

PROFESOR: Layonet Salvador Buenrostro Pérez.

MATERIA: Diseño e Implementación de Aplicación móvil

ALUMNO: OLVERA AVILES AURELIO DANIEL

MATRICULA: 161100092

Diseño UI

## INDÍCE

N.P	TEMA	PAG.
1	Marco Teórico	3
	• 2.1 SharedPreferences	3
	• 2.2 PreferencesActivity	4
	• 2.3 Sistema de archivos interno y externo	6
	• 2.4. Base de datos en SQLite	8
2	Desarrollo	13
3	Resultados	36
4	Conclusión	39
5	Referencias	40

## INDÍCE FIGURAS

N.P	TEMA	PAG.
1	Figura 1. Opción de Checkbox	4
2	Figura 2. Opción EditTextPreference	4
3	Figura 3. Opción ListPreference	5
4	Figura 4. Atributos ListPreference	5
5	Figura 5. Opción MultiSelectListPreference	6
6	Figura 6. Ejemplo completo opciones PreferencesActivity	6
7	Figura 7. Creación de la clase área	9
8	Figura 8. Creación de la clase contrato	10
9	Figura 9. Creación de la base de datos	11

## MARCO TEORICO

### 2.1 SharedPreferences

Las preferencias no son más que datos que se guardan para ser compartidos con otros métodos y/o actividades. En Android hay varias maneras de almacenar datos: Puedes hacerlo a través de una base de datos local y/o remota, a través de las Shared Preferences o combinar ambas formas. Normalmente se usan para almacenar información personal, opciones de presentación, etc. La limitación la ponemos nosotros, ya que está en la imaginación del propio diseñador de la aplicación. (Saúl Cintero 2020)

La gestión de estas preferencias se centraliza en la clase SharedPreferences, que representa una colección de preferencias. A la hora de instanciarlas, estas tienen un identificador, que es la clave o nombre que le hayamos puesto nosotros, y un modo de acceso:

**MODE\_PRIVATE:** Nuestra aplicación tiene acceso a estas preferencias.

**MODE\_WORLD\_READABLE:** Todas las aplicaciones tienen acceso en modo lectura a estas preferencias, pero únicamente nuestra aplicación puede modificarlas.

**MODE\_WORLD\_WRITABLE:** Todas las aplicaciones pueden leer y escribir sobre estas preferencias.

Para hacer una llamada a nuestras preferencias, debes incluir el siguiente código:

**SharedPreferences prefs = getSharedPreferences("MyPreferences", Context.MODE\_PRIVATE);**

Una vez referenciada nuestra colección de preferencias, ahora podemos obtener sus valores mediante el método get y escribir mediante el método put.

Por ejemplo, para obtener el valor de una preferencia llamada "url" de tipo String deberíamos poner lo siguiente:

**String url = prefs.getString("url", "www.google.com");**

Con esto le estamos diciendo que obtenga en valor de la preferencia "url" o, si estuviese vacío, obtuviese el valor por defecto, que en este caso es "www.google.com".

También podríamos haber hecho lo siguiente:

**String url = prefs.getString("url", "");**

En el cual el valor por defecto estaría vacío.

Ten en cuenta que también admiten otros métodos para el resto de tipos de datos básicos, como son:

- getBoolean()
- getInt()
- getFloat()
- getLong()

Para insertar un valor o actualizarlo, hay que hacerlo utilizando el objeto SharedPreferences. Editor. Una vez referenciado el editor, se ha de realizar un put sobre el tipo de dato básico correspondiente y validamos mediante el método commit(). Siguiendo el ejemplo de la URL:

**SharedPreferences prefs = getSharedPreferences("MyPreferences", Context.MODE\_PRIVATE);**

```
SharedPreferences.Editor editor = prefs.edit();  
  
editor.putString("url", "www.saulcintero.com");  
  
editor.commit();
```

## 2.2 PreferencesActivity

Un mecanismo que nos permite gestionar fácilmente las opciones de una aplicación permitiéndonos guardarlas en XML de una forma transparente para el programador. En aquel momento sólo vimos cómo hacer uso de ellas mediante código, es decir, creando nosotros mismos los objetos necesarios (SharedPreferences) y añadiendo, modificando y/o recuperando «a mano» los valores de las opciones a través de los métodos correspondientes (getString(), putString(), ...). Sin embargo, ya avisamos de que Android ofrece una forma alternativa de definir mediante XML un conjunto de opciones para una aplicación y crear por nosotros las pantallas necesarias para permitir al usuario modificarlas a su antojo. A esto dedicaremos este segundo artículo sobre preferencias. (Sgoliver 2020)

Dentro de cada categoría podremos añadir cualquier número de opciones, las cuales pueden ser de distintos tipos, entre los que destacan:

- CheckBoxPreference. Marca seleccionable.
- EditTextPreference. Cadena simple de texto.
- ListPreference. Lista de valores seleccionables (exclusiva).
- MultiSelectListPreference. Lista de valores seleccionables (múltiple).

Cada uno de estos tipos de preferencia requiere la definición de diferentes atributos, que iremos viendo en los siguientes apartados.

### CheckBoxPreference

Representa un tipo de opción que sólo puede tomar dos valores distintos: activada o desactivada. Es el equivalente a un control de tipo checkbox. En este caso tan sólo tendremos que especificar los atributos: nombre interno de la opción (android:key), texto a mostrar (android:title) y descripción de la opción (android:summary). Veamos un ejemplo:

```
<CheckBoxPreference  
    android:key="opcion1"  
    android:title="Preferencia 1"  
    android:summary="Descripción de la preferencia 1" />
```

Figura 1. Opción de Checkbox

### EditTextPreference

Representa un tipo de opción que puede contener como valor una cadena de texto. Al pulsar sobre una opción de este tipo se mostrará un cuadro de diálogo sencillo que solicitará al usuario el texto a almacenar. Para este tipo, además de los tres atributos comunes a todas las opciones (key, title y summary) también tendremos que indicar el texto a mostrar en el cuadro de diálogo, mediante el atributo android:dialogTitle. Un ejemplo sería el siguiente:

```
<EditTextPreference  
    android:key="opcion2"  
    android:title="Preferencia 2"  
    android:summary="Descripción de la preferencia 2"  
    android:dialogTitle="Introduce valor" />
```

Figura 2. Opción EditTextPreference

## ListPreference

Representa un tipo de opción que puede tomar como valor un elemento, y sólo uno, seleccionado por el usuario entre una lista de valores predefinida. Al pulsar sobre una opción de este tipo se mostrará la lista de valores posibles y el usuario podrá seleccionar uno de ellos. Y en este caso seguimos añadiendo atributos. Además de los cuatro ya comentados (key, title, summary y dialogTitle) tendremos que añadir dos más, uno de ellos indicando la lista de valores a visualizar en la lista y el otro indicando los valores internos que utilizaremos para cada uno de los valores de la lista anterior (Ejemplo: al usuario podemos mostrar una lista con los valores español y francés, pero internamente almacenarlos como «ESP» y «FRA»).

Estas listas de valores las definiremos también como ficheros XML dentro de la carpeta /res/xml. Definiremos para ello los recursos de tipos <string-array> necesarios, en este caso dos, uno para la lista de valores visibles y otro para la lista de valores internos, cada uno de ellos con su ID único correspondiente. Veamos cómo quedarían dos listas de ejemplo, en un fichero llamado codigospaises.xml:

```
<?xml version="1.0" encoding="utf-8" ?>
<resources>
    <string-array name="pais">
        <item>España</item>
        <item>Francia</item>
        <item>Alemania</item>
    </string-array>
    <string-array name="codigopais">
        <item>ESP</item>
        <item>FRA</item>
        <item>ALE</item>
    </string-array>
</resources>
```

Figura 3. Opción ListPreference

En la preferencia utilizaremos los atributos android:entries y android:entryValues para hacer referencia a estas listas, como vemos en el ejemplo siguiente:

```
<ListPreference
    android:key="opcion3"
    android:title="Preferencia 3"
    android:summary="Descripción de la preferencia 3"
    android:dialogTitle="Indicar Pais"
    android:entries="@array/pais"
    android:entryValues="@array/codigopais" />
```

Figura 4. Atributos ListPreference

## MultiSelectListPreference

A partir de Android 3.0.x / Honeycomb. Las opciones de este tipo son muy similares a las ListPreference, con la diferencia de que el usuario puede seleccionar varias de las opciones de la lista de posibles valores. Los atributos a asignar son por tanto los mismos que para el tipo anterior.

```
<MultiSelectListPreference
    android:key="opcion4"
    android:title="Preferencia 4"
    android:summary="Descripción de la preferencia 4"
    android:dialogTitle="Indicar Pais"
    android:entries="@array/pais"
    android:entryValues="@array/codigopais" />
```

