Trabajo Fin de Grado

Buscador de Fistosanitarios sobre iOS y FiWare

Autor

Pedro Ismael Artero Gasca

Director

F. Javier Zarazaga Soria

Escuela de Ingeniería y Arquitectura

2016

Me gustaría agradecer este trabajo a mi familia,

a los amigos que me han acompañado durante esta carrera

y a todos los profesores con los que he tratado,

con especial atención a mi tutor del proyecto, Javier Zarazaga.

Resumen ejecutivo

Este trabajo aborda el desarrollo íntegro de una primera versión viable de una aplicación, sobre el Sistema Operativo iOS, destinada a la búsqueda y consulta de diversos fitosanitarios que se puedan utilizar en los diferentes países de la Unión Europea. Para ello se parte de los resultados del proyecto Pestifinder que ya ofrece a través de Internet servicios de búsqueda análogos a los desarrollados en este proyecto. Este proyecto, desarrollado sobre FIWARE, ofrece una API mediante servicios web que posibilita el acceso a la funcionalidad sin necesidad de pasar por la aplicación que se ejecuta en el navegador de Internet. Es esta la puerta que se ha utilizado para desarrollar este TFG.

La aplicación ha sido construida en el lenguaje de programación Swift, y para su correcto funcionamiento también se ha partido del diseño de una arquitectura software flexible orientada a dar fácil mantenimiento ante la evolución de la oferta de servicios de Pestifinder. Además la aplicación ofrece un comportamiento con conexión a Internet y otro sin dicha conexión que le permite la continuidad funcional.

Finalmente, se establecen diferentes aproximaciones de modelo de negocio que posibilitan rentar el trabajo desarrollado y establecer una línea de crecimiento y evolución del sistema gestado en este proyecto.

Declaración de Autoría

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

D./Dª. Pedro Ismael Artero Gasca,

con nº de DNI 73131811 - E en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Grado, (Título del Trabajo) Buscador de Fistosanitarios sobre iOS y FiWare

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 21 de Abril de 2016

Fdo: Pedro Ismael Artero Gasca

Índice

1 Contexto del TFG

1.1 El proyecto Pestifinder

Si un agricultor español quiere vender sus productos en mercados europeos debe tener especial cuidado en los fitosanitarios que utiliza ya que en los países de la Unión Europea no están autorizados los mismos pesticidas. Esto supone un gran esfuerzo porque se debe consultar la normativa de los distintos países donde se quiera distribuir el producto y asegurarse de que los pesticidas que va a utilizar están permitidos. En estos momentos hay un marco legal común a nivel europeo para conseguir el uso sostenible de los plaguicidas[1], pero su transposición a las legislaciones nacionales no siempre se ha realizado de la misma manera, y es por este motivo por el que puede ocurrir el hecho de que un pesticida esté permitido en un país, pero no en otro.

El proyecto Pestifinder, desarrollado por la empresa GeospatiumLab (http://www.geoslab.com) en colaboración con la Universidad de Zaragoza, ofrece una solución al problema anterior. Dicho proyecto ofrece un buscador web (http://pestifinder.geoslab.com) que ayuda a los agricultores y asesores en la gestión de plagas a elegir el fitosanitario más adecuado para sus cultivos en función de distintos parámetros, entre los que se encuentra el país destinatario. Los resultados de dicho buscador muestran los distintos fitosanitarios que se podrán utilizar para la consulta realizada, así como una información detallada de cada uno de estos. Sin embargo se trata de una aplicación web que cuenta con algunas limitaciones asociadas a su propia versatilidad de multiplataforma (dado que puede ejecutarse sobre cualquier navegador). Así mismo, esta solución se encuentra completamente vinculada al proyecto PestiFinder que a su vez ofrece muchas posibilidades de la mano de su arquitectura basada en servicios web (tal y como se indica en la siguiente sección). Esta arquitectura de servicios web es la que posibilita que se puede desarrollar un sistema que acceda a los servicios ofrecidos por Pestifinder sin necesidad de ejecutar el buscador web directamente. Es esta la puerta tecnológica que va a ser usada en este proyecto para poder ofrecer nuevas capacidades.

1.2 Marco tecnológico

Pestifinder es un proyecto que se construye sobre FIWARE. FIWARE o FI-WARE (https://www.fiware.org) es una plataforma, impulsada por la Unión Europea, para el desarrollo y despliegue global de aplicaciones de Internet del Futuro. FIWARE intenta proveer de una arquitectura totalmente abierta, pública y libre así como de un conjunto de especificaciones que permita a los desarrolladores, proveedores de servicios, empresas y otras organizaciones desarrollar productos que satisfagan sus necesidades, sin dejar de ser abierta e innovadora. FIWARE fue financiado por el VII Programa Marco de la Unión Europea dentro de su proyecto de colaboración público privada para el Internet del Futuro (FI-PPP - Future Internet Public-Private Partnership).

Al tratarse de un servicio alojado en esta plataforma, para acceder al mismo es necesario disponer de unas credenciales específicas de acceso. Con el fin de simplificar esta conexión, y

[1] directiva 2009/128/CE, http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2009:309:0071:0086:es:PDF

tal y como se verá más adelante en esta memoria, se ha desarrollado una arquitectura que incluye un servicio web que enmascara este acceso al resto del sistema construido. De este modo se consigue asilar el resto de desarrollos de las peculiaridades de acceso a la plataforma. Esto es importante dado que la misma no se encuentra en un estado 100% estable y está sujeta a cambios (que en alguna ocasión afectan a los protocolos de conexión). De este modo, ante una modificación de la plataforma, no será necesario reconstruir más que una parte de este servicio web.

2 Motivación y objetivos

Tal y como se ha mencionado anteriormente, Pestifinder ofrece tan solo posibilidades de funcionalidad basadas en el navegador de Internet. Esto acarrea limitaciones a la hora, por ejemplo, de poder ofrecer funcionalidades sin conexión a datos. Este proyecto se motiva en la explotación de las posibilidades que los servicios web subyacentes a Pestifinder ofrecen para el desarrollo de nuevos productos que añadan valor y permitan generar nuevas oportunidades de negocio.

El objetivo de este proyecto será principalmente desarrollar una aplicación móvil sobre el Sistema Operativo iOS (Sistema Operativo móvil desarrollado por Apple para sus dispositivos) que ofrezca un nivel de funcionalidad análogo al que se ofrece desde PestiFinder. Además, se realizará un Middleware entre la aplicación móvil y la aplicación web, se especificará tecnológicamente un producto viable, y se definirá un modelo de negocio para la aplicación.

La aplicación móvil se desarrollará para el Sistema Operativo iOS, utilizando el lenguaje de programación Swift (Swift se trata de un lenguaje de programación multiparadigma, utilizado para programar para los diferentes Sistemas Operativos de Apple). Se ha elegido este lenguaje puesto que es el lenguaje utilizado habitualmente para desarrollar las aplicaciones de iOS en las versiones más actuales. Esta aplicación se desarrollará en Xcode, que es el entorno de desarrollo que ofrece Apple para compilar y ejecutar Swift.

Dado que se trata de una aplicación para móviles, se ha decidido utilizar replicación de datos, en medida de lo posible, cuando sea necesario y teniendo en cuenta la disposición de una conexión WiFi y así evitar gastar una posible tarifa de datos móviles.

3 Análisis y Diseño

En primer lugar, antes de plantear una solución al problema anteriormente descrito, se ha optado por realizar un análisis del contexto en el que se encuentra este proyecto, de este análisis se sacarán los requisitos mínimos que se deberán cubrir en el desarrollo, unos casos de uso de las actividades que podrá realizar el usuario en la aplicación, así como un análisis del marco tecnológico con el que nos encontramos antes de empezar.

Una vez analizado el problema, antes de proceder a implementar se realizará una tarea de diseño de la solución que abarcará tanto un diseño gráfico como tecnológico para tener una primera idea más concreta de la aplicación antes de proceder a su implementación.

3.1 Requisitos

Tras estudiar el problema planteado, se han obtenido los siguientes requisitos, que deberá satisfacer la aplicación antes de ser entregada.

- Requisitos Funcionales

Número	*Descripción del Requisito*
RF1	La aplicación debe permitir crear un usuario en base a una cuenta de correo electrónico
RF2	La aplicación debe permitir al usuario dar valor a los parámetros de búsqueda, que serán: Cultivo, Plaga, Nombre del fitosanitario, Sustancia Activa, Marca y Destino
RF3	Los valores de parámetros disponibles que ofrezca la aplicación deberán actualizarse
RF4	La aplicación debe mostrar los fitosanitarios que se adecuen a la búsqueda hecha por el cliente
RF5	La aplicación debe ofrecer información detallada de un fitosanitario concreto
RF6	La aplicación debe ofrecer un servicio con y sin conexión a internet.

Tabla 1. Requisitos Funcionales

- Requisitos No Funcionales

Número	*Descripción del Requisito*
RNF1	La solución obtendrá los datos que ofrezca Pestifinder
RNF2	La aplicación replicará datos, según se crea conveniente, para ofrecer un comportamiento sin conexión a internet.
RNF3	La aplicación deberá poder correr en el Sistema Operativo iOS 8 o superior. *
RNF4	No se expondrán las credenciales de acceso a Pestifinder, ni a Fi-Ware, en la aplicación.
RNF5	Se establecerá un mecanismo de rentabilización de la aplicación.
RNF6	Se almacenarán las direcciones de correo electrónico de los clientes que usen la aplicación

Tabla 2. Requisitos No Funcionales

*Esta decisión se ha tomado dado que es una minoría, y cada vez tienden a ser menos, los dispositivos que tienen instalados Sistemas Operativos anteriores. Actualmente, 1 de abril de 2016, un 95% de los usuarios de iOS utiliza la versión iOS 8 o superior (79% iOS 9). [1]

3.2 Casos de uso

Para especificar el comportamiento que ofrecerá la aplicación móvil al usuario se ha desarrollado el diagrama de Casos de Uso que se muestra a continuación.

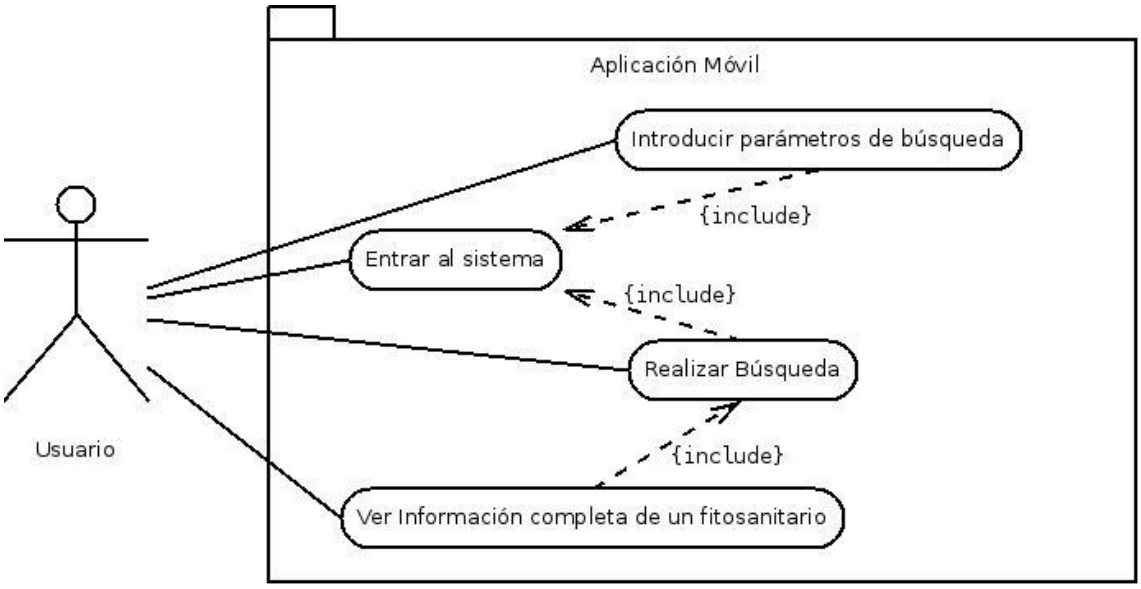

Ilustración 1. Diagrama de Casos de Uso

Este diagrama contiene un Actor, que es el usuario final que interactúe con la aplicación, y diversas funcionalidades que se pueden ver en los casos de uso, y que se detallan a continuación:

- Entrar al sistema: Para entrar a la aplicación, se deberá introducir un correo electrónico válido en la aplicación (si ya se ha registrado se comprobará su existencia, y si no estaba registrado se registrará), si ya se ha introducido en una primera entrada a la aplicación esta deberá recordarlo y no se le pedirá nada al usuario.

- Introducir parámetros de búsqueda: Para llegar a este caso, primeramente se debe estar registrado en el sistema. En este caso se podrán filtrar los datos para realizar la posible posterior búsqueda.

- Realizar Búsqueda: Se debe haber entrado en la aplicación para realizar este caso. En esta acción se realizará la búsqueda de los fitosanitarios encontrados según los parámetros de búsqueda seleccionados y se le mostrarán al usuario. En caso de que el usuario no hubiera seleccionado ningún parámetro de búsqueda, no se filtran los resultados y se le muestran todos los fitosanitarios al usuario.

- Ver información completa de un fitosanitario: Una vez realizada una búsqueda, se puede seleccionar un pesticida y la aplicación mostrará la información del producto correspondiente.

3.3 FI-Ware

Para analizar el marco tecnológico en el que se encuentra Pestifinder, hay un elemento a destacar por su novedad y desconocimiento como es FI-Ware. FI-Ware se trata de un proyecto, impulsado por la Unión Europea, que tiene por objetivo ofrecer una plataforma de middleware sobre la que desarrollar aplicaciones y servicios web.

Pestifinder ofrece una API (Interfaz de Programación de Aplicaciones) de Servicios Web. Al residir esta aplicación web en la plataforma FI-Ware, esta última será la encargada de gestionar las credenciales de autenticación del cliente y controlar el acceso de este a Pestifinder. El procedimiento de FI-Ware para acceder a una aplicación es el siguiente:

1. Cliente se registra en FI-Ware.
2. Cliente compra la aplicación deseada.
3. Cliente puede acceder a la URL del producto adquirido, previa autenticación con FI-Ware.

