TECNOLÓGICO
NACIONAL DE MEXICO

Escuela: Instituto Tecnologico de Milpa Alta.

Alumnos :

Linares Carbajal Luis Antonio.

- **Luis Villavicencio Hernandez.**
- **Sergio Zhony Miguel Garcia.**
- **Yamin De La Cruz Jimenez .**
- **Johan Gerardo Gonzales Lopez.**

Profesor: Maximiliano Roman Salgado.

Materia: Programacion para dispositivos moviles

Facultad: Ingenieria en sistemas computacionales.

Proyecto Mapache(MilpaMusic).

Semestre:7°

Contenido

Objetivo:

- Crear un reproductor de musica funcional .

objetivos secundarios:

- Lograr una interfaz intuitiva para el usuario.
- Crear un apartado para usuarios premium.
- Tener un buen repertorio musical.

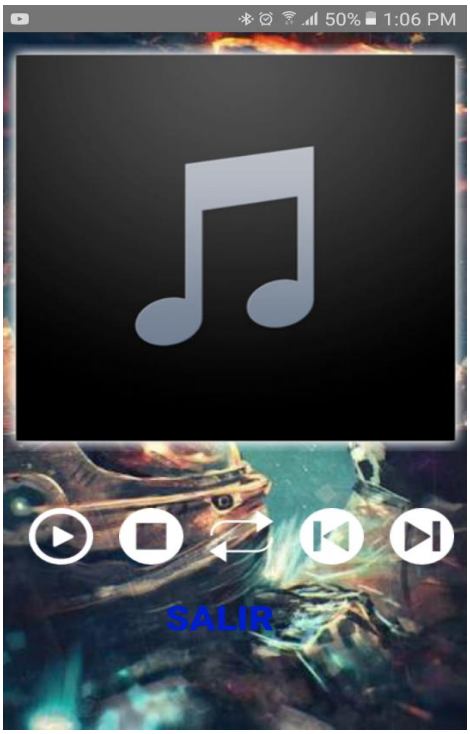

Ilustración 1(caratula)

Introduccion:

En este tutorial documento se dará una breve introducción del proceso de elaboración, así como el funcionamiento de (**Proyecto Mapache (Milpa Music)**), sus características y posteriormente un pequeño manual de usuario para que su funcionamiento y uso sea el más óptimo.

Lenguaje de desarrollo:

(Proyecto Mapache (Milpa Music)), está diseñado en Android Studio 4.1.0. se implementó como lenguaje fuente de programación Java, y XML. más adelante se verá este proceso.

Desarrollo de (Proyecto Mapache (Milpa Music))

Antes de comenzar a usar o desarrollar se recomienda descargar leer con detalle este documento para entender como funciona o como fue desarrolado el reproductor de musica.

Es importante mencionar que para que el reproductor de musica detecte un archivo de audio debe hacerse lo siguiente.

1. En la carpeta (res) del proyecto que se esta realizando en android Studio se creara otra carpeta con el nombre (raw).

Nota: es importante que la carpeta (raw) sea creada en el directorio res estrictamente y con los caracteres en minunsculas tal cual se esta indicando de lo contrario no se detectara la carpeta y causara problemas en el funcionamiento del reproductor.

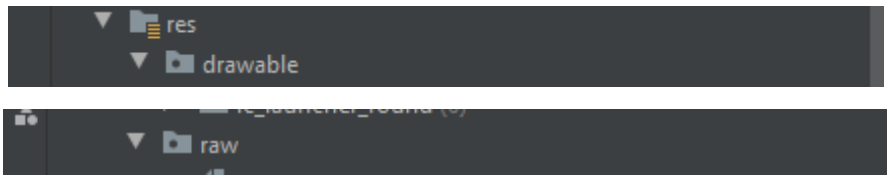

Ilustración 2(directorios)

Una ves creada la carpeta (raw) en el directorio (res). Se colocaran los archivos de audio que se incluiran en el reproductor de muscia.

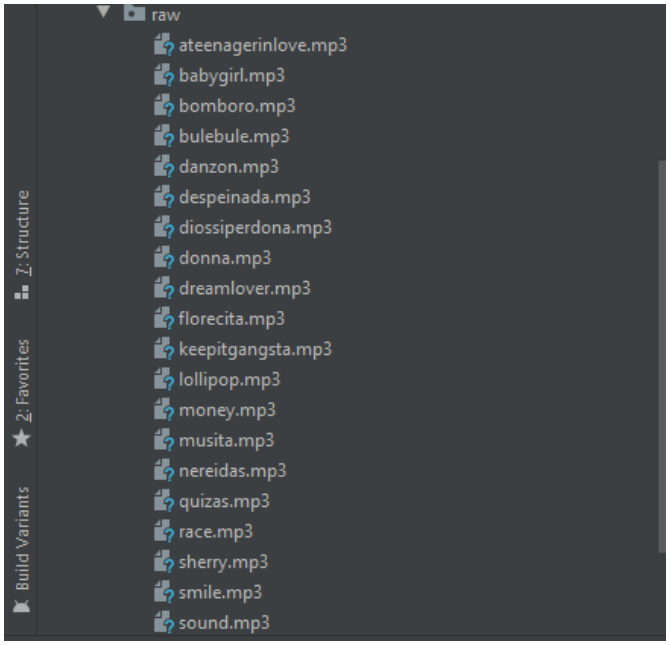

Ilustración 3(archos mp3)

Nota: todos los archvos deben ser formato .mp3 y deben ser ingresados con caracteres en minusculas y sin sombolos tal cual se muestra en la ilustracion 3 (archivos mp3).

2. Para insertar las imágenes, logos, portadas.etc. Que usara el reproductor es importante ir al directorio res.

En el directorio res exite una carpeta llamada (drawable) aquí se insertaran los archivos que usara el reproductor.

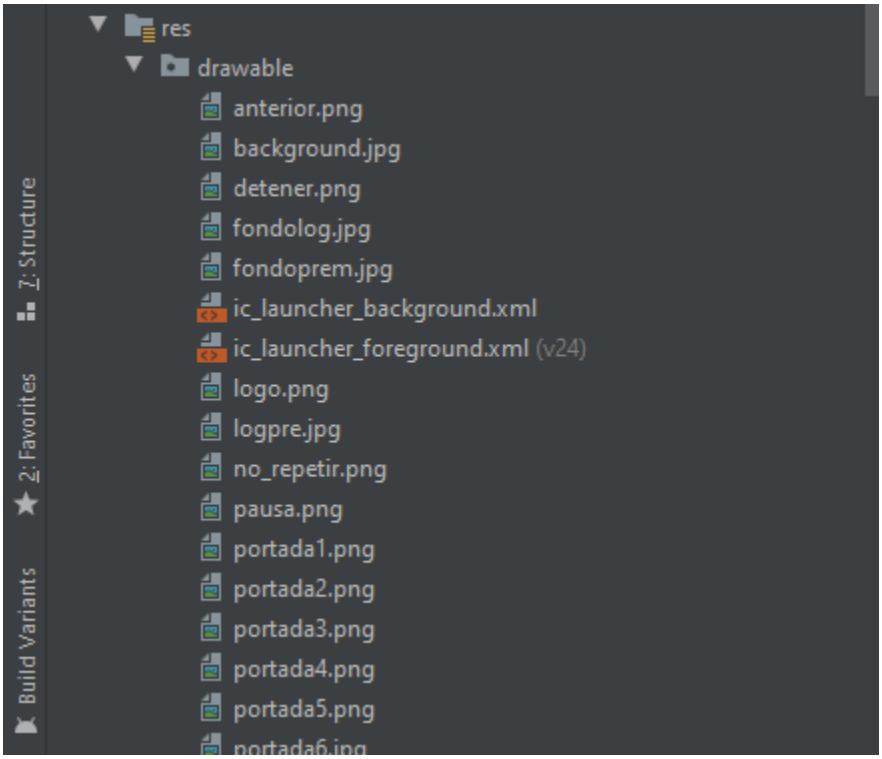

Ilustración 4(imágenes)

3. Ahora que ya se ingresaron todos los archivos que se usaran se procedeara a crear en nuestro main_activity.xml la interfas grafica de los usuarios Free.

Ilustración 5(interfas free)

Esta infermas de usuario gratuita es muy simple y facil de elaborar . cabe mencionar que los usuarios free tienen acceso a una biblioteca de soundtracks muy limitada por lo que si decean acceder a la biblioteca musical completa es necesario logearse en su cuenta premium para disfrutar de mas contenido.

Para lograr que se vea asi se debe revisar el codigo en el main_activity.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"
    android:background="@drawable/background"
    tools:context=".MainActivity">
```

Ilustración 6(activity 1)

Primero se cambia el fondo que tendra dicha interfaz grafica con el comando android:background="@drawable/background" para que no se vea el fondo blanco que nos pone el activity por defecto.

Para que el reproductor de musica cambie de caratula cuando salte o atrace a otra cancion se implementa estas lineas de codigo.

```
<ImageView
    android:id="@+id/imageView"
    android:layout_width="wrap_content"
    android:layout_height="380dp"
    android:layout_marginEnd="8dp"
    android:layout_marginRight="8dp"
    android:layout_marginTop="16dp"
    android:contentDescription="@string/app_name"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:srcCompat="@drawable/portada1" />
```

Ilustración 7(portada de canción código)

De esta manera posicionamos una imagen de las que se ingresaron en el drawable para las portadas de las canciones . cabe mencionar que esta linea de codigo posiciona una portada mas no la cambia . la parte del cambio de portada es logico y esto se mostrara mas adelante.

Una ves ingresando estas lineas de codigo nos tiene que mostrar en el diseño algo similar a la siguiente imagen:

Ilustración 8(image View portada)

Ahora hay que diseñar los botones que usara el reproductor de musica.

```
<Button
    android:id="@+id/btn_play"
    android:layout_width="50dp"
    android:layout_height="50dp"
    android:layout_marginStart="28dp"
    android:layout_marginLeft="28dp"
    android:layout_marginTop="448dp"
    android:background="@drawable/reproducir"
    android:onClick="PlayPause"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

  <Button
    android:id="@+id/btn_stop"
    android:layout_width="50dp"
    android:layout_height="50dp"
    android:layout_marginStart="96dp"
    android:layout_marginLeft="96dp"
    android:layout_marginTop="448dp"
    android:background="@drawable/detener"
    android:onClick="Stop"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

```
<Button
   android:id="@+id/btn_repetir"
   android:layout_width="50dp"
   android:layout_height="50dp"
   android:layout_marginStart="164dp"
   android:layout_marginLeft="164dp"
   android:layout_marginTop="448dp"
   android:background="@drawable/no_repetir"
   android:onClick="Repetir"
   app:layout_constraintStart_toStartOf="parent"
   app:layout_constraintTop_toTopOf="parent" />
<Button
   android:id="@+id/btn_anterior"
   android:layout_width="50dp"
   android:layout_height="50dp"
   android:layout_marginStart="228dp"
   android:layout_marginLeft="228dp"
   android:layout_marginTop="448dp"
   android:background="@drawable/anterior"
   android:onClick="Anterior"
   app:layout_constraintStart_toStartOf="parent"
   app:layout_constraintTop_toTopOf="parent" />
<Button
   android:id="@+id/btn_siguiente"
   android:layout_width="50dp"
   android:layout_height="50dp"
   android:layout_marginStart="292dp"
   android:layout_marginLeft="292dp"
   android:layout_marginTop="448dp"
   android:background="@drawable/siguiente"
   android:onClick="Siguiente"
```

```
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
<Button
    android:id="@+id/btn1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="76dp"
    android:layout_marginLeft="76dp"
    android:layout_marginTop="528dp"
    android:background="#00FFC107"
    android:onClick="Boton1"
    android:text="Iniciar secion"
    android:textColor="#FAF8F8"
    android:textSize="30dp"
    android:textStyle="bold"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    >
</Button>
```

Lo que se iso fue crear botones dandoles un margen y cordenadas para su posicion asi como un background para que en lugar de que nos aparesca un boton sin formato nos ponga una imagen que al momento de accionarla siga funcionando como un boton .

Nuestra interfaz ahora debe verse de esta manera.

Ilustración 9(interfaz usuarios free)

Ahora tenemos se debe programar la parte logica de los botones para que hagan las siguientes funciones:

a) Al momento de dar click en el boton reproducir este debe cambiar su icono de play a pausa , es decir si no se esta reproduciendo un sonido debe estar en modo play , pero al empesar a reproducir este debe cambiar a modo pausa y si esta reproduciendo debe mostrar su icono en modo pausa y al pausarse la musica debe estar en modo play como se ve en las siguientes ilustraciones.

Ilustración 11(modo play)

Ilustración 10(modo pausa)

b) Cuando se pulse el boton stop debe parar la musica y nos debe regresar al inicio de la lista de reproduccion .

c) Al accionar el boton repetir nos debe mostrar un toast si se esta repitiendo o no se esta repitiendo. Si se esta repitiendo debe cambiar el color del boton a rojo .

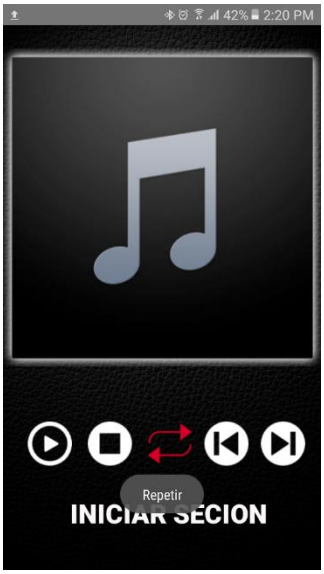

Ilustración 13(repetir)

Ilustración 12(no repetir)

d) Al accionar el los botones de siguiente y anterior deben de saltar o atrasar la cancion según el boton clickeado. Y en caso de que saltemos una cancion y la lista de reproduccion ya haya terminado debe mostrar un toast indicandonos que ya no hay mas musica para reproducir.

Ilustración 14(saltar/atrasar)

Para esto se deben usar las siguientes lineas de codigo en el java del activity que se esta trabajando.

```
package com.example.reproductorrap;
import android.content.Intent;
import android.media.MediaPlayer;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
public class MainActivity extends AppCompatActivity {
   Button play_pause, btn_repetir,btn1;
   MediaPlayer mp;
   ImageView iv;
   int repetir = 2, posicion = 0;
   MediaPlayer vectormp [] = new MediaPlayer [7];
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
      play_pause = (Button)findViewById(R.id.btn_play);
      btn_repetir = (Button)findViewById(R.id.btn_repetir);
      iv = (ImageView)findViewById(R.id.imageView);
      vectormp[0] = MediaPlayer.create(this, R.raw.race);
      vectormp[1] = MediaPlayer.create(this, R.raw.sound);
      vectormp[2] = MediaPlayer.create(this, R.raw.tea);
      vectormp[3] = MediaPlayer.create(this, R.raw.babygirl);
      vectormp[4] = MediaPlayer.create(this, R.raw.keepitgangsta);
      vectormp[5] = MediaPlayer.create(this, R.raw.smile);
      vectormp[6] = MediaPlayer.create(this, R.raw.money);
   }
```

```
public void Boton1(View view){
    btn1 = (Button)findViewById(R.id.btn1);
    btn1.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent i = new Intent(getBaseContext(), MainActivity2.class);
            Toast.makeText(getApplicationContext(), "inicia secion", Toast.LENGTH_SHORT).show();
            startActivity(i);
        }
    });
  }
  //Método para el botón PlayPause
  public void PlayPause(View view){
    if(vectormp[posicion].isPlaying()){//nos dira si se esta reproduccioendo una cacncion
        vectormp[posicion].pause();//a la hora de precionar el boton pausa , pausara la muscia
        play_pause.setBackgroundResource(R.drawable.reproducir);//aqui cambiara la imagen del boton de pausa a reproducir
        Toast.makeText(this, "Pausa", Toast.LENGTH_SHORT).show();
    } else {
        vectormp[posicion].start();//si no se esta reproducciendo se empezara a reproducir
        play_pause.setBackgroundResource(R.drawable.pausa);//si esta reproduciendo cambia el aspecto del boton a pausa
        Toast.makeText(this, "Play", Toast.LENGTH_SHORT).show();
    }
  }
```

```
//Método para el botón Stop
    public void Stop(View view){
        if(vectormp[posicion] != null){
            vectormp[posicion].stop();
            vectormp[0] = MediaPlayer.create(this, R.raw.race);
            vectormp[1] = MediaPlayer.create(this, R.raw.sound);
            vectormp[2] = MediaPlayer.create(this, R.raw.tea);
            vectormp[3] = MediaPlayer.create(this, R.raw.babygirl);
            vectormp[4] = MediaPlayer.create(this, R.raw.keepitgangsta);
            vectormp[5] = MediaPlayer.create(this, R.raw.smile);
            vectormp[6] = MediaPlayer.create(this, R.raw.money);
            posicion = 0;
            play_pause.setBackgroundResource(R.drawable.reproducir);
            iv.setImageResource(R.drawable.portada1);
            Toast.makeText(this, "Stop", Toast.LENGTH_SHORT).show();
        }
    }
    //Método repetir una pista
    public void Repetir(View view){
        if(repetir == 1){
            btn_repetir.setBackgroundResource(R.drawable.no_repetir);
            Toast.makeText(this, "No repetir", Toast.LENGTH_SHORT).show();
            vectormp[posicion].setLooping(false);
            repetir = 2;
        } else {
            btn_repetir.setBackgroundResource(R.drawable.repetir);
            Toast.makeText(this, "Repetir", Toast.LENGTH_SHORT).show();
            vectormp[posicion].setLooping(true);//este metodo setloopin nos permite siclar
valores booleanos
            repetir = 1;
        }
    }
```

```
//Método para saltar a la siguiente canción.
   public void Siguiente(View view){
      if(posicion < vectormp.length -1){ //el menos 1 es para que no haya desbordamiento de memoria
         if(vectormp[posicion].isPlaying()){
            vectormp[posicion].stop();
            posicion++;
            vectormp[posicion].start();
            if(posicion == 0){
               iv.setImageResource(R.drawable.portada1);
            } else if(posicion == 1){
               iv.setImageResource(R.drawable.portada2);
            }else if(posicion == 2){
               iv.setImageResource(R.drawable.portada3);
            }else if(posicion == 3){
               iv.setImageResource(R.drawable.portada4);
            }else if(posicion == 4){
               iv.setImageResource(R.drawable.portada5);
            }else if(posicion == 5){
               iv.setImageResource(R.drawable.portada6);
            }else if(posicion == 6){
               iv.setImageResource(R.drawable.portada7);
            }
         } else {
            posicion++;
            if(posicion == 0){
               iv.setImageResource(R.drawable.portada1);
            } else if(posicion == 1){
               iv.setImageResource(R.drawable.portada2);
            }else if(posicion == 2){
               iv.setImageResource(R.drawable.portada3);
            }else if(posicion == 3){
               iv.setImageResource(R.drawable.portada4);
```

```
        }else if(posicion == 4){
            iv.setImageResource(R.drawable.portada5);
        }else if(posicion == 5){
            iv.setImageResource(R.drawable.portada6);
        }else if(posicion == 6){
            iv.setImageResource(R.drawable.portada7);
        }
      }
    } else {
      Toast.makeText(this, "No hay más canciones", Toast.LENGTH_SHORT).show();
    }
}
```

```
//Método para regresar a la canción anterior
   public void Anterior(View view){
      if(posicion >= 1){
         if(vectormp[posicion].isPlaying()){
            vectormp[posicion].stop();
            vectormp[0] = MediaPlayer.create(this, R.raw.race);
            vectormp[1] = MediaPlayer.create(this, R.raw.sound);
            vectormp[2] = MediaPlayer.create(this, R.raw.tea);
            vectormp[3] = MediaPlayer.create(this, R.raw.babygirl);
            vectormp[4] = MediaPlayer.create(this, R.raw.keepitgangsta);
            vectormp[5] = MediaPlayer.create(this, R.raw.smile);
            vectormp[6] = MediaPlayer.create(this, R.raw.money);
            posicion--;
            if(posicion == 0){
               iv.setImageResource(R.drawable.portada1);
            } else if(posicion == 1){
               iv.setImageResource(R.drawable.portada2);
            }else if(posicion == 2){
               iv.setImageResource(R.drawable.portada3);
            }else if(posicion == 3){
               iv.setImageResource(R.drawable.portada4);
            }else if(posicion == 4){
               iv.setImageResource(R.drawable.portada5);
            }else if(posicion == 5){
               iv.setImageResource(R.drawable.portada6);
            }else if(posicion == 6){
               iv.setImageResource(R.drawable.portada7);
            }
            vectormp[posicion].start();
         }
```

```
else {
          posicion--;
            if(posicion == 0){
               iv.setImageResource(R.drawable.portada1);
            } else if(posicion == 1){
               iv.setImageResource(R.drawable.portada2);
            }else if(posicion == 2){
               iv.setImageResource(R.drawable.portada3);
            } else if(posicion == 3){
               iv.setImageResource(R.drawable.portada4);
            }else if(posicion == 4){
               iv.setImageResource(R.drawable.portada5);
            }else if(posicion == 5){
               iv.setImageResource(R.drawable.portada6);
            }else if(posicion == 6){
               iv.setImageResource(R.drawable.portada7);
            }
         }
      } else {
         Toast.makeText(this, "No hay más canciones", Toast.LENGTH_SHORT).show();
      }
   }
}
```

Lo que hace esta parte del codigo es colocar en arreglos las canciones para que las valla asignando en una posicion y de esta manera se le asigne una portada a cada posicion del arreglo para que cuando cambien de cancion cambie la portada .

Los botones cambian de icono al accionarse por que estamos ingresando una operación logica que va a cambiar el background del boton que se pulse como es el caso del boton de reproducir que cambia su color y en el caso del boton play que cambia su icono cuando este es accionaado.

El boton iniciar secion es para los usuarios que son premium y quieren ingresar al repertorio completo de musica . este boton lo que ace es diriguirlos a otro activity por medio de un metodo OnClick .

4. Ahora diseña la interfas del login para que los usuarios premium se puedan logear y disfrutar de toda la musica.

Para eso en el activity_main2.xml se implementara un <LinearLayout> para que puedamos empesar a darle forma a nuestra interfas y quede dde la siguiente manera:

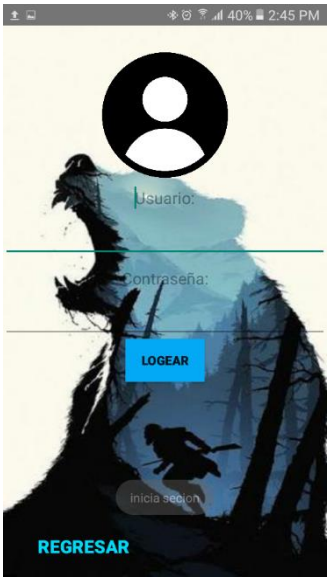

Ilustración 15(login)

Lo que hace este activiti es que nos pide ingresar un usuario y contraseña para validarla en sus datos y si lo encuentra nos mande al activity3 donde es la interfas grafica de los usuarios premium.

Ahora diseñaremos este login con las siguientes lineas de codigo .

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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"
   android:orientation="vertical"
   android:background="@drawable/fondolog"
   tools:context=".MainActivity2">
   <ImageView
      android:layout_marginTop="30dp"
      android:layout_width="140dp"
      android:layout_height="140dp"
      android:src="@drawable/logo"
      android:layout_gravity="center”>
   </ImageView>
   <EditText
      android:id="@+id/USER"
      android:layout_width="match_parent"
      android:layout_height="90dp"
      android:hint="Usuario:"
      android:textAlignment="center"
      android:gravity="center_horizontal">
   </EditText>
   <EditText
      android:id="@+id/PASS"
      android:layout_width="match_parent"
      android:layout_height="90dp"
      android:layout_y="40dp"
      android:hint="Contraseña:"
      android:inputType="textPassword"
```

```
    android:textAlignment="center"
    android:gravity="center_horizontal" />
  <Button
    android:id="@+id/btn3"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:layout_x="150dp"
    android:layout_y="690dp"
    android:background="#03A9F4"
    android:text="logear"
    android:textStyle="bold"
    />
  <Button
    android:id="@+id/btn2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:layout_marginTop="160dp"
    android:layout_marginRight="90dp"
    android:background="#00D2D6D8"
    android:onClick="Boton2"
    android:text="regresar"
    android:textColor="#00DBFB"
    android:textSize="20dp"
    android:textStyle="bold" />
</LinearLayout>
```

Lo que hace este codigo es insertar una imagen png y dimensionarla para poner el icono que identificamos como inicio de secion.

Tambien usamos los editText para ingresar caracters y un **InputType** =”Textpassword” que es para ocultar las contraseñas . y hacer segura la interfaz grafica.

Ahora tenemos que hacer funcional la interfas grafica para que valide nuestro usuario y contraseña que ingresemos y al momento de dar click en logear nos mande al activity3 donde se desarrollara la interfas de usuarios premium .

Ahora nos posecionamos en el java del activity del login para completar la parte logica de la interfaz.

Y ponemos las siguientes lineas de codigo.

```
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
public class MainActivity2 extends AppCompatActivity {
   EditText user,pass;
   Button boton1,btn2;
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main2);
      user = (EditText) findViewById(R.id.USER);
      pass = (EditText) findViewById(R.id.PASS);
      boton1 = (Button) findViewById(R.id.btn3);
      boton1.setOnClickListener(new View.OnClickListener() {
         @Override
         public void onClick(View v) {
            Hilo h1 = new Hilo();
            h1.run();
            h1.start();
            String Var_Usuario = user.getText().toString();
            String Var_Contra = pass.getText().toString();
            if (Var_Usuario.equals("Chuco") && Var_Contra.equals("12345")){
               Toast.makeText(getApplicationContext(), "Bienvenido a premium Chuco",
Toast.LENGTH_SHORT).show();
             Intent i= new Intent(getBaseContext(),MainActivity3.class);
             startActivity(i);
            }
```

```
            else
                Toast.makeText(getApplicationContext(), "Usuario o Contraseña No Valida",
Toast.LENGTH_SHORT).show();
        }
    });
  }
  public void Boton2(View view) {
    btn2 = (Button) findViewById(R.id.btn2);
    btn2.setOnClickListener(new View.OnClickListener() {
      @Override
      public void onClick(View v) {
        Intent i = new Intent(getBaseContext(), MainActivity.class);
        Toast.makeText(getApplicationContext(), "inicio", Toast.LENGTH_SHORT).show();
        startActivity(i);
      }
    });
  }
}
```

Como notaste estamos implementando herencia e una clase llamada Hilo. Este hilo nos sirve para hacer un proceso mientras se esta ejecutando otro.

Por lo que haora crearemos en el nuestro manifest una clase llamada Hilo y pondremos las siguientes lineas de codigo para que funcione correctamente.

```
import android.util.Log;

public class Hilo extends Thread {

   public void run(){

      try {

         Thread.sleep(2000);

         Log.d("Etiqueta 1", "Resultado.....");

      }catch (InterruptedException e){

         e.printStackTrace();

      }

   }

}
```

Lo que hace este codigo es que valide que el usuario y contraseña que has ingresado sean correctos . de ser correctos debe permitirte la entrada al activity3 de la interfas grafica para usuarios premium. De ser incorrecto el usuario o la contraseña que has ingresado, debe mostrart un toast con el mensaje “Contraseña o usuario incorrecto”.

Como viste tambien tenemos un boton (“Regresar”). En caso de que no seas premium y no tengas usuario o contraseña debes clickear regresar y te enviara a la interfas de usuarios Free.

5. El desarrollo de la interfas grafica de usuarios premium es muy sencillo solo reultiliza el codigo de la interfas de usuarios Free pero en esta asegurate de introducir en los arreglos todos los archivos para los usuarios premium , es decir oda la biblioteca de canciones , portadas, y por su puesto introduce un fondo mas llamativo para el usuario.

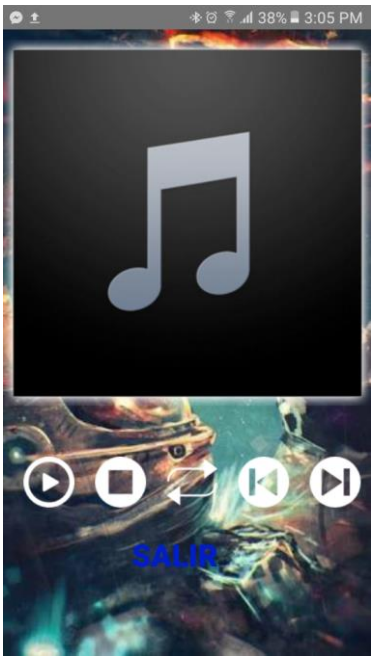

Ilustración 16(premium)