2022/23

MEMORIA PROYECTO "MYPETS"

2º DESARROLLO APLICACIONES MULTIPLATAFORMA
DAVID GONZÁLEZ CABRERO

Índice

Índice de imágenes

Índice de tablas

1. Introducción.

1.1. Resumen.

Este es el documento de memoria para el proyecto final del Ciclo Formativo de Grado Superior de “Desarrollo de Aplicaciones Multiplataforma”. En él, encontraremos diversas secciones, donde se detallan: la fase de estudio de mercado, la fase de planificación, la fase de desarrollo y la puesta en marcha.

El proyecto centra su esfuerzo en la creación de una aplicación móvil, comenzando por un análisis básico del mercado. Realizado este punto, pasamos a la etapa de planificación y diseño y como punto final, a la instalación del “software” en los dispositivos y la redacción de la documentación.

Este “software” únicamente se ha desarrollado para el sistema “Android”. Los lenguajes empleados son: “SQL”, para las consultas en “SQLite” y “Java”, para la programación orientada a objetos. El “IDE” empleado ha sido “Android Studio” y el terminal de pruebas ha sido un “PIXEL 4”.

“MYPETS” es una aplicación de la que vamos a poder disfrutar en nuestro terminal móvil: se trata de poder contar con una agenda virtual para nuestras mascotas. Por ejemplo, podremos llevar un control del peso de las mismas y ver su evolución a lo largo del tiempo, así como verificar el ejercicio que realizan y las valoraciones que les podremos otorgar. Además, podremos anotar citas: para la peluquería, baños, visitas al veterinario o cualquier otro tipo de evento relacionado con ellas.

1.2. Integrantes del equipo.

Este proyecto está realizado íntegramente por una única persona, habiéndose adquirido conocimientos de fuentes externas en momentos puntuales.

El único integrante del equipo es David González Cabrero.

1.3. Justificación.

Un proyecto de fin de grado se justifica por sí solo. Constituye la parte más importante para todo técnico: sintetizar lo aprendido hasta la fecha, para convertir una idea inicial en un producto tangible. El producto se ha estudiado, planificado, analizado y diseñado para su posterior implementación e implantación.

Este proyecto surge a raíz de la necesidad de llevar un control sobre todas las actividades que realiza una mascota.

El producto ha sido desarrollado utilizando “Android Studio Dolphin 2021.3.1 Patch 1”.

1.4. Método utilizado.

El método utilizado para la implementación ha sido un ciclo de vida en cascada.

Dicho ciclo está compuesto por las siguientes etapas:

1. Análisis previo y planificación.
2. Análisis de requisitos.
3. Diseño.
4. Implementación.
5. Pruebas.
6. Entrega del proyecto y presentación.

1.5. Análisis previo.

1.5.1. Objetivos de mercado.

Mi proyecto va a consistir en crear una pequeña agenda para nuestras mascotas, donde podremos almacenar sus citas concertadas con los veterinarios, sus vacunas, revisiones y demás eventos relacionados que tengamos reservados exclusivamente para ellas.

No se trata de una aplicación que requiera conexión a internet, ya que no interactuaremos con otros usuarios en la misma, ni con software terceros.

Esta aplicación va dirigida, en general, a todo tipo de público. Actualmente, en España, existen aproximadamente unos 29 millones de mascotas. Esto quiere decir que existe un gran público objetivo y que la aplicación estará disponible para personas de todas las edades. Se ofrecerá una interfaz sencilla, intuitiva y de fácil manejo.

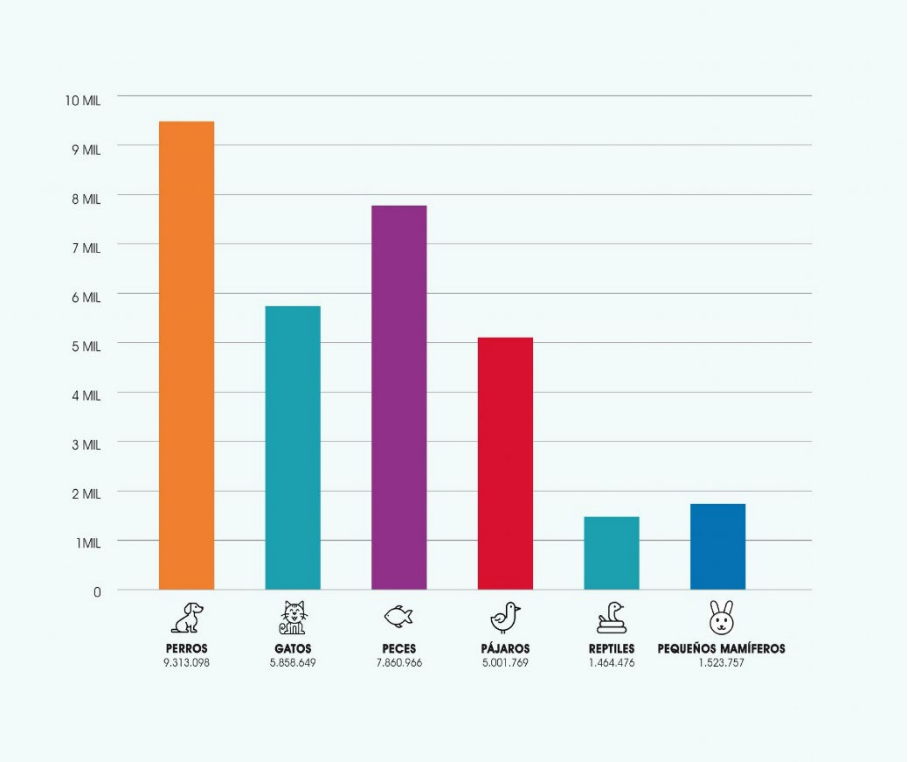

Ilustración 1: Gráfica de mascotas en España

En principio, la aplicación será desarrollada únicamente para los dispositivos móviles de Android. Tomando como referencia el gráfico inferior, observaremos cómo más del 80% de los dispositivos móviles del mercado son Android. No se descarta en un futuro poder implementarla para iOS, pero supone un sobrecoste no asumible en este momento del desarrollo.

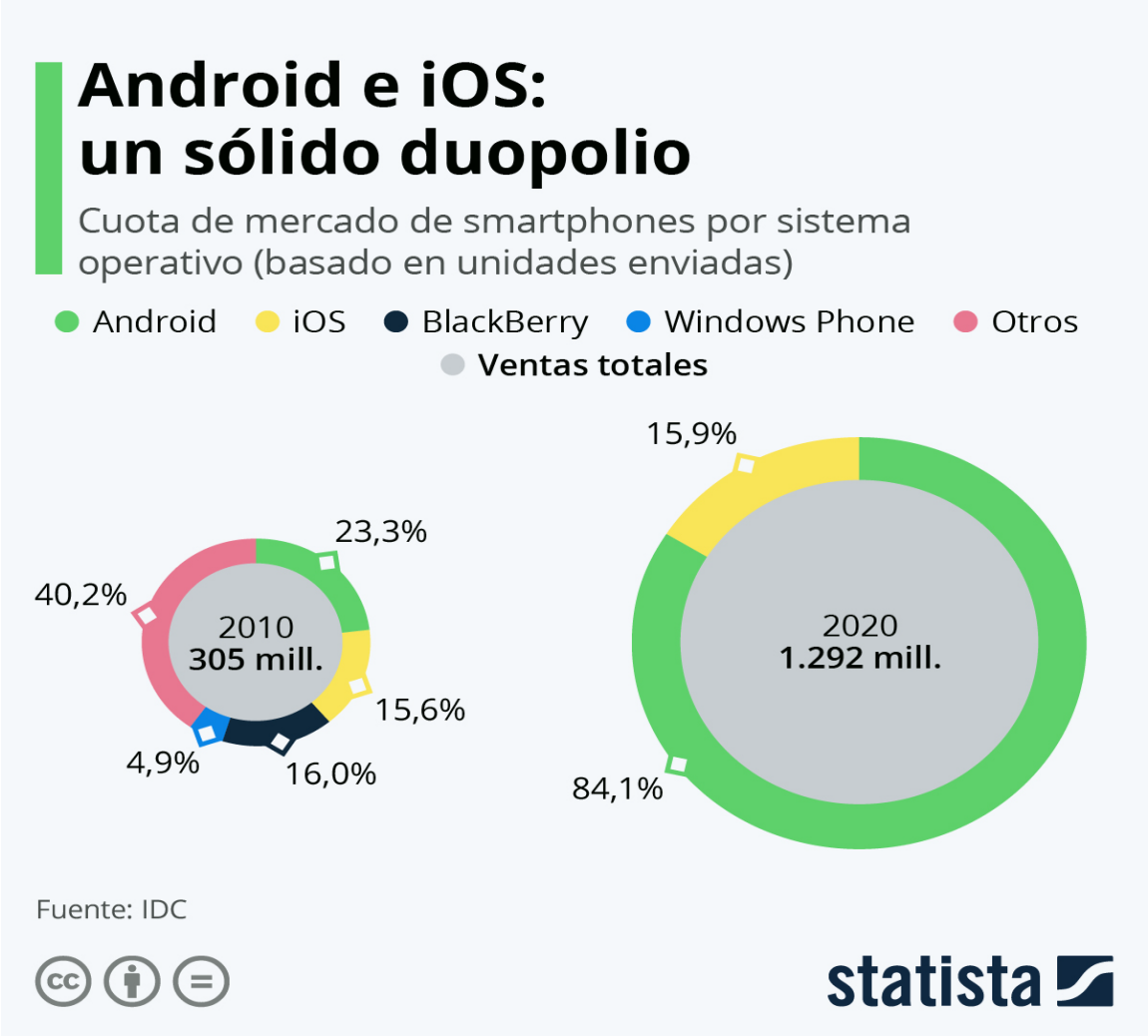

Ilustración 2: Cuota de mercado por S.O.