Figura 5. Opción MultiSelectListPreference

Como ejemplo completo, veamos cómo quedaría definida una pantalla de opciones con las 3 primeras opciones comentadas (ya que probaré con Android 2.2), divididas en 2 categorías llamadas por simplicidad «Categoría 1» y «Categoría 2». Llamaremos al fichero «opciones.xml».

```
<PreferenceScreen
    xmlns:android="http://schemas.android.com/apk/res/android">
    <PreferenceCategory android:title="Categoría 1">
        <CheckBoxPreference
            android:key="opcion1"
            android:title="Preferencia 1"
            android:summary="Descripción de la preferencia 1" />
        <EditTextPreference
            android:key="opcion2"
            android:title="Preferencia 2"
            android:summary="Descripción de la preferencia 2"
            android:dialogTitle="Introduce valor" />
    </PreferenceCategory>
    <PreferenceCategory android:title="Categoría 2">
        <ListPreference
            android:key="opcion3"
            android:title="Preferencia 3"
            android:summary="Descripción de la preferencia 3"
            android:dialogTitle="Indicar Pais"
            android:entries="@array/pais"
            android:entryValues="@array/codigopais" />
    </PreferenceCategory>
</PreferenceScreen>
```

Figura 6. Ejemplo completo opciones PreferencesActivity

## 2.3 Sistema de archivos interno y externo

### ¿Qué son los sistemas de archivo?

Tu disco duro externo, el disco duro interno de tu ordenador, un USB o una tarjeta SD. Todos ellos son unidades de almacenamiento, lo que quiere decir que cuando los formateas estás creando la infraestructura en las que van a alojarse los datos que después le quieras meter.

Es aquí donde entra en juego el sistema de archivos, un componente del sistema operativo que se encarga de administrar la memoria de cada unidad. Se encargan de asignarle a los archivos el espacio que necesiten, ordenarlos, permitir el acceso a ellos y administrar el espacio libre de las unidades de almacenamiento.

Es como un bibliotecario, que ordena y registra la posición exacta en la que se ha escrito un fichero dentro de la unidad, y así tu sistema operativo puede acceder rápidamente a ellos y saber dónde empieza y acaba cada uno.

Siguiendo con la analogía bibliotecaria, de la misma manera que cada bibliotecario puede tener su método para organizar los libros, cada sistema de archivo hace lo mismo, organizando y gestionando los datos de maneras diferentes. Cada sistema de archivos tiene sus propias ventajas y limitaciones, por lo que es importante conocerlos para elegir el que mejor se ajusta a cada necesidad que tengas. (Xataka Basic 2020)

### **Características de los principales sistemas de archivos**

Como hemos dicho, hay diferentes tipos de sistemas de archivos cada uno con sus ventajas y desventajas. Algunos de ellos seguro que los has visto más de una vez, y puede que otros no tanto. Algunos de los más conocidos son los FAT32, exFAT, NTFS, HFS+, ext2, ext3 y ext4.

#### **Sistema de archivos FAT32**

Habiéndose establecido en 1996, es uno de los viejos rockeros del mundo de los sistemas de archivo, robusto pero anticuado. Eso sí, es tremendamente versátil gracias a su enorme compatibilidad con prácticamente todos los dispositivos y sistemas operativos, razón por la que la mayoría de unidades USB que te compres estarán formateadas con él.

Su mayor y más popular limitación es que sólo permite guardar archivos de hasta 4 GB, por lo que si quieres guardar un único archivo que ocupe más que eso no te va a quedar más remedio que formatear con otro sistema de archivos. Su lado positivo es que es perfectamente compatible con Windows, macOS y GNU/Linux, y funciona sin problemas en los viejos USB 2.0.

#### **Sistema de archivos exFAT**

Podríamos referirnos al sistema exFAT como una actualización al FAT32 introducida por Microsoft en Windows Vista con la intención de acabar con los quebraderos de cabeza que provoca la limitación de 4 GB de su hermano mayor.

En cuestión de compatibilidad puedes usarlo en Windows, macOS o GNU/Linux, aunque sólo en las versiones más recientes como a partir de Windows XP SP3 u OS X 10.6.5 Snow leopard. Es un sistema de archivos muy recomendado para unidades externas como un USB o tarjeta SD donde vayas a guardar archivos de más de 4 GB y no quieras tener problemas de compatibilidad.

#### **Sistema de archivos NTFS**

Se trata de otra alternativa al sistema FAT32 promovida por Microsoft, de hecho, es el sistema de archivos que Windows utiliza por defecto. Sin los límites del tamaño máximo de archivo del FAT32, el NTFS se convierte en una muy buena opción para discos duros y otras unidades externas, por lo menos si eres usuario de Windows.

Y es que su mayor desventaja es que no es totalmente compatible con todos los sistemas operativos. Por ejemplo, de forma nativa macOS puede leer las unidades formateadas con él, pero no puede escribir en ellas. Esto quiere decir que si tienes un disco duro con NTFS no podrás guardar nada de tu Mac a no ser que lo formatees con otro sistema de archivos.

#### **Sistema de archivos HFS+**

De la misma manera que el NTFS es uno de los actuales sistemas de archivo de referencia en Windows, Apple creó el sistema HFS+ a su medida. Se da la casualidad de que mientras los sistemas GNU/Linux pueden trabajar con él sin problemas, en Windows sólo podrás leer el contenido de los discos formateados con él, pero no escribir en ellos.

Eso hace de este sistema de archivos uno perfecto si estamos dentro del ecosistema de Apple utilizando sus dispositivos. Pero si eres usuario de Windows vas a necesitar utilizar cualquiera de los otros.

#### **Sistema de archivos Ext2, ext3 y ext4**



Y terminamos con esta última familia de sistemas de archivos. Así como Apple y Microsoft tienen sus propios sistemas, estos tres (cada uno evolución del anterior) son los utilizados por las distribuciones GNU/Linux. El principal inconveniente es que sólo puede ser utilizado en esta familia de sistemas operativos. (Xataka Basic 2020)

## 2.4. Base de datos en SQLite

### ¿Por qué usar bases de datos SQLite?

En Android existen varias formas, sobre cómo guardar datos de forma persistente.

Entre los métodos más comunes encontramos:

- Bases de datos SQLite
- SharedPreferences
- Almacenamiento en disco

Guardar datos de forma persistente significa que los datos persisten incluso si se apaga y se vuelve a prender el dispositivo.

Generalmente cuando cerramos una aplicación, Android va a liberar los recursos que ésta tenía asignados.

Nuestras variables que estaban definidas dejarán de existir. Es por eso que necesitamos guardar estos datos si son importantes.

Y, hay que tener en cuenta que estos datos se están registrando de forma local.

Es decir, estas 3 opciones almacenan los datos en el mismo dispositivo, sin requerir de una conexión a internet. (PYM 2020)

### ¿Cuándo usar SQLite en Android?

Las preferencias son muy fáciles de usar, y resultan muy prácticas.

Si tenemos que registrar datos simples como, el idioma que prefiere el usuario, el skin que quiere usar, fecha y hora de la última acción que realizó, podemos usar SharedPreferences sin problema alguno.

Las preferencias nos permiten guardar datos bien puntuales.

A diferencia de ello, las bases de datos nos permiten guardar información más elaborada.

Si tienes experiencia usando MySQL, SQLServer, Oracle, PostgreSQL o algún otro gestor de bases de datos relacionales, esto te resultará muy familiar.

Sucede que SQLite nos ofrece también la posibilidad de crear bases de datos relacionales.

Puedes imaginar a SQLite como un MySQL pequeño, o una simplificación de los antes mencionados.

Se escogió esta opción porque nos permitirá crear bases de datos muy ligeras, de forma que nuestra app podrá interactuar con los datos incluso desde dispositivos con pocos recursos.

Si solo necesitas guardar un entero, una cadena u otro valor en específico, basta con usar SharedPreferences.

Pero, si en cambio requieres de una estructura para tus datos, SQLite es una muy buena opción.



