DAMAGOTCHI

Proyecto final de desarrollo
CFGS Desarrollo de Aplicaciones Multiplataforma
IES Puig Castellar (Santa Coloma de Gramenet)
Tutor de proyecto: Luis Elía

Iria Calvo Blanco
Curso 2025 / 2026

Resumen del proyecto

Este proyecto consiste en el desarrollo de una aplicación móvil para dispositivos Android que combina el entretenimiento con el apoyo durante el embarazo. La idea principal es crear una experiencia interactiva inspirada en el concepto de Tamagotchi, en la que el usuario cuida de un personaje que representa a una mujer embarazada, incorporando además contenido informativo y funcionalidades útiles para el día a día.

La aplicación se organiza en tres secciones principales. En primer lugar, el modo juego, donde el usuario interactúa con el personaje y debe atender diferentes indicadores como la energía, la alimentación, la hidratación o el descanso, simulando así el cuidado diario. En segundo lugar, la sección de comunidad, pensada como un espacio de intercambio donde los usuarios pueden compartir experiencias o encontrar apoyo. Por último, el tablón de anuncios, en el que se muestran consejos, recomendaciones y mensajes informativos adaptados a la etapa del embarazo en la que se encuentra la persona en la vida real.

Una de las características más importantes del proyecto es el uso de dos líneas de tiempo. Por un lado, el juego simula el embarazo en un total de 30 días, distribuidos en 10 días por trimestre. Por otro, se tiene en cuenta el tiempo real del usuario, lo que permite ofrecer un seguimiento más preciso y contenido personalizado según su situación.

El desarrollo se ha llevado a cabo mediante una metodología incremental, lo que ha permitido ir incorporando funcionalidades de forma progresiva hasta conseguir una aplicación funcional, intuitiva y orientada a ofrecer tanto entretenimiento como apoyo real.

Palabras clave

Embarazo
Android
Juego educativo
Cuidado prenatal
Acompañamiento
Salud maternal
Comunidad

Abstract

This project consists of the development of a mobile application for Android devices that combines entertainment with support during pregnancy. The main idea is to create an interactive experience inspired by the Tamagotchi concept, in which the user takes care of a character representing a pregnant woman, while also incorporating informative content and useful daily features.

The application is organized into three main sections. First, the game mode, where the user interacts with the character and manages different indicators such as energy, nutrition, hydration, and rest, simulating daily care. Second, the community section, designed as a space where users can share experiences and find support. Finally, the notice board, where tips, recommendations, and informative messages are displayed, adapted to the stage of pregnancy the user is in real life.

One of the most important features of the project is the use of two timelines. On one hand, the game simulates pregnancy over a total of 30 days, divided into 10 days per trimester. On the other hand, the application also considers real time, allowing for more accurate tracking and personalized content based on the user's situation.

The project has been developed using an incremental methodology, enabling the progressive implementation of features until achieving a functional, intuitive application focused on both entertainment and real support.

Keyword

Pregnancy
Android application
Educational game
Prenatal care
User support
Maternal health
Community interaction

ÍNDICE

1. Introducción

Damagotchi es una aplicación móvil para dispositivos Android inspirada en la mecánica del clásico juego Tamagotchi, adaptada al contexto del embarazo. En este caso, el personaje principal es una mujer embarazada cuyo bienestar depende de las decisiones del usuario a través de las interacciones diarias, lo que permite simular el cuidado y la atención necesarios durante este proceso.

El proyecto se enmarca dentro del desarrollo de aplicaciones móviles con un enfoque interactivo y educativo. Su propósito principal no es únicamente ofrecer entretenimiento, sino también facilitar el acceso a información relacionada con el cuidado prenatal de una forma más visual, accesible y dinámica. Para ello, la aplicación integra diferentes elementos como medidores de salud, consejos y recordatorios prácticos que pueden aplicarse en la vida real, si el usuario lo desea.

Además, la aplicación se estructura en varias secciones que enriquecen la experiencia. Por un lado, el modo juego, donde se desarrolla la interacción principal con el personaje. Por otro, una sección de comunidad, pensada para el intercambio de experiencias entre usuarios. Y, finalmente, un tablón de anuncios donde se muestran mensajes, recomendaciones y consejos adaptados a la etapa del embarazo.

De esta forma, se busca acompañar al usuario en su proceso de aprendizaje mediante una experiencia virtual que refuerce el impacto positivo de cada acción, tanto dentro del juego como en su posible aplicación en la vida real.

A lo largo de esta memoria se explicará el proceso completo de desarrollo del proyecto, desde su planteamiento inicial hasta la implementación final de la aplicación.

1.1 Contexto

El proyecto se sitúa en el ámbito de las aplicaciones móviles relacionadas con la salud y el bienestar, concretamente en el área del seguimiento del embarazo y el cuidado prenatal. En los últimos años, el uso de aplicaciones móviles en este ámbito ha crecido considerablemente, ofreciendo a los usuarios acceso a información, seguimiento del progreso del embarazo y herramientas de apoyo.

Actualmente, existen diversas aplicaciones que permiten realizar un seguimiento del embarazo, proporcionando datos sobre el desarrollo del feto, recomendaciones médicas generales y recordatorios relacionados con hábitos saludables. Estas aplicaciones suelen centrarse en ofrecer información detallada y seguimiento basado en el tiempo real del embarazo, facilitando así el acceso a contenido útil para el usuario.

Sin embargo, muchas de estas soluciones presentan ciertas limitaciones. En la mayoría de los casos, se trata de aplicaciones con un enfoque principalmente informativo, lo que puede hacer que la experiencia del usuario resulte poco interactiva o incluso monótona a largo plazo. Además, no suelen incorporar elementos que fomenten la participación activa del usuario más allá de la consulta de información.

Por otro lado, aunque existen aplicaciones de entretenimiento basadas en mascotas virtuales, estas no están orientadas a un contexto educativo o de salud, lo que limita su utilidad más allá del ocio.

En este contexto, el desarrollo de soluciones que combinen interactividad, entretenimiento y contenido educativo resulta especialmente relevante en la actualidad, ya que permite mejorar la implicación del usuario y facilitar el aprendizaje de forma más dinámica y accesible.

1.2 Justificación

El desarrollo de este proyecto surge de la necesidad de ofrecer una alternativa diferente a las aplicaciones actuales relacionadas con el embarazo. Tal y como se ha visto en el contexto, la mayoría de las soluciones existentes se centran principalmente en proporcionar información y seguimiento, pero suelen dejar de lado la parte interactiva, lo que puede hacer que el usuario pierda interés con el tiempo.

Damagotchi propone una mejora en este sentido al combinar el componente informativo con una experiencia de juego interactiva. A través del cuidado de un personaje virtual, el usuario no solo recibe información, sino que participa de forma activa en el proceso, lo que puede favorecer a una mayor implicación y comprensión de los hábitos relacionados con el bienestar prenatal. Esta combinación permite transformar una actividad pasiva, como la lectura de información, en una experiencia más dinámica y cercana.

Además, la aplicación introduce elementos diferenciadores como la división en secciones (modo juego, comunidad y tablón de anuncios) y el uso de dos líneas de tiempo, una simulada dentro del juego y otra basada en el tiempo real. Esto permite ofrecer tanto una experiencia de entretenimiento como un seguimiento más personalizado, adaptado a la situación actual del usuario.

El origen de este proyecto también está relacionado con una experiencia personal. Durante el curso pasado, en situación de embarazo, pude comprobar que muchas de las aplicaciones disponibles se centran principalmente en ofrecer información en formato de texto, con numerosos artículos y poco dinamismo. Aunque existen aplicaciones como Embarazo+, que fue una de mis principales inspiraciones por su forma más visual e interactiva de mostrar el crecimiento del embrión, en general sigue faltando una mayor interacción por parte del usuario.

A partir de esta experiencia surge la idea de desarrollar una aplicación diferente, en la que el usuario no sólo reciba información, sino que también participe activamente a través del juego. En este caso, se propone vivir el proceso desde fuera, cuidando a un personaje, lo que permite ofrecer una experiencia más dinámica, cercana y fácil de entender.

El resultado del proyecto puede tener una utilidad real como herramienta de apoyo complementaria, especialmente para aquellas personas que buscan una forma más accesible y visual de informarse sobre el embarazo.

1.3 Objetivos

El objetivo de este proyecto es desarrollar una aplicación móvil para dispositivos Android que combine un enfoque educativo con una experiencia interactiva, centrada en el acompañamiento durante el embarazo. A través de esta aplicación, se pretende representar de forma visual y dinámica las distintas fases del embarazo, facilitando la comprensión de los cambios físicos y emocionales que se producen durante este proceso.

Teniendo en cuenta esto, se espera desarrollar una aplicación educativa para Android basada en un sistema interactivo que permita simular el cuidado de una mujer embarazada, integrando información sobre el embarazo y fomentando el aprendizaje mediante la participación activa del usuario.

Desglosando este objetivo global, se plantean los siguientes objetivos específicos:

- ➢ Implementar un personaje principal con el que el usuario pueda interactuar y al que deba cuidar diariamente.
- ➢ Simular la rutina diaria mediante la integración de actividades relacionadas con el embarazo.
- ➢ Proporcionar información educativa sobre el embarazo, incluyendo nutrición y salud prenatal.
- ➢ Incorporar indicadores de estado que reflejen el bienestar físico y emocional del personaje en función de las decisiones del usuario.
- ➢ Fomentar la comprensión del proceso del embarazo a través de una experiencia práctica e interactiva.
- ➢ Integrar diferentes secciones dentro de la aplicación, como el modo juego, la comunidad y el tablón de anuncios.
- ➢ Permitir el seguimiento del embarazo tanto en tiempo simulado como en tiempo real.

En conjunto, estos objetivos permiten definir una aplicación que combina aprendizaje y entretenimiento, ofreciendo una experiencia interactiva que facilita la comprensión del embarazo de forma más cercana y accesible. El resultado esperado es una herramienta útil tanto para usuarios que viven esta experiencia como para su entorno, aportando valor educativo y práctico.

2. Estrategia y planificación

2.1 Estrategia de desarrollo y viabilidad

El proyecto parte de la idea de desarrollar una aplicación móvil inspirada tanto en aplicaciones de seguimiento del embarazo como en juegos de mascotas virtuales tipo Tamagotchi. Durante una experiencia de embarazo personal, me surgió la necesidad de encontrar aplicaciones que ofreciesen un seguimiento dinámico e interactivo, ya que muchas de ellas estaban centradas principalmente en artículos informativos y contenido estático. Aunque aplicaciones como Embarazo+ sirvieron como inspiración inicial por su forma más visual de representar el crecimiento de un embrión, seguía faltando una mayor participación del usuario dentro de la experiencia.

A partir de esta idea, se planteó el desarrollo de una aplicación que combinase entretenimiento e información educativa mediante el cuidado de un personaje virtual. Para facilitar el inicio del proyecto y trabajar sobre una estructura moderna de Android, se tomaron como referencia ejemplos desarrollados con Jetpack Compose disponibles en:

- ➔ Compose Samples

Además, para las mecánicas principales del juego se utilizó como inspiración y base del proyecto:

- ➔ TamaPet.