Una vez indicado el sistema sobre el que se desarrollará la aplicación, solamente quedará indicar su versión. La aplicación podrá funcionar en cualquier dispositivo **Android** con un nivel de API igual o superior a 21, que correspondería a la versión **Android 5.0** o superiores, la cual poseen más de un 85% de terminales móviles.

1.5.2. Situación actual.

Actualmente, el mercado de las aplicaciones móviles está en alza y resulta probable que se mantenga así durante muchos tiempo. Como es lógico, no todos los sectores ni todas

las aplicaciones funcionan igual, ni van dirigidas al mismo público objetivo. A continuación, observemos el siguiente gráfico sobre la evolución de las aplicaciones de pago a nivel mundial.

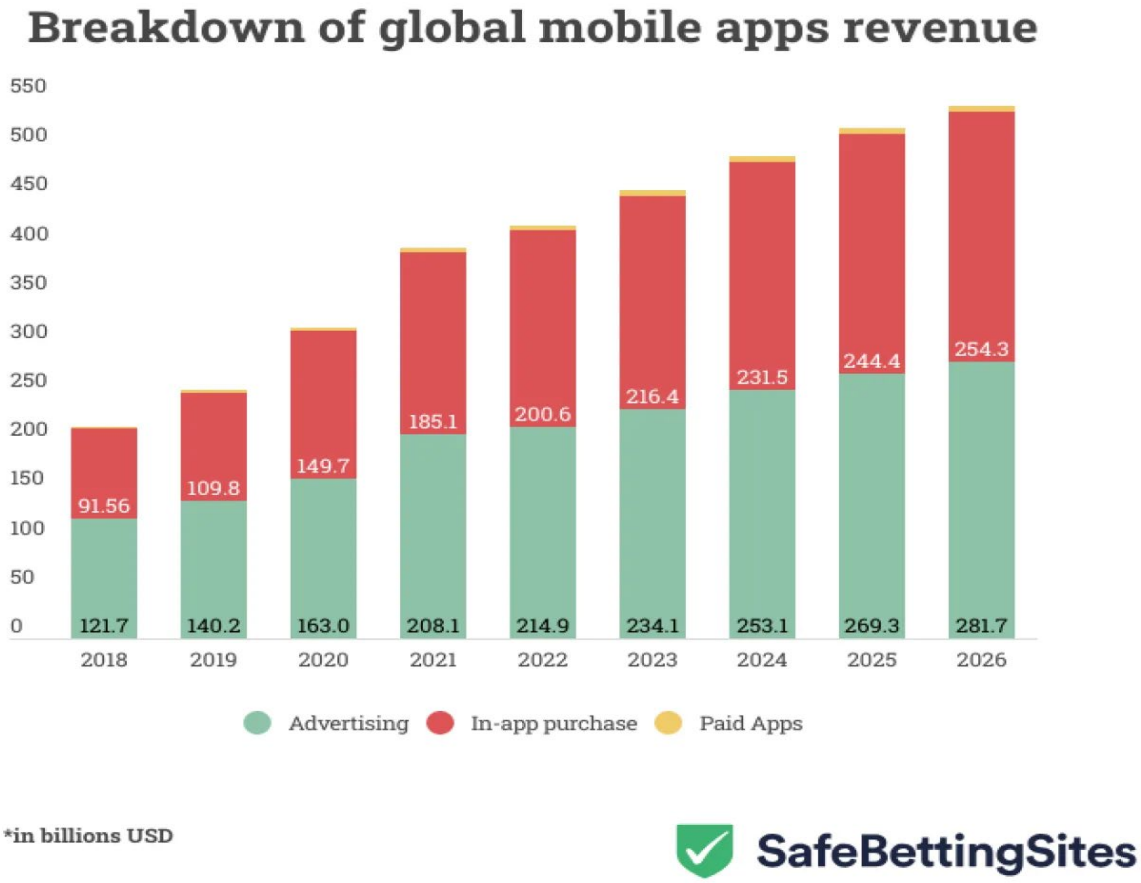

Ilustración 3: Gráfico evolución aplicaciones de pago

Las ventajas de los dispositivos móviles son varias, pero por ejemplo, en la aplicación de este anteproyecto, es una muy simple: si pierdes el dispositivo, no pierdes la agenda o la información sobre tu mascota. Siempre puedes tener copias de seguridad o un respaldo de la información en otra ubicación. Se pueden programar recordatorios, etc.

1.6. Planificación.

He decidido dividir el proyecto en tres etapas. Dentro de cada etapa tenemos diferentes actividades, con hitos que marcan el comienzo de la siguiente actividad, como podremos ver más adelante.

1.6.1. Fase de diseño.

Esta fase corresponde al período comprendido entre el 11 de octubre y el 26 de octubre de 2022. En esta etapa, seleccionamos las ideas a partir de las cuales comienza el desarrollo de la aplicación. Se han realizado algunas de las siguientes actividades:

- Analizar e instalar el “software” a emplear para el desarrollo del proyecto:
  - “Android Studio”.
  - “Microsoft Office”.
  - “Microsoft Proyect”.
  - “Gimp”.
- Se ha desarrollado el plan de trabajo.

1.6.2. Fase de desarrollo y pruebas.

Esta fase corresponde al período comprendido entre el 26 de octubre y el 25 de noviembre de 2022. En esta fase, se ha codificado la aplicación. Se han realizado algunas de las siguientes actividades:

- Desarrollo de vistas.
- Desarrollo de “cardviews”.
- Desarrollo de traducciones.
- Desarrollo de métodos.

- Conexiones con base de datos.
- Tratamiento de errores.
- Control de errores.
- Creación de entidades.

1.6.3. Fase de documentación y presentación.

Esta fase corresponde al período comprendido entre el 25 de noviembre y el 19 de diciembre de 2022. En dicha fase se realizarán las siguientes actividades:

- Confeccionar la memoria del proyecto.
- Realizar la presentación del mismo.
- “Demo” en vivo de la aplicación.

1.7. Entorno de desarrollo.

Para el desarrollo del proyecto final del Ciclo, se ha utilizado un equipo con las siguientes prestaciones:

- Procesador Intel I5 11th Gen.
- 32Gb Ram DDR4.
- 1Tb SSD.
- Gráfica RTX3060 Ti.
- Sistema operativo: Windows 10 Pro.

A continuación, se detalla el “software” necesario para realizar dicho proyecto.

- “Android Studio: Dolphin” como entorno integrado de desarrollo(IDE).
- “Gimp”: para el desarrollo de imágenes e iconos.
- “MEmu”: para la emulación de dispositivos móviles.
- “Microsoft Word”: para la elaboración de la memoria.

- “Microsoft Project”: para la elaboración del diagrama Gantt.

1.8. Diagrama de Gantt inicial.

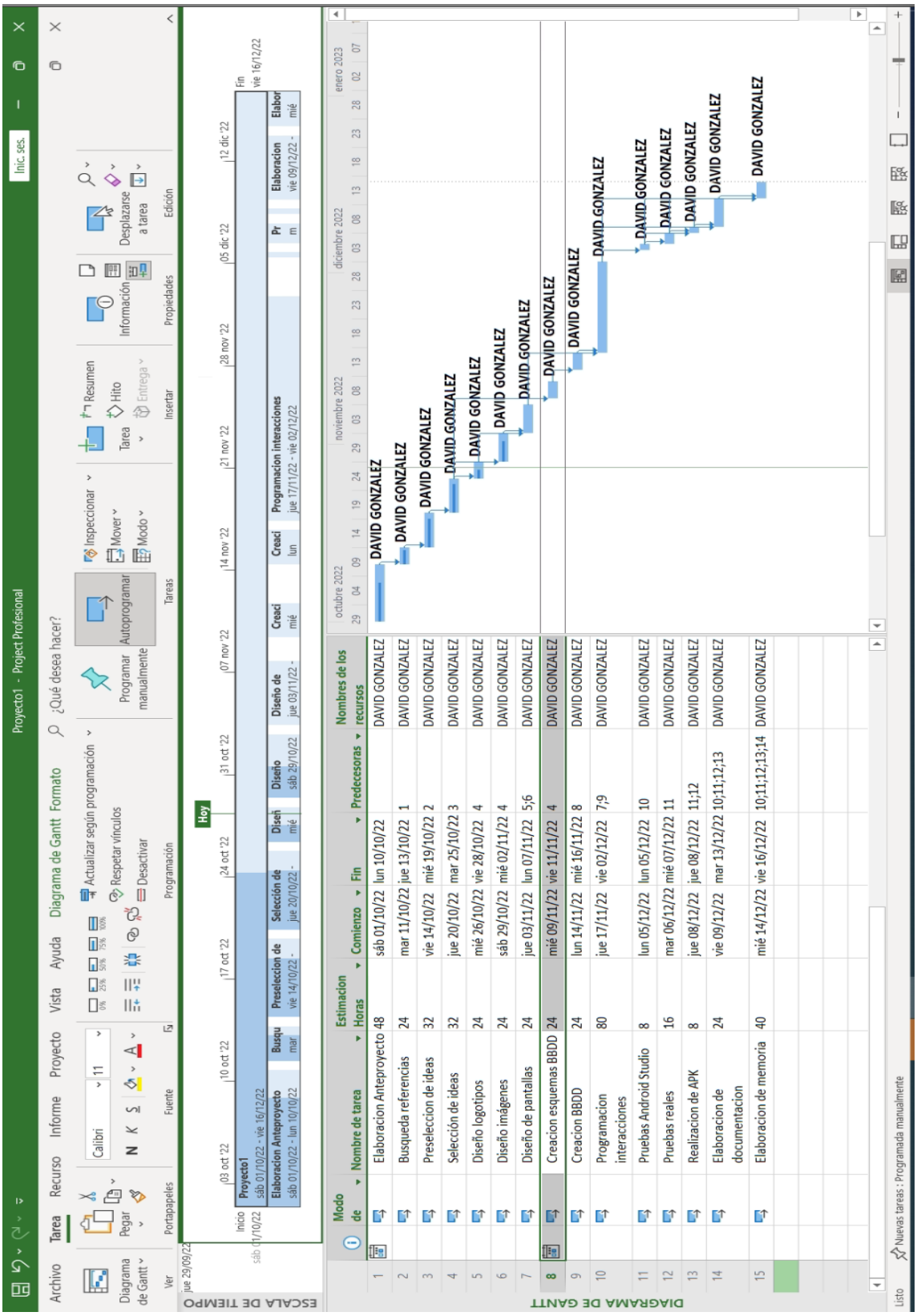

Ilustración 4: Diagrama de Gantt inicial.

