[NT5] NOTA TÉCNICA Nº. 5
DESARROLLO MÓVIL

Partes de Android Studio :

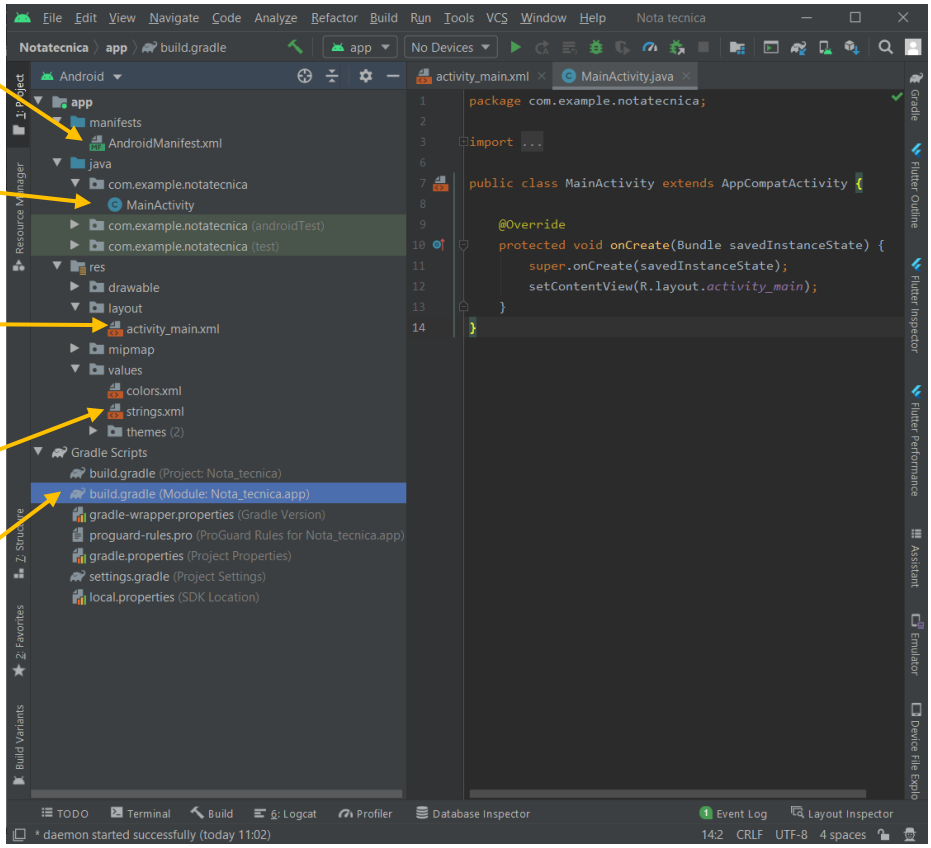
Contiene la estructura de la app y los permisos necesarios su funcionamiento

Primera activity que aparece, se define la funcionalidad

Construcción del diseño que se muestra al usuario

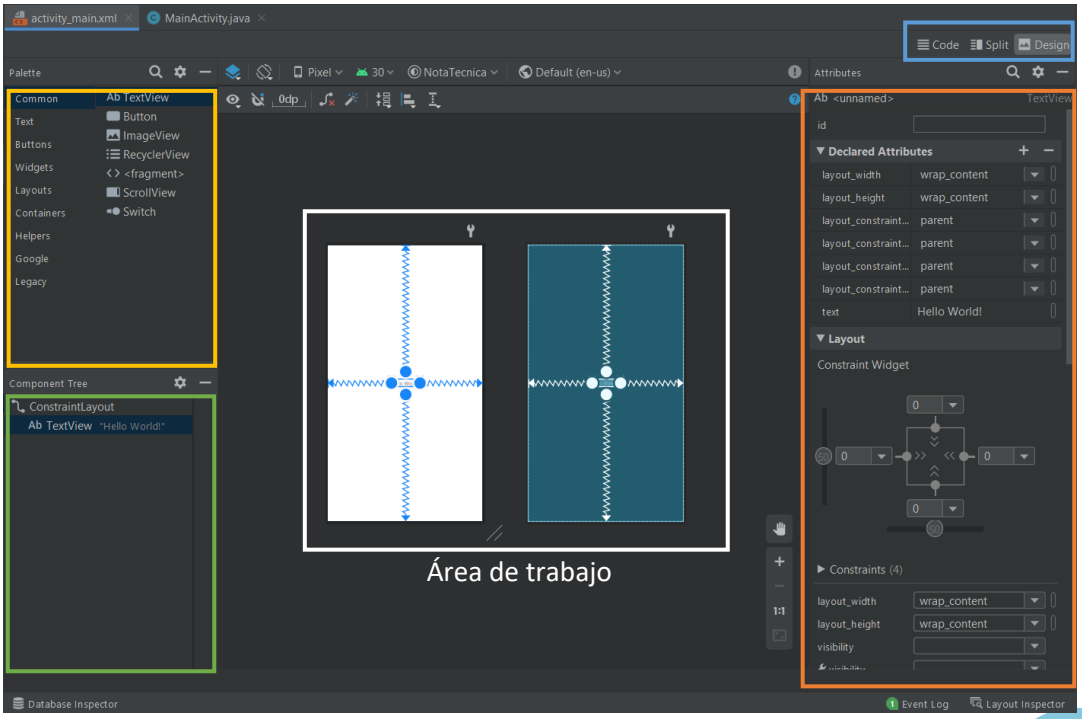
Contiene los strings que se usan con frecuencia, ayuda a reducir espacio

Incluye las dependencias de la app



Área de selección de componentes

Lista de componentes utilizados



Selección de modo de trabajo

Área de configuraciones

[NT5] NOTA TÉCNICA Nº. 5
DESARROLLO MÓVIL

El diseño también puede ser realizado utilizando únicamente el código correspondiente:

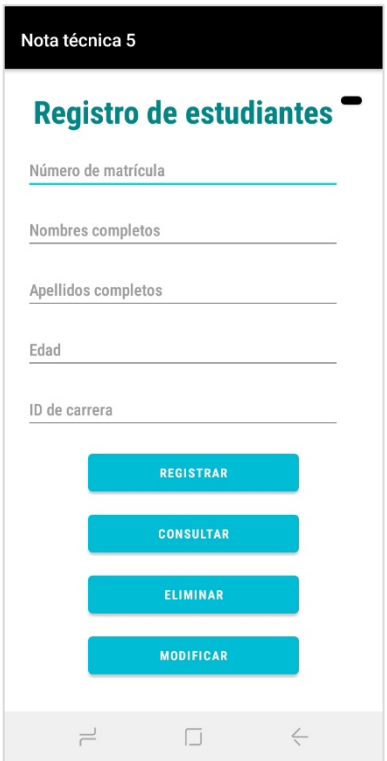
activity_main.xml | MainActivity.java

Code | Split | Design

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

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Escenario:

En la empresa Adita S.A. se requiere la creación de una aplicación móvil para el registro de estudiantes, la cual se desarrolla con Android Studio y SQLite. A continuación, se muestra paso a paso el desarrollo de la aplicación móvil con cada una de sus funcionalidades correspondientes.



[NT5] NOTA TÉCNICA NO. 5
DESARROLLO MÓVIL

La base de datos del “Sistema Académico” se muestra a continuación:

Sistema Académico - Tabla "estudiante"

id_estudiante	nombre_est	apellidos_est	edad	id_carrera
201707698	Diana	Once	21	INGMECATRONICA
201707836	Pablo	Guitierrez	21	TURISMO
201708935	Joselin	Cortez	19	INGTELEMATICA
201707767	Luis	Moyón	20	INGMECATRONICA
201789512	Daniela	García	21	INGTELECO
201856487	Clara	Zambrano	19	LICBIOLOGIA
201820158	Carlos	Flores	20	INGCOMPUTACION
202008898	Jaime	Rodriguez	19	LICECONOMIA

[NT5] NOTA TÉCNICA NO. 5
DESARROLLO MÓVIL

Paso 1: Creación del diseño de la app.

A continuación, se muestra el código de los componentes utilizados y sus respectivas configuraciones, se tomó en cuenta cada uno de los parámetros requeridos en la tabla presentada anteriormente.

```
<TextView
    android:id="@+id/textView_registro"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="20dp"
    android:layout_marginLeft="20dp"
    android:layout_marginTop="30dp"
    android:layout_marginEnd="20dp"
    android:layout_marginRight="20dp"
    android:text="Registro de estudiantes"
    android:textSize="30dp"
    android:textColor="@color/teal_700"
    android:textStyle="bold"
    android:fontFamily="sans-serif-condensed-medium"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

```
<EditText
    android:id="@+id/editTextNombres"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="20dp"
    android:layout_marginLeft="20dp"
    android:layout_marginTop="20dp"
    android:layout_marginEnd="20dp"
    android:layout_marginRight="20dp"
    android:ems="10"
    android:fontFamily="sans-serif-condensed-medium"
    android:hint="Nombres completos"
    android:inputType="textPersonName"
    android:textStyle="bold"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/editTextNumMat" />
```

También se puede establecer el tipo de texto a ingresar en el campo EditText

```
<EditText
    android:id="@+id/editTextEdad"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="20dp"
    android:layout_marginLeft="20dp"
    android:layout_marginTop="20dp"
    android:layout_marginEnd="20dp"
    android:layout_marginRight="20dp"
    android:ems="10"
    android:fontFamily="sans-serif-condensed-medium"
    android:hint="Edad"
    android:inputType="number"
    android:textStyle="bold"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/editTextApellidos" />
```

```
<Button
    android:id="@+id/button_registrar"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="80dp"
    android:layout_marginLeft="80dp"
    android:layout_marginTop="15dp"
    android:layout_marginEnd="80dp"
    android:layout_marginRight="80dp"
    android:fontFamily="sans-serif-condensed-medium"
    android:onClick="registro"
    android:text="Registrar"
    android:textStyle="bold"
    app:backgroundTint="@color/teal_700"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/editTextIdCarrera" />
```

[NT5] NOTA TÉCNICA Nº. 5
DESARROLLO MÓVIL

Paso 2. Desarrollo del código java [Base de datos interna]

1. Es necesario crear un archivo .java para establecer la base de datos de SQLite.

```
import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

import androidx.annotation.Nullable;

public class AdminSQLiteOpenHelper extends SQLiteOpenHelper {

    public AdminSQLiteOpenHelper(@Nullable Context context, @Nullable String name,
                                 @Nullable SQLiteDatabase.CursorFactory factory, int version) {
        super(context, name, factory, version);
    }

    @Override
    public void onCreate(SQLiteDatabase sqLiteDatabase) {
        sqLiteDatabase.execSQL("create table estudiantes(id_estudiante int primary key," +
                "nombres text, apellidos text, edad int, id_carrera text)");
    }

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

2. Se deben obtener los EditText creados en el .xml para extraer la información que ingrese el usuario:

```
EditText matricula, nombres, apellidos, edad, idCarrera;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    matricula = (EditText)findViewById(R.id.editTextNumMat);
    nombres = (EditText)findViewById(R.id.editTextNombres);
    apellidos = (EditText)findViewById(R.id.editTextApellidos);
    edad = (EditText)findViewById(R.id.editTextEdad);
    idCarrera = (EditText)findViewById(R.id.editTextIdCarrera);
}
```

3. Se establece el método para guardar la información:

```
public void registro(View view) {
    AdminSQLiteOpenHelper admin = new AdminSQLiteOpenHelper( context: this,
            name: "administracion", factory: null, version: 1);
    SQLiteDatabase bd = admin.getWritableDatabase();

    String matriculaText = matricula.getText().toString();
    String nombresText = nombres.getText().toString();
    String apellidosText = apellidos.getText().toString();
    String edadText = edad.getText().toString();
    String idCarreraText = idCarrera.getText().toString();

    if(!matriculaText.isEmpty()||!nombresText.isEmpty()||!apellidosText.isEmpty()||
            !edadText.isEmpty()||!idCarreraText.isEmpty()){
        bd.execSQL("insert into estudiantes (id_estudiante,nombres,apellidos,edad,id_carrera) "
                +"values ("+matriculaText+",'"+nombresText+"','"+apellidosText+"',"
                +edadText+",'"+idCarreraText+"')");
        bd.close();
        matricula.setText("");
        nombres.setText("");
        apellidos.setText("");
        edad.setText("");
        idCarrera.setText("");
        Toast.makeText( context: this, text: "Se cargaron los datos del estudiante",
                Toast.LENGTH_SHORT).show();
    }else{
        Toast.makeText( context: this, text: "Por favor, llene todos los campos.",
                Toast.LENGTH_SHORT).show();
    }
}
```

Como se puede observar, en primer lugar creamos una instancia de la clase .java de la base de datos creada en el primer paso. Obtenemos la base de datos ya creada, obtenemos el contenido de cada uno de los EditText y los ingresamos en la base de datos haciendo uso del método execSQL(), fijamos como vacíos los EditText con el fin de prepararlos para un nuevo ingreso de datos y finalmente cerramos la base de datos.

4. En el archivo .xml, agregar la siguiente línea al botón registro para activar el método creado:

```
android:onClick="registro"
```

5. Para consultar al estudiante por su número de matrícula:

```
public void consultaMatricula(View view) {
    AdminSQLiteOpenHelper admin = new AdminSQLiteOpenHelper( context: this,
            name: "administracion", factory: null, version: 1);
    SQLiteDatabase bd = admin.getReadableDatabase();

    String matriculaText = matricula.getText().toString();

    if(!matriculaText.isEmpty()){
        Cursor fila = bd.rawQuery(
                sql: "select nombres,apellidos,edad,id_carrera from estudiantes where " +
                        "id_estudiante=" + matriculaText, selectionArgs: null);
        if (fila.moveToFirst()) {
            nombres.setText(fila.getString( columnIndex: 0));
            apellidos.setText(fila.getString( columnIndex: 1));
            edad.setText(fila.getString( columnIndex: 2));
            idCarrera.setText(fila.getString( columnIndex: 3));
            Toast.makeText( context: this, text: "Consulta exitosa.",
                    Toast.LENGTH_SHORT).show();
        } else{
            Toast.makeText( context: this, text: "No existe un estudiante con dicha matrícula",
                    Toast.LENGTH_SHORT).show();
        }
        bd.close();
    }else{
        Toast.makeText( context: this, text: "Ingrese un número de matrícula.",
                Toast.LENGTH_SHORT).show();
    }
}
```

En este caso se obtiene la base de datos de manera Readable, es decir, que pueda ser leída. Utiliza el método rawQuery() para poder realizar la consulta a la base de datos. Se debe notar claramente que el tipo de sentencia utilizada para realizar la consulta es similar a la utilizada en MySQL
utilizando las palabras clave *select*, *from* y *where*. Se recorre el Cursor para obtener los datos.

[NT5] NOTA TÉCNICA Nº. 5
DESARROLLO MÓVIL

6. Para modificar los datos de un estudiante:

```
public void modificarInformacion(View v) {
    AdminSQLiteOpenHelper admin = new AdminSQLiteOpenHelper( context: this,
            name: "administracion", factory: null, version: 1);
    SQLiteDatabase bd = admin.getWritableDatabase();

    String matriculaText = matricula.getText().toString();
    String nombresText = nombres.getText().toString();
    String apellidosText = apellidos.getText().toString();
    String edadText = edad.getText().toString();
    String idCarreraText = idCarrera.getText().toString();

    if(!matriculaText.isEmpty()){
        if(nombresText.isEmpty()||apellidosText.isEmpty()||edadText.isEmpty()
                ||idCarreraText.isEmpty()){
            Toast.makeText( context: this, text: "Ingrese todos los datos.",
                    Toast.LENGTH_SHORT).show();
        }
        //try {
        bd.execSQL("update estudiantes set id_estudiante="+matriculaText+",nombres='"
                +nombresText+"',apellidos='"+apellidosText+"',edad="+edadText+",id_carrera='"
                +idCarreraText+"' where id_estudiante="+matriculaText);
        matricula.setText("");
        nombres.setText("");
        apellidos.setText("");
        edad.setText("");
        idCarrera.setText("");
        bd.close();
        Toast.makeText( context: this, text: "Se modificaron los datos.", Toast.LENGTH_SHORT).show();
    }else{
        Toast.makeText( context: this, text: "Ingrese una matricula.",
                Toast.LENGTH_SHORT).show();
    }
}
```

Para modificar los datos de un estudiante se debe obtener la base de datos de manera Writable, es decir, en la cual se pueda escribir los datos. En este caso de utiliza el método execSQL(), tal como en el paso 3, donde se agregaban los estudiantes a la base de datos.

7. Para eliminar a un estudiante de la base de datos:

```
public void eliminarEstudiante(View v) {
    AdminSQLiteOpenHelper admin = new AdminSQLiteOpenHelper( context: this,
            name: "administracion", factory: null, version: 1);
    SQLiteDatabase bd = admin.getWritableDatabase();
    String matriculaText = matricula.getText().toString();

    if(!matriculaText.isEmpty()){
        bd.execSQL("delete from estudiantes where id_estudiante=" + matriculaText);
        bd.close();
        matricula.setText("");
        nombres.setText("");
        apellidos.setText("");
        edad.setText("");
        idCarrera.setText("");
        Toast.makeText( context: this, text: "Estudiante eliminado de la base de datos.",
                Toast.LENGTH_SHORT).show();
    }else{
        Toast.makeText( context: this, text: "Ingrese un número de matrícula.",
                Toast.LENGTH_SHORT).show();
    }
}
```

En este caso, al igual que el paso 3 y 6, se obtiene la base de datos de manera Writable y se usa el método execSQL().

8. Para los métodos de los pasos 5, 6 y 7, se agregan al .xml de la misma manera que el paso 4.

[NT5] NOTA TÉCNICA Nº. 5
DESARROLLO MÓVIL

Paso 2. Desarrollo del código java [Base de datos interna]

1. Se crean los archivos necesarios para los query en el servidor que hospeda la página web.
2. Nos aseguramos de que los permisos necesarios están establecidos en el archivo Manifest de Android Studio.
3. Se establece el método para guardar la información:

```
public void registro(View view) {
    String URL_registro = URL+"ingreso_estudiante.php";

    String matriculaText = matricula.getText().toString();
    String nombresText = nombres.getText().toString();
    String apellidosText = apellidos.getText().toString();
    String edadText = edad.getText().toString();
    String idCarreraText = idCarrera.getText().toString();

    if(!matriculaText.isEmpty()||!nombresText.isEmpty()||!apellidosText.isEmpty()||
            !edadText.isEmpty()||!idCarreraText.isEmpty()){
        StringRequest strRq = new StringRequest(Request.Method.POST, URL_registro,
                new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                matricula.setText("");
                nombres.setText("");
                apellidos.setText("");
                edad.setText("");
                idCarrera.setText("");
            }
        }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                Toast.makeText(getApplicationContext(),error.toString(),
                        Toast.LENGTH_SHORT).show();
            }
        }){
            @Override
            protected Map<String, String> getParams() throws AuthFailureError {
                Map<String, String> parametros = new HashMap<>();
                parametros.put("matricula",matricula.getText().toString());
                parametros.put("nombres",nombres.getText().toString());
                parametros.put("apellidos",apellidos.getText().toString());
                parametros.put("edad",edad.getText().toString());
                parametros.put("idCarrera", idCarrera.getText().toString());
                return parametros;
            }
        };
        RequestQueue requestQueue = Volley.newRequestQueue( context: this);
        requestQueue.add(strRq);

        Toast.makeText( context: this, text: "Se cargaron los datos del estudiante",
                Toast.LENGTH_SHORT).show();
    }else{
        Toast.makeText( context: this, text: "Por favor, llene todos los campos.",
                Toast.LENGTH_SHORT).show();
    }
}
```

Se utiliza un String Request para manejar la información, los EditText quedarán vacíos una vez se obtenga una respuesta por parte del servidor web. Se envían los parámetros haciendo uso de un Map, definiendo correctamente sus id y su contenido. Hay que tener en cuenta que el id colocado en Android Studio debe coincidir en el colocado en el servidor web.

4. Para consultar al estudiante por su número de matrícula:

```
public void consultaMatricula(View view) {
    String matriculaText = matricula.getText().toString();
    String URL_consultaMat = URL+"busqueda_matricula.php?matricula="+matriculaText;

    if(!matriculaText.isEmpty()){
        JsonArrayRequest jsonArrayRequest = new JsonArrayRequest(URL_consultaMat,
                new Response.Listener<JSONArray>() {
                    @Override
                    public void onResponse(JSONArray response) {
                        JSONObject jsonObject = null;
                        for (int i = 0; i < response.length(); i++) {
                            try {
                                jsonObject = response.getJSONObject(i);
                                nombres.setText(jsonObject.get("nombres").toString());
                                apellidos.setText(jsonObject.get("apellidos").toString());
                                edad.setText(jsonObject.get("edad").toString());
                                idCarrera.setText(jsonObject.get("idCarrera").toString());
                                Toast.makeText(getApplicationContext(), text: "Consulta exitosa."
                                        ,Toast.LENGTH_SHORT).show();
                            } catch (JSONException je) {
                                Log.e( tag: "ERROR_JSON",je.getMessage());
                                Toast.makeText(getApplicationContext(),
                                        text: "No existe un estudiante con dicha matrícula",
                                        Toast.LENGTH_SHORT).show();
                            }
                        }
                    }
                }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                Log.e( tag: "ERROR_CONEXION",error.getMessage());
                Toast.makeText(getApplicationContext(), text: "Usuario o contraseña incorrecta.",
                        Toast.LENGTH_SHORT).show();
            }
        });
        RequestQueue requestQueue = Volley.newRequestQueue( context: this);
        requestQueue.add(jsonArrayRequest);
    }else{
        Toast.makeText( context: this, text: "Ingrese un número de matrícula.",
                Toast.LENGTH_SHORT).show();
    }
}
```

En este caso, como se desea saber la información de un estudiante, se obtiene un Json Array, del cual se obtendrá la información en forma de un Json Object, al cual se le especificará la clave del dato que se desea obtener.

5. Para modificar los datos de un estudiante:

[NT5] NOTA TÉCNICA Nº. 5
DESARROLLO MÓVIL

```
public void modificarInformacion(View v) {
    String URL_modificar = URL+"modificar_estudiante.php";
    String matriculaText = matricula.getText().toString();
    String nombresText = nombres.getText().toString();
    String apellidosText = apellidos.getText().toString();
    String edadText = edad.getText().toString();
    String idCarreraText = idCarrera.getText().toString();

    StringRequest strRq = new StringRequest(Request.Method.POST,URL_modificar,
            new Response.Listener<String>() {
                @Override
                public void onResponse(String response) {
                    matricula.setText("");
                    nombres.setText("");
                    apellidos.setText("");
                    edad.setText("");
                    idCarrera.setText("");
                    Toast.makeText(getApplicationContext(), text: "Se modificaron los datos.",
                            Toast.LENGTH_SHORT).show();
                }
            }, new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error) {
            Toast.makeText(getApplicationContext(),error.toString(), Toast.LENGTH_SHORT).show();
        }
    }){
        @Override
        protected Map<String, String> getParams() throws AuthFailureError {
            Map<String, String> parametros = new HashMap<~>();
            parametros.put("matricula",matricula.getText().toString());
            parametros.put("nombres",nombres.getText().toString());
            parametros.put("apellidos",apellidos.getText().toString());
            parametros.put("edad",edad.getText().toString());
            parametros.put("idCarrera", idCarrera.getText().toString());
            return parametros;
        }
    };
    RequestQueue requestQueue = Volley.newRequestQueue( context: this);
    requestQueue.add(strRq);

    if(!matriculaText.isEmpty()){
        if(nombresText.isEmpty()||apellidosText.isEmpty()||edadText.isEmpty()
                ||idCarreraText.isEmpty()){
            Toast.makeText( context: this, text: "Ingrese todos los datos.",
                    Toast.LENGTH_SHORT).show();
        }

    }else{
        Toast.makeText( context: this, text: "Ingrese una matrícula.",
                Toast.LENGTH_SHORT).show();
    }
}
```

Para modificar la información de un estudiante, se utiliza el mismo código del paso 3.

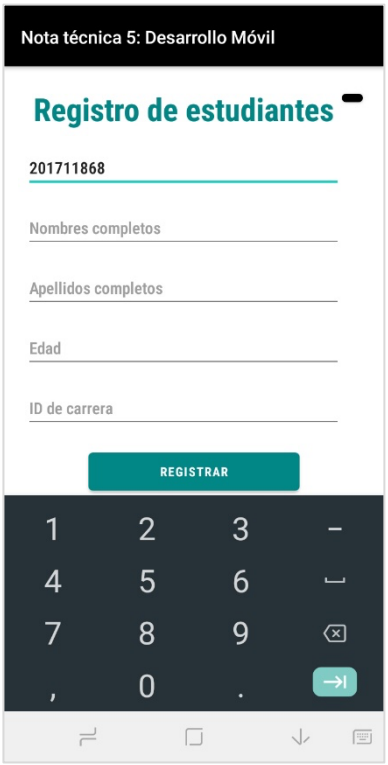
6. Para eliminar a un estudiante de la base de datos:

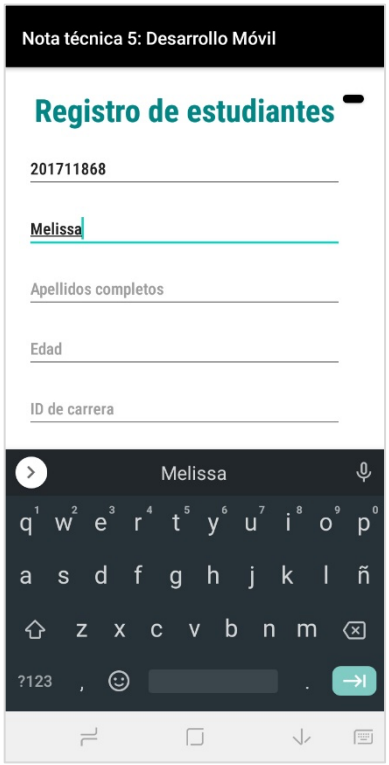
```
public void eliminarEstudiante(View v) {
    String matriculaText = matricula.getText().toString();
    String URL_eliminar = URL+"eliminar_estudiante.php";

    if(!matriculaText.isEmpty()){
        StringRequest strRq = new StringRequest(Request.Method.POST, URL_eliminar,
                new Response.Listener<String>() {
                    @Override
                    public void onResponse(String response) {
                        Toast.makeText(getApplicationContext(),
                                text: "Estudiante eliminado de la base de datos.",
                                Toast.LENGTH_SHORT).show();
                    }
                }, new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                Toast.makeText(getApplicationContext(),error.toString(),
                        Toast.LENGTH_SHORT).show();
            }
        }){
            @Override
            protected Map<String, String> getParams() throws AuthFailureError {
                Map<String, String> parametros = new HashMap<~>();
                parametros.put("matricula",matriculaText);
                return parametros;
            }
        };
        RequestQueue requestQueue = Volley.newRequestQueue( context: this);
        requestQueue.add(strRq);
        Toast.makeText( context: this, text: "Estudiante eliminado de la base de datos.",
                Toast.LENGTH_SHORT).show();
    }else{
        Toast.makeText( context: this, text: "Ingrese un número de matrícula.",
                Toast.LENGTH_SHORT).show();
    }
}
```

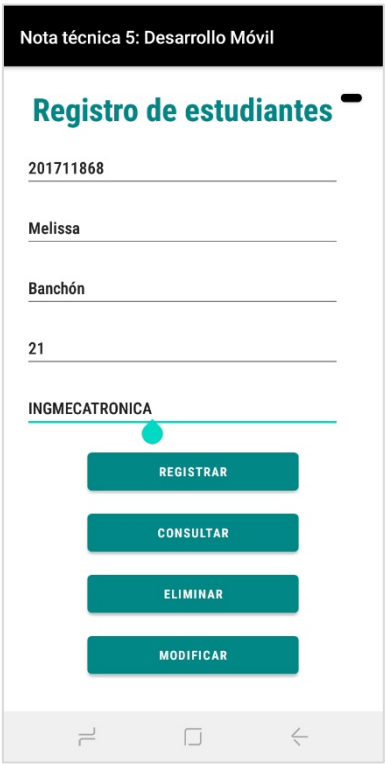
Para eliminar a un estudiante, se utiliza el código de la misma manera que en el paso anterior, esta vez, solo enviando la matrícula del estudiante a eliminar.

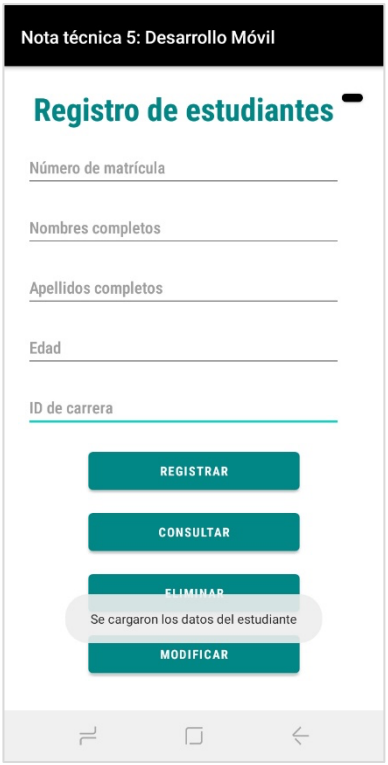
[NT5] NOTA TÉCNICA Nº. 5
DESARROLLO MÓVIL

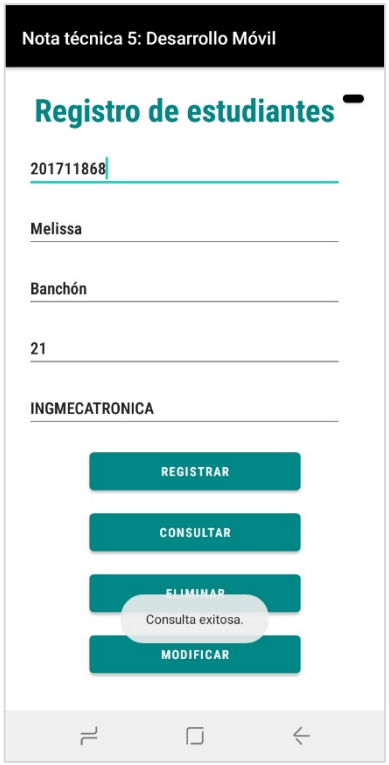
Funcionamiento de la aplicación móvil:

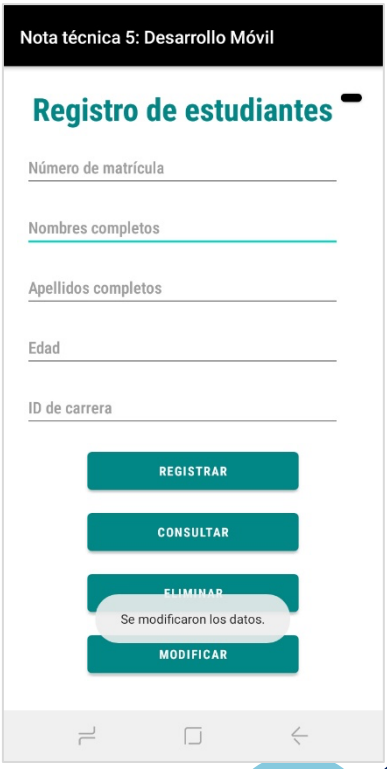












[NT5] NOTA TÉCNICA NO. 5
DESARROLLO MÓVIL