MeinPlaner

Aplicación de gestión de bienes personales

Iván Cenizo Domínguez

08/06/2025

2.º Grado Superior, Desarrollo Aplicaciones Multiplataformas

IES SAN JOSÉ

Tutor: Sánchez Yufera, Carlos Alberto

ÍNDICE:

RESUMEN

Meinplaner es una aplicación móvil diseñada para gestionar bienes personales como coches, casas o seguros. Permite al usuario almacenar información de manera sencilla, recibir notificaciones y no olvidar fechas importantes. La aplicación se sincroniza con el calendario de Google para crear eventos automáticamente y puede enviar avisos por notificaciones mediante la App. Con un diseño claro y sencillo, es accesible para cualquier usuario.

PALABRAS CLAVE

Tipo, Vehículos, Seguros, Hogar, Fechas, calendario, bienes personales, recordatorios

INTRODUCCIÓN

En la actualidad, es común que las personas deban atender múltiples bienes, como un coche que requiere revisión o un seguro que debe renovarse. A menudo, esta información se anota en papeles o se pierde el control. Meinplaner surge como una solución sencilla:

1. Es una aplicación que permite registrar y gestionar toda esta información desde el móvil.
2. Ofrece recordatorios para tareas importantes.
3. Funciona en dispositivos Android .
4. Utiliza herramientas como el inicio de sesión con Google o la integración con el calendario para facilitar su uso.

ESTADO DEL ARTE

Existen aplicaciones que gestionan aspectos individuales como coches o alquileres, pero Meinplaner integra todo en una sola aplicación, permitiendo al usuario centralizar la información.

A diferencia de otras, se conecta con el calendario del usuario y almacena los datos tanto en el móvil como en la nube, ofreciendo así mayor seguridad y completitud.

ESTUDIO DE VISIBILIDAD

Análisis DAFO

- **Debilidades**: El desarrollo se hizo por una sola persona, lo que puede hacer que algunas partes tarden más. Además, algunas herramientas usadas eran nuevas y hubo que aprender a usarlas durante el proyecto.

- **Amenazas**: Ya existen otras aplicaciones que hacen cosas parecidas. Si no se ofrece algo diferente o útil, puede costar que la gente la use.

- **Fortalezas**: La app es útil porque junta todo en un solo sitio (vehículos, seguros, casa...). Además, está pensada para que sea fácil de usar y práctica.

- **Oportunidades**: Hoy en día mucha gente usa el móvil para organizar su vida. No hay muchas apps que lo hagan todo junto, así que esta tiene su sitio.

Estudio de mercado

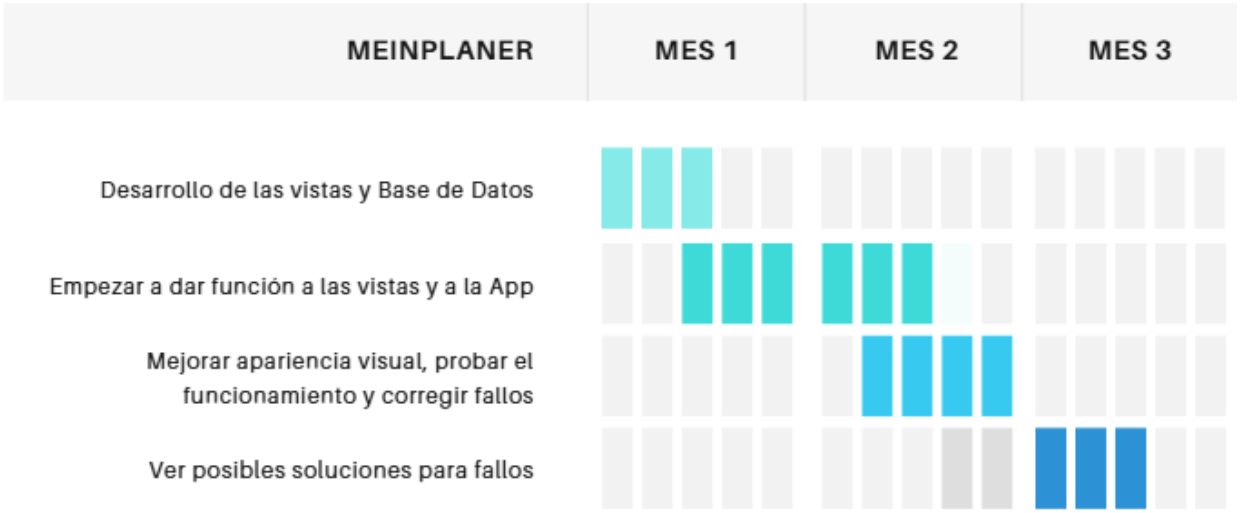
Hay muchas apps para cosas concretas (como seguros o coches), pero pocas que las junten todas. **MeinPlaner** puede gustar a personas que quieren tener organizados todos sus bienes personales en una sola app, sin tener que usar varias diferentes.