1.9. Diagrama de Gantt final.

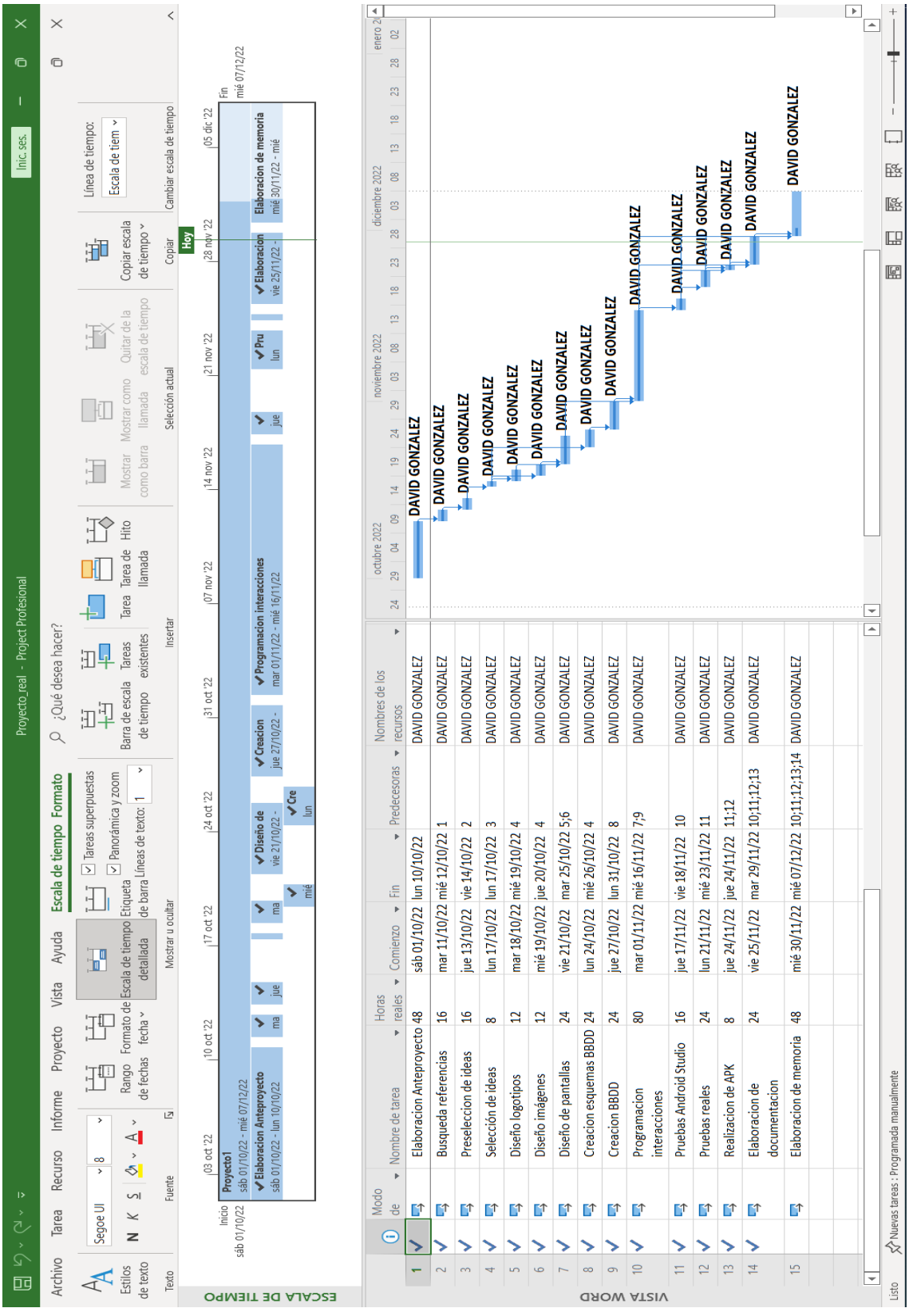

Ilustración 5: Diagrama de Gantt final.

1.10. Costes estimados del proyecto.

Realizando una valoración de los costes de este proyecto, me he basado en la estimación inicial de horas, tomando como referencia un precio de 25€ por hora de trabajo.

A continuación, se adjunta la tabla de estimación de costes:

Tabla 1: Costes estimados del proyecto

Tarea	Estimación Horas	Coste hora	Subtotal
Elaboración anteproyecto	48 h	25 €	1200 €
Búsqueda de referencias	24 h	25 €	600 €
Preselección de ideas	32 h	25 €	800 €
Selección de ideas	32 h	25 €	800 €
Diseño de logotipos	24 h	25 €	600 €
Diseño de imágenes	24 h	25 €	600 €
Diseño de pantallas	24 h	25 €	600 €
Creación de esquemas de “BBDD”	24 h	25 €	600 €
Creación de “BBDD”	24 h	25 €	600 €
Programación de interacciones	80 h	25 €	2000 €
Pruebas “Android Studio”	8 h	25 €	200 €
Pruebas reales	16 h	25 €	400 €
Realización de los “APK”	8 h	25 €	200 €
Elaboración de documentación	24 h	25 €	600 €
Elaboración de memoria	40 h	25 €	1000 €
		TOTAL	10800 €

1.11. Costes reales del proyecto.

Una vez finalizado el proyecto y habiendo obtenido las horas reales de duración del mismo, se ha confeccionado la tabla de costes definitiva. He fijado la hora de trabajo en 25€.

A continuación, se adjunta la tabla de costes reales:

Tabla 2: Costes reales del proyecto

Tarea	Horas Reales	Coste hora	Subtotal
Elaboración anteproyecto	48 h	25 €	1200 €
Búsqueda de referencias	16 h	25 €	400 €
Preselección de ideas	16 h	25 €	400 €
Selección de ideas	8 h	25 €	200 €
Diseño de logotipos	12 h	25 €	300 €
Diseño de imágenes	12 h	25 €	300 €
Diseño de pantallas	24 h	25 €	600 €
Creación de esquemas de “BBDD”	24 h	25 €	600 €
Creación de “BBDD”	24 h	25 €	600 €
Programación de interacciones	80 h	25 €	2000 €
Pruebas “Android Studio”	16 h	25 €	400 €
Pruebas reales	24 h	25 €	600 €
Realización de los “APK”	8 h	25 €	200 €
Elaboración de documentación	24 h	25 €	600 €
Elaboración de memoria	48 h	25 €	1200 €
		TOTAL	9600 €

2. Análisis.

2.1. Requisitos.

2.1.1. Requisitos no funcionales.

Son aquellos que consisten en una característica requerida del sistema, del proceso de desarrollo, o de cualquier otro aspecto del mismo. Podemos definir algunos como:

- Sistema operativo: La “app” solo funcionará sobre “Android”, a partir del “API 24”, para ser más exactos.
- Interfaz de usuario: Sencilla, intuitiva y fácil de manejar. Todas las fichas o pestañas son similares, para facilitar el aprendizaje por parte del usuario.
- Paleta de colores: Se adjunta la paleta de colores empleada (sencilla y sin colores estridentes, que puedan resultar molestos para la vista). No se ha empleado la paleta entera.

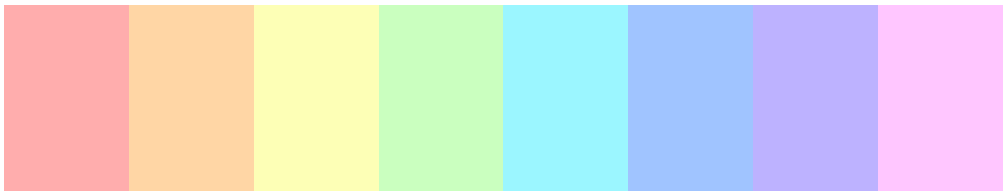

Ilustración 6: Paleta de colores utilizada.

2.1.2. Requisitos funcionales.

Son aquellos que definen una función del “software” o de alguno de sus componentes. Una función es descrita como un conjunto de entradas, comportamientos y salidas. Podemos definir algunos como:

- Creación de una o varias mascotas: podemos, desde la aplicación, manejar una o varias mascotas, cada una con sus propias citas y eventos.
- Asignación de peso por día y para cada una de las mascotas: para cada mascota, podremos asignar un valor diario de peso y lo veremos individualmente en cada una de ellas.
- Asignación de eventos por fecha, hora y mascota: podemos gestionar citas independientes para cada mascota, marcando fecha y hora.
- Asignación de cita veterinaria por fecha, hora y mascota: podemos gestionar citas independientes para cada mascota, marcando fecha y hora.
- Asignación de entrenamientos por fecha y mascota: podemos asignar una actividad deportiva individual a cada mascota.
- Eliminación de mascotas, eventos, citas médicas, peso y entrenamientos: podemos eliminar cada elemento de cada mascota de manera independiente y a su vez, podemos eliminar también cada mascota, si así lo deseamos.
- Conexión con la base de datos: se realiza una conexión a la base de datos nativa de “Android SQLite” para la inserción, consulta y modificación de los datos.

