



Ciclo 01-2025

FORO 2

Diseño y Programación de Software Multiplataforma

[REPOSITORIO GITHUB](#)

Integrantes

Integrantes	Carnet
Victor Amilcar Elías Peña	EP171613
Melissa Vanina López Peña	LP223029
Oscar Alexander Quintanilla Rodriguez	QR200363
David Ernesto Ramos Vásquez	RV230544
Oscar Dave Guerra Pacheco	GP200006

Fecha de entrega : 5 de marzo de 2025.

ÍNDICE

Opción 1.....	3
Características.....	3
Ventajas.....	4
Desventajas.....	4
Consideraciones importantes.....	4
Conclusión de opción 1.....	5
Opción 2.....	5
Características.....	5
Ventajas.....	6
Desventajas.....	6
Consideraciones importantes.....	7
Conclusión de opción 2.....	7
Configuración de Firebase.....	8
Registrar la Aplicación Web.....	9
Configurar Google Cloud Console para OAuth.....	11
Crear usuarios de prueba en Email/Password.....	13
Paso 1 : instalación de dependencias.....	14
Paso 2 : Configuración de Firebase en el proyecto.....	14
Paso 3: Configuración de la pantalla de carga.....	15
1. Importaciones.....	15
2. Configurar el código para que funcione en versiones Móviles.....	16
3. Manejo de la respuesta de Google.....	17
4. Flujo para la autenticación.....	17
5. Vista a renderizar.....	19

Opciones de Autenticación en React Native

Opción 1

Características

La autenticación mediante correo electrónico y contraseña es una de las opciones más básicas y ampliamente utilizadas en Firebase. Esta opción permite que los usuarios creen cuentas utilizando su dirección de email, lo cual es familiar para la mayoría de personas. Es especialmente útil en aplicaciones que requieren una relación más directa con el usuario, como sistemas de suscripción, plataformas educativas o servicios que envían información personalizada.

Entre sus principales características se encuentra la posibilidad de implementar fácilmente mecanismos de recuperación de cuenta mediante verificación por correo electrónico, brindando mayor control al desarrollador sobre la gestión de usuarios. Firebase proporciona una API sencilla para registrar nuevos usuarios, iniciar sesión, cerrar sesión, y verificar el estado de autenticación.

Además, este método permite agregar niveles adicionales de seguridad, como verificación de email o autenticación de dos factores, si se desea reforzar la protección. Aunque requiere que el sistema gestione la seguridad de las contraseñas, Firebase ofrece buenas prácticas y herramientas integradas para facilitar este proceso.

Ventajas

Opción de Autenticación	Ventajas
Correo Electrónico	Fácil de implementar, común entre usuarios
Google	Rápido, seguro, sin recordar contraseñas
Facebook	Ideal para apps sociales, acceso a perfil
Teléfono (SMS)	Alta conveniencia, no requiere email

Desventajas

Opción de Autenticación	Desventajas
Correo Electrónico	Debe manejarse seguridad de contraseñas
Google	Configuración adicional en Google Cloud
Facebook	Dependencia de Facebook, requiere permisos
Teléfono (SMS)	Límites de uso, posibles fallos de SMS

Consideraciones importantes

Opción de Autenticación	Consideraciones
Correo Electrónico	Implementar recuperación/verificación de email
Google	Requiere SHA-1 y configuración OAuth
Facebook	Registrar app en Facebook Developers
Teléfono (SMS)	Verificar soporte de país y operador

Conclusión de opción 1

Firestore ofrece una amplia variedad de métodos de autenticación que permiten adaptar la seguridad y experiencia del usuario según el tipo de aplicación. Las opciones como Google y Teléfono son muy convenientes para los usuarios, mientras que el método de correo electrónico ofrece mayor control. Cada método implica ventajas y desafíos técnicos específicos, por lo que la elección correcta dependerá del público objetivo, la facilidad de implementación y los requisitos de seguridad. Analizar estas variables es clave para tomar decisiones informadas al momento de diseñar el sistema de autenticación de una app.

Opción 2

Características

La autenticación con Google en Firestore permite a los usuarios iniciar sesión utilizando su cuenta de Google de forma rápida y segura. Es una opción moderna y muy utilizada que mejora la experiencia del usuario al evitar la necesidad de recordar contraseñas, reduciendo así la fricción en el proceso de ingreso.

Este método utiliza el sistema OAuth 2.0 para verificar la identidad del usuario, lo que proporciona una capa adicional de seguridad. Para su integración, es necesario habilitar el proveedor de Google en Firestore Authentication y configurar correctamente los datos del proyecto en la consola de Google Cloud, incluyendo la clave SHA-1 y la pantalla de consentimiento OAuth.

