



Trabajo de investigación

Investigación y Desarrollo de Autenticación con Android con

Kotlin utilizando Firebase

-Desarrollo de Software para Móviles DSM941 G01T -

Docente:

Ing. Alexander Alberto Siguenza Campos

Integrantes:

- William Alberto García Gómez - GG212522
- Christian Yahir López Hernández - LH212531
- Yahir Stewart Sibrian Arriola - SA212551
- Alberto Joseph Mendoza Moreno - MM200462
- Juan Jose Lue Valdez -LV231966

Fecha:

Índice

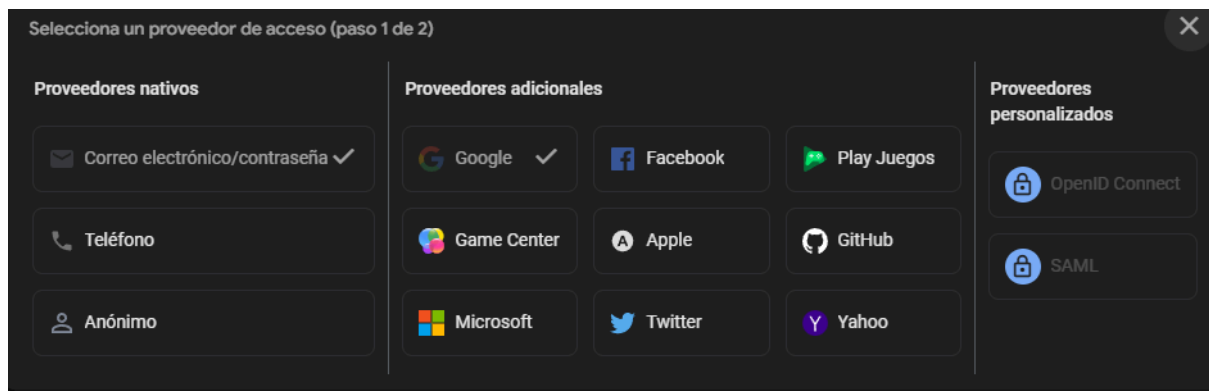
¿Qué es Firebase?.....	3
Tipos de autenticación.....	3
- Autenticación de correo electrónico y contraseña.....	3
- Autenticación redes sociales.....	4
- Autenticación anónima.....	4
- Autenticación telefónica.....	4
Implementación de la autenticación.....	5
- Autenticación por correo y contraseña.....	5
- Autenticación Google.....	7
- Agregar Firebase al proyecto.....	9
Implementar lógica.....	10
- Registro correo y contraseña.....	10
Inicialización.....	10
Registro.....	10
Navegación a login.....	11
- Login correo y contraseña.....	12
Inicialización.....	12
Login con correo y contraseña.....	13
Navegación a pantalla de registro.....	13
- Login mediante Google.....	14
Vistas de la App.....	15
Registro.....	15
Inicio de sesión.....	16
Mensajes de errores.....	17
GitHub.....	19
- Video funcionamiento.....	19

¿Qué es Firebase?

Firebase de Google es una plataforma en la nube para el desarrollo de aplicaciones web y móvil. Se destaca por su enfoque en la integración con aplicaciones de Android, iOS y web, y ofrece funcionalidades que abarcan desde bases de datos en tiempo real, autenticación de usuarios, almacenamiento de archivos, hasta análisis de datos y notificaciones push.

También, cuenta con diversas opciones de autenticación, lo que permite gestionar de manera sencilla el acceso seguro a aplicaciones móviles y web.

Tipos de autenticación



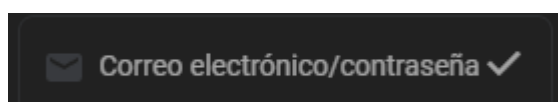
- Autenticación de correo electrónico y contraseña

También, cuenta con diversas opciones de autenticación, lo que permite gestionar de manera sencilla el acceso seguro.

Permite crear cuentas personalizadas, y verificar el correo electrónico y poder restablecer las contraseñas. Entre sus ventajas se encuentra:

- Familiar para los usuarios.
- Fácil de implementar, y bajo costo.
- Personalización de la experiencia del usuario.

Dentro de las desventajas, se tiene un riesgo y probabilidad de errores de contraseña, y al no tener acceso al correo se podría perder acceso.



- Autenticación redes sociales

Permite un inicio de sesión con cuentas de Google, Facebook, X (Twitter), Play Juegos, Apple, GitHub, Microsoft, Yahoo y Game Center, y garantiza un acceso rápido y seguro. Ventajas de inicio de sesión con redes sociales:

- Reducción del tiempo de registro.
- Uso de cuentas existentes en otras plataformas.

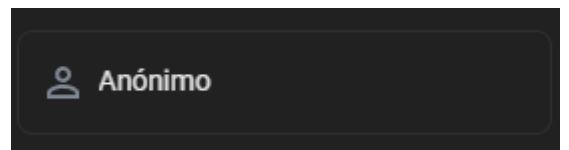
Desventajas de inicio de sesión con redes sociales:

- Dependencia de terceros.
- Posibles cambios en las API de los proveedores.

- Autenticación anónima

Garantiza un acceso a la aplicación sin necesidad de registrarse, y es ideal para pruebas o funcionalidades de invitado. La desventaja de este método de autenticación, es la limitante a ciertas funciones dependiendo la aplicación. Por otro lado, en sus ventajas se tiene:

- Reducción del proceso de registro.
- Posibilidad de visualizar y explorar la aplicación.



- Autenticación telefónica

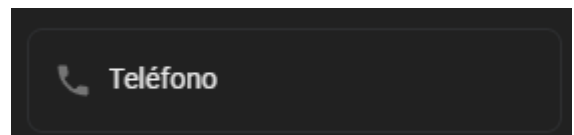
Esta autenticación permite una verificación mediante SMS, y ofrece ingreso a los usuarios sin correo electrónico.

Ventaja:

- Mayor seguridad al evitar ataques de phishing.
- Facilidad de uso en dispositivos móviles.

