

Plant Guardian: Aplicación móvil para el cuidado inteligente de plantas de exterior e interior

Fiorella Lucia Queirolo Chavez

14 de junio de 2026

Resumen– Plant Guardian es una aplicación móvil para Android diseñada para facilitar el cuidado inteligente de plantas de interior y exterior. Funciona como un asistente botánico personal dirigido a usuarios que carecen de conocimientos sobre el cuidado vegetal. Mediante la integración de inteligencia artificial y visión por computador, la aplicación identifica especies a partir de una fotografía, genera diagnósticos sobre el estado de salud de la planta y crea rutinas de cuidado personalizadas adaptadas a las condiciones meteorológicas locales en tiempo real. Finalmente, para motivar la constancia del usuario, la plataforma integra un sistema de gamificación que recompensa las tareas completadas con puntos de experiencia, niveles y logros.

Palabras clave– Aplicación móvil, Inteligencia artificial, Cuidado de plantas, Gamificación, Android, Reconocimiento visual.

Abstract– Plant Guardian is an Android mobile application designed to support the intelligent care of both indoor and outdoor plants. It acts as a personal botanical assistant for users with little or no prior knowledge of plant care. Using artificial intelligence and computer vision, the application identifies plant species from a photograph, generates health diagnostics and creates personalized care routines adapted to real-time local weather conditions. Finally, to encourage consistency in daily plant maintenance, the platform incorporates a gamification system that rewards completed tasks with experience points, levels, and achievements.

Keywords– Mobile application, Artificial intelligence, Plant care, Gamification, Android, Visual recognition.

1 INTRODUCCIÓN

EL cuidado de plantas en el hogar es una actividad cada vez más extendida, pero que a menudo se ve dificultada por la falta de conocimientos botánicos. La mayoría de usuarios desconoce aspectos básicos como la frecuencia de riego, la exposición adecuada a la luz o la detección temprana de enfermedades, lo que afecta negativamente a la salud y al desarrollo de sus plantas.

En este contexto surge Plant Guardian, una aplicación móvil para Android que actúa como un asistente botánico

personal. Mediante reconocimiento visual y análisis de síntomas, permite a los usuarios gestionar la salud de su colección de plantas, generando rutinas de cuidado personalizadas adaptadas al clima local. Además, la plataforma integra un sistema de gamificación que premia la constancia en las rutinas de cuidado mediante puntos de experiencia, niveles y logros.

El presente documento describe el proceso completo de desarrollo de Plant Guardian, partiendo de la definición de objetivos y del análisis del estado del arte, incluyendo un estudio comparativo de aplicaciones similares. Seguidamente, se aborda la planificación del proyecto, junto con la especificación de requisitos funcionales y no funcionales.

El desarrollo técnico se centra en la implementación del sistema, incluyendo la arquitectura, la base de datos y la interfaz de usuario. Finalmente, se presentan los resultados de la evaluación, las conclusiones y las mejoras futuras.

- E-mail de contacto: fiorellaqc17@gmail.com
- Menció: Enginyeria del Software
- Trabajo autorizado por: Sergio Bachiller Rubia (Ciències de la Computació i Intel·ligència Artificial)
- Curso 2025/26

2 OBJETIVOS

La meta principal de este proyecto es el desarrollo de un Producto Mínimo Viable (MVP)[1] que consolide las funcionalidades esenciales de Plant Guardian, divididas en tres objetivos principales:

- **Desarrollar un sistema inteligente de asistencia para el cuidado de plantas:** Implementar el reconocimiento automático de plantas, el diagnóstico preventivo de patologías vegetales y la creación de una rutina personalizada para cada planta mediante la integración de APIs externas.
- **Diseñar una interfaz gamificada centrada en la experiencia de usuario:** Crear una experiencia interactiva basada en recompensas e insignias que fomente el compromiso a largo plazo y promueva hábitos responsables en el cuidado de las plantas.
- **Implementar un repositorio de datos centralizado para el seguimiento y la personalización de cada planta:** Desarrollar un *backend* estructurado con una base de datos que permita almacenar las rutinas personalizadas y los datos de las plantas de cada usuario.

3 ESTADO DEL ARTE

El cuidado doméstico de plantas cuenta actualmente con diversas aplicaciones móviles basadas en inteligencia artificial disponibles en plataformas de descarga como Play Store.

Esta sección analiza las soluciones existentes, las tecnologías subyacentes y las carencias del mercado actual, con el objetivo de justificar el valor diferencial de Plant Guardian.

3.1. Aplicaciones comerciales existentes

Se han seleccionado tres aplicaciones representativas del sector: Plant Parent [2], Plant Master [3] y Planti [4].

Para la selección de estas aplicaciones, se aplicaron dos criterios de relevancia utilizados habitualmente en análisis comparativos de mercado [5]: mínimo **10.000 descargas** y **valoración superior a 4 estrellas** en la Play Store.

TABLA 1: COMPARATIVA DE FUNCIONALIDADES

Funcionalidad	P. Parent	AI P. Master	Planti
Reconocimiento visual	✓	✓	✓
Diagnóstico de síntomas	✓	✓	×
Rutinas personalizadas	✓	✓	✓
Datos meteorológicos	×	×	×
Historial de cuidados	✓	×	×
Gamificación	×	×	×

Del análisis se extraen dos conclusiones principales. En primer lugar, el reconocimiento visual y el diagnóstico de síntomas se han convertido en funcionalidades estándar del sector.

En segundo lugar, se observa que ninguna de las aplicaciones analizadas integra datos meteorológicos locales durante el análisis ni mecanismos de gamificación.

Además, durante las pruebas realizadas, se comprobó que las funcionalidades más avanzadas, como las rutinas personalizadas o el análisis de enfermedades, suelen estar bloqueadas tras modelos *freemium* con muros de pago.

3.2. Reconocimiento vegetal e IA generativa

El reconocimiento visual de estas especies se apoya en motores especializados como **Pl@ntNet** [6] o **Trefle** [7] que procesan imágenes mediante redes neuronales entrenadas sobre grandes bases de datos botánicas.

Además, el uso de modelos de lenguaje multimodal como **Gemini** [8] o **OpenAI** [9] abre la posibilidad de combinar en un único proceso la identificación de especies, el contexto ambiental y la generación de recomendaciones.

Esta integración de inteligencias artificiales que realicen el procedimiento de análisis de manera compacta está poco presente en las aplicaciones comerciales analizadas, ya que los pasos entre identificación y análisis quedan separados.

Por esto, una de las principales aportaciones que se busca con Plant Guardian es ofrecer recomendaciones personalizadas adaptadas al contexto del usuario a partir de la identificación de la planta, de manera compacta.

3.3. Gamificación para fomentar el cuidado

La gamificación, entendida como la aplicación de mecánicas de juego (puntos, niveles, logros) en contextos no lúdicos, ha demostrado ser eficaz para fomentar comportamientos regulares y el compromiso a largo plazo del usuario [10]. En el ámbito de las aplicaciones de salud y hábitos, su integración incrementa significativamente la retención y la frecuencia de uso.

Como muestra la Tabla 1, ninguna de las aplicaciones analizadas incorpora estos mecanismos. Por esto, se busca que Plant Guardian cubra esta falta de funcionalidades mediante un sistema de XP, niveles y recompensas desbloqueables que incentiva la constancia en las rutinas de cuidado.

3.4. Elección de la plataforma

El uso de la plataforma Android para el desarrollo se basa principalmente en su amplia presencia en el mercado español. Según datos de StatCounter Global Stats [11], entre 2025 y 2026 **Android cuenta con una cuota de mercado de entre el 75 % y el 80 %**, mientras que iOS se sitúa entre el 20 % y el 25 %.

