

**MANUAL BACKEND**

**APP: FERRELECTRICOS COSTA AZUL**



**UNIVERSIDAD  
NACIONAL  
DE COLOMBIA**

**DOCENTE:**

**NÉSTOR BOLIVAR PULGARÍN.**

**INTEGRANTES:**

**BECERRA GONZÁLEZ, ISABEL SOFÍA.**

**ORTIZ PÉREZ, LUIS FERNEY.**

**PROYECTO FINAL PROGRAMACIÓN ORIENTADA A OBJETOS.**

## Contenido

### Diseño y arquitectura

|   |    |
|---|----|
| Diagramas de casos de uso: .....            | 03 |
| Entorno de Desarrollo: Android Studio. .... | 04 |
| Configuración Gradle (Module. App) .....    | 06 |

### Bases de datos (Firebase):

|                         |    |
|-------------------------|----|
| Authentication: .....   | 09 |
| Real Time Database..... | 10 |

### Personas-Controladores:

|                              |    |
|------------------------------|----|
| Usuario/Cliente: .....       | 11 |
| Usuario/Empleado: .....      | 11 |
| Usuario/Administrador: ..... | 12 |

### Modelos dentro del paquete MODEL com. Ferrelectricos Costa Azul.

|                 |    |
|-----------------|----|
| Cliente: .....  | 13 |
| Producto: ..... | 15 |
| Empleado: ..... | 16 |

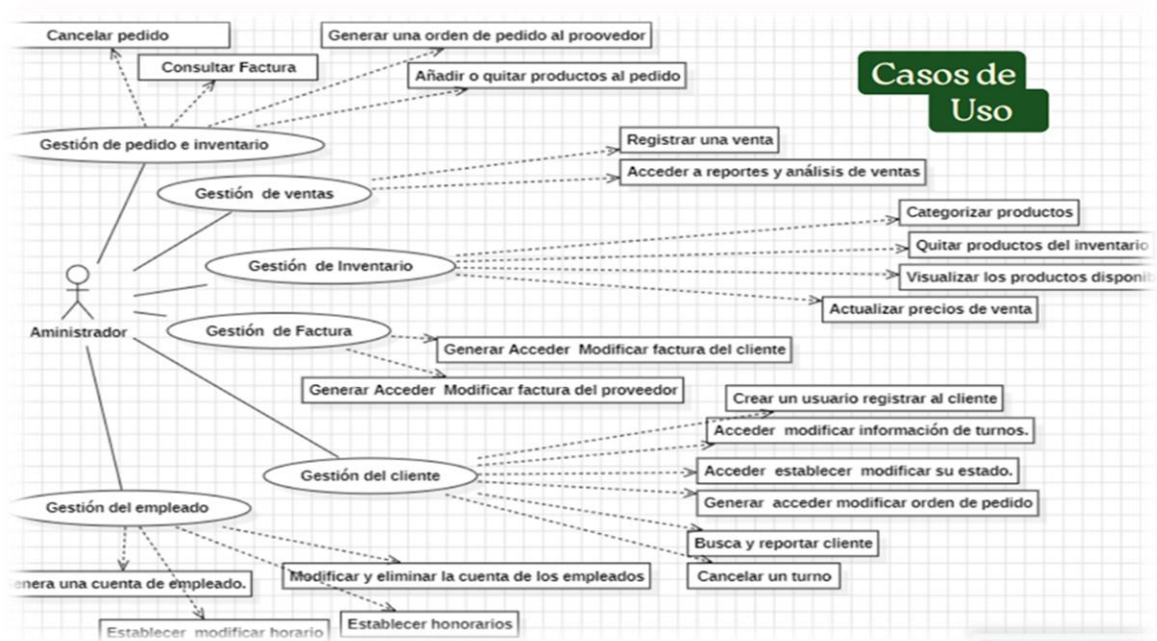
### Actividades View:

|                     |    |
|---------------------|----|
| Herramientas: ..... | 17 |
| Interfaz .....      | 18 |

