**Laboratorio 9**

**Administración de Bases de Datos en Android Studio**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tipo de trabajo:** | **X** | **Individual** |  | **Grupal** |

**Fecha:** Haga clic aquí o pulse para escribir una fecha.

**Paralelo:** Elija un paralelo.

**Nombre:** Haga clic aquí para escribir texto.

# Objetivo

Gestionar una base de datos en Android Studio para el almacenamiento y procesamiento de la información.

# Duración

Ciento veinte (120) minutos.

# Materiales y Herramientas

* Software Android Studio

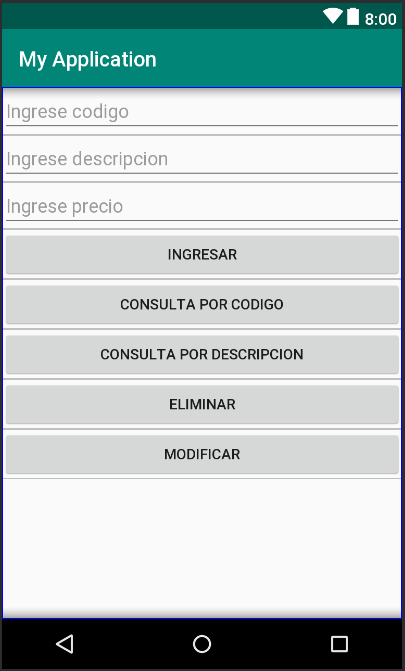
# Escenario

En la práctica de laboratorio utilizará código SQL para definir una base de datos que contendrá tablas, campos, para almacenar registros de artículos de un inventario en general.

# Procedimiento

Paso 1:Crear un proyecto nuevo

1. Iniciar el software Android Studio y crear un nuevo proyecto llamado *laboratorio9TLM.*
2. Diseñar el “Activity” el cual tenga la siguiente apariencia:



1. Crear una clase que herede la clase *“SQLiteOpenHelper”*, dar clic derecho sobre la carpeta que contiene los archivos *“JAVA”* y seleccionar *“New” 🡪 “Java Class”*; en la ventana que aparece poner el nombre de la clase *“AdminSQLiteOpenHelper”* y en el bloque *“Superclass”* poner: *“android.database.sqlite.SQLiteOpenHelper”*
2. Implementar los métodos *“onCreate”, “onUpgrade”* y el constructor de la clase.

|  |  |
| --- | --- |
|  | public class AdminSQLiteOpenHelper extends SQLiteOpenHelper {  public AdminSQLiteOpenHelper(Context context, String name, SQLiteDatabase.CursorFactory factory, int version) {  super(context, name, factory, version);  }  @Override  public void onCreate(SQLiteDatabase db) {  db.execSQL("create table articulos(codigo int primary key,descripcion text,precio real)");  }  @Override  public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {  }  } |

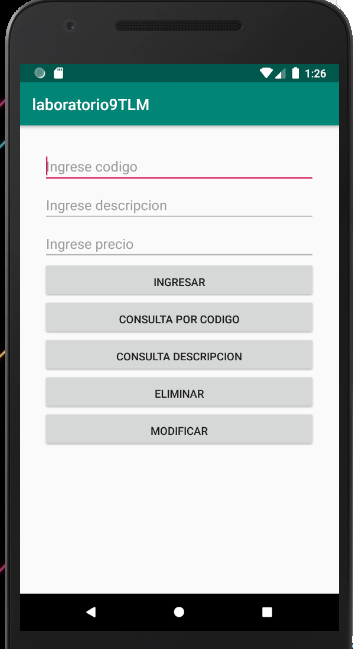
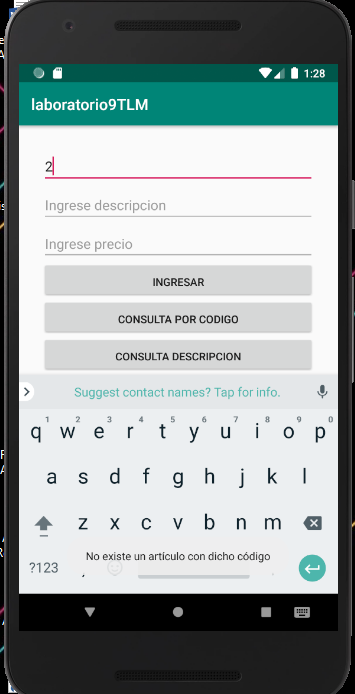
1. Crear 3 atributos en la clase *“MainActivity”* e inicializarlos en el constructor*.*

|  |  |
| --- | --- |
|  | private EditText et1,et2,et3; |
|  | et1=(EditText)findViewById(R.id.et1);  et2=(EditText)findViewById(R.id.et2);  et3=(EditText)findViewById(R.id.et3); |
|  | La declaración de los atributos NO va dentro del constructor, solo la inicialización. |

1. Crear las respectivas funciones de cada botón.

|  |  |
| --- | --- |
|  | public void ingresar(View v) {  AdminSQLiteOpenHelper admin = new AdminSQLiteOpenHelper(this,  "administracion", null, 1);  SQLiteDatabase bd = admin.getWritableDatabase();  String cod = et1.getText().toString();  String descri = et2.getText().toString();  String pre = et3.getText().toString();  bd.execSQL("insert into articulos (codigo,descripcion,precio) values ("+cod+",'"+descri+"',"+pre+")");  bd.close();  et1.setText("");  et2.setText("");  et3.setText("");  Toast.makeText(this, "Se cargaron los datos del artículo",  Toast.LENGTH\_SHORT).show();  }  public void consultaporcodigo(View v) {  AdminSQLiteOpenHelper admin = new AdminSQLiteOpenHelper(this,  "administracion", null, 1);  SQLiteDatabase bd = admin.getReadableDatabase();  String cod = et1.getText().toString();  Cursor fila = bd.rawQuery(  "select descripcion,precio from articulos where codigo=" + cod, null);  if (fila.moveToFirst()) {  et2.setText(fila.getString(0));  et3.setText(fila.getString(1));  } else  Toast.makeText(this, "No existe un artículo con dicho código",  Toast.LENGTH\_SHORT).show();  bd.close();  }  public void consultapordescripcion(View v) {  AdminSQLiteOpenHelper admin = new AdminSQLiteOpenHelper(this,  "administracion", null, 1);  SQLiteDatabase bd = admin.getReadableDatabase();  String descri = et2.getText().toString();  Cursor fila = bd.rawQuery(  "select codigo,precio from articulos where descripcion like '" + descri +"'", null);  if (fila.moveToFirst()) {  et1.setText(fila.getString(0));  et3.setText(fila.getString(1));  } else  Toast.makeText(this, "No existe un artículo con dicha descripción",  Toast.LENGTH\_SHORT).show();  bd.close();  }  public void eliminar(View v) {  AdminSQLiteOpenHelper admin = new AdminSQLiteOpenHelper(this,  "administracion", null, 1);  SQLiteDatabase bd = admin.getWritableDatabase();  String cod= et1.getText().toString();  bd.execSQL("delete from articulos where codigo = "+cod);  bd.close();  et1.setText("");  et2.setText("");  et3.setText("");  Toast.makeText(this, "Se borró el artículo con dicho código",  Toast.LENGTH\_SHORT).show();  }  public void modificacion(View v) {  AdminSQLiteOpenHelper admin = new AdminSQLiteOpenHelper(this,  "administracion", null, 1);  SQLiteDatabase bd = admin.getWritableDatabase();  String cod = et1.getText().toString();  String descri = et2.getText().toString();  String pre = et3.getText().toString();  bd.execSQL("update articulos set codigo="+cod+",descripcion='"+descri+"',precio="+pre+" where codigo="+cod);  bd.close();  Toast.makeText(this, "se modificaron los datos", Toast.LENGTH\_SHORT)  .show();  } |

Actividad 1:Presente capturas del dispositivo virtual o real ejecutando la aplicación en el Paso 1.

Pregunta 1:¿Para qué se utiliza la clase Toast en la práctica e indique que limitaciones encuentra?

Muestra un mensaje de corta duración cuando se ha ingresado un artículo, cuando no existe un artículo buscado según el código o descripción, en caso de querer eliminar o modificar la información de uno de ellos. Una de las limitaciones que se puede observar es que son mensajes que muestran una información deseada o estado de la aplicación, pero no muestran errores de ejecución y son casi imperceptibles a la vista cuando aparecen.

Pregunta 2:Explique la diferencia entre la función *“rawQuery”* y la función *“execSQL”*

rawQuery se utiliza para consultar la base de datos con una instrucción SELECT y devuelve un conjunto de filas y columnas en un Cursor String[ ]. Por otro lado, execSQL presenta instrucciones en la base de datos como INSERT, UPDATE o DELETE cuando no se está interesado en la cantidad de filas modificadas o el id de la fila del ultimo inserto.

Pregunta 3:¿Qué es un *“Primary key”* y un *“Foreign Key”*?

Primary Key identifica de forma única una fila o registro de una tabla, por lo que sus valores no se pueden repetir entre filas. Foreign Key es un campo de X tabla que se relaciona entre si con otra tabla Y cuyo campo es una Primary Key.

**RETO (25 ptos):** Realice una base de datos similar al de la práctica, pero en PhpMyAdmin y genere una app para poder conectarse con la misma.

# Conclusiones y Recomendaciones

**Conclusiones:**

* Se logro alojar