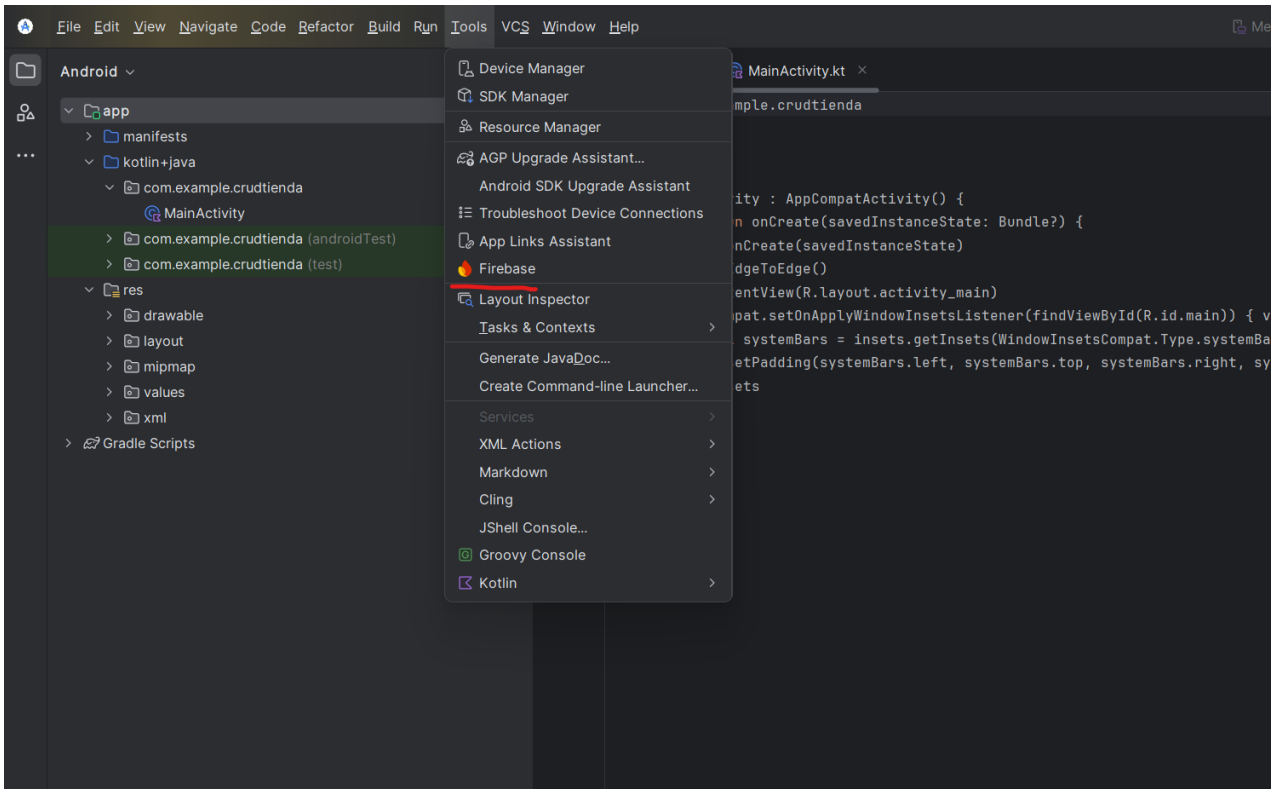
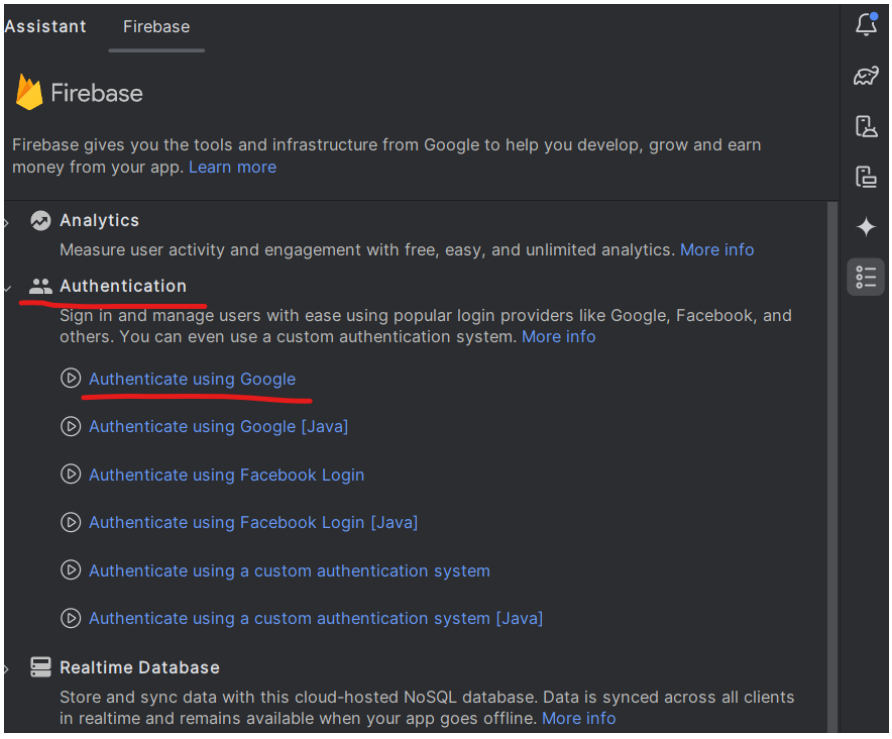
GUIA DEL PROYECTO

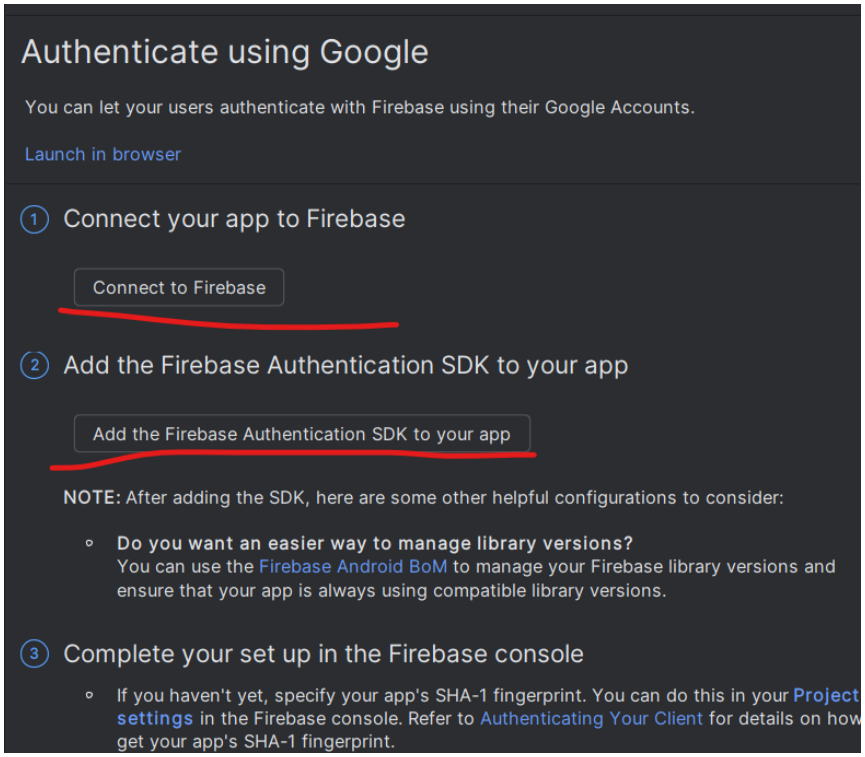
Despues de crear el proyecto haremos las configuraciones para enlazarlo al proyecto de firebase:



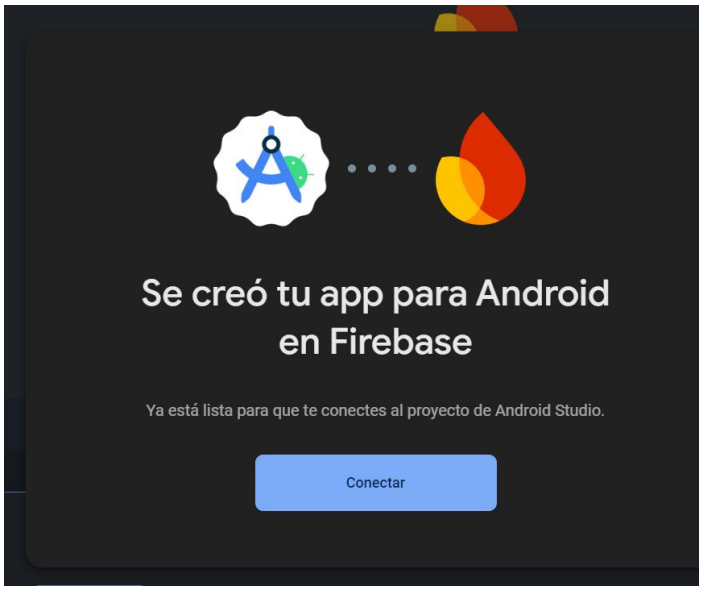
Seleccionaremos esta opcion para habilitar el login de google:



Y pincharemos en estas dos opciones para conectar con firebase e instalar el sdk:



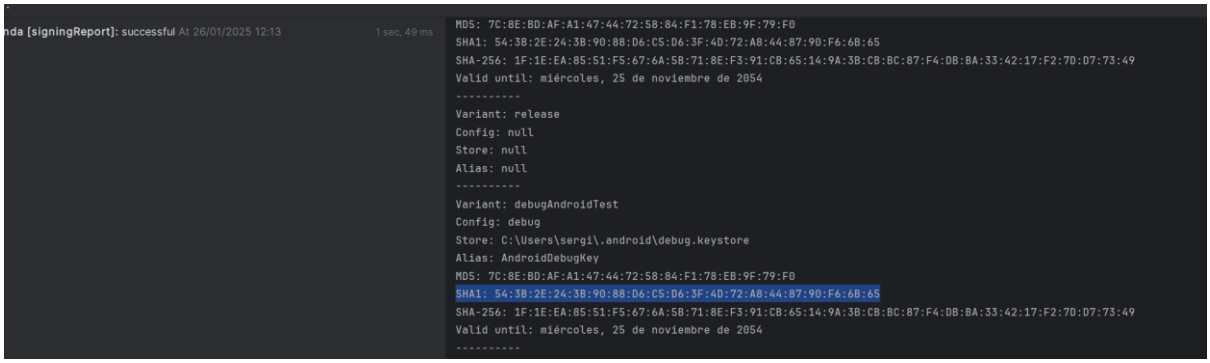
Seleccionamos el proyecto y daremos en conectar:



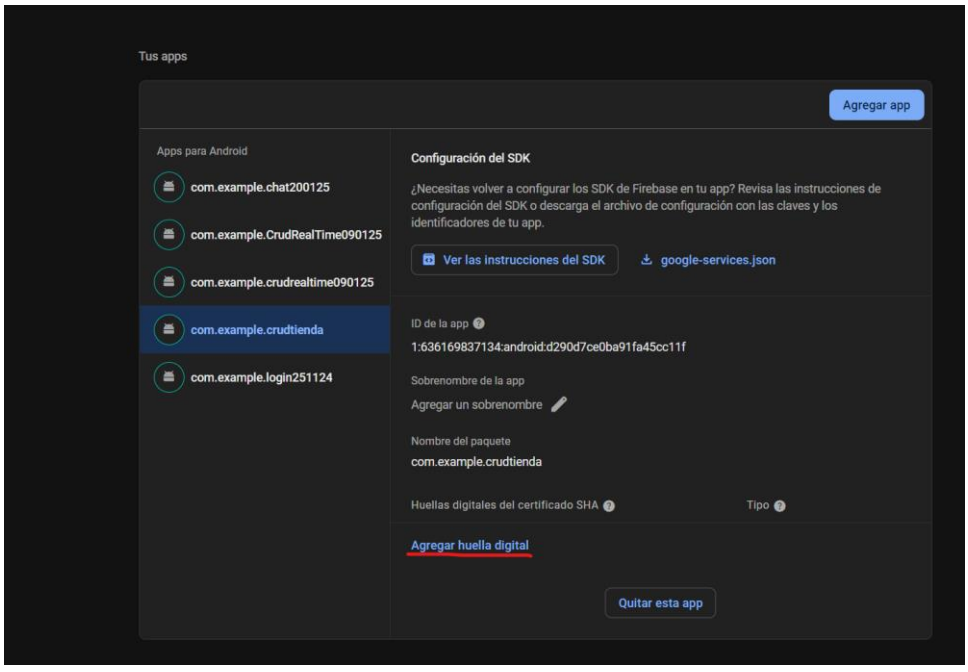
Nos aparecerá esta pantalla si se ha conectado correctamente:

Luego de estos dos pasos introduciremos este comando para crear la clave SHA1 y añadirla en nuestro firebase desde el apartado de Graddle seleccinamos el terminal:

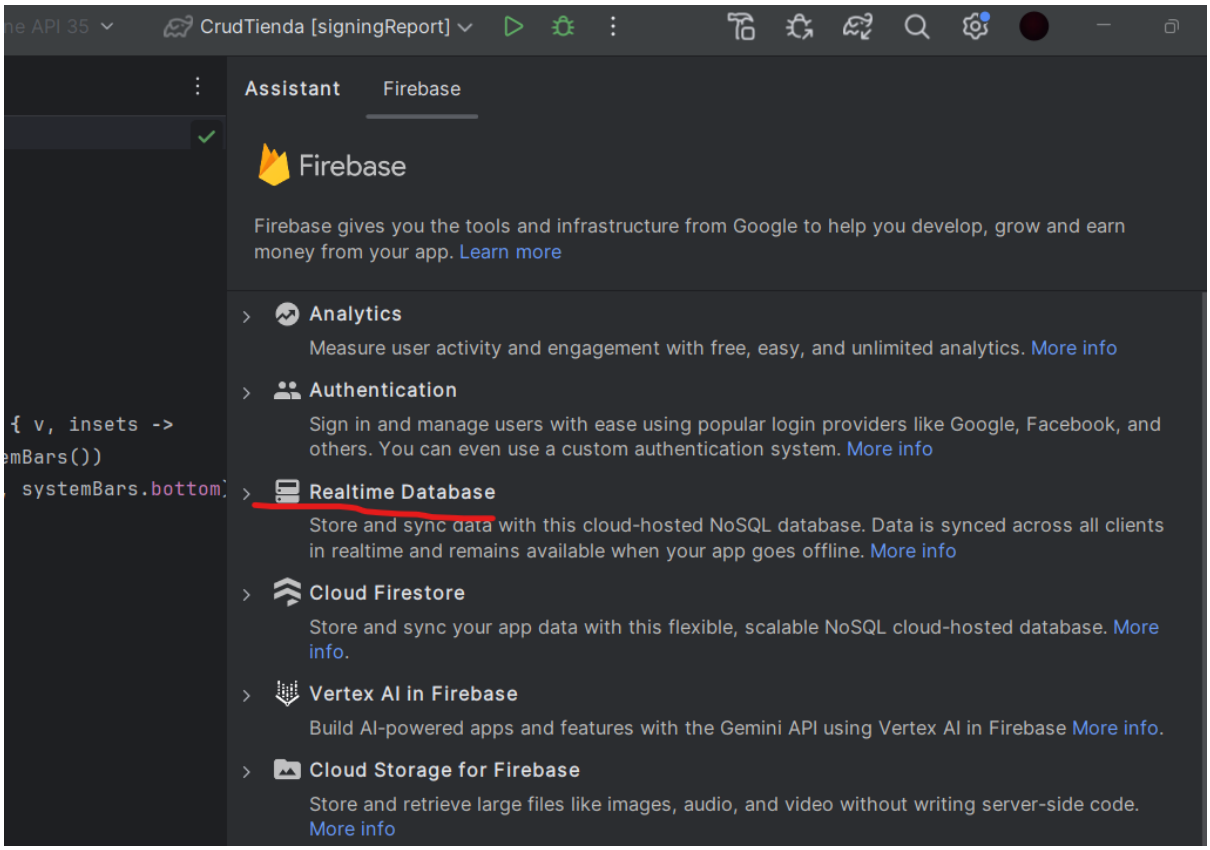
Comando: **signingReport**



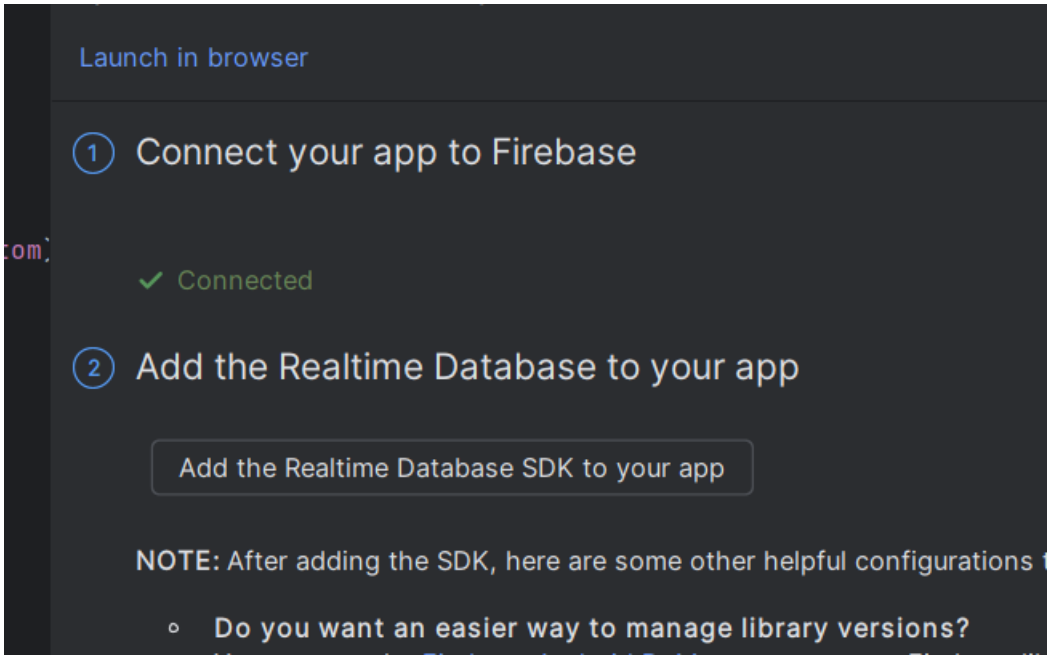
Y aqqui en el apartado de apps de nuetro proyecto firebase la añadimos:



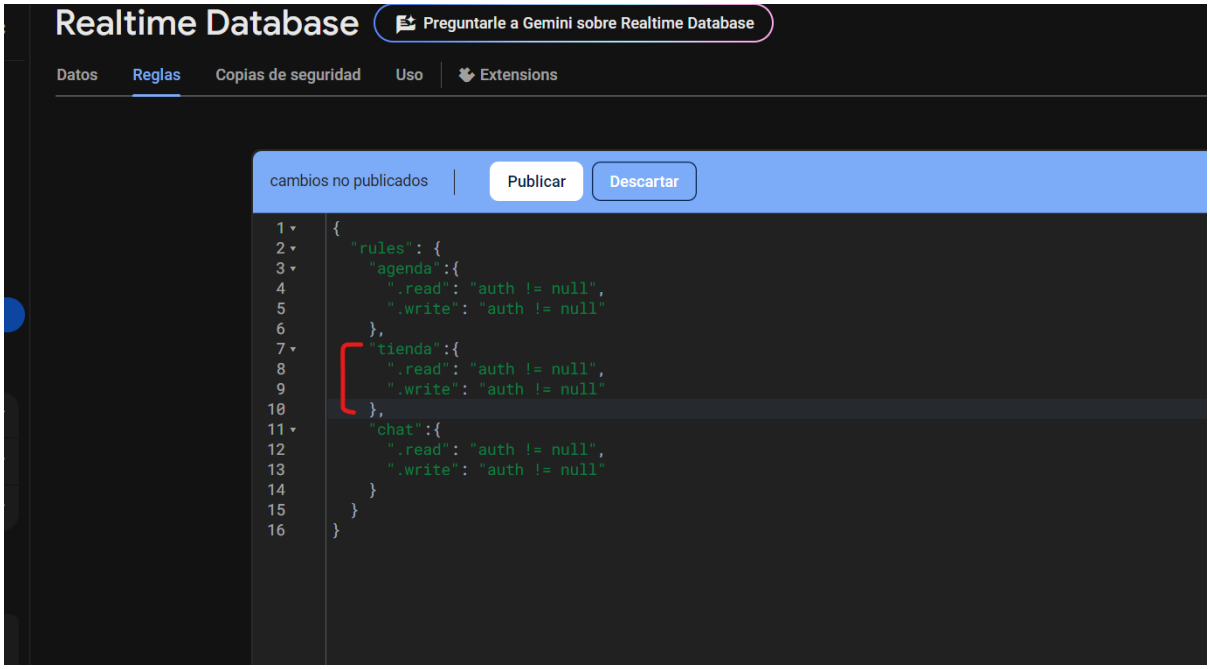
Despues añadiremos la funcionalidad de RealtimeDatabase:



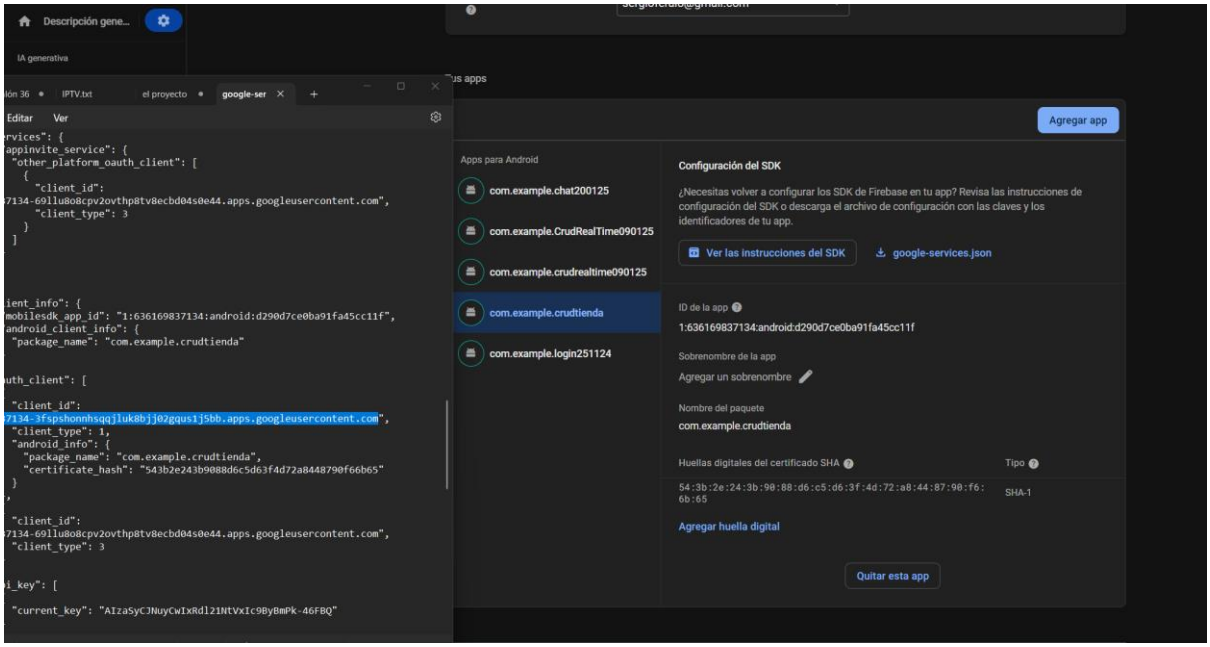
Y añadirmos el sdk:

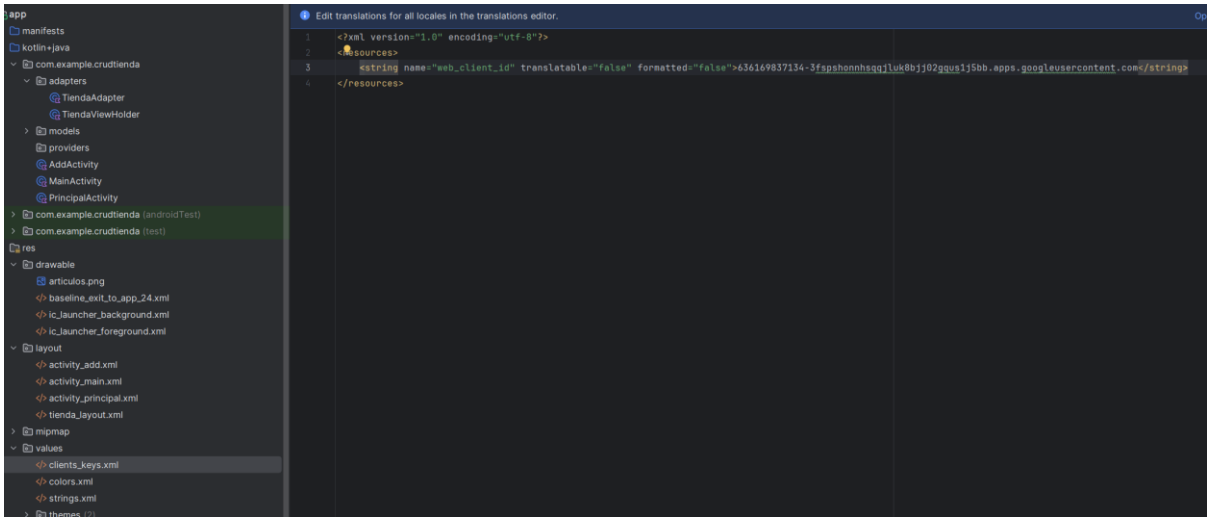


Y en el apartado de RealtimeDatabase de Firebase añadiremos el la regla sobre la que se daran los permisos de lectura y escritura de nuestra base de datos:



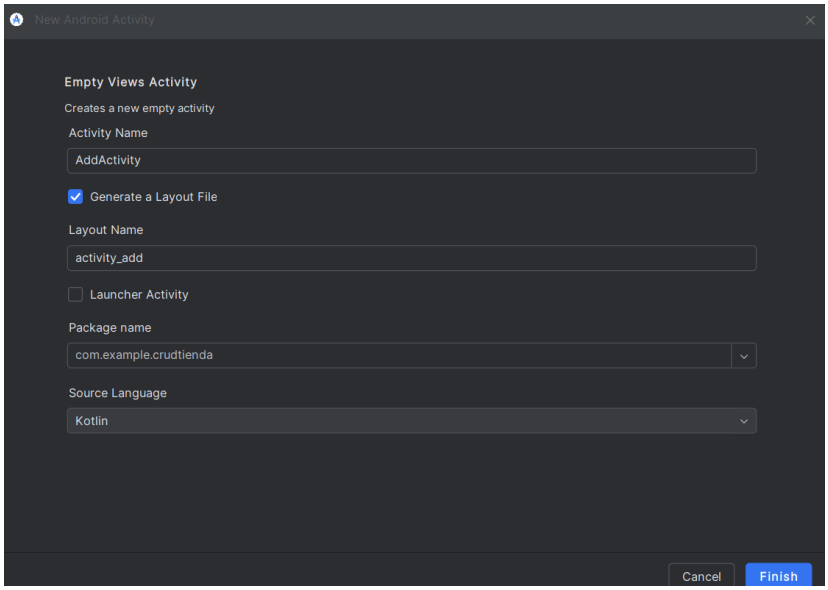
Por ultimo crearemos un archivo xml en el values que denominaremos clients_keys donde guardaremos la key correspondiente a nuetra aplicacion que encontraremos en el archivo json:



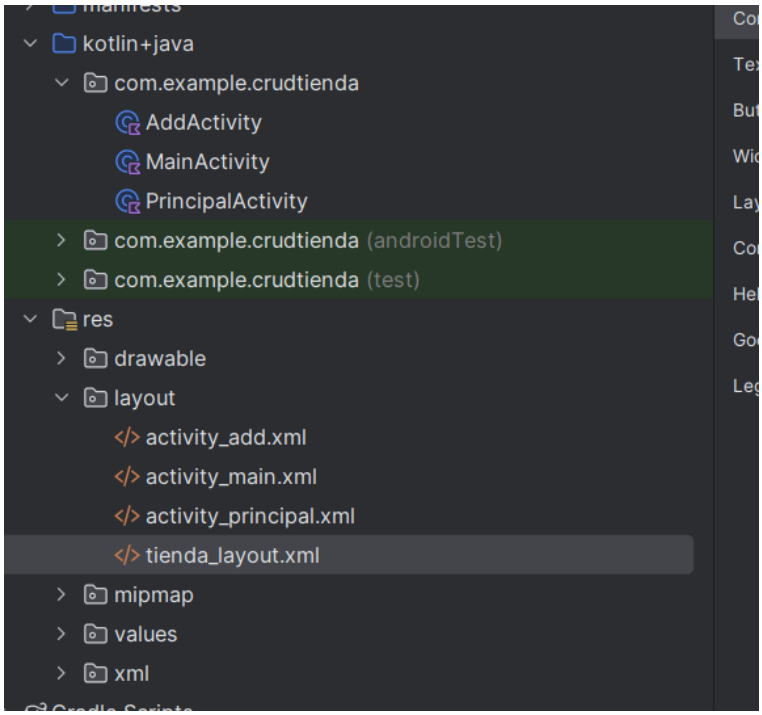


Despues de todo ya nos iremos a nuetro gradle build y añadiremos la implementacion de la autenticacion de google y el binding:

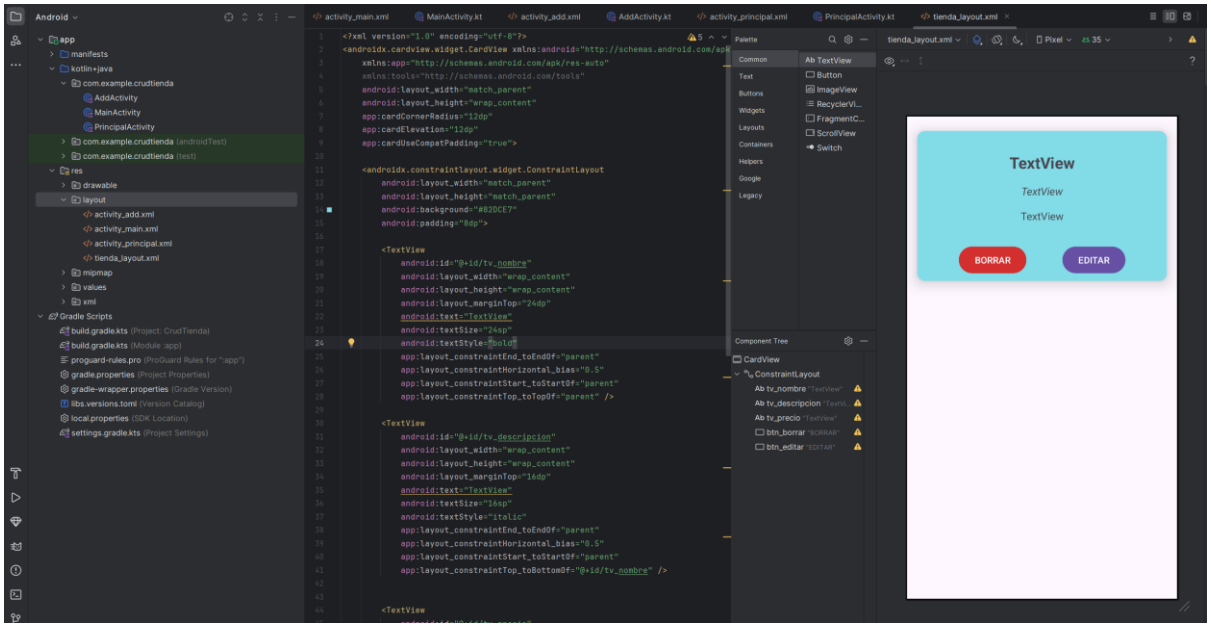
Crearemos los activities y sus respectivos layouts:



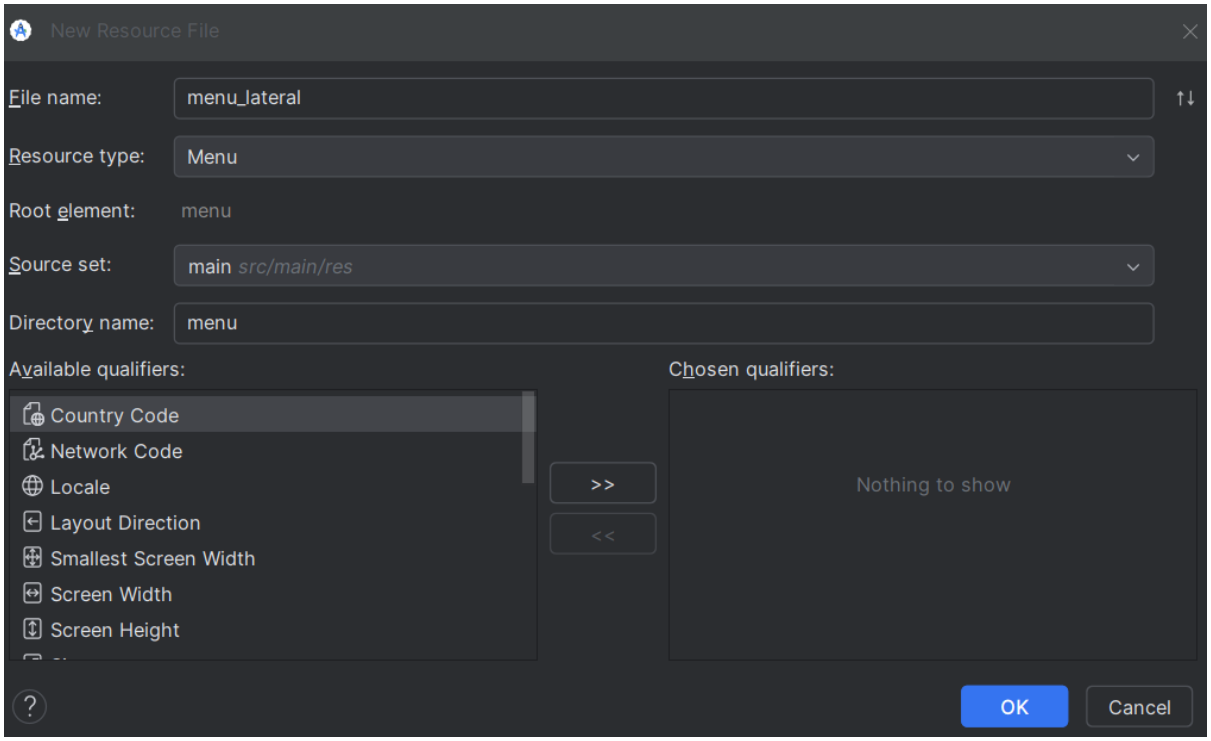
Y añadiremos el layout de tienda para el cardview de cada articulo:



El cual nos quedaria tal que asi:



Crearemos tambien un layout de menu:

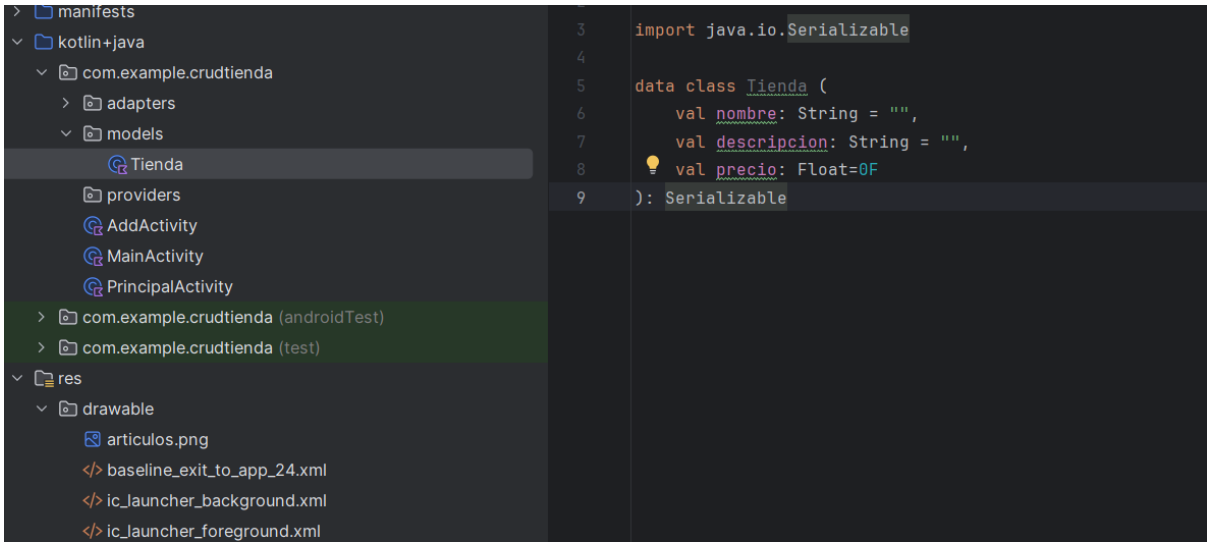


```xml
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/item_borrar"
        android:title="Borrar Todo" />
    <item
        android:id="@+id/item_logout"
        android:title="Cerrar Sesion" />
    <item
        android:id="@+id/item_salir"
        android:icon="@drawable/baseline_exit_to_app_24"
        android:title="Salir" />
</menu>
```

El cual lo incluiremos en el Layout principal:

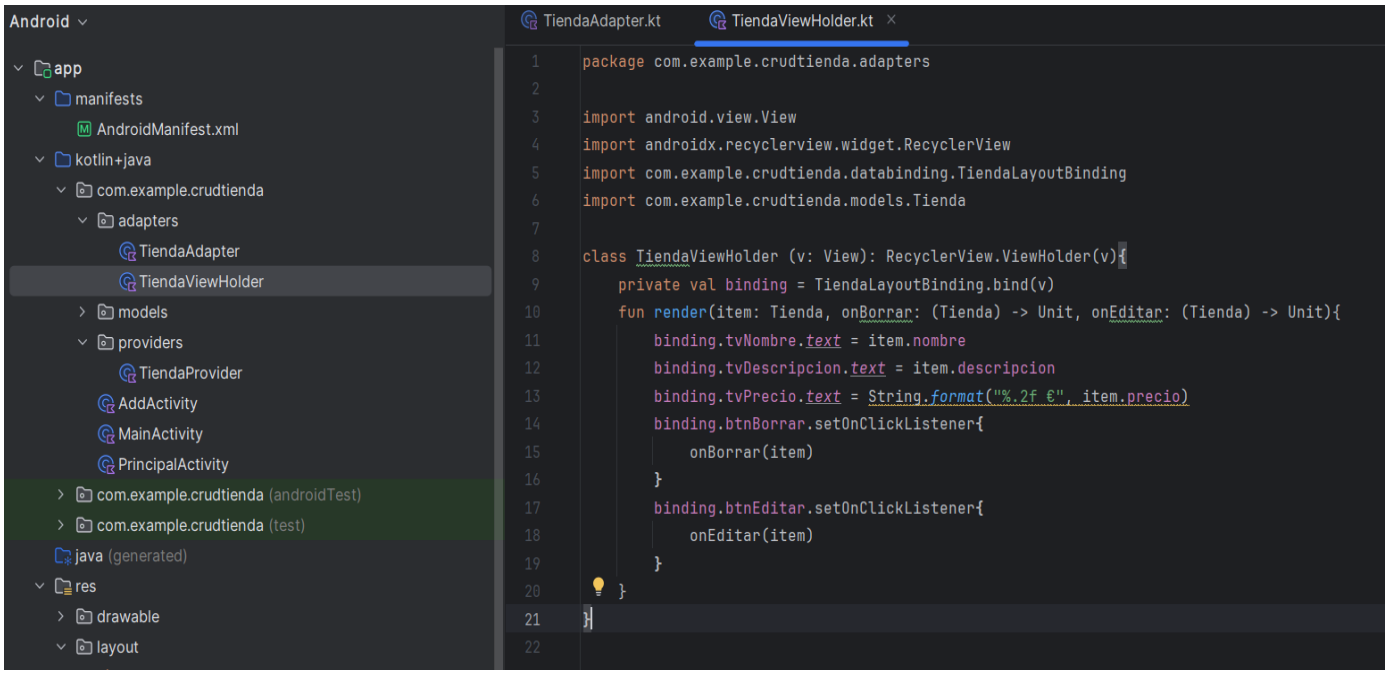
```xml
            app:layout_constraintTop_toTopOf="parent"
            app:srcCompat="@drawable/articulos" />
    </androidx.constraintlayout.widget.ConstraintLayout>
    <!--Para pintar el menu -->
    <com.google.android.material.navigation.NavigationView
        android:id="@+id/navigationview"
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_gravity="start"
        app:menu="@menu/menu_lateral"
        />
</androidx.drawerlayout.widget.DrawerLayout>
```