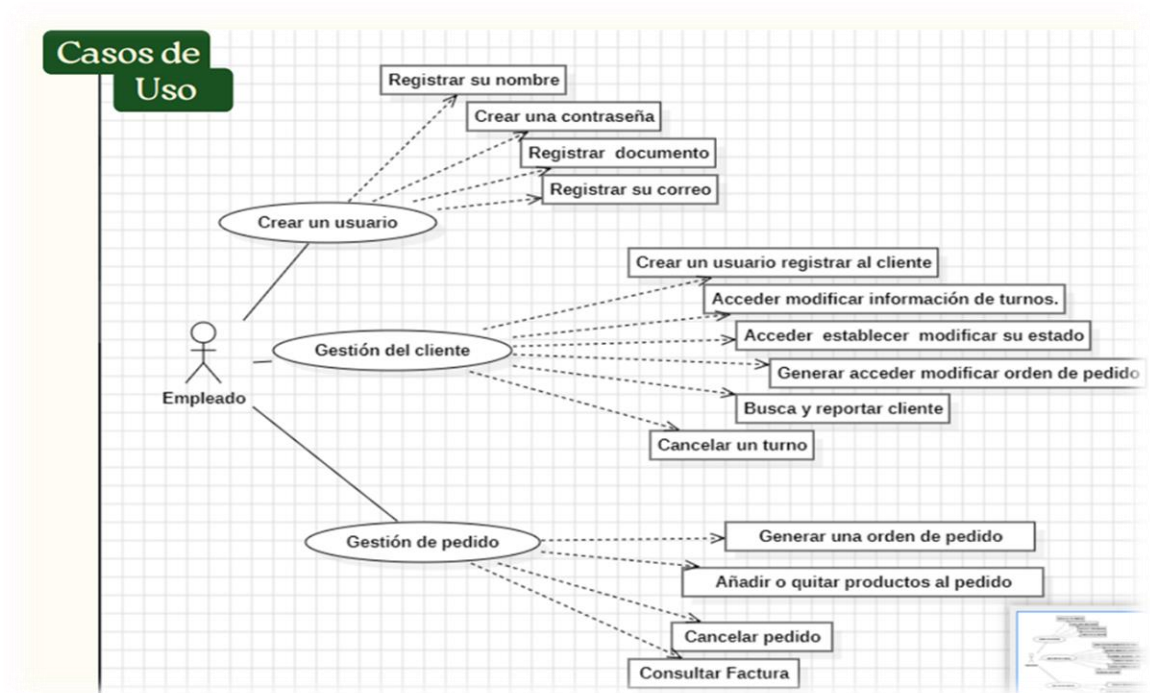
## DISEÑO Y ESTRUCTURA

### Casos de Usos:

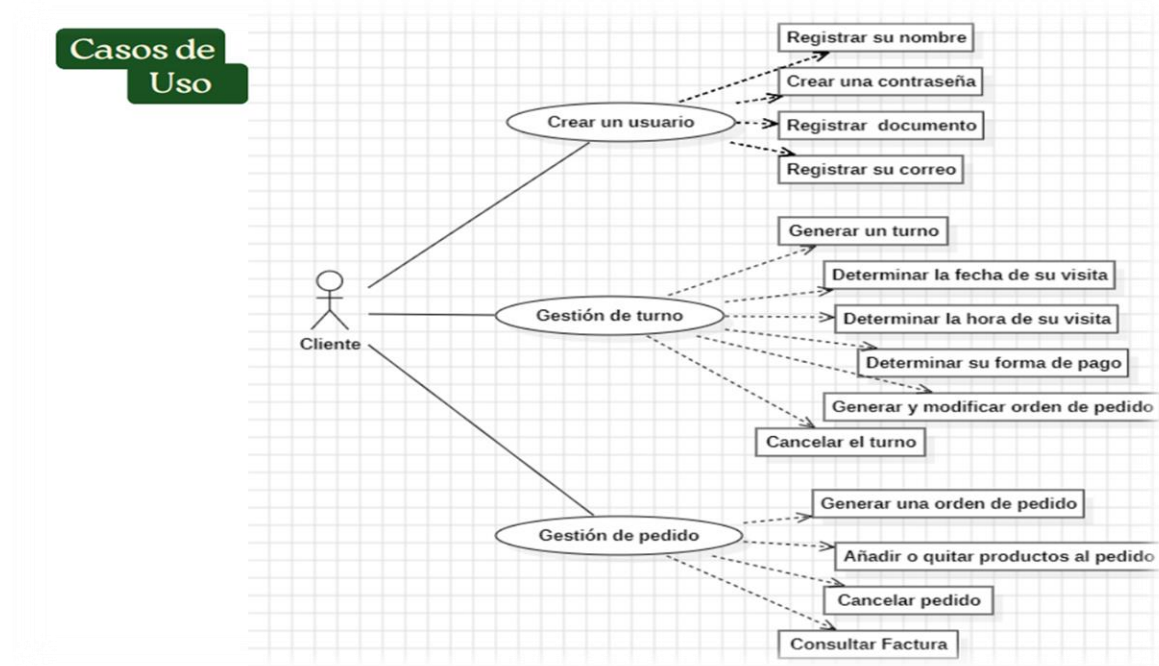
#### Administrador:



#### Empleado:



## Cliente:



## ENTORNO DE DESARROLLO:

El entorno de desarrollo apropiado para la realización de una aplicación móvil que se adapta a varios modelos de sistema Android es Android Studio. Con este entorno de desarrollo pudimos manejar un lenguaje de programación Java y al mismo tiempo acceder a múltiples herramientas de desarrollo de aplicaciones para sistemas Android.

Para la realización del Software se debe tener instalado Android Studio, cuya actualización más reciente corresponde a la versión 2022.3.1, en nuestro caso utilizaremos un SDK mínimo de 28 (número de versión) que aplica para Android 9 en adelante que aplica para más del 80% de los dispositivos, sin embargo, la aplicación podrá ser compatible en el SDK 34 el más reciente.

## GRADLE:

Android Studio usa Gradle, un paquete de herramientas de compilación avanzadas, para automatizar y administrar el proceso de compilación y, al mismo tiempo, definir configuraciones de compilación personalizadas y flexibles.

Gradle proporciona selección de dependencias personalizables y reglas de sustitución. Por eso es más flexible y podemos manejar múltiples variables y distintos tiempos de ejecución.

## Configuración build.gradle.kts

```
plugins { this: PluginDependenciesSpecScope
    id("com.android.application")
    id("com.google.gms.google-services")
}
```

```
android { this: BaseAppModuleExtension
    namespace = "com.example.ferrelectricoscostaazul"
    compileSdk = 34

    defaultConfig { this: ApplicationDefaultConfig
        applicationId = "com.example.ferrelectricoscostaazul"
        minSdk = 28
        targetSdk = 34
        versionCode = 1
        versionName = "1.0"

        testInstrumentationRunner = "androidx.test.runner.AndroidJUnitRunner"
    }
}
```

```
buildscript { this: ScriptHandlerScope
    dependencies { this: DependencyHandlerScope
        classpath("com.google.gms.google-services:4.4.0")
    }
}

// Top-level build file where you can add configuration options common to all sub-projects/modules.
plugins { this: PluginDependenciesSpecScope
    id("com.android.application") version "8.1.3" apply false
}
```

Para Poder llevar a cabo la elaboración del programa se deben tener instalados las siguientes plataformas SDK:

**Languages & Frameworks > Android SDK**

Manager for the Android SDK and Tools used by the IDE

Android SDK Location:  [Edit](#) [Optimize disk space](#)

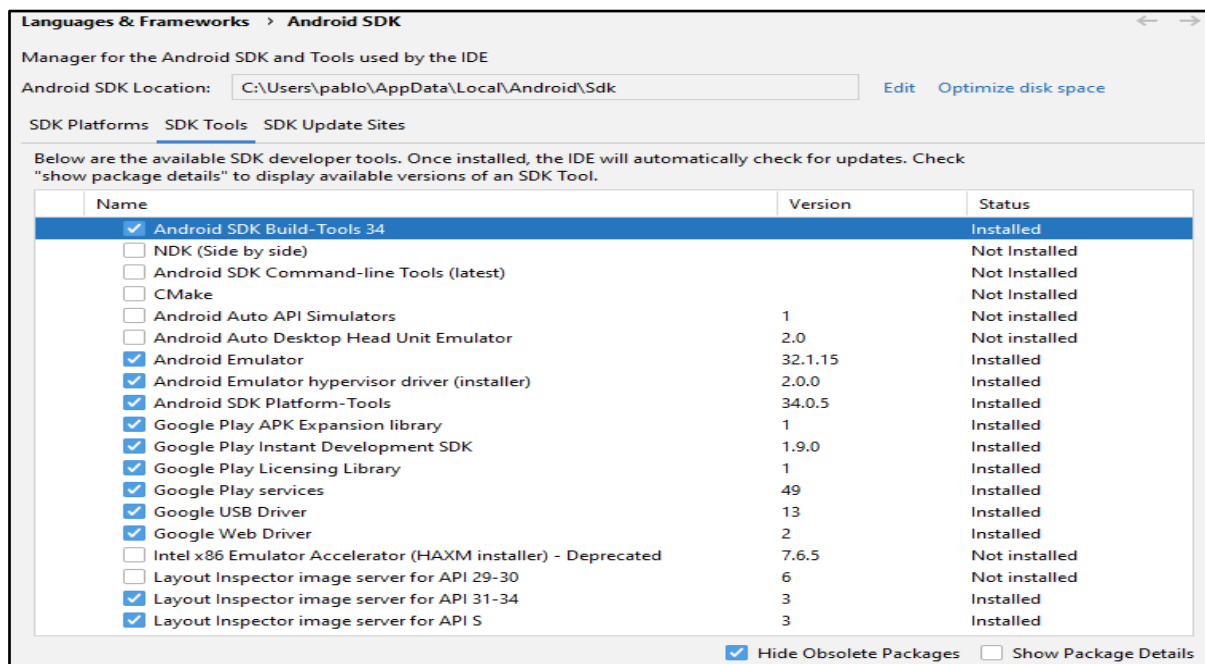
[SDK Platforms](#) [SDK Tools](#) [SDK Update Sites](#)

