

 <p>UNIVERSIDAD DON BOSCO FACULTAD DE INGENIERÍA ESCUELA DE COMPUTACIÓN VIRTUAL</p> <p>Desarrollo de Software para Móviles DSM941</p>	<p>Ciclo II - 2024</p>
---	-------------------------------

Foro II

Docente:

Alexander Alberto Siguenza Campos

GT: 01T

Integrantes	
Fredy Antonio Alfaro Santos	AS231934
Marco José Mejía Tario	MT172007
Jennifer Patricia Zaldana Beltran	ZB212784
Daniela Alejandra Fuentes Guardado	FG210531
Diego Antonio Fuentes Guardado	FG230381

GitHub: <https://github.com/fredy34009/Login-DSM>

Opciones de Autenticación en firebase

Firebase Authentication es una solución robusta para manejar el inicio de sesión y la autenticación de usuarios en aplicaciones web y móviles.

Correo Electrónico y Contraseña

Características:

- Permite a los usuarios registrarse y autenticarse utilizando su correo electrónico y una contraseña.
- Incluye funcionalidades integradas para gestionar contraseñas olvidadas.
- Proporciona validación básica para direcciones de correo electrónico.

Ventajas:

- Fácil de implementar.
- Familiar para los usuarios.
- Alta personalización en el flujo de autenticación.

Desventajas:

- Los usuarios deben recordar sus contraseñas.
- Mayor riesgo si no se implementan medidas de seguridad adicionales (como autenticación de dos factores).

Consideraciones:

- Asegúrate de usar HTTPS para proteger las credenciales.
- Implementa políticas de contraseñas seguras y monitorea intentos de fuerza bruta.

Inicio de Sesión Social (Google, Facebook, Apple, etc.)

Características:

- Permite a los usuarios registrarse y autenticarse mediante sus cuentas en plataformas sociales.
- Soporta múltiples proveedores como Google, Facebook, Twitter, Microsoft y Apple.

Ventajas:

- Simplifica el registro y login, ya que los usuarios no necesitan crear una nueva contraseña.
- Aumenta la tasa de conversión al reducir la fricción en el registro.

Desventajas:

- Dependencia de terceros; problemas de autenticación si el proveedor tiene interrupciones.
- Puede requerir la configuración de claves API y SDK adicionales.

Consideraciones:

- Configura correctamente las claves API en Firebase Console.
- Cumple con los términos y políticas de cada proveedor (como las políticas de Apple para apps en iOS).

Número de Teléfono

Características:

- Permite la autenticación mediante el envío de códigos SMS al número de teléfono del usuario.
- Utiliza la verificación en dos pasos para aumentar la seguridad.

Ventajas:

- Ideal para aplicaciones donde los usuarios pueden no tener correos electrónicos.
- Proporciona un nivel adicional de seguridad al usar SMS.

Desventajas:

- Dependencia de los servicios de SMS; podría no ser fiable en ciertas regiones.
- Los costos de los mensajes SMS pueden acumularse.

Consideraciones:

- Verifica la cobertura del servicio de SMS en las regiones donde operará la aplicación.
- Cumple con las regulaciones de privacidad y telecomunicaciones locales.

Anónimo

Características:

- Permite a los usuarios interactuar con la aplicación sin registrarse.
- Puede vincularse más tarde con otros métodos de autenticación.

Ventajas:

- Baja fricción inicial; ideal para probar la aplicación o para usuarios que no desean registrarse de inmediato.
- Se puede convertir en una cuenta permanente al agregar un método de autenticación.

Desventajas:

- No proporciona información del usuario (como correos electrónicos o números de teléfono).
- Puede llevar a la pérdida de datos si el usuario elimina la app antes de vincular su cuenta.

Consideraciones:

- Utiliza esta opción para simplificar el acceso inicial, pero proporciona incentivos para que los usuarios vinculen sus cuentas.

Proveedor Personalizado (Custom Auth)

Características:

- Permite implementar un sistema de autenticación propio utilizando tokens generados en el backend.

Ventajas:

- Total flexibilidad y control sobre el flujo de autenticación.
- Útil para sistemas heredados o integraciones complejas.

Desventajas:

- Mayor complejidad en la implementación.
- Requiere gestión de la infraestructura y seguridad adicional.

Consideraciones:

- Asegúrate de seguir las mejores prácticas para generar y verificar tokens seguros.
- Ideal para escenarios que requieren requisitos específicos no cubiertos por otros métodos.

Sin Contraseña (Passwordless)

Características:

- Autenticación mediante enlaces mágicos enviados por correo electrónico o códigos enviados por SMS.

Ventajas:

- Elimina la necesidad de recordar contraseñas.
- Mejora la experiencia del usuario, especialmente en dispositivos móviles.

Desventajas:

- Dependencia de servicios de correo electrónico o SMS.
- Riesgo si el correo o número de teléfono del usuario es comprometido.

Consideraciones:

- Implementa medidas de seguridad adicionales, como autenticación de dos factores.

Consideraciones Generales para Todas las Opciones:

1. Seguridad:

- Usa HTTPS para todas las comunicaciones.
- Habilita autenticación de dos factores (si es compatible).
- Mantén actualizado el SDK de Firebase Authentication.

2. Regulaciones:

- Cumple con las leyes locales de privacidad, como GDPR (Europa) o CCPA (California).
- Implementa políticas claras sobre la gestión de datos personales.

3. Experiencia del Usuario:

- Diseña flujos de autenticación intuitivos y rápidos.
- Proporciona mensajes claros en caso de errores o problemas.

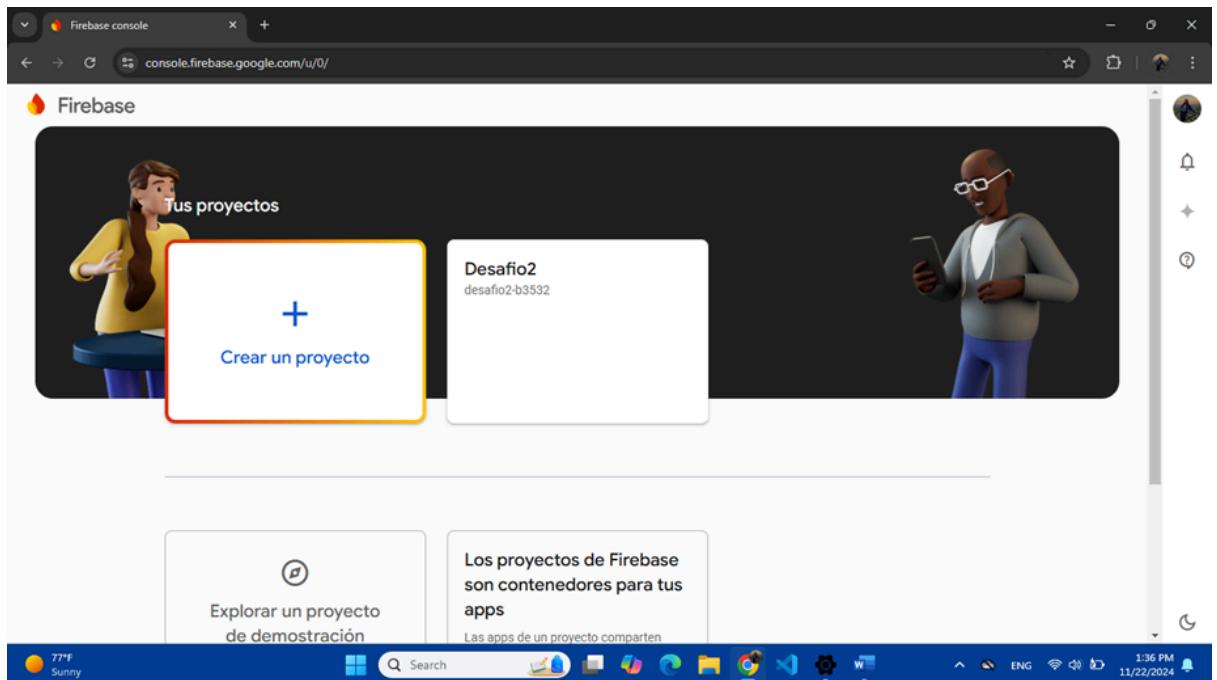
4. Escalabilidad:

- Firebase Authentication está diseñado para escalar automáticamente, pero considera las limitaciones de costos de mensajes SMS y otros servicios externos.

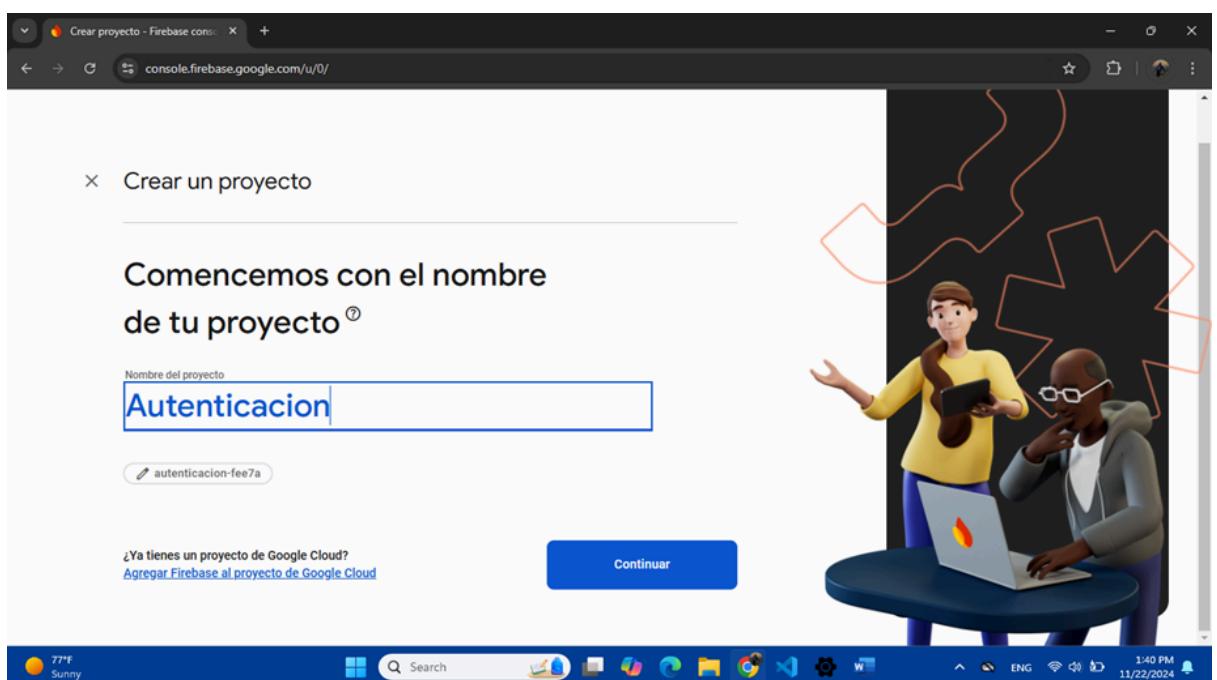
Implementación de autenticación por correo electrónico

1. Crear un proyecto en Firebase

- Nos dirigimos a la consola de Firebase <https://console.firebaseio.google.com/u/0/>



- Ingresar el nombre, y seguir los pasos de configuración y activar Analytics si se necesita



× Crear un proyecto

Google Analytics habilita las siguientes funciones:



Pruebas A/B ?



Segmentación de usuarios y
orientación a ellos en los productos de
Firebase ?



Registros de rutas de navegación en
Crashlytics ?



Activadores de Cloud Functions ?
basados en eventos



Informes ilimitados y gratuitos ?



Habilitar Google Analytics para este proyecto

Opción recomendada



[Anterior](#)

[Continuar](#)

Configurar Google Analytics

Elige o crea una cuenta de Google Analytics ?



Default Account for Firebase



Crea una nueva propiedad en esta cuenta automáticamente

Cuando se cree el proyecto, también se creará una nueva propiedad de Google Analytics en la cuenta de Google Analytics que elijas y se vinculará a tu proyecto de Firebase. Este vínculo permitirá el flujo de datos entre los productos. Los datos que se exportan de tu propiedad de Google Analytics a Firebase están sujetos a las Condiciones del Servicio de Firebase, mientras que los datos de Firebase que se importan a Google Analytics están sujetos a las Condiciones del Servicio de Google Analytics.

[Obtén más información](#)

[Anterior](#)

[Crear proyecto](#)

2. Configurar la Autenticación en Firebase

- Dar click en el menú de la izquierda y seleccionar “Authentication” y dar click en “empezar”

The screenshot shows the Firebase console interface. On the left, there's a sidebar with categories like Compilación (App Check, App Hosting, Authentication, Data Connect, Extensions, Firestore Database, Functions, Hosting, Machine Learning, Realtime Database, Storage) and Ejecución (Spark). The 'Authentication' item is highlighted with a red oval. The main content area is titled 'Autenticacion' and features a large yellow and red logo. It says 'Comienza por agregar Firebase a tu app' and 'Agrega una app para comenzar'. Below this are icons for iOS+, Android, code, audio, and video. At the top right, there are links for 'Plan Spark', 'Registrarse', and a button for '¿Todo listo para comenzar? Cuéntale a Gemini sobre tu proyecto'.

This screenshot shows the 'Authentication' landing page from the Firebase documentation. The title is 'Authentication'. Below it, a large text block reads: 'Autentica y administra usuarios de una gran variedad de proveedores sin ejecutar código en el servidor.' A large red arrow points down to the 'Comenzar' button. There are two buttons at the bottom: 'Comenzar' (in orange) and 'Preguntarle a Gemini' (in blue).

- Nos dirigirá a la pestaña “Método de Inicio de sesión” o “Métodos de acceso”

The screenshot shows the Firebase console's Authentication providers page. The URL is console.firebaseio.google.com/u/0/project/autenticacion-fee7a/authentication/providers. The left sidebar shows a project structure with 'Autenticación' selected. The main navigation bar has tabs: 'Usuarios', 'Método de acceso' (which is highlighted with a red oval), 'Plantillas', 'Uso', 'Configuración', and 'Extensions'. Below the tabs, it says 'Proveedores de acceso' and 'Agrega tu primer método de acceso y comienza a utilizar Firebase Auth'. There are three columns of providers: 'Proveedores nativos' (Correo electrónico/contraseña, Teléfono, Anónimo), 'Proveedores adicionales' (Google, Facebook, Play Juegos, Game Center, Apple, GitHub, Microsoft, Twitter, Yahoo), and 'Proveedores personalizados' (OpenID Connect, SAML).

- Daremos click en el siguiente proveedor que será “Correo electrónico/contraseña”

This screenshot is from the same page as the previous one, but a large red arrow points to the 'Correo electrónico/contraseña' button in the 'Proveedores nativos' section. The rest of the interface is identical to the first screenshot.

- Nos dirigirá a unas opciones las cuales habilitaremos ambas y daremos click en guardar

The screenshot shows the 'Authentication' section of the Firebase console. It displays two configuration options with enablement checkboxes:

- Correo electrónico/contraseña**: A checkbox labeled 'Habilitar' with a red arrow pointing to it.
- Vínculo del correo electrónico (acceso sin contraseña)**: A checkbox labeled 'Habilitar' with a red arrow pointing to it.

Below these options is a note: "La autenticación sin contraseña y con vínculo de correo electrónico requiere realizar más pasos de configuración. Sigue los que correspondan a tu plataforma." with links for Apple, Android, and Web.

At the bottom right are 'Cancelar' and 'Guardar' buttons, with 'Guardar' being circled in red.

- Al guardarla nos dirá que el “correo se habilito” y de esta manera se habilita un correo electrónico con Firebase

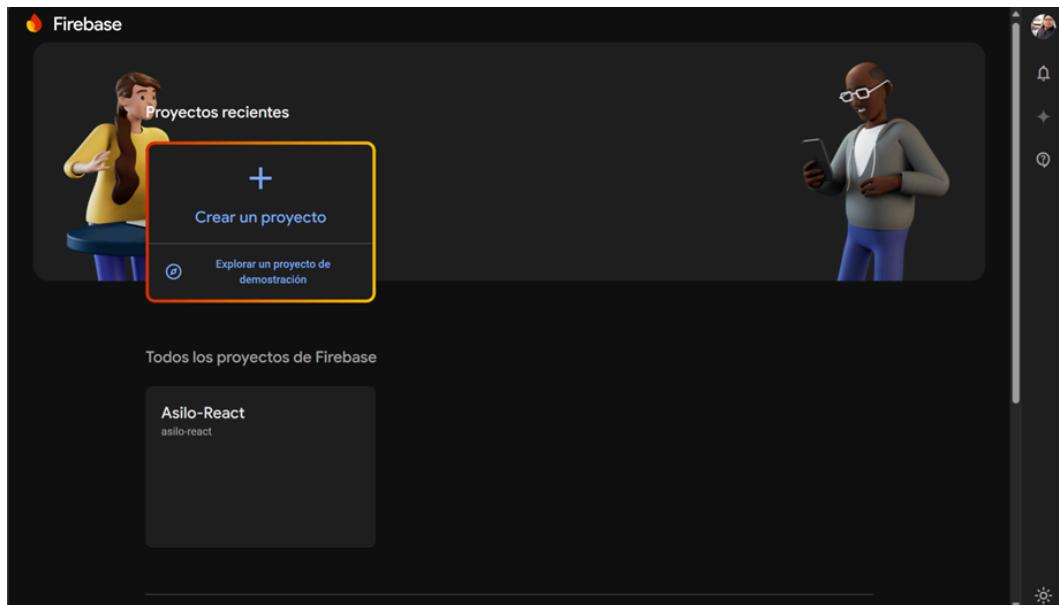
The screenshot shows the 'Authentication' section of the Firebase console after saving the configurations. A success message 'Se habilitó Correo electrónico/contraseña.' is displayed in a green box, which is circled in red.

The 'Correo electrónico/contraseña' row in the 'Proveedores de acceso' table now shows 'Habilitado' with a green checkmark.

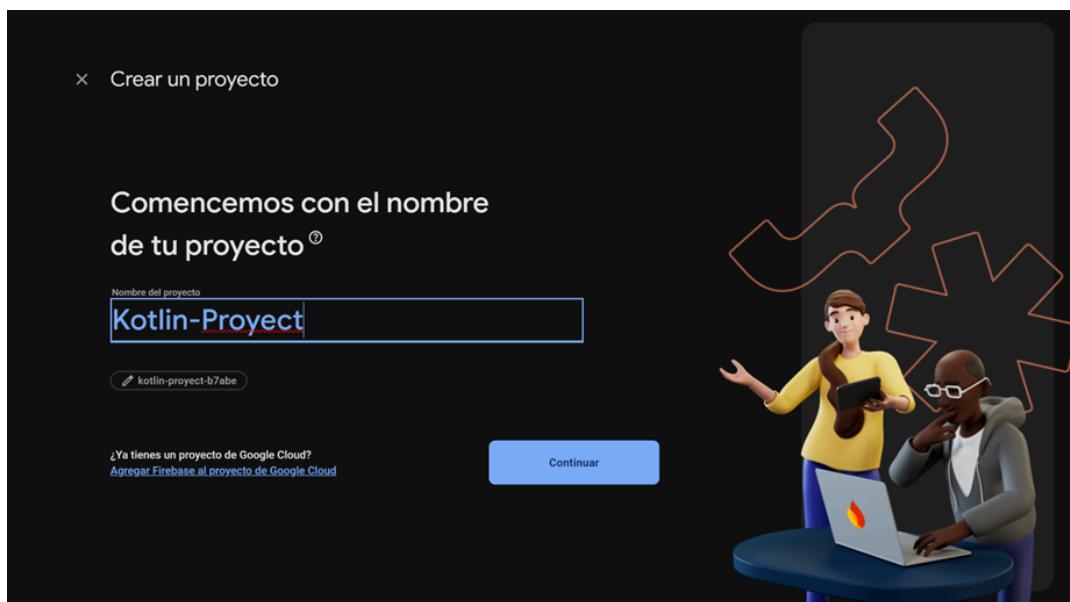
Below the table, under 'Opciones avanzadas', there is a section for 'Autenticación de varios factores mediante SMS' with a note about Identity Platform and a 'Actualizar para habilitar' button.

Autenticación por medio de los servicios de Google

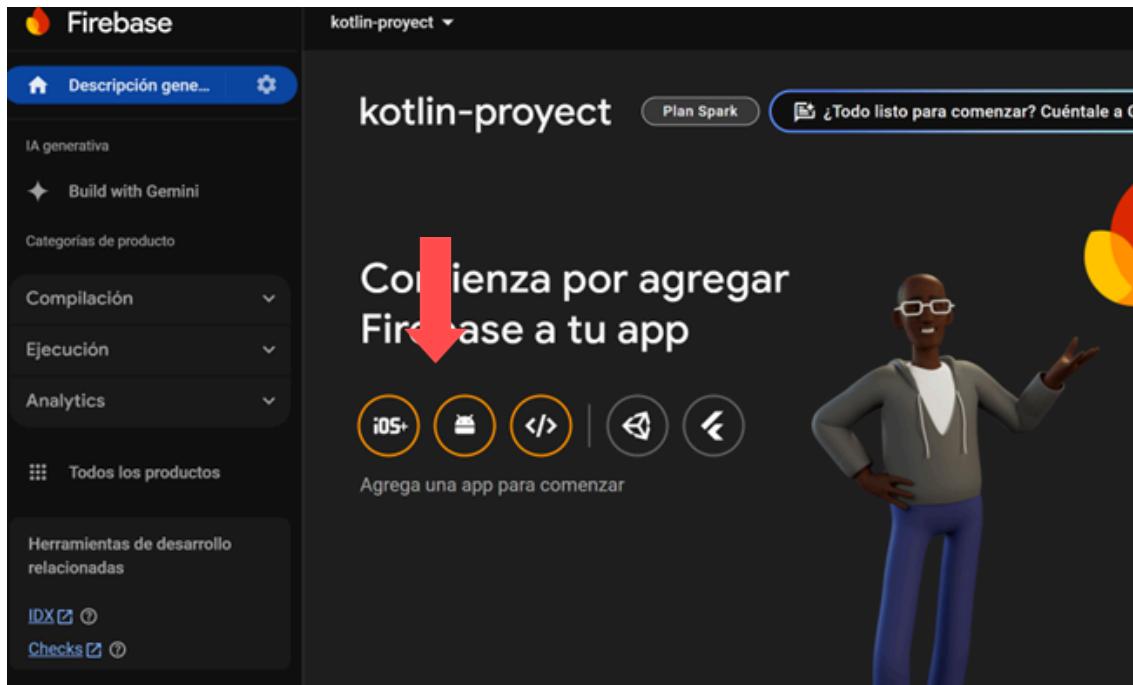
- Primero para comenzar con la autenticación por medio de los servicios de Google, debemos acceder al panel principal de Firebase y daremos clic en crear un proyecto



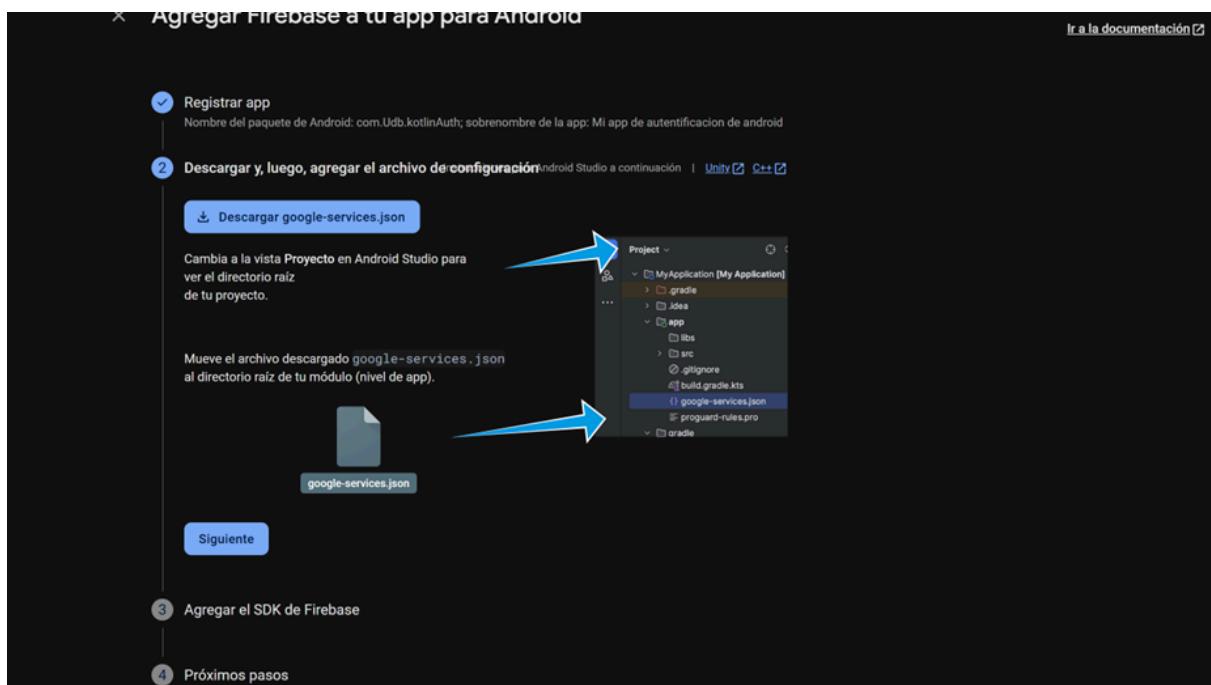
- Deberemos darle un nombre identificativo para luego utilizarlo en nuestra aplicación



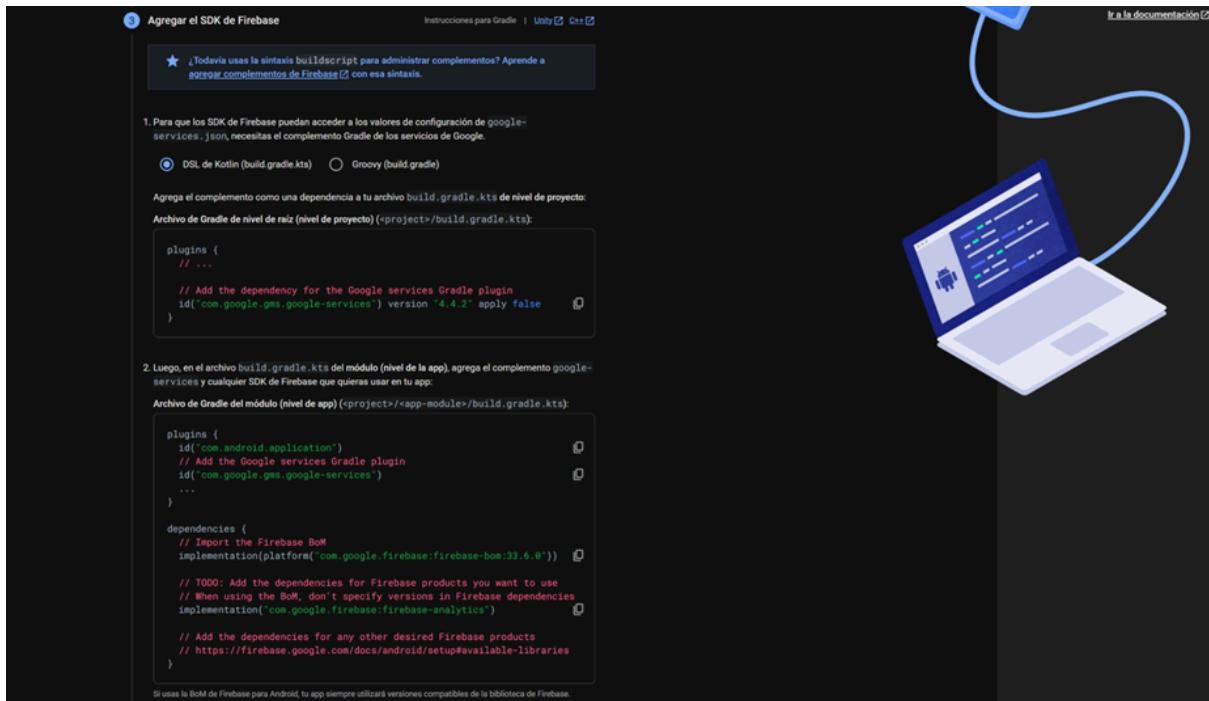
- Luego en nuestra consola seleccionamos el proyecto y nos mostrará el panel de control de nuestro proyecto, aquí seleccionaremos el ícono de Android, para poder agregarlo a nuestra aplicación.



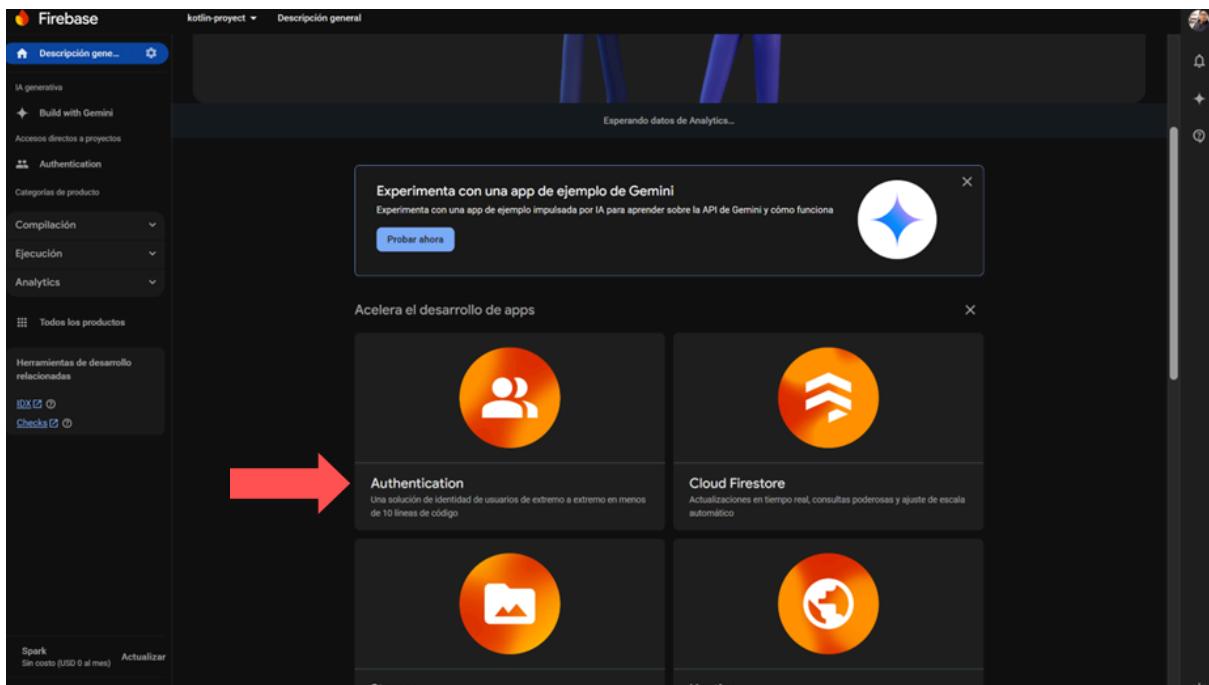
- Luego deberemos llenar los datos, para que Firebase genere el archivo de configuración, que luego deberemos utilizar en nuestra aplicación



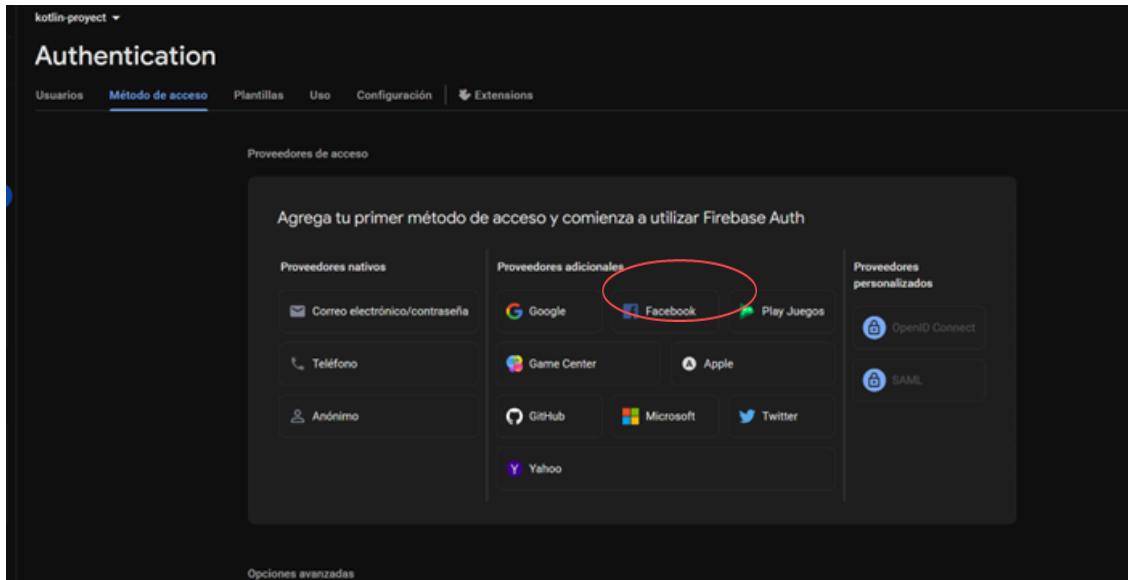
- Al descargar el archivo, podremos importarlo en nuestro proyecto, luego de esto deberemos agregar las líneas de código necesarias para poder utilizar este servicio, líneas que nos proporciona el mismo Firebase



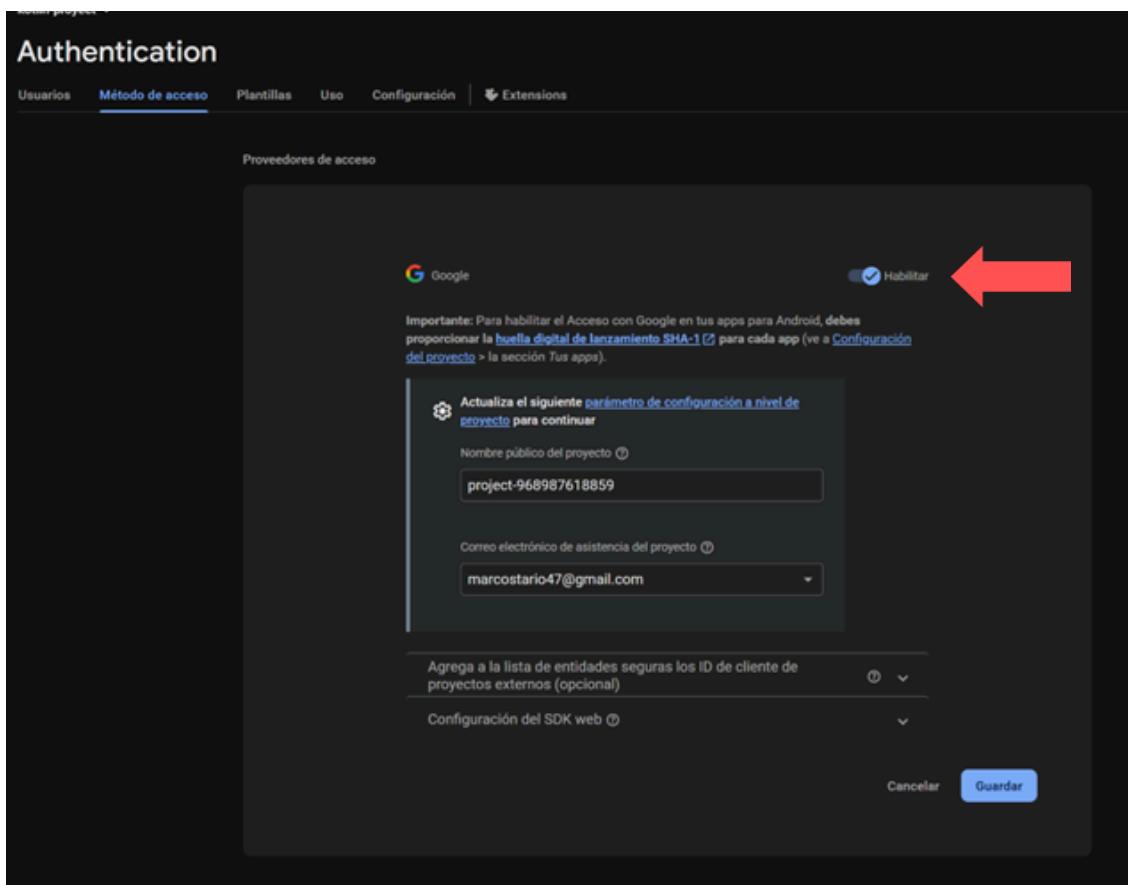
- Por consiguiente, regresaremos a la consola principal del proyecto y daremos click en la opción de Autentificación.



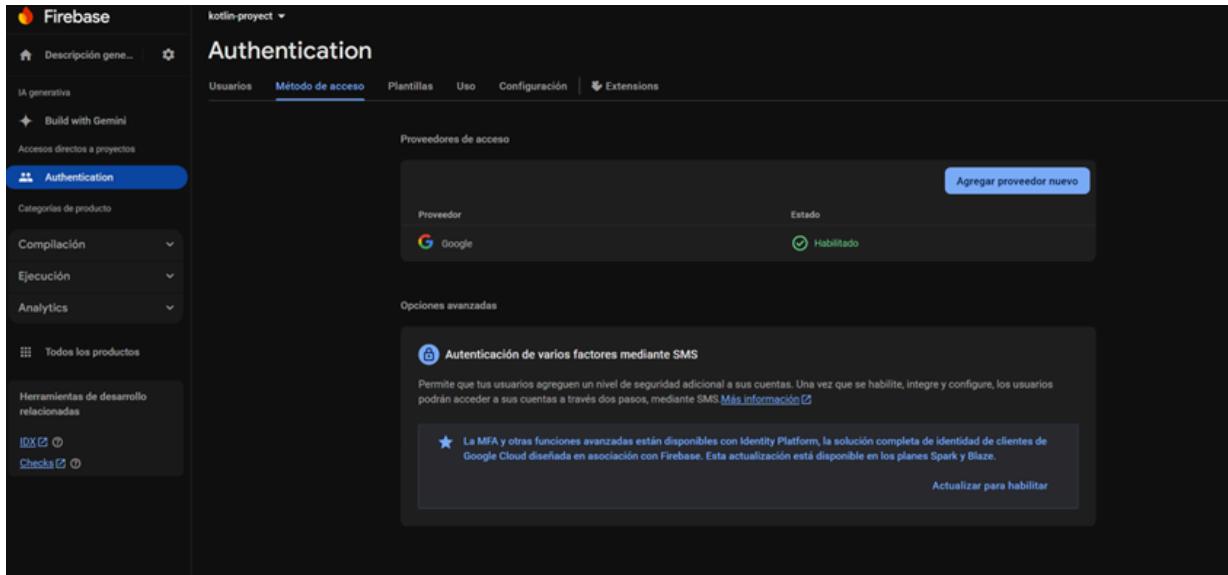
- En este menú se nos preguntará con qué tipo de autentificación queremos trabajar y seleccionaremos los servicios de Google



- Habilitaremos el inicio de sesión por Google y brindaremos un correo de asistencia al proyecto



- Siguiendo estos pasos ya tendremos configurado desde Firebase el inicio de sesión por Google. Solo deberemos en nuestra aplicación incluir el botón para utilizar estos servicios



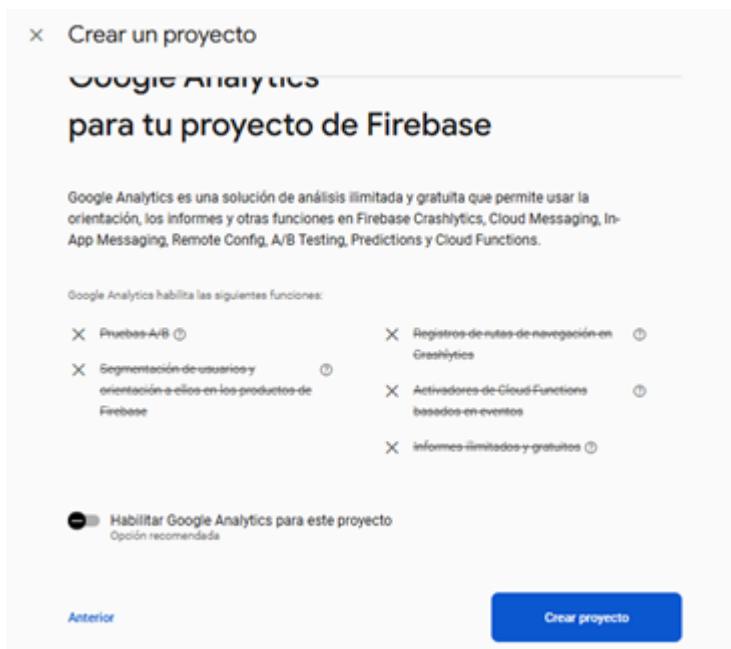
The screenshot shows the Firebase Authentication console for a project named "kotlin-project". The "Authentication" tab is selected. Under "Proveedores de acceso", there is a table with one row for "Google". The "Estado" column shows a green checkmark and the word "Habilitado". A blue button labeled "Agregar proveedor nuevo" is visible. Below the table, under "Opciones avanzadas", there is a section for "Autenticación de varios factores mediante SMS" with a note about enabling two-step verification. A button labeled "Actualizar para habilitar" is at the bottom of this section.

Desarrollo de Pantalla de Login

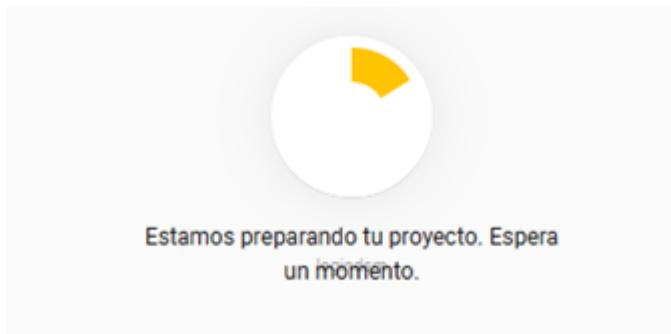
- Crear un nuevo proyecto en firebase (se debe tener una cuenta)



- Deshabilitamos google analytics y damos clic en crear proyecto



- Esperar hasta que el proyecto se cree



- Seleccionamos la plataforma para el acceso a firebase para nuestro caso seleccionaremos android



- Ingresamos el nombre del paquete para nuestro caso (com.dsm.login) y generamos una clave SHA-1 y presionamos en registrar app

Agregar Firebase a tu app para Android

1 Registrar app

Nombre del paquete de Android (?)

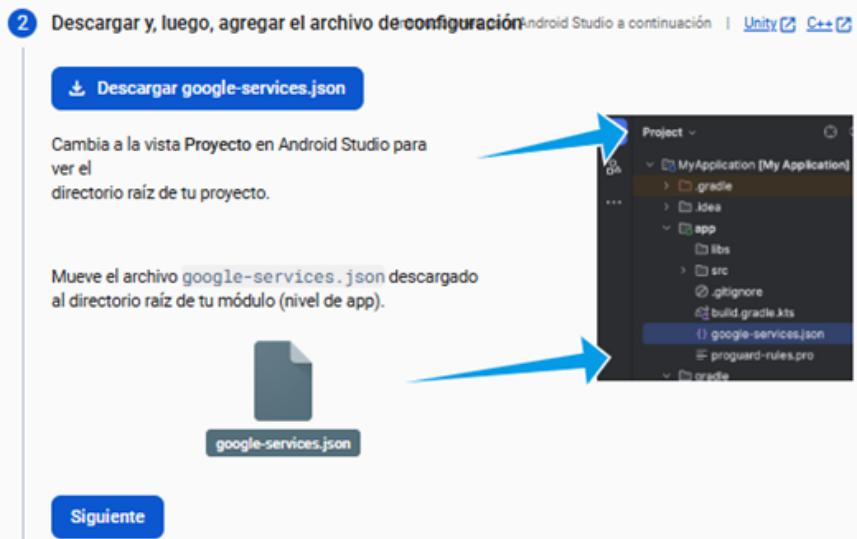
Sobrenombre de la app(Opcional) (?)

Certificado de firma SHA-1 de depuración (opcional) (?)

Obligatoria para Dynamic Links y el Acceso con Google o la compatibilidad con un número de teléfono en Auth. Puedes editar las claves SHA-1 en Configuración.

Registrar app

- Descargamos el archivo google-services.json que contiene toda la información para acceder desde nuestra App en android estudio



- Agregamos el SDK de firebase a nuestro proyecto importando las librerías necesarias y siguiendo los pasos mostrados

3 Agregar el SDK de Firebase Instrucciones para Gradle | Unity | C++ |

★ ¿Todavía usas la sintaxis buildscript para administrar complementos? Aprende a [agregar complementos de Firebase](#) con esa sintaxis.

1. Para que los SDK de Firebase puedan acceder a los valores de configuración de google-services.json, necesitas el complemento Gradle de los servicios de Google.

DSL de Kotlin (build.gradle.kts) Groovy (build.gradle)

Agrega el complemento como una dependencia a tu archivo build.gradle.kts de nivel de proyecto:

Archivo de Gradle de nivel de raíz (nivel de proyecto) (<project>/build.gradle.kts):

```
plugins {
    // ...

    // Add the dependency for the Google services Gradle plugin
    id("com.google.gms.google-services") version "4.4.2" apply false
}
```

2. Luego, en el archivo build.gradle.kts del módulo (nivel de app), agrega los complementos google-services y cualquier SDK de Firebase que quieras usar en tu app:

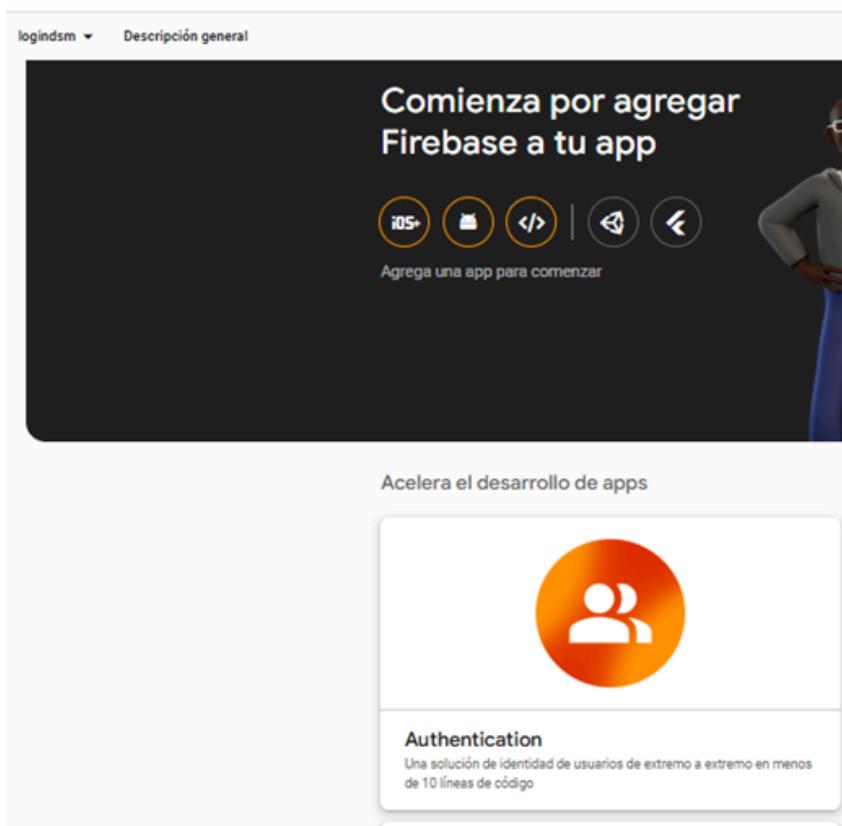
Archivo de Gradle del módulo (nivel de app) (<project>/<app-module>/build.gradle.kts):

```
plugins {
    id("com.android.application")
    // Add the Google services Gradle plugin
    id("com.google.gms.google-services")
    ...
}

dependencies {
    // Import the Firebase BoM
    implementation(platform("com.google.firebase:firebase-bom:33.6.0"))

    // TODO: Add the dependencies for Firebase products you want to use
    // When using the BoM, don't specify versions in Firebase dependencies
    // https://firebase.google.com/docs/android/setup#available-libraries
}
```

- Seleccionamos siguiente , luego ir a consola esto nos mandara a nuestro proyecto creado ahí seleccionamos Authentication



- Luego nos desplazamos a la pestaña métodos de acceso y seleccionamos los proveedores (correo electrónico /contraseña y Google para nuestro caso, dar clic en habilitar y guardar)

- Para correo y contraseña

Proveedor	Estado
Correo electrónico/contraseña	<input checked="" type="checkbox"/> Habilitar
Vínculo del correo electrónico (acceso sin contraseña)	<input checked="" type="checkbox"/> Habilitar

Borrar proveedor Cancelar Guardar

- Para acceso con google

Importante: Para habilitar el Acceso con Google en tus apps para Android, debes proporcionar la huella digital de lanzamiento SHA-1 para cada app (ve a [Configuración del proyecto](#) > la sección *Tus apps*).

Actualiza la siguiente [configuración a nivel de proyecto](#) para continuar

Nombre público del proyecto

Correo electrónico de asistencia del proyecto

Agrega a la lista de entidades seguras los ID de cliente de proyectos externos (opcional)

Configuración del SDK web

Cancelar Guardar

- Crear un usuario para acceso con correo y contraseña

- Damos clic en la pestaña usuarios, clic en agregar usuario, ingresamos el correo y la contraseña y presionamos agregar usuario y ya estaría listo nuestro acceso a la api de firebase

Buscar por dirección de correo electrónico, número de teléfono o UID de usuario

Identificador Proveedores Fecha de creación Fecha de acceso UID de usuario

Agrega un usuario con correo electrónico y contraseña

Correo electrónico: prueba@gmail.com Contraseña: 12345678

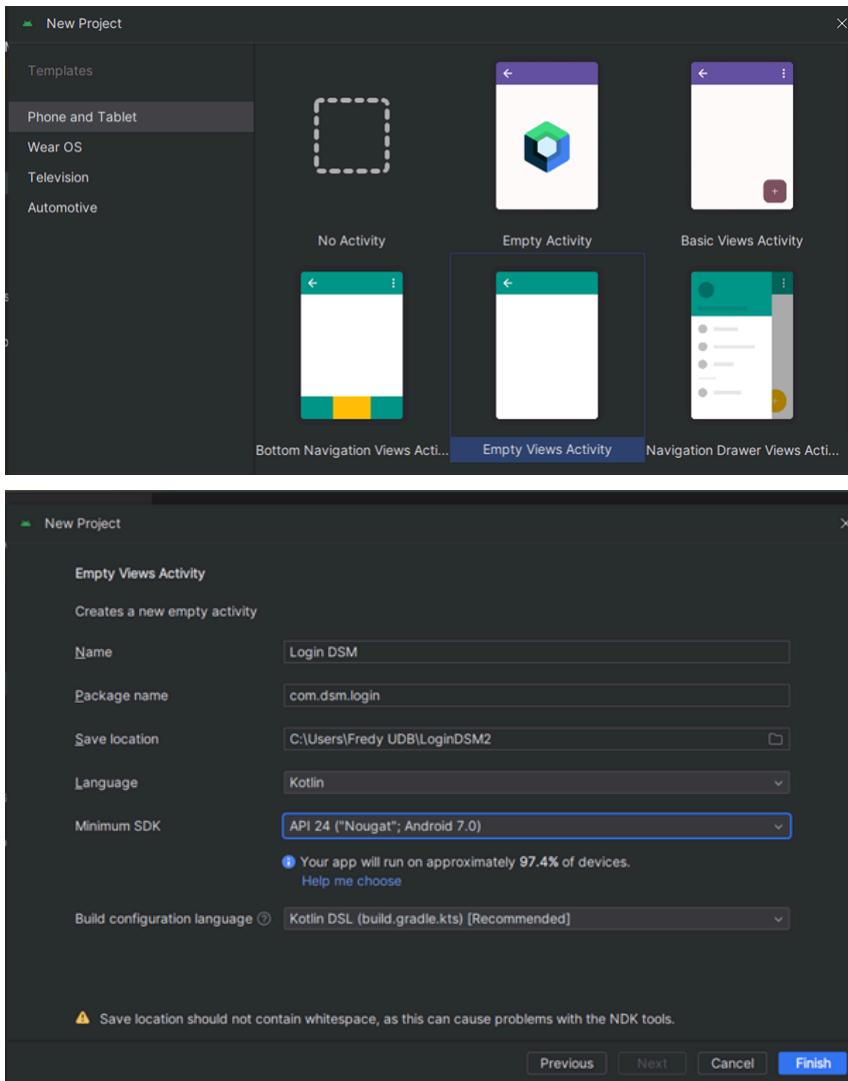
Cancelar Agregar usuario

- Así se vería la lista con usuarios

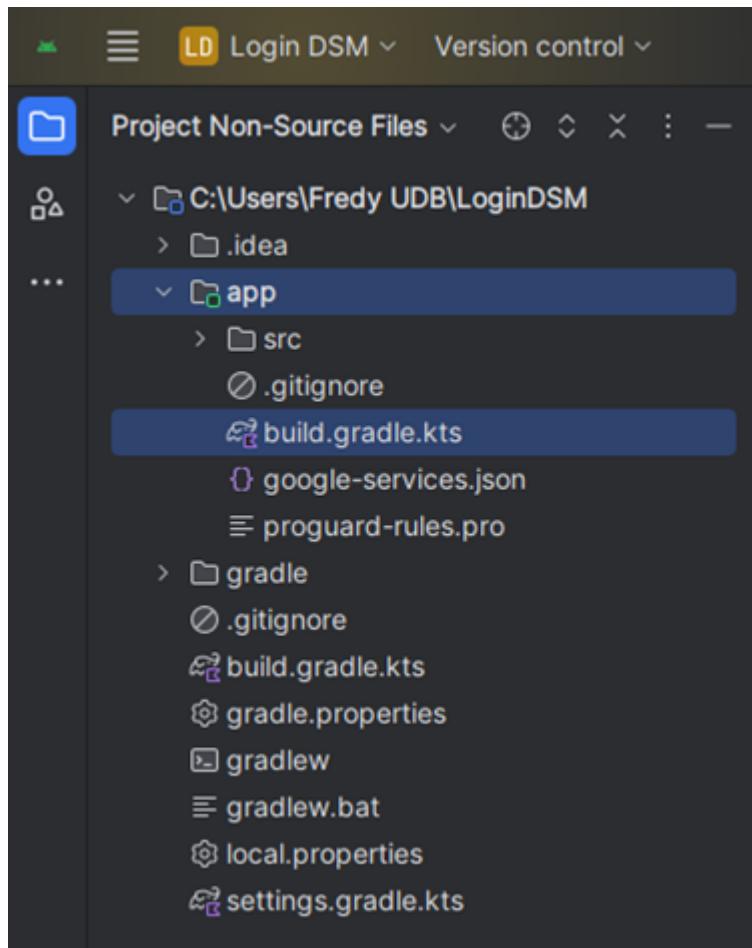
Buscar por dirección de correo electrónico, número de teléfono o UID de usuario					Agregar usuario		
Identificador	Proveedores	Fecha de creación	Fecha de acceso	UID de usuario			
freddyantonioalfaro@gm...		17 nov 2024	18 nov 2024	AI26L6Py7qRVDhbHBsMHO8x...			
prueba@gmail.com		10 nov 2024	17 nov 2024	IGiYzkxIVcDvjZXdrMjDOFDWk...			
Filas por página:					50		1 – 2 of 2

Creación del proyecto en android studio

- Crear un proyecto en blanco y seleccionamos Kotlin como lenguaje de programación y finish con esto se crea el proyecto



- Copiamos el archivo google-services.json que se descargó cuando se creó la App en Firebase a la ruta app/ google-services.json



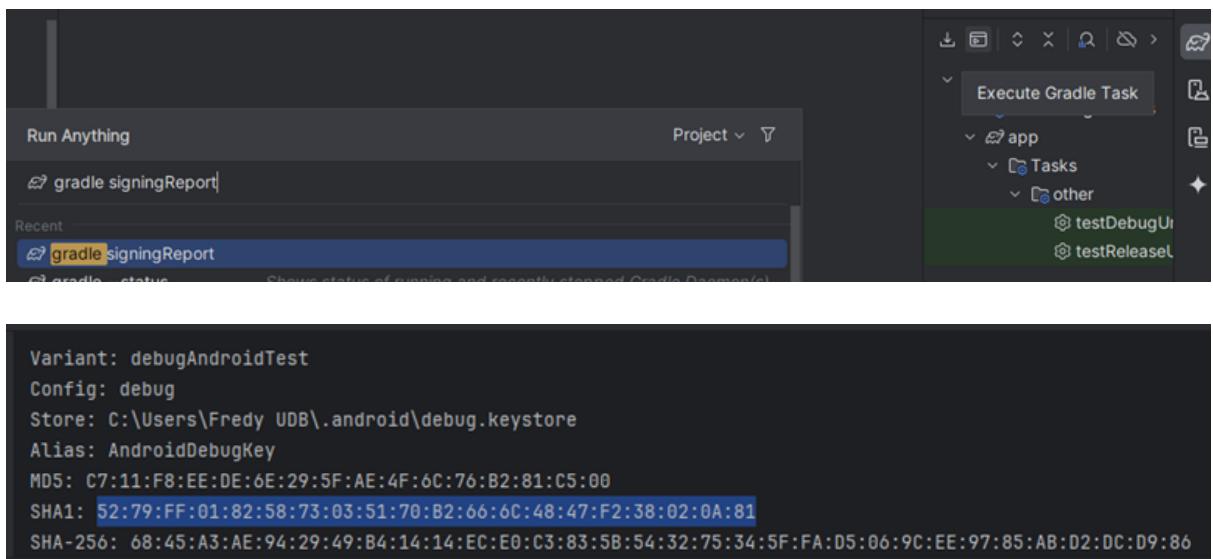
- También agregamos las dependencias de firebase a build.gradle.kts de la ruta app

```
    androidTestImplementation('libs.androidx.espresso.core')
        implementation(platform("com.google.firebase:firebase-bom:33.5.1"))
        implementation ("com.google.firebase:firebase-auth:22.1.0")
        implementation ("com.google.android.gms:play-services-auth:20.7.0")
```

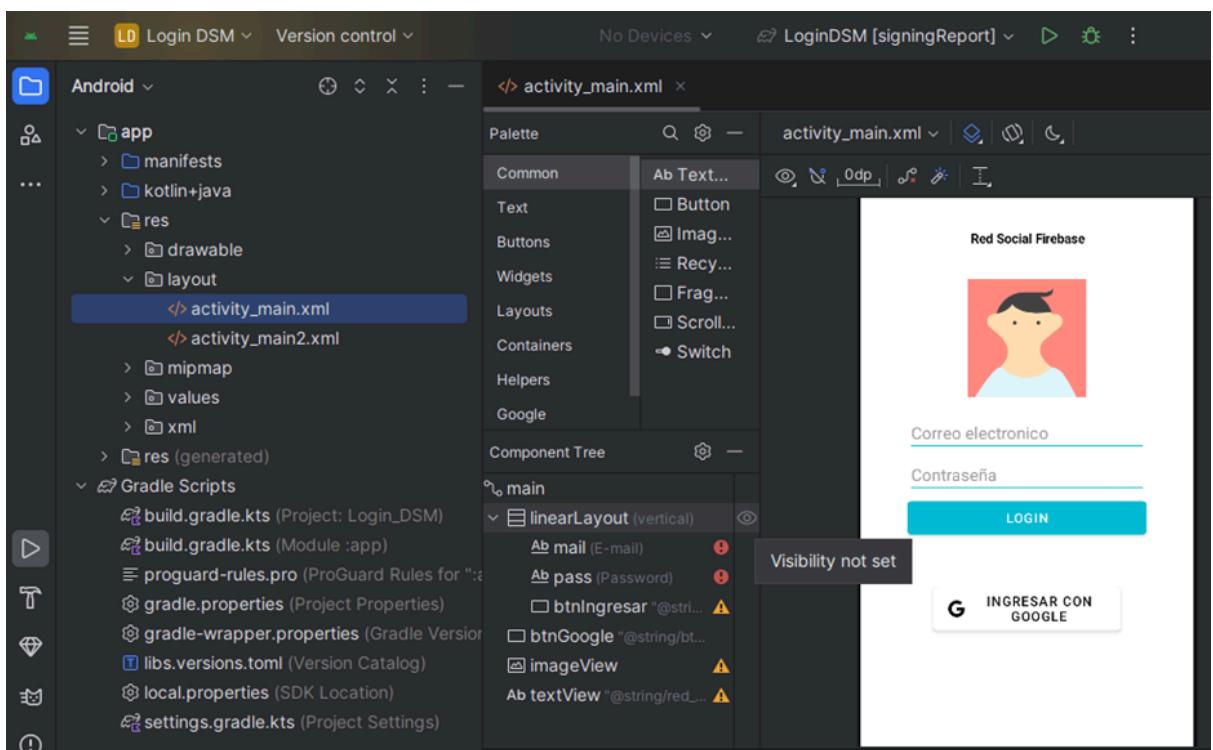
- Agregar plugins en el archivo build.gradle.kts del directorio raíz del proyecto

```
plugins {
    this: PluginDependenciesSpecScope
    alias(libs.plugins.android.application) apply false
    alias(libs.plugins.jetbrains.kotlin.android) apply false
    id("com.google.gms.google-services") version "4.4.2" apply false
}
```

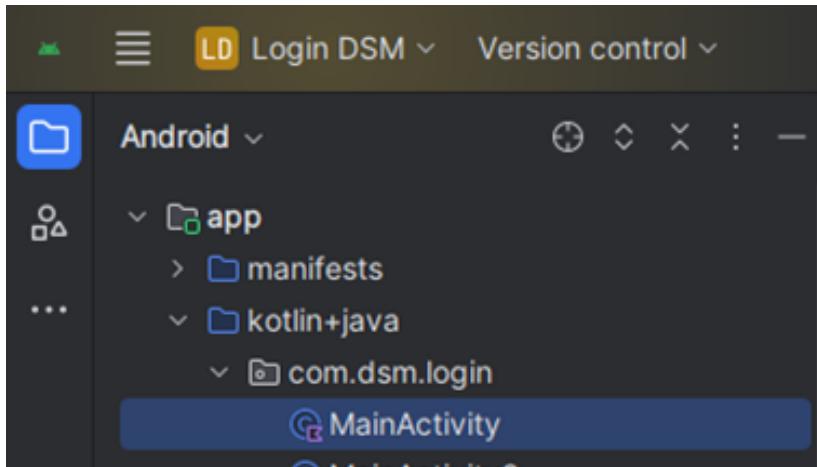
- Obtener el SHA-1 de nuestra app clic en el elefante que está a mano derecha , luego Execute Gradle Task
- Luego escribir gradle signingReport y dar enter esto mostrará el SHA-1 que es el que se ingresa en firebase cuando pide el SHA-1



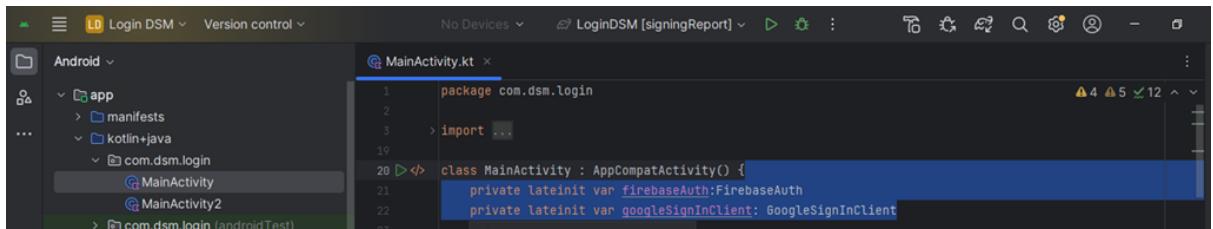
- Diseñar la interfaz gráfica del login
 - Abrimos el activity_main.xml y arrastramos botones texto e imágenes a nuestro gusto quedando de la siguiente forma el diseño



- Codificando la App, abrimos el archivo MainActivity



- Creamos las referencias a firebase con



- Iniciar firebase y las Referencias hacia la vista Xml

```

private lateinit var firebaseAuth: FirebaseAuth
private lateinit var googleSignInClient: GoogleSignInClient

override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)

    firebaseAuth = FirebaseAuth.getInstance()

    val btnIngresar :Button=findViewById(R.id.btnIngresar)
    val email:TextView=findViewById(R.id.mail)
    val pass:TextView=findViewById(R.id.pass)
    val btngGoogle:Button=findViewById(R.id.btnGoogle);
}

```

- Validación del campo email y contraseña

```
btnIngresar.setOnClickListener()
{
    if(!email.text.isEmpty() || !email.text.isBlank() && (!pass.text.isEmpty() || !pass.text.isBlank()))
    {
        signIn(email.text.toString(), pass.text.toString())
    }
    else
    {
        Toast.makeText(baseContext, text: "Error correo o contraseña vacio",Toast.LENGTH_SHORT).
    }
}
```

- Lógica para logueo con correo y contraseña en firebase

```
private fun signIn(email: String, pass: String)
{
    try {
        Log.d( tag: "email", email)
        firebaseAuth.signInWithEmailAndPassword(email, pass)
            .addOnCompleteListener(this) { task->
                if(task.isSuccessful)
                {
                    val user=firebaseAuth.currentUser
                    Toast.makeText(baseContext, user?.uid.toString(),Toast.LENGTH_SHORT).show()
                    val i=Intent( packageContext: this,MainActivity2::class.java)
                    startActivity(i);
                }
                else
                {
                    Toast.makeText(baseContext, text: "Credenciales invalidas",Toast.LENGTH_SHORT).show()
                }
            }
    }catch (e:Exception)
    {
        Log.e( tag: "Error",e.toString())
        Toast.makeText(baseContext, text: "Error "+e,Toast.LENGTH_LONG).show()
    }
}
```

- Lógica para iniciar sesión con cuenta google

```
val gso = GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
    .requestIdToken(getString(R.string.default_web_client_id)) // ID configurado en google
    .requestEmail()
    .build()

googleSignInClient = GoogleSignIn.getClient( activity: this, gso)
btngoogle.setOnClickListener()
{
    it: View!
    signInWithGoogle()
}
```

```

private fun signInWithGoogle() {
    val signInIntent = googleSignInClient.signInIntent
    launcher.launch(signInIntent)
}

private val launcher=registerForActivityResult(ActivityResultContracts.StartActivityForResult)
{
    result->
    if(result.resultCode==Activity.RESULT_OK)
    {
        val task = GoogleSignIn.getSignedInAccountFromIntent(result.data)
        handleResults(task);
    }
    else
    {
        Toast.makeText(baseContext, text: "Ha ocurrido un error",Toast.LENGTH_SHORT).show()
    }
}

private fun handleResults(task: Task<GoogleSignInAccount>)
{
    if(task.isSuccessful)
    {
        val account:GoogleSignInAccount?=task.result
        if (account!=null)
        {
            updateUI(account)
        }
        else
        {
            Toast.makeText(baseContext, text: "Ha ocurrido un Error ",Toast.LENGTH_SHORT).show()
        }
    }
}

private fun updateUI(account: GoogleSignInAccount) {
    val credential=GoogleAuthProvider.getCredential(account.idToken,null)
    firebaseAuth.signInWithCredential(credential).addOnCompleteListener{ it: Task<AuthResult!>
        if(it.isSuccessful)
        {
            Toast.makeText(baseContext, text: "Iniciando",Toast.LENGTH_SHORT).show()
            val i=Intent( packageContext: this,MainActivity2::class.java)
            startActivity(i);
        }
        else
        {
            Toast.makeText(baseContext, text: "Error "+it.exception.toString(),Toast.LENGTH_SHORT).show()
        }
    }
}

```

Bibliografías:

- *Autentica con Firebase mediante vínculos de correo electrónico.* (n.d.).
<https://firebase.google.com/docs/auth/flutter/email-link-auth?hl=es-419>
- *Identidad federada y acceso mediante redes sociales.* (n.d.).
<https://firebase.google.com/docs/auth/flutter/federated-auth?hl=es-419>
- *Autenticación telefónica.* (n.d.).
<https://firebase.google.com/docs/auth/flutter/phone-auth?hl=es-419>
- *Autentica con Firebase mediante un sistema de autenticación personalizado.* (n.d.). <https://firebase.google.com/docs/auth/flutter/custom-auth?hl=es-419>
- *Autentica con Firebase de forma anónima.* (n.d.).
<https://firebase.google.com/docs/auth/flutter/anonymous-auth?hl=es-419>