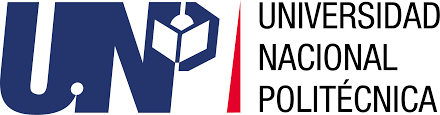
****

**Departamento de Ciencias Básicas y Tecnologías**

**Carrera: Ingeniería en Sistemas**

**Fecha:04/10/2025**

**Integrante: Jostin Antonio Obregon López**

**Docente: ING. Danilo Rodríguez Guerrero**

**Materia: Desarrollo de Aplicaciones V**

**Managua Nicaragua**

# Documentación Técnica del Proyecto: Aplicación de Cálculo de Salario

## 1. Datos Generales

Nombre del proyecto: Calcular Salario  
Lenguaje: Kotlin  
Entorno de desarrollo: Android Studio  
SDK mínimo: Android 7.0 (API 24) o superior  
Diseño de interfaz: XML  
Autor: Jostin Obregón

## 2. Descripción del Proyecto

La aplicación Calcular Salario permite al usuario ingresar su nombre completo y su salario mensual para obtener el cálculo automático de las deducciones correspondientes al INSS (7%) y al IR (10%), mostrando además el salario neto resultante.

Este documento detalla la estructura, implementación, pruebas y recomendaciones de mejora para entregar un informe técnico profesional, listo para integrarlo en un archivo Word (sin portada) para entrega académica o profesional.

**Objetivos**

**Objetivo general**

* Documentar de forma profesional la aplicación Calcular Salario, explicando la implementación y las decisiones tomadas, y proponiendo mejoras.

**Objetivos específicos**

* Describir la lógica de cálculo y su validación.
* Proveer código limpio y comentado para facilitar la evaluación.
* Sugerir mejoras de UX, accesibilidad y arquitectura para versiones futuras.

**Requisitos y entorno**

**Hardware/software mínimo para desarrollo y ejecución**

* Android Studio (última versión estable recomendada).
* JDK 11+.
* SDK Android 24+ instalado.

**Dependencias**

* No se requiere ninguna dependencia externa obligatoria para la versión actual (usa widgets nativos). Si en versiones futuras se incorpora Material Components, sí será necesario agregar la dependencia de com.google.android.material:material y ajustar temas.

## 3. Funcionalidades Principales

1. Entrada de datos: nombre completo y salario mensual.  
2. Procesamiento: cálculo de INSS e IR, y determinación del salario neto.  
3. Salida: muestra en pantalla los resultados.  
4. Botones: Calcular, Nuevo, y Salir.

## 4. Estructura del Proyecto

Archivos principales:  
  
• activity\_main.xml → Define la interfaz de usuario.  
• MainActivity.kt → Contiene la lógica del programa.  
• colors.xml → Define la paleta de colores.  
• strings.xml → Contiene los textos mostrados en la app.

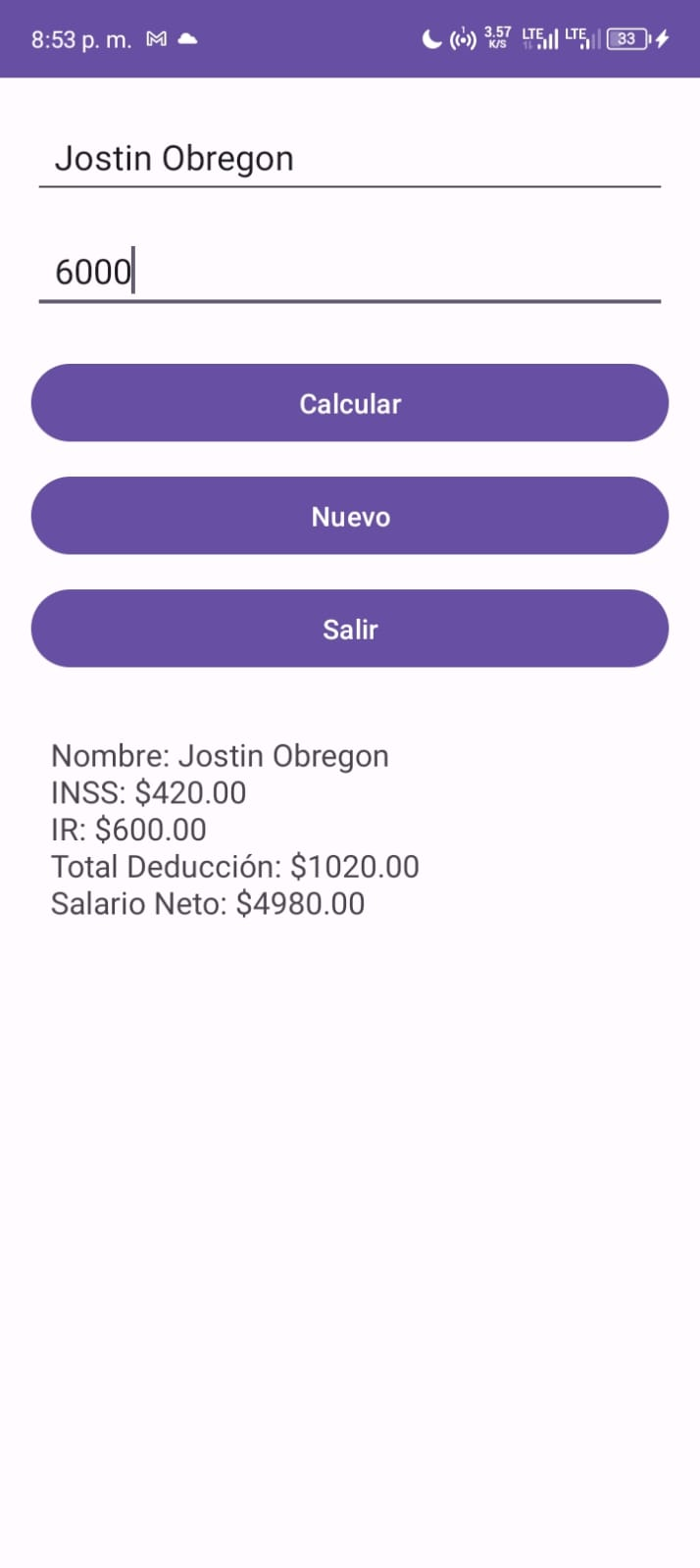
## 5. Lógica del Programa

Cálculos principales implementados en Kotlin:

* val inss = salario \* 0.07 // 7% de deducción por INSS  
  val ir = salario \* 0.10 // 10% de deducción por IR  
  val totalDeduccion = inss + ir  
  val salarioNeto = salario – totalDeduccion

## 6. Interfaz Gráfica (UI)

La aplicación presenta un diseño vertical simple con campos de entrada, botones de acción y una sección de resultados. Se utilizan colores suaves y una tipografía legible.



## 7. Ejecución del Proyecto

1. Abrir el proyecto en Android Studio.  
2. Conectar un dispositivo o iniciar un emulador.  
3. Presionar 'Run app'.  
4. Ingresar datos y presionar Calcular para ver los resultados.

## 8. Resultados Esperados

Ejemplo de salida:

Nombre: Jostin Obregón  
INSS: $420.00  
IR: $600.00  
Total Deducción: $1020.00  
Salario Neto: $4980.00

## 9. Buenas prácticas aplicadas

* Separación mínima entre UI y lógica (la función de cálculo está aislada).
* Textos en strings.xml para facilitar traducción.
* Formateo consistente con NumberFormat.
* Validaciones y mensajes claros para el usuario.
* Uso de minHeight para cumplir guidelines de accesibilidad táctil.

## 10. Conclusión

La aplicación Calcular Salario demuestra el uso correcto de Kotlin, validación de datos y manejo de interfaz XML. Es funcional, ligera y cumple con su propósito educativo y práctico.

**11. explicación de código**

**Activity\_main.xml**

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:padding="16dp">  
  
 <!-- Campo para el nombre completo -->  
 <EditText  
 android:id="@+id/etNombre"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:minHeight="48dp"  
 android:padding="12dp"  
 android:hint="@string/nombre\_completo"  
 android:inputType="textPersonName"  
 android:autofillHints="name"  
 android:layout\_marginBottom="10dp"/>  
  
 <!-- Campo para el salario mensual -->  
 <EditText  
 android:id="@+id/etSalario"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:minHeight="48dp"  
 android:padding="12dp"  
 android:hint="@string/salario\_mensual"  
 android:inputType="numberDecimal"  
 android:autofillHints="salary"  
 android:layout\_marginBottom="20dp"/>  
  
 <!-- Botón Calcular -->  
 <Button  
 android:id="@+id/btnCalcular"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:minHeight="48dp"  
 android:text="@string/calcular"  
 android:layout\_marginBottom="10dp"/>  
  
 <!-- Botón Nuevo -->  
 <Button  
 android:id="@+id/btnNuevo"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:minHeight="48dp"  
 android:text="@string/nuevo"  
 android:layout\_marginBottom="10dp"/>  
  
 <!-- Botón Salir -->  
 <Button  
 android:id="@+id/btnSalir"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:minHeight="48dp"  
 android:text="@string/salir"  
 android:layout\_marginBottom="20dp"/>  
  
 <!-- TextView para mostrar resultados -->  
 <TextView  
 android:id="@+id/tvResultados"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="@string/resultados"  
 android:textSize="16sp"  
 android:padding="10dp"/>  
</LinearLayout>

El diseño usa un **LinearLayout vertical** que organiza los elementos de arriba hacia abajo.

* **EditText etNombre:** campo para ingresar el nombre completo.
* **EditText etSalario:** campo para ingresar el salario mensual (solo números decimales).
* **Button btnCalcular:** calcula el salario neto.
* **Button btnNuevo:** limpia los campos y resultados.
* **Button btnSalir:** cierra la aplicación.
* **TextView tvResultados:** muestra el resultado del cálculo.

En conjunto, la interfaz es **simple, clara y funcional**, permitiendo al usuario ingresar datos, ejecutar el cálculo y ver los resultados fácilmente.

**MainActivity. Kt**

package com.syscraft.calcularsalario  
  
  
import android.os.Bundle  
import android.widget.Button  
import android.widget.EditText  
import android.widget.TextView  
import android.widget.Toast  
import androidx.appcompat.app.AppCompatActivity  
  
class MainActivity : AppCompatActivity() {  
  
 override fun onCreate(savedInstanceState: Bundle?) {  
 super.onCreate(savedInstanceState)  
 setContentView(R.layout.*activity\_main*)  
  
 val etNombre = findViewById<EditText>(R.id.*etNombre*)  
 val etSalario = findViewById<EditText>(R.id.*etSalario*)  
 val btnCalcular = findViewById<Button>(R.id.*btnCalcular*)  
 val btnNuevo = findViewById<Button>(R.id.*btnNuevo*)  
 val btnSalir = findViewById<Button>(R.id.*btnSalir*)  
 val tvResultados = findViewById<TextView>(R.id.*tvResultados*)  
  
 btnCalcular.setOnClickListener **{** val nombre = etNombre.*text*.toString().*trim*()  
 val salarioStr = etSalario.*text*.toString().*trim*()  
  
 if(nombre.*isEmpty*()) {  
 Toast.makeText(this, "Ingrese el nombre completo", Toast.*LENGTH\_SHORT*).show()  
 return@setOnClickListener  
 }  
  
 if(salarioStr.*isEmpty*()) {  
 Toast.makeText(this, "Ingrese el salario mensual", Toast.*LENGTH\_SHORT*).show()  
 return@setOnClickListener  
 }  
  
 val salario = salarioStr.*toDoubleOrNull*()  
 if(salario == null || salario <= 0) {  
 Toast.makeText(this, "Ingrese un salario válido", Toast.*LENGTH\_SHORT*).show()  
 return@setOnClickListener  
 }  
  
 val inss = salario \* 0.07 // 7% INSS  
 val ir = salario \* 0.10 // 10% IR  
 val totalDeduccion = inss + ir  
 val salarioNeto = salario - totalDeduccion  
  
 tvResultados.*text* = """  
 Nombre: $nombre  
 INSS: $${"%.2f".*format*(inss)}  
 IR: $${"%.2f".*format*(ir)}  
 Total Deducción: $${"%.2f".*format*(totalDeduccion)}  
 Salario Neto: $${"%.2f".*format*(salarioNeto)}  
 """.*trimIndent*()  
 **}** btnNuevo.setOnClickListener **{** etNombre.*text*.clear()  
 etSalario.*text*.clear()  
 tvResultados.*text* = "Resultados"  
 **}** btnSalir.setOnClickListener **{** finish()  
 **}** }  
}

Este archivo contiene la lógica principal de la aplicación.

* onCreate: inicia la actividad y conecta el código con los elementos del diseño XML (findViewById).
* btnCalcular:
  + Toma los datos del nombre y salario.
  + Verifica que no estén vacíos y que el salario sea válido.
  + Calcula las deducciones:
    - INSS = 7% del salario.
    - IR = 10% del salario.
  + Muestra en el TextView el detalle del cálculo y el salario neto.
* btnNuevo: limpia los campos y reinicia el texto de resultados.
* btnSalir: cierra la aplicación con finish().

En resumen:  
El código controla las acciones del usuario, valida los datos y realiza los cálculos necesarios para mostrar el salario neto con sus deducciones de forma clara y precisa.

<resources>  
 <color name="colorPrimary">#1976D2</color>  
 <color name="colorPrimaryDark">#0D47A1</color>  
 <color name="colorAccent">#FFC107</color>  
 <color name="colorBackground">#F5F5F5</color>  
 <color name="colorText">#212121</color>  
 <color name="colorButtonText">#FFFFFF</color>  
</resources>

El archivo **colors.xml** define la paleta de colores de la aplicación, estableciendo los tonos principales, de acento, fondo y texto para mantener una apariencia visual coherente y agradable; incluye el color azul como base, un acento amarillo para resaltar elementos, un fondo gris claro para contraste y texto en tonos oscuros y blancos para asegurar buena legibilidad.

<resources>  
 <string name="app\_name">MiApp</string>  
 <string name="nombre\_completo">Nombre completo</string>  
 <string name="salario\_mensual">Salario mensual</string>  
 <string name="calcular">Calcular</string>  
 <string name="nuevo">Nuevo</string>  
 <string name="salir">Salir</string>  
 <string name="resultados">Resultados</string>  
</resources>

El archivo **strings.xml** contiene los textos mostrados en la interfaz de la aplicación, como etiquetas, nombres de botones y mensajes, permitiendo una mejor organización y facilitando la traducción del contenido; en este caso define el nombre de la app y los textos “Nombre completo”, “Salario mensual”, “Calcular”, “Nuevo”, “Salir” y “Resultados”.