

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

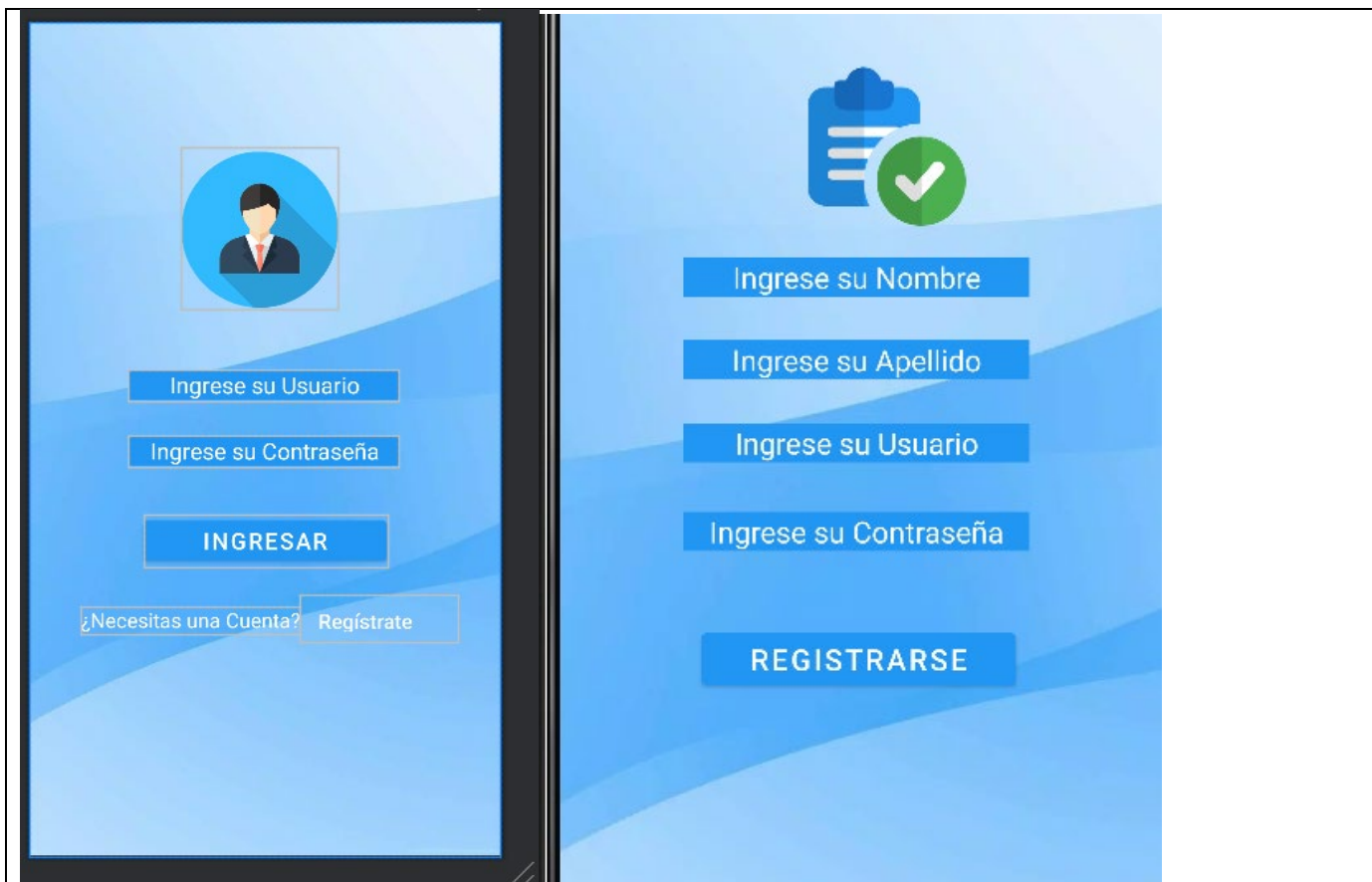
		<h1>EXAMEN INTERCICLO</h1>	
CARRERA: Ingeniería en ciencias de la computación		ASIGNATURA: Plataformas Móviles	
NRO. PRÁCTICA:		TÍTULO PRÁCTICA: EXAMEN INTERCICLO	
OBJETIVO ALCANZADO: <ul style="list-style-type: none"> - Se logró cumplir con todos los requerimientos establecidos en el examen planteado. - Se logró aprender el funcionamiento de los CustomDialog en kotlin - Se logró entender de mejor manera el consumo de APIS - Se logró implementar el envío de la información a las redes sociales 			
ACTIVIDADES DESARROLLADAS			
<p>Con base a la práctica Desarrollo de una app de búsqueda de películas utilizando lenguaje de programación nativo se pide agregar nuevas funcionalidades que permitan la gestión de usuarios, películas favoritas y reseñas en la aplicación. Para lo cual, la información antes mencionada deberá ser persistida en la base de datos interna del dispositivo Android (SQLite).</p> <p>Gestión de usuarios</p> <p>Si no ha iniciado sesión en la aplicación, no podrá realizar búsquedas de películas. Entonces, tendrá la posibilidad de,</p> <ul style="list-style-type: none"> • Registrarse como nuevos usuarios en la aplicación (<i>nueva actividad</i>) • Iniciar sesión como usuario registrado en la aplicación (<i>nueva actividad</i>) • Cerrar sesión de usuario en la aplicación (<i>en el menú de opciones</i>) <p>Para resolver el planteamiento de los usuarios y bueno de toda la aplicación se ha trabajo por medio de la creación de clases junto con sus respectivos atributos.</p>			

```
class Usuario {  
    var id: Int = 0  
    var nombre: String = ""  
    var apellido: String = ""  
    var nickname: String = ""  
    var password: String = ""  
  
    constructor(){  
        this.id = 0;  
        this.nombre = ""  
        this.apellido = ""  
        this.nickname = ""  
        this.password = ""  
    }  
}
```

```
class ReseniaModel {  
    var id: Int = 0  
    var peliculaId: Int = 0  
    var resenia: String = ""  
    var usuarioId: Int = 0  
    var imdbID: String = ""  
  
    constructor() {  
        this.id = 0  
        this.peliculaId = 0  
        this.resenia = ""  
        this.usuarioId = 0  
        this.imdbID = ""  
    }  
  
    constructor(id: Int, peliculaId: Int, resenia: String, usuarioId: Int, imdbID: String) {  
        this.id = id  
        this.peliculaId = peliculaId  
        this.resenia = resenia  
        this.usuarioId = usuarioId  
        this.imdbID = imdbID  
    }  
  
    constructor(peliculaId: Int, resenia: String, usuarioId: Int, imdbID: String) {  
        this.peliculaId = peliculaId  
        this.resenia = resenia  
        this.usuarioId = usuarioId  
        this.imdbID = imdbID  
    }  
}
```

```
class Pelicula {  
    var id: Int = 0  
    var valoracion: String = ""  
    var usuarioId: Int = 0  
    var imdbID: String = ""  
  
    constructor(){  
        this.id = 0;  
        this.valoracion = ""  
        this.usuarioId = 0;  
        this.imdbID = ""  
    }  
  
    constructor(id: Int, valoracion: String, usuarioId: Int, imdbID: String) {  
        this.id = id  
        this.valoracion = valoracion  
        this.usuarioId = usuarioId  
        this.imdbID = imdbID  
    }  
  
    constructor(valoracion: String, usuarioId: Int, imdbID: String) {  
        this.valoracion = valoracion  
        this.usuarioId = usuarioId  
        this.imdbID = imdbID  
    }  
}
```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		



