Universidad Continental

Desarrollo de Aplicaciones Móviles

Guía de Trabajo

Universidad Continental
Material publicado con fines de estudio
Código: ASUC01584

Presentación

Taller de Proyectos I, es una asignatura que permite al estudiante diseñar e implementar aplicaciones para plataformas móviles.

Por ese motivo los contenidos diseñados se enfocan en el uso de un Entorno de desarrollo como Android Studio, Views, Layouts, NavDreawers y otros elementos integradores, además de emplear persistencia de datos tanto locales como remotas, servicios Web y sensores, orientado finalmente a que las apps desarrolladas puedan integrar soluciones empresariales multiplataforma.

Al finalizar la asignatura el estudiante será capaz de diseñar, construir y probar aplicaciones móviles con criterios de calidad y eficiencia del producto software entregado, empleando metodologías y herramientas pertinente.

Dada la naturaleza de la asignatura es importante que preveas la instalación previa de Android Studio y al menos un dispositivo virtual,

Pedro Yuri Marquez Solis

Primera unidad

Semana 1 – Sesión 2

Elementos de un proyecto de Android Studio

Sección: ……..……..	Apellidos : ……..……
Docente : Pedro Yuri Marquez Solis	Nombres : ……
Unidad : Unidad 1	Fecha: …../……/……. Duración: 60 min

I. Propósito: Identificar cada uno de los elementos de un proyecto de Android Studio.

II. Descripción de la actividad a realizar (caso)

Una app de Android Studio está compuesta por los elementos que veremos a continuación, por lo que es importante que identifiques cada uno de ellos.

• Servicios. Los servicios son componentes que se ejecutan en el background de la aplicación, ya sea actualizando fuentes de información, atendiendo a diversos eventos, o activando la visualización de notificaciones en una actividad. Se utilizan para llevar a cabo procesamiento que debe ser realizado de manera regular, incluso en el caso en el que nuestras actividades no sean visibles o ni siquiera estén activas

• Proveedores de contenidos. Permiten almacenar y compartir datos entre aplicaciones. Los dispositivos Android incluye de serie un conjunto de proveedores de contenidos nativos que permiten acceder a datos del terminal, como por ejemplo los contactos o el contenido multimedia

• Intents. Los intents constituyen una plataforma para el paso de mensajes entre aplicaciones (y también dentro de una misma aplicación). Emitiendo un intent al sistema declara la intención de tu aplicación de que se lleve a cabo una determinada acción. El sistema será el encargado de decidir quién lleva a cabo las acciones solicitadas

• Receptores. Permiten a tu aplicación hacerse cargo de determinadas acciones solicitadas mediante Intents. Los receptores iniciarán automáticamente la aplicación para responder a un Intent que se haya recibido, haciendo que sean ideales para la creación de aplicaciones guiadas por eventos

• Widgets. Se trata de componentes visuales que pueden ser añadidos a la ventana principal de Android

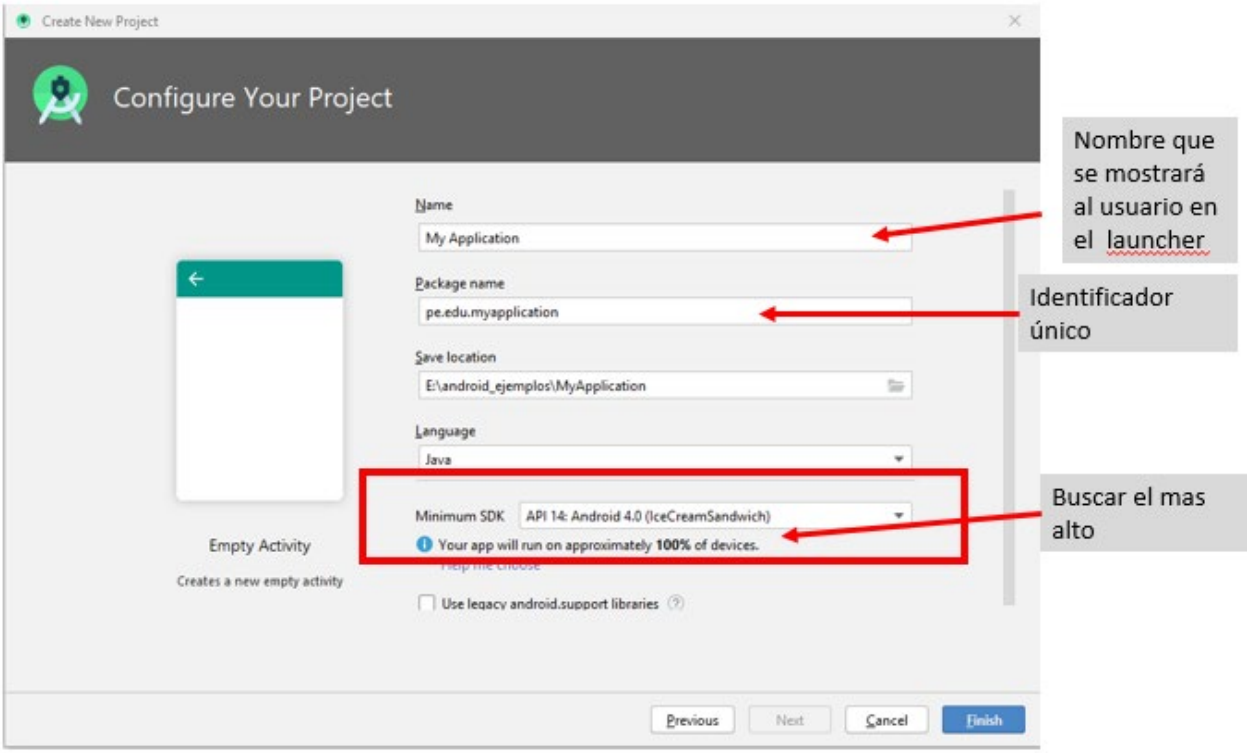
• Notificaciones. Las notificaciones permiten comunicarse con el usuario sin necesidad de robar el foco de la aplicación activa actualmente o de interrumpir a la actividad actual. Pror ejemplo,

cuando un dispositivo recibe un mensaje de texto, avisa al usuario mediante luces, sonidos o mostrando algún icono

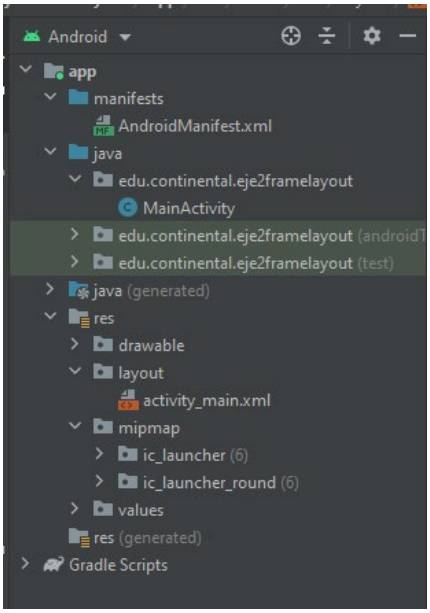
III. Procedimientos

Instrucciones:

1. Instala Android Studio, para ello puedes emplear la siguiente guía:
2. Crea un nuevo proyecto de Android Studio:
   - En el cuadro de diálogo completa como se muestra.



3. En el lado derecho tienes la vista del proyecto Android , el cual iremos revisando en sus componentes principales:



Apertura cada de las carpetas y archivos que se indican a continuación:

3.1 AndroidManifest.xml:

Enlazador de los archivos, los describe e indica como interactúan.

- Guarda configuración general de la aplicación.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="edu.continental.testgame">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="testGame"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.TestGame">
        <activity
            android:name=".MainActivity"
            android:theme="@android:style/Theme.Holo.Light.NoActionBar.Fullscreen"
            android:exported="true"  >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

TAREA A EFECTUAR:

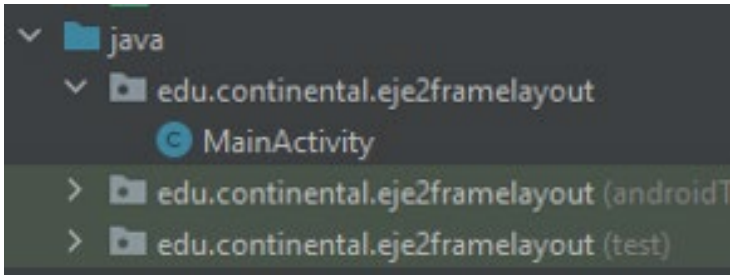
Cambia la línea **Android.label por : MiPrimeraApp**

3.2 Carpeta java

Guarda las clases java que definen la lógica de la app.

Agrupados por packages.
Si son muchas clases se recomienda crear packages.

Estructura por defecto de una clase de java en Android Studio.



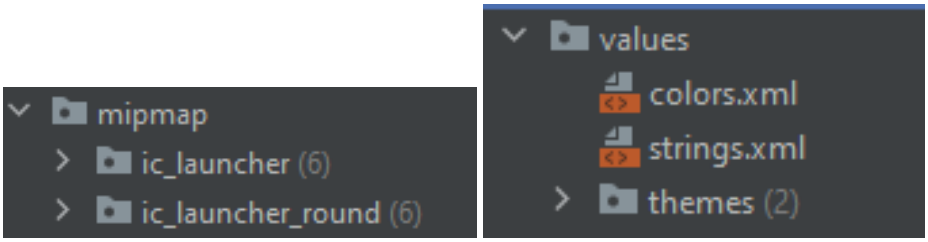

```
package edu.continental.eje2framelayout;

import ...

public class MainActivity extends AppCompatActivity {

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

TAREA A EFECTUAR:

Crea un nuevo package llamado **Entidades**, para ello efectúa click derecho sobre la carpeta Java new Package, ingresa el nuevo nombre del package.

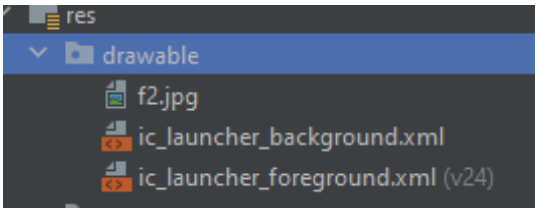
3.3 Res:

Todos los recursos de la aplicación se almacenan bajo la carpeta res/del proyecto. Dentro de esta carpeta encontramos diferentes subcarpetas para distintos tipos de recursos. Existen nueve tipos principales de recursos que tendrán su propia subcarpeta: valores simples, Drawables, layouts, animaciones, etilos, menús, searchables, XML y recursos raw. Al compilar nuestra aplicación estos recursos serán incluidos en el paquete apk que puede ser instalado en el dispositivo. Durante el proceso de compilación se generará también una clase R que contendrá referencias a cada uno de los recursos. Esto nos permitirá referenciar a los recursos desde nuestro código fuente



Drawable
una subcarpeta por cada densidad de la pantalla, xml o mapa de bits.

- Mipmap conserva, para el icono de la app.
- Values: definir textos, arrays, colores, estilos.



TAREA A EFECTUAR:

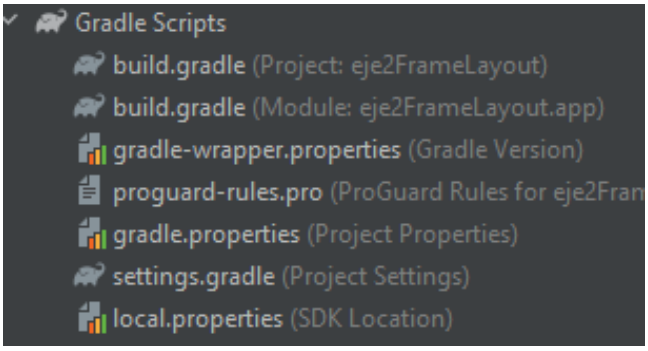
Crea un nuevo color en colors.xml

1. Ingresa a res values colors.xml
2. Ingresa el texto que se muestra en la línea 10.

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <resources>
3      <color name="purple_200">#FFBB86FC</color>
4      <color name="purple_500">#FF6200EE</color>
5      <color name="purple_700">#FF3700B3</color>
6      <color name="teal_200">#FF03DAC5</color>
7      <color name="teal_700">#FF018786</color>
8      <color name="black">#FF000000</color>
9      <color name="white">#FFFFFFFF</color>
10     <color name="fondoFormulario">#2F3068</color>
11
12 </resources>
```

Gradle - scripts

El sistema de compilación de Android compila recursos y código fuente de la app y los empaqueta en APK que puedes probar, implementar, firmar y distribuir. Android Studio usa Gradle, un paquete de herramientas de compilación avanzadas, para automatizar y administrar el proceso de compilación.



Detalles a tener en cuenta Gradle Scripts

- **compileSdkVersion:** es la versión de Android que utilizan las herramientas de compilación (Gradle) para compilar la aplicación para su lanzamiento, ejecución o depuración.
- **applicationId**: es el identificador único de la aplicación cuando ésta sea publicada en Google Play. No pueden existir 2 aplicaciones publicadas con el mismo identificador.
- **minSdkVersion**: es la versión mínima del sistema operativo Android necesaria para ejecutar la aplicación. La aplicación no podrá ser instalada en un dispositivo que tenga una versión inferior a ésta.
- **targetSdkVersion:** conocida como versión SDK destino, es la versión de Android en la que se creó la aplicación para que se ejecute. Indica que hemos probado nuestra aplicación hasta la versión que especificamos en esta propiedad.
- **versionCode:** es un número que es usado para determinar si una versión es más reciente que otra. Este numero no es mostrado a los usuarios pero sirve para definir el numero de versión dentro de la Play Store.
- **versionName:** es una cadena de texto, su único propósito es mostrar el número de versión de la aplicación a los usuarios de Google Play.

targetSdkVersion

Para todos los propósitos prácticos, en la mayoría de las aplicaciones deberemos establecer targetSdkVersion a la última versión de la API. Esto asegurará que nuestra aplicación se vea lo mejor posible en los dispositivos Android más recientes. Si no especificamos targetSdkVersion , por defecto es minSdkVersion .

Debe por tanto cumplirse:

minSdkVersion <= targetSdkVersion <= compileSdkVersion

Lo ideal es:

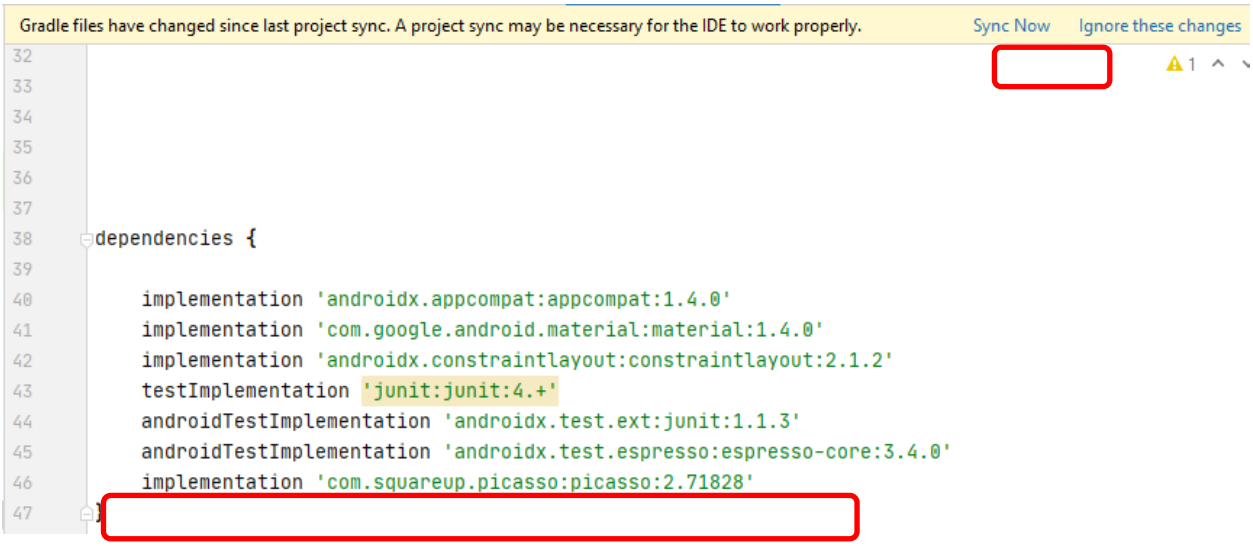
minSdkVersion (la menor posible) <= targetSdkVersion == compileSdkVersion (la versión más reciente de SDK)

TAREA A EFECTUAR:

1. Ingresa a gradle Scripts build.gradle, apertura el archivo y efectúa la siguiente agregación, lo cual nos permitirá mostrar imágenes directamente desde el Internet.

```
implementation 'com.squareup.picasso:picasso:2.71828'
```

2. Finalmente debes efectuar clic a **Sync Now**


Gradle files have changed since last project sync. A project sync may be necessary for the IDE to work properly.
Sync Now
Ignore these changes
dependencies {
implementation 'androidx.appcompat:appcompat:1.4.0'
implementation 'com.google.android.material:material:1.4.0'
implementation 'androidx.constraintlayout:constraintlayout:2.1.2'
testImplementation 'junit:junit:4.+'
androidTestImplementation 'androidx.test.ext:junit:1.1.3'
androidTestImplementation 'androidx.test.espresso:espresso-core:3.4.0'
implementation 'com.squareup.picasso:picasso:2.71828'
}

Semana 2 – Sesión 2

Layouts

Sección: ……..……..	Apellidos : ……..……
Docente : Pedro Yuri Marquez Solis	Nombres : ……
Unidad : Unidad 1	Fecha: …../……/……. Duración: 60 min

I. Propósito: Obtener un Activity con la aplicación del Layout LinearLayout .

II. Descripción de la actividad a realizar (caso.)

Los Layouts permiten ubicar los Views dentro de un Activity, básicamente un Layout es un elemento que representa el diseño de la interfaz de usuario de componentes gráficos como una actividad, fragmento o widget. Los layouts pueden ser creados a través de archivos XML o con código Java de forma programática

Algunos de los view groups más populares son:

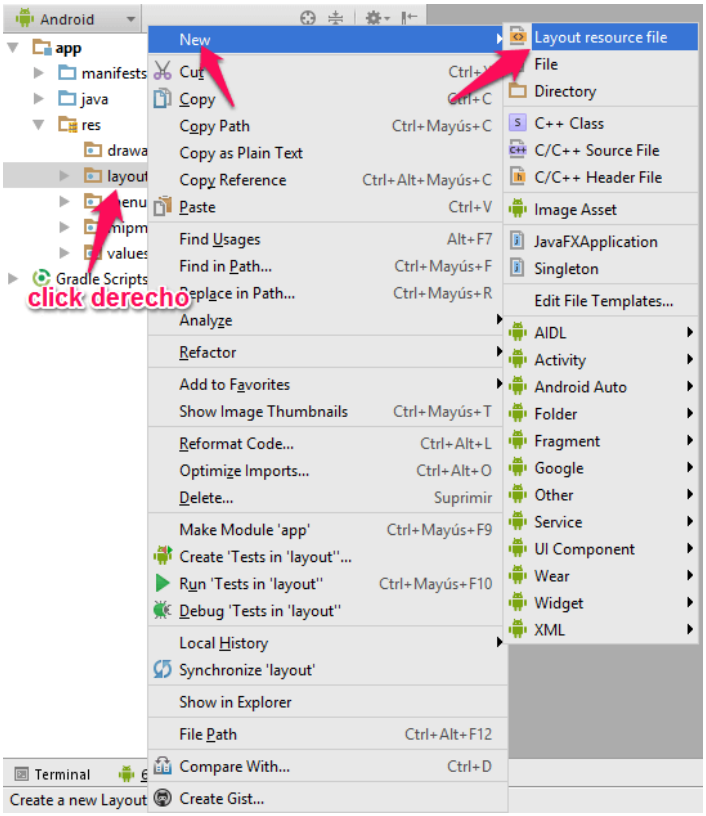
- ❑ LinearLayout
- ❑ FrameLayout
- ❑ RelativeLayout
- ❑ TableLayout
- ❑ GridLayout

En la presente actividad generaremos interfaces utilizando Views y las propiedades del LinearLayout.

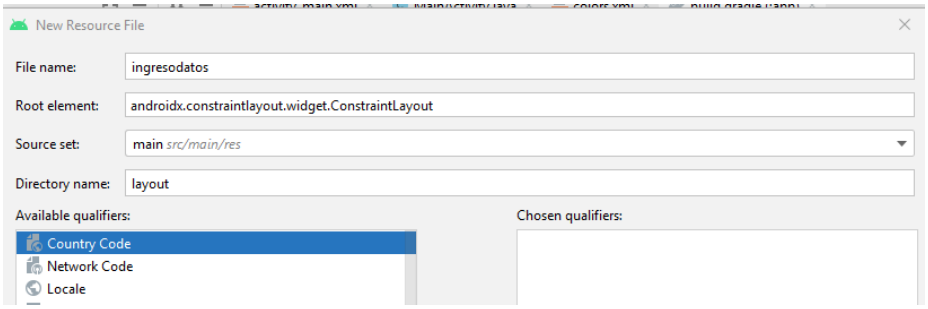
III. Procedimientos

Instrucciones:

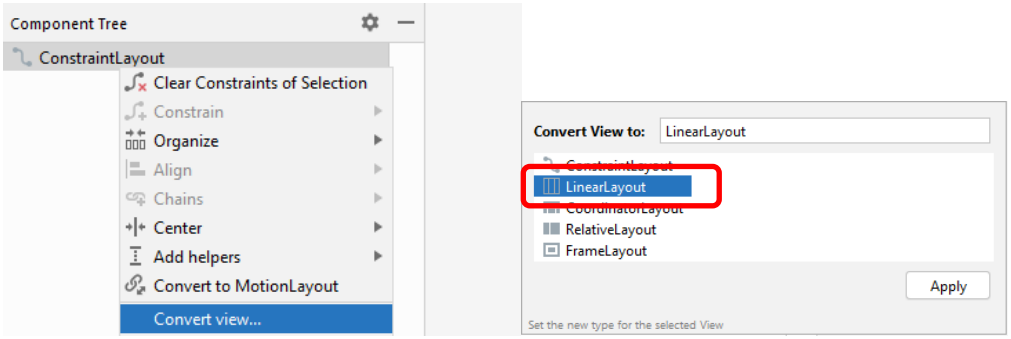
1. Crea un nuevo proyecto, denominado **Layouts_XYZ**, donde xyz representan tus iniciales (Esto con fines de evaluación).
2. Luego de haber creado el proyecto elige en las plantillas de actividades la de Empty Activity.
3. A continuación, vamos a agregar un nuevo layout:



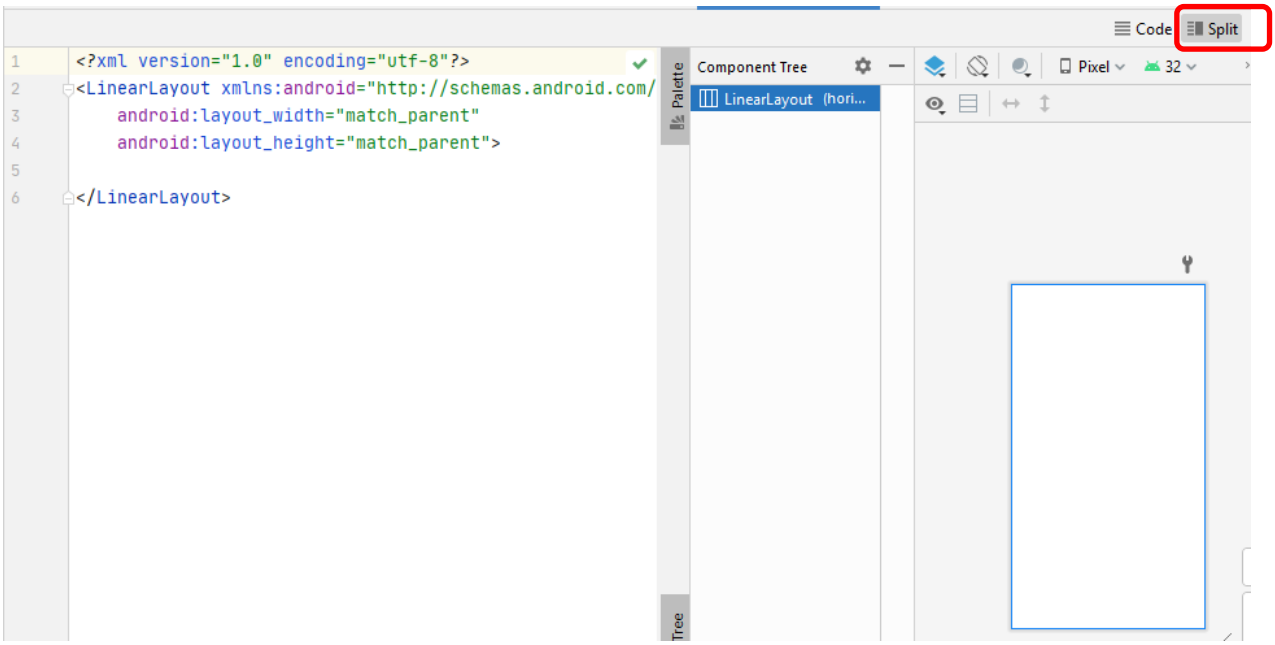
4. Proporcionar el nombre Ingresodatos



5. Efectúa clic derecho sobre el árbol de componentes y elige Convert View a continuación LinearLayout



6. Configura la vista del diseñador para que se muestre tanto el código xml, como el diseño obtenido, efectúa clic sobre el botón Split.