Ahondado más en este proceso de autenticación, se sigue el protocolo OAuth2 [2] . Dicho protocolo es comúnmente utilizado para el acceso de API's (Por ejemplo, las API's de Google o Twitter, entre otras, lo utilizan). Un resumen de este protocolo, es que el cliente se autentica en un servidor y este le devuelve unas credenciales de acceso (en este proceso se llamará Token de acceso), posteriormente se comprobará en la API si las credenciales (token) tiene los privilegios de acceso necesarios. De esta manera, se evita la exposición de la contraseña.

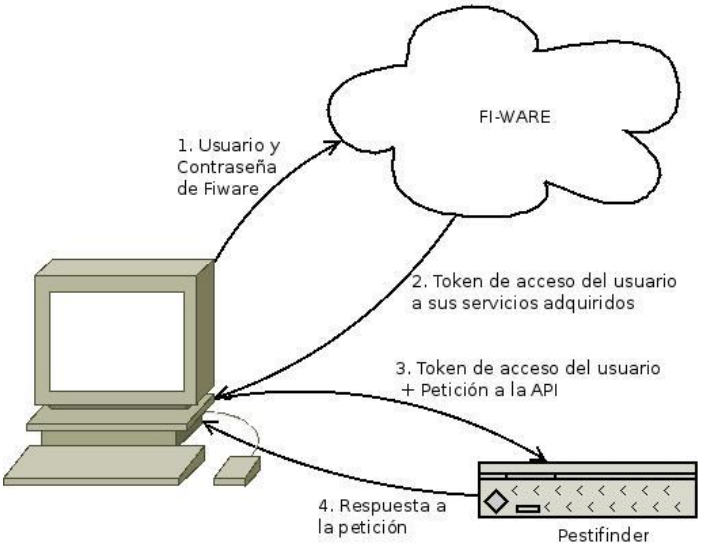

Ilustración 2. Estructura de Pestifinder

El funcionamiento de la autenticación del cliente y el acceso a Pestifinder tiene el comportamiento descrito en la imagen, y en el que se entiende como la contraseña de FI-Ware no se expone a los servicios alojados en la plataforma y como en la comunicación con Pestifinder tendrá un papel clave el token proporcionado por FI-Ware.

3.4 Mapa de Navegación

Se ha realizado un mapa de navegación para representar las pantallas que contendrá la interfaz gráfica, así como la navegación entre ellas.

Ilustración 3. Mapa de Navegación

Este mapa de navegación se ha diseñado buscando que tenga un comportamiento similar al que ofrece la versión web de Pestifinder. A pesar de que la información que debe mostrarse al usuario será de una manera similar a la representada, en la implementación se adecuará la interfaz gráfica a las guías de estilo que marca Apple para iOS.

Para no sobrecargar la imagen de líneas con direcciones, no se han dibujado las direcciones de retorno a la pantalla anterior. Esta acción se podrá hacer desde todas las

pantallas pulsando en el botón situado en la esquina superior izquierda, excepto para volver a la primera pantalla, en la que se pide un email para registrarse (entrar al sistema), ya que esta petición solo debería hacerse una vez, la primera vez que se utiliza la aplicación.

A continuación se va a explicar brevemente el comportamiento de la pantalla menú (pantalla 2), ya que es la raíz desde la que se accede a mayor número de pantallas. Desde esta escena se accederá a las siguientes:

- Info: Tras pulsar en el botón de la esquina superior derecha, nos llevará a una pantalla en la que se mostrará una breve descripción del producto diseñado, aplicación Pestifinder.

- Lista: A esta vista se accederá tras pulsar en todas las opciones de parámetros excepto la llamada “Nombre de Pesticida”. En esta pantalla dispondremos de una lista de la que podremos seleccionar el elemento por el que se desee filtrar en la búsqueda. Dado que esta lista es bastante grande en algunos casos, se ha dispuesto una barra de búsqueda para que la opción deseada sea más fácil de encontrar por el cliente.

- NombrePesticida: A esta vista se accederá tras seleccionar la opción de parámetro llamada “Nombre de Pesticida”, y servirá para introducir un filtro por nombre en la búsqueda de fitosanitarios.

- Resultados: Se accederá a esta pantalla tras seleccionar el botón de búsqueda. Aquí se mostrarán los resultados de la búsqueda mostrándose el nombre del fitosanitario y la marca a la que pertenece. Al igual que en la pantalla “Lista” se dispondrá de una barra de búsqueda para poder filtrar la búsqueda total. Una vez encontrado el elemento buscado, se podrá seleccionar e ir a la pantalla “Producto” en la que se mostrará una descripción mucho más detallada del fitosanitario seleccionado.

3.5 Arquitectura de la solución

Una vez hecha una primera aproximación de lo que será la aplicación de manera gráfica de cara al usuario, se va a proceder al diseño de una arquitectura de los diferentes componentes que se van a utilizar para el funcionamiento de la aplicación de manera que se acople al marco tecnológico en el que se encuentra Pestifinder y que se resume en el apartado anteriormente descrito de FI-Ware.

Dicha solución ha ido cambiando y mejorando a lo largo del desarrollo de la aplicación. En un primer momento, la solución más cómoda era obtener el token de acceso a Pestifinder e introducirlo en la aplicación, esta solución se desechó ya que estas credenciales de acceso tienen una fecha de expiración, y en conclusión no era una solución duradera.

Posteriormente se planteó que la aplicación se autenticara en FI-Ware y obtuviera esta las credenciales de acceso a Pestifinder. Esto se podía hacer de dos maneras, la primera era introducir en la aplicación un usuario y contraseña de FI-Ware y que esta obtuviera el token definitivo para comunicarse con la API web. Esto expone credenciales a las que una tercera persona no debería tener acceso. La segunda opción sería que un usuario de nuestra aplicación entrara con una cuenta de FI-Ware, esto haría que el acceso a nuestra aplicación para alguien que no tenga una cuenta previa sea un proceso tedioso. Por estas razones se optó para diseñar otra solución.

A continuación se ha desarrollado un mapa conceptual de la solución final planteada.

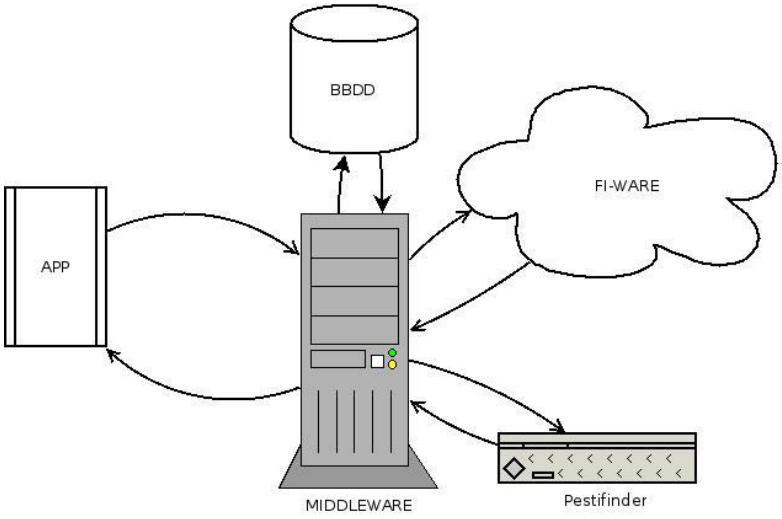

Ilustración 4. Arquitectura de la solución

En esta solución se introduce un middleware, un servidor intermediario entre nuestra aplicación y el servicio web Pestifinder. De esta manera, como reza el refranero español, “se matarían dos pájaros de un tiro”. Por un lado no se expondría ninguna credencial en la aplicación, y además, Pestifinder en un futuro puede cambiar sus servicios web, y ofrecernos las respuestas de otra manera a como nos las ofrece ahora, con esta solución no tendríamos que tocar la aplicación de iOS (con todo lo que esto conllevaría, código fuente, que las versiones anteriores no funcionen, que haya que subir nueva aplicación a la web…) si no que solamente deberíamos manipular la información que nos ofrece el servicio y devolverla al cliente de manera análoga a como lo hacía.

Otra ventaja de esta solución es que se trata de una solución escalable, y en un futuro se podría emplear este servidor intermedio para futuras mejoras.

Diagrama de secuencia

Para ayudar a comprender la solución diseñada, se ha realizado un diagrama de secuencia para que se vea de una forma más precisa la interacción entre los distintos componentes del sistema.

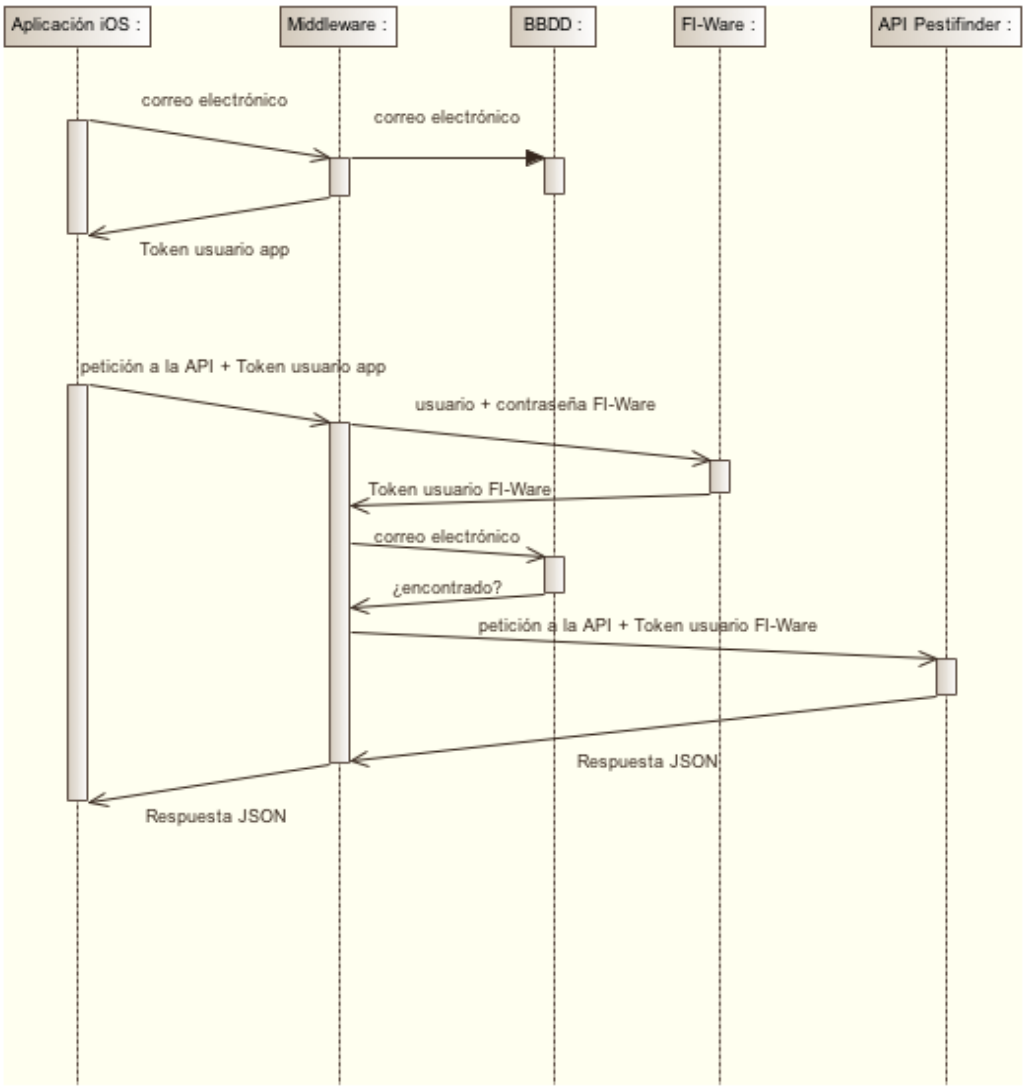

Ilustración 5. Diagrama de Secuencia

1. La aplicación envía el identificador de usuarios (correo electrónico del usuario) al servidor (Middleware en la ilustración). Este comprueba si está en la base de datos, y si no crea el usuario en ésta. Tras esta comprobación le devuelve un token de acceso a nuestro servidor.
2. La aplicación envía una petición (obtener una búsqueda, una lista…) a nuestro servidor, incluyendo el token obtenido en el paso anterior.
3. Nuestro servidor comprueba si el token es válido.

4. El Servidor tiene las credenciales de acceso a FI-Ware (usuario y contraseña), obteniendo de este el token de acceso a Pestifinder.
5. El servidor trasmite la petición a la API de Pestifinder con el token obtenido de FI-Ware y ésta le devuelve el resultado.
6. Nuestro servidor devuelve el resultado a la aplicación.

3.6 Modelo de negocio

Cuando se realiza un proyecto como este hay que analizar y diseñar diferentes aspectos y uno de ellos es la viabilidad, tanto tecnológica como comercial. Una vez se despliegue la aplicación, hay que rentabilizarla. Para dicho propósito se plantearán varias vías de negocio y se decidirá cuales se van a implementar en esta primera versión viable del producto.

3.6.1 Banners de Publicidad

Uno de los métodos de rentabilización de una aplicación para móvil más utilizado es la publicidad. Estos pueden ser de varios tipos y aparecer de diferentes formas. En esta aplicación se ha optado por introducir un banner de publicidad en la parte inferior de la pantalla del menú principal de la aplicación, así como de banners que ocuparán más espacio en la pantalla y se insertarán en las esperas de resultados, cuando se intuya que dicha espera vaya a ser suficiente como para que el usuario tenga tiempo para visualizarlo correctamente. Se han elegido estos escenarios para mostrar publicidad y de este tipo, de manera que interrumpa en la menor medida de lo posible la comodidad del usuario a la hora de utilizar esta aplicación.