La estrategia de desarrollo se planteó de forma progresiva y modular, priorizando inicialmente el funcionamiento básico del juego y la interacción con el personaje. Posteriormente, se añadieron funcionalidades complementarias como la comunidad y el tablón de anuncios, pero durante el desarrollo se decidió dividir la aplicación en diferentes secciones independientes para mejorar la organización y la experiencia del usuario.

Además, tras una primera versión más sencilla del juego, se optó por separar el funcionamiento de la aplicación en dos líneas temporales diferentes: una línea interna del juego, donde el embarazo se desarrolla en 30 días divididos en tres trimestres, y otra basada en el tiempo real del usuario, utilizada para mostrar consejos, mensajes y recomendaciones adaptadas a su etapa real del embarazo.

En cuanto a la viabilidad, el proyecto se considera realizable dentro del tiempo disponible, aunque ha supuesto un reto debido al aprendizaje de nuevas tecnologías. Mientras que en cursos anteriores el desarrollo Android se había trabajado principalmente con XML y Java, este proyecto ha requerido aprender Kotlin y adaptarse a estructuras de desarrollo más actuales. Para asegurar una versión funcional y estable dentro del tiempo establecido, se decidió priorizar las funcionalidades esenciales del sistema y dejar posibles ampliaciones futuras en segundo plano.

2.2 Metodología

La metodología escogida para el desarrollo del proyecto ha sido Kanban, ya que permite gestionar el trabajo de forma visual, flexible y progresiva. Este enfoque resultó adecuado

para un proyecto de estas características, donde algunas funcionalidades y decisiones fueron evolucionando durante el desarrollo. Gracias a esta metodología, fue posible trabajar de manera continua sobre la aplicación, revisando y ajustando tareas según las necesidades que iban surgiendo.

La organización del trabajo se realizó mediante una división de tareas por bloques funcionales, permitiendo desarrollar y revisar cada parte del proyecto de forma independiente. Este enfoque facilitó el seguimiento del progreso y permitió reorganizar determinadas tareas cuando fue necesario.

Para la gestión y seguimiento de las tareas se utilizó la herramienta Trello, que permitió organizar visualmente el progreso del proyecto mediante tableros y tarjetas. Cada tarea se fue clasificando según su estado, diferenciando aquellas pendientes, iniciadas, en prueba o finalizadas, facilitando así una visión más clara del avance general del proyecto y de las prioridades en cada momento.

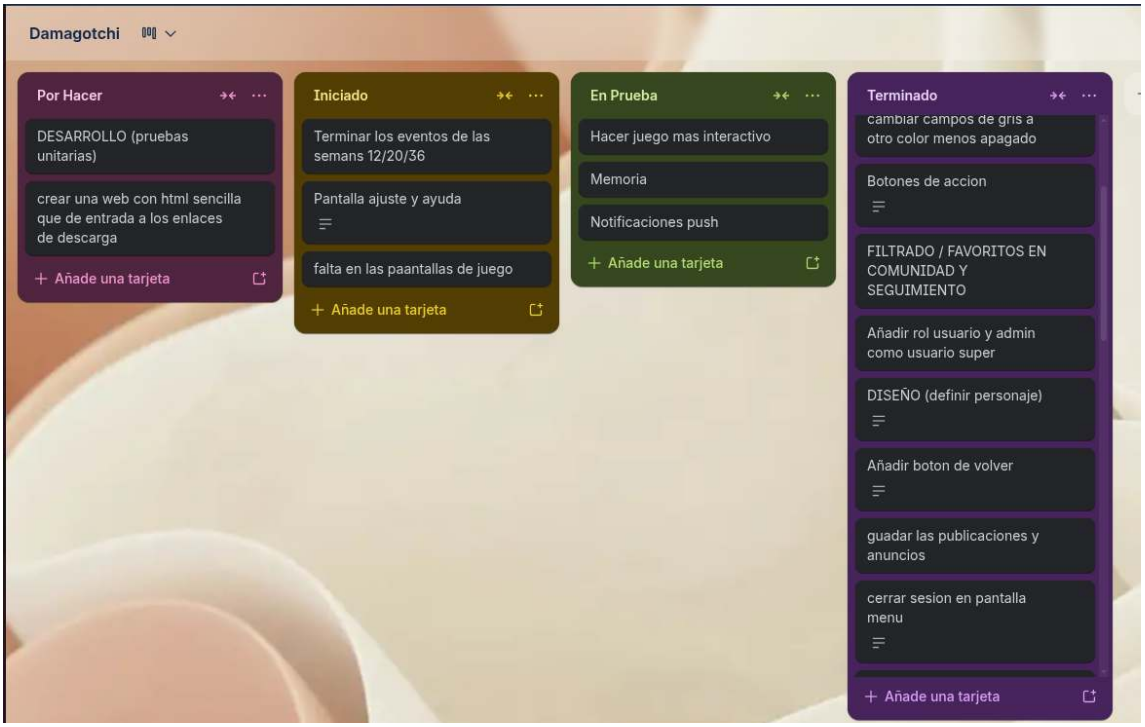

Imagen1. Gestión de tareas y control del progreso del proyecto utilizando Trello.

Como complemento a esta organización, también se utilizó un diagrama de Gantt para planificar las tareas a más largo plazo y visualizar la duración aproximada de cada fase del proyecto. La combinación de ambas herramientas permitió mantener una planificación general del proyecto junto con un seguimiento más flexible del trabajo diario.

Diagrama de Gantt.xlsx

Además, para el almacenamiento y control de versiones del código se utilizó GitHub, permitiendo guardar de forma segura los cambios realizados y mantener un seguimiento continuo de la evolución del proyecto.

2.3 Planificación

La planificación del proyecto se organizó en diferentes fases de trabajo desarrolladas de forma progresiva, permitiendo avanzar desde la idea inicial hasta la implementación final de la aplicación.

En una primera fase se realizó la investigación y análisis de aplicaciones relacionadas con el seguimiento del embarazo y los juegos tipo Tamagotchi, además de definir los objetivos y el enfoque general del proyecto.

Posteriormente, se preparó el entorno de desarrollo y comenzó el aprendizaje de las tecnologías necesarias, especialmente Kotlin y Jetpack Compose. Durante esta etapa también se definió la estructura inicial de la aplicación.

La siguiente fase se centró en el desarrollo del núcleo principal del proyecto, implementando el sistema básico del juego y la interacción con el personaje virtual. Una vez completada esta parte, se añadieron funcionalidades complementarias como la comunidad, el tablón de anuncios y el sistema de seguimiento basado en el tiempo real.

Finalmente, las últimas fases estuvieron dedicadas a la corrección de errores, mejoras visuales, realización de pruebas y elaboración de la documentación final del proyecto.

La distribución temporal de estas tareas se organizó mediante un diagrama de Gantt, utilizado como apoyo para mantener una planificación general del desarrollo y controlar el progreso del proyecto.

3. Análisis

3.1 Casos de uso

El sistema cuenta con dos tipos de roles: el usuario registrado y el administrador. A continuación, se documentan los casos de uso más representativos del sistema.

Identificador	CU-01
Nombre	Registrarse
Actor	Usuario
Precondiciones	El usuario no debe tener una cuenta registrada previamente.
Flujo principal	El usuario accede al formulario de registro, introduce sus datos y completa el proceso de creación de cuenta.
Flujo alternativo	El sistema muestra un aviso si el usuario ya existe o si algún campo es incorrecto.
Postcondiciones	La cuenta del usuario queda registrada correctamente en el sistema.

Identificador	CU-02
Nombre	Iniciar sesión
Actor	Usuario
Precondiciones	El usuario debe disponer de una cuenta registrada previamente.
Flujo principal	El usuario introduce sus credenciales y accede a la aplicación.
Flujo alternativo	El sistema muestra un error si las credenciales son incorrectas.
Postcondiciones	El usuario accede a su cuenta y puede continuar su progreso.

Identificador	CU-03
Nombre	Cuidar personaje
Actor	Usuario
Precondiciones	El usuario interactúa con el personaje virtual para mantener sus medidores de salud en niveles adecuados.

Flujo principal	El usuario accede al modo juego, consulta los medidores y realiza acciones de cuidado como alimentar, descansar o realizar actividad física, dependiendo las necesidades de la mascota.
Flujo alternativo	Si los medidores bajan demasiado, el sistema muestra un aviso de situación de riesgo que afecta al desarrollo del juego.
Postcondiciones	El personaje mejora su estado y continúa avanzando dentro del embarazo.

Identificador	CU-04
Nombre	Consultar Seguimiento
Actor	Usuario
Precondiciones	Permite consultar información relacionada con la etapa actual (real) del embarazo.
Flujo principal	El usuario accede al apartado de seguimiento y visualiza consejos y recomendaciones adaptadas a su etapa, que también pueden guardar en favoritos.
Postcondiciones	El usuario obtiene información personalizada y actualizada.

Identificador	CU-05
Nombre	Interactuar con la comunidad
Actor	Usuario y Administrador
Precondiciones	El usuario debe haber iniciado sesión en la aplicación.
Flujo principal	El usuario o administrador accede a la sección de comunidad y puede publicar, responder publicaciones existentes, dar “likes” o guardar las publicaciones a favoritos.
Flujo alternativo	El sistema muestra un aviso si ocurre un error al publicar o interactuar con el contenido.
Postcondiciones	La interacción realizada queda guardada y visible dentro de la comunidad.

Identificador	CU-06
Nombre	Gestionar publicaciones de seguimiento

Actor	Administrador
Precondiciones	El administrador debe haber iniciado sesión en la aplicación.
Flujo principal	El administrador accede a la sección de seguimiento y crea publicaciones informativas, consejos y recomendaciones relacionadas con las distintas etapas del embarazo, utilizando información contrastada y basada en fuentes fiables.
Flujo alternativo	El sistema muestra un aviso si ocurre un error al publicar o interactuar con el contenido.
Postcondiciones	Las publicaciones quedan almacenadas y visibles para los usuarios dentro de la sección de seguimiento.

Identificador	CU-07
Nombre	Configuración
Actor	Usuario
Precondiciones	El usuario debe haber iniciado sesión en la aplicación.
Flujo principal	El usuario accede al apartado de configuración y puede modificar diferentes opciones de la aplicación, como el sonido, las notificaciones, el idioma o aspectos del perfil.
Flujo alternativo	El sistema muestra un aviso si no es posible guardar los cambios realizados.
Postcondiciones	La configuración personalizada queda guardada y aplicada dentro de la aplicación.

3.2 Requisitos funcionales

Los requisitos funcionales describen las principales funcionalidades que debe ofrecer Damagotchi y las acciones que el sistema debe ser capaz de realizar desde el punto de vista del usuario y del propio funcionamiento interno de la aplicación. Los requisitos funcionales se han agrupado en grupos para facilitar su lectura.

