# UNIVERSIDAD AUTÓNOMA DE MADRID

## ESCUELA POLITÉCNICA SUPERIOR


**Grado en Ingeniería Informática**

# TRABAJO FIN DE GRADO

**Identificación de imágenes en dispositivos móviles Android para tareas interactivas**

**1516_012_SITI**

**Guido Martín Pepa**

**Tutor: Francisco Jurado Monroy**

**Ponente: Rosa María Carro Salas**

**Junio 2016**

# Identificación de imágenes en dispositivos móviles Android para tareas interactivas

**AUTOR: Guido Martín Pepa**
**TUTOR: Francisco Jurado Monroy**

**Escuela Politécnica Superior**
**Universidad Autónoma de Madrid**
**Junio 2016**

# Resumen (castellano)

Uno de los objetivos fundamentales del uso de la tecnología, es facilitar al usuario realizar una tarea determinada.

Con respecto a lo último mencionado, el objetivo de este proyecto es permitir la identificación de imágenes en dispositivos móviles para posibilitar tareas interactivas. En particular, como marco de aplicación se ha desarrollado una *app* que permite realizar el seguimiento de todos aquellos medicamentos que debe tomar una persona, de la forma más fácil e intuitiva mediante el reconocimiento de la caja de dicho medicamento. Este Trabajo de Fin de Grado es uno de los dos componentes funcionales que forman parte del proyecto de investigación MedeTect, desarrollado con ayuda de la Fundación Vodafone España y Fundación de la Universidad Autónoma de Madrid.

El sector de la población al cual va dirigido esta aplicación es en general, todas aquellas personas que por un motivo u otro necesiten ayuda para cumplir con la toma de sus medicamentos. Al tratarse de una aplicación, no se restringe su uso a ninguna persona, ya que el objetivo es ayudar a aquel que lo necesite.

Con el objetivo de que la aplicación sea fácil y cómoda en cuanto a su uso, se ha desarrollado adaptando el diseño y tipo de interacción para que cualquier persona pueda manejarla sin ningún problema ni complicaciones. Para ello, se han seguido las últimas guías de diseño de interfaces para dispositivos móviles contando con principios de accesibilidad, y se ha diseñado un tipo de interacción basada en el tratamiento de imágenes para que el usuario no tenga que manipular complicados códigos o nombres de medicamentos.

# Palabras clave (castellano)

Smartphone, dispositivos móviles, aplicación, Android, Material Design, OpenCV, OCR, Feature Matching, Template Matching, Histogram Comparison

# Abstract (English)

One of the main objectives about using technology is to make any task easier for the user.

In this regard, the aim of this Project is to enable the identification of images using mobile devices as a way to allow interactive tasks. In particular, as application framework, an app has been developed to allow us to track all those medicines that a person should take by recognizing the medicine box in an easy and intuitive way. This Final Degree Project is one of the two functional components that compose the research project MedeTect, developed with support from Vodafone Spain Foundation and Foundation of the Autonomous University of Madrid.

In general, the people this application is targeted are all those who, because one or another reason, needs assistance to comply with taking their medications. Because it is an application, its use is not restricted to any specific person, since the objective is to help everyone who needs it.

In order to make the application easier to use and comfortable, it has been developed by adapting the design and type of interaction so that anyone can handle it without problems or complications. To do this, we have followed the latest guidelines for interface design on mobile devices relying on principles of accessibility, and we have designed a type of interaction based on image processing so that the user does not need to manage complicated codes or names of medicines.



# Keywords (inglés)

Smartphone, mobile devices, app, Android, Material Design, OpenCV, OCR, Feature Matching, Template Matching, Histogram Comparison.

## *Agradecimientos*

A lo largo de esta carrera, me han acompañado una gran cantidad de amigos. Tengo que agradecer a cada uno de ellos la compañía y los momentos de diversión que me aportaron cada día. En especial, quiero agradecer a mis compañeros Jose y Julián con los que mantenía contacto a diario.

Por otro lado, agradecer a mi familia todo el apoyo que me han aportado durante el transcurso de la carrera.

A mi compañero Mario, con el que cooperé para terminar este proyecto.

A mi profesor Alejandro Bellogin Kouki, por la ayuda y dirección en el desarrollo de la aplicación como parte de las prácticas en empresa.

Por último y no menos importante, agradecer a mi tutor Francisco Jurado Monroy toda la ayuda que me ha dado para desarrollar este TFG, además de la oportunidad de poder realizarlo junto a él.

# ÍNDICE DE CONTENIDOS

# ÍNDICE DE FIGURAS

# ÍNDICE DE TABLAS

# 1 Introducción

A lo largo de los últimos años, el campo de la tecnología ha sufrido un gran cambio. Anteriormente, el desarrollo de programas tenía como principal objetivo el que estos fueran empleados en un ordenador. Con los avances tecnológicos actuales, se ha cambiado la concepción de un sistema que ya existía, el móvil, para convertirlo en lo que a día de hoy se conoce como *Smartphone* (Dispositivos móviles inteligentes).

Poco a poco, se ha incrementado en una gran medida la utilización de los *Smartphone*. Tanto es así que, en la actualidad, es muy poco común que un ciudadano no posea esta clase de dispositivo.

Como consecuencia de este cambio, actualmente el desarrollo de programas para dispositivos móviles, conocidos como "aplicaciones" o *apps*, es el principal objetivo de muchas empresas. Esto se debe a que el desarrollo en dispositivos móviles posee numerosas ventajas frente al desarrollo en ordenadores como, por ejemplo, complejidad, difusión y facilidad de uso por parte de los usuarios.

A lo largo de este capítulo se detallará cuál ha sido la motivación para desarrollar el presente Trabajo de Fin de Grado, identificando cuáles son los objetivos funcionales y no funcionales para el mismo.

## *1.1 Motivación*

A lo largo de la vida, las personas pueden verse en la necesidad de tener que seguir algún tipo de tratamiento médico.

El campo de la medicina y la farmacología ha ido evolucionando ofreciendo una gran cantidad de productos y medicamentos que permiten ser aplicados en multitud de tratamientos médicos. Para diferenciar unos medicamentos de otros, cada uno de ellos posee un nombre único. El problema surge cuando, por causas de la edad o por cualquier otro motivo, no es posible la correcta lectura de estos nombres, se poseen numerosos medicamentos, se padece algún grado de déficit de memoria que impide recordar los momentos de las tomas o los nombres de medicamentos, etc.

En este sentido, la principal motivación de este Trabajo de Fin de Grado, es permitir que cualquier persona pueda realizar el seguimiento de sus tratamientos por sí mismos, con la ayuda de su dispositivo móvil. Para ello, implementar una interacción que resulte cómoda al usuario evitando que este deba recordar complicados nombres o códigos resulta fundamental. Así, se propone el uso de técnicas de identificación de imagen para la construcción de la interacción y su integración con el flujo de uso de la aplicación.

Una de las funciones principales de la aplicación, es la de confirmar una toma del usuario. Es decir, el momento en el cual el usuario debe medicarse. Ante esta situación, surgen diversos problemas como, por ejemplo, que el usuario confunda el medicamento que debe tomar en ese momento. Para solventar esta situación, la aplicación avisará al usuario que debe tomar un medicamento determinado y, para confirmar que el usuario ha escogido el medicamento correcto, la aplicación incluye dos métodos que permiten saber si los medicamentos coinciden: OCR (Reconocimiento Óptico de Caracteres) y utilizar una imagen transparente del medicamento a tomar.

## 1.2 Objetivos

Dada la motivación anterior, los objetivos principales que se plantean para este Trabajo de Fin de Grado son los siguientes:

Objetivos teóricos:

- OT1: Ampliar los conocimientos sobre la programación para dispositivos móviles.
- OT2: Aprender a utilizar herramientas que permitan la realización de un trabajo coordinado entre los diferentes integrantes del proyecto.
- OT3: Crear una interfaz para que cualquier usuario pueda interactuar de forma fácil y cómoda con la aplicación.
- OT4: Incorporar diferentes algoritmos para la identificación de imágenes, que serán utilizados tanto para el registro de un medicamento como para la confirmación de una toma.
- OT5: Integración de los anteriores puntos en una aplicación final.

Objetivos específicos:

- OE1: Identificación del sistema operativo idóneo para el desarrollo de la aplicación.
- OE2: Identificación y empleo de herramientas para trabajo y desarrollo colaborativo
- OE3: Identificación de guías para el diseño de interfaces en aplicaciones móviles, teniendo en cuenta criterios de usabilidad y accesibilidad. Con ello, se busca que la aplicación que se desarrolle sea intuitiva y atractiva para el usuario final.
- OE4: Identificación de algoritmos para identificación y tratamiento de imágenes, con el fin de que los procesos de registro y confirmación no deban realizarse mediante la introducción explicita de complicados nombres o códigos por parte del usuario, permitiendo así una interacción más cómoda.
- OE5: Desarrollo de una *app* que permita dar de alta tratamientos médicos e implemente un sistema de notificaciones o alarmas gracias al cual el usuario pueda saber cuándo es el momento en el que debe medicarse y cuál es el medicamento correspondiente, empleando para ello las librerías propias del sistema operativo identificado en el primer punto, las guías de diseño de interfaces analizadas en el segundo y permitiendo la introducción de la información de manera interactiva mediante el tratamiento de imágenes con los algoritmos señalados en el tercer punto.

## *1.3 Organización de la memoria*

La memoria consta de los siguientes capítulos:

**Capítulo 1:** Se muestra el propósito, la motivación y se enumeran los diferentes objetivos del Trabajo de Fin de Grado.

**Capítulo 2:** Se realiza un análisis del estado del arte. En él se introduce y analiza brevemente el tema de la identificación de imágenes, las distintas librerías sobre tratamiento de imágenes investigadas y un pequeño análisis del mercado, en el que se muestran diferentes aplicaciones existentes que ya utilizan la identificación de imágenes.

**Capítulo 3:** Se muestran todas las decisiones tomadas en cuanto al diseño de la aplicación para su posterior desarrollo.

**Capítulo 4:** Se explica cómo se ha conseguido la implementación de la aplicación y de las diferentes funcionalidades que abarca este Trabajo de Fin de Grado. Además, se detallan todas las herramientas utilizadas para desarrollar este proyecto.

**Capítulo 5:** Se analizan todos los algoritmos empleados en la elaboración del proyecto mediante una serie de pruebas. Además, se realiza una comparación para saber cuál de todos ellos es el más adecuado para la posterior integración con la aplicación final.

**Capítulo 6:** Se detallan las conclusiones sacadas con la elaboración de este Trabajo de Fin de Grado y se realiza un análisis sobre las posibilidades futuras de la aplicación.

# 2 Estado del arte

En este capítulo, se analizarán los distintos elementos que componen la parte fundamental de la aplicación que se va a desarrollar. A lo largo del capítulo, se exponen los motivos por los cuales se ha seleccionado el sistema operativo Android, y se analizan los resultados que se han obtenido de la investigación de las distintas herramientas existentes en la actualidad que permiten desarrollar la tecnología que utilizará la aplicación. Por último, se muestra un análisis del mercado actual de aplicaciones móviles, con ejemplos de *apps* que poseen funcionalidades similares al proyecto en el que se va a trabajar.

## *2.1 Identificación de imágenes*

Cuanto una persona observa una foto en la que aparecen personas, es capaz de identificar claramente dónde aparece cada individuo y puede diferenciarlos entre ellos. Desde el punto de vista de un ordenador, cada persona en la foto es considerada como un conjunto más de los pixeles que componen una imagen.

Con el avance de la tecnología, se ha podido conseguir que el ordenador identifique y "reconozca" dichos pixeles como personas.

En esto consiste la identificación y reconocimiento de imágenes. A partir de un conjunto de pixeles que *a priori* no tienen ninguna relación entre ellos, obtener información mucho más detallada sobre lo que se encuentra en dicha imagen. Es decir, reconocer y tratar ciertos elementos que se encuentren en esa imagen.

En la actualidad, esta tecnología es empleada por numerosas empresas con diferentes objetivos. Por ejemplo, el buscador Google [1] permite la búsqueda de información a partir de una imagen, en lugar de utilizar texto. Es decir, reconoce el contenido de una imagen seleccionada por un usuario, y obtiene una serie de resultados que tienen una elevada relación con dicha imagen

El potencial de esta tecnología es increíble y parece no tener límites en cuanto a situaciones en las que poder utilizarla.

En el presente Trabajo de Fin de Grado se aplican técnicas de identificación de imágenes en dispositivos móviles a fin de proporcionar una interacción más cómoda entre el usuario y este tipo de dispositivos.

## *2.2 Análisis de sistemas operativos para dispositivos móviles*

El objetivo del proyecto es ayudar a todas aquellas personas que les pueda resultar difícil seguir el tratamiento de uno o varios de sus medicamentos, así como a personas que, por diversos motivos, se puedan olvidar de cuándo y de qué medicamento deben tomar una dosis. En la actualidad, el uso de dispositivos móviles es habitual e incluso, en algunos casos, una necesidad. Dentro de los dispositivos móviles encontramos numerosos sistemas operativos, pero entre todos ellos, el que predomina en el mercado actualmente, es Android. Los datos mostrados en la Figura 2-1 se han extraído de la página International Data Corporation (IDC) [2].

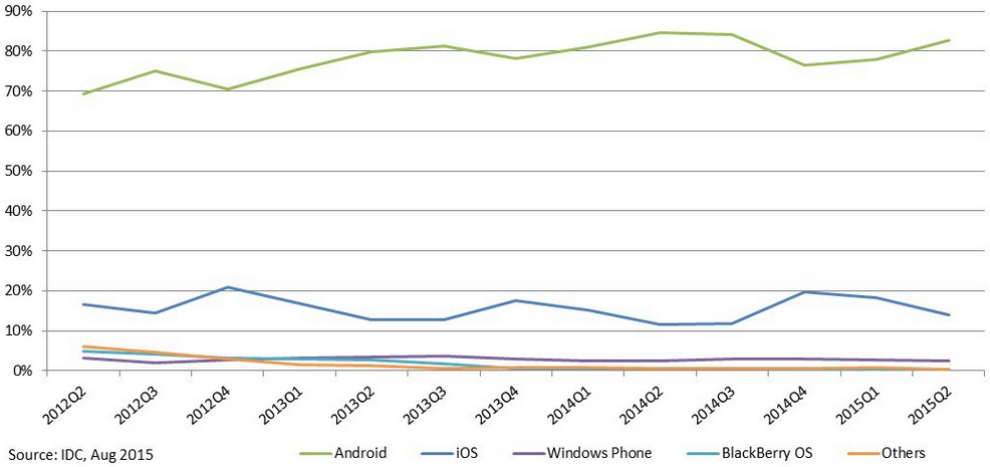



**Figura 2-1: Porcentaje de ventas mundial de SO Smartphone.**

Tras estudiar los diferentes sistemas operativos donde se podría desarrollar la aplicación, se ha llegado a la conclusión de que el más adecuado es Android. El motivo principal por el cual se ha seleccionado esta opción, es porque el objetivo de la aplicación es abarcar al mayor número de público posible. El elevado porcentaje de uso de dicho sistema operativo, asegura cumplir este objetivo. Además, es un sistema operativo que se encuentra en continua evolución, gracias a que el desarrollo del mismo está respaldado por la empresa Google.

Una vez seleccionado el sistema operativo, se deben estudiar las diferentes versiones para comprobar cuál de todas ellas cumple con los requisitos y objetivos que se desean alcanzar con la aplicación.

Como se ha explicado anteriormente, Android es un sistema operativo en continua evolución. Tanto es así, que en el periodo en el que se ha desarrollado la aplicación, se ha publicado una nueva versión (6.0 - Marshmallow). Por tanto, es normal que la mayoría de personas, actualice la versión que posee actualmente su sistema. Esto provoca, que algunas versiones comiencen a quedarse en desuso o a perder una gran cantidad de usuarios. Por ello, la mayoría de dispositivos comenzarán a funcionar bajo estas versiones.

Tal y como se ha mencionado, entre los objetivos del proyecto se busca que en el desarrollo de la aplicación se incluyan elementos que sean tanto intuitivos como atractivos para el usuario final (en cuanto a la interfaz del usuario), aplicando un diseño lo más novedoso posible.

Con la información anteriormente expuesta y estudiando la Figura 2-2 extraída de la página Android Developers [3], se ha decidido desarrollar la aplicación para todos aquellos sistemas Android que posean una versión igual o superior a 4.4 (KitKat).

Dado que la aplicación se implementará con una interfaz de usuario novedosa en cuanto a su diseño, no es posible abarcar más versiones que las mencionadas. Esto se debe a que las funcionalidades que se desarrollarán en la aplicación, no están soportadas por las versiones anteriores.
Por todos los motivos expuestos anteriormente, se considera razonable el hecho de que la aplicación no se encuentre disponible y, por tanto, carezca de soporte en versiones anteriores a las ya mencionadas.

| Version | Codename | API | Distribution |
|---|---|---|---|
| 2.2 | Froyo | 8 | 0.1% |
| 2.3.3 - 2.3.7 | Gingerbread | 10 | 2.0% |
| 4.0.3 - 4.0.4 | Ice Cream Sandwich | 15 | 1.9% |
| 4.1.x | Jelly Bean | 16 | 6.8% |
| 4.2.x | | 17 | 9.4% |
| 4.3 | | 18 | 2.7% |
| 4.4 | KitKat | 19 | 31.6% |
| 5.0 | Lollipop | 21 | 15.4% |
| 5.1 | | 22 | 20.0% |
| 6.0 | Marshmallow | 23 | 10.1% |

*Data collected during a 7-day period ending on June 6, 2016.*
*Any versions with less than 0.1% distribution are not shown.*

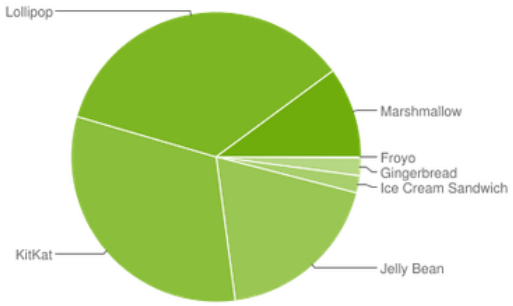



**Figura 2-2: Versiones de Android y sus porcentajes de distribución**

## 2.3 Guías para el diseño de interfaces de usuario para dispositivos móviles

A fin de reducir la carga cognitiva de los usuarios a la hora de manejar una nueva aplicación, así como de mantener una homogeneidad estética y de usabilidad entre las aplicaciones, es imprescindible tener en cuenta las guías de diseño existentes al respecto para cualquier aplicación a implementar.

Dado que el sistema operativo sobre el que implementar la *app* será Android, la principal guía de diseño de que se dispone es *Material Design* [4] desarrollada por Google. Con ello se consigue que, independientemente del dispositivo Android que posea el usuario, podrá hacer uso de la aplicación sin necesidad de un manual para conocer la funcionalidad de cada uno de los elementos de la interfaz, haciendo su uso más intuitivo.

Basándose en la metáfora de materiales tridimensionales táctiles (los elementos de la interfaz aparecen como elementos que pueden tocarse gracias al uso de materiales, luces y sombras), *Material Design* proporciona guías sobre el estilo (colores, iconos, fuentes, etc.), distribución de elementos en la interfaz (*layout*), componentes gráficos a emplear (tipos de botones, elementos de entrada y salida, etc.), patrones de elementos gráficos (notificaciones, pantallas de permisos, tipo de navegación, etc.).

## 2.4 Mercado actual de apps que emplean identificación de imágenes

En la actualidad, existen numerosas herramientas y aplicaciones que utilizan la identificación de imágenes. A continuación, se presentan algunos ejemplos.

Tal vez el más conocido es Google Goggles [5]. Esta aplicación desarrollada por Google, permite la búsqueda de información a partir de una imagen tomada con un dispositivo

móvil. Entre otras funcionalidades, se puede sacar una foto a un cuadro del cual se desea extraer más información, y se muestran en la aplicación todos los resultados referentes a dicho cuadro. De esta forma, se puede ver información sobre el autor de dicho cuadro, año en el que se ha pintado, etc. En la Figura 2-3, se puede observar un ejemplo simple del funcionamiento de esta aplicación.


**Figura 2-3: Ejemplo funcionamiento de la aplicación Google Goggles**

Desde el punto de vista comercial, empresas como Zalando [6], disponen de *apps* que permiten que todos aquellos usuarios que dispongan de un *Smartphone*, tengan la oportunidad de buscar productos de la tienda a partir de una imagen. Es decir, si un cliente encuentra en una revista una prenda de moda que desearía comprar, puede tomar una foto con su dispositivo móvil, y la aplicación le mostrará la información que necesite. En la Figura 2-4, se puede observar el funcionamiento de la aplicación.

Otra aplicación que usa la tecnología de la identificación de imágenes es PlantNet [7]. Esta aplicación permite la identificación automática de plantas a partir de la comparación de una imagen con una base de datos botánica. Es decir, compara la imagen de la planta que toma el usuario con una serie de imágenes almacenadas en una base de datos con el fin de obtener el nombre botánico de una planta. En la Figura 2-5, se puede observar una pantalla que muestra la información relativa a una planta.

En conclusión, todas aquellas herramientas que a través de una imagen puedan obtener información acerca de lo contenido en dicha imagen, están utilizando algoritmos de reconocimiento de imágenes.

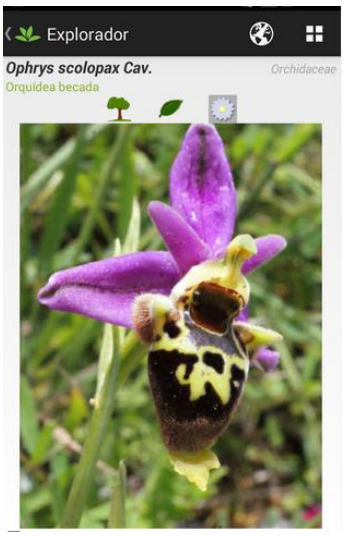



**Figura 2-5: Pantalla de la aplicación PlantNet**


**Figura 2-4: Pantalla de la aplicación Zalando**

## 2.5 Algoritmos de identificación de imágenes

A la hora de identificar una imagen, uno de los procedimientos más comunes es el de comparar la imagen observada con una previamente almacenada, es decir, se realiza la identificación por medio de la comparación de imágenes.

Existen numerosos algoritmos que permiten comparar dos imágenes entre sí. Entre todos estos algoritmos, se deben escoger aquellos que se adapten al proyecto y que aporten unos buenos resultados.

Para el presente trabajo se han identificado 4 algoritmos: *Template Matching* [8], *Feature Matching* [9], *Histogram Comparison* [10] y Optical Character Recognition (OCR) [11].

La técnica más sencilla es la *Histogram Comparison* (Comparación de histogramas) [10]. Esta técnica consiste en agrupar todos los pixeles que posean una misma característica (color, intensidad) en diferentes grupos para poder, posteriormente, hacer la comparación de las imágenes. La principal ventaja, es la facilidad de uso y la rapidez del algoritmo. Por otro lado, una de las principales desventajas es que, aplicando este algoritmo, no se obtienen resultados precisos (ya que los colores son una característica muy variable dentro de una imagen). Es decir, varían dependiendo de los colores que forman la imagen, obteniéndose resultados diferentes si se comparan imágenes iguales, pero con diferentes colores (cuando se debería identificar que ambas imágenes son iguales independientemente de su color).

Tratando de subsanar las restricciones y limitaciones de la comparación de histogramas, se encuentra la técnica del *Template Matching* [8]. Ésta consiste en identificar una imagen dentro de otra desplazando la primera sobre la segunda, y cubriendo todos los pixeles posibles. La principal ventaja es que no se centra en una característica tan variable como lo es el color, sino que se realizan comparaciones a nivel de pixel. La principal desventaja, es que busca similitud entre pixeles, en lugar de realizar una detección del elemento que se encuentra en la imagen. Es decir, si se comparan imágenes completamente diferentes, se obtiene un porcentaje de similitud alto, en el caso de que ambas imágenes posean una serie de pixeles similares.

Como una alternativa mucho más elaborada se encuentra la técnica del Feature Matching [9], que consiste en detectar una serie de características de cada imagen y compararlas entre sí. La principal ventaja es que identifica si un elemento u objeto aparece en ambas imágenes. La desventaja es que se debe configurar una serie de parámetros del algoritmo, con el objetivo de obtener unos resultados más precisos. Además, el tiempo de ejecución y de procesamiento son altos, debido a la gran cantidad de cálculos que se deben realizar (extraer características de las imágenes, comparar cada característica de una imagen con las características de la otra imagen, etc.).

Por último, otra técnica a emplear en la aplicación es el OCR [11]. Esta técnica consiste en identificar, a partir de un conjunto de imágenes de entrenamiento, cualquier carácter dentro de una imagen. Como se explicará más adelante, para este proyecto solo es necesario identificar números. Como ventaja de emplear esta técnica, se debe destacar su elevada precisión en cuanto a resultados obtenidos. Como desventaja, es necesario que la imagen que se desea escanear se encuentre bien iluminada y con una buena nitidez (cuanto más nítida sea la imagen, mejores resultados se obtienen).

## 2.6 Librerías para tratamiento de imágenes

Existen numerosas herramientas en el mercado que permiten desarrollar las funcionalidades de las que requerirá la aplicación. De entre todas ellas, se ha realizado una pequeña selección dependiendo de las características que se buscan para la aplicación a desarrollar. En la Tabla 2-1 se puede observar la comparación de estas herramientas (librerías).

| | **Tesseract** | **Opencv/Javacv** | **Zbar** | **Zxing** | **Asprise OCR** |
|---|---|---|---|---|---|
| **Utilidad** | Lector OCR | Lector OCR y reconocimiento de imágenes | Lector código de barras | Lector códigos QR | Lector OCR y código de barras |
| **Compatibilidad Android** | Si (Android 2.2 o superior) | Si | Si | Si | Si |
| **Multiplataforma (Win.,Linux,Mac)** | Si | Si | Si | Si | Si |
| **Actualizaciones y soporte** | Si | Si | Si | Si | Si |
| **Documentación (Dificultad)** | Si (Difícil) | Si (Media) | Si (Media) | SI (Media) | Si (Media) |
| **Licencia** | Apache 2 | BSD | GNU-LGPL | Apache 2 | Software propietario |
| **Inconvenientes** | La librería requiere de entrenamiento para detectar los caracteres que se desean leer | Pequeña dificultad al instalar la librería | En ocasiones, a los medicamentos se les retira el código de barras, lo que impide su posterior lectura con el dispositivo móvil | Es necesario que el dispositivo móvil tenga previamente instalada una aplicación para leer códigos QR (Y que en el medicamento tenga un código QR) | El software es propietario (de pago) |

**Tabla 2-1: Análisis comparativo de librerías para tratamiento de imágenes**

Se han estudiado 5 librerías diferentes atendiendo a siete características: *Tesseract* [12], *OpenCV* [13], *Zbar* [14], *Zxing* [15] *y Asprise OCR* [16]. Entre todas estas herramientas, se han descartado aquellas que consisten en la lectura de un código de barras. Esto se debe a que, en ocasiones, el personal médico retira de las cajas de los medicamentos el código de barra que los identifica. Por tanto, utilizando la lectura de código de barras, no se daría soporte a la totalidad de los medicamentos existentes en el mercado.

También se han descartado aquellas que utilizan técnicas como lectura de códigos QR. Esto se debe a que, a pesar de que sería sencillo identificar cada medicamento atendiendo a un código QR, habría que obligar a los usuarios a generar un código para cada medicamento y adjuntarlo a su caja correspondiente. Como la aplicación tiene como objetivo facilitar la tarea de seguimiento de toma de medicamentos a los usuarios que la utilicen, se descarta esta técnica ya que requiere de una mayor participación y trabajo por parte del usuario.

Tras el análisis de cada una de estas herramientas, se ha seleccionado Opencv [13] para el tratamiento de imágenes, ya que es la librería que mejor se adapta al proyecto y la que más ventajas ofrece sobre las demás. Además, se trabajará con la librería Tesseract [12] para realizar la parte de detección de textos en imágenes, ya que ésta aporta una mayor precisión a la hora de reconocer la aparición de caracteres o símbolos dentro de una imagen (técnica conocida como OCR [11]).

# 3 Diseño y análisis

En este capítulo, se detallan todas aquellas decisiones de diseño que se han tomado para desarrollar la aplicación y sus diferentes funcionalidades.

## *3.1 Requisitos funcionales y no funcionales*

Antes de abordar la implementación de la aplicación, es necesario definir cuáles son los requisitos funcionales y no funcionales que se establecen para la misma.

Requisitos funcionales:

- La aplicación deberá permitir al usuario dar de alta un nuevo medicamento y su posología para llevar a cabo su seguimiento.
- La aplicación deberá implementar algún mecanismo de notificación al usuario avisándole de cuándo debe realizar una toma concreta de un medicamento.
- La aplicación deberá permitir modificar los datos relativos a un medicamento y su posología.
- El usuario debería poder confirmar que ha realizado la toma de un medicamento tanto de forma manual como de forma automática

Requisitos no funcionales:

- La interfaz de la aplicación debe ser lo más intuitiva posible para el usuario
- La interfaz de la aplicación deberá seguir el aspecto del resto de aplicaciones estándares para la plataforma.
- La aplicación deberá diseñarse con el menor número de alternativas posibles para realizar las mismas acciones.

## *3.2 Casos de uso*

A partir de los requisitos funcionales anteriormente identificados, pueden extraerse los siguientes casos de uso: registrar/editar un medicamento (Figura 3-2) y confirmación de la toma de un medicamento (Figura 3-1).

En la imagen Figura 3-2, se puede observar como se ha empleado la misma pantalla (variando la información que en ella se muestra) para poder registrar y editar un medicamento. En ambas situaciones, el usuario debe rellenar el mismo formulario.

Como se puede observar en la Figura 3-1, al usuario se le plantean dos métodos para confirmar que el medicamento escogido por el usuario coincide con el que se debe medicar y, por tanto, confirmar la toma.

Estos dos métodos son: reconocer el código nacional del medicamento y comparar las imágenes.

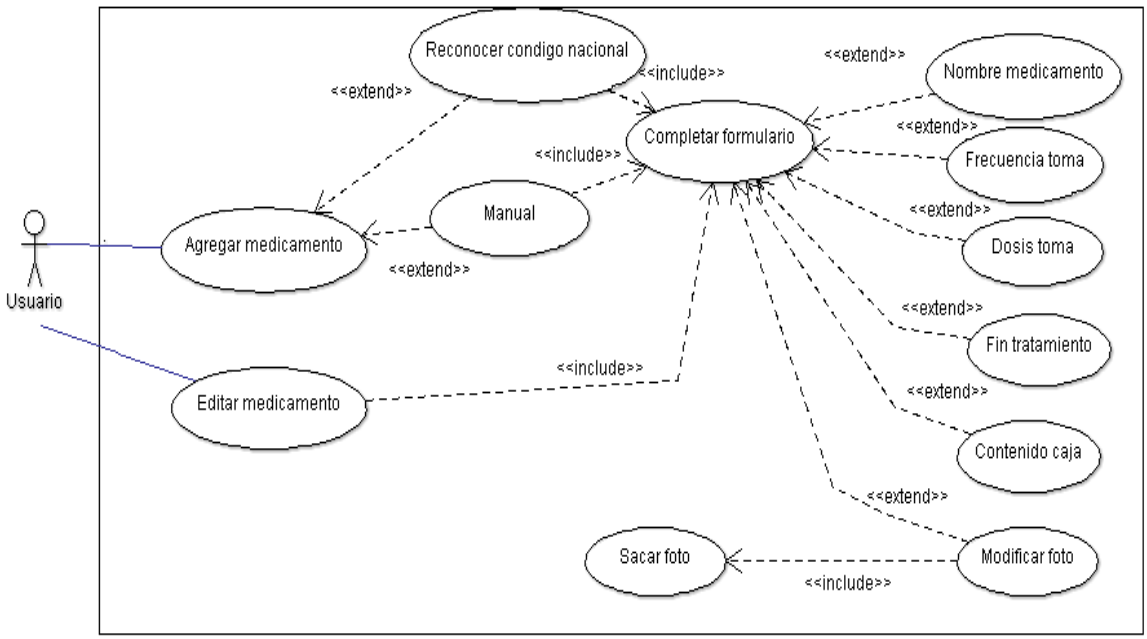



**Figura 3-2: Caso de uso para registrar/editar un medicamento**

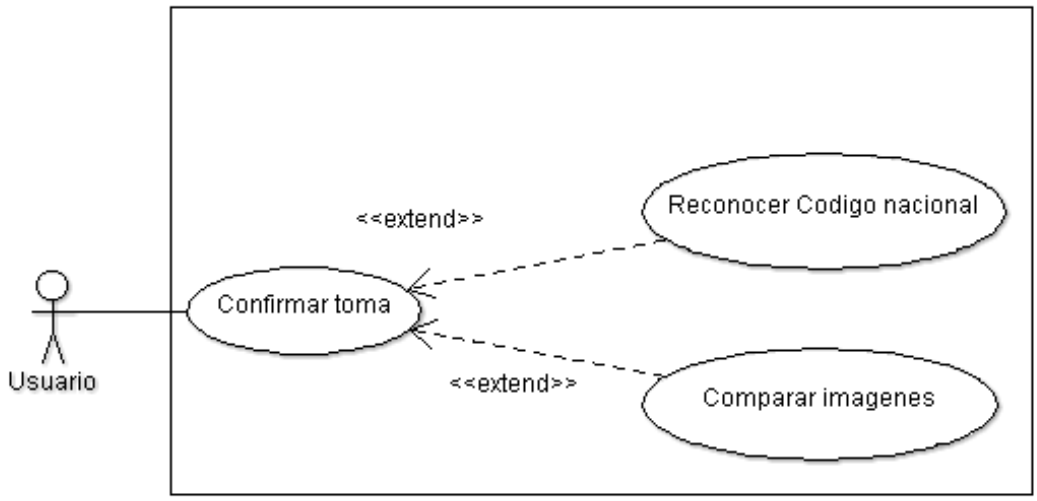



**Figura 3-1: Caso de uso para confirmar la toma de un medicamento**

El código nacional es un sistema de identificación rápido que tiene por objeto ayudar y facilitar la gestión de las oficinas de farmacia en la adquisición de los productos que se encuentran en el mercado nacional. Es decir, es una forma de identificar un medicamento determinado y, por tanto, diferenciarlo del resto (ya que este identificador es único).

Para reconocer el código nacional, el usuario deberá escanear la caja del medicamento con la cámara de su dispositivo y la aplicación detectará el código nacional.

Para comparar las imágenes, al usuario se le presentará la caja del medicamento con el que se debe medicar, en forma de transparencia. El usuario deberá ajustar dicha transparencia a la caja del medicamento para comprobar que ambas coinciden.

## 3.3 Arquitectura

A la hora de desarrollar una aplicación para el sistema operativo Android, es necesario seguir el patrón de arquitectura modelo-vista-controlador. Este patrón se caracteriza por que separa los datos (modelo) de la lógica que trabaja con ellos (controlador) y de la interfaz de usuario (vista).

Esto se debe a que en una aplicación ya se encuentra separada la vista de la lógica. Es decir, son dos capas separadas en las que se pueden trabajar de forma independiente.

En este proyecto, se ha incluido el módulo de los datos (modelo) al ser necesario trabajar con una base de datos que proporcione persistencia.

## 3.4 Diseño de la interfaz de usuario

Antes de empezar con el desarrollo de la aplicación, se prepararon una serie de maquetas en las que se planteaba una primera idea de la interfaz gráfica de la aplicación. A lo largo del desarrollo de la aplicación, se han ido modificando para cumplir con las necesidades y requisitos funcionales del proyecto, hasta alcanzar la versión final. Las maquetas sobre las cuales se basó la versión final se pueden encontrar en el Anexo D.

Dado que el objetivo es que la interfaz gráfica sea intuitiva y fácil de utilizar por cualquier persona, en ella predominan características que permiten cumplir con este objetivo. Algunos ejemplos de estas características son el uso de colores para favorecer el contraste y una lectura sencilla (el texto se lee con claridad), botones de gran tamaño, opciones con nombres representativos, etc.

Otra característica a destacar es el reducido número de pantallas. La aplicación se ha desarrollado de tal forma, que se puedan aprovechar pantallas para realizar dos acciones diferentes. Es el caso de la pantalla registro y edición de medicamentos. Ambas funcionalidades están implementadas sobre una misma pantalla. Es decir, dependiendo del botón seleccionado, aparecerá la pantalla con una funcionalidad u otra.

Gracias a estas características, la curva de aprendizaje es creciente permitiendo dominar la aplicación en el menor tiempo posible. Además, permite realizar cualquier función de las que dispone la aplicación en muy pocos pasos evitando de esta forma, que los usuarios tengan dificultades a la hora de realizar una tarea determinada.

Al utilizar un número reducido de pantallas, permite al usuario familiarizarse más rápido con la aplicación. Como resultado de esto, el usuario se sentirá más cómodo y no necesitará ayuda de ningún familiar o amigo para poder solventar cualquier problema que le pudiera surgir al intentar realizar una tarea en la aplicación.

A continuación, se detalla la funcionalidad de cada una de las pantallas y las tareas que podrá realizar el usuario en cada una de ellas.

En la mayoría de las pantallas, se puede observar que en la esquina superior izquierda está disponible un icono que permite acceder a un menú deslizante. Esto permite que la navegación para llegar a algunas pantallas, sea más rápida (se pueden ver las opciones del menú deslizante como accesos directos). Este menú y las diferentes pantallas de la aplicación, se pueden ver en el Anexo E.

1. **Menú inicial:** Primera pantalla que se muestra al usuario una vez que abre la aplicación. Está compuesta por 4 botones y el menú deslizante. En este punto, el usuario podrá realizar las tareas de registro de un medicamento, confirmar una toma, ver la lista de medicamentos o ver información de un medicamento en tiempo real (realidad aumentada). Para acceder a cada una de estas tareas, bastará con pulsar el botón correspondiente.

2. **Agregar medicamento:** Consiste en una pantalla en la que el usuario tendrá que sacar una foto a la caja del medicamento. En esta pantalla, hay dos botones cuya funcionalidad es sacar la foto y encender o apagar el flash (en el caso de que la iluminación sea escasa). Una vez sacada la foto, el usuario deberá recortarla para definir mejor los contornos del medicamento (esto facilita a la aplicación a realizar el reconocimiento de imágenes a la hora de confirmar una toma). Tras estos pasos, al usuario se le muestran dos opciones: registro manual o automático.

   Si selecciona registro automático, el usuario deberá escanear el código nacional y la aplicación detectará el medicamento. Una vez detectado el medicamento, la aplicación rellenará algunos datos del medicamento de forma automática, agilizando el registro del medicamento.
   El registro manual, se utilizará en aquellos casos en los que el usuario no dispone de la caja del medicamento (ya que no podrá realizar el escaneo del código nacional).

   Tras seleccionar una de estas opciones, se llega a la pantalla "Registrar medicamento", en la que el usuario deberá rellenar los datos necesarios para registrar finalmente el medicamento.

3. **Registrar medicamento:** Formulario que deberá completar el usuario para registrar correctamente un medicamento.

4. **Confirmar toma:** En esta pantalla, el usuario podrá confirmar la toma de un medicamento, además de mostrar una lista con las próximas tomas. De esta forma, el usuario sabrá en cuanto tiempo aproximadamente debe volver a medicarse.
   Al seleccionar un medicamento, se procederá a realizar la confirmación de la toma empleando el método seleccionado en la pantalla de ajustes (OCR o mediante reconocimiento de imagen).

5. **Lista de medicamentos:** Pantalla en la cual, el usuario podrá ver todos los medicamentos que tiene registrados en un momento determinado. Al seleccionar uno de ellos, la aplicación abre la pantalla de detalle del medicamento.

6. **Detalle del medicamento:** En esta pantalla se observa diferentes datos de interés del medicamento. Además, puede seleccionar el botón de "editar", permitiendo al usuario realizar modificaciones en el medicamento.
   Otra funcionalidad a destacar en esta pantalla, es el botón "ver Vademecum". Al seleccionarlo, al usuario se le mostrará la pantalla "Vademecum".

7. **Vademecum:** Pantalla en la que se muestra información sobre formas de uso, efectos secundarios… de un medicamento determinado.

8. **Editar medicamento:** permite modificar algunos datos del medicamento registrado. Además, permite "borrar" dicho medicamento. Esta funcionalidad, consiste en desactivar las alertas para este medicamento, es decir, es otra forma de finalizar el tratamiento.

9. **Ajustes:** pantalla en la que el usuario podrá modificar algunos aspectos de la aplicación.

10. **Tutorial:** Consiste en una serie de imágenes que ayudarán al usuario a entender el funcionamiento de aquellas pantallas cuya funcionalidad no este clara.

11. **Historial de tomas:** Permite ver en una tabla todas las confirmaciones de toma de los medicamentos, fecha de confirmación, entre otra información. Esto permite observar a una segunda persona si, por ejemplo, su familiar está realizando los tratamientos correctamente.

## *3.5 Interfaz gráfica*

Una vez se tiene una primera idea o base en cuanto al aspecto de la interfaz gráfica, se procede a su implementación. Dados los requisitos no funcionales definidos para la aplicación y atendiendo a las guías de diseño de interfaces para aplicaciones móviles que se mencionó en el capítulo de introducción, se ha utilizado en la medida de lo posible, "*Material Design*" [4].

Como se comentó anteriormente, "*Material Design*" [4] es una guía de diseño de interfaces utilizada en el sistema operativo Android, y desarrollada por Google. Al tratarse de una guía desarrollada para Android, todos aquellos elementos de la interfaz gráfica son generales para cualquier dispositivo Android. De esta forma, independientemente del dispositivo Android que posea el usuario, podrá hacer uso de la aplicación sin necesidad de un manual o tutorial para conocer la funcionalidad de cada uno de los botones que se le presentan en la interfaz (los usuarios deben haber utilizado anteriormente este sistema operativo).

En el caso de que el usuario no tenga dichos conocimientos, la aplicación incluye un pequeño tutorial, en el que se detalla la forma de interactuar con los elementos de las pantallas más importantes de la aplicación.

En ocasiones, para facilitar el uso de la aplicación por parte del sector de la población al que va principalmente orientada, se ha tenido que modificar algunos componentes de la interfaz de usuario. Esto implica, en algunos casos, no seguir el estándar definido en "*Material Design*" [4] (algunos elementos poseen un tamaño mayor al habitual indicado en el estándar).

## *3.6 Estructura de la base de datos*

La aplicación dispondrá de dos bases de datos independientes entre sí. En una de ellas, se almacenarán todos aquellos datos referentes a los tratamientos y los medicamentos de los usuarios. En la otra, llamada "Vademecum", se almacena información referente a todos los medicamentos que se pueden utilizar para mostrar información al usuario de un medicamento determinado y, además, para facilitarle el registro de un nuevo medicamento,

ya que la aplicación reconocerá el medicamento a registrar y completará parte del formulario del registro de forma automática. Todos estos datos de los medicamentos se extraen de la página del Vademecum [17].

En la base de datos propia de la aplicación, se diferencian una serie de tablas relacionadas entre sí de tal forma, que es posible extraer cualquier información necesaria de manera sencilla.
En la Figura 3-3, se muestra un diagrama entidad-relación de esta base de datos.

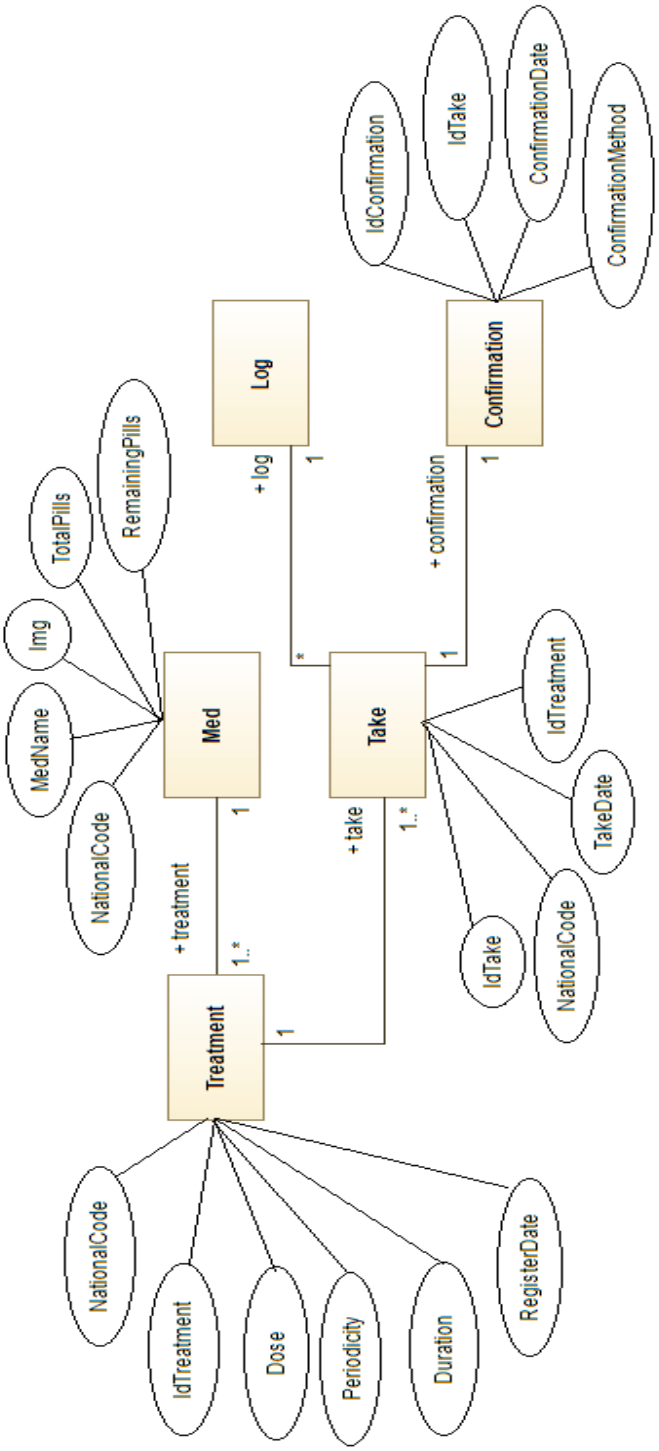



**Figura 3-3: Diagrama entidad-relación de la base de datos de la aplicación**

A continuación, se enumeran las entidades y atributos que forman cada una de estas tablas, además de las relaciones entre ellas. Ninguno de los atributos tomará valores nulos, ya que, a la hora de rellenar la tabla con información todos ellos poseen un valor determinado.

A la hora de registrar un medicamento, se tiene que tener en cuenta que éste pertenece a un tratamiento. Se debe entender como tratamiento el tiempo que nos indica el médico hasta que se debe dejar de utilizar dicho medicamento. Por tanto, registrar un medicamento implica asignarlo también a un tratamiento. En este caso, esta relación es 1 a 1, pero al tratarse de medicamentos, es posible que el paciente necesite pasado un tiempo volver a medicarse con el mismo medicamento. Para evitar que el usuario tenga que registrar otra vez el medicamento, se estableció esta relación como una 1-n. De tal forma, que un medicamento puede estar asignado a uno o más tratamientos (no de manera simultánea).
Además, estos datos permanecen en la base de datos en el caso de que se quiera observar el número de tratamientos de un paciente, entre otra información de relevancia.
Las tablas *Treatment* y *Take* poseen una relación 1-n respectivamente, ya que las tomas del medicamento se hacen con respecto a un tratamiento.

- *Treatment*: Representa un tratamiento de un medicamento. Posee los siguientes atributos:
    - *NationalCode*: identificador único del medicamento. Actúa como clave externa para relacionar esta tabla con las tablas Med y Take.
    - *IdTreatment*: identificador numérico único para diferenciar los tratamientos entre sí. Actúa como clave primaria.
    - *Dose*: indica la dosis del medicamento con la que debe medicarse el usuario.
    - *Periodicity*: indica cada cuanto tiempo debe medicarse el usuario con un medicamento determinado.
    - *Duration*: indica la fecha en la que se termina el tratamiento.
    - RegisterDate: indica la fecha en la que se registró un medicamento, es decir, la fecha en la que se creó el tratamiento.
- *Med*: Representa un medicamento.
    - *NationalCode*: identificador único del medicamento. Actúa como clave primaria para relacionar esta tabla con la tabla Treatment.
    - *MedName*: nombre del medicamento.
    - *Img*: almacena la imagen del medicamento tomada por el usuario a la hora del registro.
    - *TotalPills*: indica el número de pastillas o sobres que contiene la caja del medicamento.
    - RemainingPills: indica el número de pastillas o sobres restantes de un medicamento.

