



Actividad [#3] - [Pantalla de Autenticación]

[Desarrollo de Aplicaciones Móviles I]

Ingeniería en Desarrollo de Software

Tutor: Eduardo Israel Castillo García

Alumno: Alan David López Rojas

Fecha: 24/11/2022

Índice

Introducción.....	pág. 3
Descripción.....	pág. 3
Justificación.....	pág. 4
Investigación.....	pág. 4
Desarrollo.....	pág. 7
Interfaz.....	pág. 7
Codificación.....	pág. 9
Prueba de la aplicación.....	pág. 13
Conclusión.....	pág. 19
Bibliografía.....	pág. 20

Introducción

La transformación digital está haciendo que surjan más tecnologías que facilitan el día a día. Un claro ejemplo es el pago de servicios o transferencias a través de la 'app' móvil. Son operaciones que no requieren mucho tiempo y son muy cómodas para el cliente. Se trata de llevar operaciones bancarias de una forma más fácil al cliente, por lo que la solicitud de la creación de una aplicación con este propósito es la razón por la que se comienza con la creación de una pantalla de presentación y un Login. Continuando con lo que se hizo en la actividad 1 y 2, en esta ocasión se pide lo siguiente para el mismo sistema de servicios bancarios:

Requerimientos de interfaz:

- o Pantalla de autenticación con dos cajas de texto:

- o La primera caja servirá para ingresar el usuario.

- o La segunda caja para ingresar la contraseña.

Requerimientos funcionales:

- o El sistema validará que el usuario haya ingresado correctamente su correo. Si esto no sucede, aparecerá una pantalla modal para indicárselo al usuario.

- o El sistema validará que la contraseña que el usuario haya ingresado sea la correcta. Si esto no sucede, aparecerá una pantalla modal para indicárselo al usuario.

- o Si se ingresa un usuario y contraseña válidos, el sistema mostrará una alerta con el texto “Entró”.

Descripción

En esta siguiente actividad se complementa la aplicación de la banca móvil con un saludo personalizado, básicamente al iniciar la aplicación esta tiene que darle un saludo al usuario dependiendo de la hora ala que ingrese, por ejemplo si se ingresa en la mañana tiene que mandar un mensaje de Buenos días, si es por la tarde, Buenas tarteres, y si es por la noche, Buenas noches, por lo que al inicio de la pantalla principal se tiene que realizar la codificación para que al pasar de la Splash Screen a la pantalla de Login está,

muestre el mensaje de saludo y posteriormente hacer el resto de las funcionalidades que se realizaron en la actividad 2.

Justificación

La calidad en el servicio al cliente no es un tema reciente dentro de las empresas, ya que desde siempre los clientes han exigido el mejor trato y la mejor atención al adquirir un producto o servicio, seguido de la confiabilidad, calidad, tiempos razonables, precios, y constante innovación por parte de las mismas, por lo cual cada una de las organizaciones se han visto a la tarea de buscar diversas alternativas para enriquecer dichas exigencias, para la aplicación ofrezca esta calidad de atención al cliente de servicios bancarios, se utilizara un saludo personalizado dependiendo de la hora del día en la cual el usuario ingrese a la aplicación.

Investigación

- ¿Cuáles son las funciones de un emulador en Android Studio?

Android Emulator simula dispositivos Android en una computadora para que se pueda probar una app en diferentes dispositivos y niveles de API de Android sin necesidad de contar con los dispositivos físicos. El emulador ofrece las siguientes ventajas:

Flexibilidad: además de poder simular una variedad de dispositivos y niveles de API de Android, el emulador incluye configuraciones predefinidas para varios teléfonos y tabletas Android, dispositivos Wear OS y Android TV.

Alta fidelidad: El emulador proporciona casi todas las funciones de un dispositivo Android real. Puede simular llamadas y mensajes de texto entrantes, especificar la ubicación del dispositivo, utilizar diferentes velocidades de red, probar sensores de rotación y otros sensores de hardware, acceder a Google Play Store y mucho más.

Velocidad: En algunos casos, probar una app en el emulador es más rápido y fácil que hacerlo en un dispositivo físico. Por ejemplo, se puede transferir datos con mayor velocidad al emulador que a un dispositivo conectado mediante USB.

En la mayoría de los casos, el emulador es la mejor opción para hacer pruebas. En esta página, se describen las funciones principales del emulador y cómo comenzar a usarlas

- ¿Qué es un Splash Screen?

El “Splash Screen” es una pantalla que carga justo cuando se abre la Aplicación y que generalmente tiene algún aviso que indica al usuario que algo está sucediendo mientras se carga el sistema o simplemente un mensaje de bienvenida , acompañadas por lo general de alguna imagen comúnmente el logotipo.