Gestión de usuarios

- ➢ El sistema deberá permitir el registro de nuevos usuarios.
- ➢ El sistema deberá permitir el inicio y cierre de sesión.
- ➢ El sistema deberá almacenar el progreso y la información del usuario.

Sistema de juego

- ➢ El sistema deberá permitir interactuar con un personaje virtual basado en una mujer embarazada.
- ➢ El sistema deberá mostrar diferentes medidores relacionados con el bienestar del personaje, como energía, hambre, hidratación, higiene, actividad y descanso.
- ➢ El sistema deberá actualizar el estado del personaje según las acciones realizadas por el usuario.
- ➢ El sistema deberá mostrar diferentes estados físicos del personaje según el avance del embarazo.
- ➢ El sistema deberá permitir la evolución del embarazo mediante un sistema dividido en trimestres.
- ➢ El sistema deberá permitir recuperar parcialmente el estado del personaje tras situaciones de riesgo o enfermedad.

Seguimiento y notificaciones

- ➢ El sistema deberá mostrar consejos, mensajes y recomendaciones adaptadas a la etapa del embarazo.
- ➢ El sistema deberá generar recordatorios relacionados con el cuidado y bienestar del personaje.
- ➢ El sistema deberá mostrar avisos o situaciones de riesgo cuando los medidores del personaje alcancen valores bajos.
- ➢ El sistema deberá gestionar un sistema de notificaciones dentro de la aplicación.
- ➢ El sistema deberá diferenciar entre el tiempo interno del juego y el seguimiento basado en tiempo real.
- ➢ El sistema deberá actualizar automáticamente el contenido mostrado según la etapa real del embarazo del usuario.

Comunidad y publicaciones

- ➢ El sistema deberá permitir a los usuarios acceder a la comunidad de la aplicación.
- ➢ El sistema deberá permitir visualizar publicaciones realizadas por otros usuarios.
- ➢ El sistema deberá permitir realizar publicaciones, responder mensajes y reaccionar mediante “likes”.
- ➢ El sistema deberá permitir guardar publicaciones en favoritos.
- ➢ El sistema deberá permitir al administrador publicar contenido informativo dentro de la sección de seguimiento.

Configuración y personalización

- ➢ El sistema deberá permitir modificar opciones de configuración como notificaciones, creación de un alias, idioma y modificación del perfil

3.3 Requisitos no funcionales

Los requisitos no funcionales definen las características de calidad y comportamiento general que debe mantener Damagotchi para ofrecer una experiencia interactiva, visual y accesible para el usuario. Estos requisitos los agrupamos de la misma manera que los funcionales para facilitar la lectura.

Usabilidad

- ➢ La aplicación deberá ofrecer una interfaz intuitiva y fácil de utilizar.
- ➢ El diseño deberá facilitar la navegación entre las diferentes secciones de la aplicación.
- ➢ Los elementos visuales deberán ser claros y comprensibles para el usuario.
- ➢ La aplicación deberá mantener una estética visual basada en colores suaves y elementos fácilmente reconocibles

Rendimiento

- ➢ La aplicación deberá responder de forma fluida a las interacciones del usuario en dispositivos Android.
- ➢ Los cambios entre pantallas y secciones deberán realizarse sin tiempos de espera excesivos.
- ➢ El sistema deberá actualizar los medidores y estados del personaje en tiempo real sin afectar al rendimiento general de la aplicación.

Mantenibilidad

- ➢ El código de la aplicación deberá estar organizado de forma modular para facilitar futuras modificaciones y ampliaciones.
- ➢ La estructura del proyecto deberá permitir añadir nuevas funcionalidades sin afectar al funcionamiento principal del sistema.

Experiencia visual y sonora

- ➢ La aplicación podrá incorporar efectos visuales y sonoros para mejorar la experiencia del usuario.
- ➢ El sistema deberá utilizar transiciones suaves, indicadores visuales y mensajes dinámicos para representar el estado del personaje, el paso del tiempo y los cambios producidos durante el desarrollo del juego.

Seguridad y almacenamiento

- ➢ La aplicación deberá mantener la persistencia del progreso del juego entre sesiones y cargar desde la base de datos la información correspondiente a publicaciones, anuncios y contenido de seguimiento.

3.4 Análisis de alternativas tecnológicas

Para este proyecto se han valorado distintas alternativas en las decisiones tecnológicas más relevantes. A continuación se explica el razonamiento seguido en cada caso.

Motor de desarrollo

Alternativa	Valoración
LbGDX	Framework orientado al desarrollo de videojuegos multiplataforma. Se valoró inicialmente por su utilidad para crear mecánicas de juego tipo Tamagotchi y por la posibilidad de ampliar el proyecto a otras plataformas en el futuro. Además, permitía trabajar con Java, un lenguaje ya conocido y utilizado anteriormente durante el ciclo formativo, lo que facilitaba el inicio del proyecto. Sin embargo, resultaba menos adecuado para integrar de forma sencilla funcionalidades más propias de una aplicación móvil, como la comunidad, el seguimiento o el tablón de anuncios.
Android Studio	Entorno oficial para crear aplicaciones Android. Permite trabajar con Kotlin y Jetpack Compose, integrar navegación, pantallas, almacenamiento y conexión con base de datos de forma más directa. Aunque supuso comenzar a trabajar con tecnologías y estructuras nuevas, ofrecía una mayor flexibilidad y una mejor adaptación al enfoque final del proyecto.
Elección: Android Studio con desarrollo nativo Android, ya que se adapta mejor a Damagotchi al combinar modo juego, comunidad y tablón de anuncios dentro de una misma aplicación.	

Control de Versiones

Elección: GitHub, ya que permite mantener un mejor control de la evolución del proyecto, almacenar el código de forma segura y recuperar versiones anteriores en caso de errores o cambios importantes durante el desarrollo.

Gestión de datos

Alternativa	Valoración
PostgreSQL	Sistema de base de datos relacional robusto y adecuado para manejar información estructurada y consultas complejas. Se valoró por su estabilidad y capacidad para gestionar grandes volúmenes de datos. Sin embargo, requería una configuración y gestión más compleja para las necesidades actuales del proyecto.
Firebase	Plataforma orientada al desarrollo de aplicaciones móviles con sincronización de datos en tiempo real e integración sencilla con Android. Permitía gestionar publicaciones, anuncios y contenido compartido de forma más dinámica, además de facilitar futuras funcionalidades relacionadas con usuarios o almacenamiento online.
Alm. local	Se valoró como una opción sencilla para guardar el progreso del juego y el estado del personaje directamente en el dispositivo. Resultaba adecuado para manejar información básica sin necesidad de depender constantemente de una conexión externa.

Elección: Se optó por una **combinación** de **almacenamiento local** y **Firebase**. El progreso del modo juego y el estado del personaje se almacenan localmente en el dispositivo, mientras que Firebase se utiliza para gestionar publicaciones, anuncios y contenido compartido dentro de la aplicación.

4. Diseño

4.1 Arquitectura del sistema

El proyecto Damagotchi sigue el patrón de arquitectura MVVM (Model-View-ViewModel), recomendado para el desarrollo de aplicaciones Android. Este patrón permite separar las responsabilidades de cada capa, facilitando el mantenimiento, la escalabilidad y facilita las pruebas y el mantenimiento del código.

Capa Model
Representa los datos de la app. Pet.kt encapsula el estado del personaje y sus medidores. PetPrefs.kt persiste los datos localmente con DataStore. Firebase Firestore almacena los datos de comunidad y usuarios en la nube.

Capa ViewModel
Conecta los datos con la interfaz. PetViewModel gestiona la lógica del juego — acciones, medidores y evaluación final. TransicionViewModel controla el tiempo ficticio, el ciclo día/noche y los avisos por trimestre. Ambos exponen datos mediante StateFlow y SharedFlow.

Capa View
Construida con Jetpack Compose. Cada pantalla obtiene información de los flujos del ViewModel con collectAsState() y se actualiza automáticamente. La navegación se gestiona con Navigation Compose a través de AppNav.kt

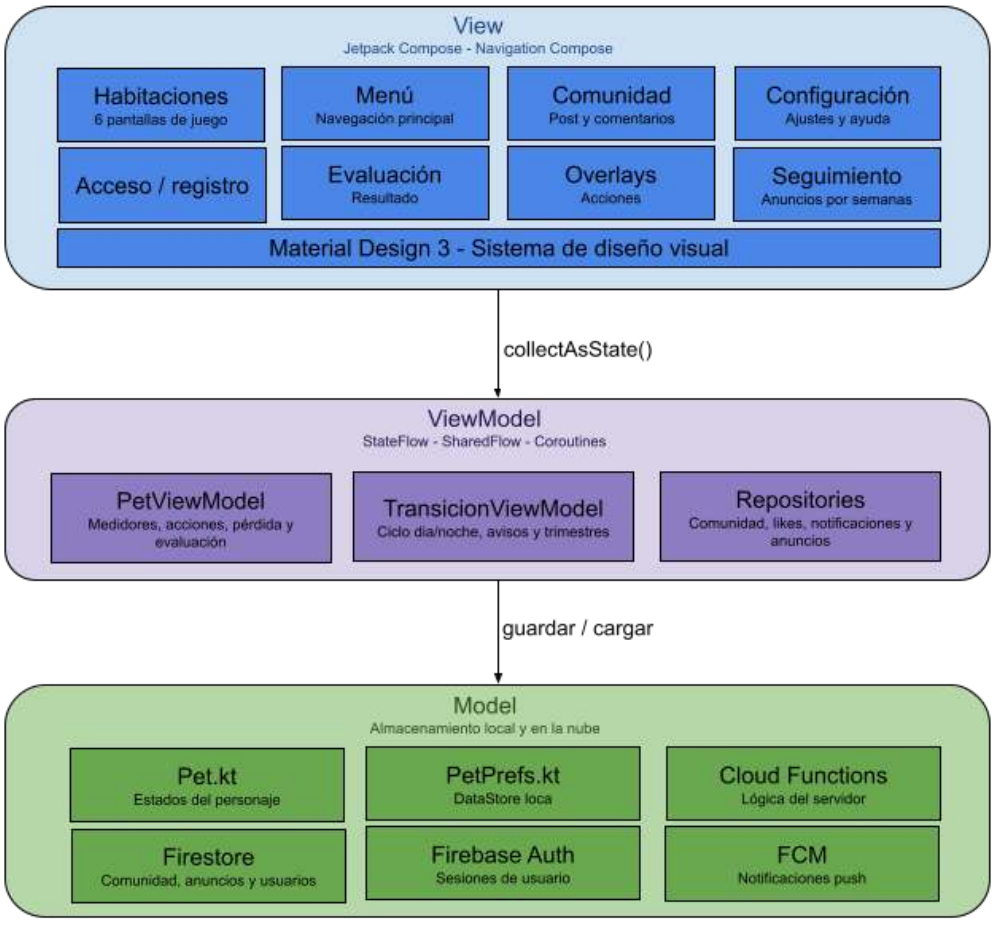

Imagen2. Diagrama de la arquitectura de Damagotchi

4.2 Modelo de datos

El modelo de datos de Damagotchi se organiza en diferentes entidades relacionadas con la gestión de usuarios, el progreso del modo juego, la comunidad y el sistema de seguimiento. La estructura diferencia las funcionalidades principales de la aplicación y cómo se relacionan entre sí.
La entidad Users almacena la información básica de los usuarios registrados dentro de la aplicación. Cada usuario dispone de un estado de juego asociado mediante la entidad Pet, encargada de guardar los diferentes medidores relacionados con el bienestar del personaje, como energía, hambre, sed, limpieza, actividad y descanso, además del progreso del embarazo dentro del juego.
La parte social de la aplicación se gestiona mediante las entidades Pub_Comunidad, Comentarios y Likes, permitiendo a los usuarios realizar publicaciones, interactuar con otros usuarios y reaccionar al contenido compartido dentro de la comunidad.
Por otro lado, la entidad Admin representa a los usuarios encargados de gestionar la sección de seguimiento de la aplicación. A través de la entidad Anuncios, el administrador publica contenido informativo, consejos y recomendaciones relacionados con las distintas etapas del embarazo y el bienestar prenatal.
Este modelo permite mantener separadas las funcionalidades del modo juego, la comunidad y el sistema de seguimiento, facilitando una organización más clara de la información dentro de la aplicación.