Esto permite que la aplicación pueda llegar a un mayor número de usuarios y facilita la validación de la propuesta de valor del proyecto. Además, centrar el desarrollo en Android simplifica el proceso de realización de entrevistas y pruebas con usuarios, ya que reduce las posibles limitaciones de acceso derivadas del dispositivo utilizado.

Por otro lado, desde el punto de vista técnico, Android también resulta una opción adecuada por su integración con los servicios de Google y por la facilidad que ofrece Android Studio como entorno de desarrollo. Por último, al haber utilizado previamente esta herramienta en otros proyectos, la curva de aprendizaje se reduce y permite centrar más esfuerzo en otras partes del desarrollo.

4 METODOLOGÍA Y PLANIFICACIÓN

Esta sección describe el marco de trabajo utilizado para organizar y gestionar el desarrollo del proyecto, así como la planificación temporal establecida desde el inicio.

4.1. Metodología SCRUM

Para este proyecto se ha optado por la metodología Scrum [12], combinando los roles de *product owner*, *scrum master* y desarrollador al tratarse de un proyecto individual.

Las tareas se organizan por *sprints* siguiendo los hitos de entrega del proyecto, manteniendo la flexibilidad de integrar tareas pendientes o adelantar requerimientos según las necesidades del desarrollo. El seguimiento se realiza mediante Trello[13], configurado con el flujo de trabajo de Scrum.

4.2. Fases del proyecto

El desarrollo se estructura en cuatro fases (Tabla 2) consecutivas que cubren todo el ciclo de vida del proyecto, priorizando la implementación de los tres objetivos principales.

TABLA 2: FASES DEL PROYECTO

Fase	Descripción	Inicio	Fin
Fase 1	Definición, análisis y diseño	12/02	08/03
Fase 2	Desarrollo de la BD, APIs y lógica	09/03	12/04
Fase 3	Desarrollo del frontend e integración	13/04	24/05
Fase 4	Pruebas y finalización de la app	25/05	18/07

Para organizar el proyecto, se ha creado un diagrama WBS (Apéndice A.1), para definir las subtarefas, y un diagrama de Gantt (Apéndice A.2), que marca el seguimiento de las tareas.

4.3. Estado actual y seguimiento

Actualmente, el proyecto se encuentra en su fase final y se ha consolidado como una **aplicación funcional** (MVP) que cumple los objetivos definidos inicialmente.

El desarrollo ha seguido la planificación establecida, manteniendo las tareas al día durante cada iteración. Además, el ritmo de trabajo ha permitido adelantar algunos hitos y dedicar tiempo a la implementación de funcionalidades y mejoras no contempladas en un principio.

Entre ellas se incluyen efectos visuales destinados a mejorar la experiencia de usuario y hacer la aplicación más atractiva e intuitiva.

A pesar del grado de madurez alcanzado por el sistema, se han identificado diversas oportunidades de ampliación y mejora vistas por usuarios externos al desarrollo. Estas se detallarán en secciones posteriores.

5 REQUISITOS

En esta sección se describen los requisitos funcionales y no funcionales del sistema, que definen el comportamiento y la calidad de la aplicación.

5.1. Toma de requisitos

Para alinear el sistema con las necesidades reales, la toma de requisitos partió de un **análisis cualitativo** mediante entrevistas a usuarios.

Se seleccionó una muestra representativa de tres personas que encajaban en el perfil del usuario final: propietarios de plantas en el hogar, usuarios de Android y sin conocimientos avanzados de botánica.

Las entrevistas permitieron obtener conclusiones que definieron directamente el alcance del MVP:

- **Principales problemas con las plantas:** Los usuarios coinciden en la dificultad para determinar la frecuencia de riego adecuada, identificar plagas de forma temprana y comprender las necesidades climáticas de cada especie.
- **Funcionalidades más demandadas:** Ninguno utilizaba aplicaciones especializadas, pero todos priorizaron la necesidad de identificar plantas mediante fotografías, recibir recordatorios de cuidados y obtener información botánica simplificada.
- **Expectativas de Diseño (UX):** Expresaron una clara preferencia por interfaces intuitivas, evitando la sobrecarga de información y priorizando resúmenes visuales rápidos del estado general de su colección.
- **Sin gamificación:** Dos de los usuarios indicaron que no consideraban necesario un sistema de progresión, puesto que su interés principal se centraba exclusivamente en el cuidado de las plantas.

A partir de estos resultados, se estableció que Plant Guardian debía priorizar la facilidad de uso, ocultando la complejidad de las decisiones botánicas mediante la automatización inteligente.

Además, aunque algunos usuarios no mostraron interés por la gamificación, se decidió mantenerla debido a que es una funcionalidad clave para diferenciar Plant Guardian de otras aplicaciones similares. Por eso, se diseñó de forma que complementara la experiencia de uso sin convertirse en el elemento principal de la aplicación.

5.2. Requisitos funcionales

Basados en las conclusiones anteriores, los requisitos funcionales (detallados en el Apéndice A.3) se agrupan en los siguientes módulos:

- **Gestión de usuarios:** Controla el acceso de las cuentas de cada usuario para que puedan guardar su colección de plantas.
- **Gestión de plantas:** Es el núcleo que permite identificar especies por foto, diagnosticar plagas y recibir guías de cuidados específicos.
- **Gestión meteorológica:** Obtiene datos climáticos en tiempo real según la ubicación del usuario.
- **Gestión de recompensas:** Implementa el sistema de puntos, niveles y logros para incentivar el cuidado diario.

5.3. Requisitos no funcionales

Para dar respuesta a las expectativas de experiencia de usuario identificadas en las entrevistas, se establecieron los siguientes requisitos no funcionales, orientados principalmente a la calidad percibida del sistema (detallados en el Apéndice A.4):

- **Usabilidad:** Asegura que la interfaz sea intuitiva y accesible para perfiles desde principiantes hasta expertos.
- **Rendimiento:** Define los tiempos de respuesta óptimos para el reconocimiento de imágenes y la agilidad de la navegación.
- **Disponibilidad:** Garantiza que el usuario pueda acceder a sus recordatorios de cuidado en cualquier momento.

6 DESARROLLO DEL SISTEMA

En esta sección se presenta la arquitectura general de Plant Guardian y los principales componentes del sistema, las APIs externas, el sistema de gamificación y las herramientas utilizadas durante el desarrollo.

6.1. Arquitectura del sistema

La arquitectura de Plant Guardian sigue un **modelo cliente-servidor** organizado en tres capas: *frontend*, *backend* y base de datos, complementadas con la integración de servicios externos mediante APIs.

En la siguiente imagen se puede ver el diagrama de la arquitectura de manera simplificada.

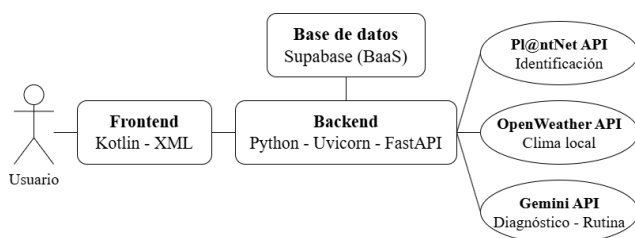


Fig. 1: Arquitectura del sistema.

6.1.1. Frontend

El *frontend* se ha desarrollado en **Android Studio** utilizando Kotlin para la lógica del flujo de la aplicación y XML para los elementos visuales. La comunicación con el *backend* se realiza a través de Retrofit[14], que actúa como puente HTTP entre ambas capas.

6.1.2. Backend

El *backend*, desarrollado en Python con **FastAPI** y **Uvicorn**, constituye el núcleo lógico del sistema.