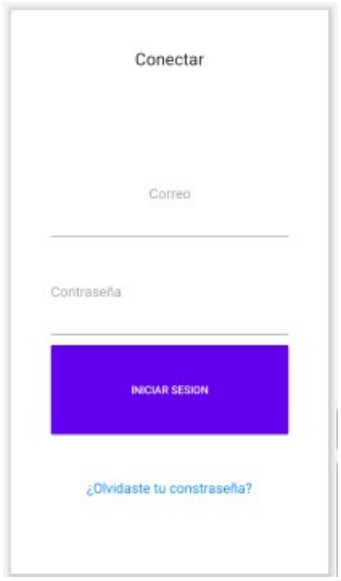


7. En la vista código inserta la siguiente codificación:

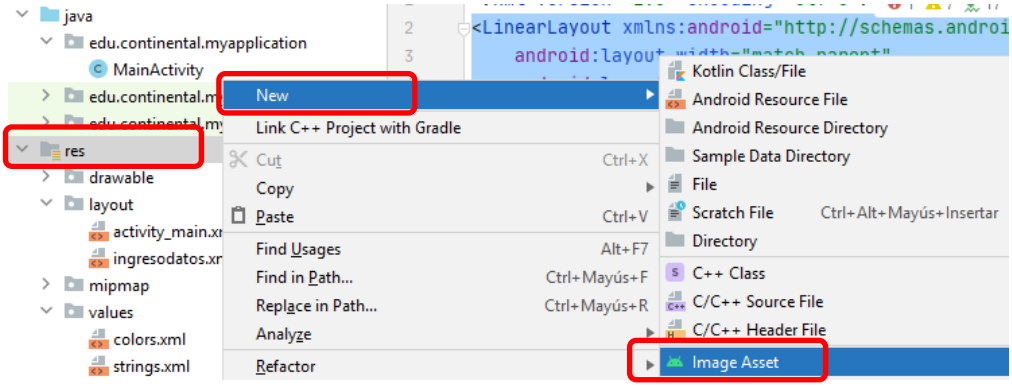
```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="48dp">

    <TextView
        android:id="@+id/texto_conectar"
        android:layout_width="wrap_content"
        android:layout_height="0dp"

        android:layout_gravity="center_horizontal"
        android:layout_weight="1"
        android:text="Conectar"
        android:textAppearance="?android:attr/textAppearanceLarge" />

    <EditText
        android:id="@+id/input_usuario"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_gravity="center_horizontal"
        android:layout_weight="1"
        android:drawableLeft="@drawable/ic_mail"
        android:gravity="center"
        android:hint="Correo"
        android:inputType="textEmailAddress"
        />

    <EditText
        android:id="@+id/input_contrasena"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_gravity="center_horizontal"
        android:layout_weight="1"
        android:ems="10"
        android:hint="Contraseña"
        android:inputType="textPassword"
        android:drawableLeft="@drawable/ic_text"
        />
```

```
    <Button
        android:id="@+id/boton_iniciar_sesion"
        style="?android:attr/buttonStyleSmall"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_gravity="center_horizontal"
        android:layout_weight="1"
        android:text="Iniciar Sesion" />

    <TextView
        android:id="@+id/texto_olvidaste_contrasena"
        android:layout_width="wrap_content"
        android:layout_height="0dp"
        android:layout_gravity="center_horizontal"
        android:layout_weight="1"
        android:gravity="center_vertical"
        android:text="¿Olvidaste tu constraseña?"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:textColor="#0E8AEE" />
</LinearLayout>
```

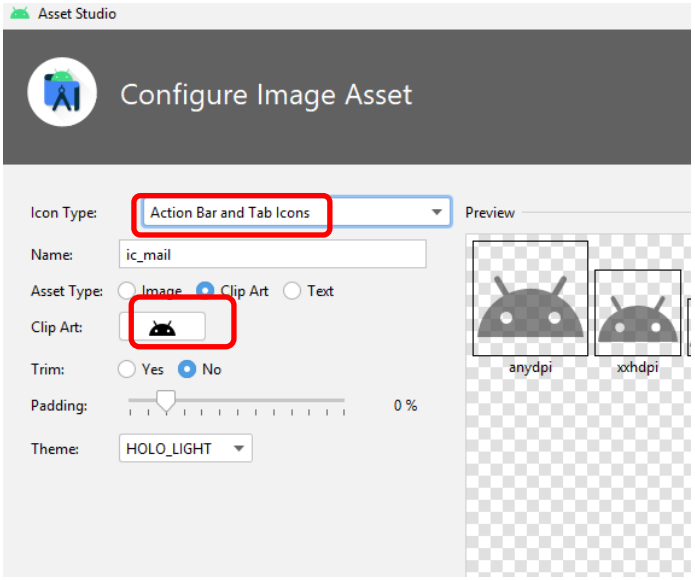
8. Debe obtener una interface como la siguiente:



9. Agregue los recursos faltantes, para ello siga la siguiente secuencia:

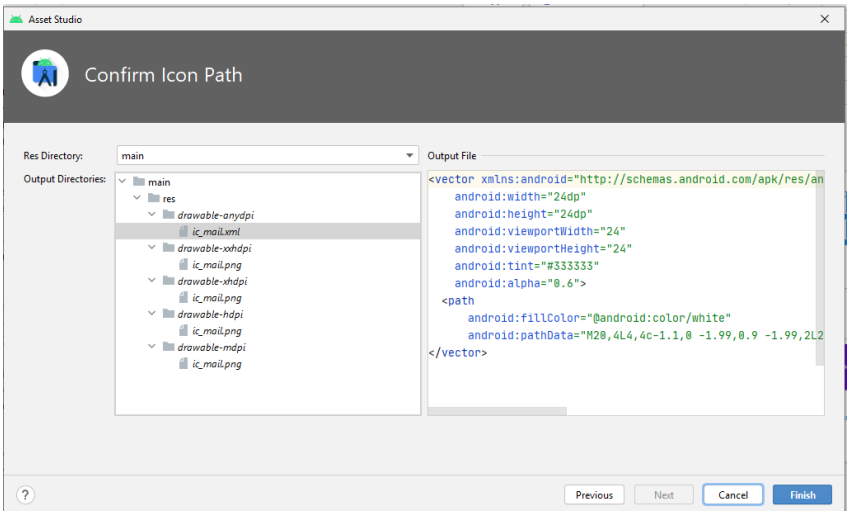


10. En la pantalla de **Configure Image Asset**, en **IconType** elija Action Bar and tab icons, en **Name** inserte ic_mail, luego para elegir el ícono haga clic en **Clip Art**

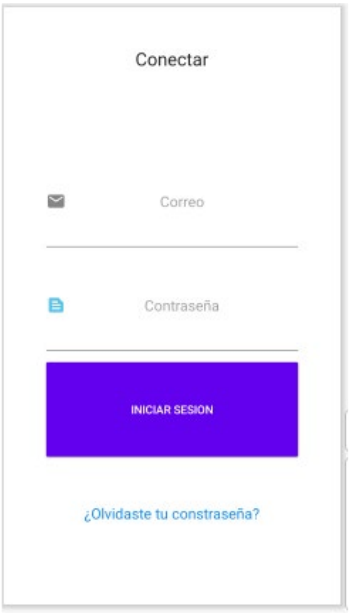


11. Para facilitar la ubicación de un ícono inserte el texto mail en el buscador

12. En la pantalla de Confirm Icon Path elija la opción **Finish.**



13. Repita el proceso para el ícono de **ic_text.**
14. El diseño final debería verse como se muestra:



Semana 3 – Sesión 2

Sección: ……..……..	Apellidos : ……..……
Docente : Pedro Yuri Marquez Solis	Nombres : ……
Unidad : Unidad 1	Fecha: …../……/……. Duración: 60 min

Creación de Menús

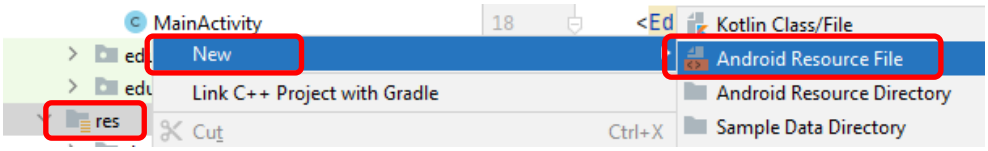
I. Propósito: Permitir al usuario definir y usar menús como elemento integrador.

II. Descripción de la actividad a realizar (caso.)

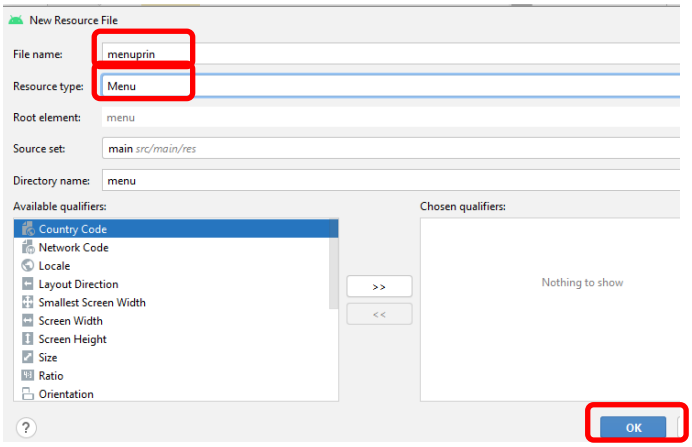
En las soluciones móviles es común emplear varias Actividades, las cuales requieren ser invocadas, un elemento que facilita este proceso es el View de tipo Menpu.

III. Procedimientos

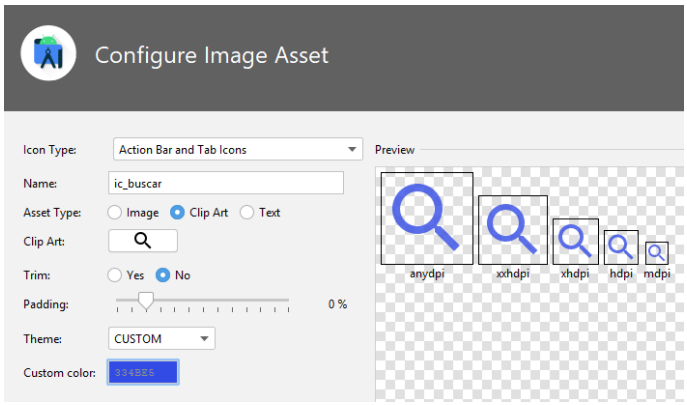
1. Apertura del proyecto **Layouts_XYZ.**
2. Agrega las actividades **Buscar, Insertar y Detalles**, todas basadas en la plantilla Empty.
3. Ahora procede a crear un menú, para ello en res efectúa clic derecho y elige New Android Resource File

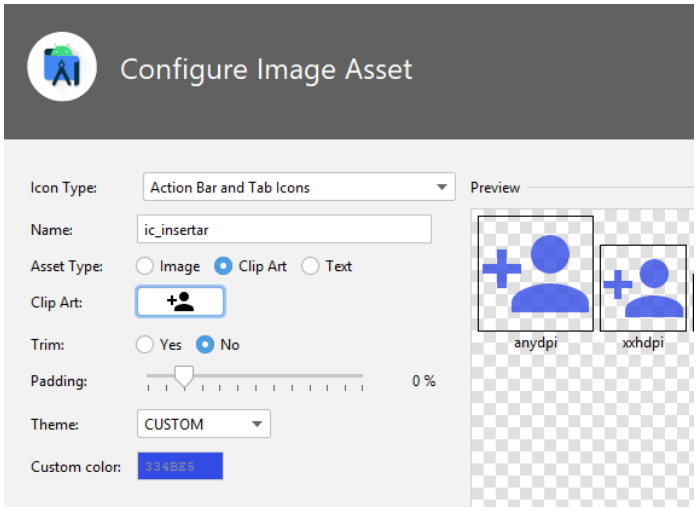


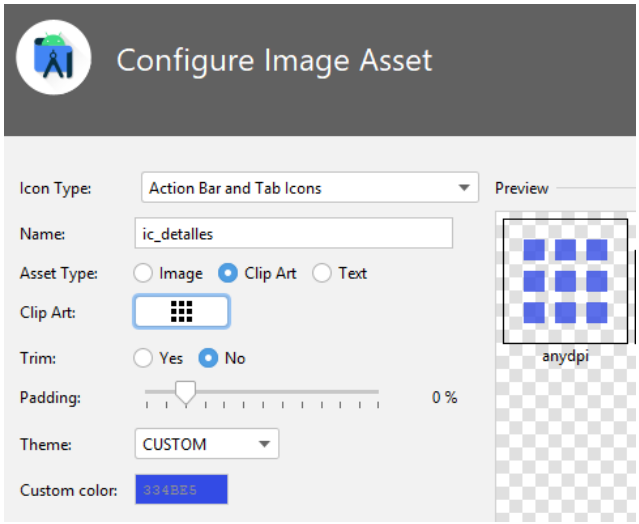
4. En el cuadro de dialogo que se muestra ingresa el File name: menuprin, y en Resource Type elige el tipo : Menu. Finalmente click al botón OK.



5. Ahora necesitamos crear los íconos correspondientes a cada opción del menú, para ello repite los pasos efectuados en la semana 2, paso número 2, cuidando asignar los siguientes propiedades:







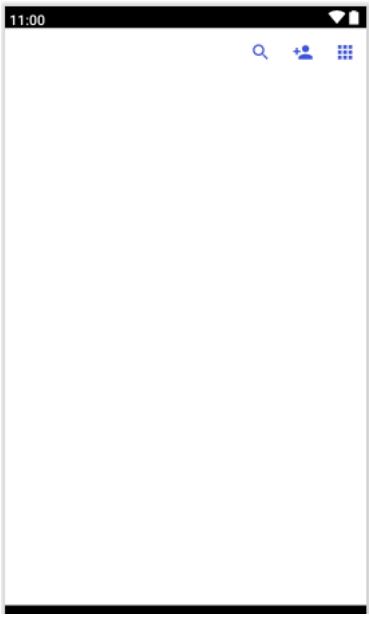
6. Redefina los colores en colors.xml, tal como se muestra:

```
<color name="purple_200">#FFBB86FC</color>
<color name="purple_500">#FF6200EE</color>
<color name="purple_700">#FF3700B3</color>
<color name="teal_200">#FF03DAC5</color>
<color name="teal_700">#FF018786</color>
<color name="fondo1">#E6E1EC</color>
<color name="fondo2">#DFD5D5</color>
<color name="fondo3">#FF03DAC5</color>
<color name="black">#FF000000</color>
<color name="white">#FFFFFFFF</color>
<color name="fondoFormulario">#2F3068</color>
```

7. Modifique el archivo menuprin.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:android="http://schemas.android.com/apk/res/android"
    >

    <item android:title="Buscar"
        android:id="@+id/optbuscar"
        android:icon="@drawable/ic_buscar"
        app:showAsAction="always"/>
    <item android:title="Insertar"
        android:id="@+id/optInsertar"
        android:icon="@drawable/ic_insertar"
        app:showAsAction="always"/>
    <item android:title="Detalles"
        android:id="@+id/optdetalles"
        android:icon="@drawable/ic_detalles"
        app:showAsAction="always"/>
</menu>
```

8. El diseño del menú deberá verse de la siguiente forma:



9. Ahora debemos implementar el menú en el activity_main que es donde se quiere mostrar inicialmente. Para ello efectúa los siguientes pasos.
10. Insertar el menú en el MainActivity, para ello agrega los métodos onCreateOptionsMenu y onOptionsItemSelected ayudándote del autocompletado de Android Studio.

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.menuprin, menu);
    return super.onCreateOptionsMenu(menu);
}

@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {
    switch (item.getItemId())
    {
        case R.id.optbuscar:{

            break;
        }
        case R.id.optInsertar:{
            Intent  i = new Intent(this,Insertar.class );
            startActivity(i);
            break;
        }
        case R.id.optdetalles:{
            Intent  i = new Intent(this,Detalles.class );
            startActivity(i);
            break;
        }
    }

    return super.onOptionsItemSelected(item);
}
```

11. Finalmente compruebe la operación del menú.


My Application
Hello World!

Semana 4 – Sesión 2

Sección: ……..……..	Apellidos : ……..……
Docente : Pedro Yuri Marquez Solis	Nombres : ……
Unidad : Unidad 1	Fecha: …../……/……. Duración: 60 min

Empleo de NavDreawer

I. Propósito: Emplear navDreawer como elemento integrador y optimizador de la app .

II. Descripción de la actividad a realizar (caso.)

Fragments:

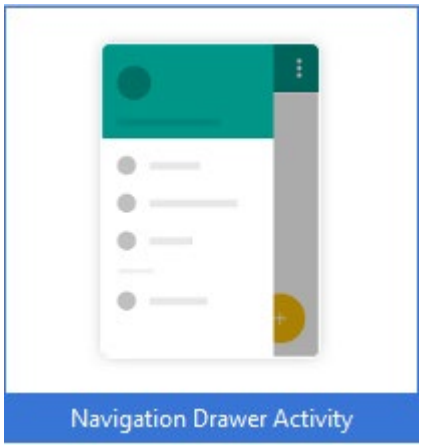
Un fragmento es una miniactividad contenida dentro de una actividad anfitriona, manejando su propio diseño (un recurso layout propio) y ciclo de vida.

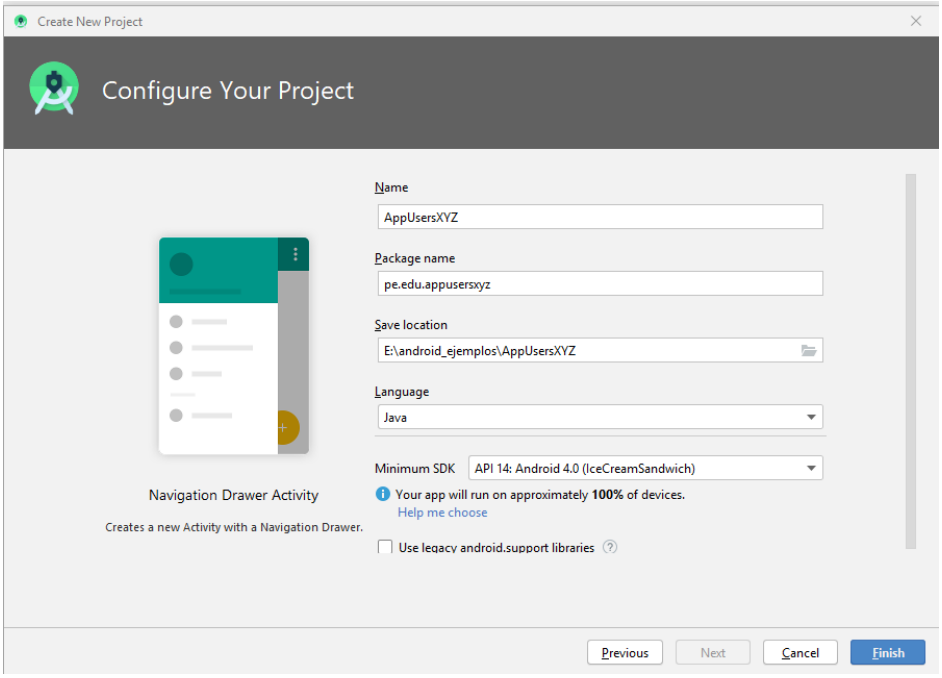
Los fragment permiten reusar código y ahorrar tiempo de diseño a la hora de desarrollar una aplicación, en cualquier tipo de tamaño de pantalla y orientación, se pueden reutilizar desde distintas actividades.

El uso de fragmentos dinámicos se facilita con el empleo de la plantilla NavDreawer.

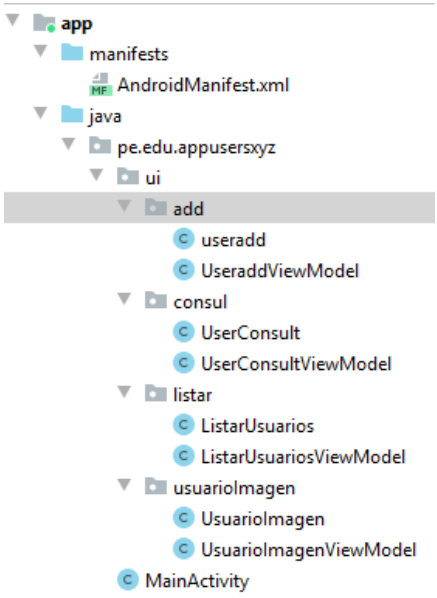
III. Procedimientos

1. **Creamos un proyecto navDreawer, al cual denominaremos AppUsersXYZ**

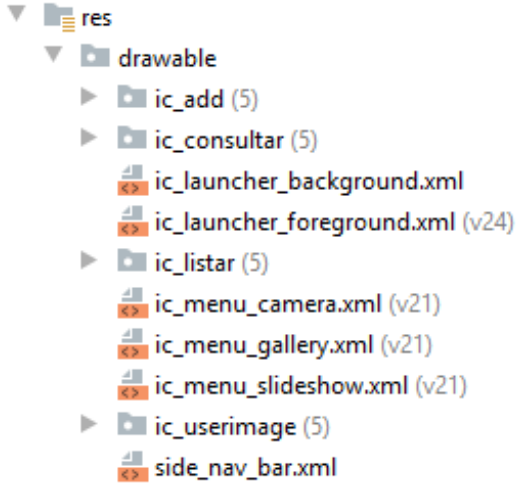




2. **Agregar fragments, cuidando organizarlos en packages como se muestra:**



3. Agregar los elementos gráficos del menú:



4. Configurar los siguientes elementos:

4.1 Activity_main_drawer.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    tools:showIn="navigation_view">

    <group android:checkableBehavior="single">
        <item
            android:id="@+id/nav_add"
            android:icon="@drawable/ic_add"
            android:title="Insertar usuario" />
        <item
            android:id="@+id/nav_consultar"
            android:icon="@drawable/ic_consultar"
            android:title="Consultar usuario" />
        <item
            android:id="@+id/nav_listar"
            android:icon="@drawable/ic_listar"
            android:title="Listar usuario" />
        <item
            android:id="@+id/nav_userimage"
            android:icon="@drawable/ic_userimage"
            android:title="Listar usuario" />
```

```
    </group>
</menu>
```

5. Definir el contenido del archivo strings.xml

```
<resources>
    <string name="app_name">TestYMS</string>
    <string name="navigation_drawer_open">Open navigation drawer</string>
    <string name="navigation_drawer_close">Close navigation drawer</string>
    <string name="nav_header_title">Android Studio</string>
    <string name="nav_header_subtitle">android.studio@android.com</string>
    <string name="nav_header_desc">Navigation header</string>
    <string name="action_settings">Settings</string>
    <string name="menu_add">Agregar Usuario</string>
    <string name="menu_consultar">Consultar</string>
    <string name="menu_listar">Listar</string>
    <!-- TODO: Remove or change this placeholder text -->
    <string name="hello_blank_fragment">Hello blank fragment</string>
    <string name="title_activity_splash">Splash</string>
    <string name="hello_world">Hello Square World!</string>
    <string name="logo">eSports</string>
</resources>
```

6. Define el contenido de mobile_navigation.xml

```
<?xml version="1.0" encoding="utf-8"?>
<navigation 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:id="@+id/mobile_navigation"
    app:startDestination="@+id/nav_add">

    <fragment
        android:id="@+id/nav_add"
        android:name="pe.edu.appusersxyz.ui.add.useradd"
        android:label="Insertar"
        tools:layout="@layout/user_add_fragment">

    </fragment>
    <fragment
        android:id="@+id/nav_consultar"
        android:name="pe.edu.appusersxyz.ui.consul.UserConsult"
        android:label="Consultar"
        tools:layout="@layout/user_consult_fragment">
    </fragment>

    <fragment
        android:id="@+id/nav_listar"
        android:name="pe.edu.appusersxyz.ui.listar.ListarUsuarios"
        android:label=  "Listar"
        tools:layout="@layout/listar_usuarios_fragment" />

    <fragment
        android:id="@+id/nav_userimage"
        android:name="pe.edu.appusersxyz.ui.usuarioImagen.UsuarioImagen"
        android:label="Usuario imagen"
        tools:layout="@layout/usuario_imagen_fragment" />
</navigation>
```

7. Configurando el MainActivity:

```
    mAppBarConfiguration = new AppBarConfiguration.Builder(
            R.id.nav_add, R.id.nav_consultar, R.id.nav_listar)
            .setDrawerLayout(drawer)
            .build();
    NavController navController = Navigation.findNavController( activity: this, R.id.nav_host_fragment);
    NavigationUI.setupActionBarWithNavController( activity: this, navController, mAppBarConfiguration);
    NavigationUI.setupWithNavController(navigationView, navController);
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
```

SPLASH ACTIVITY

Agregar una Activity con el nombre splash

8. **Definir los colores a emplear:**

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="purple_200">#FFBB86FC</color>
    <color name="purple_500">#FF6200EE</color>
    <color name="purple_700">#FF3700B3</color>
    <color name="teal_200">#1A9804</color>
    <color name="teal_700">#FF018786</color>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
    <color name="yeloow">#F8FF3B</color>
</resources>
```

9. **Establecer una gradiente para el fondo:**

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <gradient
        android:startColor="@color/yeloow"
        android:endColor="@color/teal_200"
        android:angle="270"
        ></gradient>
</shape>
```

10. **Agregar una nueva fuente**

- Clic derecho a res new folder folder Font
- Dentro del folder Font pegar la nueva familia de fuentes.

11. **Agregar un nuevo gráfico para el logo:**

- Arrastrar el archivo png, considerando que el nombre se encuentre todo en minúsculas, sin espacios en blanco. Recuerda el gráfico se moverá del origen así que considera tener una copia.

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout 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:background="@drawable/gradientesplash"
    tools:context=".Splash">

 <TextView
     android:id="@+id/logo"
     android:layout_width="467dp"
     android:layout_height="155dp"
     android:layout_gravity="center"
     android:gravity="center"
     android:fontFamily="@font/starlight"
     android:rotationX="33"400
     android:paddingTop="80dp"
     android:text="@string/logo"
     android:textColor="@color/white"
     android:textSize="120dp"></TextView>

 <ImageView
     android:layout_width="200dp"
     android:layout_height="300dp"
     android:src="@drawable/android09"
     android:layout_gravity="bottom|center"

     ></ImageView>
</FrameLayout>
```

12. Agregar un Nuevo tema

```
<style name="Theme.TestYMS.NoActionBar">
    <item name="windowActionBar">false</item>
    <item name="windowNoTitle">true</item>
</style>
```

13. Configurar el AndroidManifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="edu.yms.testyms">

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.TestYMS.NoActionBar">
        <activity
            android:name=".Splash"
            android:label="@string/title_activity_splash">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name=".MainActivity"
            android:label="@string/app_name"
            android:theme="@style/Theme.TestYMS.NoActionBar">

        </activity>
    </application>

</manifest>
```

14. Agregar las propiedades del splash para que se muestren a toda pantalla

```
public class Splash extends AppCompatActivity {
    TextView logo;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_NO_TITLE);
        getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
WindowManager.LayoutParams.FLAG_FULLSCREEN);
        logo = findViewById(R.id.logo);
        setContentView(R.layout.activity_splash2);
    }
}
```

15. Agregar la carga del MainActivity desde el splash

```
public class Splash extends AppCompatActivity {
    TextView logo;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_NO_TITLE);
        getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
WindowManager.LayoutParams.FLAG_FULLSCREEN);
        logo = findViewById(R.id.logo);
        setContentView(R.layout.activity_splash2);
        esperarYCerrar(2000);
    }
    public void esperarYCerrar(int milisegundos) {
        Handler handler = new Handler();
        handler.postDelayed(new Runnable() {
            public void run() {
                // acciones que se ejecutan tras los milisegundos
                llamarIndex();
            }
        }, milisegundos);
    }
    void llamarIndex(){
        Intent i = new Intent(this, MainActivity.class);
        startActivity(i);
    }
}
```

Segunda unidad

Semana 5 – Sesión 2

Persistencia de datos

Sección:	Apellidos : .....
Docente : Pedro Yuri Marquez Solis	Nombres : .....
Unidad : Unidad 2	Fecha: ...../...../..... Duración: 60 min

I. Propósito: Efectuar persistencia de datos en un archivo.

II. Descripción de la actividad a realizar (caso.)

Los datos que una aplicación móvil puede almacenar, pueden ser de cualquier índole, desde booleanos, cadenas, flotantes y enteros hasta la lista de contactos, fotografías, records de juegos y todo lo que necesite.
Android ofrece distintas posibilidades para guardar información, siendo los mecanismos más utilizados:

- Archivo Preferencias
- archivos de texto plano
- base de datos: local o remota

En la siguiente guía se desarrolla un ejemplo de almacenamiento en un archivo de preferencias.

III. Procedimientos

1. Crear un nuevo proyecto denominado preferencias_xyz.
2. Modificar el archivo activity_main.xml como se muestra a continuación:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">


    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
```

```
        android:layout_marginStart="32dp"
        android:text="User Name"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.066"></TextView>


    <EditText
        android:id="@+id/edtusername"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="12dp"
        android:padding="20dp"
        app:layout_constraintTop_toBottomOf="@+id/textView"
        tools:layout_editor_absoluteX="0dp"></EditText>

    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="56dp"
        android:layout_marginTop="60dp"
        android:text="Email"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/edtusername"
        app:layout_constraintVertical_bias="0.0"></TextView>

    <EditText
        android:id="@+id/edtemail"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="32dp"
        android:padding="20dp"
        app:layout_constraintTop_toTopOf="@+id/textView2"
        tools:layout_editor_absoluteX="0dp"></EditText>

    <Button
        android:id="@+id/btnescribir"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="8dp"
        android:text="Guardar"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/edtemail" />

    <Button
        android:id="@+id/btnleer"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="36dp"
        android:text="Leer"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/btnescribir" />

    <Button
        android:id="@+id/btnsalir"
        android:layout_width="365dp"
        android:layout_height="36dp"
        android:layout_marginTop="76dp"
        android:text="Salir"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/btnleer"
        app:layout_constraintVertical_bias="0.0"
        tools:layout_editor_absoluteX="23dp" />

    <CheckBox
        android:id="@+id/chkArchivo"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Archivo"
```

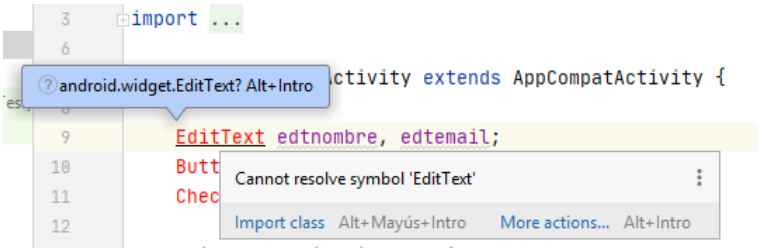
```
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.804"
        app:layout_constraintStart_toEndOf="@+id/btnescribir"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.499" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

3. Modificar el archivo MainActivity.class

```
public class MainActivity extends AppCompatActivity implements View.OnClickListener {

    EditText edtnombre, edtemail;
    Button btnguardar, btnleer, btnsalir;
    CheckBox chktipo;

    private static final String ARCHIVO_PREFRENCIAS="Preferencias_app";
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        edtnombre= findViewById(R.id.edtusername);
        edtemail=findViewById(R.id.edtemail);
        chktipo=findViewById(R.id.chkArchivo);
        btnguardar=findViewById(R.id.btnescribir);
        btnleer=findViewById(R.id.btnleer);
        btnsalir=findViewById(R.id.btnsalir);
        btnguardar.setOnClickListener(this);
        btnleer.setOnClickListener(this);
        btnsalir.setOnClickListener(this);

    }

    @Override
    public void onClick(View view) {
        String username=edtnombre.getText().toString();
        String email = edtemail.getText().toString();
        switch (view.getId()){
            case  R.id.btnescribir:
            {
                if (chktipo.isChecked()){
                    //guardar en archivo
                    try{
                        OutputStreamWriter fichero=new
OutputStreamWriter(openFileOutput("datos.txt", Context.MODE_PRIVATE));
                        fichero.write(username+";"+email);
                        fichero.flush();
                        fichero.close();
                        Toast.makeText(this, "Guardado en archivo", Toast.LENGTH_SHORT).show();
                    }
                    catch (Exception exception){
                        Toast.makeText(this, "Error al crear el archivo",
Toast.LENGTH_SHORT).show();
                    }

                }
                else
                { //guardar en sharedPreferences
                    SharedPreferences preferencias= getSharedPreferences(ARCHIVO_PREFRENCIAS, 0);
                    SharedPreferences.Editor editor=preferencias.edit();
                    editor.putString("username", username);
                    editor.putString("email", email);
                    editor.commit();
                    Toast.makeText(this, "Guardado en Preferencias", Toast.LENGTH_SHORT).show();

                }
                break;
            }

            case R.id.btnleer:{
                if (chktipo.isChecked()){
                    //leer desde archivo
```

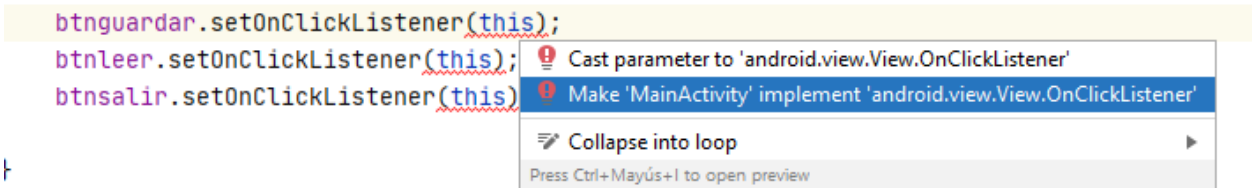
```
                    String contenido;
                    try{
                        BufferedReader fichero= new BufferedReader(new
InputStreamReader(openFileInput("datos.txt")));
                        contenido=fichero.readLine();
                        fichero.close();
                        username=contenido.substring(0,contenido.indexOf(";"));
                        email=contenido.substring(contenido.indexOf(";"), contenido.length());
                        Toast.makeText(this, "Leyendo desde archivo, el usuario es "+username
                                + "El email es "+ email , Toast.LENGTH_SHORT).show();

                    }
                    catch (Exception e){
                        Toast.makeText(this, "Error al leer", Toast.LENGTH_SHORT).show();
                    }
                }
                else
                {//leer desde archivo de preferencias
                    SharedPreferences preferencias= getSharedPreferences(ARCHIVO PREFRENCIAS,0);
                    if (preferencias.contains("username")){
                        username=preferencias.getString("username","");
                        email=preferencias.getString("email","");
                        Toast.makeText(this, "Recuperado desde preferencias "
                                +username + " > "+ email , Toast.LENGTH_SHORT).show();
                    }
                }
                break;

            }
        }
    }
}
```

4. Cuando el IDE de Android Studio muestre errores con instrucciones marcadas de rojo es probable que requiera agregar las importaciones respectivas de las clases.



5. Para resolverlo situarse sobre la instrucción con el error, presionar Alt+enter y elegir la opción import class..

6. En el caso de las instrucciones setOnClickListener con error repita el proceso indicado anteriormente, elija la segunda opción make 'MainActivity' implement para



Semana 6 – Sesión 2

Sección: ……..……..	Apellidos : ……..……..
Docente : Pedro Yuri Marquez Solis	Nombres : ……..
Unidad : Unidad 2	Fecha: …../……/……. Duración: 60 min

Persistencia de datos con Sqlite

I. Propósito: Elaborar un proyecto con persistencia de datos empleando base de datos SqLite..

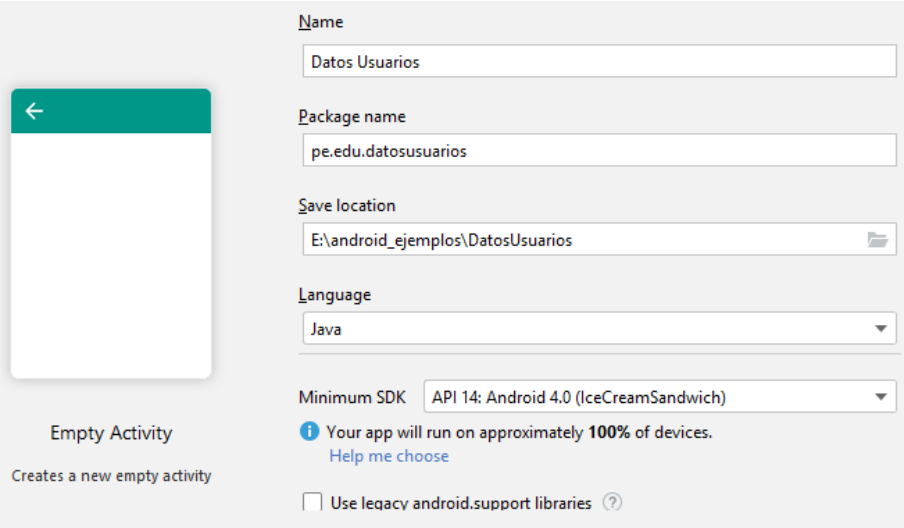
II. Descripción de la actividad a realizar (caso.)

SQLite es un gestor de bases de datos relacional y es de código abierto. En el caso del sistema Android, SQLite se implementa como una librería, por lo cual no necesita ser instalado, no necesita iniciar procesos, no necesita una configuración inicial. SQLite se implementa como una librería C compacta, en lugar de ejecutarse en un proceso propio. Debido a ello, cualquier base de datos que creemos será integrada como parte de su correspondiente aplicación.
En esta práctica implementaremos una base de datos SqLite para efectuar procesos CRUD.

III. Procedimientos

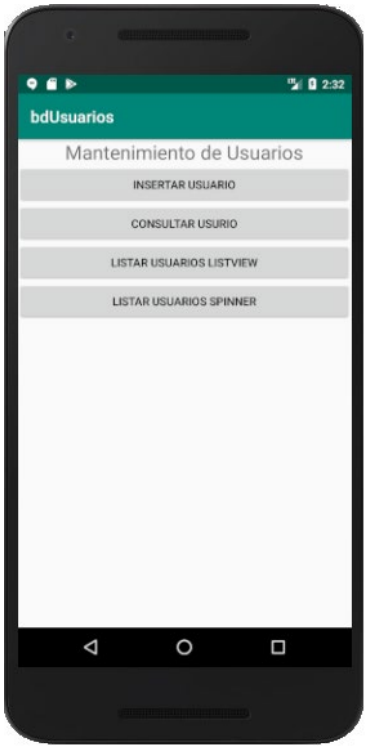
Instrucciones:

1. **Iniciamos creando un proyecto en blanco al cual denominaremos Datos usuarios**



2. Diseño general de la aplicación:

El activity_main.xml, tendrá el siguiente diseño.



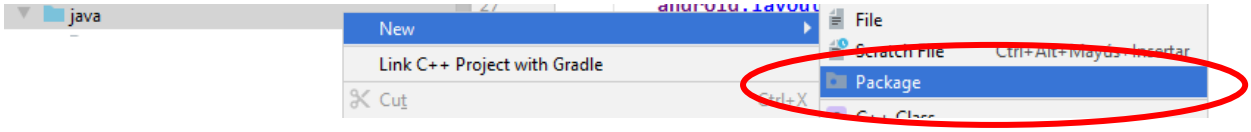
3. Modificar el Código del activity_main.xml

```
<?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">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:gravity="center"
        android:text="Mantenimiento de Usuarios"
        android:textSize="24sp" />

    <Button
        android:id="@+id/btnInsUsuario"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Insertar usuario" />

    <Button
        android:id="@+id/btnConsulUsuario"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:onClick="llamarConsultar"
        android:text="Consultar usurio" />

    <Button
```

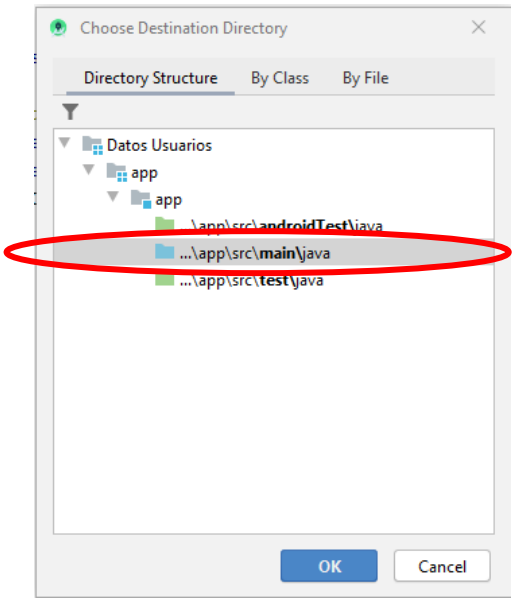
```
        android:id="@+id/btnlistarusuarios"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Listar Usuarios listView" />

    <Button
        android:id="@+id/lstusuSpinner"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Listar usuarios Spinner" />

</LinearLayout>
```

4. Establecer la Estructura del proyecto:

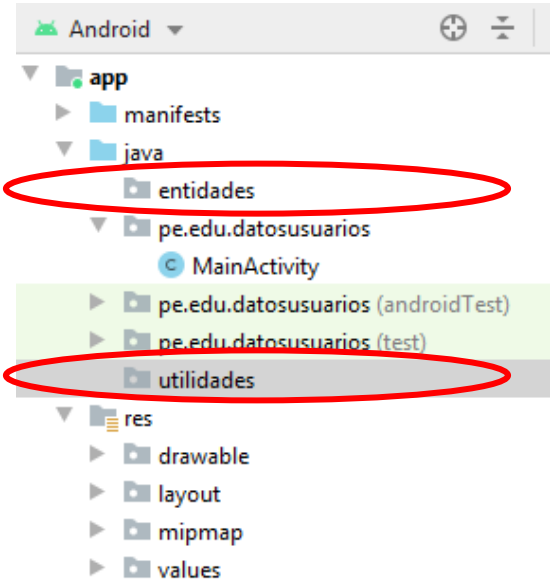
Para mantener el orden adecuado del proyecto es importante que desde el inicio generemos dos nuevos paquetes, el primero llamado entidades y el segundo utilidades, siguiendo la secuencia:



En el cuadro de dialogo elegimos:



Luego de crear los paquetes el proyecto queda como se muestra en la imagen:



5. CREACIÓN DE LA BASE DE DATOS SQLITE

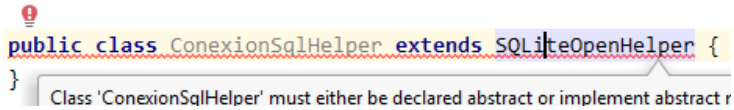
a. Crear la clase Usuario:

Dentro del paquete entidades crear la clase Java Usuario, que contiene la siguiente definición:

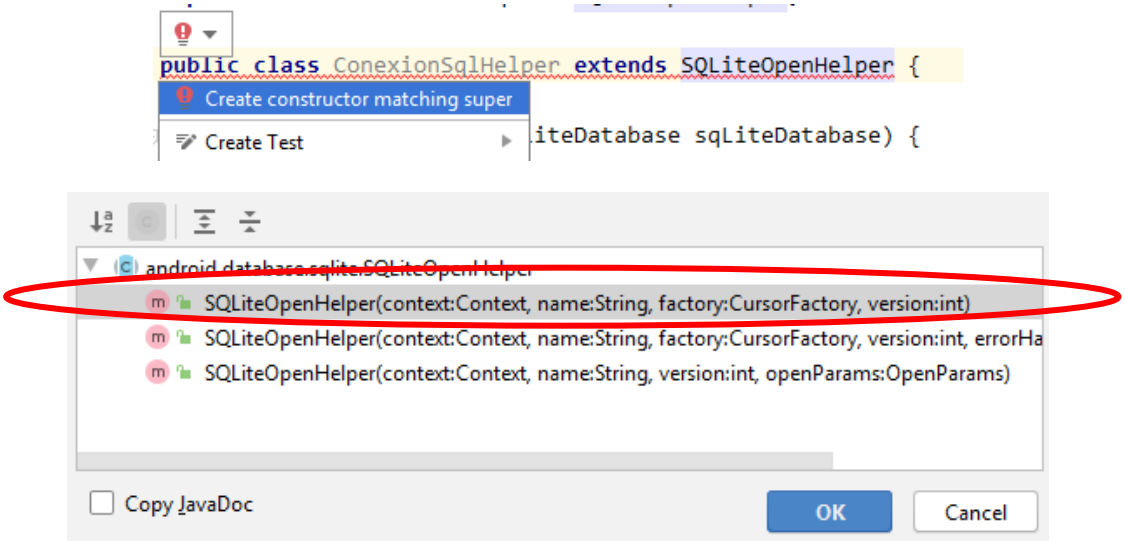
```
public class Usuario {
    public Integer id;
    public String nombre;
    public String telefono;
    public Usuario(){
        this.id=1;
        this.nombre="nnn";
        this.telefono="0000";
    }
    public Usuario(Integer id, String nombre, String telefono) {
        this.id = id;
        this.nombre = nombre;
        this.telefono = telefono;
    }

    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 getTelefono() {
        return telefono;
    }

    public void setTelefono(String telefono) {
        this.telefono = telefono;
    }
}
```

6. Crear la clase ConexionSqlLiteHelper dentro del paquete entidades:

Al extender la clase Android Studio, mostrará el icono de error para generar los métodos onCreate y onUpgrade y posteriormente su constructor.



Creamos el constructor y seleccionamos el primero



Quedaría como se muestra:

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

    @Override
    public void onCreate(SQLiteDatabase sqLiteDatabase) {

    }

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

    }
}
```

7. Crear la clase Utilidades dentro del paquete utilidades, debe contener las siguientes constantes, esto con la finalidad de mejorar el control de nuestra aplicación.

```
public class Utilidades {

    public static final String TABLA_USUARIO = "usuario";
    public static final String CAMPO_ID = "id";
    public static final String CAMPO_NOMRE = "nombre";
    public static final String CAMPO_TELEFONO = "telefono";
    public static final String CREAR_TABLA_USUARIO = "CREATE TABLE " + TABLA_USUARIO
            + "(" + CAMPO_ID + " INTEGER, " + CAMPO_NOMRE + "  TEXT, " + CAMPO_TELEFONO + " TEXT"
            + ")";
}
```

Retornamos a la clase ConexionSQliteHelper y la dejamos como se muestra:

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

import androidx.annotation.Nullable;

import utilidades.Utilidades;

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 versionAntigua, int versionNueva) {
        db.execSQL("DROP TABLE IF EXISTS usuarios");
        onCreate(db);
    }
}
```

8. Insertar los datos a SQLite:

Agregamos la actividad denominada InsertarUsuario, y al archivo activity_insertarUsuario.xml el siguiente código:

```
<?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=".InsertarUsuario">

    <TextView
        android:id="@+id/textView"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:gravity="center"
        android:text="Registrar usuario"
```

```
        android:textSize="24sp" />

    <EditText
        android:id="@+id/edtIUid"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="14"
        android:inputType="textPersonName"
        android:hint="Documento" />

    <EditText
        android:id="@+id/edtIUnom"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="14"
        android:inputType="textPersonName"
        android:hint="Nombre" />

    <EditText
        android:id="@+id/edtIUtele"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="14"
        android:inputType="textPersonName"
        android:hint="telefono" />

    <Button
        android:id="@+id/btnIUSregusuario"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="registrar usurio" />
</LinearLayout>
```

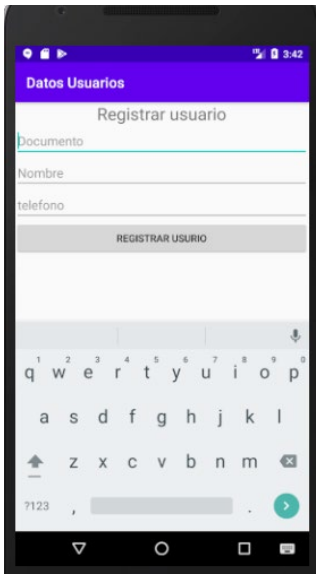
Se obtiene lo siguiente:

9. **Finalmente implementamos el activity Insertarusuario.**

```
public class InsertarUsuario extends AppCompatActivity implements View.OnClickListener {
EditText edtId, edtNom, edttel;
Button btninsertar;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_insertar_usuario);
        edtId=findViewById(R.id.edtIUid);
        edtNom=findViewById(R.id.edtIUnom);
        edttel=findViewById(R.id.edtIUtele);
        btninsertar=findViewById(R.id.btnIUSregusuario);
        btninsertar.setOnClickListener(this);
    }
    @Override
    public void onClick(View view) { registrarUsuarios(); }
    private void registrarUsuarios() {
        ConexionSqlLiteHelper conn=new ConexionSqlLiteHelper( context: this, name: "bd_usuarios", factory: null,  version: 1);
        SQLiteDatabase db= conn.getWritableDatabase();
        ContentValues values = new ContentValues();
        values.put(Utilidades.CAMPO_ID,edtId.getText().toString());
        values.put(Utilidades.CAMPO_NOMRE,edtNom.getText().toString());
        values.put(Utilidades.CAMPO_TELEFONO, edttel.getText().toString());
        Long idresultado= db.insert(Utilidades.TABLA_USUARIO, Utilidades.CAMPO_ID, values);
        Toast.makeText( context: this,  text: "Id de registro" + idresultado, Toast.LENGTH_SHORT).show();
        db.close();
    }
}
```

10. Modificamos el activity_main, para que pueda llamar a InsertarUsuarios:

```
public class MainActivity extends AppCompatActivity {
Button btnInsusuario, btnConsulUsuario;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        btnInsusuario=findViewById(R.id.btnInsUsuario);

    }
    public void llamarInsUsuario(View v){
        Intent i= new Intent( packageContext: this, InsertarUsuario.class);
        startActivity(i);
    }
```



Comprobar mediante SQLiteDatabaseBrowserPortable que se ha registrado el ingreso.

11. Consultar datos de la BD SQLite:

Agregar el activity ConsultarUsuario, en su respectivo activity_ConsultarUsuario.xml agregar la siguiente codificación:

```
<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=".ConsultarUsuario">

    <TextView
        android:id="@+id/textView2"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="Buscar usuario"
        android:textSize="30sp" />

    <EditText
        android:id="@+id/edtCUid"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textPersonName"
        android:hint="Documento" />

    <EditText
        android:id="@+id/edtCUnombre"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:inputType="textPersonName"
        android:hint="Nombre"/>

    <EditText
        android:id="@+id/edtCUTel"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="14"
        android:inputType="textPersonName"
        android:hint="Telefono" />

    <Button
        android:id="@+id/btnconsulUsuario"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Buscar" />

</LinearLayout>
```

12. El layout debería quedar como se muestra:

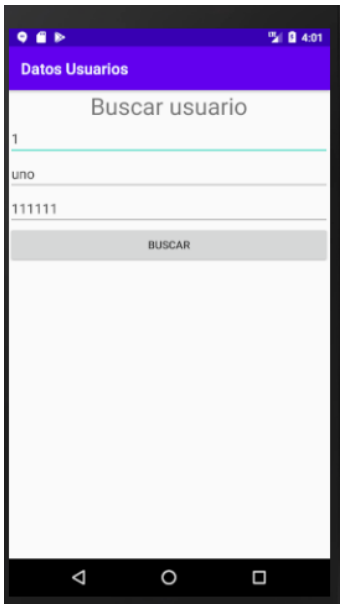
a. El código del activity ConsultarUsuario.java debe quedar como se muestra:

```
public class ConsultarUsuario extends AppCompatActivity implements View.OnClickListener {
    EditText edtId, edtNom, edttel;
    Button btnbuscar;
    ConexionSQLiteHelper conn;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_consultar_usuario);
        edtId=findViewById(R.id.edtCUid);
        edtNom=findViewById(R.id.edtCUnombre);
        edttel=findViewById(R.id.edtCUTel);
        btnbuscar=findViewById(R.id.btnconsulUsuario);
        btnbuscar.setOnClickListener(this);
        conn=new ConexionSQLiteHelper(this, "bd_usuarios",null,1);
    }

    @Override
    public void onClick(View view) {
        consultar();
    }

    private void consultar() {
        SQLiteDatabase db= conn.getReadableDatabase();
        String[] parametros={edtId.getText().toString()};
        String[] campos={Utilidades.CAMPO_NOMRE,Utilidades.CAMPO_TELEFONO};
        try{
            Cursor cursor=db.query(Utilidades.TABLA_USUARIO,campos,
                    Utilidades.CAMPO_ID+"=?",parametros,null,
                    null,null);
            cursor.moveToFirst();
            edtNom.setText(cursor.getString(0));
            edttel.setText(cursor.getString(1));
            cursor.close();
        }
        catch (Exception e){
            Toast.makeText(this, "El registro no existe", Toast.LENGTH_SHORT).show();
        }
    }
}
```

b. Modificamos el activity_main para que pueda mostrar a ConsultarUsuario.

```
public class MainActivity extends AppCompatActivity {
    Button btnInsusuario, btnConsulUsuario;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        btnInsusuario = findViewById(R.id.btnInsUsuario);

    }

    public void llamarInsUsuario(View v) {
        Intent i = new Intent(this, InsertarUsuario.class);
        startActivity(i);
    }

    public void llamarConsultar(View v) {
        Intent i = new Intent(this, ConsultarUsuario.class);
        startActivity(i);
    }
}
```

Resultado de la búsqueda:



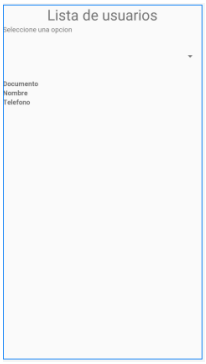
13. Consultar con un spinner la información de la base de datos

Agregar una nueva actividad denominada ListarUsersSpinners, y el siguiente código en su layout respectivo.

```
<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=".listarUsersSpinner">
    <TextView
        android:id="@+id/textView3"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:text="Lista de usuarios"
        android:textSize="30sp" />
    <TextView
        android:id="@+id/textView4"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Seleccione una opcion" />
    <Spinner
        android:id="@+id/cmbUsuarios"
        style="@style/Widget.AppCompat.Spinner.DropDown.ActionBar"
        android:layout_width="match_parent"
        android:layout_height="93dp"
        android:spinnerMode="dialog" />
    <TextView
        android:id="@+id/txtDocumento"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Documento"
        android:textStyle="bold"/>
    <TextView
        android:id="@+id/txtNombre"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Nombre"
        android:textStyle="bold"/>
    <TextView
        android:id="@+id/txtTelefono"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Telefono"
        android:textStyle="bold"/>

</LinearLayout>
```

14. **El diseño resultante debe verse como:**



15. **Modificar el código en el activity ListaUsersSpinner**

```
public class ListarUsersSpinner extends AppCompatActivity {

   Spinner cmbPersonas;
```

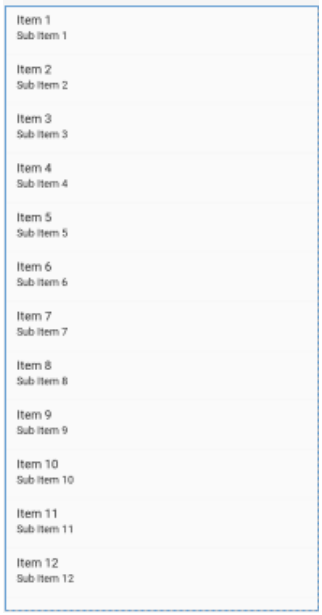
```
    TextView txtId, txtNom, txtTel;
    ArrayList<String> listaUsuarios;
    ArrayList<Usuario> UsuariosList;
    ConexionSQLiteHelper conn;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.activity_listar_users_spinner);
   cmbPersonas=findViewById(R.id.cmbUsuarios);
   txtId=findViewById(R.id.txtDocumento);
   txtNom=findViewById(R.id.txtNombre);
   txtTel=findViewById(R.id.txtTelefono);
   conn= new ConexionSQLiteHelper(this,"bd_usuarios",null,1);
   consultarUsuarios();
   ArrayAdapter<CharSequence> adaptador= new
ArrayAdapter(this,android.R.layout.simple_spinner_item, listaUsuarios);
   cmbPersonas.setAdapter(adaptador);
   cmbPersonas.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
 @Override
 public void onItemSelected(AdapterView<?> parent, View view, int position, long idl) {
if (position !=0) {
    txtId.setText(UsuariosList.get(position - 1).getId().toString());
    txtNom.setText(UsuariosList.get(position - 1).getNombre().toString());
    txtTel.setText(UsuariosList.get(position - 1).getTelefono().toString());
}
else{
    txtId.setText("");
    txtNom.setText("");
    txtTel.setText("");
}
 }
 @Override
 public void onNothingSelected(AdapterView<?> adapterView) {
 }
  });
    }
    private void consultarUsuarios() {
   SQLiteDatabase db= conn.getReadableDatabase();
   Usuario usuario=null;
   UsuariosList = 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.setTelefono(cursor.getString(2));
 Log.i("id",usuario.getId().toString());
 Log.i("Nombre",usuario.getNombre().toString());
 Log.i("Telefono",usuario.getTelefono().toString());
 UsuariosList.add(usuario);
  }
   obtenerListaUsuarios();
    }
    private void obtenerListaUsuarios() {
   listaUsuarios= new ArrayList<String>();
   listaUsuarios.add("Seleccione");
   for ( int i=0;i<UsuariosList.size();i++ ){
 listaUsuarios.add(UsuariosList.get(i).getId()+" - "+ UsuariosList.get(i).getNombre()+" - "+
UsuariosList.get(i).getTelefono());
   }
    }
     }
```

16. **Consultar a un ListView desde una BD SQLite**

Agregar la actividad ListarUsuarios, implementar su layout considerando el siguiente código:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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=".listarUsuarios">

    <ListView
        android:id="@+id/lstUsuarios"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />
</RelativeLayout>
```

El layout queda así:



17. **Implementar la activity con el siguiente código:**

```
public class ListarUsuarios extends AppCompatActivity {
    ListView lstusers;
    ConexionSQLiteHelper conn;
    ArrayList<String> listaInformacion;
    ArrayList<Usuario> listaUsuarios;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_listar_usuarios);
        lstusers=findViewById(R.id.lstUsuarios);
        conn= new ConexionSQLiteHelper(this,"bd_usuarios",null,1);
        consultarListaUsuarios();
        ArrayAdapter adaptador = new ArrayAdapter(this, android.R.layout.simple_list_item_1,
listaInformacion);
        lstusers.setAdapter(adaptador);
        lstusers.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> adapterView, View view, int i, long l) {

                String informacion="Id: " + listaUsuarios.get(i).getId()+"\n";
                informacion+="Nombre: "+ listaUsuarios.get(i).getNombre()+"\n";
```

```
                informacion+="telefono: " + listaUsuarios.get(i).getTelefono()+"\n";
                Toast.makeText(ListarUsuarios.this, informacion, Toast.LENGTH_SHORT).show();
            }
        });
    }

    private void consultarListaUsuarios() {
        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.setTelefono(cursor.getString(2));
            listaUsuarios.add(usuario);
        }
        obtenerLista();
    }

    private void obtenerLista() {
        listaInformacion= new ArrayList<String>();
        //listaInformacion.add("Documen   Nombre    Telefono");
        for (  int i=0; i<listaUsuarios.size();i++){
            listaInformacion.add(listaUsuarios.get(i).getId()+" - "+listaUsuarios.get(i).getNombre() + " -
" + listaUsuarios.get(i).getTelefono());


        }
    }
}
```

Semana 7 – Sesión 2

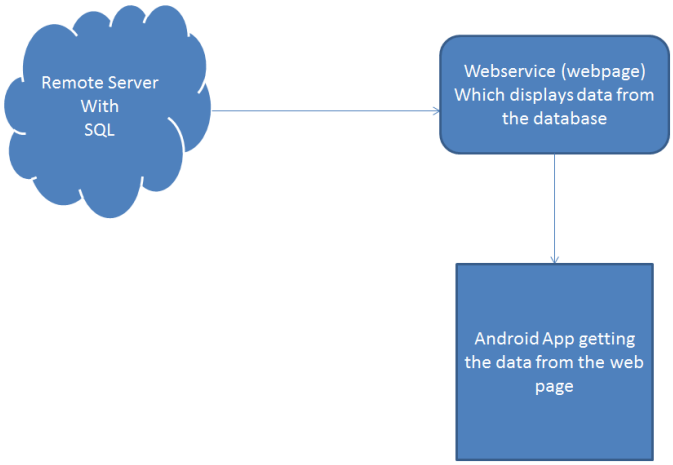
Base de datos de acceso remoto

Sección:	Apellidos : .....
Docente : Pedro Yuri Marquez Solis	Nombres : .....
Unidad : Unidad 2	Fecha: ...../...../..... Duración: 60 min

I. Propósito: Implementar una app básica con acceso a base de datos remota

II. Descripción de la actividad a realizar (caso.)

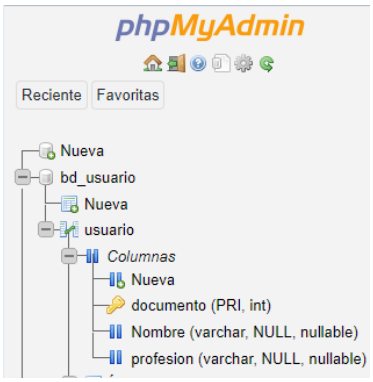
Cuando las aplicaciones móviles se conectan a un servidor web, normalmente no buscan obtener un documento web, sino que lo que hacen es acceder a servicios.
Estos servicios nos pueden servir para realizar alguna operación en el servidor, o para obtener información en un formato que nuestra aplicación sea capaz de entender y de procesar.



III. Procedimientos

Instrucciones:

1. **Creamos la base de datos base de datos *bd_usuario* y la tabla *usuario* en un servidor mysql, con la estructura de la tabla que se muestra**



Debes ingresar al menos 2 registros.

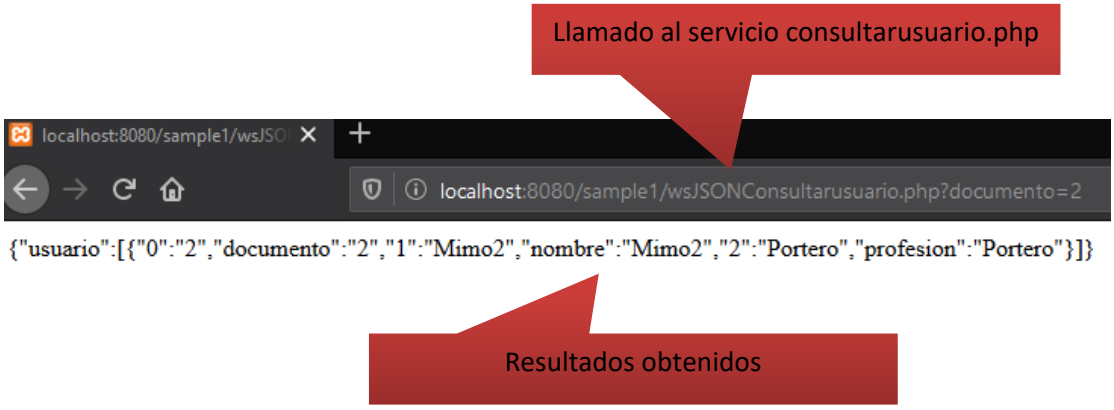
2. **Webservice:**

   Debes tener instalado un servidor web **Apache , Php y MySql.**

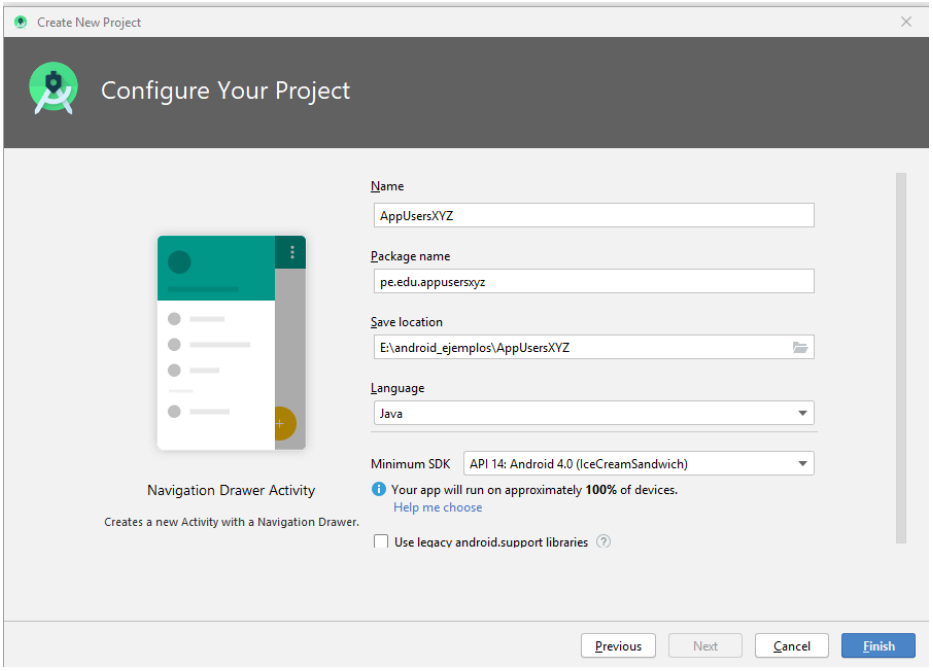
   A continuación vamos a crear el webservice en php, para ello, en la estructura de archivos de un servidor apache –php, por ejemplo en **C:\xampp\htdocs** creamos un directorio denominado **sample1** y dentro de ese directorio creamos el scripts php consultarusuario.php

```
<?php
$hostname_localhost="localhost";
$database_localhost="bd_usuario";
$username_localhost="root";
$password_localhost="";

$json=array();

    if(isset($_GET["documento"])){
        $documento=$_GET['documento'];


$conexion=mysqli_connect($hostname_localhost,$username_localhost,$password_localhost,$dat
abase_localhost);

        $consulta="select documento, nombre, profesion from usuario where documento=
'{$documento}'";
        $resultado=mysqli_query($conexion,$consulta);

        if($registro=mysqli_fetch_array($resultado)){
            $json['usuario'][]=$registro;
        }
        else{
            $resultar["documento"]=0;
            $resultar["nombre"]='No registra';
            $resultar["profesion"]='No registra';
            $json['usuario'][]=$resultar;
            }
        mysqli_close($conexion);
        echo json_encode($json);
    }
    else{
            $resultar["success"]=0;
            $resultar["message"]='WS no retorna';
            $json['usuario'][]=$resultar;
            echo json_encode($json);
        }
?>
```

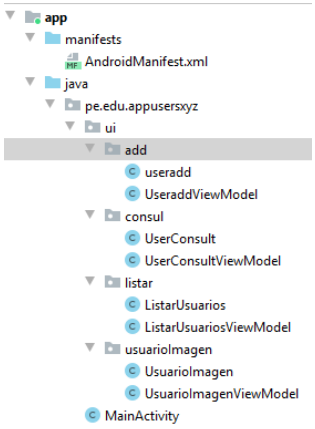
3. **Pruebas desde el browser:**
   Puedes realizar las pruebas del script de consulta del siguiente modo:



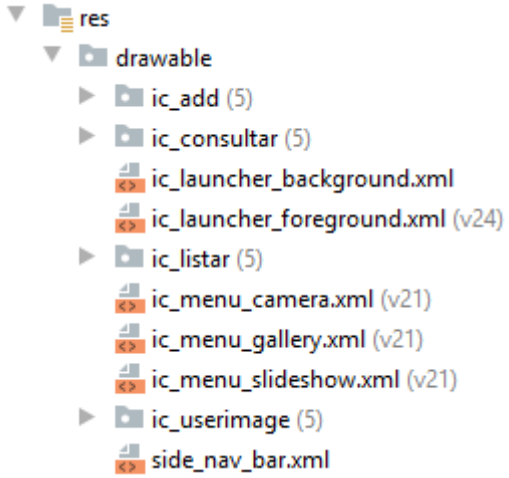
4. **Continuando, vamos creando un proyecto navDreawer, al cual denominaremos AppUsersXYZ**



16. **Agregar fragments, cuidando organizarlos con packages como se muestra:**



17. Agregar los elementos gráficos del menú:



5. Activity_main_drawer.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    tools:showIn="navigation_view">

    <group android:checkableBehavior="single">
        <item
            android:id="@+id/nav_add"
            android:icon="@drawable/ic_add"
            android:title="Insertar usuario" />
        <item
            android:id="@+id/nav_consultar"
            android:icon="@drawable/ic_consultar"
            android:title="Consultar usuario" />
        <item
            android:id="@+id/nav_listar"
            android:icon="@drawable/ic_listar"
            android:title="Listar usuario" />
        <item
            android:id="@+id/nav_userimage"
            android:icon="@drawable/ic_userimage"
            android:title="Listar usuario" />
    </group>
</menu>
```

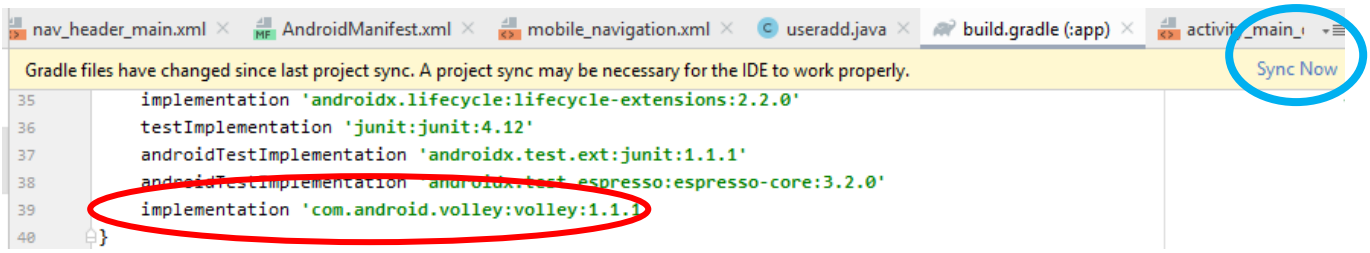
6. **Modificar el mobile_navigation.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<navigation 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:id="@+id/mobile_navigation"
    app:startDestination="@+id/nav_add">

    <fragment
        android:id="@+id/nav_add"
        android:name="pe.edu.appusersxyz.ui.add.useradd"
        android:label="Insertar"
        tools:layout="@layout/user_add_fragment">

    </fragment>
    <fragment
        android:id="@+id/nav_consultar"
        android:name="pe.edu.appusersxyz.ui.consul.UserConsult"
        android:label="Consultar"
        tools:layout="@layout/user_consult_fragment">
    </fragment>

    <fragment
        android:id="@+id/nav_listar"
        android:name="pe.edu.appusersxyz.ui.listar.ListarUsuarios"
        android:label=  "Listar"
        tools:layout="@layout/listar_usuarios_fragment" />

    <fragment
        android:id="@+id/nav_userimage"
        android:name="pe.edu.appusersxyz.ui.usuarioImagen.UsuarioImagen"
        android:label="Usuario imagen"
        tools:layout="@layout/usuario_imagen_fragment" />
</navigation>
```

7. **Agregar en el manifest el permiso de acceso a internet.**

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="pe.edu.fragmentdinamico">
    <uses-permission android:name="android.permission.INTERNET"></uses-permission>
    <application
```

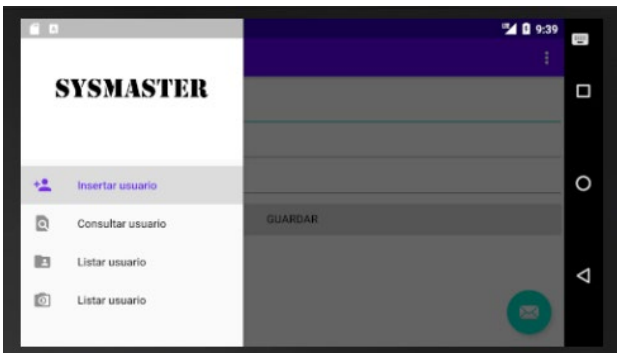
8. **Agregar la referencia a la implementación de la librería volley, no olvides sincronizar el proyecto dando click a SyncNow**



9. **Agregar los View a user_consult_fragment.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ui.consul.UserConsult">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">
    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="#F44336"
        android:gravity="center"
        android:text="Consultar usuario"
        android:textColor="#FFCCBC"
        android:textSize="36sp" />
    <TextView
        android:id="@+id/textView2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Documento"
        android:textColor="#EF5350"
        android:textSize="30sp" />
    <EditText
        android:id="@+id/camDoc"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:ems="10"
        android:hint="Documento"
        android:inputType="textPersonName" />
    <TextView
        android:id="@+id/txtNombre"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="24sp" />
    <TextView
        android:id="@+id/txtProfesion"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="24sp" />
    <Button
        android:id="@+id/btnConsutar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="consultar"
        android:background="@color/colorAccent"
        android:textColor="@android:color/white"/>
    </LinearLayout>
</FrameLayout>
```

10. **La ejecución de prueba debe mostrar un diseño similar al indicado:**



11. Implementar las clases java+

```
public class UserConsult extends Fragment  implements
Response.Listener<JSONObject>,Response.ErrorListener {
    EditText camDocu;
    TextView txtNom, txtPro;
    Button btnBuscar;
    ProgressDialog progreso;
    RequestQueue request;
    JsonObjectRequest jsonObjectRequest;
    private OnFragmentInteractionListener mListener;
    public UserConsult() {
        // Required empty public constructor
    }
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        View vista = inflater.inflate(R.layout.user_consult_fragment, container, false);
        camDocu = vista.findViewById(R.id.camDoc);
        txtNom = vista.findViewById(R.id.txtNombre);
        txtPro = vista.findViewById(R.id.txtProfesion);
        btnBuscar = vista.findViewById(R.id.btnConsutar);
        request = Volley.newRequestQueue(getContext());
        btnBuscar.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                cargarWebService();
            }


        });
        return vista;
    }

    private void cargarWebService() {
        progreso = new ProgressDialog(getContext());
        progreso.setMessage("consultando ...");
        progreso.show();
        String url = "http://192.168.1.14:8080/sample1/wsJSONConsultarUsuario.php?documento="
+ camDocu.getText().toString();
        url = url.replace(" ", "%20");
        jsonObjectRequest = new JsonObjectRequest(Request.Method.GET, url, null, this, this);
        request.add(jsonObjectRequest);
        Toast.makeText(getContext(), "Resultado", Toast.LENGTH_SHORT).show();
    }

    @Override
    public void onErrorResponse(VolleyError error) {
        progreso.hide();
        Toast.makeText(getContext(),"No se pudo consultar"+ error.toString(),
Toast.LENGTH_SHORT).show();
        Log.i("Error ", error.toString());
    }

    @Override
    public void onResponse(JSONObject response) {
        progreso.hide();
        Toast.makeText(getContext(), "Mensaje " + response, Toast.LENGTH_SHORT).show();
        Usuario miUsuario = new Usuario();
        JSONArray json = response.optJSONArray("usuario");
        JSONObject jsonObject = null;
        try {
            jsonObject = json.getJSONObject(0);
            miUsuario.setNombre(jsonObject.optString("nombre"));
            miUsuario.setProfesion(jsonObject.optString("profesion"));
            miUsuario.setNombre(jsonObject.optString("nombre"));
        } catch (JSONException e) {
            e.printStackTrace();
        }

        txtNom.setText(miUsuario.getNombre());
        txtPro.setText(miUsuario.getProfesion());
    }

    public interface OnFragmentInteractionListener {
        // TODO: Update argument type and name
        void onFragmentInteraction(Uri uri);
    }
}
```

Semana 8 – Sesión 1

Sección: ……..……..	Apellidos : ……..……..
Docente : Pedro Yuri Marquez Solis	Nombres : ……
Unidad : Unidad 2	Fecha: …../……/……. Duración: 60 min

Base de datos Remota 2

I. Propósito: Implementar una app básica con acceso a base de datos remota

II. Descripción de la actividad a realizar (caso.)

Continuamos con las operaciones CRUD a aplicar sobre la base de datos remota.

III. Procedimientos

Instrucciones:

1. **Modifica el diseño del layout userAdd.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<FrameLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".ui.add.add">
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">
        <TextView
            android:id="@+id/textView"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Agregar usuarios" />
        <EditText
            android:id="@+id/campoDoc"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:hint="Documento"
            android:inputType="textPersonName"
            android:textColor="#7ED4DF" />
        <EditText
            android:id="@+id/campoNom"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:hint="Nombre"
            android:inputType="textPersonName"
            android:textColor="#7ED4DF" />
        <EditText
            android:id="@+id/campoPro"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:ems="10"
            android:hint="Profesion"
```

```
            android:inputType="textPersonName"
            android:textColor="#7ED4DF" />
        <Button
            android:id="@+id/btnGuardar"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:text="Guardar" />
    </LinearLayout>
</FrameLayout>
```

2. **Se debe obtener un resultado en la interface como se muestra:**

3. **Apertura el proyecto de la semana 7, modifica el activity USERADD tal como se muestra**

```
public class add extends Fragment implements Response.ErrorListener,
Response.Listener<JSONObject> {
EditText txtnom,txtdoc,txtprof;
Button btnguardar;
ProgressDialog progreso;
RequestQueue request;
JsonObjectRequest jsonObjectRequest;


    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        View vista= inflater.inflate(R.layout.fragment_add, container, false);
        txtnom=  vista.findViewById(R.id.campoNom);
        txtdoc  =  vista.findViewById(R.id.campoDoc);
        txtprof=  vista.findViewById(R.id.campoPro);
        btnguardar = vista.findViewById(R.id.btnGuardar);
        request= Volley.newRequestQueue(getContext());
        btnguardar.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                cargarWebService();
            }
        });
        return vista;

    }

    private void cargarWebService() {
        progreso = new ProgressDialog(getContext());
        progreso.setMessage("Enviando datos");
        progreso.show();
```

```
        String url= "http://192.168.1.8/wsJSONRegistro.php?documento="+
txtdoc.getText().toString() +
                "&nombre="+
txtnom.getText().toString()+"&profesion="+txtprof.getText().toString() ;
        url=url.replace(" ", "%20");
        jsonObjectRequest = new JsonObjectRequest(Request.Method.GET, url, null, this,
this);
        request.add(jsonObjectRequest);
    }


    @Override
    public void onErrorResponse(VolleyError error) {
        progreso.hide();
        Toast.makeText(getContext(), "No se pudo insertar", Toast.LENGTH_SHORT).show();
        Log.i("Error", error.toString());
    }

    @Override
    public void onResponse(JSONObject response) {
        Toast.makeText(getContext(), "Regsitro exitoso", Toast.LENGTH_SHORT).show();
        progreso.hide();
        txtdoc.setText("");
        txtnom.setText("");
        txtprof.setText("");
    }
}
```

Tercera unidad

Semana 9- Sesión 2

Sección: ……..……..	Apellidos : ……..……
Docente : Pedro Yuri Marquez Solis	Nombres : ……
Unidad : Unidad 3	Fecha: …../……/……. Duración: 60 min

Geolocalización mediante GPS

I. Propósito: Emplear los servicios de GoogleCloud para geolocalizar un dispositivo móvil

II. Descripción de la actividad a realizar (caso.)

Para lograr la geolocalización se requiere triangular los valores obtenidos de al menos 3 satélites. De ahí podemos calcular tanto la Logitud y Latitud de un dispositivo.

Longitud (Meridianos)

Medida cartográfica que expresa la distancia angular desde cualquier punto de la superficie de la Tierra, con respecto al meridiano base o meridiano de Greenwich que se toma como el 0°, en dirección Este u Oeste.

Latitud (Paralelos)

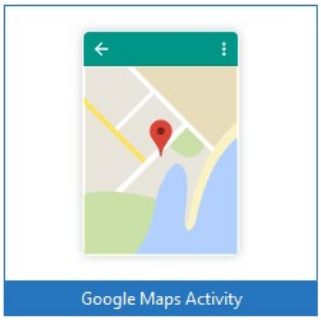
Distancia angular entre la línea ecuatorial de la Tierra y un punto específico de la Tierra que proporciona la localización de un punto en dirección Norte o Sur desde el ecuador y está expresada en medidas angulares desde los 0° hasta los 90°N del polo Norte o los 90°S del polo Sur.

En esta práctica elaboramos una app básica que permita geolocalizar un dispositivo.

III. Procedimientos

Instrucciones:

1. **Crear un nuevo proyecto denominado geolocalizacionXYZ.**
2. **Agrega una nueva Activity efectúa clic derecho sobre el directorio java Activity Gallery elige Google Maps Activity**

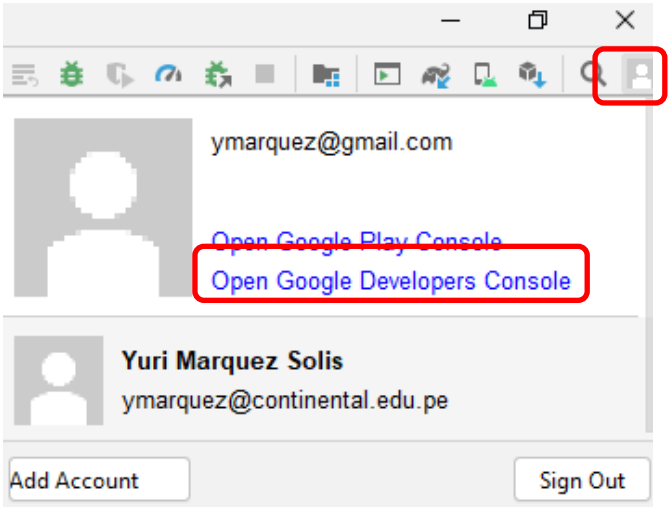


3. **Apertura el archivo MpasActivity que se ha agregado por defecto, encontraras un código similar al siguiente:**

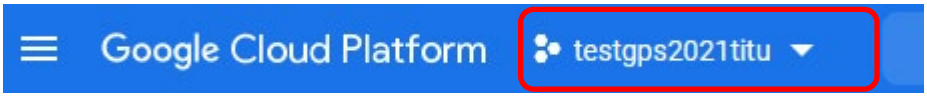
```
public class MapsActivity extends FragmentActivity implements OnMapReadyCallback {

    private GoogleMap mMap;
    private ActivityMaps2Binding binding;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        binding = ActivityMaps2Binding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());

        // Obtain the SupportMapFragment and get notified when the map is ready to be
used.
        SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
                .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);
    }


    @Override
    public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;

        // Add a marker in Sydney and move the camera
        LatLng sydney = new LatLng(-34, 151);
        mMap.addMarker(new MarkerOptions().position(sydney).title("Marker in Sydney"));
        mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));
    }
}
```

4. **Obtener el identificador sha del proyecto:**

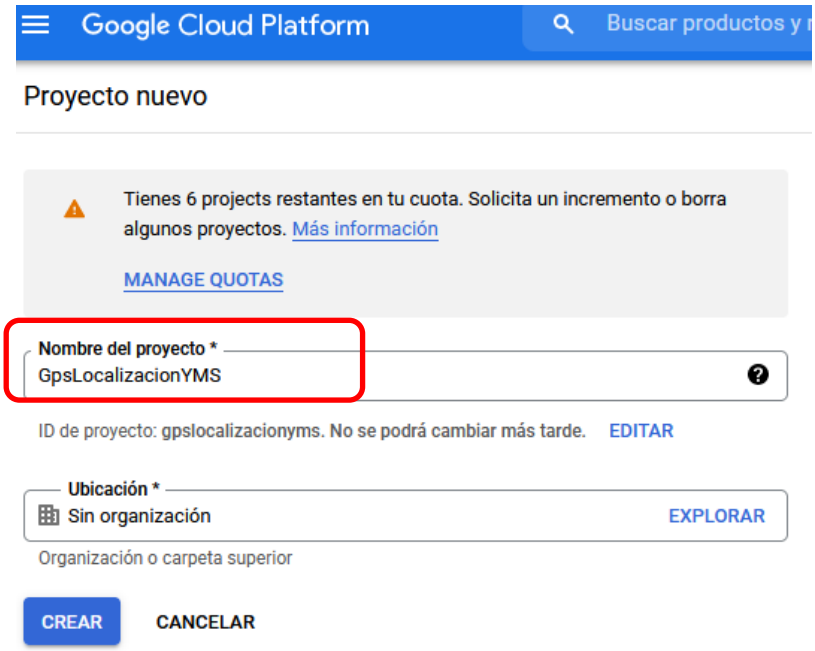
   Para ello debes autenticar tu sesión de Android Studio, en la esquina superior derecha encontraras un ícono desde el cual debes ingresar a una cuenta de Gmail. Se recomienda una diferente a la de la Universidad Continental



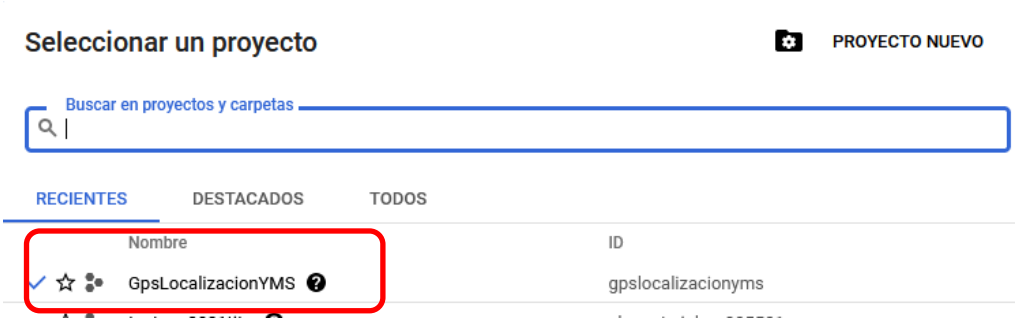
5. **Luego de autenticarte elige la opción Open Google Developers Console.**
6. **En la barra elige proyecto nuevo.**



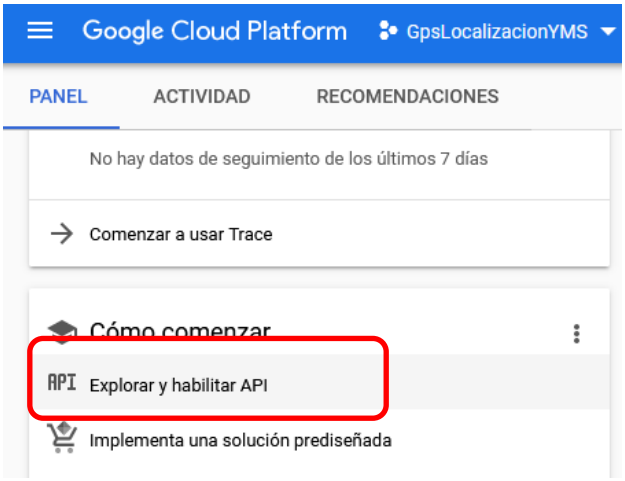
7. **Ingresa el nombre del proyecto**



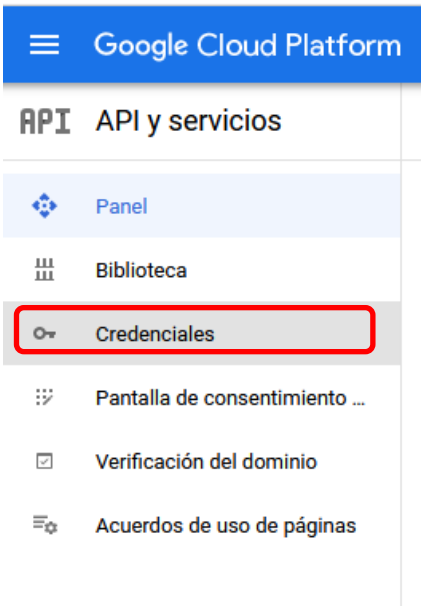
8. **Luego de crear el proyecto retorna a seleccionar proyecto y elige tu nuevo proyecto**



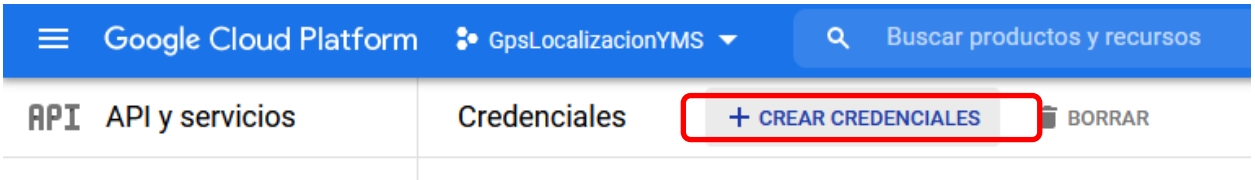
9. **En el panel de la derecha elige la opción Explorar y habilitar API:**

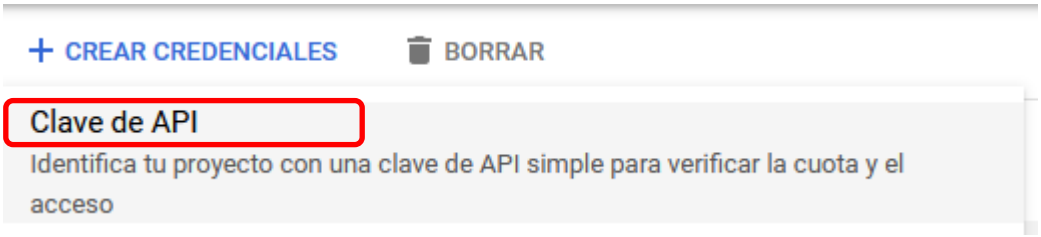


Luego la opción credenciales:



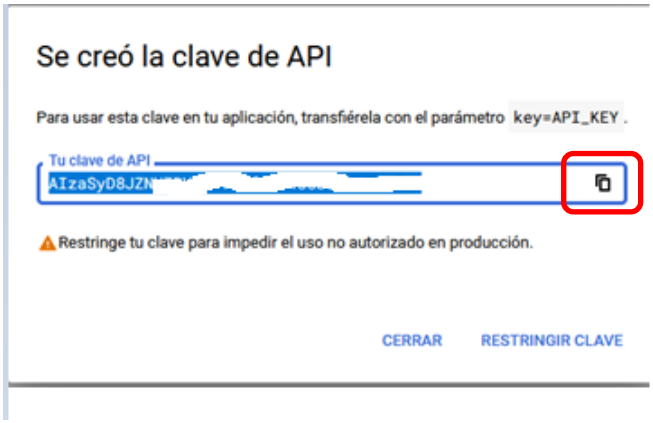
10. **Ahora elige la opción CREAR CREDENCIALES y clave de API**





11. **Se mostrará un cuadro indicando que se creó la clave de API, efectúa clic sobre el botón de copiar.**

}



12. **Modifica el archivo AndroidManifest.xml y pega la clave de API que has obtenido:**

```
<meta-data
    android:name="com.google.android.geo.API_KEY"
    android:value="AI[redacted]" />
```

13. **Ahora puedes ejecutar tu aplicación, la cual se mostrará por defecto con la ubicación de Sydney en Australia.**

Semana 10- Sesión 2

Sección:	Apellidos : .....
Docente : Pedro Yuri Marquez Solis	Nombres : .....
Unidad : Unidad 3	Fecha: ...../...../..... Duración: 60 min

Servicios Web con FireBase

I. Propósito: Utilizar los servicios de Firebase.

II. Descripción de la actividad a realizar (caso.)

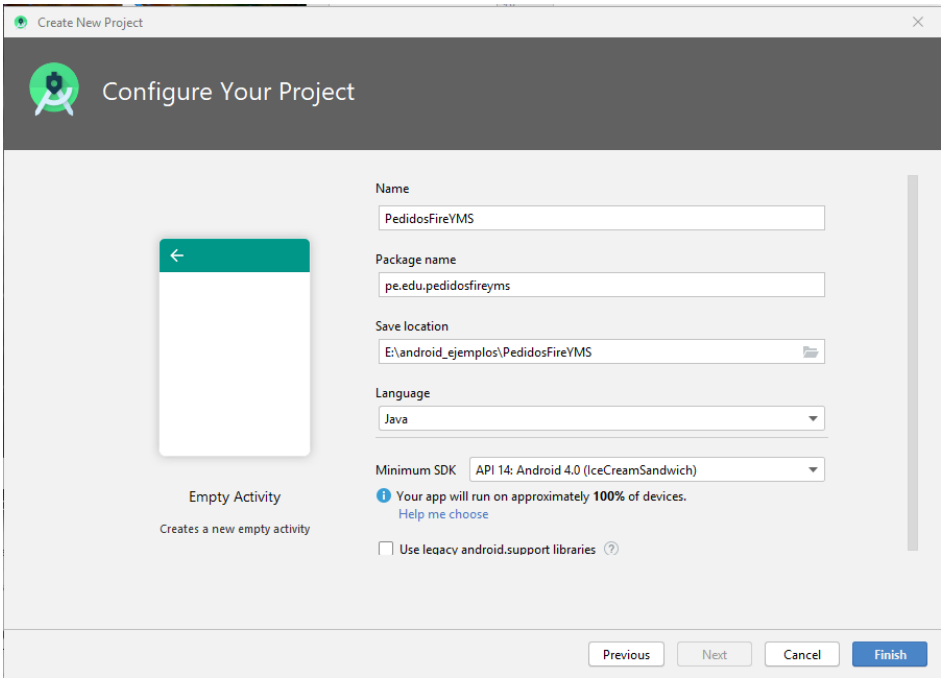
Firebasede Google es una plataforma en la nube para el desarrollo de aplicaciones web y móvil. Disponible para distintas plataformas (Android, iOS, Web, C++ ,Unity)con lo que es más rápido trabajar en el desarrollo

Firebasepuede englobarse como un servicio del tipo Mobile Backendas a Service(MBaaS).

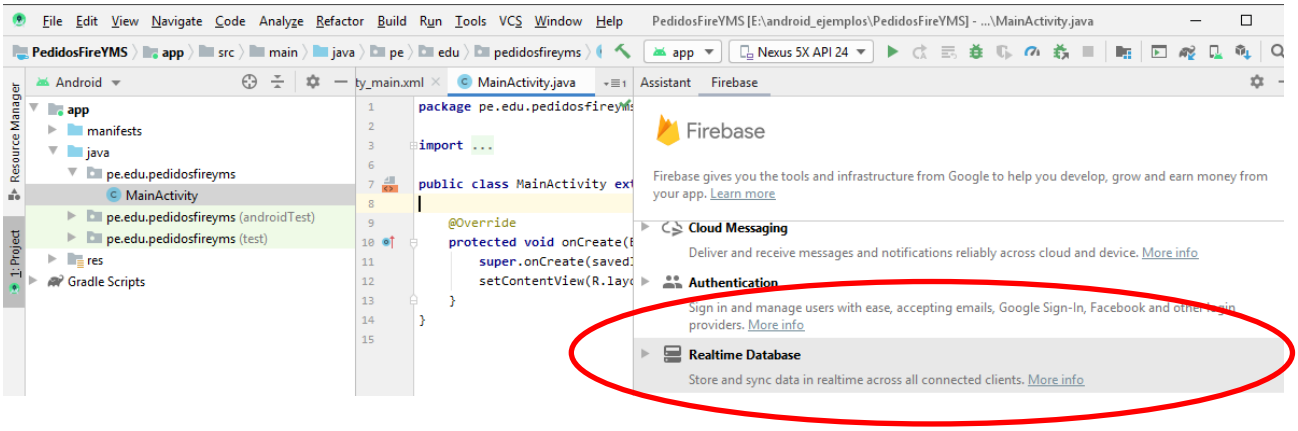
III. Procedimientos

Instrucciones:

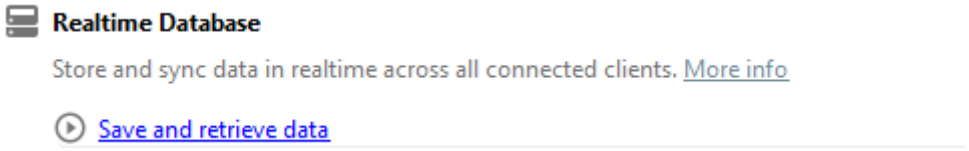
1. **Iniciamos creando un proyecto en blanco al cual denominaremos PedidosFire**



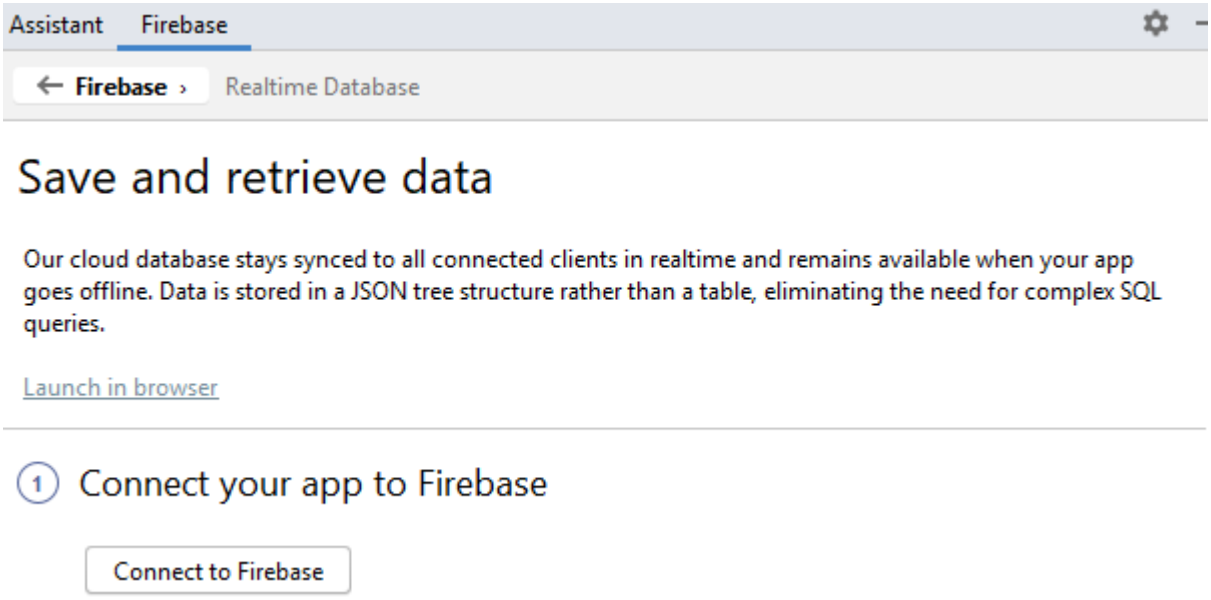
2. **Dirigirse al menú Tools, elegir la opción Firebase, se mostrará el lateral derecho como se muestra, elegir la opción RealTime eDataBase.**



Elegir la opción Save and retrieve data.

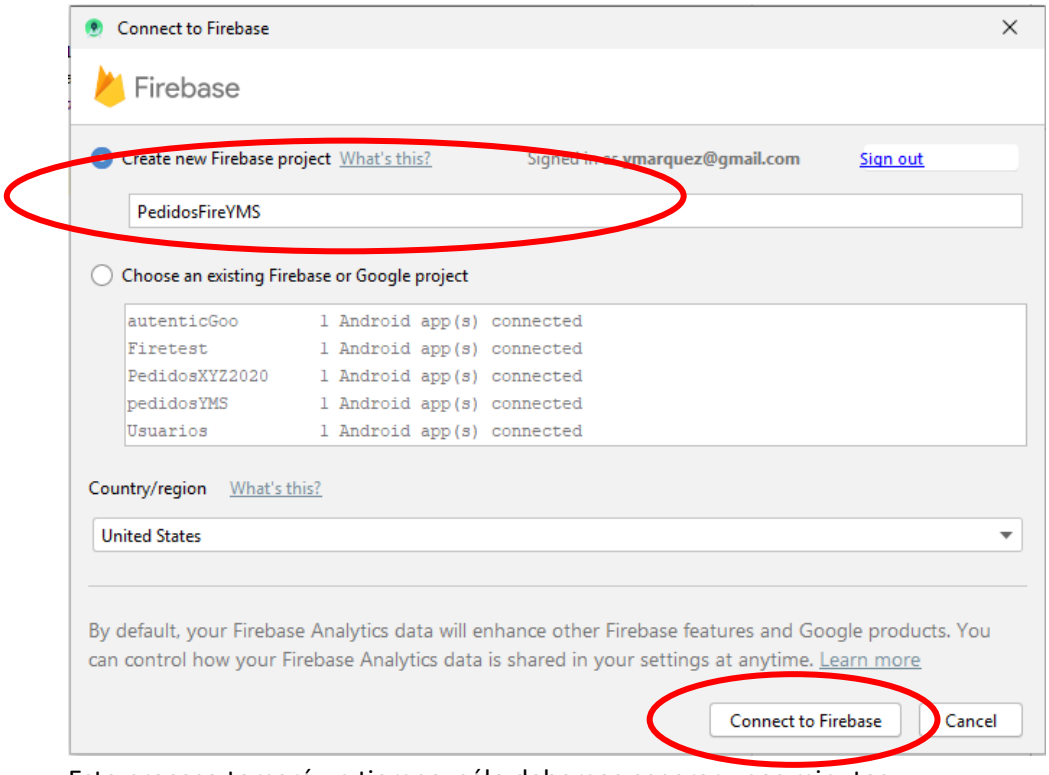


3. **Ahora se muestra la secuencia del proceso**: Iniciar con el primer paso con el botón Connect to FireBase.



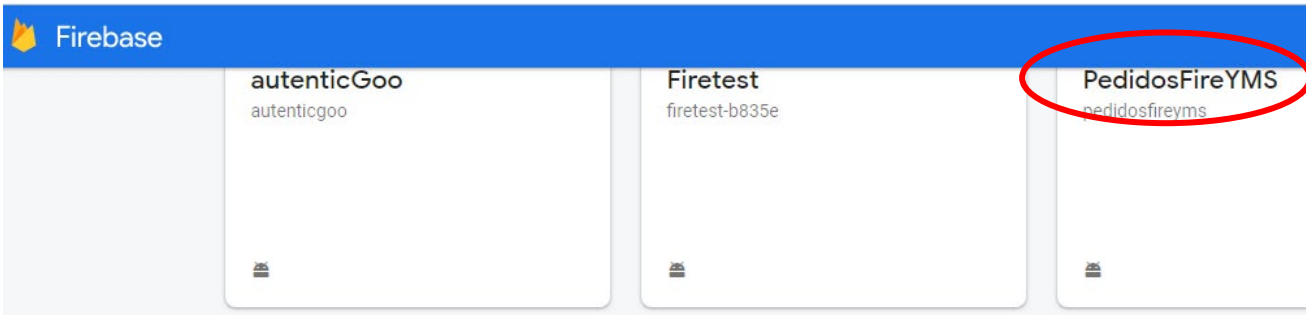
El proceso que se realizará activará (Connect to Firebase) creará un proyecto en el FireBase del desarrollador, el cual debe tener una cuenta de google obligatoriamente.

4. **Se mostrará una ventana que permite elegir el nombre del proyecto, darle click a Connect to Firebase**

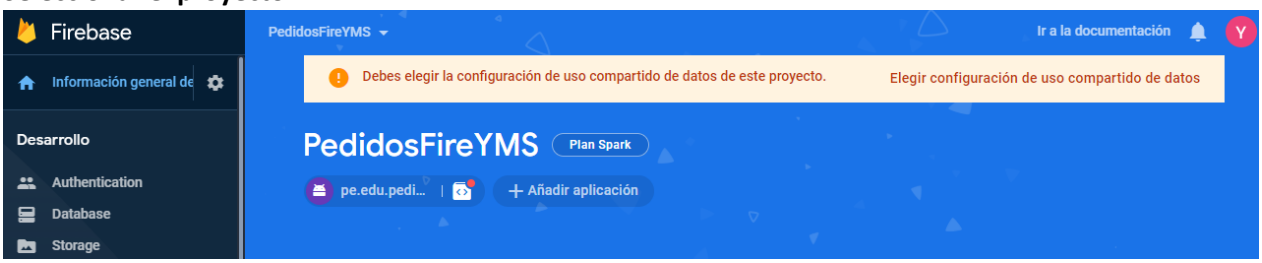


Este proceso tomará un tiempo, sólo debemos esperar unos minutos.

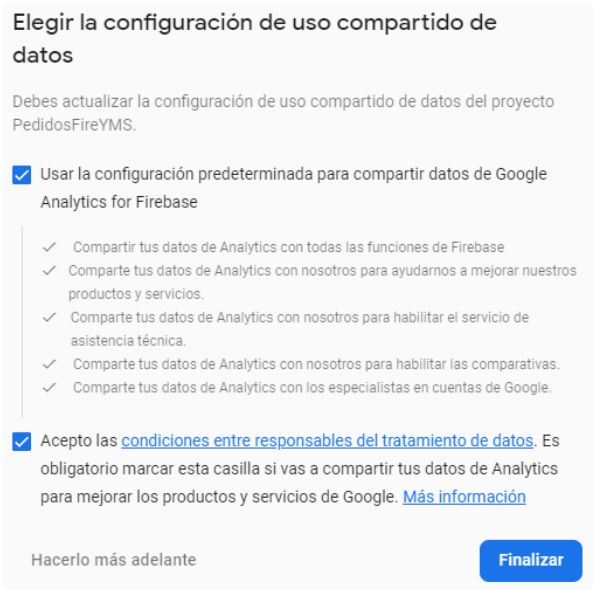
5. **Dirigirse a la consola de Firebase (en el buscador escribir firebase console) y visualizar la lista de proyectos:** Se mostrará la consola con el proyecto creado.



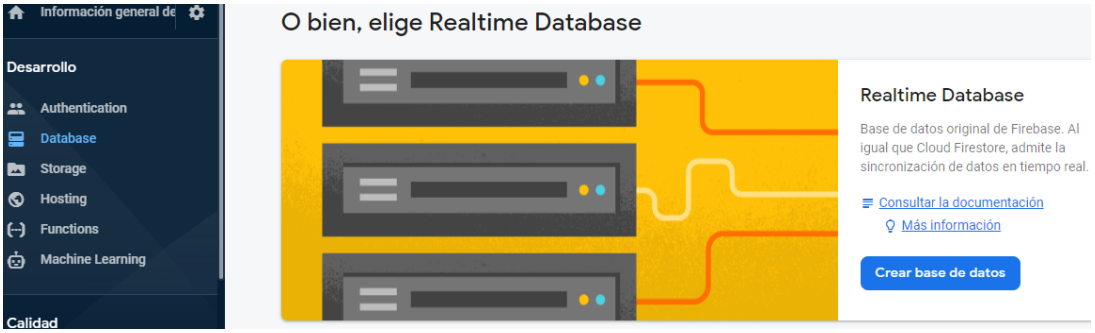
6. **Seleccionar el proyecto :**



Seleccionar la opción de uso compartido de datos:



7. **Ahora elegir en el menú lateral la opción dataBase y RealTime Database:**

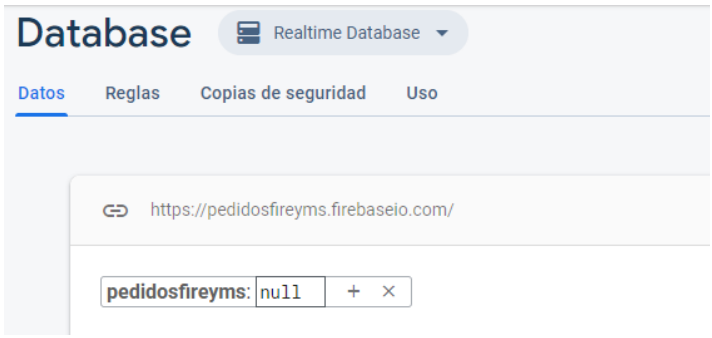


En la ventana que se muestra ahora elegir: Empezar con el modo de prueba.

Se mostrará el nodo raíz de la base de datos creada:

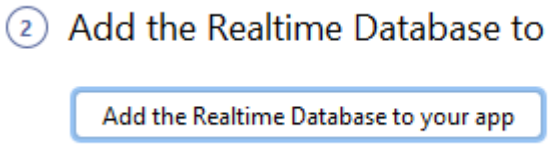
Modificar las reglas a:

```
{
  "rules": {
    ".read": "true",
    ".write": "true",
  }
}
```

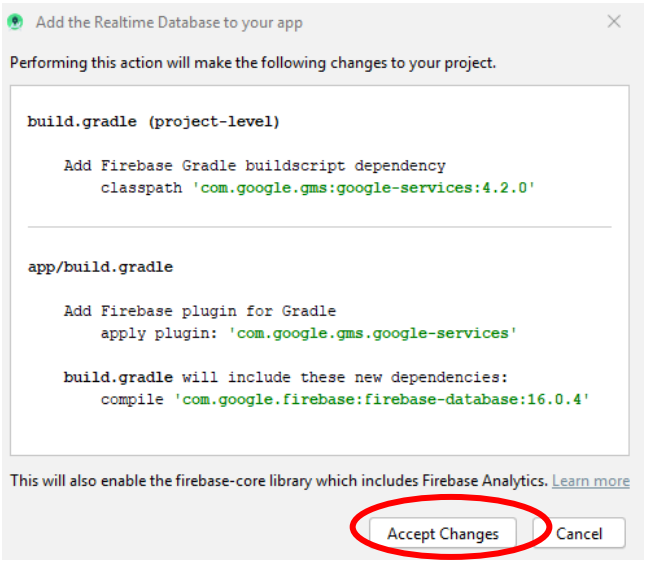


8. **Agregar la base de datos a la aplicación**

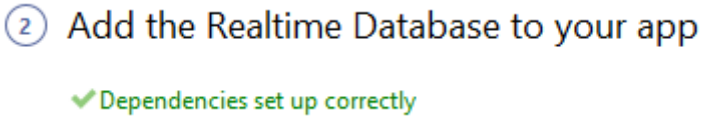
Retornamos al asistente y agregamos la base de datos a la aplicación.



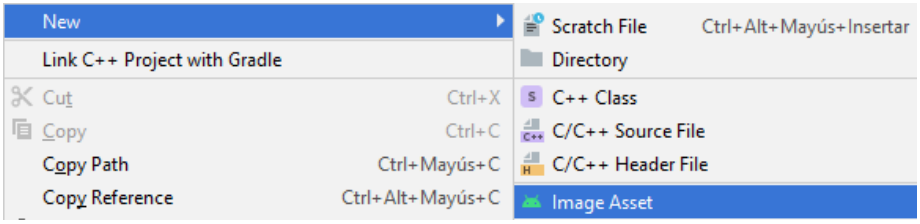
Se mostrará un cuadro de diálogo, darle clic a Accept Changes.



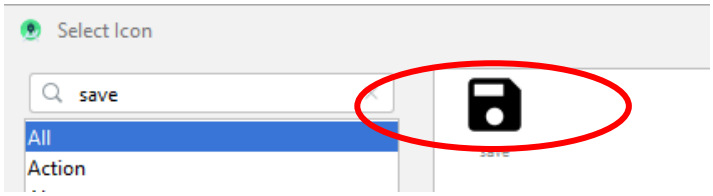
El asistente mostrará una respuesta como esta indicativa que ya se tiene la conexión con la base de datos del proyecto creado.



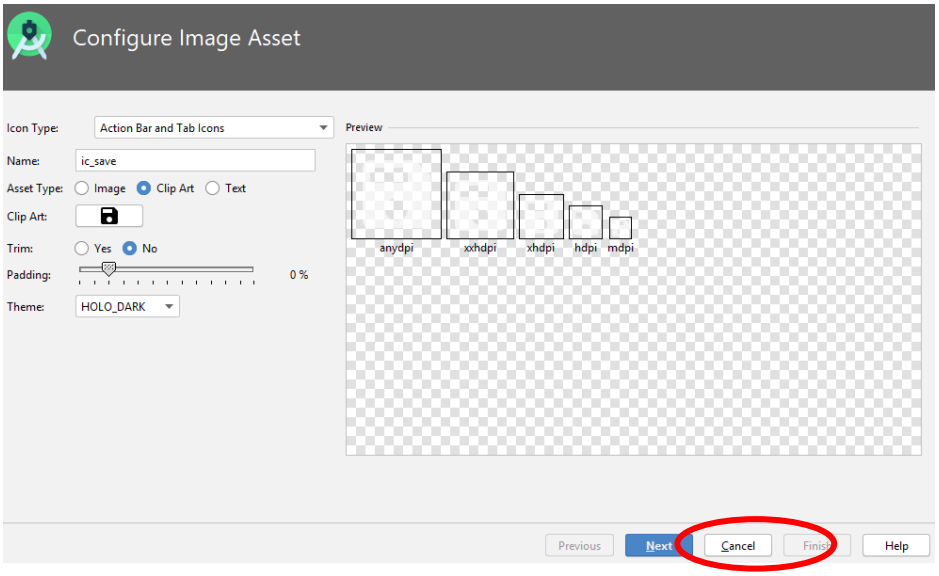
9. Obtenemos los elementos gráficos del menú:



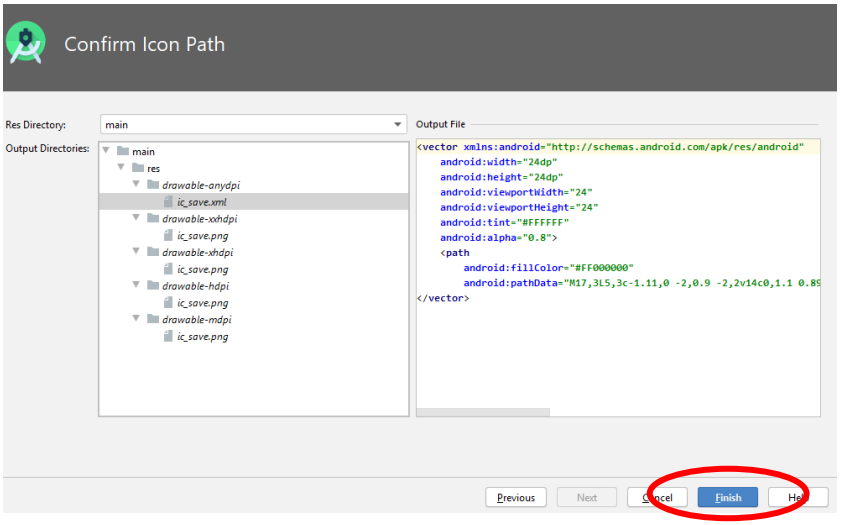
En la ventana que se muestra elegir: en Icon Type: Action Bar and tab Icons, y en el name ingresar por ejemplo ic_save. Se sugieren : ic_add, ic_update, ic_list para el resto de recursos icon.
Para asignar el ícono efectuar clic al botón ClipArt y efectuar la búsqueda, elegir el icon a asociar.



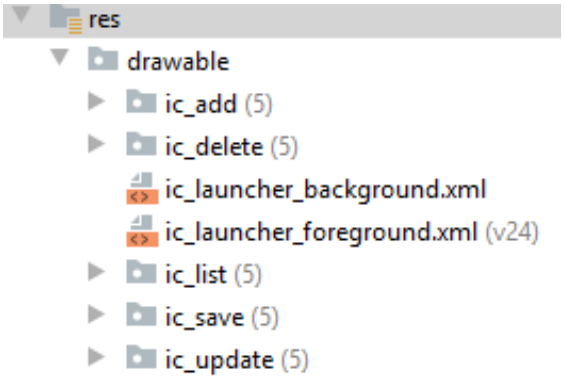
Se mostrarán las versiones del icon elegido, darle clic a next:



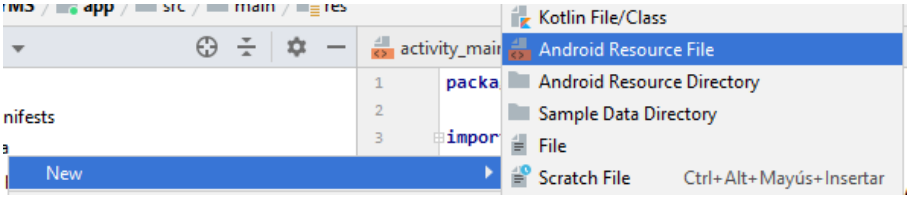
Finalmente confirmar la creación del recurso xml:



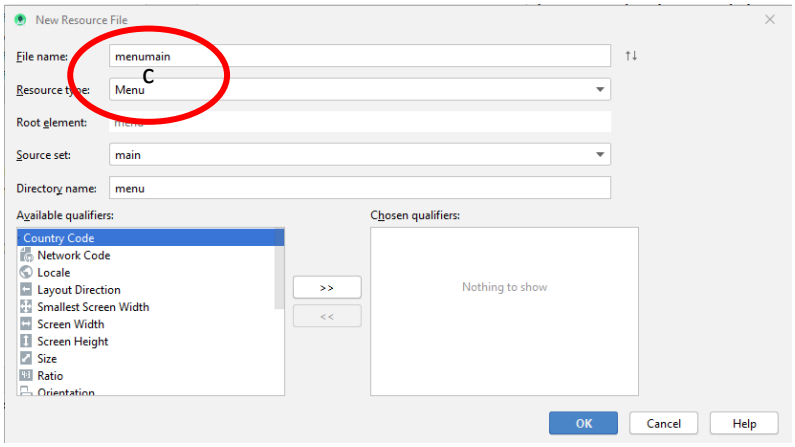
10. Repetir el proceso hasta obtener en res los siguientes recursos:



11. Ahora vamos a crear el menú: para ello crea un nuevo Android Resource File



Ingresar los siguientes datos:

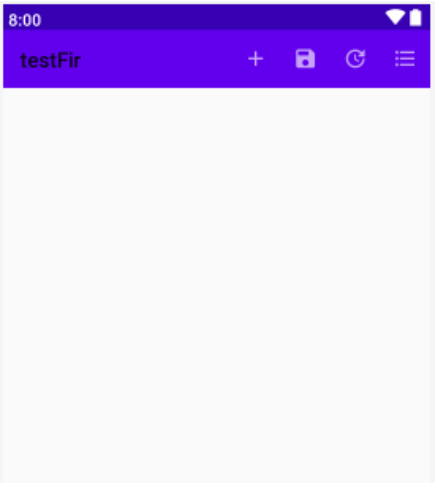


Click en Ok.

12. Crear el menú:

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item android:id="@+id/icon_add"
        android:icon="@drawable/ic_add"
        android:title="Agregar"
        app:showAsAction="always">
    </item>
    <item android:id="@+id/icon_save"
        android:icon="@drawable/ic_save"
        android:title="Guardar"
        app:showAsAction="always">
    </item>
    <item android:id="@+id/icon_update"
        android:icon="@drawable/ic_update"
        android:title="Actualizar"
        app:showAsAction="always">
    </item>
    <item android:id="@+id/icon_list"
        android:icon="@drawable/ic_list"
        android:title="Listar"
        app:showAsAction="always">
    </item>
</menu>
```

El menu creado deberá apreciarse como se muestra:



13. Insertar el menú en el MainActivity:

```
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.menu1, menu);
        return super.onCreateOptionsMenu(menu);
    }

    @Override
    public boolean onOptionsItemSelected(@NonNull MenuItem item) {
        switch (item.getItemId())
        {
            case R.id.icon_add:{
                Toast.makeText(this, "agregar", Toast.LENGTH_SHORT).show();
                break;
            }
            case R.id.icon_save:{
                Toast.makeText(this, "guardar", Toast.LENGTH_SHORT).show();
                break;
            }
        }
        return super.onOptionsItemSelected(item);
    }
```

14. Implementación de la clase Pedido:

```
public class Pedidos {

    private String idpedido;
    private String tipoped;
    private String distrito;
    private String descripcion;
    private String entregar;
    private String direccion;


    public Pedidos() {
    }

    public Pedidos(String idpedido, String tipoped, String distrito, String descripcion, String
entregar, String direccion) {
        this.idpedido = idpedido;
        this.tipoped = tipoped;
        this.distrito = distrito;
        this.descripcion = descripcion;
        this.entregar = entregar;
        this.direccion = direccion;
    }

    public String getIdpedido() {
        return idpedido;
    }

    public void setIdpedido(String idpedido) {
        this.idpedido = idpedido;
    }

    public String getTipoped() {
        return tipoped;
    }

    public void setTipoped(String tipoped) {
        this.tipoped = tipoped;
    }

    public String getDistrito() {
        return distrito;
    }

    public void setDistrito(String distrito) {
        this.distrito = distrito;
    }
```

```
        public String getDescripcion() {
            return descripcion;
        }

        public void setDescripcion(String descripcion) {
            this.descripcion = descripcion;
        }

        public String getEntregar() {
            return entregar;
        }

        public void setEntregar(String entregar) {
            this.entregar = entregar;
        }

        public String getDireccion() {
            return direccion;
        }

        public void setDireccion(String direccion) {
            this.direccion = direccion;
        }
}
```

15. Implementar la clase MyFireBaseApp

Con esta clase lograremos la sincronización de la aplicación cuando estemos operando en modo desconectado

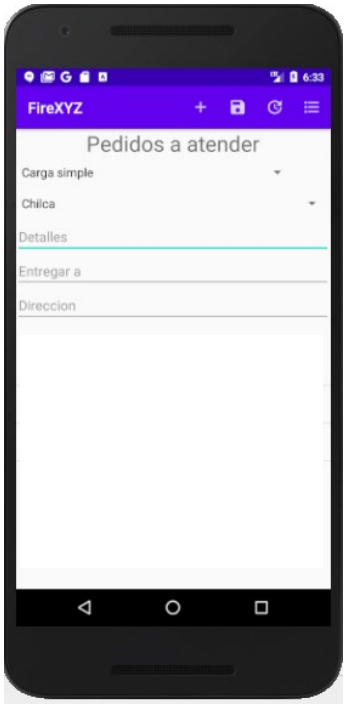
```
import com.google.firebase.database.FirebaseDatabase;

public class MyFireBaseApp  extends android.app.Application{
    @Override
    public void onCreate() {

        super.onCreate();
        FirebaseDatabase.getInstance().setPersistenceEnabled(true);
    }
}
```

16. Agregar los datos a mostrar en las listas:

```
<resources>
    <string name="app_name">PedidosXYZ2020</string>
    <string-array name="tiposped">
        <item>"Carga simple"</item>
        <item>"Carga pesada"</item>
        <item>"Sobre"</item>
    </string-array>
    <string-array name="distritos">
        <item>"Chilca"</item>
        <item>"El tambo"</item>
        <item>"Huancayo"</item>
        <item>"Chupaca"</item>
    </string-array>
    <string name="mensa_tipoPed">
        "Indique el tipo de pedido"
    </string>
    <string name="mensa_distrito">
        "Elija el distrito"
    </string>

</resources>
```

17. Definir el Layout del MainActivity:

```
<?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:gravity="center"
        android:text="Pedidos a atender"
        android:textSize="28dp"></TextView>

    <Spinner
        android:id="@+id/spTipoPed"
        android:layout_width="366dp"
        android:layout_height="38dp"
        android:entries="@array/tiposped"
        android:prompt="@string/mensa_tipoPed">

    </Spinner>

    <Spinner
        android:id="@+id/spdistrito"
        android:layout_width="match_parent"
        android:layout_height="43dp"
        android:entries="@array/distritos"
        android:prompt="@string/mensa_distrito"></Spinner>

    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="textMultiLine"
        android:hint="Detalles"
        android:id="@+id/edtdetalles"
        ></EditText>
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Entregar a"
        android:id="@+id/edtentregar"
        >
    </EditText>
    <EditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Direccion "
        android:id="@+id/edtdireccion"
        >
    </EditText>


    <Space
        android:layout_width="match_parent"
        android:layout_height="34dp" />

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



18. Implementación del MainActivity:

```
public class MainActivity extends AppCompatActivity  {
        Spinner sptipo, spdistrito;
        EditText edtdetalle, edtdireccion, edtentregar;

        ListView listViewPedidos;
        private List<Pedido> listPedidos = new ArrayList<Pedido>();
        ArrayAdapter<Pedido> arrayAdapterPedidos;

        private DatabaseReference mDataBase;
        private FirebaseDatabase firebaseDatabase;
        Pedido ped_sel;
        @Override
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);
            sptipo=findViewById(R.id.spTipoPed);
            spdistrito=findViewById(R.id.spdistrito);
            edtdetalle=findViewById(R.id.edtdetalles);
            edtdireccion=findViewById(R.id.edtdireccion);
            edtentregar=findViewById(R.id.edtentregar);

            listViewPedidos=findViewById(R.id.lstpedidos);
            mDataBase= FirebaseDatabase.getInstance().getReference();

            inicializarFirebase();
            listarPedidos();
            listViewPedidos.setOnItemClickListener(new AdapterView.OnItemClickListener() {
                @Override
                public void onItemClick(AdapterView<?> adapterView, View view, int i, long l)
{
                    ped_sel=(Pedido) adapterView.getItemAtPosition(i);
                    edtdetalle.setText(ped_sel.getDescripcion());
                    edtentregar.setText(ped_sel.getEntregar());
                    edtdireccion.setText(ped_sel.getDireccion());
```

```
                    //buscar el indice del texto a mostrar
                    String qtipopedido=ped_sel.getTipoped();
                    String qDistrito=ped_sel.getDistrito();
                    sptipo.setSelection(obtenerPosicionItem(sptipo,qtipopedido));
                    spdistrito.setSelection(obtenerPosicionItem(spdistrito, qDistrito));
                }
            });
        }

        public  static  int obtenerPosicionItem(Spinner spinner, String texto){
            int posicion=0;
            for (int i= 0;i<spinner.getCount();i++){
                if (spinner.getItemAtPosition(i).toString().equalsIgnoreCase(texto))
posicion=i;
            }
            return posicion;
        }

        private void listarPedidos() {
            mDataBase.child("Pedido").addValueEventListener(new ValueEventListener() {
                @Override
                public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
                    listPedidos.clear();
                    int n=0;
                    for (DataSnapshot objSnapShot : dataSnapshot.getChildren()){
                        Pedido p=objSnapShot.getValue(Pedido.class);
                        n++;
                        listPedidos.add(p);
                        arrayAdapterPedidos = new ArrayAdapter<Pedido>(MainActivity.this,
android.R.layout.simple_list_item_1,listPedidos);
                        listViewPedidos.setAdapter(arrayAdapterPedidos);
                    }
                }

                @Override
                public void onCancelled(@NonNull DatabaseError databaseError) {

                }
            });
        }



        private void inicializarFirebase(){
            FirebaseApp.initializeApp(this);
            firebaseDatabase=FirebaseDatabase.getInstance();
            mDataBase=firebaseDatabase.getReference();
        }

        public void registrarpedido(){
            String tipo= sptipo.getSelectedItem().toString();
            String distrito = spdistrito.getSelectedItem().toString();
            String descripcion= edtdetalle.getText().toString();
            String entrega = edtentregar.getText().toString();
            String direccion = edtdireccion.getText().toString();

            if (!TextUtils.isEmpty(descripcion) ){
                String id = mDataBase.push().getKey();
                Pedido pedido = new Pedido(id,tipo,distrito,descripcion,entrega,direccion);
                mDataBase.child("Pedido").child(id).setValue(pedido);
                Toast.makeText(this, "pedido registrado", Toast.LENGTH_SHORT).show();
            }
            else Toast.makeText(this, "debe agregar una descripción",
Toast.LENGTH_SHORT).show();
        }
```

```
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.menu1, menu);
        return super.onCreateOptionsMenu(menu);
    }

    @Override
    public boolean onOptionsItemSelected(@NonNull MenuItem item) {
        switch (item.getItemId()){
            case R.id.icon_save:{
                registrarpedido();
                break;
            }
            case R.id.icon_update:{
                actualizarPedido();
                break;
            }
            case R.id.icon_add:{
                //Limpiar controles
                Toast.makeText(this, "Limpiar controles", Toast.LENGTH_SHORT).show();
                break;
            }
            case R.id.icon_list:{
                Toast.makeText(this, "listado ", Toast.LENGTH_SHORT).show();
                break;
            }

        }
        return true;
    }
    public void actualizarPedido(){
        Pedido p = new Pedido();
        p.setIdpedido(ped_sel.getIdpedido());
        p.setDescripcion(edtdetalle.getText().toString());
        p.setDireccion(edtdireccion.getText().toString());
        p.setEntregar(edtentregar.getText().toString());
        p.setDistrito(spdistrito.getSelectedItem().toString());
        p.setTipoped(sptipo.getSelectedItem().toString());
        mDataBase.child("Pedido").child(p.getIdpedido()).setValue(p);
        Toast.makeText(this, "actualizado", Toast.LENGTH_SHORT).show();

    }
}
```

Semana 11- Sesión 2

Sección:	Apellidos : .....
Docente : Pedro Yuri Marquez Solis	Nombres : .....
Unidad : Unidad 3	Fecha: ...../...../..... Duración: 60 min

Servicio de autenticación

I. Propósito: Emplear un servicio de autenticación.

II. Descripción de la actividad a realizar (caso.)

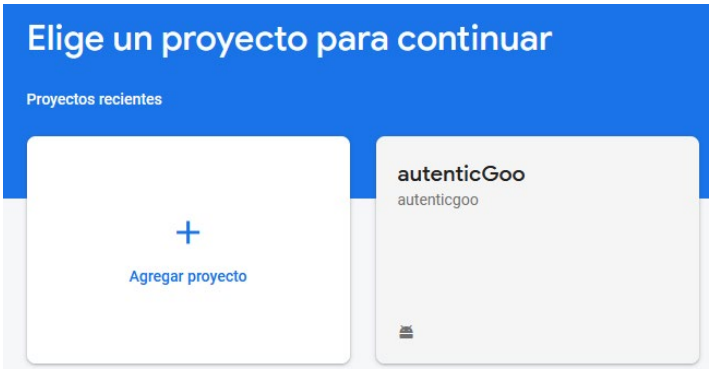
Existen diversas formas de autenticar a un usuario. FireBase provee del servicio Autenticacion con el cual se puede facilitar bastante este proceso y es el que se mostrará a continuación.

III. Procedimientos

Instrucciones:

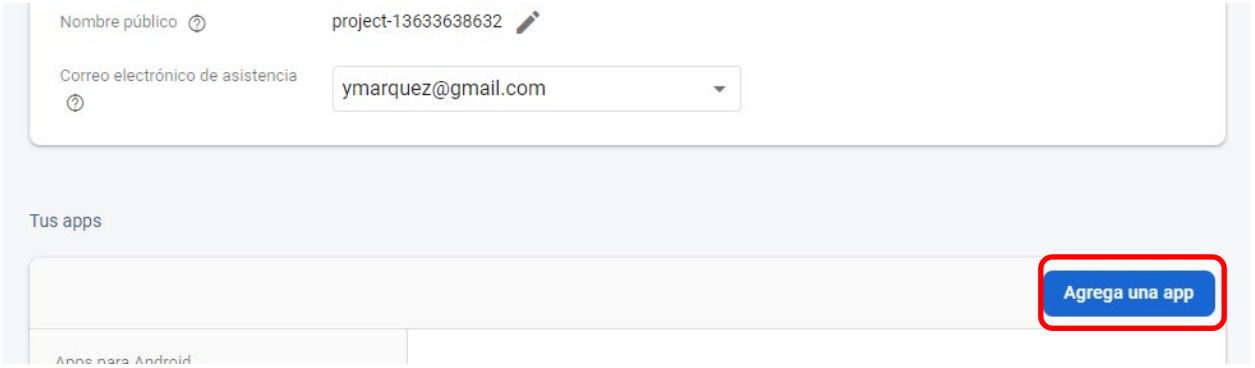
Algunas funciones se van actualizando constantemente en Android Studio y en Firebase, por lo que te recomendamos seguir los siguientes pasos:

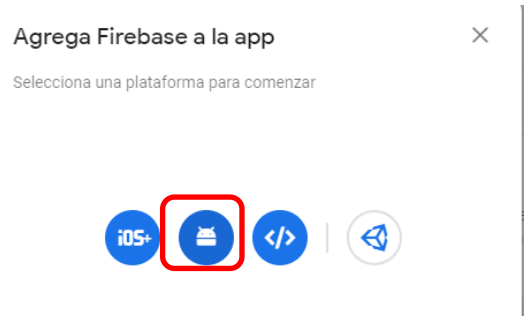
1. **En Firebase Crea un nuevo proyecto o elige uno que ya tengas creado**



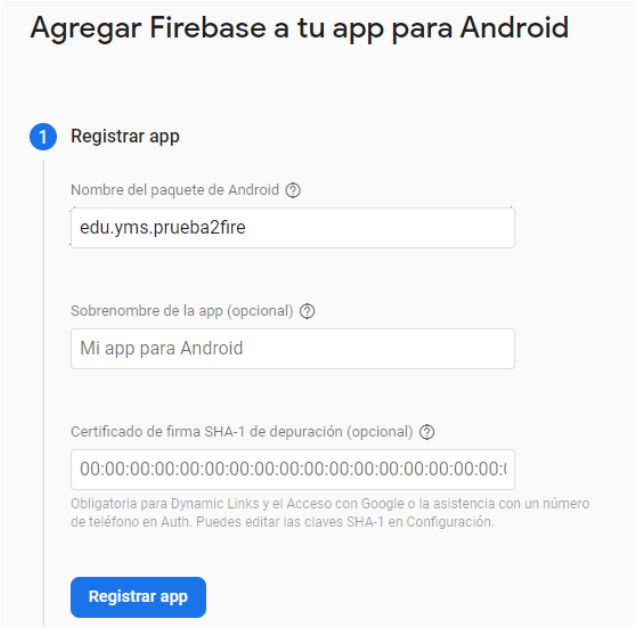
2. **Crea un nuevo proyecto en Android Studio, puedes emplear inicialmente la plantilla de Empty Activity.**
3. **Copia el nombre del proyecto al portapapeles.**
4. **Ingresa a la configuración del proyecto de Firebase.**

5. **Puedes elegir agregar una app o si es un proyecto nuevo elige aplicación Android.**

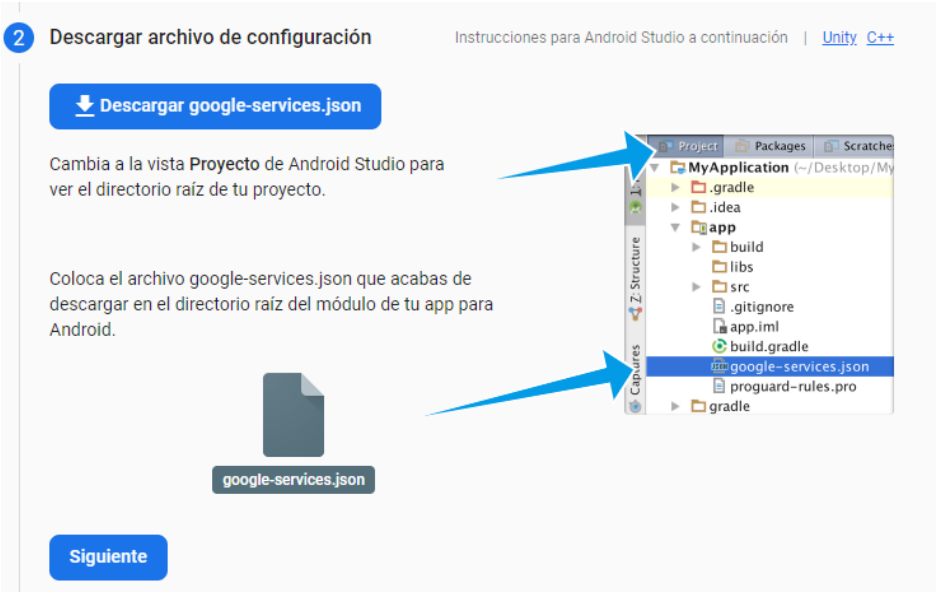




6. **Ingresa los datos solicitados y luego clic a Registrar app**



7. **Descarga el archivo googleservices.json, al directorio tal como se muestra en la imagen:**



8. **Agrega Android Studio referencias indicadas al Grade Scripts, tanto del proyecto como del módulo.**

Los gradle Scripts deben quedar como se muestra:

Project:

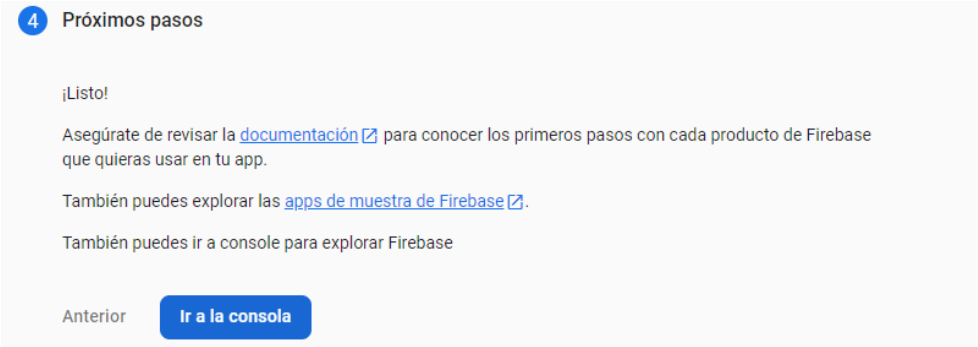
```
buildscript {
    repositories {
        google()
        mavenCentral()
    }
    dependencies {
        classpath "com.android.tools.build:gradle:4.2.2"
        classpath 'com.google.gms:google-services:4.3.10'

        // NOTE: Do not place your application dependencies here; they belong
        // in the individual module build.gradle files
    }
}

allprojects {
    repositories {
        google()
        mavenCentral()
        //jcenter() // Warning: this repository is going to shut down soon
    }
}

task clean(type: Delete) {
    delete rootProject.buildDir
}
```

Module:

```
plugins {
    id 'com.android.application'
    id 'com.google.gms.google-services'
}

android {
    compileSdkVersion 32
    buildToolsVersion '30.0.2'

    defaultConfig {
        applicationId "edu.yms.prueba2fire"
        minSdkVersion 23
        targetSdkVersion 32
        versionCode 1
        versionName "1.0"

        testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
    }

    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-
optimize.txt'), 'proguard-rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
```

```
        targetCompatibility JavaVersion.VERSION_1_8
    }
}

dependencies {

    implementation 'androidx.appcompat:appcompat:1.4.0'
    implementation 'com.google.android.material:material:1.4.0'
    implementation 'androidx.constraintlayout:constraintlayout:2.1.2'
    implementation 'com.google.firebase:firebase-auth:21.0.1'
    testImplementation 'junit:junit:4.+'
    androidTestImplementation 'androidx.test.ext:junit:1.1.3'
    androidTestImplementation 'androidx.test.espresso:espresso-core:3.4.0'
    implementation platform('com.google.firebase:firebase-bom:29.0.3')
    implementation 'com.google.firebase:firebase-analytics'
    implementation 'com.google.firebase:firebase-database:20.0.3'
    implementation 'com.basgeekball:awesome-validation:4.2'
    implementation 'com.google.guava:listenablefuture:9999.0-empty-to-avoid-
conflict-with-guava'
    implementation 'com.google.firebase:firebase-core:9.6.1'
}
```

9. **Finalmente concluye y dirigete a la consola.**



10. **Crea la base de datos en Real Time DataBase.**

11. **Modifica el MainActivity para efectuar una agregación por defecto:**

```
public class MainActivity extends AppCompatActivity {
Button button;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
```

```
            FirebaseDatabase database = FirebaseDatabase.getInstance();
            DatabaseReference myRef = database.getReference("message");

            myRef.setValue("Hello, World! test1 ");
        }
    }
```

12. Agrega las siguientes actividades:

- **a. HomeActivity**
- **b. RecuperarPassActivity**
- **c. RegistrarActivity**

13. Implementa el layouts respectivos tal como se muestra:

- **a. 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:layout_height="match_parent"
    android:orientation="vertical"
    android:gravity="center"
    tools:context=".RegistrarActivity">
    <TextView
        android:textSize="25sp"
        android:textColor="#000"
        android:gravity="center"
        android:text="Registrar cuenta"
        android:layout_width="match_parent"
        android:layout_height="wrap_content">
    </TextView>

    <EditText
        android:id="@+id/et_mail"
        android:layout_marginBottom="10dp"
        android:textColorHint="@color/colorPrimary"
        android:hint="Ingrese correo"
        android:layout_width="match_parent"
        android:layout_margin="20dp"
        android:gravity="center"
        android:layout_height="wrap_content">

    </EditText>

    <EditText
        android:id="@+id/et_pass"
        android:layout_marginBottom="10dp"
        android:textColorHint="@color/colorPrimary"
        android:hint="Ingrese password"
        android:layout_width="match_parent"
        android:layout_margin="20dp"
        android:gravity="center"
        android:layout_height="wrap_content">
    </EditText>
    <Button
        android:id="@+id/btn_registrar"
        android:textColor="#fff"
        android:text="Registrar"
        android:background="@color/colorPrimary"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content">
```

```
    </Button>
</LinearLayout>
```

14. **Implementa la lógica del RegistrarActovoty.java**

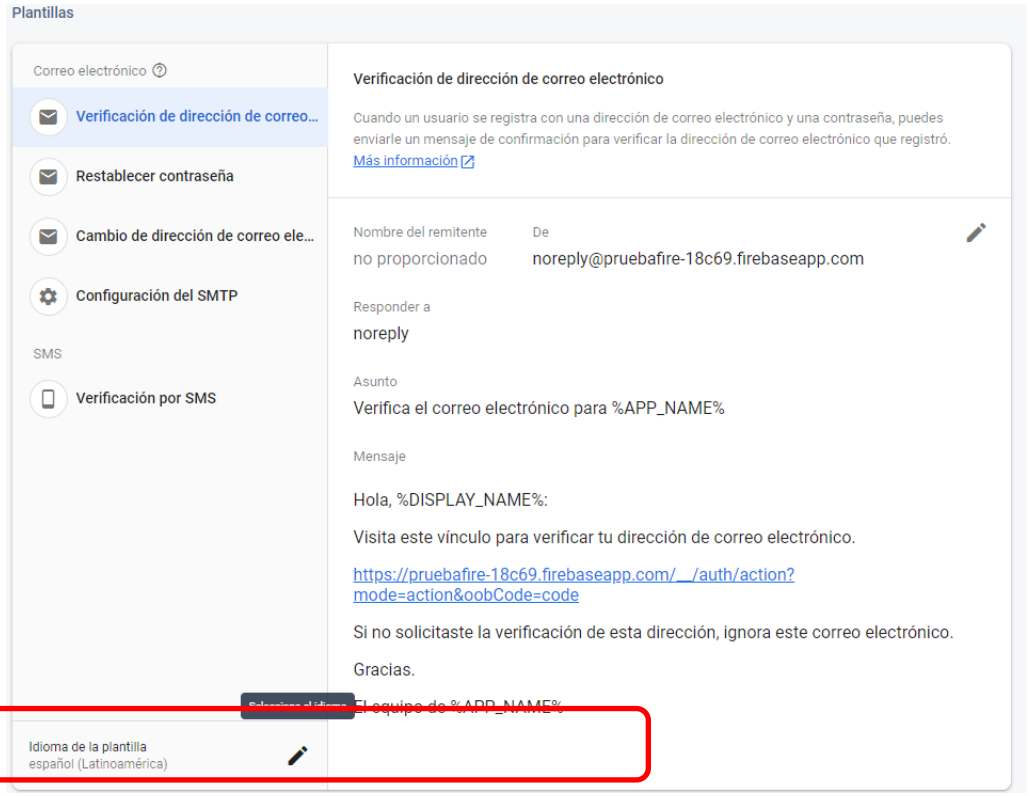
```
public class RegistrarActivity extends AppCompatActivity {
    EditText et_mail,et_pass;
    Button btn_registrar;
    AwesomeValidation awesomeValidation;
    FirebaseAuth firebaseAuth;


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

        firebaseAuth = FirebaseAuth.getInstance();
        awesomeValidation = new AwesomeValidation(ValidationStyle.BASIC);
        awesomeValidation.addValidation(this,R.id.et_mail,
Patterns.EMAIL_ADDRESS,R.string.invalid_mail);
        awesomeValidation.addValidation(this,R.id.et_pass,".{6,}",R.string.invalid_password);


        et_mail = findViewById(R.id.et_mail);
        et_pass = findViewById(R.id.et_pass);
        btn_registrar = findViewById(R.id.btn_registrar);

        btn_registrar.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {

                String mail = et_mail.getText().toString();
                String pass = et_pass.getText().toString();

                if(awesomeValidation.validate()){

firebaseAuth.createUserWithEmailAndPassword(mail,pass).addOnCompleteListener(new
OnCompleteListener<AuthResult>() {
                        @Override
                        public void onComplete(@NonNull Task<AuthResult> task) {
                            if(task.isSuccessful()){
                                Toast.makeText(RegistrarActivity.this, "Usuario creado con
exito", Toast.LENGTH_SHORT).show();
                                finish();
                            }else {
                                String errorCode = ((FirebaseAuthException)
task.getException()).getErrorCode();
                                dameToastdeerror(errorCode);
                            }
                        }
                    });
                }else {
                    Toast.makeText(RegistrarActivity.this, "Completa todos los datos..!!",
Toast.LENGTH_SHORT).show();
                }
            }
        });
    }

private void dameToastdeerror(String error) {

        switch (error) {

            case "ERROR_INVALID_CUSTOM_TOKEN":
                Toast.makeText(RegistrarActivity.this, "El formato del token personalizado es
incorrecto. Por favor revise la documentación", Toast.LENGTH_LONG).show();
                break;

            case "ERROR_CUSTOM_TOKEN_MISMATCH":
                Toast.makeText(RegistrarActivity.this, "El token personalizado corresponde a
una audiencia diferente.", Toast.LENGTH_LONG).show();
                break;

            case "ERROR_INVALID_CREDENTIAL":
```

```
                Toast.makeText(RegistrarActivity.this, "La credencial de autenticación
proporcionada tiene un formato incorrecto o ha caducado.", Toast.LENGTH_LONG).show();
                break;

            case "ERROR_INVALID_EMAIL":
                Toast.makeText(RegistrarActivity.this, "La dirección de correo electrónico
está mal formateada.", Toast.LENGTH_LONG).show();
                et_mail.setError("La dirección de correo electrónico está mal formateada.");
                et_mail.requestFocus();
                break;

            case "ERROR_WRONG_PASSWORD":
                Toast.makeText(RegistrarActivity.this, "La contraseña no es válida o el
usuario no tiene contraseña.", Toast.LENGTH_LONG).show();
                et_pass.setError("la contraseña es incorrecta ");
                et_pass.requestFocus();
                et_pass.setText("");
                break;

            case "ERROR_USER_MISMATCH":
                Toast.makeText(RegistrarActivity.this, "Las credenciales proporcionadas no
corresponden al usuario que inició sesión anteriormente..", Toast.LENGTH_LONG).show();
                break;

            case "ERROR_REQUIRES_RECENT_LOGIN":
                Toast.makeText(RegistrarActivity.this,"Esta operación es sensible y requiere
autenticación reciente. Inicie sesión nuevamente antes de volver a intentar esta solicitud.",
Toast.LENGTH_LONG).show();
                break;

            case "ERROR_ACCOUNT_EXISTS_WITH_DIFFERENT_CREDENTIAL":
                Toast.makeText(RegistrarActivity.this, "Ya existe una cuenta con la misma
dirección de correo electrónico pero diferentes credenciales de inicio de sesión. Inicie
sesión con un proveedor asociado a esta dirección de correo electrónico.",
Toast.LENGTH_LONG).show();
                break;

            case "ERROR_EMAIL_ALREADY_IN_USE":
                Toast.makeText(RegistrarActivity.this, "La dirección de correo electrónico ya
está siendo utilizada por otra cuenta..   ", Toast.LENGTH_LONG).show();
                et_mail.setError("La dirección de correo electrónico ya está siendo utilizada
por otra cuenta.");
                et_mail.requestFocus();
                break;

            case "ERROR_CREDENTIAL_ALREADY_IN_USE":
                Toast.makeText(RegistrarActivity.this, "Esta credencial ya está asociada con
una cuenta de usuario diferente.", Toast.LENGTH_LONG).show();
                break;

            case "ERROR_USER_DISABLED":
                Toast.makeText(RegistrarActivity.this, "La cuenta de usuario ha sido
inhabilitada por un administrador..", Toast.LENGTH_LONG).show();
                break;

            case "ERROR_USER_TOKEN_EXPIRED":
                Toast.makeText(RegistrarActivity.this, "La credencial del usuario ya no es
válida. El usuario debe iniciar sesión nuevamente.", Toast.LENGTH_LONG).show();
                break;

            case "ERROR_USER_NOT_FOUND":
                Toast.makeText(RegistrarActivity.this, "No hay ningún registro de usuario que
corresponda a este identificador. Es posible que se haya eliminado al usuario.",
Toast.LENGTH_LONG).show();
                break;

            case "ERROR_INVALID_USER_TOKEN":
                Toast.makeText(RegistrarActivity.this, "La credencial del usuario ya no es
válida. El usuario debe iniciar sesión nuevamente.", Toast.LENGTH_LONG).show();
                break;

            case "ERROR_OPERATION_NOT_ALLOWED":
                Toast.makeText(RegistrarActivity.this, "Esta operación no está permitida.
Debes habilitar este servicio en la consola.", Toast.LENGTH_LONG).show();
                break;
```

```
            case "ERROR_WEAK_PASSWORD":
                Toast.makeText(RegistrarActivity.this, "La contraseña proporcionada no es
válida..", Toast.LENGTH_LONG).show();
                et_pass.setError("La contraseña no es válida, debe tener al menos 6
caracteres");
                et_pass.requestFocus();
                break;
        }
    }
}
```

15. Diigete a Authentication en Firebase y configúralo de la siguiente forma:

16. En Templates cambia el idioma a Esáñol Latinoamericano



17. **Ahora prueba que se puedan agregar usuarios desde la actividad RegistrarActivity.**
18. **Puedes descargar el proyecto completo desde:**
    https://github.com/ymarquezs/appAutenticacion

Semana 12- Sesión 2

Entnediendo los permisos en Android

Sección: ……..……..	Apellidos : ……..……..
Docente : Pedro Yuri Marquez Solis	Nombres : ……
Unidad : Unidad 3	Fecha: …../……/……. Duración: 60 min

I. Propósito: Definir y entender los permisos de aplicaciones y usarios en Android.

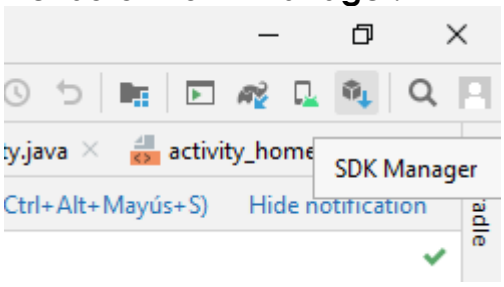
II. Descripción de la actividad a realizar (caso.)

La seguridad es un aspecto clave de todo sistema. Si nos descargáramos una aplicación maliciosa de Internet o de Google Play Store, esta podría leer nuestra lista de contactos, averiguar nuestra posición GPS, mandar toda esta información por Internet y terminar enviando cientos de mensajes SMS.

III. Procedimientos

Instrucciones:

1. **Asegúrate de tener instalada una versión de Android 6.0 o posterior. Para verificarlo pulsa en el botón SDK Manager:**



2. **Crea un nuevo proyecto con los siguientes datos:**
   Application name: Permisos en Android
   Package name: edu.conti.permisosXYZ
   ☑ Phone and Tablet
   Minimum SDK: API 22
   Agrega una activity: **Basic Activity**

3. **Añade las líneas subrayadas y elimina las tachadas en el método onCreate() de MainActivity.**

```
4. public void onClick(View view) {
   borrarLlamada();
   ~~Snackbar.make(view, "Replace with your …", Snackbar.LENGTH_LONG)~~
   ~~.setAction("Action", null).show();~~
    }
```

5. Declara la siguiente variable al principio de la clase:

private View vista;

6. En el método onCreate() añade:

vista = findViewById(R.id.content_main);

7. En la etiqueta <RelativeLayout> de content_main.xml añade:

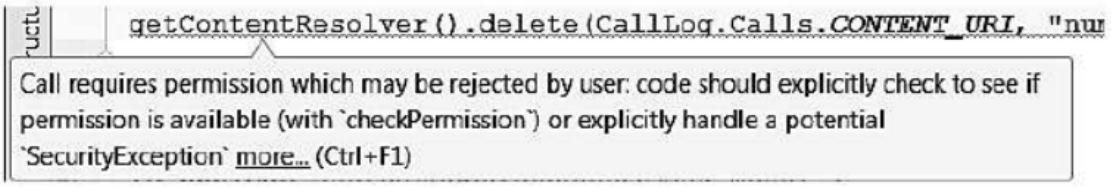
android:id="@+id/content_main"

8. Añade el siguiente método:

```
private void borrarLlamada() {
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
        getContentResolver().delete(CallLog.Calls.CONTENT_URI,
"number='555555555'",null);
    }
    Snackbar.make(vista, "Llamadas borradas del registro.",
            Snackbar.LENGTH_SHORT).show();

}
```

El código anterior elimina del registro de llamadas del teléfono todas las llamadas cuyo número sea 555555555. La segunda línea muestra un cuadro de texto tipo Snackbar para avisar que la acción se ha realizado.

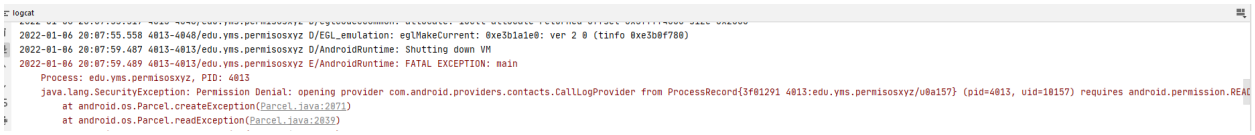
En versiones anteriores Android Studio advierte de que estamos actuando de forma no es correcta:



En versiones mas recientes debemos agregar las condicionales de versión del SDK}

9. Ejecuta la aplicación , si pulsas el botón flotante la aplicación debería colgarse.

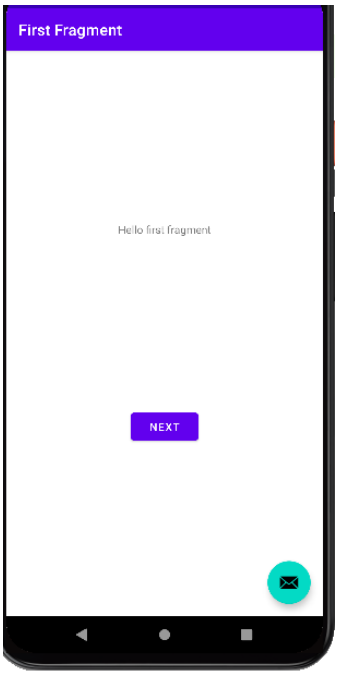
10. Abre el LogCat para verificar la causa del error:



Es decir, la aplicación se ha detenido porque está realizando una acción que requiere de la solicitud de un permiso.

11. Añade en AndroidManifest.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="edu.yms.permisosxyz">
    <uses-permission android:name="android.permission.WRITE_CALL_LOG"/>
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="permisosXYZ"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.PermisosXYZ">
        <activity
            android:name=".MainActivity2"
```



Si ejecutas de nuevo el proyecto en un dispositivo con una versión anterior a la 6.0, podrás verificar que ya no se produce el error. , en cambio si ejecutas ahora en un dispositivo con versión 6.0 (si no dispones de uno utiliza un emulador), observarás que el error continúa. Cuando instalamos una aplicación no se le concede ningún permiso peligroso.

Como acabamos de comprobar la aplicación anterior va a funcionar correctamente en dispositivos con una versión anterior a la 6.0. Sin embargo, cuando se ejecute en la nueva versión, se producirá un error. Aunque hemos visto cómo el usuario puede evitarlo, no es desde luego la forma correcta de trabajar. A partir de Android Marshmallow trabajar con acciones que necesiten de un permiso va a suponer un esfuerzo adicional para el programador.

Fuente : (Tomás Gironés, 2018)

Cuarta unidad

Semana 13- Sesión 2

Sección: ……..……..	Apellidos : ……..……..
Docente :	Nombres : ……..
Unidad : Unidad 4	Fecha: …../……/……. Duración: 60 min

Sensores

I. Propósito: Implementar un proyecto que emplee sensores..

II. Descripción de la actividad a realizar (caso.)

Los sensores integrados en los dispositivos móviles permitentener unconocimiento delentornoquele rodeaydeinteractuar conelmismo. Androidinteractúa y toma datosdesdesensores de movimiento, ambientales, de posición entre otros.

En este laboratorio desarrollamos un proyecto con sensores.

III. Procedimientos

Instrucciones:

1. Modifica el código del 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">
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:id="@+id/btnLienzo"
        android:text="Lienzo"
        ></Button>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="X"
        android:id="@+id/txtx"
        android:layout_weight="1"
        android:textSize="40dp"
        android:gravity="center_vertical"
        android:layout_gravity="center"
     />
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Y"
        android:id="@+id/txty"
        android:layout_weight="1"
```

```
            android:textSize="40dp"
            android:gravity="center_vertical"
            android:layout_gravity="center"
            />
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Z"
            android:id="@+id/txtz"
            android:layout_weight="1"
            android:textSize="40dp"
            android:gravity="center_vertical"
            android:layout_gravity="center"
            />

    </LinearLayout>
```

El diseño debe quedar como se muestra:

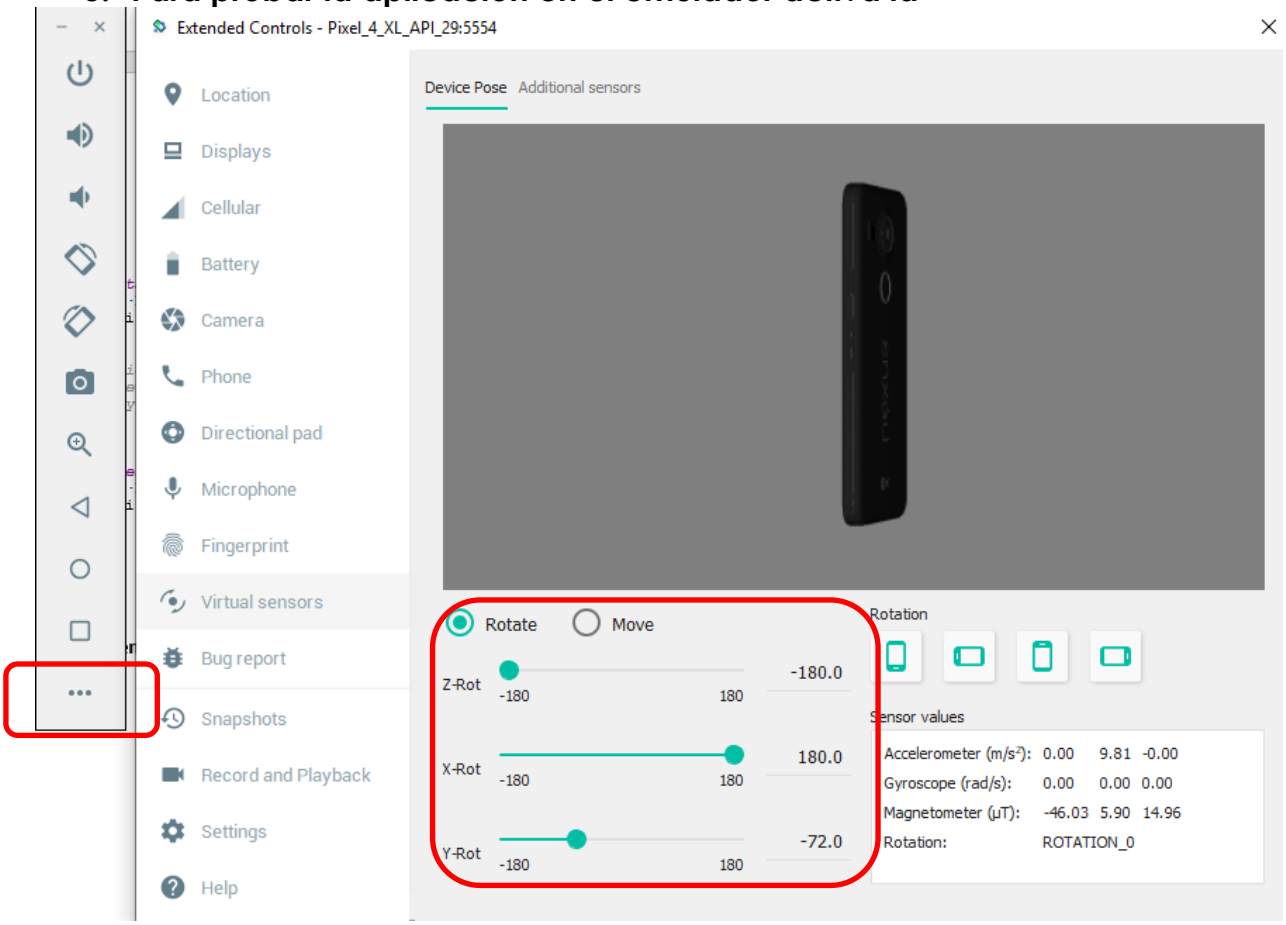
2. Modifica el MainActivity.java con el código que se muestra a continuación:

```
public class MainActivity extends AppCompatActivity {
    TextView txtx,txty,txtz;
    SensorManager sensorManager;
    Sensor sensor;

   // elLienzo miLienzo;

    //sensores generan datos, para recibir los datos generados por los sensores
    SensorEventListener sensorEventListener;
    boolean ejecutar=false;
    int movs=0;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
       // miLienzo = new elLienzo(this);
        //setContentView(miLienzo);

        txtx=findViewById(R.id.txtx);
        txty=findViewById(R.id.txty);
        txtz=findViewById(R.id.txtz);
//        Toolbar myToolbar = (Menu) findViewById(R.id.men_principal);
//        setSupportActionBar(myToolbar);
```

```
        sensorManager=(SensorManager)getSystemService(SENSOR_SERVICE);
        sensor=sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
        //implicitamente se le dice que traiga los datos desde el puerto de memoria del
acelerómetro
        if (sensor==null) {
            Toast.makeText(this, "Acelerómetro no disponible",
Toast.LENGTH_SHORT).show();
            finish();}
        sensorEventListener = new SensorEventListener() {
            @Override
            public void onSensorChanged(SensorEvent valorensor) {
                float x=valorensor.values[0];
                float y=valorensor.values[1];
                float z=valorensor.values[2];

                txtx.setText(String.valueOf(x));
                txty.setText(String.valueOf(y));
                txtz.setText(String.valueOf(z));
                if (x < 0) {
                    getWindow().getDecorView().setBackgroundColor(Color.YELLOW);

                }
                if (x>0) { getWindow().getDecorView().setBackgroundColor(Color.RED);
                    movs++;
                    ejecutar=true;
                       }
                if (movs==3 && ejecutar){
                    movs=0;
                    ejecutar=false;
                    playsound();

                    }
                }

            @Override
            public void onAccuracyChanged(Sensor sensor, int i) {

            }
        };
        iniciar();
    }

    private void playsound() {
        MediaPlayer  mediaPlayer= MediaPlayer.create(this, R.raw.grit);
        mediaPlayer.start();

    }

    private void iniciar() {
        sensorManager.registerListener(sensorEventListener, sensor,
SensorManager.SENSOR_STATUS_ACCURACY_LOW);
    }
    private void stop(){
        sensorManager.unregisterListener(sensorEventListener);
    }

    @Override
    protected void onPause() {
        stop();
        super.onPause();
    }

    @Override
    protected void onResume() {
        iniciar();
        super.onResume();
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.menu, menu);
        return super.onCreateOptionsMenu(menu);
    }

    @Override
```

```
    public boolean onOptionsItemSelected(@NonNull MenuItem item) {
        switch (item.getItemId()){
            case  R.id.men_principal:{
                Intent i = new Intent(this, MainActivity.class);
                startActivity(i);
                break;
            }
            case  R.id.men_listasensores:{
                Intent i = new Intent(this, ListaSensores.class);
                startActivity(i);
                break;
            }
//            case  R.id.men_lienzo:{
//                Intent i = new Intent(this, elLienzo.class);
//                startActivity(i);
//                break;
//            }

            case  R.id.men_nivel:{
                Intent i = new Intent(this, sensorgrafico.class);
                startActivity(i);
                break;
            }
            }
        return true;
        }

}
```

3. **Para probar la aplicación en el emulador activa la**



Semana 14- Sesión 2

Sección:	Apellidos : .....
Docente : Pedro Yuri Marquez Solis	Nombres : .....
Unidad : Unidad 4	Fecha: ...../...../..... Duración: 60 min

Uso de GitHub en el desarrollo móvil

I. Propósito: Utilizar un gestor de versiones para controlar del desarrollo de un proyecto.

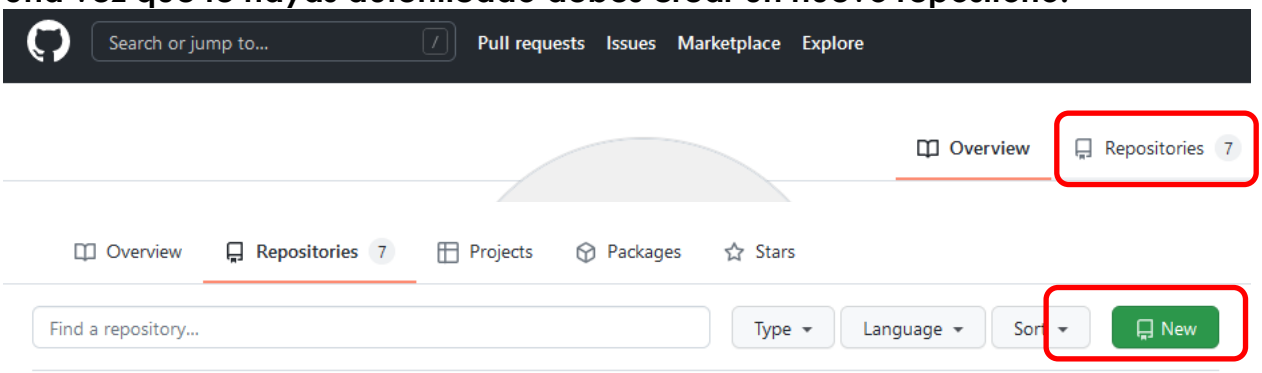
II. Descripción de la actividad a realizar (caso.)

El uso de un software de Control de versiones es de suma importancia, inclusive si se está desarrollando un proyecto en forma personal.
En esta practica se muestra como utilizar Github para utilizar el software de control de versiones GIT.

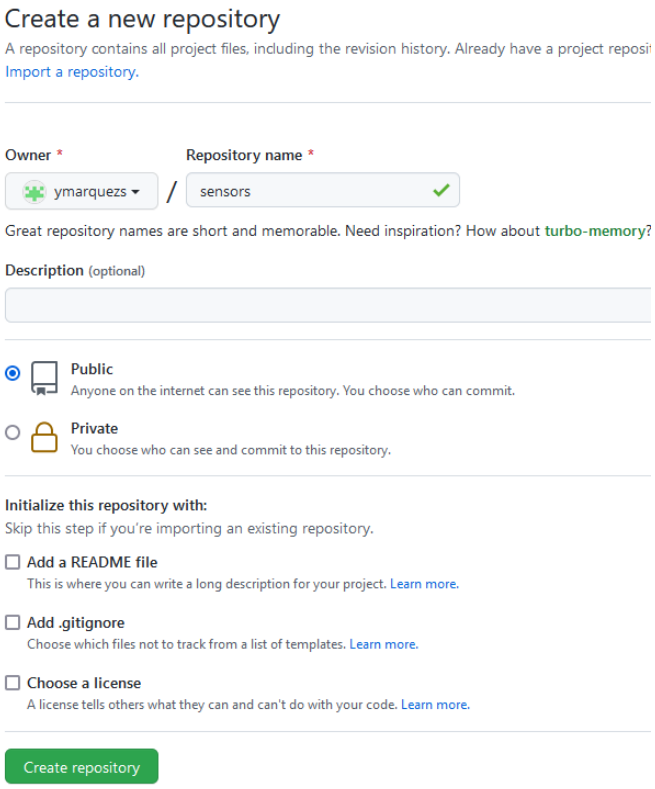
III. Procedimientos

Instrucciones:

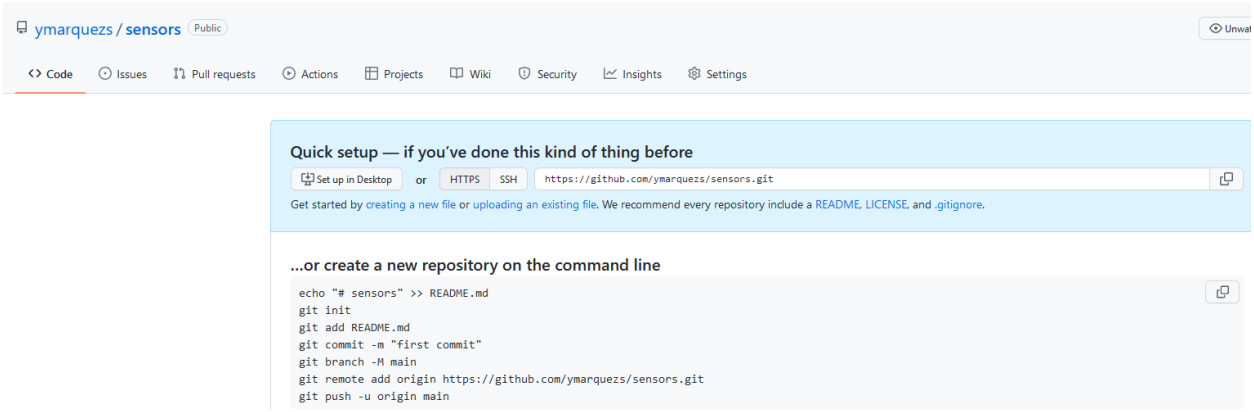
1. **Debes poseer una cuenta en guthub.com**
2. **Una vez que te hayas autenticado debes crear un nuevo repositorio.**



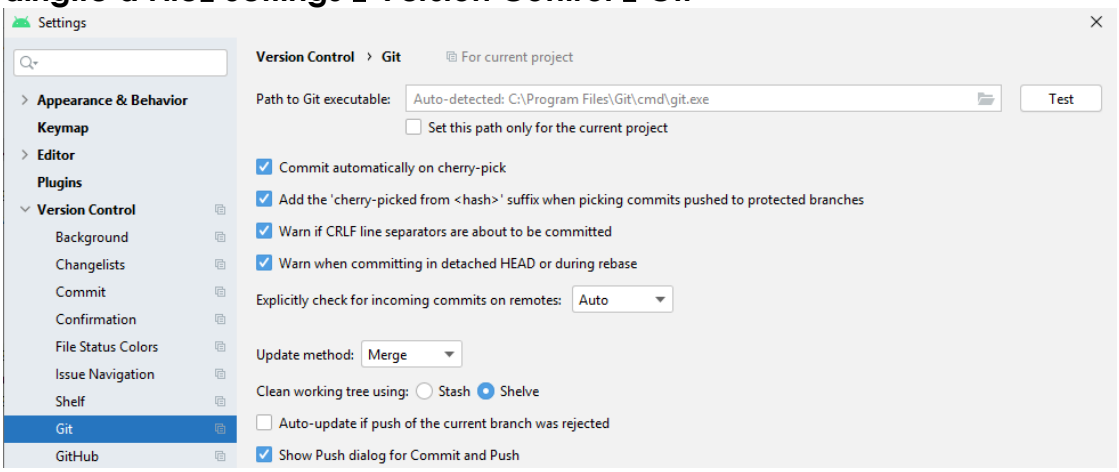
Ingresa el nombre del repositorio y si es necesario brinda una descripción del proyecto.
Puedes hacer que sea de acceso Público.
Finalmente crea el repositorio.