Each Android SDK Platform package includes the Android platform and sources pertaining to an API level by default. Once installed, the IDE will automatically check for updates. Check "show package details" to display individual SDK components.

| Name  | API Level                    | Revision | Status              |
|---|------------------------------|----------|---------------------|
| <input type="checkbox"/> Android UpsideDownCakePrivacySandbox Preview | UpsideDownCakePrivacySandbox | 2        | Not installed       |
| <input checked="" type="checkbox"/> Android API 34                    | 34                           | 2        | Installed           |
| <input checked="" type="checkbox"/> Android API 34                    | 34-ext8                      | 1        | Installed           |
| <input type="checkbox"/> Android UpsideDownCake Preview               | UpsideDownCake               | 4        | Not installed       |
| <input type="checkbox"/> Android TiramisuPrivacySandbox Preview       | TiramisuPrivacySandbox       | 9        | Not installed       |
| <input checked="" type="checkbox"/> Android 13.0 ("Tiramisu")         | 33                           | 3        | Installed           |
| <input type="checkbox"/> Android 13.0 ("Tiramisu")                    | 33-ext4                      | 1        | Not installed       |
| <input type="checkbox"/> Android 13.0 ("Tiramisu")                    | 33-ext5                      | 1        | Not installed       |
| <input type="checkbox"/> Android 12L ("Sv2")                          | 32                           | 1        | Not installed       |
| <input type="checkbox"/> Android 12.0 ("S")                           | 31                           | 1        | Partially installed |
| <input checked="" type="checkbox"/> Android 11.0 ("R")                | 30                           | 3        | Installed           |
| <input type="checkbox"/> Android 10.0 ("Q")                           | 29                           | 5        | Not installed       |
| <input checked="" type="checkbox"/> Android 9.0 ("Pie")               | 28                           | 6        | Installed           |
| <input type="checkbox"/> Android 8.1 ("Oreo")                         | 27                           | 3        | Not installed       |
| <input type="checkbox"/> Android 8.0 ("Oreo")                         | 26                           | 2        | Not installed       |
| <input checked="" type="checkbox"/> Android 7.1.1 ("Nougat")          | 25                           | 3        | Installed           |
| <input type="checkbox"/> Android 7.0 ("Nougat")                       | 24                           | 2        | Not installed       |
| <input type="checkbox"/> Android 6.0 ("Marshmallow")                  | 23                           | 3        | Not installed       |

☐ Hide Obsolete Packages ☒ Show Package Details

Además de las siguientes SDK tools, esto nos permitirá correr la aplicación en el emulador determinado por el dispositivo instalado predeterminado:



## LENGUAJE DE PROGRAMACIÓN:

El complemento de Android para Gradle 3.0.0 y las versiones posteriores admiten todas las funciones del lenguaje Java 7 y un subconjunto de funciones del lenguaje Java 8 que varían según la versión de la plataforma. Permitiendo así el uso del lenguaje de programación JAVA en su última versión.

```
compileOptions { this: CompileOptions
    sourceCompatibility = JavaVersion.VERSION_1_8
    targetCompatibility = JavaVersion.VERSION_1_8
}
```

## DEPENDENCIAS:

Las dependencias implementadas y agregadas, fueron preestablecidas a la hora de la instalación de Android Studio para permitir el desarrollo de la aplicación en el emulador como las de **Constrain Layout** que hace parte del desarrollo de la interfaz, adempas del **AppCompat**, para poder implementar actividades funcionales dentro de la aplicación.

Posteriormente se agregaron las dependencias necesarias, requeridas para la conexión con la base de datos firebase, en este caso para la conexión con la base de datos Real Time Database y la función de Authentication.

```
dependencies { this: DependencyHandlerScope

    implementation("androidx.appcompat:appcompat:1.6.1")
    implementation("com.google.android.material:material:1.10.0")
    implementation("androidx.constraintlayout:constraintlayout:2.1.4")
    implementation("com.google.firebase:firebase-auth-ktx:22.1.2")
    implementation("com.google.firebase:firebase-core:21.1.1")
    implementation("com.google.firebase:firebase-database:20.3.0")
    implementation("com.google.android.gms:play-services-auth:20.7.0")
    testImplementation("junit:junit:4.13.2")
    androidTestImplementation("androidx.test.ext:junit:1.1.5")
    androidTestImplementation("androidx.test.espresso:espresso-core:3.5.1")
}
```

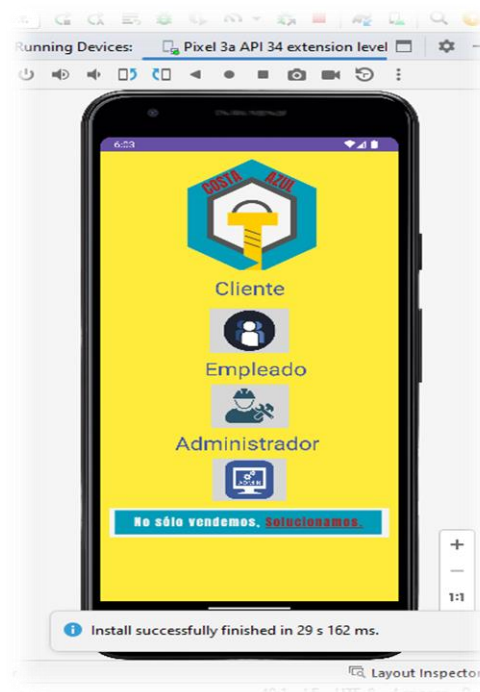
Dependencias para la creación del proyecto

## EMULADOR:

Permite realizar los tests correspondientes a la aplicación y sus modificaciones en tiempo real e implementando propiedades propias de interfaz gráfica de aplicaciones móviles en Android.

| Device Manager  |     |           |         |   |
|---|-----|-----------|---------|---|
| Virtual Physical  |     |           |         |   |
| Create Device   |     |           |         |   |
| Device  | API | Size o... | Actions |   |
| Pixel 3a API 34 extension level...<br>Android API 34 Google APIs   x86_64 | 34  | 12 GB     | ▶       | ⋮ |
| Pixel 4 API 30<br>Android 11.0 Google Play   x86                          | 30  | 12 GB     | ▶       | ⋮ |

Devises Instalados para poder probar el SDK correspondiente (34)



