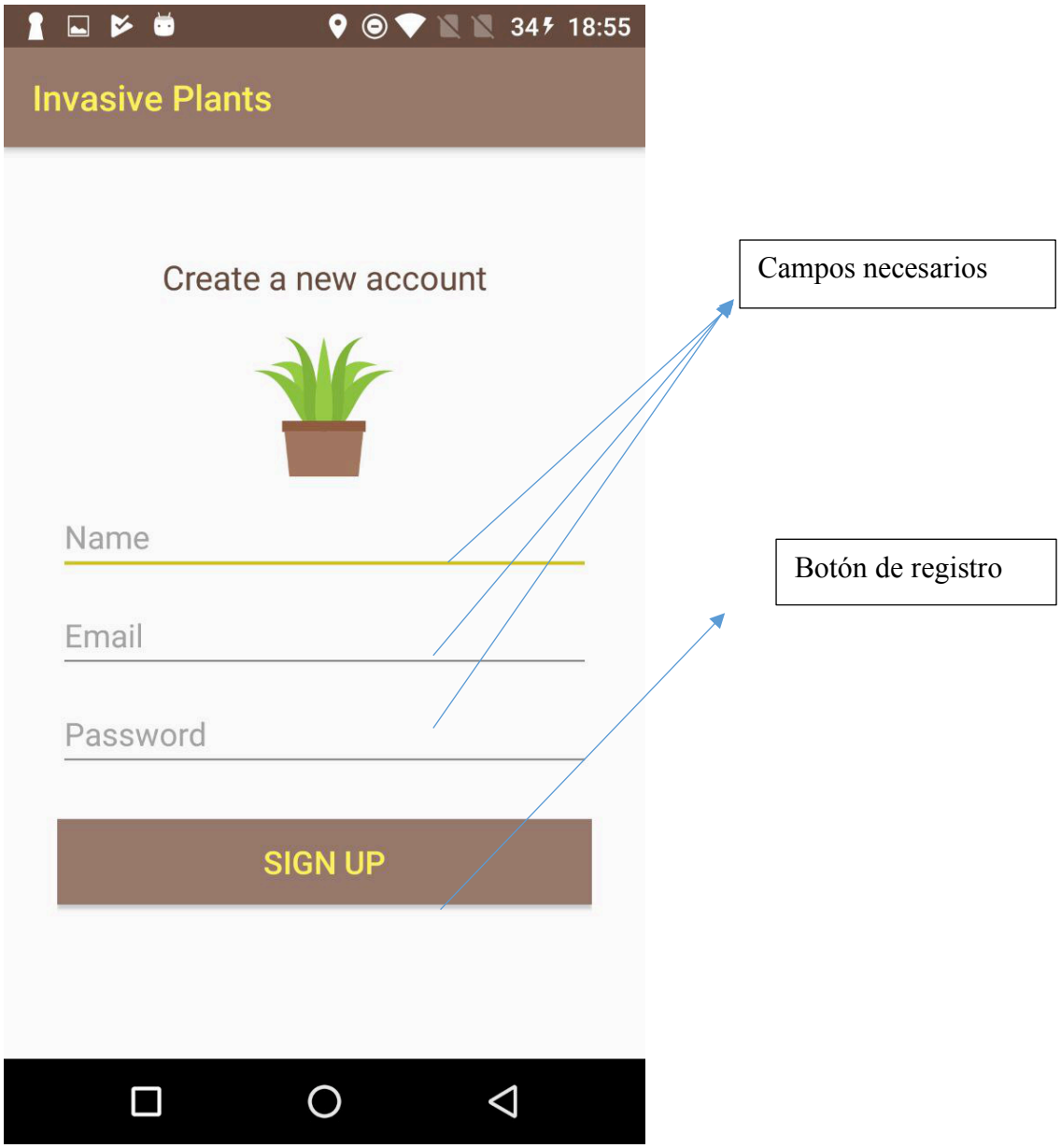
8.1.1. Login

Nada más iniciar la aplicación, esta nos pedirá un usuario y contraseña, en caso de tener cuenta, sino nos deberemos ir hacia el registro, este lo encontramos pulsando en el texto “I do not have an account”.