Ahora crearemos el modelo tienda, el cual tendrá los atributos de cada articulo:

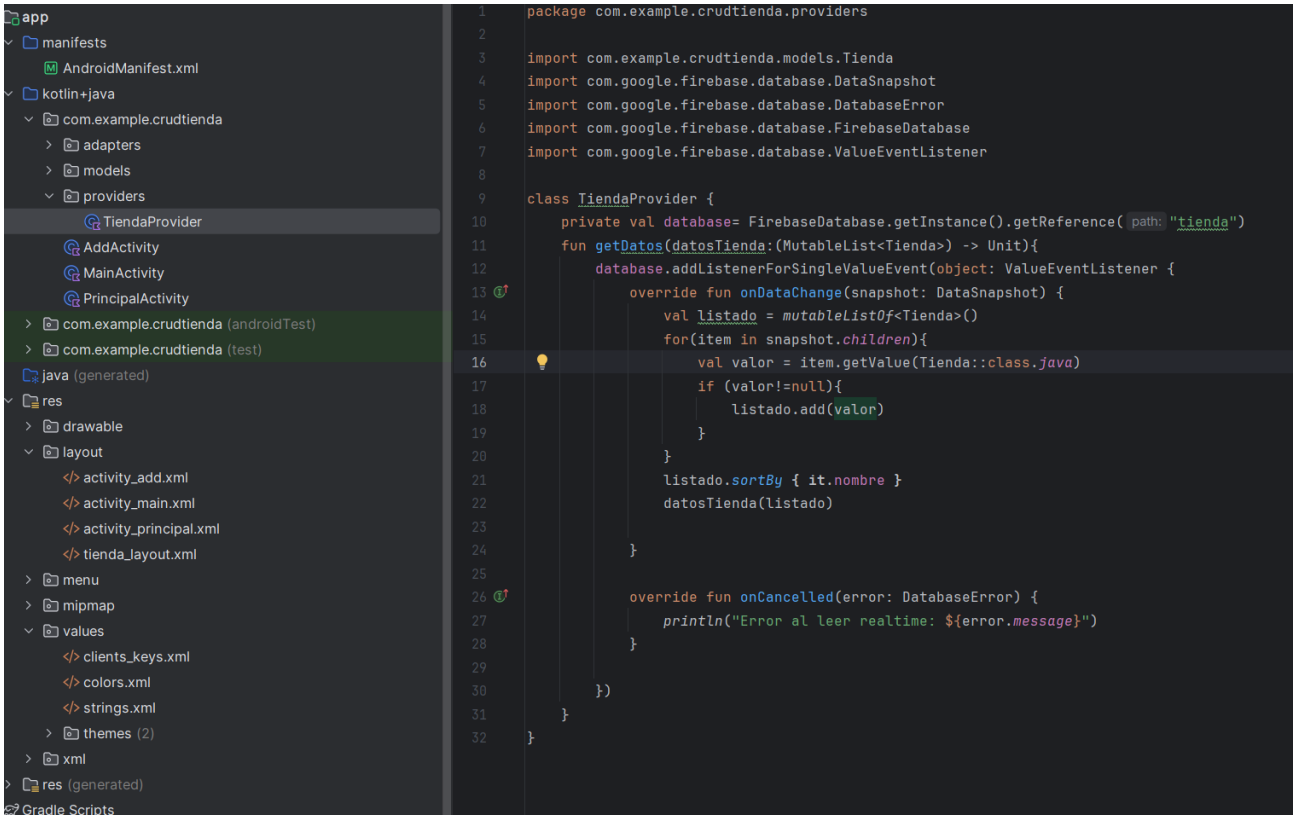


Creamos ahora el adapter y view holder para la visualizacion de cada articulo

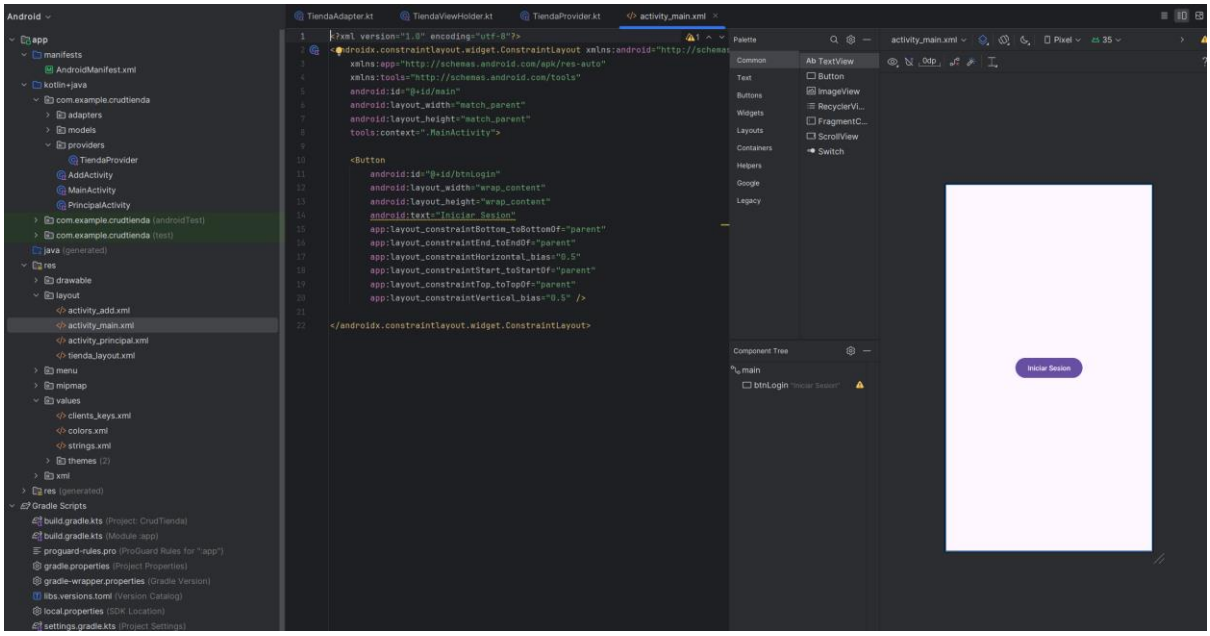
```kotlin
import android.view.LayoutInflater
import android.view.ViewGroup
import androidx.recyclerview.widget.RecyclerView
import com.example.crudtienda.R
import com.example.crudtienda.models.Tienda

class TiendaAdapter (
    var lista: MutableList<Tienda>,
    private val onBorrar: (Tienda) -> Unit,
    private val onEditar: (Tienda) -> Unit
    ): RecyclerView.Adapter<TiendaViewHolder>() {
        override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): TiendaViewHolder{
            val view = LayoutInflater.from(parent.context).inflate(R.layout.tienda_layout, parent, attachToRoot: false)
            return TiendaViewHolder(view)
        }

        override fun getItemCount()= lista.size

        override fun onBindViewHolder(holder: TiendaViewHolder, position: Int) {
            holder.render(lista[position], onBorrar, onEditar)
        }
}
```



Ahora crearemos el Provider para indicar la estructura en la que se guardaran los datos en nuestro RealTime Firebase:



Ahora desarrollaremos nuestro login, primero el layout que esta vez sera muy sencilla con un simple boton para iniciar sesion:



Y el activity:

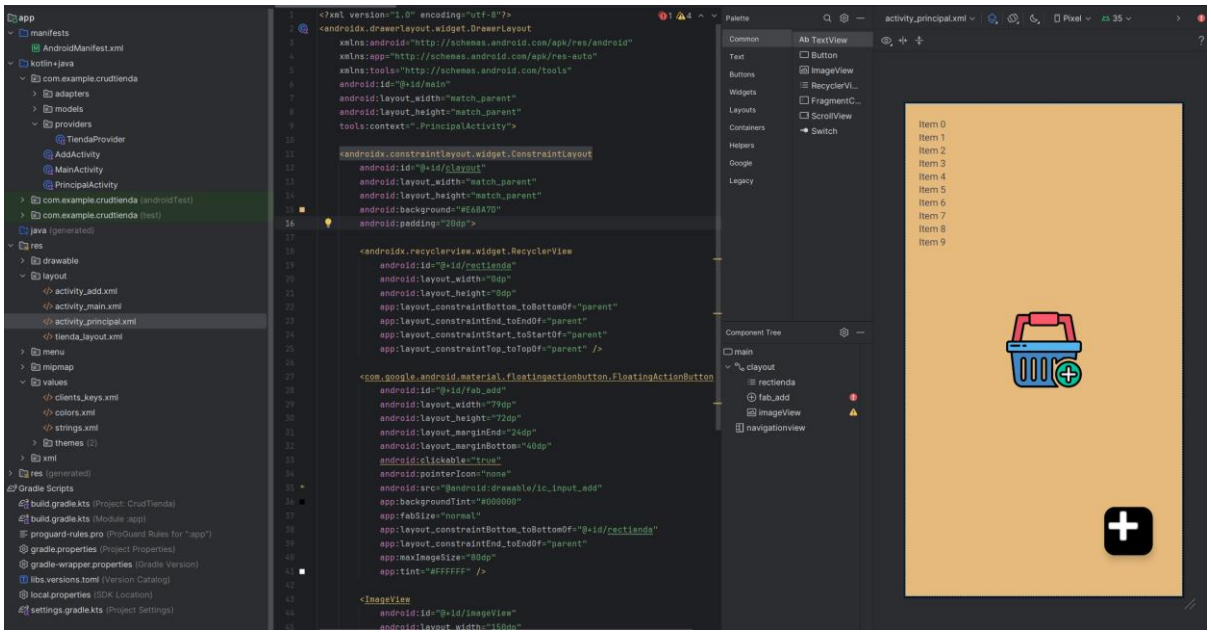
```
class MainActivity : AppCompatActivity() {
    private  val
responseLauncher=registerForActivityResult(ActivityResultContracts.S
tartActivityForResult()){
```

```
        if(it.resultCode== RESULT_OK){
            val datos=
GoogleSignIn.getSignedInAccountFromIntent(it.data)
            try{
                val cuenta=datos.getResult(ApiException::class.java)
                if(cuenta!=null){
                    val credenciales=
GoogleAuthProvider.getCredential(cuenta.idToken, null)

FirebaseAuth.getInstance().signInWithCredential(credenciales)
                        .addOnCompleteListener{
                            if(it.isSuccessful){
                                irActivityPrincipal()
                            }
                        }
                        .addOnFailureListener {
                            Toast.makeText(this,
it.message.toString(), Toast.LENGTH_SHORT).show()
                        }
                }
            }catch(e: ApiException){
                //  Log.d("ERROR de API:>>>>", e.message.toString())
            }
        }
        if(it.resultCode== RESULT_CANCELED){
            Toast.makeText(this, "El usuario canceló el registro",
Toast.LENGTH_SHORT).show()
        }
    }

    lateinit var binding: ActivityMainBinding
    private lateinit var auth: FirebaseAuth

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        binding=ActivityMainBinding.inflate(layoutInflater)
        setContentView(binding.root)

ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) {
v, insets ->
            val systemBars =
```

```
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top,
systemBars.right, systemBars.bottom)
            insets
        }
        auth=Firebase.auth
        setListeners()
    }
    //-----------------------------------------------------------------
--------------------------------
    private fun setListeners() {
        binding.btnLogin.setOnClickListener {
            login()
        }
    }
    //-----------------------------------------------------------------
--------------------------------
    private fun login() {
        val googleConf=
GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
            .requestIdToken(getString(R.string.web_client_id))
            .requestEmail()
            .build()
        val googleClient=GoogleSignIn.getClient(this, googleConf)

        googleClient.signOut() //Fundamental para que no haga login
automatico si he cerrado session

        responseLauncher.launch(googleClient.signInIntent)
    }

    //-----------------------------------------------------------------
--------------------------------
    private fun irActivityPrincipal() {
        startActivity(Intent(this, PrincipalActivity::class.java))
    }
    //-----------------------------------------------------------------
--------------------------------
    override fun onStart() {
        //Si ya tengo sesión iniciada nos saltamos el login
        super.onStart()
        val usuario=auth.currentUser
```

```
        if(usuario!=null) irActivityPrincipal()
    }
}
```

Despues de haber iniciado sesion con nuestra cuenta de google nos dirigira a la pagina principcal que sera la siguiente:



En la que encontraremos metodos para las funciones del menu, recuperar los registros de la base de datos, y funciones de los botones de cada registro:

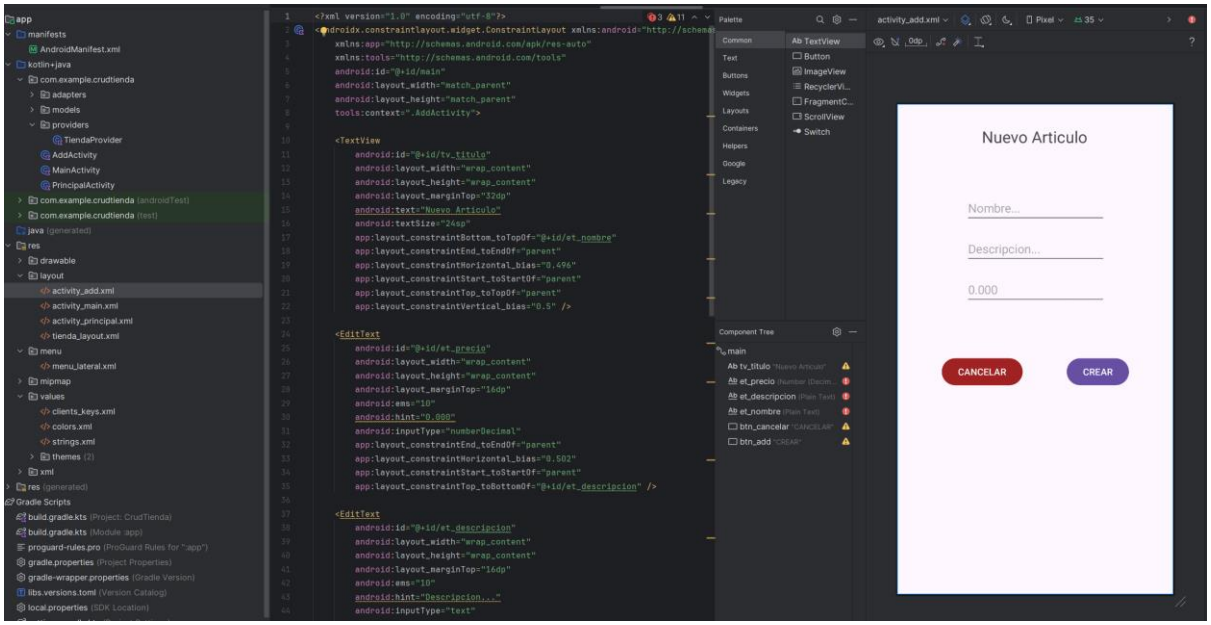
```
class PrincipalActivity : AppCompatActivity() {
    private lateinit var binding: ActivityPrincipalBinding
    var adapter = TiendaAdapter(mutableListOf<Tienda>(),
        { item -> borrarItem(item) },
        { item -> editarItem(item) })
    private lateinit var auth: FirebaseAuth
    private lateinit var database: DatabaseReference

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        binding = ActivityPrincipalBinding.inflate(layoutInflater)

        setContentView(binding.root)

ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) {
v, insets ->
            val systemBars =
```

```
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top,
systemBars.right, systemBars.bottom)
            insets
        }
        auth = Firebase.auth
        database =
FirebaseDatabase.getInstance().getReference("tienda")
        setRecycler()
        setListeners()
        setMenuLateral()
    }

    private fun setMenuLateral() {
        binding.navigationview.setNavigationItemSelectedListener {
            when (it.itemId) {
                R.id.item_logout -> {
                    auth.signOut()
                    finish()
                    true
                }

                R.id.item_salir -> {
                    finishAffinity()
                    true
                }

                R.id.item_borrar -> {
                    borrarTodo()
                    true
                }

                else -> false

            }
        }
    }

    private fun borrarTodo() {
        database.removeValue().addOnCompleteListener{
            if(it.isSuccessful){
                Toast.makeText(this, "Todo borrado",
```

```
Toast.LENGTH_SHORT).show()
                recuperarDatosTienda()
            }else{
                Toast.makeText(this, "Error al borrar",
Toast.LENGTH_SHORT).show()
            }
        }

    }

    private fun setRecycler() {
        val layoutManayer = LinearLayoutManager(this)
        binding.rectienda.layoutManager = layoutManayer

        binding.rectienda.adapter = adapter
        recuperarDatosTienda()
    }

    private fun recuperarDatosTienda() {
        val tiendaProvider = TiendaProvider()
        tiendaProvider.getDatos { todosLosRegistros ->
            binding.imageView.visibility =
                if (todosLosRegistros.isEmpty()) View.VISIBLE else
View.INVISIBLE
            adapter.lista = todosLosRegistros
            adapter.notifyDataSetChanged()
        }
    }

    private fun setListeners() {
        binding.fabAdd.setOnClickListener {
            irActivityAdd()
        }
    }

    private fun irActivityAdd(bundle: Bundle? = null) {
        val i = Intent(this, AddActivity::class.java)
        if (bundle != null) {
            i.putExtras(bundle)
        }
        startActivity(i)
    }
```

```
    private fun borrarItem(articulo: Tienda) {
        val nombre = articulo.nombre // Usar el nombre como
identificador único

        database.child(nombre).removeValue()
            .addOnSuccessListener {
                val position = adapter.lista.indexOf(articulo)
                if (position != -1) {
                    adapter.lista.removeAt(position)
                    adapter.notifyItemRemoved(position)
                    Toast.makeText(this, "Artículo borrado",
Toast.LENGTH_SHORT).show()
                }
            }
            .addOnFailureListener {
                Toast.makeText(this, "Error al borrar el artículo",
Toast.LENGTH_SHORT).show()
            }
    }

    private fun editarItem(item: Tienda) {
        val b = Bundle().apply {
            putSerializable("TIENDA", item)
        }
        irActivityAdd(b)
    }

    override fun onResume() {
        super.onResume()
        recuperarDatosTienda()
    }
}
```