Viabilidad temporal

El proyecto se ha podido hacer dentro del tiempo del curso. Se ha organizado por partes y se ha cumplido con los plazos que se pusieron desde el principio.

Planificación temporal

El trabajo se dividió así:



ANÁLISIS DE REQUISITOS

Durante el desarrollo de **MeinPlaner**, se han identificado los siguientes requisitos básicos para que la aplicación funcione correctamente y sea útil para el usuario:

Requisitos funcionales

- **Inicio de sesión y registro**: el usuario puede crear una cuenta o iniciar sesión con su cuenta de Google.
- **Pantalla principal**: desde aquí se accede a las distintas secciones: vehículos, hogar, seguros y calendario.
- **Añadir y editar bienes**: el usuario puede guardar información sobre sus vehículos, casas o seguros, y editarla cuando quiera.
- **Notificaciones**: la app avisa de vencimientos importantes como la ITV o un seguro.
- **Sincronización en la nube**: si el usuario entra con su cuenta en otro móvil, se descargan sus datos automáticamente.

Requisitos no funcionales

- La aplicación no envía los vencimientos de los seguros o ITV al Gmail
- El sistema debe ser seguro en cuanto al acceso a los datos del usuario.

DISEÑO

El diseño de la aplicación **MeinPlaner** se ha dividido en varias partes para representar su estructura, funcionamiento y aspecto visual antes de comenzar con la programación. A lo largo del desarrollo, algunos de estos esquemas han sido adaptados y mejorados.

Diseño conceptual (Entidad-Relación)

Se definió una única entidad principal llamada Bien, que agrupa diferentes tipos como vehículos, hogares o seguros. Cada bien tiene atributos específicos según su tipo.

Diseño lógico (Relacional)

Se representó la entidad principal como una tabla con columnas que agrupan todos los campos necesarios, incluyendo un campo tipo para distinguir entre vehículo, hogar o seguro.

Diseño físico

Se creó una única tabla en la base de datos local (Room) con los siguientes campos:

- id: identificador único (clave primaria)
- tipo: indica si es un vehículo, hogar o seguro
- Campos comunes: nombre, fecha, tipo, etc.
- Campos específicos: matrícula, marca, número de póliza, fecha de ITV, compañía, etc.

Este diseño permite que todos los datos estén en la misma tabla, lo que facilita la sincronización con Firebase Firestore.

Orientación a objetos

- **Diagrama de clases (estructura de código)**

  Se definieron clases en Kotlin que representan los elementos clave de la app:

  - MeinPlaner: clase principal de datos (usada por Room y Firebase)
  - MeinPlanerViewModel: conecta la base de datos con la interfaz
  - MeinPlanerDao: contiene las funciones para insertar, borrar, actualizar y buscar datos
  - Clases por actividad: cada pantalla tiene su propia clase (MainActivity, LoginActivity
- **Diagrama de secuencia (relevante)**

  Un ejemplo sería el flujo cuando el usuario añade un nuevo vehículo:

  1. Usuario pulsa “+”.
  2. Se abre el formulario.
  3. El usuario introduce los datos.
  4. Al pulsar "guardar", los datos se guardan en Room.
  5. Se sincronizan con Firebase.
  6. Se programa la notificación.