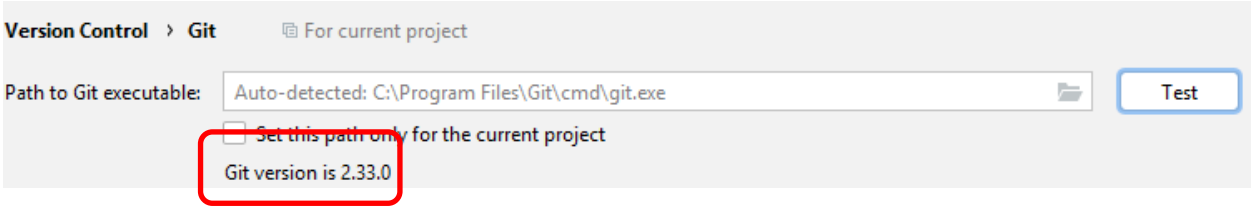
3. **Resultado obtenido:**



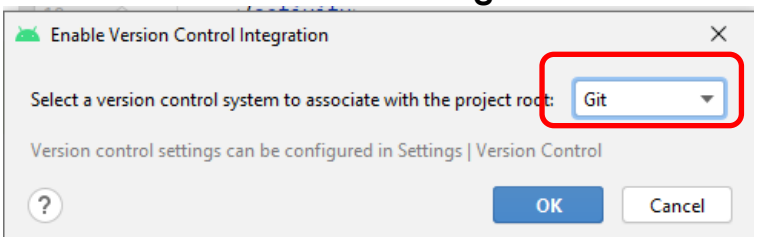
4. **En Android Studio lo primero que debes hacer es instalar GIT, para lo cual dirigite a File Settings Version Control GIt**



Efectúa clic spbre el botón Test, de no tener instalado se mostrará un mensaje y un link para instalar Git.

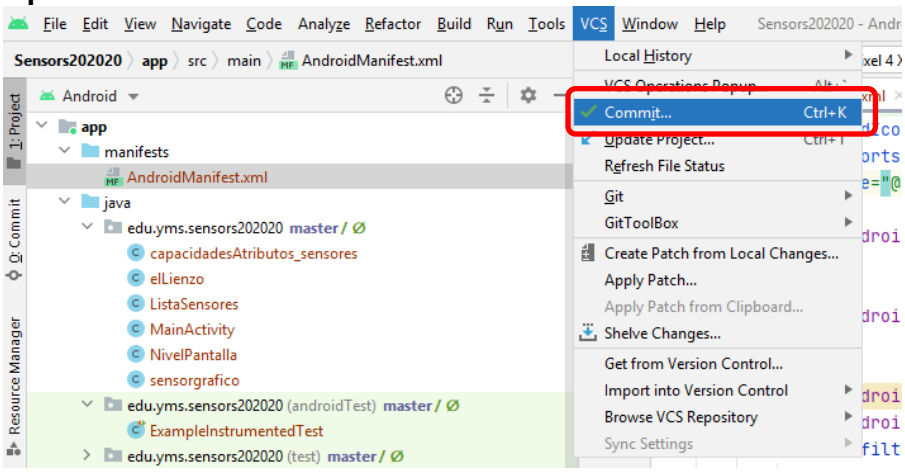


5. **Una vez comprobado que se tiene instalado Git dirigete al menú VCS y elige la opción Enable Version Control Integration.**



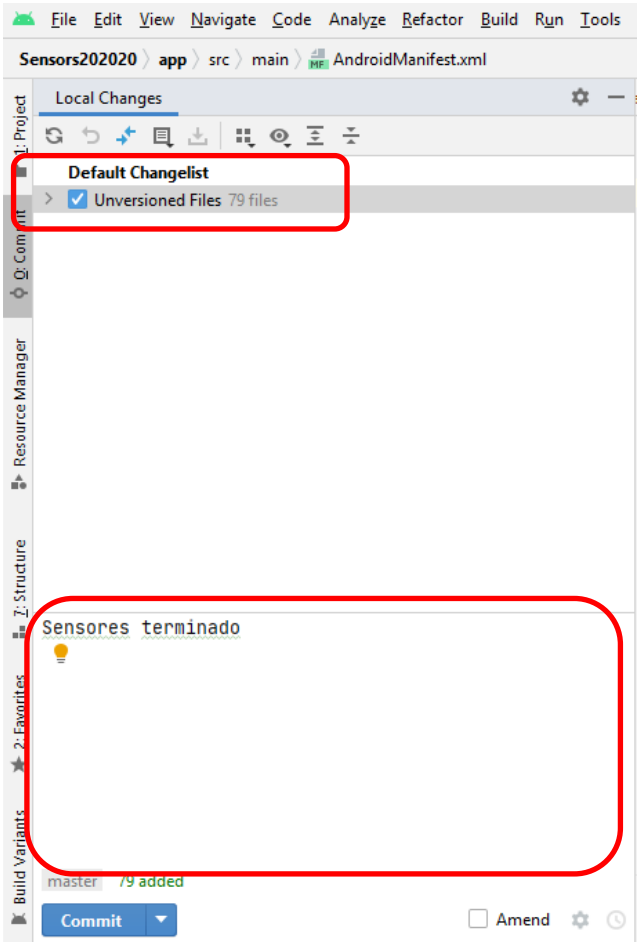
Elige Git y click a Ok.

6. **Ahora podrás apreciar que en el menú VCS se tiene la opción Commit, además que el color de los archivos ha cambiado.**



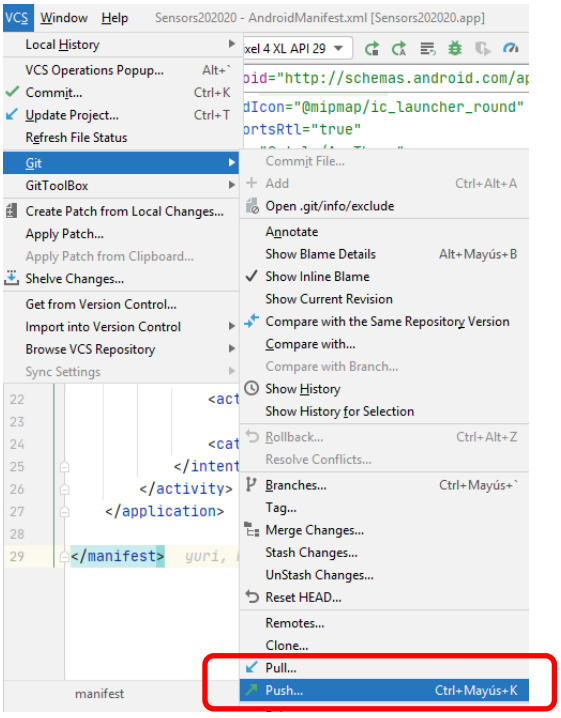
7. **Seleccionar la opción Commit y en el panel que se muestra a continuación:**

Selecciona los archivos que serán controlados, en este caso para seleccionar todos puedes seleccionar el checkBox de Unversioned Files.
Luego debes agregar un comentario.
Al darle clic a Commit únicamente estas preparando los archivos para ser controlados y comparados.

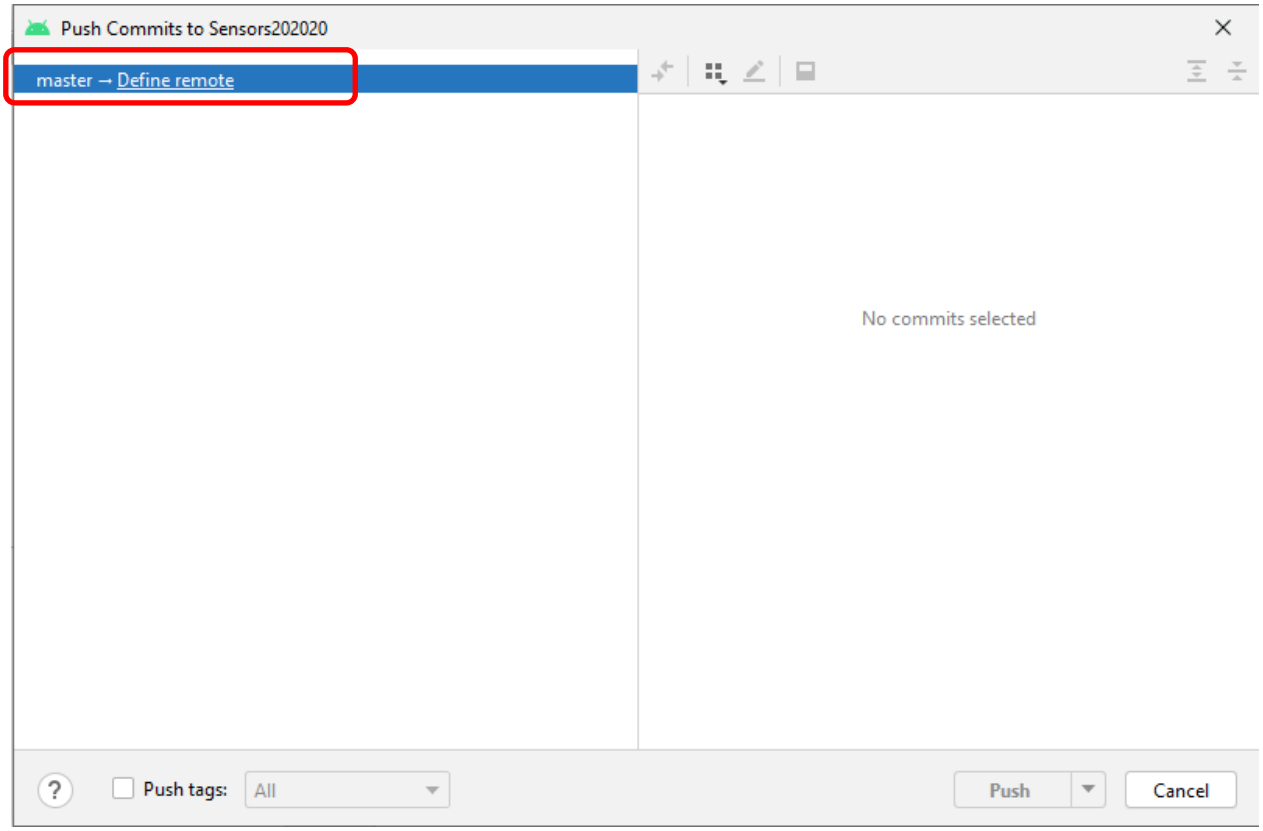


8. Enviar los archivos al repositorio:

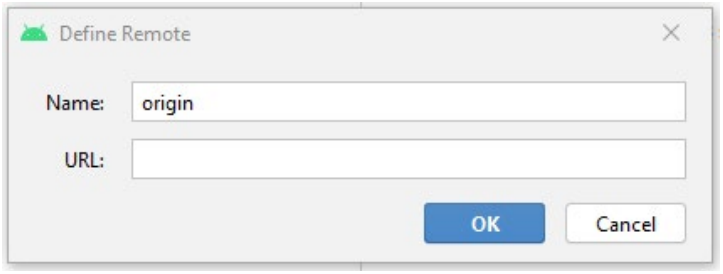
Para ello debes elegir en el menú VCS Git Push.



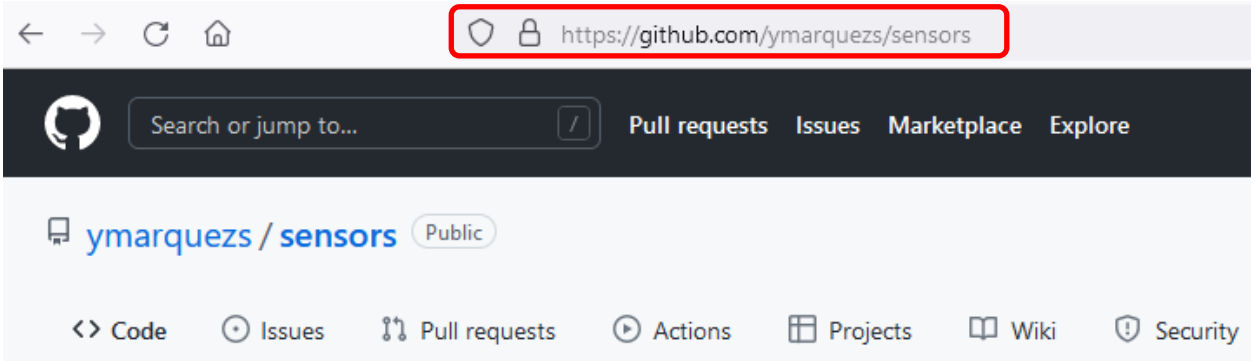
9. **Ahora debes definir el repositorio:**



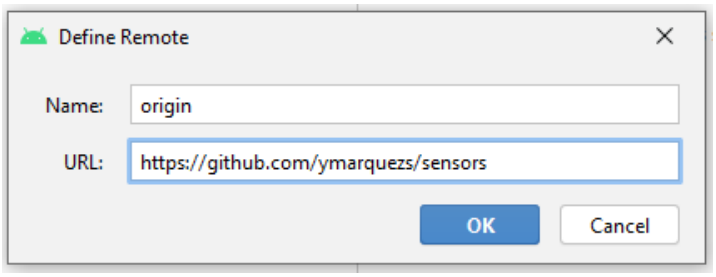
Al efectuar clic sobre deine remote se mostrará un cuadro de dialogo para indicar el URL del repositorio:



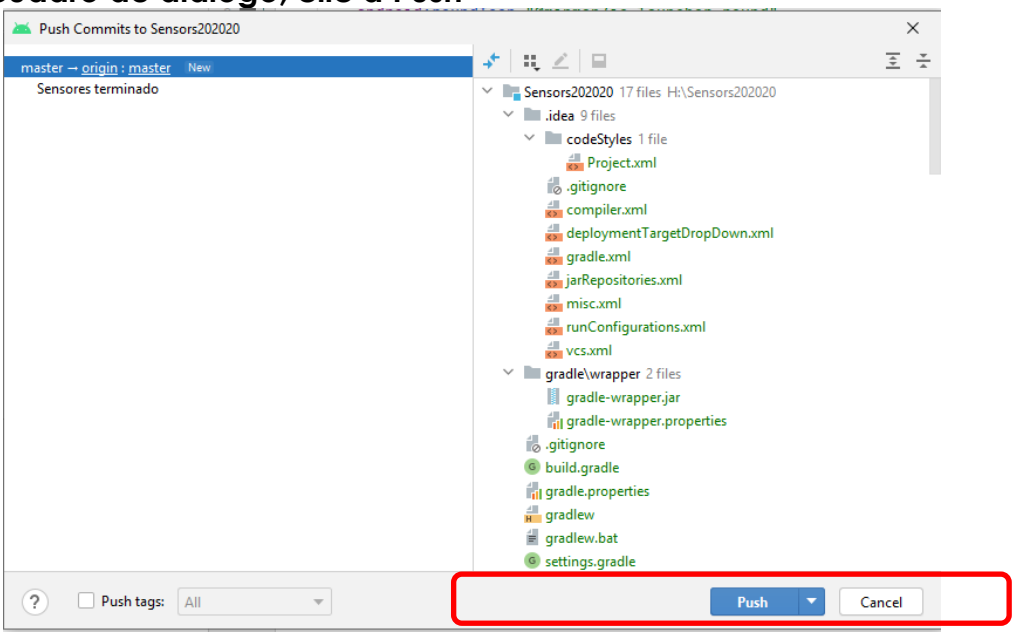
10. **Para obtener el URL debes dirigirte a GitHub y copiar de la dirección del navegador el url correspondiente:**



Debe quedar como se muestra:

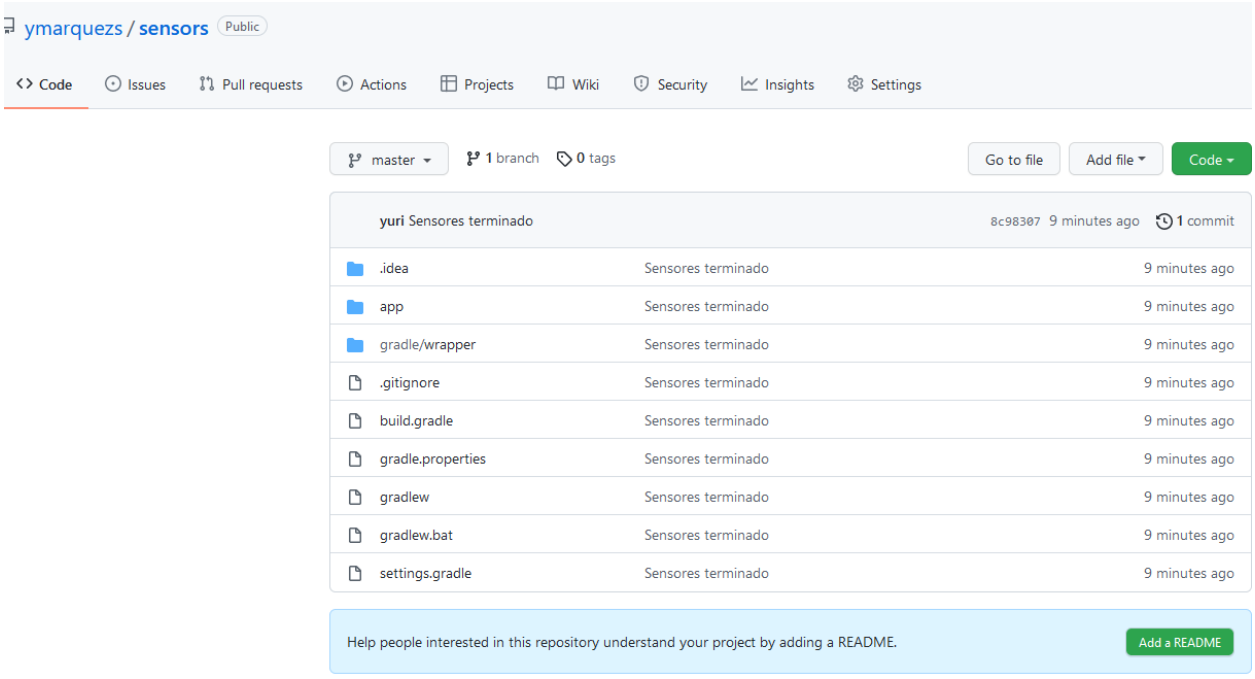


11. En el cuadro de diálogo, clic a Push



12. Se pedirá las credenciales de GitHub las cuales deben ser ingresadas para completar el proceso.

Finalmente se debe ver como se muestra.



Semana 15- Sesión 2

Sección: ……..……..	Apellidos : ……..……..
Docente : Pedro Yuri Marquez Solis	Nombres : ……
Unidad : Unidad 4	Fecha: …../……/……. Duración: 60 min

Definición de requerimientos de un proyecto Móvil

I. Propósito: Definir criterios de requerimientos funcionales y no funcionales de software para dispositivos móviles.

II. Descripción de la actividad a realizar (caso.)

El desarrollo de software para dispositivos móviles debe considerar criterios que no son necesariamente los mismos que para aplicaciones de escritorio o aplicaciones Web, por ello es importante que en esta practica puedas entender como se definen..

III. Procedimientos

Instrucciones:

1) Emplea la siguiente plantilla para listar los requerimientos funcionales de tu proyecto:

Nombre del Proyecto :

Describe brevemente las características de la aplicación.

Requisitos comerciales
¿Por qué decidió crear una aplicación móvil?
● Para compartir tu experiencia única
● Para crear una fuente de ingresos adicional
● Para mejorar los procesos comerciales actuales
● Para obtener un retorno de la inversión
● Otra razón
¿Cuál es el objetivo principal de su proyecto?
● Para lanzar un nuevo negocio, producto o servicio en un nuevo mercado.

• Para mejorar el conocimiento de la marca fuera del sitio web.
• Para realizar mejoras, rediseñar o crear una nueva versión de la aplicación actual
• Algo más
¿A qué categoría pertenece tu aplicación?
• Juego de azar
• Entretenimiento
• Comercio electrónico
• Educación
• Estilo de vida
• Utilidad
• Viaje
• Otro
¿Cuáles son sus objetivos comerciales financieros y no financieros?
Objetivos financieros: Quiero capturar una participación de mercado del X% en Y meses.
Objetivos no financieros: Quiero ser calificado como la mejor aplicación móvil en su categoría en Apple App Store y Google Play Store en una fecha específica.
¿Qué esperas que haga tu aplicación?
Describir la funcionalidad principal
Ofrecer una propuesta de valor única
¿Quiénes son sus competidores directos e indirectos?
Enumere de tres a cinco competidores principales en su nicho (junto con enlaces)
Indique las características que le gustan y las que no le gustan en los productos de sus competidores

¿Cuál es su visión de producto? Reemplaza las palabras entre paréntesis para describir la visión de su producto
Para (sus usuarios objetivo) que (necesitan o quieren cambiar algo), (nombre de su aplicación móvil) es una aplicación móvil que proporcionará (una característica excelente). A diferencia de (modelo de negocio actual o competidores), mi aplicación proporcionará (principales ventajas).
Elija su modelo de monetización:
● Publicidad pagada ● Compras en la aplicación ● Suscripción Freemium ● Suscripción premium ● Algo más

Requisitos de usuario	
Describe los roles de usuario en tu aplicación:	
● Invitado / usuario habitual / usuario de pago	
● Comprador vendedor	
● Cliente / albacea	
● Estudiante /profesor	
● Proveedor / administrador	
● Tu clasificación	
En función de los roles de usuario, cree hasta tres posibles personajes de usuario teniendo en cuenta los siguientes criterios:	
Demografía (edad, sexo, estado familiar, nivel de educación, tipo de trabajo, ubicación)	
Psicografía (puntos débiles, objetivos, necesidades, problemas vitales, actitudes, motivaciones, opiniones)	
Comportamiento en el mercado (aplicaciones utilizadas, tipos de servicios / bienes comprados, motivos para usar la aplicación o comprar el producto o servicio, solvencia)	
Determine las preferencias de sus usuarios objetivo en términos de:	

Tipo de dispositivo: teléfono inteligente, tableta, computadora de escritorio, reloj inteligente, TV inteligente
Plataforma: iOS, Android, multiplataforma
Describe el recorrido del usuario:
Dibuje una ruta típica que sus usuarios tomarán dentro de su aplicación para obtener los resultados deseados
Agregar enlaces a bocetos de posibles interfaces de aplicaciones
Requisitos del sistema
Describa las funciones que desea que su aplicación proporcione a los usuarios: Enumere hasta tres funciones imprescindibles
Agregue enlaces, si los hay, a ejemplos de cómo debe verse una característica en particular
¿Qué tipo de contenido le gustaría agregar a su aplicación?
● Videos ● Audio ● Animaciones ● Imagenes ● RSS Feeds ● Otro
¿Qué servicios, servidores y bases de datos actuales utiliza?
¿Con qué aplicaciones, servicios y bases de datos de terceros necesita que se integre su aplicación? (pasarelas de pago, redes sociales, etc.)

¿Con qué versiones del sistema operativo debería ser compatible su aplicación?
Describa sus requisitos de UI:
• Estilo de aplicación móvil
• Esquema de colores
• Logo
• Iconos
• Botones
• Imagenes
• Fuentes
• Enlace a las pautas de marca que el equipo debe seguir
¿Tiene perfiles de aprovisionamiento actuales en Apple App Store y / o Google Play Store?
. ¿Con qué hardware necesita sincronizarse su aplicación? (dispositivos portátiles, drones, etc.)
. Describe los criterios de calidad de tu aplicación con respecto a:
• Usabilidad
• Rendimiento
• Seguridad
• La seguridad
• Otros atributos de calidad
. ¿A qué idiomas debería traducirse su aplicación?

Otros requerimientos
¿Cuáles son las restricciones y limitaciones dentro de las cuales debe trabajar el equipo?
Reglas del negocio
Estándares de la industria
Legislación gubernamental
Otras posibles limitaciones
¿Cuál es el cronograma y el presupuesto de su proyecto?
¿Cuándo espera comenzar y terminar el proyecto?
¿Cuál es el presupuesto aproximado (USD) que puede asignar al proyecto?
¿Qué servicios le gustaría solicitar a su equipo de desarrollo de software?
● Desarrollo de aplicaciones móviles de ciclo completo
● Desarrollo de sitios web
● Soporte y mantenimiento continuos
● Promoción y marketing
● Diseño de interfaz
● IT consulting
● Additional services

Lista de referencias

I. Bibliografía

Básica:

Ribas, J. (2013). Desarrollo de aplicaciones para Android. Anaya Multimedia-Anaya Interactiva.

Complementaria:

Amaya, Y (2020) Guía Metodológica Ágil Para el Desarrollo de Aplicaciones Móviles. Editorial Académica Española

II. Recursos digitales:

andro4all (2021, 25 de setiembre) . Android Studio, guía de iniciación: qué es, cómo descargar e instalar, y 4 cosas que puedes hacer con él. https://andro4all.com/guias/android/android-studio-descargar-instalar-guia-trucos

Ágil, I. d. (03 de 01 de .....). https://blog.hubspot.es/sales/producto-minimo-viable. Obtenido de https://youtu.be/F78byq2Ye50

Benitez, P. R. (03 de 01 de .....). https://economiatic.com/. Obtenido de https://economiatic.com/producto-minimo-viable/

ConsultoraPMO. (..... de 01 de 3). Scrum Ejemplo Práctico Product Backlog. Obtenido de en https://www.youtube.com/watch?v=573jB8DQPEU

casanova, s. (03 de 01 de .....). samuelcasanova.com. Obtenido de samuelcasanova

https://proyectosagiles.org. (03 de 01 de .....). https://proyectosagiles.org. Obtenido de https://proyectosagiles.org/graficos-trabajo-pendiente-burndown-charts/

Tomás Gironés, T. (2018). El gran libro de Android. Barcelona: MARCOMBO, S.A.

Yo Androide (2021, 25 de setiembre) Android Studio: Curso Completo Desde Cero. https://yoandroide.xyz/curso-completo-android-studio-desde-cero/