En un primera versión se han optado por introducir los banners que proporciona Apple en su plataforma de publicidad llamada iAd, se ha optado por esta plataforma debido a la facilidad que ofrece en la integración de su publicidad con las aplicaciones iOS.

A continuación se muestran situaciones en las que se integraría la publicidad.

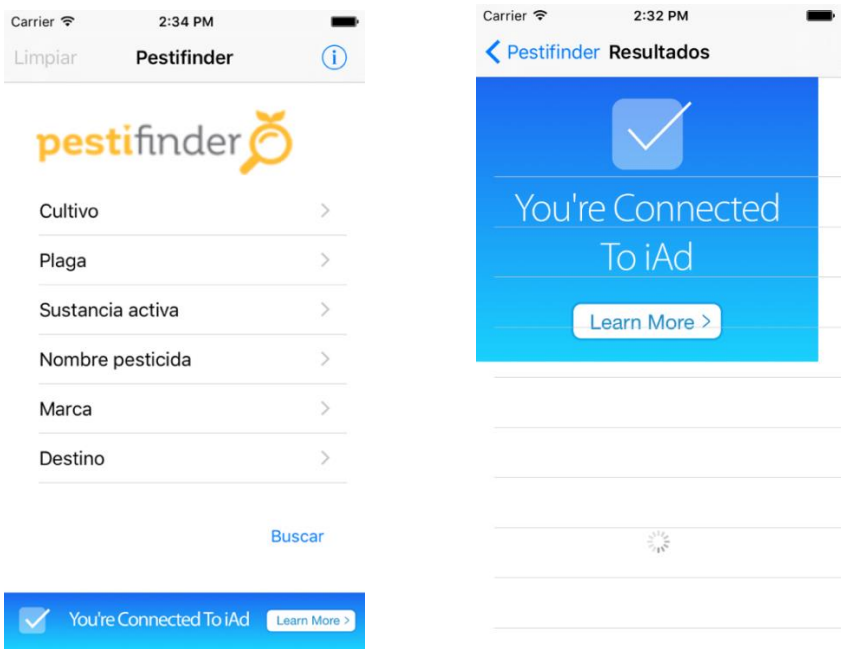

Ilustración 6. Ejemplo de publicidad

Posteriormente, se propone avanzar más en este ámbito e implementar la publicidad con otros proveedores de esta, e incluso desarrollando un sistema de gestión de publicidad propio en el que se facture al publicitado los tiempos que se han visto sus anuncios y donde los anuncios estuvieran precargados (por ejemplo, cargando la publicidad con conexión a red WiFi). De esta manera no habría que recurrir a una conexión a internet para mostrar publicidad y se evitaría sobrecargar la red de datos, ya que para los usuarios que tienen una conexión de datos con un ancho de banda bajo, esto puede ser una verdadera molestia.

3.6.2 Envío de publicidad

Otra manera de introducir publicidad sería aprovechando el sistema de notificaciones que tiene el Sistema Operativo iOS, y mandar publicidad al usuario por medio de esta vía. Para realizar esto, lo ideal sería almacenar más información del usuario (GPS, cultivos que más consulta, plagas…) para enviarle publicidad relacionada con lo que tiene y lo que podría llegar a necesitar. Por ejemplo, un agricultor que ha descargado la aplicación y suele buscar maíz, se le podría mandar publicidad sobre maquinaria que no procedería enviarle a un señor que tiene almendros.

Dado que tenemos el correo electrónico de nuestros usuarios, este se podría utilizar para enviarles correos electrónicos con un criterio similar al utilizado en el envío de publicidad por notificaciones.

3.6.3 Posicionamiento

Lo que se va a implementar es un buscador y, si hay muchos resultados (se ordenarán alfabéticamente para hacer más fácil la búsqueda al usuario), cabe la posibilidad de que el usuario solamente vea los primeros resultados. Es en este caso, cuando se podría vender a las marcas interesadas tener un buen posicionamiento en este buscador.

3.6.4 Vender el servicio

Otro método que se ha pensado para rentabilizar este sistema es vender el servicio de búsqueda a una empresa dedicada al sector agrícola a la que le pudiera interesar incluir la búsqueda de fitosanitarios en una aplicación o sistema propio. Por ejemplo, empresas como Gaysa (que se dedica a la venta de maquinaria fitosanitaria) o Bayer (dedicada a ámbitos químico-farmacéuticos, entre ellos los fitosanitarios) podrían interesarles introducir un servicio de búsqueda que ofrece este sistema (incluso filtrado para que solamente salgan los productos de algunas de estas).

4 Implementación

Una vez realizado el análisis y diseño de esta aplicación se va a proceder a su implementación. La implementación se ha dividido en diferentes etapas. Primeramente se creó la conexión entre la aplicación (cliente) y la API Pestifinder sin tener en cuenta un servidor intermedio que gestione la comunicación. Posteriormente se implementó la lógica de la aplicación, a la par que su interfaz gráfica. Una vez el tronco de la aplicación estaba terminado, se desarrolló el servidor intermediario (debido a que el cliente se preparó para cambiar su servidor de datos por medio de un fichero, esta tarea fue muy sencilla). Llegados a este punto se ha procedido a desarrollar los puntos que le dan un plus a la aplicación, como son la publicidad o la replicación de datos, así como el criterio para la disposición de éstos.

Para el desarrollo de esta aplicación se han utilizado las siguientes tecnologías (algunas de estas se explican más detalladamente en el **Anexo I**):

- Swift
- Core Data
- PHP
- MySQL
- Amazon Web Services

A continuación se van a detallar de una manera más detallada las implementaciones del Servidor, Cliente.

4.1 Servidor

Para desarrollar el servidor se ha utilizado el lenguaje PHP, debido a su facilidad para el desarrollo y despliegue de Servicios Web. En el diagrama de secuencia, visto anteriormente (**ver diagrama de secuencia**), se ve el funcionamiento de este servidor. En esta sección explicaremos más detalladamente como ha sido su implementación y despliegue.

El servidor PHP dispone de una API con varios Servicios Web entre los que destacaremos:

- /api/access: Este servicio web recibe por GET un parámetro llamado user que corresponderá al correo electrónico del usuario. El servicio comprueba si existe esta dirección de correo en la base de datos y lo introduce en caso negativo. A continuación genera un token (credencial) de autenticación para comunicarse con el cliente. En este proceso no se intercambian contraseñas, por lo que el uso del token, podría no tener sentido en un principio, pero si se sigue adelante y se implementa una autenticación con contraseña, si que se debería utilizar dicho token.

- /api/[plagues | crops | active-substances | producers | countries] : Estos servicios trasladan la petición correspondiente, tras comprobar si el token es válido, a la API de Pestifinder y una vez recibe respuesta la redirige a la aplicación.

- /api/pesticides : Recibe los parámetros de búsqueda por método GET, se preparan para enviar a la API y se envía la petición a Pestifinder (previamente se ha comprobado el Token).

- /api/pesticides/: Recibe un identificador de un fitosanitario, se comprueba el Token del cliente y si está autorizado se le realiza dicha petición a la API y se le devuelve al cliente la información adecuada del fitosanitario que había seleccionado.

Durante el desarrollo de este proyecto se ha desplegado el servidor (tanto el servidor MySQL como el servidor PHP) en instancias de Amazon Web Services. Concretamente se han utilizado:

- Amazon RDS: Se trata de una plataforma que ofrece Amazon como servicio en la nube. Esta plataforma ofrece distintos Gestores de base de datos, entre ellos MySQL 5.7 que ha sido el seleccionado para este proyecto. Se ha seleccionado Amazon RDS como servidor de base de datos durante el desarrollo de este trabajo debido que ofertaba un tiempo gratuito y se trata de un servicio fácilmente configurable y desplegable.

- Amazon Beanstalk: Se trata de otra plataforma en la nube ofrecida por Amazon. Esta plataforma ha sido muy útil para lanzar un servidor web en tiempo muy breve, muy fácil de utilizar, pues tan apenas hay que subir un fichero comprimido de los ficheros .php. Y te ofrece una URL a la que acceder a la página web que acaba de lanzar. Este servicio se ha utilizado para situar el servidor web que realizará la función de middleware en el sistema.

4.2 Cliente

El cliente se ha desarrollado en el lenguaje de programación Swift, debido a que es el lenguaje que se utiliza habitualmente para programar aplicaciones de los sistemas operativos iOS. Esta implementación se ha desarrollado en el entorno de desarrollo Xcode, ya que es el comúnmente utilizado para este propósito. Este entorno ofrece las funcionalidades de compilar y simular la aplicación en virtualizaciones de distintos dispositivos iOS de diferentes versiones, así como una herramienta gráfica para el desarrollo de la interfaz gráfica de la aplicación.

Cabe resaltar de la implementación de la aplicación cliente que se ha seguido el patrón de arquitectura software llamado MVC (Modelo Vista Controlador), de esta manera se separa la parte lógica de la aplicación, la obtención de los datos y la representación de la información.

4.2.1 Api

En primer lugar se ha desarrollado una clase llamada Api, que se encargará de la comunicación con el servidor, esta clase se inicializará al comienzo de la aplicación, obteniendo la URL del servidor de un fichero de configuración de la aplicación. Esta clase ofrecerá varios métodos que se encargarán de realizar las peticiones al servidor correspondiente y tratar sus datos para devolverlos de la manera requerida según el objetivo. Estos métodos serán de tres tipos principalmente:

- Método init: Se llama para asignar las variables globales del objeto como la URL o el token a utilizar.

- Métodos get: Son métodos a los que se les pasan ciertos parámetros y llaman al servidor para devolver a la aplicación los datos que necesite. Estos métodos reciben

los datos en formato JSON, y lo tratan para obtener una estructura de datos que se ajuste a lo necesitado en cada caso.

- Método getToken: Se utilizará para enviar una solicitud de registro/entrada al sistema, y devuelve el token que se utilizará para interactuar con nuestro servidor. Este método se ha implementado una vez se ha realizado a posteriori del servidor intermediario.

Estos dos últimos métodos son asíncronos, de manera que la aplicación no se queda bloqueada esperando la respuesta de internet, puesto que esto podría dar lugar a largas esperas con apariencia de bloqueo.

4.2.2 Interfaz Gráfica y Controladores

Para diseñar y desarrollar la interfaz gráfica hay principalmente dos alternativas, mediante programación o mediante la herramienta gráfica que proporciona Xcode. Se ha utilizado esta última ya que se ha previsto un número de horas menor utilizando esta herramienta que desarrollando dicha interfaz elaborando código. Esta herramienta ayuda además a una visualización previa de cómo quedan los diferentes elementos de la interfaz mientras se diseña esta. Así como que también dispone de los diferentes componentes gráficos que ofrece Apple y que ayudan a seguir sus Guías de estilo. A continuación se muestra una captura del diseño de la interfaz en esta herramienta.

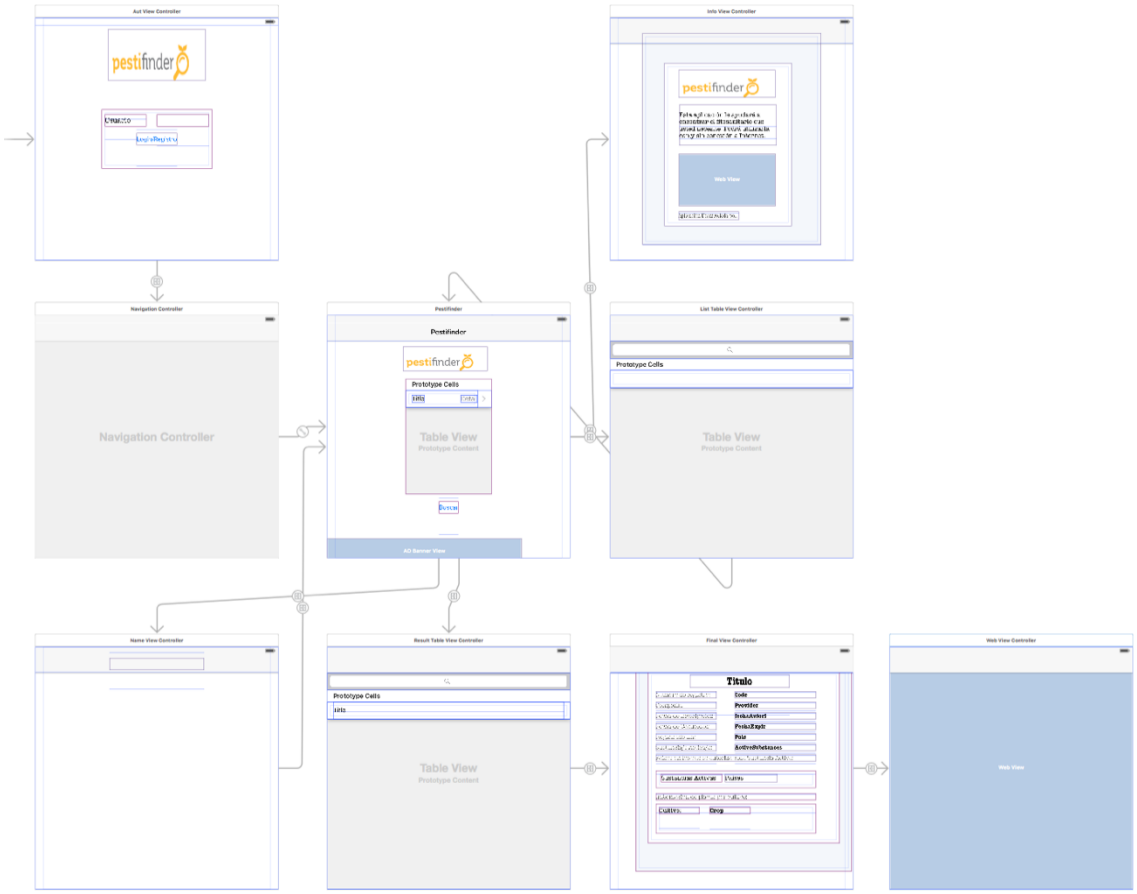

Ilustración 7. Diseño de la Interfaz en Xcode

En esta imagen se pueden apreciar los distintos componentes de las diferentes pantallas, así como los “segues” (conexiones que conectan diferentes pantallas entre las que va a haber una transición).