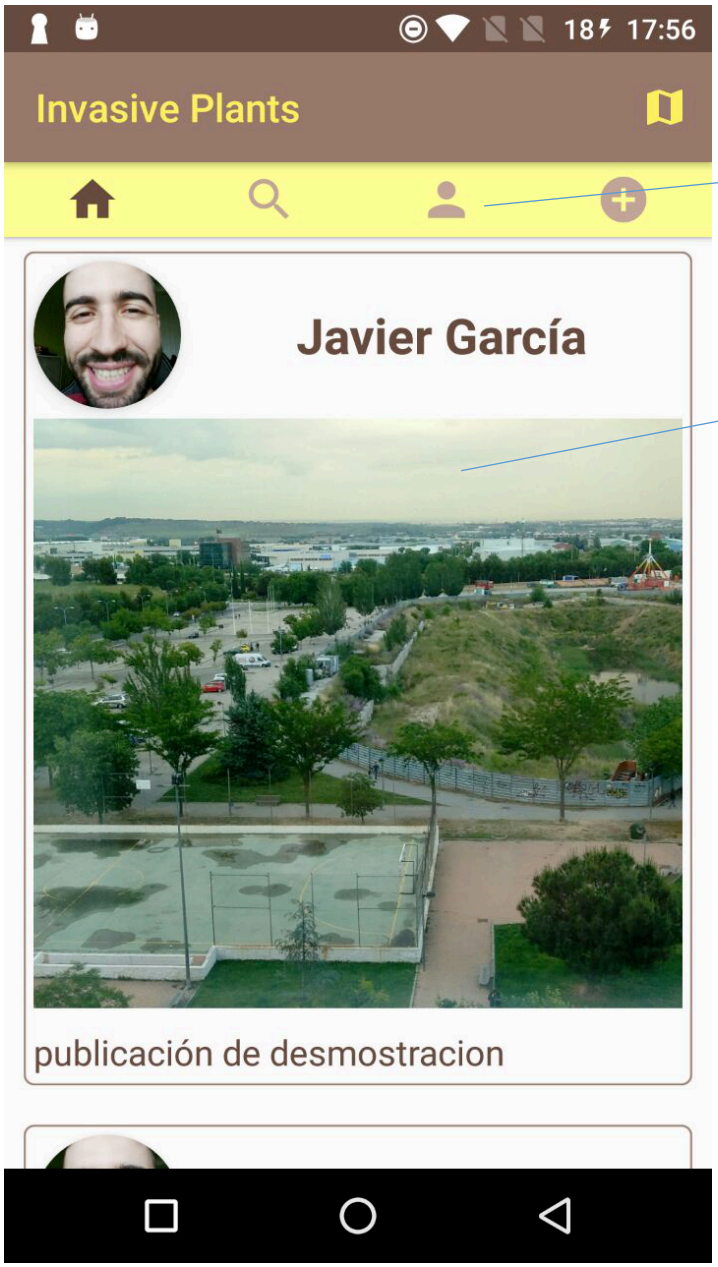
8.1.2. **Registro**

Como se ha comentado antes, en caso de no disponer de cuenta el usuario puede crearse una nueva, al crear el usuario, la aplicación le redirigirá a la ventana de log in para poder acceder a la aplicación con esta ya creada.



8.1.3. **Ventana principal**

En ella encontramos las publicaciones de usuarios ordenadas por fecha y la barra de navegación, en la que encontramos el botón de mostrar mapa, en el se mostrara un mapa con los marcadores que los usuarios han ido dejando con sus publicaciones para interactuar con la aplicación.

