Corporación Brothersoft

Informe técnico aplicación Android

Contenido

[1. Creamos una aplicación Android 2](#_Toc53422397)

[2. Abrir la aplicación 5](#_Toc53422398)

[3. Agregamos una Actividad 6](#_Toc53422399)

[4. Ponemos la actividad Login como prima ventana que se carga 7](#_Toc53422400)

[5. Insertamos una clase Singleton y configuración de aplicación 9](#_Toc53422401)

[6. Agregar la librería Voley (peticiones Get y Post ) para hacer peticiones http al web Api 12](#_Toc53422402)

[7. Diseño de formulario 13](#_Toc53422403)

[8. Estructura de archivos 14](#_Toc53422404)

[9. CambioContrasenaActivity 15](#_Toc53422405)

[10. HistorialActivity 16](#_Toc53422406)

[11. InvitadoActivity 17](#_Toc53422407)

[12. LoginActivity 18](#_Toc53422408)

[13. MainActivity 19](#_Toc53422409)

[14. PagarActivity 20](#_Toc53422410)

[15. RecuperarContrasenaActivity 21](#_Toc53422411)

[16. RegistroActivity 22](#_Toc53422412)

[17. ResultadomalActivity 23](#_Toc53422413)

[18. ResultadoMalInvitadoActivity 24](#_Toc53422414)

[19. ResultadookActivity 25](#_Toc53422415)

[20. ResultadoOKInvitadoActivity 26](#_Toc53422416)

[21. ListItemPagos 27](#_Toc53422417)

[22. ListItemHistorial 28](#_Toc53422418)

[23. TerminoCondicionesDialog 29](#_Toc53422419)

[24. TextCaptcha 29](#_Toc53422420)

[25. Singleton 29](#_Toc53422421)

[26. Captcha 30](#_Toc53422422)

[27. Recibo 30](#_Toc53422423)

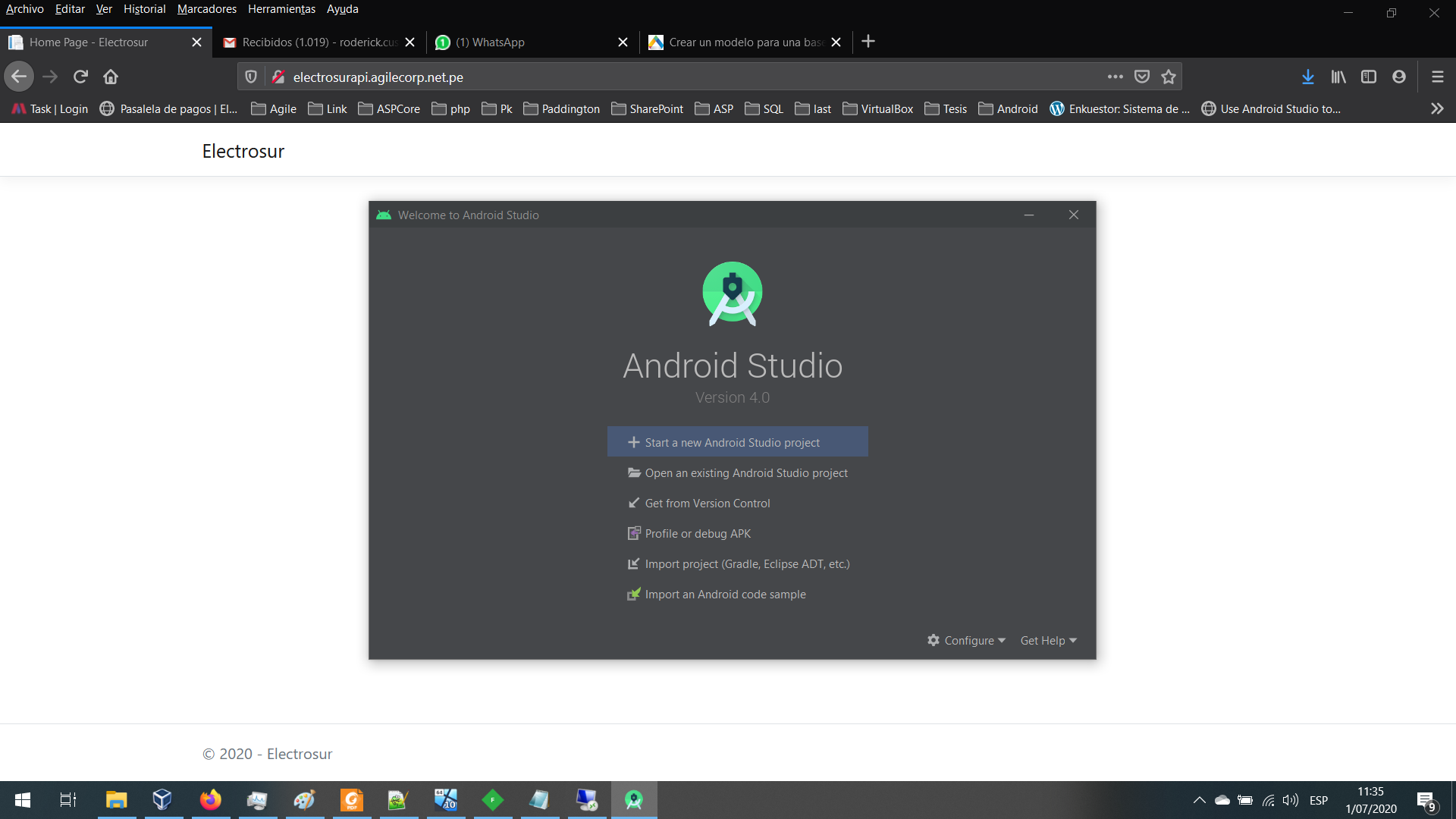
[28. AndroidManifest 30](#_Toc53422424)

[29. Recursos drawable 31](#_Toc53422425)

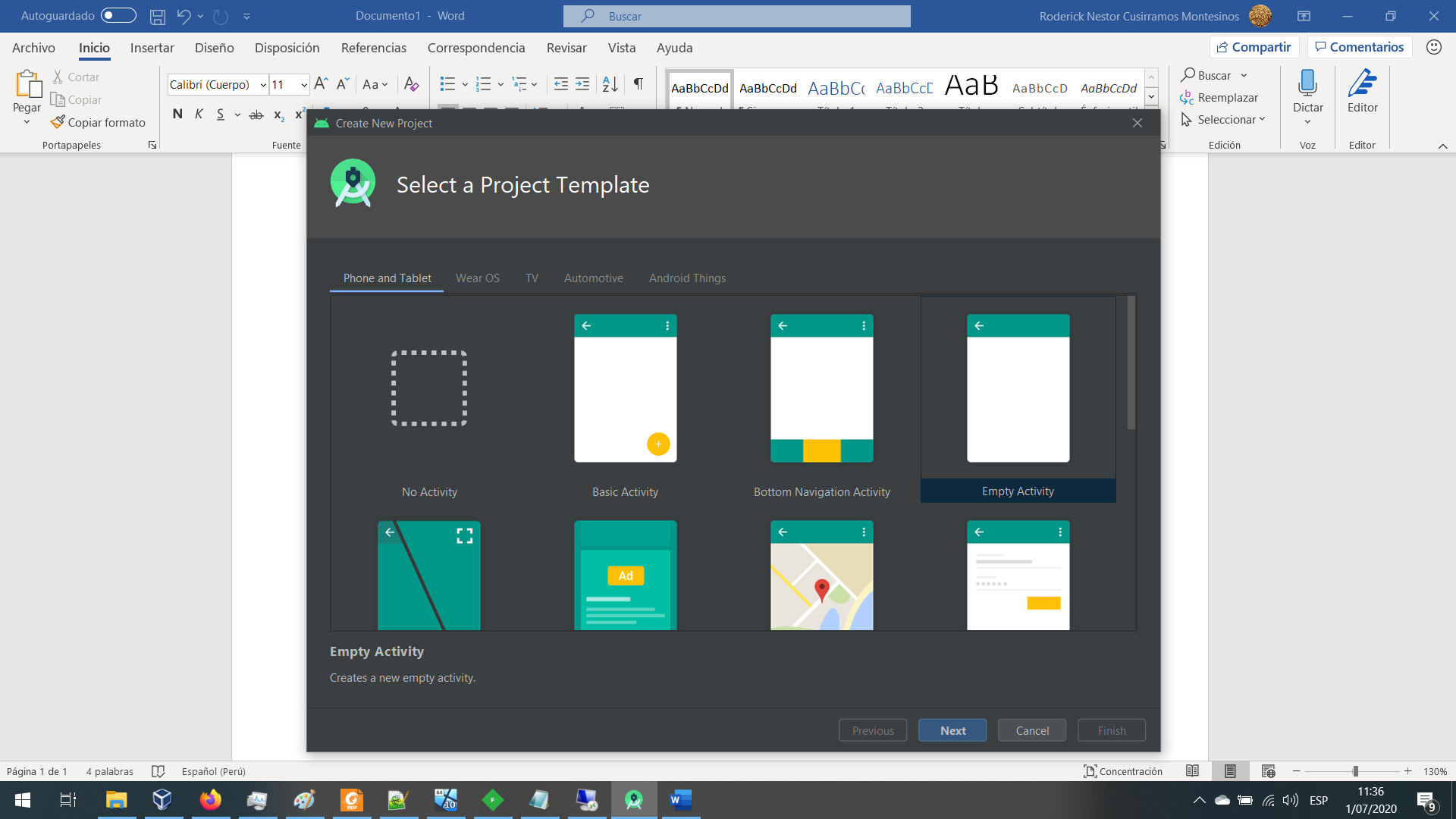
[30. Recursos layout 32](#_Toc53422426)