Una vez se accede a la aplicación, la primera pantalla a la que se accede es la que se puede apreciar en la esquina superior izquierda. En esta pantalla se comprueba si hay un usuario y token en el sistema, pasando automáticamente a la siguiente pantalla en caso afirmativo, y pidiéndole los datos al usuario en caso negativo.

Una vez se tiene al usuario en el sistema, se activa el controlador de la navegación, este controlador situará una barra de navegación fija en la parte superior de la pantalla, en la que se situarán los botones de ir a la pantalla anterior (no disponible en todas las pantallas), el titulo de la pantalla y se dejará libre la opción de incluir o sustituir botones en ella. Por ejemplo, en la pantalla del menú, se ha sustituido el botón de ir a la pantalla anterior, ya que no tendría sentido en esta situación, por un botón en el que se puedan limpiar los campos con información. Para implementar la interfaz de las pantallas para insertar los parámetros de búsqueda, excepto en la de insertar el nombre del fitosanitario, se han utilizado los elementos llamados TableView (esta elección se ha realizado tras consultar las guías de estilo y ver que se ajustaba perfectamente a nuestra necesidad). La lista de resultados se mostrará de manera similar, y todas estas pantallas tendrán una barra de buscador en la parte superior para filtrar los resultados por el nombre y sea más fácil de usar para el usuario. Por último otra pantalla a destacar será la pantalla de un producto que se haya seleccionado en la lista de resultados. Esta pantalla se ha diseñado en dos partes, una parte con etiquetas posicionadas de manera estática donde se insertarán los datos obtenidos y otra parte donde las etiquetas se insertarán dinámicamente según se obtengan los resultados. Y desde esta pantalla se podrá acceder a otra en la que se cargará una página con la información íntegra que proporcionan los distintos ministerios sobre cada uno de los fitosanitarios.

Se ha decidido el uso de los elementos TableView, el botón de información de la barra de navegación y el uso de esta barra, ya que en las Guías de Estilo de Apple viene marcado el uso de estos elementos. [3], [4], [5], [6]

Cada una de estas pantallas, serán asociadas a diferentes clases que se encargarán de controlar el comportamiento de dichas vistas (para indicar estas clases se ha utilizado la nomenclatura: “nombre de la vista” + “ViewController.swift”). El comportamiento de estas vistas, en general, será implementar la lógica de la aplicación y rellenar las vistas con los datos obtenidos o generados. Estos datos en un principio solamente se obtenían de internet, ofreciendo un comportamiento análogo al que ofrece la web de Pestifinder, posteriormente y una vez implementada un sistema de réplica de datos, también se obtendrán los datos que almacene el dispositivo. Esto se va a ver de manera más detallada en la siguiente sección de Persistencia de Datos.

4.2.3 Persistencia de Datos

Obtener persistencia en Swift se puede realizar de diferentes maneras. Principalmente Apple ofrece cuatro alternativas:

- Keychain: Ofrece un almacenamiento de usuarios y contraseñas seguros, encriptados. Es especialmente útil para ese ámbito pero no se ha encontrado utilidad en nuestra aplicación, y por ello se ha desechado.

- NSUserDefault: Ofrece una persistencia de datos que se acostumbra a utilizar para indicar preferencias de usuario y pequeños datos que puedan cambiar con frecuencia en nuestra aplicación. Este sistema nos será útil para almacenar el usuario (correo electrónico del sistema), el token de acceso al servidor y posteriormente, cuando haya una réplica de los datos descargados de internet, también se va a utilizar para guardar fechas de la última actualización de estos.

- Ficheros plist: Se utiliza para configurar por defecto una aplicación, por defecto existe un fichero llamado info.plist, que indicará a la aplicación si va a tener permiso para hacer llamadas HTTP o utilizar la geolocalización. Dicho de otra manera, la aplicación no podrá utilizar estas funcionalidades si no se ha indicado en el fichero. Este tipo de ficheros se utilizará para almacenar la url donde se encuentre el servidor al que nos tenemos que dirigir.

- Core Data: Este va a ser el sistema de persistencia de datos más complejo utilizado en el proyecto. Se trata de un framework que ofrece Apple para almacenamiento de objetos en la aplicación. Básicamente su comportamiento es similar al que tiene un ORM (Mapeo Objeto Relacional). Además posee una herramienta para crear cómo será el tipo de objetos, sus atributos y sus relaciones.

Debido a que tiene un comportamiento en la aplicación menos obvio, se va a explicar un poco más en detalle el uso de este último framework. Core Data se utilizará para dar persistencia a los datos como lista de cultivos, de destino… Y las diferentes listas que configurarán nuestros parámetros de búsqueda y que si no fuera por dicho almacenamiento en disco deberían cargarse de internet (con las esperas y molestias que estas conllevan y más si no se dispone de conexión vía WiFi y tenemos una conexión lenta o con poca cobertura de red). Además de estos datos, también se van a replicar los resultados de búsquedas, dado el número de posibilidades que ofrece una búsqueda así como los datos que almacenan (hay búsquedas de más de 5.000 resultados) se ha optado por guardar solamente las 20 últimas búsquedas. Se ha elegido este número debido a que mientras haya una conexión a internet se van a poder realizar todas las búsquedas, pero sin conexión a internet, parece un número razonable puesto que la dupla Cultivo/Plaga que puede manejar un agricultor no acostumbra a ser muy amplia y parece un número razonable para ofrecer fuera de línea. Dicho de otra manera, un agricultor no acostumbra a tener demasiados cultivos diferentes y sus búsquedas serán habitualmente parecidas. A continuación se muestra un esquema de nuestro modelo de datos.

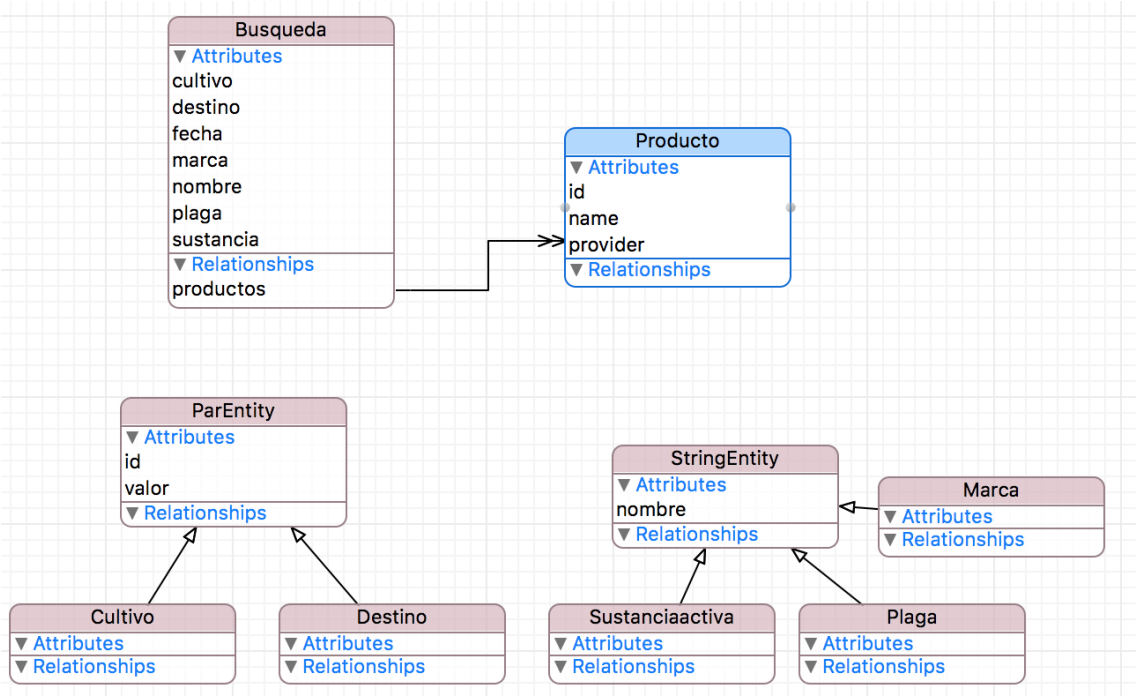

Ilustración 8. Modelado de objetos

En este esquema se puede ver como las listas de Cultivo y Destino, almacenarán duplas id-valor, puesto que los datos que se reciben y envían al servidor de Pestifinder se trabajan de esta manera, mientras que Marca, Sustancia Activa y Plaga solamente tienen el campo nombre. Asimismo, otro elemento relevante es la manera en la que se almacenan los objetos búsqueda. Una búsqueda estará compuesta por los parámetros de búsqueda habituales, la fecha en la que se hizo dicha búsqueda y por ultimo un número indefinido de productos. La fecha será utilizada para, una vez las 20 búsquedas estén almacenadas, si se va a introducir una nueva, poder borrar la primera búsqueda que se hizo.

El motivo por el que se ha utilizado Core Data en vez de una base de datos relacional como SQLite, ha sido debido a que no hay un gran número de relaciones entre los datos y que tras haber probado ambas implementaciones, la escritura en SQLite se apreció bastante más lenta para los mismos datos. Además la manera de trabajar SQLite en Swift es más tediosa que con Core Data. La supuesta desventaja de Core Data frente a SQLite sería una carga mayor de Memoria RAM, pero tras varias pruebas con una carga importante de datos (**ver pruebas de Volumen en Anexo III**), no se ha visto ningún uso de memoria fuera de lo normal.

El acceso a los datos se hacen a través de una clase llamada CoreDataController, de esta forma se separa el acceso al gestor de datos haciéndose éste transparente para la capa controlador que requiera de estos datos.

5 Gestión

En esta sección se va a explicar la gestión que se ha llevado para la realización de este proyecto. Esta gestión incluye la planificación previa del trabajo, en la que se ha abordado el problema, dividiéndolo y marcando sus plazos de realización. Posteriormente se verán el coste total de horas que se han invertido en el proyecto y los riesgos que se han previsto.

5.1 Planificación previa del trabajo

A continuación se muestra un diagrama de Gantt para ilustrar la planificación de este proyecto. Este diagrama muestra el inicio, el cierre y los plazos del proyecto.

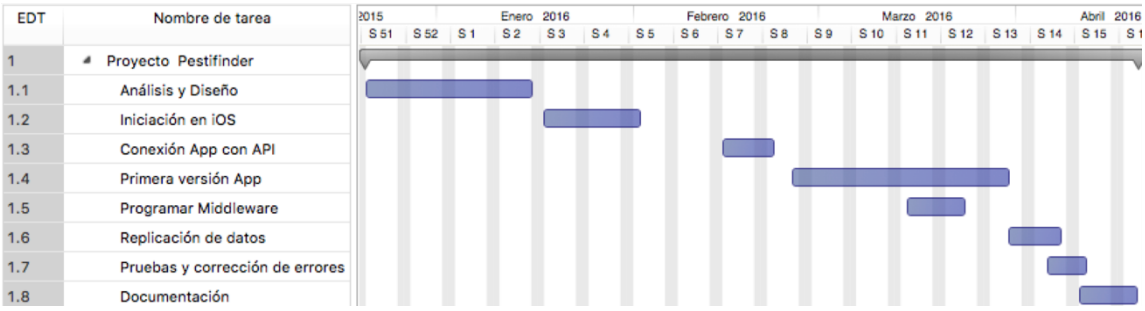

Ilustración 9. Diagrama de Gantt

En el diagrama de Gantt anterior se aprecia como se inicia el proyecto a mediados de diciembre y se finaliza con la redacción de la documentación a mediados de abril. Dicho diagrama muestra como primeramente se realiza un análisis y un diseño, donde se ha analizado el contexto tecnológico donde se encuentra Pestifinder y se ha diseñado el sistema a construir. Posteriormente, dado el desconocimiento del lenguaje de programación necesario, se ha necesitado un tiempo para iniciarse en éste antes de poder comenzar. Tras esta tarea, se ha pasado a la implementación, dicha implementación ha comenzado con la conexión entre la aplicación y Pestifinder para la obtención de los datos y su preparación. La implementación ha seguido con la interfaz gráfica, así como su comportamiento, dando una primera versión de la aplicación, y a la par que se finalizaba esta tarea, se ha realizado la implementación del middleware, acabando la implementación de este, antes de finalizar la primera versión de la aplicación.

Una vez se ha tenido la versión inicial de la aplicación, en la que se tiene un comportamiento funcional, pero solamente con conexión a internet, se ha procedido a implementar la replicación de datos en la aplicación. Tras este paso, se ha tenido una aplicación totalmente funcional, hecho esto, se han realizado una serie de pruebas de la aplicación y se han corregido errores de la aplicación, ya que a pesar de que durante la implementación se han depurado errores, otros errores habían quedado sin descubrir.

Acabada la aplicación, el proyecto finaliza con la entrega de esta memoria. Siendo la duración total del proyecto de 4 meses, aunque durante los dos primeros se avanzara a una velocidad menor debido a que no fue hasta mediados de febrero cuando se tuvo total disponibilidad para el proyecto.

5.2 Coste total de horas

Para contabilizar las horas invertidas en el proyecto, se han utilizado hojas de esfuerzo en las que se introducía el tiempo invertido cada día en un número determinado de tareas concretas, ya que es un método fiable para saber el número de horas que ha llevado la realización de un proyecto y su desglose de tareas.

