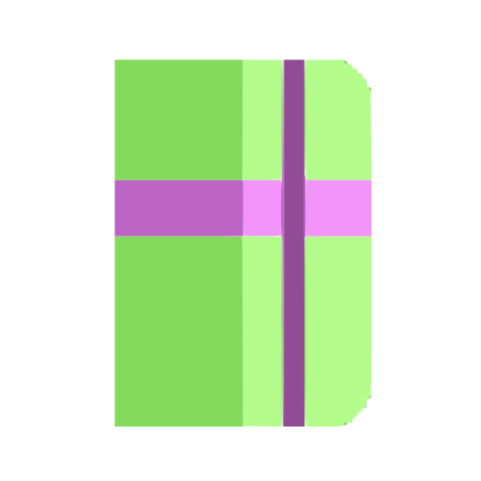
**PROYECTO**

**NoteApp**

****

**CICLO FORMATIVO DE GRADO SUPERIOR:**

**Desarrollo de Aplicaciones Multiplataforma**

**AUTORES:**

**Alejandro Rojas Ruiz**

**Iker Fernández**

**Enrique Moran**

**Licencia**

**Esta obra está bajo una licencia Reconocimiento-Compartir bajo la misma licencia 3.0 España de Creative Commons. Para ver una copia de esta licencia, visite http://creativecommons.org/licenses/by-sa/3.0/es/ o envie una carta a Creative Commons, 171 Second Street, Suite 300, San Francisco, California 94105, USA.**

**[ DEDICATORIAS/AGRADECIMIENTOS]**

Los autores del proyecto, si lo desean, podrán dedicar y/o agradecer el proyecto a quienes consideren oportuno.

**RESUMEN**

NoteApp es una Aplicación Android que podemos categorizar como una App de productividad, concretamente de gestión de tiempo y tareas.

El objetivo de NoteApp es cubrir la necesidad de una Aplicación funcional para poder recordar tareas y eventos pero que a la vez tenga un funcionamiento clásico e intuitivo que recuerde al funcionamiento de una agenda en papel tradicional, por lo tanto el diseño y funcionamiento de la misma irán dirigidos a la semejanza con una agenda en papel

Para ello la idea es desarrollar una aplicación con una pantalla principal que permita recordarnos que tareas debemos realizar en el día, así como introducir nuevas, modificar o eliminar las mismas. Alrededor de todo ello implementamos un diseño atractivo de ventanas con temas personalizables

Conseguimos así que el usuario pueda aumentar su productividad considerablemente, además de que reducir el espacio y peso que supondría una agenda tradicional en papel, así como la mejor aceptación de la nueva tecnología al ser lo más semejante posible al método que el usuario venía usando toda la vida

Se parte de una aplicación ya desarrollada previamente por nosotros mismos que cumple las necesidades más básicas y el esqueleto de la propia App. El objetivo del proyecto es tratar el software de una forma empresarial de alto nivel, es decir, partimos de una aplicación principal básica, sobre la que vamos a desarrollar nuevos módulos y nueva funcionalidad, de forma que en cada sprint podamos añadir cada vez mas funcionalidades nuevas a la aplicación

**ABSTRACT**

NoteApp is an Android Application which we can categorize as an productivity App, an time management and taks app specifically.

NoteApp’s objective is to cover the necessity of an functional Application which allow the user to remind tasks and events, but also has a classic and intuitive functionality which reminds us to a traditional paper agenda, because this design and operation in this app is focused on the similarity with an old paper agenda

To do this the idea is to develop an application with a main screen that allows us to remind which task we have to perform throughout the day as well as add, modify or delete them. Surrounding this we implement an attractive and customizable windows design

We achive in this way the user’s productivity enhance considerably, in addition to reduce space and weight that an old fashioned agenda supposes, as well as a better new technology acceptation as this is similar to the old fashioned productivity methods

We start from a previously developed application by us that cover the most basic necessities and framework of the App. The project objective is to work professionally with the software, starting from an basic main application and working over it to develop new modules and functionalities, so each sprint we can add more and more new functionalities to the app

Índice de contenido

[1 INTRODUCCIÓN 8](#__RefHeading__2216_364215148)

[2 NECESIDADES DEL SECTOR PRODUCTIVO 9](#__RefHeading__4307_478974897)

[2.1 Análisis de la situación actual 9](#__RefHeading__4644_1717060311)

[2.2 Necesidades del cliente y oportunidad de negocio 9](#__RefHeading__4646_1717060311)

[2.3 El nuevo proyecto: Nombre del proyecto 9](#__RefHeading__4648_1717060311)

[3 DISEÑO DEL PROYECTO 11](#__RefHeading__4309_478974897)

[3.1 Fases del proyecto 11](#__RefHeading__864_980353409)

[3.1.1 Análisis 11](#__RefHeading__866_980353409)

[3.1.2 Diseño 11](#__RefHeading__868_980353409)

[3.1.3 Implementación 11](#__RefHeading__870_980353409)

[3.1.4 Pruebas 12](#__RefHeading__928_1438982970)

[3.2 Objetivos a conseguir 13](#__RefHeading__874_980353409)

[3.3 Previsión de recursos materiales y humanos necesarios 13](#__RefHeading__876_980353409)

[3.4 Presupuesto económico. 13](#__RefHeading__878_980353409)

[4 PLANIFICACIÓN DE LA EJECUCIÓN DEL PROYECTO 14](#__RefHeading__4311_478974897)

[4.1 Fase de Análisis 14](#__RefHeading__938_1438982970)

[4.2 Fase de diseño 14](#__RefHeading__940_1438982970)

[4.3 Fase de Implementación 14](#__RefHeading__942_1438982970)

[4.4 Fase de pruebas 15](#__RefHeading__944_1438982970)

[5 DEFINICIÓN DE PROCEDIMIENTOS DE CONTROL Y EVALUACIÓN 16](#__RefHeading__4313_478974897)

[6 FUENTES 17](#__RefHeading__4319_478974897)

[7 ANEXOS 18](#__RefHeading__4321_478974897)

[7.1 Guía de estilo 19](#__RefHeading__7110_478974897)

Índice de figuras

[Diagrama de Gantt 10](#_Toc101911357)

# INTRODUCCIÓN

Este documento responde a la realización del **módulo de Proyecto** del CFGS en Desarrollo de aplicaciones multiplataforma. El módulo de Proyecto complementa, la formación establecida para el resto de los módulos profesionales que integran el título en las funciones de análisis del contexto, diseño del proyecto y organización de la ejecución.

* Propósito del proyecto.

Existen varios propósitos para este trabajo, en primer lugar tenemos la investigación profunda de las necesidades del sector productivo en cuanto a una aplicación Android, y la investigación de la posible viabilidad de la misma como proyecto de desarrollo, innovación e inversión productiva.

A través de este análisis detectamos un posible hueco en el mercado y una oportunidad de negocio para lo que realizamos una investigación para el desarrollo y posterior lanzamiento de la app. Esto nos obliga a investigar sobre la arquitectura de una aplicación Android, así como su interactuación con el entorno Android. Es necesario tener en cuenta las herramientas que nos ofrece el sistema, así como posibles brechas en la seguridad. Por lo tanto uno de los primeros pasos es decidir la versión mínima de SDK del proyecto, y debe ser una que nos ofrece un equilibrio entre seguridad, pragmatismo y apertura al mayor número de usuarios posible

Una vez decidida la arquitectura y el entorno, otro propósito del proyecto es lanzarnos al desarrollo real en JAVA de una aplicación Android, metiéndonos de lleno en el framework que Android Studio nos ofrece para el desarrollo JAVA. Hemos tenido que aprender, interiorizar, desarrollar e implementar todas las técnicas de desarrollo aprendidas en el ciclo, así como aprovechar las que hemos aprendido por nuestra cuenta, como librerías y frameworks nuevos para llevar a cabo un desarrollo limpio y productivo.

Este proyecto ha sido la continuación de una aplicación previamente desarrollada a lo largo del ciclo formativo. Por lo tanto el enfoque del proyecto ha sido el de ponernos en una situación real de desarrollo, en la que partimos de un producto básico que presentar al cliente, al que se le van a ir añadiendo módulos hasta convertirse en una aplicación funcional y competitiva. Pero a la vez una aplicación que puede usarse desde el momento 0, pero a la que estamos proporcionando soporte y mantenimiento constante, así como una amplia gama de mejoras que implementamos partiendo de las peticiones de nuestros usuarios

Git ha sido una parte importante de nuestro trabajo, a través de este proyecto hemos desarrollado una metodología ágil de trabajo a través de Git y Github, trabajando en distintas ramas por cada característica nueva a añadir a la aplicación.