# Creamos una aplicación Android

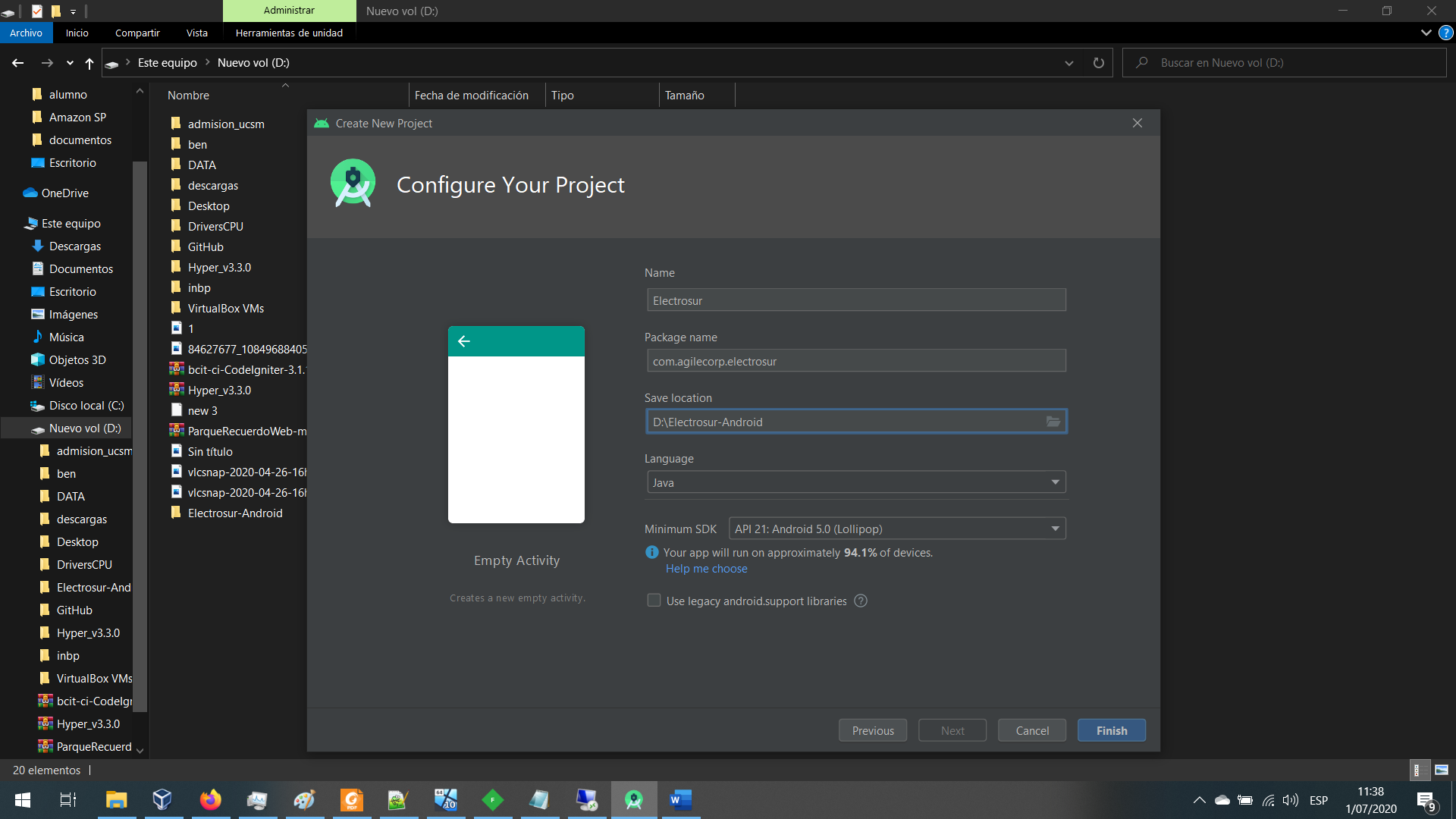
* Herramienta Android Studio
* Versión 4.0
* Link de descarga: https://developer.android.com/studio



