



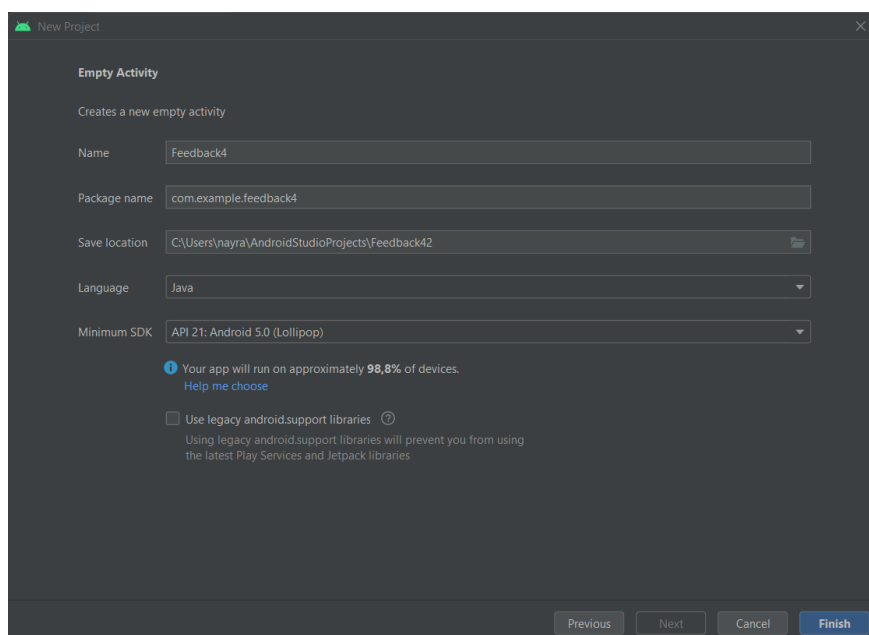
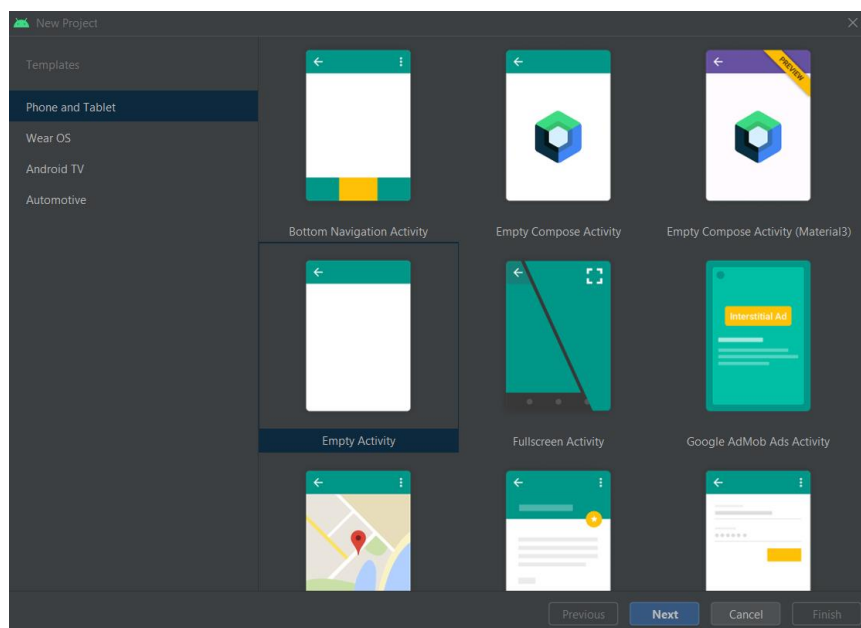
PROGRAMACIÓ MULTIMEDIA Y DISPOSITIVOS MÓVILES