Bueno como se puede ver en las imagenes tenemos nuestro diseño para la página inicial donde el usuario podrá iniciar sesión o registrarse, para ello y haciendo uso de SQLite pues hemos creado las siguientes tablas.

```
class DataBase(context: Context) :
    SQLiteOpenHelper(context, name: "BDpeliculas", factory: null, version: 1) {

    override fun onCreate(db: SQLiteDatabase) {
        val CREATE_TABLE = "CREATE TABLE usuario" +
            "(id INTEGER PRIMARY KEY AUTOINCREMENT, " +
            "nombre TEXT, apellido TEXT, nickname TEXT UNIQUE, password TEXT)"

        val CREAR_TABLA_PELICULA = "CREATE TABLE pelicula" +
            "(id_Pelicula INTEGER PRIMARY KEY AUTOINCREMENT, " +
            "valoracion TEXT, imdbID TEXT, id_Pelicula_Usuario INTEGER, FOREIGN KEY(id_Pelicula_Usuario) REFERENCES usuario(usuario_id))"

        val CREAR_TABLA_RESEÑIA = "CREATE TABLE resenia" +
            "(id_Resenia INTEGER PRIMARY KEY AUTOINCREMENT, " +
            "resenia TEXT, imdbID TEXT, id_Pelicula_Fav_Usuario INTEGER, id_Usuario_Resenia INTEGER, FOREIGN KEY(id_Pelicula_Fav_Usuario) REFEREN" +
            "FOREIGN KEY(id_Usuario_Resenia) REFERENCES usuario(usuario_id))"

        db.execSQL(CREATE_TABLE)
        db.execSQL(CREAR_TABLA_PELICULA)
        db.execSQL(CREAR_TABLA_RESEÑIA)
    }
}
```


