



*Facultad
de
Ciencias*

Desarrollo de una aplicación móvil para la combinación de prendas de vestir
Mobile development application for clothing combination

Trabajo de Fin de Grado
para acceder al

GRADO EN INGENIERÍA INFORMÁTICA

Autor: Olaya Cilleruelo Cadelo

Director: Carlos Blanco Bueno
Septiembre - 2015

Resumen

La industria de la moda crece anualmente y nosotros con ella. El número de prendas en nuestros armarios aumenta sin límite, al igual que el tiempo que invertimos en buscar algo que nos guste antes de ir a trabajar cada día. Miramos el armario esperando que nos devuelva un conjunto perfecto, pero esa respuesta nunca llega. Nosotros somos los encargados de combinar las ropas y habitualmente no tenemos el tiempo ni las ganas necesarias para hacerlo. Todo ello desemboca en utilizar la misma indumentaria, variando los complementos, para conseguir un look adecuado y rápido.

Este proyecto se centra en el desarrollo de una aplicación móvil que permita generar conjuntos formados por las prendas de vestir del usuario. Dichas ropas se irán almacenando en la aplicación conforme el usuario realice fotografías de su fondo de armario.

El usuario podrá crear manualmente combinaciones, seleccionando las prendas que considere oportunas. Dichos conjuntos podrán ser vistos posteriormente, así como combinaciones basadas en colores análogos, categorías y combinaciones generadas aleatoriamente.

Palabras clave: *aplicación, smartphone, combinaciones, prendas, android.*

Abstract

Fashion industry is growing up every year and we grow with it. Clothes' numbers increases without limit in our closets, the same as the time we spend finding something we like before working each day. We look at the closet waiting for a perfect set, but we never find the right one. We are in charge of combining clothes and usually we don't have time or the will to. This situation leads us to use the same dress, changing accessories, to get an appropriate and quick look.

This project focuses on the development of a mobile application that allows us to generate sets made of user clothes. These clothes will be stored in the application as the user performs pictures of his wardrobe.

The user will be able to make manually combinations, selecting adequate clothes. These sets will be displayed later, as well combination based on analogous colors, categories and randomly generated.

Keywords: *application, smartphone, combinations, clothes, android.*

Índice

Resumen	2
Abstract	3
Índice de figuras	6
1. Introducción	7
1.1. Objetivos	7
2. Material y métodos utilizados	8
2.1. Metodología.....	8
2.2. Tecnología y herramientas	10
2.2.1 Android Studio.....	10
2.2.2. SQLite.....	12
2.2.3. Microsoft Project	12
2.2.4. Just In Mind	13
2.2.5. Magic Draw	13
3. Presentación del problema y análisis de requisitos	13
3.1 Requisitos funcionales	13
3.2. Requisitos no funcionales.....	14
3.3. Diagrama de casos de uso	15
3.4. Casos de uso detallados	15
4. Diseño e Implementación	24
4.1. Diseño arquitectónico.....	24
4.2. Diseño e implementación de la capa de datos	28
4.3. Diseño e implementación de la capa de negocio	29
4.3.1. Selección y captura de imágenes	29
4.3.2. Almacenamiento y creación de combinaciones.....	31
4.3.3. Combinaciones aleatorias	31
4.3.4. Prendas por color.....	32
4.3.5 Combinaciones por categorías.....	32
4.4. Diseño e implementación de la capa de presentación	33
5. Pruebas	40
5.1. Pruebas unitarias	40
5.2. Pruebas de integración	41
5.3. Pruebas de sistema.....	41
5.3.1. Pruebas de rendimiento.....	41
5.3.2. Pruebas de usabilidad.....	42

5.4. Pruebas de usuario	43
6. Conclusiones y trabajos futuros	44
6.1. Conclusiones personales	44
6.2. Trabajos futuros	45
7. Bibliografía	46

Índice de figuras

Ilustración 1-Progreso del proyecto en Microsoft Project	9
Ilustración 3- Ejemplo de código inteligente.....	11
Ilustración 4-Ejemplo de layout	12
Ilustración 9- Diagrama de casos de uso	15
Ilustración 10- Diseño arquitectónico en tres capas	25
Ilustración 11- Diagrama de componentes.....	26
Ilustración 12- Operaciones entre la capa de presentación y negocio	26
Ilustración 13-Operaciones entre la capa de negocio y datos	27
Ilustración 14-Diseño de la base de datos.....	28
Ilustración 15-Ejemplo de código en SQLite	29
Ilustración 16- implementación de la selección y captura de imágenes.....	29
Ilustración 17-Implementación del método OnActivityResult.....	30
Ilustración 18-Implementación del cálculo de colores.....	30
Ilustración 19-Implementación del almacenamiento de combinaciones	31
Ilustración 20-Implementación de la presentación de las combinaciones aleatorias	31
Ilustración 22-Implementación de la captura de colores análogos.....	32
Ilustración 23-Primer prototipo de la aplicación	33
Ilustración 24-Diagrama de navegación de la aplicación.....	34
Ilustración 25-Interfaz de la aplicación	35
Ilustración 26-Interfaz de la creación de combinaciones	36
Ilustración 27-Interfaz de la sección de combinaciones	37
Ilustración 28-Interfaz de visualización del menú de categorías y su interfaz.	38
Ilustración 29-Interfaz de la sección de lavadora	39
Ilustración 30-Menú lateral.....	39
Ilustración 31-Fragmento de codigo de una de las pruebas unitarias	40
Ilustración 32- Respuesta encuesta opinión	43

1. Introducción

Según el estudio “La Sociedad de la Información en España” [2] el 81% de los teléfonos móviles en España son smartphones. Actualmente se calculan unos 26,25 millones de españoles que acceden regularmente a internet. De ellos, 20,6 % se conectan diariamente. Respecto al uso de aplicaciones, hay 23 millones de usuarios que utilizan aplicaciones y realizan una media de 3,8 millones de descargas diarias. De media, cada usuario tiene instaladas 39 aplicaciones en su dispositivo.

Al mismo tiempo, la revista Muy Interesante publicó un artículo «acerca de en qué empleamos el tiempo de nuestra vida: 25 años durmiendo, 10 años comiendo y cosas así.»[4] en el que se muestran que los hombres invierten de media 177 días en su vida vistiéndose y las mujeres 531.

Analizando esta información, se plantea dar solución al malgaste de tiempo de ambos hábitos. Para ello se pretende unificar el tiempo invertido en consultar el teléfono móvil y el periodo que dedicamos en encontrar un atuendo adecuado. La solución planteada supone la creación de una aplicación móvil dirigida a todo tipo de usuarios que permita mediante los nuevos teléfonos inteligentes, consultar a cualquier hora y sin necesidad de conexión a internet los posibles conjuntos que disponemos.

Se considera que para solventar rápidamente los problemas de vestuario, el usuario necesitará al menos poder visualizar combinaciones formadas en base a los colores de las prendas, generadas aleatoriamente y combinaciones guardadas por el mismo.

1.1. Objetivos

Los objetivos principales de este trabajo son:

- Desarrollar una aplicación móvil nativa para el sistema operativo Android que permita a los usuarios obtener y crear combinaciones a partir de sus prendas de vestir. Para ello se pueden establecer los siguientes sub-objetivos:
 - Crear una base de datos que permita el almacenamiento de datos de forma persistente.
 - Diseñar una interfaz amigable e intuitiva que facilite el uso de la aplicación y que muestre al usuario las prendas y combinaciones guardadas.
 - Diseñar los algoritmos para la combinación de prendas y permitir al usuario crear combinaciones de forma manual.

2. Material y métodos utilizados

2.1. Metodología

Para realizar este proyecto se ha decidido utilizar la metodología de desarrollo iterativo e incremental. Dicho procedimiento permite dividir el proceso en iteraciones, cada una de las cuales aumentará la funcionalidad y mejorará la calidad del software respecto a la anterior. Estas iteraciones permitirán resolver los problemas que surjan a corto plazo, mejorar de forma continua el software y tener una visión futura del proceso.

En nuestro caso, utilizaremos ocho iteraciones para completar el proceso. Así mismo, la memoria del trabajo comenzará a escribirse de forma progresiva, siguiendo el transcurso de las iteraciones.

Manejo de la cámara:

- Captura de fotografías desde la cámara
- Selección de fotografías desde la galería fotográfica

Manejo de la BBDD:

- Selección del gestor de BBDD.
- Realización de una BBDD de prueba.
- Adición de las fotografías como atributo en la BBDD.

Diseño de la BBDD:

- Elección de los atributos, entidades y relaciones necesarias.
- Diseño del diagrama.
- Implementación de la BBDD.

Hacer combinaciones a mano:

- Almacenamiento de varios datos de ejemplo.
- Visualización de distintos tipos de prendas.
- Elección de las prendas para generar la combinación.
- Creación y almacenamiento de la combinación.

Sacar combinaciones:

- Mostrar combinaciones guardadas anteriormente.

Algoritmo relleno automático del color:

- Extracción del color.
- Almacenamiento del color en la BBDD.
- Creación del algoritmo.

Algoritmo propuesta de combinaciones basadas en color:

- Elección del color.
- Aparición de conjuntos de ropa basados en el color.

Otros algoritmos de propuesta de combinaciones:

- Combinaciones aleatorias.
- Combinaciones basadas en categorías.

Así mismo se ha realizado un seguimiento del proyecto mediante la herramienta Microsoft Project[7], en la que se realizó un planificación detallada de las tareas, estipulando las iteraciones en un plazo concreto de tiempo.

A continuación se mostrará una captura del progreso del proyecto en la anteriormente mencionada herramienta.

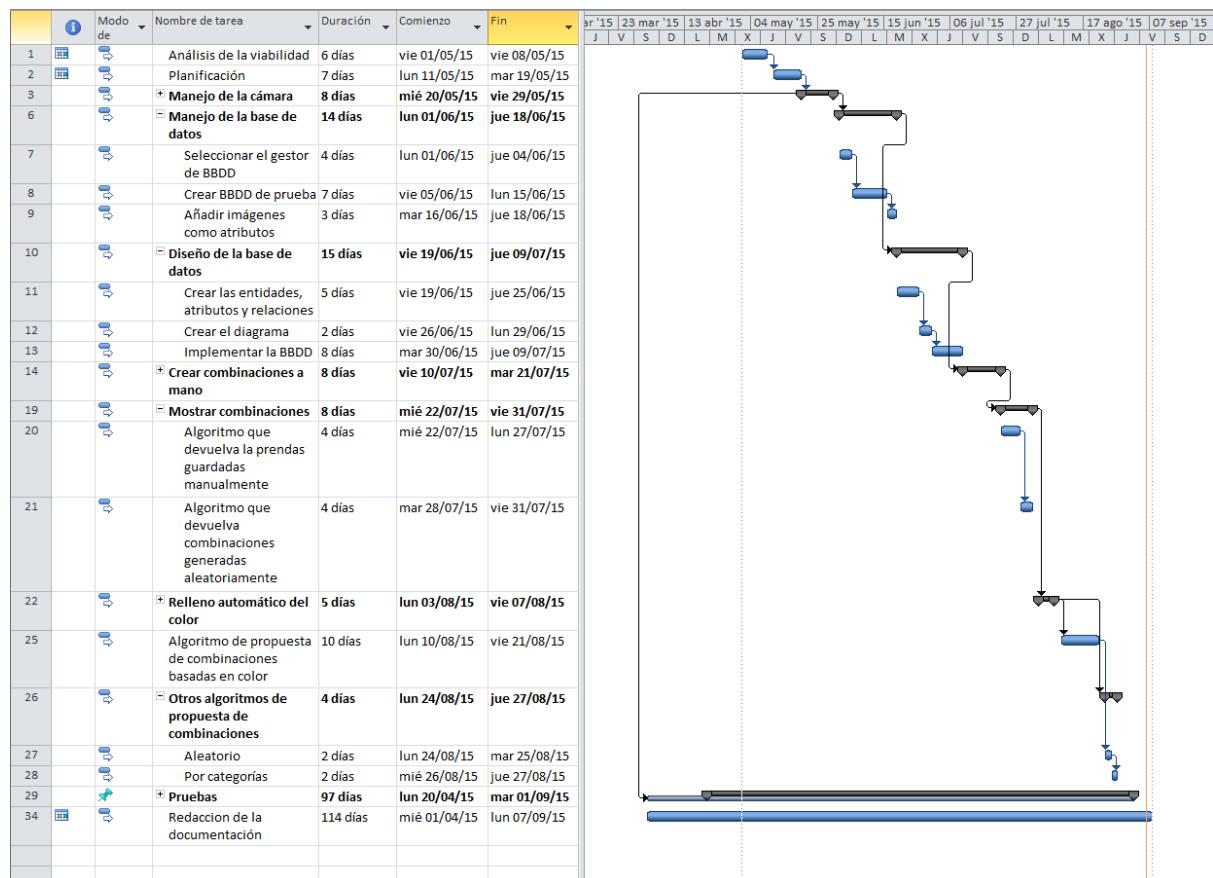


Ilustración 1-Progreso del proyecto en Microsoft Project

2.2. Tecnología y herramientas

Se expondrán y explicarán las tecnologías y herramientas necesarias para realizar el software requerido.