Diagrama de actividad

Uno de los más importantes sería el de “Inicio de sesión con Google”, que sigue los pasos de autenticación, acceso y carga de datos en el dispositivo.

Diseño UX

La app se diseñó pensando en que fuera fácil de usar desde el primer momento:

- Menú principal con iconos grandes y claros.
- Acceso rápido a cada tipo de bien.
- Formularios simples con campos obligatorios marcados.
- Colores suaves (amarillo pastel de fondo) combinados con iconos azules.
- Notificaciones visuales con iconos de alerta o recordatorio.

Mockups

Antes de empezar a programar, se diseñaron bocetos (mockups) de cada pantalla en papel , que luego se usaron como guía para crear los XML de cada Activity.

CODIFICACIÓN

La app **MeinPlaner** se ha desarrollado en **Android Studio** usando el lenguaje **Kotlin**, aprendido durante el ciclo. También se han utilizado tecnologías complementarias que aportan funcionalidad y facilidad de uso:

- **Room**: para guardar datos localmente.
- **Firebase (Auth y Firestore)**: para login con Google y sincronización en la nube.
- **Google Calendar API**: para crear eventos con fechas importantes.
- **AlarmManager**: para lanzar notificaciones locales.

Servicios y funcionalidades

- Gestión de bienes personales (vehículos, hogar, seguros).
- Guardado local y en la nube.
- Recordatorios con notificaciones.
- Interfaz simple y adaptada a móviles.

Seguridad

Se usa **autenticación con Google** (Firebase Auth) para que solo el dueño acceda a sus datos. No se guarda nada sensible sin control.

Multiplataforma (limitado)

Aunque es una app Android, **los datos se sincronizan en la nube**, así que se pueden recuperar en cualquier móvil Android con la misma cuenta. Esto asegura la funcionalidad entre diferentes dispositivos Android.

DOCUMENTACIÓN

Durante el desarrollo de **MeinPlaner** se ha creado documentación técnica y de usuario para entender y mantener la aplicación.

Documentación interna

- Se han añadido **comentarios en el código** para explicar las funciones más importantes.
- Cada clase y función incluye información sobre su **propósito**, el **autor** y la **fecha de creación/modificación**.
- Los ficheros están organizados por carpetas según su función: pantallas, base de datos, login, notificaciones, etc.

Documentación externa

Se han creado los siguientes documentos (incluidos en anexos):

- **Guía de instalación**: explica cómo abrir y ejecutar el proyecto en Android Studio.
- **Manual de usuario**: muestra cómo usar la app, con capturas de pantalla.
- **Guía técnica**: explica cómo está construida la app y las decisiones tomadas durante el desarrollo.
- **Resumen de API y dependencias**: se detallan las librerías y servicios usados (Firebase, Room, etc.).

DESPLIEGUE

Esquema de despliegue

La aplicación **MeinPlaner** se ejecuta en dispositivos Android y se conecta con los siguientes servicios externos:

Instalación y despliegue

- La app se ha desarrollado en **Android Studio**.
- Para instalarla, se compila el proyecto y se genera un archivo .apk.
- El .apk se puede instalar en cualquier móvil Android.
- Al iniciar sesión con Google, se sincronizan los datos desde Firebase automáticamente.

HERRAMIENTAS DE APOYO

Durante el desarrollo se utilizaron herramientas para gestionar diferentes versiones del proyecto, facilitando la recuperación de versiones anteriores. También se realizaron pruebas exhaustivas para asegurar el correcto funcionamiento de la aplicación.

CONTROL DE VERSIONES

Durante el desarrollo del proyecto **MeinPlaner**, el control de versiones se ha llevado a cabo de forma manual. Para ello, se han utilizado métodos sencillos pero efectivos:

- **Copias de seguridad locales**: se han guardado versiones del proyecto en distintas carpetas con fechas para evitar pérdidas de datos.
- **Control por fases**: al terminar cada parte importante (pantallas, base de datos, sincronización, notificaciones…), se generaba una copia del proyecto.
- **Organización por carpetas**: se ha mantenido una estructura ordenada para poder recuperar fácilmente versiones anteriores en caso de errores.

