



Proyecto App Móvil: “La Cueva del Cinéfilo”

¿De qué se trata nuestra app?

La Cueva del Cinéfilo es una app conformada por un catálogo de películas en las cuales el usuario podrá leer la descripción o sinopsis de la misma, ver sus especificaciones, dar un puntaje y agregar a una lista. Pretendemos que esta lista funcione como una colección personalizada de películas que el usuario aún no ha visto y que le interesaría ver en un futuro, pero también podría utilizarse como una lista de películas favoritas, depende del uso que nuestro usuario prefiera darle.

Antes de comenzar con la creación del proyecto y de escribir código, debemos analizar y tener en cuenta qué elementos van a estar presentes en nuestra app, esto nos va a servir para distinguir e identificar las activities, layouts, modelos e interfaces necesarios.

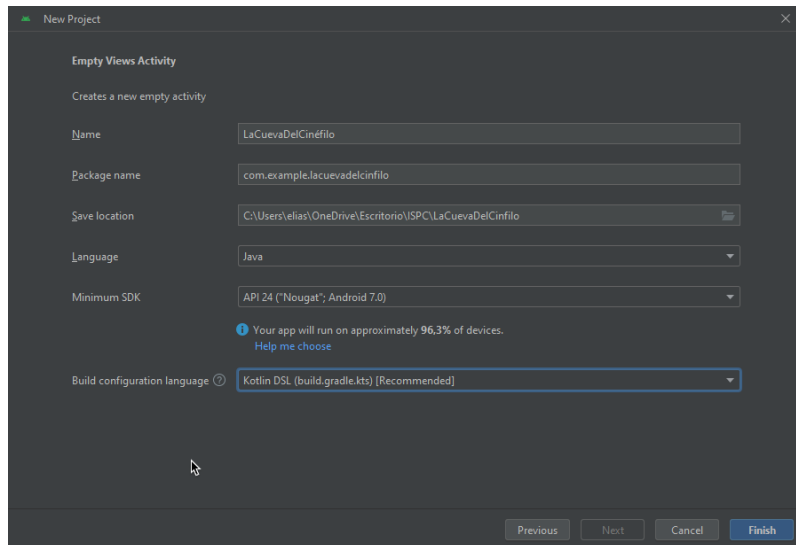
Queremos que nuestra app tenga:

- **Pantalla de login y register**
- **Pantalla principal (Catálogo)**
- **Lista personalizada (Wishlist)**
- **Formulario de contacto**
- **Vista a detalle de cada película**
- **Barra de navegación**
- **Vista a Perfil y edición del mismo.**

Comenzando a crear nuestra app

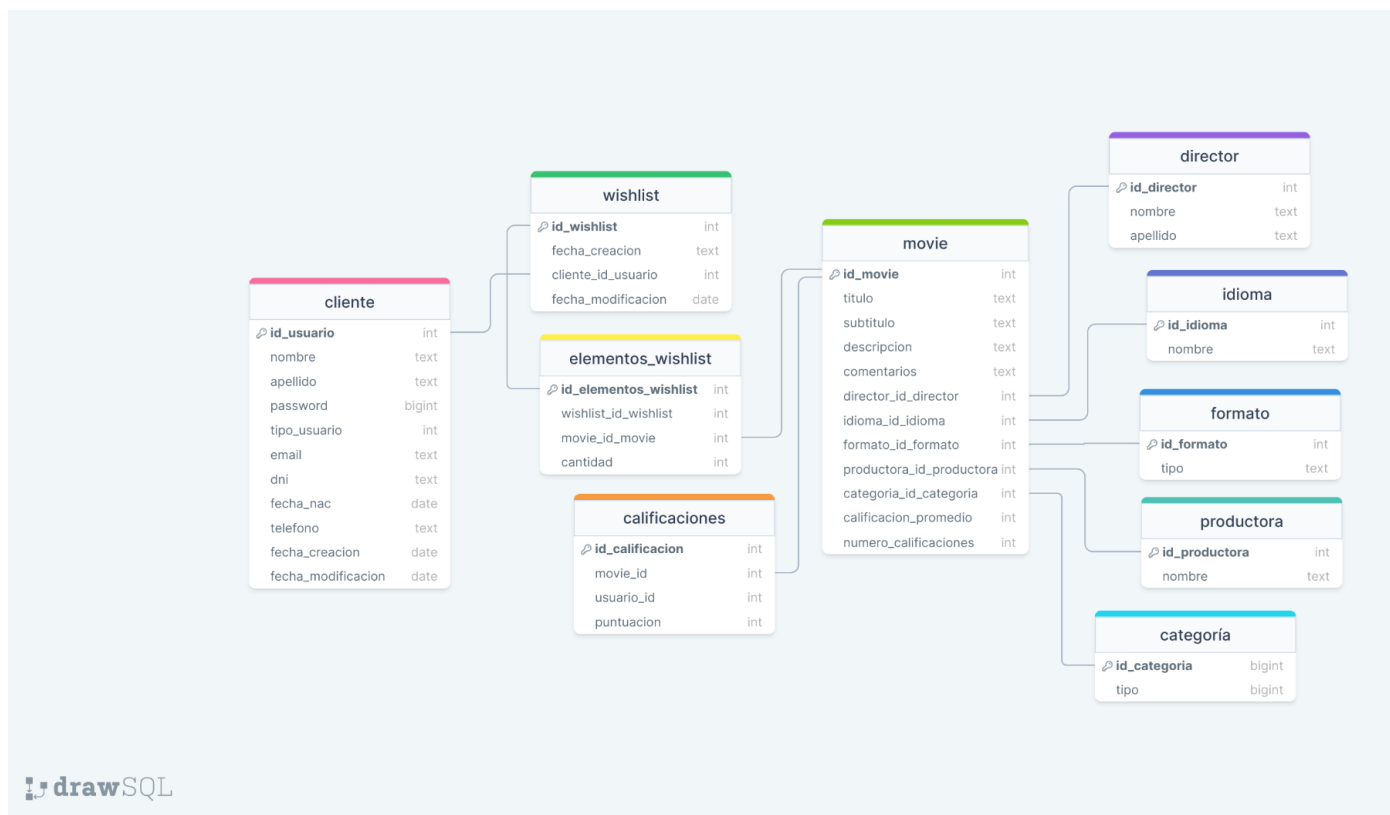
Utilizamos el IDE Android Studio y configuramos nuestro proyecto con una “Empty Views Activity” para posteriormente darle el nombre correspondiente a nuestra app y setear algunos parámetros como la ubicación de nuestro proyecto, el lenguaje con el que vamos a

programar (Java en este caso), la versión de Android que utilizaremos.



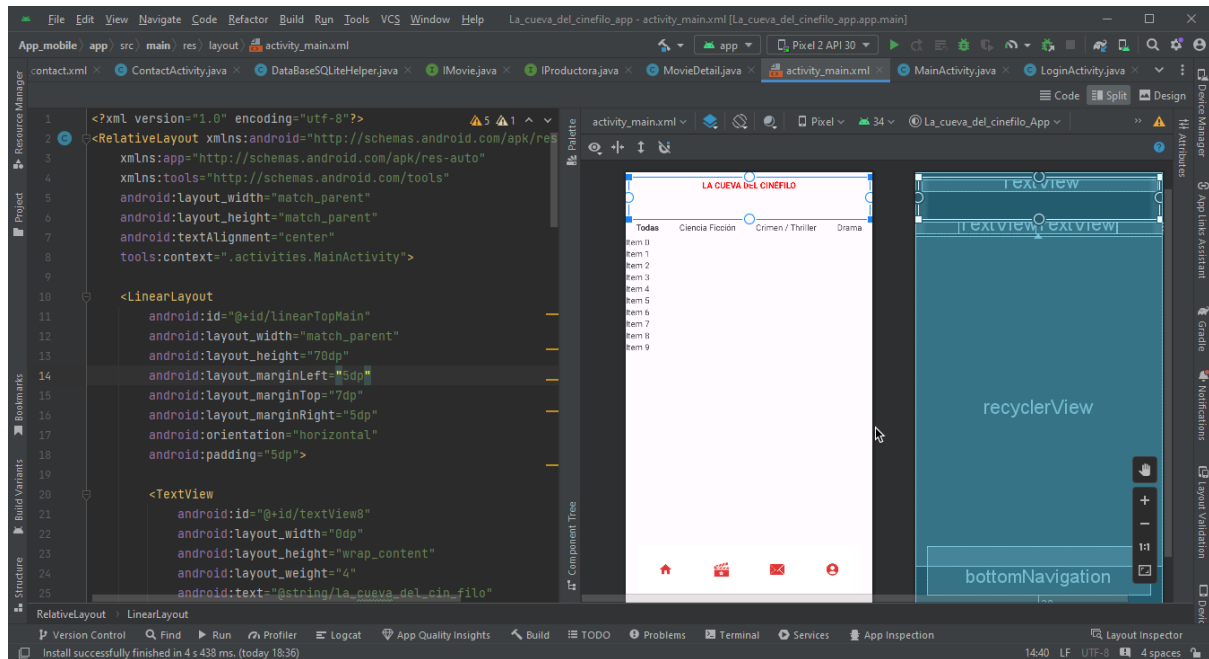
Definiendo nuestro modelo de base de datos

Posteriormente llega la hora de crear nuestra base de datos con SQLite basándonos en lo que debe tener nuestra aplicación. Yo he creado este diagrama modelo para comenzar a hacer la base de datos.

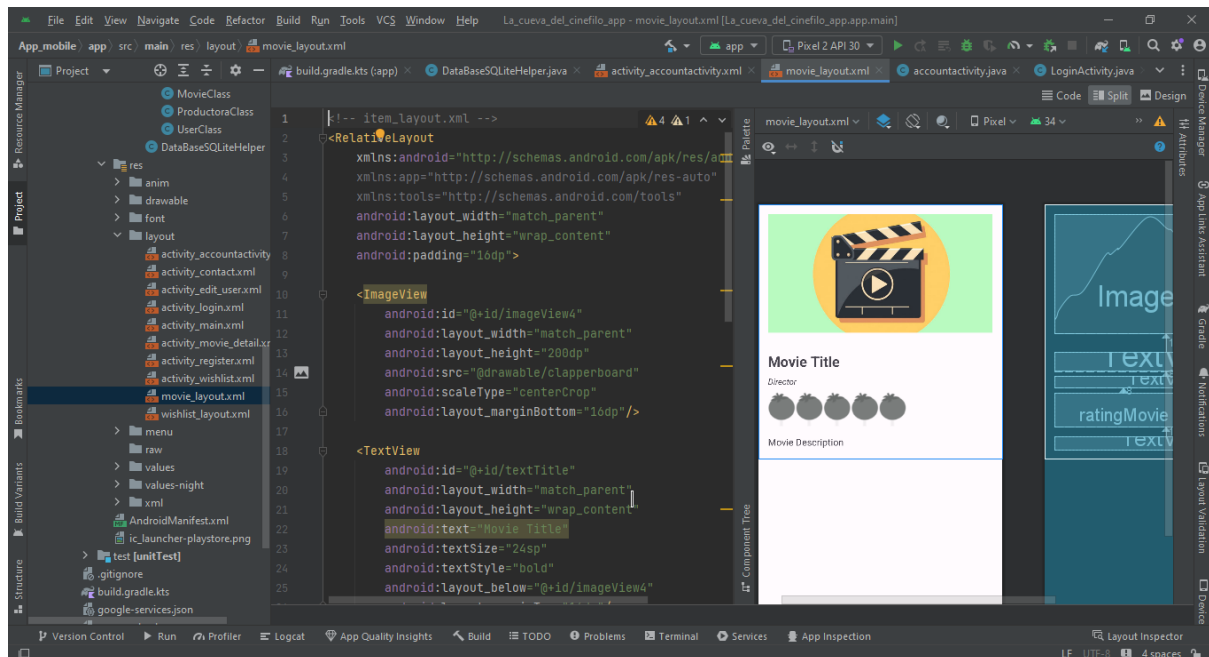


Diseño de clases y activities

El siguiente paso es el diseño de nuestras clases de Java y nuestras interfaces de usuario, para ello nos dirigimos a “*res/layout/activity_main.xml*” para configurar el apartado visual principal. En resumen, este layout representa una interfaz de usuario que consta de un encabezado con un título, una sección de categorías, una lista desplegable de elementos y una barra de navegación en la parte inferior. Utilizamos **RecyclerView** para mostrar la lista de películas.



Luego toca crear el layout de cada película, es por ello que debemos crear nuestro **movielayout**. Ésto sería cada componente o card de película que va a estar en el catálogo



A medida que vamos avanzando con el desarrollo de nuestro código vamos creando las clases correspondientes, en este caso **MovieClass** que se relaciona con la interfaz **IMovie**

```
54 usages
public class MovieClass implements IMovie {
    2 usages
    public static String COLUMN_ID = "id_movie";
    2 usages
    public static final String COLUMN_TITULO = "titulo";
    2 usages
    public static final String COLUMN_SUBTITULO = "subtitulo";
    2 usages
    public static final String COLUMN_DESCRIPCION = "descripcion";
    2 usages
    public static final String COLUMN_COMENTARIOS = "comentarios";
    2 usages
    public static final String COLUMN_DIRECTOR_ID = "director_id_director";
    2 usages
    public static final String COLUMN_IDIOMA_ID = "idioma_id_idioma";
    2 usages
    public static final String COLUMN_FORMATO_ID = "formato_id_formato";
    2 usages
    public static final String COLUMN_PRODUCTORA_ID = "productora_id_productora";
    2 usages
    public static final String COLUMN_CATEGORIA_ID = "categoria_id_categoria";
    2 usages
    public static final String COLUMN_CALIFICACION_PROMEDIO = "calificacion_promedio";
    2 usages
    public static final String COLUMN_NUMERO_CALIFICACIONES = "numero_calificaciones";
}
```

La interfaz **IMovie** define un conjunto de métodos que representan propiedades asociadas a una película en una aplicación. Cada método en la interfaz proporciona una forma de acceder a información específica sobre una película.

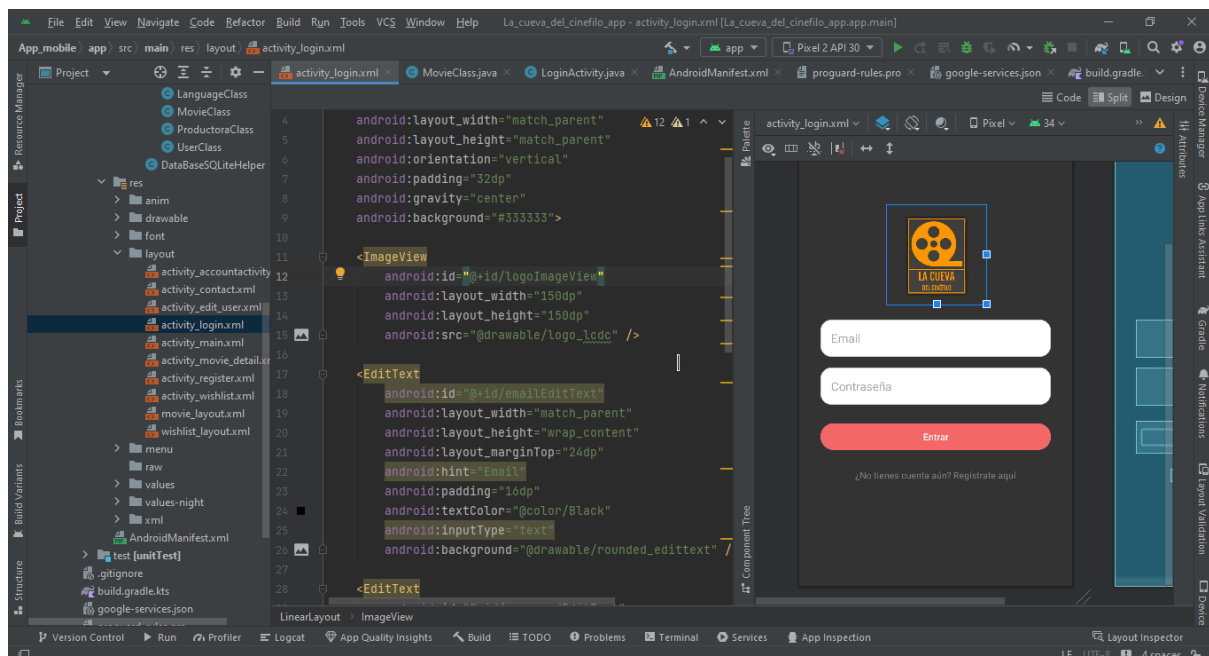
Luego desarrollé las actividades correspondientes al **login** y **register**, ya que queremos que nuestra app al abrirse pida autenticación para poder acceder a las demás funcionalidades. A medida que vamos desarrollando y completando las activities vamos a configurar la estructura de nuestro proyecto y la manera en que las activities se irán relacionando. En nuestro **AndroidManifest.xml** establecemos la actividad principal de la aplicación como LoginActivity, que se lanzará cuando la aplicación se inicie.

```

        android:name=".activities.accountactivity"
        android:exported="false" />
<activity
    android:name=".activities.MovieDetail"
    android:exported="false" />
<activity
    android:name=".activities.RegisterActivity"
    android:exported="false"
    android:label="RegisterActivity" />
<activity
    android:name=".activities.LoginActivity"
    android:exported="true">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

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

```



Ciberseguridad: implementando Firebase Auth

Para el registro y login de usuarios al principio lo había hecho de manera local con SQLite, pero luego me pareció buena idea implementar Firebase Auth únicamente para la creación

y autenticación de usuarios. Decidí implementar Firebase Auth para mejorar la seguridad de la aplicación debido a que ésta plataforma nos proporciona características y medidas de seguridad sólidas, **utiliza conexiones seguras HTTPS para transmitir datos entre la aplicación y el servidor**, lo que garantiza que la información de inicio de sesión y credenciales de usuario se mantengan **cifradas y protegidas** durante el proceso de autenticación.

Manejo de errores e inconvenientes

En términos generales el desarrollo de la aplicación no me dio problemas, pero si estuve con bastantes inconvenientes a la hora de implementar Firebase Auth. Mi idea era combinar SQLite y Firebase Auth, fue todo un desafío y verdaderamente fue gratificante haber resuelto todos los problemas.

Siendo específicos, uno de los problemas más recurrentes que tuve que enfrentar era que a la hora de loguearse como usuario el Firebase auth funcionaba, pero en local no, y esto me impedía hacer cosas como agregar películas a la wishlist, ya que el método de get user en local no estaba funcionando. Pude resolverlo a través de la documentación y además averiguando en qué parte específica del código se generaba el problema mediante **Logs** para ver en consola (**Pruebas de confirmación**), me instruí con videos y un poco de ChatGPT para resolver los bugs.

Para combinar FirebaseAuth + SQLite en local sin problemas hice el metodo **signInWithFirebase** que lo que hace cuando es exitoso el login es que guarda el ID del usuario en SharedPreferences mediante **saveUserIDToSharedPreferences**, El ID del usuario se guarda con la clave "**userid**". Posteriormente se ejecuta el método **signInLocally** que realiza la autenticación localmente después de que el usuario se ha autenticado con éxito en Firebase, se compara el ID de usuario obtenido localmente con -1 para determinar si la autenticación fue exitosa.

```
private void signInWithFirebase(String email, String password) {
    mAuth.signInWithEmailAndPassword(email, password)
        .addOnCompleteListener( activity: this, task -> {
            if (task.isSuccessful()) {
                FirebaseUser user = mAuth.getCurrentUser();
                showToast("Inicio de sesión exitoso en Firebase.");

                // Save user ID to SharedPreferences
                if (user != null) {
                    long userId = dbHelper.authenticateUser(email, password); // Replace this with the correct method
                    saveUserIDToSharedPreferences(userId);

                    // Now, try to sign in locally
                    signInLocally(email, password, user);
                }
            } else {
                showToast("Error en el inicio de sesión en Firebase. Verifica los datos e inténtalo de nuevo.");
                updateUI( user: null);
            }
        });
}
```

```

private void saveUserIDToSharedPreferences(long userId) {
    SharedPreferences sharedPreferences = getSharedPreferences( name: "UserPreferences", Context.MODE_PRIVATE);
    SharedPreferences.Editor editor = sharedPreferences.edit();
    editor.putLong( s: "userId", userId);
    editor.apply();
}

1 usage
private void signInLocally(String email, String password, FirebaseUser firebaseUser) {
    // Try to sign in locally (in SQLite)
    long userId = dbHelper.authenticateUser(email, password);
    Log.d( tag: "Local", msg: "Local User ID: " + userId);

    if (userId != -1) {
        showToast("Inicio de sesión exitoso localmente.");

        // Check if Firebase user is also authenticated
        if (firebaseUser != null) {
            // Both Firebase and local authentication successful
            updateUI(firebaseUser);
        } else {
            showToast("Esperando autenticación en Firebase...");
        }
    } else {
        showToast("Error en el inicio de sesión local. Verifica los datos e inténtalo de nuevo.");
        updateUI( user: null);
    }
}

```

Testing: casos de prueba

ID	Objetivo	Prioridad	Trazabilidad	Precondiciones	Entradas	Resultados Esperados	Resultados Reales
CP1	Ingresar a La Cueva del Cinéfilo sin datos de entrada	1	Ingresar a La Cueva del Cinéfilo y presionar login	App La Cueva del Cinéfilo	Correo electrónico y Contraseña sin valor	Visualizar un mensaje que detalle que el campo Nombre de Usuario tiene que tener un valor	Se visualiza el siguiente mensaje: Por favor, completa todos los campos.
CP2	Ingresar a La Cueva del Cinéfilo sólo con contraseña	2	Ingresar a La Cueva del Cinéfilo, ingresar solo correo electrónico y presionar login	App La Cueva del Cinéfilo	Correo electrónico con valor y Contraseña sin valor	Visualizar un mensaje que detalle que el campo Nombre de Usuario tiene que tener un valor	Se visualiza el siguiente mensaje: Por favor, completa todos los campos.
CP3	Ingresar a La Cueva del Cinéfilo con usuario y contraseña a no válidos	3	Ingresar a La Cueva del Cinéfilo, ingresar correo electrónico y contraseña aleatorios y presionar login	App La Cueva del Cinéfilo	Correo electrónico y Contraseña con valor aleatorio	Visualizar un mensaje que detalle que no corresponde a un usuario registrado	Error en el inicio de sesión de Firebase. Verifica los datos e inténtalo de nuevo

Evidencia testing: <https://youtube.com/shorts/5OSeAoqWOEM>

Cuadro de nuestros modelos, interfaces y activities

Modelos	Interfaces	Activities
CategoryClass	ICategory	accountactivity
DirectorClass	IDirector	ContactActivity
FormatClass	IFormat	EditUserActivity
LanguageClass	ILanguage	LoginActivity
MovieClass	IMovie	MainActivity
ProductoraClass	IProductora	MovieAdapter
UserClass	IUser	MovieDetail
		RegisterActivity
		WishlistActivity
		WishlistAdapter

Algunas capturas de la app:



MainActivity: Inicio / Catálogo de películas





Agregar a la Lista

Subtítulo del movie

David Fincher

Crimen/Thriller


Descripción


Dos detectives, Somerset y Mills, persiguen a un astuto asesino en serie que comete crímenes atroces basados en los siete pecados capitales. La narrativa oscura y tensa revela giros impactantes hasta su impactante conclusión.

Especificaciones

Distribución	Arnold Kopelson Productions
Formato	4K
Lenguaje	Español

MovieDetail





Ingrese su nombre

Ingrese su correo electrónico

Escriba su mensaje aquí

Enviar

Formulario de contacto



eliasgiannantonio@live.com

.....

Entrar

¿No tienes cuenta aún? Regístrate aquí

LoginActivity