En nuestro caso, el proyecto se centra en la realización de una aplicación móvil cuya labor principal es la de crear combinaciones de ropa en base a prendas guardadas. Actualmente existen 3 tipos primordiales de aplicaciones móviles, aplicaciones nativas, web e híbridas.

Las nativas son aplicaciones desarrolladas para un determinado sistema operativo y permiten acceder a todas las características del hardware del dispositivo móvil, creando una experiencia de usuario más positiva y no necesitan conexión a internet para ser utilizadas.

Las web son desarrolladas en lenguajes web independientemente del sistema operativo en el que la aplicación será usada y se ejecutan dentro del navegador web de cada dispositivo.

Las aplicaciones híbridas combinan código web y nativo obteniendo las ventajas de los dos tipos anteriores, permiten crear aplicaciones multiplataforma.

Para abordar este proyecto se ha decidido crear una aplicación nativa enfocada al sistema operativo Android.

Los principales motivos por los que se ha realizado dicha elección han sido la posibilidad de acceder de forma total a los recursos del dispositivo, tener rendimiento optimizado, así como la necesidad de no tener que estar conectado a internet para poder utilizar la aplicación. Lo que genera buenos resultados en cuanto al diseño, usabilidad y eficiencia. Cabe mencionar que se ha elegido Android debido a que actualmente se considera el sistema operativo más popular.

2.2.1 Android Studio



Ilustración 2-
logo de Android
Studio

Entorno de desarrollo integrado gratuito basado en la plataforma Android fundamentado en IntelliJ IDEA. La primera versión se dio a conocer en diciembre de 2014, convirtiéndose en IDE oficial y sustituyendo a Eclipse. Dicho programa da soporte a las plataformas Windows, Linux y Mac OS X.

Algunas de las funcionalidades que Android Studio[1] ofrece son un editor de código inteligente, desarrollo de aplicaciones multi pantalla, dispositivos virtuales para todas las formas y tamaños o plantillas para crear diseños Android u otros

componentes.

El editor de código inteligente permite autocompletar, refactorizar, analizar y compilar el código de forma rápida. Muestra la documentación relacionada con un método al posicionarte encima de su código y permite invocar acciones para inspeccionarlas en un proyecto o en códigos individuales.

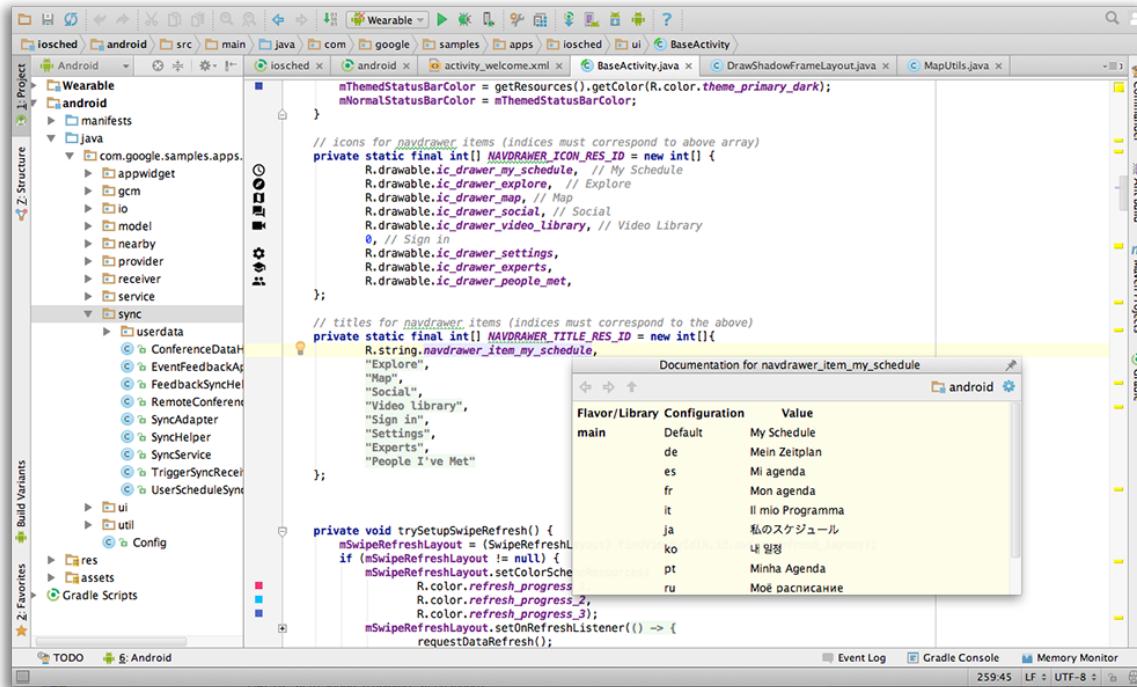


Ilustración 3- Ejemplo de código inteligente

Para realizar la parte gráfica de las aplicaciones, Android estudio combina una parte de código para los *layouts* y una vista previa dentro de su interfaz, lo que permite realizar cambios y visualizarlos instantáneamente. Estas visualizaciones pueden ser desde varios dispositivos o uno concreto.

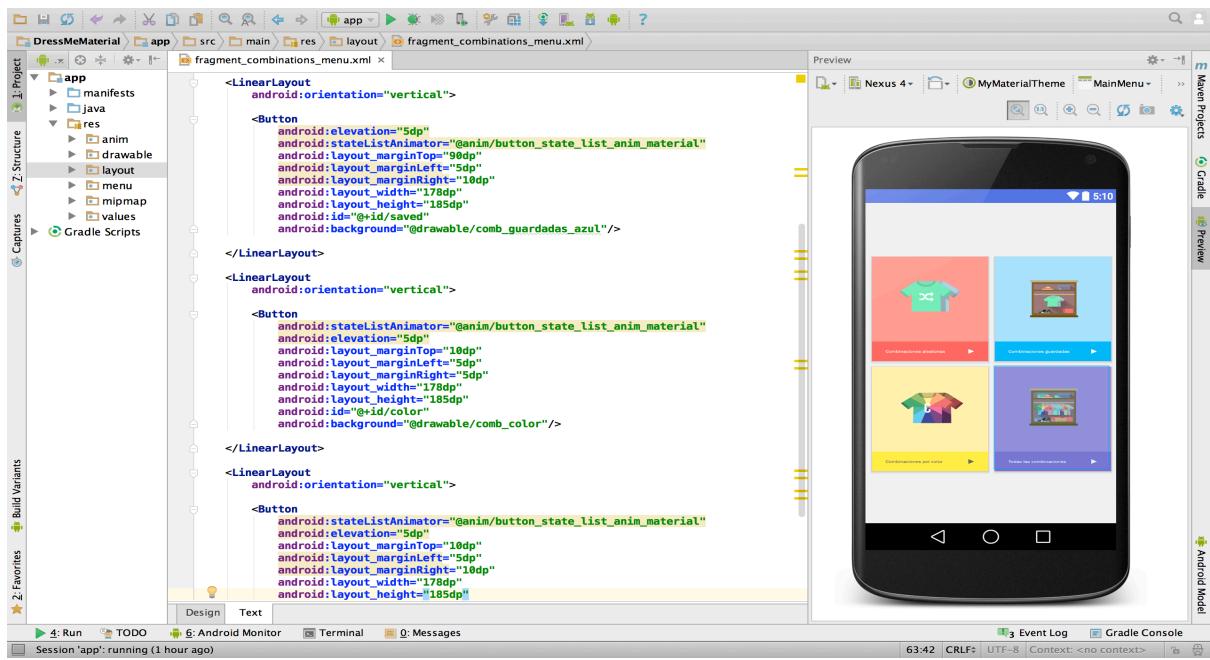


Ilustración 4-Ejemplo de layout

2.2.2. SQLite



Ilustración 5- Logo SQLite

Sistema de gestión de bases de datos relacional creado por D.Richard Hipp y cuyo lanzamiento inicial se produzco a mitad del año 2000.

que se encargan de la gestión. No precisa configuración, es transaccional y esta disponible al dominio público de los desarrolladores.

En sus últimas versiones permite crear bases de datos con un tamaño máximo de 2 terabytes y ha ampliado los tipos de datos, añadiendo el tipo BLOB. En el presente proyecto actúa de sistema gestor de la base de datos.

2.2.3. Microsoft Project



Ilustración 6-Logo Microsoft Project 2010

Software de administración de proyectos diseñado, desarrollado y comercializado por Microsoft cuya ultima versión estable se publico

en 2013. Tiene como finalidad asistir a administradores de proyectos en el desarrollo de planes, asignación de recursos a tareas, dar seguimiento al progreso, administrar presupuesto y analizar cargas de trabajo.

2.2.4. Just In Mind



Ilustración 7- Logo
JustInMind

Herramienta que permite prototipar y simular aplicaciones web 2.0, portales, intranets y sitios web, aplicaciones para móviles, aplicaciones de PC y SAP entre otros. Disponible tanto para Windows como para Mac.

Permite crear *widgets* personalizados, añadiendo imágenes, formularios y simular mediante eventos la aplicación terminada, incluyendo los carros de compras y bases de datos.

2.2.5. Magic Draw



Ilustración 8- Logo Magic
Draw

Herramienta de modelado diseñado para los analistas de negocio, analistas de software, programadores, ingenieros de control de calidad, y los escritores de documentación. Facilita el desarrollo, análisis y diseño orientado a objetos de sistemas y bases de datos (OO).

La herramienta es compatible con el estándar UML 2.3, desarrollo de código para diversos lenguajes de programación como Java, C++ o C#, así como para modelar datos.

3. Presentación del problema y análisis de requisitos

Este proyecto pretende solventar los problemas cotidianos de elección de vestuario, ya sea para actividades laborales o de ocio. El objetivo es optimizar el tiempo invertido en elegir el vestuario de forma adecuada para cualquier ocasión.

3.1 Requisitos funcionales

Con el objetivo de cumplir todas las expectativas necesarias para realizar un óptimo proyecto, se realiza el análisis de requisitos. En el análisis de requisitos funcionales se expone las funciones más importantes que el sistema software deberá realizar.

Identificador	Descripción
RF01	El usuario podrá añadir nuevas prendas a la aplicación, ya sea mediante la cámara de su dispositivo o seleccionando la prenda desde su galería fotográfica
RF04	El usuario podrá crear nuevas combinaciones de ropa
RF05	El usuario podrá visualizar combinaciones de prendas
RF06	El usuario podrá visualizar las combinaciones guardadas anteriormente.
RF07	El usuario podrá visualizar combinaciones generadas aleatoriamente
RF08	El usuario podrá visualizar combinaciones filtradas por colores
RF09	El usuario podrá visualizar combinaciones filtradas por categorías.
RF10	El usuario podrá visualizar las prendas que necesiten ser lavadas

Tabla 1-Especificación de requisitos funcionales

3.2. Requisitos no funcionales

Por el contrario, en el análisis de requisitos no funcionales se centra en listar las características generales de nuestro sistema, como por ejemplo la facilidad de uso, instalación del sistema, mantenibilidad, etc.

Identificador	Descripción	Importancia
RNF01	El sistema ha de ser diseñado para dispositivos con un sistema operativo Android.	ALTA
RNF02	El sistema ha de ser diseñado para ser compatible con cualquier dispositivo que tenga una versión de Android igual o superior a la 4.0	MEDIA

RNF03	La interfaz del sistema deberá ser fácil e intuitiva.	MEDIA
RNF04	La aplicación necesita al menos 7 mega bytes disponibles de almacenamiento dentro del dispositivo para poder ser instalada y ejecutada correctamente	MEDIA

Tabla 2-Especificación de requisitos no funcionales

3.3. Diagrama de casos de uso

A continuación se muestra el diagrama en el que están reflejados los requisitos anteriormente descritos.