Una de sus mayores ventajas es la confianza y familiaridad que los usuarios tienen con Google, lo que puede aumentar las tasas de registro e inicio de sesión. Además, Firestore permite combinar este método con Firestore para extender el perfil del usuario, y se puede complementar con verificación adicional si se requiere un mayor nivel de seguridad.

Ventajas

Ventajas
<p>Autenticación con Correo Electrónico:</p> <ul style="list-style-type: none">• Opción común y reconocida• Sin depender de terceros• Fácil de integrar con verificación y recuperación

Desventajas

Desventajas
<p>Autenticación con Correo Electrónico:</p> <ul style="list-style-type: none">• Mayor fricción para el usuario• Requiere recordar contraseña y cuidar seguridad de datos

Consideraciones importantes

Consideraciones
<p>Autenticación con Correo Electrónico:</p> <ul style="list-style-type: none">• Habilitar en consola firebase• Activar verificación de email• Manejar errores comunes y extender perfil en firestore

Conclusión de opción 2

La autenticación por correo electrónico es una de las formas más tradicionales y universales de acceso. Aunque presenta mayor fricción para el usuario en comparación con métodos sociales, su implementación es sencilla y no depende de plataformas externas. Además, permite establecer flujos robustos como verificación de cuenta, recuperación de contraseña y control total del registro de usuarios. Por ello, sigue siendo una opción sólida, especialmente en aplicaciones que requieren mayor control de acceso o no dependen de la identidad digital de redes sociales.

Autenticación por Correo Electrónico

Pasos para implementar la autenticación a través del correo electrónico (Gmail) en Firebase

Configuración de Firebase

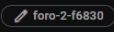
1.1 Crear proyecto en Firebase

- Ir a <https://console.firebase.google.com/>
- Dar click en “Agregar Proyecto”
- Asignar un nombre al proyecto y aceptar los términos y condiciones
- Dar click en Agregar

× Crear un proyecto

Comencemos con el nombre de tu proyecto[®]

Nombre del proyecto
Foro 2



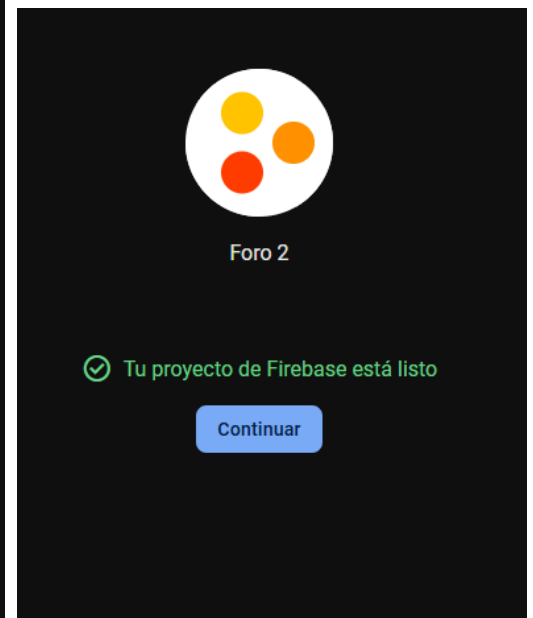
☒ Acepto las [Condiciones de Firebase](#)

☒ Confirmando que usaré Firebase exclusivamente para fines relacionados con mi trabajo, empresa, oficina o profesión.

☒ Únete al [Programa para desarrolladores de Google](#) para enriquecer tu experiencia como desarrollador con acceso a asistencia de IA, recursos de aprendizaje, insignias de perfil y mucho más.