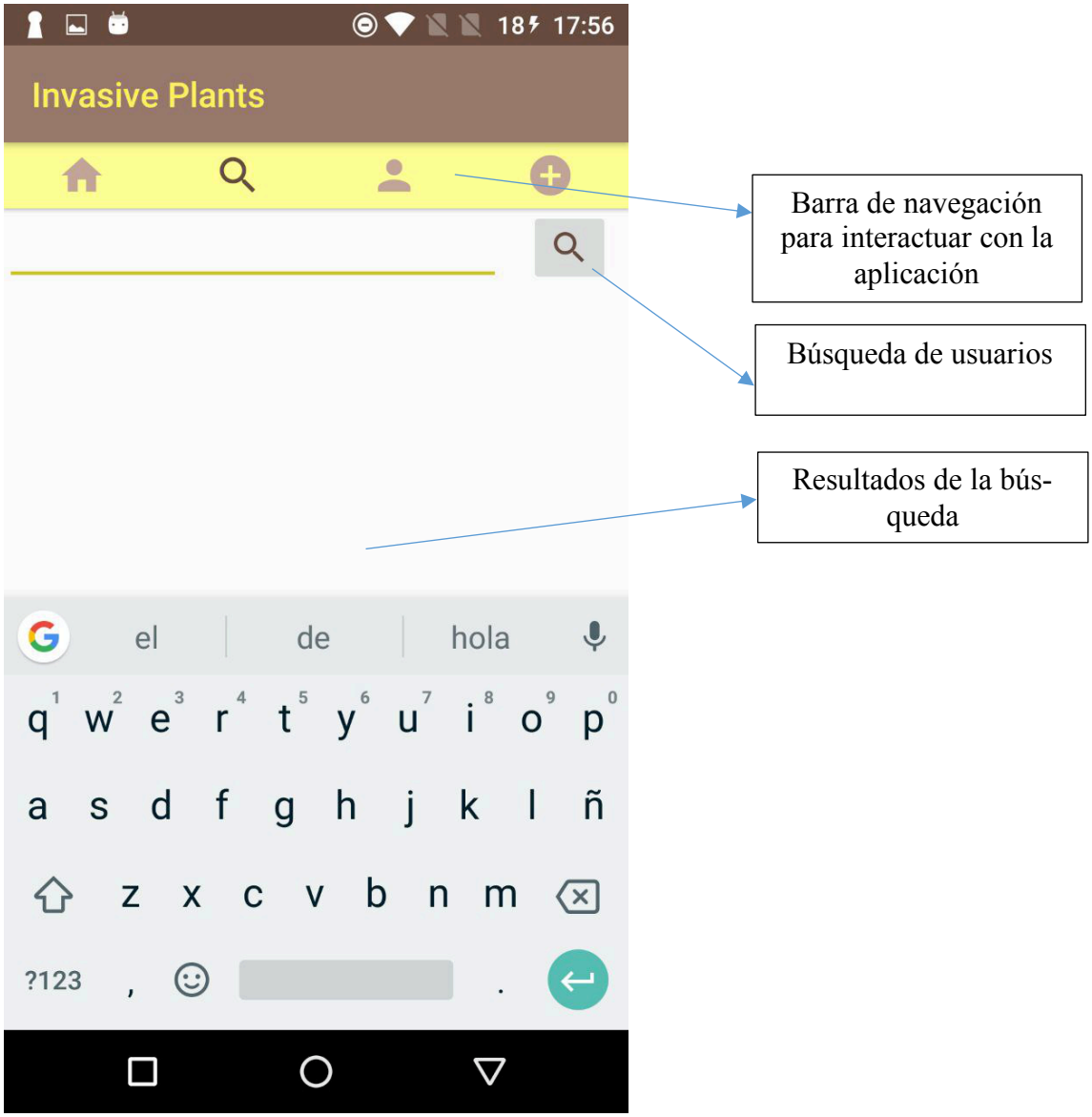


Barra de navegación para interactuar con la

Publicaciones de los usuarios

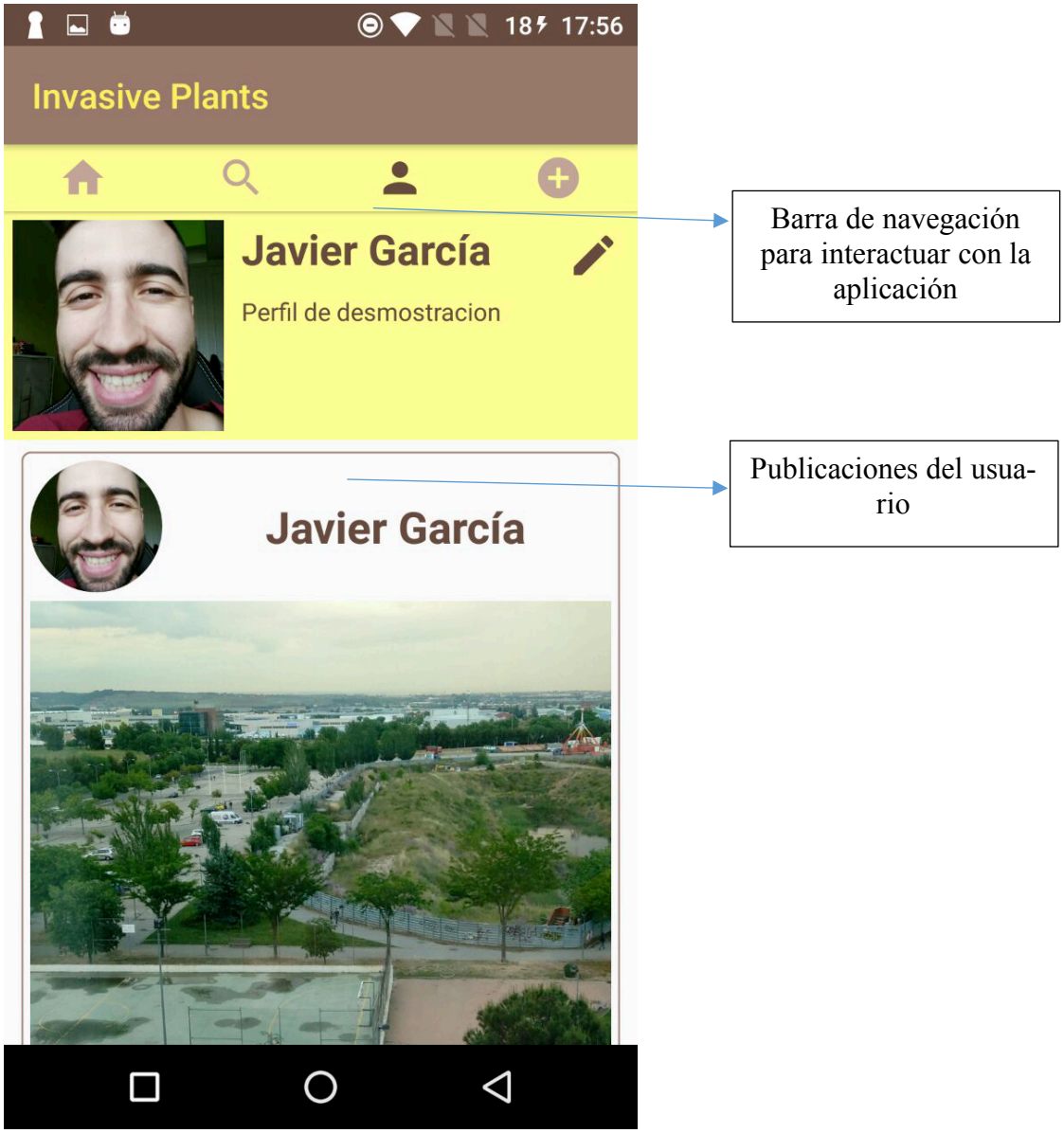
8.1.4. **Búsqueda de usuarios**

En esta actividad se permite realizar una búsqueda de usuarios, primero se introduce un nombre y este nos mostrara los resultados, en caso de existir coincidencias podremos acceder al perfil de usuario de nuestro resultado.



8.1.5. Vista de pagina de perfil de usuario

A continuación, en esta actividad podremos ver las publicaciones personales ordenadas por fecha de publicación que hayamos ido haciendo, nuestro nombre, descripción y foto de perfil. Además podremos realizar modificaciones en los 3 apartados anteriormente mencionados pulsando en el botón lápiz, para poder editar nuestro perfil.