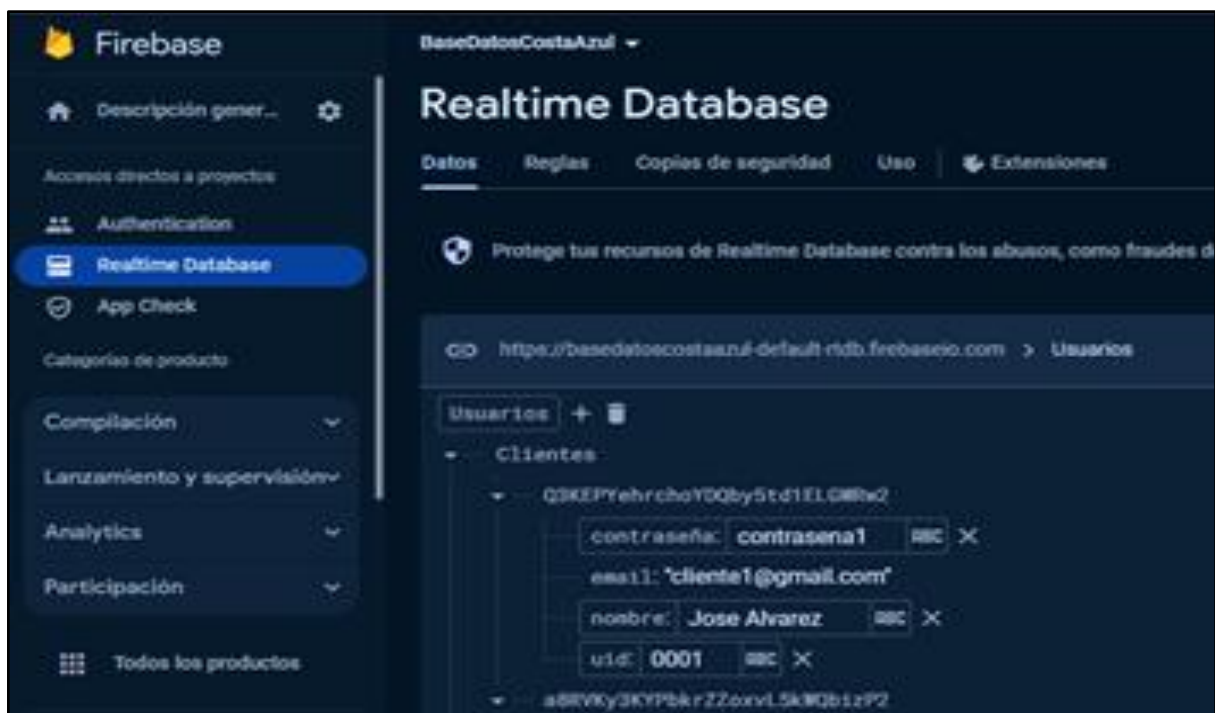


## Base de datos (Firebase)

Se definió el uso de Firebase como la base de datos de la aplicación a desarrollar dada su integración sencilla con Android Studio. Por ejemplo, el SDK para Android Studio se obtiene de forma directa. Como también, existe una conexión directa entre Android Studio y Firebase a través de una herramienta que despliega el menú de opciones, en el cual se puede seleccionar los servicios que ofrece Firebase.

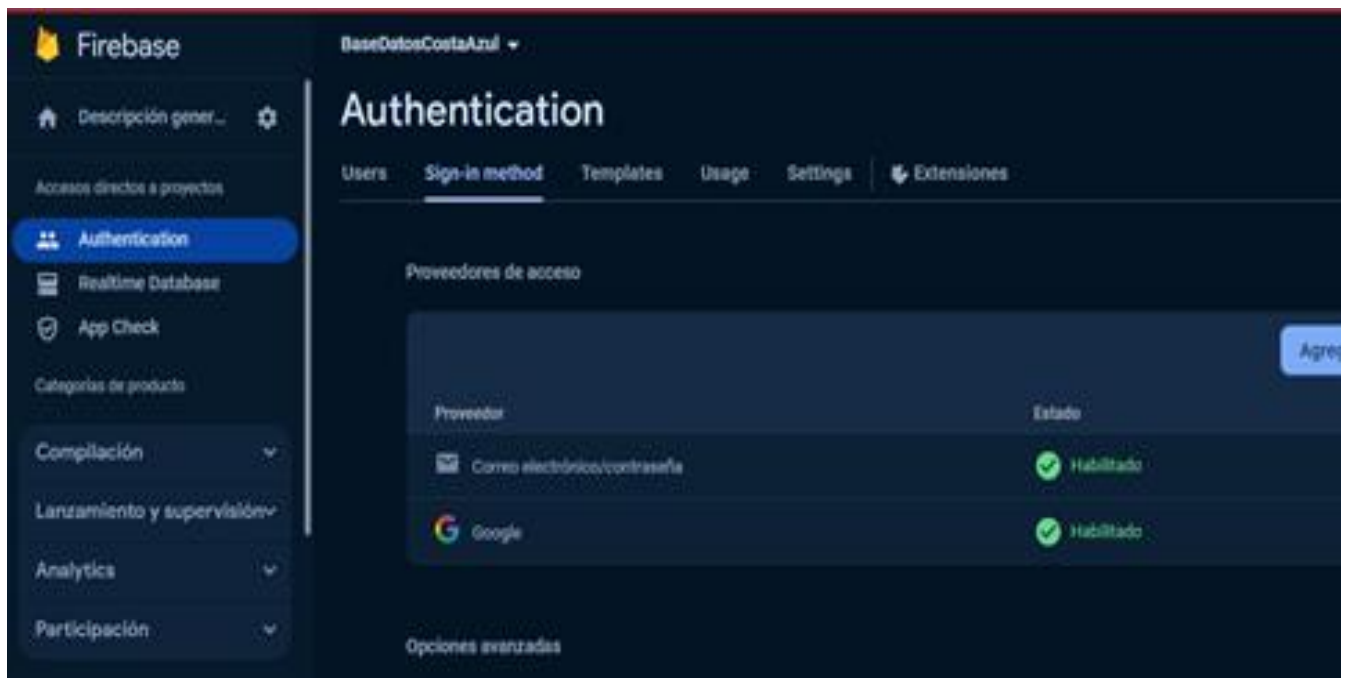
Por otra parte, Firebase hace uso de una base de datos en tiempo real, lo cual es especialmente útil para aplicaciones de Android, ya que permite la actualización instantánea de datos. Esto es crucial para proporcionar a los usuarios información en tiempo real sobre productos, clientes y empleados, etc.

Finalmente, Firebase Authentication ofrece métodos sencillos para implementar servicios de autenticación en una aplicación Android. Se puede integrar de forma sencilla opciones como inicio de sesión con correo electrónico, autenticación con Google, entre otros.