El resultado de horas ha sido 322 horas (a fecha del 18 de abril de 2016), con el siguiente desglose:

Tarea	*Horas*
Análisis y Diseño	25
Iniciación en iOS	30
Conexión App - API	30
Primera versión App	110
Programación del Middleware	40
Replicación de datos	35
Pruebas y corrección de errores	17
Documentación	35

Tabla 3. Reparto de Horas

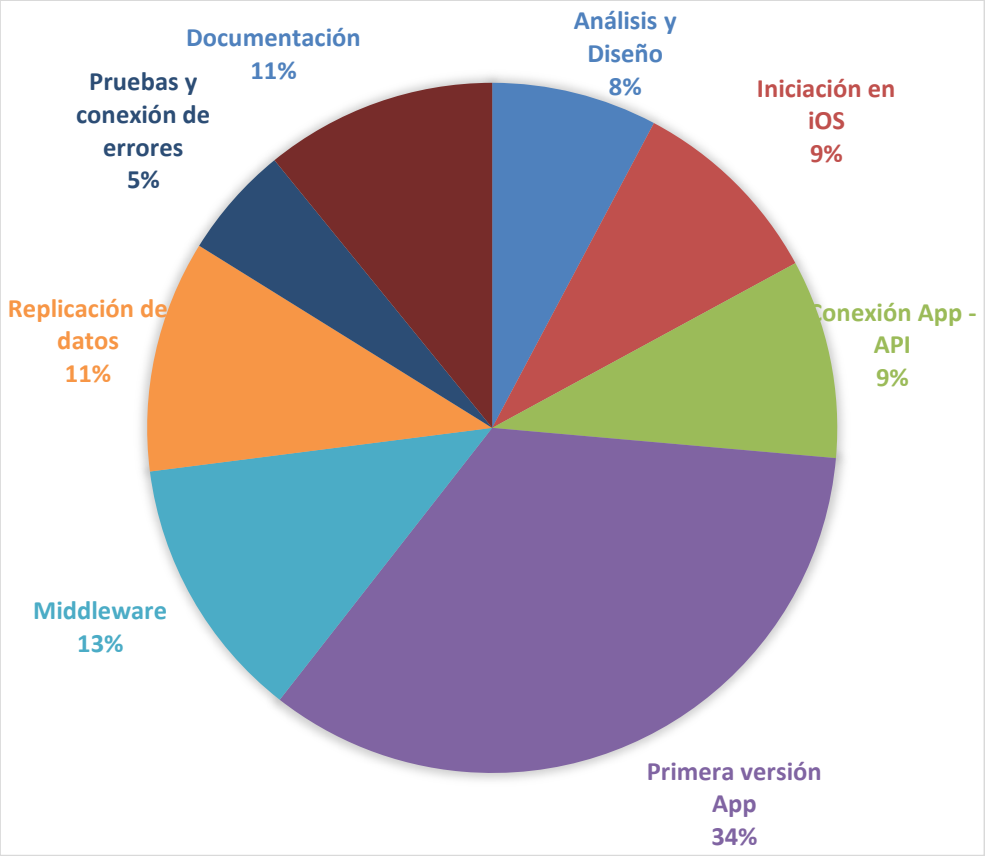

Ilustración 10. Reparto de Horas

5.3 Gestión de Riesgos

A continuación se presentan los diferentes riesgos presentes al inicio del proyecto, así como sus medidas para mitigarlos.

RIESGO	MITIGACIÓN
DESCONOCIMIENTO DE FI-WARE Y SU ACCESO	Incluir un análisis previo de FI-Ware
DESCONOCIMIENTO DEL LENGUAJE DE PROGRAMACIÓN SWIFT	Incluir una fase de aprendizaje y prever un posible desfase en tiempos esperados
PÉRDIDA DE DATOS (CÓDIGO Y MEMORIA)	Hacer copias de seguridad de manera frecuente. En diferentes sitios. Por ejemplo: Dropbox y USB.

Tabla 4. Gestión de Riesgos

6 Mejoras

En esta sección se exponen algunas posibles mejoras que se podrían llegar a realizar para tener un sistema más completo, que ofrezca un mejor servicio y que se pueda comercializar de una manera más completa

6.1 Mejorar Middleware

Una de las principales ideas para mejorar esta aplicación pasa por mejorar el middleware. Que el servidor intermediario entre la aplicación y Pestifinder no ejerza solamente como un intermediario de información, sino que almacene y procese los datos para ofrecer unos servicios más completos. Para esta mejora se propone almacenar diferentes datos como la posición, cultivos y búsquedas de los usuarios de la aplicación. De esta forma se podrían procesar los datos de manera que se tendrían los cultivos y plagas que están ocurriendo en una zona, y se les podría facilitar la búsqueda (mandarla a la aplicación sin que el usuario la pida, para tenerla precargada), así como enviar publicidad orientada a posibles necesidades actuales, o futuras, del usuario.

Otro de los cambios que se le podría hacer a este servidor, sería el cambio de ubicación, tanto el servidor que contiene los servicios web como la base de datos. Dado que aunque Amazon Web Services ofrece muchísimas ventajas, su coste sería elevado en el momento que aumenten los clientes de la aplicación y su tráfico de datos.

6.2 Publicidad pre-cargada

Otra mejora que sería adecuada para un futuro más próximo sería tener la publicidad, que se vaya a mostrar en la aplicación, precargada en esta. Ya que se evitarían esperas innecesarias, actualmente a la vez que se están cargando algunos datos, se está realizando la carga de la publicidad, por tanto se hace todavía más lenta la llegada de estos. Este inconveniente con una buena conexión a internet, pero hay que recordar que no todos los usuarios contarán con una buena conexión a internet vía WiFi, o aun teniendo contratada una buena conexión a internet, por medio de redes 3G o 4G, puede darse la situación en la que no se tenga una buena cobertura de dicha red.

7 Lecciones aprendidas y Conclusiones

El objetivo principal de la aplicación es ofrecer un servicio de búsqueda en dispositivos iOS y análogo al que se ofrecía vía web en Pestifinder, por tanto se ha tenido que realizar un análisis de esta herramienta y aplicar lo que sería ingeniería inversa. En este análisis se ha tenido que comprender el funcionamiento de FI-Ware de cara a la comunicación entre el cliente y este para acceder a un servicio. Además de éste, se ha tenido que analizar una API de servicios web sobre los que no se podía ejercer ningún cambio, aprendiendo así a aprovechar los datos de la manera en la que se nos proporcionen, sin tener posibilidad de realizar cambios sobre la fuente.

Uno de los objetivos que tenía personalmente era lidiar con una tecnología con la que no hubiera trabajado, realizar "algo" que tuviera un funcionamiento vía web y, además de esto, me llamaba la idea de aprender a programar aplicaciones para dispositivos móviles. Por tanto este trabajo me ha permitido cubrir los objetivos que tenía marcados antes de decantarme en un proyecto. He desarrollado una aplicación móvil con búsqueda en línea y fuera de línea y he aprendido un lenguaje de programación nuevo, así como un sistema operativo para el que no había programado. Aparte de haber aprendido sobre un nuevo sistema y lenguaje, también he aprendido a manejar una plataforma tan útil o completa como es Amazon WS.

Finalmente, como última lección aprendida, cabe resaltar la realización de tareas que (aunque ya se habían practicado durante la carrera) las considero muy importantes como han sido el análisis, diseño y las pruebas realizadas. Así como la búsqueda de medios para rentabilizar la aplicación y dejar la aplicación abierta a un crecimiento e integración de posibles mejoras.

Por último, decir que se ha realizado una primera versión viable de la aplicación. Por lo que ésta ya podría subirse a la App Store. Al ser una primera versión, se puede mejorar, pero dadas las dimensiones del proyecto, las horas invertidas y el tiempo que podrían llevar las mejoras, se ha decidido presentar el proyecto en este punto.

8 Bibliografía

[1] C. Zibreg, “iOS 9 adoption jumps to 79 percent of devices.” http://www.idownloadblog.com/2016/03/14/79-percent-ios-9-adoption/

[2] FI-Ware, “Oauth2-Based authorization and authentication in Cosmos WebHDFS at FIWARE LAB” https://www.fiware.org/tag/oauth/

[3] Apple, “Content Views, UIElements,” in *iOS Human Interface Guidelines*, pp. 321–322.

[4] Apple, “iOS Human Interface Guidelines.” https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/MobileHIG/ContentViews.html#//apple_ref/doc/uid/TP40006556-CH13-SW1.

[5] Apple, “iOS Human Interface Guidelines.” https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/MobileHIG/Bars.html#//apple_ref/doc/uid/TP40006556-CH12-SW1.

[6] Apple, “Bars, UIElements,” in *iOS Human Interface Guidelines*, 2016, pp. 265–266, 278.

[7] Apple, “Swift.” https://developer.apple.com/swift/

[8] M. Rubio Diaz, “Desarrollo iOS Core Data y Modelado de Datos.” [Online]. Available: http://www.migueldiazrubio.com/2013/09/04/desarrollo-ios-core-data-i-introduccion-y-modelado-de-datos/ .

[9] PHP, “¿Qué es PHP?” [Online]. Available: http://php.net/manual/es/intro-whatis.php .

[10] Oracle, “MySQL, The World’s Most Popular Open Source Database.” [Online]. Available: http://www.oracle.com/us/products/mysql/overview/index.html.

Anexo I. Tecnologías utilizadas

En este anexo se van a explicar las diferentes tecnologías que se han utilizado en el proyecto, y no se les ha dedicado un suficiente espacio en la memoria (por ejemplo, FI-Ware y Amazon WS).

Swift

Swift se trata de un lenguaje multiparadigma creado por Apple, que en 2015 pasó a ser de código abierto. Se trata de un lenguaje que permite diseñar y crear apps (aplicaciones) para iOS, Mac, Apple TV y Apple Watch.

Según datos ofrecidos por Apple [7] se trata de un lenguaje de programación que puede obtener un mayor rendimiento que otros lenguajes como Objective-C o Python.

Core Data

Se trata de un framework que ofrece Apple para la persistencia de datos en iOS. De entre las diferentes formas ofrecidas por Apple de asegurar dicha persistencia, se trata de la más adecuada para trabajar con volúmenes relativamente grandes de datos, establecer relaciones entre objetos y filtrar la información.

Para el uso de este framework, se deben tener en cuenta tres clases principalmente [8]:

- NSManagedObjectModel: Se trata del modelo de la aplicación. Se representa con un archivo con extensión .xcdatamodelcd.
- MSManagedObjectContext: Se trata del contexto en el que se realizarán las operaciones de lectura/escritura en el modelo.
- NSPersistentStoreCoordinator: Es el encargado de hacer la persistencia real a disco.

PHP

PHP (acrónimo recursivo de PHP: Hypertext Preprocessor) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML. [9]

La particularidad de PHP es que se trata de un lenguaje que se ejecuta en el lado del servidor y no dispone de estado.

Se han importado librerías como Slim para realizar los Servicios Web en este lenguaje, ya que facilita la implementación de estos.

MySQL

Se trata de un Sistema Gestor de Base de Datos relacional. Está considerada la base de datos Open Source más popular del mundo[10], aunque en un principio fue desarrollada por MySQL AB, en la actualidad pertenece a Oracle y se encuentra bajo licencia dual GPL/Licencia comercial.

Se han elegido tanto MySQL como PHP por su gran facilidad para la implementación.

Anexo II. Manual de usuario

En este anexo se va a explicar el comportamiento de la aplicación de cara al usuario, en este apartado se verán los diferentes procedimientos para realizar una acción determinada.

Una vez se accede a la aplicación por primera vez, se accederá a la pantalla de registro, en dicha pantalla se deberá introducir el correo electrónico del usuario.

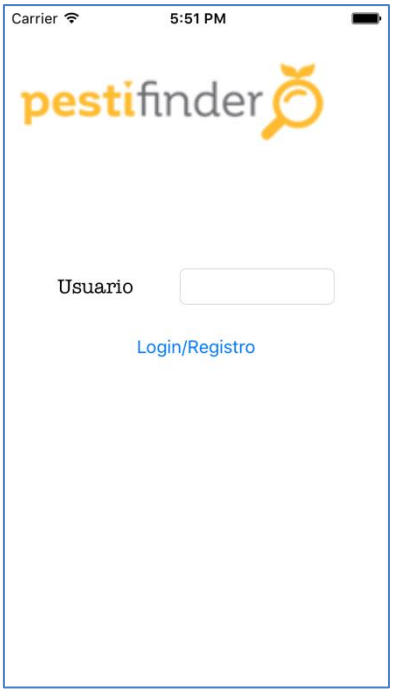

Ilustración 11. Pantalla de entrada inicial a la aplicación

Una vez introducido el nombre de usuario no se deberá volver a introducir a no ser que se reinstale la aplicación o se borren sus datos, y se pasará a la pantalla del menú, que a partir de ahora será la pantalla principal.

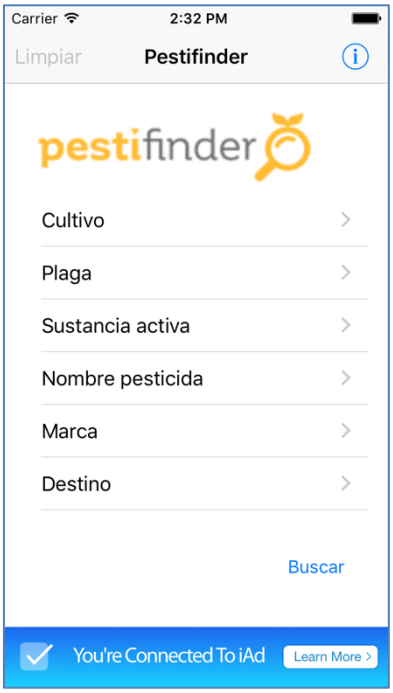

Ilustración 12. Menú principal de la aplicación

En esta pantalla se pueden ver los diferentes parámetros de búsqueda a los que se les puede dar valor. Para darles valor solamente habrá que seleccionar uno de ellos, yendo a la siguiente pantalla. En este caso se representa la selección de un cultivo.