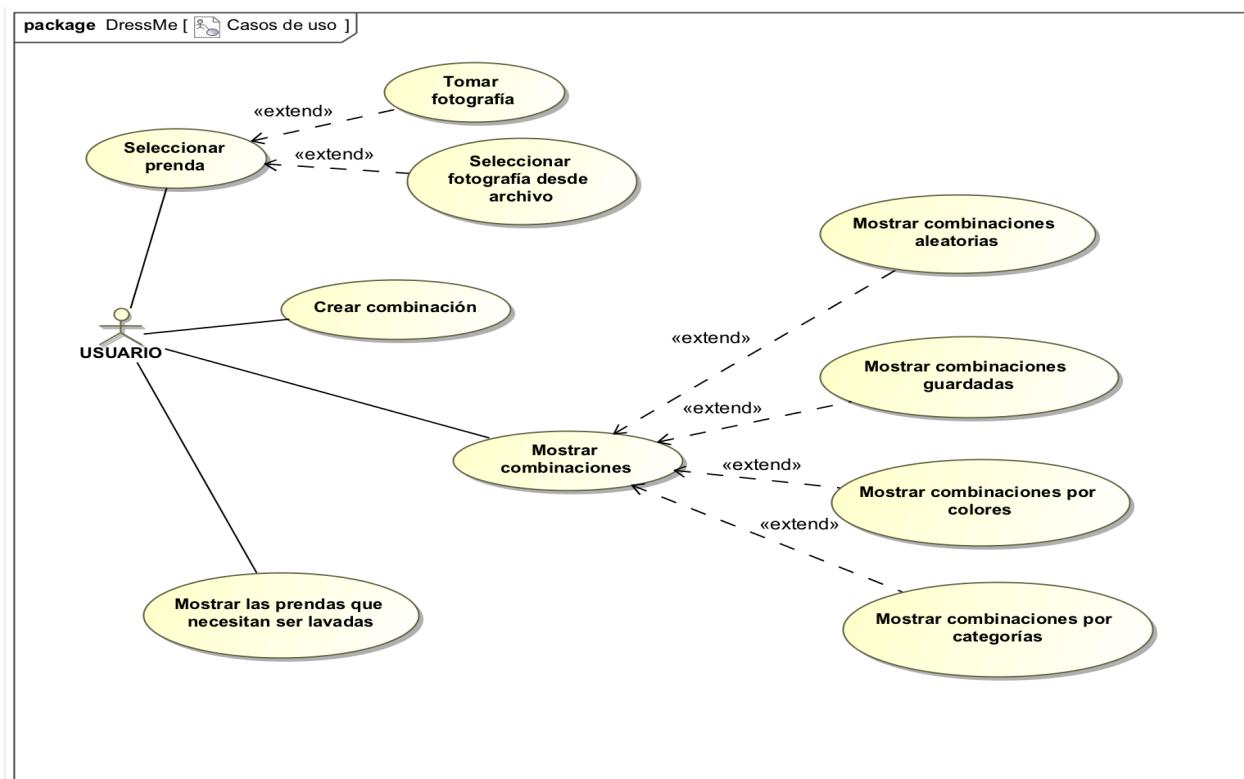


Ilustración 9- Diagrama de casos de uso

3.4. Casos de uso detallados

En las siguientes tablas se explican de forma más detallada los casos de uso mostrados en la ilustración precedente.

Identificador	Caso de uso 2
Nombre	Tomar fotografía
Descripción	El usuario selecciona la opción seleccionar prenda y elige añadir la prenda mediante el uso de la cámara del dispositivo
Actores primarios y secundarios	Usuario
Precondiciones	1. El dispositivo móvil dispone de cámara.
Postcondiciones	El usuario ha añadido de forma satisfactoria la prenda.
Escenario principal de éxito	<ol style="list-style-type: none"> 1. El usuario pulsa el botón “Seleccionar prenda”. 2. El usuario selecciona la opción tomar fotografía. 3. El usuario realiza la captura de la prenda. 4. El sistema muestra la fotografía . 5. El sistema muestra un cuestionario sobre los datos de la prenda. 6. El usuario rellena el cuestionario. 7. El usuario pulsa el botón “guardar prenda”.
Extensiones	<ol style="list-style-type: none"> 1.1. El usuario ha pulsado de forma incorrecta el botón “Seleccionar foto”. <ol style="list-style-type: none"> 3.1 La cámara no está disponible. 3.2 El usuario sale de la cámara 1.2.1 El usuario vuelve al punto 3 del escenario principal. 8.1 El usuario sale de la pantalla de la aplicación sin guardar 8.1.2 El usuario vuelve al punto 3 del escenario principal.

Identificador	Caso de uso 3
Nombre	Seleccionar fotografía desde archivo
Descripción	El usuario selecciona la opción “seleccionar prenda” y elige añadir la prenda seleccionándola desde su propia galería de imágenes.
Actores primarios y secundarios	Usuario
Precondiciones	<ol style="list-style-type: none"> 1. La aplicación ha sido instalada en el dispositivo móvil. 2. El usuario ha accedido correctamente a la aplicación. 3. El dispositivo móvil dispone de cámara.
Postcondiciones	El usuario ha añadido de forma satisfactoria la prenda.
Escenario principal de éxito	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación. 2. El usuario pulsa el botón “Seleccionar prenda”. 3. El usuario selecciona la opción seleccionar prenda desde archivo. 4. El usuario selecciona la prenda que desea en su galería de imágenes. 5. El sistema muestra la fotografía . 6. El sistema muestra un cuestionario sobre los datos de la prenda. 7. El usuario rellena el cuestionario. 8. El usuario pulsa el botón “guardar prenda”.
Extensiones	<p>1.2. El usuario ha pulsado de forma incorrecta el botón “Seleccionar foto”.</p> <p>3.1 No existen fotografía en la galería de su dispositivo.</p> <p>3.2 El usuario sale de la galería.</p> <p>3.2.1 El usuario vuelve al punto 3 del escenario principal.</p> <p>8.1 El usuario sale de la pantalla de la aplicación sin guardar</p> <p>8.1.2 El usuario vuelve al punto 3 del</p>

escenario principal.

Identificador	Caso de uso 4
Nombre	Crear combinación
Descripción	El usuario selecciona la opción "crear combinación" y desplaza las prendas de izquierda a derecha para encontrar la combinación que desea crear.
Actores primarios y secundarios	Usuario
Precondiciones	<ol style="list-style-type: none">1. La aplicación ha sido instalada en el dispositivo móvil.2. El usuario ha accedido correctamente a la aplicación.3. El usuario ha añadido prendas a la aplicación.4. La aplicación posee suficientes tipos de prendas como para realizar la combinación correctamente.
Postcondiciones	El usuario ha creado de forma satisfactoria la combinación.
Escenario principal de éxito	<ol style="list-style-type: none">1. El usuario accede a la aplicación.2. El usuario pulsa el botón "Crear combinación".3. El usuario desplaza las prendas hasta encontrar la combinación deseada.4. El sistema muestra un cuestionario sobre los datos de la prenda.5. El usuario rellena el cuestionario.

	<p>6. El usuario pulsa el botón “guardar combinación”.</p>
Extensiones	<p>2.1. El usuario pulsa un botón distinto a “Crear combinación”.</p> <p>6.1 El usuario sale de la pantalla de la aplicación sin guardar</p> <p>6.1.2 El usuario vuelve al punto 3 del escenario principal.</p>

Identificador	Caso de uso 5
Nombre	Mostrar combinaciones
Descripción	El usuario selecciona la opción “Mostrar combinaciones” y el sistema le muestra las cuatro posibles opciones de visualización.
Actores primarios y secundarios	Usuario
Precondiciones	<ol style="list-style-type: none"> 1. La aplicación ha sido instalada en el dispositivo móvil. 2. El usuario ha accedido correctamente a la aplicación. 3. El usuario ha añadido prendas a la aplicación. 4. El usuario ha creado y guardado combinaciones en el teléfono.
Postcondiciones	El usuario visualizado de forma correcta las combinaciones.

Escenario principal de éxito	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación. 2. El usuario pulsa el botón “Mostrar combinaciones”. 3. El sistema muestra las cuatro posibles opciones de visualización.
Extensiones	<ol style="list-style-type: none"> 2.1. El usuario ha pulsado un botón distinto a “Mostrar combinaciones”

Identificador	Caso de uso 6
Nombre	Mostrar combinaciones aleatorias
Descripción	El usuario selecciona “Mostrar combinaciones” y elige la opción “Mostrar combinaciones aleatorias”
Actores primarios y secundarios	Usuario
Precondiciones	<ol style="list-style-type: none"> 1. La aplicación ha sido instalada en el dispositivo móvil. 2. El usuario ha accedido correctamente a la aplicación. 3. El usuario ha añadido prendas a la aplicación. 4. La aplicación posee suficientes tipos de prendas como para realizar la combinación correctamente.
Postcondiciones	El usuario ha visualizado de manera satisfactoria las combinaciones.
Escenario principal de éxito	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación. 2. El usuario pulsa el botón “Mostrar combinaciones”. 3. El usuario selecciona la opción “Mostrar combinaciones aleatorias” 4. El sistema muestra un conjunto de combinaciones aleatorias.
Extensiones	<ol style="list-style-type: none"> 1.3. El usuario ha pulsado de forma incorrecta el botón “Combinaciones aleatorias”. 2. El sistema no tiene prendas suficientes

como para generar las combinaciones.

3.1 El sistema informa de que no hay combinaciones disponibles.

Identificador	Caso de uso 7
Nombre	Mostrar combinaciones guardadas
Descripción	El usuario escoge “Mostrar combinaciones” y elige la opción “Mostrar combinaciones guardadas”
Actores primarios y secundarios	Usuario
Precondiciones	<ol style="list-style-type: none">5. La aplicación ha sido instalada en el dispositivo móvil.6. El usuario ha accedido correctamente a la aplicación.1. El usuario ha añadido prendas a la aplicación.2. La aplicación posee suficientes tipos de prendas como para realizar la combinación correctamente.3. El usuario ha creado y guardado combinaciones.
Postcondiciones	El usuario ha visualizado de manera satisfactoria las combinaciones.
Escenario principal de éxito	<ol style="list-style-type: none">1. El usuario accede a la aplicación.2. El usuario pulsa el botón “Mostrar combinaciones”.3. El usuario selecciona la opción “Mostrar combinaciones guardadas”.4. El sistema muestra las combinaciones guardadas anteriormente por el usuario.
Extensiones	<ol style="list-style-type: none">2.1. El usuario ha pulsado de forma incorrecta el botón “Combinaciones guardadas”.3. El sistema no posee combinaciones guardadas.<ol style="list-style-type: none">3.1 El sistema informa de que no hay combinaciones disponibles.

Identificador	Caso de uso 8
Nombre	Mostrar combinaciones por colores
Descripción	El usuario selecciona “Mostrar combinaciones” y elige la opción “Mostrar combinaciones por colores”
Actores primarios y secundarios	Usuario
Precondiciones	<ol style="list-style-type: none"> 1. La aplicación ha sido instalada en el dispositivo móvil. 2. El usuario ha accedido correctamente a la aplicación. 3. El usuario ha añadido prendas a la aplicación. 4. La aplicación posee suficientes tipos de prendas como para realizar la combinación correctamente.
Postcondiciones	El usuario ha visualizado de manera optima las combinaciones.
Escenario principal de éxito	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación. 2. El usuario pulsa el botón “Mostrar combinaciones”. 3. El usuario selecciona la opción “Mostrar combinaciones por color” 4. El sistema muestra un conjunto de combinaciones por color.
Extensiones	<p>3.1. El usuario ha pulsado de forma incorrecta el botón “Combinaciones por color”.</p> <p>4.1. El sistema no tiene prendas suficientes como para generar las combinaciones.</p> <p>4.1.1. El sistema informa de que no hay combinaciones disponibles.</p>

Identificador

Caso de uso 9

Nombre	Mostrar combinaciones por categorías
Descripción	El usuario selecciona "Mostrar combinaciones" y elige la opción "Mostrar combinaciones por categorías"
Actores primarios y secundarios	Usuario
Precondiciones	<ol style="list-style-type: none"> 1. La aplicación ha sido instalada en el dispositivo móvil. 2. El usuario ha accedido correctamente a la aplicación. 3. El usuario ha añadido prendas a la aplicación. 4. La aplicación posee suficientes tipos de prendas como para realizar la combinación correctamente.
Postcondiciones	El usuario ha visualizado de manera optima las combinaciones.
Escenario principal de éxito	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación. 2. El usuario pulsa el botón "Mostrar combinaciones". 3. El usuario selecciona la opción "Mostrar combinaciones por categoría" 4. El sistema muestra menú con las categorías.
Extensiones	<ol style="list-style-type: none"> 2.1. El usuario ha pulsado de forma incorrecta el botón "Combinaciones por categoría". 4.1. El sistema no posee combinaciones para mostrar. <ol style="list-style-type: none"> 4.1.1. El sistema informa de que no hay combinaciones disponibles.