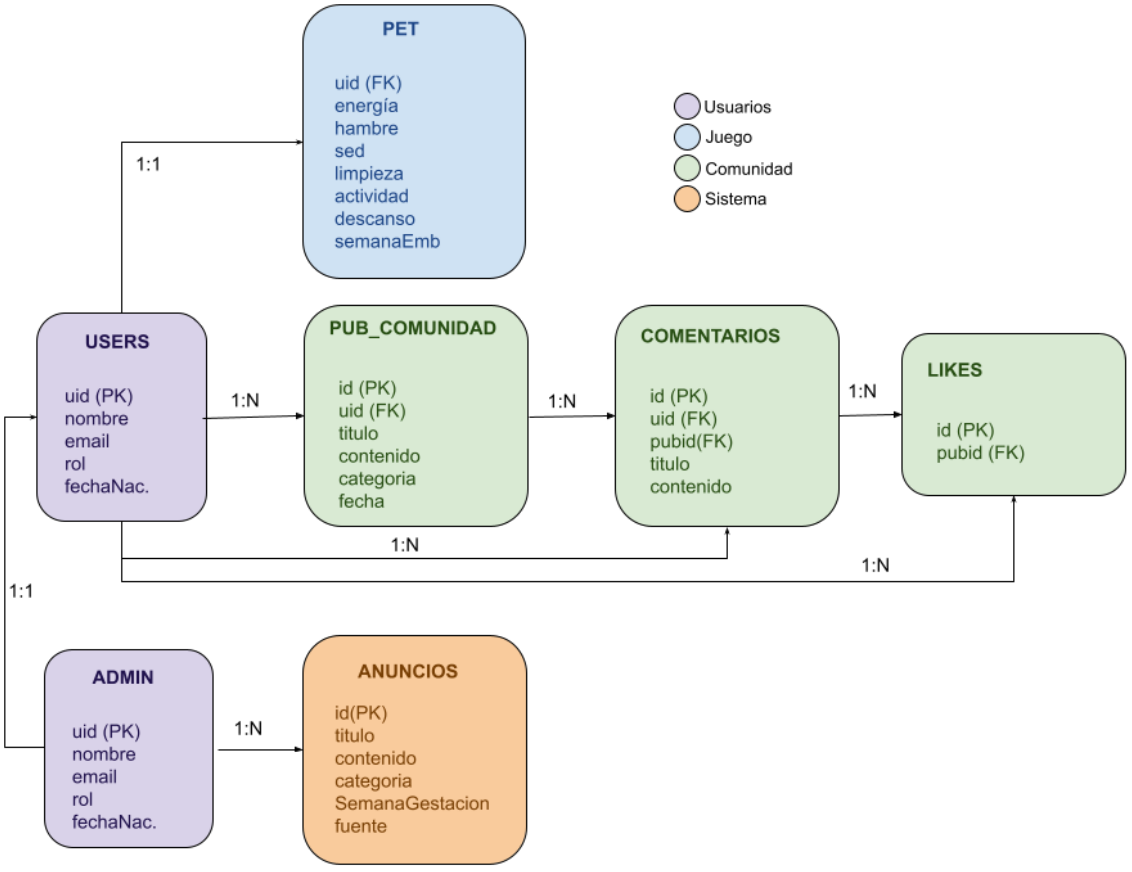

Imagen3. Diagrama Entidad - Relación

4.3 Diseño interfaz

La interfaz de Damagotchi se ha diseñado buscando una experiencia sencilla, visual y accesible. Al tratarse de una aplicación que combina juego, comunidad y tablón de anuncios, la app se ha organizado en pantallas diferenciadas para que el usuario pueda acceder fácilmente a cada funcionalidad sin mezclar demasiada información en un mismo espacio.

Las pantallas principales que se documentan son:

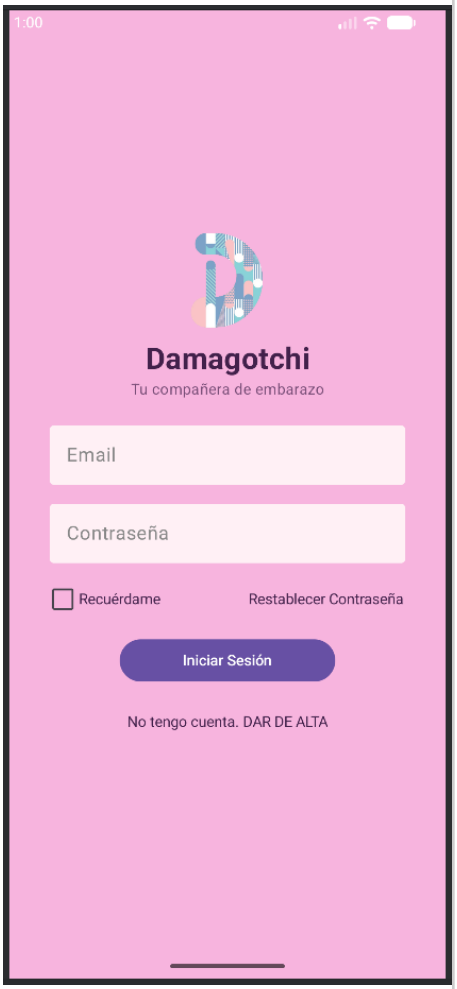

Imagen4. Login

Login. El usuario podrá darse de alta la primera vez y posteriormente usar sus credenciales para acceder a su perfil.

Imagen5. Menú principal.

Menú principal. Se muestran las tres sesiones principales por las que el usuario puede navegar libremente

Imagen6. Modo juego

Imagen7. Mi embarazo

Modo juego. Pantalla central del juego;
se representa una estancia donde se desarrolla el juego y en se ven los botones de acción correspondientes y los medidores de cuidados del personaje.

Seguimiento. Tablón de anuncios administrado por el Admin y que va mostrando información adaptada a las semanas reales del embarazo

Imagen8. Comunidad

Comunidad. Espacio en el que el usuario puede interactuar haciendo publicaciones, comentarios o guardando publicaciones en favoritos.

Imagen9. Configuración

Configuración. Pantalla en la que se pueden ajustar las preferencias del usuario

5. Desarrollo

5.1 Estructura del proyecto

El proyecto se organiza siguiendo una estructura basada en el patrón MVVM, separando la interfaz, la lógica de negocio y la gestión de datos en diferentes paquetes. Esta organización facilita el mantenimiento del código, la reutilización de componentes y la incorporación de nuevas funcionalidades.

La carpeta principal com.example.damagotchi_26 contiene la estructura principal de la aplicación. Dentro de ella se diferencian varios paquetes principales:

- data: contiene las clases relacionadas con el manejo y almacenamiento de datos.
- domain: agrupa parte de la lógica principal y modelos utilizados por la aplicación.
- navigation: gestiona la navegación entre pantallas mediante Navigation Compose.
- repository: actúa como intermediario entre las fuentes de datos y los ViewModel.
- viewmodel: contiene la lógica de estado y comunicación entre la interfaz y los datos.

La carpeta “ui” agrupa todas las pantallas y componentes visuales de la aplicación. Dentro de ella se encuentran organizadas las distintas secciones principales, como el login, menú, estancias comunidad, seguimiento, configuración o evaluación final del juego. Además, incluye componentes reutilizables y elementos relacionados con el diseño visual de la interfaz.
Por otro lado, la carpeta res almacena los recursos gráficos y de configuración de Android. Finalmente, MainActivity actúa como punto de entrada principal de la aplicación.

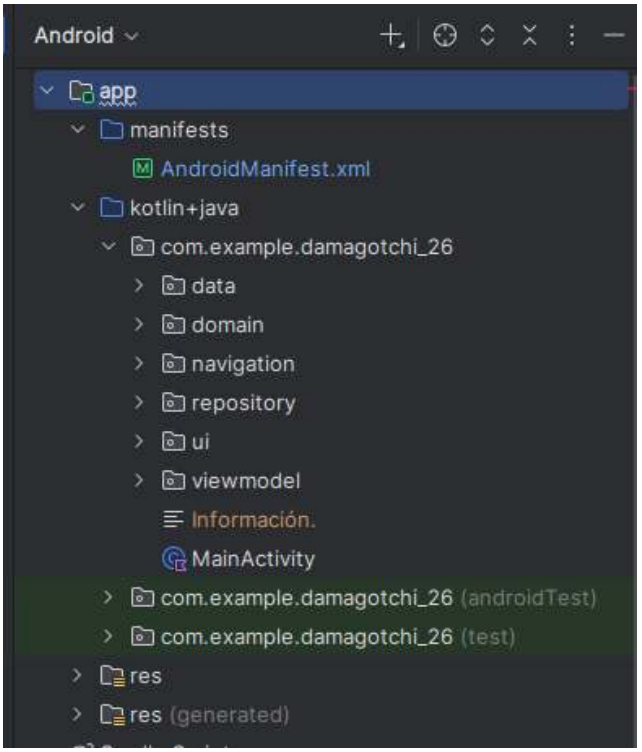

Imagen10. Estructura de carpetas Damagotchi

5.2 Implementación de funcionalidades

5.2.1 Sistema principal de juego

Personaje virtual

El modo juego representa la funcionalidad principal de Damagotchi y se basa en el cuidado diario de un personaje virtual durante las distintas etapas del embarazo. El usuario debe mantener equilibrados los medidores (energía, hambre, hidratación, higiene, actividad y descanso) relacionados con el bienestar del personaje.

El comportamiento del personaje se controla mediante un sistema de estados que varía según los valores de los medidores y las acciones realizadas por el usuario. Dependiendo de la situación, el personaje puede evolucionar entre diferentes estados de bienestar, enfermedad o recuperación, afectando directamente al desarrollo de la partida.

Para representar esta lógica se desarrolló el siguiente diagrama de estados:

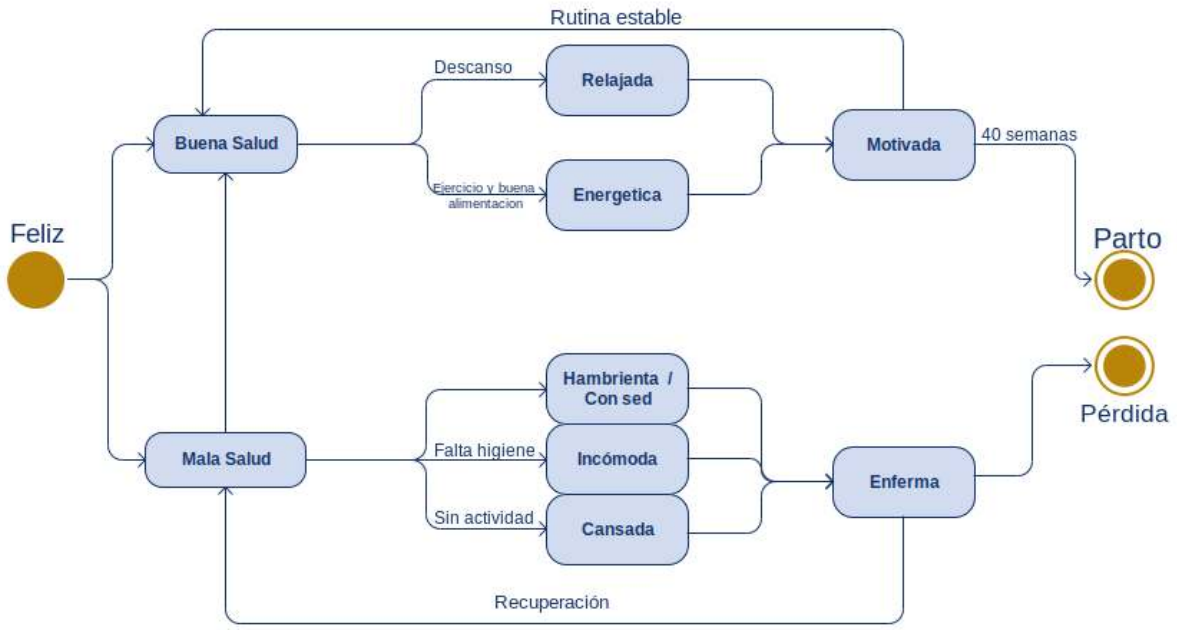

Imagen11. Diagrama de estados del personaje en Damagotchi.

Inicialmente, el personaje fue planteado como un elemento estático cuya única evolución visual consistía en el aumento progresivo del tamaño de la barriga durante los cambios de trimestre. Sin embargo, a medida que avanzó el desarrollo del proyecto, el sistema evolucionó incorporando pequeños desplazamientos laterales y diferentes expresiones faciales, como superfeliz, feliz o triste, dependiendo del estado general del personaje y de los porcentajes de sus medidores.

Además, cuando el personaje entra en un estado de enfermedad, el sistema reajusta parcialmente los medidores para permitir una recuperación progresiva y evitar una finalización inmediata de la partida.

Medidores

El sistema principal del juego se basa en diferentes medidores relacionados con el bienestar del personaje, como energía, hambre, sed, limpieza, actividad y descanso. Cada acción realizada por el usuario afecta directamente a estos valores, influyendo en el estado general del personaje y en la evolución del embarazo dentro del juego.
Inicialmente, los medidores fueron planteados como una lista fija de iconos acompañados de barras de porcentaje visibles de forma permanente en el lateral izquierdo de la pantalla. Sin embargo, durante el desarrollo se observó que esta distribución ocupaba demasiado espacio visual y recargaba la interfaz.

Por este motivo, se decidió mantener únicamente los iconos de los medidores en la columna lateral izquierda, mostrando el porcentaje detallado únicamente cuando el usuario pulsa sobre cada uno de ellos. Además, en la parte superior de la pantalla se muestran en formato texto los medidores directamente relacionados con la estancia actual del personaje. Por ejemplo, en la cocina se priorizan hambre, sed y energía, mientras que en el dormitorio se muestran principalmente descanso y energía.

Esta decisión permitió mantener una interfaz más limpia y organizada, reforzando además la relación entre las diferentes estancias y las necesidades del personaje.

Acciones

Respecto a las acciones del usuario, inicialmente los botones fueron implementados de forma básica, limitándose a modificar directamente los porcentajes de los medidores. Posteriormente, una vez desarrolladas las funcionalidades principales del proyecto, estas interacciones evolucionaron incorporando animaciones, desplazamientos y un comportamiento más dinámico, buscando una experiencia más visual e interactiva dentro del juego.

Uno de los aspectos más importantes del desarrollo fue la implementación de un sistema de acciones interactivas dentro de las distintas estancias. A diferencia de un sistema basado únicamente en botones tradicionales, las acciones fueron diseñadas para ofrecer una interacción más dinámica mediante gestos táctiles, animaciones y pequeños minijuegos desarrollados con Jetpack Compose.

Cada acción se muestra mediante overlays superpuestos sobre la habitación actual, permitiendo mantener visible el entorno principal del juego y mejorando la continuidad visual de la interfaz. Estos overlays se gestionan mediante sistemas de estados independientes para evitar que varias acciones puedan ejecutarse simultáneamente.

Muchas de las acciones utilizan sistemas de arrastre táctil mediante detectDragGestures, permitiendo que los elementos sigan el movimiento real del dedo del usuario. Acciones como la ducha, la crema o el cepillo de dientes incorporan además efectos visuales y partículas animadas para reforzar la sensación de interacción.

Otras acciones fueron planteadas como pequeños minijuegos interactivos. Entre ellos destaca el sistema de preparación de comida, donde diferentes ingredientes aparecen

dinámicamente en pantalla y deben ser capturados por el usuario antes de desaparecer. También se implementaron acciones relacionadas con dibujo y música utilizando Canvas, animaciones y actualización continua de estados mediante corrutinas.

Todas estas mecánicas fueron desarrolladas utilizando únicamente herramientas nativas de Jetpack Compose y Android, sin recurrir a librerías externas.

Temporalización

Durante las primeras fases del desarrollo, el modo juego fue planteado con un sistema de estancias independientes que representaban diferentes espacios de un hogar, incluyendo dormitorio, cocina, baño, exteriores y consulta médica. La intención inicial era integrar dentro de estas estancias tanto las mecánicas del juego como los consejos y anuncios relacionados con el seguimiento del embarazo.

Sin embargo, a medida que avanzó el desarrollo, se observó que esta estructura mezclaba dos líneas temporales diferentes: el tiempo ficticio del juego y el seguimiento basado en el embarazo real del usuario. Esto dificultaba tanto la organización de la aplicación como la claridad de la experiencia de usuario.

Por este motivo, se decidió separar ambas funcionalidades en sistemas independientes. El modo juego mantiene una evolución virtual del embarazo de 40 semanas, pero comprimida en una duración total de 30 días reales de juego, divididos en “tres trimestres”, mientras que la información de seguimiento y recomendaciones se trasladó a una sección independiente conectada a Firebase y basada en el tiempo real del embarazo.

Eventos especiales y progresión del embarazo

Con el objetivo de reforzar la sensación de progreso y acompañamiento durante el embarazo virtual, se implementó un sistema de eventos especiales vinculados a semanas concretas del juego. Estos eventos aparecen en forma de diálogos visuales y se muestran únicamente una vez, almacenando en DataStore si ya han sido vistos para evitar repeticiones en sesiones posteriores.

Por un lado, se desarrollaron mensajes automáticos asociados a los cambios de trimestre. Estos avisos aparecen en tres momentos clave del embarazo: la semana 1, que marca el inicio del recorrido; la semana 13, correspondiente al comienzo del segundo trimestre; y la semana 28, que señala el inicio del tercero. Cada mensaje está adaptado al rol del usuario y utiliza un diseño visual común basado en un diálogo oscuro con cabecera en tonos rosados, iconografía representativa y mensajes personalizados según la etapa del embarazo.

Además, se implementaron eventos especiales de ecografía vinculados a semanas concretas del embarazo. Estas pantallas muestran una imagen de ecografía junto a información relacionada con el desarrollo del bebé en esa etapa. Las ecografías se sitúan en las semanas 7, 14, 22 y 32, coincidiendo con momentos especialmente relevantes dentro del seguimiento real del embarazo.

Como elemento especialmente personal del proyecto, las imágenes utilizadas corresponden a ecografías reales del hijo de la desarrolladora. Su incorporación busca aportar una experiencia más cercana y auténtica, reforzando la sensación de acompañamiento y evolución del embarazo dentro del juego.

Para evitar conflictos entre eventos simultáneos, el sistema incorpora una prioridad de visualización. En aquellas semanas donde podrían coincidir distintos avisos, como el cambio de trimestre y una ecografía, el sistema reorganiza automáticamente su aparición para evitar saturar la experiencia del usuario.

Sistema de pérdida y recuperación

El sistema principal del juego incorpora una mecánica de control de riesgo basada en los medidores del personaje. Cada día del juego, el sistema comprueba el estado general del embarazo analizando los niveles de energía, hambre, sed, limpieza, actividad y descanso.
Cuando tres o más medidores permanecen por debajo del 20% durante varios días consecutivos, el sistema activa una situación de riesgo. En la primera ocasión, se muestra un aviso indicando que el embarazo necesita mayores cuidados y el usuario puede continuar la partida si lo desea. Para ello, los medidores se reajustan automáticamente a valores intermedios, permitiendo una recuperación progresiva del personaje.

Si la misma situación vuelve a repetirse después del aviso, el sistema activa una pérdida definitiva del embarazo y la partida finaliza mostrando automáticamente la evaluación final correspondiente.

A nivel interno, esta mecánica se controla mediante contadores que registran los días consecutivos en estado crítico y el número de veces que el personaje ha alcanzado una situación de pérdida. Esta lógica se implementa dentro de TransicionViewModel, encargado de gestionar el paso del tiempo y los estados generales del juego.

Evaluación final

Al alcanzar la semana 40 del embarazo virtual o tras una pérdida definitiva, el juego muestra una pantalla de evaluación final que resume el desempeño del usuario durante toda la partida.

La evaluación se calcula a partir del promedio general de los seis medidores del personaje — energía, hambre, sed, limpieza, actividad y descanso — acumulados progresivamente a lo largo de los días del juego. Cada vez que avanza un nuevo día, el sistema registra automáticamente el estado actual de los medidores para calcular posteriormente una valoración global del cuidado recibido durante el embarazo.