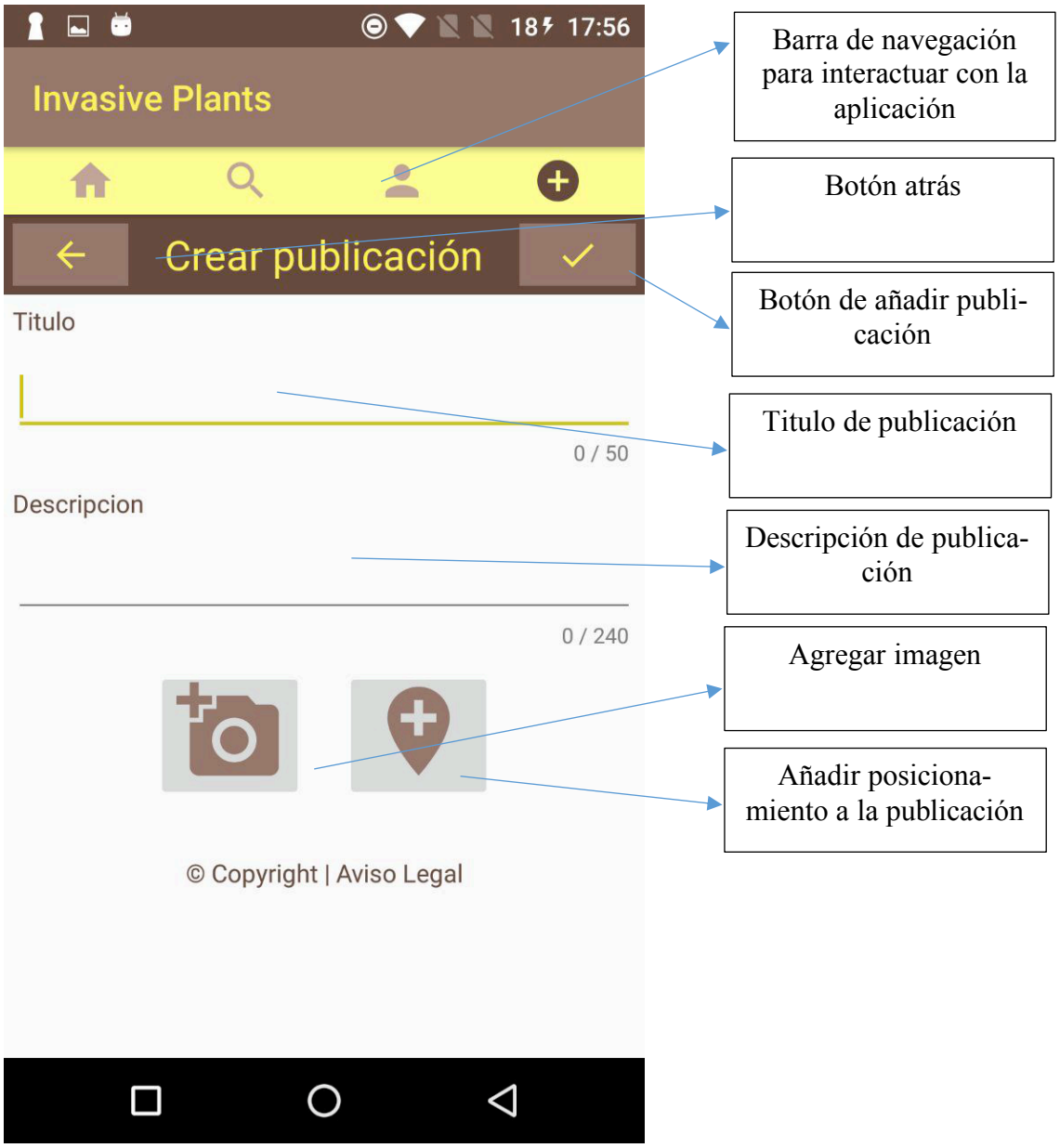


8.1.6. Actividad para modificar los datos del usuario

Como se puede observar, aquí podremos editar nuestro nombre de usuario, foto y descripción. Así los otros usuarios serán capaces de tener una idea de nuestras aspiraciones o intereses.

8.1.7. Vista de pagina de crear publicación

Si decidimos realizar una publicación, lo podremos realizar en esta ventana, en la cual se especifica los requisitos que debe cumplir para poder publicarse, como son un titulo una descripción, las coordenadas GPS y En esta actividad se podrán crear publicaciones, estas deben tener un titulo y una descripción, además de una fotografía y las coordenadas GPS en el momento de realizar la publicación.



8.1.8. **Vista de la actividad de mapa**

Aquí el usuario puede, de un vistazo, ver todas las publicaciones o hallazgos que han hecho otros usuarios. Podremos visualizarlos en Google Maps para que este nos guie hasta el.

Mapa con los marcadores de las publicaciones de usuarios

Universidad de Alcalá
Escuela Politécnica Superior

ESCUELA POLITECNICA
SUPERIOR