****

**ÚLTIMOS CAMBIOS REALIZADOS**

**Contenido**

[**1.** **REVIEWDAO** 2](#_Toc212549758)

[**2.** **LOGINACTIVITY** 3](#_Toc212549759)

[**3.** **REVIEWADAPTER** 4](#_Toc212549760)

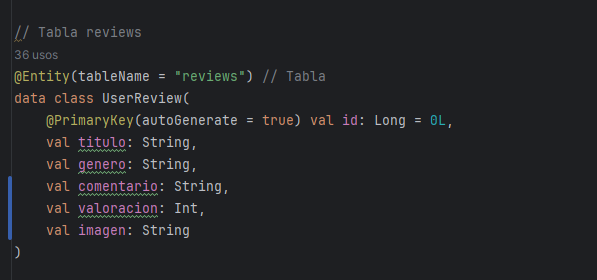
[**4.** **PROFILEACTIVITY** 6](#_Toc212549761)

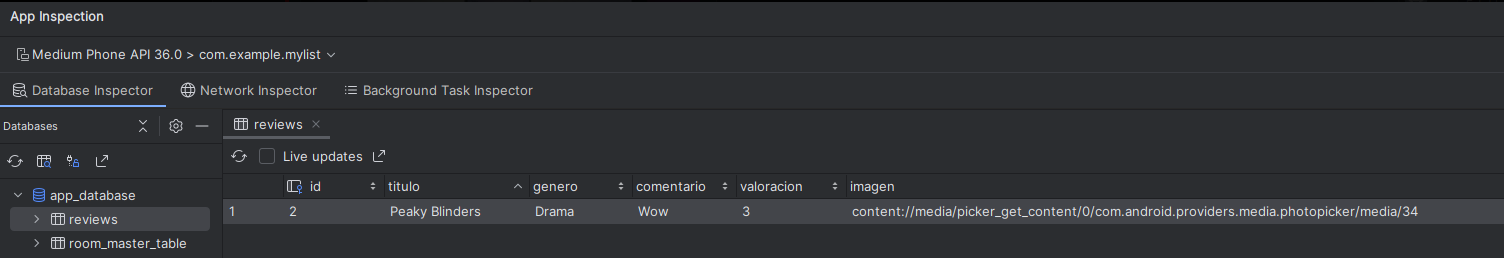
[**5.** **MAINACTIVITY** 8](#_Toc212549762)

[**6. DEPENDENCIAS** 15](#_Toc212549763)

[**7. AndroidManifest.xml** 15](#_Toc212549764)

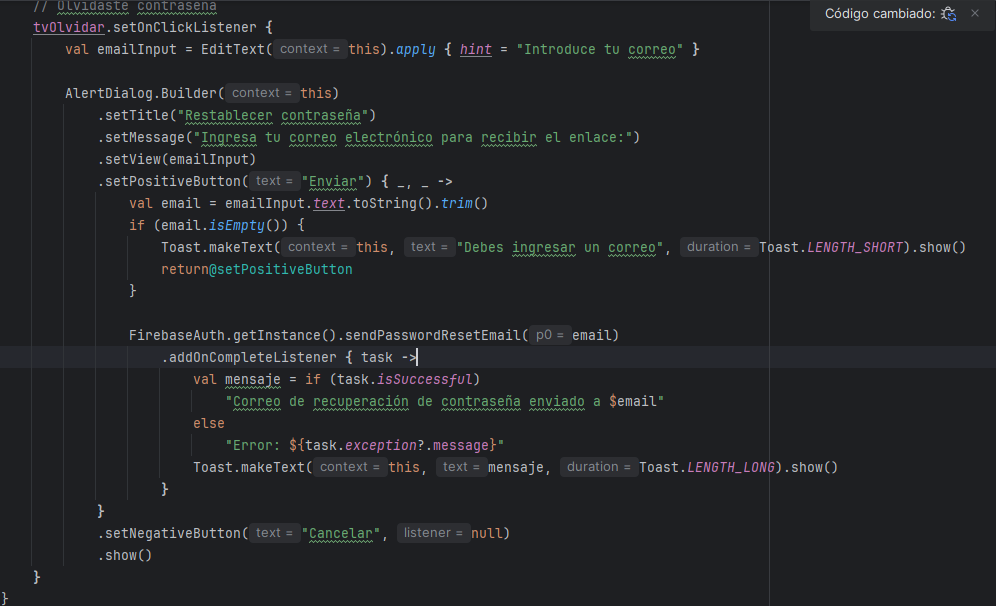
# **REVIEWDAO**

* val valoracion: Int, (guardar estrellas)
*  val imagen: String (guardar url de imagen “local”)
* Ahora se guarda en la base de datos con la valoración (RatingBar) y la imagen (url) que está guardada localmente en la memoria del móvil.



# **LOGINACTIVITY**

* Olvidaste tu contraseña



Al hacer clic en el texto **¿Has olvidado tu contraseña?**, se abre un pequeño diálogo en el que los usuarios pueden restablecer su contraseña de **Firebase**.

En este diálogo se muestra un título: **“Restablecer contraseña”**, un mensaje con el texto **“Ingresa tu correo electrónico para recibir el enlace”**, y un campo de entrada con el hint **“Introduce tu correo”**, además de dos botones: **Enviar** y **Cancelar**.

El correo introducido no puede estar vacío y tiene que tener la estructura de prueba@prueba.com

Una vez enviado, se enviará automáticamente un correo con un enlace para cambiar la contraseña, y aparecerá un pequeño toast indicando que el correo ha sido enviado correctamente.



# **REVIEWADAPTER**

* ratingBar: Muestra las estrellas de las valoraciones de las reseñas.



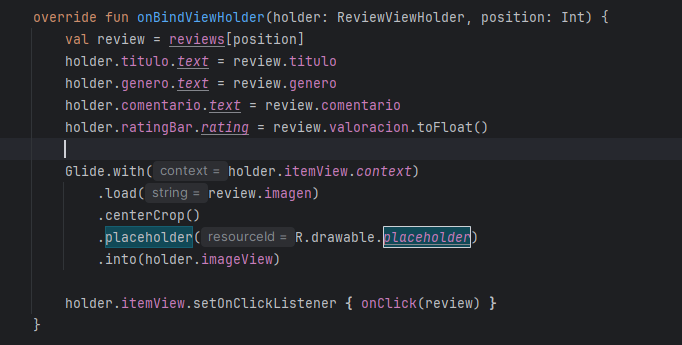
* imageView: Muestra la imagen guardada en la galería.



* **onBindViewHolder()**

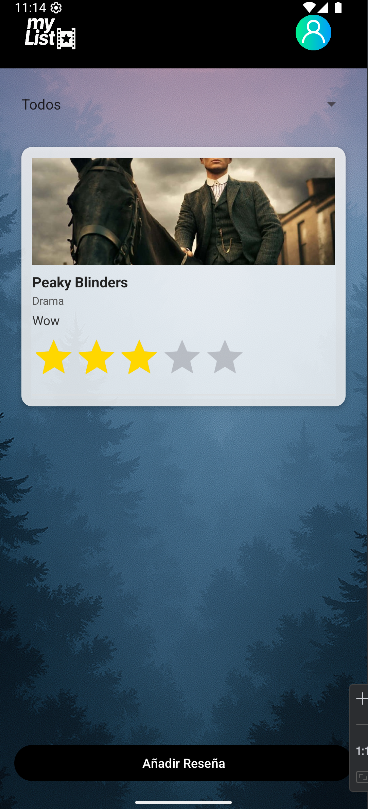
Asigna los valores del objeto UserReview a los TextView del ítem, y configura el evento setOnClickListener para detectar clics sobre cada reseña y ejecutar la acción pasada como parámetro.

El código carga la imagen de review.imagen en el ImageView con Glide y asigna un clic al elemento para ejecutar onClick(review).



Layout del diseño de la reseña (**item\_resena.xml**):



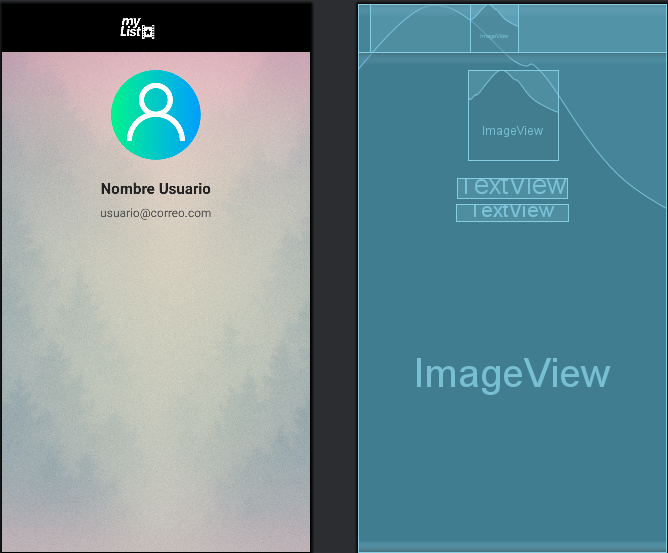


# **PROFILEACTIVITY**

**Qué hace:** Muestra la información del usuario, como su nombre y correo, y permite volver a la pantalla anterior (MainActivity).

**Layout asociado (diseño):** activity\_profile.xml





**Librerías:**

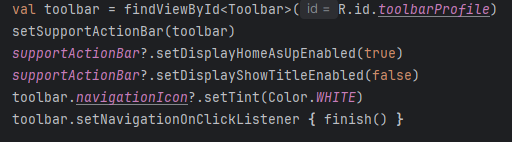
* **import android.graphics.Color:** Permite usar y manipular colores en la interfaz de Android.
* **import android.os.Bundle:** Proporciona una forma de pasar datos entre actividades o guardar el estado de la app.
* **import androidx.appcompat.app.AppCompatActivity:** Permite crear actividades que sean compatibles con versiones antiguas de Android.
* **import androidx.appcompat.widget.Toolbar:** Permite usar una barra de herramientas personalizable en la interfaz.
* **import android.widget.TextView:** Permite mostrar texto en la pantalla de la app.
* **import com.google.firebase.auth.FirebaseAuth:** Proporciona funcionalidades para autenticación de usuarios con Firebase

**Componentes:**

* FirebaseAuth auth: Instancia para manejar la autenticación de usuarios.



* Toolbar toolbar: Barra superior con botón de volver.



* TextView textName: Muestra el nombre del usuario extraído del correo.

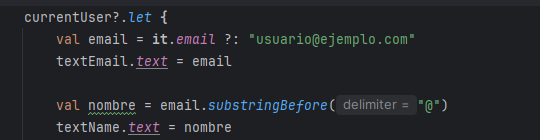


* TextView textEmail: Muestra el correo electrónico del usuario.



**Flujo:**

* Al iniciar la actividad (onCreate):
* Se obtiene la instancia de FirebaseAuth y el usuario actual (currentUser).
* Se configura la Toolbar con botón de volver y color del ícono.
* Se obtienen los TextView de nombre y correo.
* Si el usuario está logueado:
* Se muestra su correo en textEmail, y se extrae el nombre del usuario a partir del correo (todo lo que está antes de “@”) y se muestra en textName.



* Botón de volver de la Toolbar: cierra la actividad (finish()).



# **MAINACTIVITY**

* **ImageProfile:** Icono del perfil en la pantalla principal.

Al hacer click, muestra un **PopupMenu** con dos opciones:

**Ver perfil:** abre **ProfileActivity** para mostrar la información del usuario.

**Cerrar sesión:** cierra la sesión del usuario con **auth.signOut().** Redirige a **LoginActivity** y cierra MainActivity.

Texto

El contenido generado por IA puede ser incorrecto.

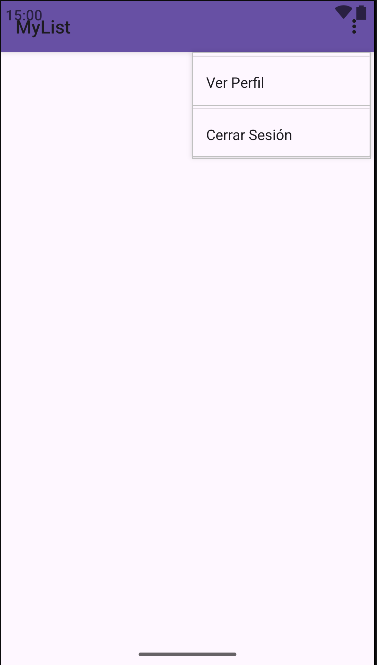
Imagen real del usuario se carga con Glide desde Firebase (currentUser.photoUrl).

Al pulsarla, se abre el PopMenu (ver perfil / cerrar sesión).

Texto

El contenido generado por IA puede ser incorrecto.

El menuInflater que es el popupmenu se obtiene de res/menú/profile\_menu.xml.



Como se vería dentro de la aplicación:

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

El contenido generado por IA puede ser incorrecto.

* **Filtrado de reseñas:**

Utiliza lifecyclopeScope y Dispatchers.IO para acceder a la base de datos sin bloquear el hilo principal, y los resultados se muestran en la interfaz a través del adaptador.

Texto

El contenido generado por IA puede ser incorrecto.

Añadir una nueva reseña:

**Texto

El contenido generado por IA puede ser incorrecto.**

Muestra un cuadro de diálogo con campos para título, género, comentario y valoración,

al confirmar, se crea un objeto UserReview y se inserta en la base de datos mediante reviewDao.insert(reseña).

* Editar o eliminar una reseña existente:

Texto

El contenido generado por IA puede ser incorrecto.

Carga los datos existentes en un diálogo editable, permite guardar los cambios, eliminar la reseña o cancelar la operación, y estas modificaciones se actualizan en la base de datos y se refresca el listado.

*val generos = resources.getStringArray(R.array.generos\_array).drop(1)*

*val adapter = ArrayAdapter(this, android.R.layout.simple\_spinner\_item, generos)*

*adapter.setDropDownViewResource(android.R.layout.simple\_spinner\_dropdown\_item)*

*generoSpinner.adapter = adapter*

**Spinner dentro del diálogo**: permite que el usuario seleccione un género para su reseña.

**drop(1)**: elimina la primera posición del array si es un título o “Selecciona género”.

**Adapter**: transforma el array en opciones visibles en el Spinner.

**Se sincroniza con el RatingBar y la imagen**, completando la reseña antes de guardarla en la base de datos.

* Seleccionar de una lista fija de géneros:

Evitando errores de escritura.

Filtrar la lista de reseñas por género.

Mantener consistencia en la base de datos.

Texto

El contenido generado por IA puede ser incorrecto.

**Array de géneros:** defines todos los géneros que quieres mostrar.

**Adapter:** convierte el array en un formato que el Spinner puede mostrar.

**Listener:** cuando el usuario selecciona un género, se llama a filtrarLista() para actualizar las reseñas mostradas.

**“Todos”**: opción que elimina el filtro y muestra todas las reseñas.

**Antes:** input manual -> propenso a errores

**Ahora:** Spinner con lista de géneros -> selección fácil, filtrado automático, coherencia en la base de datos**.**

Los géneros del spinner se obtienen de un array que se encuentra en res/values/strings.xml.

Un texto con letras negras

El contenido generado por IA puede ser incorrecto.

**Valoración visual:**

-Se añade un **RatingBar** en los diálogos de añadir o editar reseña.

- Permite al usuario seleccionar una puntuación visual (por ejemplo, 1 a 5 estrellas).

- Se guarda la puntuación en la base de datos (valoracion en UserReview).

- Se integra con la imagen y el Spinner de género para completar la reseña.

Aquí obtienes la referencia al RatingBar en el layout del diálogo (dialog\_resena.xml).

El usuario puede tocar y seleccionar el número de estrellas.

****

* **Guardar la puntuación:**

ratingBar.rating devuelve un Float.

Se convierte a Int y se guarda en la reseña.

**Texto

El contenido generado por IA puede ser incorrecto.**

Cuando editas, se carga la puntuación anterior para que el usuario pueda modificarla.

Luego se guarda de la misma forma que al añadir.

****

**Antes:** no existía un control visual para la valoración, se podía solo con texto.

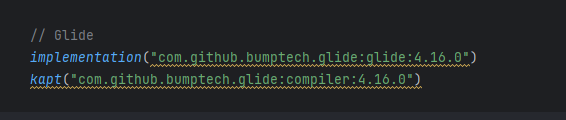
**Ahora:** hay estrellas (RatingBar) -> más intuitivo y visual.

Se integra con el resto de campos de la reseña: título, comentario, género, imagen.

**Base de datos:** cada reseña ahora tiene un campo valoración con la puntuación seleccionada.

**Editar:** la puntuación previa se carga para modificarla.

# **6. DEPENDENCIAS**

- Glide: Es una librería de Android que se usa para cargar, mostrar y manejar imágenes de forma eficiente desde URLs, recursos o archivos locales en tus aplicaciones.

# **7. AndroidManifest.xml**

Definimos la **actividad** (ProfileActivity) dentro de la aplicación.