2.2. Escenarios y componentes.

El escenario del proyecto es una aplicación móvil para la gestión de la agenda de nuestras mascotas. Los “actores” del proyecto son dos: el usuario de la aplicación y la propia aplicación.

2.3. Guión de Usuario.

Se encargará de gestionar las mascotas(altas, bajas y modificaciones). Además, también podrá generar dentro de cada una de ellas sus propias citas, diario de pesos, actividades realizadas y consultas veterinarias (pudiendo crearlas, modificarlas o eliminarlas).

2.3.1. Guión de la Aplicación.

Se encargará de realizar la carga de todos los datos de las mascotas, desde la base de datos. También será la encargada de mostrar los datos, en función del apartado donde nos encontremos.

2.3.2. Casos de uso.

A continuación, adjuntamos los casos de uso de la aplicación:

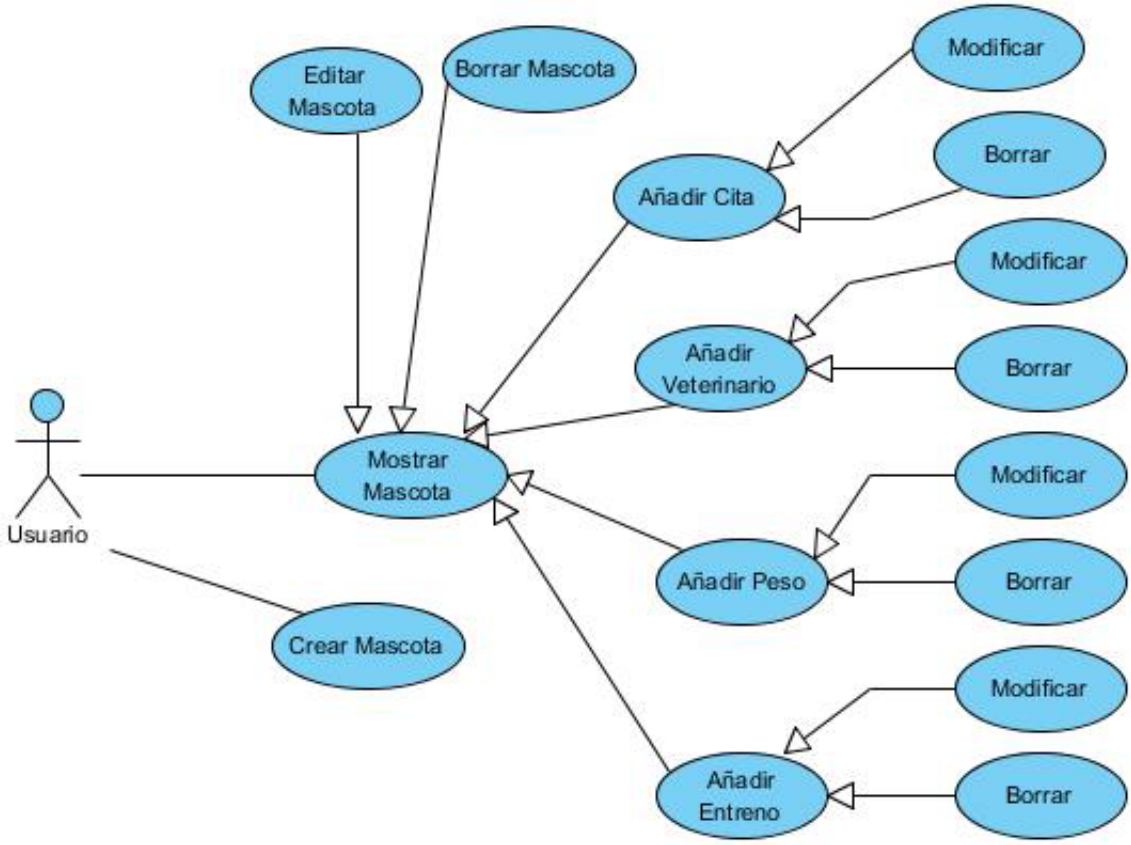

Ilustración 7: Casos de uso.

2.3.3. Especificaciones de casos de uso.

A continuación, se detalla el conjunto de casos de uso que intervienen en “MYPETS”.

2.3.4. Caso de uso: Crear mascota.

Resumen de la funcionalidad: Permite crear una mascota para su seguimiento.

Actores: **Usuario**.

Casos de uso relacionados: Ninguno.

Precondición: Ninguna.

Postcondición: Se genera la mascota para su posterior tratamiento.

Proceso normal principal:

1. Se inicia la aplicación.
2. Se conecta a la base de datos.
3. Se genera la inserción en la base de datos.
4. Se devuelve el resultado positivo de la inserción.

Alternativas de proceso y excepciones:

1) Se devuelve un error de creación de mascota.

2.3.5. Caso de uso: Mostrar mascota.

Resumen de la funcionalidad: Permite mostrar una mascota para su seguimiento.

Actores: **Usuario**.

Casos de uso relacionados: Ninguno.

Precondición: Tener una mascota creada.

Postcondición: Se muestra la información de la mascota.

Proceso normal principal:

1. Se inicia la aplicación.

2. Se conecta a la base de datos.
3. Se muestra la información de la mascota.

Alternativas de proceso y excepciones:

1) Se devuelve un error de consulta de mascota.

2.3.6. Caso de uso: Editar mascota.

Resumen de la funcionalidad: Permite editar una mascota, creada previamente.

Actores: **Usuario**.

Casos de uso relacionados: Ninguno.

Precondición: Tener una mascota creada.

Postcondición: Se modifica la información de la mascota.

Proceso normal principal:

1. Se inicia la aplicación.
2. Se conecta a la base de datos.
3. Se muestra la información de la mascota.
4. Se realiza la modificación de los datos.
5. Se devuelve el resultado positivo de la edición.

Alternativas de proceso y excepciones:

1) Se devuelve un error de modificación de mascota.

2.3.7. Caso de uso: Borrar mascota.

Resumen de la funcionalidad: Permite borrar una mascota, creada previamente.

Actores: **Usuario**.

Casos de uso relacionados: Ninguno.

Precondición: Tener una mascota creada.

Postcondición: Se borra la información de la mascota y desaparece de la base de datos.

Proceso normal principal:

1. Se inicia la aplicación.
2. Se conecta a la base de datos.
3. Se muestra la información de la mascota.
4. Se realiza el borrado de los datos.
5. Se devuelve el resultado positivo de la eliminación.

Alternativas de proceso y excepciones:

1) Se devuelve un error de borrado de mascota.

2.3.8. Caso de uso: Añadir cita.

Resumen de la funcionalidad: Permite añadir citas, para una mascota creada previamente.

Actores: **Usuario**.

Casos de uso relacionados: Ninguno.

Precondición: Tener una mascota creada.

Postcondición: Se crea una cita para una fecha y hora concretas.

Proceso normal principal:

1. Se inicia la aplicación.
2. Se conecta a la base de datos.
3. Se muestra la información de la mascota.

4. Se realiza la creación de la cita, para la mascota en cuestión.
5. Se devuelve el resultado positivo de la creación.

Alternativas de proceso y excepciones:

1) Se devuelve un error de creación de la cita de la mascota.

2.3.9. Caso de uso: Modificar cita.

Resumen de la funcionalidad: Permite modificar las citas creadas previamente para una mascota.

Actores: **Usuario**.

Casos de uso relacionados: Ninguno.

Precondición: Tener una mascota creada, con citas previstas.

Postcondición: Se modifica una cita, creada previamente.

Proceso normal principal:

1. Se inicia la aplicación.
2. Se conecta a la base de datos.
3. Se muestra la información de la mascota.
4. Se realiza la modificación de la cita en cuestión.
5. Se devuelve el resultado positivo de la modificación.

Alternativas de proceso y excepciones:

1) Se devuelve un error de modificación de la cita de la mascota.

2.3.10. Caso de uso: Borrar cita.

Resumen de la funcionalidad: Permite borrar las citas, creadas previamente para una mascota.

Actores: **Usuario**.

Casos de uso relacionados: Ninguno.

Precondición: Tener una mascota creada, con citas asignadas.

Postcondición: Se borra una cita, creada previamente.

Proceso normal principal:

1. Se inicia la aplicación.
2. Se conecta a la base de datos.
3. Se muestra la información de la mascota.
4. Se realiza el borrado de la cita en cuestión.
5. Se devuelve el resultado positivo de la eliminación.

Alternativas de proceso y excepciones:

1) Se devuelve un error de borrado de la cita de la mascota.

2.3.11. Caso de uso: Añadir cita veterinaria.

Resumen de la funcionalidad: Permite añadir citas veterinarias, para una mascota creada previamente.

Actores: **Usuario**.

Casos de uso relacionados: Ninguno.

Precondición: Tener una mascota creada.

Postcondición: Se crea una cita veterinaria, reservada en una fecha y hora concretas para la mascota.

Proceso normal principal:

1. Se inicia la aplicación.
2. Se conecta a la base de datos.
3. Se muestra la información de la mascota.
4. Se realiza la creación de la cita veterinaria para la mascota en cuestión.
5. Se devuelve el resultado positivo de la creación.

Alternativas de proceso y excepciones:

1) Se devuelve un error de creación de la cita veterinaria de la mascota.

2.3.12. Caso de uso: Modificar cita veterinaria.

Resumen de la funcionalidad: Permite modificar las citas veterinarias reservadas, para una mascota creada previamente.

Actores: **Usuario**.

Casos de uso relacionados: Ninguno.

Precondición: Tener una mascota creada previamente, con citas veterinarias asignadas.

Postcondición: Se modifica una cita veterinaria, creada previamente.

Proceso normal principal:

1. Se inicia la aplicación.
2. Se conecta a la base de datos.
3. Se muestra la información de la mascota.
4. Se realiza la modificación de la cita veterinaria en cuestión.
5. Se devuelve el resultado positivo de la modificación.

Alternativas de proceso y excepciones:

1) Se devuelve un error de modificación de la cita veterinaria de la mascota.

2.3.13. Caso de uso: Borrar cita veterinaria.

Resumen de la funcionalidad: Permite borrar las citas veterinarias, asignadas previamente a una mascota.

Actores: **Usuario**.

Casos de uso relacionados: Ninguno.

Precondición: Tener una mascota creada, con citas veterinarias asignadas.

Postcondición: Se borra una cita veterinaria, asignada previamente.

Proceso normal principal:

1. Se inicia la aplicación.
2. Se conecta a la base de datos.
3. Se muestra la información de la mascota.
4. Se realiza el borrado de la cita veterinaria en cuestión.
5. Se devuelve el resultado positivo de la eliminación.

Alternativas de proceso y excepciones:

1) Se devuelve un error de borrado de la cita veterinaria asignada a la mascota.

2.3.14. Caso de uso: Añadir peso.

Resumen de la funcionalidad: Permite añadir valores de pesaje, para una mascota creada previamente.

Actores: **Usuario**.

Casos de uso relacionados: Ninguno.

Precondición: Tener una mascota, creada previamente.

Postcondición: Se crea un valor de pesaje en una fecha concreta, para la mascota elegida.

Proceso normal principal:

1. Se inicia la aplicación.
2. Se conecta a la base de datos.
3. Se muestra la información de la mascota.
4. Se realiza el control de peso de la mascota en cuestión.
5. Se devuelve el resultado positivo de la creación.

Alternativas de proceso y excepciones:

1) Se devuelve un error de creación de la medida de pesaje de la mascota.

2.3.15. Caso de uso: Modificar peso.

Resumen de la funcionalidad: Permite modificar los pesajes asignados previamente a una mascota.

Actores: **Usuario**.

Casos de uso relacionados: Ninguno.

Precondición: Tener una mascota creada, con medidas de peso asignadas.

Postcondición: Se modifica un pesaje, creado previamente.

Proceso normal principal:

1. Se inicia la aplicación.
2. Se conecta a la base de datos.
3. Se muestra la información de la mascota.
4. Se realiza la modificación del pesaje en cuestión.
5. Se devuelve el resultado positivo de la modificación.

Alternativas de proceso y excepciones:

1) Se devuelve un error de modificación del pesaje de la mascota elegida.

2.3.16. Caso de uso: Borrar peso.

Resumen de la funcionalidad: Permite borrar los pesajes, previamente asignados a una mascota.

Actores: **Usuario**.

Casos de uso relacionados: Ninguno.

Precondición: Tener una mascota creada, con un peso asignado.

Postcondición: Se borra un pesaje, anteriormente creado.

Proceso normal principal:

1. Se inicia la aplicación.
2. Se conecta a la base de datos.
3. Se muestra la información de la mascota.
4. Se realiza el borrado del pesaje en cuestión.
5. Se devuelve el resultado positivo de la eliminación.

Alternativas de proceso y excepciones:

1) Se devuelve un error de borrado del pesaje de la mascota seleccionada.

2.3.17. Caso de uso: Añadir entrenamiento.

Resumen de la funcionalidad: Permite crear entrenamientos personalizados, que pueden valorarse mediante un sistema de puntuación y asignarse a la mascota elegida.

Actores: **Usuario**.

Casos de uso relacionados: Ninguno.

Precondición: Tener una mascota creada.

Postcondición: Se crea la evaluación de un ejercicio de entrenamiento, para la mascota elegida.

Proceso normal principal:

1. Se inicia la aplicación.
2. Se conecta a la base de datos.
3. Se muestra la información de la mascota.
4. Se realiza la creación y valoración del entrenamiento de la mascota en cuestión.
5. Se devuelve el resultado positivo de la creación del mismo.

Alternativas de proceso y excepciones:

1) Se devuelve un error de creación del entrenamiento de la mascota.

2.3.18. Caso de uso: Modificar entrenamiento.

Resumen de la funcionalidad: Permite modificar los entrenamientos, creados previamente para una mascota.

Actores: **Usuario**.

Casos de uso relacionados: Ninguno.

Precondición: Tener una mascota creada anteriormente, con entrenamientos asignados.

Postcondición: Se modifica un entrenamiento, creado previamente.

Proceso normal principal:

1. Se inicia la aplicación.
2. Se conecta a la base de datos.
3. Se muestra la información de la mascota.

4. Se realiza la modificación del entrenamiento en cuestión.
5. Se devuelve el resultado positivo de la modificación.

Alternativas de proceso y excepciones:

1) Se devuelve un error de modificación del entrenamiento de la mascota.

2.3.19. Caso de uso: Borrar entrenamiento.

Resumen de la funcionalidad: Permite borrar los entrenamientos asignados previamente para una mascota.

Actores: **Usuario**.

Casos de uso relacionados: Ninguno.

Precondición: Tener una mascota creada, con ejercicios de entrenamientos asignados para ella.

Postcondición: Se borra un entrenamiento, creado previamente.

Proceso normal principal:

1. Se inicia la aplicación.
2. Se conecta a la base de datos.
3. Se muestra la información de la mascota.
4. Se realiza el borrado del entrenamiento en cuestión.
5. Se devuelve el resultado positivo de la eliminación.

Alternativas de proceso y excepciones:

1) Se devuelve un error de borrado del entrenamiento de la mascota.

2.4. Diagrama de clases.

A continuación, se adjunta el diagrama de clases.

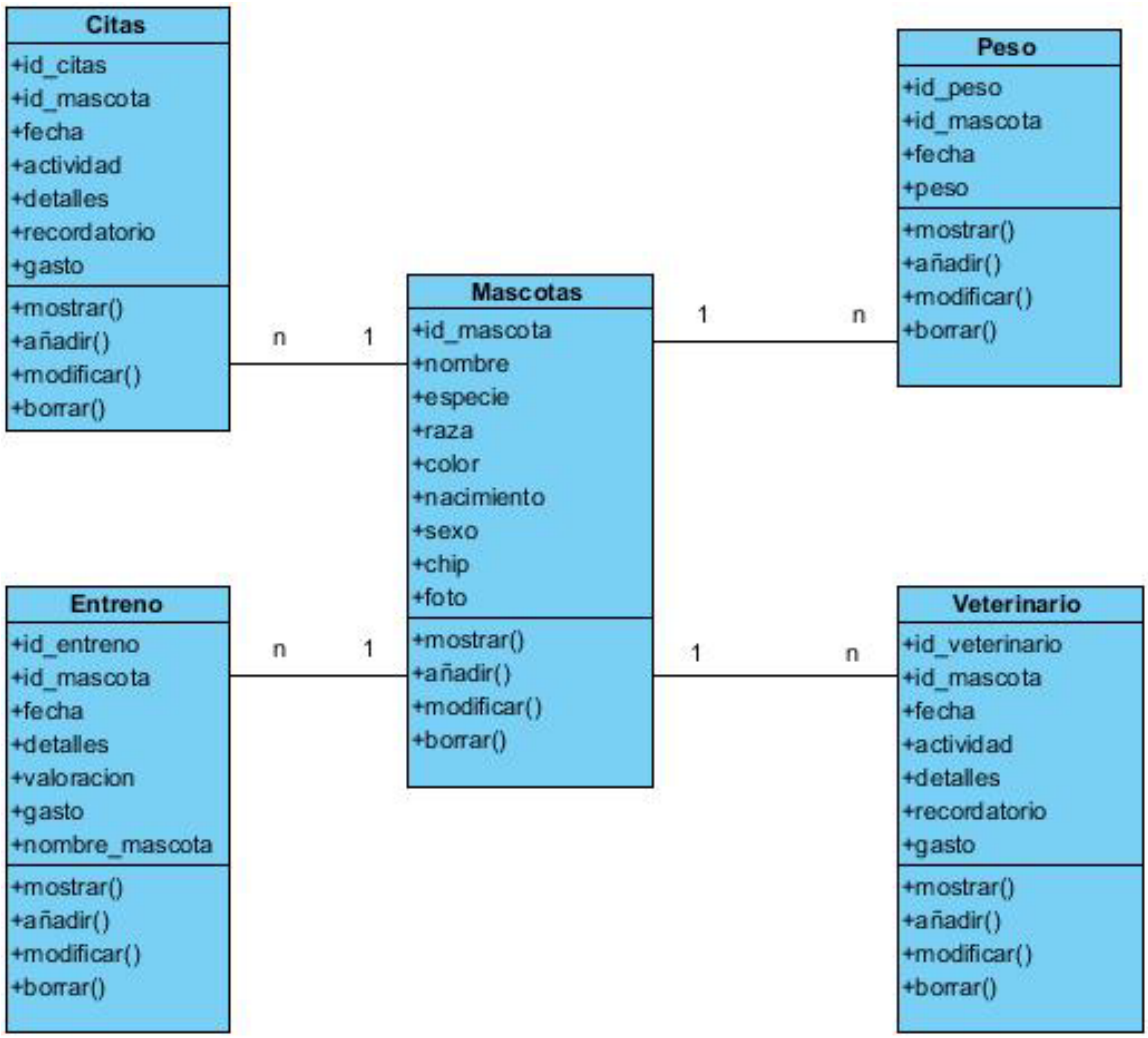

Ilustración 8: Diagrama de clases.

3. Diseño.

3.1. Arquitectura del “software”.

La arquitectura propuesta se basa en construir una aplicación nativa de “Android”, que consta de cuatro capas.

En mi proyecto, en particular, el sistema consumirá servicios compartidos por otras aplicaciones, concretamente “SQLite”.

Las capas del sistema son las siguientes:

1. Capa de presentación.
2. Capa intermedia.
3. Capa de negocio.
4. Capa de persistencia.

3.1.1. Capa de presentación.

Se trata de la capa superior, que se compone de código “XML”. En esta capa, ubicamos los elementos que componen las distintas vistas o pantallas de la aplicación. En el caso de “MYPETS”, lo tenemos compuesto por una serie de “layouts”: con “recyclerviews” dinámicos y “cardviews” asociados a los mismos.

3.1.2. Capa intermedia.

La responsabilidad de esta capa es gestionar la lógica de negocio utilizada por la interfaz del sistema. El objetivo de esta capa es evitar duplicidades de código, reutilizarlo en la medida de lo posible y gestionarlo de forma eficaz.

3.1.3. Capa de negocio.

En esta parte, disponemos de toda la lógica perteneciente a nuestra aplicación: el mapeo, con las bases de datos y los tipos de datos que existen en nuestra persistencia.

3.1.4. Capa de persistencia.

Es la capa inferior, el corazón de nuestra aplicación. Esta capa, únicamente realiza comunicaciones con su capa superior. Es en ella donde se persiste la información, es decir, donde ésta queda almacenada en nuestra base de datos.

3.2. Base de datos.

Esta aplicación almacena la información en una base de datos “SQLite”, ya que se trata de un sistema nativo y muy fiable, a la par que altamente potente para el manejo de los datos.

3.2.1. Diseño conceptual.

A continuación, adjuntamos el diseño conceptual de la base de datos planificada para nuestra aplicación.

Como podemos observar, hemos definido diferentes entidades y cada una de ellas, cuenta con sus propios atributos y características propias.

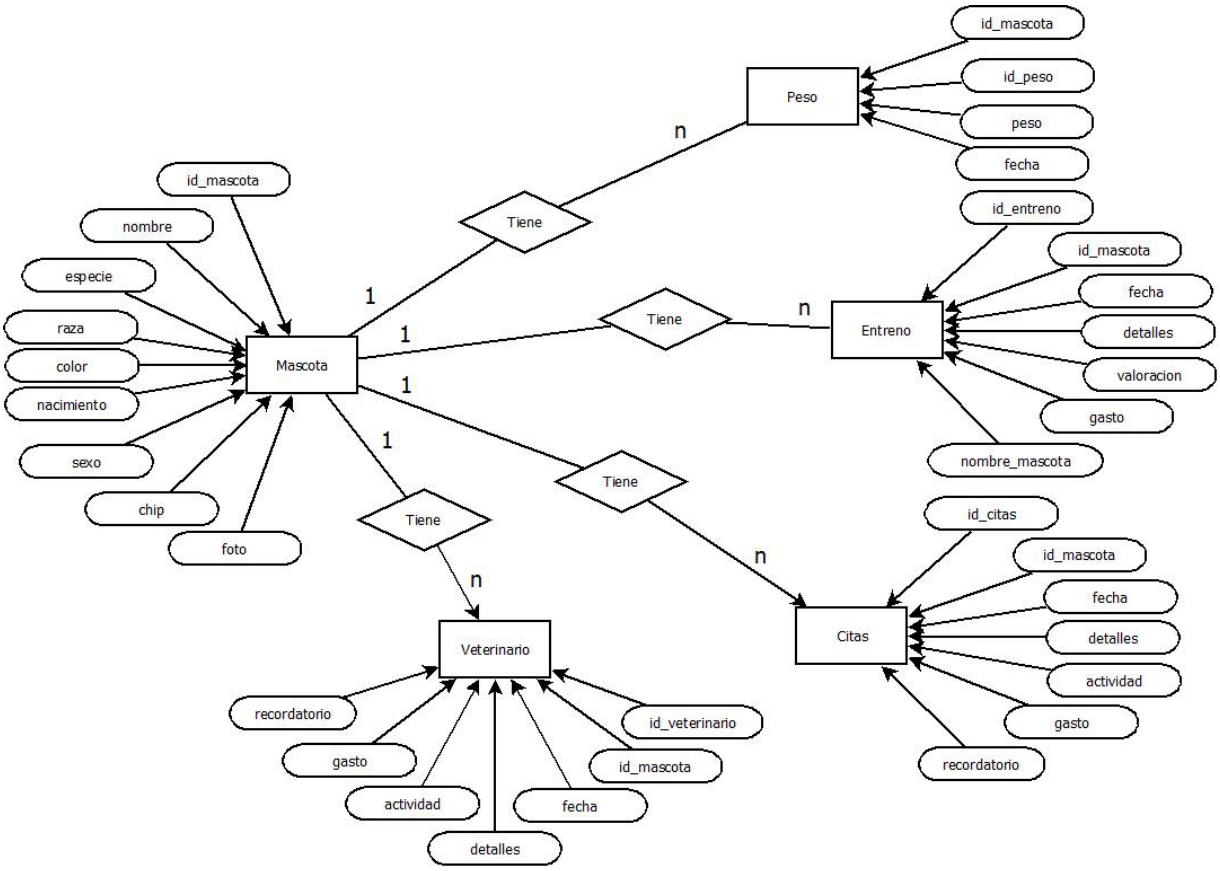

Ilustración 9: Diseño conceptual.

3.2.2. Diseño lógico(tablas normalizadas).

Para la aplicación, hemos implementado como normalización la tercera forma normal(3FN). El planteamiento que he realizado es el siguiente:

Tenemos una mascota que posee un identificador único y autoincremental y que además, contiene sus atributos: nombre, raza, sexo, etc.

Cada elemento que podemos añadir a la mascota (como por ejemplo: un pesaje, una cita con el veterinario, un entrenamiento o un evento especial) tendrá su identificador único y autoincremental, además de una clave foránea, que será el identificador de la mascota. Asumiendo estas consideraciones, lo que obtendremos serán unas tablas en “3FN”.

A continuación, se adjunta el diseño lógico:

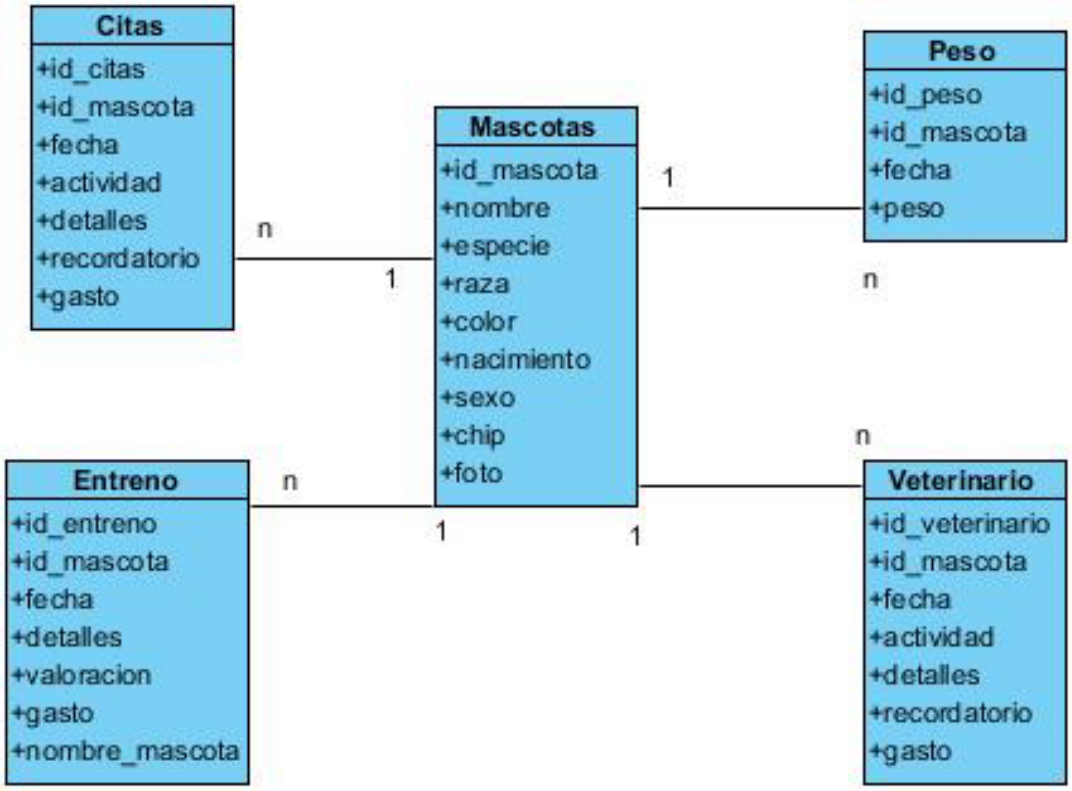

Ilustración 10: Diseño lógico.

3.3. Prototipado gráfico.

Para la aplicación, hemos realizado todo el diseño de manera nativa en “Android Studio”(mediante “layouts” en el propio “IDE”) y únicamente, estará disponible para terminales móviles.

Cada “layout”, tiene unos elementos de diseño: “imageview”, “button”, etc. A su vez, a cada elemento le he asignado un identificador único, de cara a la posterior implementación del código.

Se adjuntan “pantallazos”, con una pequeña explicación de su función:

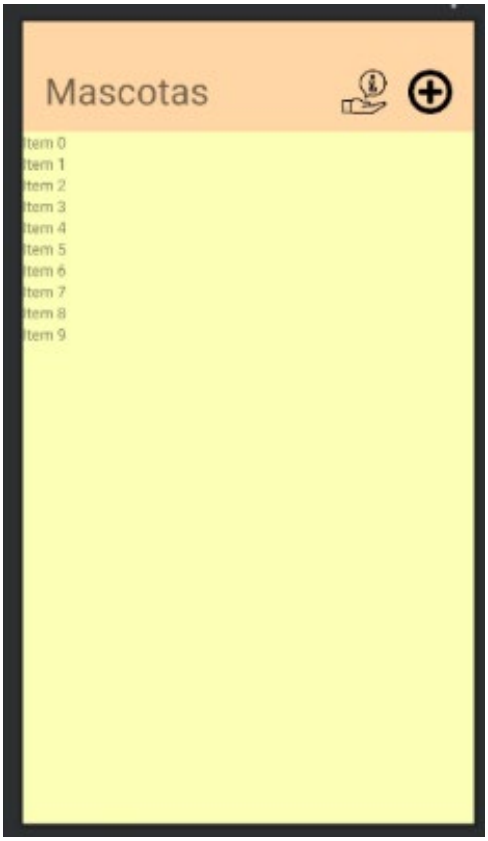

Ilustración 11: Layout “Main Activity”.

“Layout” Main_Activity

Este “layout” es el punto de carga inicial de la aplicación. Desde el mismo, podemos:

- Crear una mascota.
- Editar una mascota.
- Ver las mascotas creadas.

“Layout” Nueva_Mascota_Activity

Este “layout” se muestra cuando pulsamos el botón (+) de la parte superior derecha del “main_activity”. Desde él, podemos crear una mascota indicando los datos de la misma, como por ejemplo: nombre, especie, raza, sexo, fecha de nacimiento, etc.

Ilustración 12: Layout “Nueva Mascota”.

Ilustración 13: Layout “Tarjeta Mascotas”.

“Layout” Tarjeta_Mascotas.

En este “layout”, es donde aparecen las mascotas que tenemos creadas. Desde aquí, podemos ver sus citas, medidas de pesaje, entrenamientos, etc. Si pulsamos sobre la propia ficha, editaremos los datos de la mascota.

“Layout” Editar_Mascota_Activity

Este “layout” se nos mostrará al pulsar sobre cualquier punto de la “tarjeta_mascotas”, excepto en los botones inferiores. Desde él, podemos modificar los valores de una mascota creada previamente.

Ilustración 14: Layout “Editar Mascota”.

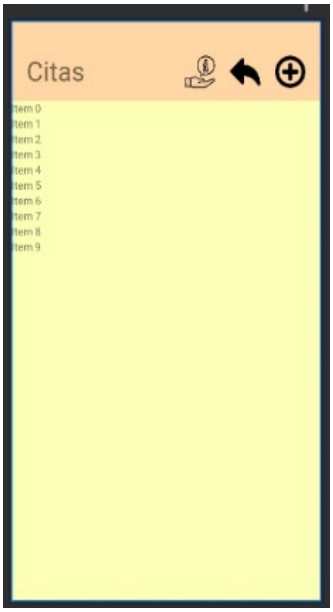

Ilustración 15: Layout “Mostrar Cita”.

“Layout” Mostrar_Cita_Activity

Este “layout” aparecerá al pulsar sobre el botón de citas de la “tarjeta_mascotas”. Desde él, veremos las citas que tengamos concertadas para nuestras mascotas (“recyclerview” central) y además, podremos volver a la pantalla principal, así como obtener un “pop-up” de ayuda o crear una nueva cita.

“Layout” Nueva_Cita_Activity

Este “layout” se nos mostrará al pulsar sobre el botón (+) de la parte superior derecha de “mostrar_cita”. Desde el mismo, podemos crear las diferentes citas elegidas para nuestra mascota.

Ilustración 16: Layout “Nueva Cita”.

“Layout” Tarjeta_citas

Ilustración 17: Layout “Tarjeta Citas”.

Este “layout” nos facilitará los datos de las citas que tengamos asignadas para la mascota y nos mostrará su nombre, los detalles a realizar en la cita, así como la fecha y hora de la misma.

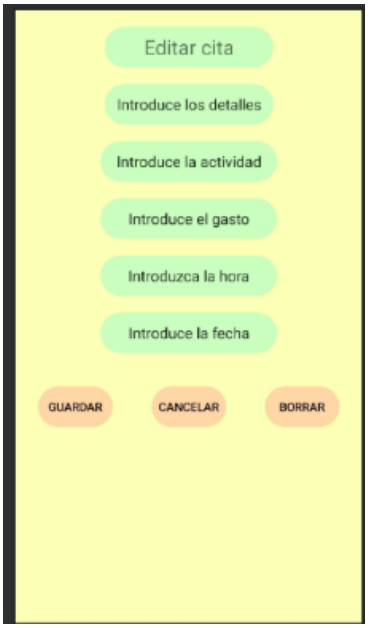

Ilustración 18: Layout “Editar Cita”.

“Layout” Editar_Cita_Activity

Este “layout” aparecerá al pulsar cualquier parte de la “tarjeta_citas”. Desde él, podemos modificar los datos de una cita que hayamos creado anteriormente.

También podremos eliminar dicha cita, si así fuese necesario.

“Layout” Mostrar_Veterinario_Activity

Este “layout” se visualizará al pulsar sobre el botón de “citas veterinarias” en la “tarjeta_mascotas”. Desde este “activity”, veremos las citas médicas que tengamos creadas previamente y podremos crear otras nuevas, pulsando sobre el botón (+) de la parte superior.

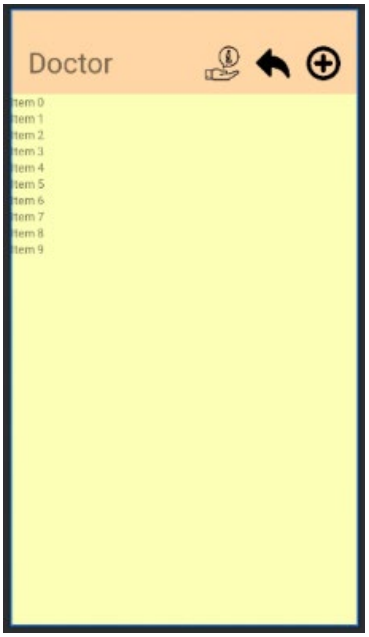

Ilustración 19: Layout “Mostrar Veterinario”.

Ilustración 20: Layout “Nuevo Veterinario”.

“Layout” Nuevo_Veterinario_Activity

Este “layout” se nos mostrará al pulsar sobre el botón (+) del “layout” “mostrar_veterinario”. Desde el mismo, podremos crear una nueva cita veterinaria para nuestra mascota.

“Layout” Tarjeta_Veterinario

Este “layout” aparecerá en el “recyclerview” de “mostrar_veterinario”. En él, veremos todos los detalles de nuestra cita veterinaria.

Ilustración 21: Layout “Tarjeta Veterinario”.

Ilustración 22: Layout “Editar Veterinario”.

“Layout” Editar_Veterinario_Activity

Este “layout” se visualizará al pulsar sobre cualquier zona de la “tarjeta_veterinario”. Desde el mismo, podremos modificar una cita veterinaria creada previamente. Si quisiéramos eliminarla, desde aquí mismo podríamos realizarlo.

“Layout” Mostrar_Peso_Activity

Este “layout” se nos mostrará al pulsar sobre el botón de “datos de pesaje” de la “tarjeta_mascotas”. Desde este “activity”, veremos un gráfico de evolución de los pesajes de nuestra mascota, en la parte superior. En la parte inferior, visualizaremos los pesajes que tengamos creados previamente y además podremos añadir otros nuevos, pulsando sobre el botón (+).

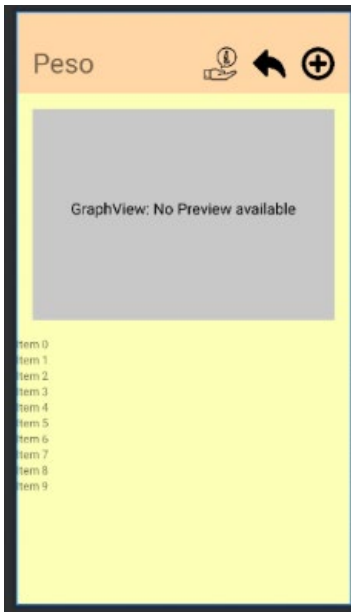

Ilustración 23: Layout “Mostrar Peso”.

Ilustración 24: Layout “Nuevo Peso”.

“Layout” Nuevo_Peso_Activity

Este “layout” aparecerá al pulsar sobre el botón (+) del “activity” “mostrar_peso”. Desde él, podremos crear un nuevo pesaje para nuestra mascota, indicando una fecha y el peso asignado.

“Layout” Tarjeta_Peso

Este “layout” se nos mostrará en el “recyclerview” de “mostrar_peso”. En él, veremos todos los detalles de los pesajes.

Ilustración 25: Layout “Tarjeta Peso”.

Ilustración 26: Layout “Editar Peso”.

“Layout” Editar_Peso_Activity

Este “layout” se nos mostrará al pulsar sobre cualquier zona de la “tarjeta_peso”. Desde él, podremos modificar un pesaje guardado anteriormente. Si quisiéramos eliminarlo, podríamos hacerlo desde aquí mismo.

“Layout” Mostrar_Entreno_Activity

Este “layout” se presentará al pulsar sobre el botón de “entrenamientos” de la “tarjeta_mascotas”. Desde este “activity”, veremos el deporte que ha realizado nuestra mascota, incluyendo la valoración que le hayamos asignado. Si fuese necesario añadir más elementos, pulsaríamos el botón (+).

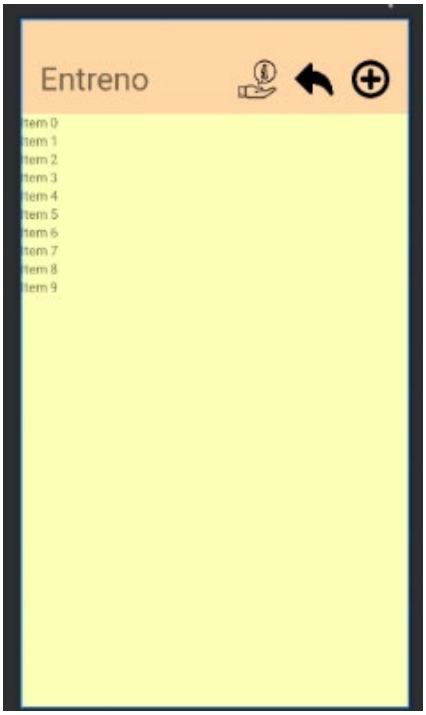

Ilustración 27: Activity “Mostrar Entreno”.

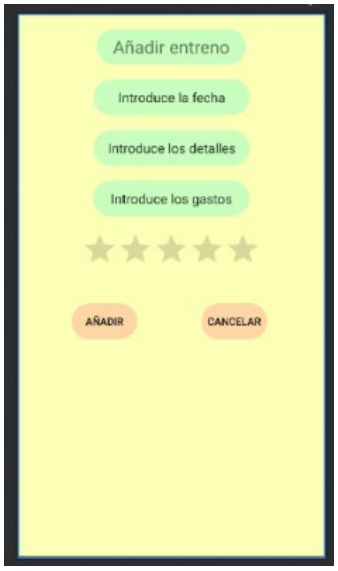

Ilustración 28: Layout “Nuevo Entreno”.

“Layout” Nuevo_Entreno_Activity

Este “layout” aparecerá al pulsar sobre el botón (+), del “layout” “mostrar_entreno”. Desde el mismo, podremos crear una nueva actividad deportiva que hayamos realizado con nuestra mascota. También podremos darle una puntuación de 1 a 5 estrellas.

“Layout” Tarjeta_Entreno

Este “layout” se visualizará en el “recyclerview” de “mostrar_entreno”. En él, veremos todos los detalles de los entrenamientos realizados y su puntuación.

Ilustración 29: Layout “Tarjeta Entreno”.

Ilustración 30: Layout “Editar Entreno”.

“Layout” Editar_Entreno_Activity

Este “layout” se presentará al pulsar sobre cualquier parte de la “tarjeta_entreno”. Desde el mismo, podremos modificar o eliminar un entrenamiento creado previamente.

4. Implementación.

4.1. Estructura del código.

El código de mi aplicación lo he dividido en 4 secciones principales, donde se albergan los distintos fragmentos de código, para realizar las funciones requeridas.

4.1.1. Adaptadores.

En esta sección he dispuesto 5 ficheros, donde generamos el adaptador para mostrar los datos en las “cardviews”. Tenemos 1 adaptador para las mascotas, otro para eventos, además del asignado para citas veterinarias, el de los pesajes y por último, el de los entrenamientos.

El código de estos ficheros estará al final de este documento (como anexo).

4.1.2. Consultas Base de datos.

En esta sección, disponemos de 6 ficheros:

Tenemos un “DbHelper”, donde creamos la base de datos con sus tablas y las actualizaciones que reciba la misma.

Tenemos también los ficheros, donde realizamos las consultas, inserciones y edición de los distintos elementos de la aplicación, es decir: mascotas, eventos, citas veterinarias, pesajes y deporte realizado.

De igual manera, el código de esta sección figurará al final del documento (como anexo).

4.1.3. Entidades.

En esta área, disponemos de 5 ficheros. En ellos, se encuentran dispuestos cada uno de los cinco objetos(eventos, citas veterinarias, mascotas, pesajes y entrenamientos), con sus respectivos “getters” y “setters”.

4.1.4. Resto de clases.

A continuación, tenemos el resto de clases, donde para cada uno de los objetos(mascotas, eventos, citas veterinarias, pesajes y deportes) existen 3 tipos de clases asignadas: una, para el apartado de la visualización de los datos, otra para el apartado de creación y la siguiente, para el apartado de modificación.

Desde estas clases, hacemos llamamientos a las entidades, a las conexiones a la base de datos y a los adaptadores, en función de las necesidades a realizar que tengamos. Llegado el caso, pasaremos los datos que sean necesarios al realizar el llamamiento al método de la clase.

4.2. Pruebas.

4.2.1. Realización de las pruebas.

Una vez finalizada la implementación del proyecto, pasamos a la fase de pruebas.

A través de ellas, intentamos evitar en la medida de lo posible que se puedan producir fallos, o si estos se producen, que puedan incomodar al usuario de la aplicación.

La intención es probar todas las funcionalidades desarrolladas y detectar todos los errores, para poder subsanarlos antes de lanzar la aplicación y así obtener un “software” en óptimas condiciones.

Al concluir la implementación de cada módulo, se han realizado pruebas unitarias sobre el mismo.

Ejemplos:

- ✓ Eliminación del fichero de base de datos y tratamiento del error.
- ✓ Inserción de valores nulos y tratamiento de los mismos.
- ✓ Inserción de valores fuera de rango y tratamiento de ellos.
- ✓ Inserción de cadenas en variables numéricas, para el tratamiento de los errores.
- ✓ Eliminación de datos sin previa inserción y gestión de los posibles fallos.

Durante las pruebas realizadas, no solamente se ha testeado el código, sino que también se ha comprobado que la aplicación se visualiza correctamente. Además, se verifica que la distribución de los elementos sobre la pantalla es uniforme y ajustada al diseño.

También se ha examinado la traducción de la misma.

4.2.2. Resultados de las pruebas.

La aplicación ha pasado las pruebas satisfactoriamente. No obstante, durante las mismas, se detectaron mejoras de diseño.

Ejemplos:

- “Datepicker” de selección de fechas con doble dígito.
- Duplicidades de código en la traducción: se han simplificado y unificado.
- Añadir títulos a cada “activity”, así como “toast”, para indicar aquello que estamos realizando.

Cuando se han detectado fallos, se han ido añadiendo al documento para solventarlos. Una vez finalizada la implementación de la aplicación, se procederá a realizar las pruebas de la misma al completo.

5. Asignaturas.

Dentro del Ciclo Formativo se han estudiado una serie de asignaturas.

En este punto, vamos a tratar de relacionar cada una de ellas con las tareas asignadas a este proyecto.

- **Sistemas informáticos y entornos de desarrollo**: ambas asignaturas proporcionan el conocimiento adecuado para instalar todo el “software” necesario para la realización de este proyecto: máquinas virtuales, emuladores, “Android Studio”, “Word”, etc.

- **Bases de datos y acceso a datos:** con los conocimientos adquiridos a lo largo del curso, hemos podido desarrollar e implementar la base de datos y las consiguientes consultas a la misma.

- **Sistemas de gestión empresarial y empresa e iniciativa emprendedora:** se han adquirido habilidades para el estudio y la planificación de este proyecto y para poder llevarlo a cabo: estimación de horas, costes y en su caso, la elaboración de todo el material adjunto a las mismas.

- **Desarrollo de interfaces , lenguaje de marcas e inglés:** he adquirido los conocimientos precisos para poder realizar todo el apartado gráfico de la aplicación y además, poder hacerlo de una forma correcta y satisfactoria. En este caso concreto, la aplicación está disponible en dos idiomas, castellano e inglés, utilizando los recursos disponibles a nuestro alcance.

- **Programación , programación de servicios y procesos, programación multimedia y de dispositivos móviles:** me han dotado de las habilidades necesarias para poder llevar a cabo esta aplicación y que además, realice las funciones para las que está prevista. También, para la subsanación de los errores detectados en sus distintas fases de desarrollo, o de los test y las pruebas realizadas a la misma.

- **Formación y orientación laboral:** esta asignatura está más centrada en el ámbito laboral, por lo que considero que, en este caso concreto, estos conocimientos no son aplicables.

.

6. Líneas futuras.

La aplicación, en algún momento determinado, habrá que complementarla con funciones que, en la actualidad, no están desarrolladas: bien por falta de presupuesto o de tiempo.

Algunas de las posibles mejoras serán:

- Desarrollo para dispositivos IOS.
- Creación de galería de imágenes de cada mascota.
- Creación de apartado de copias de seguridad.
- Geolocalización de mascotas en caso de pérdida.
- Localización de servicios cercanos a nuestra ubicación.
- Creación de apartado para clínicas y servicios profesionales.
- Traducción multi-idioma.

Todas las mejoras requerirán de su análisis pormenorizado y de su correspondiente estudio, para su posterior implementación.

7. Conclusiones.

Desde mi punto de vista, el proyecto ha constituido un gran reto personal para mí. Este trabajo es el final de una interesante e intensa etapa de estudio, donde se han puesto en práctica los conocimientos adquiridos a lo largo de la misma.

Esta aplicación ha satisfecho las necesidades presentadas en la planificación inicial, cubriendo los requisitos de los posibles clientes. Se ha desarrollado de manera nativa para “Android”, realizando un aprendizaje del “IDE”, para en un futuro implementarlo en dispositivos IOS.

Cabe destacar que se han adquirido y mejorado conocimientos sobre:

- Implementación de librerías externas.
- Reforzar el aprendizaje de “SQL”.
- Reforzar el aprendizaje de “Java”.
- Reforzar el aprendizaje de “XML”.
- Reforzar el aprendizaje del diseño de bases de datos.

Por último, he detectado que seguir una planificación resulta bastante complicado, sin disponer de una experiencia previa. Esta se ha basado en una estimación de horas y al no poseer dicha experiencia, creo que algunas partes han quedado escasas de tiempo y otras, en cambio, han resultado ser demasiado extensas. No obstante, es factible mejorarlo con práctica y esfuerzo.

8. Bibliografía.

Algunas de las fuentes externas consultadas son las siguientes:

https://developer.android.com/studio/intro

https://docs.github.com/es

https://stackoverflow.com/