- *Take*: Representa la toma de un medicamento por parte del usuario.
    - *IdTake*: identificador único de cada toma. Actúa como clave primaria.
    - *NationalCode*: identificador único de un medicamento. Actúa como clave externa para relacionar esta tabla con la tabla *Med*.
    - *TakeDate*: indica la fecha en la que se debe realizar la próxima toma.
    - *IdTreatment*: identificador único de un tratamiento. Actúa como clave externa para relacionar esta tabla con la tabla *Treatment*. A pesar de que es posible acceder a la información de la tabla *Treatment* a partir del *NationalCode* de esta tabla, se ha incluido este campo "redundante" para

facilitar las consultas a la base de datos (acceso más rápido con este campo a la tabla *Treatment*).

Las tablas *Take* y *Confirmation* poseen una relación 1 a 1 (a cada una de las tomas le corresponde una confirmación). Es decir, cuando al usuario se le avisa de una toma de un medicamento, debe confirmar que el medicamento escogido es el correcto (cada toma implica una confirmación).

- *Confirmation*: Representa la confirmación de una toma de un medicamento.
  - *IdConfirmation*: identificador único de cada confirmación de una toma. Actúa como clave primaria.
  - *IdTake*: identificador único de una toma de un medicamento. Actúa como clave externa para relacionar esta tabla con la tabla Take.
  - *ConfirmationDate*: indica la fecha en la que se ha realizado la confirmación. Esto permite saber si el usuario confirma la toma en el momento en el que la aplicación le muestra la notificación o si, por cualquier motivo, realiza la confirmación pasados unos minutos desde esta notificación.
  - *ConfirmationMethod*: indica la forma en la que se ha confirmado la toma del medicamento. La confirmación de un medicamento es posible realizarlo de 2 formas: escaneando la caja (detectando el código nacional o comparando la caja del medicamento con una imagen) o de forma manual con un botón situado en una de las pantallas de la aplicación. Se han diseñado ambos casos ya que es posible que el usuario, a la hora de realizar una toma, no disponga de la caja del medicamento (no podría realizar el escaneo de la caja). De esta forma, le damos una opción al usuario de confirmar la toma sin la necesidad de disponer de la caja del medicamento. A pesar de que el usuario dispone de diferentes formas de confirmar una toma, se intenta fomentar el uso de la cámara en lugar del botón. Si el usuario puede cumplir el mismo objetivo de una forma más fácil y rápida (pulsando un botón sin necesidad de buscar la caja del medicamento y escanearla con la cámara), optará por esta vía y no utilizará el método de escaneo con la cámara en ningún momento.

Por último, en el diagrama se muestra una entidad con el nombre Log. Consiste en una vista, es decir, la aplicación dispone de una pantalla en la que se muestra información sobre las tomas de los medicamentos y sus confirmaciones. Esta entidad es la encargada de mostrar toda la información relevante para el usuario, que se extrae a partir del resto de las tablas de la base de datos.

## *3.7 Diagramas de transición de estados*

A continuación, se muestra en un diagrama de transición de estados, todas las relaciones entre las pantallas de la aplicación (Figura 3-4) y la forma que tiene el usuario para pasar de una a otra (normalmente a través de un botón en la pantalla).
Como se puede observar en el diagrama, las opciones a través de las cuales el usuario podrá salir de la aplicación son seleccionando una opción en el menú desplegable o llegando al menú inicial y pulsando el botón "Atrás" del dispositivo móvil.

En el diagrama se han omitido algunas pantallas que utilizan la cámara del dispositivo. Por ejemplo, tras sacar la foto para registrar un medicamento, la aplicación abre una pantalla en

la cual el usuario debe recortar la foto. Tras este paso, el usuario tiene la posibilidad de realizar un registro automático del medicamento, por lo que se abre una pantalla en la cual debe escanear la caja del medicamento empleando la cámara de su dispositivo móvil.

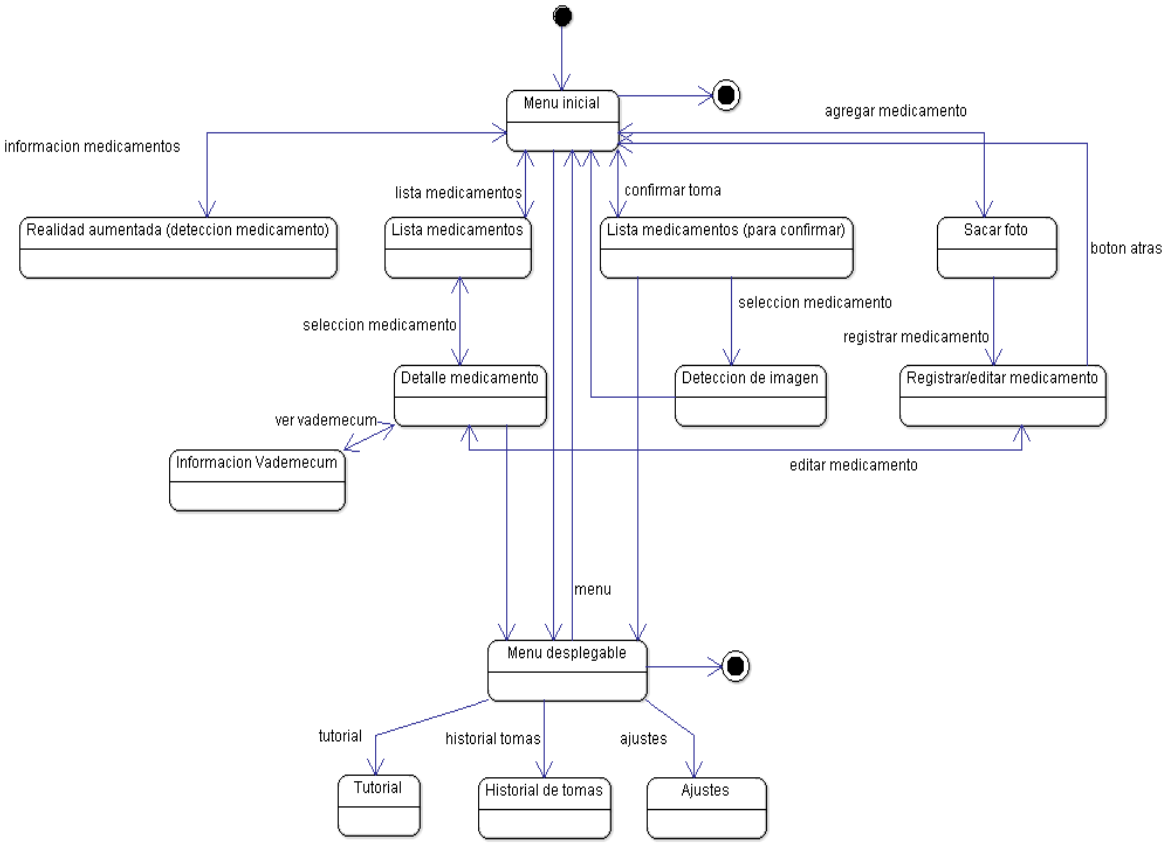



**Figura 3-4: Diagrama de transición de estados de la aplicación**

## *3.8 Notificaciones*

A fin de implementar la posibilidad de que el dispositivo móvil avise al usuario de que es el momento de realizar una toma, tal y como se definieron en los requisitos de la aplicación, en Android se dispone del mecanismo de las notificaciones.

El objetivo de una notificación es avisar o alertar al usuario sobre algo. En este proyecto, es el módulo encargado de alertar al usuario que es la hora de medicarse. Cuando llega dicha hora, además de avisar al usuario, en el dispositivo aparece una notificación en la parte superior de la pantalla que indica que es la hora de medicarse.

En la aplicación, se usan diferentes formas de alerta para captar la atención del usuario, como por ejemplo el encendido de un led (en el caso de que el dispositivo posea uno), la vibración del dispositivo móvil, o la reproducción de un sonido. El usuario podrá activar o desactivar estas alertas para configurarlo a su gusto.

En la Figura 3-5, se muestra a modo de ejemplo, una notificación de toma de un medicamento determinado:

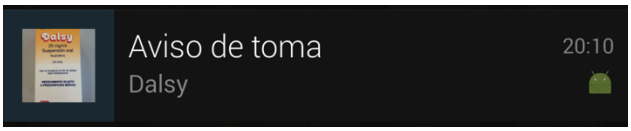



**Figura 3-5: Notificación de toma de un medicamento**

En el caso de que no se reciban muchas notificaciones, el usuario podrá "abrir o extender" esta notificación para ver la imagen del medicamento en un mayor tamaño.

Al seleccionar la notificación, se abrirá la aplicación para realizar la confirmación de la toma de dicho medicamento.

# 4 Desarrollo

En esta sección se detallan los diferentes algoritmos que se investigarán para, tras realizar una serie de pruebas, seleccionar aquel que se integrará en la aplicación. Además, se describen las herramientas que han sido utilizadas durante el desarrollo del proyecto.

## *4.1 Herramientas utilizadas*

### 4.1.1 Entornos de desarrollo

Para la elaboración de este proyecto, se han utilizado dos entornos de desarrollo. Cada uno de estos entornos se ha utilizado para la elaboración de los diferentes módulos que formarán la aplicación final.

Por una parte, se ha utilizado el entorno Android Studio [18] para la elaboración de todos aquellos módulos referentes a la aplicación. Además de la integración en la aplicación del módulo encargado de la identificación de imágenes sobre el que trata este Trabajo de Fin de Grado. La razón por la cual se ha seleccionado este entorno es porque, al pertenecer a la empresa Google, se encuentra actualizado con las últimas novedades en cuanto al sistema operativo Android, al cual va destinado la aplicación.

Por otro lado, se ha empleado el entorno Eclipse [19] para el desarrollo exclusivo del módulo de identificación de imágenes. En este entorno, se han realizado todas las pruebas necesarias para identificar el algoritmo más adecuado para utilizarlo en la aplicación.

### 4.1.2 Control de versiones

Al tratarse de un proyecto que implica su integración en otro más grande y por tanto elaborado por más personas, es necesario emplear un sistema de control de versiones. De esta forma, los integrantes del proyecto pueden trabajar de forma independiente y de manera coordinada, a la hora de realizar la integración de todos los módulos desarrollados. En particular, el sistema de control de versiones empleado es Git [20] y el servidor utilizado para albergar el repositorio es Bitbucket [21].

La razón por la cual se ha seleccionado Git [20], es porque destaca por su facilidad de uso y flexibilidad en la gestión y versionado de código, además de estar integrada perfectamente con el entorno de desarrollo Android Studio [18]. Esto permite agilizar todas aquellas tareas que consisten en integrar, en una misma versión, las diferentes modificaciones realizadas por los integrantes del proyecto.

### 4.1.3 Control de tareas

Como se ha explicado anteriormente, la realización de un trabajo coordinado para llevar a cabo este proyecto ha sido muy importante. La herramienta utilizada para la división de tareas a realizar por los integrantes del proyecto es Trello [22]. Esta herramienta permite definir de forma ordenada, todas aquellas tareas y objetivos que se deben cumplir, además de establecer entre otras características, la persona encargada de realizar cada una de estas tareas. Esta herramienta se fundamenta en el proceso empleado en proyectos de equipo conocido como “Scrum”, si bien su uso no está restringido exclusivamente al empleo de este.

Se ha seleccionado esta herramienta ya que era la mejor forma de organizar todas aquellas tareas necesarias para el desarrollo completo de la aplicación. De esta forma, cada integrante tenía marcadas, desde el inicio de cada una de las iteraciones, todas aquellas tareas en las que debía trabajar para cumplir con el objetivo final de desarrollar la aplicación.

### 4.1.4 Base de datos

Para todas aquellas tareas relacionadas con la base de datos, se ha empleado la herramienta SQLite [23]. Consiste en un sistema de gestión de base de datos liviano.

Se ha empleado debido a su facilidad de uso y a que permitía realizar una serie de pruebas en cuanto a definición de tablas, consultas SQL, etc., permitiendo así una comprobación previa a la integración de la base de datos con la aplicación final. De esta forma, se ha comprobado que el módulo encargado de la base de datos funciona correctamente, y es posible realizar su integración junto con los demás módulos de la aplicación.

### 4.1.5 Diagramas y maquetas

Para la realización de los diferentes diagramas y maquetas que representan de forma gráfica cómo será el diseño de la aplicación, se han empleado las herramientas ArgoUML [24], Modelio [25] y Moqups [26].

Estas herramientas han sido utilizadas anteriormente por los integrantes del proyecto. Por esta razón, todas las tareas relacionadas con el diseño de la aplicación, se han llevado a cabo de forma rápida y eficiente. Permitiendo así, en lugar de emplear el tiempo en el aprendizaje de una nueva herramienta, emplearlo en el desarrollo de otros módulos de la aplicación.

Para el diseño de las maquetas se ha empleado la herramienta Moqups [26] ya que es una herramienta fácil de utilizar y porque posee diferentes elementos, por ejemplo, la imagen de un dispositivo móvil (en ella se van incluyendo los diferentes elementos que formarán el diseño de una pantalla), que nos permite tener una idea mucho más definida de cómo será el aspecto final de una pantalla determinada. Además, sirve como una referencia en cuanto al tamaño de los componentes que se desean incluir en la pantalla, permitiendo tener una visión un poco más detallada de la ubicación y el tamaño que debería ocupar cada uno de los elementos.

## *4.2 Identificación de imágenes*

La tecnología de identificación de imágenes consiste en a partir de una imagen, detectar un elemento determinado y, además, reconocerlo diferenciándolo de los demás.

Para la identificación de imágenes, existen numerosos algoritmos, entre todos ellos, se han escogido para su investigación aquellos con los que mejores resultados se podrían obtener. El objetivo es encontrar cuál de todos estos algoritmos es el más adecuado para la aplicación y con el que se obtengan mejores resultados. Todos estos algoritmos poseen un sistema de funcionamiento diferente. Para la integración de la aplicación, es necesario obtener el grado o porcentaje de similitud entre ambas imágenes. Por esta razón, cuando se haya seleccionado un algoritmo para la integración, se debe convertir el resultado que devuelva dicho algoritmo en forma de porcentaje de similitud.

Se han sometido todos estos algoritmos a un caso de prueba (situación que se dará cuando el usuario utilice la aplicación) para determinar cuál de todos ellos es el que mejores resultados aporta, para posteriormente, incluirlo en la aplicación final.
Este caso de prueba, consiste en observar los resultados que se obtienen de aplicar los diferentes algoritmos para reconocer un medicamento en una imagen.

Para ello, se utilizará la imagen de la caja de un medicamento determinado. En este caso, el medicamento escogido para la realización de la prueba es Cefazolina (Figura 4-1). Esta imagen debería ser similar a la que dispondrá el usuario a la hora de registrar este medicamento en la aplicación.

A continuación, se supone el caso de que el usuario debe medicarse con este medicamento. Por ello, se incluye una imagen en la que se encuentra la caja del medicamento en una mesa (Figura 4-2). Es decir, se establece el escenario en el cual, el usuario con la utilización de la cámara de su dispositivo, intentará escanear el medicamento para confirmar que es el correcto para, posteriormente, medicarse con él.


**Figura 4-1: Caja del medicamento Cefazolina**


**Figura 4-2: Caja del medicamento Cefazolina situado en una mesa**

Estos algoritmos suelen poseer diferentes métodos de comparación o parámetros que se pueden modificar. En esta sección, se analizarán los resultados obtenidos y esperados de estos algoritmos y, más tarde en la sección de pruebas, se analizará con una mayor profundidad y detalle cada uno de estos aspectos modificables de los algoritmos.

### 4.2.1 *Template Matching*

Es una técnica de procesamiento digital de imágenes para la búsqueda de un área de una imagen que coincida o sea similar con una imagen a modo de plantilla.

Para emplear esta técnica, se necesitan dos imágenes: una en la que se espera encontrar la plantilla y otra que actuará como plantilla.

Para identificar el área que coincida o sea similar a la plantilla, el procedimiento a seguir es desplazar (1 pixel de izquierda a derecha, y de arriba a abajo) la plantilla a lo largo de la imagen principal. En cada posición, una métrica calcula el nivel de similitud entre la plantilla y la imagen. Tras realizar este desplazamiento por toda la imagen, se puede obtener el área con mayor similitud. En la Figura 4-3, se muestra un ejemplo del proceso completo y los resultados obtenidos.

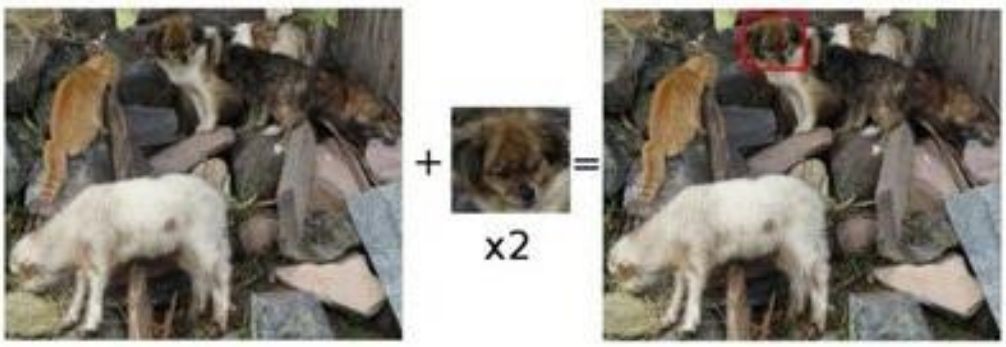



**Figura 4-3 : Extraída de la página de OpenCV sobre el algoritmo Template Matching**

Para que se pueda observar de una forma más visual los resultados obtenidos, si el algoritmo "detecta una imagen dentro de la otra", se dibuja un rectángulo negro marcando la zona con mayor similitud. El algoritmo encuentra el pixel con mayor grado de similitud y, empleando dicho pixel como la esquina superior izquierda del rectángulo, se dibuja el área que ocupa la plantilla (partiendo del ancho y alto de la plantilla). Es decir, se dibuja el rectángulo independientemente de la posición real en la que se encuentra la caja del medicamento.

Tras comprender el funcionamiento de este algoritmo, se procede a aplicar en el caso de prueba y se obtiene el siguiente resultado (Figura 4-4).


**Figura 4-4: Resultado de aplicar el algoritmo Template Matching en el caso de prueba**

Como se puede observar en la imagen anterior, el algoritmo detecta el medicamento correctamente (con un método de comparación determinado). Es decir, se puede pensar que en un caso ideal como es este (el usuario coloca la caja del medicamento justo en frente), el algoritmo funciona perfectamente. Como es posible que no todos los usuarios coloquen de esta forma la caja del medicamento, debemos realizar la misma prueba, pero situando la caja del medicamento en una posición diferente.

En esta ocasión, la caja del medicamento se ha colocado diagonalmente (Figura 4-5) y los resultados obtenidos han sido los siguientes (Figura 4-6).

**Figura 4-5: Caja en posición diagonal**

**Figura 4-6: Resultados obtenidos al aplicar el algoritmo Template Matching con el medicamento Cefazolina**

Como se puede observar, el algoritmo continúa detectando el medicamento. Llegado este punto, solo queda comprobar una situación más para confirmar que el algoritmo funciona correctamente. Para ello, se empleará en este mismo escenario, la imagen de otro medicamento (Figura 4-7). La elección de este nuevo medicamento se ha realizado dado a su alta similitud con el medicamento anterior.

En el caso de que el algoritmo funcione correctamente, debería detectar que estos medicamentos son diferentes entre sí y, por tanto, no mostrar ninguna zona de similitud (rectángulo negro en la imagen).


**Figura 4-7: Caja del medicamento Cefotaxima**

**Figura 4-8: Resultados obtenidos tras ejecutar Template Matching con el medicamento Cefotaxima**

Se procede a analizar los resultados obtenidos (Figura 4-8). Como se puede observar, el algoritmo sigue detectando una alta coincidencia entre las imágenes, pero no identifica que ambas son diferentes.

En vista a los resultados obtenidos, este algoritmo no es el adecuado para la integración con el proyecto. El motivo es que el algoritmo, detecta similitudes a nivel de pixel, por lo que no es posible identificar que dos imágenes similares, sean diferentes entre sí. No es condición suficiente detectar similitudes entre los pixeles para decir que dos imágenes coinciden.

Además, otro aspecto desfavorable en cuanto a la utilización de este algoritmo es la dependencia del tamaño de ambas imágenes. No se puede realizar la búsqueda de una

plantilla con un tamaño mayor a la imagen en la que se desea encontrar dicha plantilla. Esto puede provocar un comportamiento y unos resultados no deseados en la aplicación.

### 4.2.2 Histogramas

Consiste en agrupar todos los pixeles que posean una misma característica (color, intensidad) en diferentes grupos. Tras representarlo en una gráfica, se puede diferenciar la cantidad de pixeles que pertenecen a un determinado grupo. Para comparar la similitud entre dos imágenes, se debe comparar cada uno de los histogramas de los diferentes canales RGB (rojo, verde, azul) de las imágenes.
A continuación, se muestra un ejemplo de la utilización de esta técnica sobre una imagen de un paisaje (Figura 4-10) y el resultado que se obtiene (Figura 4-9):

**Figura 4-10: Extraída de la pagina de OpenCV sobre el algoritmo Histogram Comparison**

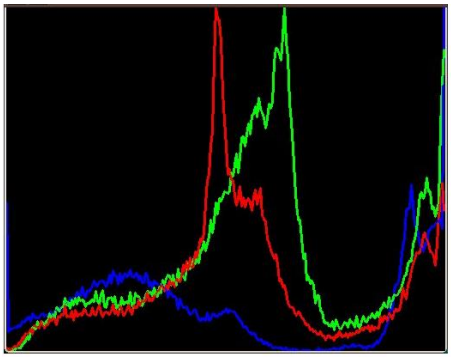


**Figura 4-9: Extraída de la página de OpenCV sobre el algoritmo Histogram Comparison**

Se procede a aplicar este algoritmo al escenario de prueba. A continuación se pueden observar los histogramas (Figura 4-12 y Figura 4-11) para cada una de las imágenes del caso de prueba:

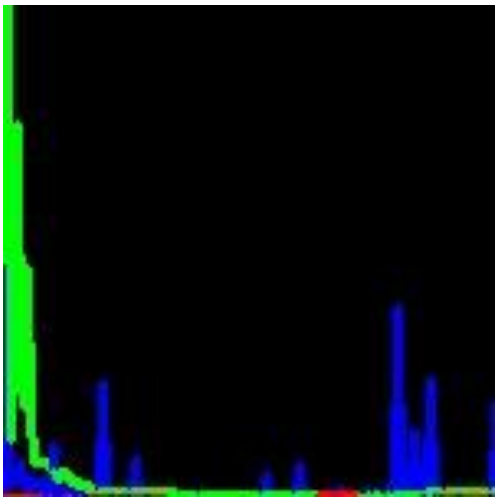


**Figura 4-12: Histograma obtenido de la imagen del medicamento Cefazolina (Figura 4-1)**

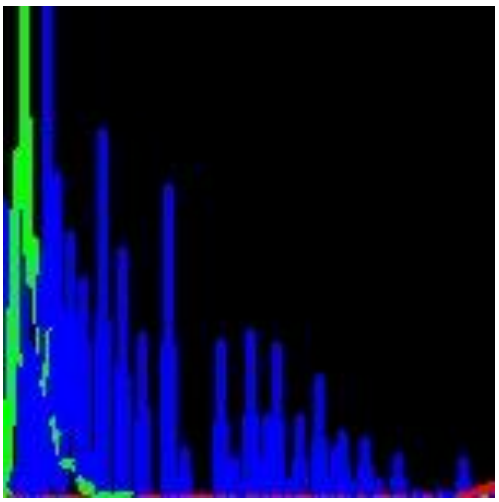


**Figura 4-11: Histograma obtenido de la imagen del medicamento Cefazolina sobre una mesa (Figura 4-2)**

Analizando los resultados obtenidos, se puede ver que ambos histogramas tienen una estructura bastante diferente. Esto se debe a que la tecnica de los histogramas depende de muchos factores, como pueden ser la luminosidad de las imágenes, los colores, etc.
Al depender de estas caracteristicas, se puede dar el caso de obtener resultados diferentes si, por ejemplo, se situa la caja del medicamento en una mesa blanca o en una mesa con un mantel de color rojo. En ambas situaciones, los canales RGB variarán de tal forma que analizando solamente los histogramas, la conlcusión a la cual se llegaría sería que ambas imágenes son completamente diferentes (A pesar de que la misma caja del medicamento este presente en ambas imágenes).

Por esta razón, este algoritmo no es adecuado para la integración con la aplicación.

### 4.2.3 *Feature Matching*

Esta técnica se basa en la búsqueda de características similares entre dos imágenes dadas. Para encontrar estas características se utiliza un método conocido como fuerza bruta (existen diferentes métodos). Es decir, se selecciona una característica de una imagen y se compara con todas las demás características de la segunda imagen. Haciendo un cálculo de distancias para comparar unas con otras, se obtiene la característica más próxima y, por tanto, más similar.

El algoritmo se divide en dos partes: obtención de características similares y comprobar coincidencias entre ellas.
La primera parte, consiste en obtener una serie de puntos clave de una imagen. Es decir, se busca puntos que sean característicos de la imagen. Este procedimiento lo realizan los denominados detectores o descriptores.
Una vez realizado este procedimiento para las dos imágenes que se desean comparar, se pueden analizar y detectar cuales de estos puntos coinciden en ambas imágenes y, por tanto, se puede tener una idea aproximada de que tan similares son las imágenes.
La segunda parte del algoritmo es la encargada de encontrar la similitud entre los descriptores de ambas imágenes. Este procedimiento lo realizan los denominados *matchers* o comparadores. Para realizar esta búsqueda, el comparador debe comprobar la similitud de cada descriptor de una imagen con cada uno de los descriptores de la otra imagen. Es decir, debe comparar todos los descriptores de una imagen con todos los de la otra.

En la Figura 4-13 se muestra un ejemplo de este algoritmo aplicado a dos imágenes:

**Figura 4-13: Extraída de la página de OpenCV sobre el algoritmo Feature Matching**

Como observamos en la imagen anterior, el algoritmo ha encontrado una serie de puntos característicos de la imagen de la caja que se desea buscar. Una vez detectados todos estos puntos, se buscan las coincidencias en los puntos característicos de la otra imagen.
En el resultado final, se muestra con una línea azul la coincidencia entre los puntos de ambas imágenes.

Otro aspecto que hay que destacar, es que este algoritmo no depende de la posición del objeto dentro de la otra imagen. Esto nos permite ampliar el rango de posibilidades a las cuales se va a enfrentar la aplicación cuando un usuario intente confirmar una toma. Con la utilización de este algoritmo, se evita el conflicto de que el usuario no coloque la caja del medicamento en una posición adecuada. Por tanto, se abarcarían todos los problemas que dependían de la posición en la que se encontraba la caja del medicamento.

Otro aspecto favorable de la utilización de este algoritmo, es la independencia de tamaño de ambas imágenes. Al tratarse de la detección de unos puntos característicos de la imagen, el tamaño no afecta en ningún momento a la realización de esta tarea.

Este algoritmo posee una serie de características que, dependiendo del valor con el que se hayan configurado, los resultados obtenidos pueden variar. En la sección 5 “Integración, pruebas y resultados”, se analizan estas características de una forma más detallada.

A continuación, se procede a aplicar este algoritmo al caso de prueba y se analizan los resultados (Figura 4-14):


**Figura 4-14: Resultado obtenido al aplicar el algoritmo Feature Matching en el caso de prueba**

Como se puede observar, se ha detectado una serie de coincidencias entre un conjunto de puntos de ambas imágenes. En vistas al resultado obtenido, podemos aplicar el algoritmo en otra prueba.
En este caso, se variará la posición de la caja del medicamento (Figura 4-5). De esta forma, se confirma que el algoritmo funciona independientemente de la posición que ocupen los elementos dentro de las imágenes.

Tras la ejecución de esta prueba, se obtiene el resultado mostrado en la Figura 4-16.


**Figura 4-16: Resultado de ejecutar el algoritmo Feature Matching variando la posición de la caja del medicamento**


**Figura 4-15: Resultado de ejecutar el algoritmo Feature Matching con el medicamento Cefotaxima (Figura 4-7)**

Al igual que en el anterior caso, el resultado obtenido es el esperado. Además, observamos que la posición de la caja del medicamento no influye en el procedimiento que realiza el algoritmo.

Por último, para comprobar el correcto funcionamiento del algoritmo, se utilizará un medicamento similar (Figura 4-7) al del caso de prueba anterior y se analizará el resultado obtenido (Figura 4-15):

Se puede comprobar que el algoritmo continúa funcionando correctamente. El algoritmo ha detectado una serie de puntos que coinciden en ambas imágenes. El problema que se plantea a continuación, es cómo se puede saber si el medicamento a buscar coincide con la caja que escanea el usuario. Para resolver este problema, es necesario realizar una serie de pruebas en las cuales se modificarán una serie de características del algoritmo. La solución a este problema consiste en, a partir de los resultados obtenidos de aplicar el algoritmo, obtener un porcentaje de similitud entre ambas imágenes. De esta forma, se puede diferenciar entre los dos casos de prueba anteriores. Cuanto mayor sea este porcentaje, mayor es la similitud entre ambas imágenes. Es decir, se espera que este porcentaje sea mayor (aunque no por una gran diferencia) en la segunda prueba que en la anterior.

Tras el análisis del algoritmo, se puede comprobar que es el más adecuado para la integración con el proyecto. Tras realizar una serie de pruebas para confirmar el correcto funcionamiento del algoritmo, se procede a su integración con la aplicación.

### 4.2.4 OCR

Consiste en una técnica que identifica automáticamente todos los símbolos o caracteres pertenecientes a un determinado alfabeto en una imagen. Tras realizar un procesamiento previo de la imagen, se busca un patrón (carácter o símbolo del alfabeto) dentro de la imagen y, una vez encontrado, se devuelve en forma de dato (comprensible o reconocible para un ordenador) para que se pueda trabajar con él.

Para que esta técnica funcione correctamente, la imagen debe tener una buena resolución, por el contrario, la búsqueda del patrón en ella puede dar lugar a resultados no deseados. El nivel de resolución necesario para el correcto funcionamiento de esta técnica, es fácilmente

alcanzable por cualquier dispositivo de la actualidad (cuanto más antiguo sea el dispositivo, más dificultades se pueden encontrar para emplear esta técnica).

En la Figura 4-17, se muestra un ejemplo de la utilización de esta técnica para detectar los números y caracteres de una matrícula de un vehículo.


**Figura 4-17: Resultado de aplicar la técnica OCR en la matrícula de un vehículo**

En la aplicación, el usuario deberá escanear el código nacional de la caja del medicamento. Este código nacional, contiene un número de control que la aplicación ignora. Este número de control no aporta ninguna información de interés para la aplicación. De esta forma, el formato que se utiliza para representar el código nacional de un medicamento, consiste en un conjunto formado por 6 números.

Para comprobar el funcionamiento de este algoritmo, se ha escaneado con la cámara del dispositivo móvil la caja de un medicamento, y se ha comprobado que el código nacional del medicamento escaneado es correcto. Además, al tratarse de una técnica que se debe ejecutar constantemente y requiere de una resolución de imagen determinada (el algoritmo realiza una búsqueda constante hasta que se escanea un código nacional correcto), se han realizado una serie de pruebas en las cuales se modificaron las condiciones de luminosidad del entorno. En todos estos casos, el resultado del algoritmo es el esperado.
Por estas razones, se ha decidido integrarlo a la aplicación como otro método para la confirmación de toma de un medicamento. Además, este algoritmo es utilizado también en el registro de un medicamento. El usuario, tras sacar una foto a la caja del medicamento a registrar, tiene la posibilidad de escanear el código nacional del medicamento para que la aplicación complete automáticamente una serie de datos en el formulario de registro.

## *4.3 Base de datos*

Para almacenar todos los medicamentos que registra el usuario al utilizar la aplicación, es necesario la utilización de una base de datos.

En la aplicación se pueden diferenciar dos bases de datos: una propia de la aplicación y otra formada por un gran número de medicamentos (“Vademecum”).

A la hora de registrar un medicamento, la aplicación se encarga de buscar si dicho medicamento existe o no para facilitar al usuario el proceso de registro de un medicamento. Es decir, si el medicamento se encuentra en el Vademecum [17], la aplicación rellena algunos de los campos que son necesarios para el registro del medicamento, ahorrando así tiempo al usuario y facilitándole la realización de esta tarea. Además, de esta base de datos se puede extraer información de interés de cada uno de los medicamentos como, por ejemplo, efectos secundarios, modo de empleo, etc.

Toda esta información es accesible por el usuario a través de la pantalla denominada "Vademecum".

En la base propia de la aplicación, se almacenarán todos los medicamentos que registra el usuario, los diferentes tratamientos, etc.

La aplicación debe estar constantemente accediendo a la base de datos para poder así, extraer toda la información necesaria de los medicamentos. Para evitar que la aplicación se ralentice debido a un alto grado de procesamiento por la gestión de la base de datos, las pantallas de la aplicación se han organizado de tal forma que no sea necesario realizar un gran número de accesos a la base de datos. Para ello, al abrir la aplicación (en la pantalla en la que se muestra la lista de medicamentos) se realiza una primera consulta a la base de datos para extraer la mayor cantidad de información sobre los medicamentos cuya información se mostrará a lo largo de la aplicación. Es decir, tras esta consulta, la información sobre los medicamentos fluye entre las pantallas de la aplicación evitando así, la constante sobrecarga que implicaría realizar una consulta siempre que se deseara obtener información de un medicamento determinado. Se ha seleccionado esta pantalla para realizar esta primera consulta ya que es el punto a partir del cual se llega a las demás pantallas que muestran información sobre un medicamento.

En ciertas pantallas, con la aplicación de esta técnica no se obtenía ningún beneficio, por lo que se ha tenido que realizar una nueva consulta a la base de datos. Un ejemplo de estas pantallas es "Vademecum". Al tratarse de la única pantalla en la cual se muestra información detallada de un medicamento, es innecesario trasmitir esta información de pantalla en pantalla y una mejor opción, es realizar una nueva consulta específica para esta pantalla.

# 5 Integración, pruebas y resultados

Para comprobar que todo el trabajo se ha realizado correctamente y que la aplicación cumple con todos los requisitos planteados, se han llevado a cabo una serie de pruebas que permiten confirmar este objetivo. Además, para la selección correcta del algoritmo de identificación de imágenes a integrar en la aplicación final, se han realizado una serie de pruebas que permite compararlos entre sí y, de esta forma, confirmar que la elección del algoritmo es la adecuada.

## *5.1 Pruebas sobre la interfaz de usuario*

Al tratarse de una aplicación que debe cumplir una serie de requisitos (pocas pantallas, botones de gran tamaño, contraste de colores…), se ha prestado bastante atención a esta parte del proyecto.

Los objetivos de estas pruebas son comprobar el correcto funcionamiento y validar la aplicación. Se han ejecutado numerosas veces a lo largo de la vida del proyecto, ya que siempre que se agrega una pantalla, se debe comprobar que la integración se ha realizado de forma correcta y que las transiciones entre ésta y las demás pantallas funcionan de la forma adecuada.

Las pruebas que se han realizado son las siguientes:

- Pruebas sobre la navegación entre pantallas. Para ello, se ha realizado una navegación por la aplicación, de tal forma que se puedan cubrir todas las transiciones entre las pantallas y así poder comprobar que las pantallas mostradas en todas las situaciones eran las correctas. Las posibles transiciones entre pantallas han sido explicadas anteriormente en el apartado 3.7.
- Pruebas sobre el flujo de información entre pantallas. Como se ha explicado anteriormente en el apartado 4.3, es necesario comprobar que la información que se pasa de una pantalla a otra es correcta.
- Pruebas individuales de las pantallas de la aplicación. Se ha comprobado que todos aquellos elementos que componen la interfaz de cada pantalla funcionan correctamente.
- Pruebas sobre la dependencia de información entre pantallas. Consiste en comprobar que la información que se muestra en cada pantalla es correcta. Por ejemplo, al modificar alguna propiedad en la pantalla “Ajustes”, comprobar que todas aquellas pantallas que se ven afectadas por esta modificación, funcionan correctamente. A modo de ejemplo, si en la pantalla de ajustes se selecciona orden alfabético como forma de ordenación de los medicamentos, la pantalla “Lista de medicamentos” debe mostrar los medicamentos en dicho orden.

## *5.2 Pruebas de usabilidad*

Estas pruebas consisten en seleccionar a un grupo de usuarios y solicitarles que realicen una serie de tareas necesarias para comprender la forma en que se debe utilizar la aplicación (registro de un medicamento, confirmación de una toma, modificar algún ajuste…). Una vez que el usuario haya realizado las tareas solicitadas, se le pide que rellene un formulario gracias al cual, se obtiene información relevante sobre aquellos apartados que se deben modificar para mejorar la aplicación.

Estas pruebas se han realizado una vez se disponía de una versión avanzada de la aplicación. Con los resultados obtenidos de estas pruebas, se puede mejorar la aplicación hasta llegar a un punto en el cual los usuarios puedan utilizar sin ninguna complicación la aplicación.

Con estas pruebas, se intenta abarcar un amplio grupo de usuarios que resulte representativo de los diferentes sectores en los cuales se divide la población. Es decir, se buscará personas con diferentes edades. De esta forma, se obtiene información representativa sobre la aplicación por parte de todos los posibles usuarios. Estas pruebas con diferentes rangos de edades son necesarias, ya que es posible que usuarios jóvenes no encuentren dificultades en cuanto al uso de la aplicación, sin embargo, a una persona de avanzada edad o que no se encuentre familiarizado con la tecnología de los *Smartphone*, le puede resultar más difícil la utilización de la aplicación.

El formulario que deben completar los usuarios al terminar de realizar la tarea que se les haya solicitado se encuentra en el Anexo F.

Este cuestionario permite calcular el *System Usability Scale* (SUS) [27]. Una vez el usuario haya completado el formulario, se calcula una puntuación (varia dentro del rango 0 a 100) que indica la usabilidad del sistema, en este caso, la aplicación. Para obtener una puntuación que sea representativa del grado de usabilidad de la aplicación, es necesario un gran número de usuarios, por lo que no se ha calculado dicho valor. En su lugar, se ha pedido a un pequeño grupo de usuarios que realicen la tarea de registrar un medicamento (una de las principales funciones de la aplicación) y se han recopilado todas las sugerencias que han mencionado los usuarios. En la Tabla 5-1 se recogen las respuestas de estos usuarios y se calcula el valor del SUS [27].

| **Preguntas formulario** | **Usuario 1ª** | **Usuario 2ª** | **Usuario 3ª** |
|---|---|---|---|
| Edad | **57** | **52** | **42** |
| Pienso que me gustaría utilizar este sistema con frecuencia | 4 | 5 | 4 |
| Me pareció que el sistema es innecesariamente complejo | 1 | 2 | 1 |
| Me pareció que el sistema era fácil de usar | 4 | 4 | 5 |
| Creo que necesitaría el apoyo de un técnico para poder utilizar este sistema | 2 | 2 | 1 |
| Me pareció que las diversas funciones de este sistema estaban bien integradas | 4 | 5 | 4 |
| Pensé que había demasiada inconsistencia en este sistema | 1 | 1 | 1 |
| Me imagino que la mayoría de la gente debería aprender a utilizar este sistema muy rápidamente | 4 | 4 | 4 |
| Me pareció que el sistema es muy complicado de usar | 1 | 3 | 1 |
| Me sentí seguro utilizando el sistema | 4 | 4 | 4 |
| Tuve que aprender muchas cosas antes de que pudiera utilizar este sistema | 1 | 1 | 2 |
| **Puntuación (SUS)** | **85** | **82.5** | **87.5** |

**Tabla 5-1: Respuestas de los usuarios al formulario y calculo del *System Usability Scale* (SUS)**

Tras realizar esta prueba y analizar las respuestas marcadas en los formularios, se ha llegado a las siguientes conclusiones:

- A pesar de los esfuerzos invertidos en el diseño de la interfaz y la apuesta por proporcionar una interacción más cómoda mediante el uso de técnicas basadas en

reconocimiento de imágenes, es interesante destacar que la aplicación puede resultar algo compleja para personas de avanzada edad.

Por este motivo, se debe agregar una nueva funcionalidad a la aplicación. Además de la sección de tutorial del menú desplegable, si es la primera vez que se inicia la aplicación se debería mostrar al usuario una navegación por toda la aplicación y sus funciones. De esta forma, usuarios que no estén familiarizados con los aspectos del sistema operativo Android, podrán utilizar la aplicación sin ninguna complicación.

- Se debe mejorar el proceso de captura de imagen para reconocer el medicamento. Al registrar un medicamento, se pide al usuario que tome una foto de la caja del medicamento. Tras este paso, la aplicación abre una pantalla para recortar la imagen y eliminar así el resto de elementos que no sean la caja del medicamento. Esta transición tarda lo suficiente como para que el usuario se pregunte si la foto se ha realizado correctamente. Para solucionar esto, bastaría con agilizar este proceso de captura y recorte de la imagen o simplemente indicar al usuario con un icono que espere un instante.

- Los usuarios no tienen muy claro como confirmar el registro del medicamento. Se debe incorporar un botón que sea más llamativo.

En futuras versiones de la aplicación, se corregirán dichas sugerencias y se procederá a realizar las mismas pruebas para confirmar que las modificaciones se han realizado correctamente.

## *5.3 Pruebas de los diferentes algoritmos para la identificación de imágenes*

En la sección 4 de "Desarrollo" se han presentado los algoritmos que han sido investigados para la posterior integración con la aplicación. Se han visto los resultados obtenidos a partir de un caso de prueba y cuál de todos aquellos era el más adecuado para la integración con la aplicación.
Todos estos algoritmos poseen una serie de características que se pueden modificar con el objetivo de mejorar los resultados obtenidos.

Para estas pruebas, se han empleado las mismas imágenes que en el caso de prueba de la sección 4 ("Desarrollo") pero modificando algunos parámetros de los algoritmos. Además, para realizar unas pruebas más exhaustivas sobre los algoritmos, se ha decidido realizar un pre procesamiento a las imágenes convirtiéndolas a escala de grises (en los casos en los que era posible). El objetivo de esta decisión es evitar que el algoritmo detecte características comunes que dependan de luminosidad o colores de las imágenes.

Esta técnica solo es aplicable sobre aquellos algoritmos cuyo funcionamiento no se base en los colores de las imágenes. En el caso de los histogramas, no es conveniente emplear esta técnica ya que los resultados obtenidos tras la ejecución del algoritmo, podrían no ser los deseados.

Para realizar todas las pruebas de los algoritmos, se ha diseñado un programa que permita extraer información relevante sobre cada uno de los algoritmos. El programa se encarga de

mostrar todos los resultados obtenidos tras variar estas características para cada uno de los algoritmos a probar.

Una vez obtenidos los resultados para un algoritmo, se puede analizar cuál es el valor adecuado para cada una de las características de cada uno de los algoritmos.

Para automatizar estas pruebas, se ha diseñado un programa cuya estructura se muestra en la Figura 5-1. En este diagrama se incluyen solo los métodos y atributos relevantes para la explicación.

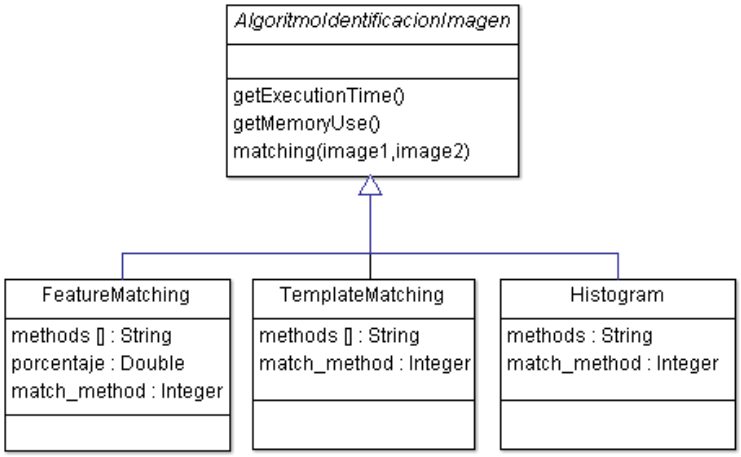



**Figura 5-1 : Estructura del programa diseñado para realizar las pruebas**

Como se puede observar, el programa consiste en una clase abstracta "AlgoritmoIdentificacionImagen", que contiene métodos para calcular el uso de memoria, tiempo de ejecución y el propio método de comparación de imágenes.

Se ha diseñado de esta forma, ya que todos los algoritmos van a recibir dos imágenes que son las que deben comparar entre sí. Además, se agregan los métodos de cálculo del tiempo y del uso de memoria en el caso de que sea necesario para una comparación más detallada. De esta forma, siempre que se desee agregar otro algoritmo para la investigación, basta con crear una clase que lo represente y programar la herencia correspondiente.

Todos los algoritmos que se han investigado, se representan mediante sus respectivas clases que heredan de la anteriormente mencionada. Además, estas clases incluyen una serie de atributos necesarios para cada uno de estos algoritmos:

- *Methods*: Consiste en una serie de cadenas de texto que permiten especificar los diferentes métodos que tiene un algoritmo determinado para comparar dos imágenes.
- *Match_method*: Atributo que permite recorrer cada uno de los métodos del algoritmo. De esta forma, se comprueban diferentes modos de uso del algoritmo.
- *Porcentaje*: Atributo que solo se encuentra disponible en un algoritmo, ya que es el que se integrará en la aplicación. La aplicación debe obtener un porcentaje de similitud para comprobar que tan similares son ambas imágenes y, de esta forma, continuar o no con el proceso encargado de la confirmación de una toma de un medicamento. Además, dado los resultados de los otros algoritmos, es difícil

extraer una característica que se pueda expresar en un porcentaje y que sea representativa del resultado obtenido de la ejecución del algoritmo.

Los resultados obtenidos de la ejecución de cada algoritmo se muestran por pantalla. A pesar de que la forma de representar los resultados de cada algoritmo sean diferentes, todos ellos devuelven una o varias imágenes en la que se pueden ver dichos resultados.

Una vez comprendida la estructura del programa, se procede a ejecutar todos los algoritmos investigados y a analizar los resultados obtenidos.

### 5.3.1 Pruebas con el algoritmo *Template Matching*

Este algoritmo posee 6 posibles métodos para encontrar las similitudes entre las imágenes. Los resultados obtenidos tras modificar este parámetro se pueden encontrar en el Anexo C.

Tras analizar los resultados, se llega a la conclusión de que utilizando diferentes métodos de *matching*, se llega a resultados similares. A pesar de que los resultados sean iguales en la mayoría de los casos, otro aspecto a tener en cuenta es el tiempo de ejecución consumido para obtener dicho resultado.

Para obtener una distinción más detallada, se procede a calcular los tiempos de ejecución, representados en la Tabla 5-2:

| **Método** | **Tiempo de ejecución (milisegundos)** |
|---|---|
| Imgporc.TM_SQDIFF | 1588 |
| Imgporc.TM_SQDIFF_NORMED | 1580 |
| Imgporc.TM_CCORR | 1393 |
| Imgporc.TM_CCORR_NORMED | 1589 |
| Imgporc.TM_CCOEFF | 1501 |
| Imgporc.TM_CCOEFF_NORMED | 1611 |

**Tabla 5-2: Comparación de los diferentes métodos del algoritmo Template Matching**

Como se puede observar, los tiempos de ejecución son diferentes dependiendo del método utilizado. Con vistas a los resultados mostrados en el Anexo C, los métodos a valorar son "TM_SQDIFF", "TM_CCOEFF" y "TM_CCOEFF_NORMED".
Dado que los tiempos entre estos métodos no son muy diferentes, se opta por utilizar el método "TM_CCOEFF_NORMED", ya que es una versión normalizada y se han obtenido buenos resultados tanto de la versión sin normalizar como de la normalizada. A diferencia del método "TM_SQDIFF", cuya versión normalizada no aporta los resultados deseados.

### 5.3.2 Pruebas con el algoritmo *Histogram Comparison*

Para comparar dos imágenes, extraemos de cada una de ellas su histograma, diferenciando los canales RGB que la constituyen.

Para comparar el grado de similitud de dos histogramas, se utilizan una serie de métricas. Podemos diferenciar 4 tipos de métricas. Cada una de ellas posee un rango en el cual se muestra el grado de similitud máximo y mínimo. Gracias a esto, podemos saber el grado de similitud de nuestras imágenes dependiendo de su posición dentro de estos rangos.
Para las métricas Imgproc.CV_COMP_CORREL y Imgproc.CV_COMP_INTERSECT, un valor alto indica una mayor similitud. En cambio, en las métricas

Imgproc.CV_COMP_CHISQR y Imgproc.CV_COMP_BHATTACHARYYA, un valor bajo indica una mayor similitud.

En la Tabla 5-3, se muestran los resultados obtenidos de emplear las diferentes métricas:

| Métrica | Rango de similitud | Histograma Azul | Histograma Verde | Histograma Rojo |
|---|---|---|---|---|
| **Imgproc.CV_COMP_CORREL** | [-1,1] | 0.45 | 0.54 | 0.67 |
| **Imgproc.CV_COMP_CHISQR** | [0, unbounded] | 17266.78 | 2256.81 | 2651.27 |
| **Imgproc.CV_COMP_BHATTACHARYYA** | [0,1] | 0.60 | 0.44 | 0.41 |
| **Imgproc.CV_COMP_INTERSECT** | [0, unbounded] | 537.69 | 659.58 | 1545.64 |

**Tabla 5-3: Comparación de las métricas del algoritmo *Histogram Comparison***

Ambas imágenes poseen un grado de similitud alto, ya que una incluye a la otra. El único aspecto que varía es el entorno en el que se encuentra la caja del medicamento. Teniendo esto en cuenta, debemos fijarnos en aquellas métricas cuyos resultados tengan un alto grado de similitud.

Todas las métricas poseen un tiempo de ejecución similar, por lo que se deben analizar principalmente los resultados de la comparación de los diferentes canales de colores.

Tras analizar los resultados, se ha comprobado que tanto la métrica "CV_COMP_CORREL" como "CV_COMP_BHATTACHARYYA" son las mejores opciones ya que los resultados obtenidos se aproximan a los deseados.

### 5.3.3 Pruebas con el algoritmo *Feature Matching*

Como se ha explicado anteriormente en la sección 4.1.3, en este algoritmo se diferencian dos elementos importantes, los *matchers* y los detectores de características. En estas pruebas, se estudiarán los 3 detectores de características más utilizados (SURF, ORB y SIFT) y dos algoritmos comparadores de los descriptores (BRUTEFORCE_HAMMING, BRUTEFORCE_SL2).

Se ha decidido utilizar estos algoritmos comparadores en las pruebas ya que son los algoritmos que se recomiendan en la página de la librería OPENCV. Se recomienda que para los algoritmos SIFT y SURF se utilice BRUTEFORCE_SL2 y para ORB se utilice BRUTEFORCE_HAMMING.

Una vez se tenga en claro los parámetros que se van a utilizar en las pruebas, hay que mencionar que se va a utilizar una característica más. Esta característica es una cota y no es un parámetro que se pueda modificar en los algoritmos, se ha agregado al programa para intentar obtener unos mejores resultados en las pruebas. El objetivo de incluir esta característica en el programa es para poder obtener un porcentaje de similitud entre ambas imágenes. De esta forma, se pueden diferenciar medicamentos que sean similares ya que los porcentajes obtenidos serán diferentes.

A la hora de integrar el algoritmo *Feature Matching* [9] con la aplicación, además, se podrá indicar un porcentaje de similitud mínimo que se deba superar para que dos imágenes sean consideradas iguales o similares.

Anteriormente se ha explicado que dos descriptores son similares cuanto más próximos sean (empleando un cálculo de distancias). Esta característica consiste en establecer una cota que actúa a modo de filtro. Empleando esta cota, se consigue eliminar todas aquellas distancias que no aporten información relevante para la comparación de las imágenes. Al comparar los descriptores de dos imágenes (una vez calculadas las distancias), se puede establecer un mínimo de distancia en la cual se considera que dicha comparación es buena. Es decir, si esta cota se establece con un elevado valor, la mayoría de distancias calculadas entre los descriptores de ambas imágenes, serán tomadas como válidas y relevantes, a pesar de que puedan resultar erróneas o ser características no importantes de las imágenes (se podrían considerar como ruido). En cambio, si se establece la cota con un valor muy bajo, se encontrarán pocas distancias que se encuentren dentro del rango válido, por lo que la similitud entre ambas imágenes resultará errónea.

A continuación, se analizan los resultados obtenidos para cada uno de los algoritmos de detección de características SURF (Figura 5-4), ORB (Figura 5-2) y SIFT (Figura 5-3).

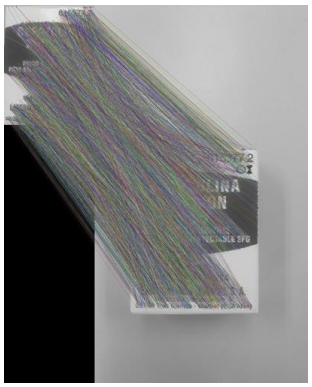


**Figura 5-4: Resultado de aplicar el Algoritmo Feature Matching con SURF como método de deteccion de características**

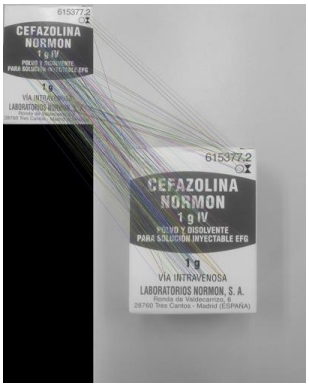



**Figura 5-2: Resultado de aplicar el Algoritmo Feature Matching con ORB como método de detección de características**

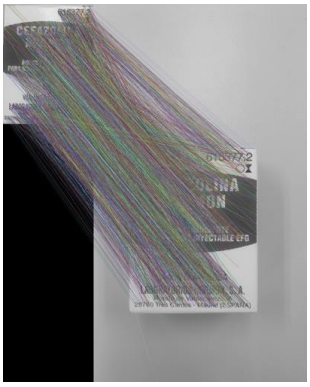


**Figura 5-3: Resultado de aplicar el Algoritmo Feature Matching con SIFT como método de deteccion de características**

Como podemos observar, los algoritmos SURF y SIFT encuentran muchas más coincidencias (7993 y 5998 respectivamente) que ORB (500). A pesar de que ORB es mucho más rápido que los otros algoritmos, es preferible emplear un poco más de tiempo de ejecución para obtener unos mejores resultados finales. Por estos motivos, se descarta la utilización del algoritmo ORB y se continúan probando los otros algoritmos.

Tras comprobar la efectividad de cada uno de los algoritmos con este caso de prueba, se procede a establecer la cota para obtener un valor del porcentaje de similitud.

Para ello, se variará de forma automatizada el valor de la cota y, de esta forma, se obtendrá para dichos valores de la cota, un número de comparaciones relevantes o key points (eliminando el posible ruido). Con el número de comparaciones realizadas en total y la cantidad de key points, se puede obtener la relación de porcentaje entre ambas, que representará el grado de similitud entre las imágenes.

Los rangos en los cuales establecer el valor de la cota varían para cada algoritmo. Por esta razón, se han realizado varias pruebas (modificando dichos rangos) hasta obtener un rango

que permita calcular un porcentaje de similitud que se adecúe a las imágenes con las que van a trabajar los algoritmos. Estos valores son dentro del rango de 0 hasta la distancia máxima encontrada entre los descriptores de las imágenes para SIFT. Para SURF el rango es de 0 a 1.

Teniendo en cuenta las imágenes del caso de prueba, se busca un porcentaje de similitud alto al estar incluida una imagen en la otra. Es decir, buscamos una cota en la que el porcentaje de similitud varíe entre el rango del 80% al 90%. Para incrementar la efectividad de esta cota, se debe realizar la misma prueba para una gran cantidad de imágenes, de tal forma, que se llegue a una cota que permita encontrar los porcentajes adecuados para cualquier imagen.

En las figuras Figura 5-5 y Figura 5-6 se observará como varía la cantidad de key points dependiendo del valor de la cota para cada uno de los algoritmos de detección de características. En el caso de SIFT (Figura 5-5), se incrementará la cota con la distancia mínima encontrada entre los descriptores de las imágenes. En el caso de SURF (Figura 5-6), el incremento será de 0.01 unidades. Utilizando estos incrementos, podemos representar en una gráfica todos los valores obtenidos y analizar los resultados. Así se tiene:

- SIFT: Total de key points 5998
- SURF: Total de key points 7993

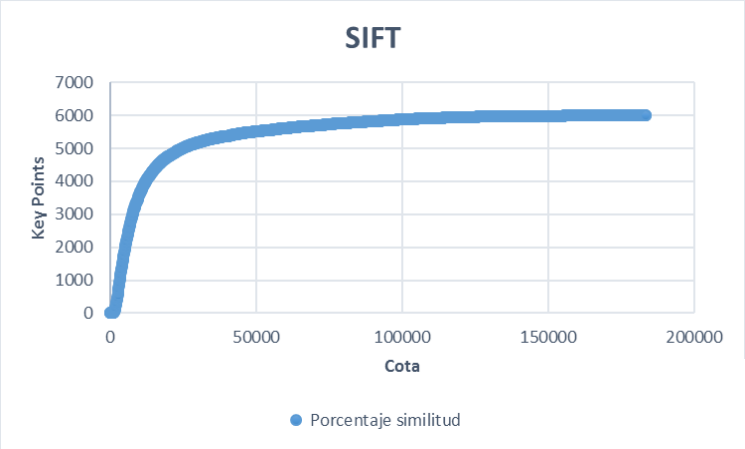



**Figura 5-5: Resultados tras variar la cota para el algoritmo SIFT**

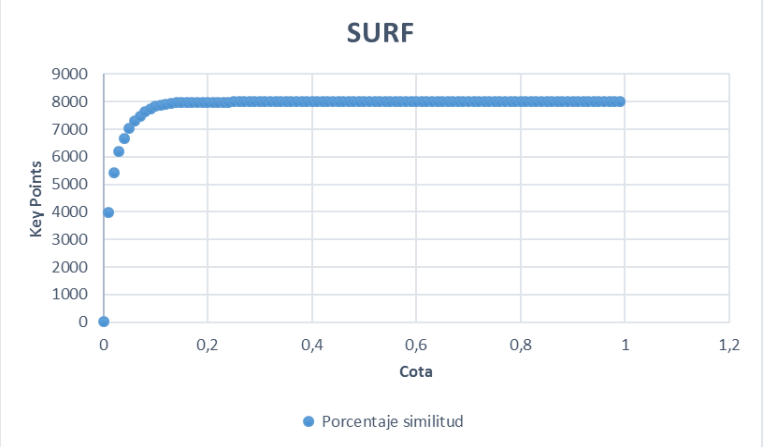



**Figura 5-6: Resultados de variar la cota para el algoritmo SURF**

Como se puede observar en las figuras, a medida que aumenta la cota, la cantidad de key points también aumenta. Si se continúa incrementando la cota, se llega hasta el punto en el

cual, dado que el valor de la cota es tan alto, se considera que ningún key point es ruido y el total de key points encontrados en la imagen es relevante (en porcentaje el 100% de los key points).

Una vez comprobado cómo varía la cantidad de key points, debemos establecer el valor que se utilizará como cota a la hora de comparar dos imágenes. Como se ha mencionado anteriormente, se busca un valor de cota con el cual se obtenga un porcentaje de similitud comprendido en el rango de 80% a 90%.

Para seleccionar el valor de la cota, se ejecuta el mismo algoritmo en dos situaciones diferentes. En una de ellas, la caja del medicamento se encuentra dentro de la otra imagen y ambas cajas son idénticas. En la otra situación, se realiza el mismo procedimiento con una caja bastante similar, pero diferente (Figura 4-7).

El objetivo principal de estas pruebas es obtener un valor de cota que permita diferenciar el porcentaje de similitud entre estas dos cajas de medicamentos similares. Se busca una cota con la que se obtenga un porcentaje de similitud más alto al utilizar la caja de medicamento roja (Figura 4-1) frente a la caja azul (Figura 4-7). Esto es debido a que la caja roja aparece en la imagen y la azul, a pesar de que pueda tener características muy similares, no se encuentra en la imagen.

Empleando los porcentajes de similitud, se podrá diferenciar entre cajas de medicamento muy similares entre sí. De esta forma, se resuelve el problema planteado en la sección 4.2.3

A continuación, se detallan los resultados de las pruebas:

- SIFT: Analizando los resultados obtenidos se observa que, para un valor de cota de 49100 unidades, se obtienen unos porcentajes de similitud de 91.81% y 70.83% al utilizar la imagen de la caja roja y la caja azul respectivamente.

- SURF: Tras analizar los resultados obtenidos, se observa que para un valor de cota de 0.06, los porcentajes de similitud obtenidos son 91.44% y 88.5% al utilizar la imagen de la caja roja y la caja azul respectivamente.

Tras estas comparaciones se puede ver que, para un mismo valor de la cota, es conveniente utilizar el algoritmo SIFT ya que la diferencia entre los porcentajes de similitud de ambas cajas es mayor que empleando el algoritmo SURF. Es decir, el algoritmo SIFT para un valor de cota determinado, diferencia mucho mejor cada una de las características de las imágenes empleadas.

Para comprobar que la cota establecida es la correcta y el algoritmo SIFT funciona adecuadamente, solo falta estudiar el caso en el cual se busca una caja de medicamento totalmente diferente (Figura 5-7) a las utilizadas en las otras pruebas.


**Figura 5-7: Caja del medicamento Augmentine**

Tras configurar el algoritmo SIFT con los parámetros anteriormente estudiados, se obtiene el resultado mostrado en la Figura 5-8, con 602 Key Points, una cota de 49100 y un porcentaje de similitud de 35.54 %. Es decir, el resultado parece el deseado (35.54%) ya que estas cajas no tenían muchas características en común.

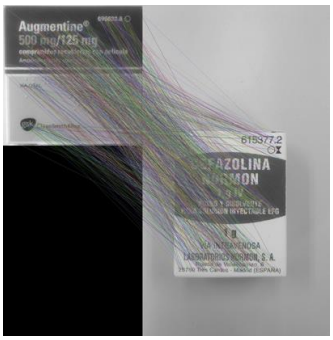


**Figura 5-8: Resultado de aplicar el Algoritmo Feature Matching con SIFT como método de detección de características**

Como se ha mencionado anteriormente, para seleccionar el valor de cota idóneo habría que repetir estas pruebas con una mayor cantidad de cajas de medicamentos. A pesar de que estas pruebas no se han realizado con una gran cantidad de cajas de medicamentos, se puede tomar estos parámetros de configuración del algoritmo como punto de partida para futuros experimentos.

Como conclusión, queda demostrado que el algoritmo SIFT es el más adecuado para la integración con la aplicación de todos aquellos algoritmos probados para la identificación de imágenes.

# 6 Conclusiones y trabajo futuro

En este apartado se muestran las conclusiones finales de todo el trabajo realizado, desde un punto de vista técnico y otro personal. Además, se incluye un apartado que trata sobre las posibles mejoras o trabajo futuro relacionado con este proyecto.

## *6.1 Conclusiones*

Una vez finalizado el proyecto, se deben mencionar las diferentes conclusiones a las que se ha llegado con la elaboración de este proyecto.

### 6.1.1 Cumplimiento de objetivos

Durante el desarrollo del TFG, se buscaba cumplir una serie de objetivos determinados. Una vez finalizado el TFG, se puede concluir que los objetivos planteados al inicio del proyecto y mencionados en la sección 1.2 se han cumplido satisfactoriamente, obteniéndose como resultado la implementación de una aplicación que permite realizar el seguimiento de los medicamentos de un usuario mediante una interacción basada en el reconocimiento de imágenes. Para ello se ha hecho uso de algoritmos que permiten reconocer e identificar imágenes, se han realizado pruebas sobre los diferentes algoritmos con el objetivo de integrar en la aplicación aquel que aporte unos mejores resultados y se ha diseñado una base de datos que permite gestionar todos los medicamentos que necesita el usuario.

### 6.1.2 Conclusiones personales

Como experiencia personal, el estudiante se ha dado cuenta de sus aptitudes para:

- Trabajar en equipo y tomar decisiones de forma conjunta.
- Realizar una investigación sobre algoritmos que permitan realizar una actividad determinada con la que no se ha trabajado nunca (identificación de imágenes) y cumplir con unos objetivos marcados.
- Aprender la programación de diferentes elementos del sistema operativo Android con los que no se ha trabajado anteriormente.
- Utilizar diferentes herramientas que facilitan el desarrollo de un proyecto en todos los niveles posibles: planificación, implementación, etc.
- Plantear diferentes alternativas para poder realizar una tarea determinada, de tal forma que no es necesaria la detención del proyecto para tratar con los posibles problemas que puedan surgir.
- Rápida resolución de problemas que puedan surgir durante el desarrollo de un proyecto.

La realización de este trabajo ha sido posible gracias a la formación que ha recibido el estudiante a lo largo de toda la carrera de Grado en Ingeniería Informática, implantada en la Escuela Politécnica Superior de la Universidad Autónoma de Madrid.