Desventajas:

- Costos asociados a los mensajes SMS.
- Limitaciones en algunos países o regiones.
- Dependencia de redes de telecomunicaciones



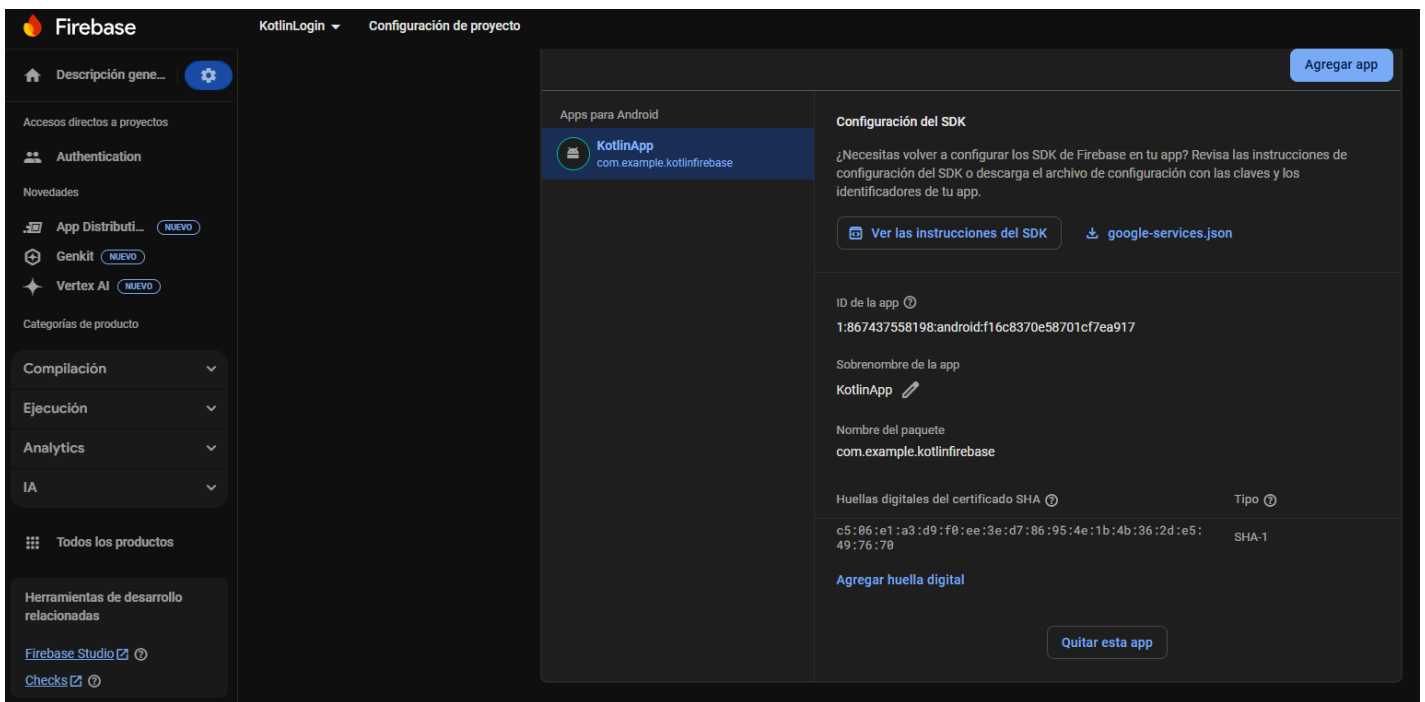
Implementación de la autenticación

- Autenticación por correo y contraseña

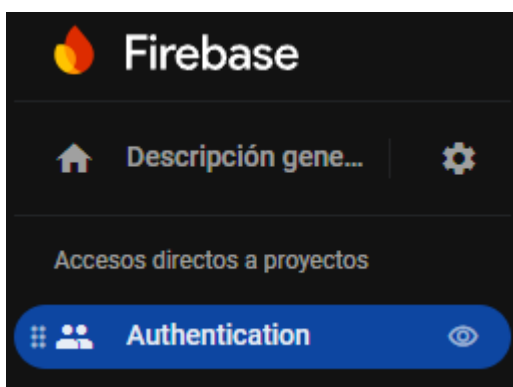
Esta autenticación permite una verificación mediante SMS, y ofrece ingreso a los usuarios sin correo electrónico.

Para poder integrar una autenticación por correo electrónico y contraseña en Firebase se puede realizar lo siguiente:

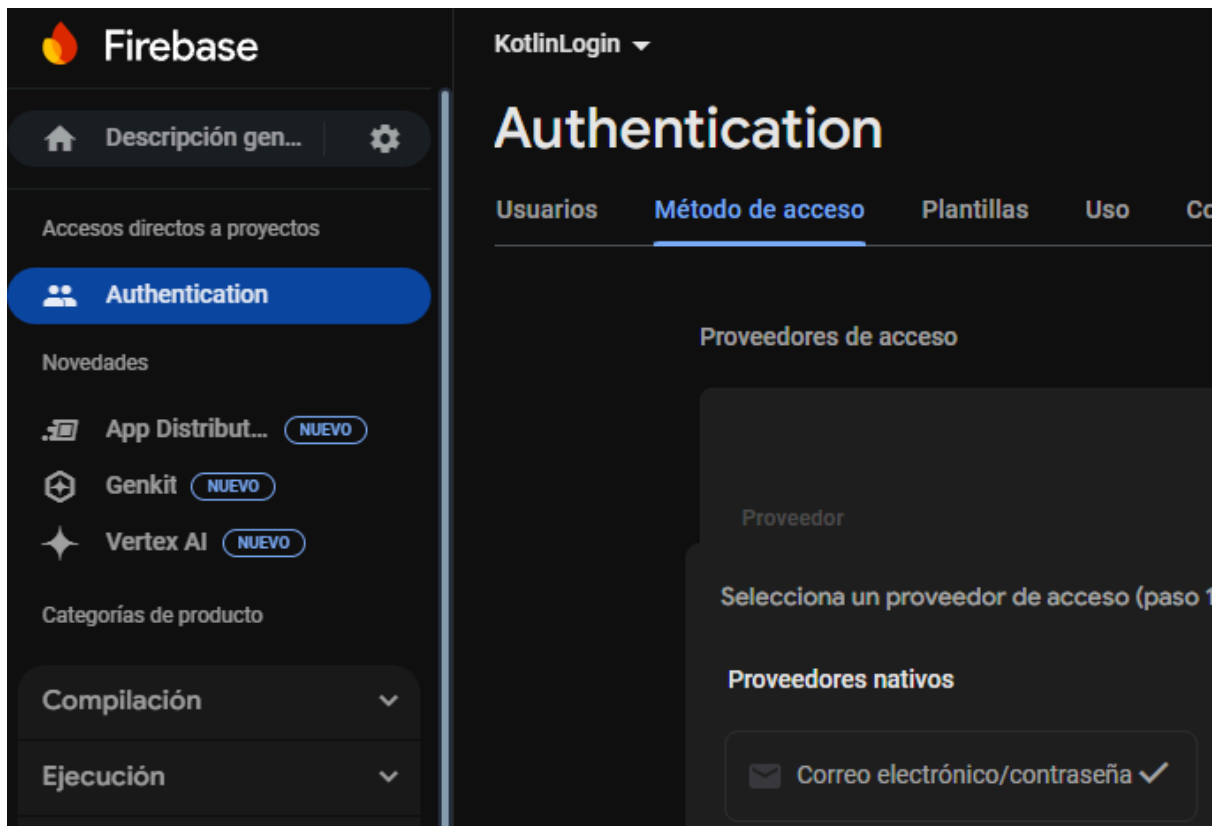
1. Crear un nuevo proyecto en Firebase
2. Configurar el nombre del proyecto
3. Cambiar configuración de proyecto
4. Agregar una App