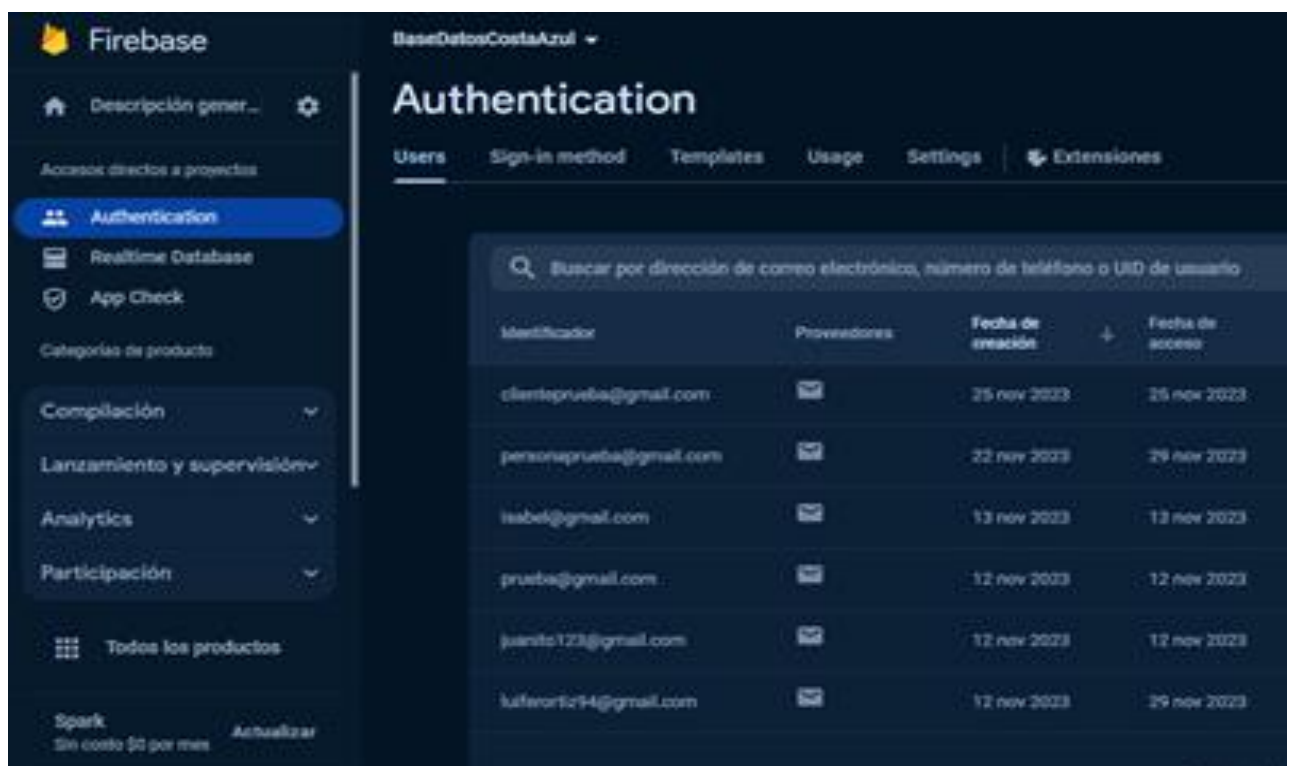


Registro y consulta de datos de tipo cliente.





Servicios activos de autenticación.



Registro de correos y contraseñas de usuarios de la App.

## CONTROLADORES:

### AuthController (Mediante Authentication en Firebase):

Se encarga de crear y administrar los inicios de sesión de Usuarios (clientes, Administrador, Empleado) mediante un token de inicio de sesión único. Todo este proceso con el fin de aumentar la seguridad de todos los usuarios.

- **Cliente:**

Se realiza el Sing In con Google generando un Token determinado que sirve para referenciar a este cliente en específico perteneciente a la base de datos, generando un "id" único y registrando los cambios que se realicen a este.

```

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

GoogleSignInOptions gso = new GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
    .requestIdToken("910607048212-h611b6vtgpsfn6gvg2p41en6db50ilf1.apps.goog...")
    .requestEmail()
    .build();

mGoogleSignInClient = GoogleSignIn.getClient( activity: this, gso);
}

1 usage
private void singInWithGoogle(){
    Intent singInIntent = mGoogleSignInClient.getSignInIntent();
    startActivityForResult(singInIntent, RC_SING_IN);
}

@Override
public void onActivityResult(int requestCode, int resultCode, Intent data){
    super.onActivityResult(requestCode, resultCode, data);

    if(requestCode == RC_SING_IN){
        Task<GoogleSignInAccount> task = GoogleSignIn.getSignedInAccountFromIntent(data);
        try {
            GoogleSignInAccount account = task.getResult(ApiException.class);
            firebaseAuthWithGoogle(account.getIdToken());
        } catch (Exception e) {
            // Handle the exception
        }
    }
}

```

Validación y verificación del usuario, comprueban que los datos ingresados a la aplicación cumplan con las condiciones necesarias para ser registrado como un usuario de la aplicación de forma correcta, en este caso utilizando la herramienta Usuario y contraseña.

```

1 usage
private void firebaseAuthWithGoogle(String idToken){
    AuthCredential credential = GoogleAuthProvider.getCredential(idToken, accessToken: null);
    mAuth.signInWithCredential(credential)
        .addOnCompleteListener( activity: this, new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if(task.isSuccessful()){
                    Intent intent = new Intent( packageContext: LoginActivity.this, MenuCliente.class);
                    startActivity(intent);
                    finish();
                }else{
                    Toast.makeText( context: LoginActivity.this, text: "Fallo en iniciar sesion", Toast.LENGTH_SHORT ).show();
                }
            }
        });
}

```

```

public void validate(){

    String email = emailEditText.getText().toString().trim();
    String password = contrasenaEditText.getText().toString().trim();

    if(email.isEmpty() || !Patterns.EMAIL_ADDRESS.matcher(email).matches()){

        emailEditText.setError("Correo No Valido");
        return;
    }else{
        emailEditText.setError(null);
    }

    if(password.isEmpty() || password.length() < 8){
        contrasenaEditText.setError("Minimo 8 caracteres");
        return;
    } else if (!Pattern.compile( regex: "[0-9]").matcher(password).find()) {

        contrasenaEditText.setError("Usa al menos un numero");
        return;
    }else{
        contrasenaEditText.setError(null);
    }

    iniciarSesion(email,password);
}

```

Validación de datos ingresados por el usuario.

- **Administrador/Empleado:**

Define los métodos con los cuales la aplicación interactúa con las cuentas de Administrador y Empleado en el proceso de registro autorizando el acceso a la información dependiendo del Authcontroller que envía un Token único gracias a Firebase:

```

1 usage
private void firebaseAuthWithGoogle(String idToken){
    AuthCredential credential = GoogleAuthProvider.getCredential(idToken, accessToken: null);
    mAuth.signInWithCredential(credential)
        .addOnCompleteListener( activity: this, new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if(task.isSuccessful()){
                    Intent intent = new Intent( packageContext: LoginEmpleadoActivity.this, MenuEmpleadoActivity.class);
                    startActivity(intent);
                    finish();
                }else{
                    Toast.makeText( context: LoginEmpleadoActivity.this, text: "Fallo en iniciar sesion",Toast.LENGTH_SHORT ).show()
                }
            }
        });
}
}

```

Administrador de inicio de sesión de Administrador/Empleados.

## MODELO DE LA APP:

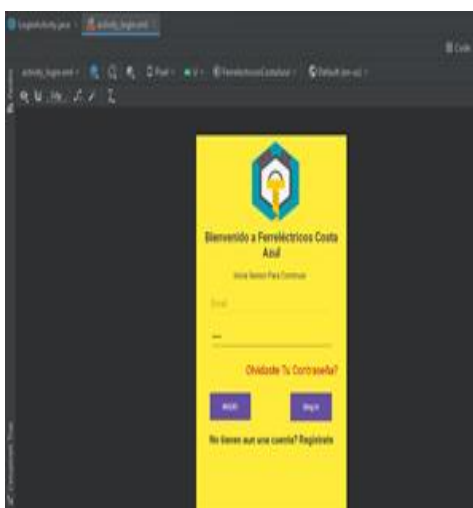
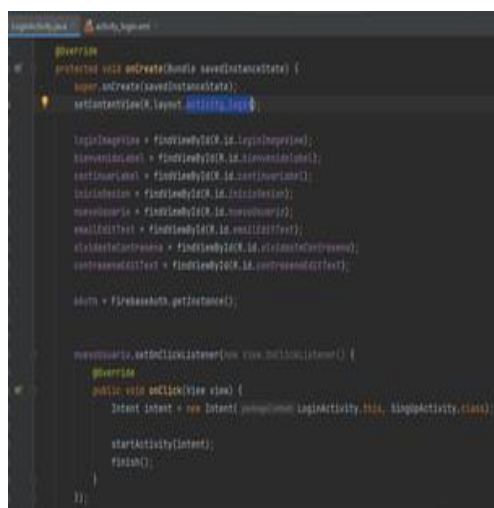
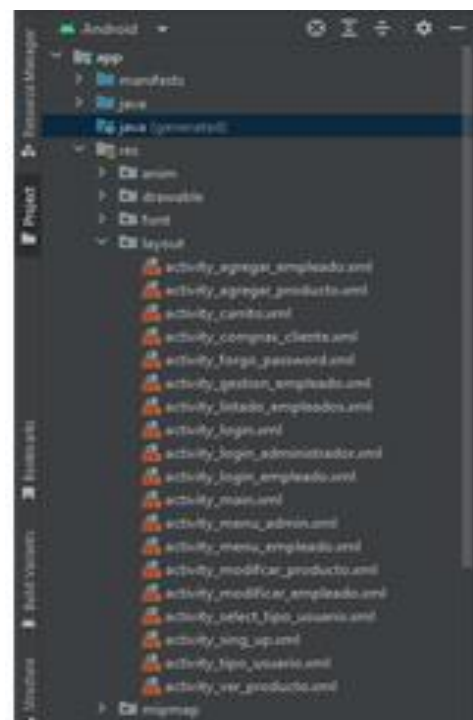
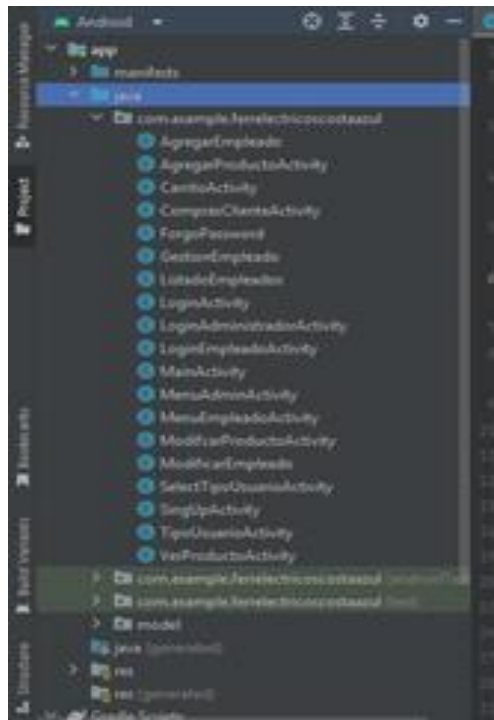
Una Activity en Android es un componente de la aplicación que representa una pantalla con la cual el usuario puede interactuar.

Cada pantalla visible es gestionada por una Activity.

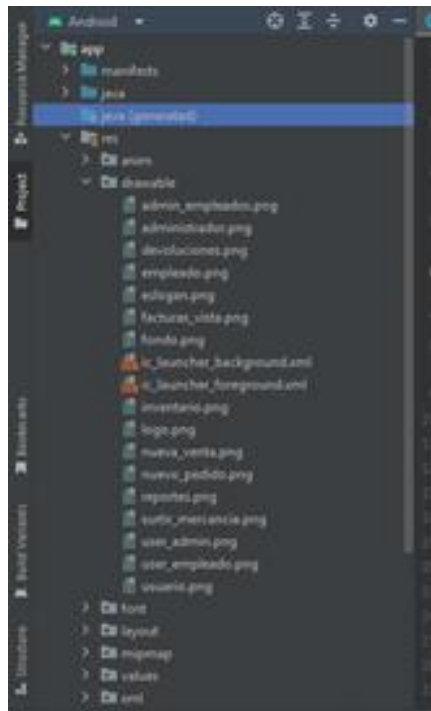
En el modelo de la App se hizo uso de Activity (Java) y XML Layout, directamente relacionadas entre sí. Los archivos XML de diseño (layout) definen la estructura y la

apariciencia de la interfaz de usuario, incluyendo vistas como botones, campos de texto, imágenes, etc. La interacción con las vistas, como manejar eventos de clic en un botón, se realiza en el código Java de la actividad.

Las siguientes son las Activities de la App.

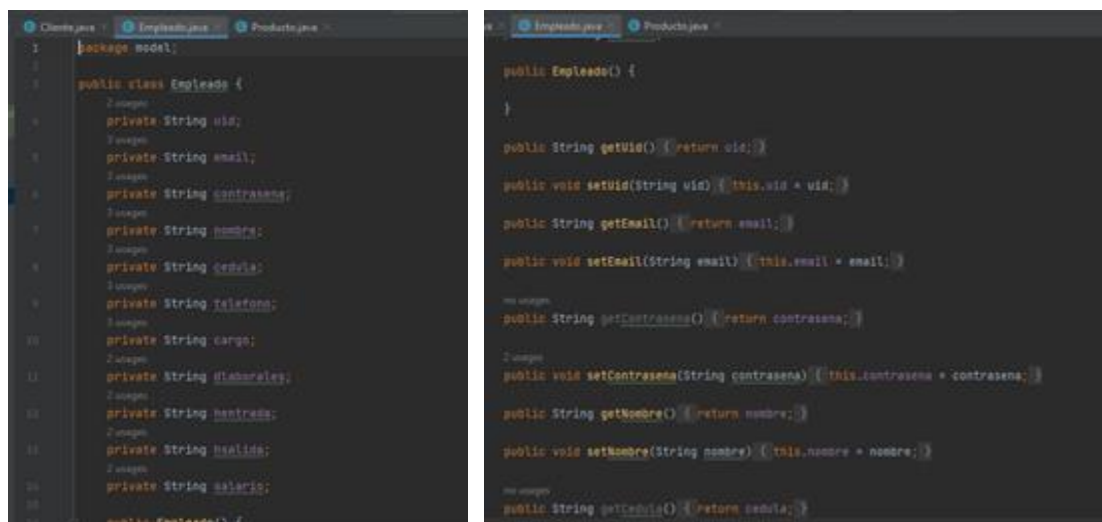


Por otra parte, al hacer uso de imágenes, es necesario incluir estas en la carpeta drawable.

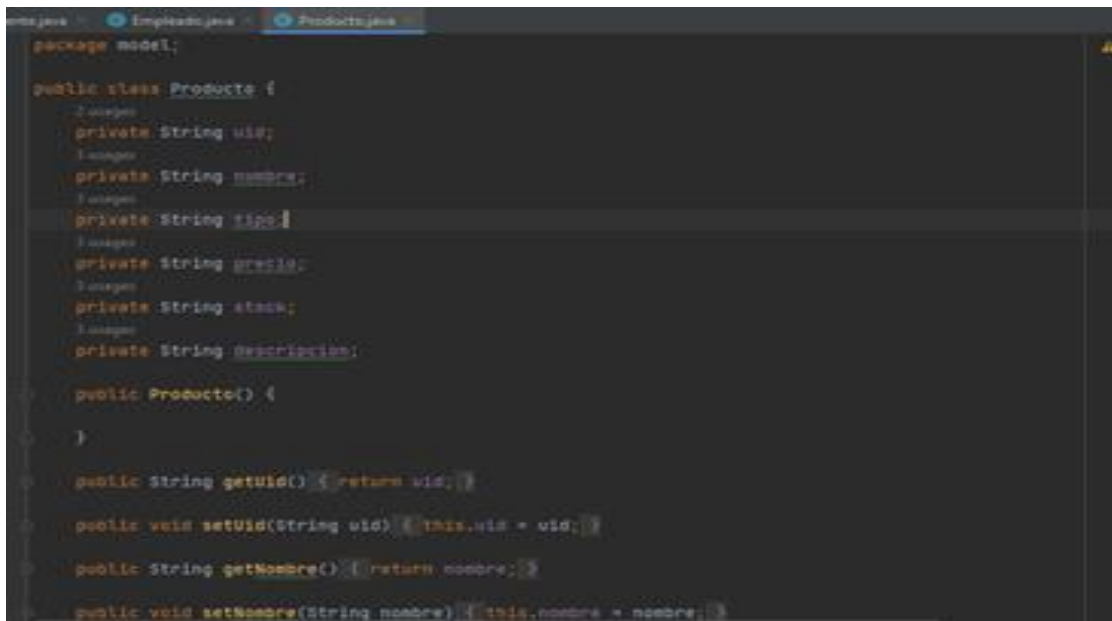


Repositorio de imágenes.

Para las clases java Empleado, Cliente y Producto se le definieron sus respectivos atributos que mediante el uso de su respectivo constructor y los métodos Setters y Getters, estos atributos son modificados o consultados por el usuario en los diferentes casos de uso de estos.



Clase Empleado.



```

package model;

public class Producto {
    //atributos
    private String uid;
    //atributos
    private String nombre;
    //atributos
    private String tipo;
    //atributos
    private String precio;
    //atributos
    private String stock;
    //atributos
    private String descripcion;

    public Producto() {
    }

    public String getUid() {return uid;}

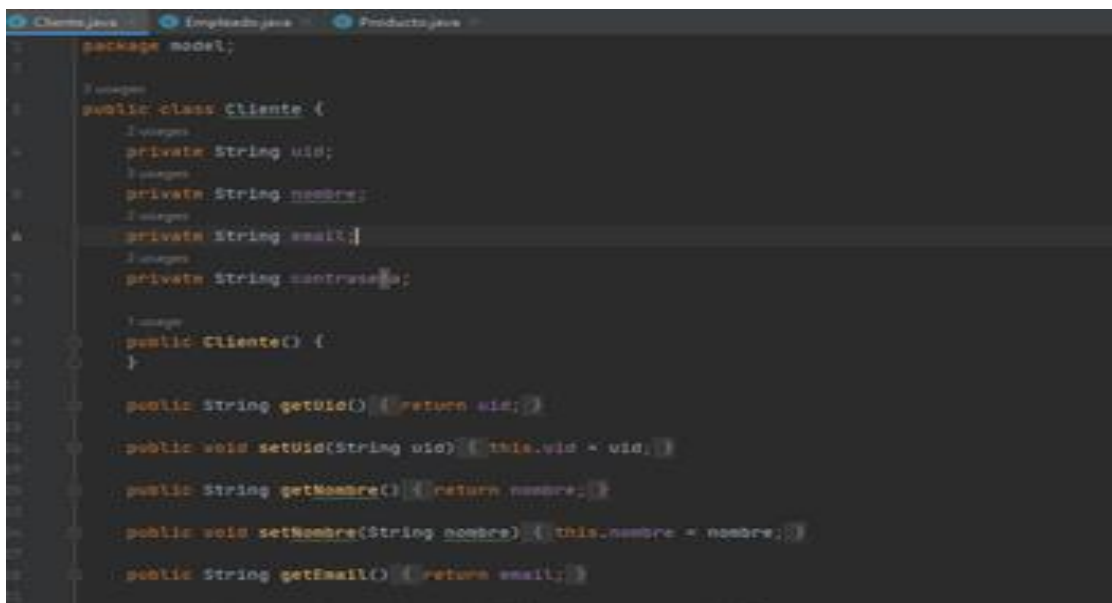
    public void setUid(String uid) {this.uid = uid;}

    public String getNombre() {return nombre;}

    public void setNombre(String nombre) {this.nombre = nombre;}
}

```

**Clase Producto.**



```

package model;

public class Cliente {
    //atributos
    private String uid;
    //atributos
    private String nombre;
    //atributos
    private String email;
    //atributos
    private String contraseña;

    public Cliente() {
    }

    public String getUid() {return uid;}

    public void setUid(String uid) {this.uid = uid;}

    public String getNombre() {return nombre;}

    public void setNombre(String nombre) {this.nombre = nombre;}

    public String getEmail() {return email;}
}

```

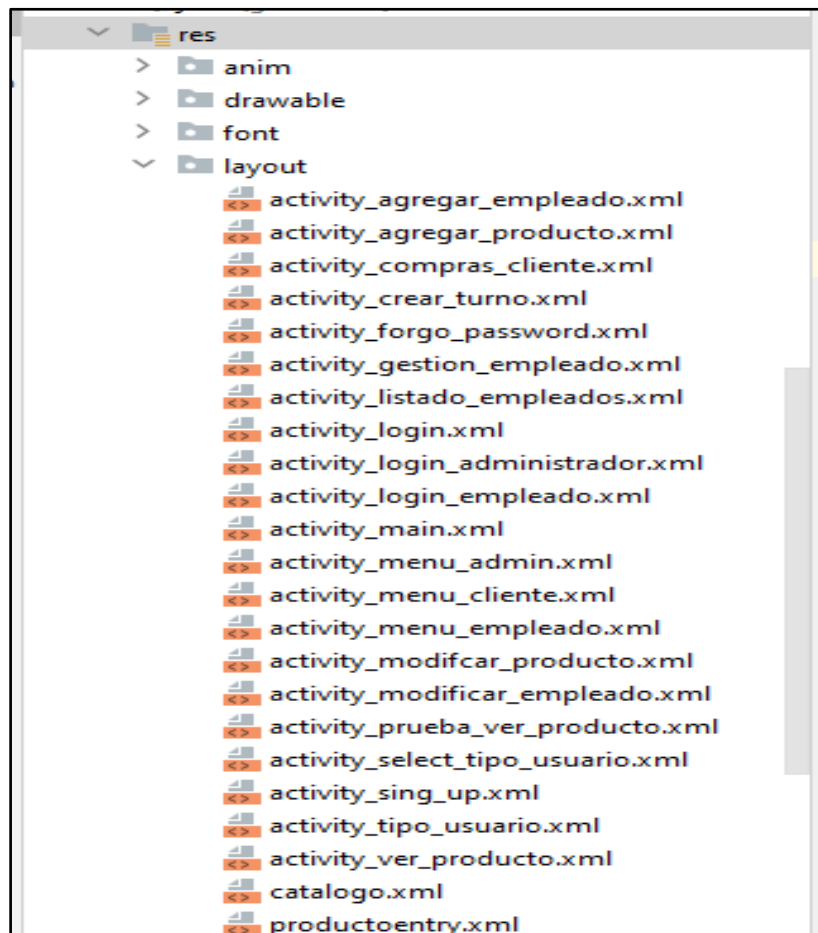
**Clase Cliente.**

## ACTIVIDADES VIEW:

### Herramientas:

Cada Activity en Android Studio está asociada a una interfaz gráfica específica. En nuestro caso los principales elementos que involucran recursos gráficos consisten en formularios de llenado de información, es decir, ingresar datos, y listas de visualización y modificación que interactúan entre las diferentes necesidades de los usuarios.

## Paquete de recursos gráficos de Android Studio



## Formularios Ingreso de Datos:

### Codificación formularios

```
<TextView
    android:id="@+id/textView7"
    android:layout_width="81dp"
    android:layout_height="8dp"
    android:layout_marginTop="296dp"
    android:text="Precio"
    android:textSize="20sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.048"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<TextView
    android:id="@+id/textView8"
    android:layout_width="81dp"
    android:layout_height="8dp"
    android:layout_marginTop="380dp"
    android:text="Stock"
    android:textSize="20sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.051"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

### Visualización formularios

Agregar Producto

|             |             |
|-------------|-------------|
| Nombre      | Nombre      |
| Tipo        | Tipo        |
| Precio      | Precio      |
| Stock       | Stock       |
| Descripción | Descripción |

Registrar



```

<EditText
    android:id="@+id/descripcionEditText"
    android:layout_width="265dp"
    android:layout_height="52dp"
    android:layout_marginTop="20dp"
    android:ems="10"
    android:hint="Descripcion"
    android:inputType="text"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.906"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/stockEditText" />

<EditText
    android:id="@+id/precioEditText"
    android:layout_width="300dp"
    android:layout_height="65dp"
    android:layout_marginTop="8dp"
    android:ems="10"
    android:hint="Precio"
    android:inputType="text"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.921"
    app:layout_constraintStart_toStartOf="parent"

```

## Codificación listas

```

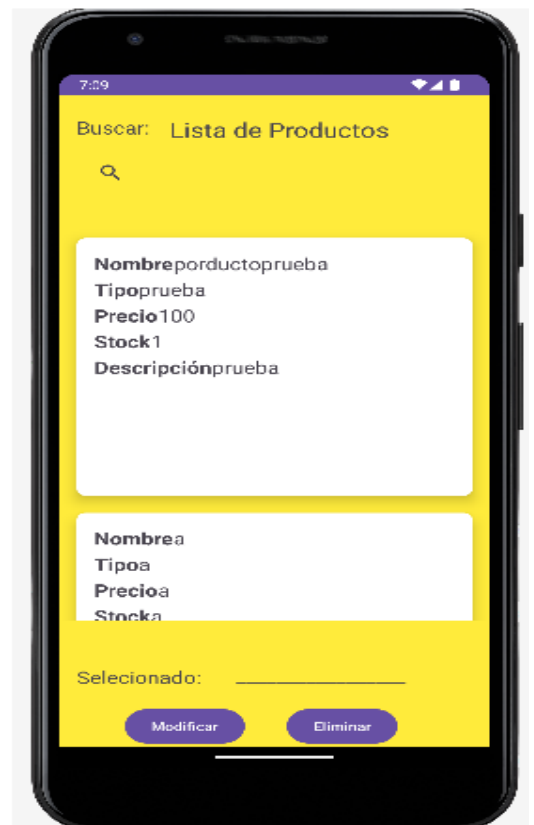
<TextView
    android:id="@+id/textView5"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_marginTop="24dp"
    android:text="Buscar:"
    android:textSize="20sp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/RecyclerView"
    android:layout_width="385dp"
    android:layout_height="456dp"
    android:layout_marginBottom="108dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/BarraBuscador" />

</androidx.constraintlayout.widget.ConstraintLayout>

```

## Visualización listas



## Codificación menús

```
<ImageView
    android:id="@+id/surtirproductoImageView"
    android:layout_width="134dp"
    android:layout_height="114dp"
    android:layout_marginStart="43dp"
    android:layout_marginTop="12dp"
    android:layout_marginEnd="52dp"
    android:layout_marginBottom="8dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toStartOf="@+id/inventarioImageView"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/surtirproductoTextView"
    app:srcCompat="@drawable/surtir_mercancia" />

<ImageView
    android:id="@+id/inventarioImageView"
    android:layout_width="127dp"
    android:layout_height="109dp"
    android:layout_marginTop="12dp"
    android:layout_marginEnd="68dp"
    android:layout_marginBottom="13dp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/inventarioTextView"
```

## Visualización menús

