



UNIVERSIDAD EUROPEA DE MADRID

ESCUELA DE ARQUITECTURA, INGENIERÍA Y DISEÑO

Grado en Ingeniería Informática

TRABAJO DE FIN DE GRADO

**Aplicación para el asesoramiento en
inversiones y finanzas personales**



Alejandro Hernán Colmenarejo

Dirigido por

Rafael Muñoz Gil

Curso 2019-2020

TÍTULO: Aplicación para el asesoramiento en inversiones y finanzas personales.

AUTOR: Alejandro Hernán Colmenarejo.

TITULACIÓN: Grado en Ingeniería Informática.

DIRECTOR DEL PROYECTO: Rafael Muñoz Gil.

FECHA: 13 de Julio de 2020

RESUMEN

Desde la creación de la primera bolsa de valores, las personas han utilizado su dinero para invertirlo en sectores determinados. Estos junto con la creación de internet ha supuesto que, con el paso del tiempo, crezca de una manera descomunal. Este crecimiento ha sido tal que ya incluso se puede invertir de forma automática.

Este tipo de inversión automática se lleva a cabo por los llamados Roboadvisors, que son asesores automáticos que gestionan de manera automatizada las inversiones de los usuarios.

Normalmente, este tipo de movimientos monetarios los realizan personas con gran capital económico. En este proyecto lo que se trata es de intentar fomentar que las personas con poco capital puedan invertir, ya que lo que se va a recomendar son fondos donde el importe mínimo para poder invertir es asequible a toda la población.

También se plantea desarrollar una aplicación Android basada en los Roboadvisors, donde se recomendará dónde invertir según unos parámetros que el usuario decide. Estos parámetros no tienen por qué ser fijos, se pueden personalizar según le plazca al usuario.

Para que este Roboadvisor pueda recomendar según la personalización que el usuario ha elegido, se van a utilizar algoritmos de Machine Learning para el aprendizaje automático del recomendador. Esto significa que cada vez que un usuario se registre en la aplicación, este Roboadvisor utilizará los usuarios ya registrados, para recomendarle a este nuevo usuario una inversión.

ABSTRACT

Since the creation of the first stock exchange, people have used their money to invest in specific sectors. This fact, together with the creation of the Internet, has caused an enormous increase in this type of investments over the past few years.. This growth has been such that it can even be invested automatically.

This type of automated investment is carried out by the so-called Roboadvisors, which are automated advisors that manage the users' investments in an automated way.

Normally, this type of monetary movement is carried out by people with great economic capital. In this project the aim is to try to encourage people with little capital to invest, since what is going to be recommended are funds where the minimum amount to be able to invest is affordable to the whole population.

It also aims to develop an Android application based on the Roboadvisors, which will recommend where to invest according to some parameters that the user decides. These parameters do not have to be fixed, they can be customized according to the user's needs.

In light of the above information, Roboadvisor has to be able to recommend according to the customization that the user has chosen, Machine Learning algorithms will be used for automatic learning of the recommender. This means that every time a user registers in the application, this Roboadvisor will use the already registered users, to recommend this new user an investment.

AGRADECIMIENTOS

Por fin, después de 6 años ya se ve la luz al final del camino, un camino lleno curvas y baches. Este camino si lo hubiera hecho solo, no lo hubiera acabado nunca, es por eso por lo que es fundamental saber de quién te rodeas y saber agradecerse, como pueden ser amigos, compañeros e incluso una disciplina deportiva como las artes marciales mixtas.

Todo esto me ha ayudado a desconectar cuando mi cabeza no aguantaba más, pero, sobre todo, doy gracias a mis padres por insistir en guiarme cuando me veían perdido y descentrado.

Gracias a ellos mi camino tomó otro rumbo, y es que a la experiencia no hay que negarle la razón, aunque nos cuente. Este rumbo fue esta universidad, la cual me acogió con los brazos abiertos.

Yo siempre he sido una persona muy negativa, pero el poder realizar el deporte de artes marciales mixtas me ha enseñado que no hay que temer fallar, sino aprender sobre ello y no volver a fallar en lo mismo más veces.

Por ultimo y no menos importante, doy las gracias a mi tutor Rafael Muñoz Gil por guiarme a realizar este proyecto y aclararme las ideas para yo poder encauzar este final de un camino que cierra su etapa para dejar paso a otros.

TABLA RESUMEN

| | DATOS |
|--|--|
| Nombre y apellidos: | Alejandro Hernán Colmenarejo |
| Título del proyecto: | Aplicación para el asesoramiento en inversiones y finanzas personales |
| Directores del proyecto: | Rafael Muñoz Gil |
| El proyecto se ha realizado en colaboración de una empresa o a petición de una empresa: | NO |
| El proyecto ha implementado un producto: (esta entrada se puede marcar junto a la siguiente) | SI |
| El proyecto ha consistido en el desarrollo de una investigación o innovación: (esta entrada se puede marcar junto a la anterior) | NO |
| Objetivo general del proyecto | Realizar una aplicación que facilite que los usuarios puedan invertir su dinero en fondos de inversión, los cuales podrán elegirse automáticamente o de forma personalizada. |

Índice

| | |
|--|----|
| RESUMEN..... | 3 |
| ABSTRACT | 4 |
| TABLA RESUMEN | 6 |
| Capítulo 1. RESUMEN EL PROYECTO | 12 |
| 1.1 Contexto y justificación | 12 |
| 1.2 Planteamiento del problema | 12 |
| 1.3 Objetivos del proyecto | 12 |
| 1.4 Resultados obtenidos | 12 |
| 1.5 Estructura de la memoria..... | 12 |
| Capítulo 2. ANTECEDENTES / ESTADO DEL ARTE | 13 |
| 2.1 Contexto y justificación..... | 13 |
| 2.2 Planteamiento del problema | 14 |
| Capítulo 3. OBJETIVOS | 15 |
| 3.1 Objetivo global..... | 15 |
| 3.2 Objetivos concretos | 15 |
| 3.3 Objetivos fuera de alcance | 15 |
| 3.4 Beneficios para el cliente | 16 |
| Capítulo 4. DESARROLLO DEL PROYECTO | 17 |
| 4.1 Análisis de requisitos | 17 |
| 4.1.1 Requisitos funcionales..... | 17 |
| 4.1.2 Requisitos no funcionales..... | 18 |
| 4.1.3 Tabla de requisitos | 19 |
| 4.2 Arquitectura de la aplicación..... | 21 |
| 4.2.1 Android | 21 |
| 4.2.2 Firebase | 22 |
| 4.2.3 Machine Learning | 23 |
| 4.3 Diseño | 25 |
| 4.3.1 Diagrama de casos de uso | 25 |
| 4.3.2 Mockups..... | 26 |
| 4.3.3 Diagrama de clases | 32 |

| | | |
|--------------|---|----|
| 4.3.4 | Diagrama de la arquitectura de la aplicación | 33 |
| 4.3.5 | Recursos requeridos..... | 33 |
| 4.4 | Resultados del proyecto | 34 |
| 4.4.1 | Código en Python – Entrenamiento | 36 |
| 4.4.2 | Código en Python – Clasificación | 36 |
| 4.4.3 | Código en Java – Interpretación del código Python | 37 |
| Capítulo 5. | PRESUPUESTO | 40 |
| Capítulo 6. | VIABILIDAD Y PLAN DE RECURSOS | 41 |
| 6.1 | Estudio de viabilidad..... | 41 |
| 6.2 | Estudio de viabilidad económica..... | 41 |
| 6.3 | Plan de recursos..... | 41 |
| Capítulo 7. | CONCLUSIONES..... | 42 |
| Capítulo 8. | FUTURAS LINEAS DE TRABAJO | 43 |
| Capítulo 9. | BIBLIOGRAFIA..... | 44 |
| Capítulo 10. | ANEXO I | 49 |
| 10.1 | Manual de usuario | 49 |
| 10.1.1 | Pantalla de inicio | 49 |
| 10.1.2 | Pantalla aplicación..... | 52 |
| 10.2 | Manual de instalación | 55 |
| 10.2.1 | Android Studio | 55 |
| 10.2.2 | Firebase Realtime Database | 55 |
| Capítulo 11. | ANEXO II | 56 |

Índice de Figuras

| | |
|---|----|
| Ilustración 1 – Gráfico Android | 21 |
| Ilustración 2 - Android | 22 |
| Ilustración 3 -Distancia euclidiana | 23 |
| Ilustración 4 - Corchete de Iverson | 24 |
| Ilustración 5 - Casos de uso..... | 25 |
| Ilustración 6 - MockUp | 26 |
| Ilustración 7 - MockUp | 27 |
| Ilustración 8 - MockUp | 27 |
| Ilustración 9 - MockUp | 28 |
| Ilustración 10 - MockUp | 28 |
| Ilustración 11 - MockUp | 29 |
| Ilustración 12 - MockUp | 29 |
| Ilustración 13 - MockUp | 30 |
| Ilustración 14 - MockUp | 30 |
| Ilustración 15 - MockUp | 31 |
| Ilustración 16 - MockUp | 31 |
| Ilustración 17 - Diagrama de clases | 32 |
| Ilustración 18 – Arquitectura | 33 |
| Ilustración 19 - Chaquopy | 34 |
| Ilustración 20 – Chaquopy | 35 |
| Ilustración 21 - Chaquopy | 35 |
| Ilustración 22 - Creación modelo | 36 |
| Ilustración 23 - Clasificación | 37 |
| Ilustración 24 - Instancia Python | 37 |
| Ilustración 25 - BBDD | 38 |
| Ilustración 26 – Graficas y carteras..... | 39 |
| Ilustración 27 - Anexo Manual Usuario | 49 |
| Ilustración 28 - Anexo Manual Usuario | 49 |
| Ilustración 29 - Anexo Manual Usuario | 50 |
| Ilustración 30 - Anexo Manual Usuario | 50 |

| | |
|---|----|
| Ilustración 31 - Anexo Manual Usuario | 51 |
| Ilustración 32 - Anexo Manual Usuario | 51 |
| Ilustración 33 - Anexo Manual Usuario | 51 |
| Ilustración 34 - Anexo Manual Usuario | 51 |
| Ilustración 35- Anexo Manual Usuario | 52 |
| Ilustración 36 - Anexo Manual Usuario | 53 |
| Ilustración 37 - Anexo Manual Usuario | 53 |
| Ilustración 38 - Anexo Manual Usuario | 53 |
| Ilustración 39 - Anexo Manual Usuario | 54 |
| Ilustración 40 - Anexo Manual Usuario | 54 |
| Ilustración 41 - Anexo Manual Usuario | 54 |
| Ilustración 42 - Anexo Manual Usuario | 54 |
| Ilustración 43 - Anexo Diagrama de clases | 57 |

Índice de Tablas

| | |
|----------------------------|----|
| Tabla 1- Requisitos..... | 20 |
| Tabla 2 - Presupuesto..... | 40 |

Capítulo 1. RESUMEN EL PROYECTO

1.1 Contexto y justificación

Desde que se creó la primera bolsa de valores, las personas han ido invirtiendo su dinero para poder obtener beneficios, pero también hay que decir que existe las pérdidas.

Este proyecto se ha creado para que el usuario con un menor capital pueda realizar dicha actividad sin problema ni conocimiento alguno.

1.2 Planteamiento del problema

El tema de inversiones es principalmente para personas / empresas con mucho capital. En este proyecto lo que se propone es conseguir una aplicación que sirva para usuarios con un menor capital y que quieran poder invertir su dinero.

1.3 Objetivos del proyecto

Realizar una aplicación que facilite que los usuarios puedan invertir su dinero en fondos de inversión, los cuales podrán elegirse automáticamente o de forma personalizada.

1.4 Resultados obtenidos

Recomendación de un fondo de inversión según unos parámetros introducidos por el usuario.

1.5 Estructura de la memoria

Capítulo 1: resumen del proyecto

Capítulo 2: se contextualiza y se pone en situación histórica.

Capítulo 3: se muestran los objetivos que se pretenden en el proyecto

Capítulo 4: explicación detallada del desarrollo del proyecto

Capítulo 5: presupuesto

Capítulo 6: explicación de la viabilidad y plan de recursos del proyecto

Capítulo 7: se exponen unas conclusiones

Capítulo 8: se muestran las futuras líneas de trabajo

Capítulo 9: recogida de información y referencias

Capítulo 10: anexos

Capítulo 2. ANTECEDENTES / ESTADO DEL ARTE

2.1 Contexto y justificación.

En 1460 en Amberes, Bélgica, se creó la primera bolsa de valores la cual continúa hasta la actualidad. El proceso de modernización desde entonces ha sido enorme y con una gran expansión por todo el mundo.

Tiene tal peso en la economía de la sociedad que puede provocar crisis económicas y también puede hacer que se gane mucho dinero. Es por ello por lo que las personas buscan invertir su dinero en acciones, fondos, criptomonedas, etc.

El Deutsche Bank reconoce que los RoboAdvisors suponen un desafío para la industria de gestión de activos clásica. Cree que para el inversor la mejor opción es un híbrido, una combinación del asesoramiento automatizado y el asesoramiento tradicional.

La diferencia entre los RoboAdvisors y los bancos tradicionales es muy amplia. Por ejemplo, en Estados Unidos los asesores financieros cobran un 1% de comisión por gestionar una cartera de hasta 100.000 \$. Sin embargo, los RoboAdvisors cobran entre un 0,15% y 0,67%, incluso con servicios gratuitos, para invertir hasta 10.000 \$.

La figura de los RoboAdvisors ayuda a atraer al mundo de las inversiones a las familias con menos formación financiera.

Un ejemplo de RoboAdvisors es “Popcoin” de Bankinter, que con poco más de dos años de existencia, ha superado los 10 millones de euros bajo gestión en carteras de inversión y planes de pensiones, creciendo en los últimos 6 meses un 90%.

En contraposición, el servicio similar que ofrece la banca privada tradicional tiene un importe mínimo de inversión de 350.000€, mientras que “Popcoin” admite inversiones a partir de sólo 1.000€.

2.2 Planteamiento del problema

Lo que este proyecto pretende realizar es ir acercando al mundo de las inversiones a todo aquel que no tenga un conocimiento pobre o casi nulo mediante el uso de los Roboadvisors.

Este proyecto se va a centrar en que cualquier persona, dando igual el capital que posea, pueda acceder al mundo de las inversiones, incluso si tener conocimiento alguno.

Este es el verdadero motivo por el que se quiere realizar esta aplicación. Lo ideal es que se facilite el movimiento de dinero personal, capital, entre varios activos, fondos de inversión, de manera automática o incluso personalizable. De esta manera la persona inversora puede ver como su dinero fluctúa. En función de varios factores como el objetivo y el riesgo, se tiene que tener en cuenta la habilidad individual del usuario, que será lo que última instancia definirá el éxito o fracaso de la inversión.

Se podría añadir que este proyecto ofrecerá un recomendador de fondos de inversiones para personas con un capital indiferente. De esta manera no es necesaria la opción de contratar un gestor financiero, que, si bien es útil, con las cantidades tan bajas que se van a recomendar invertir, no sería tan necesario.

Capítulo 3. OBJETIVOS

3.1 Objetivo global

1. El objetivo del proyecto es crear un asesor financiero, RoboAdvisor, personalizable, que lo puedan usar tanto personas que estén al día en el mundo de las inversiones como personas que no tengan conocimiento alguno de este mundo.

3.2 Objetivos concretos

1. Desarrollar el almacenamiento de los datos en una base de datos en tiempo real en Firebase.
2. Creación de un test para determinar el perfil inversor.
3. Creación de carteras de inversión según el perfil inversor definido.
4. Recomendar fondos de inversión al cliente, según el tipo de cartera que se ha creado.
5. Los únicos tipos de fondos de inversión que se van a utilizar son los fondos indexados. Los demás fondos no se contemplan tratar.

3.3 Objetivos fuera de alcance

1. Implementación de fondos gestionados, planes de pensiones u otros que no sean los fondos indexados.
2. No se plantea realizar un cobro de las comisiones para sacar rentabilidad a la aplicación.
3. No se plantea tratar riesgo de divisa.
4. Modificación de carteras una vez creadas.

3.4 Beneficios para el cliente

Con este RoboAdvisors ahora podrán invertir no solo personas que tenga mucho capital, sino también individuos con una situación económica más modesta. Lo que se quiere decir con esto, es que ahora el pequeño inversor cada vez está siendo más importante. Con esta aplicación se puede invertir en grandes abanicos de fondos, de esta manera, si no se tiene conocimiento sobre lo que se invierte, gracias a las estadísticas de la aplicación, el cliente puede despreocuparse de estar pendiente de sí pierde dinero o de cuando sacarlo, basta con fijar una cifra y la aplicación realizará dichas operaciones automáticamente.

Gracias a que están automatizadas las inversiones, no se necesita un asesor financiero real, por lo que el coste de las comisiones es bastante menor para el cliente potencial.

Por último, esta aplicación tampoco sustituye el papel de los asesores financieros tradicionales, sino que lo complementa, como se ha mencionado anteriormente, ayudará a atraer a clientes con menor formación financiera al mundo de la inversión.

Capítulo 4. DESARROLLO DEL PROYECTO

4.1 Análisis de requisitos

4.1.1 Requisitos funcionales

4.1.1.1 *Gestión de usuarios*

1. Registro de usuarios: Relleno de datos de un formulario, con su correspondiente validación y verificación de cuenta.
2. Identificación de los usuarios: Introducción de los datos correspondientes y verificación de que sean correctos
3. Cerrar la sesión: El usuario cierra sesión y se muestra la pantalla principal.
4. Cuenta del usuario: El usuario podrá ver sus datos guardados en la aplicación
5. Carteras el usuario: Según el resultado del test podrá ver unos fondos u otros donde se ha invertido su dinero
6. Dar de baja al usuario: El usuario se puede dar de baja y se borrarán sus respectivos datos.

4.1.1.2 *Aplicación*

1. Pantalla principal: Donde se podrá registrar, iniciar sesión y recuperar la contraseña.
2. Pantalla del test: El usuario realizará el test para determinar su perfil inversor y una cartera asociada a ello.
3. Pantalla usuario: Aquí se podrá ver, crear, editar y borrar carteras.
4. Notificaciones: Se podrán activar o desactivar las notificaciones

4.1.2 Requisitos no funcionales

4.1.2.1 Documentación

1. Documentación del código: En caso de que haya que modificar el código, todo será más entendible y fácil de localizar.
2. Manual de usuario: Con este manual el usuario podrá guiarse por la aplicación.

4.1.2.2 Seguridad

1. Autenticación por defecto: El usuario podrá autenticarse mediante email y contraseña.
2. Autenticación por Google: El usuario podrá autenticarse mediante su cuenta de Google.
3. Cifrado de los datos: Los datos personales serán cifrados mediante Firebase.

4.1.2.3 Compatibilidad

1. Aplicación Android: Solo estará disponible para versiones de Android 5.0 y superiores.
2. Conexión a internet: Se necesitará tener todo el tiempo una conexión de datos móviles o WIFI.

4.1.2.4 Rendimiento

1. Base de datos Firebase: La base de datos se realizará en Firebase y será en tiempo real.

4.1.2.5 Interfaz

1. Sencillez: Se creará una interfaz sencilla que atraiga al usuario y sea intuitiva.

4.1.3 Tabla de requisitos

| ID | Nombre | Categoría | Descripción |
|-------------|--------------------------------|---------------------|--|
| ReqFun_01 | Registro usuarios | Gestión de usuarios | Relleno de datos de un formulario, con su correspondiente validación y verificación de cuenta. |
| ReqFun_02 | Identificación de los usuarios | Gestión de usuarios | Introducción de los datos correspondientes y verificación de que sean correctos. |
| ReqFun_03 | Cerrar sesión | Gestión de usuarios | El usuario cierra sesión y se muestra la pantalla principal. |
| ReqFun_04 | Cuenta de usuario | Gestión de usuarios | El usuario podrá ver sus datos guardados en la aplicación. |
| ReqFun_05 | Carteras usuario | Gestión de usuarios | Según el resultado del test podrá ver unos fondos u otros donde se ha invertido su dinero. |
| ReqFun_06 | Dar de baja usuario | Gestión de usuarios | El usuario se puede dar de baja y se borrarán sus respectivos datos. |
| ReqFun_07 | Pantalla principal | Aplicación | Donde se podrá registrar, iniciar sesión y recuperar la contraseña. |
| ReqFun_08 | Pantalla test | Aplicación | El usuario realizará el test para determinar su perfil inversor y una cartera asociada a ello. |
| ReqFun_09 | Pantalla usuario | Aplicación | Aquí se podrá ver, crear, editar y borrar carteras. |
| ReqFun_10 | Notificaciones | Aplicación | Se podrán activar o desactivar las notificaciones. |
| ReqNoFun_01 | Documentación del código | Documentación | En caso de que haya que modificar el código, todo será más entendible y fácil de localizar. |
| ReqNoFun_02 | Manual de usuario | Documentación | Con este manual el usuario podrá guiarse por la aplicación. |
| ReqNoFun_03 | Autenticación por defecto | Seguridad | El usuario podrá autenticarse mediante email y contraseña. |

| | | | |
|-------------|--------------------------|----------------|--|
| ReqNoFun_04 | Autenticación por Google | Seguridad | El usuario podrá autenticarse mediante su cuenta de Google. |
| ReqNoFun_05 | Cifrado de datos | Seguridad | Los datos personales serán cifrados mediante Firebase. |
| ReqNoFun_06 | Aplicación Android | Compatibilidad | Solo estará disponible para versiones de Android 5.0 y superiores. |
| ReqNoFun_07 | Conexión a internet | Compatibilidad | Se necesitará tener todo el tiempo una conexión de datos móviles o WIFI. |
| ReqNoFun_08 | Firebase Database | Rendimiento | La base de datos se realizará en Firebase y será en tiempo real. |
| ReqNoFun_09 | Interfaz sencilla | Interfaz | Se creará una interfaz sencilla que atraiga al usuario y sea intuitiva. |

Tabla 1- Requisitos

4.2 Arquitectura de la aplicación

Para realizar esta aplicación se han utilizado distintas tecnologías como Android, Firebase y Machine Learning, las cuales serán explicadas a continuación.

4.2.1 Android

Se ha decidido usar esta tecnología porque es la que predomina en España como se puede observar en el gráfico:

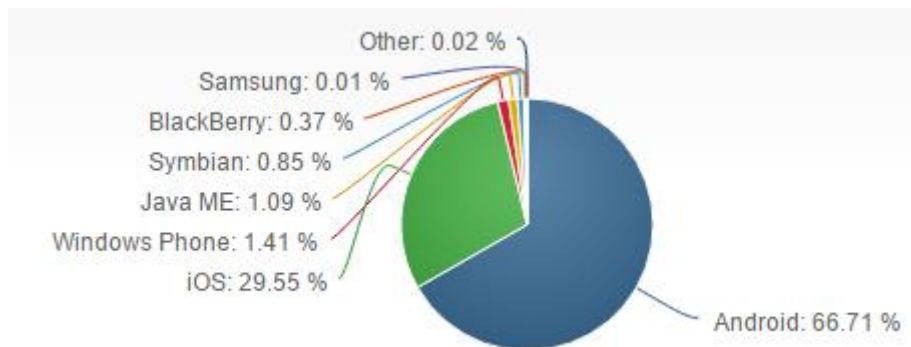


Ilustración 1 – Gráfico Android

Es un sistema operativo de código abierto desarrollado por Google y basado en el kernel de Linux.

Desde que surgió, 2005, han ido saliendo nuevas versiones. Este proyecto hará compatible la aplicación para smartphones que tenga una versión mayor o igual a la 5.0. Esto se debe a la cuota de mercado de las versiones.

Cuota de mercado de las versiones Android más utilizadas en 2018.

- Android Lollipop 5.x: 17.9%
- Android Marshmallow 6.x: 21.3%
- Android Nougat 7.x: 28.2%
- Android Oreo 8.x: 21.5%
- Android Pie 9.x: 10.4%

| | | | |
|--------------------------|-------------|-------------------------|---------|
| Lollipop | 5.0 – 5.1.1 | 12 de noviembre de 2014 | 21 – 22 |
| Marshmallow | 6.0 – 6.0.1 | 5 de octubre de 2015 | 23 |
| Nougat | 7.0 – 7.1.2 | 15 de junio de 2016 | 24 – 25 |
| Oreo | 8.0 – 8.1 | 21 de agosto de 2017 | 26 – 27 |
| Pie | 9.0 | 6 de agosto de 2018 | 28 |
| Android 10 ⁵⁷ | 10.0 | 3 de septiembre de 2019 | 29 |




| | |
|---|------------------------------|
|  | Última versión |
|  | Versión antigua pero vigente |
|  | Versión descontinuada |

Ilustración 2 - Android

Una aplicación en Android se puede hacer tanto en Kotlin como en Java. Debido al aprendizaje realizado durante estos años en este último, se ha decidido su aplicación a un caso más real como es la realización de este proyecto. Para realizar dicha aplicación Android en Java se ha utilizado el programa Android Studio, ya que este facilita un emulador en caso de que no se pueda conectar el móvil al ordenador para ejecutar las pruebas.

4.2.2 Firebase

Es una plataforma desarrollada por Google que sirve para desarrollar aplicaciones web y móviles.

De este servicio de Firebase se va a utilizar Firebase Authentication y RealtimeDatabase.

Firebase Authentication proporciona servicios de backend, SDK fáciles de usar y bibliotecas de IU ya elaboradas para autenticar a los usuarios en su app. También admite la autenticación mediante contraseñas, números de teléfono, Google, Facebook y Twitter, etc.

Firebase RealtimeDatabase es una base de datos alojada en la nube. Los datos se almacenan en formato JSON y se sincronizan en tiempo real con cada cliente conectado. Cuando se compilan apps multiplataforma como en nuestro caso es Android, todos los clientes comparten una instancia de RealtimeDatabase y reciben actualizaciones automáticamente con los datos más recientes.

Gracias a la API de Firebase, se pueden enlazar las funciones de Firebase Authentication y Firebase RealtimeDatabase con Android Studio.

4.2.3 Machine Learning

Clasificación supervisada: a partir de un conjunto de datos inicial, el objetivo es el de clasificar todas las instancias nuevas de forma correcta. Este conjunto de datos está formado por varios atributos descriptivos y un atributo objetivo. En este caso los atributos descriptivos serían la edad, ingresos y puntuación. El atributo objetivo sería un identificador del fondo de inversión.

El algoritmo utilizado para la recomendación de fondos de inversión en este proyecto es el K-NN, K-Nearest-Neighbours. Este algoritmo no genera un modelo propio después del entrenamiento, sino que aprende en el momento en que se prueban los datos, por lo que entra en el grupo de “*Lazy Learning methods*”.

La diferencia con el K-Means, que era el algoritmo que se iba a utilizar al principio, es que en K-NN la “K” significa la cantidad de puntos vecinos que se tiene en cuenta para clasificar los “N” grupos y en el K-Means, la “K” significa la cantidad de grupos, clústeres que se quieren clasificar.

El funcionamiento de este algoritmo es el siguiente:

- 1- Calcula la distancia entre los ítems que se van a clasificar y el resto de los ítems del dataset del entrenamiento.
- 2- Selecciona los “K” elementos más cercanos.
- 3- Realizar una “votación de mayoría” entre los k puntos: los de una clase/etiqueta que “dominen” decidirán su clasificación final.

El valor que se tiene que dar a “K” es muy importante en el apartado 3, ya que terminará por definir que grupos pertenecen a los puntos llamados “fronteras”. Cuantos mas puntos tenga “K”, mas tardará en procesar y responder el algoritmo.

La distancia entre puntos se mide mediante la distancia Euclidiana, cuya formula es:

$$d(x_i, x_j) = \sqrt{\sum_{r=1}^p (x_{ri} - x_{rj})^2}$$

Ilustración 3 -Distancia euclidiana

La fase de entrenamiento del algoritmo consiste en almacenar los vectores característicos y las etiquetas de las clases de los ejemplos de entrenamiento. El algoritmo de entrenamiento es:

“Para cada ejemplo $\{x, f(x)\}$, donde $x \in X$, agregar el ejemplo a la estructura representando los ejemplos de aprendizaje”

En la fase de clasificación, la evaluación del ejemplo es representada por un vector en el espacio característico. Se calcula la distancia entre los vectores almacenados y el nuevo vector y se selecciona los “K” mas cercanos. Este nuevo ejemplo es clasificado con la clase que mas se repite en los vectores seleccionados. El algoritmo de clasificación es el siguiente:

“Dado un ejemplar x_q que debe ser clasificado, sean x_1, \dots, x_k los k vecinos más cercanos a x_q en los ejemplos de aprendizaje:

$$\hat{f}(x) \leftarrow \arg \max_{v \in V} \sum_{i=1}^k [v = f(x_i)]$$

Ilustración 4 - Corchete de Iverson

Donde se usa la notación de corchete de Iverson.

El valor $\hat{f}(x_q)$ devuelto por el algoritmo como un estimador de $f(x_q)$ es solo el valor más común de f entre los k vecinos más cercanos a x_q .”

Cuando el algoritmo usa todos los atributos, es cuando mejor resultados otorga, el problema es que puede haber demasiados atributos irrelevantes que dominen sobre la clasificación.

Este sesgo es posible corregirlo asignando peso a las distintas distancias que tiene cada atributo, así se hace a los atributos más relevantes. La otra opción

4.3 Diseño

4.3.1 Diagrama de casos de uso

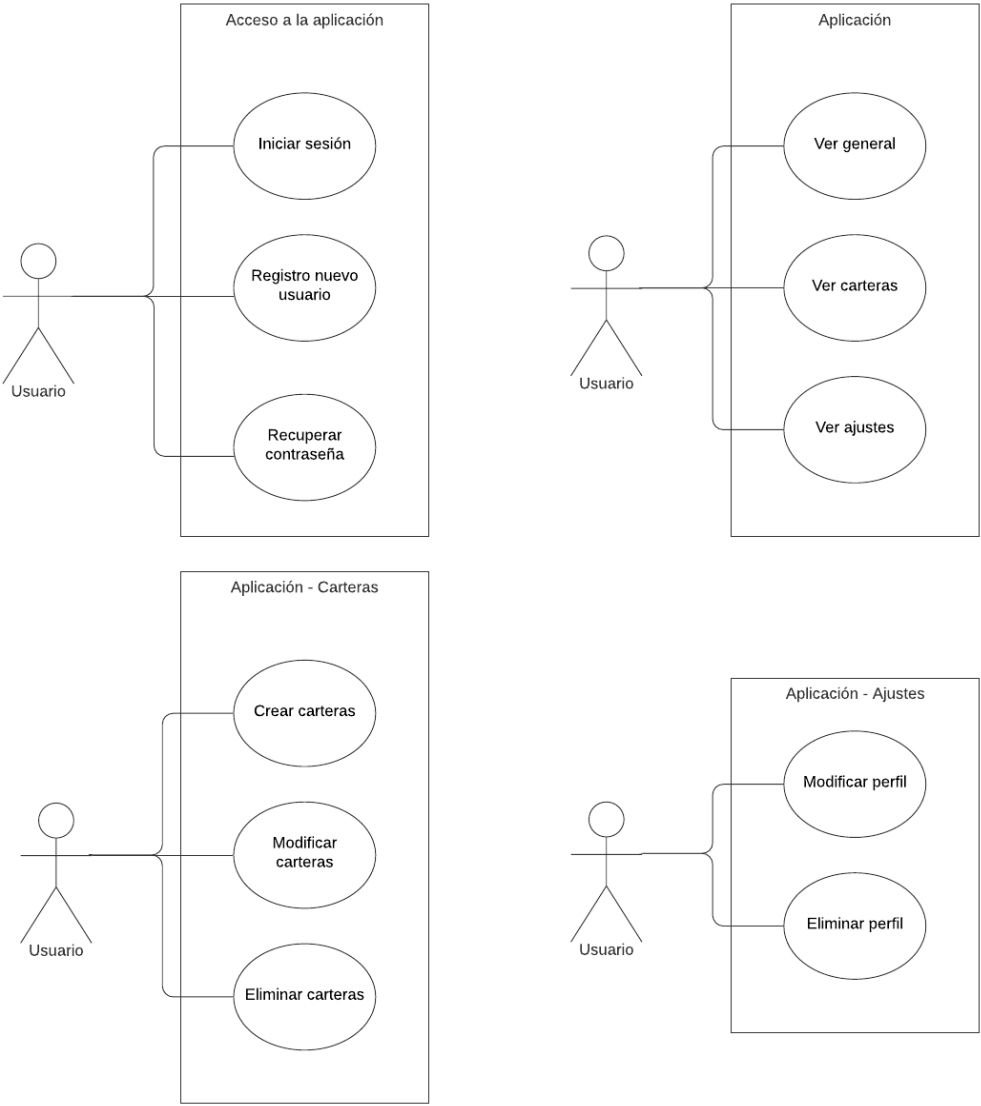


Ilustración 5 - Casos de uso

4.3.2 Mockups

Para la realización de los mockups de la aplicación se ha utilizado el software Ninja Mockup, el cual tiene una versión gratuita. Una de las mejores características de esta herramienta, es que te permite realizar una demo con los mockups creados, aunque obviamente, está demo no tiene funcionalidad alguna, sino que lo que hace es ir viajando entre las diferentes pantallas que se han creado. Para viajar entre las ventanas, se realiza un enlace entre ellas en los botones.

A continuación, se muestra un esquema de cómo están estructuradas las pantallas.

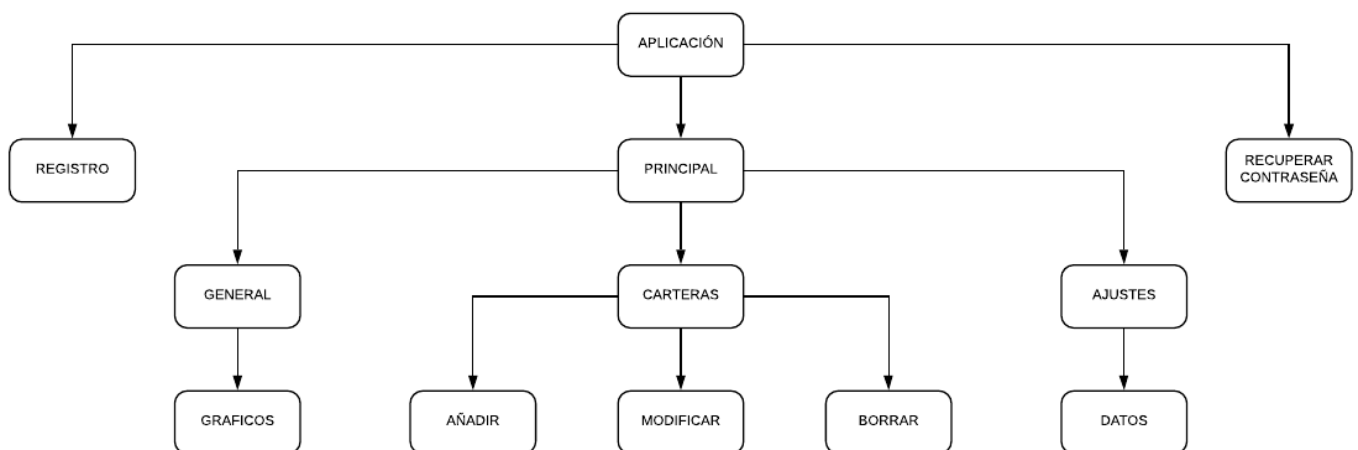


Ilustración 6 - MockUp

La pantalla principal, según se abre la aplicación que se muestra es el inicio de sesión, desde donde se podrá:

1. Iniciar sesión.
2. Registrarse.
3. Recuperar la contraseña.



Ilustración 7 - MockUp

La pantalla “Principal” es la que se ve cuando se inicia sesión en la aplicación y es la encargada de mostrar:

1. Una vista general de tu situación
2. Una vista de las diferentes carteras que tiene el usuario
3. Una vista de los ajustes. Estas tres pantallas serán explicadas más adelante.

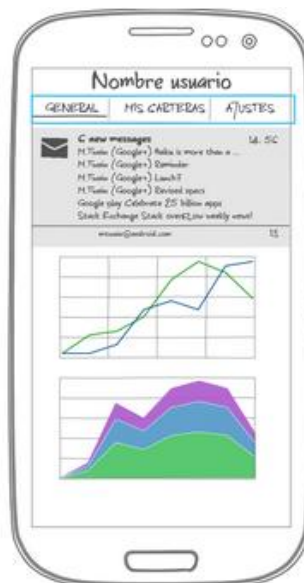


Ilustración 8 - MockUp

En el apartado de “Mis Carteras” se podrá ver un listado de todas las carteras creadas.

En cada cartera se puede:

- Ver los detalles
- Modificar la cartera.
- Borrar la cartera.

Además, se puede añadir carteras, pulsando el botón de añadir.

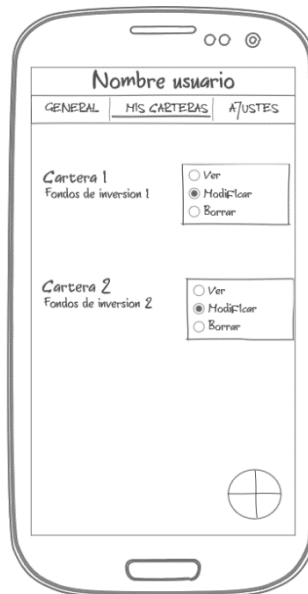


Ilustración 9 - MockUp

Ver detalles

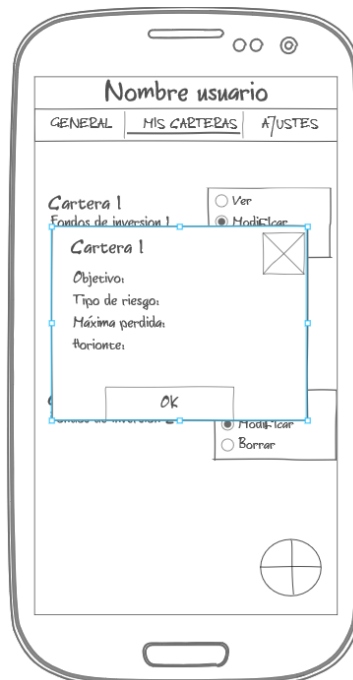


Ilustración 10 - MockUp

Modificar cartera

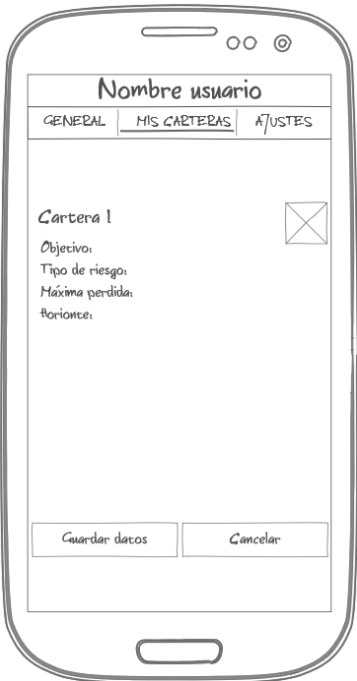


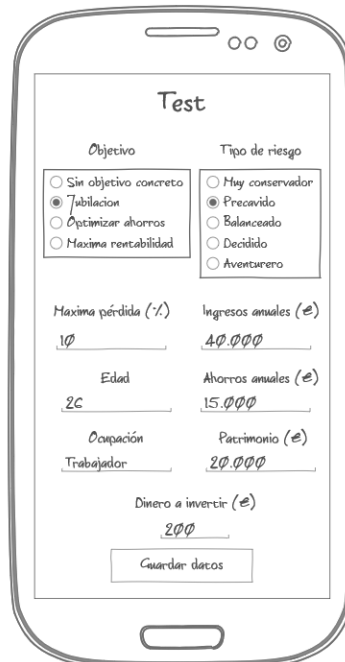
Ilustración 11 - MockUp

Borrar cartera



Ilustración 12 - MockUp

Al pulsar el botón añadir, redirige a la ventana para realizar un test y poder determinar el perfil inversor de la cartera que se va a crear.



Test

Objetivo

☐ Sin objetivo concreto
☒ Jubilación
☐ Optimizar ahorros
☐ Máxima rentabilidad

Tipo de riesgo

☐ Muy conservador
☒ Precautivo
☐ Balanceado
☐ Decidido
☐ Aventurero

Máxima pérdida (%)

Ingresos anuales (€)

Edad

Ahorros anuales (€)

Ocupación

Patrimonio (€)

Dinero a invertir (€)

Guardar datos

Ilustración 13 - MockUp

Como último apartado en la vista “Principal” está la vista de “Ajustes”. En esta pantalla se podrán ver los datos del usuario, lo cuales son editables y si se pulsa el botón “Modificar datos” se guardarán dichas variaciones. Por otro lado, está el botón “Realizar test”, el cuál llevará a una pantalla para realizar un test y poder crear una cartera según el perfil inversor que haya resultado de las contestaciones dadas por el usuario en el test. Los otros dos botones son “Cerrar sesión”, donde el usuario se desloguea de la aplicación y “Borrar cuenta”, donde se borran los datos del usuario del registro en la base de datos.



AJUSTES

DATOS

Ilustración 14 - MockUp

Por otro lado, tenemos la pantalla de Registro en la que el usuario introducirá los datos personales necesarios para poder registrarse.



A hand-drawn mockup of a smartphone screen. At the top, the status bar shows signal, battery, and time icons. The screen displays a registration form titled "DATOS". The form contains ten input fields, each with a label above it: "DNI", "NOMBRE", "APELLIDOS", "PAIS", "CIUDAD", "DOMICILIO", "TELEFONO", "IBAN", "EMAIL", and "CONTRASEÑA". Below the fields is a button labeled "Registrarse". The phone's home button is visible at the bottom.

Ilustración 15 - MockUp

Por último, tenemos la pantalla para Recuperar la contraseña, donde un usuario que no recuerde la contraseña, podrá recuperarla introduciendo el correo con el que se registró en la aplicación y de esta manera recibir un email en su bandeja de entrada para restablecerla.



A hand-drawn mockup of a smartphone screen. At the top, the status bar shows signal, battery, and time icons. The screen displays a password recovery form. At the top of the form is a placeholder for an email address, represented by a rectangle with an 'X' inside. Below this is the text "Introduce tu email con el que estás registrado para recibir la contraseña". Underneath is an input field labeled "Email". At the bottom of the form is a button labeled "Enviar". The phone's home button is visible at the bottom.

Ilustración 16 - MockUp

[illegible]

Ilustración 17 - Diagrama de clases

Para verlo mejor ir al apartado de Anexo II.

Explicación conceptual del diagrama de clases

- *MainActivity*: clase para mostrar la pantalla inicial.
- *PrincipalActivity*: clase para mostrar los Fragmentos después de una validación de datos.
- *RegistroActivity*: clase para el relleno del formulario de registro.
- *RecuperarContraseñaActivity*: clase para la recuperación de la contraseña.
- *GeneralFragment*: clase para la visualización de los datos de las carteras creadas.
- *CarteraFragment*: clase para la visualización de las carteras creadas.
- *AjustesFragment*: clase para la visualización y tratado de datos del usuario.
- *CrearCarteraActivity*: clase para la creación de carteras.
- *Usuarios*: clase con los atributos de los usuarios.
- *Fondos*: clase con los atributos de los fondos.
- *Carteras*: clase con los atributos de las carteras.
- *Validaciones*: clase con las validaciones de datos como DNI, teléfono o IBAN.
- *Autenticaciones*: clase que comprueba que el inicio de sesión es correcto.
- *Validaciones*: clase que relaciona los CardViews con los RecyclerViews
- *PageAdapter*: clase que crea las relaciones entre los Fragments.
- *FondosAdapter*: clase que relaciona los CardViews con los RecyclerViews.
- *CarterasAdapter*: clase que relaciona los CardViews con los RecyclerViews.

4.3.4 Diagrama de la arquitectura de la aplicación

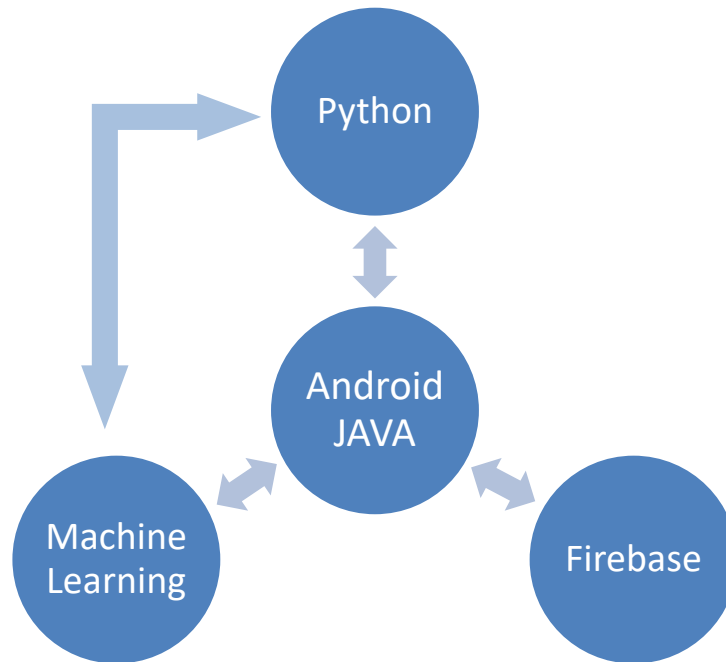


Ilustración 18 – Arquitectura

4.3.5 Recursos requeridos

1. Firebase
2. Android – Java
3. Android Studio
4. RapidMiner
5. Excel
6. Python
7. Chaquopy
8. LineChart

4.4 Resultados del proyecto

Para poder realizar la clasificación de los datos mediante la lectura del modelo generado en el entrenamiento, ha sido necesario el uso de Chaquopy, lo que permite, mediante un SDK de Python, poder ejecutar el código de Machine Learning que tenemos en Python, ya que proporciona mayor simplicidad que al llevarlo a cabo en Java

Para ello se ha tenido que solicitar una licencia Open-Source para que no saltaran alarmas cuando se ejecutara dicho fragmento de código.

Se ha seguido la documentación para poder implementar todo lo necesario en Android para su funcionamiento, como dependencias, una carpeta específica para Python, el NDK y las librerías que se han creído necesarias.

```
defaultConfig {
    applicationId "com.pfgAlex.miroboadvisor1_0"
    sourceSets{
        main{
            python{
                srcDirs = ["src/main/python"]
            }
        }
    }
    minSdkVersion 16
    targetSdkVersion 29
    versionCode 1
    versionName "1.0"
    testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    ndk {
        abiFilters "armeabi-v7a", "x86"
        abiFilters "arm64-v8a"
    }
    python{
        buildPython "C:/Users/Alex/AppData/Local/Programs/Python/Python38-32/python.exe"
        pip {
            install "numpy"
            install "tensorflow"
            install "scikit-learn"
            install "pandas"
            install "matplotlib"
            install "joblib"
        }
    }
}
```

Ilustración 19 - Chaquopy

```
buildscript {
    repositories {
        google()
        jcenter()
        maven { url "https://chaquo.com/maven" }
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:4.0.0'
        classpath 'com.google.gms:google-services:4.3.3' // Google Services plugin
        classpath "com.chaquo.python:gradle:7.0.3"
        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
    }
}
```

Ilustración 20 – Chaquopy

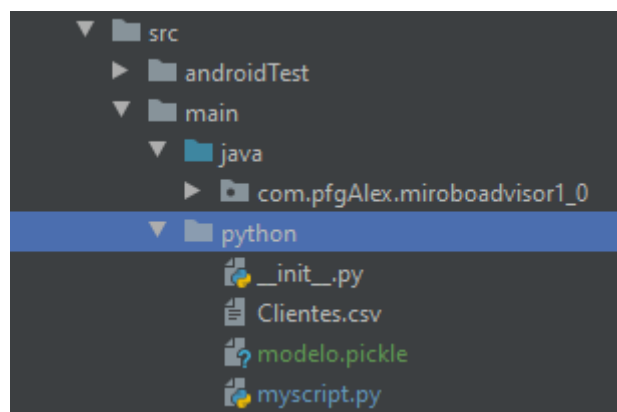


Ilustración 21 - Chaquopy

4.4.1 Código en Python – Entrenamiento

Para generar el modelo es necesario tratar dos parámetros:

- X: atributos que va a pasarse por parámetro a la clasificación.
- Y: atributo con el que se va a realizar la recomendación.

Se ha elegido un k=27 ya que así se recogen todos los fondos, aunque en cuanto a rendimiento sea más lento, dará más efectividad.

Lo que retorna esta función es el modelo.

```
def train():  
  
    import pandas as pd  
    from sklearn.neighbors import KNeighborsClassifier  
  
    #Lectura del dataset  
    data = pd.read_csv(join(dirname(__file__), 'Clientes.csv'), sep=';')  
  
    x = data[["Age", "Income", "Score"]].copy()  
    y = data[["Fondos"]].copy()  
  
    #Agrupación de nuevo en clusters  
    knn_new = KNeighborsClassifier(n_neighbors=27)  
    knn_new.fit(x, y.values.ravel())  
  
    return knn_new
```

Ilustración 22 - Creación modelo

4.4.2 Código en Python – Clasificación

La clasificación recibe por parámetros los atributos que se han introducido en el modelo: edad, ingresos y ahorros.

Se carga el modelo llamando a la función del entrenamiento. El resultado es el identificador del fondo que ha recomendado mediante una predicción usando la edad, ingresos y una puntuación.

```
def classification(edad, ingresos, ahorros):  
  
    classifier = train()  
    score = (ahorros/(ingresos*edad))*100  
    result = classifier.predict([[edad, ingresos, score]])  
  
    return result[0]
```

Ilustración 23 - Clasificación

4.4.3 Código en Java – Interpretación del código Python

Como bien dice la documentación de Chaquopy, hay que crear una instancia de Python.

```
Python py = Python.getInstance();  
PyObject pyf = py.getModule( s: "myscript"); //fichero python  
PyObject obj = pyf.callAttr( key: "test", Integer.parseInt(num_edad), Integer.parseInt(num_ingresos), Integer.parseInt(num_ahorros));
```

Ilustración 24 - Instancia Python

Una vez creada la instancia de Python, se crea un objeto de Python, el cual va a hacer referencia al archivo donde se encuentra el código que se quiere ejecutar.

Para poder pasarle los parámetros necesarios a la función, también se necesita crear un objeto.

Este último objeto será el que contenga el contenido del resultado de la recomendación

Si los datos introducidos, no se corresponden con un fondo, el algoritmo otorgará el más aproximado.

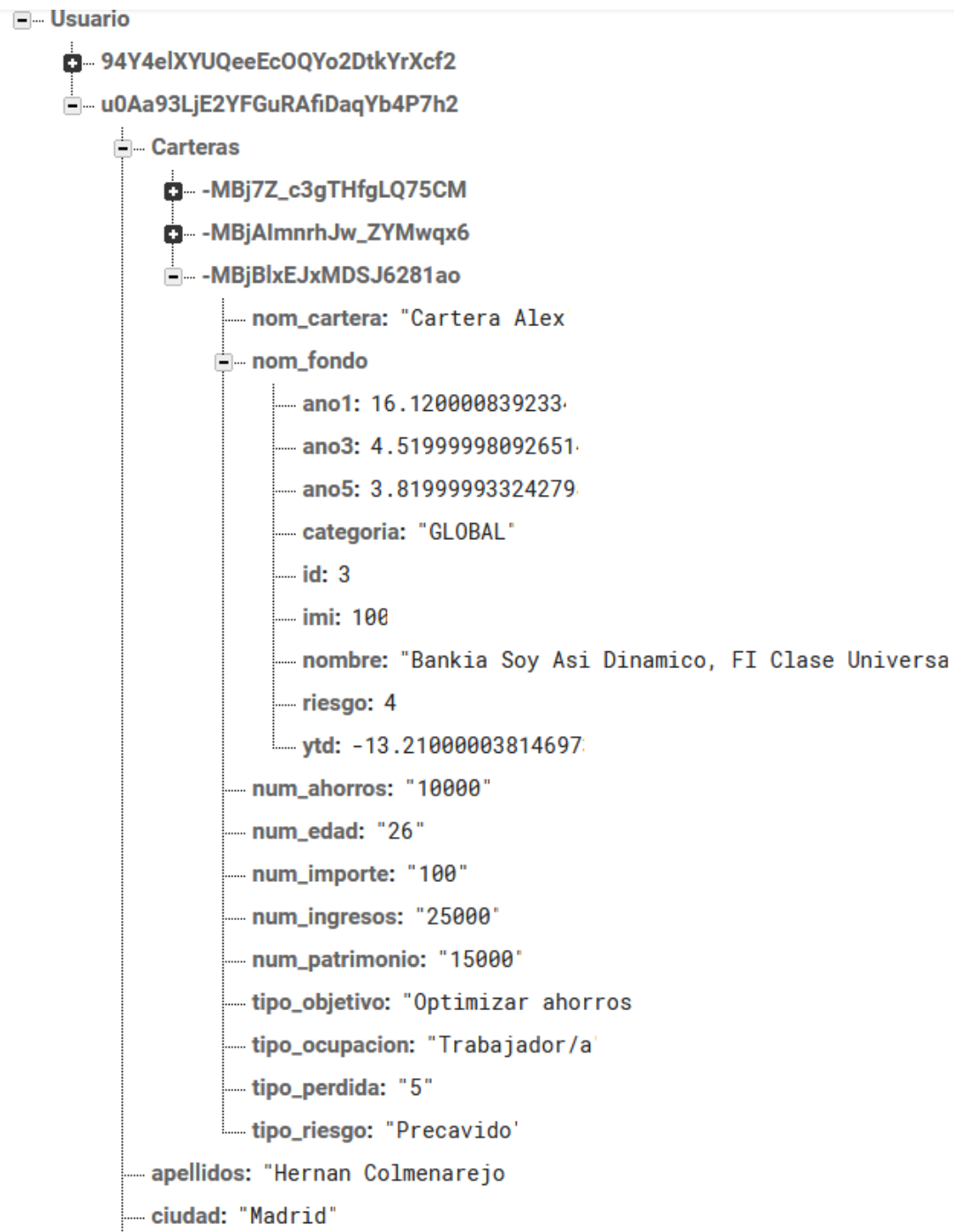


Ilustración 25 - BBDD

Esta recomendación está basada en un dataset de 12 fondos de inversión variados y un dataset de 200 clientes.

Se han repartido esos 12 fondos entre los 200 clientes y a partir de ahí, el usuario cuando crea una cartera se fija en qué cliente con el resultado de la clasificación (el id del fondo) se parece más a él en cuanto a edad, ingresos y puntuación, una vez hecho eso, a la cartera se le asocia un fondo.

La puntuación se obtiene de una fórmula:

$$\frac{\text{ahorros}}{\text{ingresos} * \text{edad}} * 100$$

El resultado en la aplicación sería el siguiente:

Por un lado, se tiene las carteras que se han otorgado según unos datos introducidos por el usuario. Estos datos introducidos son el ingreso anual, los ahorros anuales, la edad y el importe máximo a invertir. De esta manera se crearía una tarjeta, la cual sería una cartera.

A su vez, pero en el apartado de vista general, se crearía otra tarjeta, pero para la identificación de la variación de este fondo de inversión asignado mediante una gráfica.

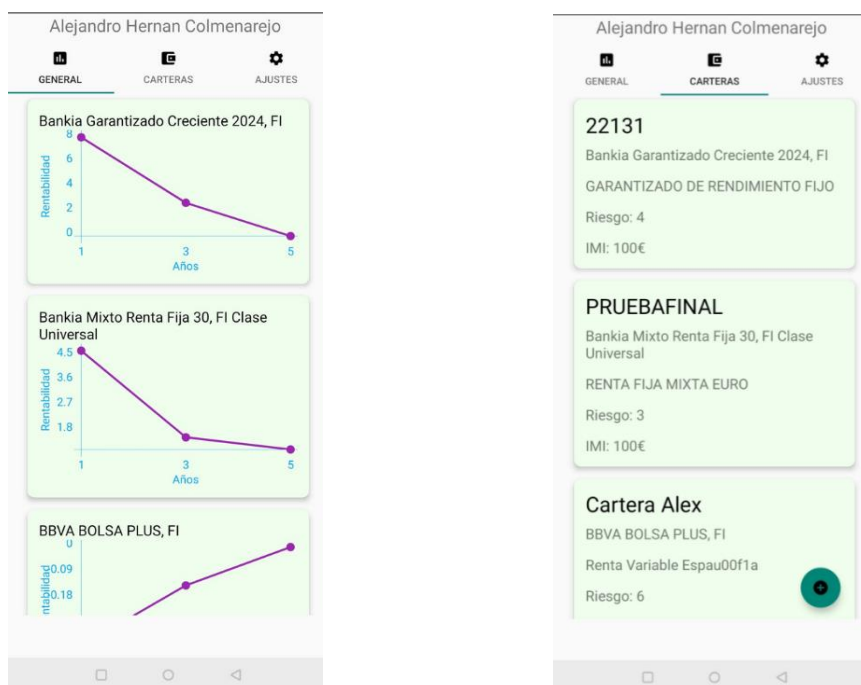


Ilustración 26 – Gráficas y carteras

Capítulo 5. PRESUPUESTO

El total de horas destinadas a este proyecto son 450 horas, que, distribuidos en meses, según el diagrama de GANTT del apartado anterior, serían 5 meses.

Para programar la aplicación se necesita un ordenador, y para probar la aplicación un smartphone.

| Elemento | Precio | Total |
|---|---------------|---------|
| Ordenador | 1500€ | 1500€ |
| Móvil | 400€ | 400€ |
| Programador | 1376€ /mes | 6882€ |
| Subscripción Google Cloud para Firebase (primer año gratuito) | 300€ anuales* | Gratis* |
| Android Studio Software | Gratis | Gratis |
| Conexión Internet | 50 €/mes | 250€ |
| Licencia de Windows 10 | 110€ | 110€ |
| Alta desarrollador Google Play | 25€** | 25€** |

Tabla 2 - Presupuesto

*Primer año de almacenamiento gratuito.

**Solo es 25€ por darse de alta

Coste total del proyecto: 11.285 €

Capítulo 6. VIABILIDAD Y PLAN DE RECURSOS

6.1 Estudio de viabilidad

La aplicación del Machine Learning con JAVA es más tediosa que con Python. Se ha hecho un análisis para realizar la aplicación Android en Python, pero se ha considerado más conveniente buscar información sobre el Machine Learning aplicado con JAVA. La ventaja que se ha observado en parte de la investigación es que Firebase, donde almacenaremos nuestra base de datos, tiene herramientas para usar el Machine Learning en Android.

Todo lo anteriormente citado tiene un efecto.

Si el ordenador que se utiliza para realizar la aplicación no cumple los requisitos puede que algunas funcionalidades se pierdan. Por ejemplo, si no se tiene una tarjeta gráfica y se usa la integrada del ordenador, los iconos del móvil, en el emulador aparecerán invisibles.

6.2 Estudio de viabilidad económica

El punto de mayor debilidad desde el punto de vista económico se encuentra en el almacenamiento en cloud. La razón se encuentra en el tipo de contrato que presentan, el cual, una vez superado cierto número de consultas o almacenamiento procede a pedir ciertos pagos.

6.3 Plan de recursos

Un alumno, como papel de Ingeniero Informático Junior, recién salido de la universidad, el cual tendrá un coste de unos 1376 € al mes. Esta información ha sido obtenida de fuentes sobre salarios de informáticos según el tipo de perfil, como se ha comentado al principio, el perfil asociado es Junior, ya que no tiene experiencia alguna en el sector.

También se utilizará Firebase, que es una base de datos de Google Cloud, la cual en principio es gratuita el primer año, pero con capacidad limitada. Aun así, tenemos disponibles 100 conexiones simultáneas a la base de datos, 10.000 autenticaciones al mes, 1GB de almacenamiento o el ML Kit (Machine Learning Kit) entre otros.

Finalmente, la herramienta de programación de la aplicación será Android Studio y su lenguaje de programación será JAVA.

Capítulo 7. CONCLUSIONES

El desarrollar una aplicación nativa en Android Java ha supuesto demasiados problemas como sobre todo ha sido el de poder ejecutar código Python. Esto tras un largo periodo de tiempo de investigación y pruebas se ha solventado con un SDK de Python llamada Chaquopy, cual permite la ejecución de código en Python para poder así implementar Machine Learning en Android Java.

Otro problema encontrado ha sido con la librería scikit-learn y la incompatibilidad de arquitecturas en cuanto a la ejecución de los modelos. Este motivo ha hecho que no se pueda ejecutar un modelo si no ha sido entrenado en la plataforma donde se ejecuta, Android.

Me hubiera gustado haber podido realizar varias funcionalidades más, pero con tanta investigación he perdido demasiado tiempo. A parte, pienso que ha sido un proyecto bastante tedioso y duro, aunque a la par entretenido.

Por otro lado, gracias a este proyecto he podido adentrarme dentro del mundo de la economía un poco por encima y me ha parecido bastante atractivo. También me ha permitido aprender un tipo de base de datos nuevo como es Firebase y recordar mis conocimientos sobre Android que estudie en el grado superior de Desarrollo de Aplicaciones Multiplataforma. Este conocimiento me he dado cuenta de que pasados 6 años había quedado casi obsoleto, podemos decir que es lo bonito de la informática, pero a la vez lo más duro, ya que uno no se puede quedar atrás, sino que siempre tiene que estar al día en las tecnologías que van saliendo y en las actualizaciones de las tecnologías ya existentes.

.

Capítulo 8. FUTURAS LINEAS DE TRABAJO

Como futuras líneas de trabajo se consideran las siguiente:

- Cifrado de la contraseña en Firebase.
- Comprobaciones de existencia del DNI, IBAN, teléfono.
- Crear la posibilidad de modificar o borrar una cartera una vez se haya creado, si el fondo que se ha recomendado es no satisfactorio para el usuario.
- Mejorar interfaz gráfica ya que no es adaptable para algunos dispositivos.
- Comprobar que todos los datos de la creación de la cartera se han introducido.
- Mejora de la recomendación introduciendo más parámetros.

Todo esto anteriormente no se ha tenido en cuenta al hacer el proyecto porque había otras prioridades.

Lo que más tiempo supondría hacer sería la mejora de la interfaz, junto al borrado y modificación de carteras y la mejora del algoritmo.

Capítulo 9. BIBLIOGRAFIA

- Capitalista, L. H. (2020, January 15). *Robo Advisors* → *La Mejor guía para Invertir en automático*. La Hormiga Capitalista; La Hormiga Capitalista. - <https://lahormigacapitalista.com/como-invertir-robo-advisors/>
- *Salarios para empleos de Programador/a junior en España | Indeed.es*. (n.d.). Retrieved January 28, 2020, from <https://www.indeed.es/salaries/programador-junior-Salaries?period=yearly>
- *Los “robo advisors” crecen y amenazan el monopolio de la banca en gestión de fondos*. (2017, August 31). Expansión.com. <https://www.expansion.com/mercados/2017/08/31/59a6fc4c468aeb92188b469a.html>
- EP. (2019, December 22). *El “robo advisor” de Bankinter, Popcoin, supera los 10 millones de euros dos años después de su lanzamiento*. Expansión; Expansion. <https://www.expansion.com/empresas/banca/2019/12/22/5dff4480e5fdeab1638b45e9.html>
- *Conoce tu perfil inversor en Finanbest | Finanbest Inversiones Inteligentes*. (n.d.). Retrieved January 23, 2020, from <https://www.finanbest.com/test-de-idoneidad/#paso-0>

- *Roboadvisor - Carteras de Inversión | Openbank Wealth*. (n.d.).
Roboadvisor - Carteras de Inversión | Openbank Wealth. Retrieved
January 23, 2020, from [https://www.openbank.es/inversiones/robo-
advisor-gestion-carteras](https://www.openbank.es/inversiones/robo-advisor-gestion-carteras)
- Bauguess, S. W. (2017). *The Role of Big Data, Machine Learning, and AI
in Assessing Risks: A Regulatory Perspective*.
<https://doi.org/10.2139/ssrn.3226514>
- *Deutsche Bank - Carteras modelo*. (2020, January 10).
[https://www.deutsche-bank.es/pbc/data/es/carteras-modelo-estrategia-
pa.html](https://www.deutsche-bank.es/pbc/data/es/carteras-modelo-estrategia-pa.html)
- Hernández, E., González, A., Pérez, B., de Luis Reboredo, A., &
Rodríguez, S. (2018). Virtual Organization for Fintech Management.
*Distributed Computing and Artificial Intelligence, Special Sessions, 15th
International Conference*, 201–210.
- Hudson, C. (2018, November 28). *Ten Applications of AI to Fintech*.
Medium; Towards Data Science. [https://towardsdatascience.com/ten-
applications-of-ai-to-fintech-22d626c2fdac](https://towardsdatascience.com/ten-applications-of-ai-to-fintech-22d626c2fdac)
- de los proyectos Wikimedia, C. (2014, September 5). *Android Studio -
Wikipedia, la enciclopedia libre*. Wikimedia Foundation, Inc.
https://es.wikipedia.org/wiki/Android_Studio#Requisitos_del_sistema
- Alex Narváez Programming. (2017, noviembre 20). *Android y
Firebase 1 - Autenticación - Registro de Usuarios*. Recuperado de
<https://www.youtube.com/watch?v=DIAGb5V9Myo>

- *Android - how to make a scrollable constraintlayout?* (2017, marzo 29). Recuperado de <https://stackoverflow.com/questions/43098150/android-how-to-make-a-scrollable-constraintlayout>
- Coding Demos. (2018a, febrero 25). *Android Tablayout Example With ViewPager And Menu Items (Explained)*. Recuperado de https://www.youtube.com/watch?v=Vxiy_h5hNII
- Coding Demos. (2018b, octubre 8). *Android Firebase - How to Delete a User Account Programmatically From Firebase (Explained)*. Recuperado de <https://www.youtube.com/watch?v=Gp4SsHpzDdg>
- colaboradores de Wikipedia. (2020, abril 27). *Android*. Recuperado de <https://es.wikipedia.org/wiki/Android>
- *Create swipe views with tabs using ViewPager |* . (s. f.). Recuperado 5 de mayo de 2020, de <https://developer.android.com/guide/navigation/navigation-swipe-view>
- *Develop for Android*. (s. f.). Recuperado 5 de mayo de 2020, de <https://material.io/develop/android/>
- *Firebase Authentication*. (s. f.). Recuperado 5 de mayo de 2020, de <https://firebase.google.com/docs/auth>
- *Firebase Realtime Database*. (s. f.). Recuperado 5 de mayo de 2020, de <https://firebase.google.com/docs/database>

- *Fondos indexados: ¿Qué son? ¿Cómo funcionan?* (2018, julio 11). Recuperado de <https://es.investing.com/analysis/fondos-indexados-que-son-como-funcionan-200223245>
- *How to use button in fragment* - Quora. (s. f.). Recuperado 5 de mayo de 2020, de <https://www.quora.com/How-do-I-use-button-in-fragment>
- *Inhabilitar un boton*. (s. f.). Recuperado 5 de mayo de 2020, de <https://es.stackoverflow.com/questions/23263/inhabilitar-un-boton>
- *Login | InbestMe* - Inbestme. (s. f.). Recuperado 5 de mayo de 2020, de <https://investor.inbestme.com/>
- Mato, M. (2018, junio 11). *Cómo crear Swipe Views con tabs distintas en la Action Bar de forma fácil*. Recuperado de <https://medium.com/@manuelmato/c%C3%B3mo-crear-swipe-views-con-tabs-distintas-en-la-action-bar-de-forma-f%C3%A1cil-efe2badf9e53>
- Sociedad Androide. (2017, junio 11). *Android Studio - Autenticacion Con Firebase 2/3*. Recuperado de <https://www.youtube.com/watch?v=5uBlbTVsObs>
- Web, D. M. A. M. J.-. (s. f.). *Características y tabla comparativa de los sistemas operativos móviles más usados*. Recuperado 5 de mayo de 2020, de <https://jmacuna.tecnoblog.guru/2017/03/sistemas-operativos-moviles.html>

- *Neares Neighbours*. (2007–2019). Recuperado de <https://scikit-learn.org/stable/modules/neighbors.html#classification>
- colaboradores de Wikipedia. (s. f.). *K vecinos más próximos*. Recuperado 8 de julio de 2020, de https://es.wikipedia.org/wiki/K_vecinos_m%C3%A1s_pr%C3%B3ximos
- *El algoritmo K-NN y su importancia en el modelado de datos*. (s. f.). Recuperado 8 de julio de 2020, de <https://www.analiticaweb.es/algoritmo-knn-modelado-datos/>
- Na. (2018, July 10). *Clasificar con K-Nearest-Neighbor ejemplo en Python*. Aprende Machine Learning. <https://www.aprendemachinelearning.com/clasificar-con-k-nearest-neighbor-ejemplo-en-python/>
- Smith, M. (s. f.). *The easiest way to use Python in your Android app*. Recuperado de <https://chaquo.com/chaquopy/>

Capítulo 10. ANEXO I

10.1 Manual de usuario

10.1.1 Pantalla de inicio

10.1.1.1 Inicio de sesión

Se introducen los datos que se ha registrado, es decir, para poder entrar en la aplicación es necesario usar el email y la contraseña. Una vez introducidos se puede presionar el botón “INICIO SESIÓN”, el cual redirigirá a la pantalla “GENERAL”

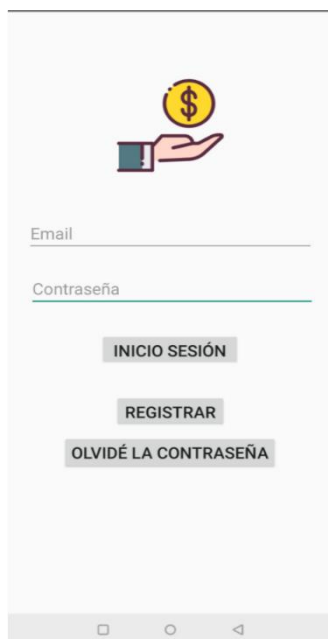


Ilustración 27 - Anexo Manual Usuario

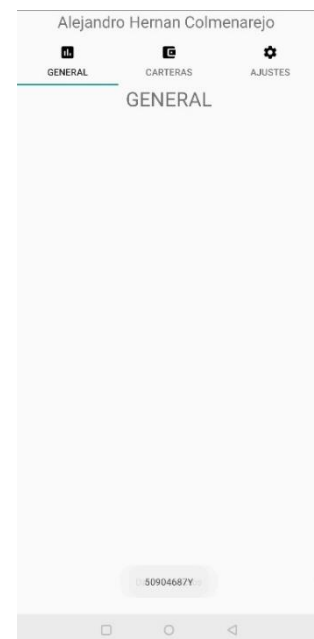


Ilustración 28 - Anexo Manual Usuario

10.1.1.2 Registro de usuario

Para registrarse, se tiene que pulsar en el botón “REGISTRAR”, el cuál redirige a la pantalla donde se tiene que rellenar un formulario. Una vez se hayan rellenado todos los datos, se termina guardando los datos en la base de datos mediante la acción de pulsar “REGISTRARSE”, la cual redirigirá a la pantalla inicial para poder iniciar sesión.

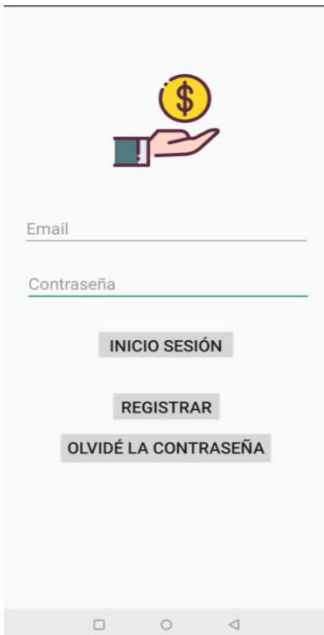


Ilustración 29 - Anexo Manual Usuario



Ilustración 30 - Anexo Manual Usuario

10.1.1.3 Recuperación de contraseña

Si por alguna circunstancia, el usuario no recuerda su contraseña, esta, se puede recuperar haciendo clic en el botón “OLVIDÉ LA COTRASEÑA”. Se introduce el email de la cuenta la cual no recordamos la contraseña, se presiona el botón “ENVIAR CONTRASEÑA” y llegará un correo con un enlace en donde pedirá que se introduzca la nueva contraseña.

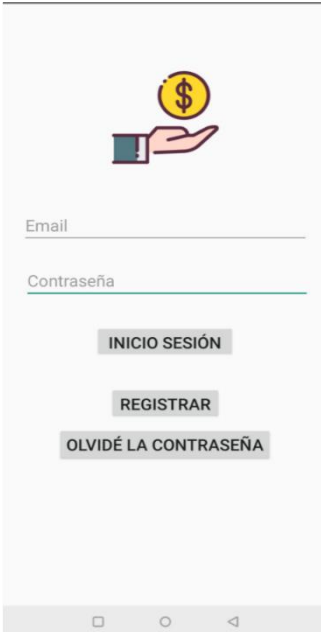


Ilustración 31 - Anexo Manual Usuario


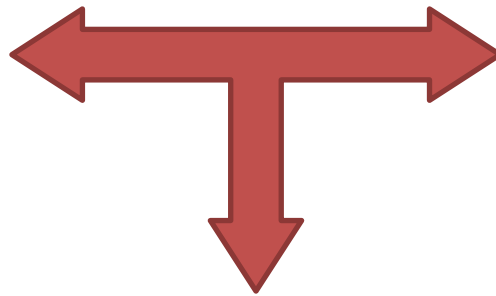


Ilustración 32 - Anexo Manual Usuario

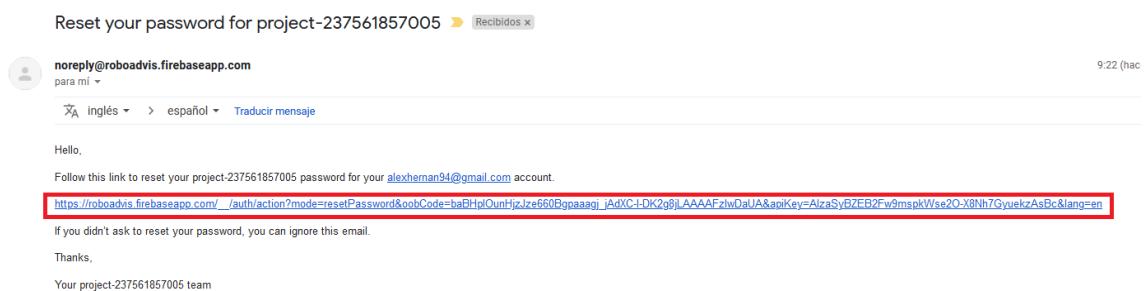


Ilustración 34 - Anexo Manual Usuario

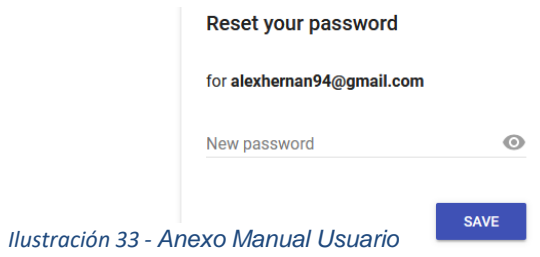


Ilustración 33 - Anexo Manual Usuario

10.1.2 Pantalla aplicación


10.1.2.1 General

En esta pantalla se podrán observar los gráficos que tienen los fondos que se han dado al usuario y poder así observar su variación en 1, 3 y 5 años atrás.



Ilustración 35- Anexo Manual Usuario

10.1.2.2 Carteras

Aquí se podrán observar los datos de las carteras que se han creado. Si se desea crear una cartera nueva, habrá que pulsar el botón  para poder así rellenar los datos necesarios. Si se tiene alguna duda, se podrá pulsar el botón “¿?”

Para finalizar la creación de una cartera, basta con pulsar el botón “CREAR CARTERA”. Se podrá observar el nombre de la cartera, el nombre del fondo, el tipo de categoría del fondo, el riesgo del fondo y el ingreso mínimo a invertir que necesita ese fondo.



Ilustración 37 - Anexo Manual Usuario

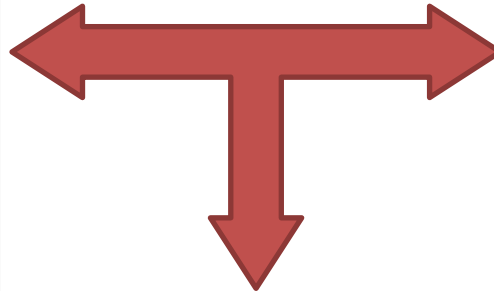



Ilustración 36 - Anexo Manual Usuario

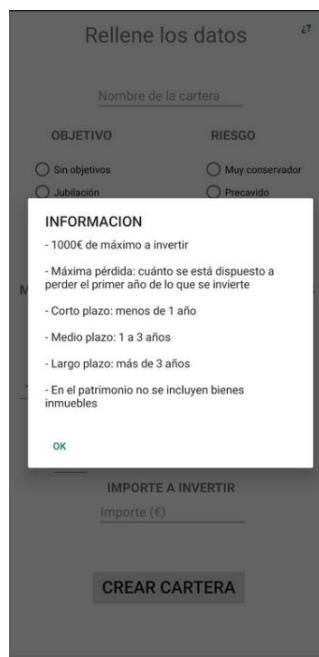


Ilustración 38 - Anexo Manual Usuario

10.1.2.3 Ajustes

Aquí se ve los datos que registró el usuario. Estos datos son modificables mediante el botón “MODIFICAR”, el cual activará el botón “GUARDAR DATOS” y se desactivarán los demás botones.



Ilustración 39 - Anexo Manual Usuario

Ilustración 40 - Anexo Manual Usuario

Si se quiere borrar la cuenta o cerrar la sesión, basta con pulsar los botones “BORRAR CUENTA” o “CERRAR SESIÓN”.

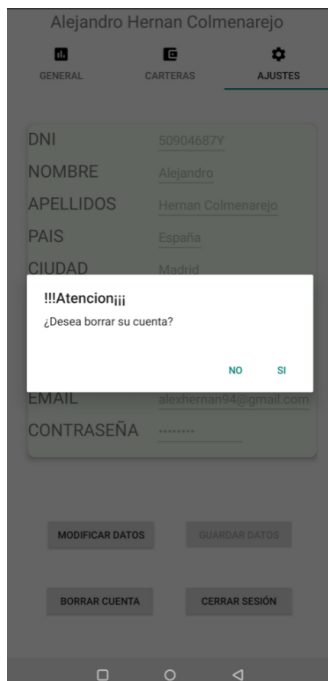


Ilustración 42 - Anexo Manual Usuario

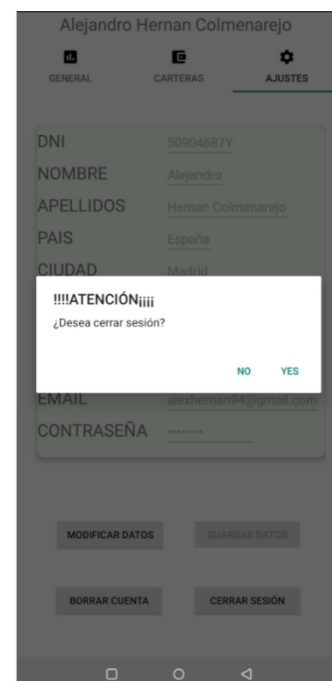


Ilustración 41 - Anexo Manual Usuario

10.2 Manual de instalación

1. Si se quiere continuar con la programación de dicha aplicación, solo es necesario obtener el código de la página de Github: https://github.com/alexhernan94/MiRoboadvisor1_0
2. Lo clonamos en un directorio y mediante el uso de un IDE como puede ser Android Studio, lo abrimos.
3. Una vez con el proyecto abierto, esperamos que se actualicen los plugs-in y otras herramientas. Para poder tener acceso a Firebase, Chaquopy o LineChart, entre otros, se debe compilar el proyecto. Este proceso tardará un rato y puede que pida reiniciar el Android Studio o actualizar el Java, SDK, Firebase Plug-in, etc.
4. Cuando el proceso de compilación termina, ya podemos continuar trabajando en el proyecto.

10.2.1 Android Studio

1. Se descarga de la página oficial:
<https://developer.android.com/studio#downloads>
2. Para la instalación se puede seguir este tutorial de la página oficial:
<https://developer.android.com/studio/install>

10.2.2 Firebase Realtime Database

1. Lo primero que hay que hacer es agregar el proyecto Android a Firebase, para ello está este tutorial oficial:
<https://firebase.google.com/docs/android/setup?hl=es-419>
2. Una vez agregado, hay que crear la base de datos y agregar el SDK de la base de datos Realtime a la aplicación:
<https://firebase.google.com/docs/database/android/start?hl=es-419>

Capítulo 11. ANEXO II

- Usuario
 - id_user
 - DNI
 - Nombre
 - Apellidos
 - Domicilio
 - Ciudad
 - País
 - Iban
 - Teléfono
 - Email
 - Contraseña
 - Carteras
 - id_cartera
 - Nombre
 - Ahorros
 - Edad
 - Importe
 - Ingresos
 - Patrimonio
 - Objetivo
 - Ocupación
 - Pérdida
 - Riesgo
 - Fondo
 - Id
 - Nombre
 - Tipo
 - IMI
 - Riesgo
 - YTD
 - 1 años
 - 3 años
 - 5 años
- Fondos
 - id_fondos
 - Id
 - Nombre
 - Tipo
 - IMI
 - Riesgo
 - YTD
 - 1 años
 - 3 años
 - 5 años