Se estructura en tres subcapas: los *routers*, que reciben y dirigen las peticiones HTTP, la capa *logic*, que concentra la lógica de negocio, y los *services*, responsables de la integración con las APIs externas.

6.1.3. Base de datos

La base de datos está gestionada mediante **Supabase**[15], una plataforma de Backend as a Service (BaaS) basada en PostgreSQL que ofrece autenticación, almacenamiento y acceso a datos en tiempo real a través de una API REST, encargándose de la persistencia de los datos del sistema.

Su estructura se organiza en tres bloques: gestión del usuario y su progresión, gestión de plantas y tareas, y el sistema de recompensas. Esto último se explica más en la Sección 6.3.

6.2. APIs externas

El sistema integra tres APIs externas que forman el núcleo inteligente de la aplicación y trabajan conjuntamente para generar rutinas personalizadas.

A continuación, se describe cada una de ellas junto con la justificación de su uso frente a otras alternativas, así como su integración en la aplicación final.

6.2.1. Identificación de plantas con PI@ntNet

PI@ntNet permite la identificación automática de especies vegetales a partir de fotografías, apoyándose en modelos de visión por computador entrenados con grandes bases de datos botánicas.

A diferencia de motores de visión artificial generalistas (como Google Cloud Vision), PI@ntNet está entrenado exclusivamente sobre una de las mayores bases de datos botánicas del mundo, lo que garantiza un índice de precisión significativamente mayor en el reconocimiento.

Además de esta ventaja técnica, su plan gratuito (*free tier*) ofrece una cuota de peticiones muy superior a la de otras alternativas comerciales como Trefle, lo que permite soportar la carga de uso durante el desarrollo sin incurrir en costes elevados.

En la Figura 2 se muestra el proceso de identificación de una planta mediante la aplicación.

6.2.2. Datos meteorológicos con OpenWeather

La API de **OpenWeather** provee datos meteorológicos locales en tiempo real, extrayendo parámetros clave para el diagnóstico botánico como la temperatura, la humedad, las precipitaciones y la velocidad del viento [16].

Su selección se fundamenta en su posición como estándar de la industria y en la solidez de su integración RESTful, la cual cuenta con una documentación exhaustiva. Además, desde una perspectiva operativa, su plan gratuito permite realizar un alto volumen de llamadas diarias.

Esto asegura una actualización constante de los datos ambientales, permitiendo mantener un alto nivel de precisión en las recomendaciones y rutinas generadas. Esta fiabilidad técnica impacta directamente en la experiencia final, ya que los usuarios valoran de forma positiva la exactitud de la información mostrada.

En la Figura 3 se muestran los datos meteorológicos obtenidos en tiempo real a través de la API.

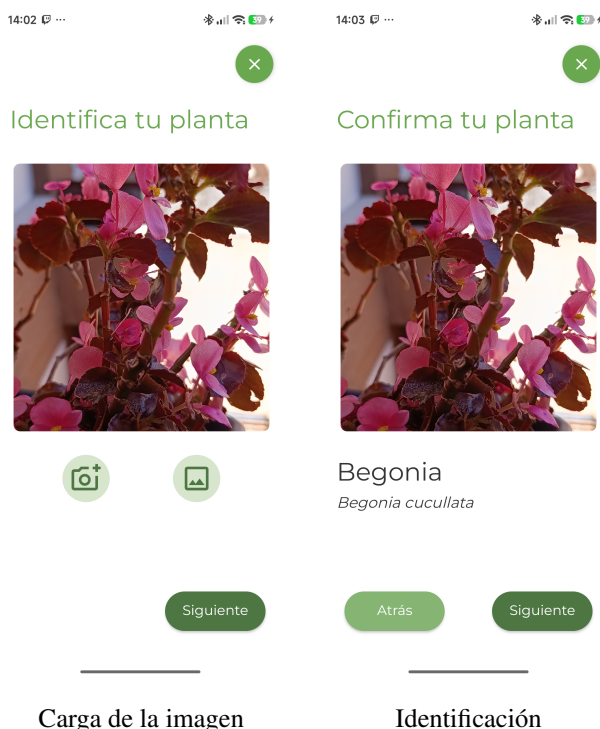


Fig. 2: Identificación de Pl@ntNet para la planta Begonia.



Fig. 4: Rutina generada por Gemini para la planta Begonia.



Fig. 3: Datos meteorológicos locales en tiempo real.

6.2.3. Generación de rutinas con Gemini

Como inteligencia artificial principal se emplea **Gemini**, el modelo multimodal de Google. Este modelo actúa como el “cerebro” del sistema, cruzando los datos botánicos obtenidos con el contexto climático para generar diagnósticos preventivos y rutinas de cuidado.

Su elección frente a otros modelos como los de OpenAI se debe principalmente a su integración dentro del ecosistema de Google, lo que facilita su uso en el entorno de desarrollo de Android Studio y su compatibilidad con el resto de servicios empleados en el proyecto.

Además, otro factor determinante fue el coste, ya que Gemini ofrece un acceso gratuito suficientemente amplio en sus distintas versiones para la fase de pruebas y validación del sistema.

En la Figura 4 se muestra un ejemplo de la rutina generada por Gemini para una planta concreta.

6.3. Diseño de la base de datos

La base de datos gestiona la información del usuario, sus plantas y el sistema de recompensas, organizándose en tres bloques principales que se explicarán en los siguientes apartados. Este modelo se puede consultar en el Apéndice A.5.

6.3.1. Tablas definidas

El primer bloque corresponde a la **gestión de usuario**, centralizada en la tabla *Usuario*, donde se almacenan los datos personales junto con la progresión del jugador, incluyendo nivel, experiencia (XP), tareas completadas y el avatar desbloqueable.

El segundo bloque se centra en la **gestión de plantas y tareas** y se estructura en tres tablas principales. *Planta* actúa como catálogo de especies y almacena la información de cada planta, lo que permite reutilizar datos y evitar llamadas innecesarias a las APIs. *Usuario_Planta* representa las plantas asociadas a cada usuario e incluye información como la foto, el estado de salud y los consejos generados por la IA. Por último, *Tarea* registra las acciones de cuidado junto con la experiencia (XP) obtenida y la fecha del recordatorio.

El tercer bloque corresponde al sistema de **recompensas**, que utiliza la tabla *Logro* para definir los hitos disponibles y la tabla *Usuario_Logro* para registrar los logros obtenidos por cada usuario junto con su fecha de desbloqueo.

6.3.2. Operaciones CRUD

Además del diseño del modelo de datos, fue necesario configurar adecuadamente los permisos de acceso a la base de datos para garantizar la correcta gestión de la información.

Para ello, se desarrolló un script SQL encargado de definir las **políticas de acceso** (*Row Level Security Policies*) de Supabase para cada una de las tablas del sistema.

Esta configuración permitió habilitar únicamente las operaciones CRUD[17] necesarias en cada tabla. De este modo, no todas las tablas disponen de los mismos permisos, sino que estos se adaptan a su función dentro de la aplicación.

Por ejemplo, la tabla *Logro* únicamente almacena el ca-

tálogo global de logros disponibles en la aplicación. Dado que estos registros no son creados, modificados ni eliminados por los usuarios, únicamente fue necesario conceder permisos de lectura.

En cambio, tablas como *Tarea* requieren todos los permisos, ya que sus registros se crean automáticamente al añadir nuevas plantas, se consultan para mostrar las tareas pendientes, se actualizan conforme cambia su estado o fecha programada y pueden eliminarse cuando el usuario elimina una planta asociada.

6.4. Diseño de la interfaz

El diseño de Plant Guardian se desarrolló mediante varias iteraciones para poder crear una interfaz lo más sencilla e intuitiva posible, centrada en la funcionalidad principal del sistema: el cuidado de las plantas.