Y para poder crear un usuario hemos creado el siguiente método

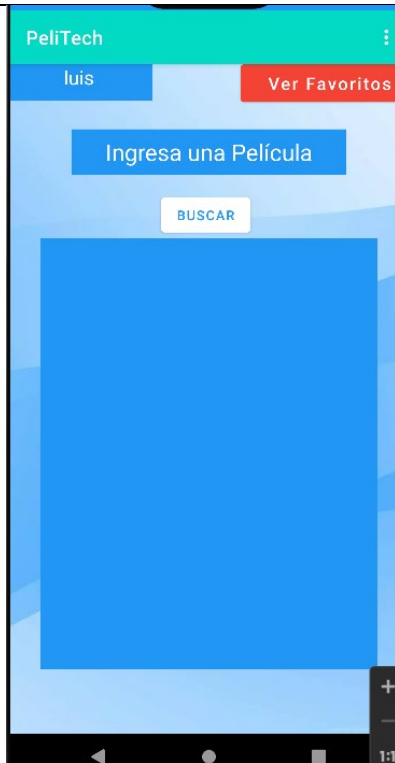
```
fun addUsuario(usuario: Usuario): Boolean {  
    val db = this.writableDatabase  
    val values = ContentValues()  
    values.put("nombre", usuario.nombre)  
    values.put("apellido", usuario.apellido)  
    values.put("nickname", usuario.nickname)  
    values.put("password", usuario.password)  
    val _success = db.insert( table: "usuario", nullColumnHack: null, values)  
    db.close()  
    return (Integer.parseInt( s: "$_success") != -1)  
}
```

Entonces ya en mi vista lo que se ha hecho es llamar a una instancia de la base de datos para poder persistir el objeto Usuario.

```
btnInicio.setOnClickListener(View.OnClickListener { it: View? -> {  
    if (txtUsuario.getText().toString() != "" && txtPassword.getText().toString() != "") {  
        val usuarioLogin = DataBase.getUsuarioEnLogin(txtUsuario.getText().toString(), txtPassword.getText().toString())  
        val pruebaotro = DataBase.getUsuarioEnLogin2(txtUsuario.getText().toString(), txtPassword.getText().toString())  
        if (usuarioLogin.nombre != "false" && pruebaotro.nombre != "false") {  
            val toast = Toast.makeText(  
                applicationContext,  
                text: "Iniciando Sesión",  
                Toast.LENGTH_SHORT  
            )  
            toast.show()  
            val intent = Intent( packageContext: this, Inicio::class.java).apply { this: Intent  
                putExtra( name: "idUser", usuarioLogin.id)  
            }  
            startActivity(intent)  
        } else {  
            val toast = Toast.makeText(  
                applicationContext,  
                text: "Datos incorrectos!!",  
                Toast.LENGTH_SHORT  
            )  
            toast.show()  
        }  
    } else {  
        val toast = Toast.makeText(  
            applicationContext,  
            text: "Por favor ingrese datos en los campos",  
            Toast.LENGTH_SHORT  
        )  
        toast.show()  
    }  
})
```

Y una vez validados todos los datos pues y haciendo de los Toast se pasara a la siguiente pagina donde se podrá hacer la búsqueda de las películas.