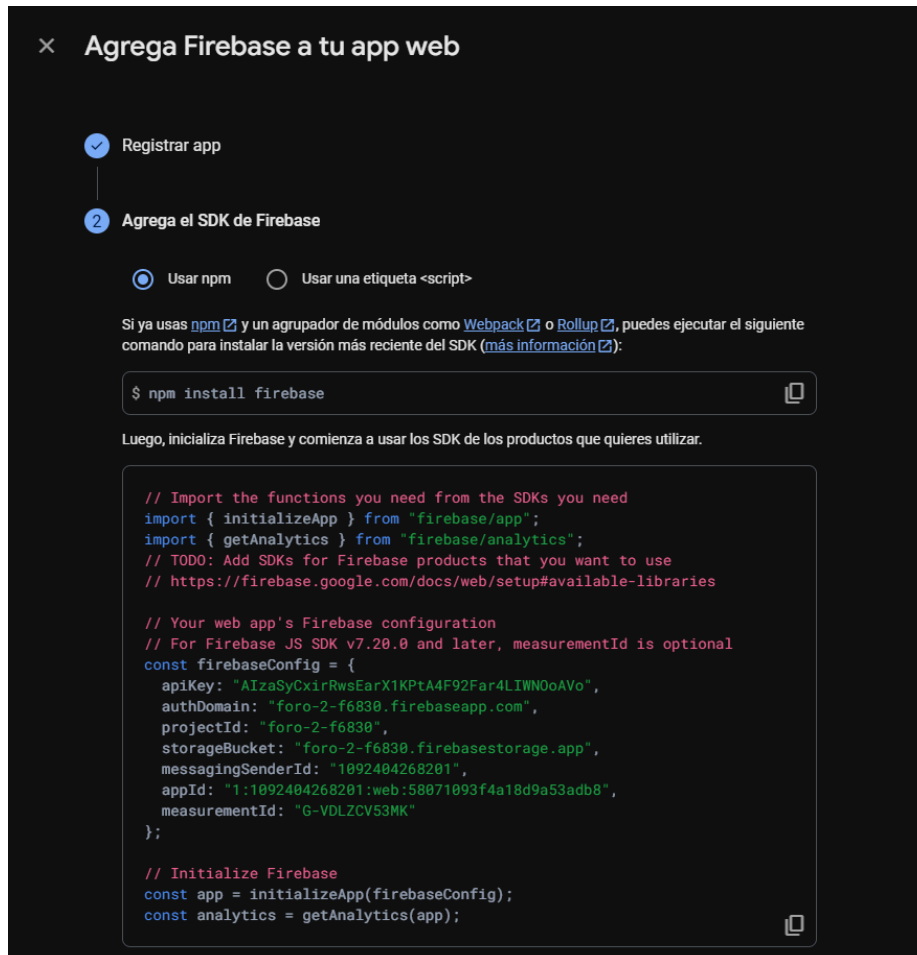
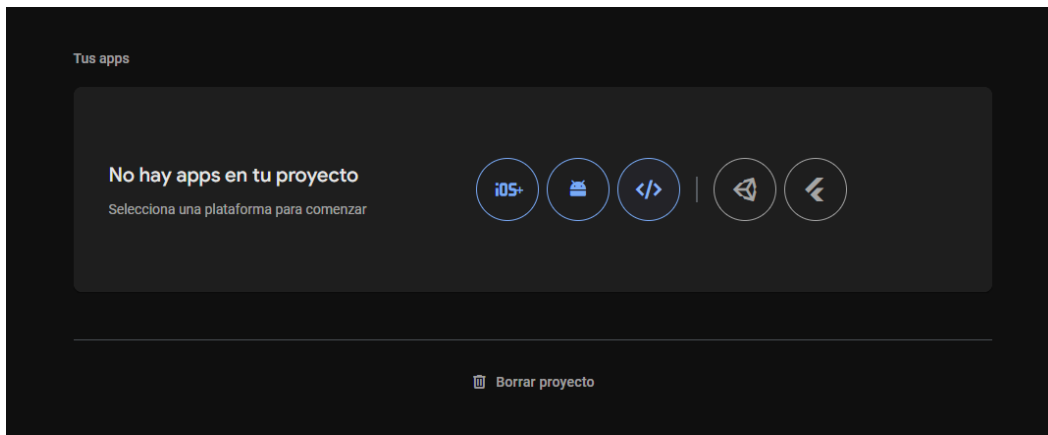
¿Ya tienes un proyecto de Google Cloud?
[Agregar Firebase al proyecto de Google Cloud](#)