* Escogemos una plantilla



* Configuración del proyecto



Nombre

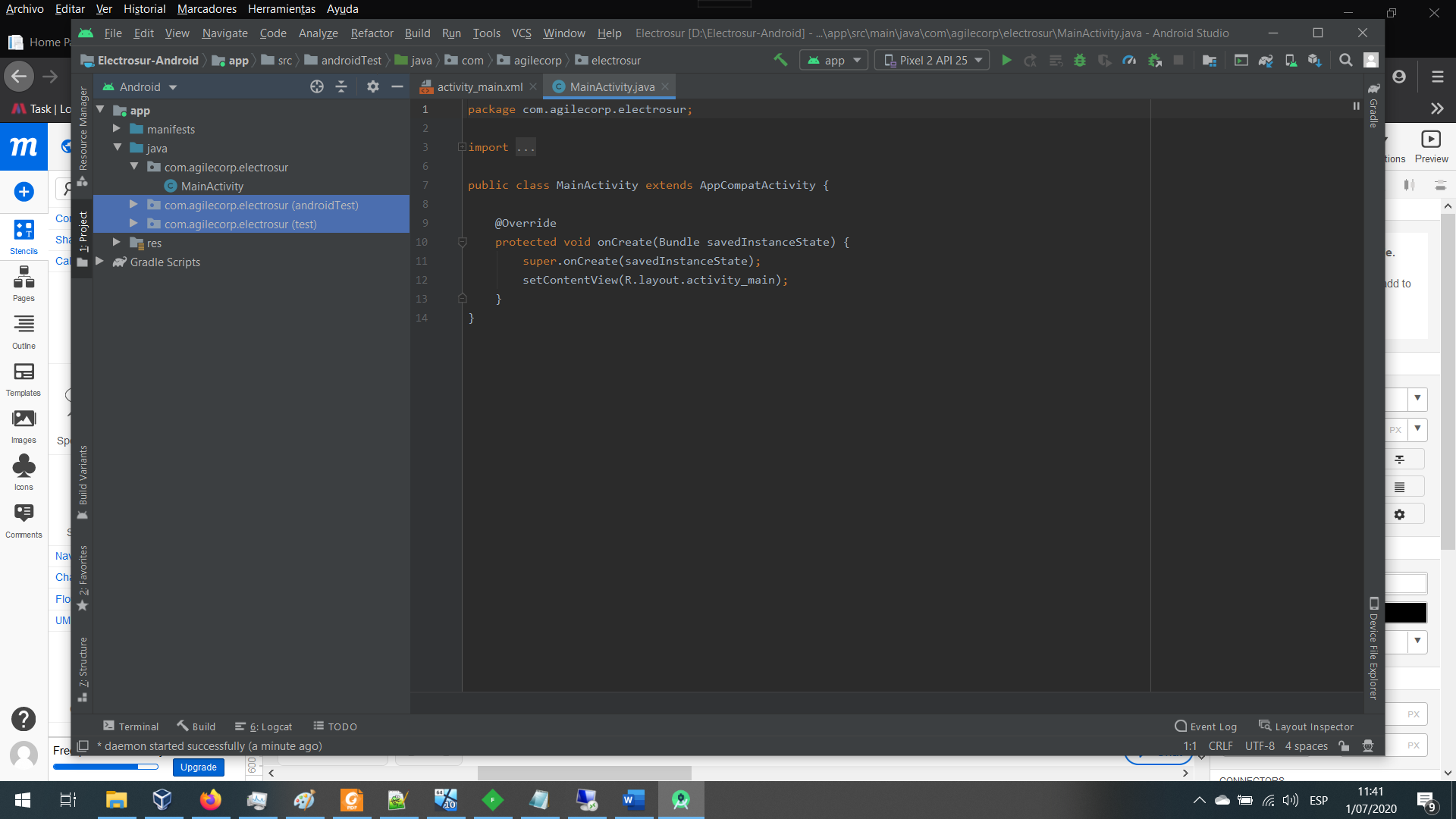
Nombre del paquete

Localización

Lenguaje

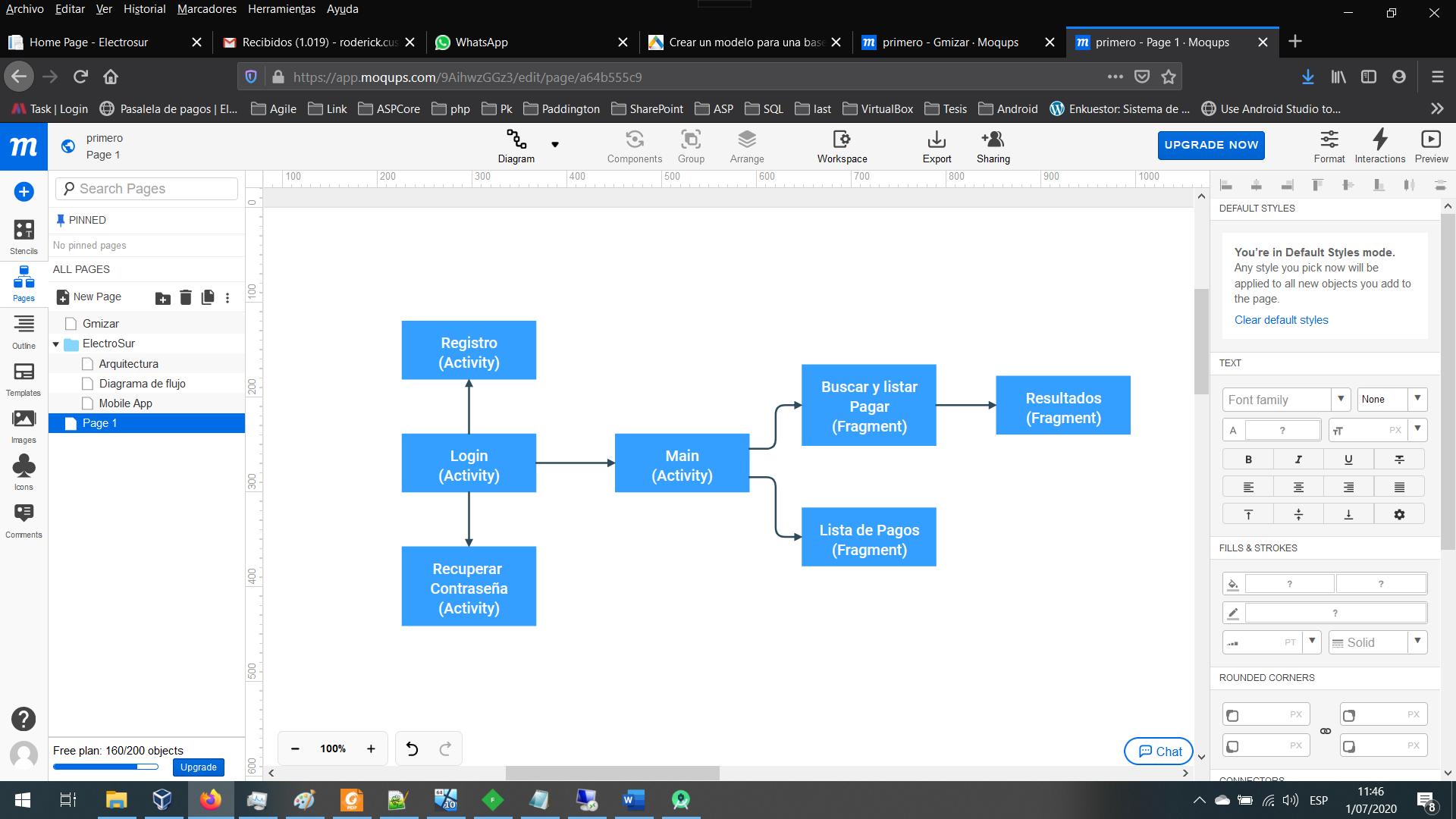
Mínimo SDK

* Proyecto



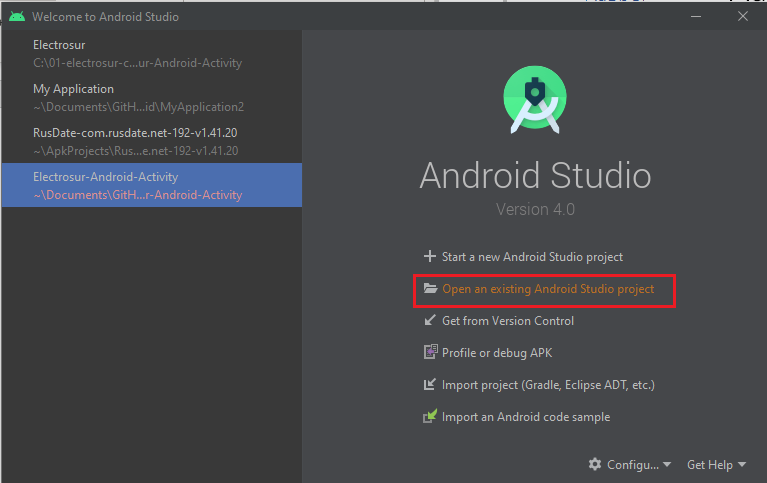
Eliminamos los paquetes de Test

Estructura de la aplicación

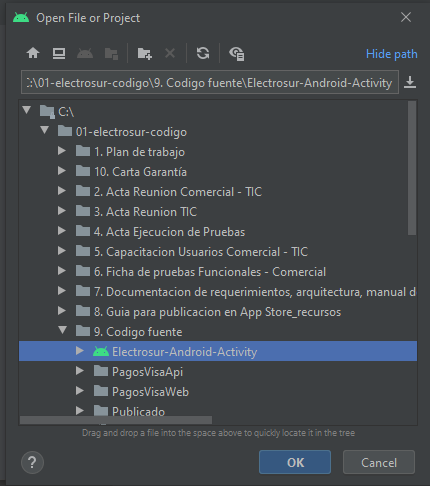


# Abrir la aplicación

Abrimos la aplicación y en la opción “Open an existing Android Studio project”



Seleccionamos la carpeta donde está la aplicación Android (“Electrosur-Android-Activity”)