Identificador	Caso de uso 10
Nombre	Mostrar prendas que necesitan ser lavadas.

Descripción	El usuario selecciona la opción “Mi Lavadora” y el sistema permite visualizar combinaciones mezcladas entre combinaciones por color, aleatorias y guardadas.
Actores primarios y secundarios	Usuario
Precondiciones	<ol style="list-style-type: none"> 1. La aplicación ha sido instalada en el dispositivo móvil. 2. El usuario ha accedido correctamente a la aplicación.
Postcondiciones	El usuario ha visualizado de manera optima las prendas que necesitan ser lavadas.
Escenario principal de éxito	<ol style="list-style-type: none"> 1. El usuario accede a la aplicación. 2. El usuario pulsa el botón “Mi Lavadora”. 3. El sistema muestra las prendas que han sido utilizadas más de dos veces.
Extensiones	<ol style="list-style-type: none"> 2.1 El usuario ha pulsado de forma incorrecta el botón “Mi Lavadora”. 3.1 No existen prendas sucias. <ol style="list-style-type: none"> 3.1.1 El sistema notifica que no es necesario lavar ninguna prenda.

4. Diseño e Implementación

4.1. Diseño arquitectónico

Para desarrollar el proyecto correctamente según los requisitos anteriormente descritos, es necesario elaborar una interfaz gráfica intuitiva, amigable y fácil de utilizar. Una base de datos donde almacenar las prendas agregadas por el usuario y sus respectivos colores, así como las combinaciones que el usuario elija o cree. Del mismo modo, se necesita crear la lógica de negocio que permita conectar la parte visual de la aplicación con el almacenamiento de datos.

Por ello, se decide realizar una diseño compuesto por una arquitectura en tres capas[9], capa de presentación, negocio y datos. Separando cada capa para conseguir independencia, pero apoyándose cada una de ellas en los servicios y facilidades ofrecidos por la capa inmediatamente anterior. Lo que proporciona una arquitectura cambiante y portátil y permite desarrollar los sistemas de forma incremental.

1. Capa de presentación 2. Capa de negocio 3. Capa de datos

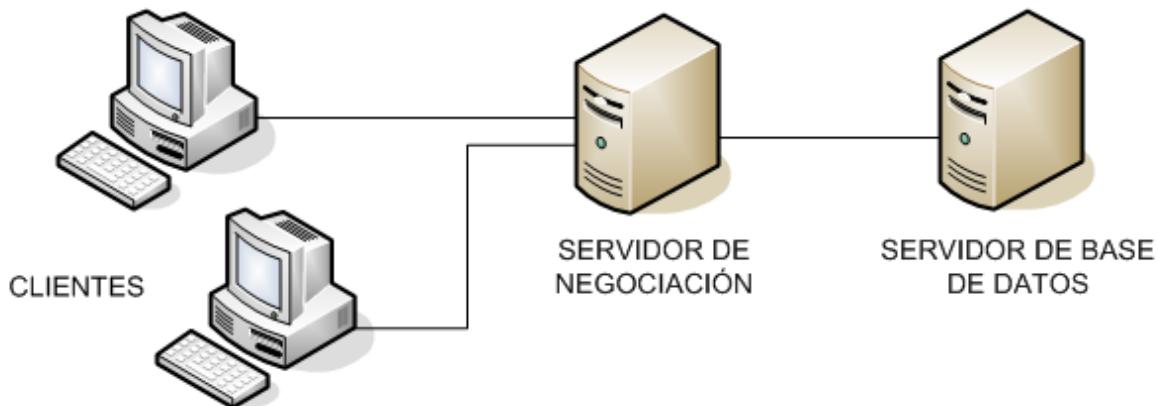


Ilustración 10- Diseño arquitectónico en tres capas

A continuación se explica detenidamente qué funciones realiza cada capa:

Capa de presentación: Proporciona facilidades de interfaz de usuario, comunicándose con el y recibiendo la información que este introduce. Así mismo, dicha interfaz debe ser entendible y accesible para el. La capa está comunicada con la capa de negocio únicamente.

Capa de negocio: Capa intermedia encargada de relacionar la capa de presentación y la capa de datos. En ella se reciben las peticiones y se envían las respuestas del usuario y se almacenan y recuperan los datos de la capa de datos.

Capa de datos: Formada por uno o varios gestores de bases de datos, es la encargada de almacenar los datos recibidos y devolverlos a la capa de negocio cuando sean necesarios, es la única capa que puede acceder a ellos.

A continuación se muestran los elementos del diseño del sistema mediante un diagrama de componentes creado a través de la herramienta Magic Draw[8].

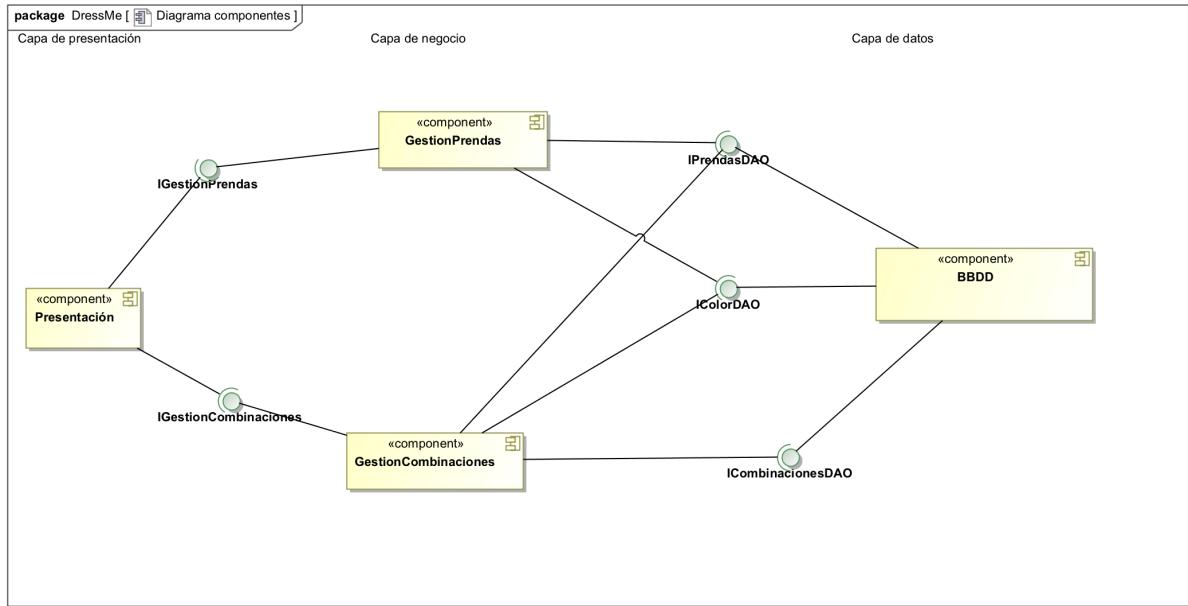


Ilustración 11- Diagrama de componentes

Y se podrá visualizar de forma más detallada las operaciones entre la capa de presentación y negocio.

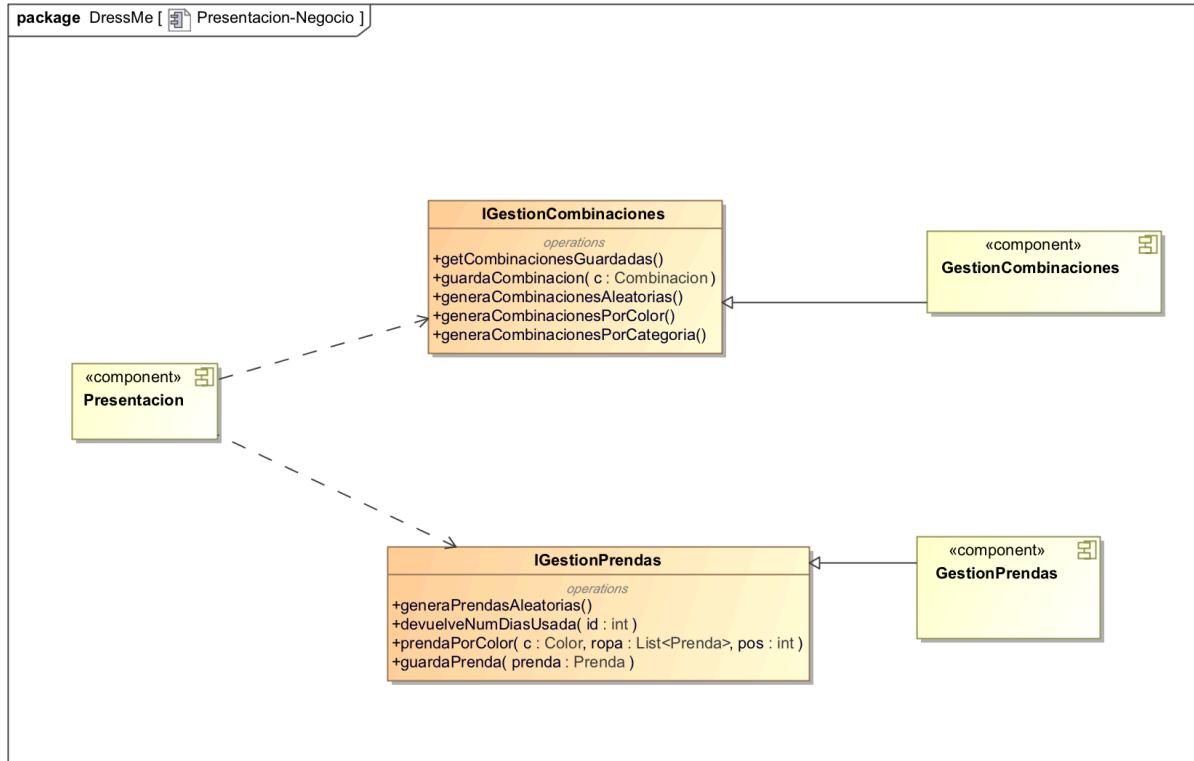


Ilustración 12- Operaciones entre la capa de presentación y negocio

Así como las operaciones de las interfaces entre la capa de negocio y datos.

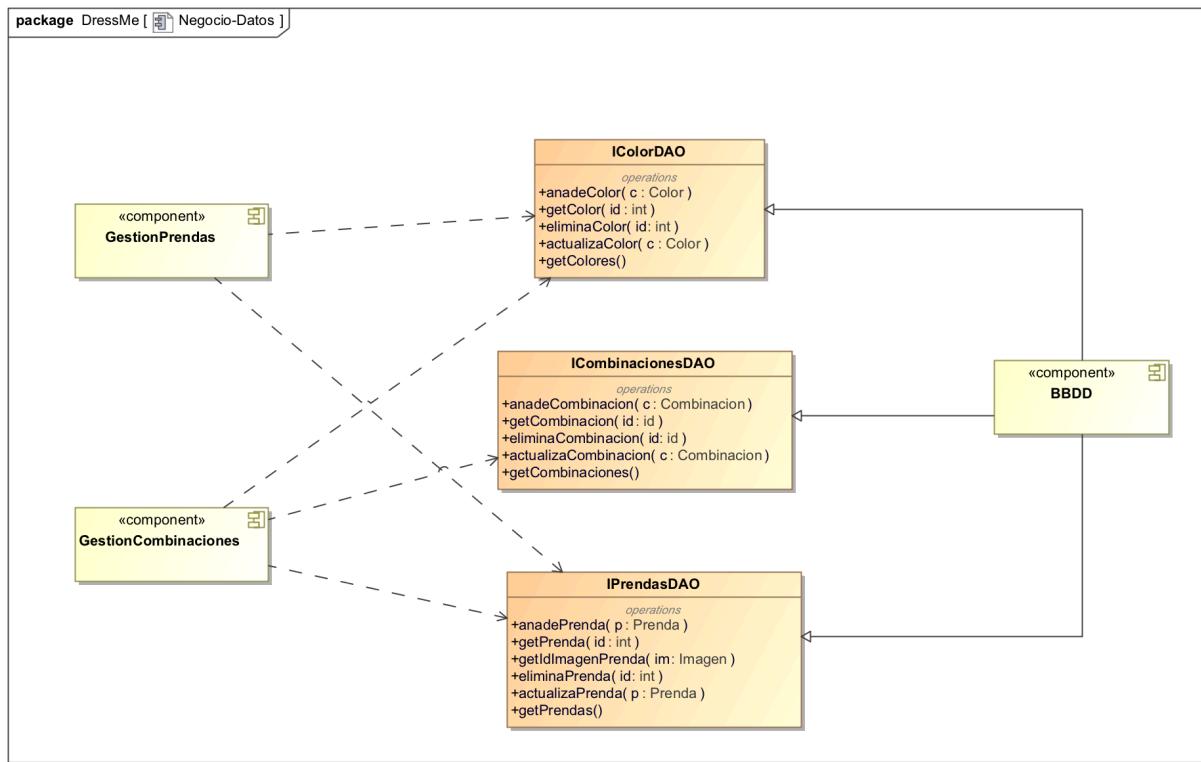


Ilustración 13-Operaciones entre la capa de negocio y datos

Una vez realizado el diseño del sistema, se procede a implementar los diseños especificados anteriormente.

En los siguientes sub apartados, se explicará en detalle la implementación seguida en cada una de las capas. Cabe mencionar que debido a la metodología incremental iterativa, dichas implementaciones pertenecen a un desarrollo global que ha ido evolucionando conforme el paso del tiempo.

4.2. Diseño e implementación de la capa de datos

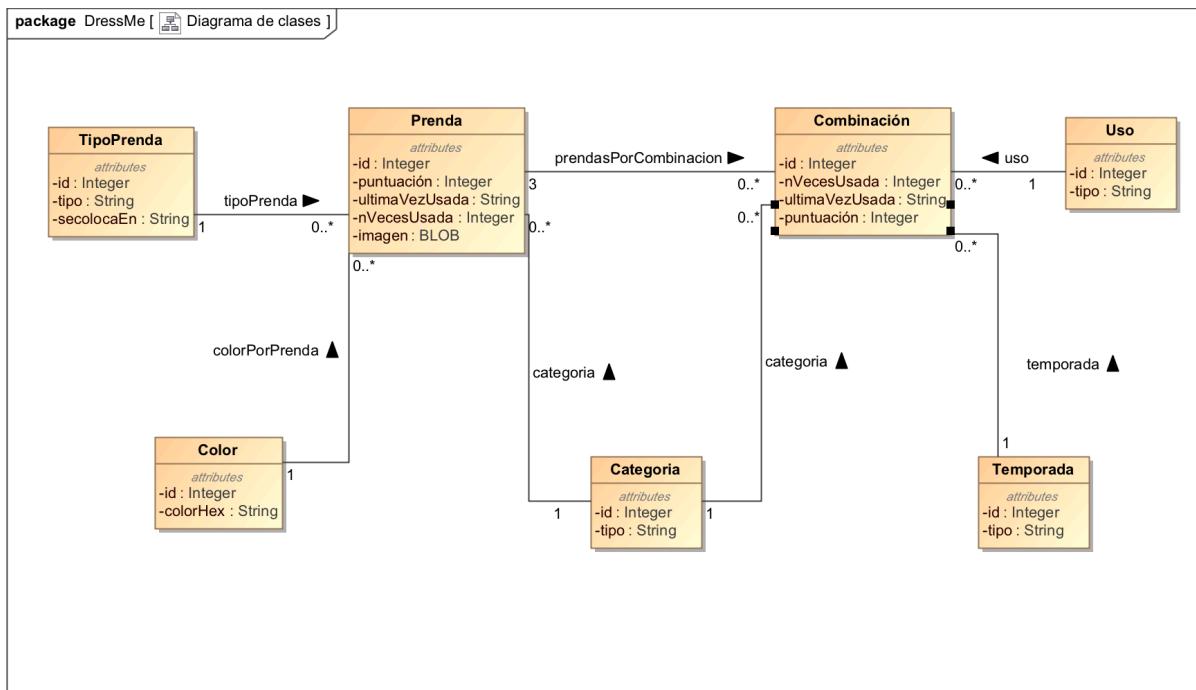


Ilustración 14-Diseño de la base de datos

En la imagen se muestra el diagrama de clases UML[6] que reflejan el diseño detallado de la base de datos. En el que la tabla principal es Prenda en donde se almacena los datos de la prenda añadida por el usuario, en donde el mismo introduce manualmente el tipo de prenda que es, la categoría en la que esta sería añadida y una puntuación que el considere. Los demás datos serán rellenados automáticamente por el sistema.

La tabla Combinación representa el conjunto de prendas que el usuario ha elegido en un determinado momento o el sistema ha creado. Sus datos son almacenados de forma similar a los de la tabla anterior, ya que parte de ellos son introducidos por el usuario cuando este cree una combinación y los otros son rellenados automáticamente por el sistema en el caso en que sean requeridos.

La tabla Color simboliza la tonalidad de la prenda que ha sido añadida por el usuario y sus datos son añadidos por el sistema en el momento en el que el usuario almacena una prenda.

Las demás tablas son utilizadas para representar el uso de cada prenda, la categoría en la que el usuario considera que debe de estar, el uso que va a darse, el tipo de prenda que es y la temporada a la que pertenece.

Para implementar esta base de datos se ha utilizado el sistema de gestión de bases de datos SQLite[10] descrito anteriormente en el apartado de tecnologías y herramientas.

A continuación se muestra un ejemplo de la forma en la que se añade una combinación en la base de datos.

```

public static int anadeCombinacion(Combinacion c){
    ContentValues valores=new ContentValues();
    valores.put("nVeces",c.getnVeces());
    String ult=null;
    if(c.getUltimaVezUsada()!=null){
        ult = c.getUltimaVezUsada().toString();
    }
    valores.put("ultimaVezUsada",ult);
    valores.put("puntuacion",c.getPuntuacion());
    valores.put("categoria",c.getCategoría());
    valores.put("temporada", c.getTemporada());
    valores.put("uso", c.getUso());

    return (int)Database.db.insert("Combinacion",null,valores);
}

```

Ilustración 15-Ejemplo de código en SQLite

Para poder insertarla, se necesita crear un contenedor de valores en el que introducir todos los datos de la nueva combinación y pasárselos a la base de datos en la inserción.

4.3. Diseño e implementación de la capa de negocio

A continuación se expondrán las funcionalidades consideradas más importantes elaboradas en la capa de negocio.

4.3.1. Selección y captura de imágenes

En primer lugar se demostrará el funcionamiento del acceso a la cámara del dispositivo y a la galería fotográfica mediante las clases *TakePicture* y *ChoosePicture*.

```

class TakePicture implements View.OnClickListener{
    @Override
    public void onClick(View v) {
        Intent intent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);

        intent.putExtra(MediaStore.EXTRA_OUTPUT, Uri.fromFile(f));
        startActivityForResult(intent, CAMERA);
    }
}

class ChoosePicture implements View.OnClickListener{
    @Override
    public void onClick(View v) {
        Intent intent = new Intent(Intent.ACTION_PICK, android.provider.MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
        intent.setType("image/*");
        startActivityForResult(Intent.createChooser(intent, "Selecciona la Foto"),FILE);
    }
}

```

Ilustración 16- implementación de la selección y captura de imágenes

Take Picture captura la imagen de la cámara, la almacena en un archivo y realiza una llamada al método *on activity result[3]* en el que se guarda la imagen en la base de datos. *ChoosePicture* actúa de forma análoga, solo

que en vez de capturar la imagen de la cámara, la toma de las imágenes del dispositivo móvil.

```
@Override  
public void onActivityResult(int requestCode, int resultCode, Intent data) {  
    if(resultCode == -1) {  
        //change to process picture fragment  
        Uri selectedImageUri;  
        String tempPath;  
        if (requestCode == CAMERA) {  
            tempPath = f.getAbsolutePath();  
        } else {  
            selectedImageUri = data.getData();  
            tempPath = getPath(selectedImageUri, getActivity());  
        }  
        mainActivity.sendPicture(tempPath);  
        mainActivity.changeFragment(DressMe.FRAGMENT_NAMES.PROCESS, "Nueva Foto",-1);  
    }  
}
```

Ilustración 17-Implementación del método OnActivityResult

Si el código de repetición es cámara entonces coge la ruta del archivo temporal que se le ha pasado a la cámara. En el caso contrario, coge la ruta del archivo de la imagen existente y se lo traslada a la actividad principal. En la cual se crea la nueva prenda y se calcula su color dominante mediante la imagen.

Para detectar el color predominante de la imagen, se calcula los pixeles totales de la imagen y se recogen los tres componentes del color predominante y se combinan para formar el color final.

```
public static List<Integer> calculateColours(Bitmap bm){  
    //el empiezo del cuadrado que vamos a coger de la imagen  
    int x=(bm.getWidth()/2)-(bm.getWidth()/4);  
    int y=(bm.getHeight()/2)-(bm.getHeight()/4);  
    //la altura y anchura del cuadrado de la imagen  
    int wi=bm.getWidth()/2;  
    int he=bm.getHeight()/2;  
  
    Bitmap croppedBm = Bitmap.createBitmap(bm,x,y,wi,he);  
    int[] pixels= new int[croppedBm.getWidth()*croppedBm.getHeight()];  
    //cogemos los pixels de la imagen que nos han pasado para buscar el color predominante  
  
    croppedBm.getPixels(pixels,0,croppedBm.getWidth(),0,0,croppedBm.getWidth(),croppedBm.getHeight());  
    int numCols=0;  
    int r=0,g=0,b=0;  
    for(int i=0;i<pixels.length;i+=10){  
        numCols++;  
        //convertir del valor a byte  
        r+=(pixels[i]>>16) & 0xFF;  
        g+=(pixels[i]>>8) & 0xFF;  
        b+=pixels[i] & 0xFF;  
    }  
    r/=numCols;  
    g/=numCols;  
    b/=numCols;  
  
    ArrayList<Integer> cols = new ArrayList<>();  
    Color col = new Color();  
    System.out.println(r+" , "+g+" , "+b);  
  
    cols.add(Color.rgb(r,g,b));  
    return cols;  
}
```

Ilustración 18-Implementación del cálculo de colores

Una vez finalizada la detección del color, se almacena la prenda en la base de datos.

4.3.2. Almacenamiento y creación de combinaciones

Para la creación de las combinaciones manualmente por el usuario, se presentan en la pantalla mediante un *flipper*. Este está compuesto por tres *layouts* en los cuales se proyectan las imágenes que componen las combinaciones. Al finalizar el usuario su creación, pulsa el botón guardar.

En ese momento, se almacena el identificador de la prenda a la que pertenece cada imagen activa. Seguidamente se procede a crear una nueva combinación en la cual se añaden los datos introducidos por el usuario y las tres prendas que hemos almacenado y se realiza la petición a la base de datos para que almacene la combinación.

```
saveCombination.setOnClickListener((v) -> {
    int seasonID = season.getSelectedItemPosition()+1;
    int usageID = usage.getSelectedItemPosition()+1;
    int categoryID = category.getSelectedItemPosition()+1;
    int ratingID = (int) rating.getRating();
    mainActivity.saveCombination(topTouchListener, middleTouchListener, bottomTouchListener, 0, ratingID, seasonID, usageID, categoryID);
});
```

Ilustración 19-Implementación del almacenamiento de combinaciones

4.3.3. Combinaciones aleatorias

Se realiza una consulta a la base de datos en la que se obtiene el número de prendas y las de prendas de cada tipo almacenadas. Se genera aleatoriamente para cada tipo de prenda una cifra que no supere el número de prendas máximas, y se utiliza para localizar la prenda dentro de la lista de prendas que nos devolvió la base de datos.

Se realiza lo mismo para cada tipo de prenda, que en este caso sería torso, piernas y pies y se proyecta en la pantalla del mismo modo que las anteriores combinaciones, para que el usuario seleccione si le gusta o no alguna de ellas.

En la siguiente imagen se muestra la forma en la que se presentan las prendas al usuario.

```
public void addView(){
    if(available > 0) {
        if(type == RANDOM) {
            generateRandomClothes();
        }
        LinearLayout ly = new LinearLayout(context);
        ly.setOrientation(LinearLayout.VERTICAL);
        LinearLayout.LayoutParams params = new LinearLayout.LayoutParams(ViewGroup.LayoutParams.MATCH_PARENT, ViewGroup.LayoutParams.MATCH_PARENT);
        params.weight = 1;
        ImageView topImage = new ImageView(context);
        topImage.setLayoutParams(params);
        topImage.setImageBitmap(clothes.get(index-1).getImagen());
        ImageView middleImage = new ImageView(context);
        middleImage.setLayoutParams(params);
        middleImage.setImageBitmap(clothes.get(index-2).getImagen());
        ImageView bottomImage = new ImageView(context);
        bottomImage.setLayoutParams(params);
        bottomImage.setImageBitmap(clothes.get(index-3).getImagen());
        ly.addView(bottomImage); ly.addView(middleImage); ly.addView(topImage);

        flipper.addView(ly);
        available--;
    }
}
```

Ilustración 20-Implementación de la presentación de las combinaciones aleatorias

4.3.4. Prendas por color

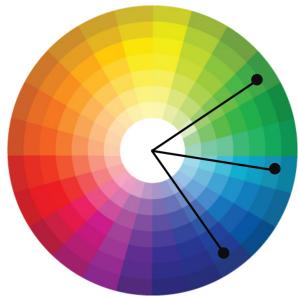


Ilustración 21-
Representación de los
colores análogos

El funcionamiento en el que se crean las prendas por color es equivalente a las anteriores. En ella se combinan las prendas siguiendo un patrón de colores. En este proyecto se decide realizar la combinación seleccionando las prendas que tengas colores análogos. Dichos colores, son aquellos que se sitúan a ambos lados de cualquier color en el círculo cromático.

```
public static List<ColorPrenda> getAnalogs(int red, int green,  
ArrayList ret = new ArrayList();  
  
float[] hsv= new float[3];  
Color.RGBToHSV(red,green,blue,hsv);  
//añadir al angulo  
hsv[0]+=50;  
hsv[0] %= 360;  
ret.add(new ColorPrenda(Color.HSVToColor(hsv)));  
hsv[0]+=50;  
hsv[0] %= 360;  
ret.add(new ColorPrenda(Color.HSVToColor(hsv)));  
  
return ret;  
}
```

Ilustración 22-Implementación de la captura de colores análogos

Se introduce en el método el color descompuesto en componentes rgb, y se calculan los otros dos colores análogos al introducido. Se transforma el modelo de color rgb a hsv, en donde la h simboliza los grados que tiene el ángulo del color. Se suma a dicho ángulo 50 grados, se comprueba que no supera los 360° del círculo cromático, se convierte a rgb y se almacena en la lista de colores. Se realiza la misma acción posteriormente para conseguir el segundo color.

4.3.5 Combinaciones por categorías

Se realiza una consulta a la base de datos acerca de las combinaciones relacionadas con cada categoría y esta nos devuelve la lista de conjuntos que forman cada una de ellas. Mediante el flipper anteriormente mencionado y siguiendo el mismo funcionamiento que los algoritmos descritos en los apartados anteriores, se muestran las combinaciones para cada una de las categorías alojadas en la base de datos.

4.4. Diseño e implementación de la capa de presentación

En esta capa se muestran las interfaces con las que el usuario interactúa. Mediante capturas de pantalla, se irá realizando un recorrido por la aplicación y se explicará el código utilizado para llevarla a cabo.

Para ello se comenzará por el primer prototipo de la aplicación, el cual ha sido creado utilizando la herramienta JustInMind[5]. Este prototipo se ha ido modelando y actualizando conforme a las iteraciones del proyecto, ampliando las funcionalidades y resolviendo los fallos encontrados. A continuación se muestra unas capturas del primer prototipo.

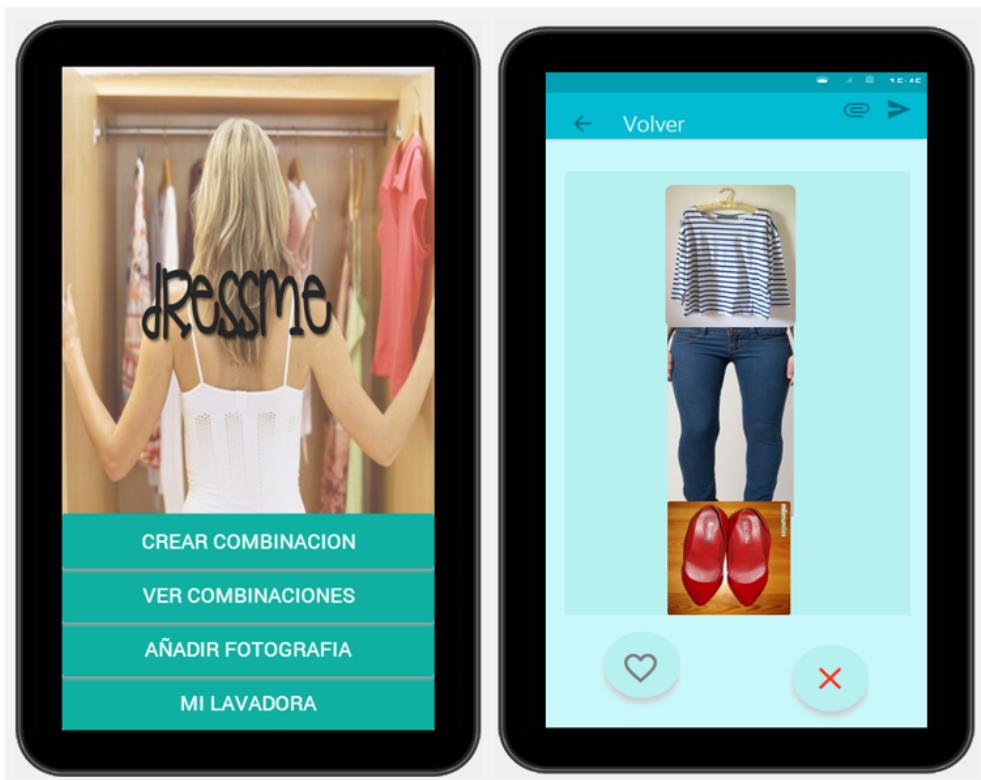


Ilustración 23-Primer prototipo de la aplicación

Con el objetivo de describir correctamente la aplicación, se muestra un mapa de navegación del sistema. En él se presentan las pantallas a las que el usuario puede acceder.

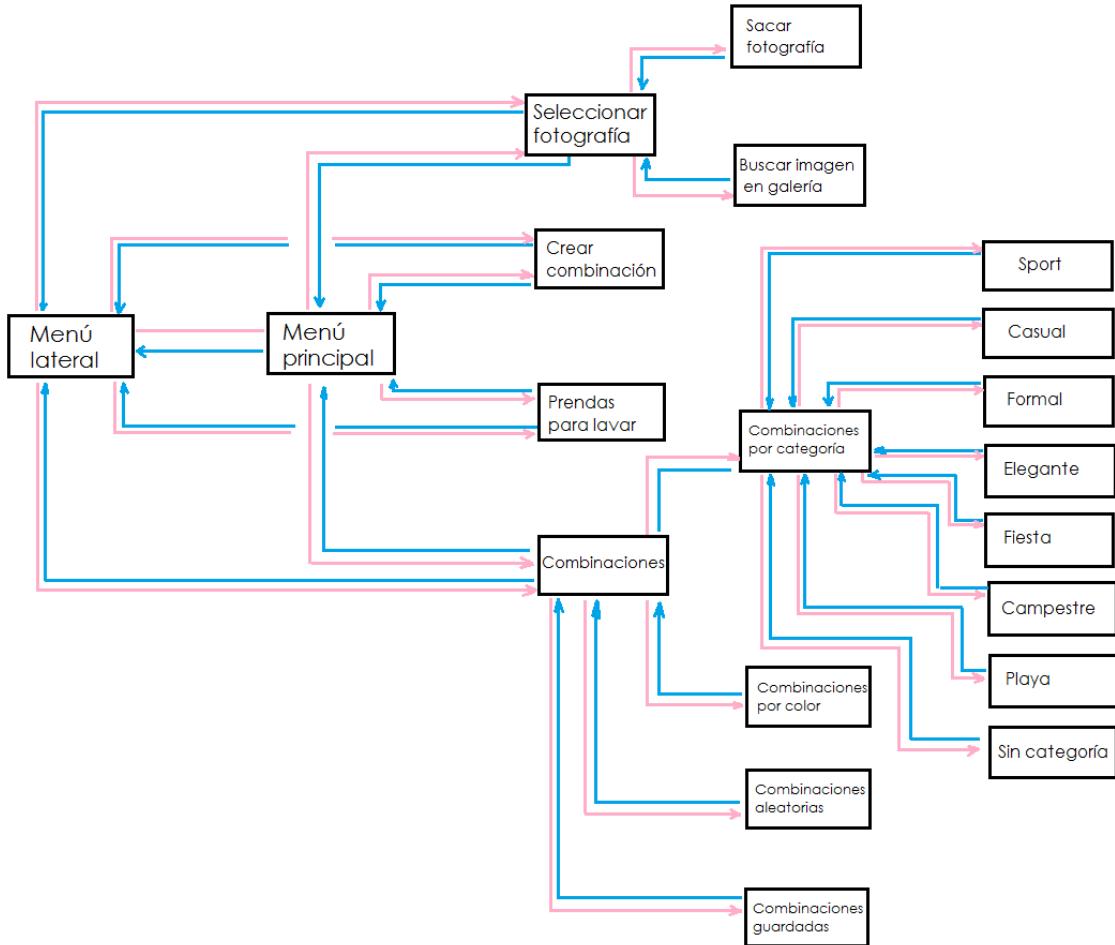


Ilustración 24-Diagrama de navegación de la aplicación

La vista principal de la aplicación está formada por cuatro botones principales con los que el usuario tiene que interactuar. Para poderla describir correctamente, se procederá a realizar la explicación de forma ordenada, empezando por la selección de la prenda, la creación de combinaciones, visualización de combinaciones y finalmente la lavadora.

El primer botón permite seleccionar la prenda que se desea añadir a la aplicación, dando la opción de capturar la fotografía desde la cámara del dispositivo o mediante la galería de imágenes.

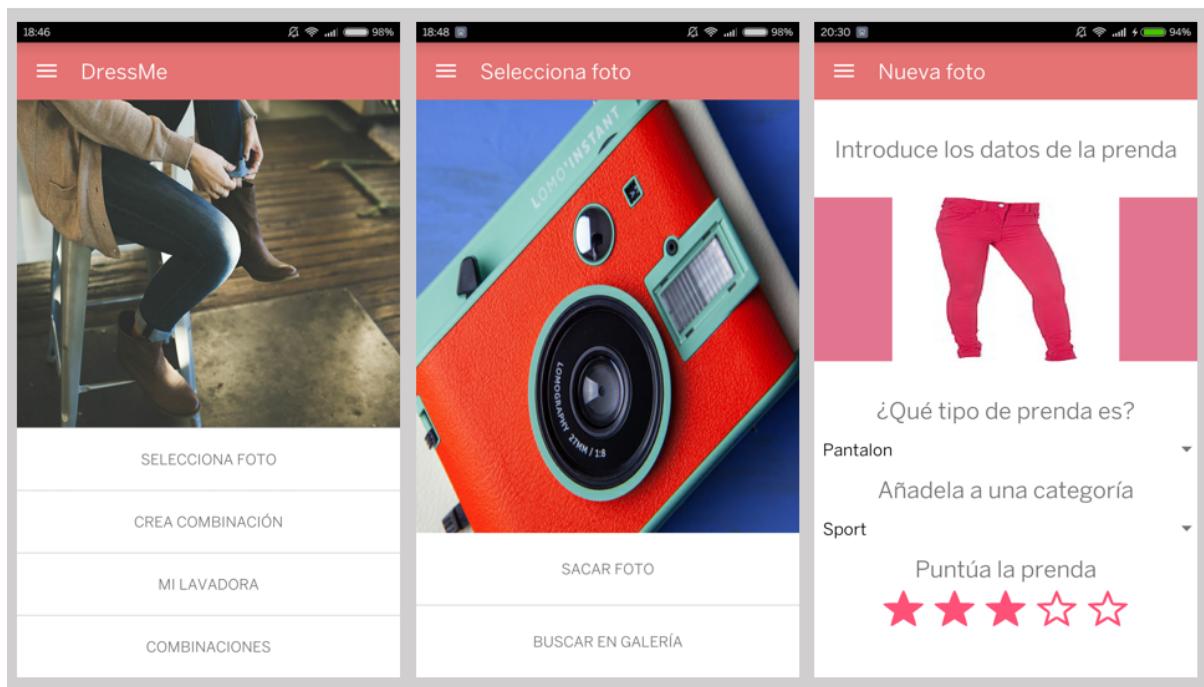


Ilustración 25-Interfaz de la aplicación

Una vez seleccionada la imagen que desea añadir, se dirige al usuario a un cuestionario donde debe completar la información de la prenda introducida. Para ello tiene que llenar el tipo de prenda que ha introducido, la categoría a la que pertenece y el grado de satisfacción que dicha prenda le produce.

El botón “Crea Combinación” permite al usuario crear de forma manual la combinación que este desee. Al pulsar el botón este será dirigido a una pantalla en la que se muestran tres imágenes que representan el tronco, las piernas y los pies.