Según el promedio obtenido, el sistema muestra una de las siguientes evaluaciones:

- Excelente — promedio igual o superior al 75%.
- Bien — promedio igual o superior al 50%.
- Mejorable — promedio igual o superior al 30%.
- Deficiente — promedio inferior al 30%.

- Pérdida del embarazo — en caso de producirse una pérdida definitiva durante la partida.

Para mantener esta información incluso tras cerrar la aplicación, los datos acumulados se almacenan localmente mediante DataStore junto al resto del progreso del personaje.

La pantalla de evaluación se activa automáticamente al alcanzar la semana 40 del embarazo virtual o tras una pérdida definitiva, cerrando así el ciclo principal del modo juego y ofreciendo al usuario una valoración final basada en las decisiones tomadas durante toda la experiencia.

Evolución

El modo juego partió de un sistema básico de botones que modificaban directamente los medidores del personaje, sin animaciones ni interacciones adicionales. A lo largo del desarrollo evolucionó hasta convertirse en una experiencia táctil completa.

La primera mejora fue el personaje dinámico, que pasó de ser una imagen estática a moverse lateralmente y cambiar de apariencia según el trimestre del embarazo y el estado emocional, permitiendo al usuario percibir visualmente el estado del juego sin consultar constantemente los medidores.

El siguiente paso fue la transformación de las acciones. Cada habitación recibió un sistema de overlays con minijuegos táctiles propios — arrastrar la esponja dejando burbujas, frotar el cepillo de dientes, inclinar el vaso para beber, llevar la cuchara a la boca del personaje o atrapar ingredientes cayendo en la cocina. El dormitorio introdujo la distinción entre siesta y dormir con animaciones diferenciadas. El parque añadió animaciones por sprite al caminar y una secuencia interactiva de estiramientos. El salón fue la habitación que más evolucionó, sustituyendo sus acciones originales por dos minijuegos: piano y pintura libre.

Por último se implementaron los sistemas de consecuencias, la pérdida del embarazo al descuidar los medidores y la evaluación final al llegar a la semana 40, cerrando el ciclo completo del juego.

Elementos sonoros y música interactiva

Con el objetivo de reforzar la inmersión dentro del modo juego, se incorporó un sistema de sonido compuesto por música de fondo, efectos ambientales y sonidos interactivos asociados a determinadas acciones.

Por un lado, el modo juego incluye un hilo musical continuo implementado mediante MediaPlayer de Android. La música se inicia automáticamente al acceder al sistema de estancias y permanece reproduciéndose en bucle durante toda la sesión de juego. Al abandonar esta sección, el sistema libera automáticamente los recursos para evitar consumo innecesario de memoria o reproducción en segundo plano.

Además, varias acciones incorporan sonidos específicos para reforzar la interacción del usuario. En el baño y la ducha se añadieron efectos ambientales relacionados con el agua,

mientras que el dormitorio incorpora un sonido de respiración durante la acción de dormir para reforzar la sensación de descanso nocturno. La siesta, en cambio, se mantuvo sin sonido para diferenciarla como una acción más breve y ligera.

La acción de piano también incorpora efectos sonoros interactivos, reproduciendo distintas notas musicales al pulsar correctamente las teclas del minijuego. De esta forma, el sistema combina tanto sonidos ambientales continuos como efectos puntuales asociados a determinadas acciones del usuario.

Persistencia local

El progreso principal del modo juego se almacena de forma local dentro del dispositivo utilizando DataStore, permitiendo conservar automáticamente el estado del personaje entre sesiones sin necesidad de conexión a internet.

Entre los datos almacenados se encuentran los valores de los medidores, el progreso del embarazo virtual, el estado general del personaje y diferentes configuraciones relacionadas con la partida. Esta persistencia permite que el usuario pueda continuar el juego desde el mismo punto al volver a acceder a la aplicación.

Inicialmente se valoró almacenar toda la información mediante Firebase, unificando el sistema del juego con las funcionalidades online de la aplicación. Sin embargo, durante el desarrollo se decidió separar ambos sistemas para evitar mezclar la lógica local del modo juego con las funcionalidades sociales y de seguimiento.

El almacenamiento local permitió simplificar la gestión del estado del personaje, mejorar el rendimiento de la aplicación y garantizar el acceso al modo juego incluso sin conexión a internet. Por otro lado, Firebase quedó reservado para las funcionalidades relacionadas con la comunidad, autenticación y sistema de seguimiento basado en datos dinámicos.

5.2.2 Sistema de seguimiento

Durante las primeras fases del desarrollo, esta funcionalidad iba a integrarse directamente dentro de las estancias del juego, especialmente en la consulta médica. Sin embargo, a medida que avanzó el proyecto, se observó que mezclar el tiempo real del embarazo con la temporalidad ficticia del modo juego dificultaba tanto la organización de la aplicación como la claridad de la experiencia de usuario. Por este motivo, se decidió separar ambas funcionalidades en sistemas independientes.

El contenido del seguimiento se organiza mediante anuncios almacenados en Firebase Firestore dentro de la colección anuncios_seguimiento. Cada publicación incluye información como título, contenido, categoría, semana de gestación y la fuente relacionada con el tema tratado.

La pantalla carga dinámicamente estos anuncios y los filtra automáticamente según la semana actual del embarazo del usuario, mostrando de forma acumulada el contenido relevante a cada etapa. Además, el sistema incorpora un buscador de texto, filtros por

categorías mediante chips desplazables y un sistema de favoritos para guardar publicaciones de interés.

Los usuarios con rol de administrador disponen de una pantalla adicional desde la que pueden crear nuevas publicaciones, seleccionando tanto la categoría como la semana del embarazo a la que irá dirigida.

Visualmente, el sistema de seguimiento fue diseñado utilizando tarjetas dinámicas y scroll vertical, buscando una experiencia de lectura más clara y diferenciada respecto al modo juego. Esta funcionalidad refuerza el componente educativo de la aplicación y complementa la experiencia interactiva principal con información real y actualizada sobre el embarazo.

5.2.3 Comunidad

La sección de comunidad fue desarrollada como un espacio de interacción entre usuarios, permitiendo compartir experiencias, dudas y consejos relacionados con el embarazo y el bienestar durante la gestación. Esta funcionalidad actúa como una pequeña red social integrada directamente dentro de la aplicación.

Los usuarios pueden crear publicaciones indicando título, contenido y categoría. Cada publicación permite además la interacción mediante comentarios, likes y sistema de favoritos, facilitando una experiencia más participativa y dinámica dentro de la comunidad.

La pantalla principal muestra las publicaciones ordenadas cronológicamente mediante un feed vertical con scroll continuo. Además, se incorporó un buscador de texto y diferentes filtros para facilitar la localización de contenido concreto dentro de la comunidad.

El sistema también incorpora opciones relacionadas con la privacidad del usuario. Desde la pantalla de configuración, cada persona puede decidir si desea mostrar su nombre real o un alias dentro de las publicaciones realizadas en la comunidad.

Uno de los aspectos más importantes de esta funcionalidad fue la implementación de un sistema de notificaciones push utilizando Firebase Cloud Messaging y Firebase Cloud Functions. Cuando un usuario crea una nueva publicación o realiza un comentario, una Cloud Function detecta automáticamente el evento en Firestore y envía una notificación al resto de usuarios que tengan activadas estas alertas.

Además de las notificaciones push del sistema, la propia pantalla de comunidad incorpora una campana de notificaciones interna que muestra un contador con las interacciones pendientes de leer. Al pulsar sobre ella, se despliega un panel inferior con las notificaciones recientes y el usuario puede navegar directamente hasta la publicación relacionada.

Finalmente, el sistema incorpora diferentes roles de usuario almacenados en Firebase. Los administradores disponen de permisos adicionales para moderar contenido y gestionar determinadas funcionalidades relacionadas con la comunidad.

Población de la base de datos de seguimiento

Dentro de la propia aplicación se desarrolló una pantalla exclusiva para administradores que permite crear y publicar anuncios del sistema de seguimiento directamente desde la propia aplicación y Firebase Firestore. Este sistema está pensado para la gestión normal del contenido, permitiendo añadir, modificar o ampliar publicaciones de forma sencilla desde la interfaz de la aplicación.
Sin embargo, durante las primeras fases del desarrollo fue necesario introducir una gran cantidad de anuncios iniciales para cubrir las distintas semanas del embarazo y categorías informativas. Debido a ello, realizar esta carga manualmente desde la aplicación resultaba demasiado lento y poco práctico.

Para agilizar este proceso se desarrolló un script de automatización denominado subirAnuncios.js utilizando Node.js y Firebase Admin SDK. Este script permite subir múltiples anuncios automáticamente a Firestore en un único proceso, definiendo previamente todos los datos dentro de un array estructurado.

Gracias a este sistema fue posible poblar rápidamente la colección anuncios_seguimiento con una base inicial completa de contenido, facilitando las pruebas y el desarrollo del sistema de seguimiento.

De esta forma, el proyecto combina ambos métodos de gestión según las necesidades: la interfaz integrada en la aplicación para la administración habitual del contenido y el script de Node.js para cargas masivas o inserciones múltiples de datos.

5.2.4 Navegación y flujo de pantallas

La navegación de la aplicación se implementó utilizando Navigation Compose, centralizando toda la gestión de pantallas dentro de un único archivo AppNav.kt, encargado de definir el grafo principal de navegación de la aplicación.

Las diferentes rutas se organizaron mediante una sealed class llamada Route, donde cada pantalla de la aplicación dispone de una ruta propia identificada mediante un path. Esta estructura permitió mantener un sistema de navegación más ordenado, seguro y fácil de mantener a medida que el proyecto fue creciendo.

El sistema utiliza un NavHost junto a un NavController para gestionar la navegación y la pila de pantallas activas. Cada sección principal de la aplicación — modo juego, comunidad, seguimiento, ayuda o configuración — se registra mediante composables independientes conectados al sistema principal de navegación.

Al iniciar la aplicación, el sistema comprueba automáticamente si el usuario mantiene una sesión activa utilizando la configuración almacenada localmente mediante DataStore. Dependiendo de este estado, la aplicación redirige automáticamente al menú principal o a la pantalla de login.

Durante el desarrollo se tomaron varias decisiones orientadas a mejorar la experiencia de navegación y evitar problemas habituales relacionados con la pila de pantallas. Por ejemplo, al cerrar sesión, la navegación utiliza popUpTo(0) { inclusive = true }, para limpiar completamente el historial de navegación e impedir que el usuario pueda regresar a pantallas protegidas utilizando el botón atrás del dispositivo. Además, se implementó launchSingleTop = true, un sistema que evita la duplicación accidental de pantallas cuando el usuario pulsa repetidamente sobre una misma opción del menú.

Visualmente, la navegación fue diseñada buscando transiciones simples y fluidas entre pantallas, manteniendo una estructura clara y coherente con el resto de la interfaz de la aplicación.

Para representar de forma más clara la estructura general de navegación y el flujo principal entre pantallas, se desarrolló el siguiente diagrama simplificado del funcionamiento de la aplicación.

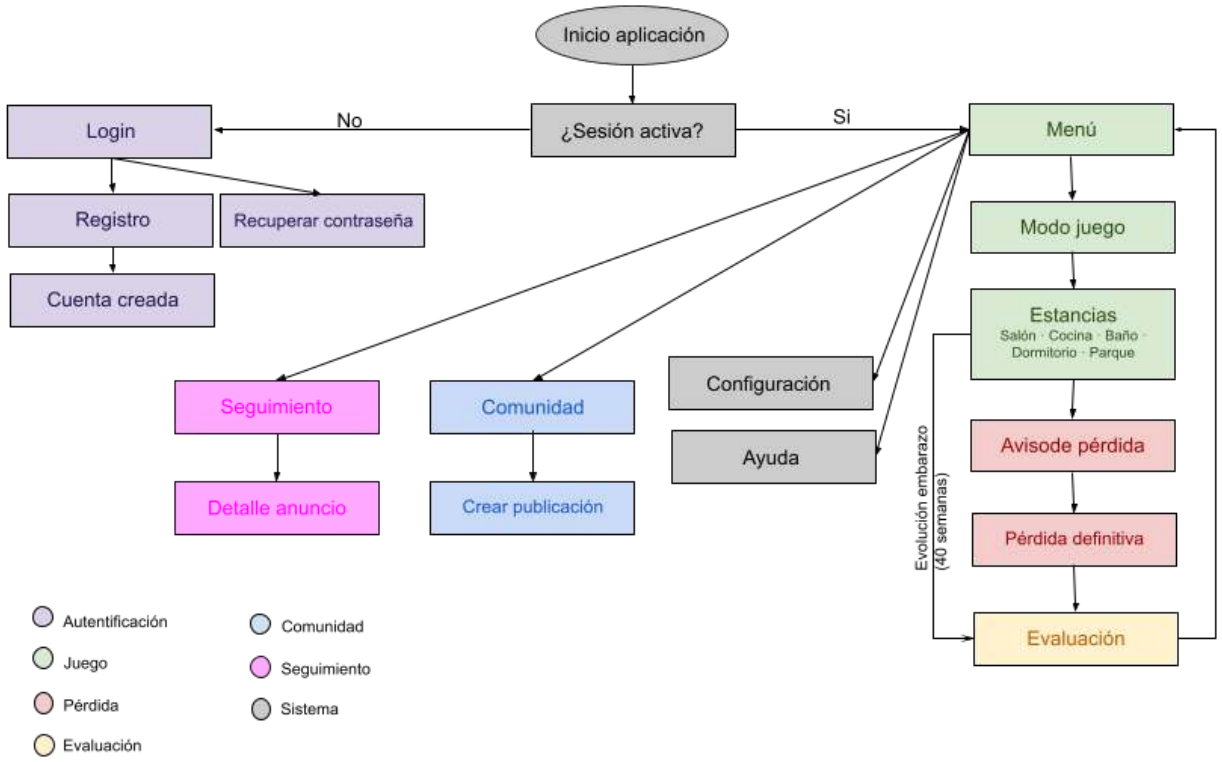

Imagen12. Flujo de navegación de Damagotch

5.2.5 Usuarios y autenticación

La autenticación de usuarios se implementó utilizando Firebase Authentication, encargado de gestionar el registro, inicio y cierre de sesión de forma segura dentro de la aplicación.

Registro

Durante el registro, el usuario introduce información básica como nombre, correo electrónico, contraseña, fecha de nacimiento, fecha de última regla, sexo del bebé y rol dentro de la aplicación. Las credenciales de acceso se almacenan mediante Firebase Authentication, mientras que el perfil completo del usuario se guarda en Firestore dentro de la colección users.

Inicio de sesión

El inicio de sesión se realiza mediante correo electrónico y contraseña utilizando Firebase Auth. Además, la aplicación incorpora una opción de “Recuérdame”, que permite mantener la sesión iniciada utilizando almacenamiento local mediante DataStore. Al iniciar la aplicación, el sistema comprueba automáticamente este estado para acceder directamente al menú principal sin pasar nuevamente por el login.

Cierre de sesión

Al cerrar sesión, la aplicación elimina la autenticación activa de Firebase y limpia los datos locales relacionados con la sesión persistente almacenada mediante DataStore.

Roles de usuario

El sistema distingue diferentes roles dentro de la aplicación, mamá, papá, acompañante y administrador. Dependiendo del rol, determinadas secciones adaptan parte del contenido mostrado al usuario.

Los administradores disponen además de permisos adicionales relacionados con la creación y gestión de publicaciones dentro del sistema de seguimiento y la comunidad.

Restablecimiento de contraseña

La aplicación incorpora un sistema de recuperación de contraseña mediante correo electrónico utilizando Firebase Authentication, permitiendo al usuario restablecer el acceso a su cuenta desde la pantalla de login.

Página web de presentación del proyecto

Como complemento a la aplicación, se desarrolló una página web estática para facilitar la presentación y distribución del proyecto. La página está alojada en GitHub Pages y fue creada con HTML, sin utilizar frameworks externos.

El diseño mantiene la misma línea visual que Damagotchi, utilizando una paleta basada en tonos morados y rosas para conservar la coherencia con la aplicación Android. La web incluye una sección principal con capturas reales de la app, una zona de estadísticas del proyecto, un resumen de las funcionalidades principales, un apartado de descarga del APK y una sección de contacto.

Además, la página se diseñó de forma responsive para adaptarse correctamente tanto a dispositivos móviles como a pantallas de escritorio.

La publicación se realizó mediante GitHub Pages a partir del repositorio público damagotchi_web, activando el despliegue desde la rama master. La página está disponible en:

https://iria141.github.io/damagotchi_web

5.3 Pruebas

Durante el desarrollo de Damagotchi no se implementaron pruebas unitarias automatizadas. La verificación del funcionamiento de la aplicación se realizó principalmente mediante pruebas manuales, utilizando las previews de Jetpack Compose y ejecutando la aplicación en distintos dispositivos Android con diferentes tamaños de pantalla y versiones del sistema.

Las pruebas se centraron en comprobar el correcto funcionamiento de las funcionalidades principales, como el registro e inicio de sesión, la navegación entre pantallas, el modo juego, la evolución de los medidores, la persistencia del progreso, la comunidad y el sistema de seguimiento.

También se realizaron pruebas visuales sobre la interfaz, revisando que las pantallas se adaptasen correctamente a diferentes resoluciones y que los elementos principales, como botones, tarjetas, medidores, overlays y animaciones, se mostrasen correctamente.

En el modo juego se probaron manualmente las diferentes acciones interactivas, comprobando el comportamiento de los gestos táctiles, los desplazamientos del personaje, los minijuegos y los avisos relacionados con los medidores. Estas pruebas permitieron detectar y corregir problemas de distribución visual, navegación y respuesta de algunas acciones.

Además, se verificó el funcionamiento de Firebase mediante pruebas manuales de registro, login, creación de publicaciones, comentarios, likes, favoritos y carga de anuncios en el sistema de seguimiento. De esta forma se comprobó que los datos se almacenaban y recuperaban correctamente desde la aplicación.

Aunque no se utilizaron pruebas automatizadas, este proceso de comprobación manual permitió validar el funcionamiento general de la aplicación y corregir errores durante el desarrollo.

Problemas encontrados y soluciones aplicadas

Durante el desarrollo aparecieron diferentes problemas relacionados principalmente con la navegación, la gestión de estados y las interacciones táctiles dentro del modo juego.

Uno de los principales problemas surgió al mezclar la temporalidad ficticia del juego con el seguimiento basado en tiempo real, lo que obligó a separar ambas funcionalidades en

sistemas independientes para simplificar la estructura de la aplicación y mejorar la experiencia de usuario.

También aparecieron problemas relacionados con overlays simultáneos y duplicación de pantallas en la navegación, solucionados mediante sistemas de control de estados y limpieza de la pila de navegación.

En las acciones táctiles, algunas implementaciones iniciales producían desfases entre el dedo del usuario y los elementos arrastrables. Este comportamiento se corrigió ajustando manualmente los offsets y la conversión de coordenadas dentro de Jetpack Compose.
Durante la fase final de pruebas en dispositivos reales se detectó que algunos elementos de la interfaz no se adaptaban correctamente a pantallas de distintos tamaños. El problema principal estaba relacionado con el uso de valores fijos en dp, que funcionaban correctamente en el dispositivo de desarrollo, pero provocaban desajustes en pantallas más pequeñas o más grandes.
Para solucionarlo, se realizaron varias mejoras responsive. En las pantallas de autenticación, como login, registro o restablecimiento de contraseña, se sustituyeron paddings horizontales fijos por anchuras proporcionales mediante fillMaxWidth, consiguiendo que los botones se adaptasen mejor al tamaño disponible.
También se ajustó el panel lateral de medidores y los botones de acción de las estancias, que inicialmente utilizaban tamaños y espaciados fijos, provocando problemas de distribución en algunos dispositivos. En el caso de los medidores laterales, se utilizó BoxWithConstraints para calcular el espacio disponible y distribuir los iconos de forma proporcional según el tamaño de pantalla. Por otro lado, en los botones de acción se modificó su distribución mediante SpaceEvenly, mejorando su adaptación a diferentes resoluciones y evitando problemas de solapamiento o falta de espacio.

6. Conclusiones

6.1 Conclusiones generales del proyecto

El desarrollo de Damagotchi ha supuesto uno de los proyectos más completos y exigentes que he realizado hasta ahora. Aunque la idea inicial partía de un sistema relativamente sencillo basado en el cuidado de un personaje virtual, el proyecto fue evolucionando progresivamente hasta incorporar funcionalidades mucho más complejas relacionadas con interacción táctil, comunidad, seguimiento informativo y sincronización online.

Uno de los mayores retos durante el desarrollo fue intentar integrar dentro del propio modo juego los anuncios y recomendaciones relacionados con el embarazo real. Inicialmente ambas funcionalidades compartían una misma estructura, pero a medida que el proyecto avanzó se comprobó que mezclar la temporalidad ficticia del juego con el seguimiento real del embarazo hacía la aplicación demasiado confusa tanto a nivel técnico como de experiencia de usuario. Tras valorar esta situación junto al tutor del proyecto, decidí separar ambas funcionalidades en sistemas independientes, convirtiéndose finalmente en una de las decisiones más importantes del desarrollo. Además, esta separación me ayudó mucho a aclarar ideas y a entender de una forma más ordenada cómo debía estructurar realmente las funcionalidades de la aplicación.

A nivel técnico, el proyecto también supuso un reto importante. Aunque Android Studio era un entorno que ya conocía previamente por haber trabajado con Java y XML, las tecnologías principales utilizadas en este proyecto eran completamente nuevas para mí, especialmente Kotlin, Jetpack Compose y Firebase. Comprender cómo funcionaban y aprender a utilizarlas correctamente requirió bastante tiempo de dedicación y aprendizaje continuo durante todo el desarrollo.