Continuar



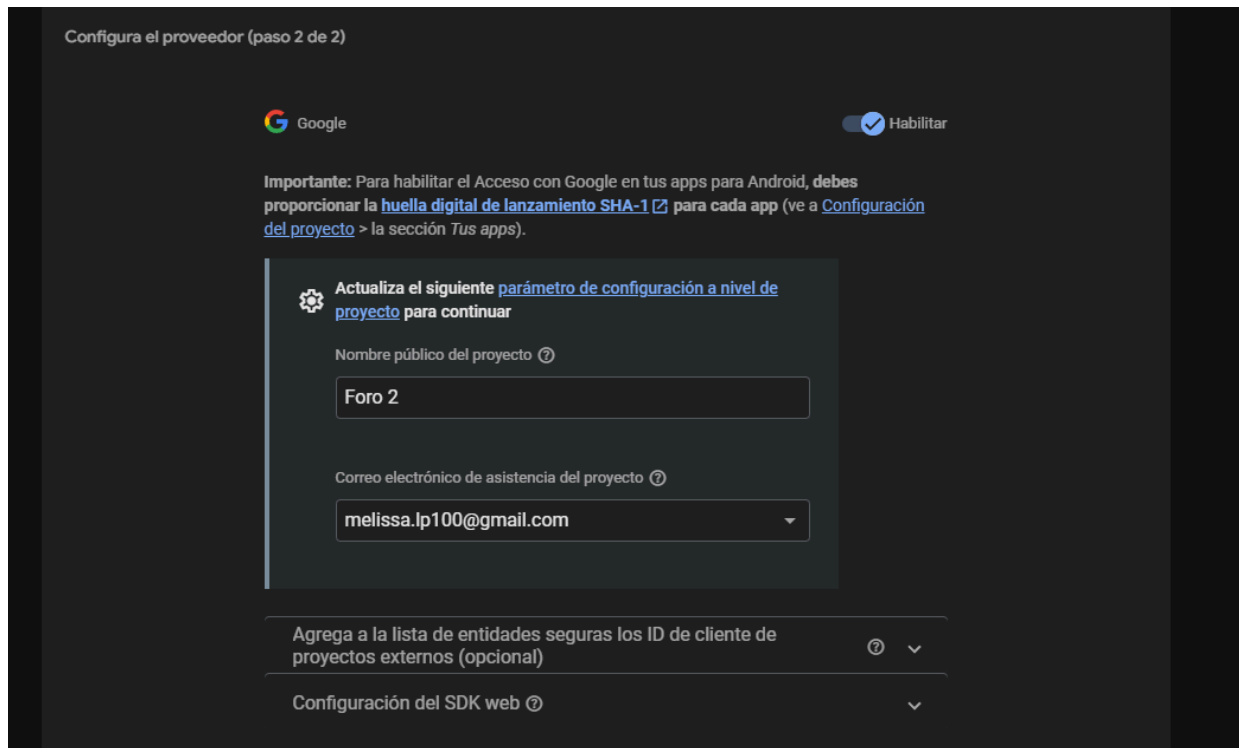
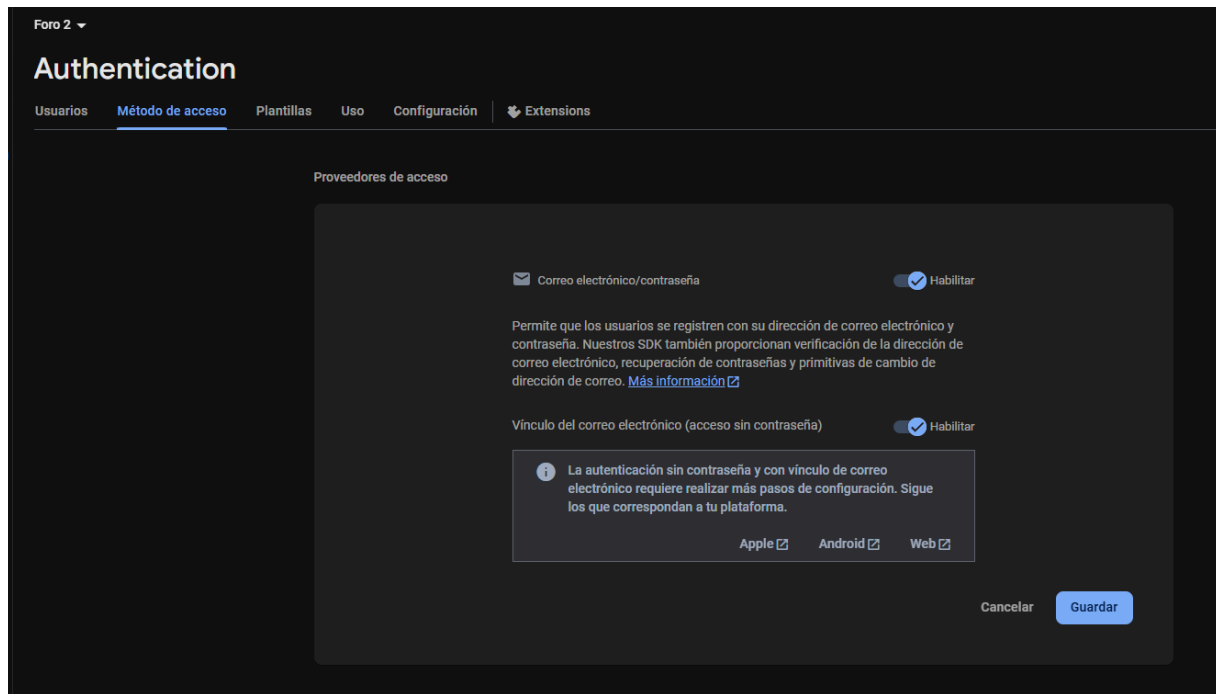
Registrar la Aplicación Web

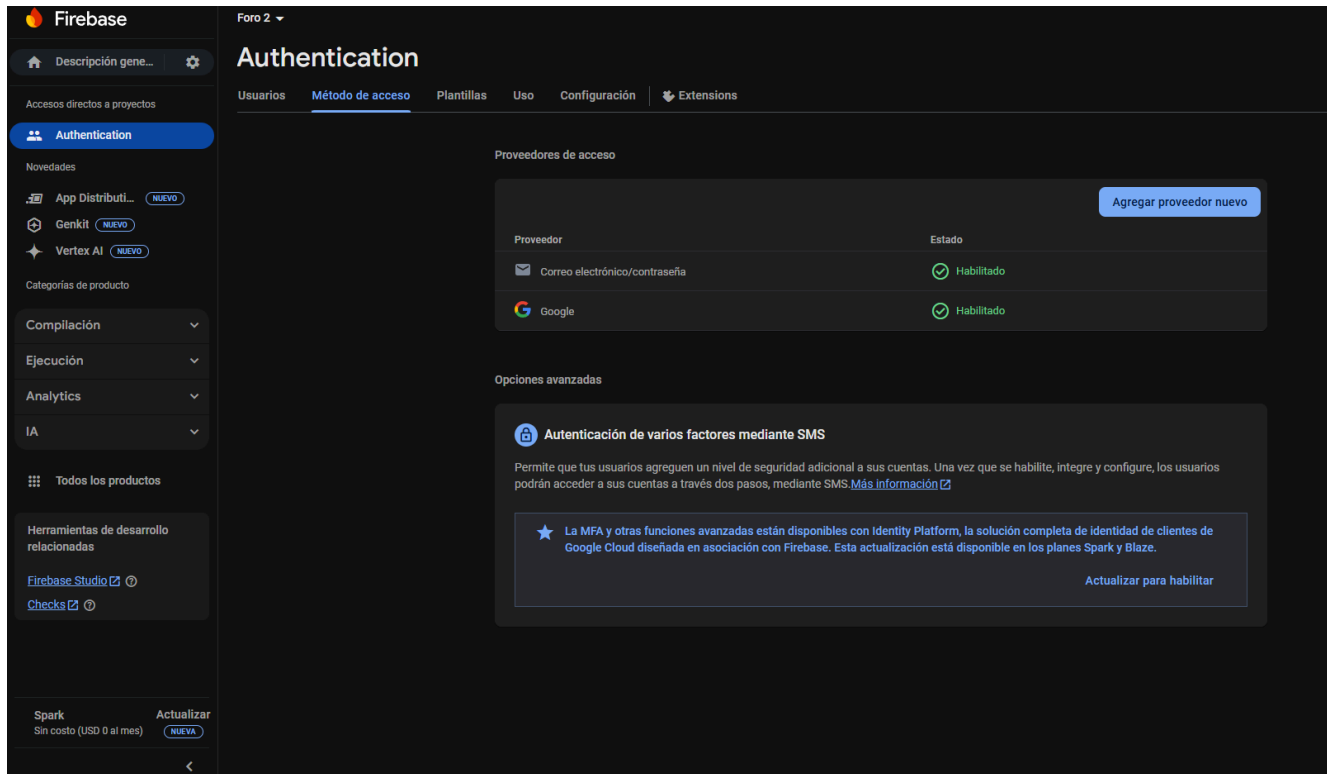
- Ir a Project Settings-> Your apps
- Seleccionar el icono </>
- Asignar nombre y registrar la App
- Copiar el fragmento de configuración que da Firebase para utilizarlo posteriormente en src/firebase/config.



1.3 Habilitar métodos de autenticación

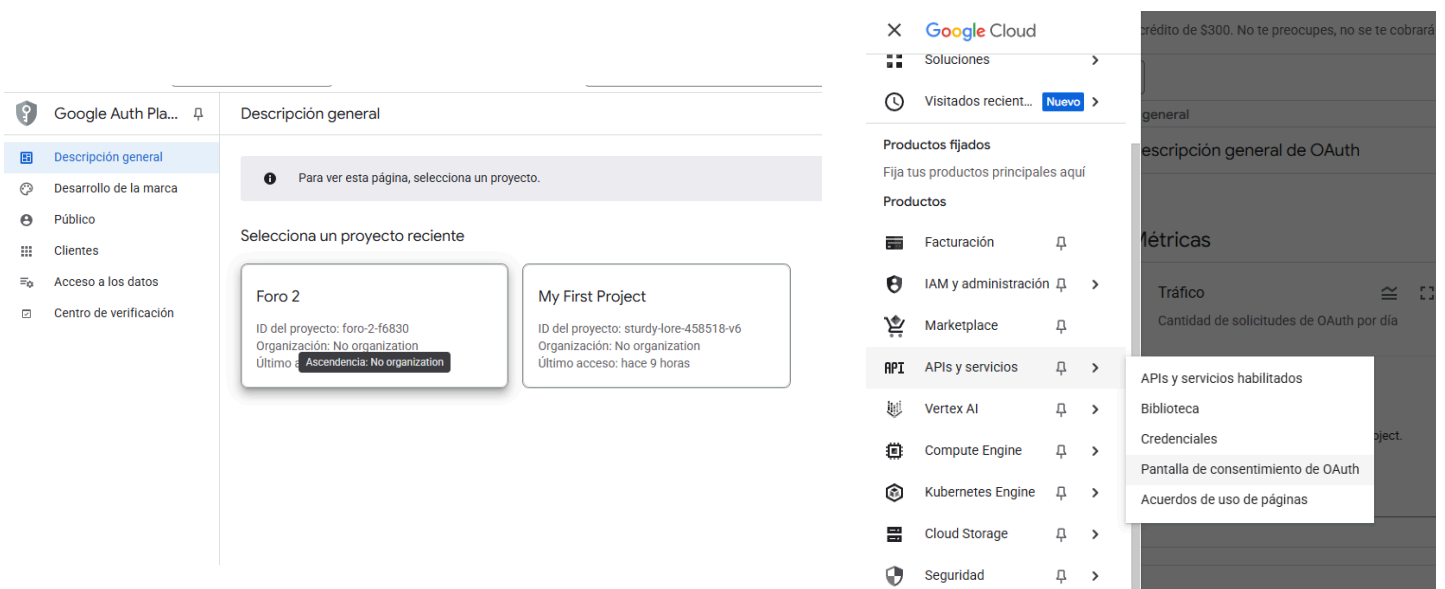
- Ir a la opción Authentication del menú
- Activar la opción Email/Password
- Activar la opción Google





Configurar Google Cloud Console para OAuth

- Ir al Google Cloud Console y dirigirse al proyecto (Foro 2)
- Buscar la opción de APIs y servicios y dar click en Pantalla de consentimiento de OAuth
- En Desarrollo de la marca poner el nombre de la aplicación y correo de soporte



Google Auth Platform / Desarrollo de la marca

Descripción general

Desarrollo de la marca

Público

Cientes

Acceso a los datos

Centro de verificación

Información de marca

Información de la aplicación

Esta información aparece en la pantalla de consentimiento y permite que los usuarios finales sepan quién eres y cómo comunicarse contigo

Nombre de la aplicación *

Foro 2

El nombre de la aplicación que solicita el consentimiento

Correo electrónico de asistencia del usuario *

melissa.lp100@gmail.com

Para que los usuarios se comuniquen contigo si tienen preguntas sobre su consentimiento. [Más información](#)

Logotipo de la app

Este es tu logotipo. Ayuda a que las personas reconozcan tu app y aparece en la pantalla de consentimiento de OAuth.

Después de subir un logotipo, deberás enviar tu app para verificarla, a menos que esté configurada solo para uso interno o tenga el estado de publicación "Prueba". [Más información](#)

- Ir a la opción de APIs y servicios->Credenciales
- Dar click en editar cliente de OAuth
- Agregar <http://localhost:5000>

Google Cloud

Foro 2

Buscar (/) recursos, documentos, prod

Google Auth Platform / Clientes / Cliente: 1092404268201-37s3b3kpjjcmkipo01jd7tnbndiaed6.apps.googleusercontent.com

Descripción general

Desarrollo de la marca

Público

Cientes

Acceso a los datos

Centro de verificación

← ID de cliente para Aplicación web [Borrar](#)

Nombre *

Web client (auto created by Google Service)

El nombre de tu cliente de OAuth 2.0. Este nombre solo se usa para identificar al cliente en la consola y no se mostrará a los usuarios finales.

Los dominios de los URI que agregues a continuación se incorporarán automáticamente a tu [pantalla de consentimiento de OAuth](#) como [dominios autorizados](#).

Orígenes autorizados de JavaScript

Para usar con solicitudes de un navegador

URI 1 *

http://localhost

URI 2 *

http://localhost:5000

URI 3 *

https://foro-2-f6830.firebaseio.com

+ Agregar URI

Crear usuarios de prueba en Email/Password



- Ir a Authentication en Firebase
- Dar click en Users
- Dar click en Add User
- Crear un usuario con email y contraseña para las pruebas


Foro 2 ▾

Authentication

[Usuarios](#) [Método de acceso](#) [Plantillas](#) [Uso](#) [Configuración](#) [Extensions](#)

i Las siguientes funciones de autenticación dejarán de estar disponibles cuando se cierre Firebase Dynamic Links el 25 de agosto de 2025: la autenticación a través de vínculos de correo electrónico en apps para dispositivos móviles, así como la compatibilidad con OAuth de Cordova para apps web. ▾

[Agregar usuario](#)  

Identificador	Proveedores	Fecha de creación ▾	Fecha de acceso	UID de usuario
team_member@gmail.c...		1 may 2025	1 may 2025	IWukzuszk1d4nTBpnMjeliCFK...

Filas por página: 50 ▾ 1 – 1 of 1 < >

Explicación del código para generar el login con Gmail

Paso 1 : instalación de dependencias

Necesitaremos instalar NPM INSTALL FIREBASE , esta librería es necesaria ya que incluye todas las dependencias para utilizar la autenticación de gmail

```
PS C:\Users\vmeli\Documents\UDB\CICLO 01-2025\DPS\Foro-2-DPS> npm install firebase

added 86 packages in 26s

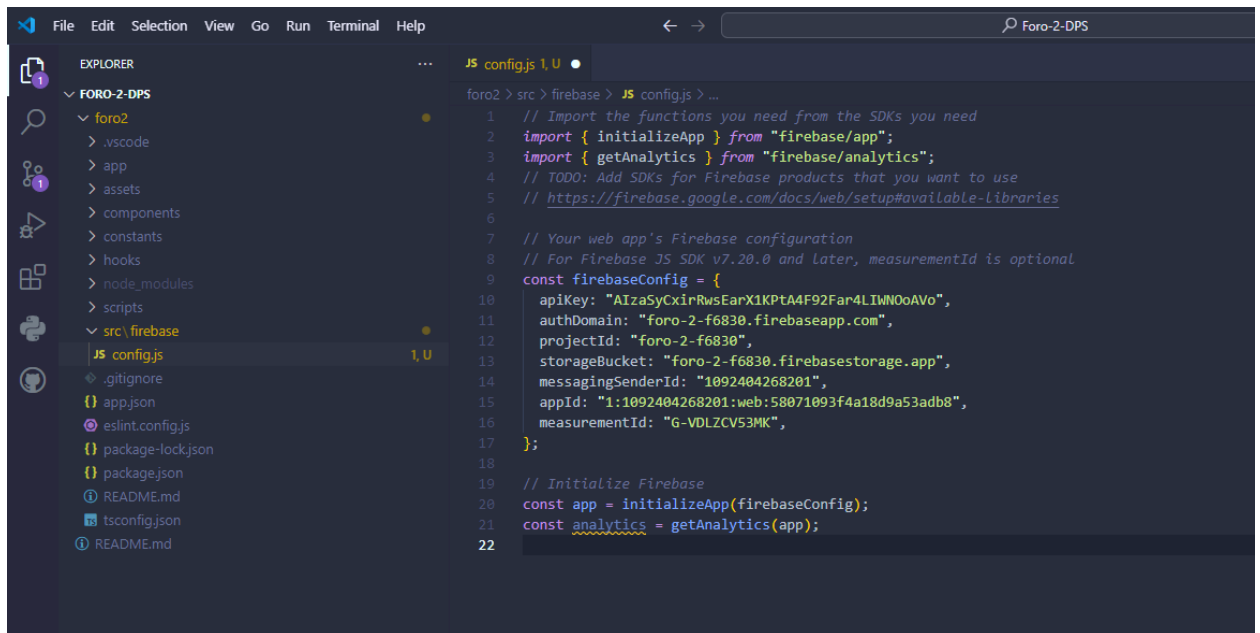
3 packages are looking for funding
  run `npm fund` for details
PS C:\Users\vmeli\Documents\UDB\CICLO 01-2025\DPS\Foro-2-DPS> |
```

Paso 2 : Configuración de Firebase en el proyecto

- **Config.js**

1. Primeramente importamos initializeApp y getAnalytics , estos son los encargados de recibir la configuración y levantar las instancias del Firebase , este es el que conecta nuestro código con los servers
2. Luego necesitamos crear el objeto de configuración del proyecto , en el tenemos que agregar los siguientes parámetros para que funcione de manera correcta
 - a. **apiKey**, esta será nuestra llave pública la cual usaremos para poder identificar el proyecto.
 - b. **auth Domain**, Este es el dominio para la autenticación.
 - c. **project id** , es el identificador que tenemos en nuestro Firebase.
 - d. **storageBucket** , este nos da la ubicación de donde se almacena la info en nuestra nube.
 - e. **appId** , el identificador de la app a la que estamos aplicando el identificador.

3. Lo único que nos faltara es inicializar el firebase , necesitamos devolver el objeto app con la configuración.



```
1 // Import the functions you need from the SDKs you need
2 import { initializeApp } from "firebase/app";
3 import { getAnalytics } from "firebase/analytics";
4 // TODO: Add SDKs for Firebase products that you want to use
5 // https://firebase.google.com/docs/web/setup#available-libraries
6
7 // Your web app's Firebase configuration
8 // For Firebase JS SDK v7.20.0 and later, measurementId is optional
9 const firebaseConfig = {
10   apiKey: "AIzaSyCxrRwsEarX1KPtA4F92Far4LIWNoOAv0",
11   authDomain: "foro-2-f6830.firebaseio.com",
12   projectId: "foro-2-f6830",
13   storageBucket: "foro-2-f6830.firebaseio.com",
14   messagingSenderId: "1092404268201",
15   appId: "1:1092404268201:web:58071093f4a18d9a53adb8",
16   measurementId: "G-VDLZCV53MK",
17 };
18
19 // Initialize Firebase
20 const app = initializeApp(firebaseConfig);
21 const analytics = getAnalytics(app);
22
```

Paso 3: Configuración de la pantalla de carga

- LoginScreen.js

1. Importaciones

- React, useState y useEffect para el manejo de estado y efectos secundarios.
- Componentes de UI de React Native (View, TextInput, TouchableOpacity, etc.).
- useRouter de expo-router para la navegación.
- auth desde src/firebase/config y funciones de Firebase Auth (signInWithEmailAndPassword, GoogleAuthProvider, signInWithPopup, signInWithCredential).