## *6.2 Trabajo futuro*

Al tratarse de un proyecto cuyo objetivo es desarrollar una aplicación móvil, éste debe estar dispuesto a incorporar una serie de mejoras o ampliaciones con el fin de obtener en

cada revisión, un producto mejor elaborado y que cumpla las necesidades de los usuarios en todo momento.

A continuación, se exponen una serie de mejoras que se podrían llevar a cabo en la aplicación:

- Incorporar funcionalidad de autocompletado cuando el usuario está modificando o registrando el nombre de un medicamento. De esta forma, si un usuario desea establecer el nombre de un medicamento, pero no recuerda perfectamente cómo se llamaba, dispone de una ayuda para encontrar el nombre buscado.
- Implementar la funcionalidad necesaria para que se permita guardar toda la configuración y medicamentos registrados de la aplicación en un servidor o en la nube. Cuando un usuario cambie de dispositivo, no será necesario que vuelva a registrar todos los medicamentos, en su lugar, importará todos los datos de la aplicación previamente almacenados en la nube o en el servidor.
- Mejorar y permitir la portabilidad de la aplicación, para poder ser utilizada en cualquier dispositivo cuyo sistema operativo sea Android.
- Incluir la opción de cambiar el idioma. Para permitir que personas del extranjero puedan utilizar la aplicación sin que el idioma sea un problema.
- Incluir una sección de estadísticas en la cual, con la utilización de gráficas o cualquier otro elemento gráfico, se permita realizar un seguimiento al usuario. Esta funcionalidad, por ejemplo, podría ser empleada por el médico del usuario permitiéndole comprobar que el usuario está siguiendo correctamente el tratamiento planteado.
- Implementar una mejora en el sistema de notificaciones de tal forma que, si un usuario no ve la notificación y se salta la toma de un medicamento, la aplicación le envíe continuamente alertas (vibración o sonido). Además, el intervalo de tiempo en el cual se envían estas alertas podría ser configurado por el usuario (para poder adaptar la aplicación a cada usuario). Con esta mejora, se evita que un usuario pueda olvidar la toma de un medicamento que pueda ser de gran importancia para su salud.

# Referencias


[1] Google Imágenes, [En línea]. Available: https://www.google.es/imghp.

[2] IDC (International Data Corporation), [En línea]. Available: http://www.idc.com/prodserv/smartphone-os-market-share.jsp.

[3] Android Developers, [En línea]. Available: https://developer.android.com/about/dashboards/index.html.

[4] Material Design, [En línea]. Available: https://material.google.com/.

[5] Google Goggles, [En línea]. Available: https://play.google.com/store/apps/details?id=com.google.android.apps.unveil&hl=es.

[6] Zalando, [En línea]. Available: https://play.google.com/store/apps/details?id=de.zalando.mobile.

[7] PlantNet identificación planta, [En línea]. Available: https://play.google.com/store/apps/details?id=org.plantnet.

[8] OpenCV-TemplateMatching, [En línea]. Available: http://docs.opencv.org/2.4/doc/tutorials/imgproc/histograms/template_matching/template_matching.html.

[9] OpenCV-FeatureMatching, [En línea]. Available: docs.opencv.org/master/dc/dc3/tutorial_py_matcher.html.

[10] OpenCV-HistogramComparison, [En línea]. Available: http://docs.opencv.org/2.4/doc/tutorials/imgproc/histograms/histogram_comparison/histogram_comparison.html.

[11] Wikipedia OCR, [En línea]. Available: https://en.wikipedia.org/wiki/Optical_character_recognition.

[12] Tesseract-OCR, [En línea]. Available: https://github.com/tesseract-ocr/tesseract.

[13] OpenCV, [En línea]. Available: http://opencv.org/.

[14] Zbar, [En línea]. Available: http://zbar.sourceforge.net/.

[15] Zxing, [En línea]. Available: https://github.com/zxing/zxing/wiki/Getting-Started-Developing.

[16] Asprise, [En línea]. Available: https://asprise.com/royalty-free-library/java-ocr-api-overview.html.

[17] Vademecum, [En línea]. Available: http://www.vademecum.es/.

[18] Android Studio, [En línea]. Available: https://developer.android.com/studio/index.html.

[19] Eclipse, [En línea]. Available: https://eclipse.org/.

[20] Git, [En línea]. Available: https://git-scm.com/.

[21] Bitbucket, [En línea]. Available: https://bitbucket.org/.

[22] Trello, [En línea]. Available: https://trello.com/.

[23] SQLite, [En línea]. Available: https://www.sqlite.org/.

[24] ArgoUML, [En línea]. Available: http://argouml.tigris.org/.

[25] Modelio, [En línea]. Available: https://www.modelio.org/.

[26] Moqups, [En línea]. Available: https://moqups.com/.

[27] System Usability Scale, [En línea]. Available: https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html.

Todas las referencias en línea fueron visitadas por última vez el 28-6-2016

# Glosario

| | |
|---|---|
| App | Aplicación informática diseñada para ser ejecutada en teléfonos inteligentes (Smartphones), tabletas y otros dispositivos móviles. |
| Códigos QR | Matriz en dos dimensiones formada por una serie de cuadrados negros sobre fondo blanco. Empleando un lector específico, muestra en la pantalla del dispositivo móvil una información determinada. Por ejemplo, una página web, correo electrónico, etc. |
| Feature Matching | Técnica que extrae las características más importantes de dos imagenes con el objetivo de compararlas entre sí y afirmar si ambas imágenes son o no similares |
| Histogram Comparison | Técnica que se emplea para comparar dos imágenes a través de los histogramas que las representan |
| Librería | Conjunto de implementaciones funcionales, codificadas en un lenguaje de programación, que ofrece una interfaz bien definida para la funcionalidad que se invoca. |
| OCR | Reconocimiento Óptico de Caracteres |
| Smartphone | Tipo de teléfono móvil caracterizado por tener una mayor capacidad de almacenar datos y realizer actividades. |
| Template Matching | Es una técnica en procesamiento digital de imágenes para la búsqueda de pequeñas partes de una imagen que coincidan con una imagen a modo de plantilla o patrón |
| Usabilidad | Facilidad con que las personas pueden utilizar una herramienta determinada con el fin de alcanzar un objetivo concreto. |

# Anexos

## A Manual de instalación

Actualmente, la aplicación se encuentra en fase de pruebas y, por este motivo, no se ha decidido publicar en la tienda de Google para aplicaciones Android llamada Play Store.
En su lugar, la aplicación puede ser descargada a través de la siguiente dirección: https://play.google.com/apps/testing/es.uam.eps.medetectApp

Los únicos requisitos para poder descargar la aplicación, es acceder a la dirección anteriormente mencionada con una cuenta de Gmail.

Una vez se acceda a dicha dirección, se podrá consultar más información sobre la aplicación.

# B Manual del programador

Para comenzar a utilizar la aplicación Medetect, es necesario disponer de los siguientes elementos:

1. Android-Studio
2. URL del repositorio donde se ubica el proyecto Medetect-App

Pasos a seguir para una instalación limpia (desde cero):

1. Acceder a la página web: http://developer.android.com/intl/es/sdk/index.html
2. Seleccionar el sistema operativo correspondiente. En el caso de que se instale en un sistema operativo que no sea Windows, se debe seleccionar la opción "Other Download Options" y elegir el sistema operativo deseado.
3. Descargar la última versión indicada de Android Studio
4. Seguir los pasos indicados en la web para configurar correctamente Android Studio. En caso de que la instalación no sea en Windows, seleccionar la opción "Show Instructions for all platforms"
5. Tras la instalación, se deberá seleccionar la opción "Open an existing Android Studio project" que se encuentra en la pantalla inicial de Android Studio ("Welcome to Android Studio")
6. Seleccionar la carpeta que contiene el proyecto Medetect-App
7. Por último, lo único que hace falta es pasar a la vista de proyecto Android (situado en la parte superior del lateral izquierdo de la pantalla), conectar un dispositivo móvil por USB o utilizar un emulador y ejecutar el proyecto con la flecha de color verde de la barra de herramientas.

## C Anexo pruebas algoritmos

- Resultados obtenidos de variar el método de comparación de imágenes en el algoritmo *Template Matching*:

A. method=CV_TM_SQDIFF


B. method=CV_TM_SQDIFF_NORMED


C. method=CV_TM_CCORR

D. method=CV_TM_CCORR_NORMED

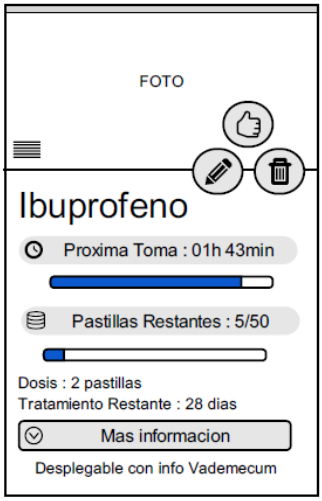

E. method=CV_TM_CCOEFF


F. method=CV_TM_CCOEFF_NORMED

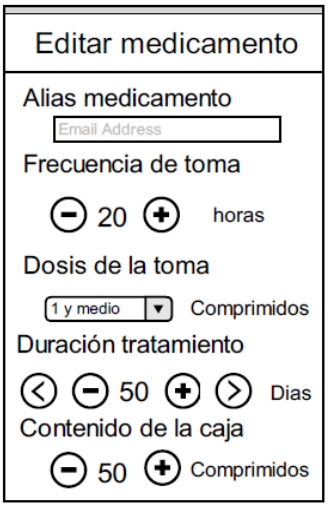
## *D Anexo maquetas de la aplicación*


**Menú incial**


**Pantalla por defecto de la aplicación (al abrir la aplicación teniendo como mínimo un medicamento)**



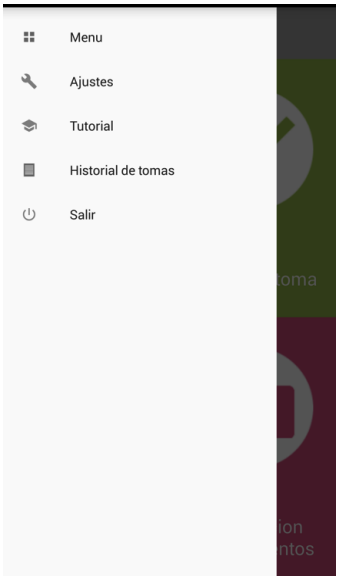

**Detalle de un medicamento**



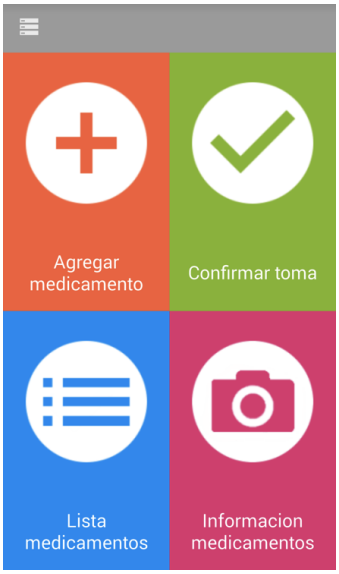

**Editar medicamento**


**Lista de medicamentos registrados**

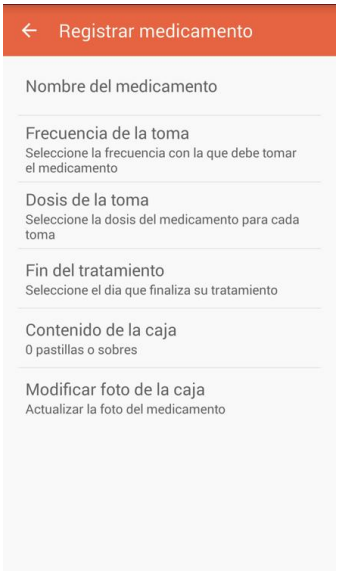
## E Anexo pantallas de la aplicación



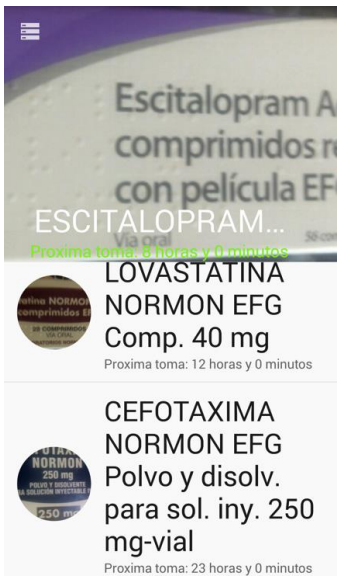

**Menú deslizante**



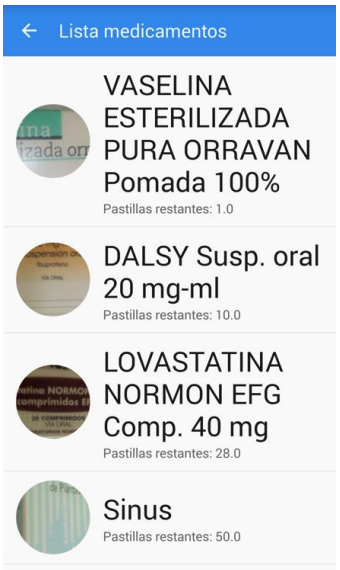

**Menú inicial**



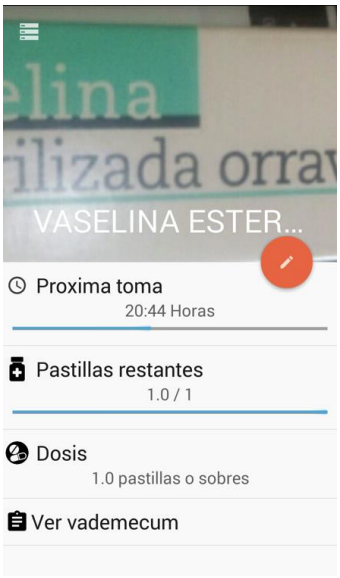

**Formulario de registro de un medicamento**



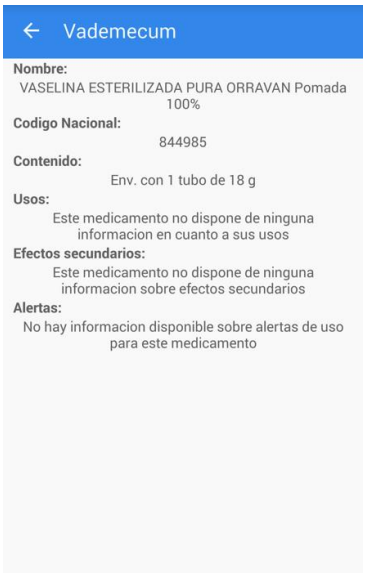

**Confirmar toma**



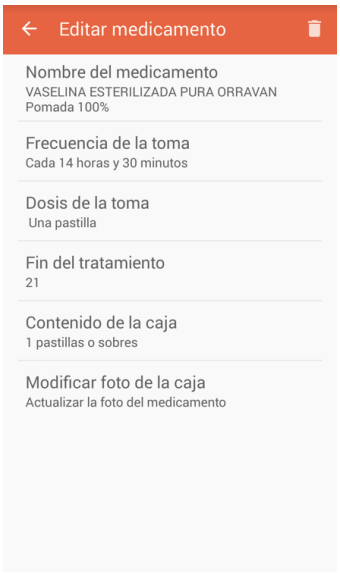

**Lista de medicamentos registrados**




**Detalle del medicamento**



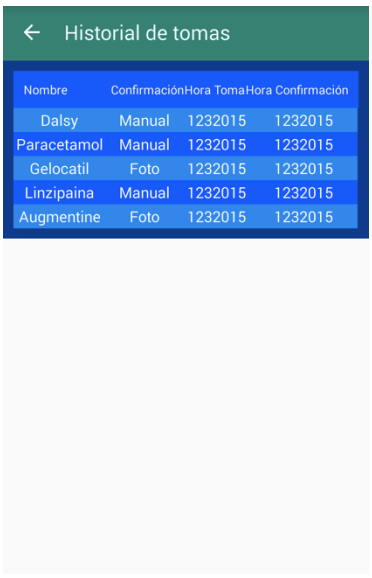

**Vademecum**




**Formulario para editar medicamento**


**Ajustes**


**Tutorial**




**Historial de tomas**

## F Anexo cuestionario SUS

Cuestionario para el cálculo del *System Usability Scale.*
Fuente: https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html

### *System Usability Scale*