1ºPortada

**CENTRO INTEGRADO MARÍA ANA SANZ PAMPLONA**

**2º GRADO SUPERIOR DESARROLLO DE APLICACIONES MULTIPLATAFORMA**

**TRABAJO DE FIN DE GRADO APLICACIÓN MÓVIL EN ANDROID PARA LA EDUCACIÓN FINANCIERA DE INVERSIONES EN INVERSIÓN EN BOLA Y CRIPTOMONEDAS**

---------------------------------------------------------------------------------------------------------------------

Autor: AITOR ZUBERO

Tutor/TFG: ENEKO

Tutor Académico: EMILIO

Indice

2ºintroduccion 3

3. Descripción inicial de la aplicación 4

[3.1Explicación del proyecto: 4](#__RefHeading___Toc1581_2392937072)

[3.2Justificación de la necesidad o problema que soluciona: 4](#__RefHeading___Toc1583_2392937072)

4. Objetivos del proyecto 6

[4.1 Resultados esperados al finalizar el desarrollo: 6](#__RefHeading___Toc1682_2392937072)

5. Recursos hardware y software 8

[5.1Recursos necesarios: 8](#__RefHeading___Toc1900_97050301)

[5.2Hardware: 8](#__RefHeading___Toc1902_97050301)

[5.3Software: 8](#__RefHeading___Toc1904_97050301)

[ANDROID STUDIO 8](#__RefHeading___Toc1906_97050301)

[FIREBASE: 13](#__RefHeading___Toc1908_97050301)

[Registro de una app en fire base 14](#__RefHeading___Toc1870_2045927908)

[FIREBASE AUTHENTICATION 17](#__RefHeading___Toc1872_2045927908)

[FIREBASE STO 21](#__RefHeading___Toc1874_2045927908)

[API 22](#__RefHeading___Toc1910_97050301)

6. Fases del desarrollo 23

[Diagrama de temporalización 23](#__RefHeading___Toc1169_2392937072 Copy 1)

[Descripción de cada fase 23](#__RefHeading___Toc1169_2392937072)

[● Fase de análisis 23](#__RefHeading___Toc1171_2392937072)

[● Fase de diseño 23](#__RefHeading___Toc1173_2392937072)

[Diagrama de la base de datos: entidades y relaciones. 23](#__RefHeading___Toc1175_2392937072)

[Diagrama de casos de uso: interacción usuario-sistema. 23](#__RefHeading___Toc1177_2392937072)

[Planificación de la interfaz: guía de estilos, mockups (hechos con GMA, a mano, Figma, etc.), y temas de la aplicación. 23](#__RefHeading___Toc1179_2392937072)

[● Fase de desarrollo 23](#__RefHeading___Toc1181_2392937072)

[● Fase de pruebas y depuración 23](#__RefHeading___Toc1183_2392937072)

[● Fase de lanzamiento 24](#__RefHeading___Toc1185_2392937072)

[7. Conclusiones 25](#__RefHeading___Toc1187_2392937072)

8. Bibliografía y referencias 26

# 1ºintroduccion

En la actualidad, el auge de las plataformas digitales de inversión ha generado un entorno en el que cada vez más personas buscan acceder al mundo del trading desde sus dispositivos móviles. Este escenario, marcado por una amplia oferta de servicios financieros y aplicaciones especializadas, representa un reto para los usuarios principiantes, quienes se enfrentan a una elevada complejidad técnica y a la falta de herramientas intuitivas que les permitan iniciarse de manera segura. Esta dificultad se acentúa por la dispersión de recursos educativos y por la existencia de interfaces poco amigables para quienes carecen de experiencia previa en los mercados financieros.

Frente a esta situación, plataformas reconocidas como eToro, Binance o TradingView han ampliado sus funcionalidades para captar a nuevos usuarios, invirtiendo en mejorar la accesibilidad de sus sistemas. No obstante, la curva de aprendizaje sigue siendo un factor determinante que aleja a muchos potenciales inversores, especialmente a los jóvenes que se sienten atraídos por la posibilidad de gestionar sus propias finanzas, pero carecen del acompañamiento adecuado para comprender los riesgos y estrategias involucradas.

En el contexto nacional, el interés por el trading ha crecido considerablemente, impulsado por la difusión de contenido en redes sociales, foros especializados y comunidades digitales. Esta tendencia ha puesto en evidencia la necesidad de aplicaciones que combinen formación, práctica y una experiencia de usuario clara y sencilla. Según datos recientes, más del 70% de los usuarios novatos abandonan las plataformas de trading en sus primeras semanas, debido a la falta de comprensión del funcionamiento del mercado y de herramientas que faciliten su integración progresiva.

En este contexto, **Neo**Tradesurge como una propuesta innovadora diseñada específicamente para personas que desean iniciarse en el trading de forma segura, guiada y sin complicaciones. La aplicación ofrece un entorno visualmente amigable que integra funcionalidades como simuladores de inversión con datos en tiempo real, un glosario financiero interactivo, alertas personalizadas sobre movimientos de mercado, noticias económicas relevantes y una sección de aprendizaje progresivo basada en lecciones breves y dinámicas.

Para el desarrollo de este proyecto, se ha utilizado la tecnología más actual, con enfoque en el lenguaje de programación Kotlin dentro del entorno de Android Studio. El backend de la aplicación se gestiona a través de Firebase, permitiendo autenticación segura de usuarios tanto mediante correo y contraseña cifrada como a través de inicio de sesión con Google. Además, se emplea **Firebase** Firestore para almacenar el historial de transacciones simuladas, preferencias del usuario y configuraciones de la app.

Para la obtención de datos del mercado en tiempo real y de noticias económicas, se han integrado APIs externas como Tiingo y Finnhub, lo que permite ofrecer información precisa y actualizada sobre acciones, criptomonedas y otros activos financieros. A su vez, la aplicación mantiene comunicación con un servidor propio para acceder a contenidos educativos y métricas personalizadas de desempeño del usuario.

Durante el desarrollo, se han implementado librerías ampliamente utilizadas en el ecosistema Android, como Retrofit para la gestión de peticiones HTTP, Glide para la carga eficiente de imágenes, MPAndroidChart para visualización de gráficos de cotizaciones y Gson para el tratamiento de datos JSON, asegurando un rendimiento fluido y una experiencia de usuario satisfactoria. EasyTrade se presenta así como una herramienta integral que busca democratizar el acceso al trading, combinando formación, práctica y tecnología en una sola aplicación móvil.

# 2. Descripción inicial de la aplicación

### 3.1Explicación del proyecto:

En el contexto actual de la tecnología, estamos siendo testigos de un crecimiento exponencial en el desarrollo de aplicaciones móviles orientadas a distintos sectores, especialmente en el ámbito financiero. Desde hace años, existen bancos digitales, plataformas de inversión y servicios vinculados a las tecnologías blockchain. Sin embargo, son pocas las herramientas accesibles y educativas que permitan a los usuarios sin experiencia iniciarse en el mundo del trading de criptomonedas y las inversiones financieras de forma segura y didáctica.

Este proyecto surge con el propósito de cubrir esa necesidad: se trata de una aplicación móvil desarrollada en Android Studio, diseñada para ofrecer una primera toma de contacto con el mundo de las inversiones sin arriesgar dinero real. A través de una cuenta de usuario, los participantes podrán simular operaciones de inversión con capital ficticio, practicar técnicas de análisis de mercado y aprender a interpretar gráficas de velas, entre otras funcionalidades.

La aplicación no solo permite experimentar cómo funciona el trading de criptomonedas, sino también observar el comportamiento de los mercados, identificar oportunidades de inversión, y tomar decisiones estratégicas basadas en datos visuales y actualizados. Todo esto en un entorno seguro, sin pérdidas económicas reales, y con el objetivo de formar a los usuarios antes de lanzarse a plataformas de inversión reales.

### 2.2Justificación de la necesidad o problema que soluciona:

El mundo de las criptomonedas y las inversiones financieras puede resultar abrumador y arriesgado para quienes no tienen conocimientos previos. Muchas personas interesadas en invertir no saben por dónde empezar, y acaban tomando decisiones precipitadas que pueden derivar en pérdidas económicas importantes.

Actualmente, existen plataformas de trading avanzadas, pero pocas están pensadas para quienes están dando sus primeros pasos. Además, la falta de educación financiera y de espacios donde aprender practicando, sin consecuencias económicas, supone una barrera de entrada para gran parte de la población.

Esta aplicación busca reducir esa barrera, ofreciendo una herramienta educativa y práctica, donde los usuarios puedan familiarizarse con el funcionamiento del mercado, entender conceptos básicos de inversión y análisis técnico, y prepararse para una experiencia real, sin poner en riesgo su dinero. De esta forma, se promueve una inversión más consciente, informada y responsable.

# 3. Objetivos del proyecto

### 3.1 Resultados esperados al finalizar el desarrollo:

Al finalizar este proyecto, se espera contar con una aplicación móvil plenamente funcional, capaz de proporcionar a los usuarios una experiencia completa de simulación de inversión. Entre los resultados clave que se pretende alcanzar, destacan los siguientes:

1. **Sistema de registro y autenticación en la nube:**
   1. La aplicación permitirá a los usuarios crear una cuenta personal y autenticarse mediante un sistema seguro alojado en la nube. Esto facilitará la gestión individual de sus datos y su acceso desde distintos dispositivos.
2. **Simulación de operaciones con capital ficticio:**
   1. Se integrará un sistema de inversión simulado en el que los usuarios podrán operar con dinero ficticio. Esto les permitirá practicar estrategias y conocer el comportamiento del mercado sin arriesgar recursos reales.
3. **Historial de actividad y seguimiento del progreso:**
   1. Cada usuario podrá consultar su historial de inversiones, analizar su evolución a lo largo del tiempo y observar el impacto de sus decisiones financieras. Esta función fomentará el aprendizaje y la mejora continua.
4. **Análisis de gráficos en tiempo real:**
   1. La aplicación incluirá herramientas visuales que permitirán observar gráficos de velas y otros indicadores clave del mercado, proporcionando una base sólida para el análisis técnico.
5. **Recomendaciones y orientación para invertir:**
   1. Se implementarán sugerencias o señales básicas que ayuden al usuario a identificar posibles momentos adecuados para invertir, en función del comportamiento de los activos y patrones del mercado.
6. **Evaluación de la preparación del usuario:**
   1. Con el uso continuado de la app, los usuarios podrán evaluar su progreso y determinar si están listos para pasar al entorno real de inversión, habiendo adquirido las habilidades necesarias para tomar decisiones informadas.

Beneficios que se esperan obtener:  
Consolidación de conocimientos técnicos adquiridos durante la formación

Este proyecto me permite poner en práctica una amplia variedad de contenidos estudiados durante mi formación, incluyendo el desarrollo en Android, el manejo de bases de datos en tiempo real, el diseño de interfaces de usuario y el uso de servicios externos en la nube. Al integrar todo esto en un solo proyecto funcional, puedo reforzar y consolidar mi aprendizaje de manera significativa.

1. Experiencia en el desarrollo de aplicaciones móviles reales
   1. Trabajar en una aplicación con características completas —registro, análisis gráfico, simulaciones, almacenamiento en la nube, etc.— me proporciona una experiencia práctica muy valiosa, cercana al tipo de proyectos que se realizan en entornos laborales reales.
2. Dominar tecnologías y herramientas actuales
   1. Uno de los principales objetivos es adquirir soltura en el uso de tecnologías ampliamente empleadas por desarrolladores profesionales. Entre ellas destacan:
   2. Firebase como plataforma Backend as a Service, integrando autenticación, almacenamiento y base de datos.
   3. Kotlin y Java como lenguajes de desarrollo para Android.
   4. Manejo de gráficos y análisis de datos financieros, conceptos útiles en el desarrollo de apps de datos en tiempo real.
3. Mejora de la capacidad de resolución de problemas
   1. Durante el proceso de desarrollo, surgen múltiples dificultades técnicas y conceptuales que requieren soluciones creativas y bien fundamentadas. Enfrentar y resolver estos desafíos me ayudará a mejorar mis habilidades de análisis y mi capacidad para encontrar soluciones eficaces, organizadas y adaptables.
4. Autonomía en la gestión de un proyecto completo
   1. Diseñar, planificar, codificar, probar y documentar una aplicación de principio a fin implica asumir un papel completo de desarrollador. Esto me ha permitido desarrollar una visión integral del ciclo de vida de una aplicación y fortalecer mis competencias como profesional autónomo, capaz de gestionar todas las fases de un proyecto tecnológico.
5. Preparación para el acceso al mundo laboral
   1. El resultado final de este proyecto constituye una carta de presentación sólida que podré utilizar en futuras entrevistas de trabajo o procesos de selección. Una app funcional, publicada o demostrable, que utiliza tecnologías actuales y resuelve un problema concreto, es una excelente forma de mostrar mis habilidades a potenciales empleadores o clientes.
6. Crecimiento personal como programador
   1. Finalmente, este proyecto ha supuesto también un crecimiento personal. Me ha permitido confiar más en mis capacidades, explorar nuevas tecnologías con autonomía y creatividad, y desarrollar una mentalidad de mejora continua, que será clave para mi evolución como desarrollador de software.

# 4. Recursos hardware y software

## 4.1Recursos necesarios:

### 4.2Hardware:

* + PC, móvil Android

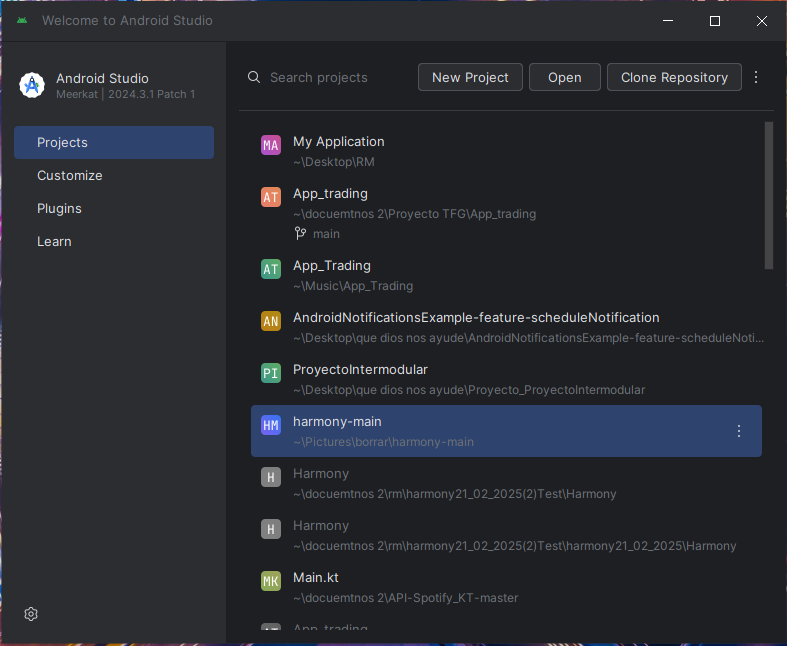
### 5.3Software:

* + Android Studio, Firebase, figma, jsoneditoronline, postman, git, github, git craken, LibreOffice, LibreOfficeClac,

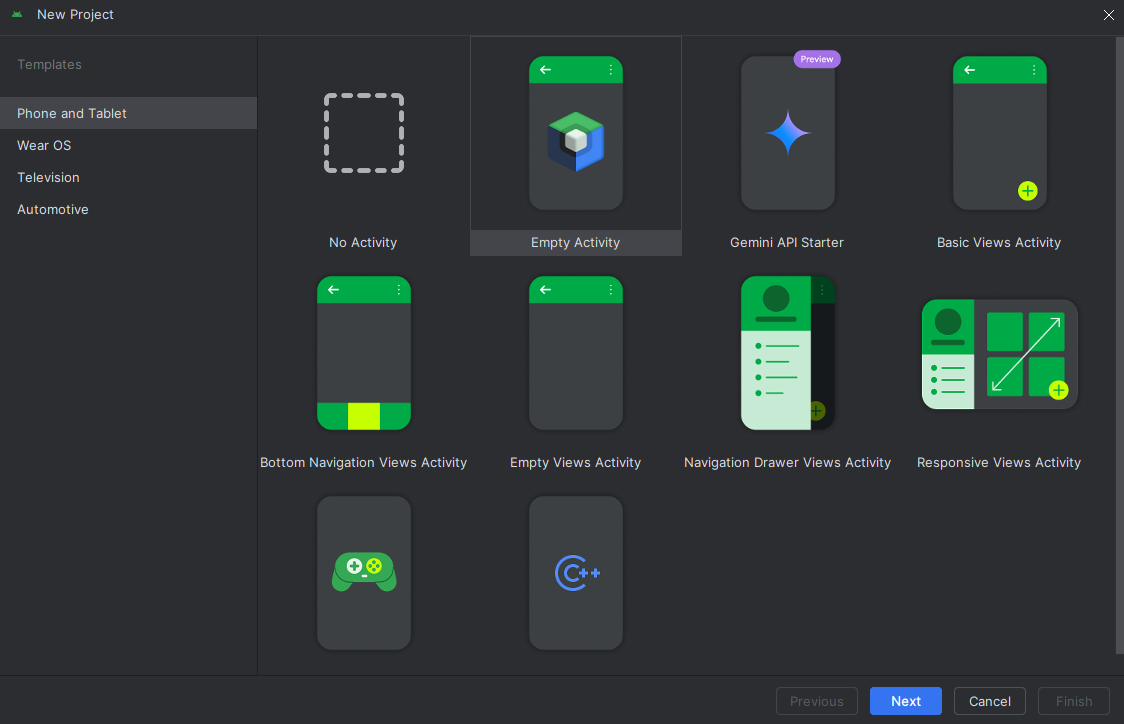
#### ANDROID STUDIO

Android Studio ha sido una herramienta clave en el desarrollo de este proyecto. Desde las fases iniciales, se utilizó esta plataforma para configurar la aplicación, llevar a cabo el desarrollo y realizar las pruebas necesarias. Su entorno de trabajo, fácil de usar y con múltiples funcionalidades integradas, como el emulador de dispositivos y el diseñador de interfaces visuales, permitió acelerar considerablemente el proceso. Además, se eligió por ser el entorno de desarrollo oficial respaldado por Google para aplicaciones Android.

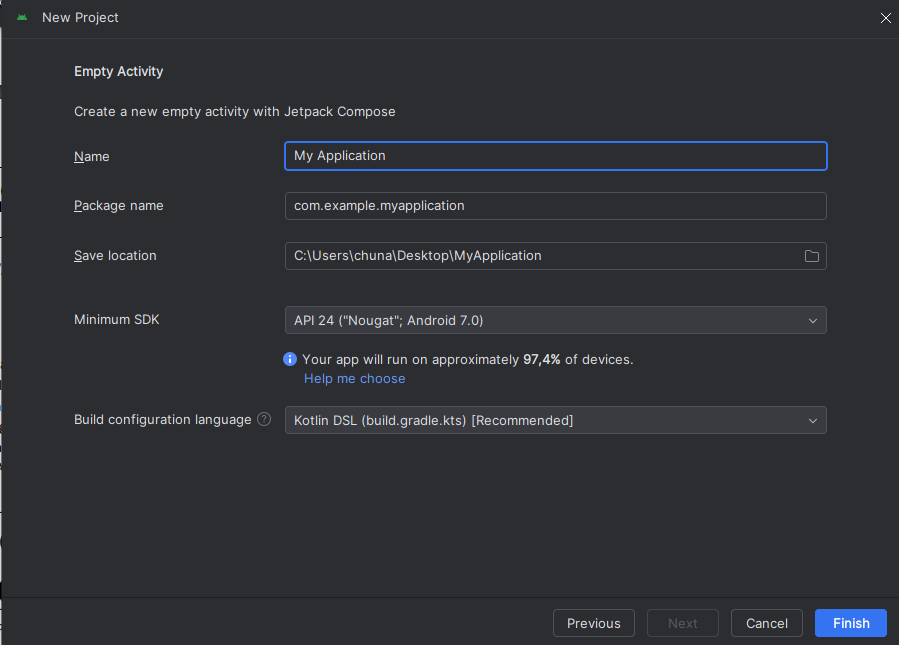
1. Configuracion del proyecto: Abrimos el android studio y nos apareera esta ventana, si es la primera vez estará vaciá de proyectos.
   1. Le damos al botón de new project



