

### **ACTIVIDAD DE APRENDIZAJE DE LA UNIDAD 3**

**Estudiante:**

Abraham Yendes – 7502220015

**Al Docente:**

JHON ARRIETA

**Asignatura:**

Desarrollo APP



**Universidad de Cartagena**

Facultad de Ingeniería

Ingeniería de Software

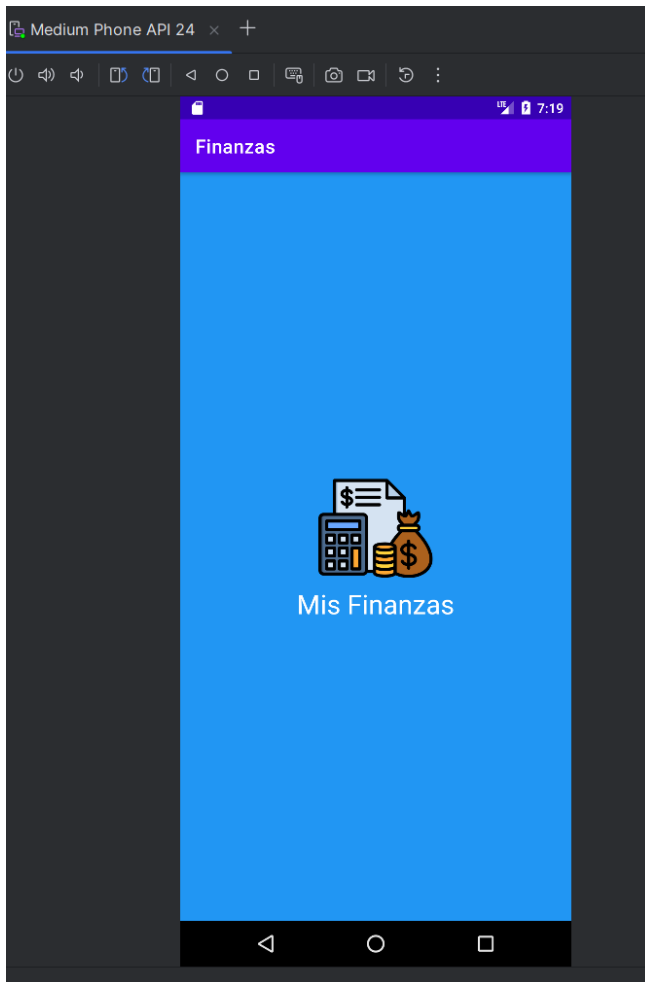
Cartagena-Bolívar

15 de junio de 2025

**ACTIVIDA 3 APP** Utilizar un IDE compatible con el **SDK de Android** sobre **Java** o Kotlin (**Android Studio**, Eclipse, VSC, Netbeans, AIDE, etc) para desarrollar una aplicación para dispositivos móviles que cumpla con los siguientes requerimientos.

- Registro y login de Usuario
- Registro de Ingresos y Gastos
- Link de Video [https://youtu.be/ctPo\\_iYa4RU](https://youtu.be/ctPo_iYa4RU)

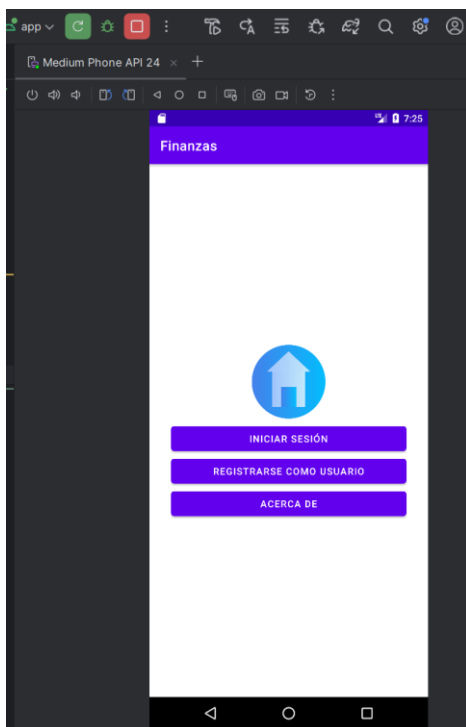
Pantalla de Bienvenida a la aplicación



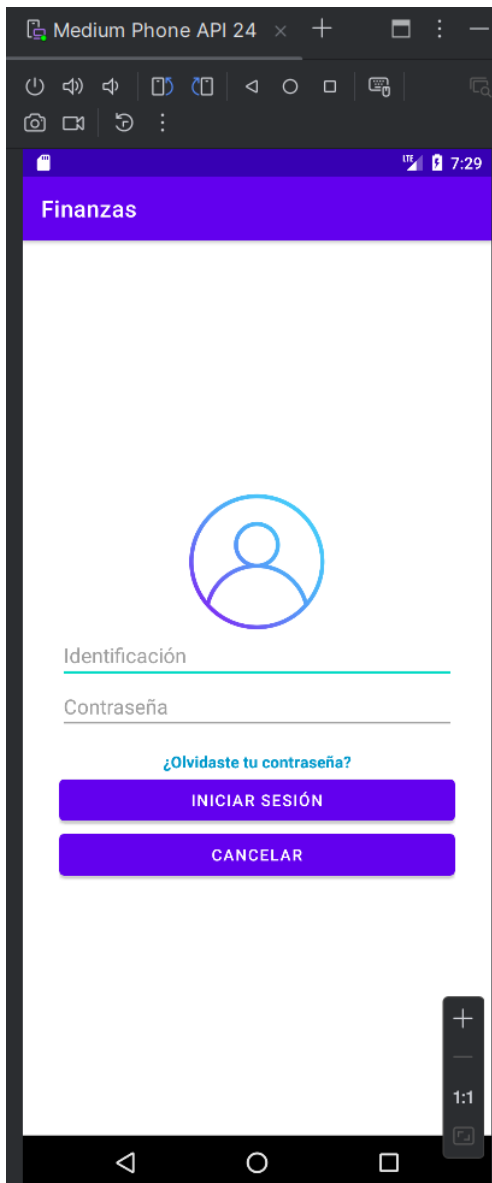
Bloque de Programación para Salir de la pantalla de Bienvenida y dirigirnos a inicio de App como transición y luego a menú Principal. Vista de programación de tiempo de animación

```
Finanzas ▾ Version control ▾ Medium Phone ▾
SplashActivity.java ×
1 package com.desarrolloapp.finanzas.activities;
2
3 import android.content.Intent;
4 import android.os.Bundle;
5 import android.os.Handler;
6 import androidx.appcompat.app.AppCompatActivity;
7
8 import com.desarrolloapp.finanzas.R;
9
10 public class SplashActivity extends AppCompatActivity {
11     @Override
12     protected void onCreate(Bundle savedInstanceState) {
13         super.onCreate(savedInstanceState);
14         setContentView(R.layout.activity_splash);
15         new Handler().postDelayed(() -> {
16             startActivity(new Intent( packageContext: this, MainMenuActivity.class));
17             finish();
18         }, delayMillis: 2000);
19     }
20 }
21
```

## Pantalla de Menú Principal



## Inicio de Sesión



```

LoginActivity.java x
15 public class LoginActivity extends AppCompatActivity {
16     private EditText etIdentificacion, etPassword;
17     private Button btnLogin, btnCancel;
18     private FirebaseAuth mAuth;
19     private DatabaseReference dbRef;
20
21     @Override
22     protected void onCreate(Bundle savedInstanceState) {
23         super.onCreate(savedInstanceState);
24         setContentView(R.layout.activity_login);
25
26         etIdentificacion = findViewById(R.id.etIdentificacion);
27         etPassword = findViewById(R.id.etPassword);
28         btnLogin = findViewById(R.id.btnLogin);
29         btnCancel = findViewById(R.id.btnCancel);
30
31         mAuth = FirebaseAuth.getInstance();
32         dbRef = FirebaseDatabase.getInstance().getReference(path: "usuarios");
33
34         btnLogin.setOnClickListener( View v -> {
35             String identificacion = etIdentificacion.getText().toString().trim();
36             String pass = etPassword.getText().toString().trim();
37
38             if (identificacion.isEmpty() || pass.isEmpty()) {
39                 Toast.makeText(context: this, text: "Ingrese identificación y contraseña", Toast.LENGTH_SHORT).show();
40                 return;
41             }
42
43             // Otpener email de identificacion para autenticar inicio de sesion
44             dbRef.child(identificacion).addListenerForSingleValueEvent(new ValueEventListener() {

```

Bloque de programación para ir a pantalla de Registro de Usuario nuevo

Finanzas

Seleccione

Número de Identificación

Contraseña

Nombre

Apellido

Seleccione

Email

Número de Teléfono

Seleccione

País

Ciudad

GUARDAR

CANCELAR

```

13  public class RegisterActivity extends AppCompatActivity {
14      3 usages
15      private Spinner spTipoIdentificacion, spGenero, spRol;
16      3 usages
17      private EditText etIdentificacion, etPassword, etNombre, etApellido, etEmail, etTelefono, etPais, etCiudad;
18      3 usages
19      private Button btnGuardar, btnCancelar;
20
21      @Override
22      protected void onCreate(Bundle savedInstanceState) {
23          super.onCreate(savedInstanceState);
24          setContentView(R.layout.activity_register);
25
26          spTipoIdentificacion = findViewById(R.id.spTipoIdentificacion);
27          spGenero = findViewById(R.id.spGenero);
28          spRol = findViewById(R.id.spRol);
29          etIdentificacion = findViewById(R.id.etIdentificacion);
30          etPassword = findViewById(R.id.etPassword);
31          etNombre = findViewById(R.id.etNombre);
32          etApellido = findViewById(R.id.etApellido);
33          etEmail = findViewById(R.id.etEmail);
34          etTelefono = findViewById(R.id.etTelefono);
35          etPais = findViewById(R.id.etPais);
36          etCiudad = findViewById(R.id.etCiudad);
37          btnGuardar = findViewById(R.id.btnGuardar);
38          btnCancelar = findViewById(R.id.btnCancelar);
39
40          btnGuardar = findViewById(R.id.btnGuardar);
41          btnCancelar = findViewById(R.id.btnCancelar);
42
43          // Configuración de spinner con arrays.xml
44          ArrayAdapter<CharSequence> adapterTipoId = ArrayAdapter.createFromResource(context, this,
45              R.array.tipos_identificacion, android.R.layout.simple_spinner_item);
46          adapterTipoId.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

```

```

btnGuardar.setOnClickListener( View v -> {

    String identificacion = etIdentificacion.getText().toString().trim();
    String tipoIdentificacion = spTipoIdentificacion.getSelectedItem().toString();
    String password = etPassword.getText().toString().trim();
    String nombre = etNombre.getText().toString().trim();
    String apellido = etApellido.getText().toString().trim();
    String genero = spGenero.getSelectedItem().toString();
    String email = etEmail.getText().toString().trim();
    String telefono = etTelefono.getText().toString().trim();
    String rol = spRol.getSelectedItem().toString();
    String pais = etPais.getText().toString().trim();
    String ciudad = etCiudad.getText().toString().trim();

    if (TextUtils.isEmpty(identificacion)) {
        etIdentificacion.setError("Campo requerido");
        return;
    }
}

```

```

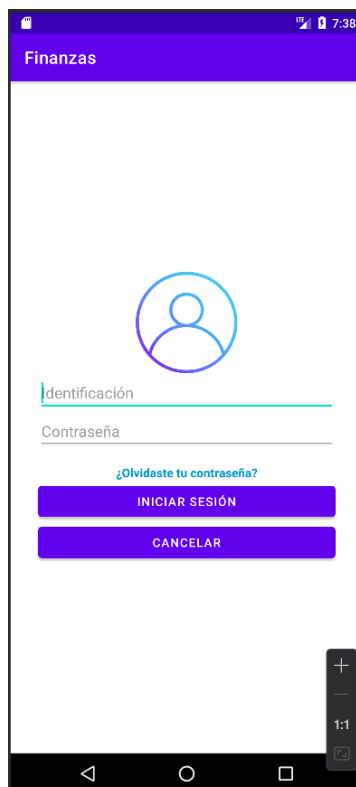
User user = new User();
user.identificacion = identificacion;
user.tipoIdentificacion = tipoIdentificacion;
user.password = password;
user.nombre = nombre;
user.apellido = apellido;
user.genero = genero;
user.email = email;
user.telefono = telefono;
user.rol = rol;
user.pais = pais;
user.ciudad = ciudad;

// Guardar en Base de Datos Firebase Realtime Database
DatabaseReference dbRef = FirebaseDatabase.getInstance().getReference( path: "usuarios");
dbRef.child(identificacion).setValue(user)
    .addOnSuccessListener( Void aVoid -> {
        Toast.makeText( context: this, text: "Registro Exitoso", Toast.LENGTH_SHORT).show();
        finish();
    })
    .addOnFailureListener( Exception e -> {
        Toast.makeText( context: this, text: "Error al registrar: " + e.getMessage(), Toast.LENGTH_SHORT).show();
    });

Cancelar.setOnClickListener( View v -> finish());

```

## Bloque de Programación para iniciar sesión



```

LoginActivity.java
13 import com.google.firebase.database.*;
14
15 public class LoginActivity extends AppCompatActivity {
16     private EditText etIdentificacion, etPassword;
17     private Button btnLogin, btnCancel;
18     private FirebaseAuth mAuth;
19     private DatabaseReference dbRef;
20
21     @Override
22     protected void onCreate(Bundle savedInstanceState) {
23         super.onCreate(savedInstanceState);
24         setContentView(R.layout.activity_login);
25
26         etIdentificacion = findViewById(R.id.etIdentificacion);
27         etPassword = findViewById(R.id.etPassword);
28         btnLogin = findViewById(R.id.btnLogin);
29         btnCancel = findViewById(R.id.btnCancel);
30
31         mAuth = FirebaseAuth.getInstance();
32         dbRef = FirebaseDatabase.getInstance().getReference("usuarios");
33
34         btnLogin.setOnClickListener() {
35             String identificacion = etIdentificacion.getText().toString().trim();
36             String pass = etPassword.getText().toString().trim();
37
38             if (identificacion.isEmpty() || pass.isEmpty()) {
39                 Toast.makeText(this, "Ingresa identificación y contraseña", Toast.LENGTH_SHORT).show();
40                 return;
41             }

```

```

// Obtener email de identificación para autenticar inicio de sesión
dbRef.child(identificacion).addListenerForSingleValueEvent(new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot snapshot) {
        if (snapshot.exists()) {
            String email = snapshot.child("email").getValue(String.class);
            if (email != null && !email.isEmpty()) {
                mAuth.signInWithEmailAndPassword(email, pass)
                    .addOnCompleteListener(LoginActivity.this, new Task<AuthResult> task -> {
                        if (task.isSuccessful()) {
                            FirebaseUser user = mAuth.getCurrentUser();
                            startActivity(new Intent(LoginActivity.this, UserMenuActivity.class));
                            finish();
                        } else {
                            Toast.makeText(LoginActivity.this, "Credenciales incorrectas", Toast.LENGTH_SHORT).show();
                        }
                    });
            } else {
                Toast.makeText(LoginActivity.this, "No se encontró el email asociado.", Toast.LENGTH_SHORT).show();
            }
        } else {
            Toast.makeText(LoginActivity.this, "Identificación no registrada.", Toast.LENGTH_SHORT).show();
        }
    }
})

@Override
public void onCancelled(DatabaseError error) {
    Toast.makeText(LoginActivity.this, "Error de base de datos.", Toast.LENGTH_SHORT).show();
}

```

Con la Identificación y el Password se recupera el correo del usuario y se realiza la autenticación para iniciar sesión.



Luego de iniciar sesión, se ingresa a registrar Ingreso y gasto en la cuenta de usuario, validando los datos y manteniendo sesión iniciada.



```

1 package com.desarrolloapp.finanzas.activities;
2
3 import android.content.Intent;
4 import android.os.Bundle;
5 import androidx.appcompat.app.AppCompatActivity;
6
7 import com.desarrolloapp.finanzas.R;
8
9 public class UserMenuActivity extends AppCompatActivity {
10     @Override
11     protected void onCreate(Bundle savedInstanceState) {
12         super.onCreate(savedInstanceState);
13         setContentView(R.layout.activity_user_menu);
14
15         findViewById(R.id.btnEditProfile).setOnClickListener( View v -> startActivity(new Intent( packageContext: this, EditProfileA
16         findViewById(R.id.btnIngresos).setOnClickListener( View v -> startActivity(new Intent( packageContext: this, IngresosActivit
17         findViewById(R.id.btnGastos).setOnClickListener( View v -> startActivity(new Intent( packageContext: this, GastoActivity.cla
18     }
19 }
20
21

```

Sa validan los datos registrados de Ingresos y Gastos en las listas

Finanzas

Nombre del Ingreso

Valor

Seleccione

Descripción

GUARDAR

EDITAR

ELIMINAR

CANCELAR

Ingreso Colombo - \$130

Fuente: Salario | salario 15/06

Unicolombo - \$150

Fuente: Negocio | Inversion en la Bolsa de Colombia

Ingreso Colombo - \$130

Fuente: Salario | salario 30/06

+

-

1:1

```
19 public class IngresosActivity extends AppCompatActivity {
30     private int spinnerSelectedIndex = 0;
31
32     @Override
33     protected void onCreate(Bundle savedInstanceState) {
34         super.onCreate(savedInstanceState);
35         setContentView(R.layout.activity_ingresos);
36
37         etNombreIngreso = findViewById(R.id.etNombreIngreso);
38         etValorIngreso = findViewById(R.id.etValorIngreso);
39         etDescripcionIngreso = findViewById(R.id.etDescripcionIngreso);
40         spFuenteIngreso = findViewById(R.id.spFuenteIngreso);
41         btnGuardarIngreso = findViewById(R.id.btnGuardarIngreso);
42         btnEditarIngreso = findViewById(R.id.btnEditarIngreso);
43         btnEliminarIngreso = findViewById(R.id.btnEliminarIngreso);
44         btnCancelarIngreso = findViewById(R.id.btnCancelarIngreso);
45         rvIngresos = findViewById(R.id.rvIngresos);
46
47         btnEliminarIngreso.setEnabled(false);
48
49         // Colocacion de arrays.xml por defecto en spinner
50         ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(
51             context: this,
52             R.array.fuentes_ingreso,
53             android.R.layout.simple_spinner_item
54         );
55         adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
56         spFuenteIngreso.setAdapter(adapter);
57
58         FirebaseUser currentUser = FirebaseAuth.getInstance().getCurrentUser();
59         if (currentUser != null) {
60             userId = currentUser.getId();
61             dbRef = FirebaseDatabase.getInstance().getReference(path: "ingresos").child(userId);
62         } else {
```

```

IngresosActivity.java x
19 public class IngresosActivity extends AppCompatActivity {
33     protected void onCreate(Bundle savedInstanceState) {
115
116         btnEditarIngreso.setOnClickListener( View v -> {
117             if (ingresoSeleccionado == null) {
118                 Toast.makeText( context: this, text: "Seleccione un ingreso para editar", Toast.LENGTH_SHORT).show();
119                 return;
120             }
121             String nombre = etNombreIngreso.getText().toString().trim();
122             String valor = etValorIngreso.getText().toString().trim();
123             String fuente = spFuenteIngreso.getSelectedItem().toString();
124             String descripcion = etDescripcionIngreso.getText().toString().trim();
125
126             if (TextUtils.isEmpty(nombre)) {
127                 etNombreIngreso.setError("Campo requerido");
128                 return;
129             }
130             if (TextUtils.isEmpty(valor)) {
131                 etValorIngreso.setError("Campo requerido");
132                 return;
133             }
134
135             ingresoSeleccionado.nombre = nombre;
136             ingresoSeleccionado.valor = valor;
137             ingresoSeleccionado.fuente = fuente;
138             ingresoSeleccionado.descripcion = descripcion;
139
140             dbRef.child(ingresoSeleccionado.id).setValue(ingresoSeleccionado)
141                 .addOnSuccessListener( Void aVoid -> {
142                     Toast.makeText( context: this, text: "Ingreso actualizado", Toast.LENGTH_SHORT).show();
143                     limpiarCampos();
144                     ingresoSeleccionado = null;
145                     btnEditarIngreso.setEnabled(false);
146                 })

```

finanzas > app > src > main > java > com > desarrolloapp > finanzas > activities > IngresosActivity

```

19 public class IngresosActivity extends AppCompatActivity {
33     protected void onCreate(Bundle savedInstanceState) {
148         });
149
150         btnEliminarIngreso.setOnClickListener( View v -> {
151             if (ingresoSeleccionado == null) {
152                 Toast.makeText( context: this, text: "Seleccione un ingreso para eliminar", Toast.LENGTH_SHORT).show();
153                 return;
154             }
155             new android.app.AlertDialog.Builder( context: this)
156                 .setTitle("Eliminar ingreso")
157                 .setMessage("¿Seguro que desea eliminar este ingreso?")
158                 .setPositiveButton( text: "Eliminar", ( DialogInterface dialog, int which) -> {
159                     dbRef.child(ingresoSeleccionado.id).removeValue()
160                     .addOnSuccessListener( Void aVoid -> {
161                         Toast.makeText( context: this, text: "Ingreso eliminado", Toast.LENGTH_SHORT).show();
162                         limpiarCampos();
163                         ingresoSeleccionado = null;
164                         btnEditarIngreso.setEnabled(false);
165                         btnEliminarIngreso.setEnabled(false);
166                     })
167                     .addOnFailureListener( Exception e -> Toast.makeText( context: this, text: "Error al eliminar", Toast
168                 })
169                 .setNegativeButton( text: "Cancelar", listener: null)
170                 .show();
171         });
172
173         btnCancelarIngreso.setOnClickListener( View v -> finish());
174         btnEditarIngreso.setEnabled(false);
175         btnEliminarIngreso.setEnabled(false);
176     }
177 }
178

```

Finanzas

Nombre del gasto

Valor

Seleccione

Descripción

GUARDAR

EDITAR

ELIMINAR

CANCELAR

Cena Restaurante - \$300.0

Categoría: Alimentación | Mercado 20/6

+

1:1

```
19 ▶ </> public class GastoActivity extends AppCompatActivity {  
    7 usages  
20     private EditText etNombreGasto, etValorGasto, etDescripcionGasto;  
    8 usages  
21     private Spinner spCategoriaGasto;  
    2 usages  
22     private Button btnGuardarGasto, btnCancelarGasto, btnEditarGasto, btnEliminarGasto;  
    3 usages  
23     private RecyclerView rvGastos;  
    3 usages  
24     private List<Gasto> listaGastos = new ArrayList<>();  
    3 usages  
25     private GastoAdapter gastoAdapter;  
    6 usages  
26     private DatabaseReference dbRef;  
    2 usages  
27     private String userId;  
    12 usages  
28     private Gasto gastoSeleccionado = null;  
29  
30     @Override  
31     protected void onCreate(Bundle savedInstanceState) {  
32         super.onCreate(savedInstanceState);  
33         setContentView(R.layout.activity_gasto);  
34  
35         etNombreGasto = findViewById(R.id.etNombreGasto);  
36         etValorGasto = findViewById(R.id.etValorGasto);  
37         etDescripcionGasto = findViewById(R.id.etDescripcionGasto);  
38         spCategoriaGasto = findViewById(R.id.spCategoriaGasto);  
39         btnGuardarGasto = findViewById(R.id.btnGuardarGasto);  
40         btnCancelarGasto = findViewById(R.id.btnCancelarGasto);  
41         btnEditarGasto = findViewById(R.id.btnEditarGasto);  
42         btnEliminarGasto = findViewById(R.id.btnEliminarGasto);  
43         rvGastos = findViewById(R.id.rvGastos);
```

```
19     public class GastoActivity extends AppCompatActivity {
31         protected void onCreate(Bundle savedInstanceState) {
63             }
64
65             // Configuración de la lista de gastos guardados
66             gastoAdapter = new GastoAdapter(listaGastos, Gasto gasto -> {
67                 gastoSeleccionado = gasto;
68                 etNombreGasto.setText(gasto.getNombre());
69                 etValorGasto.setText(String.valueOf(gasto.getValor()));
70                 etDescripcionGasto.setText(gasto.getDescripcion());
71                 for (int i = 0; i < spCategoriaGasto.getCount(); i++) {
72                     if (spCategoriaGasto.getItemAtPosition(i).toString().equals(gasto.getCategoria())) {
73                         spCategoriaGasto.setSelection(i);
74                         break;
75                     }
76                 }
77                 btnEditarGasto.setEnabled(true);
78                 btnEliminarGasto.setEnabled(true);
79             });
80             rvGastos.setLayoutManager(new LinearLayoutManager(context, this));
81             rvGastos.setAdapter(gastoAdapter);
82
83             cargarGastos();
84
85             btnGuardarGasto.setOnClickListener(View v -> {
86                 String nombre = etNombreGasto.getText().toString().trim();
87                 String valorStr = etValorGasto.getText().toString().trim();
88                 String categoria = spCategoriaGasto.getSelectedItem().toString();
89                 String descripcion = etDescripcionGasto.getText().toString().trim();
90
91                 if (TextUtils.isEmpty(nombre)) {
92                     etNombreGasto.setError("Campo requerido");
93                     return;
94                 }
95             });
96         }
97     }
98 }
```

GastoActivity.java ×

```
19     public class GastoActivity extends AppCompatActivity {
31         protected void onCreate(Bundle savedInstanceState) {
84
85             btnGuardarGasto.setOnClickListener( View v -> {
86                 String nombre = etNombreGasto.getText().toString().trim();
87                 String valorStr = etValorGasto.getText().toString().trim();
88                 String categoria = spCategoriaGasto.getSelectedItem().toString();
89                 String descripcion = etDescripcionGasto.getText().toString().trim();
90
91                 if (TextUtils.isEmpty(nombre)) {
92                     etNombreGasto.setError("Campo requerido");
93                     return;
94                 }
95                 if (TextUtils.isEmpty(valorStr)) {
96                     etValorGasto.setError("Campo requerido");
97                     return;
98                 }
99
100                 double valor;
101                 try {
102                     valor = Double.parseDouble(valorStr);
103                 } catch (NumberFormatException e) {
104                     etValorGasto.setError("Valor inválido");
105                     return;
106                 }
107
108                 String id = dbRef.push().getKey();
109                 Gasto gasto = new Gasto(id, nombre, valor, categoria, descripcion);
110
111                 dbRef.child(id).setValue(gasto)
112                     .addOnSuccessListener( Void aVoid -> {
113                         Toast.makeText( context: this, text: "Gasto guardado", Toast.LENGTH_SHORT).show();
114                         limpiarCampos();
115                     })
            }
```



© GastoActivity.java ×

```
19     public class GastoActivity extends AppCompatActivity {
31         protected void onCreate(Bundle savedInstanceState) {
118
119             btnEditarGasto.setOnClickListener( View v -> {
120                 if (gastoSeleccionado == null) {
121                     Toast.makeText( context: this, text: "Seleccione un gasto para editar", Toast.LENGTH_SHORT).show();
122                     return;
123                 }
124                 String nombre = etNombreGasto.getText().toString().trim();
125                 String valorStr = etValorGasto.getText().toString().trim();
126                 String categoria = spCategoriaGasto.getSelectedItem().toString();
127                 String descripcion = etDescripcionGasto.getText().toString().trim();
128
129                 if (TextUtils.isEmpty(nombre)) {
130                     etNombreGasto.setError("Campo requerido");
131                     return;
132                 }
133                 if (TextUtils.isEmpty(valorStr)) {
134                     etValorGasto.setError("Campo requerido");
135                     return;
136                 }
137
138                 double valor;
139                 try {
140                     valor = Double.parseDouble(valorStr);
141                 } catch (NumberFormatException e) {
142                     etValorGasto.setError("Valor inválido");
143                     return;
144                 }
145
146                 gastoSeleccionado.setNombre(nombre);
147                 gastoSeleccionado.setValor(valor);
148                 gastoSeleccionado.setCategoria(categoria);
149                 gastoSeleccionado.setDescripcion(descripcion);
```

```

GastoActivity.java x
19 public class GastoActivity extends AppCompatActivity {
31     protected void onCreate(Bundle savedInstanceState) {
119         btnEditarGasto.setOnClickListener( View v -> {
120             // TODO: Add your code here
158         })
159         .addOnFailureListener( Exception e -> Toast.makeText( context: this, text: "Error al actualizar", Toast.LENGTH_SH
160     });
161
162     btnEliminarGasto.setOnClickListener( View v -> {
163         if (gastoSeleccionado == null) {
164             Toast.makeText( context: this, text: "Selecciona un gasto para eliminar", Toast.LENGTH_SHORT).show();
165             return;
166         }
167         new android.app.AlertDialog.Builder( context: this)
168             .setTitle("Eliminar gasto")
169             .setMessage("¿Seguro que desea eliminar este gasto?")
170             .setPositiveButton( text: "Eliminar", ( DialogInterface dialog, int which) -> {
171                 dbRef.child(gastoSeleccionado.getId()).removeValue()
172                 .addOnSuccessListener( Void aVoid -> {
173                     Toast.makeText( context: this, text: "Gasto eliminado", Toast.LENGTH_SHORT).show();
174                     limpiarCampos();
175                     gastoSeleccionado = null;
176                     btnEditarGasto.setEnabled(false);
177                     btnEliminarGasto.setEnabled(false);
178                 })
179                 .addOnFailureListener( Exception e -> Toast.makeText( context: this, text: "Error al eliminar", Toast
180             })
181             .setNegativeButton( text: "Cancelar", listener: null)
182             .show();
183     });
184
185     btnCancelarGasto.setOnClickListener( View v -> finish());
186     btnEditarGasto.setEnabled(false);
187     btnEliminarGasto.setEnabled(false);

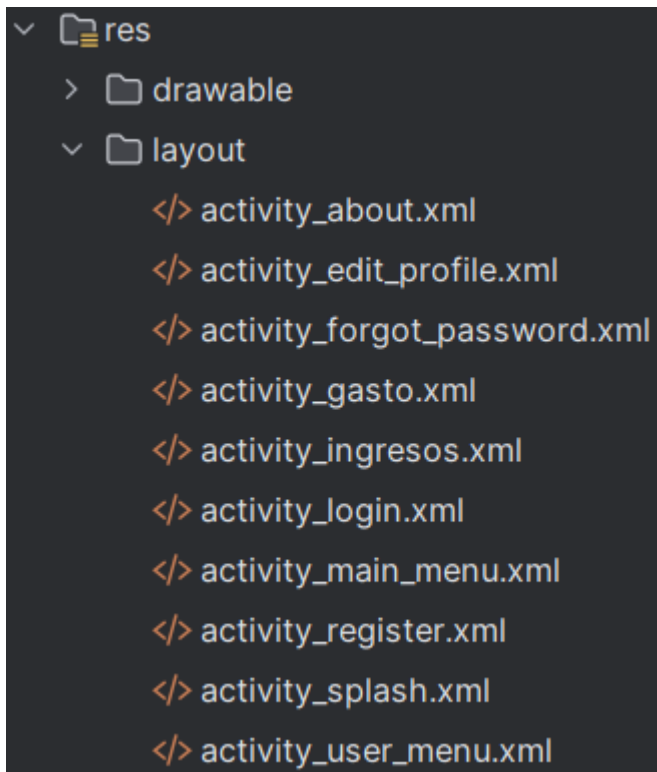
```

## AndroidManifest.xml

```
AndroidManifest.xml ×
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:tools="http://schemas.android.com/tools">
4
5      <application
6          android:allowBackup="true"
7          android:dataExtractionRules="@xml/data_extraction_rules"
8          android:fullBackupContent="@xml/backup_rules"
9          android:icon="@mipmap/ic_launcher"
10         android:label="@string/app_name"
11         android:roundIcon="@mipmap/ic_launcher_round"
12         android:supportsRtl="true"
13         android:theme="@style/Theme.Finanzas"
14         tools:targetApi="31" >
15
16         <activity android:name=".activities.SplashActivity"
17             android:exported="true"
18             android:label="@string/app_name"
19             android:theme="@style/Theme.Finanzas"
20             tools:ignore="MissingClass">
21             <intent-filter>
22                 <action android:name="android.intent.action.MAIN" />
23                 <category android:name="android.intent.category.LAUNCHER" />
24             </intent-filter>
25         </activity>
26         <activity android:name=".activities.AboutActivity" />
27         <activity android:name=".activities.MainMenuActivity" />
28         <activity android:name=".activities.LoginActivity" />
29         <activity android:name=".activities.RegisterActivity" />
30         <activity android:name=".activities.ForgotPasswordActivity" />
31         <activity android:name=".activities.UserMenuActivity" />
32         <activity android:name=".activities.EditProfileActivity" />
33         <activity android:name=".activities.GastoActivity" />

```

## Layout



```

▼ res
  > drawable
  ▼ layout
    </> activity_about.xml
    </> activity_edit_profile.xml
    </> activity_forgot_password.xml
    </> activity_gasto.xml
    </> activity_ingresos.xml
    </> activity_login.xml
    </> activity_main_menu.xml
    </> activity_register.xml
    </> activity_splash.xml
    </> activity_user_menu.xml

```

A screenshot of the Android Studio file explorer showing the 'res' directory. The 'layout' subdirectory is expanded, displaying a list of XML layout files. Each file name is preceded by an orange XML tag icon (</>).

## Arrays.xml para spinner

```
</> arrays.xml ×
1  <resources>
2      <string-array name="tipos_identificacion">
3          <item>Seleccione</item>
4          <item>Tarjeta de Identidad</item>
5          <item>Cédula de ciudadanía</item>
6          <item>Pasaporte</item>
7          <item>Cédula de extranjería</item>
8      </string-array>
9
10     <string-array name="generos">
11         <item>Seleccione</item>
12         <item>Masculino</item>
13         <item>Femenino</item>
14     </string-array>
15
16     <string-array name="roles">
17         <item>Seleccione</item>
18         <item>Usuario</item>
19         <item>Administrador</item>
20     </string-array>
21
22     <string-array name="fuentes_ingreso">
23         <item>Seleccione</item>
24         <item>Salario</item>
25         <item>Negocio</item>
26         <item>Inversiones</item>
27         <item>Otro</item>
28     </string-array>
29
30     <string-array name="categorias_gasto">
31         <item>Seleccione</item>
32         <item>Alimentación</item>
33         <item>Transporte</item>
34         <item>Salud</item>
```