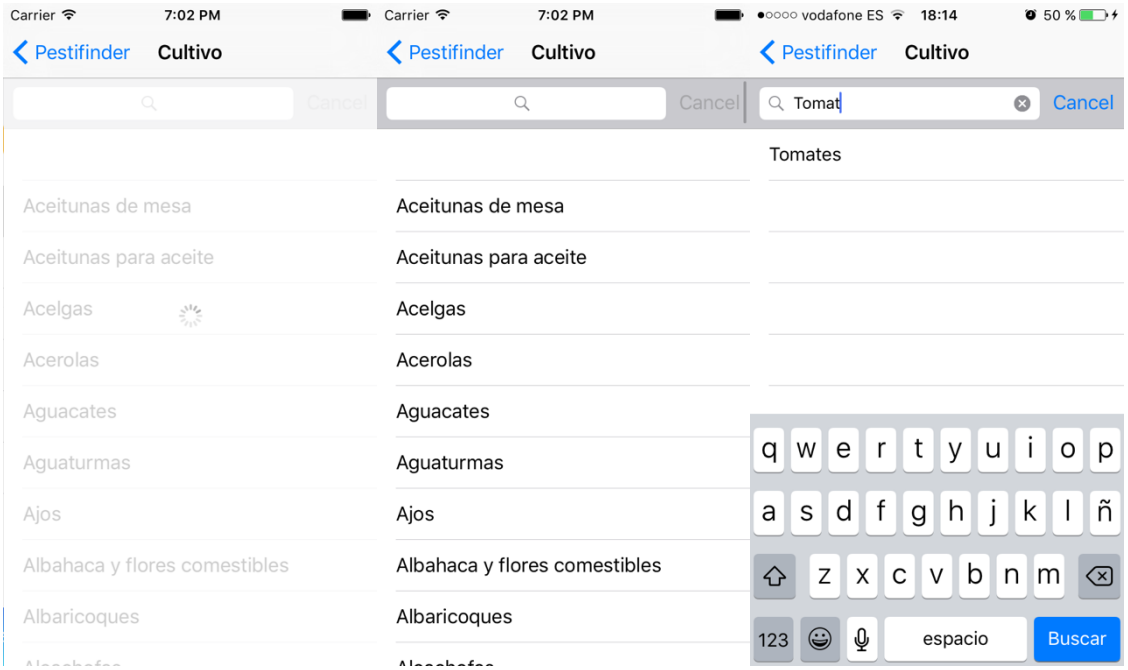

Ilustración 13. Comportamiento de la pantalla de listar cultivos

En estas pantallas se puede apreciar que hay que esperar a que lleguen los datos a la aplicación, y una vez han llegado se pueden buscar desplazándose por la pantalla o directamente buscar en la barra de búsqueda el cultivo deseado. En caso de que hubiéramos seleccionado un parámetro que ya tuviera un valor, tendríamos la posibilidad de cambiarlo o borrarlo seleccionando el primer elemento que aparece sin texto. Estas pantallas tienen un uso idéntico para seleccionar el resto de valores a los que se accede desde el menú principal, excepto para seleccionar el nombre del fitosanitario a buscar, que en ese caso, solamente se deberá rellenar un campo de texto.

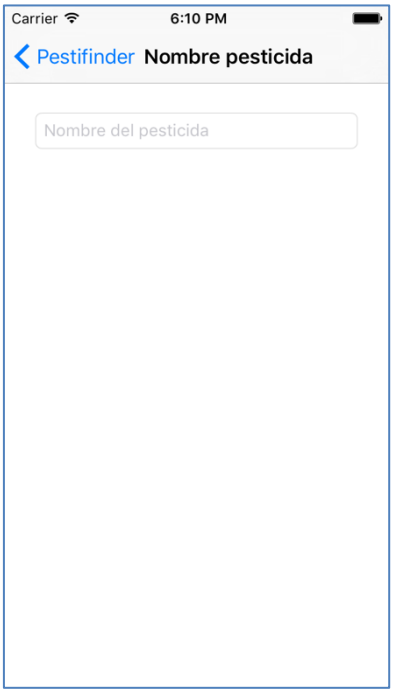

Ilustración 14. Pantalla de introducir nombre de búsqueda de fitosanitario

Una vez seleccionados los parámetros deseados, se podrán ver sus valores junto al título de dichos parámetros en la pantalla del menú principal. En este mismo escenario, también se verá activado un botón llamado “Limpiar” que pondrá todos los valores a su nivel por defecto. Y por último veremos en la parte inferior a la derecha, el botón para realizar la búsqueda.

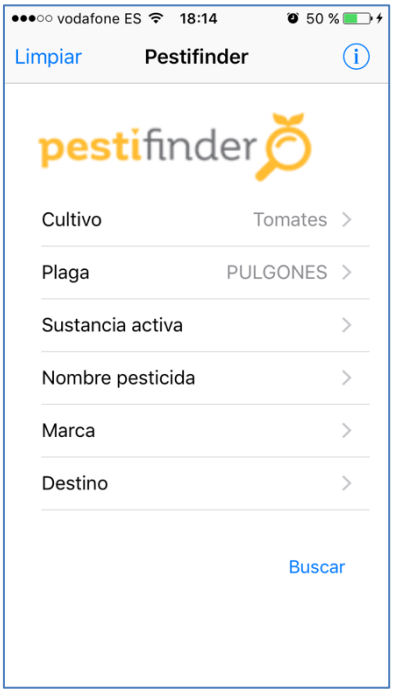

Ilustración 15. Menú principal con opciones marcadas

Una vez se haya realizado la búsqueda aparecerá un listado, parecido al de la selección de valores para los parámetros pero con los nombres y marcas de los fitosanitarios que se han encontrado como resultado. En caso de haber realizado la búsqueda sin haber dado valor a ningún parámetro, se listarán por defecto todos los fitosanitarios.

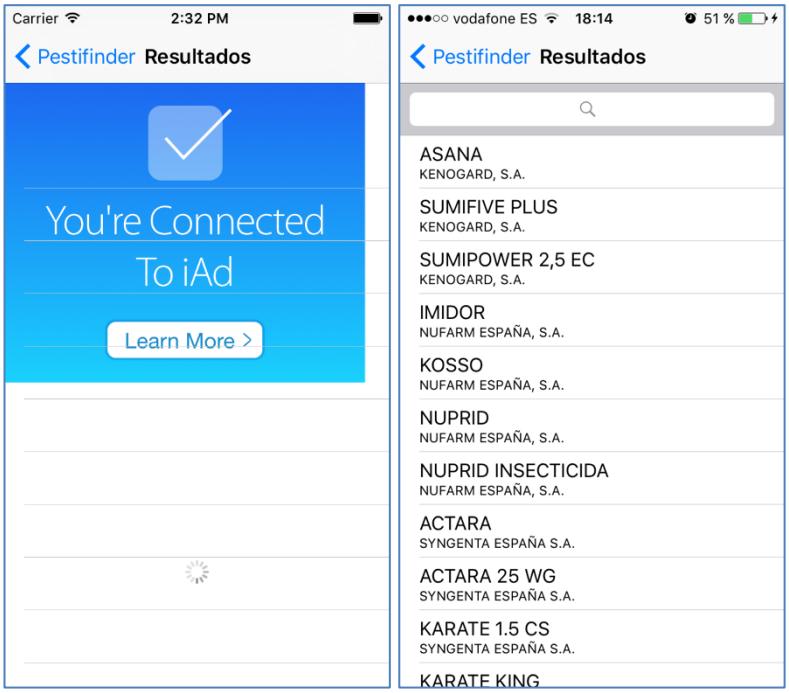

Ilustración 16. comportamiento de la búsqueda de fitosanitarios

Una vez listados los fitosanitarios, se podrá volver atrás para realiza otra búsqueda, o simplemente acceder al menú principal, o se podrá seleccionar uno de estos productos para obtener más información sobre estos. En la pantalla de descripción del producto se verá información de este, así como los distintos países en los que está permitido y las diferentes cultivos y plagas para los que se puede utilizar con sus dosis permitidas.

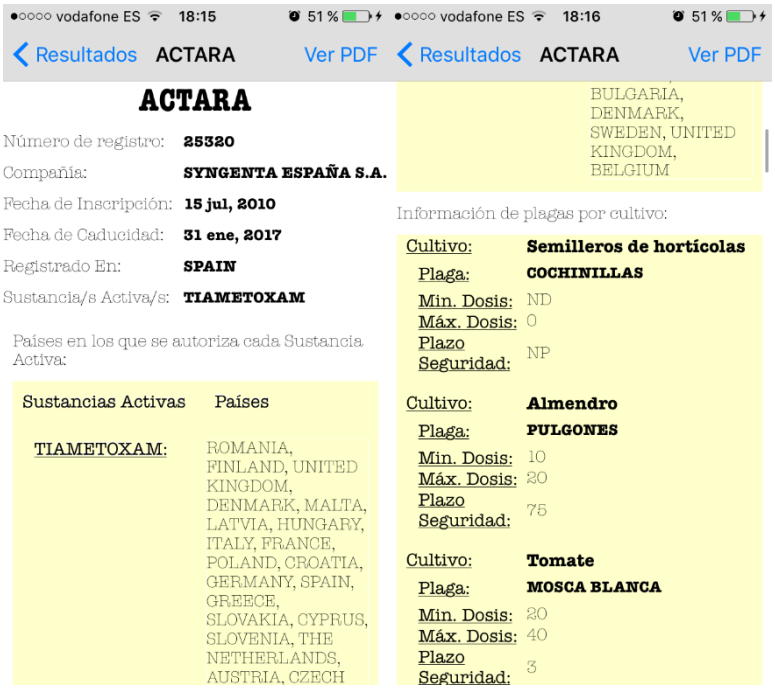

Ilustración 17. Información de un fitosanitario

Una vez consultada esta información, tenemos la opción de volver atrás y consultar otros fitosanitarios, o pulsar el botón de la esquina superior derecho para ver el documento oficial de donde se ha obtenido esta información y en la que se puede obtener información más específica del producto (En el caso de productos registrados en España, el documento se trata de un PDF del MAGRAMA, Ministerio de Agricultura y Medio Ambiente.

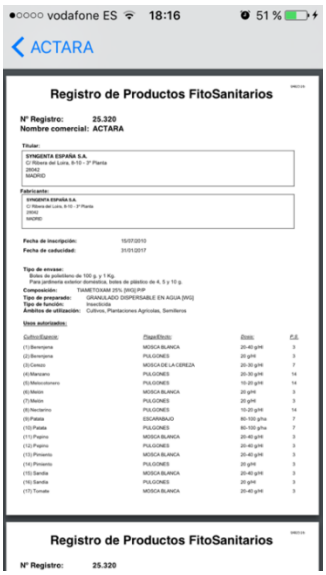

Ilustración 18**. Información precisa de un fitosanitario**

Anexo III. Pruebas

Durante la implementación se han corregido diversos errores que pudieran ocurrir producidos en el desarrollo del código, pero sería un error dejar las pruebas ahí. Una vez finalizada dicha implementación se han realizado diversas pruebas más globales para comprobar el correcto funcionamiento de la aplicación antes de dar por concluido el proyecto.

Pruebas funcionales

Unas de las principales pruebas que se han realizado, han sido las pruebas funcionales, es decir probar que se cumplen los objetivos del sistema y que éste realiza correctamente sus funciones.

Para llevar a cabo estas pruebas se comprobará que se cumplen cada uno de los requisitos funcionales descritos en el Análisis de requisitos (**ver requisitos**).

RF1, "la aplicación debe permitir crear un usuario en base a una cuenta de correo electrónico". Como se puede ver en las imágenes inferiores, se deberá introducir un usuario y este deberá ser un correo electrónico antes de acceder a las funcionalidades de la aplicación. En caso de no introducir un correo electrónico se explicará al usuario por medio de una alerta, que compruebe si el email es válido.

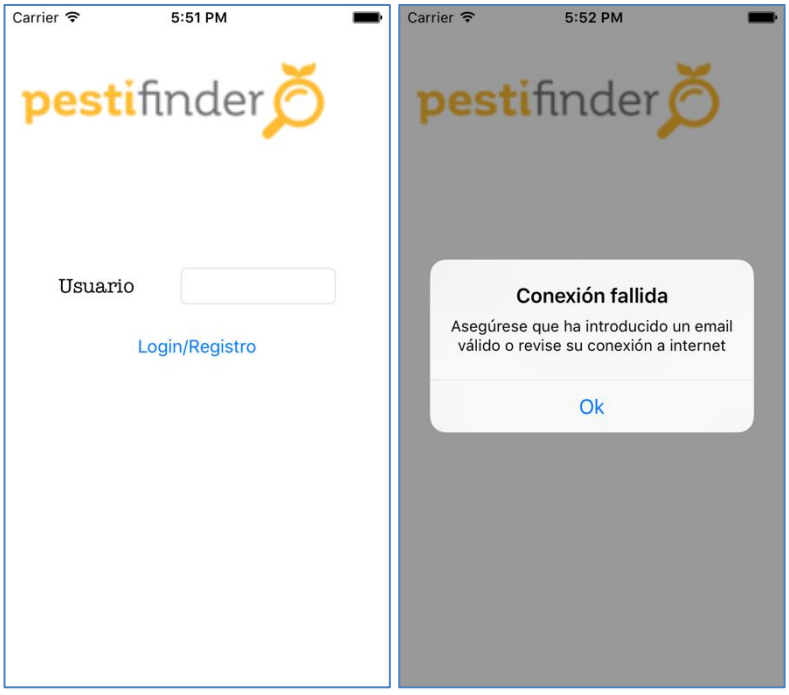

Ilustración 19. email fallido en registro

RF2, "La aplicación debe permitir al usuario dar valor a los parámetros de búsqueda, que serán: Cultivo, Plaga, Nombre del fitosanitario, Sustancia Activa, Marca y Destino".
Como se puede ver en la pantalla del menú, se muestran todos los parámetros de búsqueda y se les ha podido dar un valor a todos ellos.

Ilustración 20. Menú principal completo

RF3, "Los valores de parámetros disponibles que ofrezca la aplicación deberán actualizarse".
Cada vez que se accede a la aplicación mediante conexión WiFi, se actualizan estos valores si estos tienen más de una semana de antigüedad. Por tanto este requisito se habría realizado.

RF4, "La aplicación debe mostrar los fitosanitarios que se adecuen a la búsqueda hecha por el cliente".
Se han comprobado que los resultados que se obtienen de Pestifinder, equivalen a los presentados por la aplicación. Y dado que Pestifinder satisface este requisito, la aplicación también. A continuación se muestra uno de los ejemplos en los que se puede comprobar que la misma consulta produce los mismos resultados en Pestifinder, en sus servicios web y en la aplicación. La consulta se trata de la misma que se muestra en el RF2.