Nayra Infantes Tardón

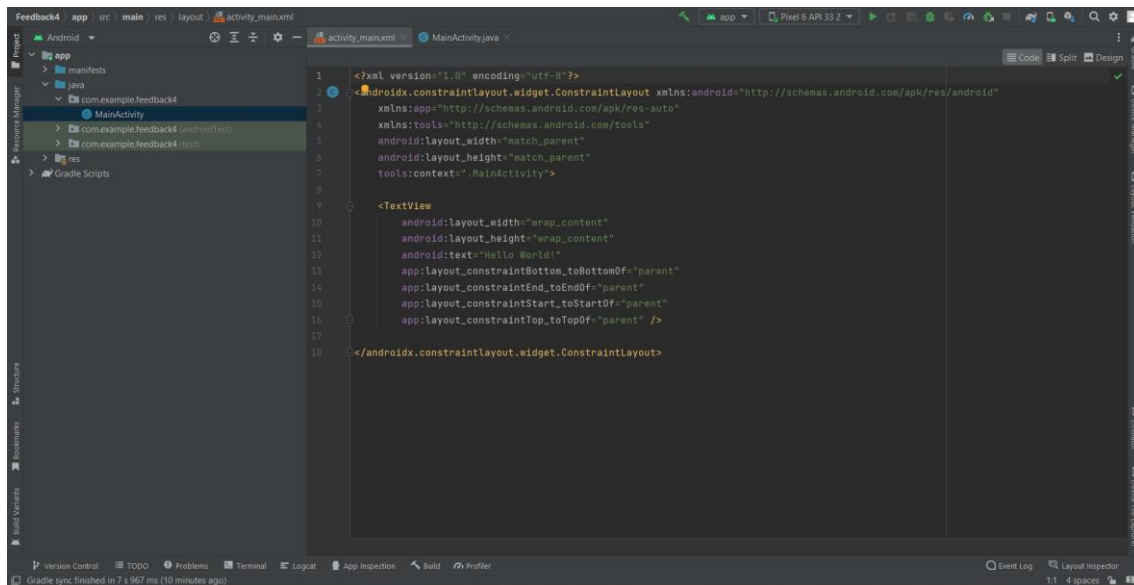
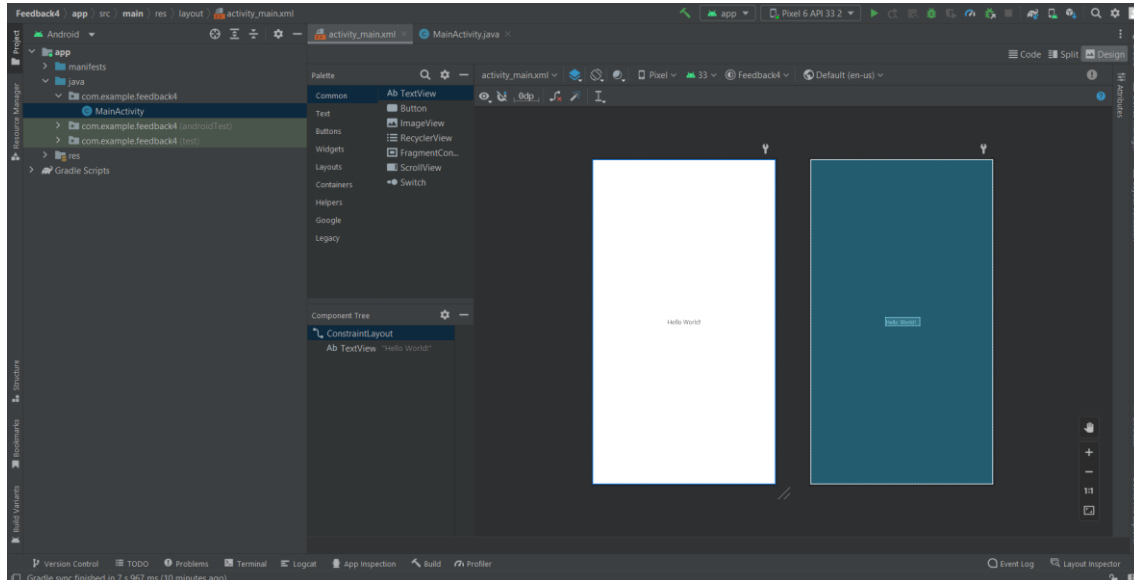
Ejercicio feedback unidad 5

Realizar una app para el sistema operativo Android con el objetivo de llevar a cabo un registro y un login de usuarios. Los datos deberán almacenarse/consultarse en una base datos utilizando para ello SQLITE y la foto que sale en las imágenes será un avatar genérico idéntico para todos los usuarios.

Para poder diseñar nuestra aplicación lo primero que debemos hacer es crear un nuevo proyecto, en nuestro caso se llamará Feedback5, partiremos de una actividad vacía para poder tener una base previa, y el lenguaje a utilizar será Java.

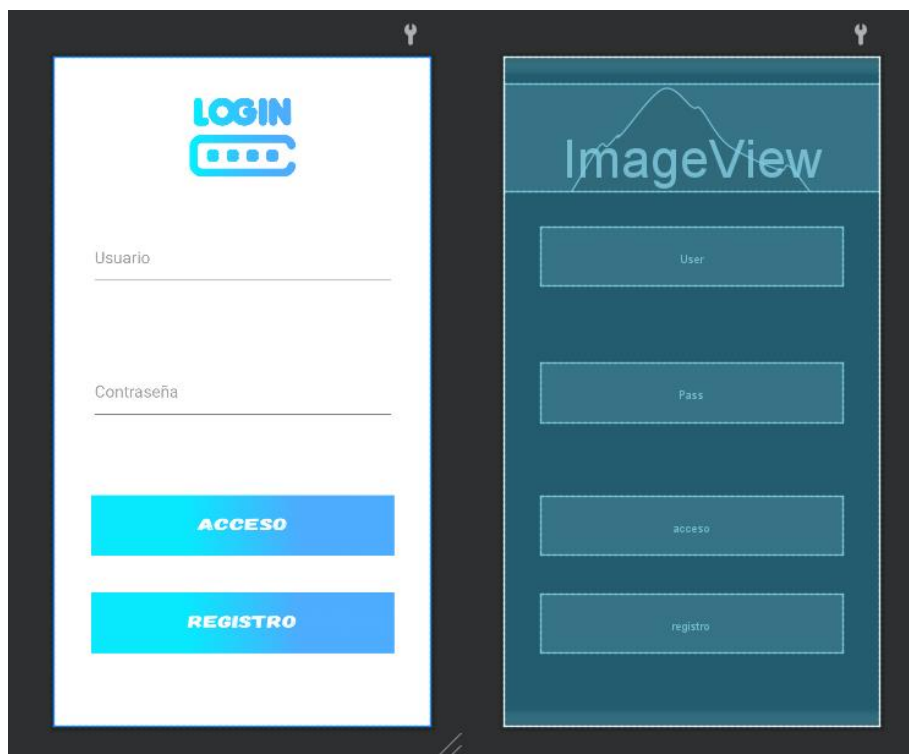


Gracias a la interfaz que nos proporciona Android Studio podemos crear nuestra agenda desde de una parte gráfica en la que funciona todo de una manera muy intuitiva o directamente a través de código en nuestros archivos .xml.



Para la realización de este ejercicio utilizaremos un `LinearLayout`, para poder posicionar de mejor manera el contenido. Nuestra aplicación estará compuesta por varias actividades, tanto la página de inicio como la página de registro están compuestas por un `ImageView` que muestra el logo de la aplicación o el icono de usuario, `ImageButtons` realizados con la herramienta Canva, y varios `EditText`; a grandes rasgos, los `EditText` en Android se consideran un `TextView` que presenta algunas modificaciones. Puede mostrarle un texto al usuario; no obstante, su función es permitirle o brindarle la opción de introducir texto. Nuestros campos de edición de texto tendrán un `id` que dependerá del tipo de dato a introducir o mostrar; un `layout width`, o ancho del grupo; un `layout height`, que define el alto; un `hint` o pista, que es el mensaje que se muestra dentro de ellos para que el usuario sepa que dato debe introducir; y un `inputType`, que determinará el tipo de entrada que se va a utilizar, entre otras propiedades.

Los botones, por otro lado, dispondrán también de un `id`, ancho y alto, se les asignará un método de nuestros archivos `.java` a través de la propiedad `onClick`, y les proporcionaremos una imagen gracias a la propiedad `background` para que la aplicación sea más vistosa.



```
<?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"
    tools:context=".MainActivity">

    <ImageView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:layout_marginTop="30dp"
        android:src="@drawable/login" />

    <EditText
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:hint="Usuario"
        android:inputType="textPersonName"
        android:layout_margin="40dp"
        android:id="@+id/User" />

    <EditText
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_weight="1"
        android:hint="Contraseña"
        android:inputType="textPassword"
        android:layout_margin="40dp"
        android:id="@+id/Pass" />

    <ImageButton
        android:id="@+id/acceso"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_margin="40dp"
        android:layout_weight="1"
        android:background="@drawable/acceso"
        android:onClick="acceso" />

    <ImageButton
        android:id="@+id/registro"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_marginHorizontal="40dp"
        android:layout_marginBottom="80dp"
        android:layout_weight="1"
        android:background="@drawable/registro"
        android:onClick="pagRegistro" />
</LinearLayout>
```

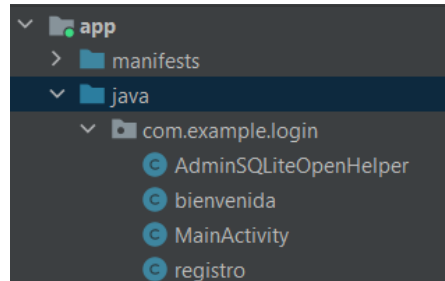
En el archivo strings.xml, ubicado en el directorio “values” que a su vez se encuentra dentro de “res”, crearemos todas las constantes a incluir en nuestro programa. Para ello debemos disponer @string/ y seguidamente el nombre de la constante definida en el archivo. Por defecto siempre podremos encontrar el string app_name que contiene el nombre de nuestra aplicación. También modificaremos en el archivo colors.xml los colores de nuestra aplicación para que sea mucho más estética.

```
<resources>
  <string name="app_name">LOGIN APP</string>
  <string name="login">LOGIN</string>
  <string name="usuario">Usuario</string>
  <string name="password">Contraseña</string>
  <string name="password2">Repetir contraseña</string>
  <string name="boton3">CONFIRMAR</string>
  <string name="form">Formulario de registro</string>
  <string name="mensaje1">BIENVENIDO</string>
  <string name="mensaje2">¡Ten un buen día!</string>
</resources>
```

```
<resources>
  <color name="purple_200">#FFB86F</color>
  <color name="purple_500">#656565</color>
  <color name="purple_700">#656565</color>
  <color name="teal_200">#FF03DAC5</color>
  <color name="teal_700">#FF018786</color>
  <color name="black">#FF000000</color>
  <color name="white">#FFFFFFFF</color>
</resources>
```

Una vez completados los pasos anteriores comenzaremos a trabajar en nuestros archivos .java.

Tendremos en total cuatro archivos que ayudarán a hacer funcionar nuestra aplicación: AdminSQLiteOpenHelper.java, MainActivity.java, Registro.java y Bienvenida.java.



La clase AdminSQLiteOpenHelper hereda de SQLiteOpenHelper. Esta clase nos permite crear la base de datos y actualizar la estructura de tablas y datos iniciales. Debemos implementar el constructor y sobrescribir los métodos onCreate y onUpgrade.

El método onCreate se le llama cuando la base de datos se crea por primera vez. En él, definimos la estructura de las tablas y cargamos los datos iniciales. En el método onUpgrade se llama cuando la base de datos debe ser actualizada. En nuestro caso, tiene por objetivo añadir tablas ya que no hemos añadido más funcionalidades de las que pedía el enunciado.

```
package com.example.login;
import ...

public class AdminSQLiteOpenHelper extends SQLiteOpenHelper{
    public AdminSQLiteOpenHelper(Context context, String name, SQLiteDatabase.CursorFactory factory, int version) {
        super(context, name, factory, version);
    }

    @Override
    public void onCreate(SQLiteDatabase BaseDeDatos) {
        BaseDeDatos.execSQL("create table usuarios(id integer primary key autoincrement, usuario text, pass text)");
    }

    @Override
    public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int i1) {

    }
}
```

En la clase Registro, crearemos un objeto de la clase AdminSQLiteOpenHelper y le pasaremos al constructor this, que hace referencia al Activity actual, "usuarios", que es el nombre de la base de datos que crearemos en el caso que no exista, null y un uno indicando que es la primera versión de la base de datos.

Posteriormente crearemos un objeto de la clase SQLiteDatabase llamando al método getWritableDatabase para que la base de datos se abra en modo lectura y escritura.

Crearemos también un objeto de la clase ContentValues y mediante el método put inicializaremos los campos a cargar, en este caso, usuario y pass que es nuestra contraseña.

En el método insert de la clase SQLiteDatabase pasaremos en el primer parámetro el nombre de la tabla, como segundo parámetro un null y por último el objeto de la clase ContentValues, llamado registro, ya inicializado.

Cuando nuestros datos se hayan cargado en la base de datos, vaciaremos los EditText y mediante un Toast indicaremos al usuario que su registro se ha completado con éxito. También hemos añadido un Intent que nos permite volver a la vista principal cuando el registro se haya completado y mediante una estructura de else e if mostraremos Toast que nos avisarán en caso de que haya un campo vacío o las contraseñas no coincidan.

```
package com.example.login;

import ...

public class registro extends AppCompatActivity {
    EditText us, pas, pas2;
    ImageButton btnGuardar;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_registro);
        us = (EditText) findViewById(R.id.us);
        pas = (EditText) findViewById(R.id.pas);
        pas2 = (EditText) findViewById(R.id.pas2);
        btnGuardar = (ImageButton) findViewById(R.id.guardar);
    }

    //Método para dar de alta usuarios
    public void Registrar(View view) {
        AdminSQLiteOpenHelper admin = new AdminSQLiteOpenHelper( context: this, name: "usuarios", factory: null, version: 1);
        SQLiteDatabase Database = admin.getWritableDatabase();

        String usuario = us.getText().toString();
        String pass = pas.getText().toString();
        String pass2 = pas2.getText().toString();

        if (!usuario.isEmpty() && !pass.isEmpty() && !pass2.isEmpty()) {
            if (pass.equals(pass2)) {
                ContentValues registro = new ContentValues();

                registro.put("usuario", usuario);
                registro.put("pass", pass);

                Database.insert( table: "usuarios", nullColumnHack: null, registro);

                Database.close();
                us.setText("");
                pas.setText("");
                pas2.setText("");

                Toast.makeText( context: this, text: "Registro exitoso", Toast.LENGTH_SHORT).show();
                Intent i= new Intent( packageContext: registro.this,MainActivity.class);
                startActivity(i);
            } else {
                Toast.makeText( context: this, text: "Las contraseñas no coinciden", Toast.LENGTH_SHORT).show();
            }
        } else {
            Toast.makeText( context: this, text: "Debes llenar todos los campos", Toast.LENGTH_SHORT).show();
        }
    }
}
```


En la clase MainActivity hemos creado dos métodos, uno llamado pagRegistro que contiene un Intent que nos lleva hasta la página de registro donde se ejecutarán las acciones del punto anterior y el método acceso que es algo más complejo ya que se encarga de realizar la consulta a la base de datos para que podamos comprobar que el usuario y contraseña existen.

En el método acceso crearemos otro objeto de la clase AdminSQLiteOpenHelper y de nuevo obtendremos una referencia de la base de datos llamando al método getWritableDatabase.

En caso de que los campos usuario y contraseña no estén vacíos, crearemos una variable de la clase Cursor y la inicializaremos con el valor devuelto por el método llamado.rawQuery, donde le indicamos que seleccione todos los datos de la tabla usuarios donde usuario coincida con el dato introducido en el EditText.

Mediante putExtra guardaremos el nombre de usuario para poder mostrarlo en la página de bienvenida y gracias a los Toast podremos avisar a los usuarios si falta algún dato o todavía no se han registrado.

```
package com.example.login;

import ...

public class MainActivity extends AppCompatActivity {
    EditText user, pass;
    ImageButton btnAcceso, btnRegistro;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        user=(EditText) findViewById(R.id.User);
        pass=(EditText) findViewById(R.id.Pass);
        btnAcceso=(ImageButton) findViewById(R.id.acceso);
        btnRegistro=(ImageButton) findViewById(R.id.registro);
    }

    public void pagRegistro(View view){
        Intent i= new Intent( packageContext: MainActivity.this,registro.class);
        startActivity(i);
    }
    //Metodo de acceso a la app
    public void acceso(View view){
        AdminSQLiteOpenHelper admin = new AdminSQLiteOpenHelper( context: this, name: "usuarios", factory: null, version: 1);
        SQLiteDatabase Database = admin.getWritableDatabase();

        String usuario = user.getText().toString();
        String password = pass.getText().toString();
        String [] usuarios= {user.getText().toString()};
        if(!usuario.isEmpty()&&!password.isEmpty()){
            Cursor consulta = Database.rawQuery( sql: "select * from usuarios where usuario=?", usuarios);

            if(consulta.moveToFirst()){
                if(consulta.getString( 0 ).equals(usuario)&&consulta.getString( 1 ).equals(password)){
                    Intent i2= new Intent( packageContext: MainActivity.this,bienvenida.class);
                    i2.putExtra( name: "username",usuario);
                    startActivity(i2);
                    Database.close();
                }
                else{
                    Toast.makeText( context: this, text: "No existe el usuario", Toast.LENGTH_SHORT).show();
                    Database.close();
                }
            }
        } else {
            Toast.makeText( context: this, text: "Debes introducir el usuario o contraseña", Toast.LENGTH_SHORT).show();
        }
    }
}
```

```




package com.example.login;

import ...

public class bienvenida extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_bienvenida);
        Bundle miBundle = getIntent().getExtras();
        TextView muestraUserName = findViewById(R.id.muestrausername);
        String userNameIntent;
        userNameIntent = miBundle.getString( key: "username");
        muestraUserName.setText(userNameIntent);
    }
}
  
```

Nuestra aplicación se debe mostrar pues de la siguiente forma:

 <p>Usuario</p> <hr/> <p>Contraseña</p> <hr/> <p>Repetir contraseña</p> <hr/> <p>GUARDAR USUARIO</p>	 <p>Usuario</p> <hr/> <p>Contraseña</p> <hr/> <p>ACCESO</p> <p>REGISTRO</p>	 <p>BIENVENIDO</p> <p>¡Ten un buen día!</p>
--	--	---