

UF 2 - Aplicación CRUD: ToDoList

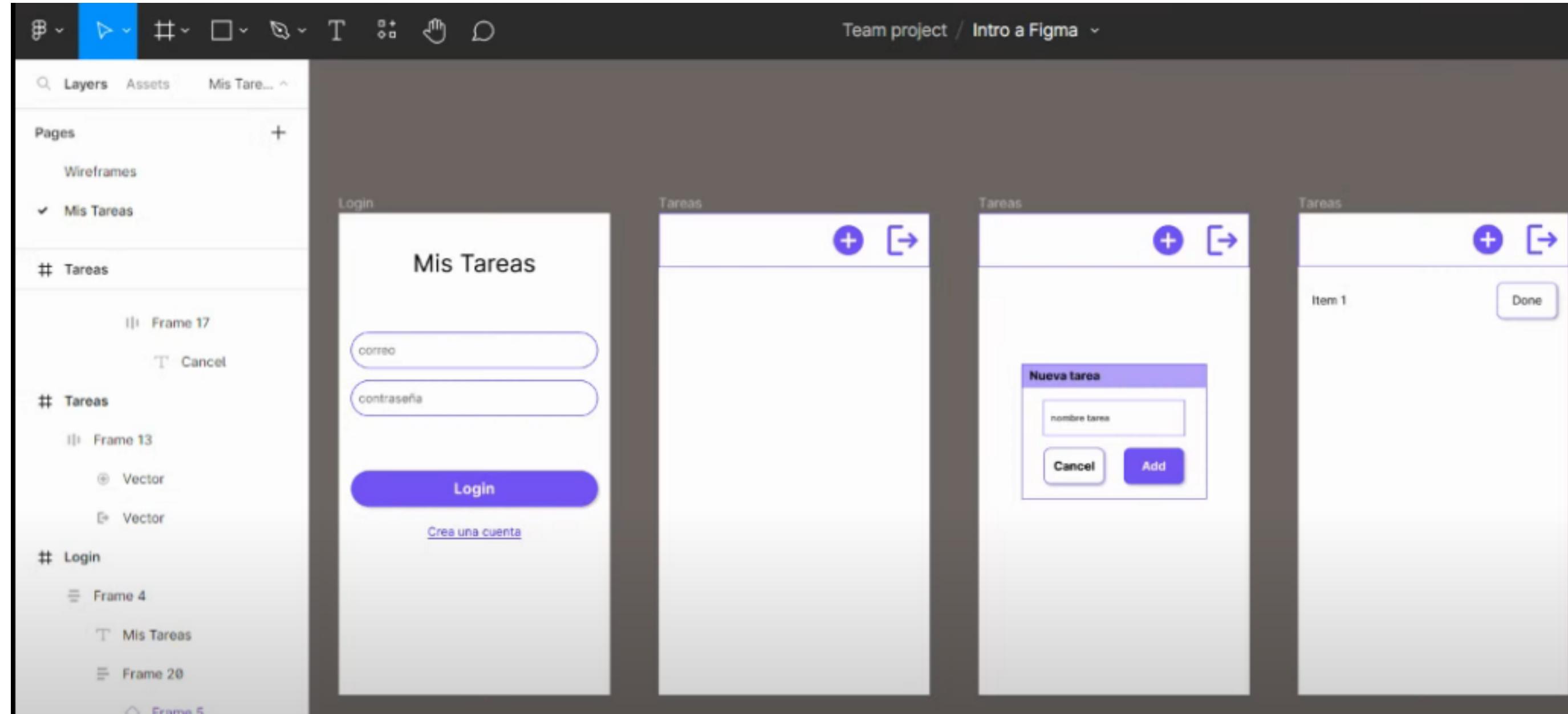


Contenido

- Diseño del proyecto
- Pantalla de login
- AppBar
- Eventos
- Toast
- Cuadros de diálogo
- ListView
- Configuración de FireBase
- Comunicación entre activities
- Login y registro en FireBase
- Altas y consultas
- Borrar datos y logout
- Acciones en Eventos

Diseño del proyecto

Para hacer un proyecto es aconsejable utilizar [figma](#). Esta aplicación web nos permite crear un diseño de como queremos que quede nuestro proyecto:



Diseño del proyecto

Cambiamos los estilos de la aplicación modificando el archivo ubicado en **res/values/themes** :

themes.xml

```
<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Base.Theme.ToDoList" parent="Theme.Material3.Light">
        <!-- Customize your light theme here. -->
        <!-- <item name="colorPrimary">@color/my_light_primary</item> -->
    </style>

    <style name="Theme.ToDoList" parent="Base.Theme.ToDoList" />
</resources>
```

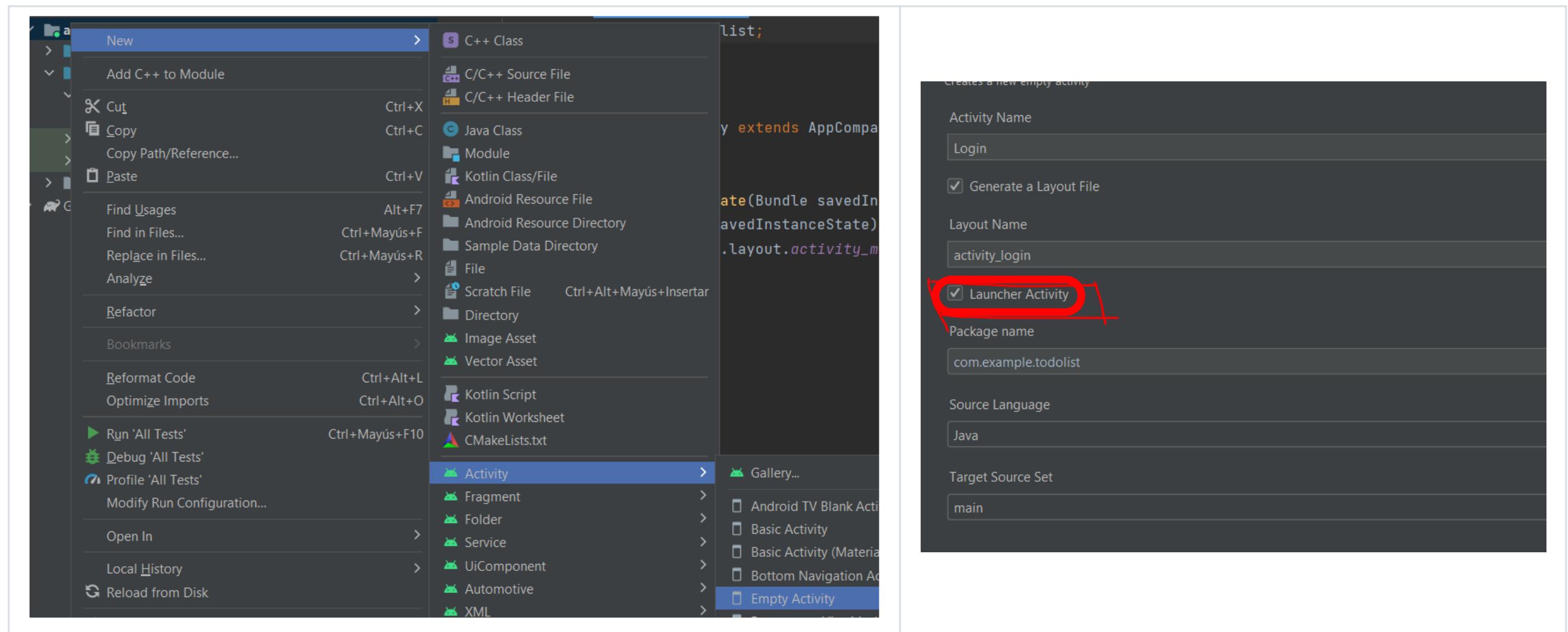
Configuramos la paleta de colores para la aplicación en el archivo ubicado en **res/values/colors** :

colors.xml

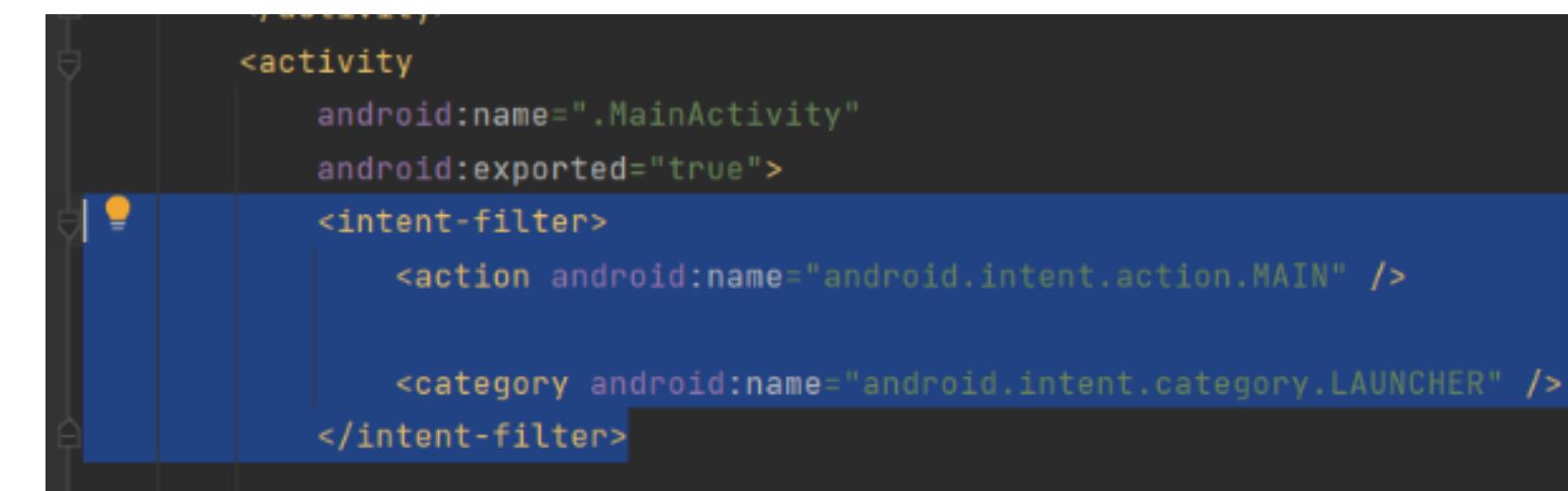
```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFF</color>
    <color name="morado">#6750A4</color>
    <color name="morado_claro">#FFBB86FC</color>
</resources>
```

Pantalla de login

Para la pantalla de login tendremos que crear una nueva activity, muy importante marcar la opción de **Launcher Activity**:

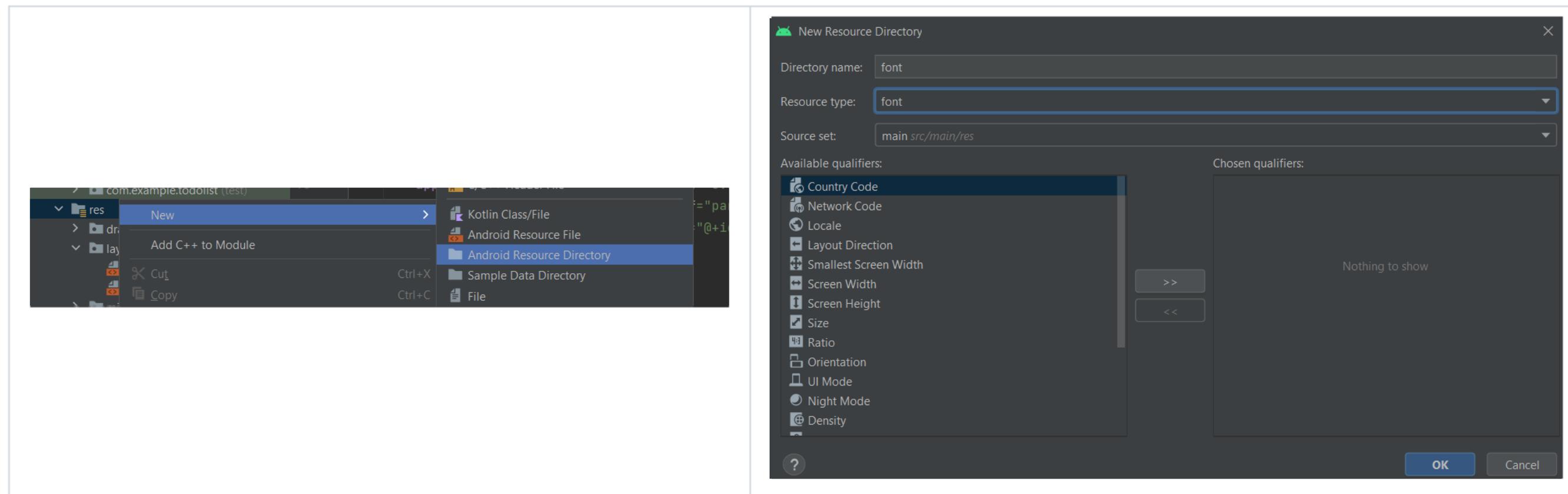


Vamos a la carpeta manifest → AndoridManifest.xml y eliminamos esto:



Pantalla de login

En la carpeta de **res** creamos una carpeta destinada a fuentes y añadimos nuestra fuente



Pantalla de login

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".Login">

    <TextView
        android:id="@+id/titulo"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="56dp"
        android:text="@string/Titulo"
        android:textColor="@color/morado"
        android:textSize="50sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        android:fontFamily="@font/fuente_titulo"/>

    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/layoutCorreo"
        android:layout_width="309dp"
        android:layout_height="72dp"
        android:layout_marginTop="32dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/titulo">

        <com.google.android.material.textfield.TextInputEditText
            android:id="@+id/cajaCorreo"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:hint="@string/hintCorreo"
            android:textColorHint="@color/morado_claro"
            android:textSize="16sp" />

    </com.google.android.material.textfield.TextInputLayout>

    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/textInputLayout"
        android:layout_width="314dp"
        android:layout_height="72dp"
        android:layout_marginTop="24dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/layoutCorreo">

        <com.google.android.material.textfield.TextInputEditText
            android:id="@+id/cajaPass"
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:hint="@string/hintPass"
            android:inputType="textPassword"
            android:textColorHint="@color/morado_claro"
            android:textSize="16sp" />

    </com.google.android.material.textfield.TextInputLayout>

    <Button
        android:id="@+id/botonLogin"
        android:layout_width="308dp"
        android:layout_height="56dp"
        android:layout_marginTop="72dp"
        android:text="@string/login"
        android:textColor="@color/white"
        android:textSize="24sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textInputLayout" />

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="24dp"
        android:text="@string/registro"
        android:textColor="@color/morado"
        android:textSize="20sp"
        android:textStyle="bold"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/botonLogin" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

```
<com.google.android.material.textfield.TextInputLayout
    android:id="@+id/textInputLayout"
    android:layout_width="314dp"
    android:layout_height="72dp"
    android:layout_marginTop="24dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/layoutCorreo">

    <com.google.android.material.textfield.TextInputEditText
        android:id="@+id/cajaPass"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:hint="@string/hintPass"
        android:inputType="textPassword"
        android:textColorHint="@color/morado_claro"
        android:textSize="16sp" />

</com.google.android.material.textfield.TextInputLayout>

<com.google.android.material.textfield.TextInputLayout
    android:id="@+id/textInputLayout"
    android:layout_width="314dp"
    android:layout_height="72dp"
    android:layout_marginTop="24dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/cajaCorreo">

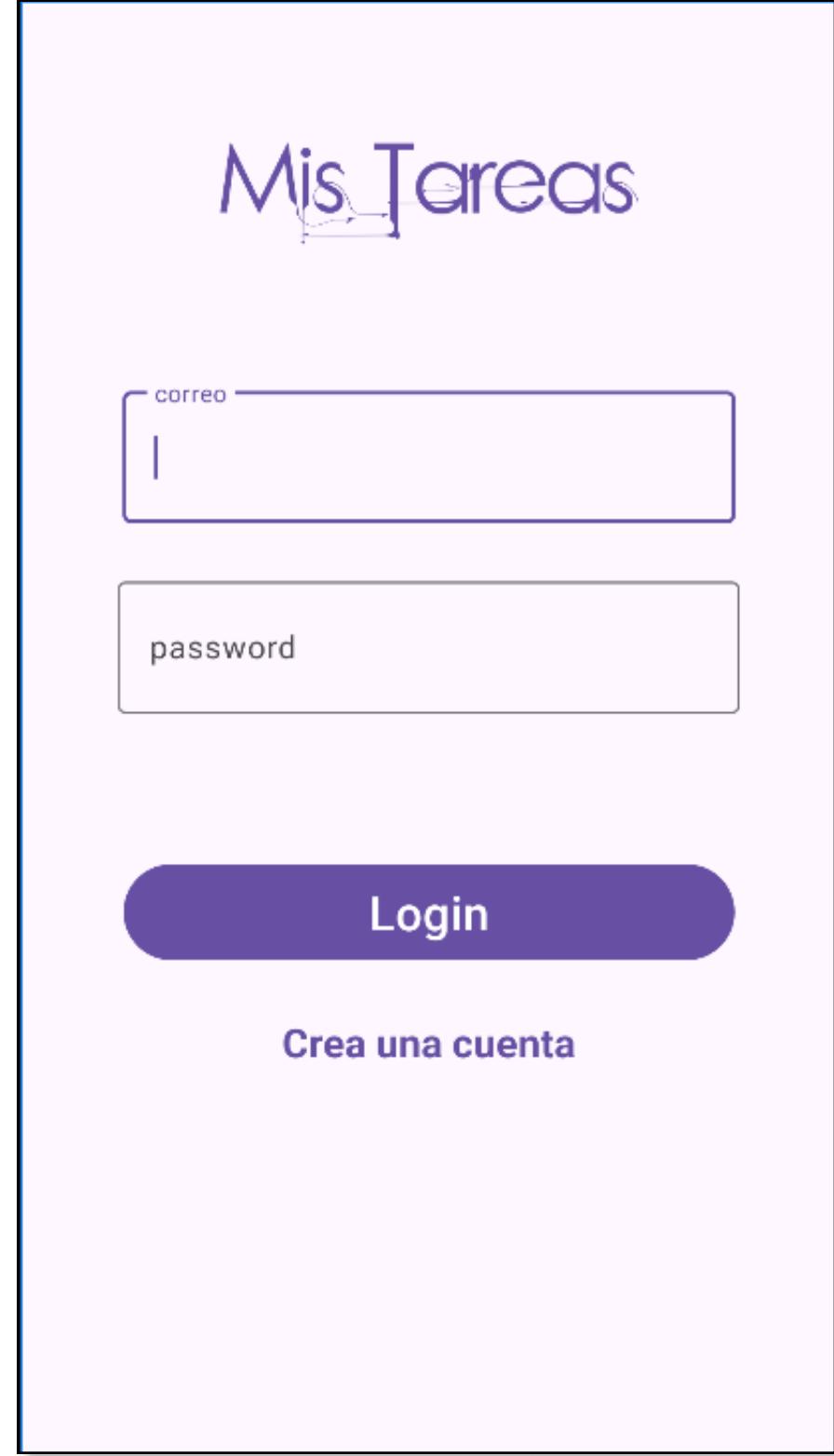
    <com.google.android.material.textfield.TextInputEditText
        android:id="@+id/cajaCorreo"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:hint="@string/hintCorreo"
        android:textColorHint="@color/morado_claro"
        android:textSize="16sp" />

</com.google.android.material.textfield.TextInputLayout>

<Button
    android:id="@+id/botonLogin"
    android:layout_width="308dp"
    android:layout_height="56dp"
    android:layout_marginTop="72dp"
    android:text="@string/login"
    android:textColor="@color/white"
    android:textSize="24sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textInputLayout" />

<TextView
    android:id="@+id/textView2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="24dp"
    android:text="@string/registro"
    android:textColor="@color/morado"
    android:textSize="20sp"
    android:textStyle="bold"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/botonLogin" />

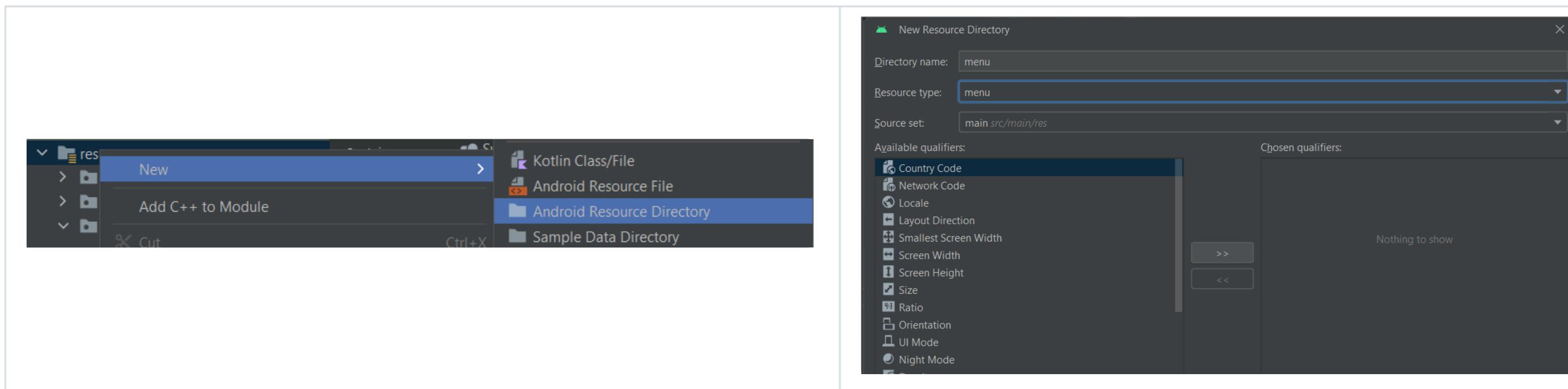
</androidx.constraintlayout.widget.ConstraintLayout>
```



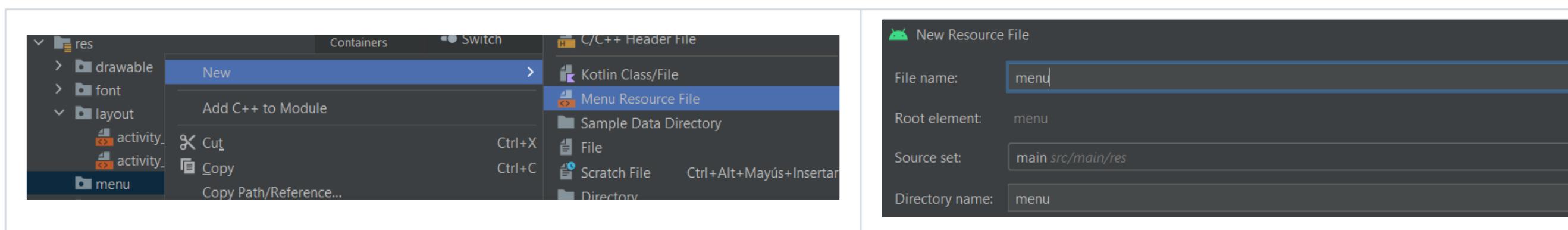
AppBar

Es recomendable usar [materialdesingicon](#) u otra web como [font.google](#) en la que podamos descargar iconos para así usarlos en nuestra app.

Creamos un directorio de trabajo Android de tipo menu.



En el activity_main borramos el textView que hay y creamos el xml que usaremos que se ubicará en la carpeta menu:



AppBar

```
menu.xml

<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
      xmlns:app="http://schemas.android.com/apk/res-auto"
      xmlns:tools="http://schemas.android.com/tools"
      tools:context=".MainActivity">
    <item
        android:id="@+id/mas"
        android:title="@string/mas"
        android:icon="@mipmap/mas"
        app:showAsAction="ifRoom"
    />

    <item
        android:id="@+id/logout"
        android:title="@string/logout"
        android:icon="@mipmap/logout"
        app:showAsAction="ifRoom"
    />
</menu>
```

android:showAsAction

Palabra clave. Indica cuándo y cómo se muestra este elemento como un elemento de acción en la barra de la app. Un elemento de menú puede aparecer como un elemento de acción solo cuando la actividad incluye una barra de la app. Valores válidos:

Valor	Descripción
ifRoom	Solo coloca este elemento en la barra de la app si hay espacio. Si no hay lugar para todos los elementos marcados como "ifRoom", se muestran como acciones los elementos que tengan los valores <code>orderInCategory</code> más bajos; los restantes aparecerán en el menú ampliado.
withText	Incluye también el texto del título (definido por <code>android:title</code>) con el elemento de acción. Puedes incluir este valor junto con uno de los otros marcadores separándolos con un canal .
never	Nunca coloques este elemento en la barra de la app. En su lugar, enumera el elemento en el menú ampliado de la barra de la app.
always	Siempre coloca este elemento en la barra de la app. Evita usar esta opción a menos que sea esencial que el elemento siempre aparezca en la barra de acción. Configurar varios elementos para que siempre aparezcan como elementos de acción puede hacer que se superpongan con otra IU en la barra de la app.

AppBar

```
MainActivity.java

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.menu, menu);
    return super.onCreateOptionsMenu(menu);
}

@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    if(item.getItemId() == R.id.mas){
        //activar el cuadro de diálogo para añadir tarea
        return true;
    }else if(item.getItemId() == R.id.logout) {
        //cierra de sesión de Firebase
        return true;
    }else return super.onOptionsItemSelected(item);
}
```



Eventos

```
Login.java

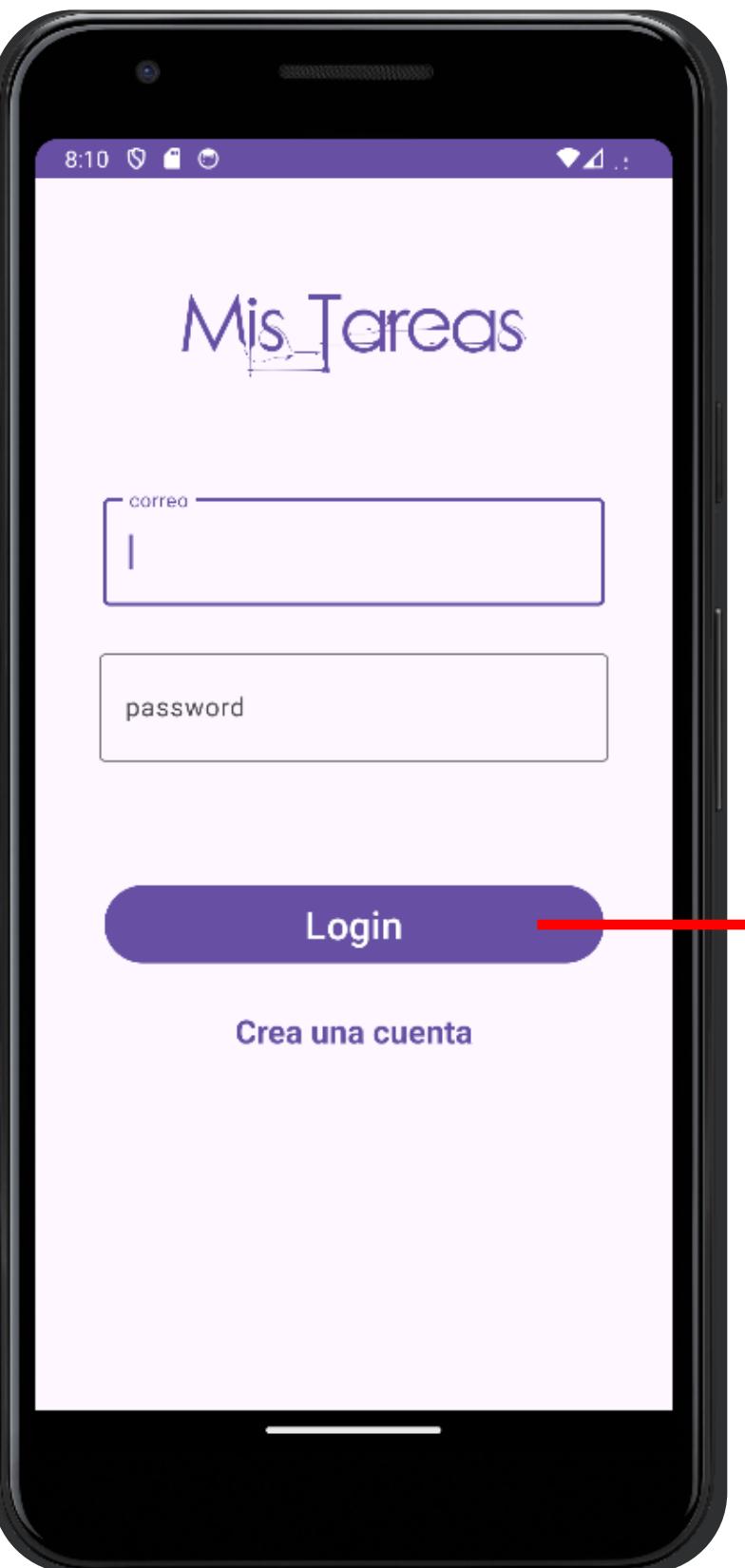
public class Login extends AppCompatActivity {

    Button botonLogin;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        getSupportActionBar().hide();

        botonLogin = findViewById(R.id.botonLogin);
        botonLogin.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent = new Intent(Login.this, MainActivity.class);
                startActivity(intent);
            }
        });
    }
}
```



Eventos

```
 Login.java

package com.rcl.todolist;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class Login extends AppCompatActivity {

    Button botonLogin;
    TextView botonRegistro;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        getSupportActionBar().hide();

        botonLogin = findViewById(R.id.botonLogin);
        botonLogin.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // LOGIN EN FIREBASE
                Intent intent = new Intent(Login.this, MainActivity.class);
                startActivity(intent);
            }
        });

        botonRegistro = findViewById(R.id.botonRegistro);
        botonRegistro.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // CREAR USUARIO EN FIREBASE
                Intent intent = new Intent(Login.this, MainActivity.class);
                startActivity(intent);
            }
        });
    }
}
```

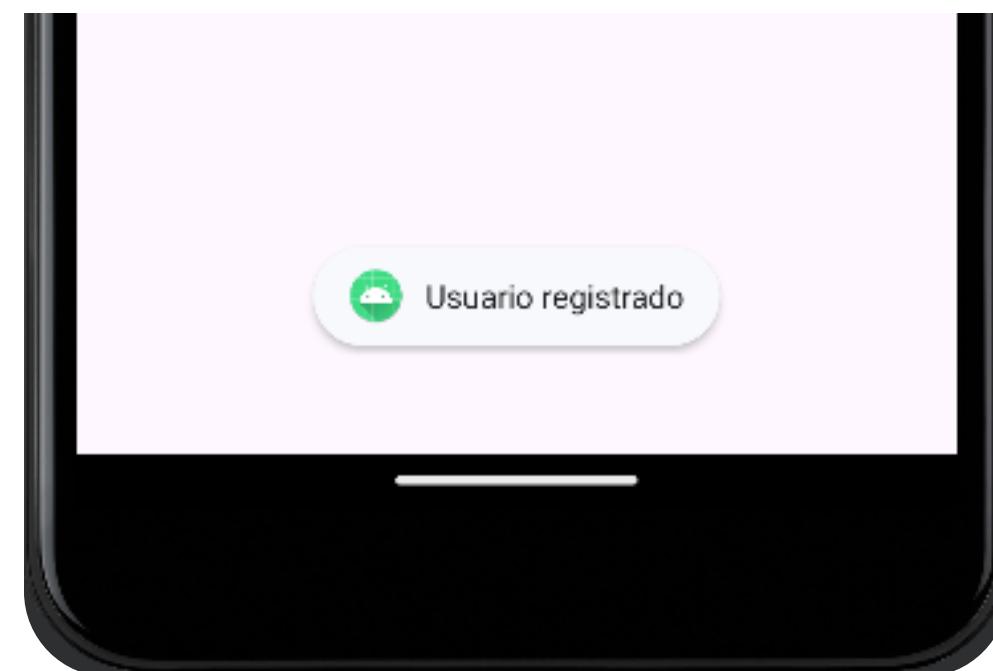
Toast

Mensaje que aparece en una parte de la pantalla (por defecto en el bottom) para dar información sobre acciones de una app.

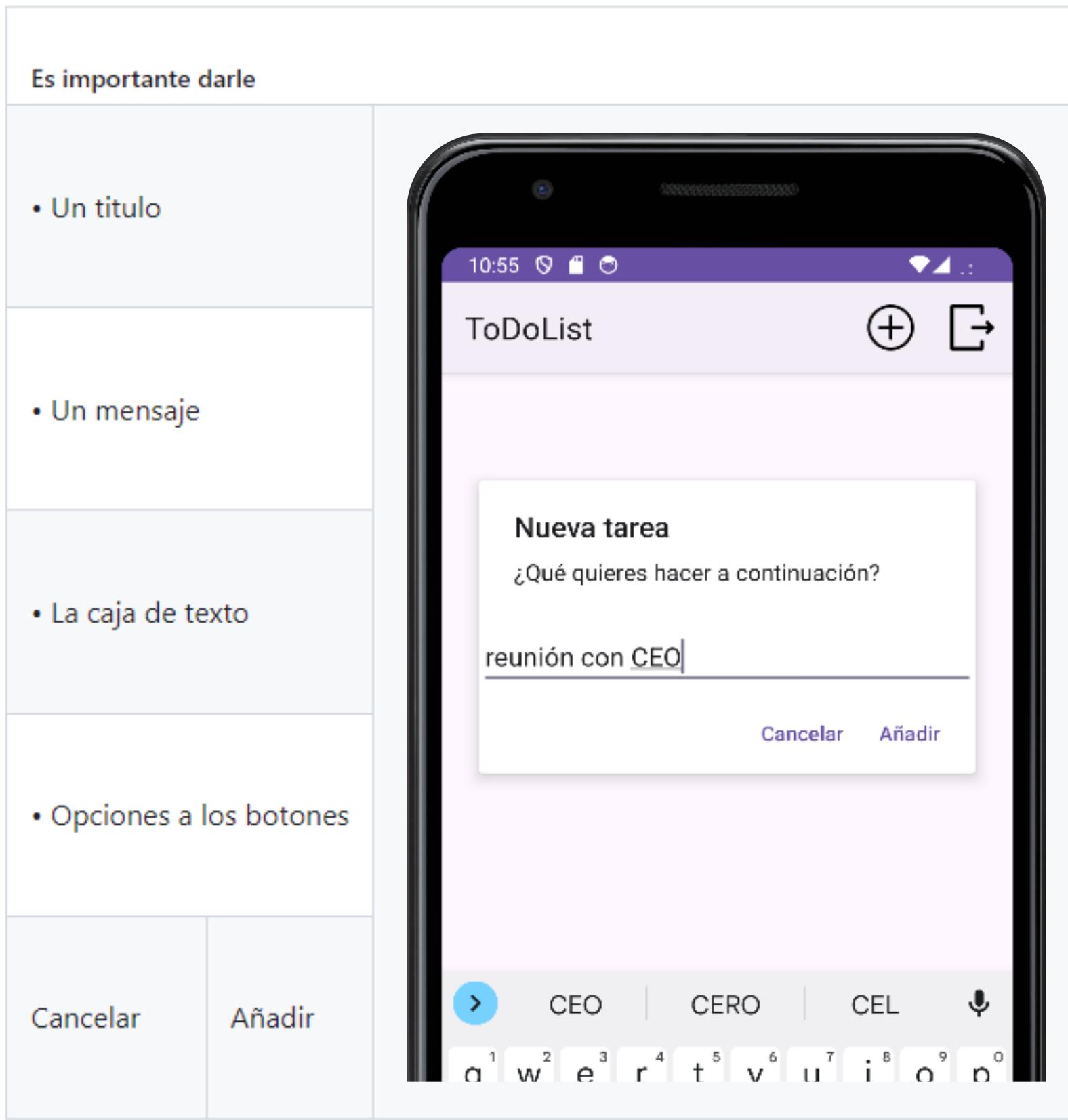
Se crea con el siguiente código para que se ejecute cuando se pulse un botón, por ejemplo.

```
Toast.makeText(this, "Usuario Registrado", Toast.LENGTH_LONG).show();
```

codetomg.com



Cuadros de diálogo



```
MainActivity.java
```

```
@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    if(item.getItemId() == R.id.mas){
        //activar el cuadro de diálogo para añadir tarea

        final EditText taskEditText = new EditText(this);
        AlertDialog dialog = new AlertDialog.Builder(this)
            .setTitle("Nueva tarea")
            .setMessage("¿Qué quieres hacer a continuación?")
            .setView(taskEditText)
            .setPositiveButton("Añadir", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    // añadir tarea a la base de datos y al listView

                    Toast.makeText(MainActivity.this,"Tarea añadida",Toast.LENGTH_SHORT).show();
                }
            })
            .setNegativeButton("Cancelar",null)
            .create();
        dialog.show();
        return true;
    }

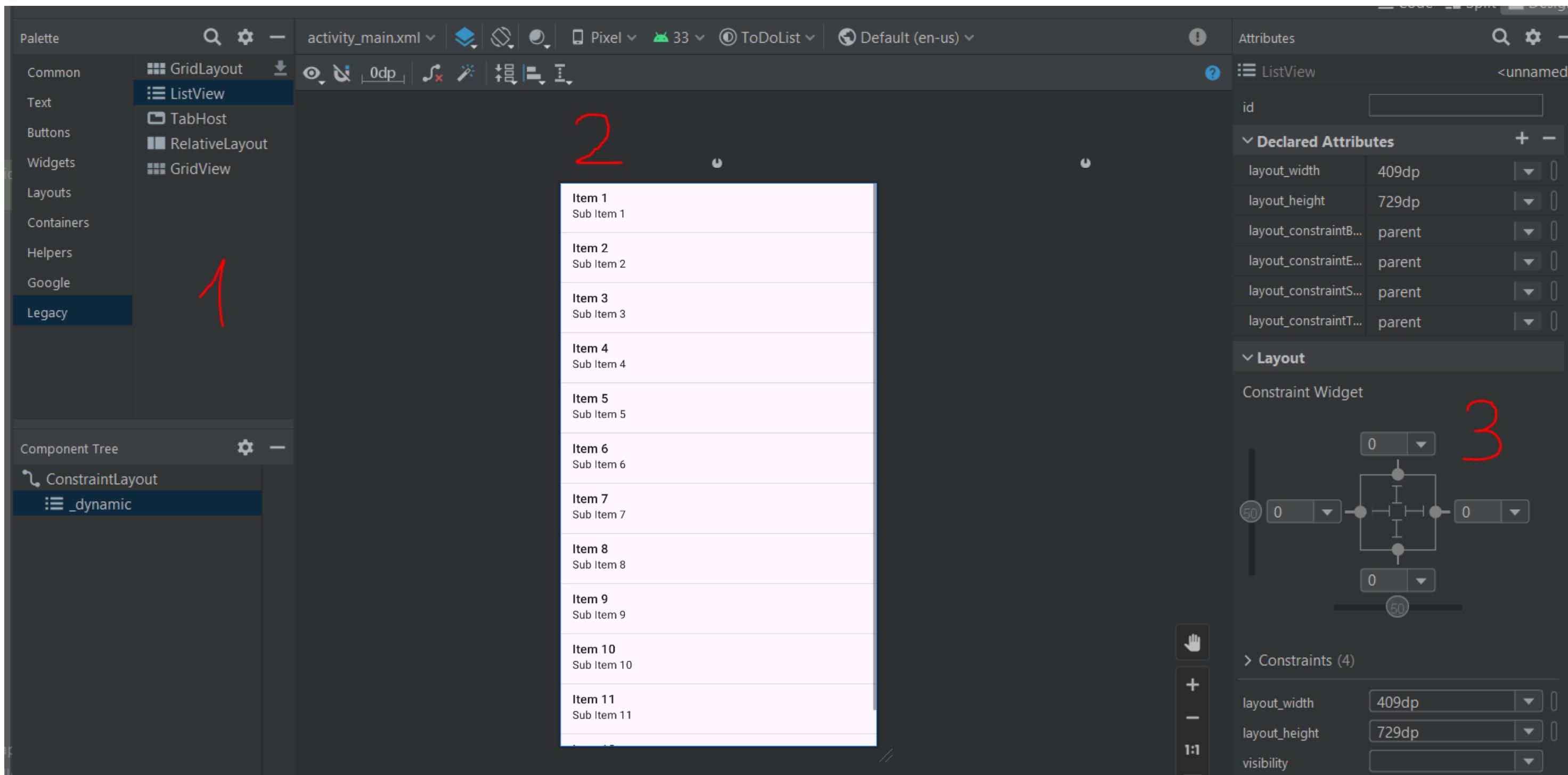
    else if(item.getItemId() == R.id.logout) {
        // cierre de sesión de Firebase

        startActivity(new Intent(MainActivity.this,Login.class));
        return true;
    }
    else return super.onOptionsItemSelected(item);
}
```

ListView

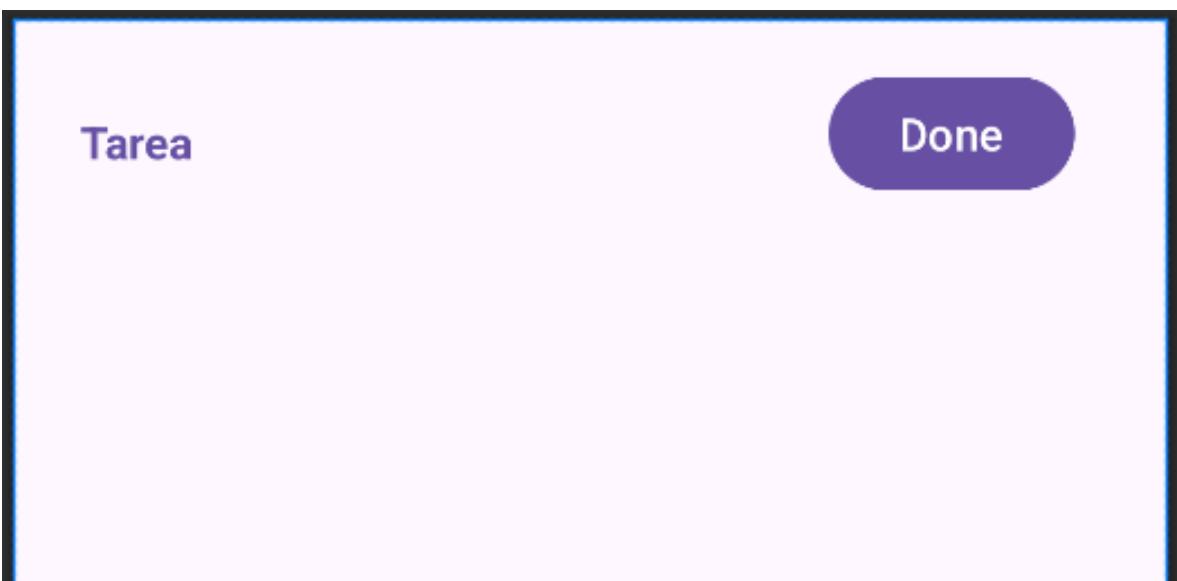
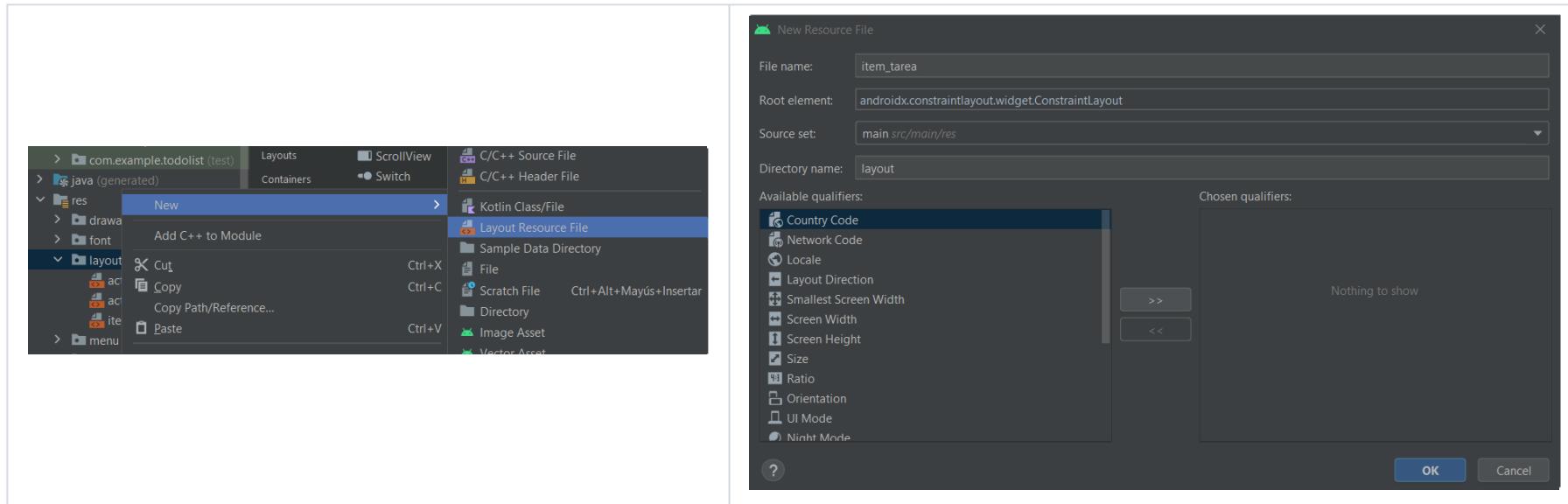
Para generar un listView tenemos que ir a activity_main.xml

1. Buscar la etiqueta listView.
2. Añadirlo a la ventana.
3. Marcarlo a los 4 extremos de la pantalla con la distancia que deseemos.



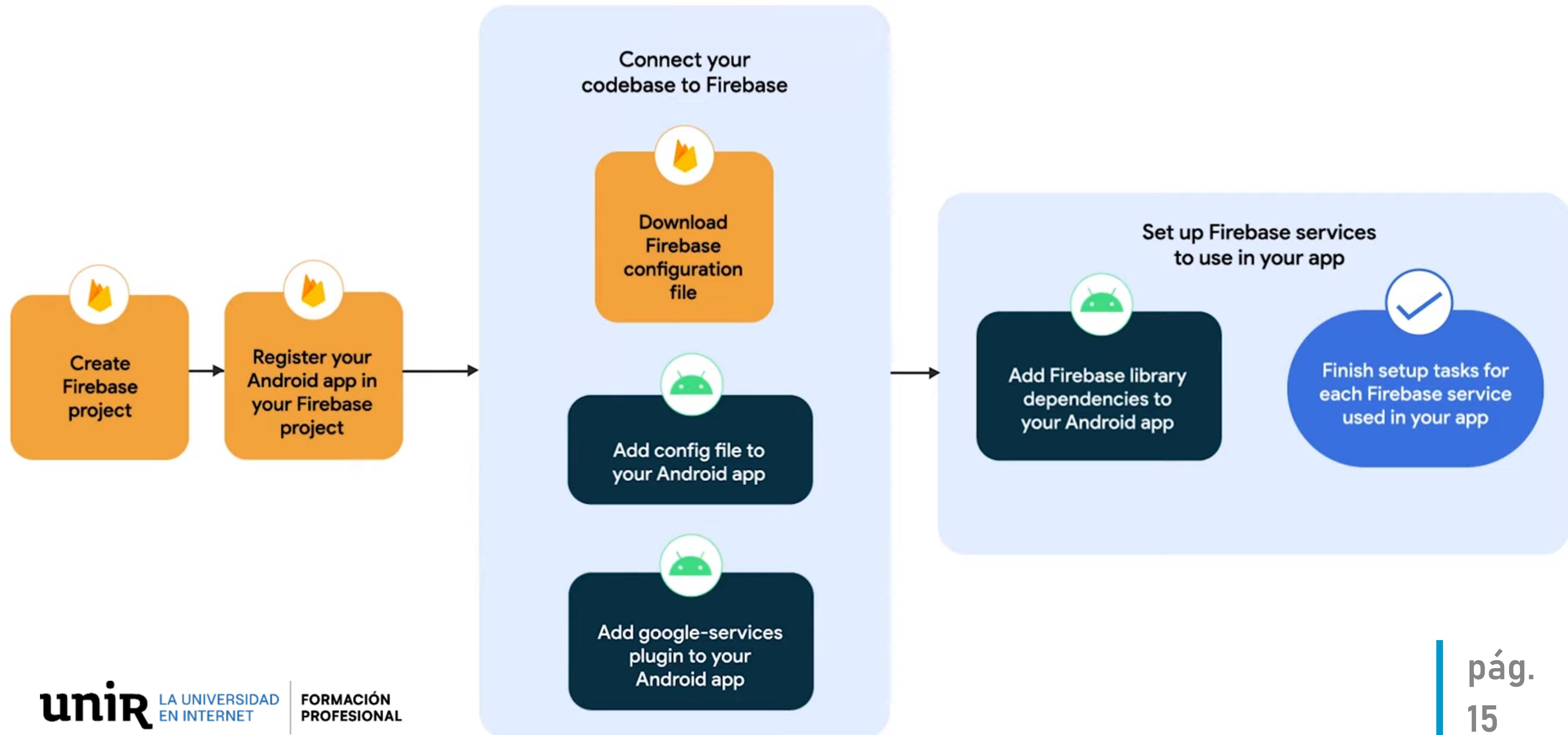
ListView

Definir el contenido que tendrá, vamos a layout y añadimos un nuevo fichero de recursos.



```
item_tarea.xml  
  
<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    xmlns:app="http://schemas.android.com/apk/res-auto"  
    xmlns:tools="http://schemas.android.com/tools"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent">  
  
    <TextView  
        android:id="@+id/textViewTarea"  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content"  
        android:layout_marginStart="24dp"  
        android:layout_marginTop="32dp"  
        android:text="@string/tarea"  
        android:textColor="@color/morado"  
        android:textSize="16sp"  
        android:textStyle="bold"  
        app:layout_constraintStart_toStartOf="parent"  
        app:layout_constraintTop_toTopOf="parent" />  
  
    <Button  
        android:id="@+id/buttonDone"  
        android:layout_width="wrap_content"  
        android:layout_height="wrap_content"  
        android:layout_marginTop="16dp"  
        android:layout_marginEnd="32dp"  
        android:text="@string/done"  
        android:textSize="16sp"  
        app:layout_constraintEnd_toEndOf="parent"  
        app:layout_constraintTop_toTopOf="parent" />  
  
</androidx.constraintlayout.widget.ConstraintLayout>
```

Configuración de FireBase



Configuración de FireBase

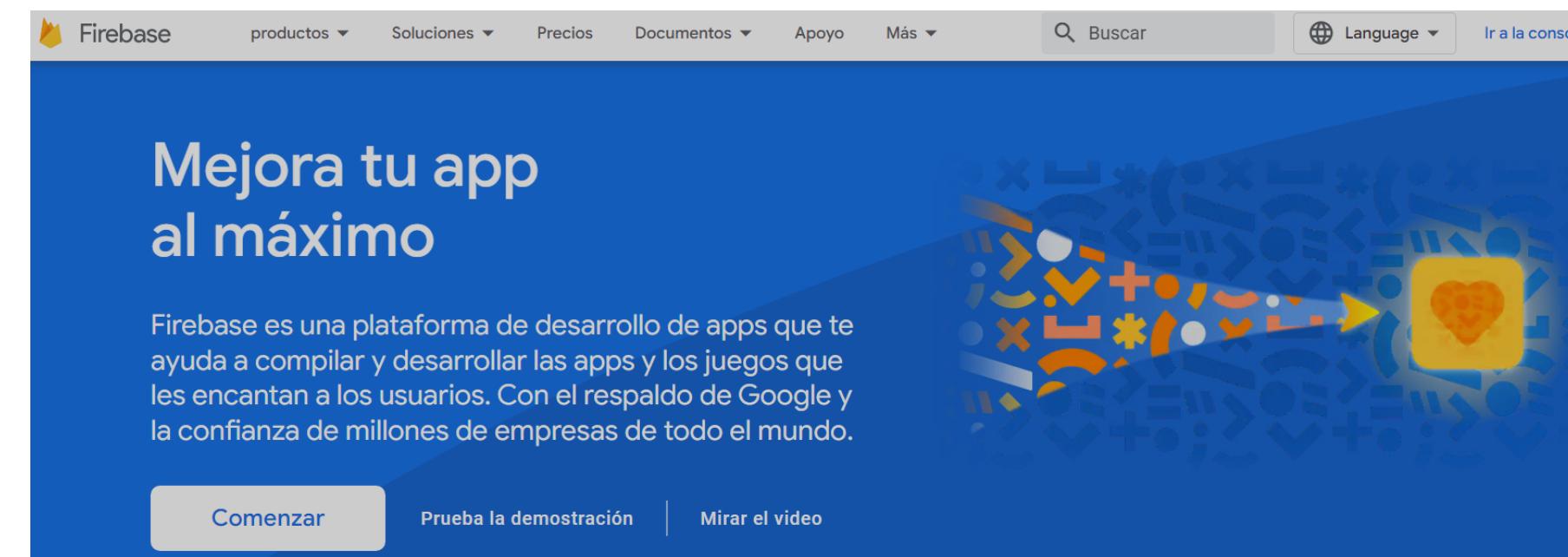
Para utilizar FireBase tenemos que tener el Android 19 (KitKat) mínimo. Para ver que cumplimos los requisitos podemos ir a build.gradle (a nivel de app) y miramos el compileSdk:

```
5 android {  
6     namespace 'com.example.todolist'  
7     compileSdk 32  
8  
9     defaultConfig {  
10         applicationId "com.example.todolist"  
11         minSdk 26  
12         targetSdk 32  
13         versionCode 1  
14         versionName "1.0"  
15  
16         testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  
17     }  
}
```

Ahora tendremos que acceder a [FireBase](#). Podemos tener una base de datos que se utilizará tanto para la app móvil como para la app web.

¿Cómo crear un proyecto?

1. Acceder a la [web](#) y darle al botón de comenzar.



2. Le damos un nombre a nuestro proyecto:

A screenshot of a project creation form. It asks for the 'nombre de tu proyecto' (Project name) which is filled with 'ToDoList'. Below the input field is a small preview of the project ID: 'todolist-41a1b'. There are two checkboxes at the bottom: 'Acepto las condiciones de Firebase' (I accept the Firebase terms of service) and 'Confirmo que usaré Firebase exclusivamente para fines relacionados con mi empresa, oficio o profesión.' (I confirm that I will use Firebase exclusively for purposes related to my company, profession or vocation). A blue 'Continuar' (Continue) button is at the bottom.

Configuración de FireBase

3. En caso de desarrollar una app que se subirá a la Play Store u otro sitio, activar la opción de Google Analytics. Si se va a usar para pruebas y poco más es mejor desactivarlo.

Google Analytics es una solución de analítica ilimitada y gratuita que permite usar la segmentación, los informes y otras funciones en Firebase Crashlytics, Cloud Messaging, In-App Messaging, Remote Config, A/B Testing y Cloud Functions.

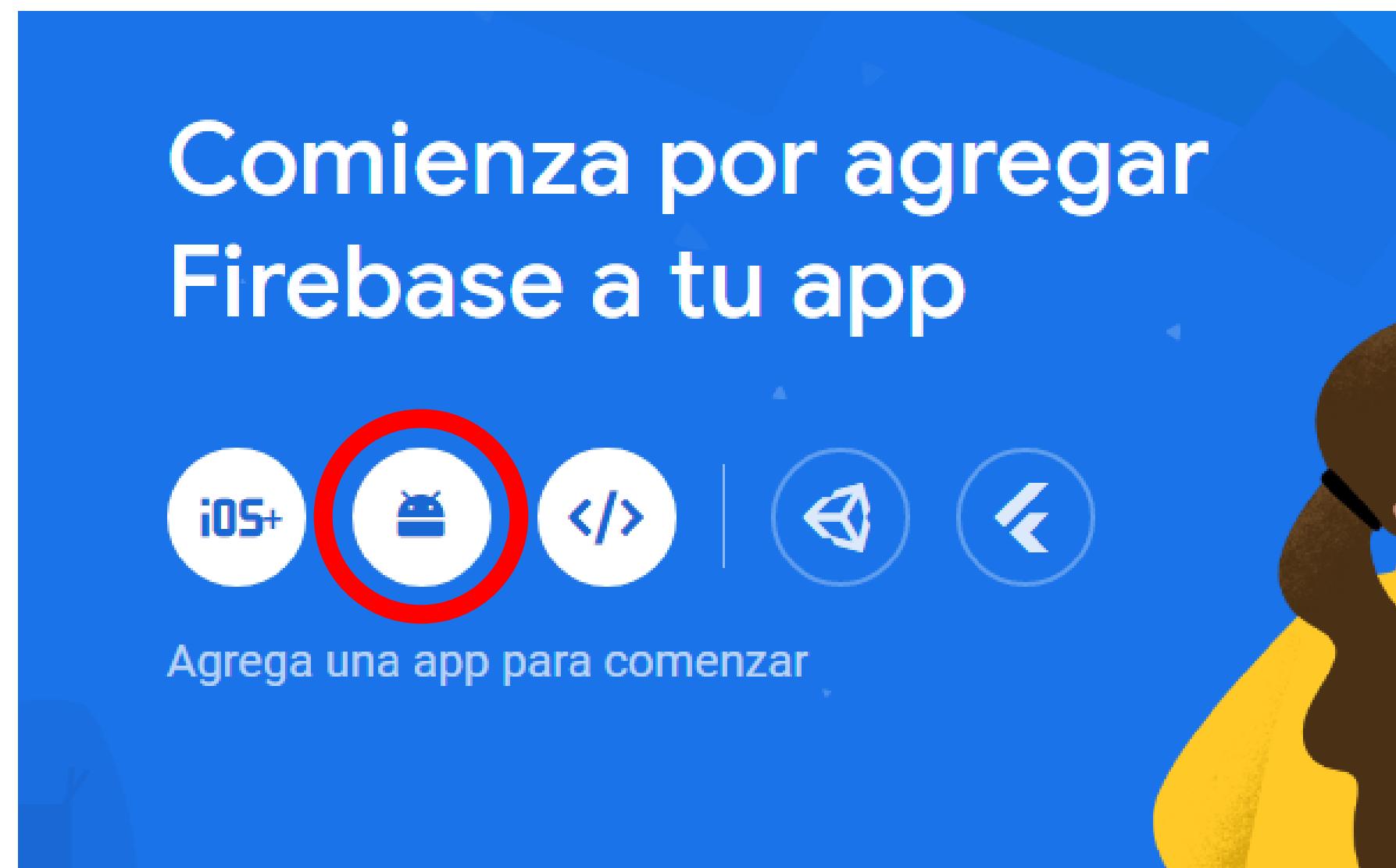
Google Analytics habilita las siguientes funciones:

- ✗ Pruebas A/B ②
- ✗ Segmentación de usuarios y orientación a ellos en los productos de Firebase ②
- ✗ Usuarios que no experimentan fallas ②
- ✗ Activadores de Cloud Functions basados en eventos ②
- ✗ Informes ilimitados y gratuitos ②

Habilitar Google Analytics para este proyecto
Recomendado

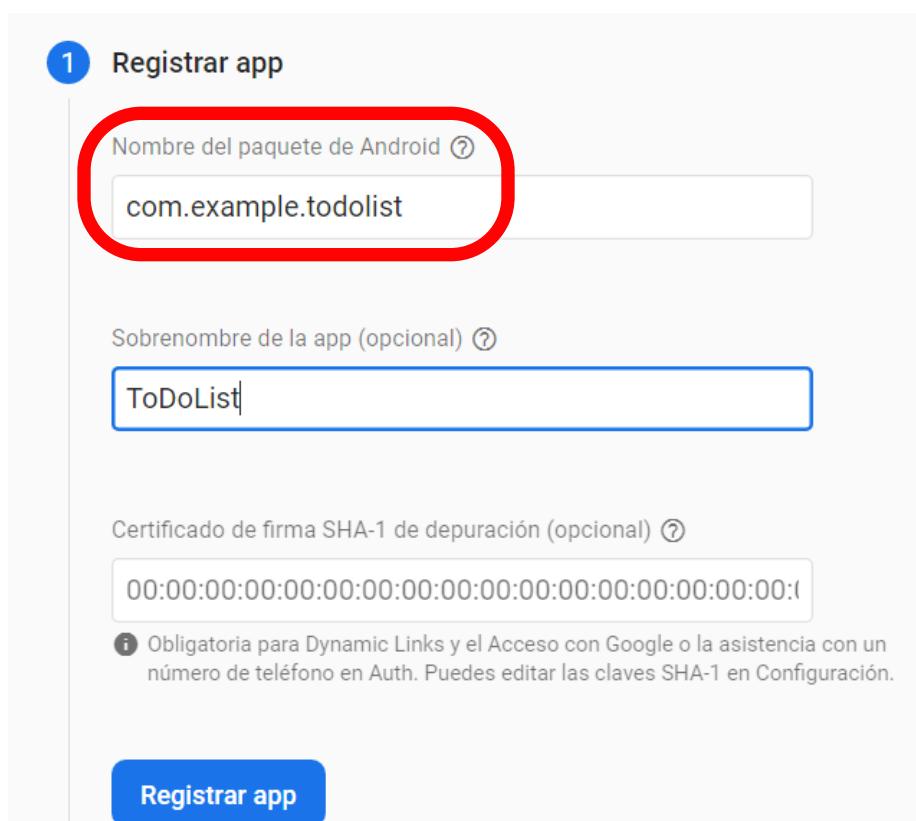
[Anterior](#) [Crear proyecto](#)

4. Indicar para qué dispositivo se utilizará:

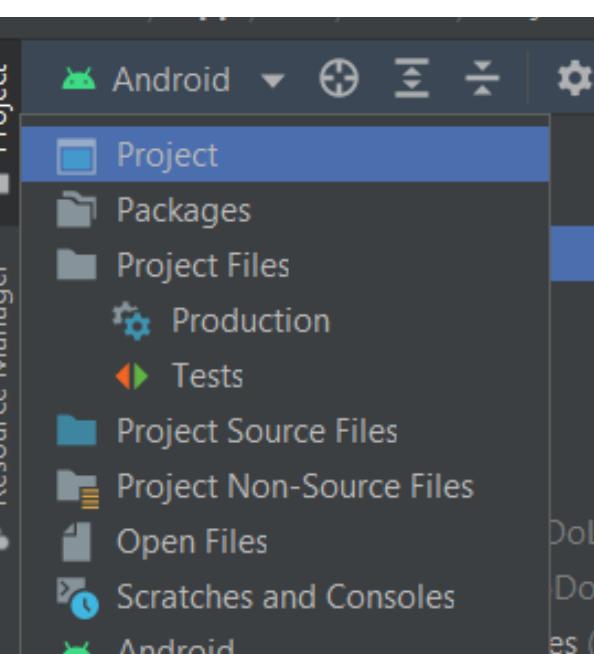


Configuración de FireBase

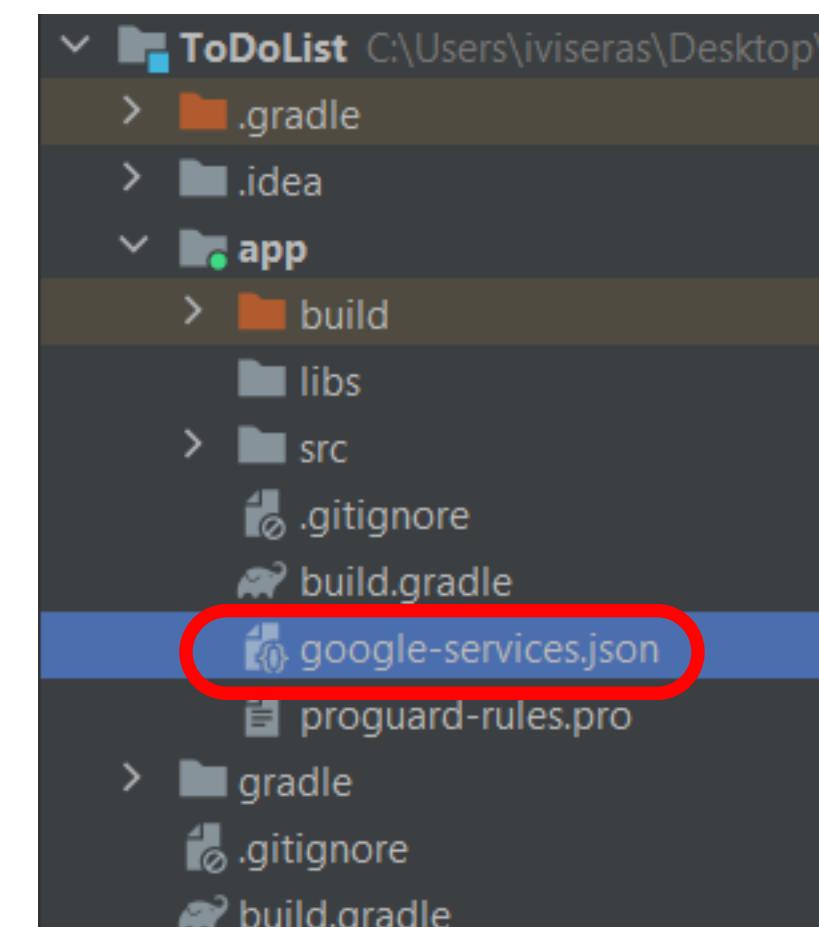
5. Para saber el nombre del paquete android vamos al fichero build.gradle (nivel app):



6. Descargamos el archivo .json que nos da la web y lo tendremos que añadir en la ubicación que nos indican, para eso tendremos que cambiar la vista de Andoid a la del proyecto:



7. Añadir el archivo a la carpeta app



8. Añadir la siguiente línea en el archivo build.gradle (a nivel de Proyecto):

```
build.gradle.kts (Project)

plugins {
    id("com.android.application") version "8.1.3" apply false
    id("com.google.gms.google-services") version "4.4.0" apply false
}
```

Configuración de FireBase

9. En el archivo build.gradle (a nivel de aplicación) añadimos lo siguiente en la sección de plugins:

```
build.gradle.kts (Module :app)

plugins {
    id("com.android.application")
    id("com.google.gms.google-services")
```

10. Añadimos lo siguiente en la sección de dependencies:

```
build.gradle.kts (Module :app)

dependencies {

    ...
    // Import the Firebase BoM
    implementation(platform("com.google.firebase:firebase-bom:32.7.0"))

    // TODO: Add the dependencies for Firebase products you want to use
    implementation("com.google.firebaseio:firebase-auth")
    implementation("com.google.firebaseio:firebase-firebase")
}
```

11. En la web de firebase ya podemos acceder a la consola.

12. En el AndroidManifest.xml tenemos que añadir los siguientes permisos, arriba del todo:

```
AndroidManifest.xml

<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
```

codetocimg.com

13. Accionamos el botón para que se sincronice el proyecto con todos los cambios:

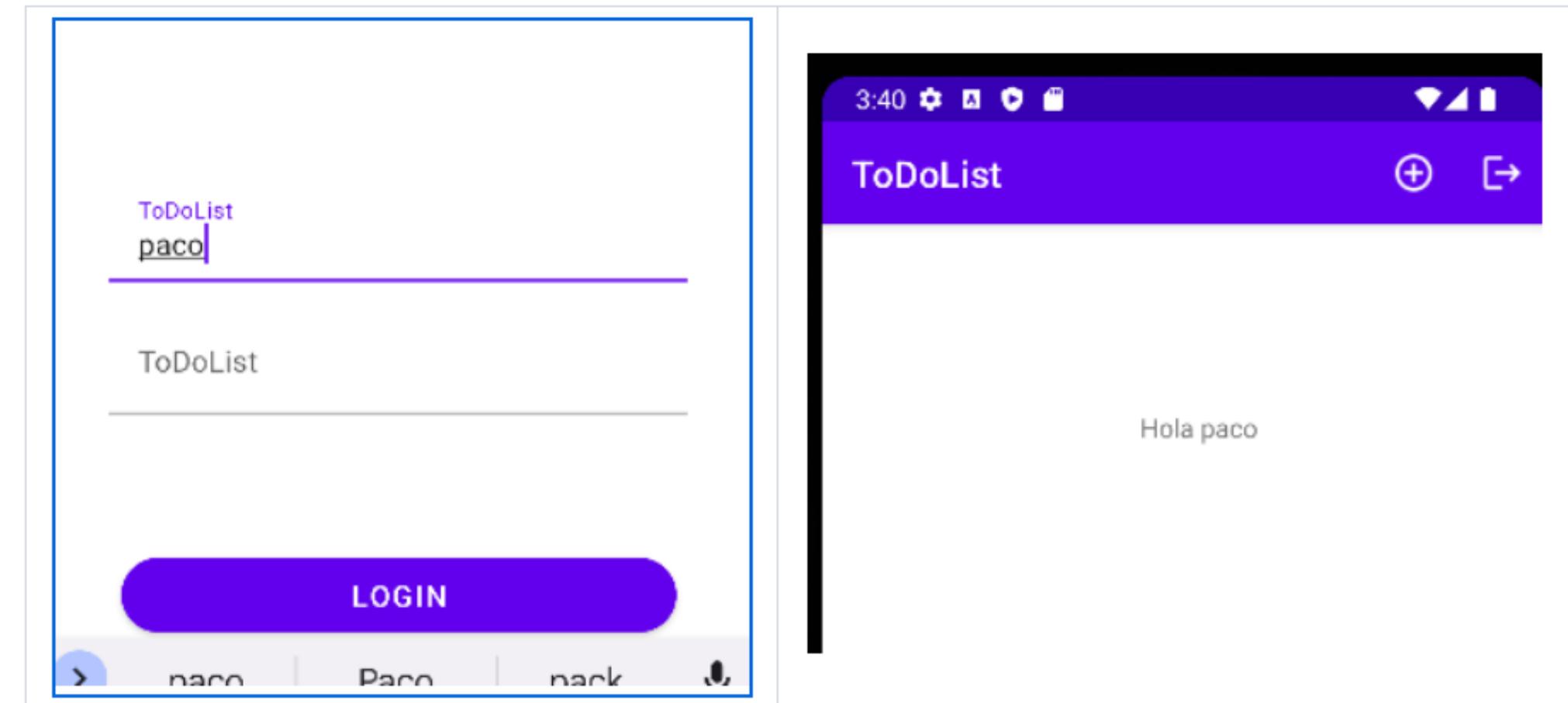
Sync Now Ignore these changes

Comunicación entre activities

Lo que queremos hacer es poder pasar un dato de una ventana a otra para eso:

```
Login.java  
... implements View.OnClickListener  
  
// añadir en onCreate  
findViewById(R.id.botonLogin).setOnClickListener(this);  
  
@Override  
public void onClick(View view) {  
    Intent intent = new Intent(this, MainActivity.class);  
    EditText nombreUsuario = findViewById(R.id.cajaNombre);  
    intent.putExtra("nombre", nombreUsuario.getText().toString());  
    startActivity(intent);  
}
```

```
MainActivity.java  
// añadir en onCreate  
String nombre = getIntent().getStringExtra("nombre");  
TextView etiquetaNomUser = findViewById(R.id.nombreUser);  
etiquetaNomUser.setText("Hola " + nombre);
```

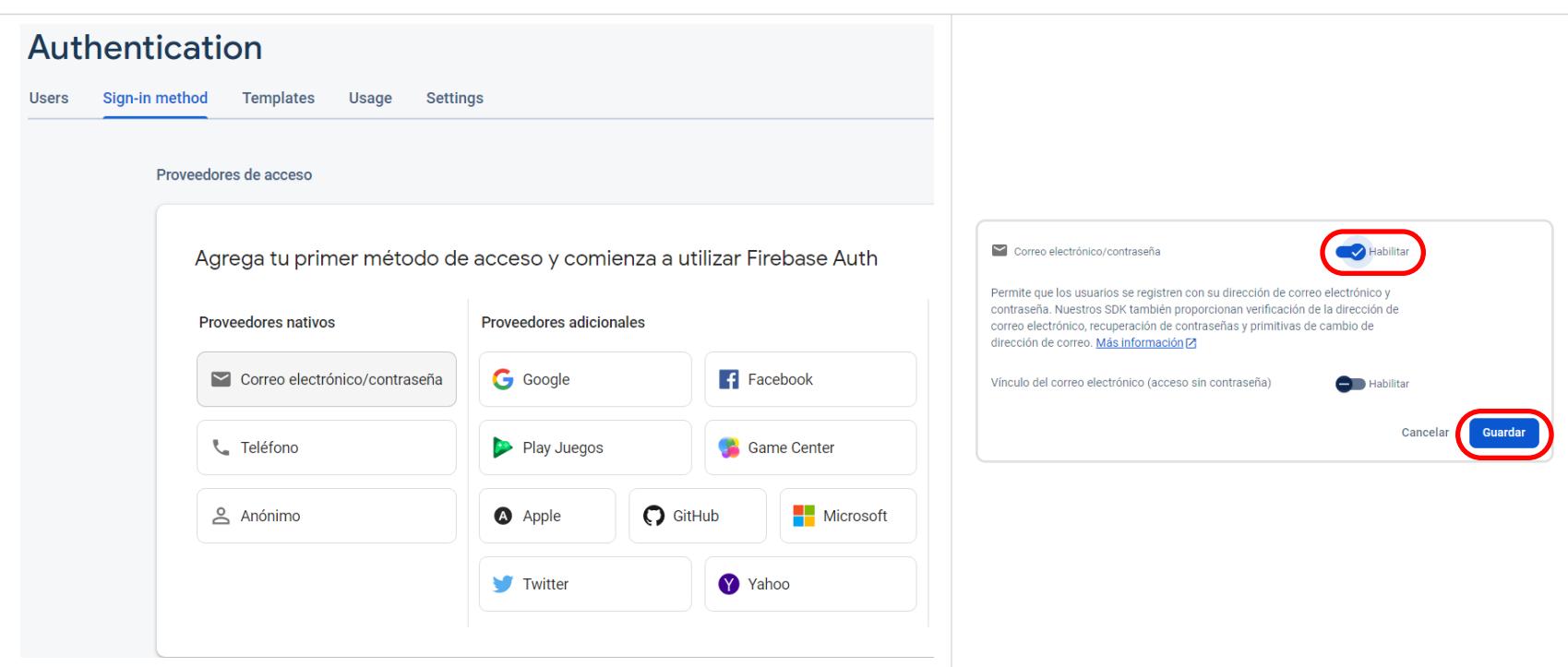


Login y registro de usuarios en Firebase

1. Accedemos a la consola de [Firebase](#) y, dentro de nuestro proyecto, vamos a la tarjeta *Authentication*:



2. Elegimos el proveedor que queramos utilizar, en nuestro caso usaremos el **correo electrónico**:



The top screenshot shows the 'Sign-in method' tab selected. Under 'Proveedores de acceso', 'Correo electrónico/contraseña' is listed with 'Estado' set to 'Habilitado'. The bottom screenshot shows the 'Users' tab selected, with a message stating 'Este proyecto todavía no tiene usuarios'.

3. Accederemos a la [guía](#) que tiene firebase y empezaremos a seguir los siguientes pasos.

4. En Login.java añadimos lo siguiente para guardar la referencia a Firebase y en el método onCreate se inicializa:

```
Login.java  
private FirebaseAuth mAuth;  
  
// onCreate  
mAuth = FirebaseAuth.getInstance();
```

Login y registro en FireBase

Acción de registro

1. Necesitamos crear dos variables en las que almacenaremos los datos de los campos correo y contraseña de la ventana login:

```
EditText emailText, passText;
```

2. Inicializamos las variables anteriores en el método onCreate:

```
emailText = findViewById(R.id.cajaCorreo);  
passText = findViewById(R.id.cajaContraseña);
```

3. Añadimos el siguiente código. Aquí es donde se da de alta al usuario en la base de datos y, por tanto, aparecerá en la consola de Firebase.

```
Login.java  
  
botonRegistro.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View v) {  
        // CREAR USUARIO EN FIREBASE  
  
        String email = emailText.getText().toString();  
        String password = passText.getText().toString();  
  
        mAuth.createUserWithEmailAndPassword(email, password)  
            .addOnCompleteListener(new OnCompleteListener<AuthResult>() {  
  
                @Override  
                public void onComplete(@NonNull Task<AuthResult> task) {  
                    if (task.isSuccessful()) {  
                        // Sign in success, update UI with the signed-in user's information  
                        Toast.makeText(Login.this, "Usuario registrado", Toast.LENGTH_SHORT).show();  
                        Intent intent = new Intent(Login.this, MainActivity.class);  
                        startActivity(intent);  
                    } else {  
                        // If sign in fails, display a message to the user.  
                        Toast.makeText(Login.this, "Authentication failed.", Toast.LENGTH_SHORT).show();  
                    }  
                }  
            });  
    }  
});
```



The screenshot shows a table with columns: Identificador, Proveedores, Fecha de creación, Fecha de acceso, and UID de usuario. There is one row with the following data:

Identificador	Proveedores	Fecha de creación	Fecha de acceso	UID de usuario
rcl@mail.com	✉	28 dic 2023	28 dic 2023	kRCrr0oORJMrHnY6133Hfafa...

At the bottom, there are navigation buttons for Filas por página (50), 1 - 1 of 1, and arrows.

Login y registro en FireBase

Acción de login

Este método lo vamos a realizar con una *función lambda*. Es recomendable usarlas ya que así no tenemos que sobrescribir el método onClick:

```
Login.java

botonLogin.setOnClickListener(view -> {
    // LOGUEAR USUARIO EN FIREBASE

    String email = emailText.getText().toString();
    String password = passText.getText().toString();

    mAuth.signInWithEmailAndPassword(email, password)
        .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful()) {
                    // Sign in success
                    startActivity(new Intent(getApplicationContext(), MainActivity.class));
                } else {
                    // If sign in fails, display a message to the user.
                    Toast.makeText(Login.this, "Authentication failed.", Toast.LENGTH_SHORT).show();
                }
            }
        });
});
```

Altas y Consultas

1. Desde la consola de Firebase accedemos al módulo de Cloud Firestore:



2. Clic en crear base de datos:



Crear base de datos

1 Establece el nombre y la ubicación 2 Reglas de seguridad

ID de la base de datos
(default)

Ubicación
eur3 (Europe)

! La configuración de ubicación es el lugar donde se almacenarán tus datos de Cloud Firestore.

! No podrás cambiar la ubicación después de configurarla. Además, la configuración de ubicación será la de tu bucket predeterminado de Cloud Storage.

Más información

Si habilitas Cloud Firestore, no podrás usar Cloud Datastore en este proyecto

Cancelar Siguiente

Crear base de datos

1 Establece el nombre y la ubicación 2 Reglas de seguridad

Después de definir la estructura de tus datos, debes crear reglas para protegerlos.

Más información

Iniciar en modo de producción

Tus datos son privados de forma predeterminada. El acceso de lectura/escritura de los clientes solo se otorgará como se indica en tus reglas de seguridad.

Comenzar en modo de prueba

Para permitir una configuración rápida, los datos se abren de forma predeterminada. Sin embargo, debes actualizar las reglas de seguridad en un plazo de 30 días a fin de habilitar el acceso de lectura/escritura a largo plazo para los clientes.

rules_version = '2';
service cloud.firestore {
 match /databases/{database}/documents {
 match /{document} {
 allow read, write: if
 request.time < timestamp.date(2024, 1, 27);
 }
 }
}

! Las reglas de seguridad predeterminadas del modo de prueba permiten que cualquier usuario con acceso a tu referencia de base de datos pueda ver, editar y borrar todos los datos durante los siguientes 30 días.

Si habilitas Cloud Firestore, no podrás usar Cloud Datastore en este proyecto

Cancelar Habilitar

Cloud Firestore

Datos Reglas Índices Uso Extensiones

(default)

+ Iniciar colección

Más funciones en Google Cloud

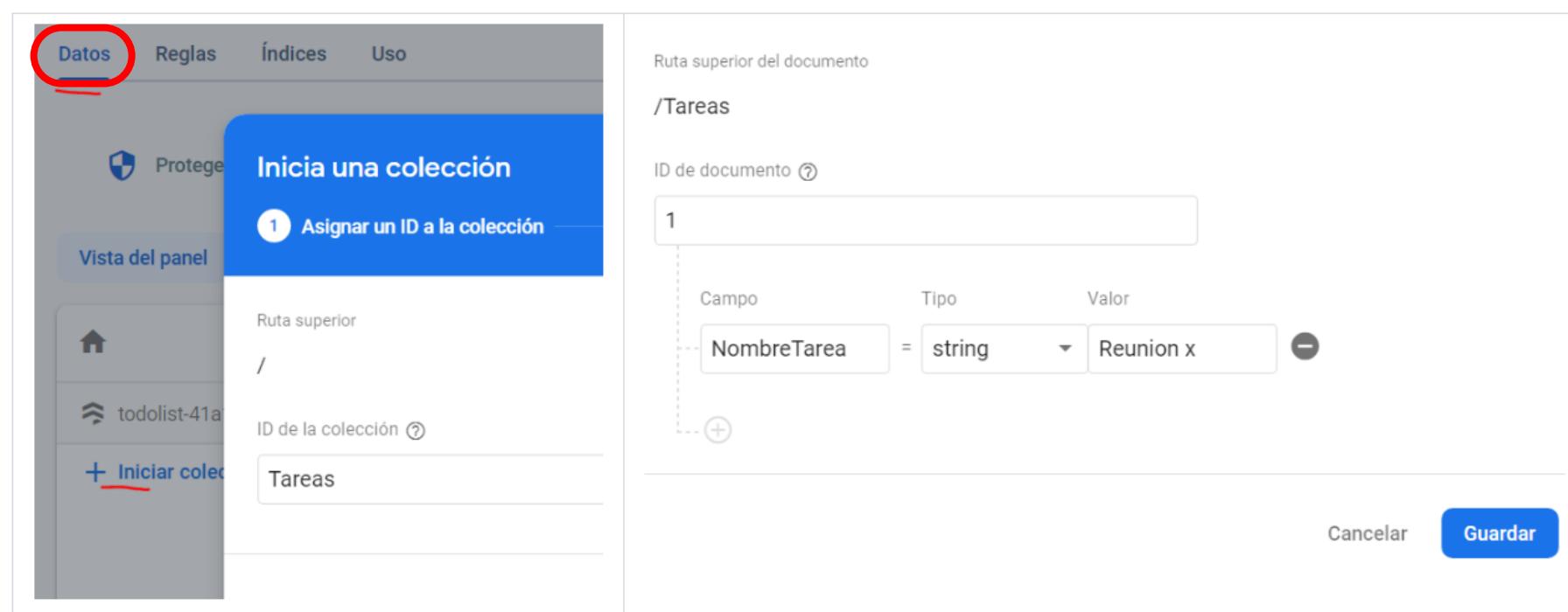
Tu base de datos está lista. Solo tienes que agregar datos.

Altas y Consultas

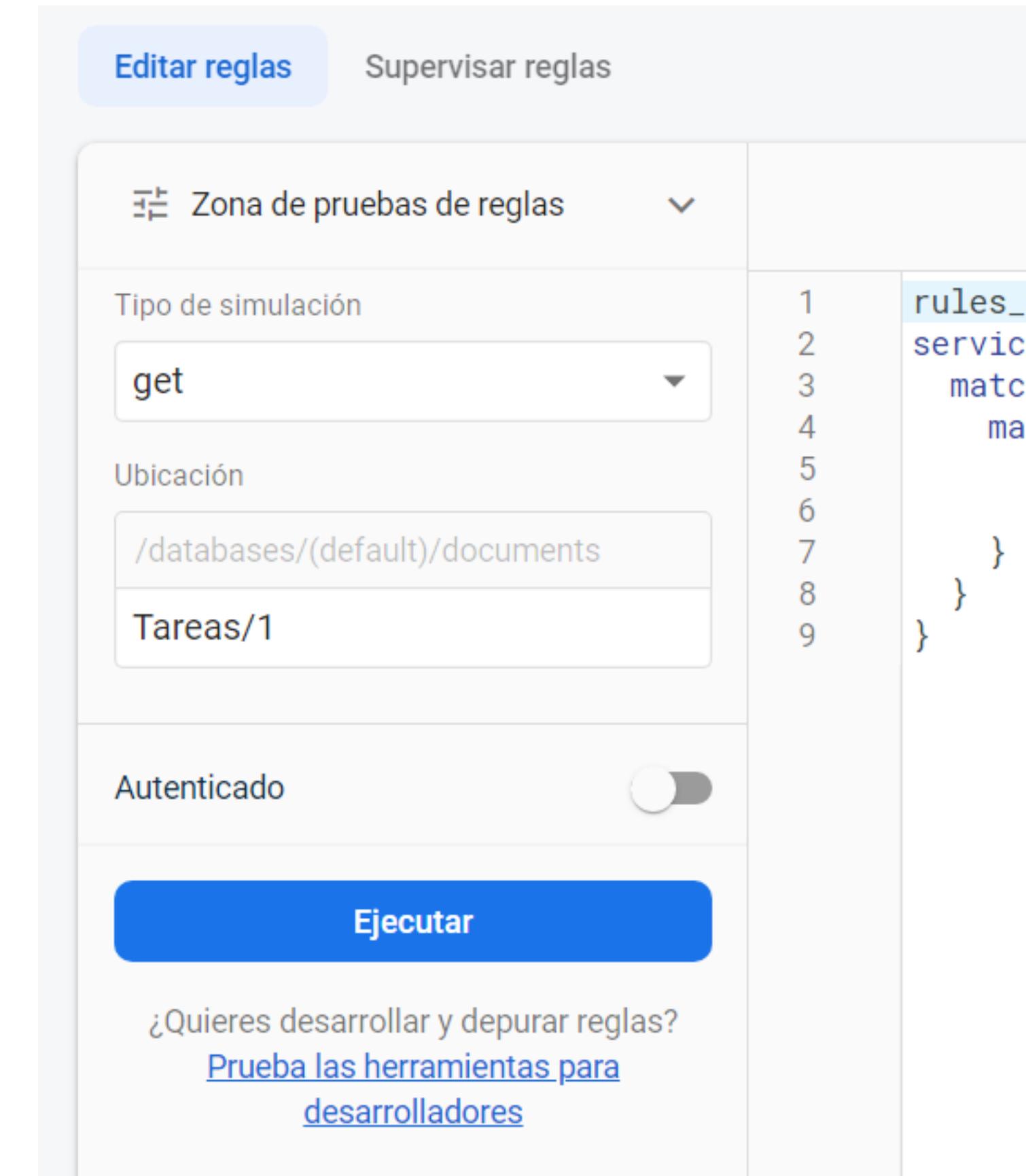
En la pestaña de reglas vamos a modificar la que se genera por defecto por esta otra. Haciendo este cambio vamos a permitir la lectura y escritura mientras que el usuario esté logeado:

```
match /{document=**} {  
  allow read, write: if request.auth != null;  
}
```

3. Ahora nos vamos a la pestaña de **Datos** y creamos una nueva colección:



4. Ahora podríamos ir a la zona de pruebas y comprobar si las acciones que indicamos se pueden realizar o no.



The screenshot shows the 'Zona de pruebas de reglas' section of the Firebase Rules Test Runner. The 'Tipo de simulación' dropdown is set to 'get'. The 'Ubicación' input field contains the path '/databases/(default)/documents/Tareas/1'. The 'Autenticado' toggle switch is turned off. At the bottom, there is a large blue 'Ejecutar' button. Below it, a message asks if you want to develop and debug rules, with a link to 'Prueba las herramientas para desarrolladores'.

Altas y Consultas

5. Nos vamos al build.gradle (nivel aplicación) y nos aseguramos de tener esta línea dentro de las dependencias:

```
implementation 'com.google.firebaseio:firebase-firebase'
```

6. En MainActivity.java añadimos todas las variables necesarias para guardar la referencia a la base de datos y al módulo de autenticación. Además de varios arrays donde iremos guardando los datos de las tareas:

```
FirebaseFirestore miBaseDatos;  
FirebaseAuth mAuth;  
String idUser;  
  
ListView listViewTareas;  
List<String> listaTareas = new ArrayList<>();  
List<String> listaIDTareas = new ArrayList<>();  
ArrayAdapter<String> mAdapterTareas;
```

7. En el método onCreate guardamos las instancias de FireStore y Authentication:

```
miBaseDatos = FirebaseFirestore.getInstance();  
mAuth = FirebaseAuth.getInstance();  
idUser = mAuth.getCurrentUser().getUid();  
  
listViewTareas = findViewById(R.id.ListView);
```

8. En el método onOptionsItemSelected, a la acción del botón positivo del cuadro de dialogo, añadimos la declaración de un mapeo de datos y lo añadimos a la bd:

```
public void onClick(DialogInterface dialog, int which) {  
    // añadir tarea a la base de datos y al ListView  
  
    String tarea = taskEditText.getText().toString();  
  
    Map<String, Object> miTarea = new HashMap<>();  
    miTarea.put("nombreTarea", tarea);  
    miTarea.put("idUsuario", idUser);  
  
    // Add a new document with a generated ID  
    miBaseDatos.collection("Tareas").add(miTarea);  
  
    Toast.makeText(MainActivity.this, "Tarea añadida", Toast.LENGTH_SHORT).show();  
}
```

Altas y Consultas

¿Cómo recibimos datos de una colección?

En este caso estamos sacando los datos en tiempo real según esta [guía](#). Crearemos un método en MainActivity para actualizar la interfaz de la app:

```
private void actualizarUI() {
    miBaseDatos.collection("Tareas")
        .whereEqualTo("idUsuario", idUser) // solo las tareas del usuario logueado
        .addSnapshotListener(new EventListener<QuerySnapshot>() {

            @Override
            public void onEvent(@Nullable QuerySnapshot value, @Nullable FirebaseFirestoreException e) {
                if (e != null) {
                    return;
                }

                //limpiar las listas
                listaIDTareas.clear();
                listaTareas.clear();

                //rellenar las listas con los datos de los documentos de la base de datos
                for (QueryDocumentSnapshot doc : value) {
                    listaIDTareas.add(doc.getId());
                    listaTareas.add(doc.getString("nombreTarea"));
                }

                //rellenar el listview con las listas

                if(listaTareas.size() == 0){ // si no hay documentos
                    listViewTareas.setAdapter(null);
                }else{
                    mAdapterTareas = new ArrayAdapter<>(MainActivity.this,R.layout.item_tarea,R.id.textViewTarea,listaTareas);
                    listViewTareas.setAdapter(mAdapterTareas);
                }
            }
        });
}
```

Tendremos que hacer una llamada a este método desde `onCreate` del MainActivity para que se ejecute.

Borrar datos y Logout

Borrar datos

```
MainActivity.java  
  
public void borrarTarea(View view){ //1  
    View parent = (View) view.getParent(); //2  
    TextView tareaTextView = parent.findViewById(R.id.textViewTarea); //3  
    String tarea = tareaTextView.getText().toString(); //4  
    int posicion = listaTareas.indexOf(tarea); //5  
  
    miBaseDatos.collection("Tareas").document(listaIDTareas.get(posicion)).delete(); //6  
}
```

1. El view hace referencia al botón que ha sido pulsado.
2. Obtenemos el view padre del botón.
3. Desde el padre podemos obtener el textView hijo.
4. Obtenemos el texto de la tarea (contenido).
5. Sacamos la posición (índice) de esa tarea en la lista.
6. Buscamos el documento de la base de datos con el identificador que está situado en ese índice para poder borrarlo.

Para que este método se pueda ejecutar tenemos que colocar en el xml del botón el llamamiento al método:

```
        android:onClick="borrarTarea"
```

Logout

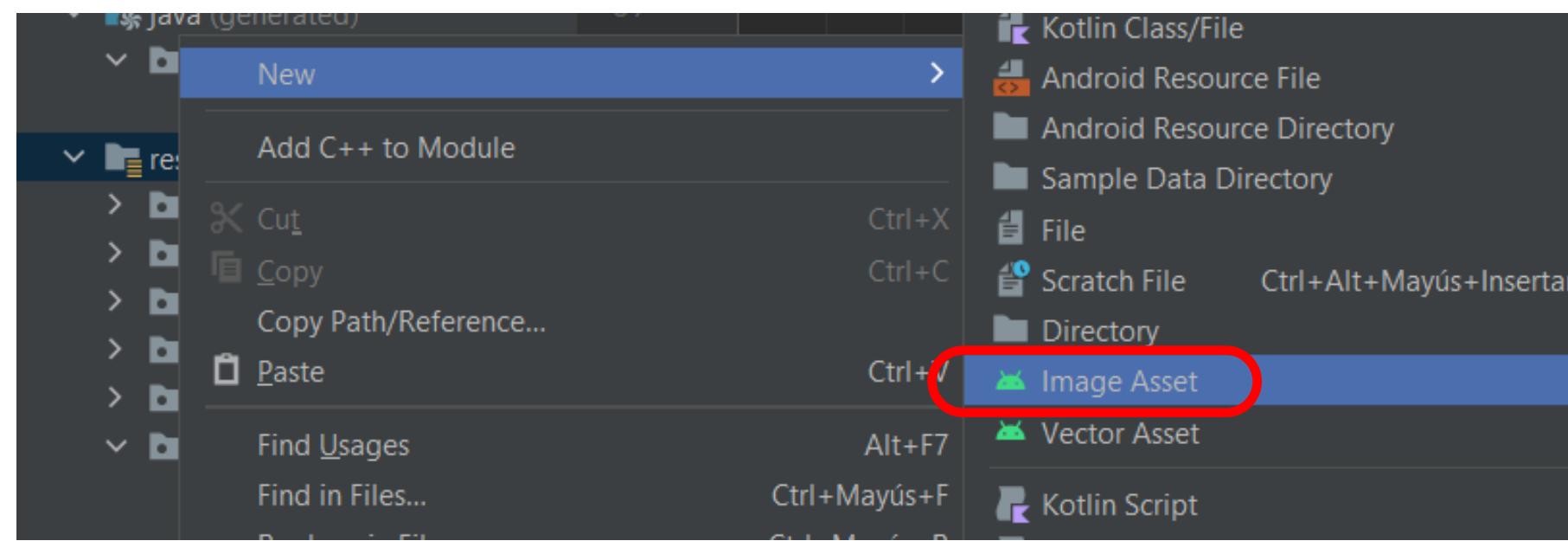
```
} else if(item.getItemId() == R.id.logout) {  
    // cierre de sesión de Firebase  
  
    mAuth.signOut();  
  
    startActivity(new Intent(MainActivity.this,Login.class));  
    finish();  
  
    return true;  
}
```

codetocimg.com

Acciones en Eventos

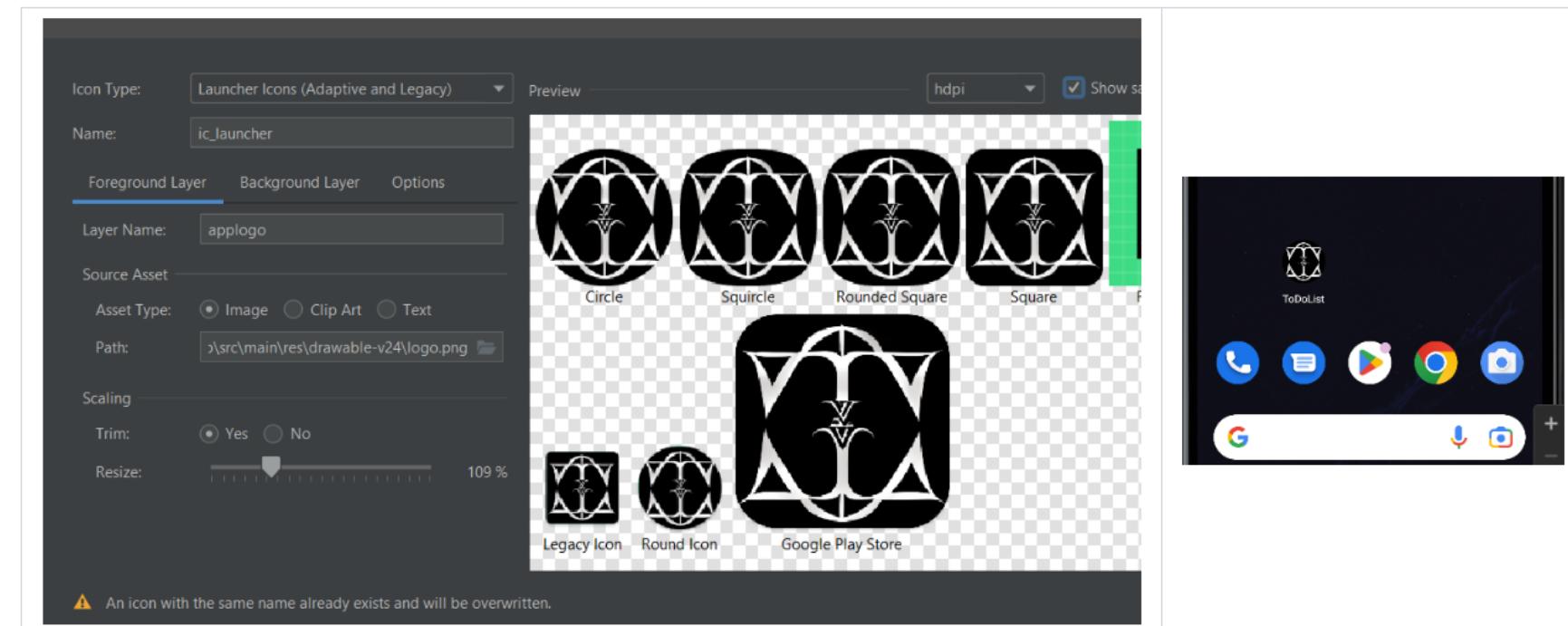
Cambiar logo app

1. Accedemos a la siguiente pestaña:



2. Indicar la foto que queremos que tenga nuestra app.

Colocar la foto en *res/drawable/*



Iconos para la app

Podemos entrar a [flaticon](#) y buscar algún logo que nos sea útil para la app.

Una vez descargado, lo guardamos en *res/drawable*, después nos vamos al archivo xml en el que lo queremos usar:

Colocar icono	Dar tamaño al icono	Dar color al icono
<code>app:icon="@drawable/cerca"</code>	<code>app:iconSize="20dp"</code>	<code>app:iconTint="@color/white"</code>

Icono a la derecha

`android:drawableRight="@drawable/cerca"`

unir LA UNIVERSIDAD
EN INTERNET