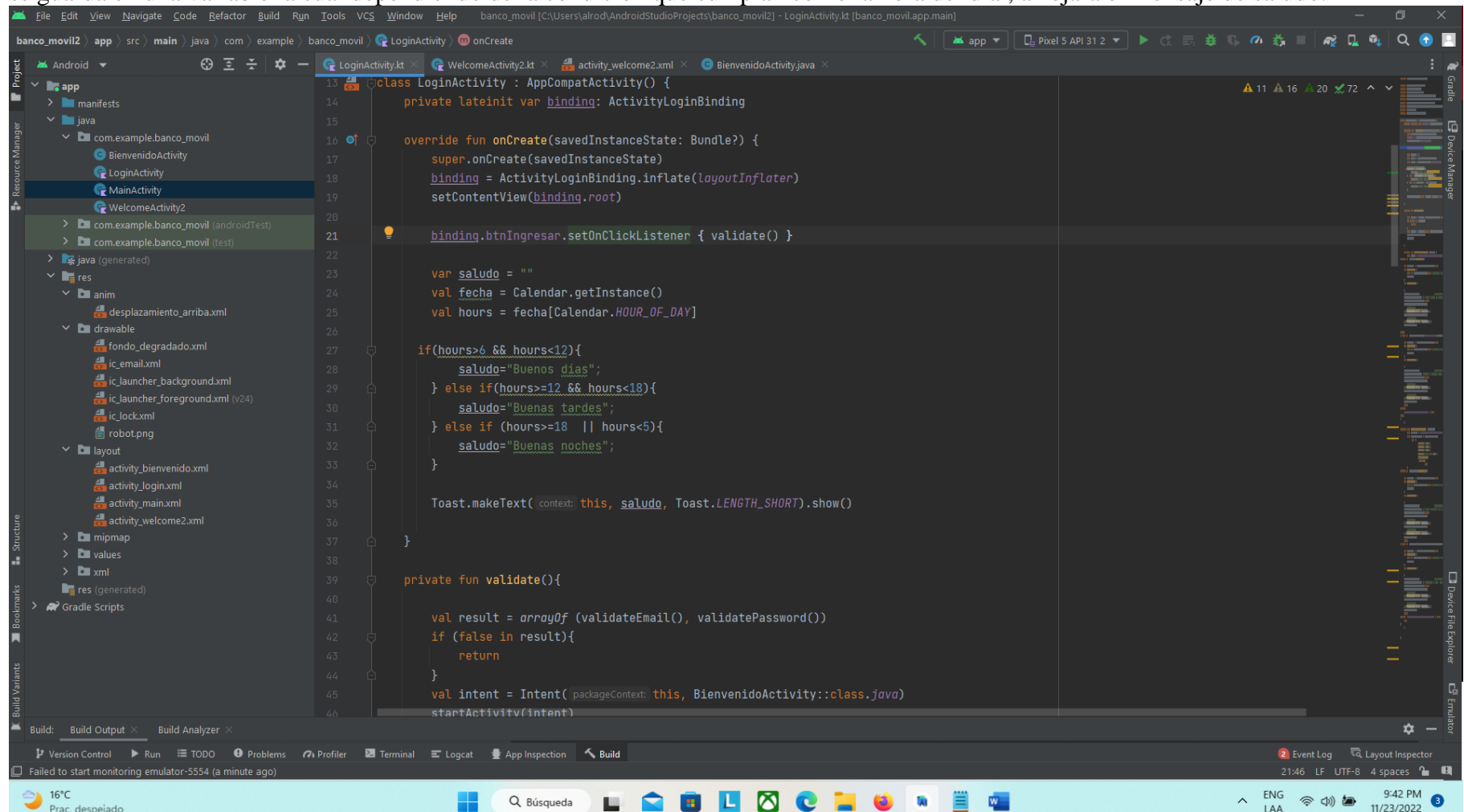
Estas pantallas de presentación son típicamente usadas por aplicaciones especialmente grandes para notificar al usuario que el programa esta en proceso de carga. Ellos proporcionan retroalimentación que es un proceso en marcha. En ocasiones, una barra de progreso en la pantalla de presentación indica el progreso. Una pantalla de Bienvenida aparece y desaparece para dar paso ala ventana principal de la aplicación. Esta Splash Screen por lo general sirven para mejorar la apariencia de una aplicación o sitio, por lo que a menudo son visualmente atractivas, por lo que pueden contener animaciones, gráficos o sonido.

En el pasado se usaba de manera incorrecta el Splash Screen ya que retrasaba intencionalmente al usuario agregando una pantalla de bienvenida que se muestra durante x segundos. Y muchas personas en la actualidad usan esta misma técnica la cual es incorrecto. Después de un tiempo Material Design proporcionó el patrón llamado Launch Screen enseñándonos así a usar el Splash Screen de manera correcta.

Desarrollo

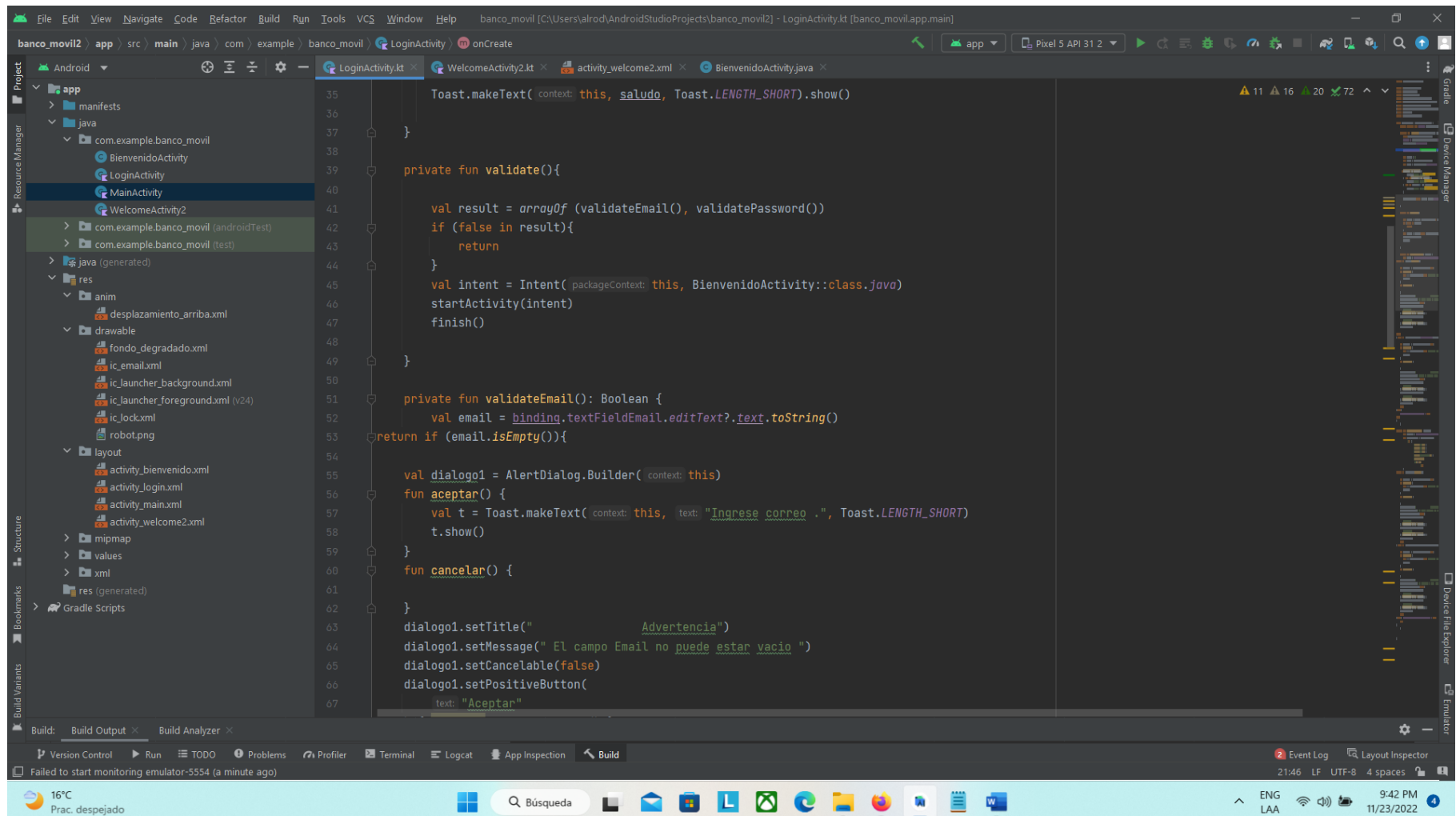
Interfaz

Se realiza el código de el mensaje personalizado, que hace el saludo dependiendo de la hora en la que ingresa el usuario, la actividad pide que dependiendo de esto, el saludo diga: Buenos días, Buenas Tardes o Buenas noches, por loque al hacer el código primero se tomo en cuenta que se tiene que obtener la hora que el sistema nos arroja a través de los comandos Calendar.getInstance() y Calendar.HOUR_OF_DAY,, esto se guarda en una variable la cual dependiendo de la condición que cumplan como la hora del día , arrojará el mensaje de saludo.



```
13 class LoginActivity : AppCompatActivity() {
14     private lateinit var binding: ActivityLoginBinding
15
16     override fun onCreate(savedInstanceState: Bundle?) {
17         super.onCreate(savedInstanceState)
18         binding = ActivityLoginBinding.inflate(layoutInflater)
19         setContentView(binding.root)
20
21         binding.btnIngresar.setOnClickListener { validate() }
22
23         var saludo = ""
24         val fecha = Calendar.getInstance()
25         val hours = fecha[Calendar.HOUR_OF_DAY]
26
27         if(hours>6 && hours<12){
28             saludo="Buenos días";
29         } else if(hours>=12 && hours<18){
30             saludo="Buenas tardes";
31         } else if (hours>=18 || hours<5){
32             saludo="Buenas noches";
33         }
34
35         Toast.makeText( context: this, saludo, Toast.LENGTH_SHORT).show()
36
37     }
38
39     private fun validate(){
40
41         val result = arrayOf (validateEmail(), validatePassword())
42         if (false in result){
43             return
44         }
45         val intent = Intent( packageContext: this, BienvenidoActivity::class.java)
46         startActivity(intent)
```

Después de eso se continua con el código de autenticación que se había hecho desde la actividad anterior.



Codificación

Login Activity

```
package com.example.banco_movil

import android.content.Intent
import android.os.Bundle
import android.widget.Toast
import androidx.appcompat.app.AlertDialog
import androidx.appcompat.app.AppCompatActivity
import androidx.core.util.PatternsCompat
import com.example.banco_movil.databinding.ActivityLoginBinding
import java.util.*
import java.util.regex.Pattern

class LoginActivity : AppCompatActivity() {
    private lateinit var binding: ActivityLoginBinding

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityLoginBinding.inflate(layoutInflater)
        setContentView(binding.root)

        binding.btnIngresar.setOnClickListener { validate() }

        var saludo = ""
        val fecha = Calendar.getInstance()
        val hours = fecha[Calendar.HOUR_OF_DAY]

        if(hours>6 && hours<12){
            saludo="Buenos días";
        } else if(hours>=12 && hours<18){
            saludo="Buenas tardes";
        } else if (hours>=18 || hours<5){
            saludo="Buenas noches";
        }

        Toast.makeText(this, saludo, Toast.LENGTH_SHORT).show()
    }

    private fun validate() {
```

```

        val result = arrayOf (validateEmail(), validatePassword())
        if (false in result){
            return
        }
        val intent = Intent(this, BienvenidoActivity::class.java)
        startActivity(intent)
        finish()
    }

    private fun validateEmail(): Boolean {
        val email = binding.textFieldEmail.editText?.text.toString()
return if (email.isEmpty()){

        val dialogo1 = AlertDialog.Builder(this)
        fun aceptar() {
            val t = Toast.makeText(this, "Ingrese correo .", Toast.LENGTH_SHORT)
            t.show()
        }
        fun cancelar() {

        }
        dialogo1.setTitle("                Advertencia")
        dialogo1.setMessage(" El campo Email no puede estar vacio ")
        dialogo1.setCancelable(false)
        dialogo1.setPositiveButton(
            "Aceptar"
        ) { dialogo1, id -> aceptar() }
        dialogo1.setNegativeButton(
            "Cancelar"
        ) { dialogo1, id -> cancelar() }
        dialogo1.show()

false
    } else if (!PatternsCompat.EMAIL_ADDRESS.matcher(email).matches()) {

        val dialogo1 = AlertDialog.Builder(this)
        fun aceptar() {
            val t = Toast.makeText(this, "Ingrese correo valido.", Toast.LENGTH_SHORT)
            t.show()
        }
        fun cancelar() {

```

```

    }
    dialogo1.setTitle("                    Advertencia")
    dialogo1.setMessage(" Correo no valido, debe tener estructura de correo electronico ")
    dialogo1.setCancelable(false)
    dialogo1.setPositiveButton(
        "Aceptar"
    ) { dialogo1, id -> aceptar() }
    dialogo1.setNegativeButton(
        "Cancelar"
    ) { dialogo1, id -> cancelar() }
    dialogo1.show()
    false
} else {
    binding.textFieldEmail.error = null
true
}
}

private fun validatePassword() : Boolean {
    val password = binding.textFieldPassword.editText?.text.toString()
    val passwordRegex = Pattern.compile(
        "^" +
        "(?=.*[0-9])" +
        "(?=.*[a-z])" +
        "(?=.*[A-Z])" +
        "(?=.*[0#$%^&!_+=])" +
        "(?=\\S+$)" +
        ".{4,}" +
        "$"
    )
}
return if (password.isEmpty()) {

    val dialogo1 = AlertDialog.Builder(this)
    fun aceptar() {
        val t = Toast.makeText(this, "Ingrese password.", Toast.LENGTH_SHORT)
        t.show()
    }
    fun cancelar() {

    }
    dialogo1.setTitle("                    Advertencia")
    dialogo1.setMessage(" El campo Password no puede estar vacio ")
    dialogo1.setCancelable(false)

```

```

        dialogo1.setPositiveButton(
            "Aceptar"
        ) { dialogo1, id -> aceptar() }
        dialogo1.setNegativeButton(
            "Cancelar"
        ) { dialogo1, id -> cancelar() }
        dialogo1.show()
        false
    } else if (!passwordRegex.matcher(password).matches()) {

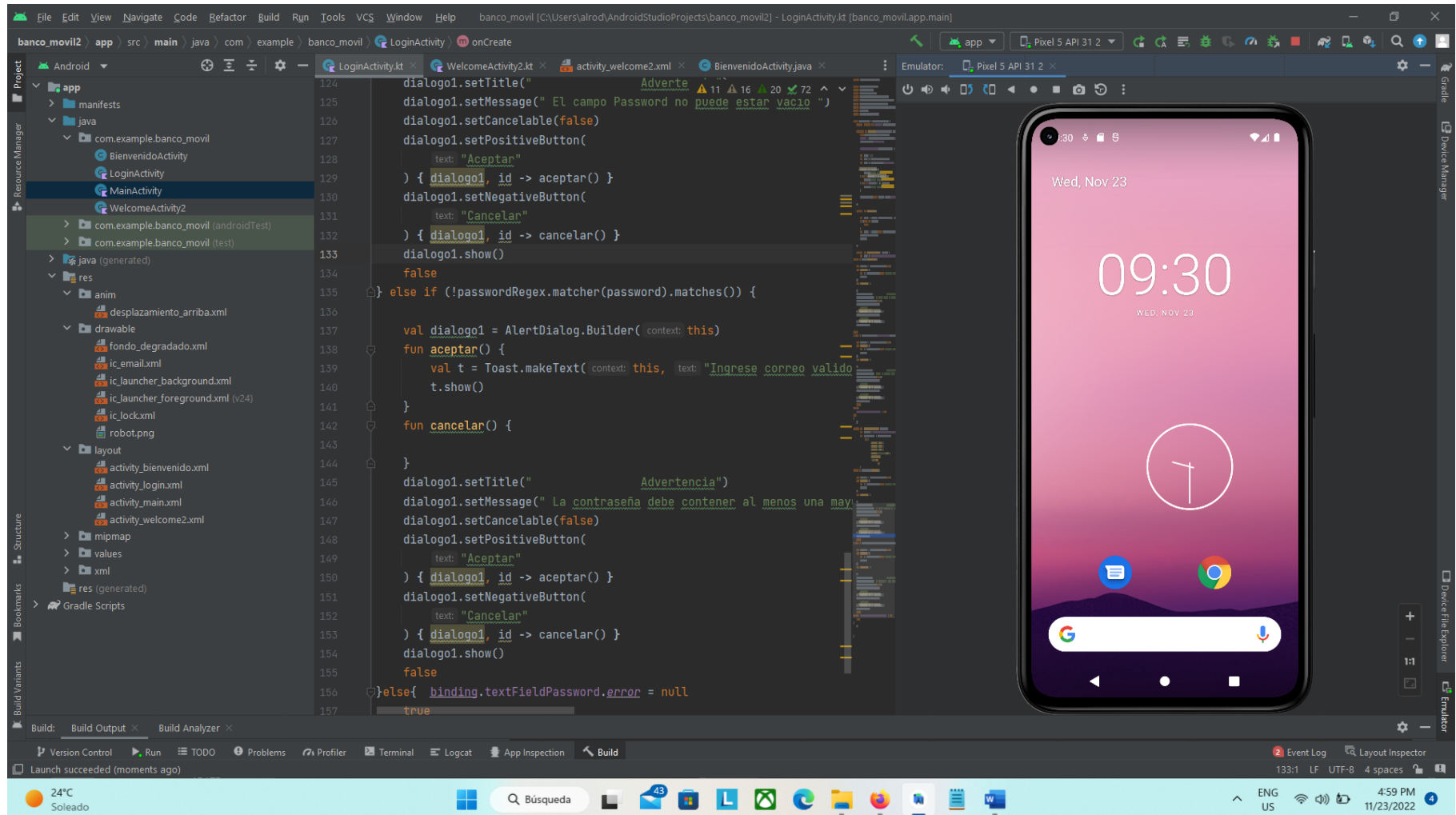
        val dialogo1 = AlertDialog.Builder(this)
        fun aceptar() {
            val t = Toast.makeText(this, "Ingrese correo valido.", Toast.LENGTH_SHORT)
            t.show()
        }
        fun cancelar() {

        }
        dialogo1.setTitle("Advertencia")
        dialogo1.setMessage("La contraseña debe contener al menos una mayuscula, una minuscula, un numero, un
caracter especial y tener al menos 4 caracteres ")
        dialogo1.setCancelable(false)
        dialogo1.setPositiveButton(
            "Aceptar"
        ) { dialogo1, id -> aceptar() }
        dialogo1.setNegativeButton(
            "Cancelar"
        ) { dialogo1, id -> cancelar() }
        dialogo1.show()
        false
    } else { binding.textFieldPassword.error = null
        true
    }
}
}

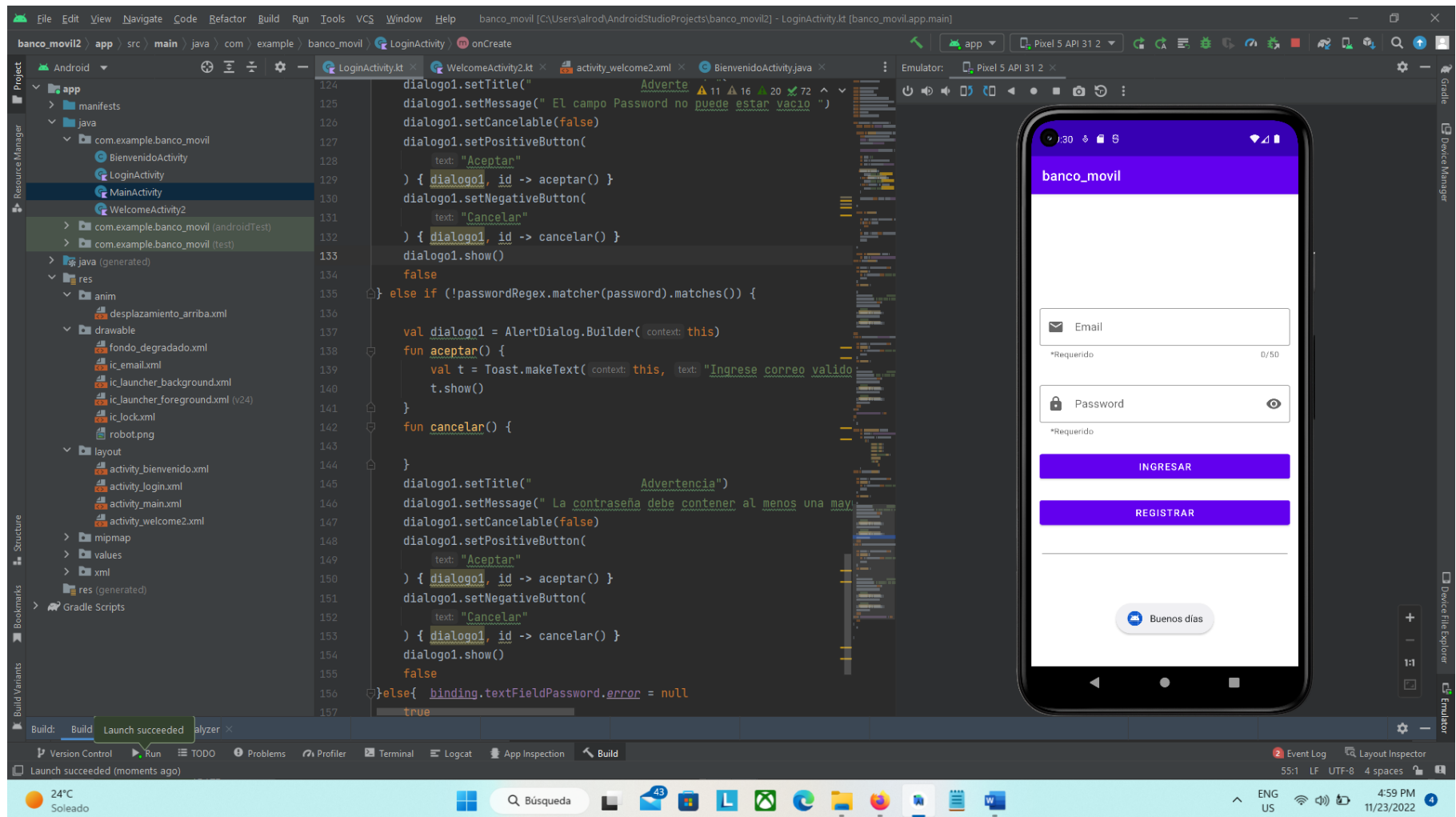
```

Prueba de aplicación

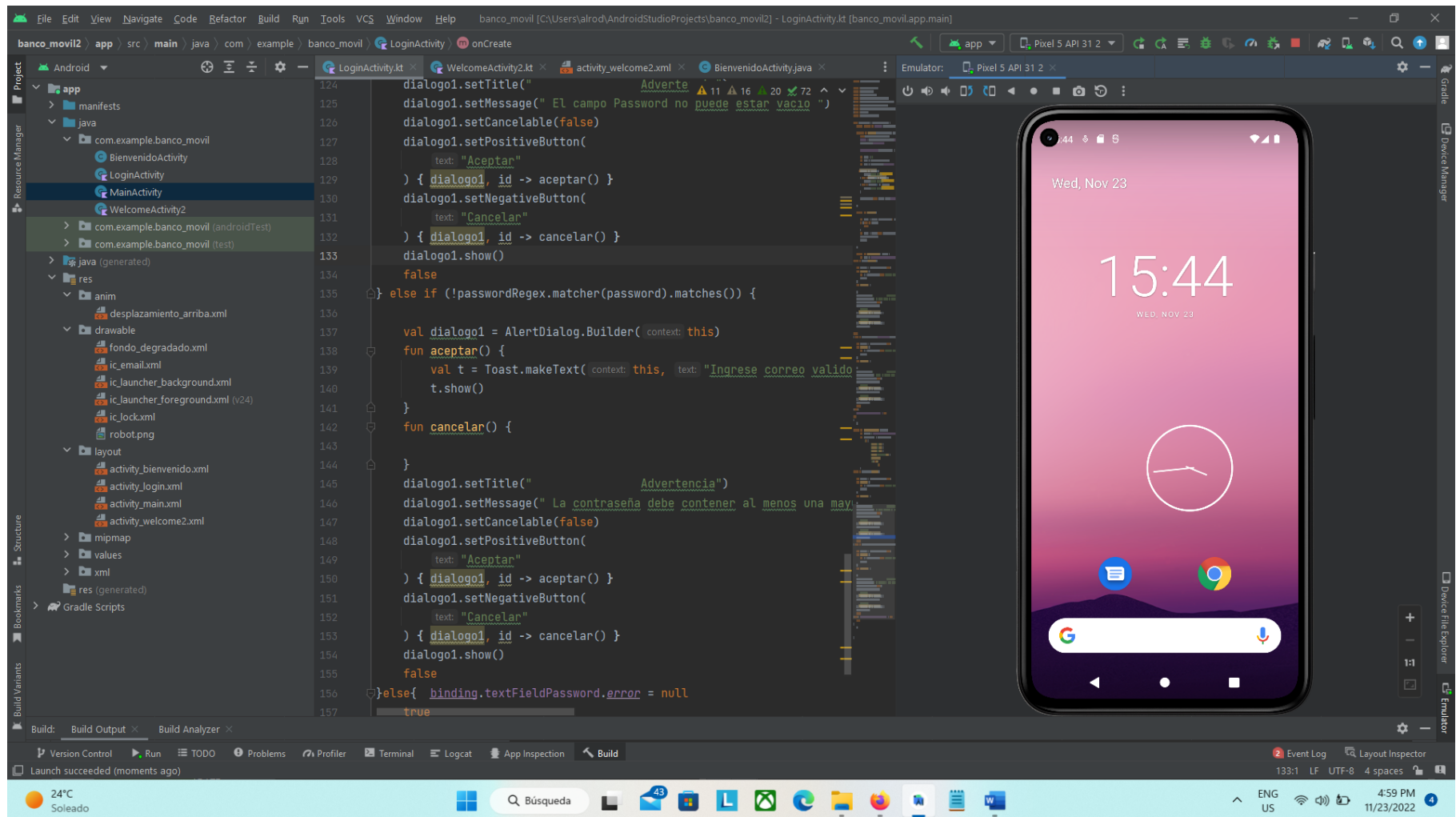
Se realiza la prueba configurando la hora a las 9:30 a.m.



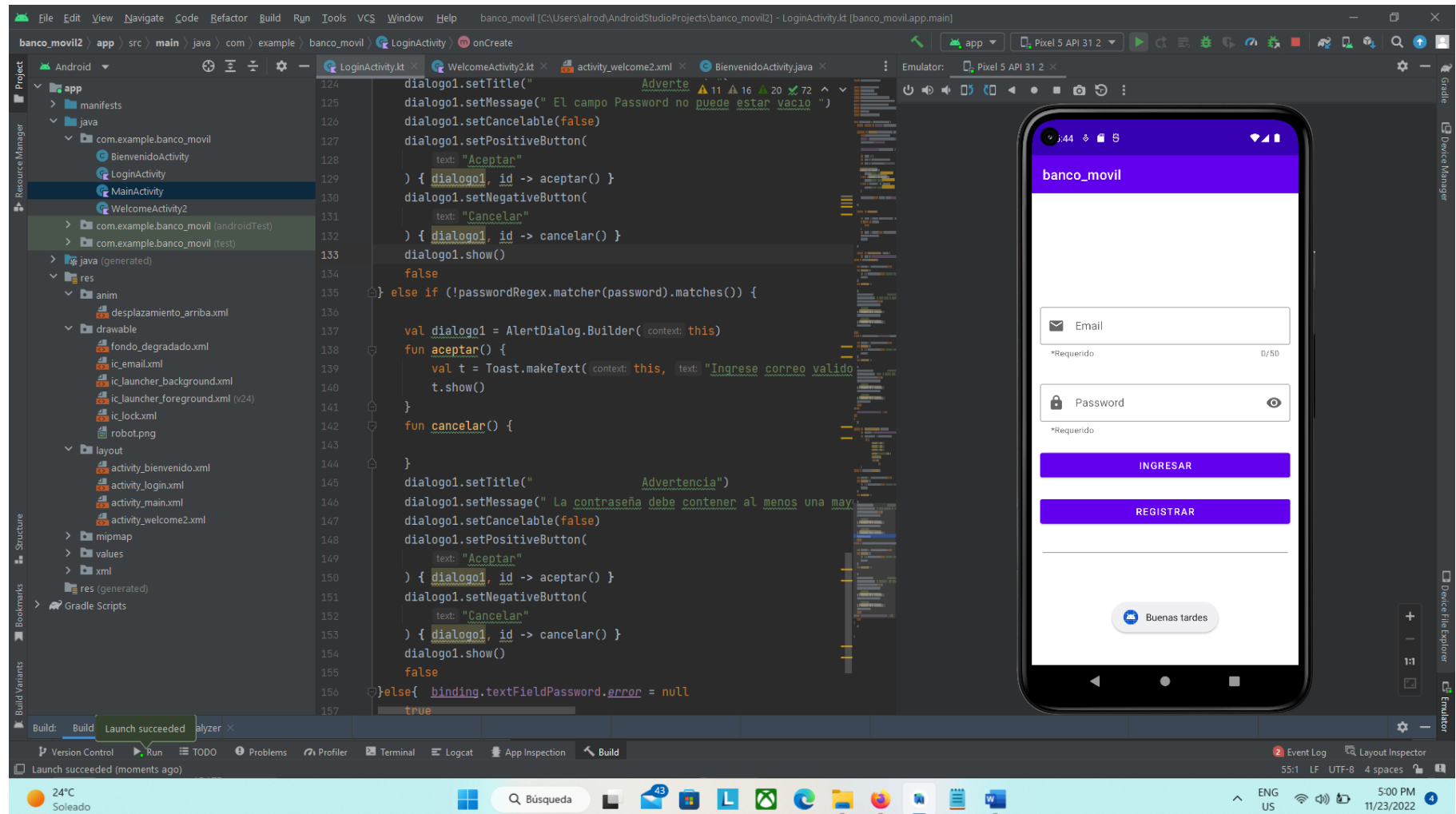
Al ejecutar la aplicación da el saludo de Buenos días.



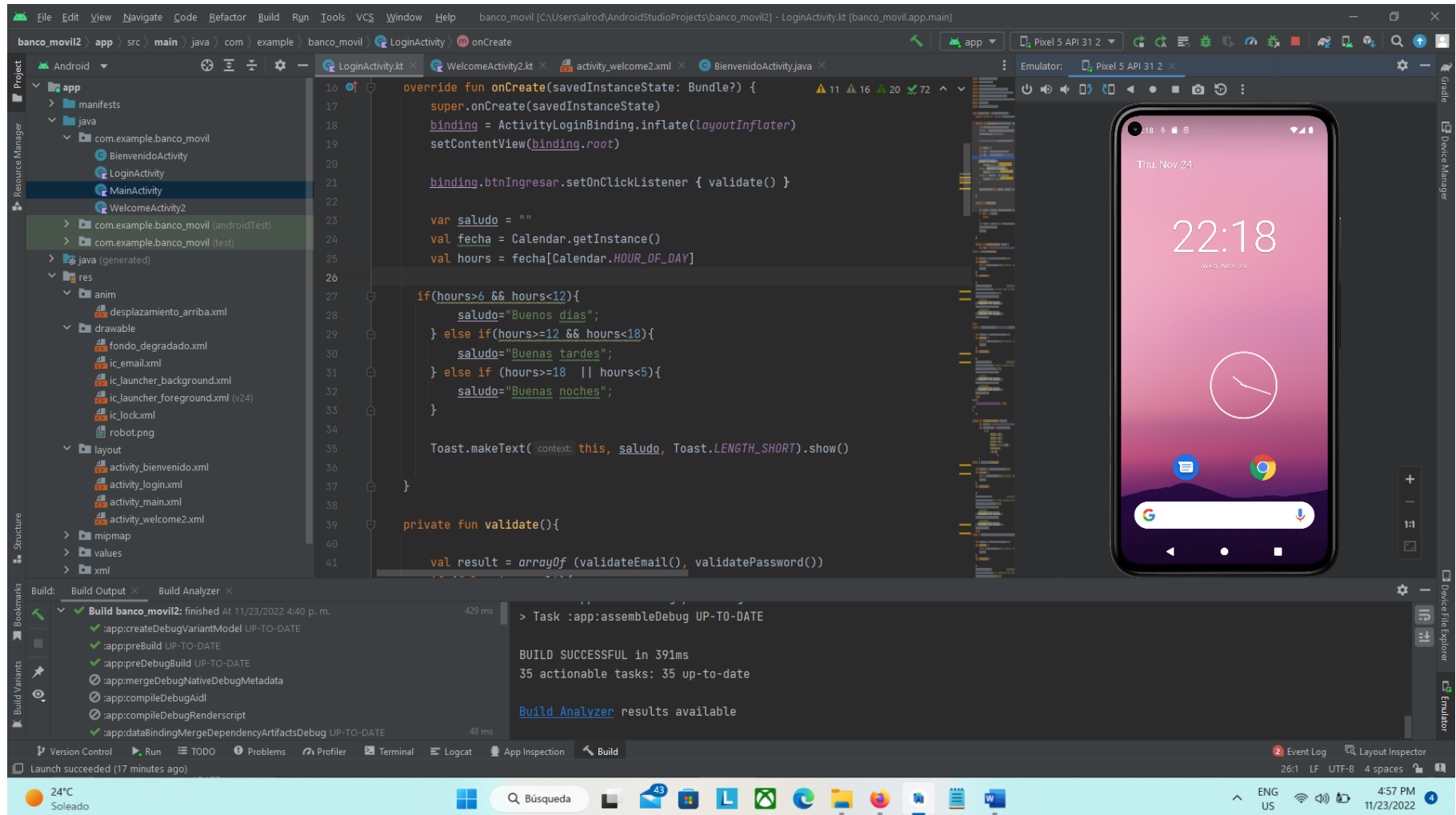
Después se configura a las 15:44 p.m. de la tarde para la siguiente condición.



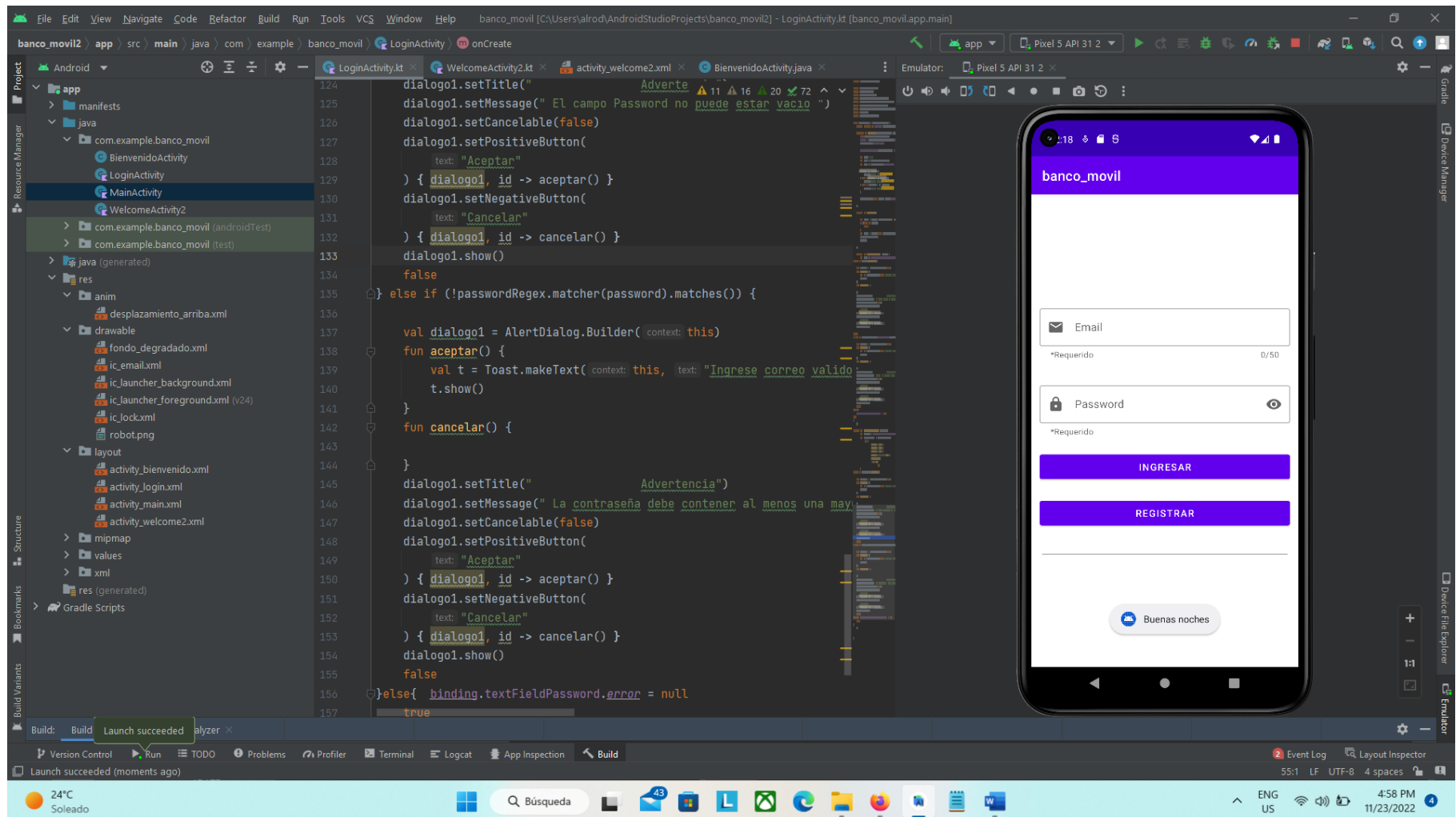
Al ejecutar la aplicación da el saludo de Buenas tardes.



Finalmente se configura la hora alas 10:18 p.m. de la noche para la ultima condición.



Al ejecutar la aplicación da el saludo de Buenas noches.



Conclusión

Cualquier persona puede hoy en día acceder a las aplicaciones e información por medio de un dispositivo móvil, presente prácticamente en cualquier lado. Además un creciente número de personas en el mundo tienen un dispositivo móvil Android o iOS a su disposición y esto los convierte en consumidores potenciales para las aplicaciones móviles. Las posibilidades de masificación del producto o servicio son enormes y esto se complementa además con nuevas y mejores funcionalidades de accesibilidad, las cuales permiten a personas con alguna discapacidad hacer uso de estos sistemas. Por lo que saber desarrollar aplicaciones móviles es fundamental para un desarrollador, para esto Android Studio facilita mucho la función de creación de estas aplicaciones, al desarrollar estas actividades me permitió conocer un poco como funciona el desarrollo de una aplicación, aunque solo fue una pequeña parte, aprendí el ciclo de vida de una aplicación, así como también manejar los Layouts y las activities para hacer el diseño y funcionalidad de la aplicación.

ENLACE DEL PROYECTO

https://drive.google.com/file/d/128s_5cqOL7EngACUyUeE9pLg7YUATI8W/view?usp=share_link

Bibliografía

Developers, Introducción a Android Studio, sitio web
<https://developer.android.com/studio/intro?hl=es-419>

Marlon Mederos Corrales, Primer acercamiento al Android Studio, Universidad tecnológica de la Habana, Monografía CUJAE 2021,
https://www.researchgate.net/publication/354474559_Primer_acercamiento_al_Android_Studio

Salvador Gómez Oliver, Manual Programación Android, Versión online 3.0, junio 2013,
<https://aluzardo.github.io/trabajo-fin-de-grado/Tutoriales/Manual%20Programacion%20Android.pdf>

Yendry Lopez Gómez, ¿Qué es un Emulador?, <https://lovtechnology.com/que-es-un-emulador/>