Al mismo tiempo, se buscó integrar los elementos de gamificación de manera equilibrada, de forma que complementaran la experiencia de usuario sin desviar la atención de la actividad principal.

6.4.1. Desarrollo del diseño

En general, se ha seguido una paleta de colores relacionada con el entorno natural, evitando contrastes excesivos, y se han aplicado componentes de Material Design de Google [18] para mantener una estética coherente y familiar para el usuario Android.

El proceso de diseño se estructuró en dos etapas. Primero se realizaron *sketches* iniciales, en donde se esbozaron las pantallas principales y se anotaron elementos específicos de la interfaz.

A partir de estos bocetos, la segunda etapa se basó en realizar un prototipo *high-fidelity* en Figma, obteniendo un diseño muy cercano al aspecto final de la aplicación.

6.4.2. Pantallas principales

En cuanto a la interfaz final, esta se organiza en cinco bloques principales. Las pantallas principales pueden consultarse en el Apéndice A.6.

- **Autenticación:** Incluye las pantallas de registro e inicio de sesión.
- **Inicio:** Muestra el clima local, accesos rápidos a las otras pantallas y un resumen del nivel del usuario.
- **Mis plantas:** Permite visualizar y buscar la colección mediante un buscador.
- **Añadir planta:** Se realiza la identificación de la planta, el análisis y la generación de una rutina personalizada, que posteriormente se guarda para su consulta. El flujo completo de este proceso puede consultarse en el Apéndice A.7.
- **Perfil:** Integra los elementos de gamificación con una barra de experiencia, logros desbloqueados e iconos de perfil que el usuario obtiene al subir de nivel.

6.4.3. Cambios realizados

Cabe destacar que la pantalla principal experimentó varias iteraciones respecto al diseño original elaborado en el *sketch* y en Figma.

A diferencia de otras pantallas, se consideró que la propuesta inicial no proporcionaba suficiente información ni elementos que facilitaran la interacción del usuario con la aplicación.

Por ello, se tomó como referencia la aplicación Planti, previamente analizada durante el estudio comparativo realizado en fases anteriores del proyecto (Sección 3.1). A partir de este análisis, se observó que una de las principales carencias es la falta de elementos de acceso rápido (*shortcuts*).

Con esta referencia, se adaptaron diversas ideas a una propuesta más minimalista y coherente con la identidad visual de Plant Guardian. Este proceso de rediseño se llevó a cabo con el apoyo de la herramienta de inteligencia artificial Claude[19], que permitió visualizar distintas alternativas de diseño incorporando los *shortcuts* mencionados.

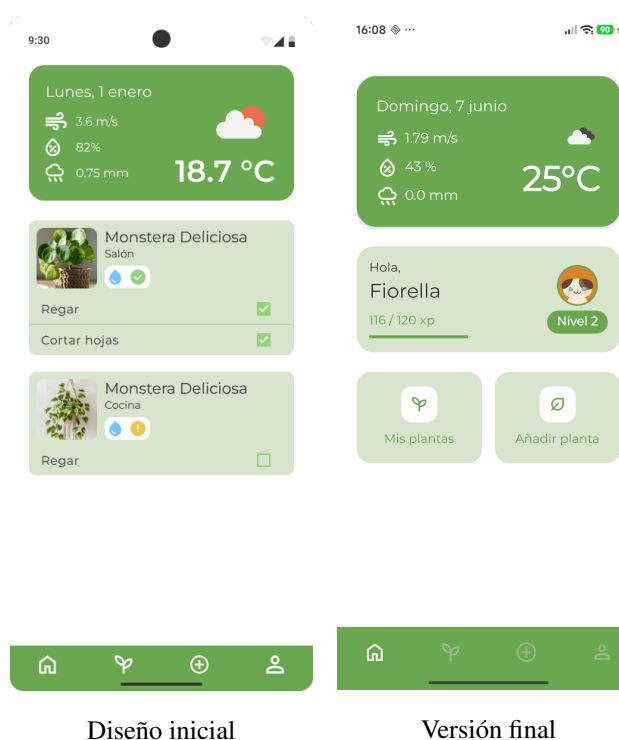


Fig. 5: Evolución del diseño de la pantalla de inicio.

6.5. Sistema de gamificación

El sistema de gamificación tiene como objetivo incentivar al usuario a cuidar sus plantas de forma constante y mantener buenos hábitos de seguimiento.

A continuación, se presentan los cuatro elementos principales que lo componen, acompañados de capturas hechas desde la aplicación donde puede verse cómo se han implementado.

6.5.1. Sistema de progresión

El sistema de progresión integra una mecánica para subir de nivel según la experiencia obtenida, sustentada en un

algoritmo simple que contiene una curva de dificultad dinámica basada en un **multiplicador de 1.2x**.

En la siguiente imagen se muestra el estado del nivel desde la pantalla de usuario.

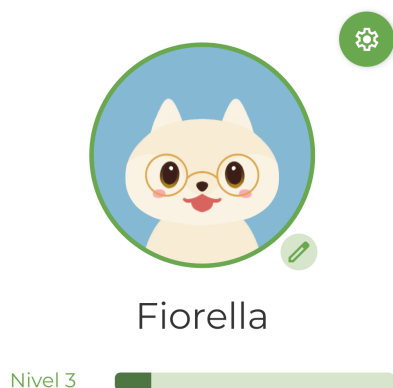


Fig. 6: Nivel de progresión del usuario.

6.5.2. Logros

Por otro lado, el usuario puede ganar logros en función de las actividades que realiza dentro de la aplicación. Estos logros se muestran en el perfil mediante iconos representativos y, actualmente, se agrupan en tres categorías: nivel alcanzado, número de plantas registradas y cantidad de tareas completadas.

Para implementar este sistema de logros, se ha diseñado un mecanismo que permite diferenciar entre los que hay disponibles y los que ha obtenido cada usuario.

Para ello, se mantiene un registro acumulativo del número total de tareas completadas a lo largo de toda la trayectoria del usuario en la aplicación, independientemente de su estado actual.



Fig. 7: Logros de plantas, tareas y niveles.

6.5.3. Iconos

Por último, los iconos de perfil desbloqueables mencionados vinculan el nivel del usuario con la disponibilidad de avatares. Es decir, el usuario solo puede seleccionar aquellos iconos para los que ha alcanzado el nivel requerido,

añadiendo así un incentivo visual al progreso dentro de la aplicación.

En la siguiente imagen se muestran todos los iconos disponibles que el usuario puede desbloquear.



Fig. 8: Iconos disponibles en la aplicación.

6.6. Herramientas de desarrollo

Para el desarrollo de Plant Guardian se han utilizado las siguientes herramientas, organizadas según su función dentro del proyecto.

En cuanto al desarrollo y control de versiones, se utiliza **GitHub** para la gestión del código fuente y el historial de cambios, **Visual Studio Code** para la implementación del *backend* y **Android Studio** para el desarrollo del *frontend*.

Para el diseño y prototipado, se utilizó **Figma** para desarrollar el prototipo *high-fidelity* y definir el flujo de navegación de la aplicación antes de su implementación.

Por último, para la gestión documental y evaluación, se empleó **Google Drive** como repositorio centralizado de documentos del proyecto, incluyendo también el **cuestionario SUS** distribuido durante las pruebas de usabilidad con usuarios.

6.7. Despliegue

El *backend* se ha desplegado mediante **Render**[20], una plataforma de alojamiento gratuita que permite mantener el servidor activo y accesible.

Para evitar la suspensión automática del servicio en el plan gratuito, se ha configurado un *cron-job*[21] que realiza llamadas periódicas cada 10 minutos, garantizando así la disponibilidad continua del sistema durante el periodo de evaluación[22].