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		




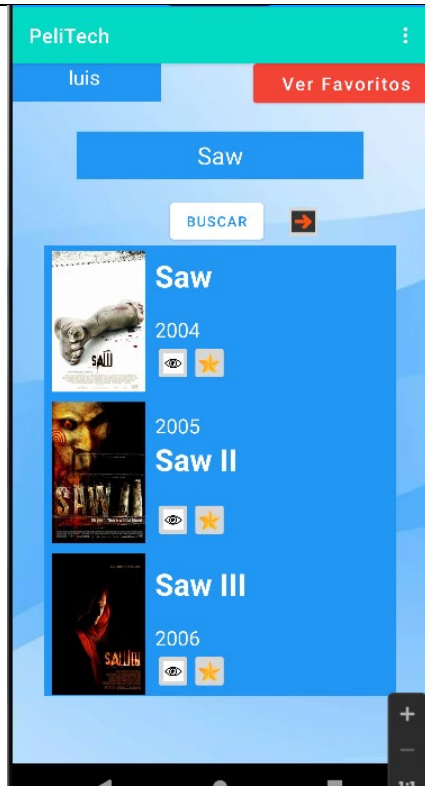
Entonces como se podrá observar existe un campo donde yo puedo ingresar el nombre de la película de la cual quiero obtener información, además se ha puesto en las esquinas superiores un TextView y un Button en los cuales estará el nombre del usuario y el botón para ver las películas favoritas del mismo.

```

val jsonObjectRequest = JsonObjectRequest(Request.Method.GET, url,
    jsonRequest: null,
    { response ->
        if (response.getString( name: "Response") == "True") {
            btnDerecha.setVisibility(View.VISIBLE);
            Log.i( tag: "Data", response.toString())
            val movies = response.getJSONArray( name: "Search")
            val movie = movies.optJSONObject( index: 0)
            if (movie != null) {
                DownloadImageFromInternet(findViewById(R.id.imgPoster)).execute(
                    movie.getString(
                        name: "Poster"
                    )
                )
                txtResultado.text = movie.getString( name: "Title")
                txtAnio.text = movie.getString( name: "Year")
                btnMore.setVisibility(View.VISIBLE);
                btnFav.setVisibility(View.VISIBLE);
                btnMore.setOnClickListener(View.OnClickListener { it:View?
                    val intent = Intent( packageContext: this, Detalle::class.java);
                    var extras: Bundle? = Bundle()
                    if (extras != null) {
                        extras.putString("ID", movie.getString( name: "imdbID"))
                        extras.putInt("idUser", usuarioInicio.id)
                        intent.putExtras(extras)
                    };
                    startActivity(intent)
                })
                btnFav.setOnClickListener(View.OnClickListener { it:View?

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		



Como se podrá observar pues una vez dado click en el botón buscar y haciendo el consumo de la API se me logran listar las respectivas películas sobre las cuales yo voy a poder o bien ver a detalle cada una o agregarla a mi lista de películas favoritas.

Para ello hemos hecho uso de lo que serian los eventos sobre estos botones, a continuación, se adjunta capturas de cómo están configurados algunos de ellos.