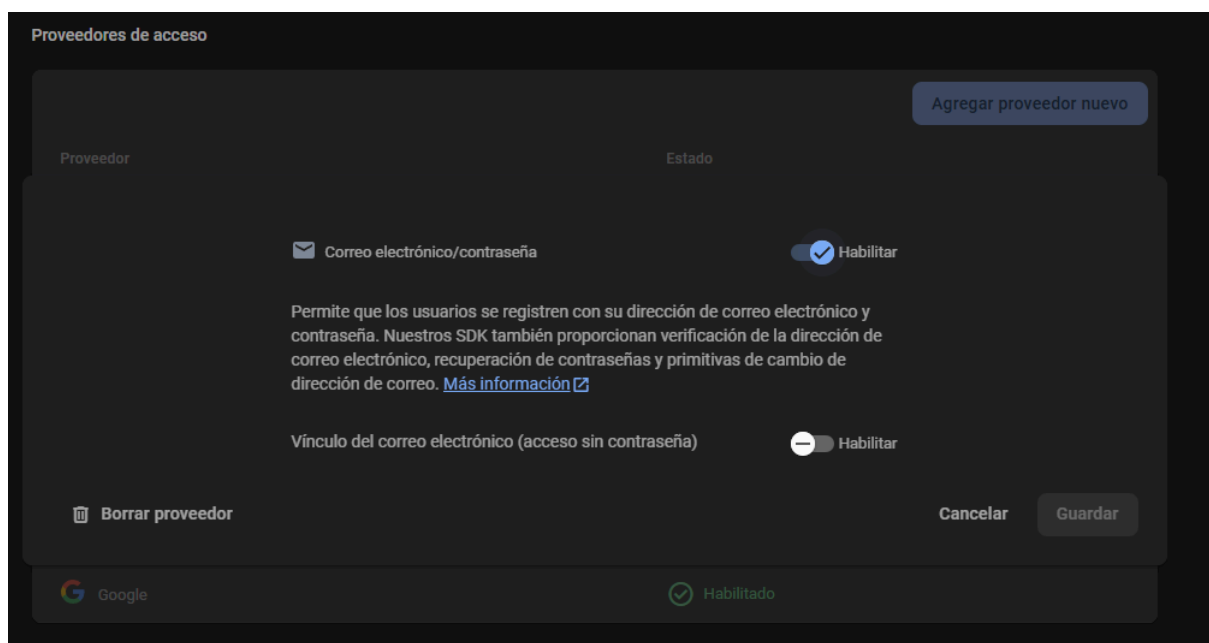
5. Seleccionar Authentication



6. Establecer Correo electrónico/contraseña



7. Guardar cambios

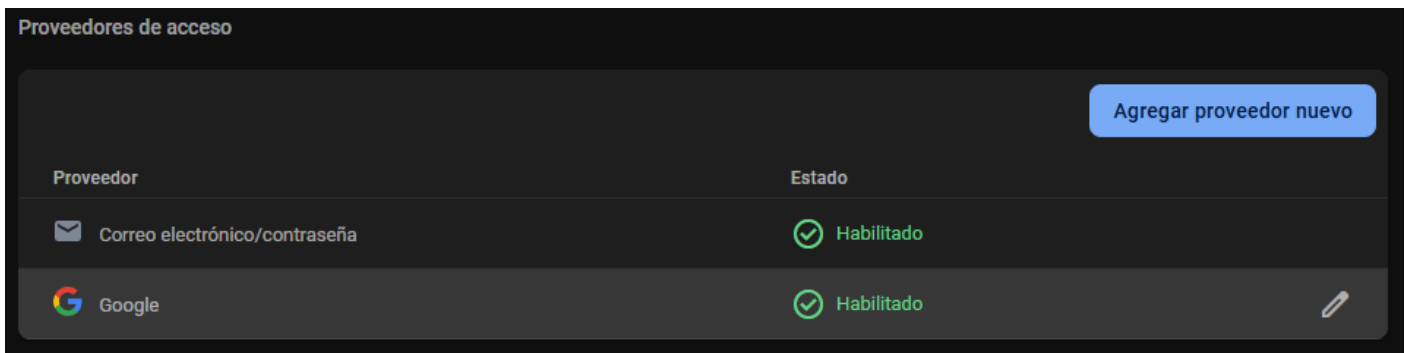


- Autenticación Google

Para poder realizar una autenticación con Google, se deben realizar los siguientes pasos:

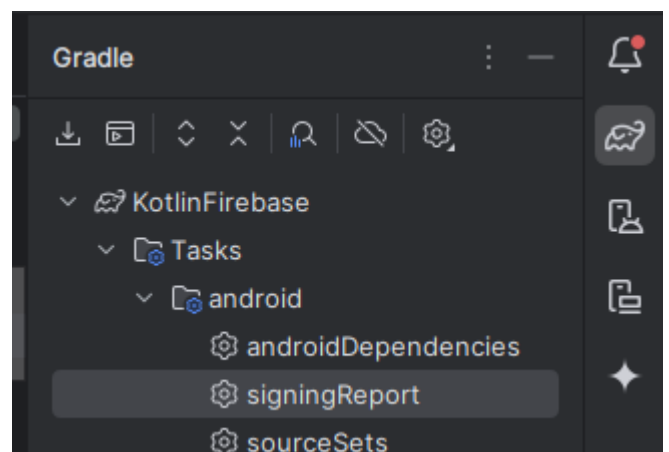
Configurar Firebase

1. Seleccionar el proyecto
2. Dirigirse a Authentication
3. Métodos de acceso
4. Establecer Google



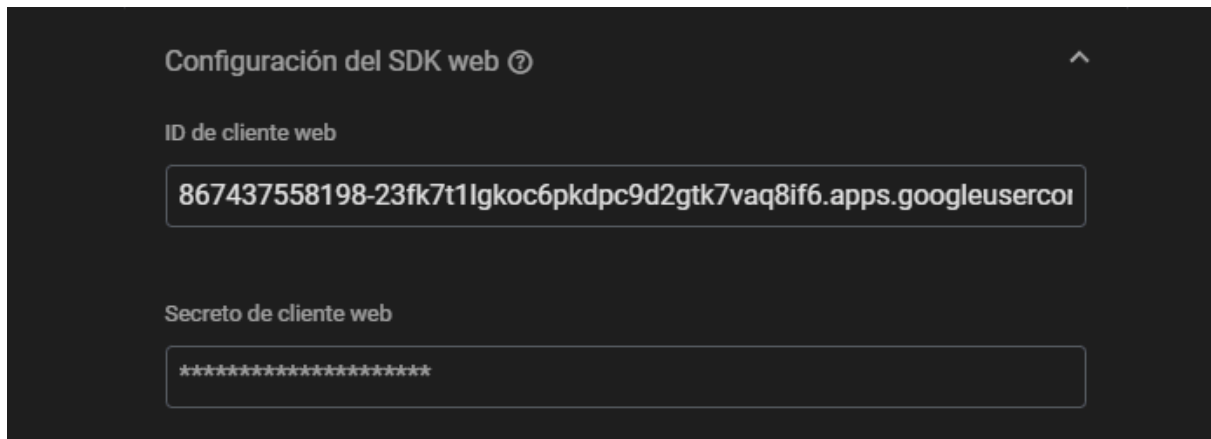
Configurar ID de cliente web

En Android Studio, se debe seleccionar la opción de Gradle, y escoger signingReport par poder generar el ID



```
Alias: null
-----
Variant: debugAndroidTest
Config: debug
Store: C:\Users\PC\.android\debug.keystore
Alias: AndroidDebugKey
MD5: 88:4A:A7:F5:45:15:07:9E:39:48:81:F1:F5:7B:B3:DD
SHA1: C5:06:E1:A3:D9:F0:EE:3E:D7:86:95:4E:1B:4B:36:2D:E5:49:76:70
SHA-256: 57:76:AC:30:B8:9A:9C:40:F5:77:1E:56:B1:DF:FA:56:E8:0C:7D:9B:63:07:1A:B4:94:BF:81:5D:CB:00:75:EF
Valid until: sábado, 10 de abril de 2055
-----
```

Una vez generado, copiar el valor del SHA-1, y pegarlo en Configuración del SDK Web. Esto permitirá vincular correctamente la app con el SDK Web de Google para autenticación.



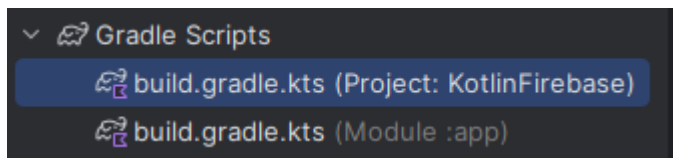
Configuración del SDK web ?

ID de cliente web

867437558198-23fk7t1lgkoc6pkdpc9d2gtk7vaq8if6.apps.googleusercontent.com

Secreto de cliente web

Agregar el complemento de Google Services como dependencia en el archivo de Gradle a nivel de raíz (a nivel de proyecto) y del módulo (a nivel de app):



Nivel proyecto:

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

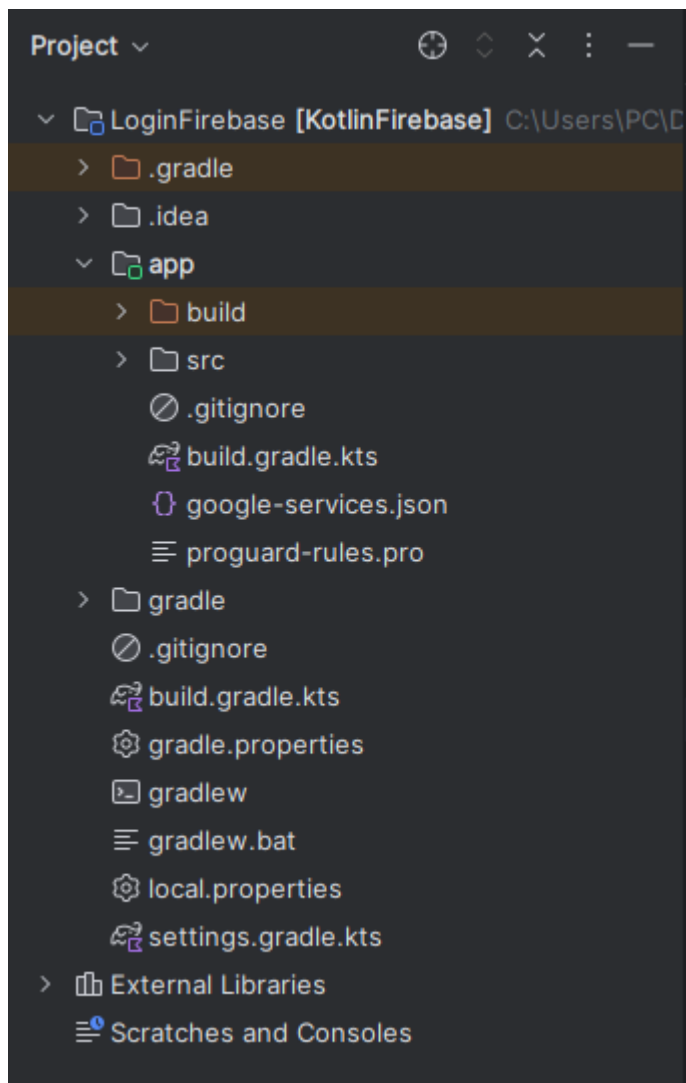
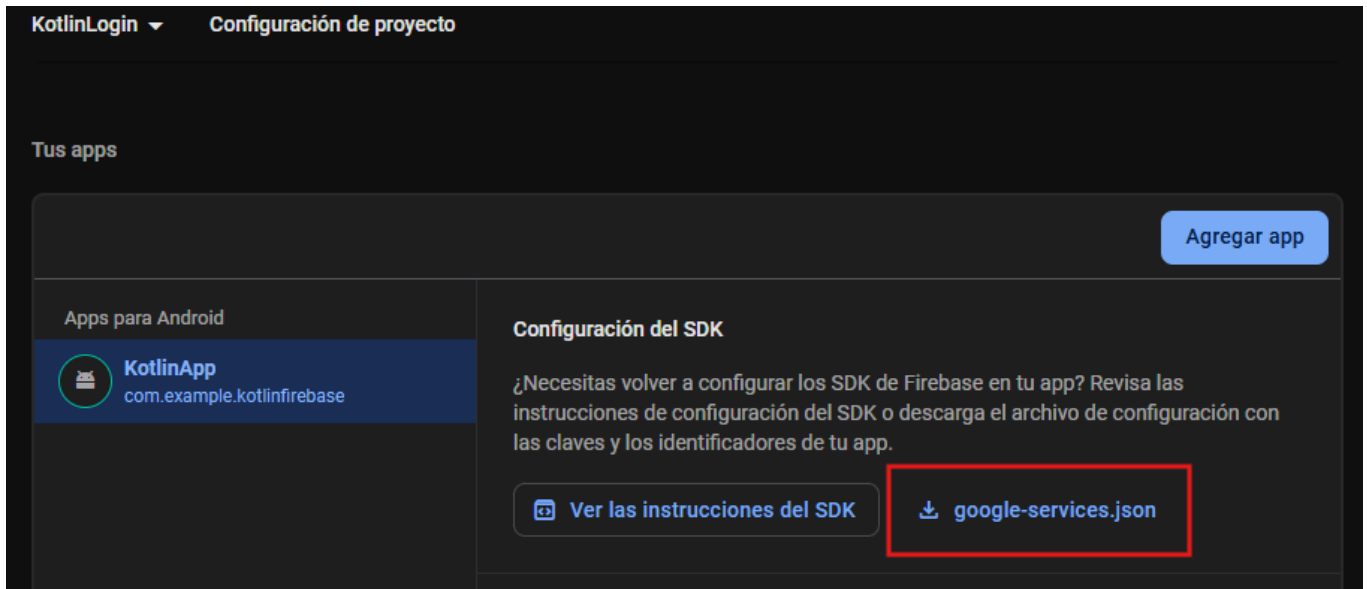
Nivel app y agregar dependencias:

```
plugins {  
    id("com.android.application")  
    id("org.jetbrains.kotlin.android")  
    id("com.google.gms.google-services")  
}
```

```
dependencies {  
    implementation(platform("com.google.firebase:firebase-bom:32.8.1"))  
    implementation("com.google.firebase:firebase-auth-ktx")  
}
```


- Agregar Firebase al proyecto

1. En Configuración del proyecto, descargar el archivo google-services.json



2. Una vez descargado, debe colocarse en la carpeta app

Implementar lógica

- Registro correo y contraseña

Inicialización

- Se vincula el layout con ActivityRegisterBinding.
- Se inicializa FirebaseAuth.

```
private lateinit var binding: ActivityRegisterBinding
private lateinit var auth: FirebaseAuth

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

    binding = ActivityRegisterBinding.inflate(layoutInflater)
    setContentView(binding.root)
    auth = FirebaseAuth
```

Registro

- Se validan los campos de correo y contraseña.
- Se verifica que el correo sea válido.

```
binding.registrarBtn.setOnClickListener{
    val email = binding.email.text.toString().trim()
    val password = binding.password.text.toString().trim()

    if (email.isEmpty()) {
        Snackbar.make(binding.root, text: "Ingresa tu correo electrónico", Snackbar.LENGTH_SHORT).show()
        return@setOnClickListener
    }
    if (password.isEmpty()) {
        binding.password.error = "La contraseña es obligatoria"
        return@setOnClickListener
    }

    if (!android.util.Patterns.EMAIL_ADDRESS.matcher(email).matches()) {
        Snackbar.make(binding.root, text: "Correo no válido", Snackbar.LENGTH_SHORT).show()
        return@setOnClickListener
    }
}
```

- Se ejecuta createUserWithEmailAndPassword. En caso de ingresar los datos validos, se inicia sesión automáticamente y se redirige a MainActivity. En caso de error, se muestra un mensaje con Toast.

```
auth.createUserWithEmailAndPassword(binding.email.getText().toString().trim(), binding.password.getText().toString().trim())
    .addOnCompleteListener(this) { task ->
        if (task.isSuccessful) {
            //inicio de sesión
            Log.d(TAG, msg: "createUserWithEmail:success")
            val user = auth.currentUser
            val intent = Intent(packageContext: this, MainActivity::class.java)
            startActivity(intent)
        } else {
            // error
            Log.w(TAG, msg: "createUserWithEmail:failure", task.exception)
            Toast.makeText(
                baseContext,
                text: "Authentication failed.",
                Toast.LENGTH_SHORT,
            ).show()
        }
    }
```

Navegación a login

- Botón **move** lleva de vuelta a LoginActivity

```
binding.move.setOnClickListener{
    val intent = Intent(packageContext: this, LoginActivity::class.java)
    startActivity(intent)
}
```

- Login correo y contraseña

Inicialización

- Se vincula el layout con ActivityLoginBinding.
- Se inicializa FirebaseAuth con Firebase.auth.

```
private lateinit var binding: ActivityLoginBinding
private lateinit var auth: FirebaseAuth
private lateinit var googleSignInClient: GoogleSignInClient
private val RC_SIGN_IN = 9001

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

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

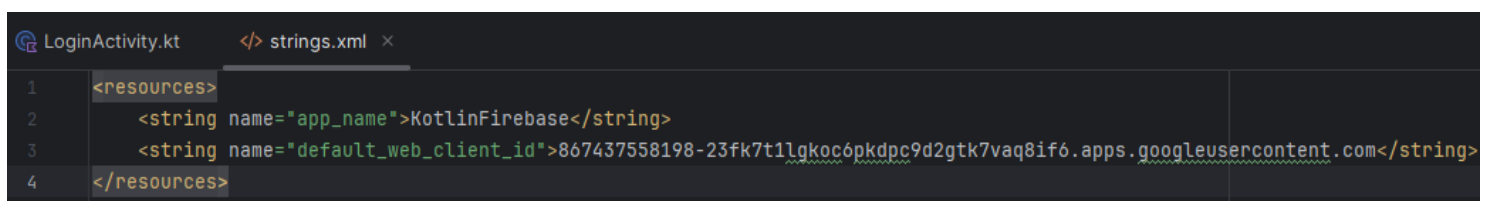
    auth = Firebase.auth
```

- Se configura el inicio de sesión con Google (GoogleSignInOptions y GoogleSignInClient)

```
// Configurar Google
val gso = GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
    .requestIdToken(getString(R.string.default_web_client_id))
    .requestEmail()
    .build()

googleSignInClient = GoogleSignIn.getClient(activity: this, gso)
```

- Configurar "default_web_client_id" en strings.xml



```
LoginActivity.kt  </> strings.xml  x
1  <resources>
2      <string name="app_name">KotlinFirebase</string>
3      <string name="default_web_client_id">867437558198-23fk7t1lgkoc6pkdpc9dgtk7vaq8if6.apps.googleusercontent.com</string>
4  </resources>
```

Login con correo y contraseña

- Se valida el email y la contraseña

```
// Botón de login con correo/contraseña
binding.iniciarBtn.setOnClickListener{
    val email = binding.email.text.toString().trim()
    val password = binding.password.text.toString().trim()

    if (email.isEmpty()) {
        Snackbar.make(binding.root, text: "Ingresa tu correo electrónico", Snackbar.LENGTH_SHORT).show()
        return@setOnClickListener
    }

    if (password.isEmpty()) {
        binding.password.error = "La contraseña es obligatoria"
        return@setOnClickListener
    }

    if (!android.util.Patterns.EMAIL_ADDRESS.matcher(email).matches()) {
        binding.email.error = "Correo no válido"
        return@setOnClickListener
    }
}
```

- Si son válidos, se llama SignInWithEmailAndPassword . En caso de éxito, se navega a MainActivity. En caso de error, se muestra el mensaje o un error de credenciales.

```
auth.signInWithEmailAndPassword(binding.email.getText().toString().trim(), binding.password.getText().toString().trim())
    .addOnCompleteListener(this) { task ->
        if (task.isSuccessful) {
            startActivity(Intent( packageContext: this, MainActivity::class.java))
            finish()
        } else {
            val exception = task.exception
            if (exception is FirebaseAuthInvalidCredentialsException) {
                // Si las Credenciales son incorrectas
                binding.password.error = "Credenciales incorrectas"
            } else {
                Snackbar.make(binding.root, text: "Error: ${exception?.message}", Snackbar.LENGTH_SHORT).show()
            }
        }
    }
}
```

Navegación a pantalla de registro

- Botón **move** lleva de vuelta a RegisterActivity

```
binding.move.setOnClickListener{
    val intent = Intent( packageContext: this, RegisterActivity::class.java)
    startActivity(intent)
}
```

- Login mediante Google

Al presionar el botón de Google, se lanza un intent que abre la interfaz de selección de cuentas de Google.

```
// Botón de login con Google
binding.googleBtn.setOnClickListener {
    val signInIntent = googleSignInClient.signInIntent
    startActivityForResult(signInIntent, RC_SIGN_IN)
}
```

Una vez que el usuario elige una cuenta y se obtiene una respuesta y en `onActivityResult` se verifica la cuenta.

Estas credenciales se usan para autenticar al usuario con Firebase mediante `signInWithCredential`. Si la autenticación es exitosa, se redirige al usuario a la pantalla principal (`MainActivity`); si falla, se muestra un mensaje de error.

```
override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {
    super.onActivityResult(requestCode, resultCode, data)

    if (requestCode == RC_SIGN_IN) {
        val task = GoogleSignIn.getSignedInAccountFromIntent(data)
        try {
            val account = task.getResult(ApiException::class.java)
            if (account != null) {
                firebaseAuthWithGoogle(account)
            }
        } catch (e: ApiException) {
            Log.w(TAG, msg: "Google sign in failed", e)
            Toast.makeText(context: this, text: "Google sign-in failed.", Toast.LENGTH_SHORT).show()
        }
    }
}

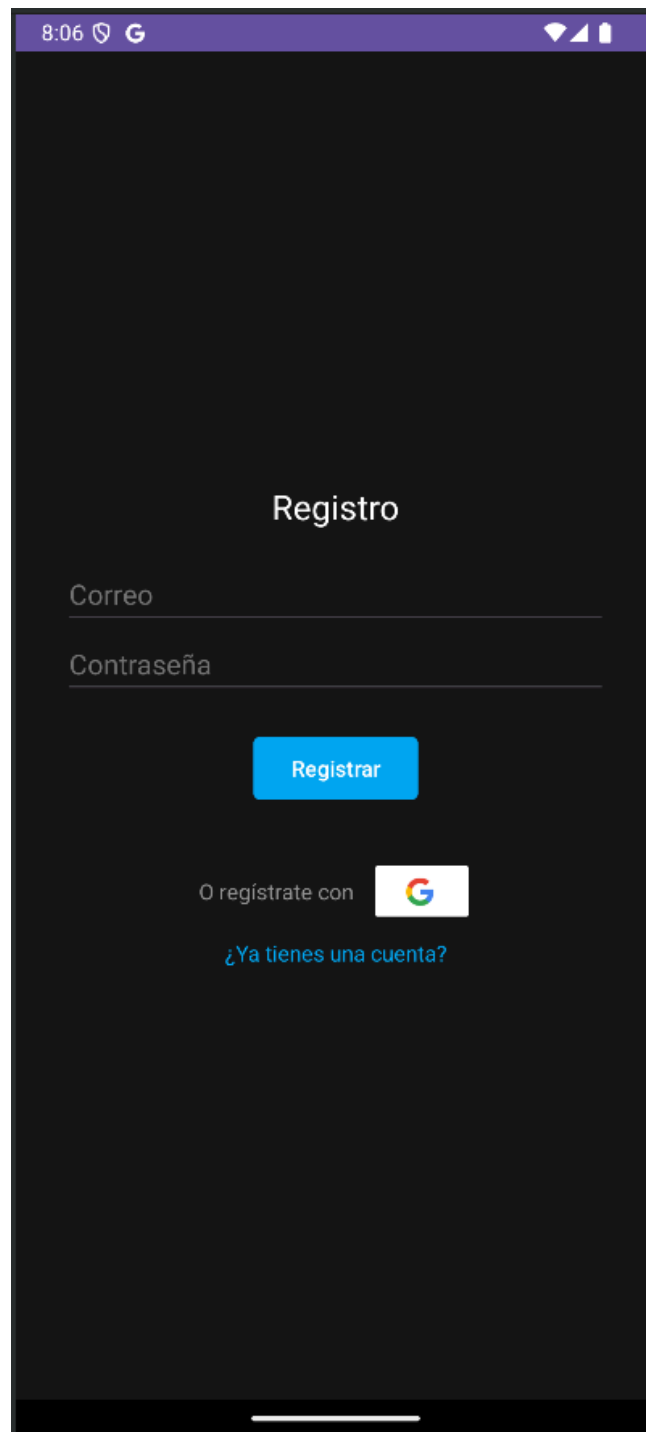
private fun firebaseAuthWithGoogle(account: GoogleSignInAccount) {
    val credential = GoogleAuthProvider.getCredential(account.idToken, null)
    auth.signInWithCredential(credential)
        .addOnCompleteListener(this) { task ->
            if (task.isSuccessful) {
                Log.d(TAG, msg: "signInWithCredential:success")
                startActivity(Intent(packageContext: this, MainActivity::class.java))
                finish()
            } else {
                Log.w(TAG, msg: "signInWithCredential:failure", task.exception)
                Toast.makeText(context: this, text: "Firebase authentication failed.", Toast.LENGTH_SHORT).show()
            }
        }
}
```

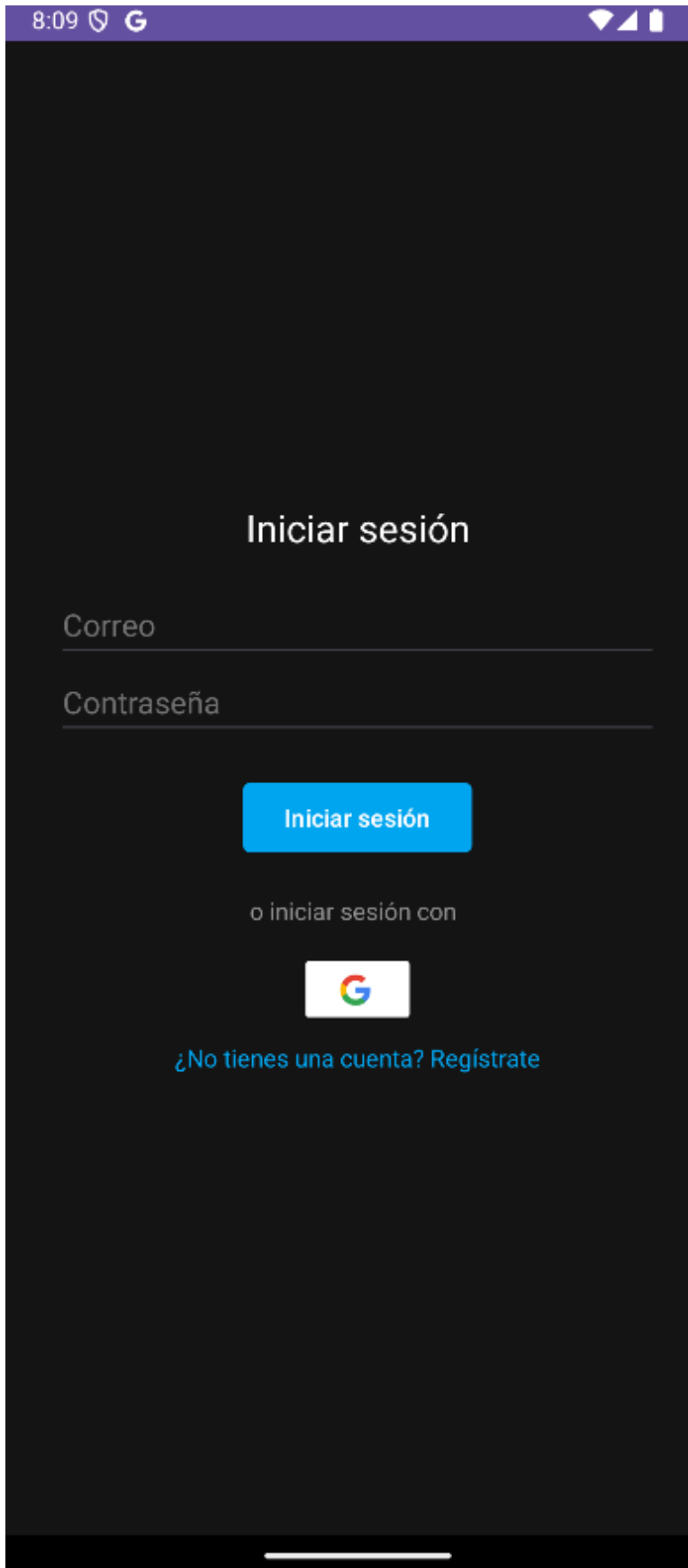
Vistas de la App

Registro

Pantalla principal que se muestra al ejecutar la app, donde los usuarios se pueden registrar a la aplicación, y crear una cuenta mediante correo y contraseña, o mediante la opción de Google.

Una vez creado, se redirige a la ventana Home (ver siguiente apartado).



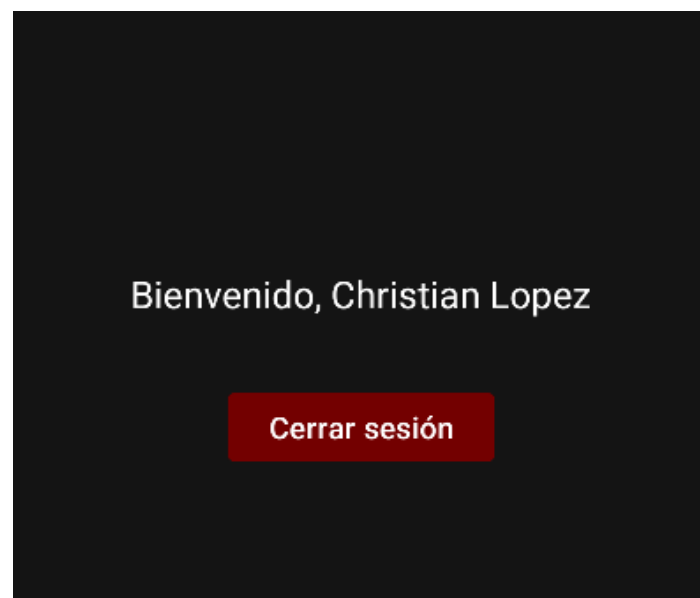


Inicio de sesión

En esta pantalla se muestran dos campos de texto para ingresar el correo y contraseña y el respectivo botón de inicio de sesión.

También, se tiene un TextView para poder ingresar al apartado de Registro, y un botón para iniciar sesión por medio de Google.

Un vez iniciado, se muestra el nombre si es mediante Google, o el correo si es por correo/contraseña.



Mensajes de errores

- Login


Iniciar sesión

Correo

Contraseña

Iniciar sesión

o iniciar sesión con



[¿No tienes una cuenta? Regístrate](#)

Ingresar tu correo electrónico


Iniciar sesión

cuentainexistente@gmail.com

.....

Iniciar sesión

o iniciar sesión con



[¿No tienes una cuenta? Regístrate](#)

Credenciales incorrectas

Iniciar sesión

correo@gmail.com

Contraseña

La contraseña es obligatoria

Iniciar sesión

o iniciar sesión con

Iniciar sesión

correo@gmail

.....

Correo no válido

Iniciar sesión

- Registro

Registro

Correo

Contraseña

Registrar

O regístrate con



¿Ya tienes una cuenta?

Ingresa tu correo electrónico

Registro

correo@gmail.com

Contraseña

Registrar

La contraseña es obligatoria

Registro

correo@

.....

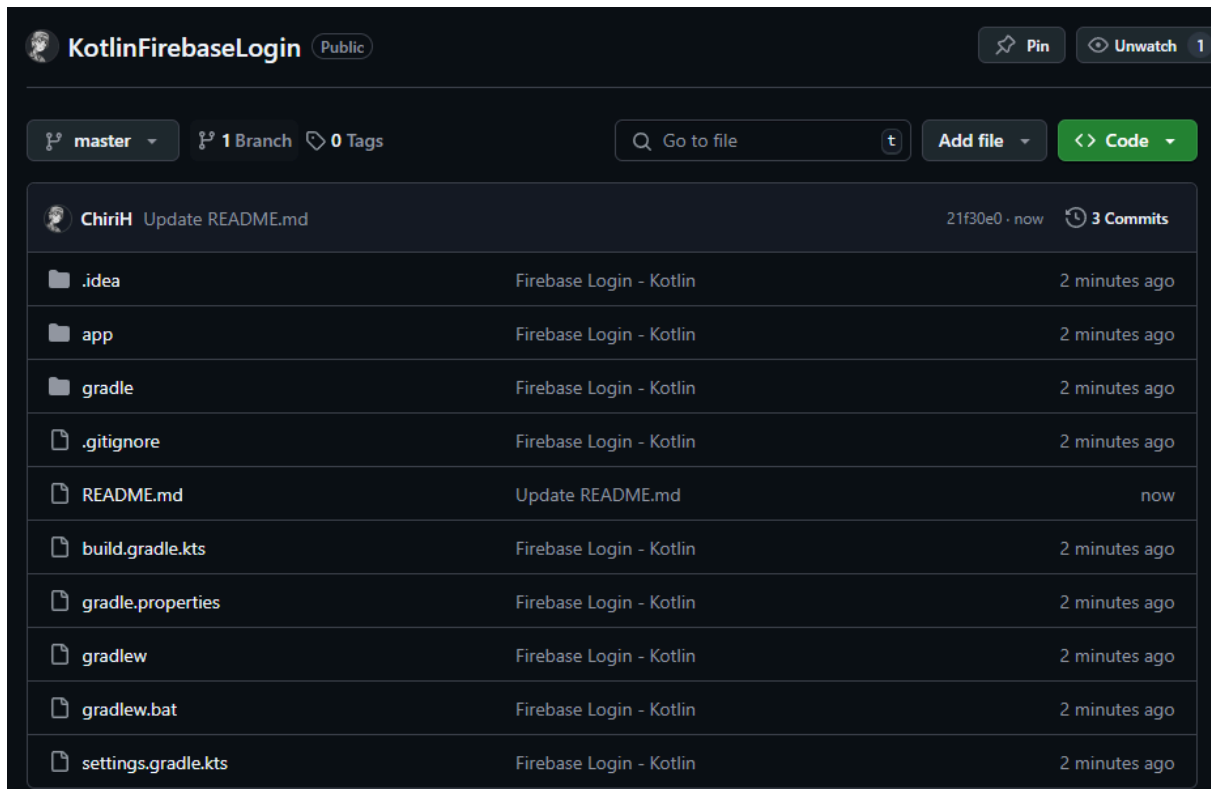
Registrar

Correo no válido

GitHub

Enlace a repositorio

<https://github.com/ChiriH/KotlinFirebaseLogin>



- Video funcionamiento

<https://drive.google.com/file/d/1scnicwIAUIWM2jCEsIRYZY-OPnvPI7Rx/view?usp=sharing>