Para darte un ejemplo: Ahora mismo, yo he terminado de desarrollar una app. Esta app se conecta con una API, que le permite consumir y guardar datos. Todo OK. Pero ahora me comentan que se necesita usar la app en modo offline. SQLite es la solución. (PYM 2020)

A modo de ejemplo, vamos a:

- Crear una base de datos con una tabla llamada "areas".
- Definir las columnas (nombres y tipos de dato) que usará esta tabla.
- Implementar las operaciones de registro, edición y eliminación de filas en esta tabla.
- Guardar datos, editarlos y recuperarlos usando las operaciones definidas.

### Paso 1: Definir un esquema inicial

Vamos a trabajar con una entidad "Area", que tendrá solo 2 datos: un ID y un nombre. La idea es entender cómo funciona todo. Si entendemos todo a la perfección entonces luego será sencillo usar tablas con más datos, y trabajar con varias tablas que están relacionadas.

He aquí nuestra clase Area:

```
public class Area {  
  
    private int id;  
    private String name;  
  
    public int getId() {  
        return id;  
    }  
  
    public void setId(int id) {  
        this.id = id;  
    }  
  
    public String getName() {  
        return name;  
    }  
  
    public void setName(String name) {  
        this.name = name;  
    }  
}
```

Figura 7. Creación de la clase área

Esta clase únicamente expresa los datos que queremos registrar. Nuestra base de datos tendrá una tabla, y las columnas de esta tabla tendrán un nombre y un tipo de dato. Esto tenemos que expresarlo a través de un Contract. Un contrato es simplemente una clase que nos permitirá guardar allí algunas constantes.

¿Eso significa que no es obligatorio su uso?

Exacto. Podríamos no definir constantes y usar directamente los valores en nuestras sentencias SQL. El punto es que, si necesitamos hacer algún cambio en el futuro, vamos a tener que hacer cambios aquí y allá. Al usar constantes, solo modificamos el valor de estas constantes, y todas nuestras sentencias usarán los nuevos valores.

He aquí el contrato:

```
public final class AreaContract {

    private AreaContract() {}

    public static class AreaEntry implements BaseColumns {
        public static final String TABLE_NAME = "areas";

        public static final String COLUMN_ID = "id";
        public static final String COLUMN_NAME = "name";
    }
}
```

Figura 8. Creación de la clase contrato

#### Acerca de BaseColumns:

Seguro notaste que la clase interna AreaEntry implementa una interfaz.

BaseColumns es una interfaz que define un atributo \_ID.

Este identificar lo genera Android, y es muy distinto al ID que vamos a guardar nosotros (el ID que guardamos por nuestra cuenta se asocia al ID que tiene cada área según nuestra API).

Es por eso que definimos el campo COLUMN\_ID por nuestra cuenta. Y usamos BaseColumns para tener nuestro código acorde con el estándar de Android.

Nos será útil posteriormente cuando usemos otras clases que requieran implementar esta interfaz.

#### Paso 2: Crear la base de datos

En el paso anterior hemos definido el esquema que tendrá nuestra base de datos. Ahora vamos a escribir el código que se encargará de crear verdaderamente la base de datos.

Para ello vamos a crear una clase llamada MyDbHelper.

Esta clase va a heredar de SQLiteOpenHelper, que es una clase abstracta, que nos ayudará a gestionar nuestra base de datos. Esta clase abstracta se encargará de usar la base de datos si ya existe, crearla si aún no existe, y actualizarla cuando sea necesario.

Gracias a que esta clase lleva un control del estado de nuestra base de datos, nosotros no tenemos que preocuparnos sobre cuándo ejecutar cada cosa.

Pero sí tenemos que decirle cómo hacerlo:

```
public class MyDbHelper extends SQLiteOpenHelper {
    private static final int DATABASE_VERSION = 1;
    private static final String DATABASE_NAME = "pym.db";

    public MyDbHelper(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }

    public void onCreate(SQLiteDatabase db) {
        db.execSQL("CREATE TABLE " + AreaEntry.TABLE_NAME + " (" +
            AreaEntry._ID + " INTEGER PRIMARY KEY," +
            AreaEntry.COLUMN_NAME + " TEXT)");
    }

    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS " + AreaEntry.TABLE_NAME);
        onCreate(db);
    }
}
```

Figura 9. Creación de la base de datos

Aquí encontramos:

- Un atributo entero **DATABASE\_VERSION**, que indica la versión de nuestra base de datos.
- Un atributo cadena **DATABASE\_NAME** que indica con qué nombre de archivo se guardará nuestra base de datos.
- Una implementación de los métodos **onCreate** y **onUpgrade**.

Partiendo de esta información debemos tener en cuenta que:

- El método **onCreate** se encargará de crear el esquema de nuestra base de datos (tablas, claves primarias, claves foráneas).
- Si altermos el esquema de nuestra base de datos, debemos aumentar el valor de **DATABASE\_VERSION**, para que nuestro helper ejecute **onUpgrade** la próxima vez.
- El método **onUpgrade** debería contener cierta lógica para migrar los datos existentes de una versión a otra.

En el ejemplo que estamos viendo, las Áreas se obtienen desde una API, y quedan almacenadas para que la aplicación pueda trabajar de modo offline.

Si en algún momento cambia el esquema de la base de datos, no hay problema en borrar la tabla y volver a cargar la información.

Estamos considerando el escenario más simple. En un futuro post vamos a ver circunstancias que requieran de un método **onUpgrade** más elaborado.

**¿Por qué no usamos una clave primaria autoincremental?**

En este caso queremos que el ID que devuelve la API para cada Area sea igual al ID almacenado de forma local.

El esquema de nuestra base de datos es relativo a las operaciones que pensamos realizar.

Por ejemplo:

- Si estamos desarrollando una aplicación que funcionará únicamente de modo offline, tiene sentido que los ID de nuestras tablas sean todas auto-incrementales.
- Si tenemos una tabla de "reportes realizados por usuarios", y queremos que nuestra app funcione tanto de modo online como offline, podríamos tener 2 tablas, una para guardar los reportes descargados desde la API, y otra para guardar reportes que se han creado de modo offline y que hacen falta publicar en nuestro servidor online.
- Pero ese no es el único camino, otros van a preferir tener una columna adicional (que existe solo de forma local) para diferenciar los reportes que se han sincronizado de los que aun no. Aquí se podría tener un ID autoincremental como clave primaria (muy aparte del ID usado de forma online), y una columna adicional contendría el ID que tiene el reporte en el servidor. Si es null significaría que aún no se ha subido, y si tiene un valor quiere decir que ya se sincronizó con el servidor y tiene un ID online. (PYM 2020)

### **Paso 3: Registrar datos usando ContentValues**

Veamos ahora cómo insertar datos.

Para hacer ello necesitamos:

- Instanciar nuestro DBHelper (pasándole como parámetro el contexto en que nos encontramos).
- Obtener una instancia de SQLiteDatabase en modo escritura apartir de nuestro helper.
- Crear un ContentValues y cargarlo con la información que deseamos registrar.
- Hacer efectiva la operación usando el método insert desde nuestra instancia SQLiteDatabase.

Si quieres que una tabla tenga valores desde su creación, entonces debes hacer la operación de inserción en el método onCreate. En ese caso los 2 primeros pasos se omiten porque ya se tiene a la mano una instancia de SQLiteDatabase. (PYM 2020)

## DESARROLLO

### Practica 1

#### MainActivity.java

```
package com.example.almacenamientoarchivos;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {
    EditText etUsuario, etPassword;
    Button btnIngresar, btnRegistrar;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        etPassword = (EditText)findViewById(R.id.etPassword);
        etUsuario = (EditText)findViewById(R.id.etUsuario);
        btnIngresar = (Button)findViewById(R.id.btnIngresar);
        btnRegistrar = (Button)findViewById(R.id.btnRegistrar);

        //Creamos la funcion del boton Ingresar
        btnIngresar.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //Traemos los datos que se relacionaron con la vista
                anteriormente String usuario=etUsuario.getText().toString();
                String password=etPassword.getText().toString();

                //Creamos una estructura de seleccion para indicar que se
                ingresarea con el usuario y la contraseña

                if (usuario.equals("Admin")&& password.equals("123")){
                    //Al ingresar los caracteres correcto, este mandara a la
                    siguiente actividad
                    Intent intent=new Intent(MainActivity.this,Ingresar.class);
                    startActivity(intent);
                    //En caso de ser lo contrario lo que pasara es que no
                    permitira el acceso y mandara un mensaje
                }else{
                    Toast.makeText(getApplicationContext(), "Usuario
```

```
Incorrecto", Toast.LENGTH_SHORT).show();
    }

    });
//Creamos la funcion del boton Registrar
    btnRegistrar.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            //Colocamos un intent para madar a la otra actividad
            Intent intent=new Intent(MainActivity.this,Registrar.class);
            //Colocamos el estar para que haga la accion
            startActivity(intent);
        }
    });
}
}
```