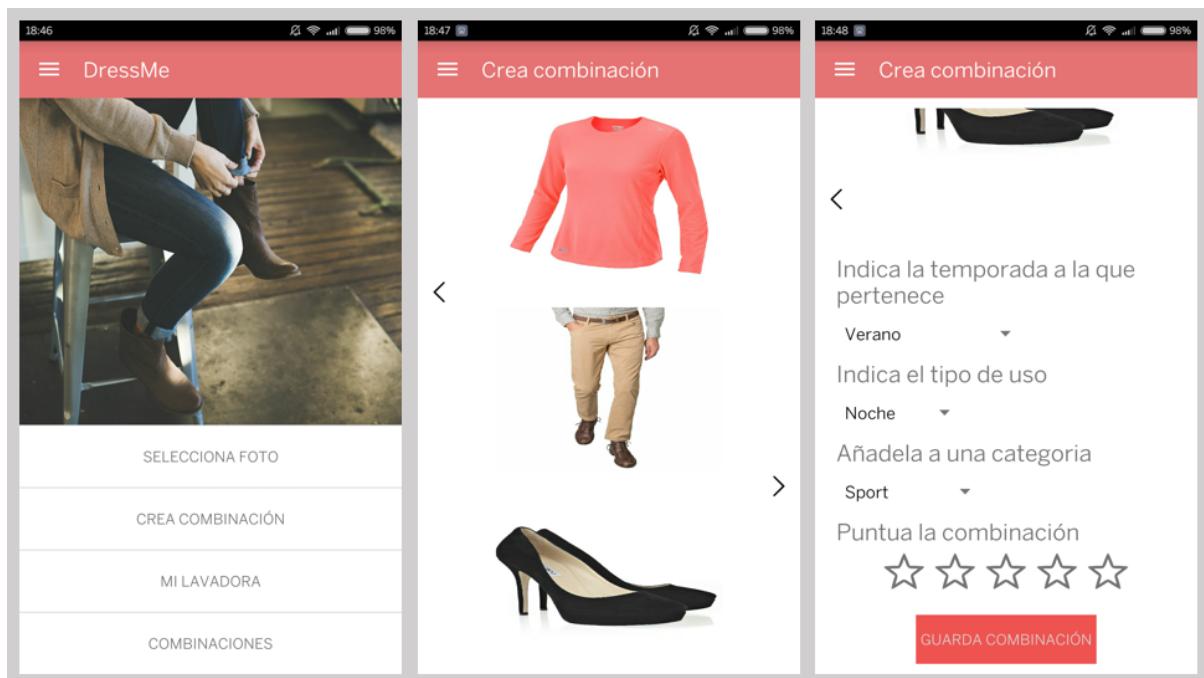


Ilustración 26-Interfaz de la creación de combinaciones

Para elegir la prenda correcta en cada parte, el usuario debe deslizar el dedo por encima de cada prenda hacia derecha o izquierda y la imagen cambiará. Una vez haya seleccionado las prendas adecuadas, este deberá llenar un formulario en el que indique la temporada a la que considera que la combinación pertenece, el tipo de uso que le dará y la categoría a la que pertenece y finalmente el grado de satisfacción hacia dicha combinación.

Respecto a la opción combinaciones, esta traslada al usuario a cuatro posibles combinaciones: combinaciones por color, combinaciones guardadas, combinaciones aleatorias y filtradas por categorías.

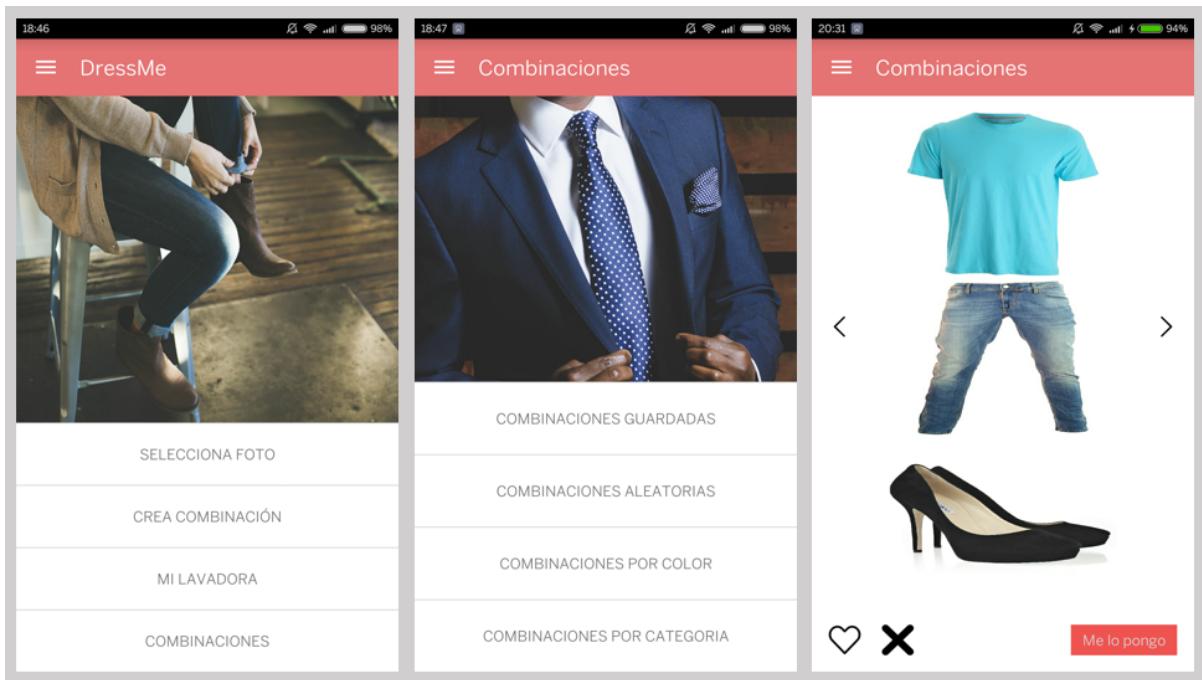


Ilustración 27-Interfaz de la sección de combinaciones

Ambas cuatro están presentadas al usuario del mismo modo, una estructura de tres piezas de ropa posicionadas verticalmente y tres botones en la parte posterior. Cada uno de ellos tiene una función, el corazón es utilizado para que el usuario señale si la combinación mostrada es de su agrado, la “X” refleja su disgusto y “Me lo pongo” evidencia su futura vestimenta.

En la opción “Combinaciones por categoría” existe un menú intermedio hasta llegar a la visualización de las combinaciones, en el cual el usuario deberá seleccionar la categoría que más se aadecue a su búsqueda. Una vez elegida, se mostrarán las combinaciones del modo anteriormente descrito.

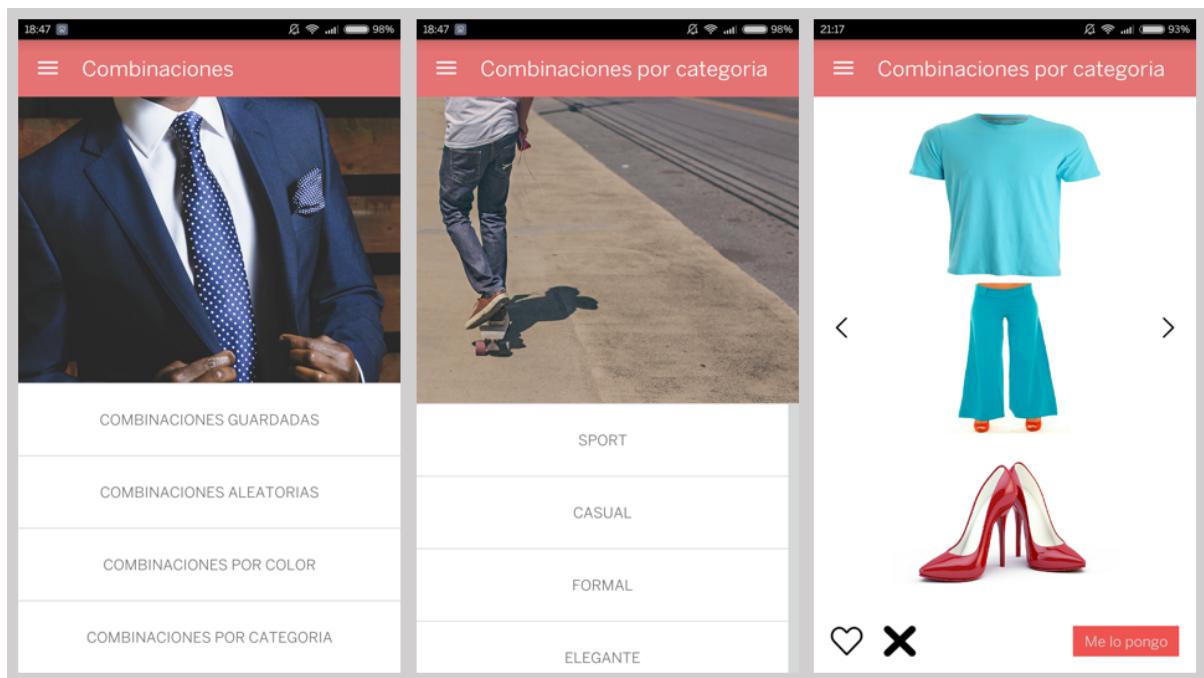


Ilustración 28-Interfaz de visualización del menú de categorías y su interfaz

Existen diferencias funcionales entre ellas, las combinaciones por color y aleatorias no están guardadas, sino que son generadas y representadas en la pantalla. En el caso en el que el usuario decida vestirse siguiendo esa combinación o indique su agrado por ella, esta será almacenada en la base de datos.

Sin embargo en las combinaciones guardadas y por categoría, al pulsar cualquiera de dichos botones, los datos de los conjuntos ya almacenados serán actualizados, modificando la fecha y el numero de veces que dicha combinación haya sido utilizada o aumentando su puntuación. Por el contrario, si el usuario pulsa la cruz, la combinación será eliminada.

Para finalizar el menú principal, “Mi Lavadora” permite al usuario ver que prendas han sido usadas más asiduamente y por lo tanto se considera que deben ser lavadas. Para ello se muestran las prendas en la pantalla de manera individual, permitiendo que el usuario visualice las siguientes prendas al deslizar el dedo por encima de ellas. Para indicar que la prenda ha sido lavada, el usuario deberá pulsar el botón en forma de lavadora. Al realizar dicha acción, se actualizará los datos de la prenda y desaparecerá de esa sección.

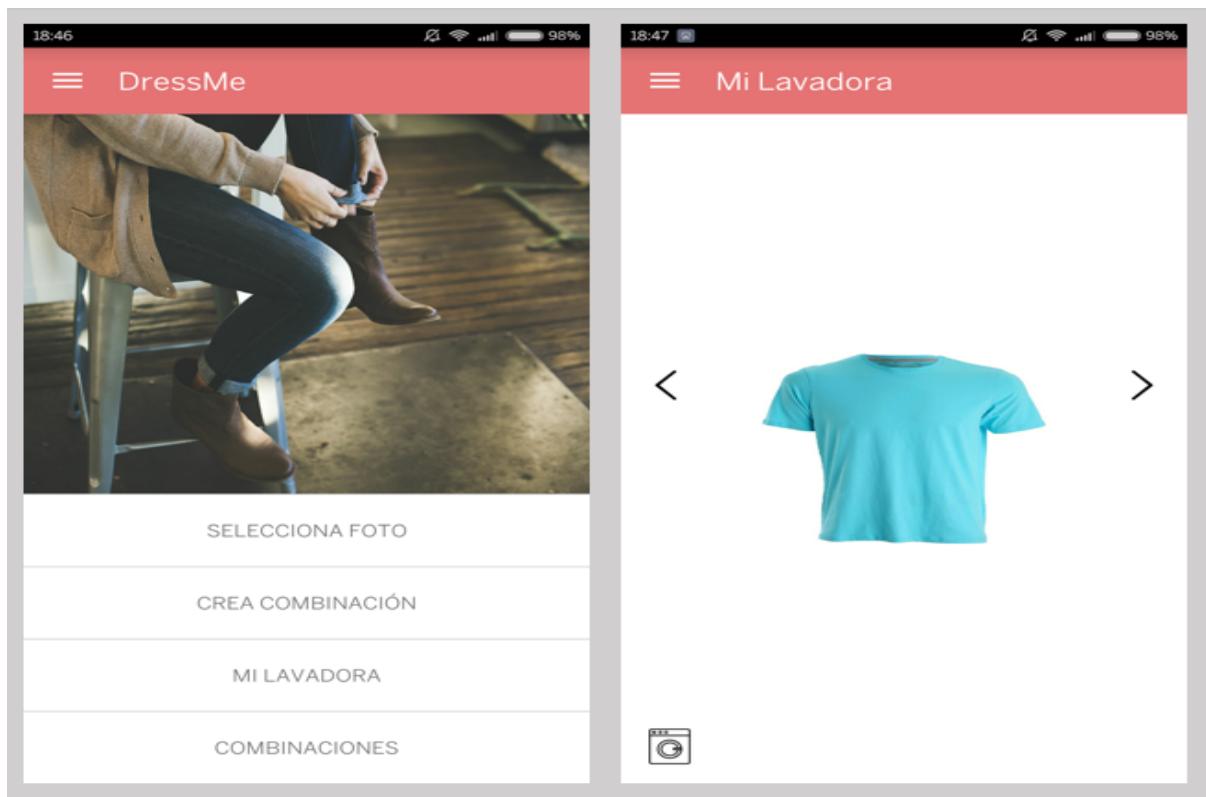


Ilustración 29-Interfaz de la sección de lavadora

La barra de navegación posibilita al usuario desplazarse a las funcionalidades que se consideran más necesarias desde cualquier punto de la aplicación como son el menú principal, las combinaciones, la creación de combinaciones y la lavadora.