Y ahora para el activity de nuestro formulario en el que crearemos los registros de articulos o editaremos el elegido:



Y crearemos su respectivo AddActivity:

```
class AddActivity : AppCompatActivity() {
    private lateinit var binding: ActivityAddBinding

    private var nombre = ""
    private var descripcion = ""
    private var precio = 0F
    private var editando=false

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()

        binding = ActivityAddBinding.inflate(layoutInflater)
        setContentView(binding.root)

ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) {
v, insets ->
            val systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top,
systemBars.right, systemBars.bottom)
            insets
        }
        recogerRegistro()
        setListeners()
```

```
        if (editando) {
            binding.tvTitulo.text = "Editar Articulo"
            binding.btnAdd.text = "EDITAR ARTICULO"
        }

    }

    private fun recogerRegistro() {
        val datos = intent.extras
        if (datos != null) {
            val tienda = datos.getSerializable("TIENDA") as Tienda
            editando = true
            nombre = tienda.nombre
            descripcion = tienda.descripcion
            precio = tienda.precio

            pintarDatos()
        }
    }

    private fun pintarDatos() {
        binding.etNombre.setText(nombre)
        binding.etDescripcion.setText(descripcion)
        binding.etPrecio.setText(precio.toString())
    }

    private fun setListeners() {
        binding.btnCancelar.setOnClickListener {
            finish()
        }
        binding.btnAdd.setOnClickListener {
            addItem()
        }
    }

    private fun addItem() {
        if (!datosOK()) return

        val database: DatabaseReference =
FirebaseDatabase.getInstance().getReference("tienda")

        val nombre = binding.etNombre.text.toString().trim()
```

```
        val descripcion =
binding.etDescripcion.text.toString().trim()
        val precio =
binding.etPrecio.text.toString().toFloatOrNull() ?: 0F

        // Crear el objeto Tienda
        val articulo = Tienda(nombre, descripcion, precio)

        // Verificar si el artículo ya existe
        database.child(nombre).addListenerForSingleValueEvent(object
: ValueEventListener {
            override fun onDataChange(snapshot: DataSnapshot) {
                if (snapshot.exists() && !editando) {
                    Toast.makeText(this@AddActivity, "El artículo ya
existe", Toast.LENGTH_SHORT).show()
                } else {
                    // Guardar el artículo en la base de datos
usando el nombre como clave

database.child(nombre).setValue(articulo).addOnSuccessListener {
                        Toast.makeText(this@AddActivity, "Artículo
agregado con éxito", Toast.LENGTH_SHORT).show()
                        finish()
                    }.addOnFailureListener {
                        Toast.makeText(this@AddActivity, "Error al
insertar el artículo", Toast.LENGTH_SHORT).show()
                    }
                }
            }

            override fun onCancelled(error: DatabaseError) {
                Toast.makeText(this@AddActivity, "Error en la
operación: ${error.message}", Toast.LENGTH_SHORT).show()
            }
        })
    }

    private fun datosOK(): Boolean {
        val nombre = binding.etNombre.text.toString().trim()
        val descripcion =
binding.etDescripcion.text.toString().trim()
        val precio =
```

```
binding.etPrecio.text.toString().toFloatOrNull() ?: 0F

        if (nombre.length < 3) {
            binding.etNombre.error = "Error, el nombre debe tener al
menos 3 caracteres"
            return false
        }

        if (descripcion.length < 10) {
            binding.etDescripcion.error = "Error, la descripción
debe tener al menos 10 caracteres"
            return false
        }

        if (precio < 1 || precio > 10000) {
            binding.etPrecio.error = "Error, el precio debe estar
entre 1 y 10000"
            return false
        }

        return true
    }

}
```

A destacar encontraremos el metodo de añadir en el que buscaremos si existe ya un articulo con el nombre ya existente y si no, añadirlo y poner el valor del nombre introducido como el id:

```kotlin
private fun addItem() {
    if (!datosOK()) return

    val database: DatabaseReference = FirebaseDatabase.getInstance().getReference( path: "tienda")

    val nombre = binding.etNombre.text.toString().trim()
    val descripcion = binding.etDescripcion.text.toString().trim()
    val precio = binding.etPrecio.text.toString().toFloatOrNull() ?: 0F

    // Crear el objeto Tienda
    val articulo = Tienda(nombre, descripcion, precio)

    // Verificar si el artículo ya existe
    database.child(nombre).addListenerForSingleValueEvent(object : ValueEventListener {
        override fun onDataChange(snapshot: DataSnapshot) {
            if (snapshot.exists() && !editando) {
                Toast.makeText( context: this@AddActivity,  text: "El artículo ya existe", Toast.LENGTH_SHORT).show()
            } else {
                // Guardar el artículo en la base de datos usando el nombre como clave
                database.child(nombre).setValue(articulo).addOnSuccessListener {
                    Toast.makeText( context: this@AddActivity,  text: "Artículo agregado con éxito", Toast.LENGTH_SHORT).show()
                    finish()
                }.addOnFailureListener {
                    Toast.makeText( context: this@AddActivity,  text: "Error al insertar el artículo", Toast.LENGTH_SHORT).show()
                }
            }
        }

        override fun onCancelled(error: DatabaseError) {
            Toast.makeText( context: this@AddActivity,  text: "Error en la operación: ${error.message}", Toast.LENGTH_SHORT).show()
        }
    })
}
```

Tal que asi: