

[**Alfonso\_Rincon\_IMDbAPP2.0**](https://github.com/alfonsaco/Alfonso_Rincon_IMDbAPP2.0)

**Alfonso Rincón Cuerva**

**2º DAM**

**Programación Multimedia y de Dispositivos Móviles**

**ÍNDICE**

[HERRAMIENTAS Y TECNOLOGÍAS 1](#_Toc179471611)

[PROBLEMA PLANTEADO 2](#_Toc179471612)

[SOLUCIÓN AL PROBLEMA 2](#_Toc179471613)

[FUNCIONES UTILIZADAS 2](#_Toc179471614)

[PRUEBAS REALIZADAS 8](#_Toc179471615)

[ESTRUCTURA DE LA APLICACIÓN 9](#_Toc179471616)

[EJECUCIÓN DE LA APLICACIÓN 15](#_Toc179471617)

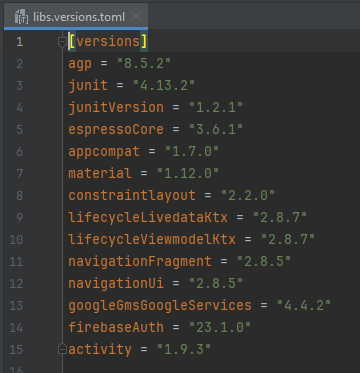
[COSAS QUE HE APRENDIDO 16](#_Toc179471618)

[BIBLIOGRAFÍA 17](#_Toc179471619)

# **LINK DEL PROYECTO:** https://github.com/alfonsaco/Alfonso\_Rincon\_IMDbAPP2.0

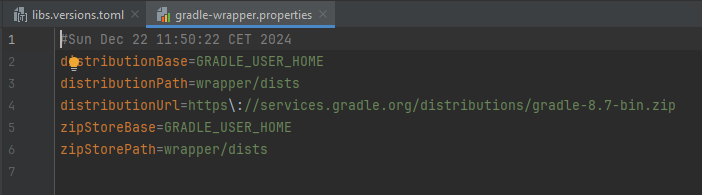
# **HERRAMIENTAS Y TECNOLOGÍAS**

**VERSIONES**

En el archivo **libs.versions.toml** encontramos versiones de diferentes bibliotecas y plugins del proyecto.

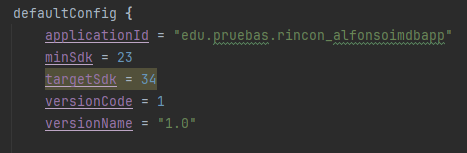
* **agp:** 8.5.2.
* **activity:** 1.9.3.
* **navigationFragment:** 2.8.5.
* **firebaseAuth:** 21.1.0.
* **googleGmsGoogleServices:** 4.4.2.

La versión de Gradle es de 8.7.



En el archivo ***build.gradle.kts***, podremos ver lo siguiente:

* **minSdk**: en esta aplicación es de 23. Esto significa que la versión de API mínima que soporta la app es la 23.
* **targetSdk**: es la versión de SDK para la cual está optimizada la app. En esta app, es de 34.



**EMULADORES**

* **Pixel API 30**: 5,0’’ 1080x1920 420dpi. ***System image***: R (API 30)
* **Pixel 5 API 30**: 6,0’’ 1080x2340 440dpi. ***System image***: R (API 30)
* **Pixel 6 API 30**: 6,4” 1080x2400 420dpi ***System image:*** R (API 30)

**PLATAFORMA**

El programa fue creado en la plataforma **Android Studio**, que es un entorno de desarrollo integrado oficial para crear aplicaciones Android. Incluye características como editor de código, diseñador de interfaces, emulador de Android y sistema de compilación ***Gradle***.

**LENGUAJES**

Los lenguajes utilizados para la creación de esta aplicación son los siguientes:

* **Java**: para la lógica de la aplicación e interactuar con los usuarios.
* **XML**: para agregar características y estilos a la aplicación.
* **Gradle**: es el sistema de compilación que utiliza Android.

# **PROBLEMA PLANTEADO**

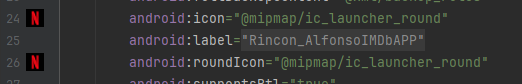
Vamos a expandir la aplicación de Rincon\_AlfonsoIMDbAPP, de forma que añadiremos diversas funcionalidades. Ahora, se guardará la información de los diferentes usuarios que la utilicen, se conectarán las bases de datos por la nube, se agregan distintas formas de Login, y funciones para editar usuarios.

# **SOLUCIÓN AL PROBLEMA**

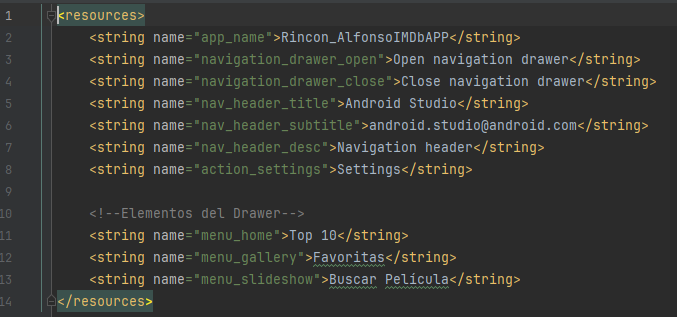
Utilizando como base la otra aplicación, se agregan funciones para el botón de Facebook, de inicio de sesión con contraseña, clases para sincronizar con el nube y clases para la nueva base de datos.

# **FUNCIONES UTILIZADAS**

**ICONO:** en el archivo AndroidManifest.xml, asignaré un nuevo icono para la aplicación. Este consiste en un diseño de una bici.



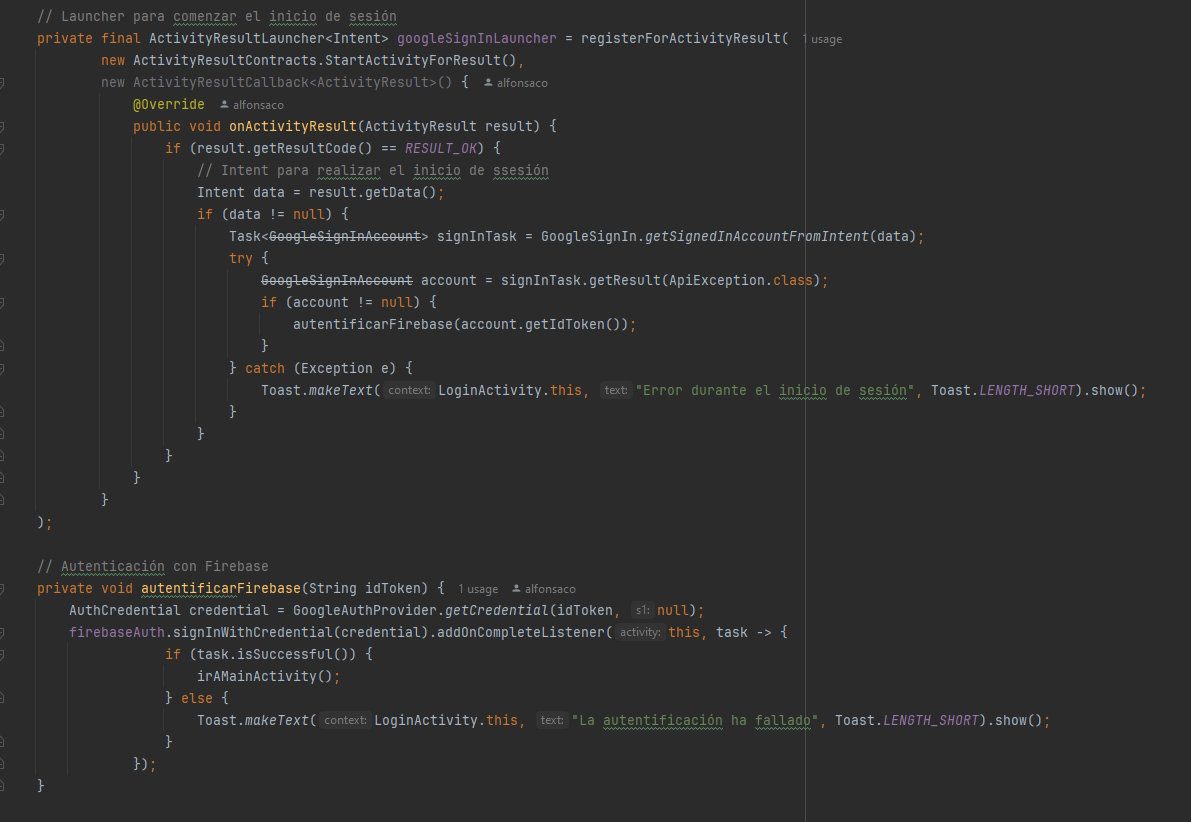
**VARIABLES PARA STRINGS:** tendrá diferentes textos que se usarán en diferentes XML.



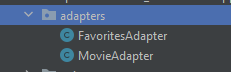
**ACTIVIDAD PRINCIPAL:** carga los dato del Email con el que ha iniciado sesión el usuario, mostrándolo en un nav. También contiene el acceso a los diferentes Fragments.



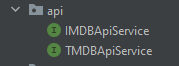
**LOGIN ACTIVITY:** esta Actividad es la primera que se ejecuta al abrir la aplicación. Nos mostrará un botón para iniciar sesión con Google, y el logo de la App. Para ello, utiliza ***Firebase***. Una vez que el usuario haya iniciado sesión, se le redigiriá a la MainActivity, donde podrá encontrar las películas.



**ADAPTERS:** estas dos claes se utilizarán para insertar los datos (Películas o series) dentro de los RecyclerView de las clases que lo utilicen. El de Favorites se utilizará para el Fragment de favoritos, mientras que el MovieAdapter se usa tanto en el Fragment de Top10, como en la actividad que muestra las películas buscadas. A su vez, el layout que estos Adapters utilizan son el ***item\_movie.xml*** *y* ***item\_favorite\_movie.xml*.**

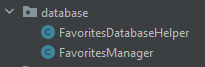


**APIS:** estas 2 clases son interfaces que utilizan la blblioteca Retrofit para poder obtener información de APIs externas. Estas utilizan las APIs IMDb y TMDB. Cada una utiliza unos Endpoints determinados, que le proporcionan un JSON con información específica.

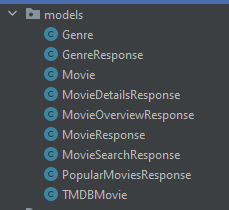


**DATABASES:** en esta carpeta tenemos 2 clases distintas. La primera, ***FavoritesDatabaseHelper***, es la que crea y gestiona la base de datos de SQLite, la cual almacena las películas favoritas del usuario. Distinguirá la ID del usuario, para así evitar que los distintos usuarios compartan la misma lista de películas favoritas.

Después, tenemos ***FavoritesManager***, la cual es la clase que FavoritesDatabaseHelper utiliza para poder interactuar con la base de datos. Esta clase tendrá todas las funciones de añadir, obtener, y eliminar las películas de la base de datos, además de una función para hacer logs y así poder ver todas las películas que hay. Esto se creó para poder observar el corecto funcionamiento de la aplicación durante su creación. Por último tenemos ***UserDatabaseHelper,*** que es la clase que crea la base de datos de los usuarios.

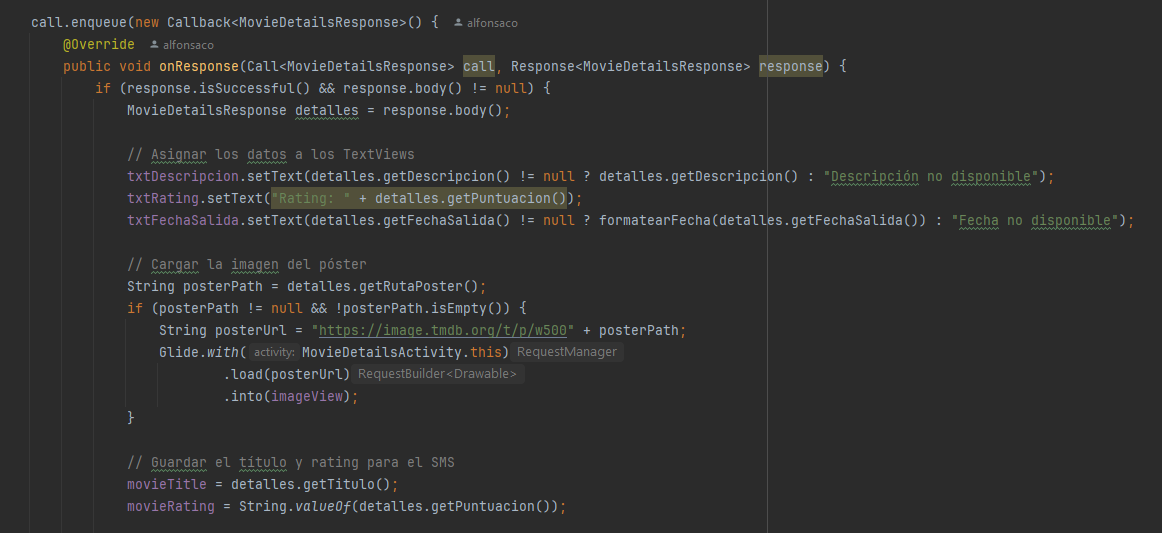


**MODELS:** todas estas clases, sirven para poder mapear los JSON de las API. Cada una mapea algo distinto. La clase ***Genre*** y ***GenreResponse*** sirven para poder obtener los géneros que están disponibles en la API de TMDB. ***Movie*** contendrá todas las funciones para poder hacer Parceables de las películas, y así poder transferir sus datos entre clases como objetos. El resto, sirven para mapear los distintos Endpoints de las API.



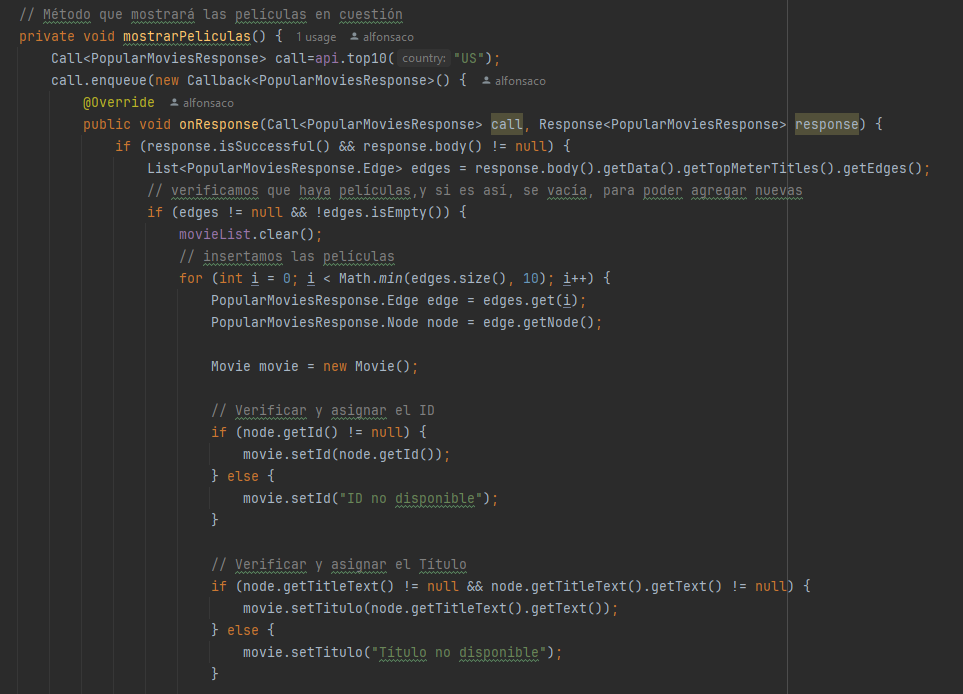
También tenemos los nuevos modelos de ***Usuarios*** y ***UsuarioDAO,*** que son los que nos van a ayudar a gestionar cuando se añaden los usuarios a la base de datos, a la externalizada

**MOVIEDETAILS:** esta clase es la que mostrará la película que ha sido seleccionada por el usuario. En la clase ***MovieAdapter*** se añaden eventos de Click y LongClick a cada una de las películas para poder ver más información de estas, así como añadirlas a la base de datos. Pues esta es la actividad que muestra los datos al hacer Click en ellas. Utiliza el layout ***activity\_movie\_details.xml***, y también tiene un botón para poder enviar información de una película en cuestión a un contacto, a través de SMS.



**MOVIELIST:** esta clase mostrará las películas que han sido buscadas por los usuarios, y filtradas por año y género. Al igual que en los Fragments, tendrán las funciones de Click y LongClick, ya que también utiliza el ***MovieAdapter***. Se llama a esta clase desde el SlideshowFragment.



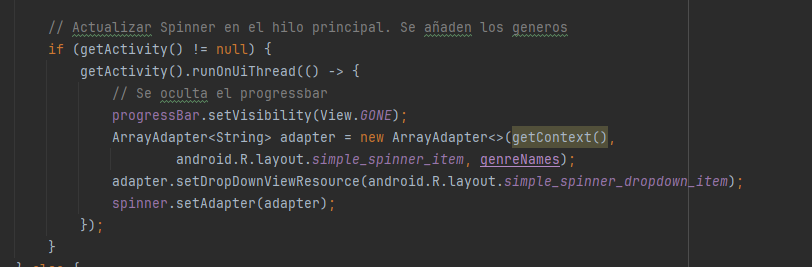


**HOMEFRAGMENT:** es el Fragment principal, el cual contiene el Top 10 de películas.

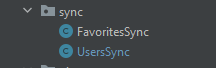
Las películas se añaden a través de un RecyclerView, el cual obtiene las películas gracias a la API de IMD, utilizando el Endpoint de get-top-meter.

**SEARCHFRAGMENT:** este Fragment es el que cargará todas las películas favoritas de cada uno de los usuarios, y las mostrará. Las películas que estén agregadas aquí, tendrán la misma función de Click que las otras, pero su LongClick será diferente, ya que, en esta, provocará que la película se elimine. También tiene un botón que permite compartir la película vía bluetooth, mostrando un diálogo con un JSON.

**SLIDESHOWFRAGMENT:** en esta clase, el usuario puede insertar el año en el que salió la película y el género. Este último estará dentro de un Spinner, que se llena con los géneros que ofrece a API de TMDB. Habrá limitaciones en el año, para que no se puedan insertar años inválidos.



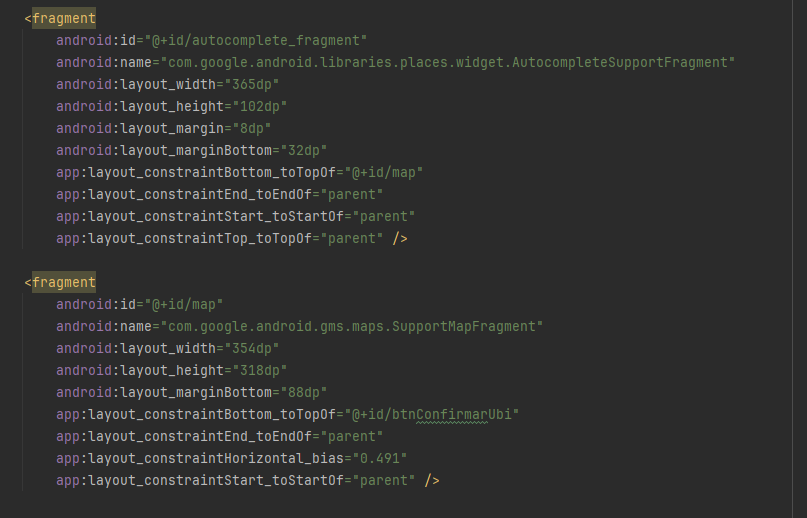
**SYNC:** esta carpeta contiene las 2 clases que se usarán para externalizar las bases de datos. Estas son FavoritesSync y UsersSync, que enviarán los datos de la base de datos local a las colecciones creadas en el Cloud de Firebase. FavoritesSync tendrá funciones para poder agregar y eliminar las películas en la base de datos externalizadas.



**EDITUSERACTIVITY:** es una Activity que servirá para que el usuario pueda editar sus datos, como su nombre, email, número de teléfono y dirección.

**KEYSTOREMANAGER:** esta clase es la que se encargará de obtener los datos, cifrarlos y guardarlos, para que en la base de datos se vean cifrados, pero luego nos de su verdadero valor.

**UBICACIONACTIVITY:** es la que usaremos para permitir al usuario utilizar el Google Maps, y así poder obtener su ubicación. Para funcionar, contiene 2 componentes de tipo Fragment, que se deben implementar en el Gradle.



# **PRUEBAS REALIZADAS**

Se realizan diversas pruebas de caja negra para verificar la funcionalidad del programa. Tras ello, se programan sus respectivas soluciones:

1. **DISTINTAS RESOLUCIONES:**

Se ha probado la aplicación en pantallas con diferentes resoluciones, además de en dispositivos móviles reales. En todos funciona correctamente.

1. **DISTINTOS INICIOS DE SESIÓN**

Se ha iniciado sesión con usuarios diferentes, de diferentes formas.

# **ESTRUCTURA DE LA APLICACIÓN**

**api:** contiene interfaces de las APIs de IMD y TMDB, para lograr conectar realizar las funciones en cuestión.

**models:** aquí hay numerosas clases de Mapeo, que sirven para obtener los datos de los JSON de las API, además de clases para crear objetos.

**adapters:** contiene todas las clases que se utilizan para los RecyclerViews.

**sync:** contiene las clases para lograr la externalización de la base de datos en el Cloud.

**ui:** contiene las clases de los 3 Fragments.

**drawable**: aquí están todos los XML, imágenes y vectores.

**mipmap**: dentro de esta carpeta están los archivos del icono de la aplicación.

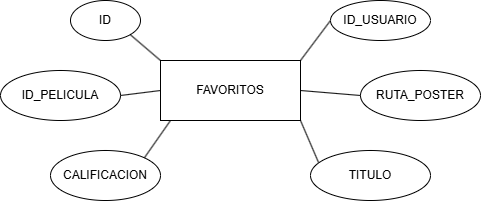
**menu:** aquí están los estilos de los elementos que hay en el menú del Main.

**values**: en sus archivos ***colors.xml*** y ***string.xml*** se encontrarán todas las variables de colores y strings que hemos creado.

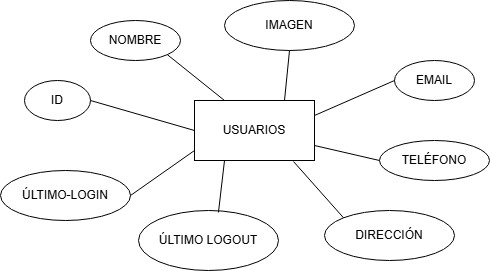
**navigation:** contiene los estilos de los Fragments.

**ESQUEMAS DE LA BASE DE DATOS**

Esta base de datos está compuesta dos tablas. La primera, favoritos, recopilará datos de la película, al igual que el id del usuario, para así poder diferenciar entre las preferencias de un usuario y otro.

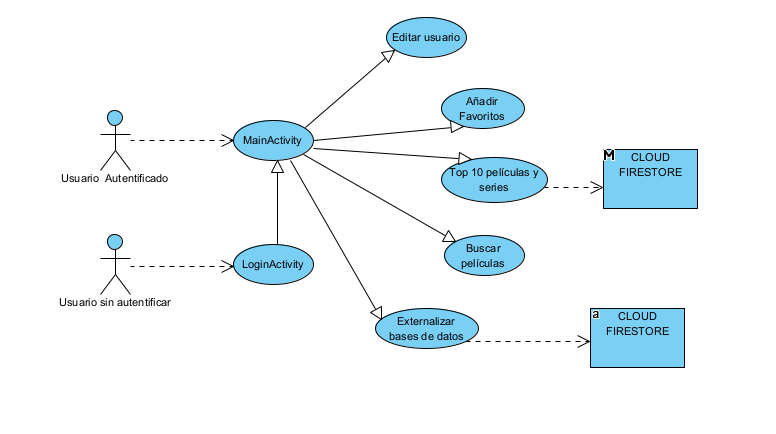
****

Esta segunda tabla es la de usuarios, que tendrá los datos de cada usuario, además de su login y logout.



**DIAGRAMAS UML**

**CASOS DE USO**



**DIAGRAMA DE CLASES**

DIAGRAMA COMPLETO

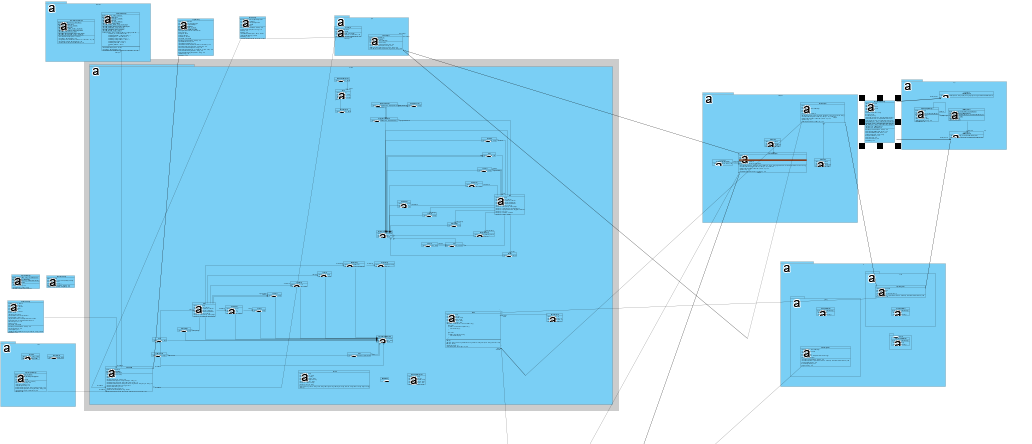
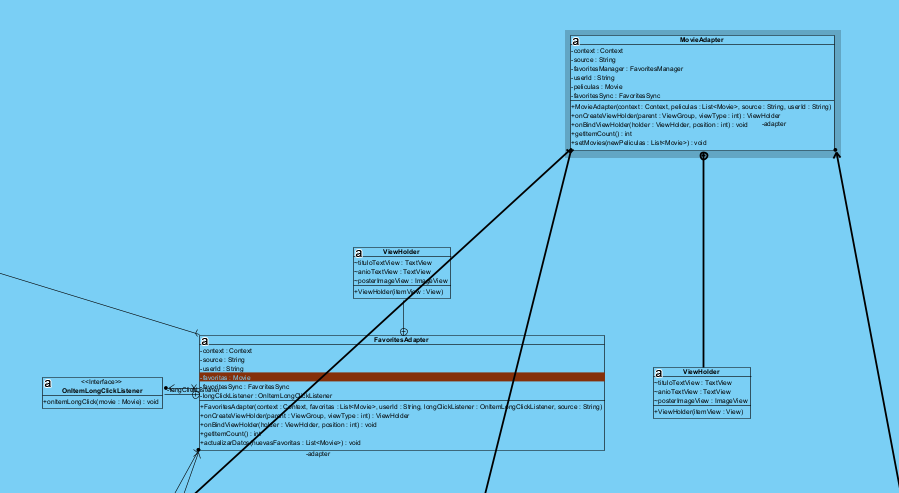
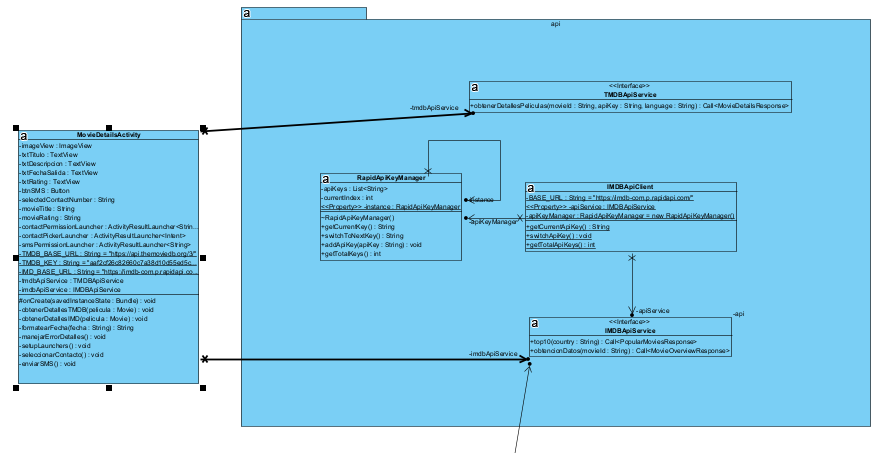
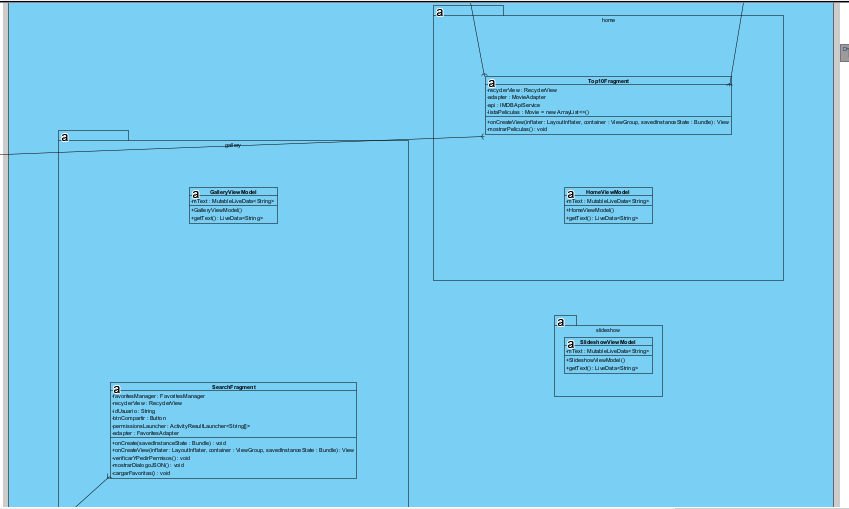
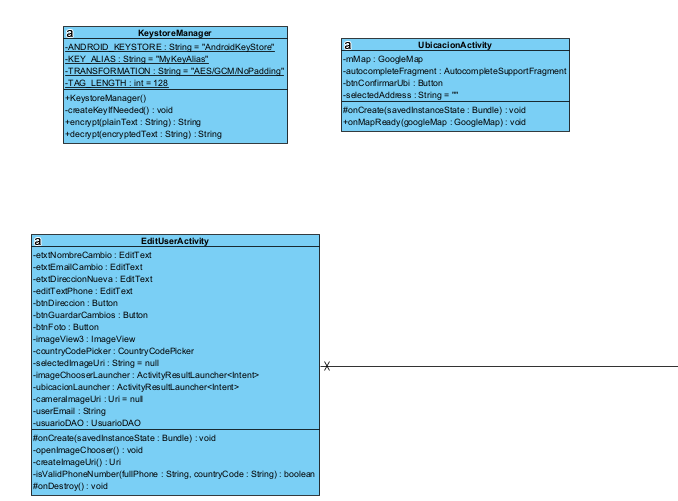
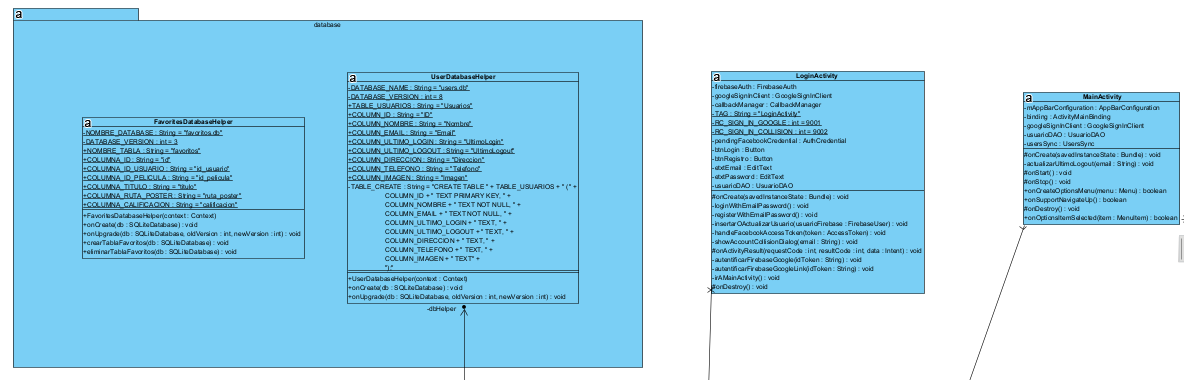


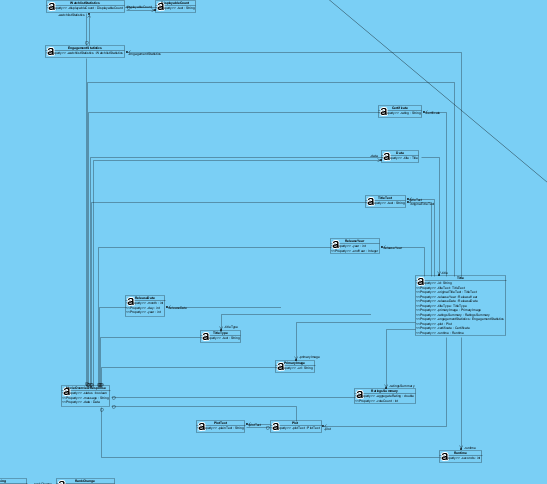
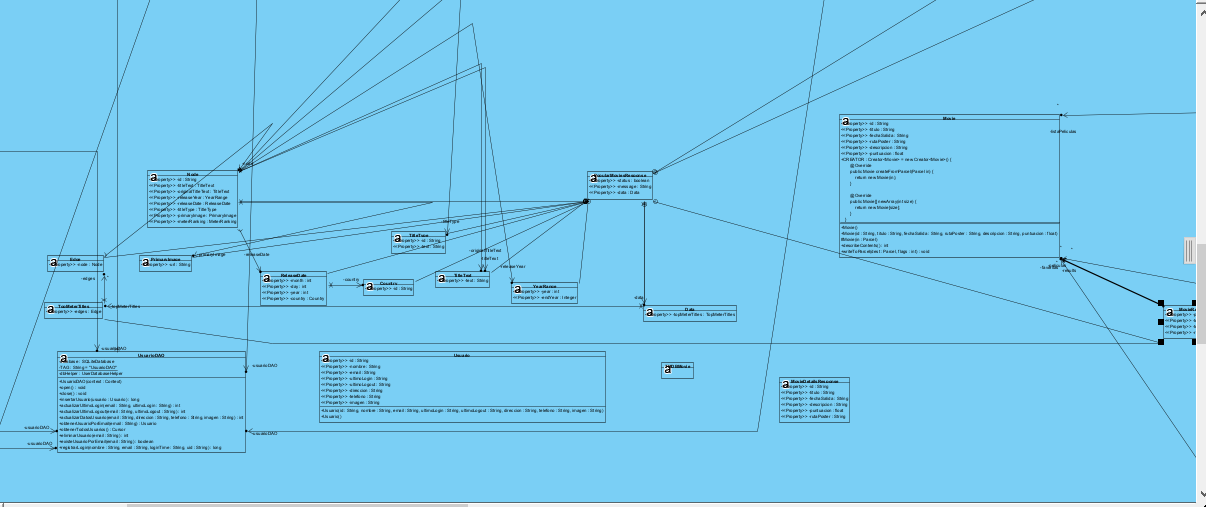
DIAGRAMA POR PARTES





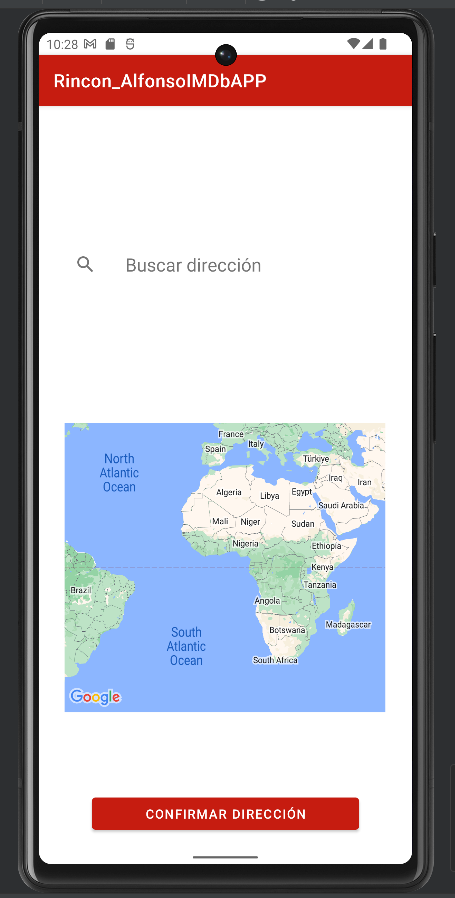
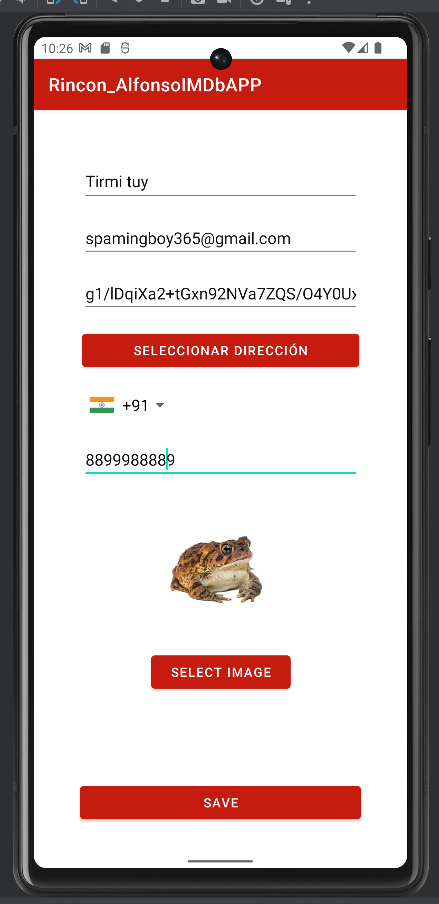
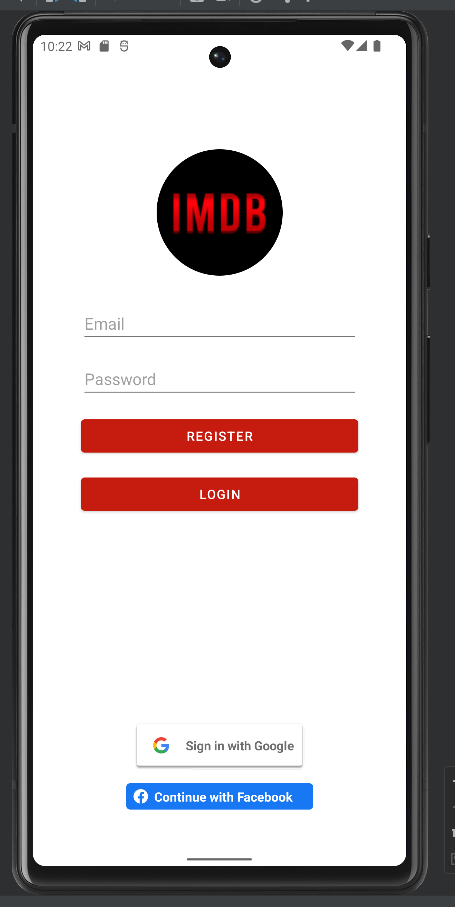


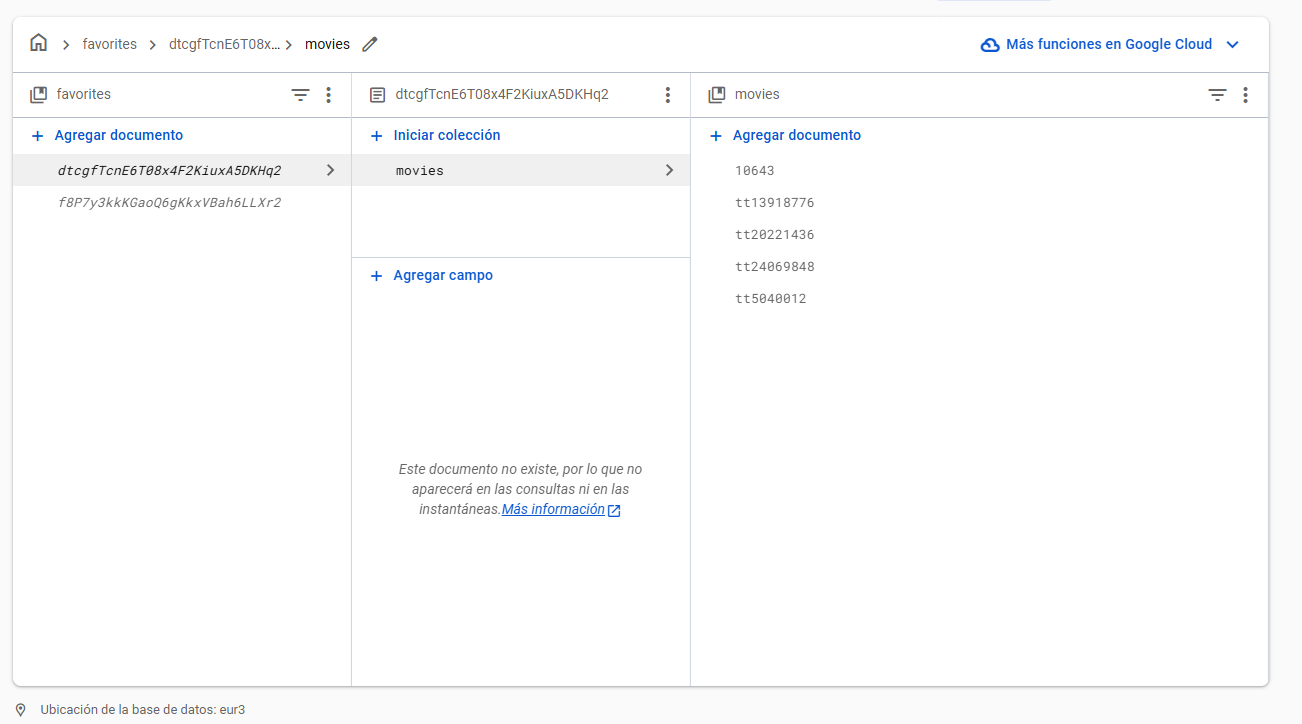




# **EJECUCIÓN DE LA APLICACIÓN**

**LOGIN ACTIVITY EDIT USER FAVORITOS FRAGMENT**



**FIREBASE CLOUD**

# **COSAS QUE HE APRENDIDO**

1. Utilizar Cloud Firestore.
2. Conectar Facebook con Firebase.
3. Agregar un botón de Facebook a Android Studio.

# **BIBLIOGRAFÍA**

MoureDev by Brais Moure. (2020, 1 mayo). *LOGIN Android Studio (Google Sign in) 🔥 FIREBASE Login Android [2020]* [Vídeo]. YouTube. https://www.youtube.com/watch?v=xjsgRe7FTCU