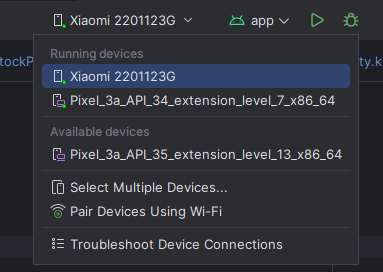
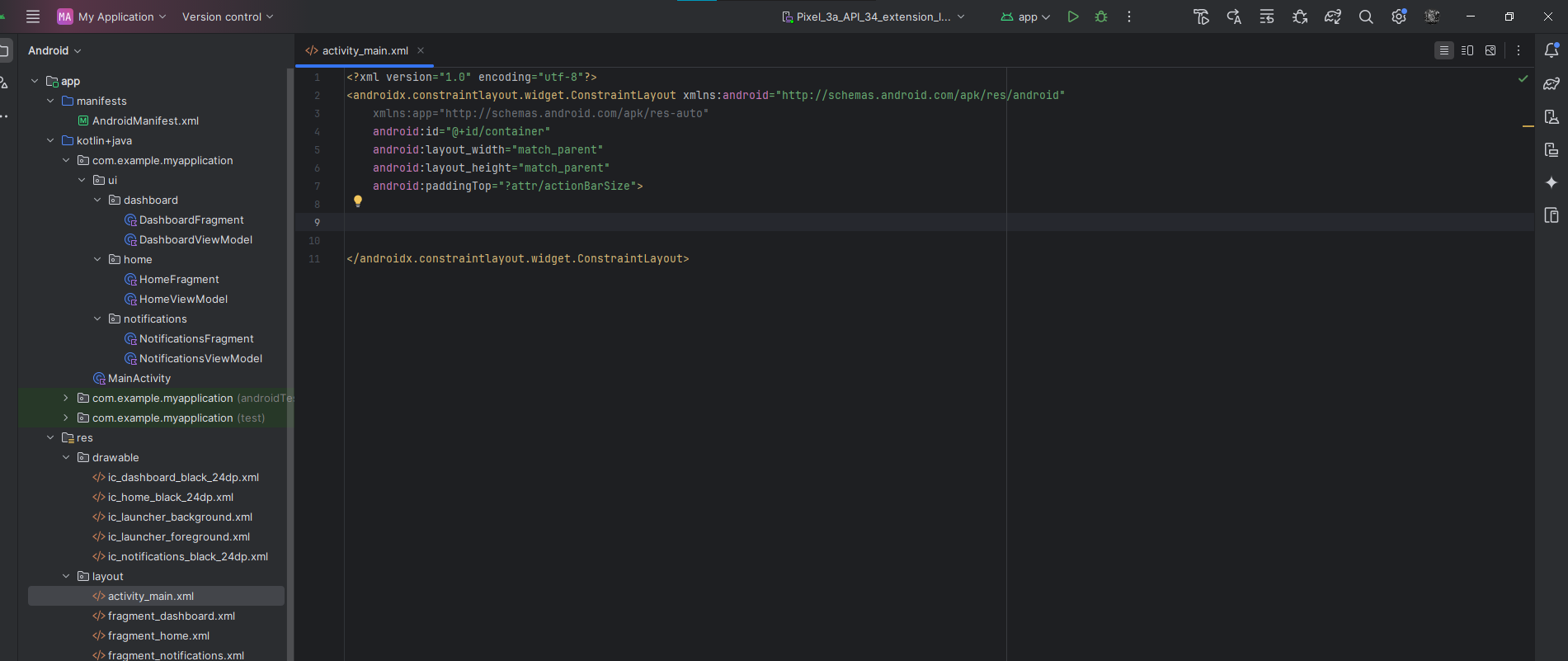
* 1. nos aparece esta ventana con unas plantillas por defecto en mi caso va ser una plantilla vaciá para empezar el proyecto con una “Empty View Activity”



* 1. En esta ventana deberemos cambiar el nombre, por el del nombre de proyecto que deseamos poner, también si queremos cambiar el nombre de la estructura de proyecto, el lugar donde queremos guardarlo, el sdk mínimo (esto lo dejo por defecto) y por ultimo el lenguaje de compilación (lo dejo por defecto pero se puede cambiar a grovy)
     1. Por ultimo le damos a finish.
  2. Se abrirá un proyecto vació como este :



* 1. Ademas de poder emular un dispositivo virtual podemos testear nuestra aplicación en un dispositivo fijo android
  2. en la parte superior podremos acceder a un desplegable que nos indicara los dispositivos



La ventana MainActivity en Android Studio es la actividad principal y usualmente es la primera pantalla que el usuario ve al iniciar la aplicación. Esta ventana actúa como el punto de entrada de la app y es donde definimos la lógica inicial para mostrar contenido, manejar interacciones básicas o iniciar flujos importantes de la aplicación.

En términos técnicos, **MainActivity** hereda de la clase **AppCompatActivity** o simplemente **Activity** en **Kotlin**, y es una de las múltiples actividades (pantallas) que pueden existir dentro de una app Android. Su ciclo de vida está gestionado por el sistema operativo Android, lo que significa que esta actividad puede estar activa, pausada o detenida dependiendo del estado de la aplicación y del uso que el usuario haga de la misma.

Cuando hablamos de que esta actividad "se ejecuta continuamente", nos referimos a que, una vez lanzada, permanece en primer plano (a menos que el usuario navegue fuera de ella o cierre la aplicación). El sistema operativo se encarga de mantenerla activa o ponerla en pausa según lo requiera la gestión de recursos del dispositivo.

La configuración de qué actividad se ejecuta primero, o es lanzada al iniciar la aplicación, está definida en el archivo **AndroidManifest.xml**. Aquí se especifica qué actividad tiene el filtro de intención (intent filter) con acciones y categorías como **MAIN** y **LAUNCHER**, lo que le indica al sistema que esta es la "actividad de entrada". Por ejemplo:

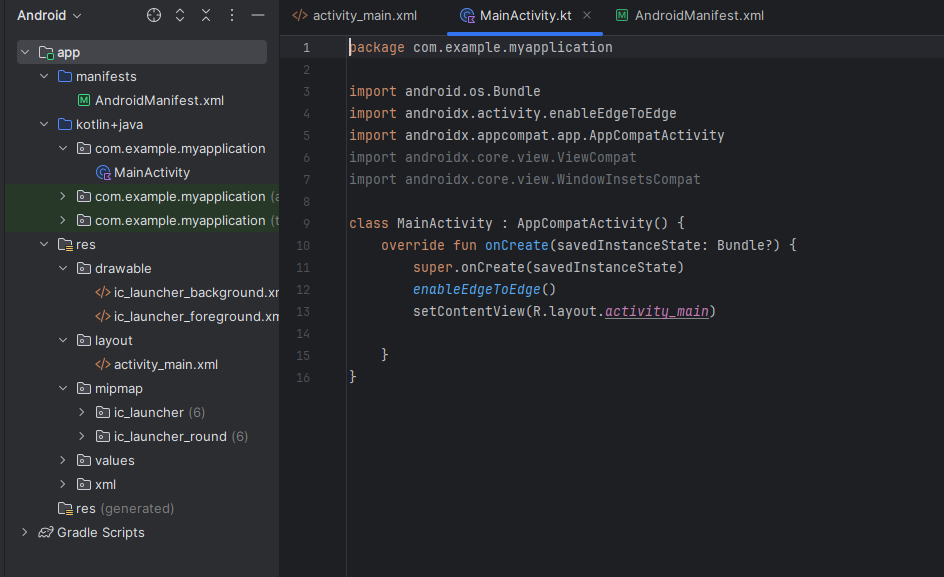
<activity  
 android:name=".MainActivity"  
 android:exported="true">  
  
 <intent-filter>  
 <action android:name="android.intent.action.MAIN" />  
  
  
 <category android:name="android.intent.category.LAUNCHER" />  
 </intent-filter>  
</activity>

Esta declaración dentro de la actividad **MainActivity** le dice a Android que esta es la ventana principal desde donde arrancará la app cuando el usuario la selecciona en el lanzador de aplicaciones.

Además, en **MainActivity** se suele definir la interfaz de usuario mediante archivos XML de layouts, o también puede hacerse de forma programática, y se puede gestionar la interacción con la app utilizando eventos de botones, navegación, llamadas a APIs, o cualquier otra funcionalidad que defina el comportamiento principal de la app.

Es importante destacar que aunque **MainActivity** es el punto inicial, las aplicaciones pueden tener múltiples actividades o fragmentos que se cargan posteriormente. También se puede cambiar cuál ventana se muestra primero modificando el **AndroidManifest.xml** para apuntar a una actividad diferente con el filtro **MAIN** y **LAUNCHER.**

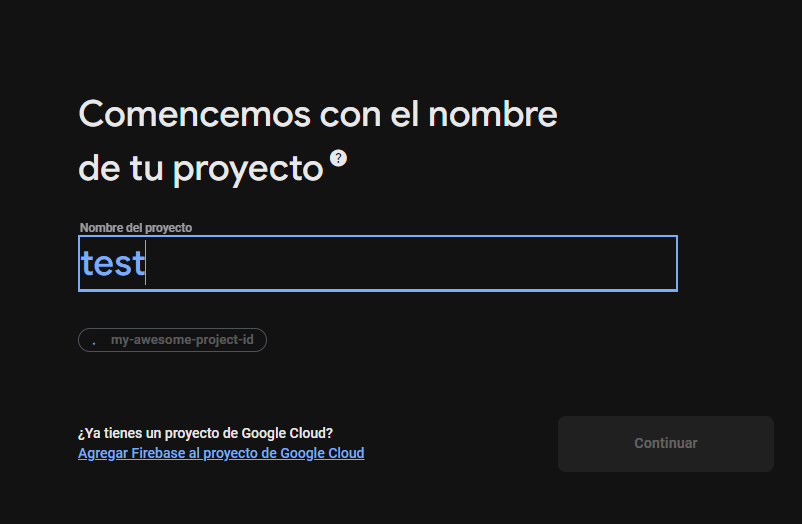
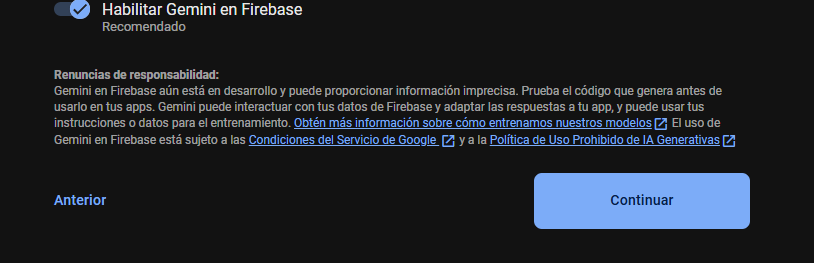
En resumen, **MainActivity** es la "ventana principal" y el punto de entrada fundamental donde comienza la ejecución de la app en Android Studio usando Kotlin. Controlar su comportamiento y configuración es esencial para definir cómo los usuarios experimentan la aplicación desde el inicio.



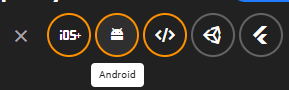
#### FIREBASE:



En la pagina inicial creamos un proyecto le ponemos el nombre de proyecto siguiente (Gemini Aceptamos integrar la Ia al proyecto)→ siguiente (se puede desmarcar//google Analytics-)→ Crear Proyecto



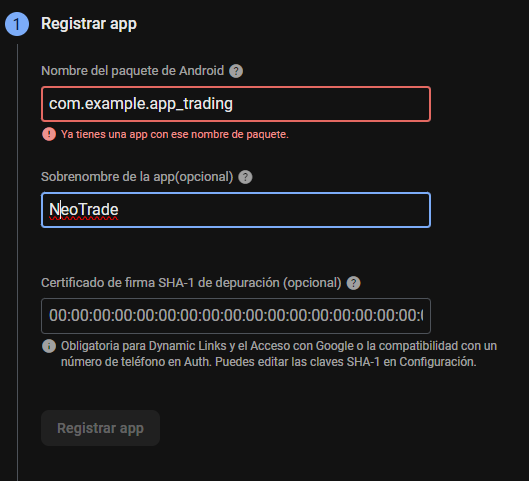
Le damos al botón que indica “Agregar app” elegimos nuestra plataforma que vayamos a implementar **Firebase**, en mi caso android stuido



Rellenamos los campos con el nombre de la carpeta del archivo encontrada en el **AndroidManifest.xml** o en el main (Ventana manifest)

<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 package="com.example.app\_trading">

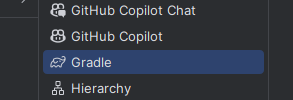
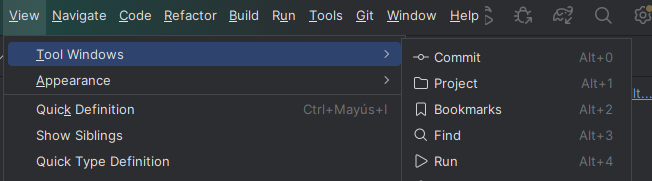
#### Registro de una app en fire base

 Rellenamos los datos que nos marcan como el nombre del parque de android y le nombre de la apliacion el certificado

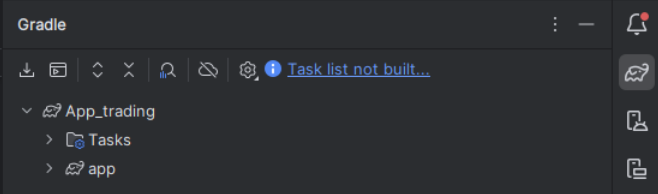
Para la firma deberemos ir a nuestro proyecto

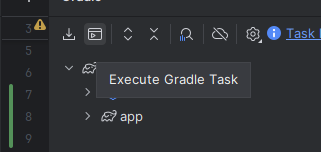


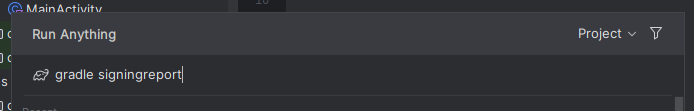
En la parte superior izquierda saldrá un desplegable que nos permitirá acceder a las opciones de view → tool Windows y abajo tendremos la opción de gradle



Le damos a gradle de ahí nos saldrá una ventana del gradle y en la ventana le damos al botón de terminal.



 Le damos al botón y ejecutamos el comando nos saldrá nuestra huella.



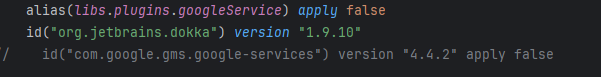
 Aquí tenemos nuestra huella digital

Seleccionamos el boton “**Descargar google-services.json**” y despues “**Siguinte**”



se nos creara un archivo llamado

con esta estructura se puede hacer de dos maneras

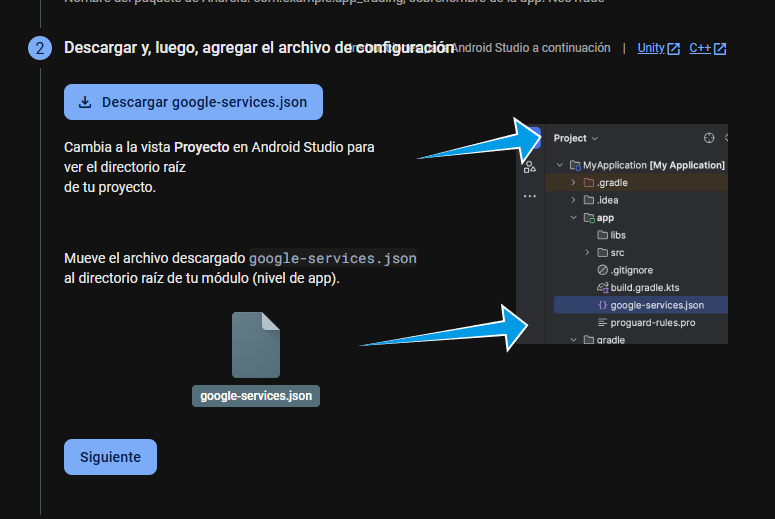


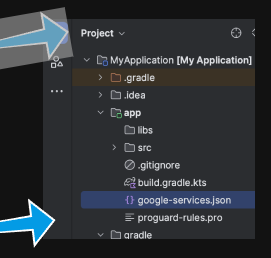
Si añadimos la linea de alias alias(*libs*.*plugins*.*googleService*) *apply* false

tendremos que añadir estas lineas de codigo:

googleServiceVersion="4.4.2"

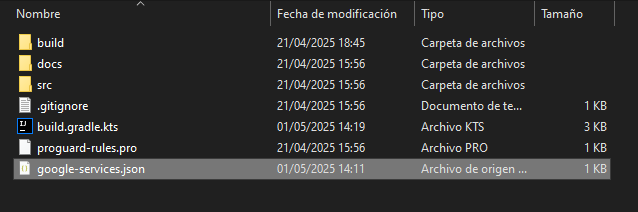
googleService={id="com.google.gms.google-services", version.ref="googleServiceVersion"}

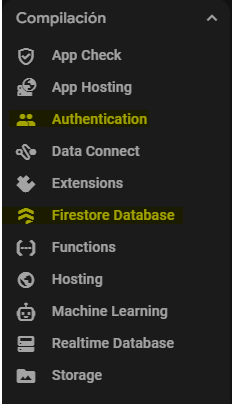




Cambia a la vista Proyecto en Android Studio para ver el directorio raíz de tu proyecto.

Mueve el archivo descargado google-services.json  
al directorio raíz de tu módulo (nivel de app)

 Seguimos el instructivo indicado por firebase

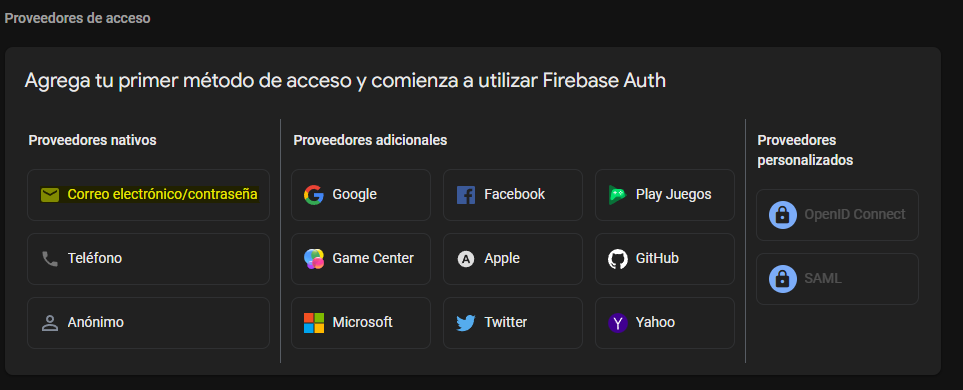


Ahora debemos meter estos módulos en Firebase

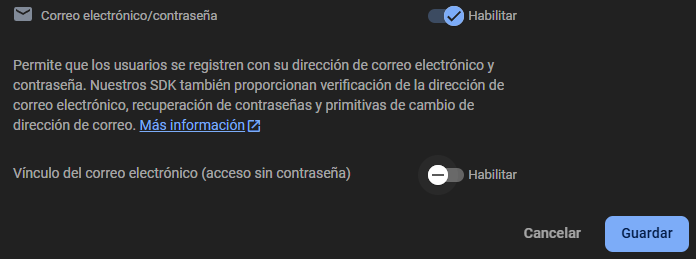
#### FIREBASE AUTHENTICATION

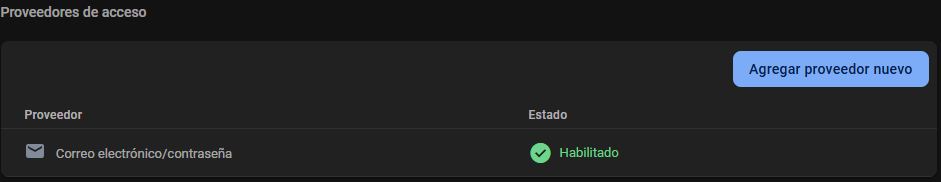
Le damos a comenzar

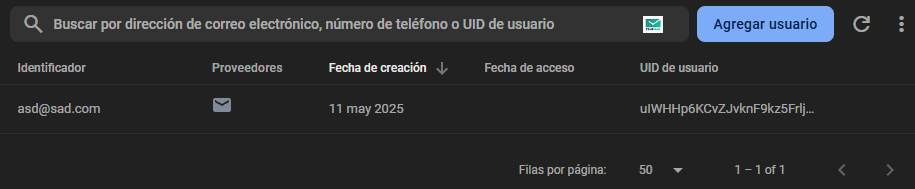
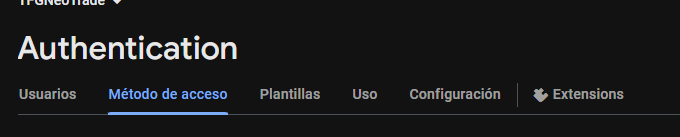
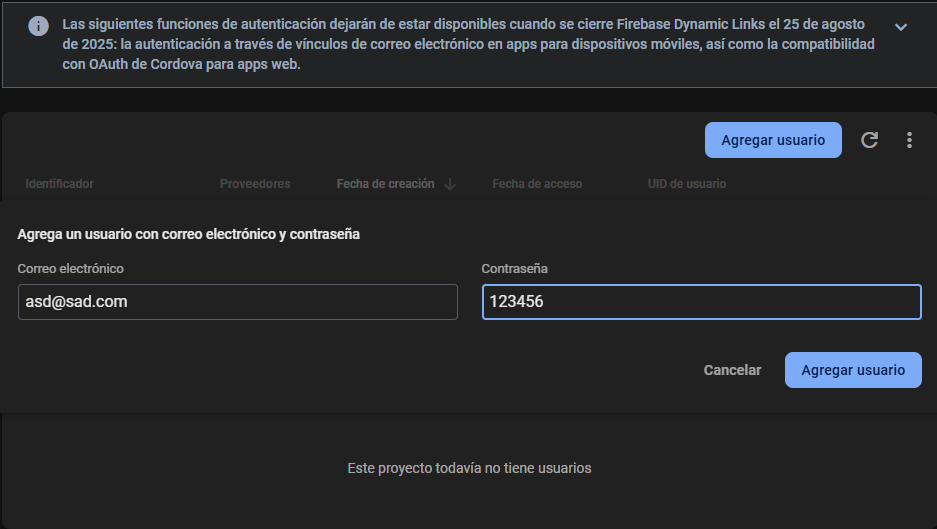
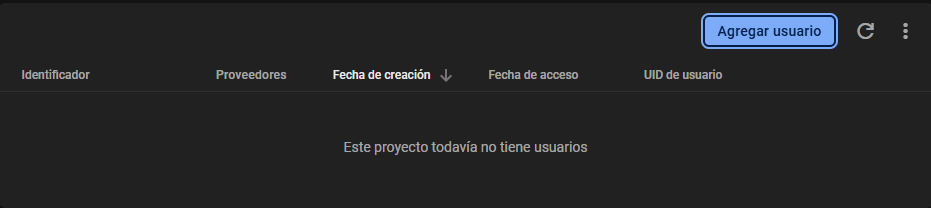
Ahora elegimos que proveedor queremos (Elegí Correo electrónico/contraseña) por ser el más a genérico



Le damos a correo y contraseña (Habilitar) → Guardar

Viculo del correo electrónico → nos permitirá hacer un registro sin contraseña:

Podemos seguir añadiendo nuevos proveedores o meter un nuevo usuario



Así podemos probar la aplicación con un usuario e ir a su documentación para poder ver las funciones que nos dan para inicio de sesión y registro.

Inicialización de Firebase Authentication

Antes de utilizar las funciones de autenticación, se debe inicializar el objeto FirebaseAuth en la actividad:

private lateinit var auth: FirebaseAuth  
  
override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_inicio\_de\_sesion*)  
  
 auth = Firebase.*auth* // Inicializa el sistema de autenticación  
}

Esto prepara el entorno para que podamos usar signInWithEmailAndPassword y createUserWithEmailAndPassword.

Registro de Usuario

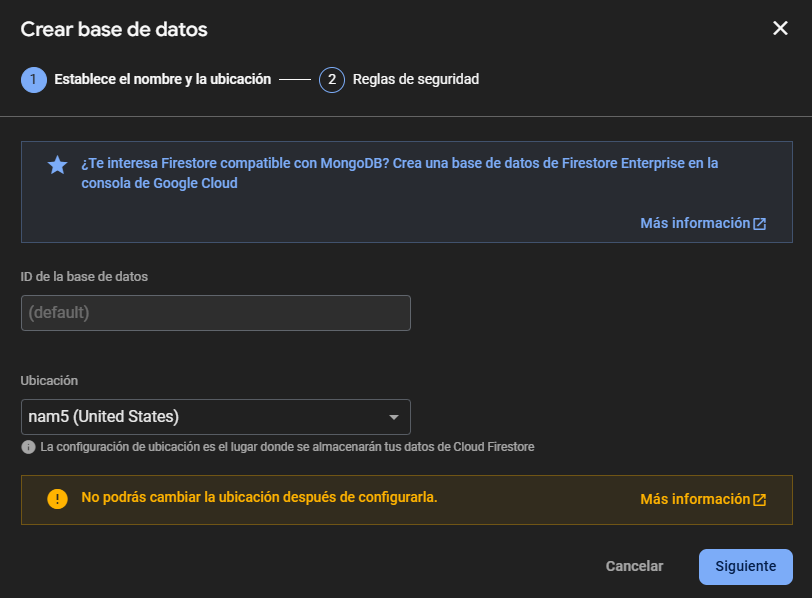
Para registrar a un nuevo usuario, se utiliza el siguiente método:

private fun crearCuentaFirebase(correo: String, contrasena: String) {  
 FirebaseAuth.getInstance().createUserWithEmailAndPassword(correo, contrasena)  
 .addOnCompleteListener **{** task **->** if (task.*isSuccessful*) {  
 Toast.makeText(this, "Registro exitoso", Toast.*LENGTH\_SHORT*).show()  
 val intent = Intent(this, MainActivity::class.*java*)  
 intent.putExtra("correo", task.*result*.*user*?.*email*)  
 intent.putExtra("Proveedor", "Usuario/Contraseña")  
 startActivity(intent)  
 } else {  
 Toast.makeText(this, "Error: ${task.*exception*?.message}", Toast.*LENGTH\_LONG*).show()  
 }  
 **}**}

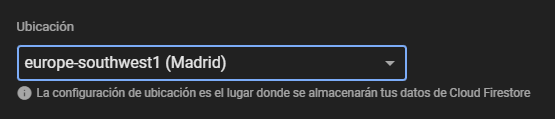
1. Se llama al método createUserWithEmailAndPassword.
2. Si la tarea (task) es exitosa, se muestra un mensaje de éxito y se redirige al usuario a la actividad principal.
3. Si falla, se muestra un mensaje con el error devuelto por Firebase.

#### FIREBASE STO

Le damos a comenzar.

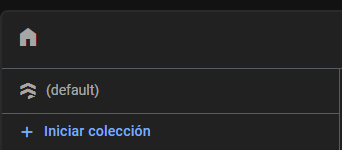


El id de la base de datos lo ponen ellos, y lo que debemos elegir es la ubicación yo puse la Europea-Madrid.

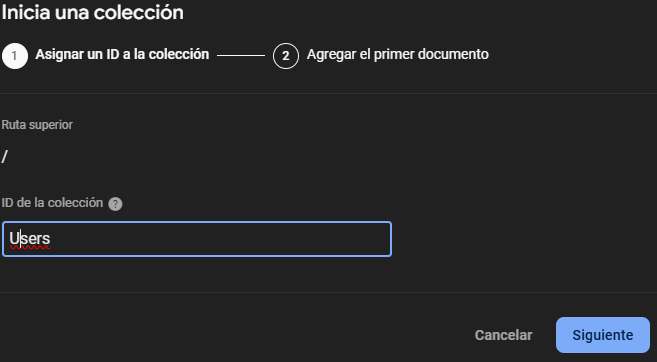
 Le damos a siguiente.



Creemos la base de datos, lo mas conveniente es la primera opción. Esperemos hasta que cargue.

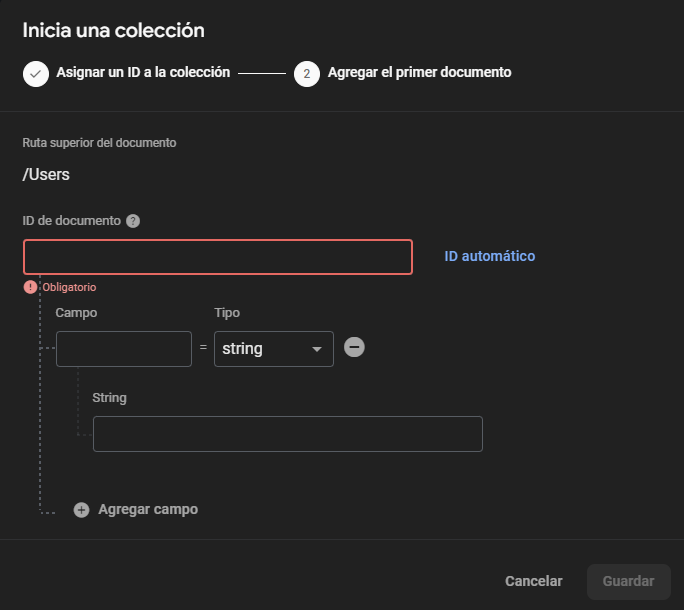
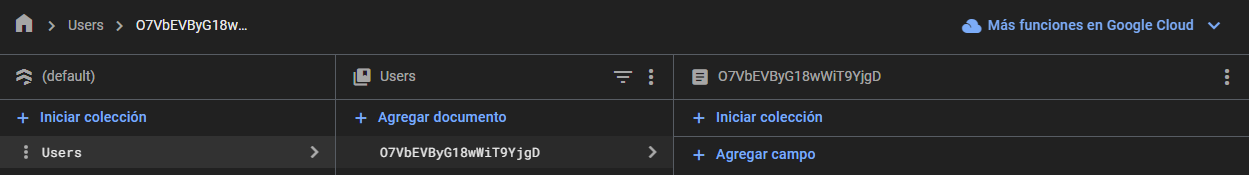
Creamos una nueva colección desde el botón de iniciar colección.

Ponemos el nombre de la colección



Aquí guardaremos datos de los usuarios.

Le damos a ID automático, y a guardar, los datos los iniciaremos desde la aplicación.



#### **API**

# Última Información de Precios

https://api.tiingo.com/tiingo/daily/<ticker>/prices

# Información Histórica de Precios

https://api.tiingo.com/tiingo/daily/<ticker>/prices?startDate=2012-1-1&endDate=2016-1-1&format=csv&resampleFreq=monthly

solicitudes de importación

encabezados = {

'Tipo de contenido': 'aplicación/json'

}

requestResponse = requests.get("https://api.tiingo.com/tiingo/daily/aapl/prices?startDate=2019-01-02&token=39184e3a30850b9362053181e27f780222ed0077", headers=headers)

imprimir(requestResponse.json())

# 6. Fases del desarrollo

## Imagen de salidaDescripción de cada fase

## ● Fase de diseño

**3.1. Fase 1: Análisis y Diseño**

**Fecha de inicio:** 04 de marzo de 2025  
**Fecha de finalización:** 13 de Mayo de 2025

**Actividades realizadas:**

* **Identificación de requisitos funcionales y no funcionales:**  
  Se comenzó con un análisis detallado del objetivo de la aplicación: crear una herramienta educativa que permita a los usuarios simular inversiones en bolsa, aprender conceptos básicos del trading y entender el funcionamiento de los mercados financieros. Se definieron los requisitos funcionales clave, como la simulación de compra/venta de activos, visualización de carteras, y acceso a contenido formativo. A su vez, se establecieron requisitos no funcionales como la usabilidad, la seguridad y la compatibilidad con distintos dispositivos Android.
* **Lluvia de ideas y estructuración de contenidos:**  
  A través de sesiones de brainstorming, se definieron los módulos principales de la aplicación: simulador, sección educativa, noticias del mercado y perfil de usuario. Se debatieron enfoques para equilibrar la utilidad con la simplicidad, enfocándose en usuarios con poca o ninguna experiencia en inversión.
* **Diseño de la arquitectura de la app y bocetos iniciales:**  
  Se diseñó la estructura lógica de la aplicación, incluyendo el flujo entre pantallas, navegación y estructura de datos. Los prototipos de baja fidelidad se realizaron en papel y posteriormente en Figma, reflejando la disposición inicial de las vistas clave.

📌 **[Aquí deberías insertar una imagen del prototipo de baja fidelidad de la interfaz principal]**

**3.2. Fase 2: Desarrollo Frontend y Backend**

**Fecha de inicio:** 8 de abril de 2024  
**Fecha de finalización:** 12 de mayo de 2024

**Actividades realizadas:**

* **Diseño de interfaces de usuario (UI):**  
  Se desarrollaron las pantallas principales utilizando **Kotlin** y **Android Studio**, garantizando una experiencia fluida tanto en orientación vertical como horizontal. Se diseñaron pantallas como el panel de inicio, operaciones bursátiles, historial de transacciones, perfil de usuario y contenido educativo.

📌 **[Aquí deberías insertar una imagen del diseño de la pantalla de simulación de compra/venta]**

* **Configuración del Backend con Firebase:**  
  Se utilizó **Firebase** para implementar la autenticación de usuarios, almacenar información de la cartera, registrar operaciones y alojar imágenes de perfil. Esta solución permitió una sincronización en la nube eficiente, reduciendo la complejidad del backend.
* **Integración con APIs externas:**  
  Para simular precios reales del mercado, se integró una API gratuita de datos bursátiles (como Alpha Vantage o Twelve Data), permitiendo a los usuarios realizar inversiones ficticias basadas en valores reales. Esta integración facilitó un entorno realista y educativo.

📌 **[Aquí deberías insertar una imagen del panel de usuario con su cartera de inversiones]**

**3.3. Fase 3: Pruebas y Depuración**

**Fecha de inicio:** 13 de mayo de 2024  
**Fecha de finalización:** 16 de mayo de 2024

**Actividades realizadas:**

* **Pruebas unitarias y de integración:**  
  Se verificó el correcto funcionamiento de los fragmentos individuales, como la autenticación, la carga de precios, la actualización de carteras y los test educativos. Se usaron herramientas de depuración integradas en Android Studio para identificar errores de lógica y problemas visuales.
* **Pruebas de experiencia de usuario (UX):**  
  Se realizaron pruebas con usuarios reales para obtener feedback sobre la facilidad de uso, comprensión del contenido y claridad de los procesos simulados. Este proceso permitió realizar ajustes finos en la disposición de botones, mensajes de ayuda y organización de contenidos.

📌 **[Aquí deberías insertar una imagen mostrando un test de funcionamiento de una operación de inversión simulada]**

* **Corrección de errores y mejora del rendimiento:**  
  Se solucionaron errores detectados durante las pruebas y se optimizó el uso de recursos, reduciendo el consumo de memoria en la carga de datos y mejorando los tiempos de respuesta de la app.

**3.4. Fase 4: Preparación para Lanzamiento**

**Fecha de finalización:** 16 de mayo de 2024

**Actividades realizadas:**

* **Generación del archivo APK:**  
  Una vez finalizado el desarrollo y validadas todas las funcionalidades, se generó el archivo APK final para su posible distribución mediante dispositivos físicos o la Play Store, cumpliendo con las políticas de Google en cuanto a permisos, dependencias y seguridad.
* **Verificación final:**  
  Se revisó el cumplimiento de los objetivos iniciales del proyecto. Se comprobó que la aplicación ofreciera una experiencia educativa efectiva, sencilla y motivadora para aprender a invertir. Se documentó todo el proceso de desarrollo y se prepararon las evidencias gráficas para su inclusión en la memoria final del TFG.

📌 **[Aquí deberías insertar una imagen de la app ejecutándose en un dispositivo físico (foto o captura de pantalla)]**

### Diagrama de la base de datos: entidades y relaciones.

Room (recomendado): Es una biblioteca de Android Jetpack que simplifica el uso de SQLite. Proporciona una API basada en anotaciones y es más fácil de usar que SQLiteOpenHelper.

#### Diagrama de casos de uso: interacción usuario-sistema.

#### Planificación de la interfaz: guía de estilos, mockups (hechos con GMA, a mano, Figma, etc.), y temas de la aplicación.

### ● Fase de desarrollo

* Explicación de las partes más complejas del código.
* Soluciones aplicadas a los problemas encontrados.
* Repositorio del código: [Enlace al repositorio, ej. GitHub]

### ● Fase de pruebas y depuración

* Pruebas realizadas destacadas.
* Errores encontrados y cómo se solucionaron.

### ● Fase de lanzamiento

* Consideraciones sobre la publicación de la aplicación (Play Store, Web, distribución, etc.)

### 7. Conclusiones

* Reflexión razonada sobre el proyecto.
* Puntos débiles y fuertes detectados.
* Posibles mejoras a futuro.

# 8. Bibliografía y referencias

[jsoneditoronline.org](http://jsoneditoronline.org/)

[console.firebase.google.com](http://console.firebase.google.com/)

[https://chatgpt.com](https://chatgpt.com/)

[https://github.com](https://github.com/)

rv

<https://www.youtube.com/watch?app=desktop&v=BQaxPwZWboA&t=410s>

<https://developer.android.com/develop/ui/views/layout/recyclerview?hl=es-419>

<https://www.youtube.com/watch?v=x31-sq3fQlM&t=105s>

Uso de la api en android studio

<https://homanhuang.medium.com/android-stock-app-retrofit-with-dagger-hilt-kotlin-coroutines-and-livedata-4d894df069fc#b803>

<https://www.youtube.com/watch?v=IiuKAmgRYeM>

<https://www.youtube.com/watch?v=dpURgJ4HkMk>

<https://www.youtube.com/watch?v=hS4qPXhru9M>

<https://firebase.google.com/support/release-notes/android>

<https://www.youtube.com/watch?v=LxABxtwhrDE>

<https://firebase.google.com/docs/auth/android/start?hl=es-419>

<https://www.youtube.com/watch?v=Mas3Lnd3WtM>

<https://www.youtube.com/watch?v=NFot9_bSFhw&t=436s>

[Android Kotlin future notification with worker with method from main activity - Stack Overflow](https://stackoverflow.com/questions/62623334/android-kotlin-future-notification-with-worker-with-method-from-main-activity)