

N1-U2 Actividad práctica sumativa - Semana 6

**MÓDULO: TALLER DE APLICACIONES MÓVILES**

**SEMANA: Semana 6**

Docente: IVAN AYALA

Estudiante: Andres Lagos

**Resumen**

Se desarrolló una aplicación móvil Android que calcula la distancia entre el dispositivo del cliente y la bodega utilizando la fórmula de Haversine, aplica reglas de negocio para determinar el costo de despacho y utiliza Firebase Authentication (correo/contraseña y Google) como SSO. Al validar credenciales, la app captura la ubicación GPS y la guarda en Firebase Realtime Database, permitiendo trazabilidad básica del despacho. El proyecto es compatible con Android Lollipop (minSdk 21) y se probó principalmente en Android Oreo y superior. Se documentó la configuración, el proceso de compilación, las pruebas y se publica el código en GitHub, con respaldo en OneDrive.

Índice

[Introducción 4](#_Toc209301178)

[Desarrollo 5](#_Toc209301179)

[Conclusión 12](#_Toc209301180)

[Bibliografía 13](#_Toc209301181)

# **Introducción**

El documento describe el proceso de construcción de una app de distribución de alimentos que incluye cálculo de distancia, costo de envío y autenticación con Google y correo/clave. Se busca integrar conceptos de geolocalización, reglas de negocio y servicios en la nube (Firebase) en una solución funcional y reproducible, apta para equipos con Android Lollipop (administrador) y versiones más recientes (clientes).

# **Desarrollo**

**a) Pantalla Login con autenticación Firebase (correo/contraseña)**

Diseñe una **LoginActivity** en Jetpack Compose con:

Campo correo y contraseña (con botón de mostrar/ocultar).

Botones **Entrar**, **Crear cuenta** y **Continuar con Google** (opcional).

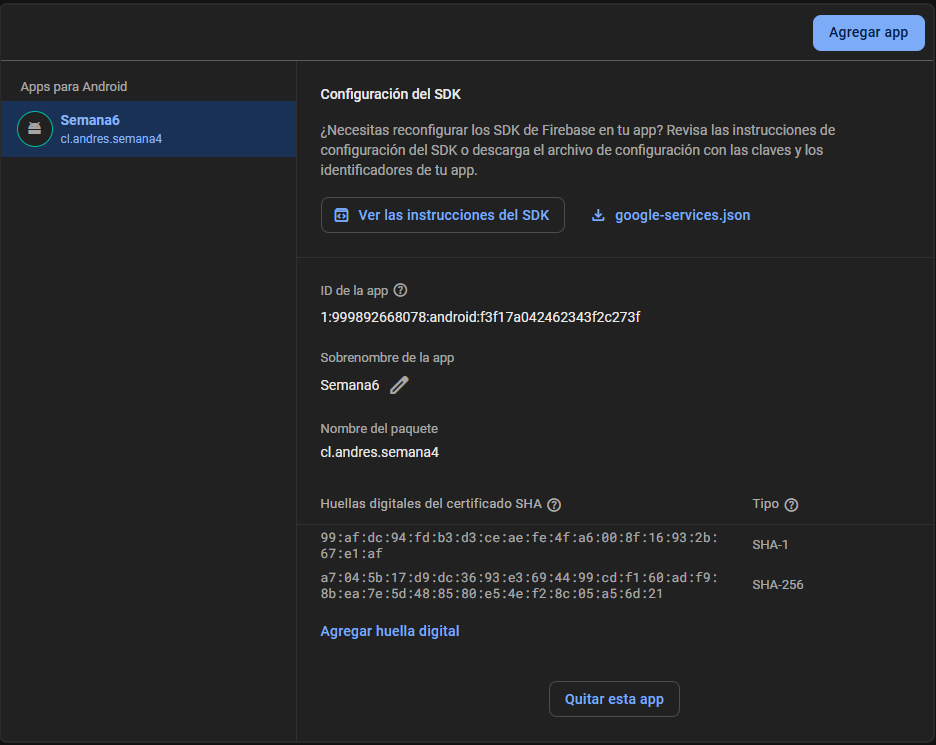
Validaciones: no se permite invocar Firebase si los campos están vacíos; se muestra un **snackbar/toast** indicando que deben completarse (evita el crash por parámetros nulos).

Flujo implementado:

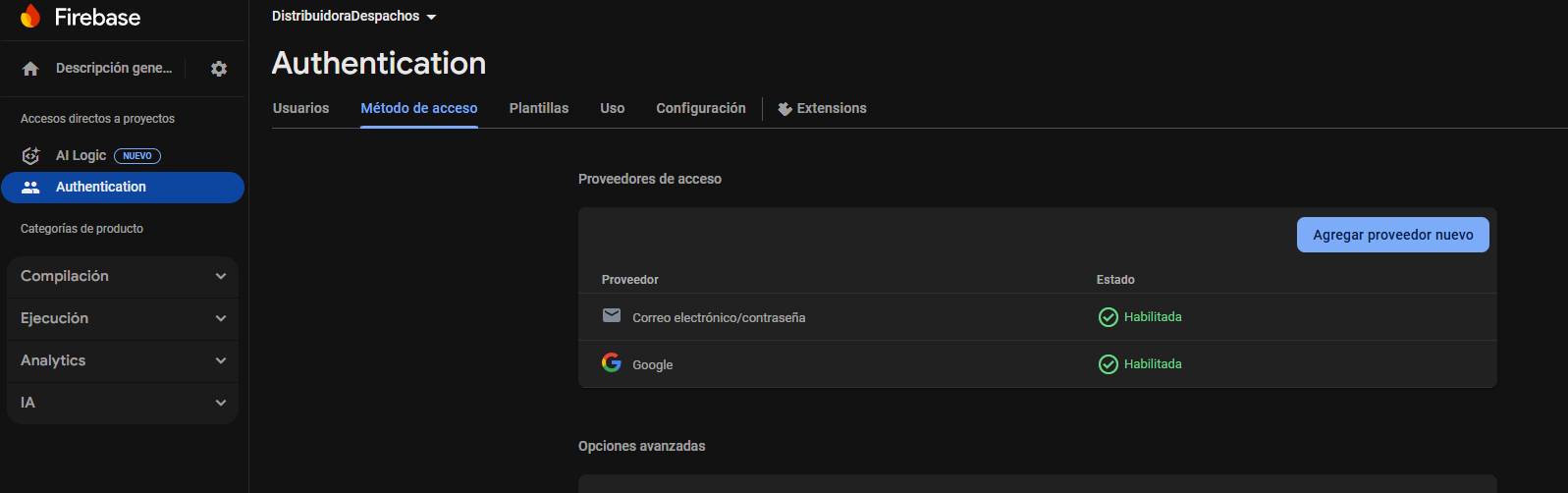
**Crear cuenta:** FirebaseAuth.getInstance().createUserWithEmailAndPassword(email, pass)  
si es exitoso, se deja logueado al usuario y se navega a **MenuActivity**.

**Entrar:** FirebaseAuth.getInstance().signInWithEmailAndPassword(email, pass)  
 si las credenciales son correctas, se abre **MenuActivity.**

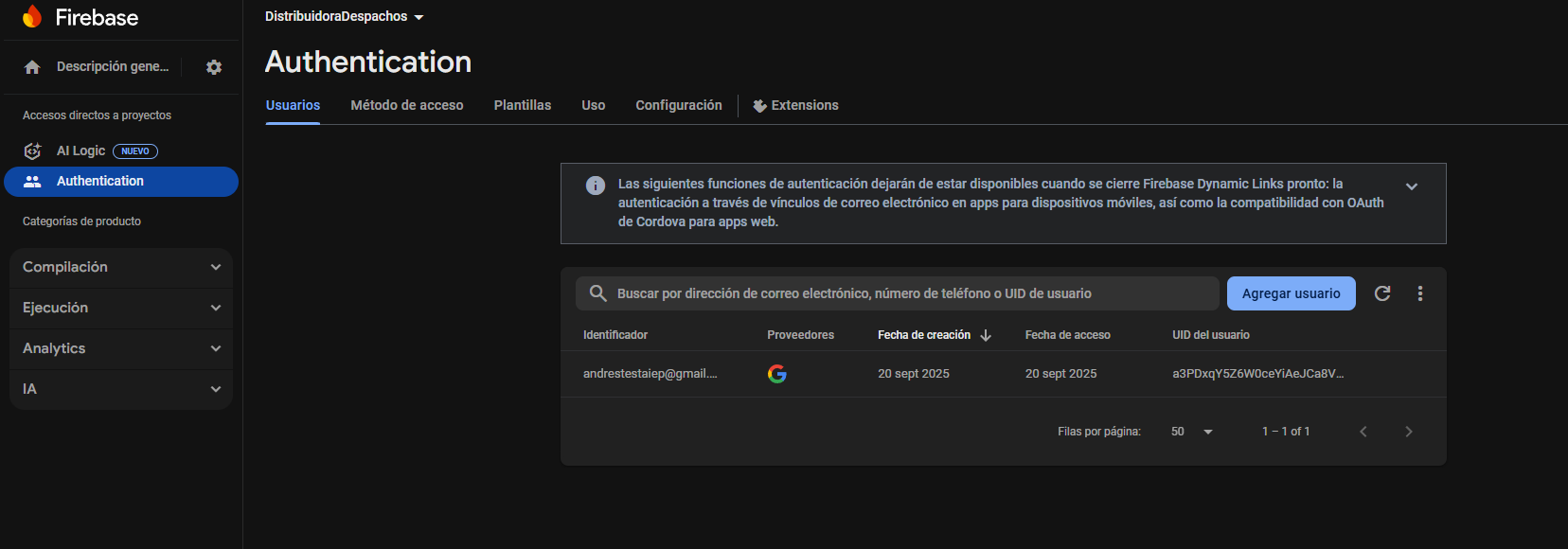
**Evidencias firebase:  
firebase google-service sha-1 sha-256:**



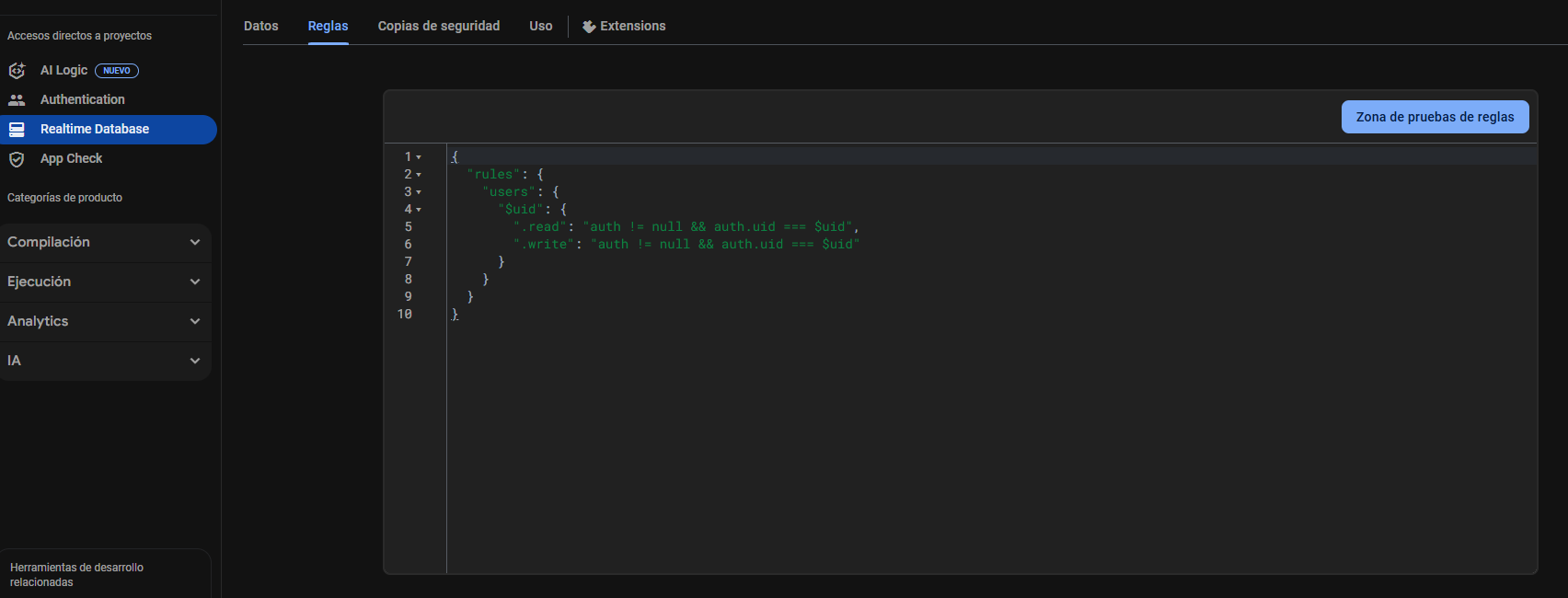
**Metodo de acceso habilitado:**



**Authentication:**



**Reglas realtime database:**



**b) Guardar posición GPS en Realtime Database si el login es válido**

En MenuActivity implemente el guardado de la última ubicación del usuario autenticado:

Al iniciar la actividad (onStart()), pedimos permisos ACCESS\_FINE\_LOCATION/ACCESS\_COARSE\_LOCATION.  
Si están concedidos → se llama a capturarYGuardarUbicacion().  
Se obtiene la ubicación con FusedLocationProviderClient:

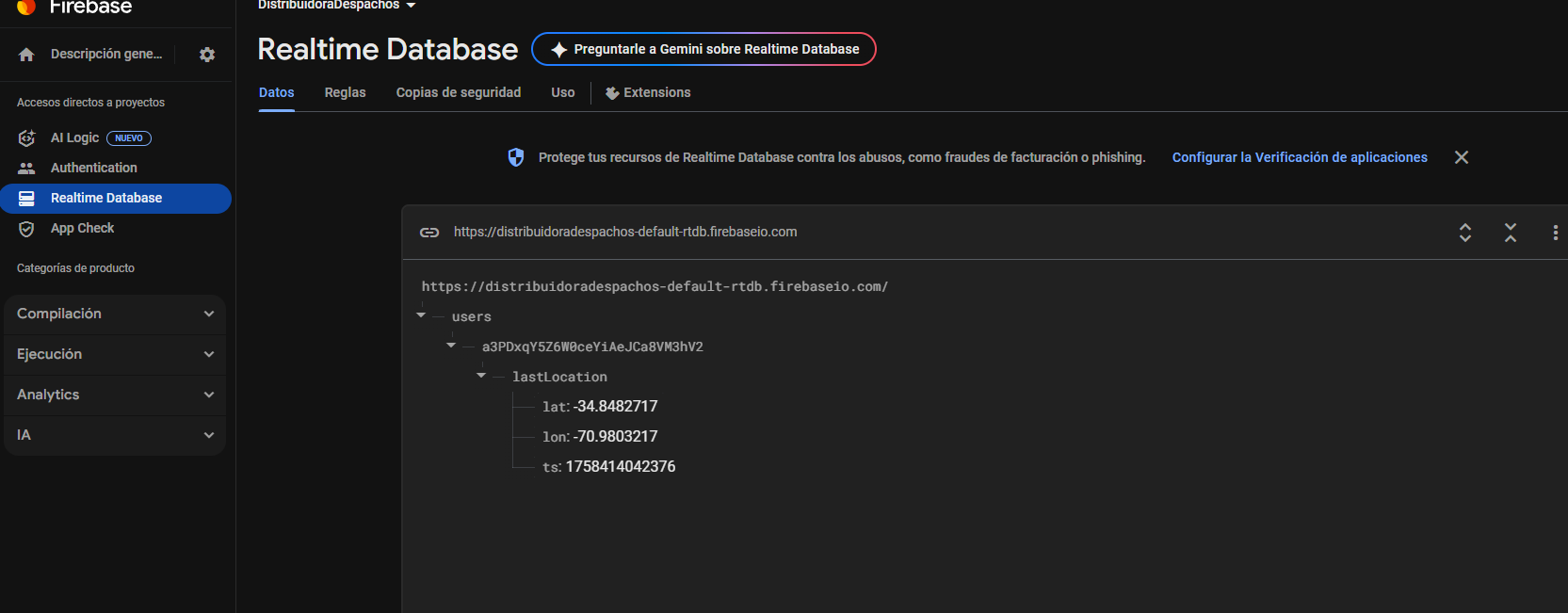
used.getCurrentLocation(Priority.*PRIORITY\_HIGH\_ACCURACY*, null)

Si hay ubicación y uid != null, se escribe en **Realtime Database**:

FirebaseDatabase.getInstance()  
 .getReference("users")  
 .child(uid)  
 .child("lastLocation")  
  
val data = *mapOf*(  
 "lat" *to* loc.*latitude*,  
 "lon" *to* loc.*longitude*,  
 "ts" *to* ServerValue.*TIMESTAMP*)

**Evidencia:**

**Captura de** **Realtime Database:**



**c) Instalación y prueba en dispositivo/emulador**

**Emulador recomendado:** AVD con **Google APIs** (ej. Pixel 6, API 30+ con Google Play).

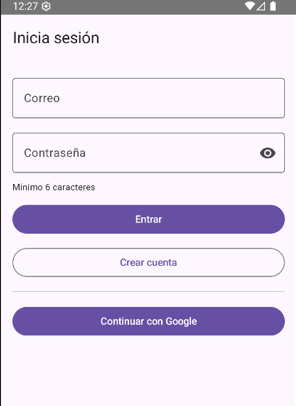
Ingresar a Play Store del emulador (confirma que tiene servicios de Google).  
Ajustar ubicación en **Extended controls → Location** para simular coordenadas.

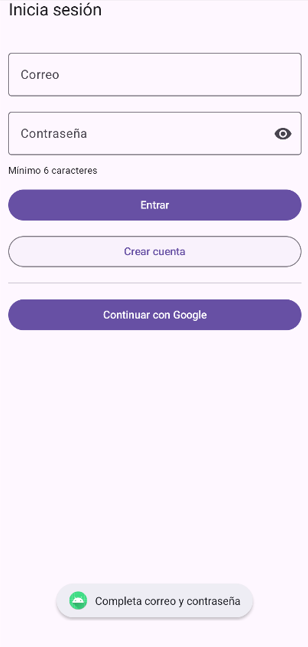
Compilaciónyejecución**:**

Sync Project with Gradle Files.  
Runseleccionando el dispositivo/emulador.

Probar:  
Login con correo/clave válidos.  
Ver que, al abrir MenuActivity, se escribe la ubicación en la DB.  
Repetir negando permisos para verificar el toast de rechazo.

**Evidencia:**

**Login app:**   
  


**Crear cuenta sin completar campos:**  
  


**Menú app:**



# **Conclusión**

El proyecto integró de forma práctica Android con Kotlin, Jetpack Compose y Firebase. Se implementó login con correo/contraseña, registro de la ubicación del usuario en Realtime Database y la lógica de negocio (radianes, Haversine y cálculo de despacho), manteniendo compatibilidad con minSdk 21. Además, el uso de GitHub con issues y ramas por feature mejoró la organización y el trabajo colaborativo. En conjunto, se reforzaron competencias técnicas y de gestión aplicables a proyectos móviles reales.

# **Bibliografía**

Firebase. (s. f.). **Autenticación con correo y contraseña en Android**. Recuperado el 20 de septiembre de 2025, de https://firebase.google.com/docs/auth/android/password-auth  
Firebase. (s. f.). **Configurar Realtime Database para Android**. Recuperado el 20 de septiembre de 2025, de https://firebase.google.com/docs/database/android/start  
Firebase. (s. f.). **Reglas de seguridad de Realtime Database**. Recuperado el 20 de septiembre de 2025, de https://firebase.google.com/docs/rules