**registrar.java**

```
package com.example.almacenamientoarchivos;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;

public class Registrar extends AppCompatActivity {
    EditText etNombre,etPaterno,etMaterno,etEdad,etUsuario,etEmail,etPassword;
    Button btnRegistrar;
    //Colocamos el nombre del archivo que contendra nestros datos
    public static final String FILE_NAME="ejemplo.txt";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_registrar);

        etNombre=(EditText)findViewById(R.id.etNombre);
        etPaterno=(EditText)findViewById(R.id.etPaterno);
        etMaterno=(EditText)findViewById(R.id.etMaterno);
        etEdad=(EditText)findViewById(R.id.etEdad);
        etUsuario=(EditText)findViewById(R.id.etUsuario);
        etEmail=(EditText)findViewById(R.id.etEmail);
        etPassword=(EditText)findViewById(R.id.etPassword);
        btnRegistrar=(Button)findViewById(R.id.btnregistrar);
    }
}
```

```

    }
    //se creael metodo onclick del boton para que haga la fncion de guardar Los
    datos
    public void guardar(View view){
        String texto=etNombre.getText().toString();
        String texto1=etPaterno.getText().toString();
        String texto2=etMaterno.getText().toString();
        String texto3=etEdad.getText().toString();
        String texto4=etUsuario.getText().toString();
        String texto5=etEmail.getText().toString();
        String texto6=etPassword.getText().toString();

        //Se lla a la siguiente clase para poder escribir los datos en el
        archivo
        FileOutputStream fileOutputStream=null;

        try {
            fileOutputStream=openFileOutput(FILE_NAME,MODE_PRIVATE);
            //Se guardan las variables declaradas
            fileOutputStream.write(texto.getBytes());
            fileOutputStream.write(texto1.getBytes());
            fileOutputStream.write(texto2.getBytes());
            fileOutputStream.write(texto3.getBytes());
            fileOutputStream.write(texto4.getBytes());
            fileOutputStream.write(texto5.getBytes());
            fileOutputStream.write(texto6.getBytes());
            //Cuadno el usurio ingrese algo, mandaremos el metoo clear para limoar
            etNombre.getText().clear();
            etPaterno.getText().clear();
            etMaterno.getText().clear();
            etEdad.getText().clear();
            etUsuario.getText().clear();
            etEmail.getText().clear();
            etPassword.getText().clear();

            //Mensaje de la ruta en donde se guardo nuestro archivo con los daros
            Toast.makeText(this, "Guardado en" + getFilesDir() + "/" +
            FILE_NAME, Toast.LENGTH_SHORT).show();
        }catch (FileNotFoundException e){
            e.printStackTrace();
        }catch (IOException e){
            e.printStackTrace();
        }finally {
            if (fileOutputStream != null){
                try {
                    fileOutputStream.close();
                }catch (IOException e){
                    e.printStackTrace();
                }
            }
        }
    }
}

```



```
ingresar.java
package com.example.almacenamientoarchivos;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

import java.io.BufferedReader;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStreamReader;
import static com.example.almacenamientoarchivos.Registrar.FILE_NAME;

public class Ingresar extends AppCompatActivity {

    TextView tvMostrar;
    Button btnMostrar;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_ingresar);

        btnMostrar = (Button)findViewById(R.id.btnMostrar);
        tvMostrar=(TextView)findViewById(R.id.tvMostrar);
    }

    public void mostrar(View view){
        FileInputStream fileOutputStream=null;
        try {
            fileOutputStream=openFileInput(FILE_NAME);

            InputStreamReader inputStreamReader=new
            InputStreamReader(fileOutputStream);

            BufferedReader bufferedReader= new
            BufferedReader(inputStreamReader);
            StringBuilder stringBuilder= new StringBuilder();

            String texto;

            while ((texto=bufferedReader.readLine())!=null){
                stringBuilder.append(texto).append("\n");

                tvMostrar.setText(stringBuilder.toString() + "\n");
            }
        }catch (FileNotFoundException e){
            e.printStackTrace();
        }catch (IOException e){
            e.printStackTrace();
        }
    }
}
```

```

    }finally {
        if (fileOutputStream!=null){
            try {
                fileOutputStream.close();
            }catch (IOException e){
                e.printStackTrace();
            }
        }
    }
}

```

#### activity\_main.xml

```

<?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="wrap_content"
        android:layout_height="277dp"
        android:src="@drawable/fondo3"
        />

    <LinearLayout
        android:layout_marginTop="15dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">

        <com.google.android.material.textfield.TextInputLayout
            android:id="@+id/tilNombre"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="32dp">

            <EditText
                android:layout_marginRight="35dp"
                android:id="@+id/etUsuario"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:hint="Usuario"
                android:inputType="text"/>
        </com.google.android.material.textfield.TextInputLayout>

        <com.google.android.material.textfield.TextInputLayout
            android:id="@+id/tilDomicilio"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_marginLeft="32dp">

```

```

        <EditText
            android:id="@+id/etPassword"
            android:layout_marginRight="35dp"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:hint="Password"
            android:inputType="textPassword" />
    </com.google.android.material.textfield.TextInputLayout>

    <Button
        android:layout_marginTop="15dp"
        android:id="@+id/btnIngresar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginLeft="75dp"
        android:layout_marginRight="75dp"
        android:text="Ingresar" />

    <Button
        android:id="@+id/btnRegistrar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginLeft="75dp"
        android:layout_marginRight="75dp"
        android:text="Registrar" />

</LinearLayout>

</LinearLayout>

activity_registrar.xml
<?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:orientation="vertical"
    android:layout_height="match_parent"
    tools:context=".Registrar">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Registrate"
        android:gravity="center"
        android:textStyle="bold"
        android:textSize="45dp"
        android:textColor="#FF0000"/>

    <LinearLayout
        android:layout_marginTop="15dp"

```

```
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:orientation="vertical">

<com.google.android.material.textfield.TextInputLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="32dp">

    <EditText
        android:layout_marginRight="35dp"
        android:id="@+id/etNombre"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Nombre"
        android:inputType="text"/>
</com.google.android.material.textfield.TextInputLayout>

<com.google.android.material.textfield.TextInputLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="32dp">

    <EditText
        android:id="@+id/etPaterno"
        android:layout_marginRight="35dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Apellido Paterno"
        android:inputType="text" />
</com.google.android.material.textfield.TextInputLayout>

<com.google.android.material.textfield.TextInputLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="32dp">

    <EditText
        android:layout_marginRight="35dp"
        android:id="@+id/etMaterno"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Materno"
        android:inputType="text"/>
</com.google.android.material.textfield.TextInputLayout>

<com.google.android.material.textfield.TextInputLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="32dp">

    <EditText
        android:id="@+id/etEdad"
        android:layout_marginRight="35dp"
```

```
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Edad"
        android:inputType="number" />
</com.google.android.material.textfield.TextInputLayout>

<com.google.android.material.textfield.TextInputLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="32dp">

    <EditText
        android:layout_marginRight="35dp"
        android:id="@+id/etUsuario"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Usuario"
        android:inputType="text"/>
</com.google.android.material.textfield.TextInputLayout>

<com.google.android.material.textfield.TextInputLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="32dp">

    <EditText
        android:id="@+id/etEmail"
        android:layout_marginRight="35dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Email"
        android:inputType="textEmailAddress" />
</com.google.android.material.textfield.TextInputLayout>

<com.google.android.material.textfield.TextInputLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginLeft="32dp">

    <EditText
        android:id="@+id/etPassword"
        android:layout_marginRight="35dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Password"
        android:inputType="textPassword" />
</com.google.android.material.textfield.TextInputLayout>

<Button
    android:layout_marginTop="5dp"
    android:id="@+id/btnregistrar"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
```

```

        android:layout_marginLeft="75dp"
        android:layout_marginRight="75dp"
        android:onClick="guardar"
        android:text="Registrar" />

```

```

    </LinearLayout>

```

```

</LinearLayout>

```

#### activity\_ingresar.xml

```

<?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=".Ingresar">

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Tu Registro"
        android:gravity="center"
        android:textStyle="bold"
        android:textSize="45dp"
        android:textColor="#FF0000"/>

    <LinearLayout
        android:layout_marginTop="15dp"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical">

        <Button
            android:layout_marginTop="5dp"
            android:id="@+id/btnMostrar"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:layout_gravity="center"
            android:onClick="mostrar"
            android:layout_marginLeft="75dp"
            android:layout_marginRight="75dp"
            android:text="Mostrar" />

        <TextView
            android:id="@+id/tvMostrar"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:gravity="center"