7 EVALUACIÓN Y RESULTADOS

Con el objetivo de evaluar el estado final de Plant Guardian y verificar el cumplimiento de los requisitos definidos, se han llevado a cabo dos fases de validación: **pruebas alfa** y **pruebas beta**. La principal diferencia entre ambas fases

radica en los participantes involucrados en el proceso de evaluación[23].

Durante la fase alfa, es el propio desarrollador quien analiza y valida el funcionamiento de la aplicación, comprobando que las funcionalidades implementadas cumplen los requisitos establecidos.

Por otro lado, en la fase beta son los usuarios finales quienes interactúan con la versión MVP del producto, proporcionando información relevante para detectar posibles errores, evaluar la experiencia de usuario e identificar oportunidades de mejora.

7.1. Pruebas alfa

En esta fase se analizaron tanto las funcionalidades implementadas como la calidad general del sistema, comprobando que el comportamiento de la aplicación se ajustaba a lo previsto.

7.1.1. Funcionalidades implementadas

Tras la revisión del sistema, la gran mayoría de las funcionalidades previstas quedaron implementadas satisfactoriamente.

Las tres que no llegaron a desarrollarse fueron el inicio de sesión mediante Google, las colecciones de plantas (sustituidas por un sistema de búsqueda equivalente) y las alertas climáticas ante condiciones meteorológicas adversas.

En todos los casos se trató de características consideradas opcionales desde las fases iniciales de planificación, por lo que su ausencia no compromete los objetivos principales del proyecto ni la experiencia de uso básica.

Desde un punto de vista técnico, su implementación en versiones futuras no presentaría dificultades relevantes y podría abordarse de forma incremental.

7.1.2. Calidad del sistema

En cuanto a la experiencia de uso, la interfaz se mostró intuitiva y visualmente clara, con una carga de información reducida y una navegación ágil hacia las funcionalidades principales.

El flujo de añadir una planta, desde la toma de fotografía hasta la generación de la rutina personalizada, funcionó de forma fluida y coherente con lo diseñado.

Respecto al rendimiento, el tiempo medio de respuesta durante la identificación de plantas se situó en torno a los cinco segundos, un valor aceptable aunque sujeto a variaciones en función de la carga de los servicios externos utilizados.

Asimismo, se optimizó el tamaño de la aplicación mediante la generación de una versión APK comprimida, reduciendo el impacto en el almacenamiento del dispositivo.

En lo relativo a la disponibilidad, durante las pruebas no se detectaron fallos en la carga de información cuando existía conexión a Internet. No obstante, el uso de planes gratuitos en los servicios externos impone ciertas limitaciones que podrían afectar a la disponibilidad en escenarios de alta demanda.

Por otro lado, el modo sin conexión no está disponible, dado que las funcionalidades principales de la aplicación dependen de servicios remotos para su funcionamiento, lo

que hace imprescindible disponer de conexión a Internet en todo momento.

7.2. Pruebas beta

Mediante estas pruebas se evalúa la experiencia de uso de la aplicación con usuarios reales que no han participado en su desarrollo, permitiendo identificar problemas de usabilidad y oportunidades de mejora desde la perspectiva del usuario final.

7.2.1. Metodología

Estas pruebas se realizaron de forma presencial con tres participantes, los mismos a quienes se les realizaron las entrevistas iniciales de requisitos.

Se trata de usuarios sin conocimientos técnicos, con experiencia básica en Android y con plantas en su hogar, ajustándose al perfil de usuario final previsto.

Cada participante realizó ocho tareas: iniciar sesión, añadir una planta, eliminar una planta, modificar su nombre, cambiar su icono de perfil, marcar una tarea como hecha y buscar otros nombres de una planta. Al finalizar, completaron un cuestionario SUS[24] y respondieron preguntas abiertas sobre su experiencia.

7.2.2. Cuestionario SUS

La aplicación obtuvo una puntuación media de **93,33 sobre 100** en el cuestionario SUS (Tabla 3), lo que significa que hay una percepción altamente positiva por parte de los usuarios.

No obstante, debe tenerse en cuenta que el **resultado puede estar sesgado** debido al tamaño reducido de la muestra, por lo que debe interpretarse como orientativo. Aun así, constituye un buen punto de partida que indica una **valoración inicial muy positiva** del sistema.

TABLA 3: RESULTADOS DEL CUESTIONARIO SUS

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Res.
P1	5	5	5	1	5	1	5	1	5	1	90
P2	5	5	5	1	5	1	5	1	5	1	90
P3	5	1	5	1	5	1	5	1	5	1	100
Media											93,33

7.2.3. Resultados y opiniones recogidas

En general, los tres usuarios se desarrollaron correctamente durante las tareas propuestas, lo que sugiere que perciben la aplicación como intuitiva y fácil de aprender.

A partir de la observación y las preguntas abiertas, los hallazgos principales se agrupan en:

- **Incidencias:** Se detectaron problemas en algunas pantallas, como una mala adaptación del teclado durante el registro, la no disponibilidad de la cámara por un conflicto de permisos, y cierta dificultad para localizar el botón de eliminar plantas o cambiar el icono del perfil.
- **Valoración positiva:** Los aspectos más apreciados fueron la claridad visual de las tareas de cuidado y la utilidad de la información botánica proporcionada (por ejemplo, la información sobre pautas de riego).

- **Puntos débiles en la gamificación:** Los elementos de gamificación (niveles e iconos) pasaron desapercibidos, ya que los usuarios centraron toda su atención en el cuidado botánico, lo cual era lo esperado según la decisión tomada después de las entrevistas.
- **Propuestas de mejora:** Las sugerencias de los usuarios incluyeron la opción de visibilizar la contraseña durante el registro, la integración de alarmas o notificaciones *push* como recordatorios, y la inclusión de un foro de opiniones entre la comunidad.

8 CONCLUSIONES

A continuación se presenta un resumen de las conclusiones obtenidas al realizar las pruebas mencionadas en los apartados anteriores junto con las mejoras que se podrían realizar en un futuro.

8.1. Cumplimiento de los objetivos

El análisis de la versión final tras las pruebas permite extraer las siguientes conclusiones sobre los objetivos alcanzados, organizadas por ámbitos:

- **Sistema inteligente de asistencia:** Apoyado en las APIs de Pl@ntNet y Gemini, se han implementado con éxito las funcionalidades núcleo de la aplicación, la identificación botánica, la generación de rutinas personalizadas basadas en el clima local y la presentación de diagnósticos accesibles.
- **Diseño de la interfaz gamificada:** El sistema de gamificación fue implementado de forma completa, integrando XP, niveles, logros e iconos desbloqueables. No obstante, **el sistema de recompensas pasó en gran medida desapercibido**, ya que se redujo intencionalmente su presencia para priorizar la funcionalidad principal de la aplicación. Para mejorar su impacto, sería necesario hacerlo más interactivo mediante elementos como notificaciones, animaciones al ganar experiencia o una mayor variedad de recompensas, evitando en todo caso saturar al usuario.
- **Implementación del repositorio centralizado:** La base de datos centralizada ha cumplido satisfactoriamente su función principal, la cual es almacenar las rutinas personalizadas, los datos de cada planta y la progresión del usuario de forma estructurada y persistente. Además, la arquitectura cliente-servidor adoptada ha demostrado ser adecuada para esta fase del proyecto. No obstante, se han detectado problemas de saturación en algunas llamadas a las APIs externas, lo que ha llevado a considerar soluciones como **OpenRouter**[25], que permitiría alternar entre múltiples modelos de IA en caso de que alguno no pueda responder, mejorando así la robustez del sistema.