Ilustración 30-Menú lateral

5. Pruebas

Para asegurar que el sistema software resuelve el problema inicialmente planteado, se realiza la verificación y validación del software mediante revisiones y pruebas. En este apartado nos centraremos en las pruebas software cuyo objetivo es comprobar dinámicamente el comportamiento de un programa para un conjunto finito de casos de prueba comparándolo con el comportamiento esperado para evaluar su calidad y mejorarla, identificando posibles defectos y problemas.

5.1. Pruebas unitarias

Mediante este tipo de pruebas se comprueba el funcionamiento aislado de cada uno de los módulos de nuestro sistema, asegurando que cada módulo funcione correctamente.

Para ello, se ha utilizado la herramienta Android Test Case que proporciona Android Studio. A continuación se muestra un pequeño ejemplo del uso de dicho programa y el código utilizado para verificar el correcto funcionamiento del módulo analizado. Para ello se crea una combinación nueva y dos tipos de categorías y se comprueba que estos datos son correctos. En el caso de las categorías se comprueba que existen dos tipos de categorías y una de ellas es Sport y que al introducir la combinación creada en la base de datos se almacenan los datos correctamente.

```
@Test
public void testDatosCombinacion(){
    assertTrue(combinacion.getnVeces()==1);
    assertTrue(combinacion.getPuntuacion()==2);
    assertTrue(combinacion.getTemporada()==1);
    assertTrue(combinacion.getCategoría() ==1);
    assertTrue(combinacion.getUso()==1);

}

@Test
public void testCategoria(){

    Categoría c=null;
    for (int i=0;i<listaCategorías.size();i++){
        if(listaCategorías.get(i).getTipo().equals("Sport")){
            c=listaCategorías.get(i);
        }
    }
    assertTrue(c.getTipo().equals("Sport"));
    assertTrue(listaCategorías.size()==2);
}
```

Ilustración 31-Fragmento de código de una de las pruebas unitarias

5.2. Pruebas de integración

Al finalizar la evaluación de las pruebas unitarias, se continua con las pruebas de integración. Se combinan gradualmente los módulos individuales del proyecto para comprobar que el funcionamiento conjunto de los componentes del sistema se realiza correctamente.

Debido a que la metodología usada para llevar este proyecto adelante ha sido la incremental iterativa, una vez finalizada cada iteración se ha comprobado el funcionamiento de todos los componentes creados en ese momento de manera conjunta. Por ello, al finalizar la aplicación, se comprueba que al ejecutarla completamente no existe ningún error al interactuar con el sistema.

Esta prueba se realiza mediante el entorno de Android Studio que posee un módulo dentro del programa que permite ver los errores que ocurren a lo largo de la ejecución de la aplicación.

5.3. Pruebas de sistema

En este tipo de pruebas verificamos que la forma en la que se comporta nuestro sistema software es el correcto, adecuado y conforme a los requisitos del sistema especificados en el análisis de requisitos no funcionales.

5.3.1. Pruebas de rendimiento

Durante el periodo de implementación del sistema, se han ido marcando varios periodos de pruebas, para verificar que los tiempos de respuesta de la aplicación no fueran desmesurados y en el caso de que los resultados no fueran favorables, solventar los problemas de forma temprana.

Para la realización de dichas pruebas se ha medido el tiempo de respuesta de distintas funcionalidades. Algunas de ellas son la duración del arranque por primera vez y en siguientes accesos, el tiempo de respuesta ante la muestra de combinaciones o el periodo invertido en cargar las prendas.

Se han utilizado los dispositivos LG G4, Xiaomi mi4, Huawei G620 y Nexus 4 desde el simulador de Android Studio.

	LG G4	Xiaomi MI4	Huawei Ascend G620s	LG Nexus 4
Procesador	2,5 Ghz	2.5Ghz	1,2 Ghz	1,5Ghz
Memoria RAM	3GB	3GB	1GB	2GB
Sistema operativo	Android 5.1	Android 4.4.4	Android 4.4.4	Android 5.0
Pantalla	5,5 pulgadas	5 pulgadas	5 pulgadas	4,7 pulgadas

A continuación se muestran los tiempos invertidos en realizar las funcionalidades anteriores desde cada dispositivo.

	LG G4	Xiaomi MI4	Huawei Ascend G620s	LG Nexus 4
Primer arranque sin prendas	0,03 segundos	0,02 segundos	0,12 segundos	0,07 segundos
Arranque con 20 prendas	0,06 segundos	0,07 segundos	0,09 segundos	0,07 segundos
Creación de combinaciones	0,04 segundos	0,04 segundos	0,08 segundos	0,05 segundos
Mostrar combinaciones aleatorias	0,05 segundos	0,05 segundos	0,06 segundos	0,05 segundos

Cabe mencionar que el tiempo de carga se incrementa debido al aumento de datos a medida que se utiliza la aplicación. Aún así, los resultados han sido satisfactorios, consiguiendo que se pueda visualizar y trabajar de forma fluida en la aplicación.

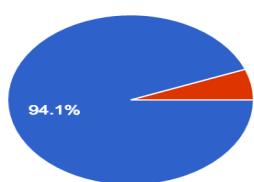
5.3.2. Pruebas de usabilidad

Se proporciona la aplicación a distintos grupos de usuarios y se observa como se desenvuelven con ella en un principio y el tiempo que tardan en

utilizarla de forma fluida posteriormente. Paralelamente se realiza una encuesta online realizando preguntas sobre aspectos visuales de la aplicación, si les ha parecido sencillo interactuar con ella, o los aspectos que mejoraría.

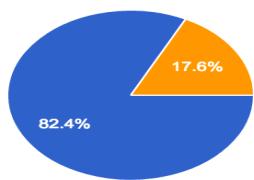
Así mismo, se cita posteriormente al mismo grupo de usuarios para comprobar si el usuario recuerda fácilmente el uso de la aplicación.

¿Mejoraría esta aplicación tu forma de organizarte?



Si	16	94.1%
No	1	5.9%

¿Te parece una interfaz amigable y fácil de utilizar?



Si	14	82.4%
No	0	0%
Regular	3	17.6%

Ilustración 32- Respuesta encuesta opinión

Ante esta prueba los resultados han sido favorables, indicando que a la gran mayoría la aplicación les ha parecido fácil, amigable e intuitiva.

5.4. Pruebas de usuario

Durante el periodo de implementación les ofrecemos a los clientes potenciales las funcionalidades que la aplicación posee y dejamos que la utilicen. Se pide que evalúen la aplicación y muestren sus opiniones acerca de si supera o cubre sus expectativas como para poder comprarla en un futuro.

Se realiza un cuestionario sobre diferentes grupos de usuarios, donde se realizan preguntas sobre los requisitos de nuestra aplicación, como si nuestro sistema es fácil de entender, accesible, etc..

6. Conclusiones y trabajos futuros

Al finalizar el trabajo es necesario valorar si todos los objetivos marcados al inicio del proyecto fueron cumplidos satisfactoriamente. El objetivo principal era desarrollar un aplicación móvil nativa que permitiese obtener y crear combinaciones de ropa. Respecto a este requisito, se puede afirmar que ha sido completado exitosamente.

El sistema constituye una aplicación nativa, habiendo sido creado exclusivamente para el sistema operativo Android. Se ha diseñado y creado su interfaz para ser ejecutada en cualquier dispositivo con dicho sistema operativo que tenga una versión igual o superior a la 4.0 e indistintamente del tamaño de la pantalla. Además se ha incidido en realizar una interfaz sencilla y fácil de usar.

Se han creado dos algoritmos para la visualización de las combinaciones, uno de ellos genera aleatoriamente prendas según en que parte del cuerpo están colocadas, como el torso, las piernas o los pies y las combina produciendo un conjunto que el usuario ve. El segundo de ellos, elige aleatoriamente una prenda y utiliza el color que esta posee para buscar otras prendas alojadas en la base de datos que se asemejen a dicho tono y las devuelve como una combinación. Además se ha creado un apartado para que el usuario elabore sus conjuntos por si no esta satisfecho con los algoritmos anteriormente descritos y que podrá visualizar posteriormente.

Respecto a las dificultades, se considera que la mayor dificultad del sistema fue la detección del color de cada imagen y el diseño del algoritmo que generara combinaciones basadas en colores análogos.

En los objetivos descritos en la introducción, se mencionaba la utilización de los conocimientos adquiridos a lo largo del periodo universitario, estos han sido empleados para la elaboración de forma estructurada de la memoria del proyecto y han servido como pautas para el correcto desarrollo software mediante una metodología incremental iterativa y una arquitectura en tres capas.

6.1. Conclusiones personales

Decidí realizar este trabajo porque me parecía una idea original, novedosa y diferente a lo que había realizado durante mis estudios. En un primer momento me pareció bastante tedioso de realizar ya que nunca había desarrollado nada parecido y la herramienta era desconocida para mi, lo que me causaba cierto respeto. Poco a poco el entorno de trabajo se fue haciendo más cercano y los problemas que en un primer momento veía imposibles de solventar, fueron resolviéndose favorablemente. Finalmente realice el trabajo con los objetivos marcados.

Después de haberlo acabado considero que he aprendido a ser

autosuficiente, autodidacta y constante en el trabajo. He conseguido superar los miedos a lo desconocido y sacar el trabajo adelante. Para ello me ha sido de gran ayuda los conocimientos obtenidos a lo largo de la titulación, que han supuesto las bases para la realización de un trabajo estructurado y coherente.

6.2. Trabajos futuros

A continuación se exponen las funcionalidades de nuestro sistema que se consideran ampliables o mejorables:

- Ampliación de los tipos de generación de combinaciones siendo interesante el enlace de la aplicación con algún *plugin* que permita el acceso a la ubicación y el tiempo y así poder devolver las combinaciones en función del clima en el que se encuentre cada usuario. Del mismo modo, se podría realizar conjuntos basados en acciones de cada usuario o filtrando los conjuntos por categorías.
- Respecto al almacenamiento de las prendas, sería recomendable que el sistema recortara la fotografía automáticamente, permitiendo ser almacenada sin el color de fondo de la imagen. O utilizar un marco en la captura de la imagen que delimitara los bordes dentro de los cuales debe ir la prenda.
- Desarrollo de la aplicación dirigido a otras plataformas móviles como IOS, Windows Phone o BlackBerry.

7. Bibliografía

- [1] Android Developers. Recuperado el 10 de 08 de 2015, de Developers: <https://developer.android.com/intl/es/sdk/index.html>
- [2] Fundación Telefónica. (2015). *La Sociedad de la Información en España 2014*. Barcelona: Ariel, S.A.
- [3] Gironés, J. T. (2013). *El gran libro de Android*. Barcelona: Marcombo S.A.
- [4] Guinea, E. G. (27 de 09 de 2009). Toniface. Recuperado el 06 de 05 de 2015, de Toniface: <http://www.toniface.es/tiempo-que-pasamos-durmiendo-y-otras-curiosidades-de-la-vida-cotidiana/>
- [5] Just in Mind. (2014). Recuperado el 20 de 04 de 2015, de <http://www.justinmind.com>
- [6] Larman, C. (2008). *UML y Patrones*. (B. M. Valle, Trad.) Madrid: Pearson.
- [7] Microsoft Project. (Microsoft, Productor) Recuperado el 01 de 04 de 2015, de <https://products.office.com/es-es/project/project-and-portfolio-management-software>
- [8] No Magic. (I. No Magic, Productor, & Wyoming Corporation) Recuperado el 20 de 05 de 2015, de www.nomagic.com
- [9] Somerville, I. (2011). *Ingeniería del Software*. (L. M. Castillo, Ed., & V. C. Olgún, Trad.) Pearson.
- [10] SQLite. Recuperado el 21 de 05 de 2015, de <https://www.sqlite.org/>