```

    )
    )
    txtResultado.text = movie.getString( name: "Title")
    txtAño.text = movie.getString( name: "Year")
    btmMore.setVisibility(View.VISIBLE);
    btmFav.setVisibility(View.VISIBLE);
    btmMore.setOnClickListener(View.OnClickListener { it:View!
        val intent = Intent( packageContext, this, Detalle::class.java);
        var extras: Bundle? = Bundle()
        if (extras != null) {
            extras.putString("ID", movie.getString( name: "imdbID"))
            extras.putInt("idUser", usuarioInicio.id)
            intent.putExtras(extras)
        };
        startActivity(intent)
    })
    btmFav.setOnClickListener(View.OnClickListener { it:View!
        var dialog = CustomDialogFragment()
        val data = Bundle()
        data.putString("bandera", "1")
        data.putString("pelicula", txtPelicula.text.toString())
        data.putInt("usuario", usuarioInicio.id)
        data.putString("imdbID", movie.getString( name: "imdbID"))
        dialog.setArguments(data);
        dialog.show(supportFragmentManager, tag: "customDialog")
    })

```

Como se podrá observar pues dentro de cada evento onclick de mi botón pues se mandan parámetros para las siguientes actividades para poder persistir y asociar las diferentes tablas de mi base de datos.


2.

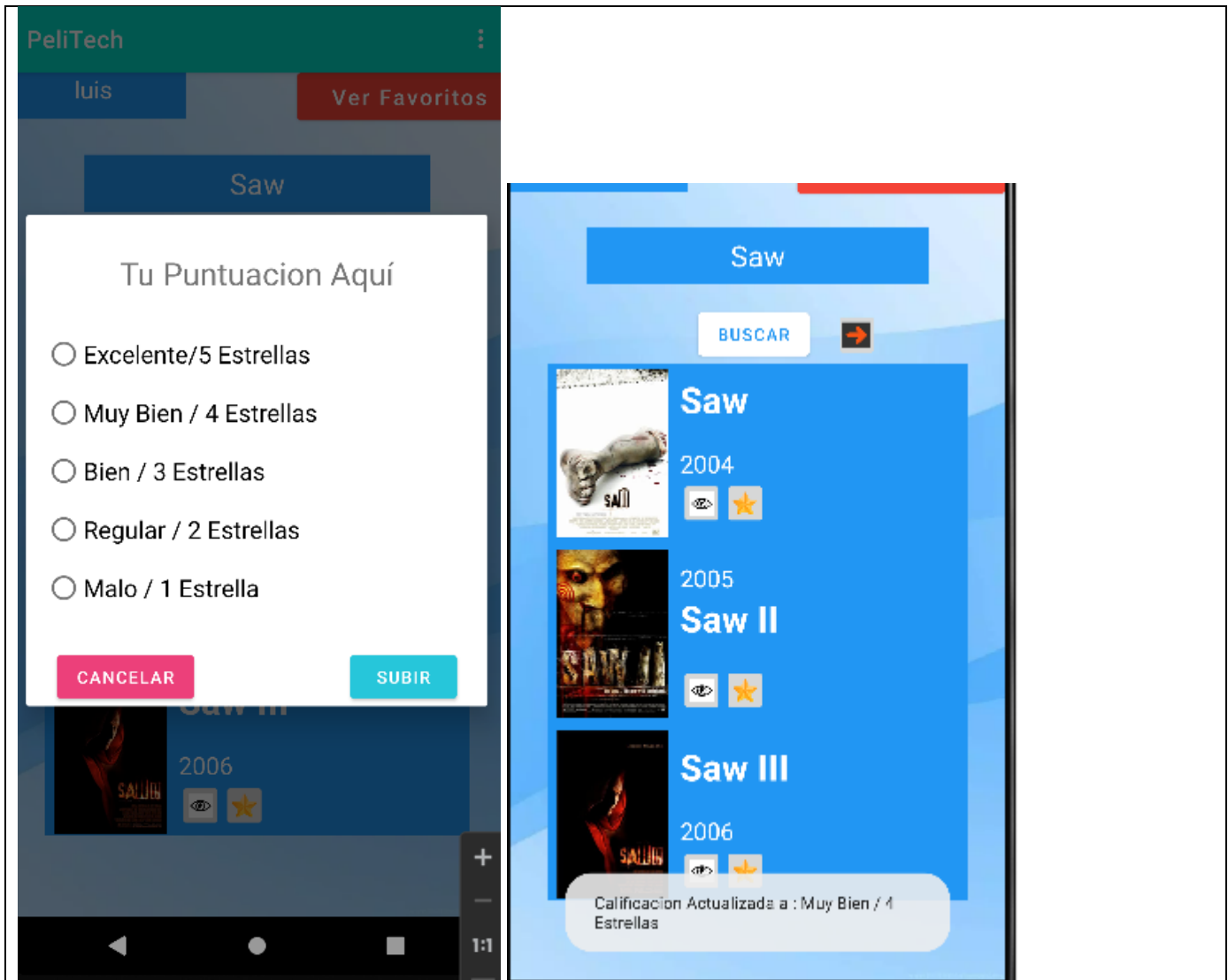
Gestión de películas favoritas

Luego, de que un usuario ha iniciado sesión en la aplicación y puede ingresar al buscador de películas tendrá la posibilidad de,

- Seleccionar de películas favoritas al momento de realizar búsquedas en la API (en actividad de búsqueda de películas existente)
- Asignar valoración de rating de la película favorita seleccionada. Puede ser un valor entre 1 a 5 estrellas. (en actividad de búsqueda de películas existente)
- Visualizar películas favoritas (nueva actividad)

Para la sección de lo que sería las películas favoritas, pues una vez hecho click en el botón con mi estrella se me abrirá un customDialog, el cual obtendrá los parámetros enviados anteriormente de mi página de búsqueda, y por medio de varios RadioButton se podrá escoger la valoración para cada una de las películas como se observa en la imagen

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		



Como se podrá observar se ha seleccionado la calificación de 4 estrellas y por medio del uso de los Toast pues se notifica al usuario