* Fases a desarrollar del proyecto:

1. Estudio de mercado
2. Análisis de la situación del sector
3. Análisis del feedback de usuarios
4. Decisiones de arquitectura
5. Organización de los componentes de trabajo mediante metodología ágil
6. Lluvia de ideas para el desarrollo de nuevos módulos
7. Implementación de módulos
8. Pruebas de módulos
9. Despliegue de módulos
10. Recogida de feedback

# NECESIDADES DEL SECTOR PRODUCTIVO

A continuación se identifican las necesidades detectadas en el sector productivo que originan la oportunidad de negocio que se detalla en los siguientes puntos.

## Análisis de la situación actual

Para el alcance de este proyecto, nos hemos movido en el sector de aplicaciones Android, subsector de aplicaciones de gestión y productividad, subsección de agendas.

En esta subsección encontramos múltiples aplicaciones ya existentes en el mercado que nos ofrecen la posibilidad de aumentar nuestra productividad a través de la escritura y recordatorio de tareas en nuestro terminal Android. A pesar de ser un tipo de aplicación altamente usado nos encontramos que, a pesar de tener una oferta relativamente amplia, en realidad es bastante reducida en comparación con otro tipo de aplicaciones como juegos, entretenimiento, visionado de contenido multimedia etc…

## Necesidades del cliente y oportunidad de negocio

Conforme al análisis de la situación actual del mercado, hemos determinado la existencia de una oportunidad de negocio: las aplicaciones de agendas que emulan una agenda en papel. Existen muchas aplicaciones de agenda en el mercado pero según hemos estudiado no existe ninguna cuyo objetivo sea emular una agenda tradicional de papel, tanto en formato como en estética y funcionamiento. El objetivo de este tipo de agenda no es solo el de satisfacer las necesidades de los clientes que aprecian más la estética y funcionamiento de este tipo de agendas, si no también el de ayudar a esos usuarios cuya transición a la era digital se les hace más difícil. Una agenda más parecida a la que lleva utilizando un usuario tradicional durante toda la vida, ejerce un refuerzo positivo y una motivación mayor a usuarios cuya transición a los formatos digitales se les hace especialmente difíciles

Para ello hemos decidido la creación de una nueva pequeña empresa de tres desarrolladores que van a ejercer distintos roles (Cloud master, Project leader y Design master) respectivamente.

Y con ello llevar a cabo la implementación, despliegue y recaudación del nuevo proyecto.

## El nuevo proyecto: NoteApp

**2.3.1Tipo de proyecto**

Tras lo expuesto anteriormente, la decisión lógica es la de formar una empresa pequeña para llevara cabo el desarrollo implementación y despliegue del proyecto. El objetivo de la empresa en principio no es la diversificación, pero no se descarta el posible objetivo de expandirse y crear más aplicaciones en el futuro.

El proyecto a desarrollar por la empresa es de desarrollo de software, pero además se deberán llevar a cabo tareas de implementación de software, de BBDD, implementación de infraestructura empresarial y de soporte a la aplicación

**2.3.2 Características requeridas al proyecto**

En cuanto a los requerimientos del proyecto:

El proyecto debe ser un desarrollo para Android, para lo cual necesitamos desarrolladores de software especializados en Java o Kotlin. En este caso puesto que somos 3 estudiantes de Desarrollo de Aplicaciones Multiplataforma y hemos sido Ampliamente formados en Java, hemos decidido abordar el proyecto en Java, siendo aún así conscientes de que es probable que la mejor alternativa sea Kotlin, pero tras un breve estudio hemos determinado que no sale rentable la inversión de tiempo de migración y formación a Kotlin para el alcance del proyecto.

Aún así, el aproximamiento con Java nos abre la puerta al uso de librerías y frameworks más extendidos y probados que en Kotlin, usaremos un montón de librerías en el proyecto que nos ayudarán con las animaciones y con la persistencia. De esta forma buscamos diferenciarnos de otros proyectos menos trabajados en este aspecto.

En cuanto a los recursos humanos necesarios para el proyecto, en principio y puesto que el presupuesto no lo permite, vamos a ser los propios incentivadores del proyecto los que vamos a formar parte de toda la fuerza humana del proyecto. Pero estimamos que trabajando de una forma continuada a lo largo del mismo, no tiene por qué ser necesaria más fuerza de trabajo para el desarrollo del mismo.

**2.3.3 Obligaciones fiscales, laborales y de prevención de riesgo**

En principio las obligaciones fiscales del proyecto no van mas allá de la conformación de una sociedad limitada. Pero puesto que el proyecto no aspira a facturar antes de terminar una cierta parte del desarrollo de la aplicación, esta conformación puede retrasarse hasta el despliegue y obtención de beneficios por parte de la empresa

Puesto que nosotros, los conformantes de la empresa somos los mismos trabajadores, tampoco tenemos en principio ninguna obligación laboral legal y aplicable para el estado del proyecto

Sin embargo puesto que hablamos de muchas horas de desarrollo, sería interesante plantear una formación en prevención de riesgos laborales en trabajo de oficina, donde podemos ver recomendaciones posturales, de setup y de descansos tanto mentales como visuales a lo largo de nuestra jornada de desarrollo.

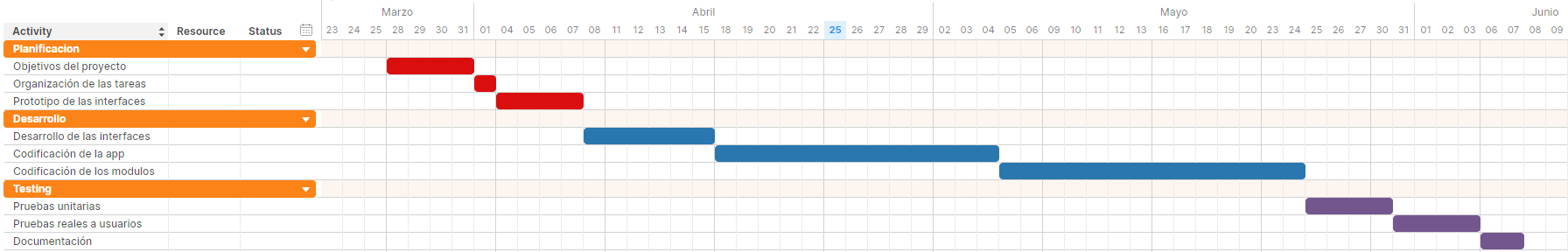
# DISEÑO DEL PROYECTO

Dando por hecho la viabilidad del proyecto, en este apartado se concretarán las fases necesarias para llevarlo a cabo, y cumplir con los objetivos que se establezcan, teniendo en cuenta los recursos necesarios.

## Fases del proyecto

El desarrollo de este proyecto se llevará a cabo en cuatro fases: análisis, diseño, implementación y pruebas, que pasan a detallarse a continuación.

Diagrama de Gantt

**

1 Diagrama de Gantt del pryecto

### Análisis

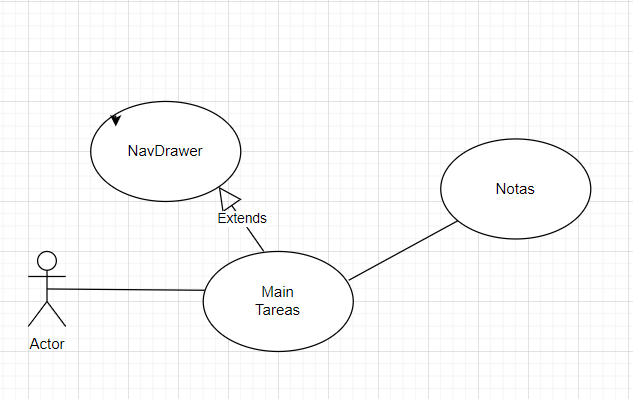
En esta fase se establecerán los requisitos del proyecto, distinguiendo entre los funcionales y no funcionales.

Los requisitos mínimos de la aplicación son estos:

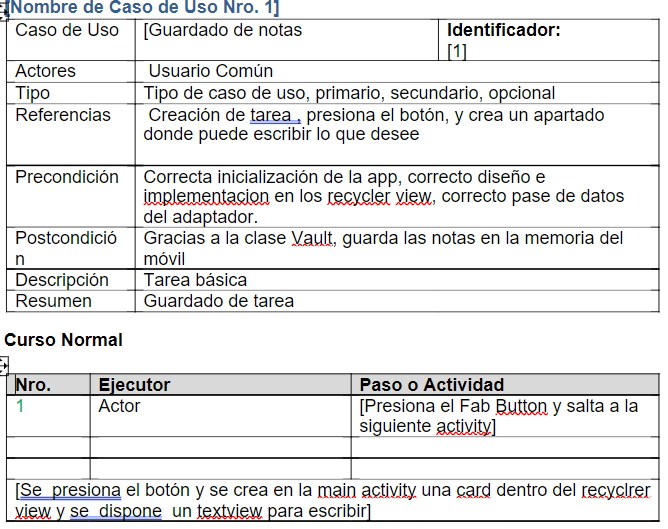
Funcionales:

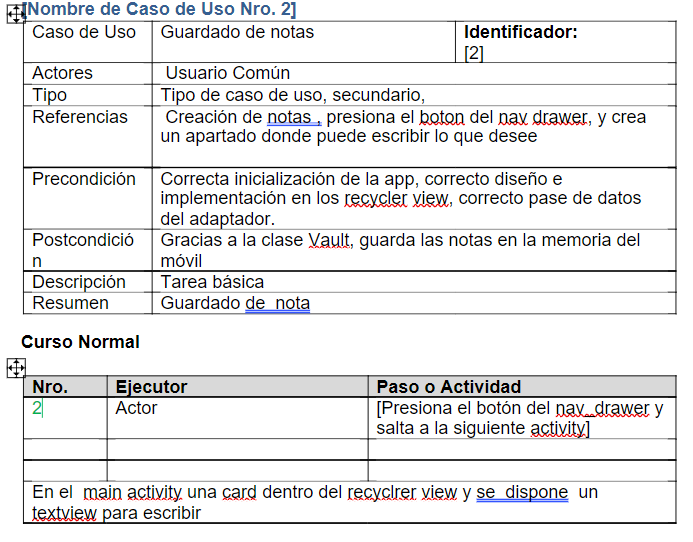
* Estabilidad en General
* Guardado y persistencia de notas
* Guardado y persistencia de tareas
* Cambios de Temas
* Persistencia de fotos dentro del apartado notas
* Elección de color en tareas y notas

Diagrama de Casos de Usos



Casos de Usos posibles





No funcionales:

* El proyecto tiene que satisfacer las necesidades de un usuario, que quiera apuntar una nota o una tarea de forma rápida.
* El diseño de la App esta basado en la típica agenda no digital, para que el usuario lo identifique con algo conocido pero mejor
* El alcance de esta aplicación es sobre todo para uso de cosas cotidianas, puede utilizarse para el ámbito laboral, pero deberian de meterse mas opciones por lo tanto es para algo rápido y funcional.
* La fiabilidad del guardado de notas se ha hecho latente gracias al uso poder guardar las notas en Google Drive y por el uso de métodos de persistencia como ObjectBox.(Base de datos embebida).
* Las restricciones mas clara es el la incapacidad de crear cuentas y poder guardar todos estos datos en una API. Aparte de mas opciones orientadas al mundo laboral

### Diseño

En esta fase se realiza una aproximación al diseño tecnológico de la solución.

Describir **cómo** realizar cada uno de los requisitos establecidos en la fase anterior.

Definir la estructura de la aplicación, el diseño de los componentes (BBDD, servidor web, clientes...)

### Implementación

Partiendo del diseño, en esta fase se construye el proyecto.

Por ejemplo, se lleva a cabo la implementación de la página web en el lenguaje que se haya determinado, la creación de las tablas de la BBDD, la carga de datos...

### Pruebas

Son muchas pruebas que pueden realizarse en un proyecto, para eliminar los posibles errores y garantizar su correcto funcionamiento. Los casos de prueba establecen las condiciones/variables que permitirán determinar si los requisitos establecidos se cumplen o no. A continuación se detallan algunos de los casos de prueba que se ejecutarán para comprobar la correcta construcción de este proyecto.

A modo de ejemplo se facilitan los campos de una posible plantilla para definir casos de pruebas, que deberá modificarse como se estime oportuno en función del proyecto y las pruebas que se consideren oportunas realizar.

* Fecha/autor/[ versión a probar]
* Caso de prueba

Identificador del caso de prueba ( nombre único)

* Descripción

Breve explicación sobre el objetivo del caso de prueba

* Condiciones de ejecución

Descripción de las condiciones de ejecución que se deben cumplir antes de iniciar el caso de prueba, por ejemplo, que se haya realizado correctamente el login en el sistema...

* Entrada

Datos necesarios para poder ejecutar la prueba

* Resultado esperado

Valor esperado para el correcto funcionamiento del proyecto

* Resultado obtenido

Valor de salida al ejecutar el caso de prueba

* Evaluación

Comparación del valor esperado y obtenido para concluir, finalmente, si el aspecto chequeado por el caso de prueba confirma el correcto funcionamiento del proyecto o elevación detallada del correspondiente error.

A modo de ejemplo se facilitan los campos de una posible plantilla para reportar los errores detectados en los casos de prueba ejecutados.

* Fecha/autor/[ versión probada ]
* Caso de prueba
* Evaluación

Comparación del valor esperado y obtenido para concluir, finalmente, si el aspecto chequeado por el caso de prueba confirma el correcto funcionamiento del proyecto o elevación detallada del correspondiente error.

* Posible causa de error

Detallar la posible causa o causas que han podido generar el error detectado.

* Posible corrección

Detallar la posible forma de corregir el problema

* Áreas afectadas

Detallar qué áreas (módulos, componentes, documentos,...) se verán afectados al implementar la corrección.

## Objetivos a conseguir

Se pueden diferenciar objetivos que son para el desarrollo (cumplimiento de los requisitos técnicos) del proyecto o bien del negocio, de la empresa, financieros,...

* Extender el uso en la empresa de nuevas metodologías, estándares, procedimientos de trabajo...
* Abrir el mercado internacional
* Satisfacer los requisitos del cliente con un valor añadido
* Facturar 100.000.00 en 2015
* Tener 150 clientes el primer año.
* Contar con 3000 visita en la página web el primer mes

## Previsión de los recursos materiales y humanos necesarios

Se tendrá en cuenta las herramientas y la formación necesaria para desarrollar las actividades que requiere el proyecto, así como el tiempo para llevarlo a cabo.

## Presupuesto económico.

Detallar el coste económico de los recursos anteriormente establecidos.

# PLANIFICACIÓN DE LA EJECUCIÓN DEL PROYECTO

A continuación se detallan las actividades/tareas/procedimientos por cada una de las fases del proyecto previamente establecidas.

[Estas tareas podrán variar en función del proyecto]

## Fase de Análisis

1. Estudio de las necesidades a cubrir

Observando nuestro entorno, podemos decir que la vida en esta época en la que la tecnología ha facilitado la posibilidad, y la necesidad, ya que así lo requiere para la mayoría de la gente común, de realizar muchas actividades a lo largo del día, es decir prácticamente nos hemos visto obligados a ser multitarea. Con lo que ello implica inmediatez y una necesidad clara de orden .

NoteApp quiere ofrecer al usuario ese orden de una manera que ya conoce, las antiguas agendas antes eran muy útiles , pero con lo antes ya expuesto hemos decidido dar un paso mas , y ofrecer esa agenda , adaptada a los nuevos tiempo, para facilitar la consecución correcta de los proyectos personales del usuario, dándole ese orden, con una herramienta básica ya conocida, la agenda tradicional.

1. Estudio de la situación actual

Actualmente en el mercado podemos encontrar aplicaciones de agendas por todo el mercado , estas son nuestras principales competidores:

* Google Calendar: Este sería nuestro principal competidor ya que el gigante google ofrece muchas herramientas útiles que han entrado prácticamente en la vida cotidiana de muchas personas , ha conseguido normalizar el uso de sus servicios.

También permite la sincronización de cuenta de gmail, permitiendo tener todo al alcance de una manera rápida.

Sabemos que esto es muy positivo, pero tiene su partes negativas, como olvidos de contraseña, no siempre tenemos que tener conexión a internet, fallos en la sincronización, además no permite el uso de notas .

NoteApp quiere destacar por ser simple y cómodo , y evitar todos estos temas que producen quebraderos de cabeza al usuario.

Aún asi también utilizamos tecnologias de google como Drive, en la que podemos guardar las tareas, por si queremos tener ambas cosas , es decir queremos compatibilizar el mundo clásico “analógico” con las tecnologías actuales.

* Evernote: Es una potente aplicación de agendas, con una interfaz intuitiva, tiene la opción de crear notas y tareas , como la nuestra pero se ha analizado lo errores mas típicos que intentaremos mejorar.
* Es obligatorio el registro, y algo que consideramos nosotros esencial no lo tiene: no puedes acceder sin una conexión de internet. La aplicación de NoteApp , su filosofia es de gratuitidad y el fácil acceso.
* El numero de notras creadas por NoteApp es ilimitada, tanto como la capacidad que tenga el dispositivo, en cambio e vernote tiene un limite de 60mb al mes en su versión gratuita
* No hay recordatorios , ni de fechas , ni en horas
* Su versión premium es muy cara

OneNote:

* Aplicación de agendas propiedad de Microsoft bastante versátil y flexible, permite organizar el contenido en blocs de notas y páginas
* Permite etiquetar las notas
* Puedes escribir con una herramienta similar a un lápiz y añadirle colores
* Puedes grabar audios e insertar archivos.
* Permite compartir estas notas

Desde nuestras perspectiva son herramientas muy útiles, pero NoteApp aboga por menos es más, pensando en un usuario común sabemos que estas herramientas la mayoría de veces pasan de manera desapercibida, un gran numero de opciones puede abrumar al usuario, por lo que la estética útil y la simpleza es nuestra máxima.

Espacio ilimitado

Otros fallos que hemos analizado son las carencias que tiene su versión movil en comparación con su versión de escritorio, falla al aplicar formatos de texto , y no permite colores cosa que NoteApp si que puede.

Agenda por defecto:

De manera predeterminada en los dispositivos Android/IOS viene con una aplicación de notas.

Son muy útiles y los fallos que vemos que podemos superar , es que generalmente estas aplicaciones caen en el olvido.

El sistema de guardado , no es del todo satisfactorio para los potenciales usuarios a los que hemos preguntado, sobre todo, suele faltar organización, tanto en horarios y en fechas lo que provoca una mezcla de tareas que abruma el usuario.

NoteApp por defecto permite organizar por fechas, por lo que es mas organizada.

Además de ser aplicaciones que llama poco la atención del usuario, son simples como la nuestra pero nosotros transmitimos esa vuelta a las agendas clásicas lo que permite la mas fácil interacción con el usuario.

1. Establecimiento de los requisitos del proyecto

Los requirimientos de nuestra aplicación van destinados a satisfacer al usuario común, esta aplicación esta pensada para el uso cotidiano , y también para el uso profesional.

El usuario tiene un acceso rápido a la aplicación , no es necesaria tener una cuenta,eso lo dejamos a opción del usuario por si necesita el guardado de estas notas.

El usuario se encuentra de una manera directa con el calendario del año,mes y dia.

De una manera llamativa encontramos un botón flotante en la posición mas intuitiva para ello (abajo según los estandares estudiados sobre las aplicaciónes móviles) y permite el crear las tareas que el usuario considere.

En esa misma pantalla el usuario puede observar las distintas opciones posibles, accede a la sección de notas.

En esa misma pantalla puede acceder a fragments de customización de la aplicación y de notas , para apuntar cosas que no necesita una fecha en concreto.

1. Valoración comparativa de las posibles soluciones

El primer paso es identificar los candidatos de nuestra posible competencia cosa que ya esta realizado en el apartado anterior de una manera general.

Experiencias: En las primeras versiones de NoteApp los usuarios consultados destacaron que no era intuitivo nuestro sistema de calendario y fechas por lo que implementamos mejoras de movimiento y posicionarlo de una manera intuitiva, en comparación con app como Google Calendar, Evernote y One Note , destacaron la rápida utilidad que tenia la aplicación , al ir mas directamente al grano .Las críticas de las ya mencionadas aplicaciones son:

* Google Calendar: Al estar intrínsecamente relacionada con la cuenta de google, tiene excesiva seguridad en persepectiva de nuestro usuario objetivo por lo tanto , puede resultar molesto, sensación de pereza, ganas de no utilizar la aplicación..., Queremos continuar en esta linea mejorando lo ya desarollado
* EverNote: Muchas opciones , que sin la versión premium puede resultar incompleto además que sufre lo mismos problemas de gratuituidad y fácil acceso.

Además de tener limitaciones en cuanto el numero de notas.

El sistema de notas esta inspirado este.

* OneNote: El sistema de audios y guardar archivos es muy bueno, en NoteApp tambien tenemos un sistema de guardado en el que puedes almacenar fotos.
* Agendas predeterminadas: Los usuarios de estas apps se quejan de la falta de organización y echan en falta una plantilla para poder organizarte de una manera mas rápida.
* Costos: NoteApp es una aplicación gratuita, la filosofía de el equipo creador quiere que sea una aplicación , que crezca por la calidad y el buen uso de los usuarios , prácticamente que se una necesidad diaria en el usuario
* Tamaño del despliegue o del mercado: Existen mucha competencia , pero con la filosofia de NoteApp pretendemos diferenciarnos del resto, por su estética, rapidez, sencillez y utilidad
* Soporte: NoteApp es un proyecto realizado por tres estudiantes de desarrollo multiplataforma, que ofrece una aplicacíon útil con una buena proyección de futuro buena, dependiendo de la acogida se vería la inversión en esto. Al ser un trabajo de Fin de grado el código queda abierto y queda en propiedad del Instituto, aparte del éxito fuera de ser un trabajo propio dicho , será un ejemplo bueno para futuros estudiantes.
* Dependencias:
  + Es una aplicación destinada para dispositivos Android.
* Confiabilidad:
  + El sistema de seguridad de NoteApp depende mucho de la responsabilidad del usuario, aun así se ofrece guardar notas en la nube de Google para la confortabilidad del usuario
* Escalabilidad
  + La escalabilidad en este contexto se refiere a la capacidad de la aplicación de manejar un cierto tamaño de problema o de cantidad de datos .En este caso ofrecemos un manejo de datos tanto como quepa en el dispositivo, o espacio disponible en la cuentra de Google Drive
  + Usabilidad:

La usabilidad mide la calidad de la interfaz aplicación-usuario. Una aplicación con buena usabilidad permite aprender a usarla en forma rápida y natural, en nuestro caso es una pantalla principal que te lleva a tareas y notas. No te pierde por distintas opciones.

* Flexibilidad

La flexibilidad del producto se refiere a su capacidad de adaptarse o ser adaptado a requerimientos que hoy no son contemplados, pero que puede ser necesario i incorporar en el futuro.

NoteApp al ser de código abierto tiene la ventaja, de que gracias a ala opinión de los usuarios, puede ser mejorado, justo en lo que los usuarios demandan.

* Interoperabilidad:
  + De momento es una aplicación destinada a la plataforma Android
* Tipo de licencia:
  + En este caso es copylefting pueden ser modificadas, pero las modificaciones deben ser publicadas bajo las mismas condiciones en que está publicado el original. La mayoría de las aplicaciones/licencias utilizan copylefting, como por ejemplo, GPL
* Aspectos legales:
  + El tipo de licencia más común, GPL, que por defecto no se dará garantía sobre el producto.

1. Identificación de las necesidades que implica el nuevo proyecto en la empresa.

Para solventar los problemas que plantea el proyecto puede ser necesario contratar personal, formarlo en determinadas metodologías/herramientas, comprar equipos...

Necesidades para realizar este proyecto son equipos que soporten Android Studio y demás dispositivos para el testeo de pruebas. El personal para este trabajo es de tres personas, suficientes para realizar este proyecto.

La metodología de trabajo es la de reparto de trabajo según las cualidades individuales de los creadores, además del uso de tableros de trabajo como Trello que permite el reparto efectivo y la obtención de ideas del equipo en general.

El máximo problema es la falta de tiempo para entregarlo en una fecha relativamente temprana.

1. Estudio de viabilidad de la solución elegida teniendo en cuenta no solo los beneficios económicos.

La solución elegida para desmarcarnos de aplicaciónes punteras ha sido la de la gratuitidad y utilidad rápida, frente a aplicaciones, con opciones de pago y la incomodidad de excesivas utilidades.

1. Corrección de posibles errores

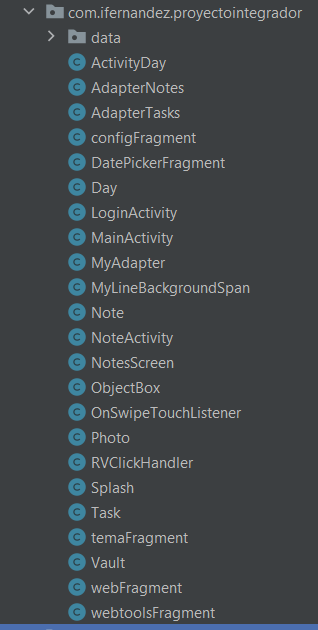
## Fase de diseño

1. Preparación del entorno de diseño
2. Diseño de la arquitectura
3. Diseño de los interfaces
4. Diseño de los datos
5. Diseño de los procedimientos
6. Corrección de posibles errores

## Fase de Implementación

1. Preparación del entorno de implementación

Esta es la estructura de Activitys,Fragments y Clases



2 Clases de NoteApp

Activitys

* ActivitySplash
  + Es la carta de presentación de nuestra App
* ActivityMain
  + Es la primera activity que se ve al iniciar la aplicación, esta contiene los días de la semana en la que se puede introducir nuestras notas, y permite de acceso a las otras activitys.
* ActivityDay
  + Contiene la fecha elegida y además un Fabbutton en el que agregar nuestras tareas
* NoteActivity
  + Permite la creación de notas personalizadas
* NoteScreen
  + Contiene un Recyclerview en la que posicionar las notas
* LoginActivity
  + Permite el acceso a cloud, para guardar en la nube las notas.

Fragments

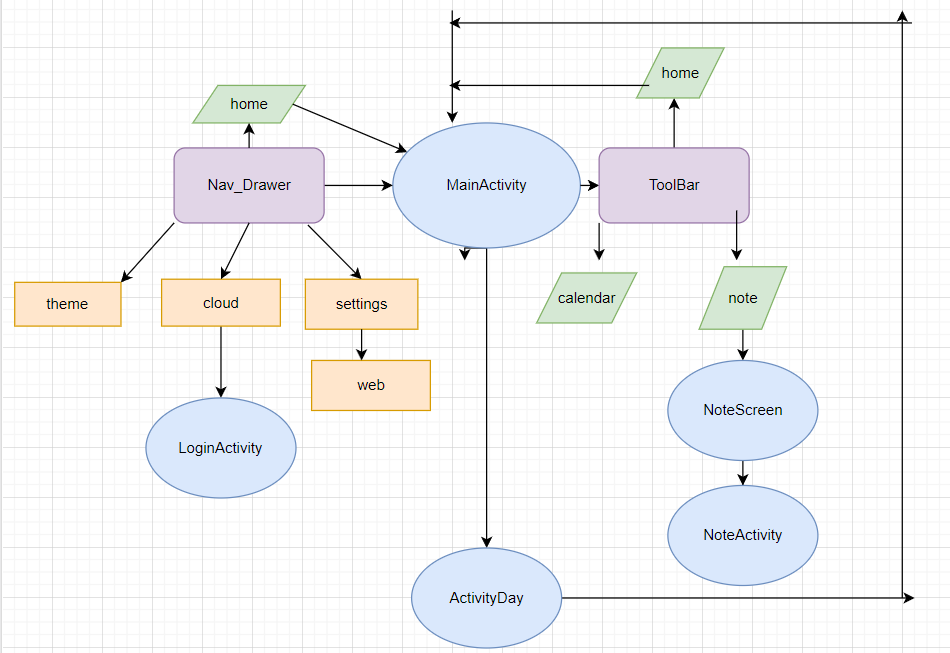
* TemaFragment:
  + Desde el nav\_drawer accede a un fragment que permite la selección de un tema para nuestra app.
* ConfigFragment
  + Desde el nav\_drawer accede a un fragment que permite ver los créditos en forma de lista de los creadores de la app.
* WebFragment
  + Al seleccionar a cualquier creador de la lista de “ListItem” permite ver sus respectivos GitHub.
* DatePickerFragment
  + Desde la toolbar accede a un calendario en el que seleccionar un dia,fecha,mes y año

Clases

* Day
  + Contiene los atributos de dia y se las pasa a una lista(tasklist)
* MyLineBackgroundSpan
  + Permite colorear nuestra tarea, para darle distinción
* Note
  + Contiene los atributos de notas (titulo,descripcion,foto)
* ObjectBox
  + Base de datos para persisitir los objetos
* OnSwipeTouchList
  + Permite desplazar lateralmente la tarjetas de la MainClass.
* Photo
  + Clase foto para implementar a las notas
* RvClickHandler
  + Escucha los clicks para realizar eventos de los RecyclerView Implementados
* Task
  + Contiene los atributos de las tareas que estan dentro de los dias de la clase Main
* Vault
  + Clase para serializar

1. Desarrollo de la arquitectura

Esta es la arquitectura que tiene la App



3 Arquitectura interna de la APP

1. Desarrollo de los interfaces, los datos y los procedimientos

\*Se excluyen métodos getter,setter y repetidos

Activitys

* ActivitySplash :
  + private void openApp(boolean LocationPermision):

Contiene un intent que hace que la aplicación pase a la MainActivity.

* ActivityMain:

Actividad principal contiene los recyclerview, navigation drawer y una toolbar.

El oncreate() recoge los datos de la lista semanal, actualiza los dias de la semana además de refrescar los recyclerview

public boolean onCreateOptionsMenu(Menu menu):

* + Infla la toolbar es decir le pasa el layout.

private void setDrawerNavView():

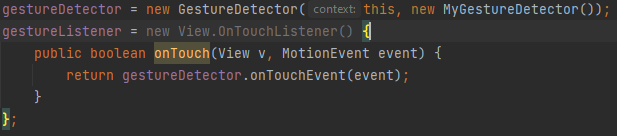
* Es el menu Lateral gracias a un switch que según el item que clickees te lleva aun fragment o activity.

Para llevarte a un fragmente lo hace gracias a fragmentmanager y si es una activity te lleva gracias a activityResultLauncher

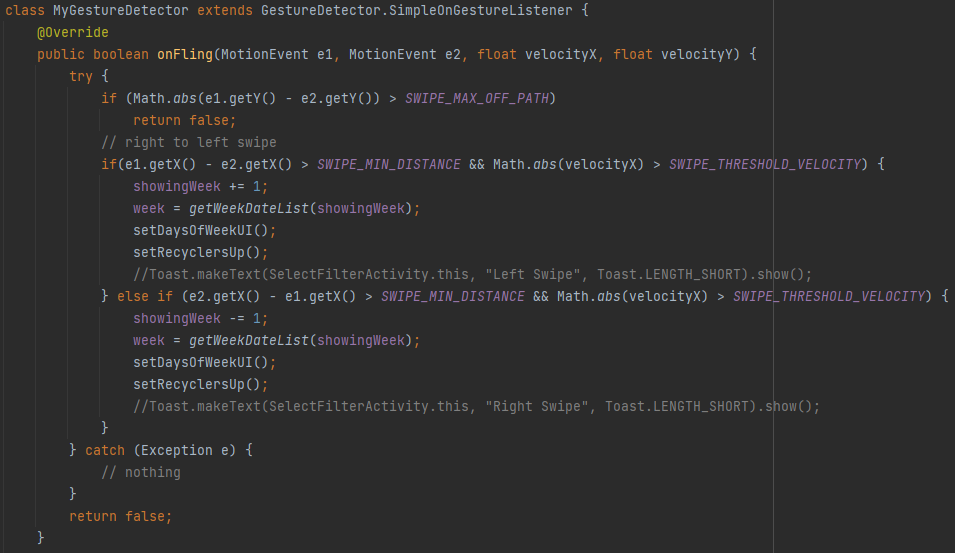
Contiene un if que permite el el paso a un fragmento, si la transaccion es true,reemplaza la vista actual por la del fragmento deseado

GestureDetector():

* Con esta clase lo que hacemos es gestionar el movimiento lateral sobre la pantalla del dispostivo , gracias a esta clase también permite que el menú lateal se despliege con soltura y fluided.



Con este fragmento de código activamos la gestión del evento deslizante y con la siguiente clase , como se puede apreciar le pasamos como parámetros el contexto del activity y nuestra clase personalizada para gestionar dicho evento



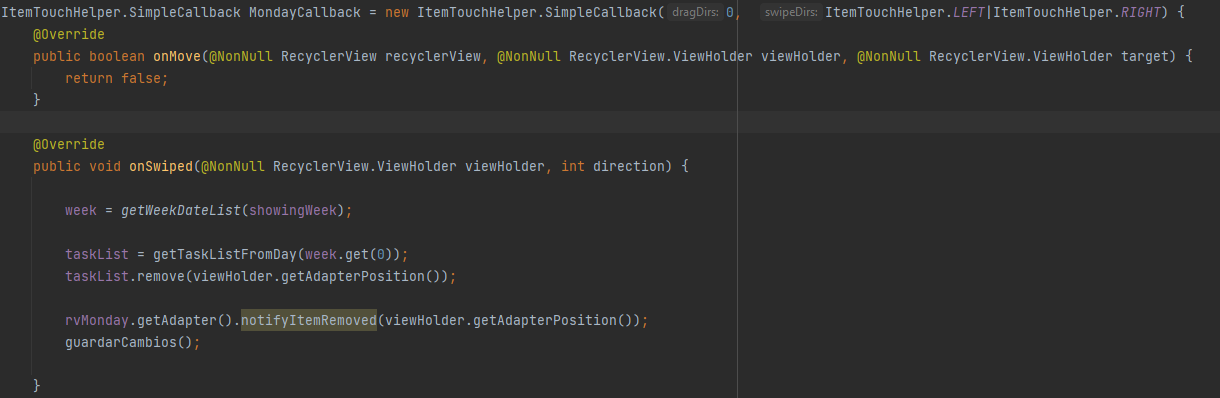
Sobreescribimos el método onFling() y y comprobando las distancias y velocidades de deslizamiento podemos manejar el evento de deslizar a izquierda o derecha, en nuestro caso si desliza a la izquierda retrocederemos a la semana anterior y actualizando los recycler para que se cargen tanto las notas como que cambien los días de la semana.

Solo faltaría asignarle este evento a una vista , nostros lo hemos asignado en el layout del contenido del main, para que en la ventana de inicio podamos navegar entre las semanas con facilidad.

private void setItemTouchUp():

* Esta método nos permite eliminar las tareas desde la ventana principal deslizando

Crea un evento onSwiped() con el que podemos gestionar el deslizar a la izquierda o a la derecha, en nuestro caso , cuando deslizes a la izquierda te marcara la tarea como completada, si deslizas a la derecha lo eliminara.



En el evento, tendremos que eliminar de la persistencia la nota que acabamos deslizar , para ello obtendremos de nuestra clase de persistencia Vault(),

La semana en la que esta nuestro usuario, con ello obtendremos la lista de tareas de ese dia que ha deslizado , ya que cada recycler representa un dia de la semana, en el caso de la imagen seria para el lunes, por lo tanto al deslizar en una nota del lunes, obtendremos la lista de las tareas del lunes, después tendremos que eliminar de esa lista el elemento deslizado.

Para ello con viewHolder.getAdapterPosition() sabremos que elemento es el deslizado y podremos eliminarlo, después actualizaremos el recyclerview para que se ordenen las tareas después de haber eliminado una de ellas y después actualizaremos la persistencia para que guarde la eliminación de la tarea.

Una vez configurado de manera correcta el evento, tendremos que asignarle el recyclerview sobre el queremos que se active el evento.

Para nuestra aplicación, tendremos que asignárselo a los 7 recyclerview que tiene la ventana principal

C:\Users\iker\AppData\Local\Microsoft\Windows\INetCache\Content.Word\itemtouch2.png

private ArrayList<Task> getTaskListFromDay(Date day):

* + Este método esta relacionado directamente con la persistencia de las tareas, permite obtener la lista de tareas de un dia en concreto.

private void setRecyclerClickEvent():

* + Con este método manejamos el evento de pulsar sobre el recyclerview de un dia en concreto, el activity de destino será ActivityDay(), donde podrá escribir y crear tareas

private void setDaysOfWeekUI():

* + Metodo para cargar el texto de los textviews, para que aparezca el dia y el mes según la semana en la que estemos.

public static ArrayList<Date> getWeekDateList(int plusWeek):

* + Nos devolverá una lista con los días de las semanas desde la de inicio hasta la que le pasemos por parámetro.

public static ArrayList<Date> getWeekDateList() :

* + Devuelve un array con el numero de los días de la semana actual.
* ActivityDay

Contiene la fecha elegida y además un Fabbutton en el que agregar nuestras tareas.

El Oncreate() carga el tema, y respectivos bundle para guardar la información de los dias.

Contiene un for que busca y carga la lista de tareas para asignarselo al dia correcto. Si no hay ningun dia guardado, crea una nueva lista de tareas, a continuación carga el recycler con las tareas que estan en “tasklist”

private void ponerTema():

* + Permite el cambio de tema de la app y gracias a las sharedpreferences permite guardar el tema en local

private void setTextviewsUI():

* + Es un switch que extrae Strings según el dia y el mes correspodiente, finalmente se visualizan en los textview.

public void onBackPressed():

* + Este método guarda todos los cambios en un fichero cuando presionas el botón atrás gracias a la clase Vault que permite su persistencia, que lo transporta gracias a un intent.

private void setUpRecycler():

* + Actualiza el recyclerview con los valores correctos.

public void addTaskButton(View view):

* + Añade una tarea al presionar el botón, y se añade a la tasklist, despues el recyclerview se actualiza con los datos sacándolo del adaptador.
* NoteActivity
  + Permite la creación de notas personalizadas

public void onCreateContextMenu(ContextMenu menu, View v, ContextMenu.ContextMenuInfo menuInfo):

* + Añade items al menu

public boolean onContextItemSelected(MenuItem item):

* + Es un listener para cuando clickeas los items del menu

private void initNote():

* + Recupera la nota abierta o crea una en caso de que no haya
* NoteScreen

private void configureButton():

* + Añade una nueva nota al clickar el fabbutton

public void openNote(int notePos):

* + Al clickar una nota se abre
* LoginActivity

El oncreate() configura la peticion de logueo con el id del usuario, correo electronico, y los datos basicos del usuario.

Además empaqueta los datos para poder pasarlo.

Construye además un botón que permite el logueo

Fragments

* TemaFragment:

De una lista de Strings en los que estan guardados los temas hace que al selecciona de un spinner , gracias a un switch nos cambie el tema seleccionado, además lo guarda en local con las sharedpreferences.

Si no seleccionas nada pasa al tema predeterminado.

* ConfigFragment
  + Desde el nav\_drawer accede a un fragment que permite ver los créditos en forma de lista de los creadores de la app.

Con glade muestra las fotos de los creadores, y con un Onclick hace que ese fragmento pase a otro gracias a fragment manager

* WebFragment
  + Al seleccionar a cualquier creador de la lista de “ListItem” permite ver sus respectivos GitHub.
* DatePickerFragment
  + Desde la toolbar accede a un calendario en el que seleccionar un dia,fecha,mes y año

Clases

* Day
  + Contiene los atributos de dia y se las pasa a una lista(tasklist)
  + Es una clase serializada.
* MyLineBackgroundSpan
  + Permite colorear nuestra tarea, para darle distinción
* Note
  + Contiene los atributos de notas (titulo,descripcion,foto)
* ObjectBox
  + Base de datos para persisitir los objetos
* OnSwipeTouchList
  + Permite desplazar lateralmente la tarjetas de la MainClass.
* Photo
  + Clase foto para implementar a las notas
* RvClickHandler
  + Escucha los clicks para realizar eventos de los RecyclerView Implementados
* Task
  + Contiene los atributos de las tareas que estan dentro de los dias de la clase Main
* Vault
  + Clase para serializar

### Persistencia del proyecto

#### Objectbox

La persistencia del proyecto base estaba realizada en base a la serialización de objetos java.

Este sistema, aunque funcional es precario, poco escalable, lento y en general no es una solución final de persistencia para un proyecto profesional.

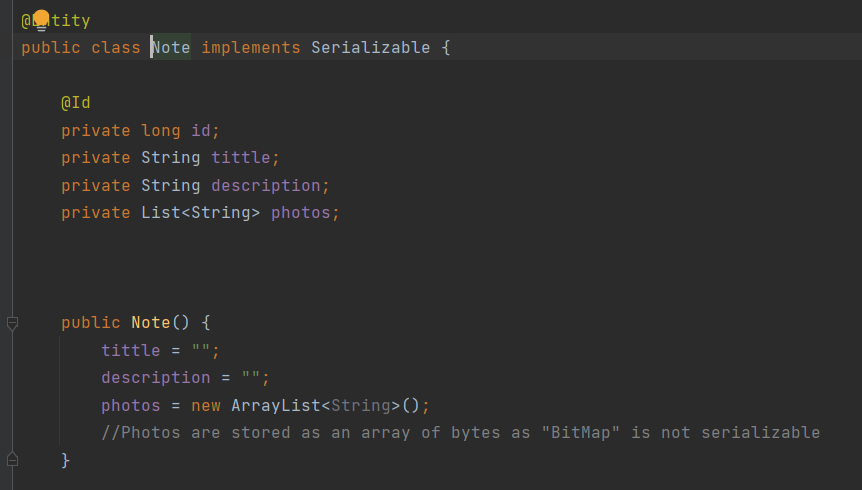
Por esto nos hemos visto obligados a migrar el sistema de persistencia de la APP. Tras una serie de investigaciones, la mejor opción que hemos podido encontrar en el mercado actual es la API objectBox



ObjectBox es una API de persistencia en BBDD desarrollada para varios lenguajes orientados a objetos (Java, C++, Go, Swift, Kotlin, Python, Dart, EdgeX)

Esta API permite la gestión automática de BBDD a través del sistema de orientación a objetos del lenguaje, es una API muy parecida a Spring JPA.

ObjectBox permite definir una clase POJO de java como una entidad a partir de la anotación @Entity como ejemplo en nuestra app nuestra entidad sería la clase nota



4 Clase POJO de Java implementada como entidad de ObjectBox

Definiendo una clase POJO con @Entity, la API entiende que esa clase es una tabla de BBDD y sus atributos columnas de la tabla, estas columnas deben ser tipos primitivos, aunque ObjectBox también admite algunos tipos objeto como puede ser el tipo Date y los List.

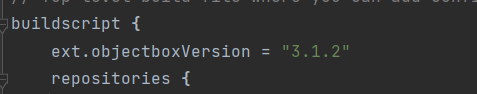
Para el uso de @Entity también son necesarios dos requisitos más: la definición de un @Id, que no es más que la columna Primary Key de la tabla. Por limitaciones del sistema ObjectBox si es completamente necesario que el atributo @Id sea definido de tipo long, o de tipo Long.

El segundo requisito es la necesidad de implementar un constructor sin parámetros para la clase entidad. Es conveniente también inicializar ciertos parámetros en este constructor vacío si no queremos que el sistema gestor de BBDD tenga problemas a la hora de gestionar tipos nulos. Si bien este requisito no es estrictamente obligatorio, es más que deseable pues Java es un lenguaje fuertemente propenso a tener problemas gestionando nulls.

##### Implementación real de la API

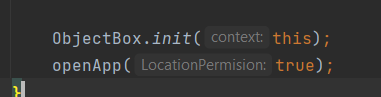
Una vez hemos definido las entidades a persistir necesarias según el alcance del proyecto, debemos inicializar la API y utilizarla en nuestro proyecto de Android Studio.

Para ello el primer paso es implementar las librerías de ObjectBox en el archivo gradle del proyecto:



5 Implementación de Objectbox en graddle

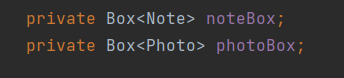
Una vez las librerías de ObjectBox están introducidas en nuestro proyecto, el siguiente paso es inicializar la API. De esta forma, ObjectBox escaneará en busca de entidades definidas en el proyecto y a partir de ellas construirá una BBDD con su correspondiente gestor de base de datos. Todo ello a un nivel por debajo del desarrollador, de forma que este solo debe preocuparse de inicializar el sistema ObjectBox

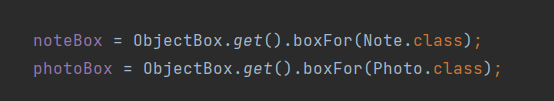


6 Inicialización de ObjectBox

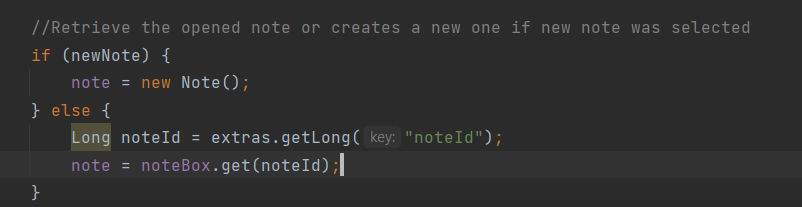
Debemos utilizar el método estático de ObjectBox en la primera Activity de nuestra APP (En nuestro caso el splash screen) Una vez hecho esto, todo el sistema de BBDD está construido y funcionando por debajo en nuestra APP.

Una vez inicializado todo el sistema, debemos utilizarlo. Esto se consigue mediante lo que ObjectBox denomina como Boxes. Un objeto Box<> no es más que la abstracción de todo el sistema de control de una tabla en concreto. A través de este objeto tenemos control total sobre la tabla. Podemos hacer insert, update, delete etc… simplemente invocando a los distintos métodos del Box

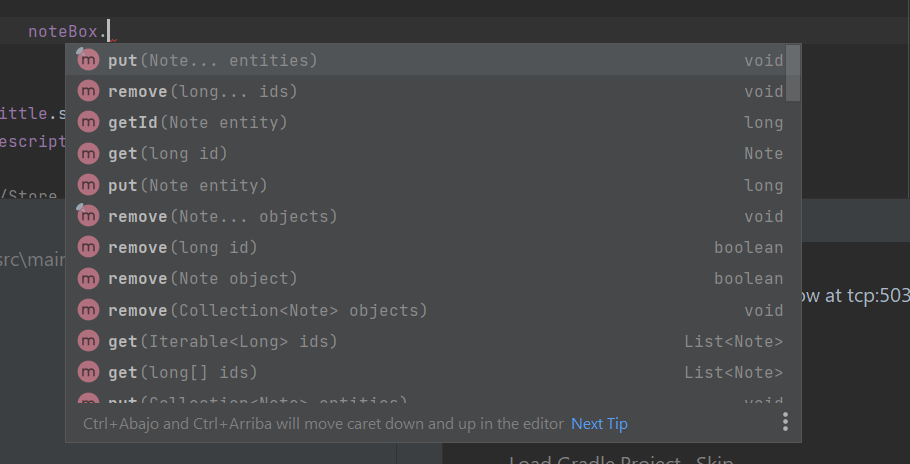




7 Definición e incialización de las Box de notas y fotos



8 Ejemplo de utilización de un box



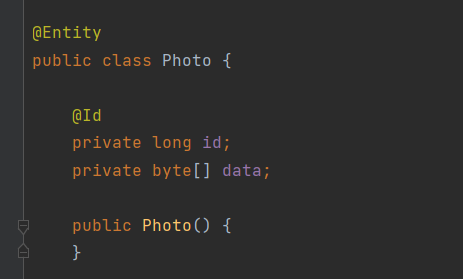
9 Ejemplo de métodos disponibles en un box

Como podemos apreciar, un box tiene infinidad de métodos para interactuar con la base de datos, evitando al usuario de tener que lidiar con complejas sentencias SQL que retrasarían el desarrollo del código.

* Put nos permite guardar una entidad (O lista de entidades) en base de datos. En caso de que el id de esa entidad ya exista en base de datos, la sentencia pasará de ser un Insert a un Update
* Remove nos permite hacer un delete de una entidad o lista de entidades
* getId nos permite conseguir el id a través de una entidad, algo especialmente útil para hacer operaciones con nuestras entidades
* get nos permite recuperar una fila entera de la base de datos y además se nos devuelve en forma del objeto POJO del que hayamos definido la entidad. Algo especialmente útil para trabajar con los datos.

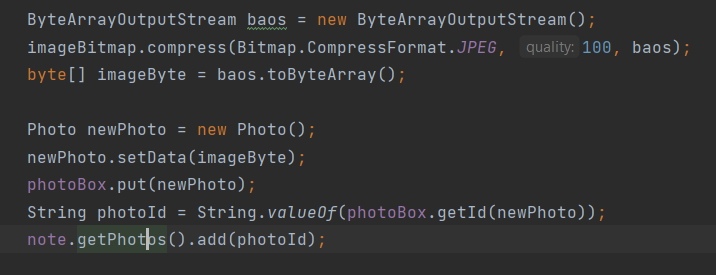
##### Persistencia de fotos

Uno de los problemas que nos hemos encontrado en el sistema de persistencia de la aplicación ha sido la necesidad de guardar fotos en la misma. Android gestiona las fotos habitualmente con el formato bitmap, pero es un formato no serializable y no apto para guardar en base de datos. Por ello al definir una entidad de foto para que ObjectBox la gestione, nos hemos visto obligados a guardar la foto como un array de bytes.



10 Entidad foto

Por ello, antes de guardar la foto en base de datos se debe convertir en un array de bits y asignarle un id para poder recuperarla luego cuando se necesite. Dando lugar a la siguiente implementación:



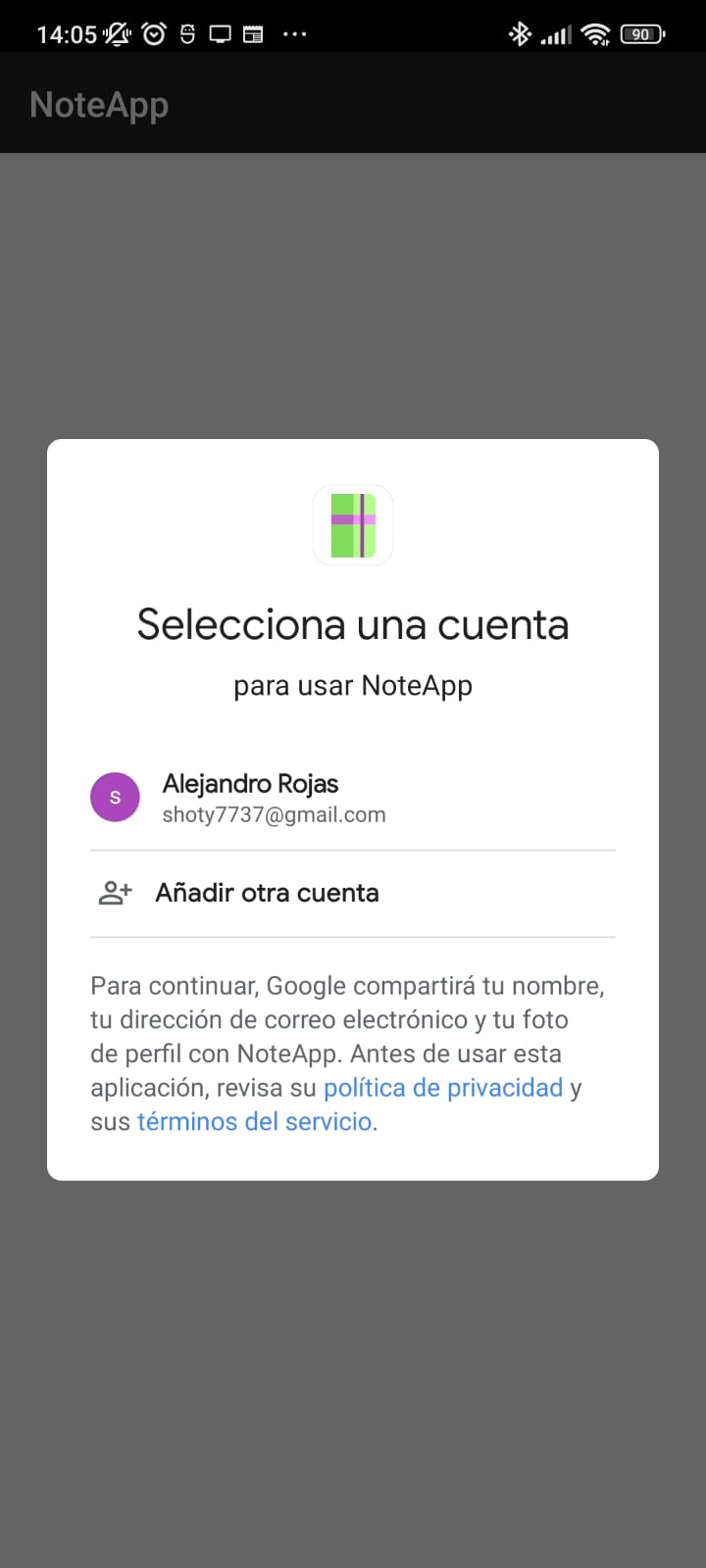
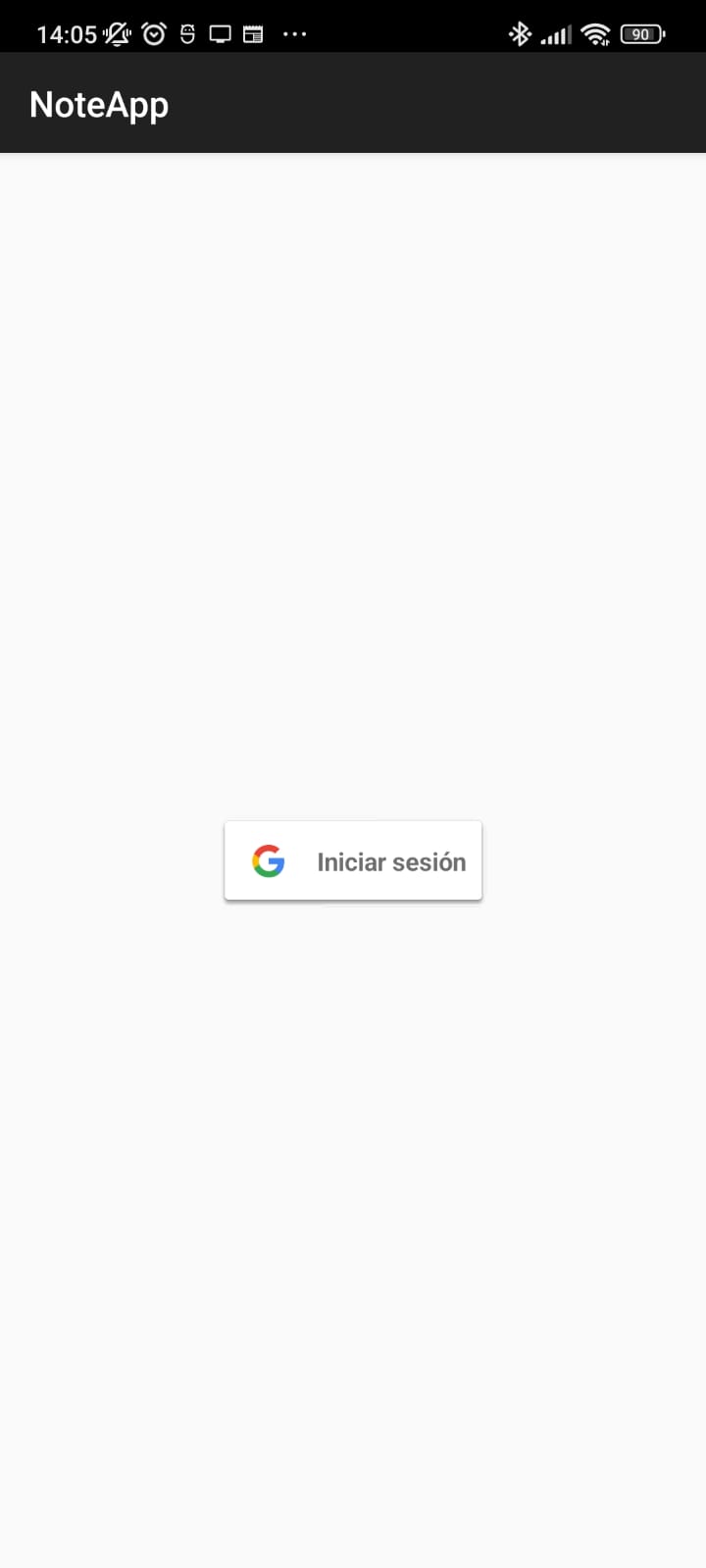
11 Implementación de persistencia de fotos

De esta forma, en primer lugar, se comprime el bitmap, pasándolo a JPEG y luego pasamos la información del JPEG a un array de bits, pudiendo así “serializar” la información y persistirla en BBDD

### Firebase y Google Login

Uno de los módulos mas importantes de la aplicación es el de Google log in. En este apartado el usuario es capaz de hacer un Log in con su cuenta de Google para persistir todos sus datos en la nube. Esto es importante ya que cualquier usuario que use a largo plazo nuestra aplicación querrá estar seguro de que sus datos no se pierden y son capaces de recuperarse en caso de pérdida o inutilización de su terminal móvil. O incluso también espera que en caso de cambio temporal o definitivo de terminal móvil pueda recuperar igualmente estos datos.

Para ello se ha determinado que una de las mejores opciones era el uso de Google Login a través de Firebase. Esto permite al usuario logearse con su cuenta de Google y, una vez hecho esto, de forma automática se guardarán los datos de su agenda en la nube de firebase, manteniendo a salvo todos los datos personales del usuario



Una vez hecho esto, se implementa un botón que el usuario podrá pulsar desde el nuevo terminal o desde la misma APP en caso de perdida de los datos locales para poder recuperar la agenda tal y como se encontraba en su último uso.

1. Corrección de posibles errores

## Fase de pruebas

1. Preparación del entorno de pruebas
2. Ejecución de las pruebas y reporte de los errores encontrados

# DEFINICIÓN DE PROCEDIMIENTOS DE CONTROL Y EVALUACIÓN

* Definir el procedimiento de evaluación de las **incidencias** que puedan presentarse durante la realización de las diferentes actividades.
* Definir el procedimiento para gestionar los posibles **cambios** en los recursos y en las actividades.

# FUENTES

Incluir las páginas web, biografía,,,consultadas.

# ANEXOS

Cualquier añadido se incluirá en este apartado.

Por ejemplo, a continuación se incluyen unas pautas a tener en cuenta a la hora de elaborar la documentación del proyecto.

## Guía de estilo

Obviamente, este anexo deberá ser eliminado del documento final a entregar.

* **Título del proyecto**

Elegir un nombre llamativo y relacionado con la temática que va a tratar.

* **Figuras y tablas**

Cualquier figura, tabla... incluida en el documento deberá tener un título a pie de página..

Incluir tablas, gráfico, mapas conceptuales...que ayuden a leer y comprender el documento.

* **Índices**

Además del índice de contenidos, ya incluido en la plantilla, se añadirá a continuación el índice de figuras, si fuera necesario.

* **Redacción**

Se evitarán las mayúsculas, salvo en los títulos y poco más.

No se emplearán formas personales (instalamos, seleccionamos...) en su lugar se utilizarán formas impersonales ( instalar, se instalará, seleccionar, se selecciona,...).

Se evitará la voz pasiva (casi siempre traducción literal del inglés). En vez de: es desarrollado para cumplir... mejor: se desarrolla para cumplir...

Se evitarán los párrafos largos.

Se utilizarán las viñetas para facilitar la lectura del documento.

* **Formato**

El documento se generará en formato pdf.

* **Entrega**

Todo el material del módulo Proyecto (documentos, ficheros fuentes, herramientas...) se entregará en formato electrónico, en una carpeta comprimida:

CICLO-CURSO-TITULO DEL PROYECTO