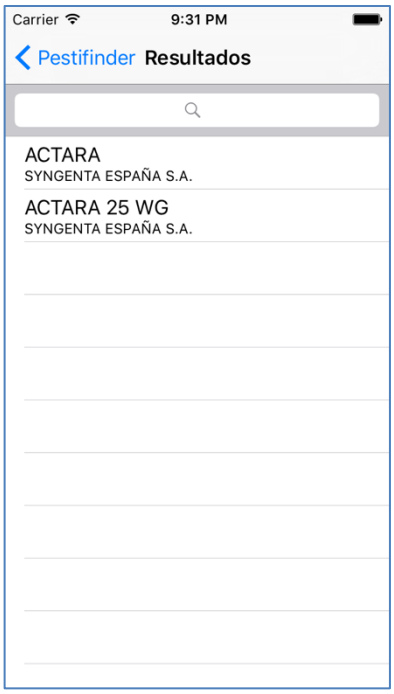

Ilustración 21. Resultados en la aplicación

Ilustración 22. Resultados en la web Pestifinder

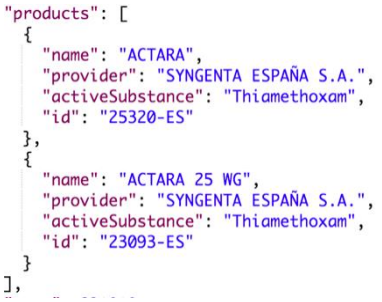

Ilustración 23. Resultados en la API

RF5, "La aplicación debe ofrecer información detallada de un fitosanitario concreto". A continuación se va a mostrar la información ofrecida por la aplicación para un fitosanitario concreto, así como la información más detallada si el usuario requiere el documento origen del fitosanitario.

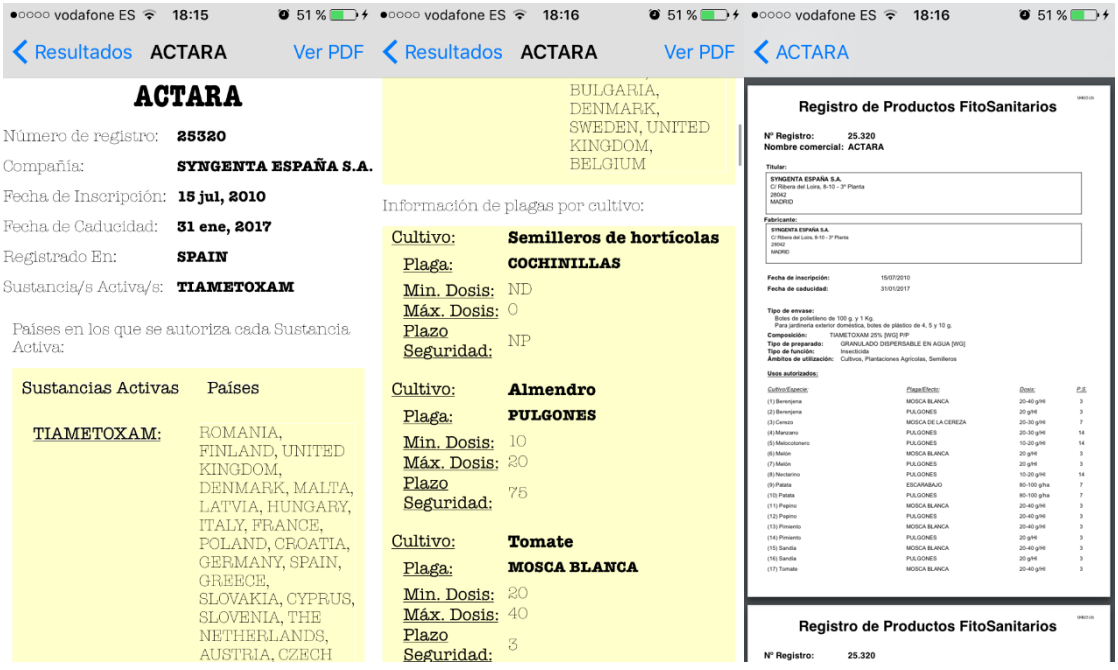

Ilustración 24. Toda la información de un fitosanitario

Como se pueden ver en estas tres pantallas se puede ver como el usuario tiene acceso por medio de la aplicación a una información detallada del producto deseado, por tanto se da por satisfecho este requisito.

RF6, "La aplicación debe ofrecer un servicio con y sin conexión a internet".

Para comprobar este requisito se ha comprobado sobretodo el uso sin conexión a internet, ya que los anteriores se realizaban con conexión a este. Se ha realizado una serie de pruebas para comprobar su correcto funcionamiento fuera de línea, se puede apreciar en el símbolo de "modo avión" en la esquina superior izquierda, en este modo no se obtiene conexión a internet.

- Listar todos los valores que puede tomar un parámetro.

Ilustración 25. Listar sin conexión

- Realizar una búsqueda anteriormente hecha.

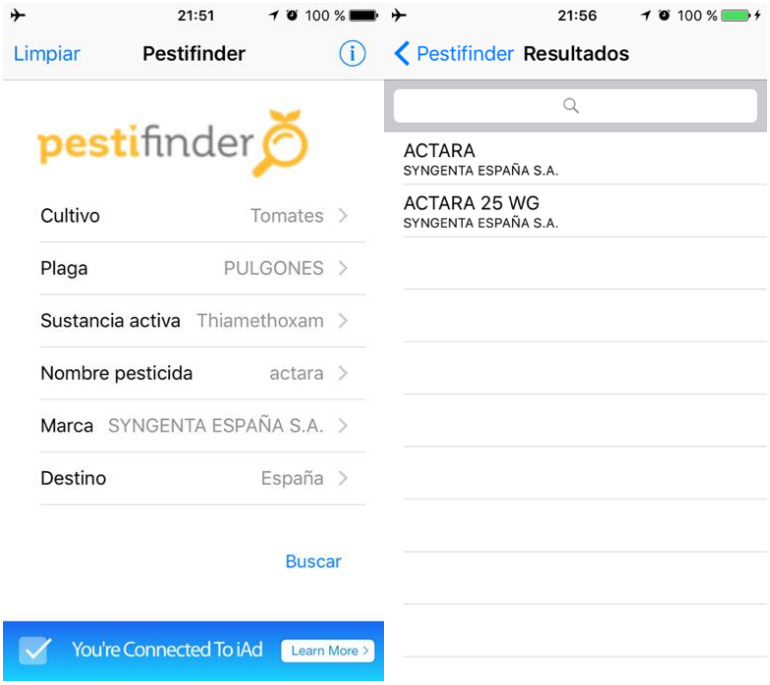

Ilustración 26. Búsqueda sin conexión

- Realizar una búsqueda que no se hubiera hecho (dado que no tenemos datos previos de esta búsqueda, no se tendrán datos).

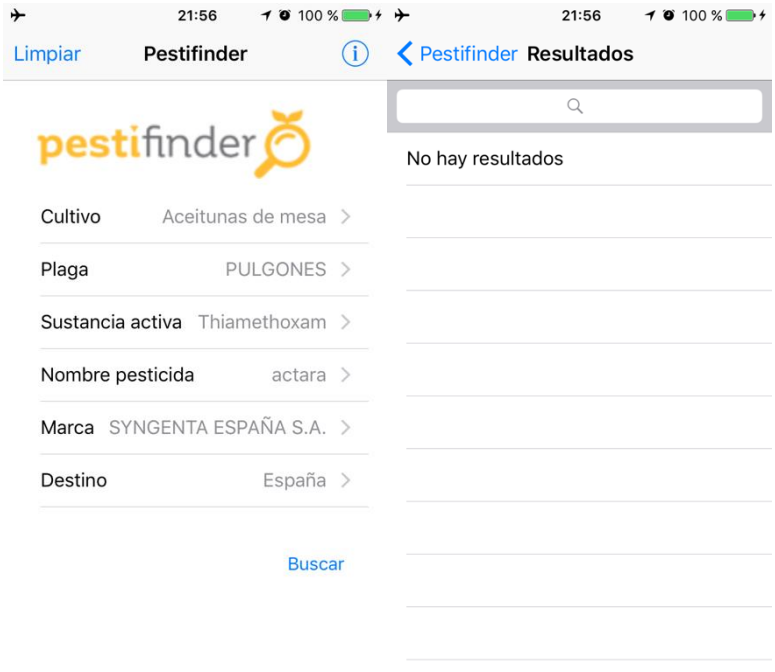

Ilustración 27. Búsqueda sin resultados sin conexión

- Intentar seleccionar un producto para obtener información más detallada de este.

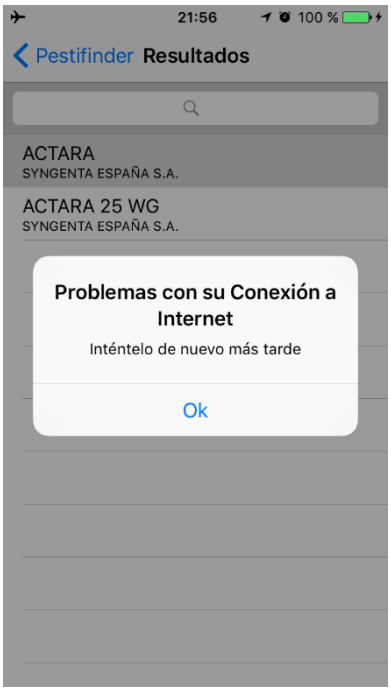

Ilustración 28. Intento de obtener información de un fitosanitario sin conexión

Tras analizar estos datos, se puede observar como en ningún momento sin conexión a internet la aplicación ha tenido un mal comportamiento con cierres inesperados y en todo momento ha cumplido su objetivo o ha mostrado una señal de falta de conexión en caso de no poder acceder a un dato concreto por la falta de conexión.

Pruebas de comunicaciones

Otra de las pruebas del sistema global que se han realizado consiste en comprobar que las interfaces entre los distintos componentes funcionan adecuadamente. Para esta comprobación se han realizado pruebas llamando al middleware con valores esperados e inesperados y comprobando sus respuestas, concluyendo esta prueba encontrando un error a corregir. Dicho error se produjo al enviar un carácter “Ñ” desde la aplicación al middleware como parámetro, lo que hacía que cuando una palabra que contenía dicha letra o signos de puntuación, entre otros, no reaccionaba el sistema de manera adecuada. Este error se solventó codificando estos parámetros con los caracteres permitidos para las URL, y en el servidor/middleware de manera similar mediante la instrucción “rawurlencode”.

Pruebas de volumen

Dado que se ha implementado una replicación de datos y se decidió por Core Data, se han realizado diferentes pruebas con el volumen esperado de datos y ver de qué manera responde el sistema para que no se produzcan futuros problemas. A continuación vamos a ver y describir diferentes mediciones objetivas que se han obtenido para comprobar si el funcionamiento de la aplicación estaba en riesgo.

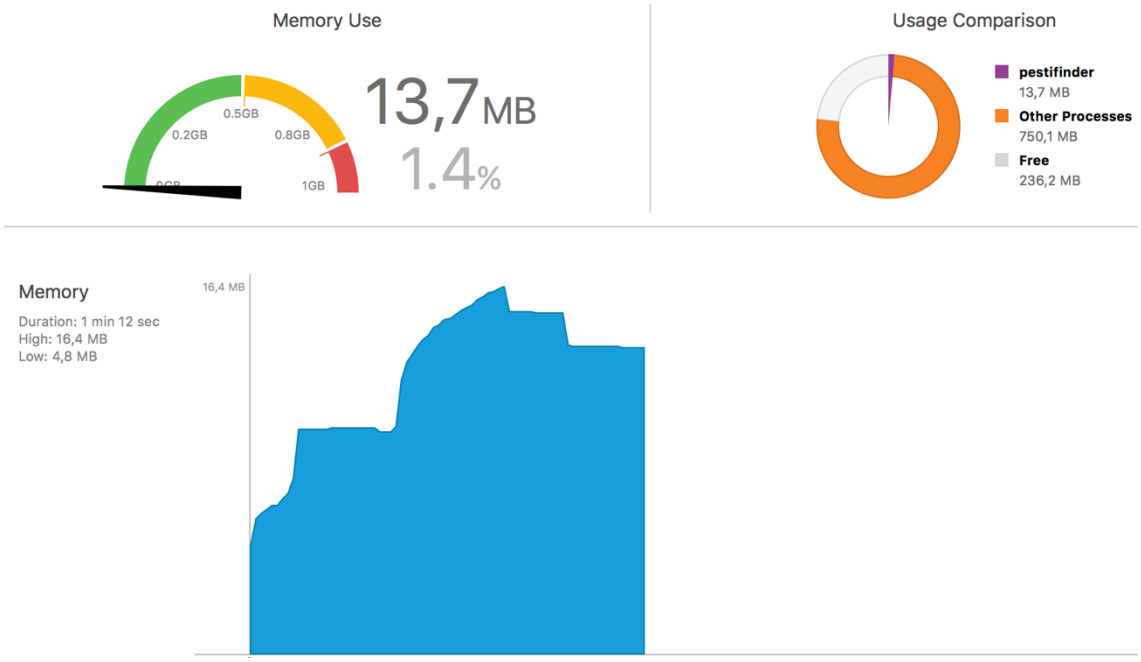

Ilustración 29. Rendimiento de memoria con carga de datos

En esta medición se ha realizado primeramente un registro del usuario en la aplicación, y una primera carga (con réplica de datos) de todas las listas de valores de parámetros, en esta última carga de datos se ve una primera subida de memoria en la RAM. Posteriormente se hace una búsqueda (con réplica de datos) bastante grande, la máxima que se puede realizar actualmente puesto que lista todos los fitosanitarios (más de 5.000), y como se puede ver, aunque se produce un pico en cuanto el uso de memoria RAM, no parece que pueda llegar a preocupar, puesto que los dispositivos con menor capacidad de memoria RAM (iPad 2 y iPhone 4s) en los que puede correr esta aplicación, disponen de 512 MB de capacidad. Llegando esta aplicación a 16,4 MB en su estado máximo. Esta medición tiene bastante sentido, puesto que Core Data primeramente carga todos los datos en memoria RAM y posteriormente los salva en el disco, por lo que puede llegar a sobrecargar esta memoria.

A continuación podemos ver una misma medición de la aplicación realizando una búsqueda sin necesidad de replicación, y no se han encontrado una diferencia demasiado significativo entre el uso corriente de esta aplicación con el uso almacenando datos. Por tanto se puede decir que en cuanto a lo que memoria se refiere, la aplicación responde adecuadamente y ha pasado la prueba del volumen.

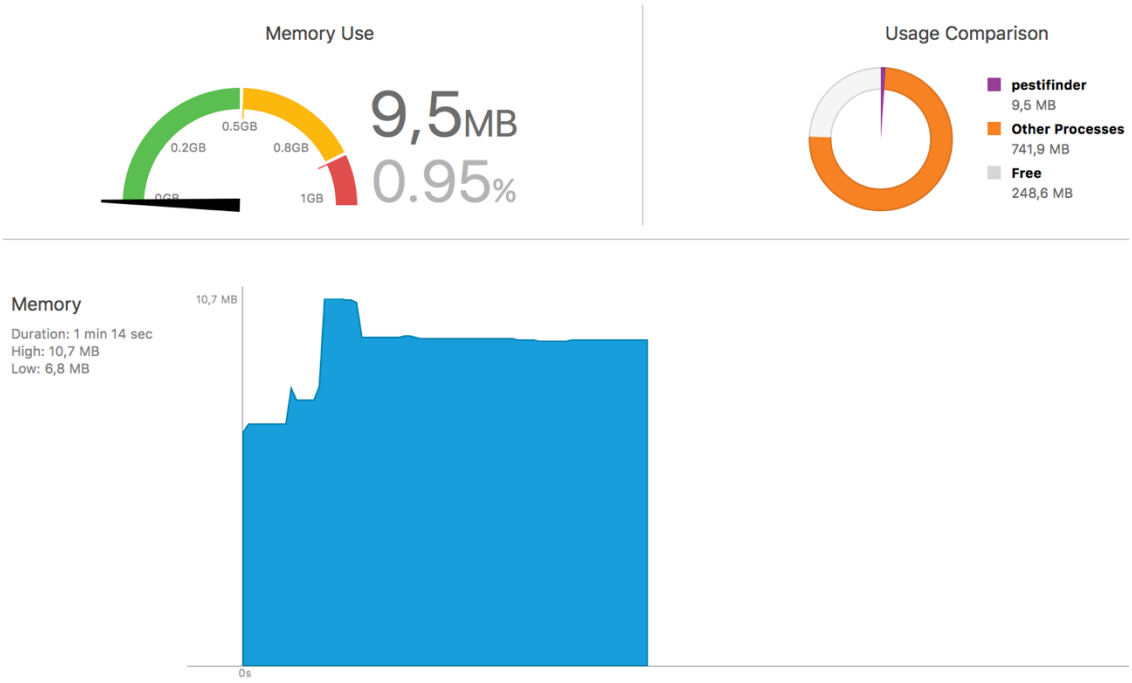

Ilustración 30. Rendimiento de memoria sin carga de datos

Anexo IV. Análisis de la competencia

Tras realizar diversas búsquedas en App Store (tienda de aplicaciones para iOS de Apple), se han encontrado algunas aplicaciones a resaltar.

Pesticide Labels

Se trata de una aplicación canadiense que permite una búsqueda de fitosanitarios. El buscador ofrecido por esta aplicación tiene solamente 4 parámetros, que hay que poner manualmente y no dispone de manera de seleccionar como valor de parámetro.

Ventajas:

- Permite guardar una búsqueda (aunque luego no se puede consultar sin conexión a internet)
- Permite marcar un fitosanitario como favorito, el cual si que se podrá consultar sin conexión a internet.

Desventajas:

- Solamente ofrece servicio para productos canadienses.
- No permite búsquedas fuera de línea.
- No permite incluir como parámetros de búsqueda aspectos útiles como cultivos y plagas a las que afecta el fitosanitario, sino que todos los filtros tienen que ver directamente con el fitosanitario (nombre, marca, sustancia activa…)
- No ofrece información orientada a su uso, solamente hacia su composición.

A continuación se muestran algunos pantallazos de esta aplicación donde se puede ver como muestra la información, como dispone de los favoritos y como no se puede consultar una búsqueda sin conexión a internet.

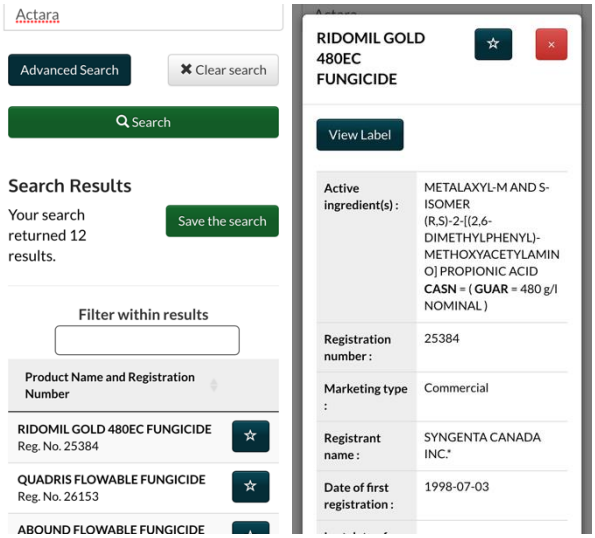

Ilustración 31. Pantallazos de la aplicación Pesticide Labels 1

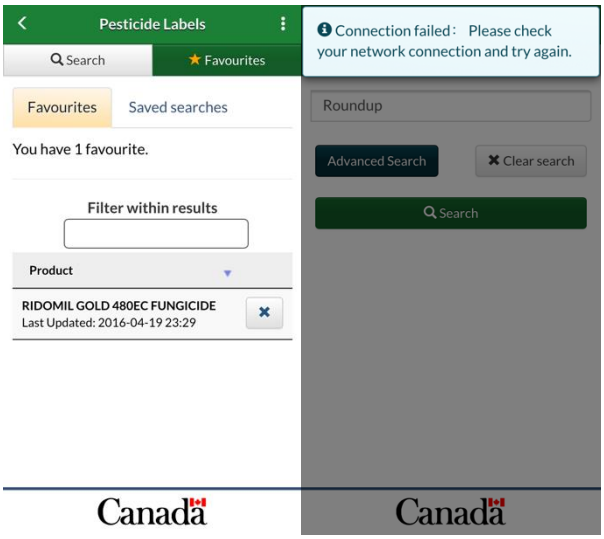

Ilustración 32. Pantallazos de la aplicación Pesticide Labels 2

CultivApp

CultivAPP es una aplicación móvil desarrollada por Inventia Agrárica SL, diseñada para ayudar a los profesionales de la agricultura a gestionar con eficacia sus explotaciones agrícolas.
Esta aplicación ofrece diversos servicios (chats, diarios de actividades…) que la aplicación que se va a desarrollar no se ha considerado incluir, así que el análisis se centrará en la búsqueda de fitosanitarios que también ofrece la aplicación CultivApp.
Según aseguran en la descripción de este producto, a través de esta aplicación se puede acceder a una base de datos, fiable y completamente actualizada, sobre los productos fitosanitarios, abonos y fertilizantes autorizados, pudiendo consultar dosis recomendadas, plazos de seguridad, recomendaciones de aplicación, etc. Además se podrán registrar las aplicaciones de los fitosanitarios a los cultivos[2].

No se ha podido comprobar el funcionamiento de esta aplicación puesto que a fecha de 18 de abril de 2016, se encuentra en "Servicio temporalmente No Disponible", así que el análisis no ha podido ser completo.

Primeramente esta aplicación pide un registro pidiendo diversos datos al usuario. La siguiente pantalla da diferentes opciones según lo que desee realizar el usuario: Vademecum, Diario/Registros, Registro de Campo y CultiChat. Otro aspecto interesante de esta apliación es que su publicidad se basa en patrocinadores propios que se muestran en la parte inferior de esta pantalla. La opción que analizaremos será Vademecum, que es la que aparentemente se utiliza para buscar fitosanitarios. Solamente permitiría filtrar la búsqueda por Cultivo, Enfermedad, Producto y Principio activo, no cargando los valores del primer y tercer parámetro y mostrando un mensaje de error al intentar seleccionar el resto de filtros.

Ventajas de la aplicación:

- En caso de que funcionara, aparentemente parece ser bastante completa y ofrece servicios que Pestifinder no abarca.

Desventajas de la aplicación:

- No parece ser una aplicación estable.

[2] Descripción de la aplicación en iTunes https://itunes.apple.com/es/app/cultivapp/id714206358?mt=8

- No permite seleccionar el país de destino del cultivo.
- No se trata de una aplicación intuitiva y fácil de utilizar.
- No ofrece servicio fuera de línea.

Se pueden ver algunas pantallas de esta aplicación a continuación:

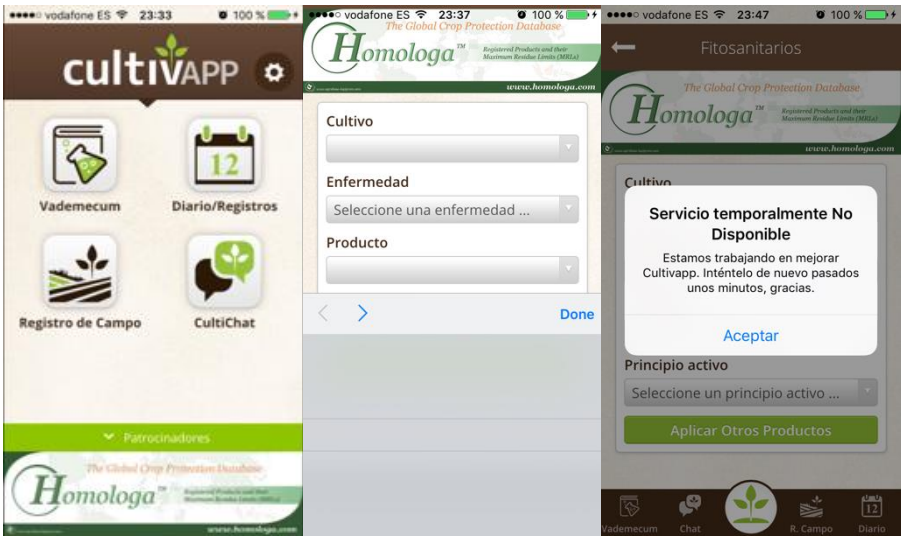

Ilustración 33. Pantallazos de la aplicación CultivApp

Esta podría ser la aplicación que represente la competencia más directa a nuestra aplicación, puesto que está en el mismo idioma, en español, que la nuestra, y trabajo sobre datos españoles. Aunque esta aplicación ofrezca más servicios que la nuestra, en lo que se refiere a buscador que al fin y al cabo es a donde va orientada nuestra aplicación, el de pestifinder es más completo.

Herbicide Guide

Se trata de una aplicación cuyo coste alcanza los 159,99 € al año. Esta aplicación ofrece un acceso a la información sobre una lista de fitosanitarios[3]. Esta aplicación es de procedencia australiana, por lo que probablemente ofrezca solamente fitosanitarios de este país.

Ventajas:

- Parece una aplicación con información bastante completa.
- Ofrece comportamiento sin internet refrescando los datos cada mes.

Desventajas:

- No ofrece versión gratuita y se trata de una aplicación bastante cara.
- No se ha podido probar la aplicación por su elevado precio.
- No parece que se pueda buscar por país de destino.

Aquí se muestran algunas imágenes de esta aplicación:

[3] Descripción de la aplicación en iTunes https://itunes.apple.com/us/app/herbicide-guide/id969359383?mt=8

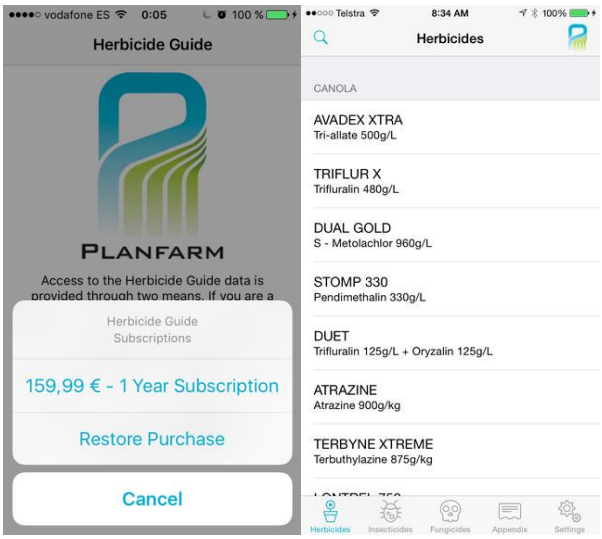

Ilustración 34. Pantallazos de la aplicación Herbicide Guide 1

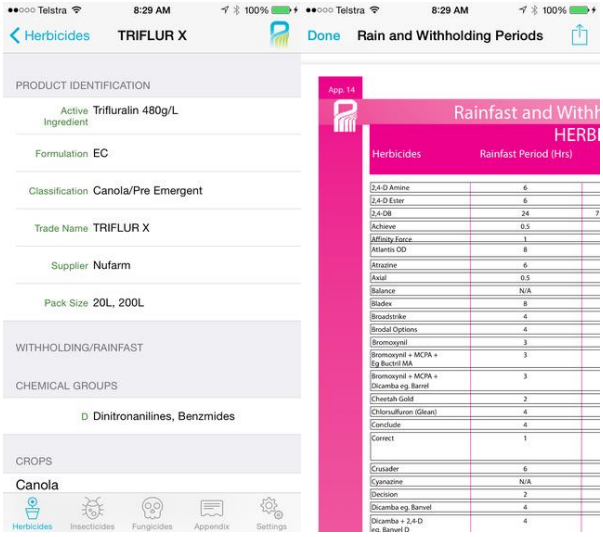

Ilustración 35. Pantallazos de la aplicación Herbicide Guide 2

Índice de Tablas

Índice de Ilustraciones