```

rootView.findViewById<View>(R.id.BtnSubir).setOnClickListener{ it: View!
    val data = arguments
    var usuario :Int = -1
    var pelicula = ""
    var bandera = ""
    var imdbID = ""
    var seleccionado :RadioGroup = rootView.findViewById(R.id.ratingRadioGroup)
    val selectedId =seleccionado.checkedRadioButtonId
    val radio = rootView.findViewById<RadioButton>(selectedId)
    val idx = seleccionado.indexOfChild(radio)
    val texto = radio.getText().toString()
    if (data != null) {
        usuario = data.getInt( key: "usuario")
        pelicula = data.getString( key: "pelicula").toString()
        bandera = data.getString( key: "bandera").toString()
        imdbID = data.getString( key: "imdbID").toString()
    }
    val p = Pelicula(texto,usuario,imdbID)
    val buscarPeliculaBandera = DataBase.buscarPelicula(p)
    println("retorno " +buscarPeliculaBandera)
    if (buscarPeliculaBandera) {
        DataBase.addPelicula(p)
        Toast.makeText(context, text: "Califcaste: $texto", Toast.LENGTH_LONG).show()
    } else {
        DataBase.actualizarPelicula(p)
        Toast.makeText(context, text: "Calificacion Actualizada a : $texto", Toast.LENGTH_LONG).show()
    }
    val updateIntent = Intent(activity, Inicio::class.java)
    var extras: Bundle? = Bundle()
    if (extras != null) {

```

```

        DataBase.actualizarPelicula(p)
        Toast.makeText(context, text: "Calificacion Actualizada a : $texto", Toast.LENGTH_LONG).show()
    }
    val updateIntent = Intent(activity, Inicio::class.java)
    var extras: Bundle? = Bundle()
    if (extras != null) {
        extras.putInt("idUser", usuario)
        extras.putString("pelicula",pelicula.toString())
        extras.putString("bandera","2")
        updateIntent.putExtras(extras)
    };

    startActivity(updateIntent)
}

return rootView
dismiss()
}

```

Sumado a esto se ha hecho la respectiva validación para que al momento de que si se ingresa una reseña el mismo usuario y de la misma película pues que está simplemente se actualice.

Continuando pues ya dando click en mi botón ver favoritos podré observar mi lista de películas favoritas



```
btnAdd.setOnClickListener(View.OnClickListener { it: View!
    val intent = Intent( packageContext, this::AddResenia::class.java);
    var extras: Bundle? = Bundle()
    if (extras != null) {
        extras.putString("ID", lista[i].imdbID)
        extras.putInt("idUser", usuarioInicio.id)
        extras.putInt("idPelículaFav", lista[i].id)
        intent.putExtras(extras)
    }

    startActivity(intent)
})

btnDelete.setOnClickListener(View.OnClickListener { it: View!
    println("ID " + lista[i].id + " Id película u " + lista[i].imdbID)


    val resenia = dataBase.buscarReseniaPorIdPelícula(lista[i].id, lista[i].usuarioId)
    println(resenia.id.toString() + " " + resenia.resenia + " " + resenia.películaId)
    dataBase.eliminarResenia(resenia.id)
    dataBase.eliminarPelícula(lista[i].id)
    finish();
    startActivity(getIntent());
    Toast.makeText(applicationContext, text: "Película Eliminada de Favoritos", Toast.LENGTH_LONG).show()
})
```

Una vez listado lo que sería mi película favorita pues tengo dos botones con los cuales o bien puedo eliminar a la película de mi lista de favoritos o añadir una reseña de dicha película, para ello de igual forma hemos ido creando varios métodos para hacer las respectivas consultas a mi base de datos.

```
fun eliminarPelícula(idPelícula: Int): Boolean {
    val db = this.writableDatabase
    val idPelicular = idPelícula.toString()
    val args = arrayOf(idPelicular)
    val _success = db.delete( table: "película", whereClause: "id_Película=?", args)
    db.close()
    return Integer.parseInt( s: "$_success") != -1
    println("Eliminar")
}