-
- `Google.useAuthRequest` de `expo-auth-session/providers/google` para el flujo OAuth en móvil.

2. Configurar el código para que funcione en versiones Móviles

Para integrar el inicio con Google en dispositivos móviles (Expo), utilizaremos el hook.

- a. **request:** configuración de la petición OAuth.
- b. **response:** será el objeto que, al completarse con éxito, incluye el token de acceso.
- c. **promptAsync():** función para iniciar el flujo de login de Google en dispositivos móviles.

```
export default function LoginScreen() {
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const [errorMessage, setErrorMessage] = useState("");
  const router = useRouter();

  const [request, response, promptAsync] = Google.useAuthRequest({
    clientId:
      "1092404268201-37s3b3kpjjlcmkipo01jd7tnbndiaed6.apps.googleusercontent.com",
  });
}
```

3. Manejo de la respuesta de Google

En este caso, este código sirve para captar la respuesta de Google y convertirla en credencial de Firebase.

```
useEffect(() => {
  if (response?.type === "success" && response.authentication) {
    const { idToken } = response.authentication;
    const credential = GoogleAuthProvider.credential(idToken);
    signInWithCredential(auth, credential)
      .then(() => {
        setErrorMessage("");
        router.replace("/home");
      })
      .catch((err) => setErrorMessage("Error con Google: " + err.message));
  }
}, [response, router]);
```

- Cuando response.type es "success", extraemos el idToken.
- Con GoogleAuthProvider.credential(idToken) creamos la credencial para Firebase.
- Finalmente, signInWithCredential completa el login y nos redirige o muestra un mensaje de error.

4. Flujo para la autenticación

Primeramente, esta función detecta en qué plataforma se está ejecutando la aplicación. Si estamos en un navegador web (**Platform.OS === "web"**), creamos un nuevo proveedor de Google (`new GoogleAuthProvider()`) y llamamos a **signInWithPopup(auth, provider)** para abrir la ventana emergente de Google.

Cuando el usuario completa el flujo correctamente, borramos cualquier mensaje de error con **setErrorMessage("")** y redirigimos a la ruta `/home` mediante `router.replace("/home")`. En caso de que se produzca un error durante este proceso, lo capturamos y mostramos su descripción al usuario.

Por otro lado, si la aplicación corre en un dispositivo móvil, antes de iniciar el flujo de Expo Auth Session comprobamos que la petición OAuth (request) ya esté preparada; si no lo está, informamos al usuario con “Google Auth no está listo.”

Una vez listo, ejecutamos `promptAsync()`, que despliega la interfaz nativa de Google y deja el manejo final del token (intercambio por credencial y redirección) en el `useEffect` configurado para procesar la respuesta.

```
const loginWithGoogle = async () => {  
  if (Platform.OS === "web") {  
    const provider = new GoogleAuthProvider();  
    try {  
      await signInWithPopup(auth, provider);  
      setErrorMessage("");  
      router.replace("/home");  
    } catch (err: any) {  
      setErrorMessage("Error con Google (web): " + err.message);  
    }  
  } else {  
    if (!request) {  
      setErrorMessage("Google Auth no está listo.");  
    } else {  
      promptAsync();  
    }  
  }  
};
```

5. Vista a renderizar

Ahora renderizamos la vista para poder mostrar el Login al usuario.

Primeramente agregaremos el diseño visual , agregaremos imágenes y las indicaciones al usuario.

Colocaremos los input necesarios para que funcione el login por medio de email y password.

Y por último tendremos la funcionalidad clave , que es el inicio de Sesión de google , aquí básicamente usaremos el método creado anteriormente y automáticamente hará la verificación yendo a los servidores a verificar que todo esté correcto y iniciara sesión.

```
<Text style={styles.or}>o</Text>

<TouchableOpacity
  style={styles.googleButton}
  onPress={loginWithGoogle}
  disabled={Platform.OS !== "web" && !request}
>
  <AntDesign name="google" size={24} color="#DB4437" />
  <Text style={styles.googleButtonText}>Iniciar con Google</Text>
</TouchableOpacity>
</View>
</SafeAreaView>
);
```