Aunque no se ha usado una plataforma como GitHub, se ha procurado llevar un control claro y organizado del avance del proyecto.

SISTEMA DE INTEGRACIÓN CONTINUA

En este proyecto no se ha utilizado un sistema de integración continua como tal (por ejemplo, GitHub Actions, Jenkins o similares), ya que se trata de un desarrollo individual y a pequeña escala.

Sin embargo, se han aplicado algunos principios básicos:

- **Pruebas frecuentes en dispositivos físicos y emuladores** para comprobar el correcto funcionamiento de la app tras cada cambio importante.
- **Revisiones periódicas del código** para asegurar que no haya errores de compilación o problemas de lógica.
- **Organización del proyecto por etapas**, trabajando funcionalidad por funcionalidad para mantener el proyecto siempre operativo.

GESTIÓN DE PRUEBAS

Tipos de pruebas realizadas:

- **Pruebas funcionales**: se ha comprobado que cada pantalla y botón realiza la acción esperada (añadir, editar, borrar, guardar...).
- **Pruebas de validación**: los formularios no permiten guardar datos vacíos o incorrectos.
- **Pruebas de sincronización**: se ha verificado que los datos se suben y bajan correctamente desde Firebase al iniciar sesión en diferentes dispositivos.
- **Pruebas de notificaciones**: se ha comprobado que las alarmas se crean correctamente y se muestran en el móvil en la fecha indicada.
- **Pruebas de usabilidad**: se ha revisado que la app sea intuitiva.

CONCLUSIONES

Grado de cumplimiento de los objetivos

Los objetivos planteados al inicio del proyecto **MeinPlaner** se han cumplido de forma satisfactoria:

- Se ha desarrollado una app funcional para gestionar bienes personales (vehículos, hogar, seguros).
- Se ha implementado correctamente el guardado local (Room) y la sincronización en la nube (Firebase).
- El sistema permite crear notificaciones y eventos para recordar fechas importantes.
- La app es estable, fácil de usar y cumple con los requisitos definidos.

Propuestas de mejora o ampliaciones futuras

Aunque la app cumple su función, se podrían añadir mejoras en el futuro:

- Crear una versión para iOS o usar tecnología multiplataforma.
- Añadir notificaciones push más avanzadas (como correos).
- Incluir estadísticas o gráficos de gastos, mantenimientos, etc.
- Permitir compartir bienes con otros usuarios (familiares, parejas...).
- Adjuntar fotos, documentos o facturas a cada bien.

BIBLIOGRAFÍA

Durante el desarrollo del proyecto **MeinPlaner**, se han consultado distintos recursos para resolver dudas, aprender nuevas tecnologías y tomar ideas de diseño y funcionalidad. A continuación se detallan las principales fuentes utilizadas:

Páginas web consultadas

- **developer.android.com**
  Página oficial de Android, con documentación sobre Kotlin, Jetpack, Room, notificaciones, etc.
- **firebase.google.com**
  Documentación de Firebase (Auth y Firestore).
- **stackoverflow.com**
  Comunidad de preguntas y respuestas de programación.
- **https://dashboard.emailjs.com/admin/account**
  Aunque no se ha podido implementar las notificaciones por correo, se ha creado la API para intentar hacerlo.

Videos de YouTube

Para algunas cosas se ha sacado información sobre algunos videos de la plataforma youtube

Inteligencia Artificial

Se ha hecho uso de herramientas de inteligencia artificial como apoyo durante el desarrollo del proyecto. Siempre que se ha utilizado, se ha comprendido la información proporcionada y en ningún momento se ha abusado de su uso.

Proyectos anteriores y ejemplos

- Se han usado algunos proyectos anteriores para sacar información sobre algunas cosas.

- Ejemplos vistos en cursos online y ejercicios propuestos por el profesorado.