8.2. Mejoras

A partir de los resultados obtenidos durante las pruebas y el grado de cumplimiento con las funcionalidades, las mejoras identificadas para la evolución de la aplicación son:

- **Correcciones críticas:** En primer lugar, se deben abordar las incidencias que afectan directamente a la experiencia de uso básica, como la corrección de los permisos de acceso a la cámara y la optimización de la pantalla de registro.
- **UX y gamificación:** Las pruebas indicaron que los usuarios confundían la configuración de datos personales con la personalización de su avatar (recompensa). Se plantea unificar estos menús e implementar *feedback* activo (retroalimentación en tiempo real) al desbloquear un logro, integrando la gamificación en el flujo principal en lugar de mantenerla aislada.
- **Funcionalidades de valor añadido:** En segundo lugar, se proponen mejoras no prioritarias que enriquecerían futuras versiones, destacando la implementación de notificaciones de recordatorio mediante alarmas y la opción de visibilidad de contraseña.
- **Expansión y comunidad:** Se contempla la viabilidad de exportar la plataforma a dispositivos iOS mediante tecnologías multiplataforma. Asimismo, se plantea incorporar funcionalidades sociales, como un foro de intercambio de experiencias, para fomentar una comunidad en torno al cuidado de plantas.
- **Arquitectura y refactorización:** Debido al tiempo limitado y a la curva de aprendizaje en la estructuración de proyectos nativos en Android Studio, algunas partes del código son susceptibles de mejora. Como trabajo futuro, se propone refactorizar los componentes mediante la adopción de patrones de diseño más robustos y una mayor separación de responsabilidades entre módulos (clases, efectos, etc.).

USO DE LA IA GENERATIVA

Durante el desarrollo de Plant Guardian se han utilizado principalmente **Gemini** y **Claude** como herramientas de apoyo, cada una con un rol diferenciado.

Gemini se empleó desde las fases iniciales para la toma de decisiones tecnológicas, la estructuración del *backend* y el diseño de la base de datos. Su uso facilitó la comprensión de las relaciones entre componentes y la organización del proyecto en Android Studio.

Claude se utilizó como apoyo en el diseño de interfaces, ayudando a comprender el funcionamiento de animaciones, transiciones y distribuciones visuales antes de trasladarlas al código definitivo.

En ambos casos, todo fragmento de código o sugerencia fue analizado y adaptado para garantizar su ajuste a los requisitos del proyecto. El uso de estas herramientas permitió agilizar el desarrollo individual y centrar el esfuerzo en la implementación funcional del MVP.

AGRADECIMIENTOS

Quiero expresar mi más sincero agradecimiento a mi pareja, León, por su ayuda y motivación constante durante estos meses de trabajo.

Asimismo, agradezco a mis compañeros Gabriel, Manel y Gerson ya que su presencia en esta etapa ha contribuido muy positivamente al resultado de este trabajo.

Finalmente, dedico un agradecimiento muy especial a mis padres. Ellos han sido un pilar fundamental y la verdadera inspiración que me llevó a crear Plant Guardian.

REFERENCIAS

- [1] Mailchimp. (2022, 22 de diciembre). *Significado de MVP - Qué es un Producto Mínimo Viable*. Recuperado el 13 de febrero de 2026, de <https://mailchimp.com/es/resources/minimum-viable-product/>
- [2] Plant Parent AI. (s.f.). *Plant Parent: AI Plant Care*. Google Play Store. Recuperado el 14 de febrero de 2026, de <https://play.google.com/store/apps/details?id=com.plantparentai.app>
- [3] Wyven. (s.f.). *AI Plant Master -Plant Care*. Google Play Store. Recuperado el 14 de febrero de 2026, de <https://play.google.com/store/apps/details?id=com.ai.plant.master.android&hl=es>
- [4] Adlunam. (s.f.). *Planti - Plant Identifier*. Google Play Store. Recuperado el 14 de febrero de 2026, de <https://play.google.com/store/apps/details?id=hu.adlunam.planti>
- [5] Zinklar. (2024, 8 de octubre). *Análisis Comparativo: Guía Práctica para Profesionales*. Recuperado el 14 de febrero de 2026, de <https://zinklar.com/es/blog/que-es-un-analisis-comparativo-una-guia-practica-para-profesionales/>
- [6] Pl@ntNet. (s.f.). *Introduction - Pl@ntNet API for developers*. Recuperado el 11 de marzo de 2026, de <https://my.plantnet.org/doc/getting-started/introduction>
- [7] Trefle. (s.f.). *Trefle API Documentation*. Recuperado el 11 de marzo de 2026, de <https://docs.trefle.io/docs/guides/getting-started>
- [8] Google. (s.f.). *Gemini API | Google AI for Developers*. Recuperado el 11 de marzo de 2026, de <https://ai.google.dev/gemini-api/docs>
- [9] OpenAI. (s.f.). *OpenAI API Documentation*. Recuperado el 11 de marzo de 2026, de <https://platform.openai.com/docs>
- [10] Interaction Design Foundation (IxDF). (2026). *What is Gamification?* Recuperado el 1 de junio de 2026, de <https://ixdf.org/literature/topics/gamification>
- [11] StatCounter Global Stats. (s.f.). *Operating System Market Share Worldwide*. Recuperado el 14 de febrero de 2026, de <https://gs.statcounter.com/os-market-share>
- [12] Atlassian. (s.f.). *¿Qué es Scrum? Una guía para el marco de trabajo ágil*. Recuperado el 13 de febrero de 2026, de <https://www.atlassian.com/es/agile/scrum>
- [13] Trello. (s.f.). *Trello*. Recuperado el 16 de febrero de 2026, de <https://trello.com/es>
- [14] Square. (s.f.). *Introduction | Retrofit*. Recuperado el 11 de abril de 2026, de <https://square.github.io/retrofit/>
- [15] Supabase. (s.f.). *Supabase Documentation: Backend as a Service Platform*. Recuperado el 20 de febrero de 2026, de <https://supabase.com/docs>
- [16] OpenWeatherMap. (s.f.). *OpenWeatherMap API guide*. Recuperado el 18 de marzo de 2026, de <https://openweathermap.org/guide>
- [17] Microsoft Learn. (s.f.). *CRUD (Crear, Leer, Actualizar, Eliminar)*. Recuperado el 22 de abril de 2026, de <https://learn.microsoft.com/es-es/iis-administration/api/crud>
- [18] Google. (s.f.). *Material Design 3*. Recuperado el 9 de mayo de 2026, de <https://m3.material.io>
- [19] Anthropic. (s.f.). *Claude AI*. Recuperado el 14 de mayo de 2026, de <https://www.anthropic.com/claude>
- [20] Render. (s.f.). *Docs + Quickstarts | Render*. Recuperado el 15 de mayo de 2026, de <https://render.com/docs>
- [21] Cron-job.org. (s.f.). *Free cronjobs - from minutely to once a year*. Recuperado el 15 de mayo de 2026, de <https://cron-job.org/en/>
- [22] FastCron. (s.f.). *Wake up Render free instances*. Recuperado el 15 de mayo de 2026, de <https://www.fastcron.com/tutorials/render-cron/>
- [23] ClickUp. (2024, 4 de diciembre). *Pruebas alfa frente a pruebas beta*. Recuperado el 1 de junio de 2026, de <https://clickup.com/es-ES/blog/243172/pruebas-alfa-frente-a-pruebas-beta>
- [24] Universitat Oberta de Catalunya. (s.f.). *Cuestionarios - Design Toolkit*. Recuperado el 1 de junio de 2026, de <https://design-toolkit.recursos.uoc.edu/es/guia/cuestionarios/>
- [25] OpenRouter. (2026). *The Unified Interface for LLMs*. Recuperado el 10 de junio de 2026, de <https://openrouter.ai>

APÉNDICE

A.1. Diagrama WBS

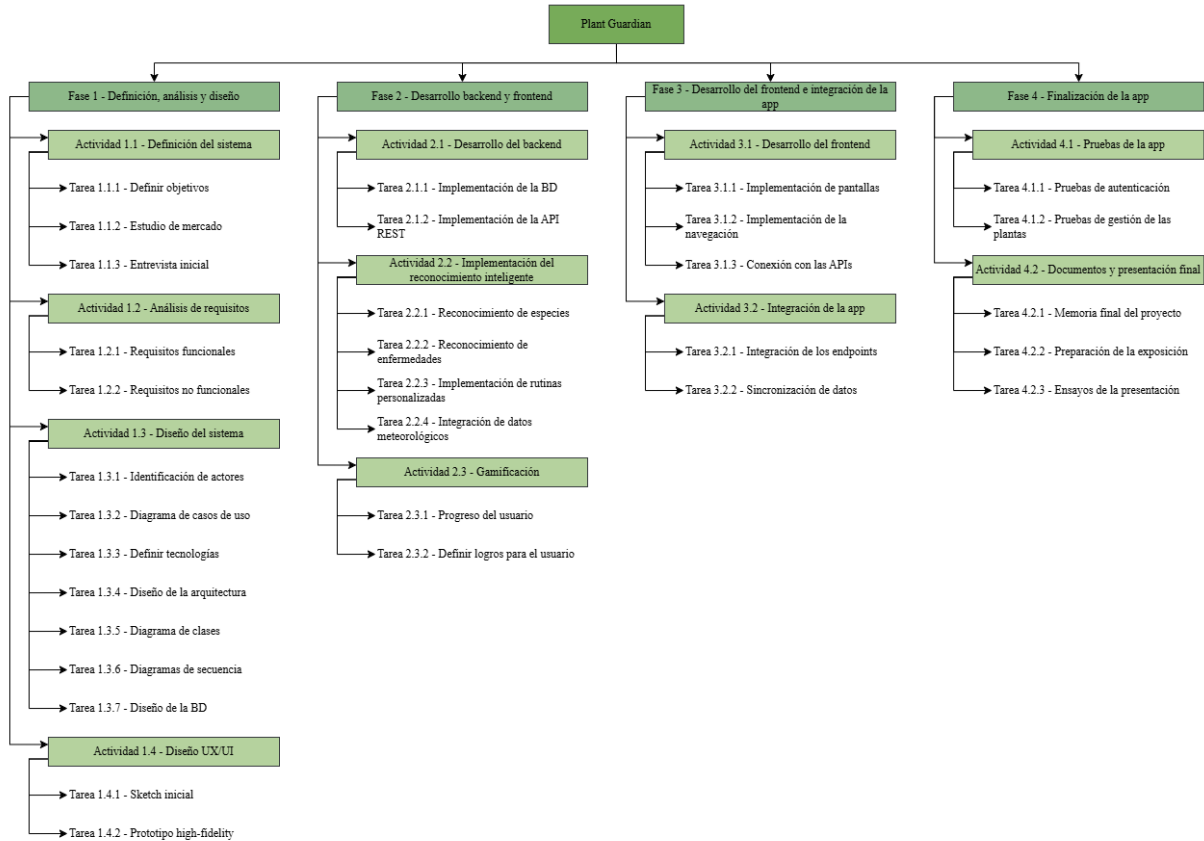


Fig. 9: Diagrama WBS del proyecto.

A.2. Diagrama de Gantt

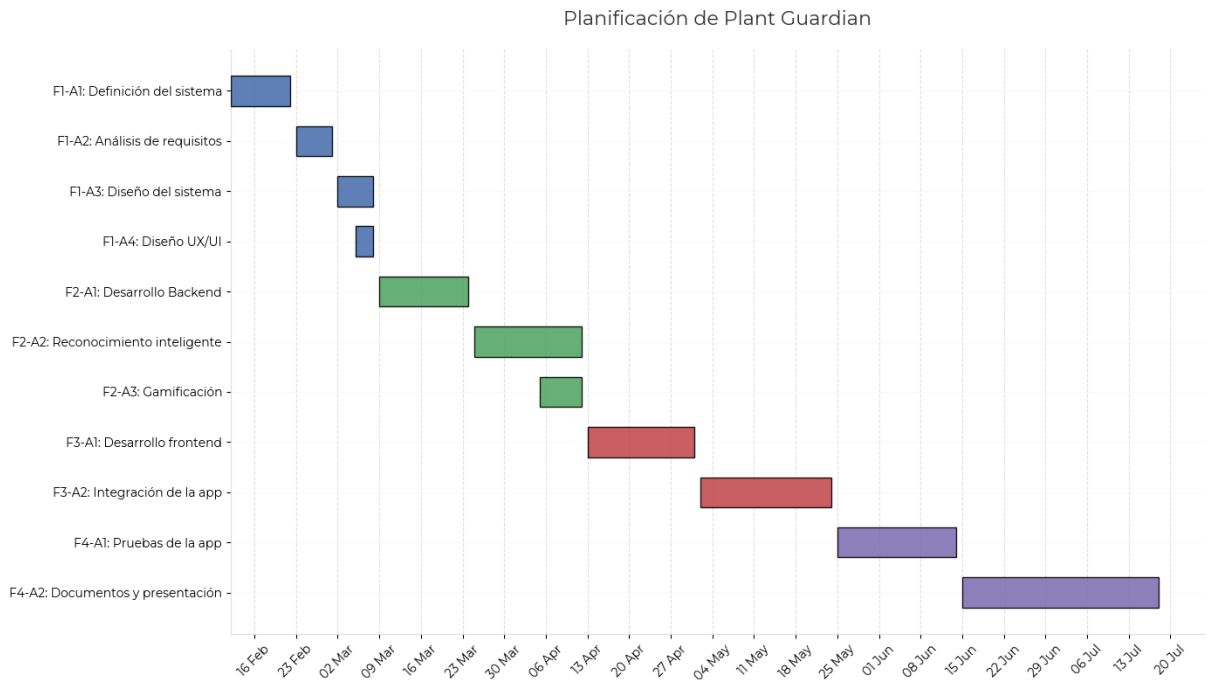


Fig. 10: Diagrama de Gantt del proyecto.

A.3. Requisitos funcionales

TABLA 4: REQUISITOS FUNCIONALES

ID	Requisito	Prioridad
1 - Gestión de usuarios		
RF-1-1	El sistema ha de permitir que el usuario pueda iniciar sesión.	Esencial
RF-1-2	El sistema ha de permitir que el usuario pueda registrarse con su email y su contraseña.	Esencial
RF-1-3	El sistema ha de permitir la gestión de los datos del usuario.	Esencial
RF-1-4	El sistema ha de permitir que el usuario pueda iniciar sesión con su cuenta de Google.	Opcional
2 - Gestión de plantas		
RF-2-1	El sistema ha de identificar la planta a partir de una foto.	Esencial
RF-2-2	El sistema debe realizar un análisis de la planta y mostrar los resultados.	Esencial
RF-2-3	El sistema ha de guardar la información de la planta.	Esencial
RF-2-4	El sistema ha de crear una rutina personalizada para cada planta.	Esencial
RF-2-5	El sistema debe mostrar una lista de acciones a realizar en la rutina personalizada.	Esencial
RF-2-6	El sistema ha de permitir crear colecciones de plantas.	Opcional
3 - Gestión meteorológica		
RF-3-1	El sistema ha de obtener la información meteorológica actual según la ubicación del usuario.	Esencial
RF-3-2	El sistema ha de mostrar la información meteorológica local.	Esencial
RF-3-3	El sistema ha de ofrecer recomendaciones en las rutinas adaptadas al clima actual.	Esencial
RF-3-4	El sistema ha de generar alertas ante condiciones climáticas.	Opcional
4 - Gestión de recompensas		
RF-4-1	El sistema ha de registrar los puntos (XP) según las acciones realizadas en cada planta.	Esencial
RF-4-2	El sistema ha de otorgar un nivel de experiencia al usuario.	Esencial
RF-4-3	El sistema ha de mostrar los logros del usuario en su perfil.	Esencial
RF-4-4	El sistema ha de otorgar recompensas (iconos de usuario o decoraciones en el perfil).	Opcional

A.4. Requisitos no funcionales

TABLA 5: REQUISITOS NO FUNCIONALES

ID	Requisito	Prioridad
1 - Usabilidad		
RNF-1-1	La interfaz debe ser intuitiva, visual y amigable con el usuario.	Esencial
RNF-1-2	La aplicación debe permitir la identificación de una planta en menos de 3 clics desde la pantalla de inicio.	Esencial
RNF-1-3	La aplicación debe tener un diseño responsive que se adapte a diferentes tamaños de pantalla.	Opcional
2 - Rendimiento		
RNF-2-1	El tiempo de respuesta para el reconocimiento de imágenes mediante IA no debe superar los 10 segundos.	Opcional
RNF-2-2	La aplicación debe ser ligera, con un tamaño de instalación inicial optimizado para no ocupar mucho espacio en el móvil.	Opcional
3 - Disponibilidad		
RNF-3-1	El sistema debe tener una disponibilidad del 99 % del tiempo.	Esencial
RNF-3-2	La aplicación debe permitir la consulta en modo offline de las fichas de plantas que ya han sido añadidas al inventario del usuario.	Opcional

A.5. Diagrama de la base de datos

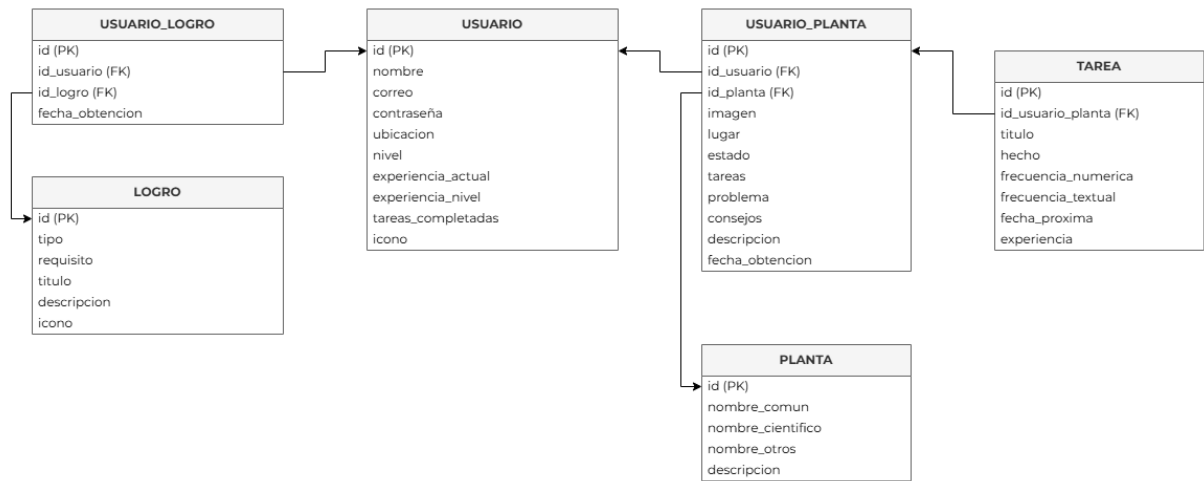


Fig. 11: Diagrama de la base de datos.

A.6. Pantallas principales



Fig. 12: Pantallas principales.

A.7. Flujo de añadir una planta

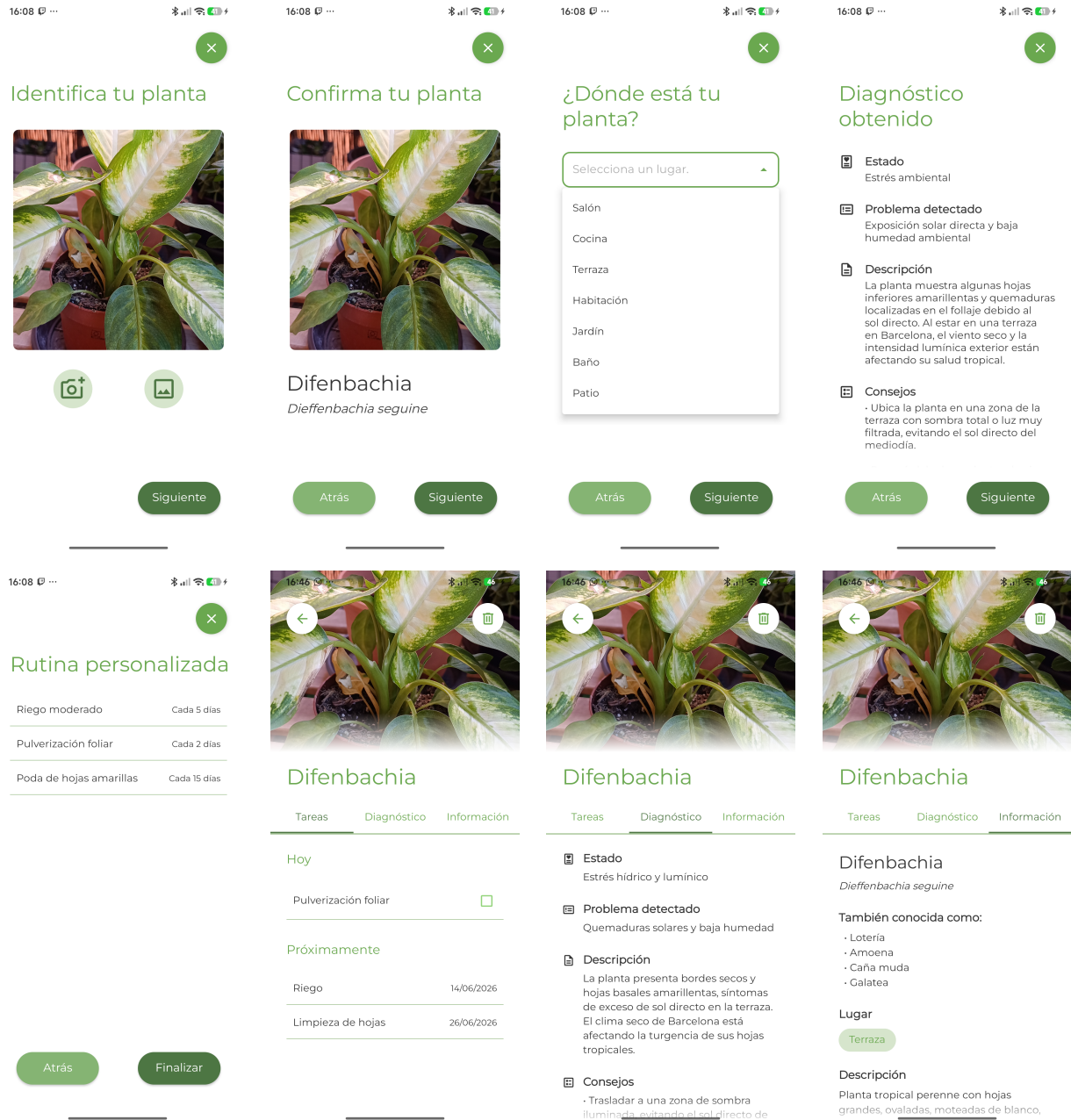


Fig. 13: Flujo de añadir una planta.