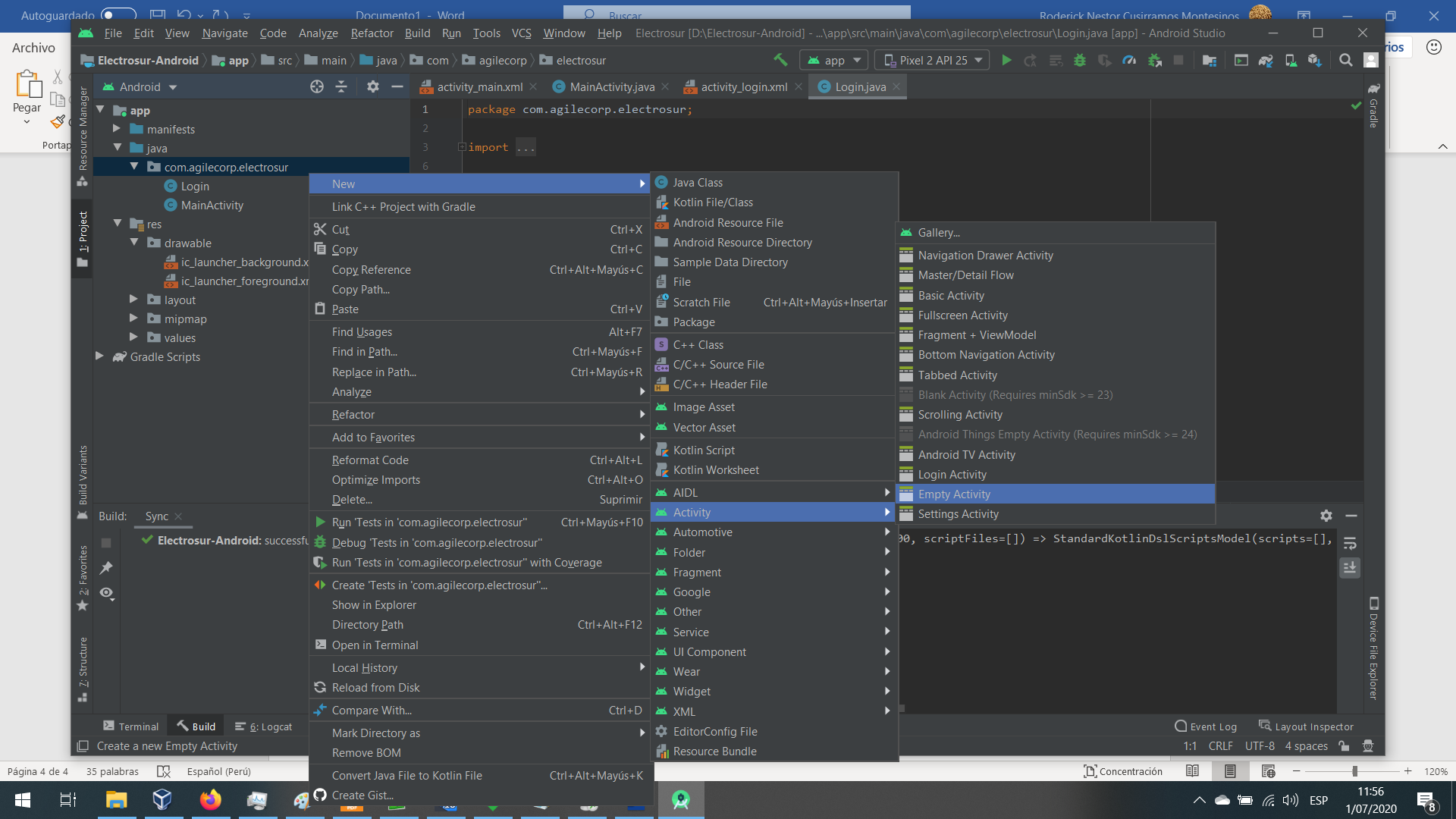
# Agregamos una Actividad

Una actividad es una pantalla donde podemos agregar controladores, una actividad está conformada por una clase y por un layout.

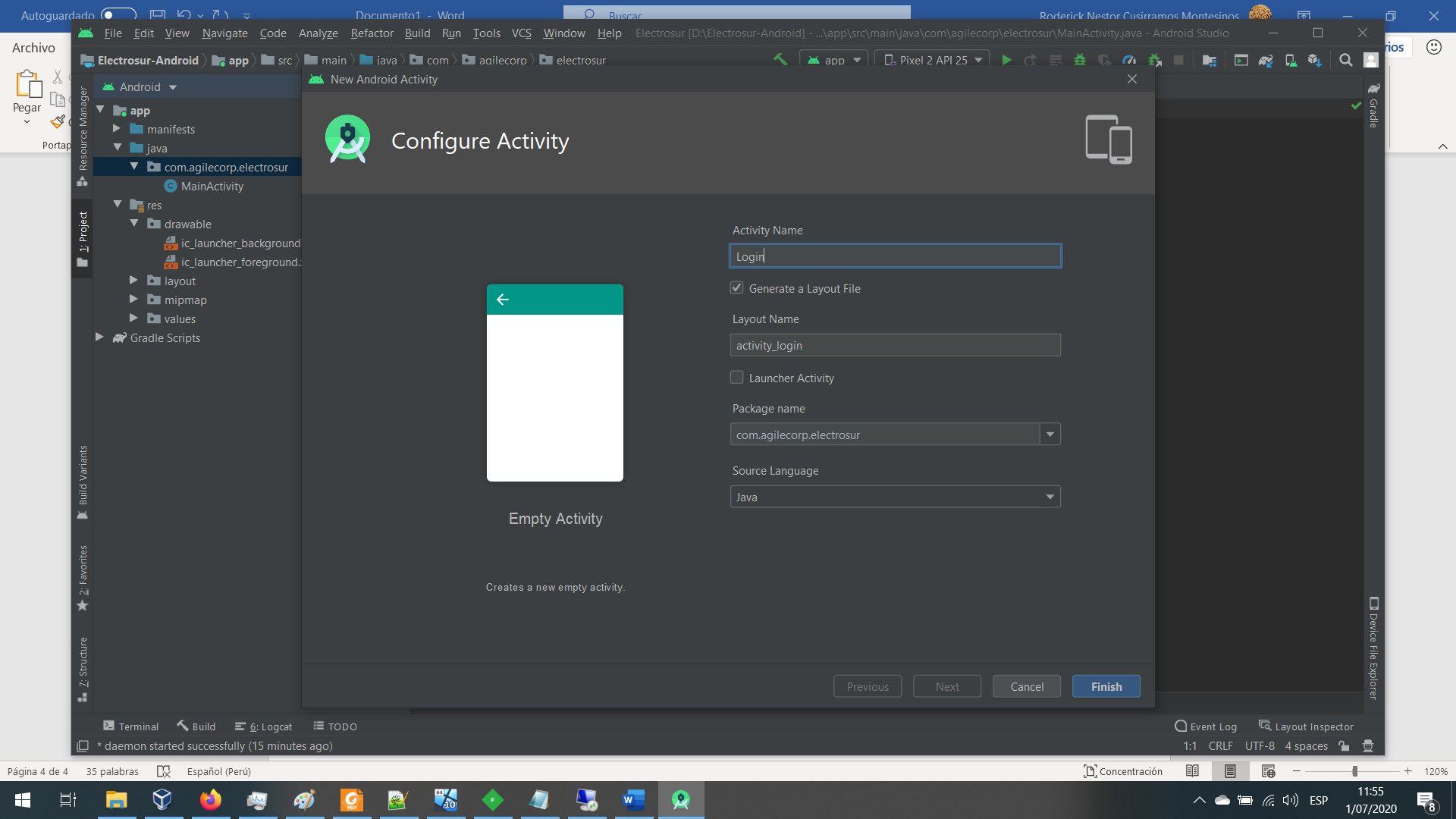
Una clase: es donde agregamos funcionalidad.

Layout: es la parte grafica donde podemos arrastrar componentes para generar nuestra pantalla.

Agregando una actividad.



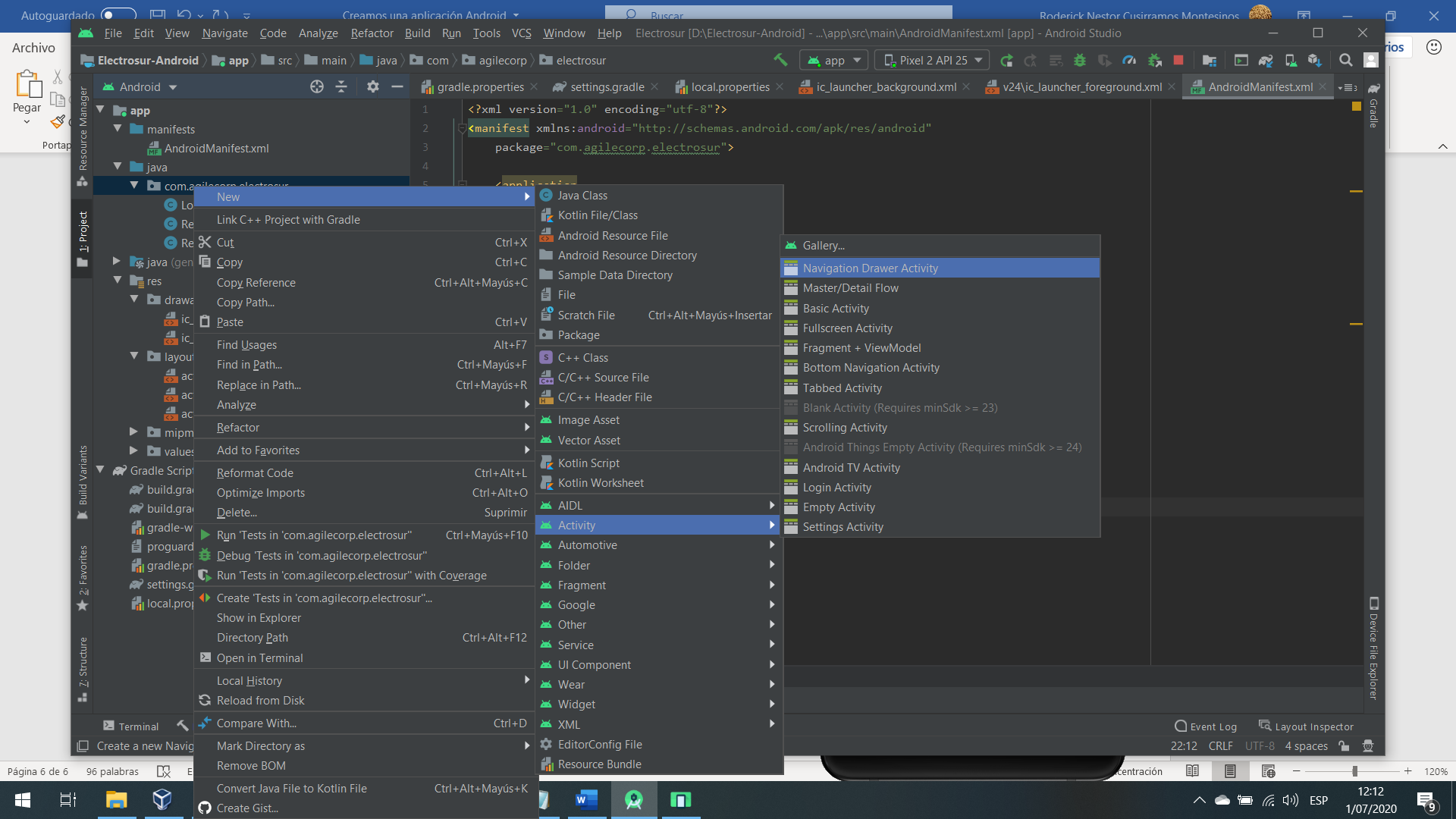
Configuracion de una actividad.

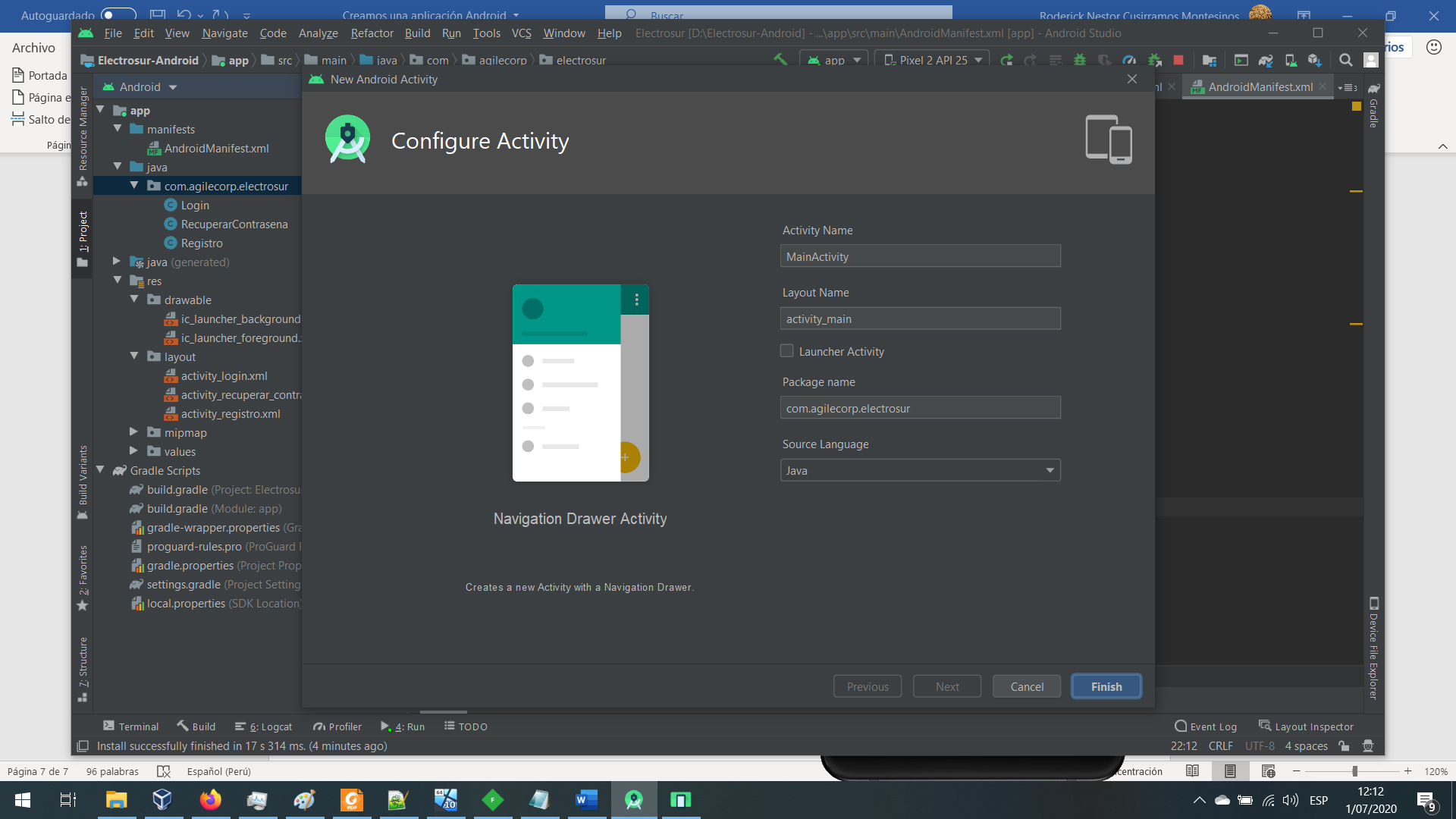


# Ponemos la actividad Login como prima ventana que se carga

|  |
| --- |
| <?xml version="1.0" encoding="utf-8"?>  <manifest xmlns:android="http://schemas.android.com/apk/res/android"  package="com.agilecorp.electrosur">  <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/AppTheme">  <activity android:name=".RecuperarContrasena"></activity>  <activity android:name=".Registro" />  <activity android:name=".MainActivity" />  <activity android:name=".Login">  <intent-filter>  <action android:name="android.intent.action.MAIN" />  <category android:name="android.intent.category.LAUNCHER" />  </intent-filter>  </activity>  </application>  <uses-permission android:name="android.permission.INTERNET" />  <uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />  </manifest> |

Eliminamos la actividad main y creamos una actividad con el mismo nombre, pero de tipo navigaton bar

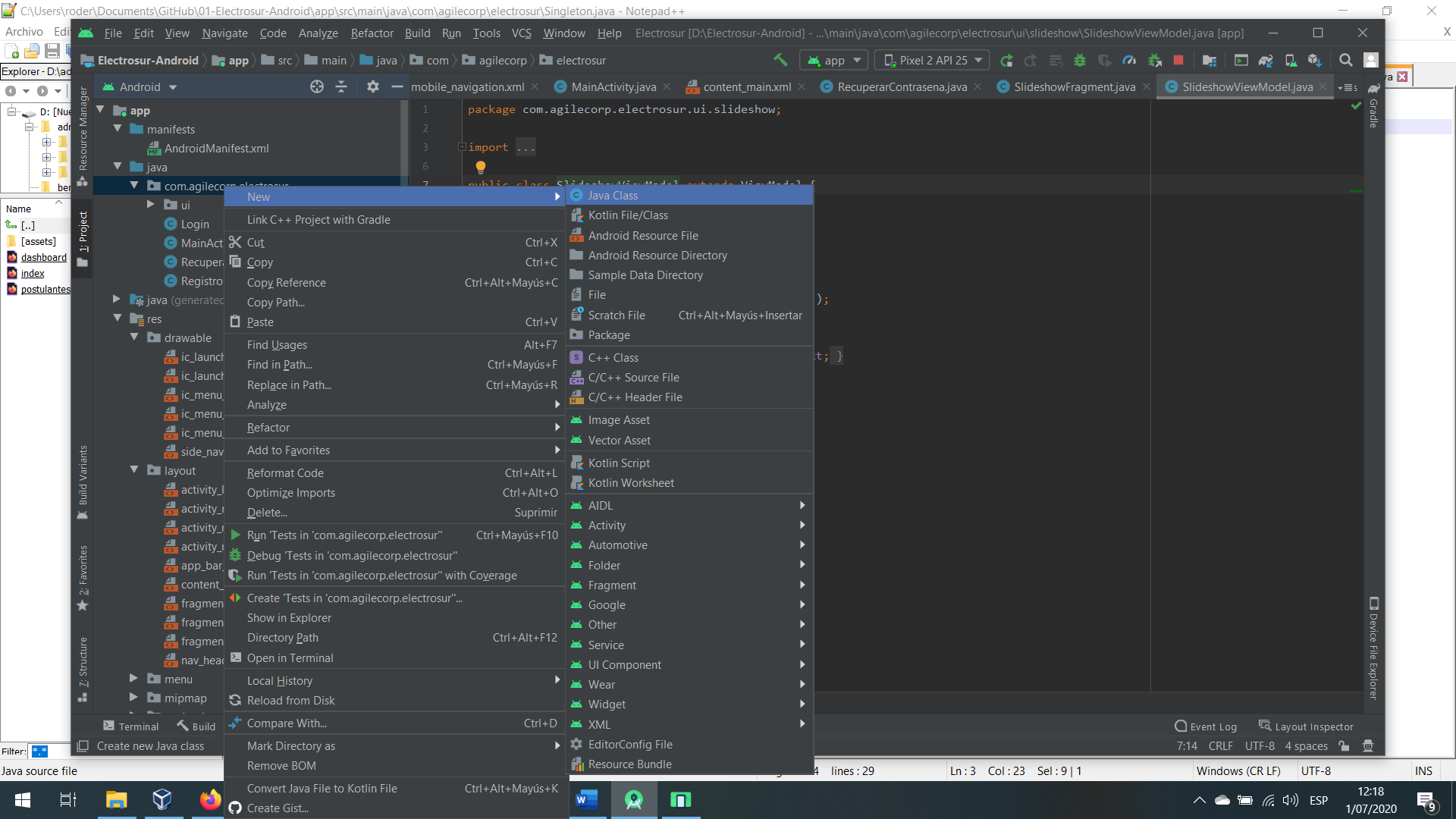




# Insertamos una clase Singleton y configuración de aplicación

Singleton o instancia única es un [patrón de diseño](https://es.wikipedia.org/wiki/Patr%C3%B3n_de_dise%C3%B1o) que permite restringir la creación de [objetos](https://es.wikipedia.org/wiki/Programaci%C3%B3n_orientada_a_objetos) pertenecientes a una [clase](https://es.wikipedia.org/wiki/Clase_(programaci%C3%B3n_orientada_a_objetos)) o el valor de un tipo a un único objeto.

Su intención consiste en garantizar que una clase solo tenga una [instancia](https://es.wikipedia.org/wiki/Instancia_(inform%C3%A1tica)) y proporcionar un punto de acceso global a ella.



Código

En esta clase debemos configurar la url del web api, los parámetros de visa y la clave secreta.

|  |
| --- |
| public class Singleton {  String Name, Email, UniqueId, Token, Url, CodigoCliente;  String VisaNet\_USERNAME, VisaNet\_PASSWORD, VisaNet\_MERCHANT,VisaNet\_VISANET\_ENDPOINT\_URL;  Recibo recibo;  String descripcion, trasaccion, card, brand, ClaveSecretaMovil;  private static final Singleton instance = new Singleton();  public static Singleton getInstance() {  return instance;  }  public String getClaveSecretaMovil() {  return ClaveSecretaMovil;  }  private Singleton() {  Url = "https://pagos.electrosur.com.pe/appApi/";  //Url = "http://electrosurapi.agilecorp.net.pe/";  VisaNet\_USERNAME = "integraciones.visanet@necomplus.com";  VisaNet\_VISANET\_ENDPOINT\_URL = "https://apitestenv.vnforapps.com/";  VisaNet\_PASSWORD = "d5e7nk$M";  VisaNet\_MERCHANT = "100128038";  ClaveSecretaMovil = "zxczc2vz3v215fg17zxczc2vz3v215fg17zxczc2vz3v215fg17zxczc2vz3v215fg17zxczc2vz3v215fg17";  }  public String getUrl() {  return Url;  }  public String VisaNetUSERNAME() {  return VisaNet\_USERNAME;  }  public String VisaNetPASSWORD() {  return VisaNet\_PASSWORD;  }  public String VisaNetMERCHANT() {  return VisaNet\_MERCHANT;  }  public String VisaNetVISANETENDPOINTURL() {  return VisaNet\_VISANET\_ENDPOINT\_URL;  }  public void setName(String Name) {  this.Name = Name;  }  public String getName() {  return Name;  }  public void setEmail(String Email) {  this.Email = Email;  }  public String getEmail() {  return Email;  }  public void setUniqueId(String editValue) {  this.UniqueId = editValue;  }  public String getUniqueId() {  return UniqueId;  }  public void setToken(String Token) {  this.Token = Token;  }  public String getToken() {  return Token;  }  public void setRecibo(Recibo recibo) {  this.recibo = recibo;  }  public Recibo getRecibo() {  return recibo;  }  public String getDescripcion() {  return descripcion;  }  public void setDescripcion(String descripcion) {  this.descripcion = descripcion;  }  public String getTrasaccion() {  return trasaccion;  }  public void setTrasaccion(String trasaccion) {  this.trasaccion = trasaccion;  }  public String getCard() {  return card;  }  public void setCard(String card) {  this.card = card;  }  public String getBrand() {  return brand;  }  public void setBrand(String brand) {  this.brand = brand;  }  public String getCodigoCliente() {  return CodigoCliente;  }  public void setCodigoCliente(String CodigoCliente) {  this.CodigoCliente = CodigoCliente;  }  } |

# Agregar la librería Voley (peticiones Get y Post ) para hacer peticiones http al web Api

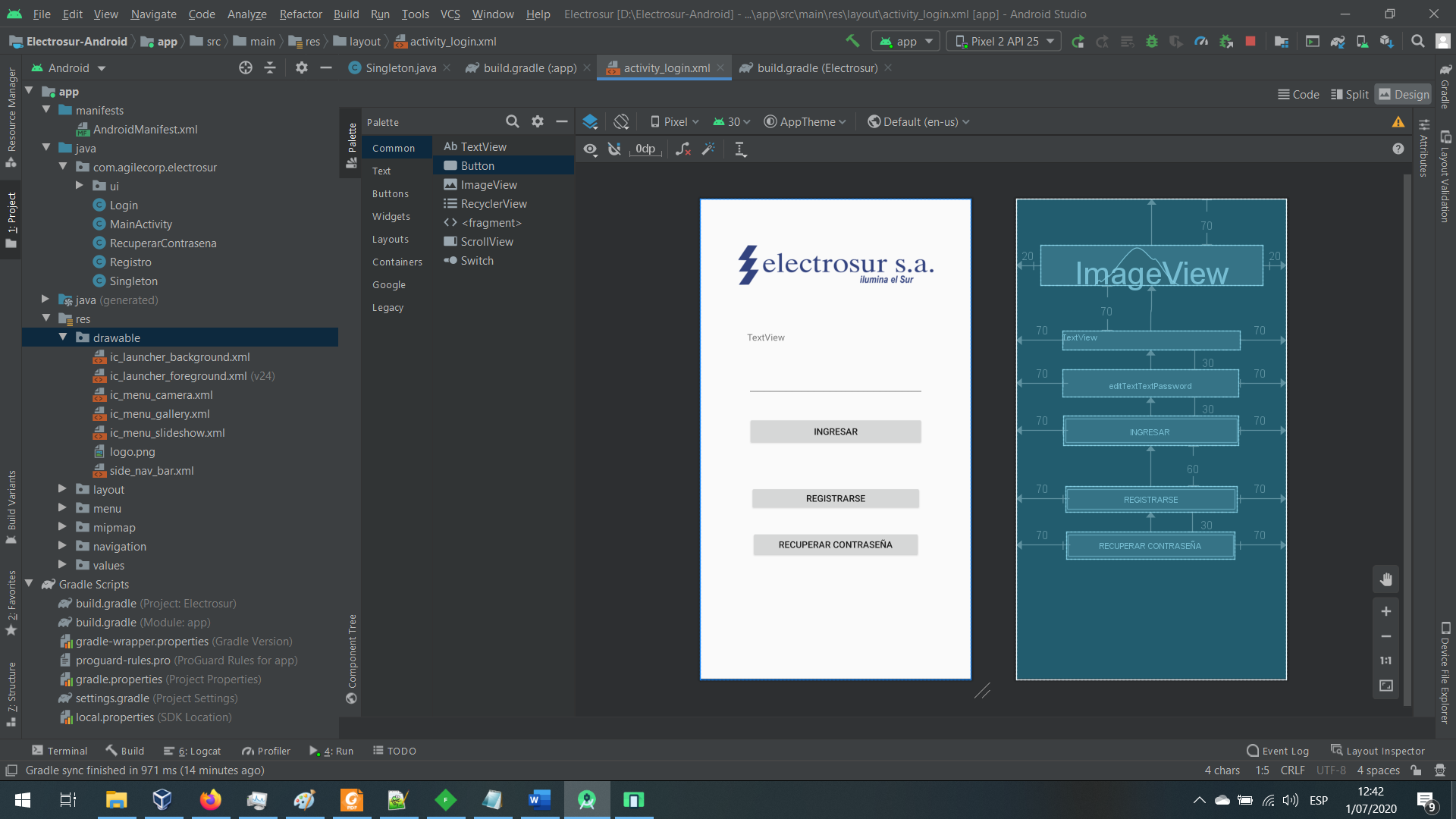
En el archive Build.gradle esta Liberia

implementation 'com.android.volley:volley:1.1.1'

|  |
| --- |
| apply plugin: 'com.android.application'  android {  compileSdkVersion 30  buildToolsVersion "30.0.0"  defaultConfig {  applicationId "com.agilecorp.electrosur"  minSdkVersion 21  targetSdkVersion 30  versionCode 1  versionName "1.0"  testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"  }  buildTypes {  release {  minifyEnabled false  proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'  }  }  }  dependencies {  implementation fileTree(dir: "libs", include: ["\*.jar"])  implementation 'androidx.appcompat:appcompat:1.1.0'  implementation 'androidx.constraintlayout:constraintlayout:1.1.3'  implementation 'androidx.legacy:legacy-support-v4:1.0.0'  implementation 'com.google.android.material:material:1.1.0'  implementation 'androidx.navigation:navigation-fragment:2.2.2'  implementation 'androidx.navigation:navigation-ui:2.2.2'  implementation 'androidx.lifecycle:lifecycle-extensions:2.2.0'  implementation 'com.android.volley:volley:1.1.1'    testImplementation 'junit:junit:4.12'  androidTestImplementation 'androidx.test.ext:junit:1.1.1'  androidTestImplementation 'androidx.test.espresso:espresso-core:3.2.0'  } |

# Diseño de formulario

Como indicamos cada actividad tiene un layout, como muestra en la imagen siguiente donde podmeos agregar controles.



# Estructura de archivos

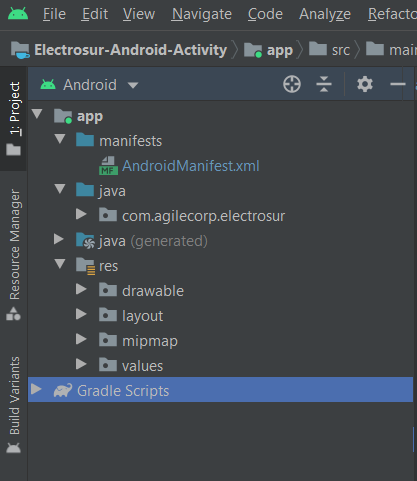
Una actividad representa una sola pantalla con una interfaz de usuario como una ventana.

Estructura de una aplicación Android

En la carpeta java es donde está el código fuente(clases).

Carpeta res/layout donde podemos encontrar los layout por cada clase de tipo Activity.

Carpeta res/drawable donde podemos encontrar iconos e imágenes.



# CambioContrasenaActivity

|  |  |
| --- | --- |
| **Tipo** | Activity |
| **Descripción** | Esta Activity la usamos para cambiar la contraseña |

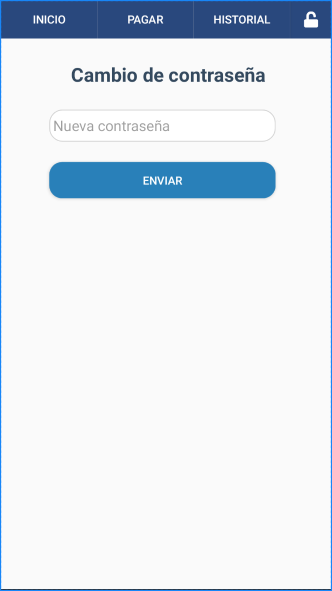
Acciones

|  |  |
| --- | --- |
| Método | Descripción |
| grabar | Envía al servidor la nueva contraseña |
| inicio | Carga la actividad MainActivity |
| pagar | Carga la actividad PagarActivity |
| historial | Carga la actividad HistorialActivity |
| cambioContrasena | Carga la actividad CambioContrasenaActivity |
|  |  |

**Webservice** : api/cambiocontrasena

**Parametros**: contrasena, usruniqueid

**Retorno**: OK o Error



# HistorialActivity

|  |  |
| --- | --- |
| **Tipo** | Activity |
| **Descripción** | Buscamos los pagos realizados por mes y año |

Acciones

|  |  |
| --- | --- |
| Método | Descripción |
| buscar | Después de crear el componente hacemos una llamada al webservices al Api “api/historial” y mostramos |
| inicio | Carga la actividad MainActivity |
| pagar | Carga la actividad PagarActivity |
| historial | Carga la actividad HistorialActivity |
| cambioContrasena | Carga la actividad CambioContrasenaActivity |
|  |  |

Webservice: api/historial

Parametros: usruniqueid, anyo, mes

Retorno: lista de pagos

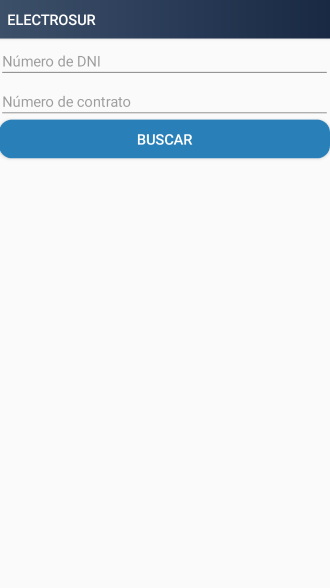


# InvitadoActivity

|  |  |
| --- | --- |
| **Tipo** | Activity |
| **Descripción** | Interfaz de pago como usuario invitado(anónimo) si registro, nos permite buscar un contrato con solo número de DNI |

Acciones

|  |  |
| --- | --- |
| Método | Descripción |
| buscar | Busca un numero de suministro |
| ListView ClickListener | Cuando hacemos clic en un ítem del listView |
| onActivityResult | Es una función que se instancia cunado se termina de pagar por el formulario de visa. |



# LoginActivity

|  |  |
| --- | --- |
| **Tipo** | Activity |
| **Descripción** | Pantalla de inicio que nos permite ingresar al sistema también hay enlaces |

Acciones

|  |  |
| --- | --- |
| Método | Descripción |
| refrescarCaptcha |  |
| login |  |
| registro |  |
| recuperarContrasena |  |
| invitado |  |



# MainActivity

|  |  |
| --- | --- |
| **Tipo** | Activity |
| **Descripción** | Actividad de inicio |

Acciones

|  |  |
| --- | --- |
| Método | Descripción |
| inicio | Carga la actividad MainActivity |
| pagar | Carga la actividad PagarActivity |
| historial | Carga la actividad HistorialActivity |



# PagarActivity

|  |  |
| --- | --- |
| **Tipo** | Activity |
| **Descripción** | Actividad para buscar un numero de contrato y pagar |

Acciones

|  |  |
| --- | --- |
| Método | Descripción |
| buscar | Buscamos un contrato |
| itemClickListener | Se ejecuta cuando hacemos clic en un ítem de la lista, carga un dialogo con el formulario de Visa. |
| onActivityResult | Una vez pagado, se ejecuta esta acción. |
| inicio | Carga la actividad MainActivity |
| pagar | Carga la actividad PagarActivity |
| historial | Carga la actividad HistorialActivity |

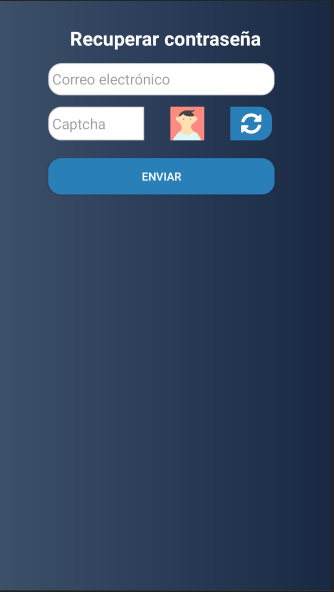


# RecuperarContrasenaActivity

|  |  |
| --- | --- |
| **Tipo** | Activity |
| **Descripción** | Recuperar contraseña |

Acciones

|  |  |
| --- | --- |
| Método | Descripción |
| refrescarCaptcha | Actualizan el código captcha. |
| grabar | Envía un correo con una nueva contraseña. |

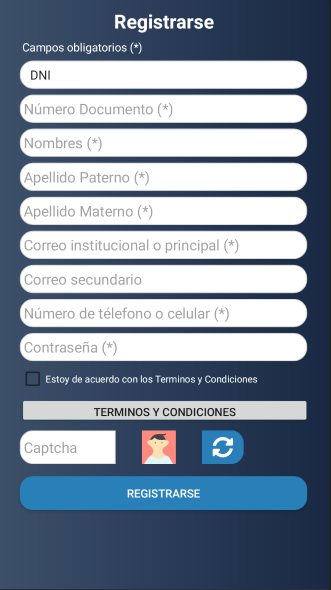


# RegistroActivity

|  |  |
| --- | --- |
| **Tipo** | Activity |
| **Descripción** | Registra a un usuario |

Acciones

|  |  |
| --- | --- |
| Método | Descripción |
| refrescarCaptcha | Actualizan el código captcha |
| dialogo | Visualiza los términos y condiciones. |
| grabar | Graba al usuario y envía un correo de confirmación de cuenta. |

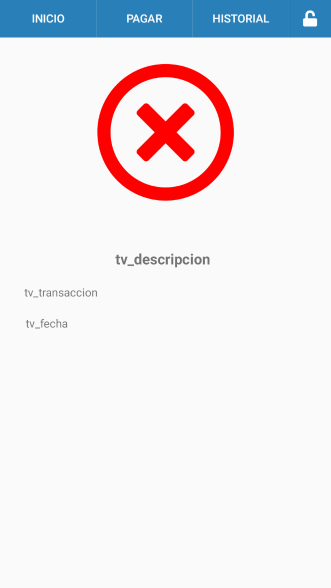


# ResultadomalActivity

|  |  |
| --- | --- |
| **Tipo** | Activity |
| **Descripción** | Se muestra el resultado erróneo de un pago. |

Acciones

|  |  |
| --- | --- |
| Método | Descripción |
| onCreate | Carga los valores de error |
| inicio | Carga la actividad MainActivity |
| pagar | Carga la actividad PagarActivity |
| historial | Carga la actividad HistorialActivity |
| cambioContrasena | Carga la actividad CambioContrasenaActivity |

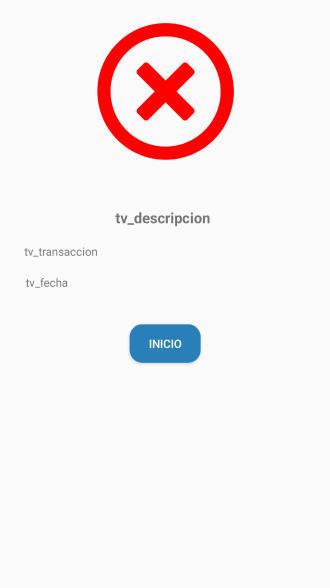


# ResultadoMalInvitadoActivity

|  |  |
| --- | --- |
| **Tipo** | Avtividad |
| **Descripción** | Se muestra el resultado erróneo de un pago. |

Acciones

|  |  |
| --- | --- |
| Método | Descripción |
| onCreate | Carga los valores de error |
| inicio | Carga la actividad InvitadoActivity |

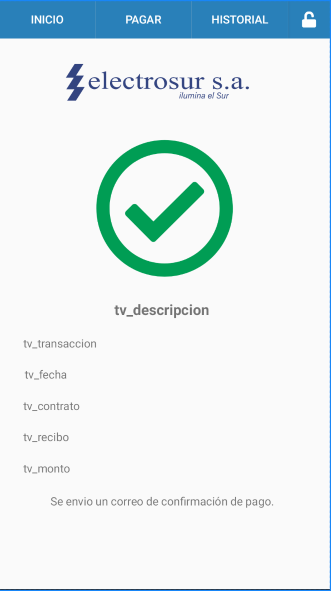


# ResultadookActivity

|  |  |
| --- | --- |
| **Tipo** | Actividad |
| **Descripción** | Muestra el resultado satisfactorio de pago |

Acciones

|  |  |
| --- | --- |
| Método | Descripción |
| onCreate | Carga los valores de pago y actualiza la conformidad de pago en el web services. |
| inicio | Carga la actividad MainActivity |
| pagar | Carga la actividad PagarActivity |
| historial | Carga la actividad HistorialActivity |
| cambioContrasena | Carga la actividad CambioContrasenaActivity |

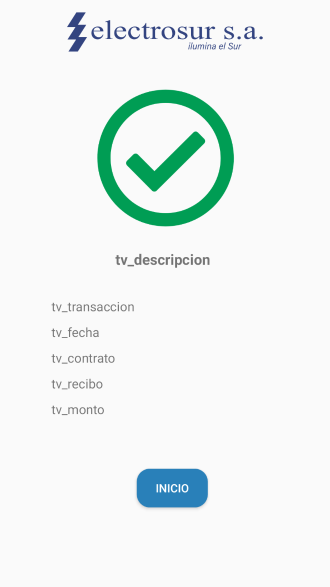


# ResultadoOKInvitadoActivity

|  |  |
| --- | --- |
| **Tipo** | Actividad |
| **Descripción** | Muestra el resultado satisfactorio de pago |

Acciones

|  |  |
| --- | --- |
| Método | Descripción |
| onCreate | Carga los valores de pago. |
| inicio | Carga la actividad InvitadoActivity |

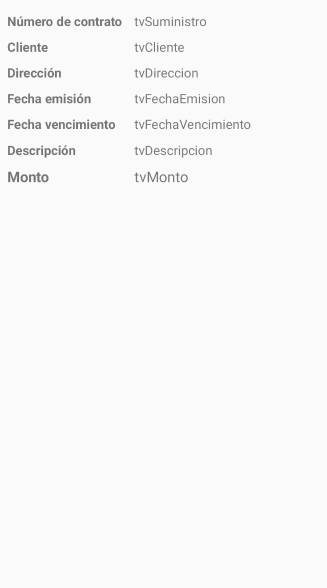


# ListItemPagos

|  |  |
| --- | --- |
| **Tipo** | BaseAdapter |
| **Descripción** | Clase que usamos para mostrar item por item |

Acciones

|  |  |
| --- | --- |
| Método | Descripción |
| getCount | Cantidad de ítem |
| getItem | Obtenemos un ítem por su posición. |
| getItemId | Obtenemos un ítem por su Id. |
| getView | Obtenemos la vista. |

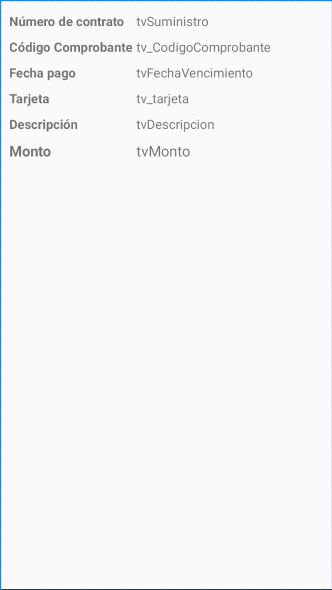


# ListItemHistorial

|  |  |
| --- | --- |
| **Tipo** | BaseAdapter |
| **Descripción** | Clase que usamos para mostrar item por item |

Acciones

|  |  |
| --- | --- |
| Método | Descripción |
| getCount | Cantidad de ítem |
| getItem | Obtenemos un ítem por su posición. |
| getItemId | Obtenemos un ítem por su Id. |
| getView | Obtenemos la vista. |



# TerminoCondicionesDialog

|  |  |
| --- | --- |
| **Tipo** | DialogFragment |
| **Descripción** | La usamos para mostrar los términos y condiciones |

Acciones

|  |  |
| --- | --- |
| Método | Descripción |
| onCreateDialog |  |
| createSimpleDialog |  |

# TextCaptcha

|  |  |
| --- | --- |
| **Tipo** | Hereda de Captcha |
| **Descripción** | Usa amos para generar la imagen de captcha de números y letras. |

Acciones

|  |  |
| --- | --- |
| Método | Descripción |
| TextCaptcha | Constructor definimos los parámetros de visualización. |
| imagen | Nos retorna la imagen de captcha. |
| generarCodigoCaptcha | Generamos el código de 4 letras o números. |

# Singleton

|  |  |
| --- | --- |
| **Tipo** | Genérico |
| **Descripción** | Singleton es un patrón de diseño que permite restringir la creación de objetos a uno solo, la utilizamos para mantener información en memoria. |

# Captcha

|  |  |
| --- | --- |
| **Tipo** | Clase abstracta |
| **Descripción** | Sirve para las implementaciones y funciones para la generación del captcha. |

# Recibo

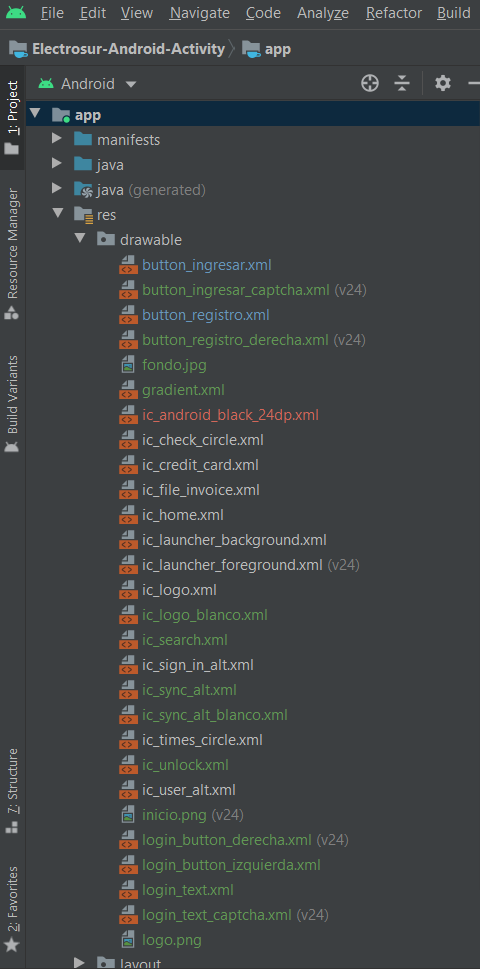
|  |  |
| --- | --- |
| **Tipo** | Genérico |
| **Descripción** | Clase contenedora |

# AndroidManifest

Archivo de configuración aquí registramos todas las actividades e indicamos cual de ellas se va iniciar.

# Recursos drawable

Lista de iconos, estilos e imágenes.



# Recursos layout

Por cada actividad tenemos un layout.