```

```
android:textStyle="bold"
android:textSize="45dp"
android:textColor="@color/teal_200"/>
```

```
</LinearLayout>
```

```
</LinearLayout>
```

## Practica 2

### MainActivity.java

```
package com.example.sharedpreference1;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
public class MainActivity extends AppCompatActivity {
    EditText etNombre, etApellidoPaterno, etApellidoMaterno, etEdad, etUsuario,
    etEmail, etPassword;
    Button btnEnviar, btnMensaje;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_registro);
        etNombre = (EditText)findViewById(R.id.etNombre);
        etApellidoPaterno = (EditText)findViewById(R.id.etApellidoPaterno);
        etApellidoMaterno = (EditText)findViewById(R.id.etApellidoMaterno);
        etEdad = (EditText)findViewById(R.id.etEdad);
        etUsuario = (EditText)findViewById(R.id.etUsuario);
        etEmail = (EditText) findViewById(R.id.etEmail);
        etPassword = (EditText)findViewById(R.id.etPassword);
        btnEnviar = (Button)findViewById(R.id.btnEnviar);
        btnMensaje = (Button)findViewById(R.id.btnMensaje);
    }
    public void guardar(View view){
        SharedPreferences preferences = getSharedPreferences("datos",
        Context.MODE_PRIVATE);
        String nombre = etNombre.getText().toString();
        String aPaterno = etApellidoPaterno.getText().toString();
        String aMaterno = etApellidoMaterno.getText().toString();
        String edad = etEdad.getText().toString();
        String usuario = etUsuario.getText().toString();
        19
        String email = etEmail.getText().toString();
        String password = etPassword.getText().toString();
        SharedPreferences.Editor obj_editor = preferences.edit();
        obj_editor.putString("nombre", nombre );
        obj_editor.putString("aPaterno", aPaterno);
        obj_editor.putString("aMaterno", aMaterno);
        obj_editor.putString("edad", edad);
```



```
obj_editor.putString("usuario", usuario);
obj_editor.putString("email", email);
obj_editor.putString("password", password);
obj_editor.commit();
Toast.makeText(this, "El registro se guardó exitosamente",
Toast.LENGTH_LONG).show();
}
public void mensaje(View view){
Intent intent = new Intent(this, mostrar.class);
startActivity(intent);
}
}
```

#### mostrar.java

```
package com.example.sharedpreference1;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Context;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
public class mostrar extends AppCompatActivity {
TextView tvNombre, tvApellidoPaterno, tvApellidoMaterno, tvEdad, tvUsuario,
tvEmail, tvPassword;
Button btnRegresar;
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_mostrar);
tvNombre = (TextView)findViewById(R.id.tvNombre);
tvApellidoPaterno = (TextView)findViewById(R.id.tvApellidoPaterno);
tvApellidoMaterno = (TextView)findViewById(R.id.tvApellidoMaterno);
tvEdad = (TextView)findViewById(R.id.tvEdad);
tvUsuario = (TextView)findViewById(R.id.tvUsuario);
tvEmail = (TextView)findViewById(R.id.tvEmail);
tvPassword = (TextView)findViewById(R.id.tvPassword);
btnRegresar = (Button)findViewById(R.id.btnRegresar);
cargarPreferencia();
}
private void cargarPreferencia() {
SharedPreferences preferences = getSharedPreferences("datos",
Context.MODE_PRIVATE);
String nombre = preferences.getString("nombre", "");
String aPaterno = preferences.getString("aPaterno", "");
String aMaterno = preferences.getString("aMaterno", "");
String edad = preferences.getString("edad", "");
String usuario = preferences.getString("usuario", "");
String email = preferences.getString("email", "");
String password = preferences.getString("password", "");
tvNombre.setText(nombre);
tvApellidoPaterno.setText(aPaterno);
```

```
tvApellidoMaterno.setText(aMaterno);
tvEdad.setText(edad);
tvUsuario.setText(usuario);
tvEmail.setText(email);
tvPassword.setText(password);
}
public void regresar(View view){
Intent intent = new Intent(this, MainActivity.class);

startActivity(intent);
}
}
```

activity\_main.xml

```
<?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=".registro">
<TextView
android:text="Registrarse"
android:textSize="40dp"
android:textAlignment="center"
android:layout_width="match_parent"
android:layout_height="wrap_content"/>
<EditText
android:id="@+id/etNombre"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Nombre"
android:inputType="text"
android:textAlignment="center"/>
<EditText
android:id="@+id/etApellidoPaterno"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Apellido Paterno"
android:inputType="text"
android:textAlignment="center"/>
<EditText
android:id="@+id/etApellidoMaterno"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Apellido Materno"
android:inputType="text"
android:textAlignment="center"/>
<EditText
android:id="@+id/etEdad"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Edad"
```

```
android:inputType="number"
android:textAlignment="center"/>
<EditText
android:id="@+id/etUsuario"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Usuario"
android:textAlignment="center"/>
<EditText
```

```
android:id="@+id/etEmail"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="E-Mail"
android:inputType="textEmailAddress"
android:textAlignment="center"/>
<EditText
android:id="@+id/etPassword"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Password"
android:inputType="textPassword"
android:textAlignment="center"/>
<Button
android:id="@+id/btnEnviar"
android:onClick="guardar"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="Registrar"/>
<Button
android:id="@+id/btnMensaje"
android:onClick="mensaje"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="Mensaje"/>
</LinearLayout>
```

```
activity_mostrar.xml
<?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=".mostrar">
<TextView
android:text="Registrarse"
android:textSize="40dp"
android:textAlignment="center"
android:layout_width="match_parent"
android:layout_height="wrap_content"/>
<TextView
android:id="@+id/tvNombre"
```

```
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Nombre"
android:inputType="text"
android:textAlignment="center"/>
<TextView
android:id="@+id/tvApellidoPaterno"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Apellido Paterno"
android:inputType="text"
android:textAlignment="center"/>
<TextView
android:id="@+id/tvApellidoMaterno"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Apellido Materno"
android:inputType="text"
android:textAlignment="center"/>
<TextView
android:id="@+id/tvEdad"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Edad"
android:inputType="number"
android:textAlignment="center"/>
<TextView
android:id="@+id/tvUsuario"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Usuario"
android:textAlignment="center"/>
<TextView
android:id="@+id/tvEmail"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="E-Mail"
android:inputType="textEmailAddress"
android:textAlignment="center"/>
<TextView
android:id="@+id/tvPassword"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Password"
android:inputType="textPassword"
android:textAlignment="center"/>
<Button
android:id="@+id/btnRegresar"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:onClick="regresar"
android:text="Regresar"/>
</LinearLayout>
```

### Practica 3

#### MainActivity.java

```
package com.example.crud;
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Toast;
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        ConexionSQLiteHelper conn = new ConexionSQLiteHelper(this, "bd_usuario",
        null, 1);
    }
    public void Registrar(View v){
        Toast.makeText(this, "ABRIENDO PAGINA...", Toast.LENGTH_SHORT).show();
        Intent insertar = new Intent(getApplicationContext(), insertar.class);
        startActivity(insertar);
    }
    public void Consultar(View v){
        Toast.makeText(this, "ABRIENDO PAGINA...", Toast.LENGTH_SHORT).show();
        Intent consultar = new Intent(getApplicationContext(), consultar.class);
        startActivity(consultar);
    }
    public void Ver(View v){
        Toast.makeText(this, "ABRIENDO PAGINA...", Toast.LENGTH_SHORT).show();
        Intent ver = new Intent(getApplicationContext(), ConsultaListView.class);
        startActivity(ver);
    }
}
```

#### ConexionSQLiteHelper.java

```
package com.example.crud;
import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import androidx.annotation.Nullable;
public class ConexionSQLiteHelper extends SQLiteOpenHelper {
    public ConexionSQLiteHelper(@Nullable Context context, @Nullable String name,
    @Nullable SQLiteDatabase.CursorFactory factory, int version) {
        super(context, name, factory, version);
    }
    @Override
    public void onCreate(SQLiteDatabase db) {
        db.execSQL(utilidades.CREAR_TABLA_USUARIO);
    }
    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        db.execSQL("DROP TABLE IF EXISTS usuario");
        onCreate(db);
    }
}
```

#### usuario.java

```
package com.example.crud;
public class usuario {
    private Integer id;
    private String nombre;
    private String domicilio;
    private String email;
    private String password;
    public usuario(){
        this.id = id;
        this.nombre = nombre;
        this.domicilio = domicilio;
        this.email = email;
        this.password = password;
    }
    public Integer getId(){ return id; }
    public void setId(Integer id){ this.id = id; }
    public String getNombre(){ return nombre; }
    public void setNombre(String nombre){ this.nombre = nombre; }
    public String getDomicilio(){ return domicilio; }
    public void setDomicilio(String domicilio){ this.domicilio = domicilio; }
    public String getEmail(){ return email; }
    public void setEmail(String email){ this.email = email; }
    public String getPassword(){ return password; }
    public void setPassword(String password){ this.password = password; }
}
```

#### ConsultaListView.java

```
package com.example.crud;
import androidx.appcompat.app.AppCompatActivity;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.Toast;
import java.util.ArrayList;
public class ConsultaListView extends AppCompatActivity {
    ListView lvPersonas;
    ArrayList<String> listaInformacion;
    ArrayList<usuario> listaUsuarios;
    ConexionSQLiteHelper conn;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_consulta_list_view);
        Toast.makeText(getApplicationContext(), "VISUALIZACION DE TODOS LOS REGISTROS", Toast.LENGTH_SHORT).show();
        conn = new ConexionSQLiteHelper(this, "bd_usuario", null, 1);
        lvPersonas = (ListView) findViewById(R.id.lvPersonas);
        consultarListaPersonas();
        ArrayAdapter<String> adaptador = new ArrayAdapter<>(this,
```



```

android.R.layout.simple_list_item_1, listaInformacion);
lvPersonas.setAdapter(adaptador);
lvPersonas.setOnItemClickListener(new AdapterView.OnItemClickListener() {
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int
    position, long id) {
        String informacion= "ID: "
        +listaUsuarios.get(position).getId()+"\n";
        informacion+="NOMBRE: "
        +listaUsuarios.get(position).getNombre()+"\n";
        informacion+="DOMICILIO: "
        +listaUsuarios.get(position).getDomicilio()+"\n";
        informacion+="EMAIL: "
        +listaUsuarios.get(position).getEmail()+"\n";
        informacion+="PASSWORD: "
        +listaUsuarios.get(position).getPassword()+"\n";
        Toast.makeText(getApplicationContext(), informacion,
        Toast.LENGTH_SHORT).show();
    }
});
}
public void consultarListaPersonas(){
    SQLiteDatabase db = conn.getReadableDatabase();

    usuario usuario = null;
    listaUsuarios = new ArrayList<usuario>();
    Cursor cursor = db.rawQuery("SELECT * FROM "
    +utilidades.TABLA_USUARIO,null);
    while (cursor.moveToNext()){
        usuario = new usuario();
        usuario.setId(cursor.getInt(0));
        usuario.setNombre(cursor.getString(1));
        usuario.setDomicilio(cursor.getString(2));
        usuario.setEmail(cursor.getString(3));
        usuario.setPassword(cursor.getString(4));
        listaUsuarios.add(usuario);
    }
    obtenerLista();
}
private void obtenerLista(){
    listaInformacion = new ArrayList<String>();
    for (int i = 0; i<listaUsuarios.size(); i++){
        listaInformacion.add(listaUsuarios.get(i).getId()+" "
        +listaUsuarios.get(i).getNombre());
    }
}
}
}

```

#### insertar.java

```

package com.example.crud;
import androidx.appcompat.app.AppCompatActivity;
import android.content.ContentValues;
import android.database.sqlite.SQLiteDatabase;
import android.os.Bundle;
import android.os.PatternMatcher;

```



```
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;
import com.google.android.material.textfield.TextInputLayout;
import java.util.regex.Pattern;
public class insertar extends AppCompatActivity {
    EditText etNombre, etDomicilio, etCorreo, etPassword;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_insertar);
        etNombre = (EditText) findViewById(R.id.etNombre);
        etDomicilio = (EditText) findViewById(R.id.etDomicilio);
        etCorreo = (EditText) findViewById(R.id.etEmail);
        etPassword = (EditText) findViewById(R.id.etPassword);
    }
    public void Guardar(View v){
        final String nombre1 = etNombre.getText().toString();
        final String domicilio1 = etDomicilio.getText().toString();
        final String email1 = etCorreo.getText().toString();
        final String password1 = etPassword.getText().toString();
        if(nombre1.isEmpty() || domicilio1.isEmpty() || email1.isEmpty() ||
        password1.isEmpty()){
            Toast.makeText(this, "No puede dejar campos vacios",
            Toast.LENGTH_SHORT).show();
        }
        else{
            ConexionSQLiteHelper conn = new ConexionSQLiteHelper(this,
            "bd_usuario", null, 1);
            SQLiteDatabase db;
            db = conn.getWritableDatabase();
            ContentValues values = new ContentValues();
            values.put(utilidades.CAMPO_NOMBRE, etNombre.getText().toString());
            values.put(utilidades.CAMPO_DOMICILIO,
            etDomicilio.getText().toString());
            values.put(utilidades.CAMPO_EMAIL, etCorreo.getText().toString());
            values.put(utilidades.CAMPO_PASSWORD,
            etPassword.getText().toString());

            Long idResultante = db.insert(utilidades.TABLA_USUARIO,
            utilidades.CAMPO_NOMBRE, values);
            Toast.makeText(getApplicationContext(), "Numero de Registro: "
            +idResultante, Toast.LENGTH_SHORT).show();
            db.close();
        }
    }
}
```

#### consultar.java

```
package com.example.crud;
import androidx.appcompat.app.AppCompatActivity;
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
```

```
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import android.widget.Toast;
import com.google.android.material.textfield.TextInputEditText;
import com.google.android.material.textfield.TextInputLayout;
public class consultar extends AppCompatActivity {
    EditText etNombre, etDomicilio, etCorreo, etPassword, etID;
    ConexionSQLiteHelper conn = new ConexionSQLiteHelper(this, "bd_usuario", null,
1);
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_consultar);
    etID = (EditText) findViewById(R.id.etID);
    etNombre = (EditText) findViewById(R.id.etNombre);
    etDomicilio = (EditText) findViewById(R.id.etDomicilio);
    etCorreo = (EditText) findViewById(R.id.etEmail);
    etPassword = (EditText) findViewById(R.id.etPassword);
}
public void Buscar(View v){
    SQLiteDatabase db;
    db = conn.getReadableDatabase();
    String [] parametros = {etID.getText().toString()};
    String [] campos =
    {utilidades.CAMPO_NOMBRE, utilidades.CAMPO_DOMICILIO,
    utilidades.CAMPO_EMAIL, utilidades.CAMPO_PASSWORD};
    try{
        Cursor cursor = db.query(utilidades.TABLA_USUARIO, campos,
        utilidades.CAMPO_ID+"=?",
        parametros, null, null, null);
        cursor.moveToFirst();
        etNombre.setText(cursor.getString(0));
        etDomicilio.setText(cursor.getString(1));
        etCorreo.setText(cursor.getString(2));
        etPassword.setText(cursor.getString(3));
        cursor.close();
    }catch (Exception e){
        Toast.makeText(getApplicationContext(), "El ID no existe",
        Toast.LENGTH_SHORT).show();
    }
}
public void Actualizar(View v){

    SQLiteDatabase db;
    db = conn.getWritableDatabase();
    String [] parametros = {etID.getText().toString()};
    ContentValues values = new ContentValues();
    values.put(utilidades.CAMPO_NOMBRE, etNombre.getText().toString());
    values.put(utilidades.CAMPO_DOMICILIO, etDomicilio.getText().toString());
    values.put(utilidades.CAMPO_EMAIL, etCorreo.getText().toString());
    values.put(utilidades.CAMPO_PASSWORD, etPassword.getText().toString());
    db.update(utilidades.TABLA_USUARIO, values, utilidades.CAMPO_ID + "=?",
    parametros);
    Toast.makeText(getApplicationContext(), "Registro Actulizado",
```

```

Toast.LENGTH_SHORT).show();
etID.setText("");
etNombre.setText("");
etDomicilio.setText("");
etCorreo.setText("");
etPassword.setText("");
db.close();
}
public void Eliminar(View v){
    SQLiteDatabase db;
    db = conn.getWritableDatabase();
    String [] parametros = {etID.getText().toString()};
    db.delete(utilidades.TABLA_USUARIO, utilidades.CAMPO_ID+ "=?",
    parametros);
    Toast.makeText(getApplicationContext(), "Registro Borrado",
    Toast.LENGTH_SHORT).show();
    etID.setText("");
    etNombre.setText("");
    etDomicilio.setText("");
    etCorreo.setText("");
    etPassword.setText("");
    db.close();
}
}
utilidades.java
package com.example.crud;
public class utilidades {
    public static final String TABLA_USUARIO = "usuario";
    public static final String CAMPO_ID = "id";
    public static final String CAMPO_NOMBRE = "nombre";
    public static final String CAMPO_DOMICILIO = "domicilio";
    public static final String CAMPO_EMAIL = "email";
    public static final String CAMPO_PASSWORD = "password";
    public static final String CREAR_TABLA_USUARIO = "CREATE TABLE usuario" +
    "("+CAMPO_ID+" INTEGER PRIMARY KEY AUTOINCREMENT,"
    +CAMPO_NOMBRE+" TEXT,"+CAMPO_DOMICILIO+" TEXT,"+CAMPO_EMAIL+"
    TEXT,"+CAMPO_PASSWORD+" TEXT)";
}

```

activity\_main.xml

```

<?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"
    tools:context=".MainActivity"
    android:orientation="vertical">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="35dp"
        android:textStyle="bold"
        android:text="USO DE SQLITE"
    >

```

```
android:gravity="center"/>
<Button
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="Registrar Usuario"
android:layout_marginTop="15dp"
android:onClick="Registrar"/>
<Button
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="Consultar Usuario"
android:layout_marginTop="15dp"
android:onClick="Consultar"/>
<Button
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="Ver usuarios en ListView"
android:layout_marginTop="15dp"
android:onClick="Ver"/>
</LinearLayout>
```

activity\_insertar.java

```
<?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"
tools:context=".insertar"
android:orientation="vertical">
<TextView
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="INSERTAR REGISTRO"
android:textSize="30dp"
android:textStyle="bold"
android:gravity="center"
android:layout_marginTop="10dp"
android:layout_marginBottom="10dp"/>
<EditText
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Nombre Usuario"
android:gravity="center"
android:textSize="20dp"
android:id="@+id/etNombre"/>
<EditText
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Domicilio"
android:gravity="center"
android:textSize="20dp"
android:id="@+id/etDomicilio"/>
<EditText
```

```
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Email"
android:gravity="center"
android:textSize="20dp"
android:id="@+id/etEmail"/>
<EditText
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Password"
android:gravity="center"
android:textSize="20dp"
android:id="@+id/etPassword"/>
<Button
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="GUARDAR"
android:onClick="Guardar"/>
</LinearLayout>
```

activity\_consultar.xml

```
<?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"
tools:context=".consultar"
android:orientation="vertical">
<TextView
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="CONSULTAR REGISTRO"
android:textSize="30dp"
android:textStyle="bold"
android:gravity="center"
android:layout_marginTop="10dp"
android:layout_marginBottom="10dp"/>
<EditText
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="ID Usuario"
android:gravity="center"
android:id="@+id/etID"/>
<Button
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="BUSCAR"
android:onClick="Buscar"/>
<EditText
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Nombre Usuario"
android:gravity="center"
```

```
android:textSize="20dp"
android:id="@+id/etNombre"/>
<EditText
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Domicilio"
android:gravity="center"
android:textSize="20dp"
android:id="@+id/etDomicilio"/>
<EditText
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Email"
android:gravity="center"
android:textSize="20dp"
android:id="@+id/etEmail"/>
<EditText
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Password"
android:gravity="center"
android:textSize="20dp"
android:id="@+id/etPassword"/>

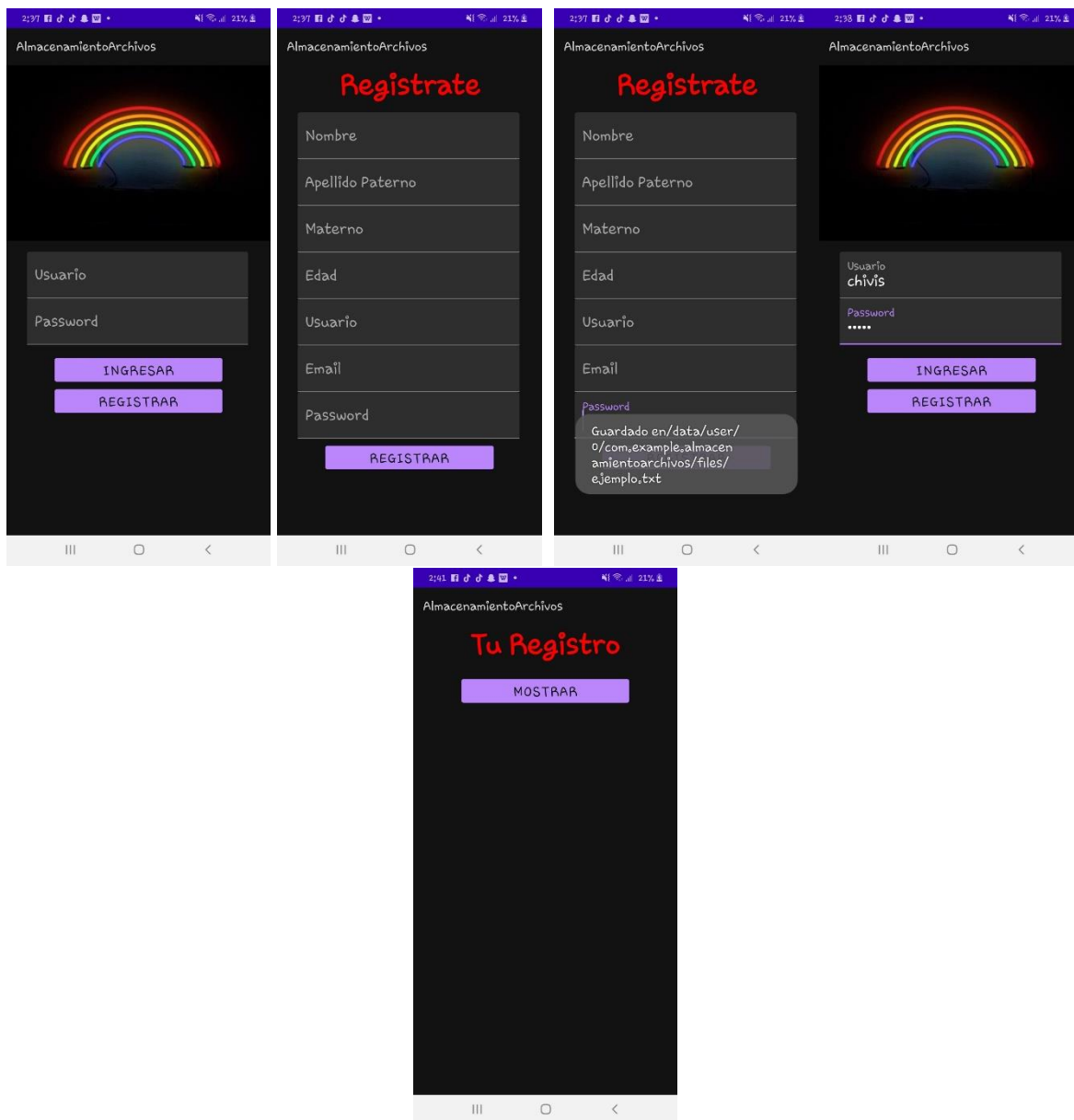
<Button
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="ELIMINAR"
android:onClick="Eliminar"/>
<Button
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="ACTUALIZAR"
android:onClick="Actualizar"/>
</LinearLayout>
```

```
activity_consulta_list_view.xml
<?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"
tools:context=".ConsultaListView"
android:orientation="vertical">
<TextView
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="VER REGISTROS"
android:textSize="30dp"
android:textStyle="bold"
android:gravity="center"
android:layout_marginTop="10dp"
android:layout_marginBottom="10dp"/>
```


```
<ListView  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:id="@+id/lvPersonas"/>  
</LinearLayout>
```

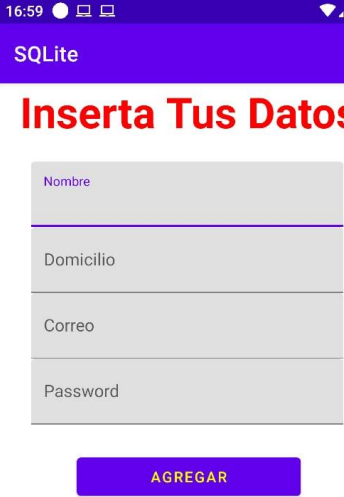


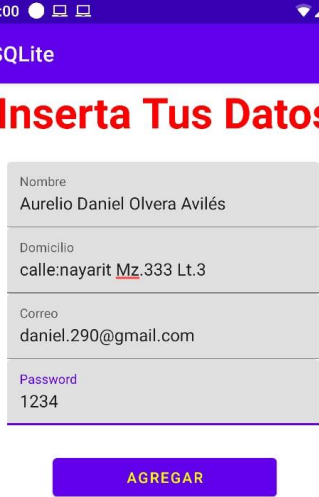
## RESULTADOS



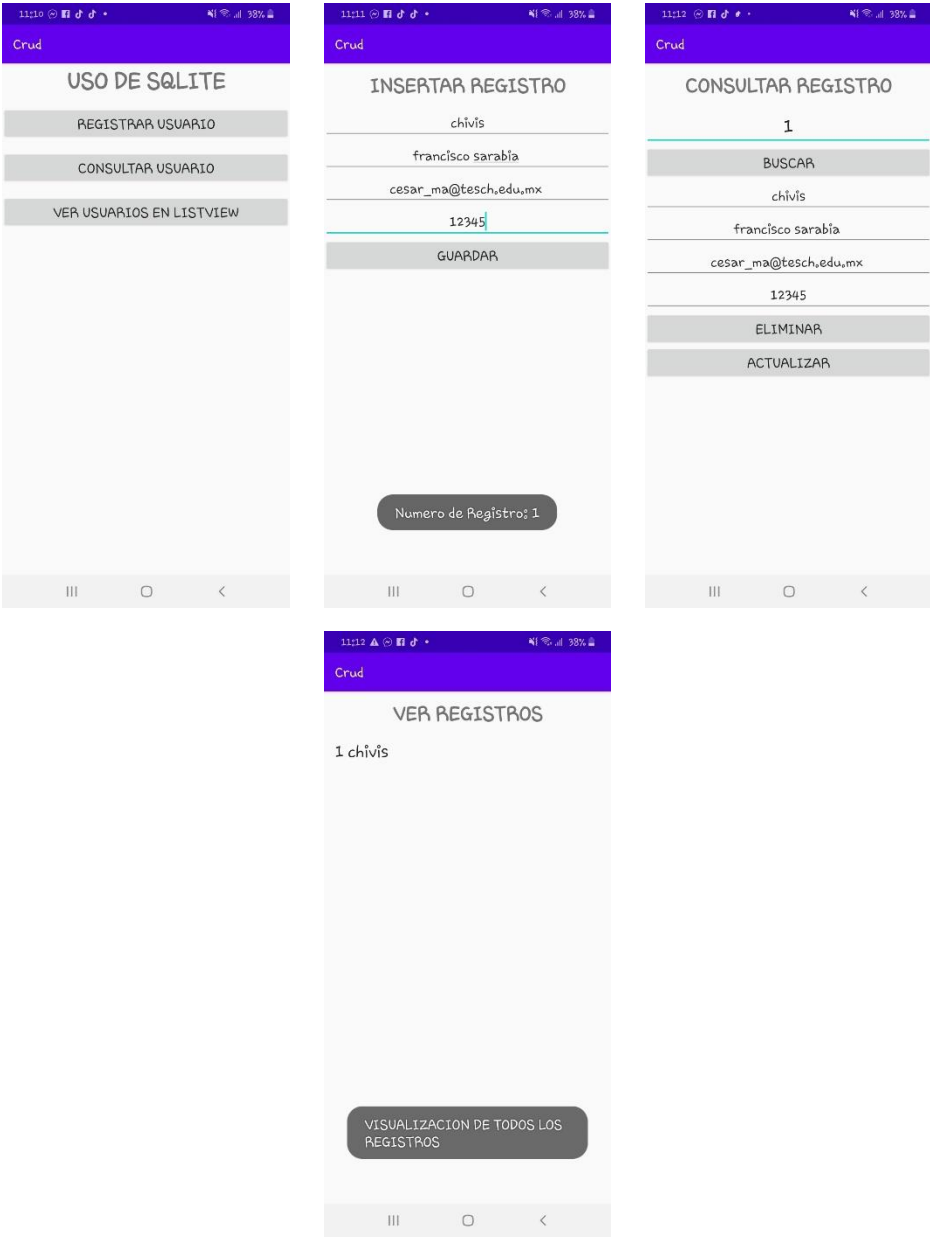
Como se muestra en estas pantallas el resultado de nuestra aplicación 3 la cual nos permitira hacer un registro y ese registro podra almacenarse en un archivo .txt el cual nos servira para poder nosotros almacenar nuestros datos, en la primera pantalla apareceran 2 botones uno de registro y otro de ingresar cuando demos en registrar nos direccionara a la siguiente pantalla en donde colocaremos nuestros datos una vez echo el registro de guardara e nuestro archivo .txt ahora si en la pantalla principal colocamos nuestro usuario y contraseña, en est apartado solo tenemos valido la opcion de admin ya que si ponemos nuestro usuario no nos permitira entrar por eso colocamos admin y la password y nos dirigiremos a la ultima pantalla donde podremos visualizar nuestro registro.







Como se observa en estas pantallas el resultado de nuestra aplicación, como vemos en la primera pantalla que será la principal en donde podremos hacer un registro de un usuario cuando demos guardar automáticamente se guardará el registro y lo almacenará en la siguiente pantalla que nos servirá para consultar nuestro registro.



Como se observa en estas pantallas el resultado de nuestra aplicación 3, como vemos el funcionamiento de SQLite la cual nos servirá para poder guardar en una base de datos los registros de los usuarios, como vemos en nuestra primera pantalla nuestros 3 botones los cuales servirán para hacer un registro, consultar nuestro registró y ver en lista nuestros registros como se ve en las pantallas anteriores.

## CONCLUSION

En la presente documentación se presentaron 3 diferentes practicas las cuales hacen referencia al compartimiento de nuestros archivos, datos y como poder almacenar los datos en la utilización de SQLite.

Todas estas practicas sirven de mucho ya que permiten hacer diferentes consultas de nuestros registros, así como también hacer diferentes tipos de registros y así de esta forma nos permiten tener una mayor seguridad de todos los datos, estas practicas se pueden utilizar bastante ya que muchas de las empresas requieren de algún método de registro y así la forma de SharedPreferences puede servir de mucho ya que tiene la funcionalidad de hacer el registro y la consulta muy rápidamente y fácil.

En la segunda parte de la documentación se realizo la practica de registro de nuestros archivos ya que se utiliza el formulario de datos personales esta aplicación nos permitirá guardar los datos en un archivo .txt en el cual aparecerán todos los registros que se hayan realizado en ese formulario, esta aplicación nos puede permitir para ambos usuarios ya sean administradores o usuarios normales los administradores pueden hacer el registro de admin y con sus respectivos datos.

En la tercera parte de la documentación se muestra el uso de SQLite el cual nos va a permitir hacer un registró de nuestros datos en una base de datos y así estén, asegurados nuestros datos ya que la última aplicación tiene el beneficio de poder consultar y eliminar nuestros datos, esto nos sirve de mucho ya que muchos de los usuarios al momento de hacer el registro se equivocan y así con la opción de eliminar pueden volver hacer el registro.

Para todas estas prácticas si se complicó un poco ya que al momento de hacer el registro la aplicación crasheaba y se cerraba, pero buscando y teniendo las herramientas necesarias se pudo hacer el registro correctamente y las aplicaciones funcionaban y no se cerraban.

## REFERENCIAS

- 1.- Ulises C (2020). *Shared Preferences*. Recuperado de:  
<http://www.saulcintero.com/preferencias-en-android-shared-preferences/>
- 2.- Agoliver (2020). *Preferences Activity*. Recuperado de:  
<https://www.sgoliver.net/blog/preferencias-en-android-ii-preferenceactivity/>
- 3.- Xataka B (2020). *Sistemas de Archivos Internos y Externos*. Recuperado de:  
<https://www.xataka.com/basics/sistemas-de-archivo-como-saber-cual-elegir-al-formatear-tu-disco-duro-o-usb>
- 4.- PYM (2020). *Bases de Datos SQLite*. Recuperado de:  
<https://programacionymas.com/blog/bases-de-datos-sqlite-en-android>