Uno de los aspectos que más satisfacción me produjo fue ver cómo el proyecto iba evolucionando poco a poco desde algo muy básico hasta una aplicación mucho más dinámica e interactiva. Especialmente importante fue la implementación de los overlays y las acciones táctiles, ya que fue el momento en el que realmente empecé a sentir que el proyecto se estaba convirtiendo en un juego y no solo en una aplicación funcional.

También considero que este proyecto me ha ayudado mucho a aprender cómo afrontar desarrollos de mayor tamaño. Al ser el primer proyecto real que realizo, muchas decisiones iniciales probablemente las plantearía de otra forma actualmente, pero precisamente eso forma parte del aprendizaje obtenido durante el proceso. Gracias al proyecto he aprendido a no intentar construir todo desde el principio, sino a comenzar por una base simple e ir desarrollándola progresivamente.

En general, considero que el resultado final ha sido incluso mejor de lo que esperaba inicialmente. Aunque todavía existen aspectos que podrían ampliarse o mejorarse en futuras versiones, creo que Damagotchi ha conseguido combinar correctamente la parte lúdica del modo juego con el componente educativo, social e informativo planteado desde el inicio del proyecto.

6.2 Consecución de objetivos

En términos generales, considero que los objetivos planteados al inicio del proyecto se han alcanzado de forma satisfactoria, aunque algunas ideas iniciales evolucionaron y cambiaron durante el desarrollo para adaptarse mejor a las necesidades reales de la aplicación.

El objetivo principal de desarrollar una aplicación interactiva relacionada con el cuidado durante el embarazo se ha cumplido mediante la creación de un modo juego centrado en el bienestar de un personaje virtual, combinado con sistemas educativos, seguimiento y comunidad.

Respecto a los objetivos específicos:

- El personaje principal interactivo fue implementado correctamente, incorporando diferentes estados emocionales, evolución visual durante el embarazo y sistemas de interacción dinámica mediante acciones táctiles y minijuegos.
- La simulación de la rutina diaria también se alcanzó mediante la integración de distintas estancias y actividades relacionadas con el cuidado del personaje, como alimentación, descanso, higiene, ejercicio y ocio.
- El componente educativo del proyecto terminó teniendo incluso más importancia de la prevista inicialmente gracias al desarrollo del sistema de seguimiento basado en anuncios y recomendaciones organizadas según las etapas del embarazo.

- Los indicadores relacionados con el bienestar físico y emocional del personaje fueron implementados mediante el sistema de medidores y estados dinámicos, permitiendo que las decisiones del usuario afecten directamente tanto al comportamiento del personaje como al resultado final de la partida.
- El objetivo de fomentar una experiencia práctica e interactiva también se considera cumplido, especialmente tras la incorporación de animaciones y minijuegos táctiles que transformaron el modo juego en una experiencia mucho más dinámica.
- La integración de diferentes secciones dentro de la aplicación también se consiguió mediante la incorporación del modo juego, la comunidad, el sistema de seguimiento y las funcionalidades de configuración y ayuda y autenticación de usuarios.
- Finalmente, el seguimiento del embarazo tanto en tiempo simulado como en tiempo real fue uno de los aspectos que más evolucionó durante el proyecto.

En conjunto, considero que los objetivos planteados no solo se han alcanzado, sino que en algunos aspectos el proyecto terminó evolucionando hacia un resultado más completo y ambicioso de lo previsto inicialmente.

6.3 Valoración de la metodología y planificación

En general, considero que la planificación inicial del proyecto se mantuvo bastante cercana al resultado final. La idea principal de desarrollar un modo juego centrado en el cuidado de un personaje embarazado se mantuvo desde el inicio hasta el final del desarrollo, aunque algunas funcionalidades fueron evolucionando y reorganizándose a medida que el proyecto avanzaba.

Uno de los principales cambios realizados fue la separación entre el modo juego y el sistema de seguimiento del embarazo real. Inicialmente ambas funcionalidades iban a convivir dentro de una misma estructura, pero tras analizar el funcionamiento de la aplicación y valorar distintas opciones junto al tutor del proyecto, se decidió separar ambas temporalidades para evitar un sistema demasiado confuso y complejo. Considero que esta decisión ayudó mucho a organizar mejor las funcionalidades y simplificar el desarrollo general de la aplicación.

Uno de los aspectos que más tiempo requirió durante el desarrollo fue adaptarme a Jetpack Compose y a la nueva estructura de trabajo.

Además, también fue necesario adaptar algunas decisiones técnicas a la complejidad real del proyecto y al tiempo disponible. Inicialmente se valoró utilizar PostgreSQL como sistema principal de base de datos, pero esta opción requería desarrollar o adaptar una API intermedia para conectar la aplicación Android con el servidor. Debido a la complejidad añadida que suponía esta arquitectura y a las necesidades reales de la aplicación, finalmente se optó por utilizar Firebase y almacenamiento local, permitiendo simplificar gran parte del desarrollo y centrarse en las funcionalidades principales del proyecto.

En cuanto a la planificación general, considero que el proceso seguido fue adecuado para un proyecto de esta envergadura. El desarrollo comenzó definiendo primero la idea principal de la aplicación, realizando esquemas y planteamientos iniciales, seleccionando posteriormente las tecnologías y sistemas necesarios y, finalmente, desarrollando progresivamente las funcionalidades. Aunque probablemente intentaría organizar mejor algunos tiempos en futuras ocasiones, en general volvería a trabajar de una forma muy similar a la utilizada en este proyecto.

6.4 Visión de futuro

Como línea de futuro, una de las primeras mejoras que se plantea es la incorporación de traducciones dentro de la aplicación. Esto permitiría ampliar el alcance de Damagotchi y hacerla más accesible para usuarios que utilicen otros idiomas, enriqueciendo además la experiencia general del proyecto.

Otra posible evolución sería ampliar el modo juego una vez finalizado el embarazo virtual. Actualmente, al llegar a la semana 40, el ciclo principal termina con la evaluación final. En una futura versión, se podría desarrollar una segunda parte enlazada con la primera, centrada en el cuidado del bebé tras el nacimiento. De esta forma, la experiencia no finalizaría con el embarazo, sino que continuaría con una nueva etapa de juego.

También se plantea mejorar el sistema de notificaciones. Además de las notificaciones actuales de la comunidad, podrían añadirse avisos relacionados con el modo juego, por ejemplo cuando los medidores del personaje alcancen valores bajos. Asimismo, se podrían incorporar eventos especiales al superar cada trimestre, mostrando mensajes de felicitación e imágenes tipo ecografía, reforzando así la sensación de avance y acompañamiento.

Estas mejoras permitirían ampliar la aplicación, hacerla más completa y mantener la motivación del usuario durante más tiempo.

7. Glosario

Término	Definición
API	Interfaz que permite la comunicación entre diferentes sistemas o aplicaciones.
Cloud Functions	Funciones ejecutadas en la nube mediante Firebase para automatizar tareas del servidor.
DataStore	Sistema de almacenamiento local utilizado en Android para guardar datos persistentes.
FCM	Firebase Cloud Messaging. Servicio utilizado para enviar notificaciones push a dispositivos Android.
Firebase	Plataforma de desarrollo de Google que ofrece servicios como autenticación, bases de datos y notificaciones.
Firestore	Base de datos en la nube de Firebase utilizada para almacenar información dinámica de la aplicación.
Jetpack Compose	Framework moderno de Android utilizado para desarrollar interfaces de usuario mediante Kotlin.
Kotlin	Lenguaje de programación utilizado para el desarrollo de la aplicación Android.
MVVM	Patrón de arquitectura que separa la interfaz, la lógica y los datos en diferentes capas.
Navigation Compose	Librería de Jetpack Compose utilizada para gestionar la navegación entre pantallas.
Notificación push	Mensaje que el sistema operativo muestra al usuario aunque la aplicación no esté abierta.
Overlay	Capa visual superpuesta sobre otra pantalla o componente de la interfaz.
Persistencia	Capacidad de almacenar datos para que permanezcan disponibles aunque la aplicación se cierre.
StateFlow	Sistema utilizado para gestionar y observar cambios de estado de forma reactiva en Kotlin.
UI	User Interface. Conjunto de elementos visuales con los que interactúa el usuario.
ViewModel	Componente encargado de gestionar el estado y la lógica entre la interfaz y los datos.

8. Bibliografía

Android Developers. (s.f.). *Jetpack Compose documentation*. Android Developers. https://developer.android.com/jetpack/compose

Android Developers. (s.f.). *Navigation Compose*. Android Developers. https://developer.android.com/develop/ui/compose/navigation

Android Developers. (s.f.). *DataStore*. Android Developers. https://developer.android.com/topic/libraries/architecture/datastore

Firebase. (s.f.). *Firebase documentation*. Google. https://firebase.google.com/docs

Firebase. (s.f.). *Cloud Firestore documentation*. Google. https://firebase.google.com/docs/firestore

Firebase. (s.f.). *Firebase Authentication documentation*. Google. https://firebase.google.com/docs/auth

Firebase. (s.f.). *Firebase Cloud Messaging documentation*. Google. https://firebase.google.com/docs/cloud-messaging

JetBrains. (s.f.). *Kotlin documentation*. Kotlin. https://kotlinlang.org/docs/home.html

Google. (s.f.). *Material Design 3*. Material Design. https://m3.material.io/

Android Developers. (s.f.). *ViewModel overview*. Android Developers. https://developer.android.com/topic/libraries/architecture/viewmodel

Android Developers. (s.f.). *StateFlow and SharedFlow*. Android Developers. https://developer.android.com/kotlin/flow/stateflow-and-sharedflow

Android Developers. (s.f.). Compose Samples. GitHub. https://github.com/android/compose-samples

Venkatesh153. (s.f.). TamaPet. GitHub. https://github.com/venkatesh153/TamaPet

“Aprender es construir poco a poco aquello que al principio parecía imposible.”

Nota sobre el uso de inteligencia artificial en esta guía

Durante el desarrollo de esta memoria se han utilizado herramientas de inteligencia artificial como apoyo puntual en tareas de redacción, reformulación de textos, organización de ideas y mejora de la expresión escrita.

El uso de estas herramientas se ha realizado siempre como complemento al trabajo personal desarrollado durante el proyecto. Tanto el planteamiento de la aplicación, las decisiones técnicas, la implementación, las funcionalidades descritas y las reflexiones incluidas en el documento han sido desarrolladas y decididas por la autora del proyecto.

La inteligencia artificial se ha empleado principalmente como herramienta de asistencia para ayudar a estructurar determinados apartados, mejorar la claridad de algunos textos o reformular explicaciones cuando ha sido necesario. En todos los casos, el contenido generado ha sido revisado, adaptado y validado manualmente para asegurar su coherencia con el proyecto desarrollado y con la intención real del documento.

El objetivo de esta nota es mantener un criterio de transparencia sobre el uso de herramientas digitales dentro del proceso de elaboración de la memoria, entendiendo la inteligencia artificial como un recurso de apoyo y no como un sustituto del trabajo técnico y personal realizado.