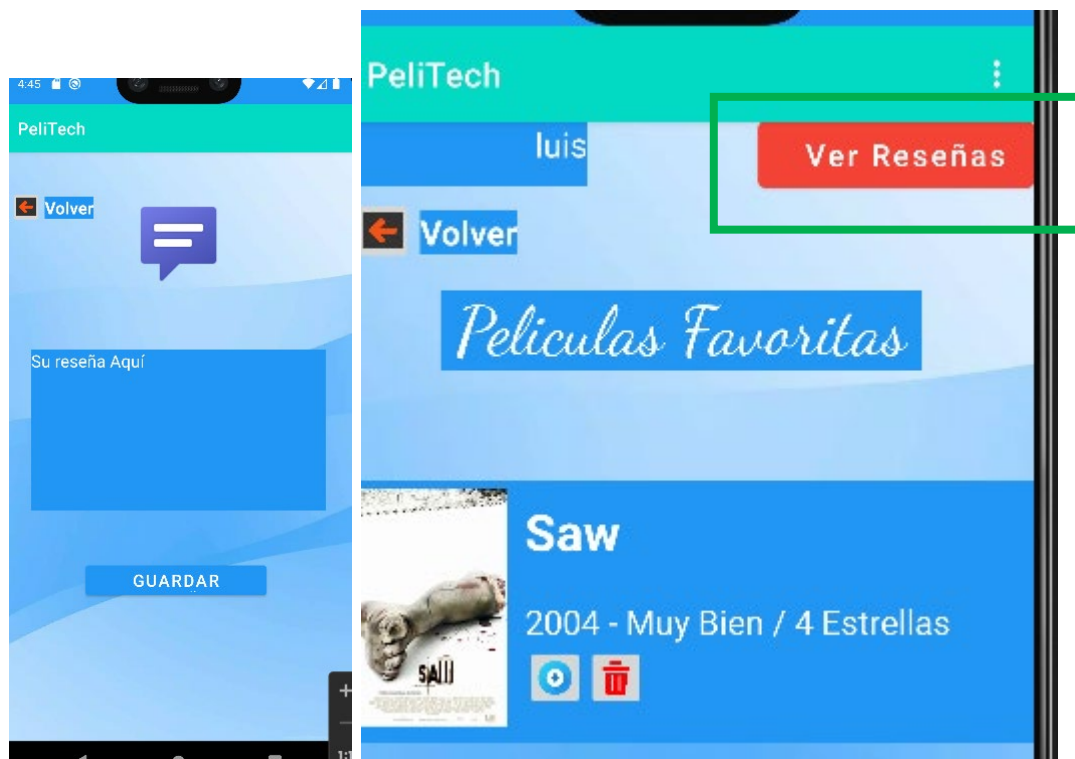
fun eliminarResenia(idResenia: Int): Boolean {
    val db = this.writableDatabase
    val idPelicular = idResenia.toString()
    val args = arrayOf(idPelicular)
    val _success = db.delete( table: "resenia", whereClause: "id_Resenia=?", args)
    db.close()
    println("Se elimino la resenia")
    return Integer.parseInt( s: "$_success") != -1
}

}
```


	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

Para esta parte y continuando con lo anteriormente mencionado pues ya dando click en el botón + puedo añadir la reseña de la respectiva película en otra actividad como se había solicitado

```
btnCrearResenia.setOnClickListener(View.OnClickListener { it: View!
    if (txtResenia.getText().toString() != "") {
        val r = ReseniaModel(idPeliculaFav,txtResenia.getText().toString(),idUsuario,imdbPelicula.toString())
        val buscarReseniaBandera = DataBase.buscarResenia(r)
        if (buscarReseniaBandera){
            DataBase.addResenia(r)
            Toast.makeText(applicationContext, text: "Reseña Creada", Toast.LENGTH_SHORT).show()
            onBackPressed()
        } else {
            DataBase.actualizarResenia(r)
            Toast.makeText(applicationContext, text: "Reseña Actualizada", Toast.LENGTH_SHORT).show()
            onBackPressed()
        }
    } else {
        val toast = Toast.makeText(
            applicationContext,
            text: "Campos Vacios!! Reseña no Creada",
            Toast.LENGTH_SHORT
        )
        toast.show()
    }
})
```



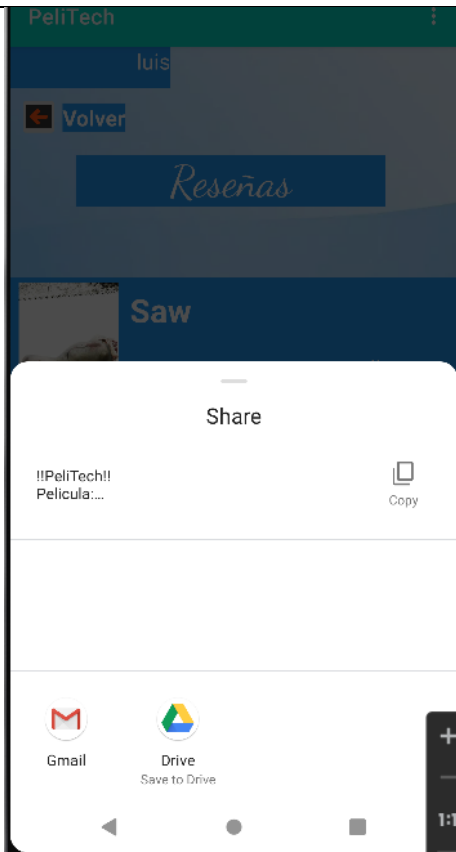
Entonces como se podrá observar tengo un MultilineText para poder escribir n líneas para la reseña y guardarla y para poder ver las mismas se dará click en el botón Ver Reseñas

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		



Como se podrá observar nos ha listado tanto la valoración que le hemos dado a la película, así como la reseña nuestra.

Finalmente, también tenemos la opción de esta reseña poderla compartir o eliminarla.




```


btnShare.setVisibility(View.VISIBLE);
btnDelete.setVisibility(View.VISIBLE);
btnShare.setOnClickListener(View.OnClickListener { it: View?
    val intent = Intent().apply { this: Intent
        action = Intent.ACTION_SEND
        putExtra(
            Intent.EXTRA_TEXT,
            value: "!!PeliTech!!" + "\n"
                + "Película: " + "\n" + "Titulo: " + titulo + "\n" + "Año: " + Anio + "\n" +
                "Valoracion: " + valoracion.valoracion + "\n" + "Reseña: " + resenia + "\n" + "Portada:
        )
        type = "text/plain"
    }
    val share = Intent.createChooser(intent, title: null)
    startActivity(share)
})
btnDelete.setOnClickListener(View.OnClickListener { it: View?
    println("ID " + lista[i].id + "Id película u " + lista[i].imdbID)
    database.eliminarResenia(lista[i].id)

    finish();
    startActivity(getIntent());
    Toast.makeText(
        applicationContext,
        text: "Reseña Eliminada con Exito",
        Toast.LENGTH_LONG
    ).show()
})
}

```

	VICERRECTORADO DOCENTE	Código: GUIA-PRL-001
	CONSEJO ACADÉMICO	Aprobación: 2016/04/06
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		

luis


Volver

Reseñas

Reseña Eliminada con Exito

!!PeliTech!!

Pelicula:


Título: Saw

Año: 2004

Valoracion: Muy Bien / 4 Estrellas

Reseña: Especial

Portada: https://m.media-amazon.com/images/M/MV5BM2M1MzI1MWYtYmM2ZC00OWY3LTk0ZGMtNmRkNzU1NzEzMWE5XkEyXkFqcGdeQXVyODUwOTkwODk@_V1_SX300.jpg




```

})
btnDelete.setOnClickListener(View.OnClickListener { it: View!
    println("ID " + lista[i].id + "Id película u " + lista[i].imdbID)
    DataBase.eliminarResenia(lista[i].id)

    finish();
    startActivity(getIntent());
    Toast.makeText(
        applicationContext,
        text: "Reseña Eliminada con Exito",
        Toast.LENGTH_LONG
    ).show()
})
layout.addView(view)

```

RESULTADO(S) OBTENIDO(S):

- Como resultados pues podemos decir que hemos logrado hacer una aplicación móvil que por medio del consumo de una API permite obtener la información de una película, además lo que se ha implementado es base de datos y sesiones de usuario a fin de que cada usuario pueda hacer reseñas con respecto a su película favorita
- Se logró entender de manera mas clara cual es la dinámica y funcionamiento del consumo de servicios y como se los puede implementar con el lenguaje de programación Kotlin

CONCLUSIONES:

En conclusión, podemos decir que el hacer este tipo de aplicaciones nos ayuda a obtener un conocimiento mucho más amplio de la arquitectura o funcionamiento de un app móvil.

También podemos decir que lo que sería la persistencia de datos es algo que es esencial al momento de realizar aplicaciones móviles pues permite guardar los datos del usuario

RECOMENDACIONES:

- Se recomienda ver la documentación para el uso de los diferentes widgets que nos proporciona Android studio pues los mismos nos facilitan mucho en el proceso de desarrollo de una app móvil
- Aprender bien los fundamentos de programación orientada a objetos

Nombre de estudiante: Luis Adrián Cabrera



Firma de estudiante: