UDB VIRTUAL

Universidad Don Bosco, El Salvador

Asignatura: Diseño y Programación de Software Multiplataforma.

Docente: Alexander Alberto Siguenza Campos.

.

Integrantes

- Jorge Miguel Alberto Cruz - AC221717.
- Diego Emerson Varela Linares - VL181980.
- Daniel Antonio Marroquin Granados MG161914
- Daniel Adonay Molina Jovel MJ150737

Actividad: Foro 2.

Fecha de entrega: 3/12/2023

Índice

Autenticación de React Native con Firebase

La mayoría de las aplicaciones requieren identificar a sus usuarios para garantizar la seguridad de sus datos y ofrecer una experiencia personalizada en diferentes dispositivos. Firebase Authentication facilita este proceso al proporcionar servicios de backend, SDK y bibliotecas de IU para autenticar a los usuarios. Ofrece opciones como autenticación por contraseña, número de teléfono y proveedores populares como Google, Facebook y Twitter. Además, se integra sin problemas con otros servicios de Firebase y sigue estándares de la industria como OAuth 2.0 y OpenID Connect, lo que facilita su incorporación en un backend personalizado.(*Firebase Authentication*, s. f.)

Correo electrónico y contraseña

Autentica a los usuarios con sus direcciones de correo electrónico y contraseñas. El SDK de Firebase Authentication proporciona métodos a fin de crear y administrar usuarios que utilizan sus direcciones de correo electrónico y contraseñas para acceder. Firebase Authentication también maneja el envío de correos electrónicos para restablecer la contraseña.(*Firebase Authentication*, s. f.)

Autenticación con número de teléfono

Envía mensajes SMS a los teléfonos de los usuarios para autenticarlos.

Integración con proveedores de identidad federada:

Google

Configuración del Proyecto Firebase:

Se utiliza Firebase para manejar la autenticación y otros servicios en la aplicación.

Se crea un nuevo proyecto en la Consola de Firebase.

Se obtiene un identificador de la aplicación y se configura en el proyecto.

Creación del Proyecto React Native:

Se crea un nuevo proyecto React Native utilizando el comando react-native init.

Se instala el paquete react-native-google-signin para habilitar la autenticación con Google.

Configuración del paquete react-native-google-signin:

Se importan módulos y componentes necesarios del paquete.

Se crean estados (loggedIn y userInfo) para manejar el estado de autenticación y la información del usuario.

Función de Inicio de Sesión:

Se implementa la función _signIn para manejar el proceso de inicio de sesión con Google.

Se utiliza el método GoogleSignin.signIn() para realizar la autenticación.

Se manejan posibles errores, como cancelación por parte del usuario o servicios de Google no disponibles.

Configuración de Firebase y Estado de Sesión:

Se utiliza el hook useEffect para configurar la autenticación de Google una vez que el componente se monta.Se inicializa la configuración del objeto de inicio de sesión de Google con GoogleSignin.configure().

Función de Cierre de Sesión:

Se implementa la función signOut para manejar la acción de cierre de sesión.

Se utilizan métodos de GoogleSignin para revocar el acceso y cerrar sesión.

Se actualizan los estados loggedIn y userInfo.

Interfaz de Usuario (UI):

Se utiliza la biblioteca React Native para crear la interfaz de usuario.Se muestra un botón de inicio de sesión con Google, que cambia a un botón de cierre de sesión una vez que el usuario ha iniciado sesión correctamente.

Se utiliza el componente GoogleSigninButton proporcionado por el paquete.

Ejecución del Proyecto:

Se ejecuta el proyecto en un emulador de dispositivos móviles.Se muestra la interfaz de usuario con el botón "Iniciar sesión con Google".

Facebook

Configuración Inicial:

Agrega Firebase a tu proyecto de JavaScript:

Asegúrate de haber integrado Firebase en tu proyecto JavaScript antes de comenzar con la autenticación.

Obtén ID de app y Secret de app en Facebook:

Ve al sitio Facebook for Developers y obtén el ID de la aplicación y el Secret de la aplicación para tu aplicación.

Habilita el Acceso con Facebook en Firebase:

En la consola de Firebase, en la sección de Autenticación, habilita el método de acceso con Facebook y proporciona el ID de la aplicación y el Secret de la aplicación de Facebook.

Asegúrate de que la URI de redireccionamiento de OAuth esté configurada en Firebase y coincida con la configuración en Facebook.

Manejo del Flujo de Acceso con el SDK de Firebase JavaScript:

Crea una instancia del objeto proveedor de Facebook:

Importa FacebookAuthProvider del paquete firebase/auth y crea una instancia del proveedor.

Opcional: Especifica permisos adicionales de OAuth 2.0:

Puedes agregar permisos adicionales utilizando addScope en el proveedor.

Opcional: Ajusta el flujo de OAuth según el idioma:

Puedes actualizar el código de idioma en la instancia de Auth antes de iniciar el flujo de OAuth.

Opcional: Especifica parámetros personalizados del proveedor de OAuth:

Puedes usar setCustomParameters para enviar parámetros personalizados junto con la solicitud de OAuth.

Autentica con Firebase a través del proveedor de Facebook:

Utiliza signInWithPopup o signInWithRedirect para autenticar al usuario.

Recupera información adicional y maneja errores:

Después de la autenticación, puedes obtener información sobre el usuario y el token de acceso.

Puedes manejar errores utilizando las promesas then y catch y acceder a códigos de error específicos.

Redireccionamiento (Opcional):

Redireccionamiento a la página de acceso:

Si prefieres redireccionar a la página de acceso en lugar de usar una ventana emergente, utiliza signInWithRedirect y luego llama a getRedirectResult para obtener el resultado.

Recupera el token de OAuth después del redireccionamiento:

Utiliza getRedirectResult para obtener el token de acceso y otra información después de redireccionar al usuario.

Notas Adicionales:

Puedes utilizar el token de acceso para acceder a la API de Facebook y obtener más datos del usuario.

La sección proporciona manejo de errores y acceso a códigos de error específicos.

En resumen, este conjunto de instrucciones proporciona pasos detallados para configurar y utilizar la autenticación con Facebook en Firebase mediante JavaScript, ya sea gestionando el flujo de acceso con el SDK de Firebase o realizándolo manualmente en entornos no basados en navegadores.

Twitter

Configuración inicial para implementar autenticación de Twitter.

Instalación de librería externa **@react-native-twitter-signin/twitter-signin** en el proyecto de React Native**.**

Luego habilitamos el proveedor de autenticación de Twitter en Firebase, en la consola de Firebase, nos aseguramos de que el proveedor de inicio de sesión de Twitter esté habilitado.

Inicializar el SDK de Twitter, es necesario inicializar el SDK antes de realizar una solicitud de inicio de sesión, con las claves de consumo(**consumer key**) y (**consumer secret**) de la cuenta de Twitter en React Native, como se muestra en fragmento de código:

```
import { NativeModules } from 'react-native';
const { RNTwitterSignIn } = NativeModules;

RNTwitterSignIn.init('TWITTER_CONSUMER_KEY',
'TWITTER_CONSUMER_SECRET').then(() =>
  console.log('Twitter SDK initialized'),
);
```

Aquí, TWITTER_CONSUMER_KEY y TWITTER_CONSUMER_SECRET deben reemplazarse con las claves y secretos reales de tu aplicación de Twitter.

Generar credenciales para Firebase

Luego de inicializar el SDK de Twitter, procederemos a realizar una solicitud de inicio de sesión con Twitter.

GitHub

Configuración inicial para implementar autenticación de GitHub.

Configurar la aplicación GitHub OAuth:

Ir a github.com/settings/developers y crear una nueva aplicación OAuth.

Agregar el esquema de la aplicación como URL de devolución de llamada autorizada,

Copiar el Client ID y Client Secret generados en GitHub.

Configurar Firebase:

Crear un nuevo proyecto en Firebase.

Habilitar la autenticación y seleccionar GitHub como proveedor.

Ingresar el Client ID y Client Secret de GitHub cuando se soliciten durante la configuración de Firebase.

Vinculación a la aplicación:

Especificar una URL personalizada scheme para la aplicación en el archivo app.json

Dependencias:

Instalar las dependencias esenciales utilizando los siguientes comandos en la terminal:

npx expo install firebase

npx expo install expo-web-browser

npx expo install expo-auth-session expo-crypto

npx expo install @react-native-async-storage/async-storage

Variables de entorno:

Crear un archivo. env con las variables de entorno necesarias, que contienen las credenciales de GitHub y Firebase.

Conectar la aplicación a Firebase:

Crear un archivo para las credenciales de backend y para inicializar la aplicación Firebase.

Ejercicio práctico

npm y paquetes expo utilizados:

npm install --save @react-native-firebase/app

npm i @react-native-firebase/auth

npx expo install @react-native-google-signin/google-signin

npx expo install expo-dev-client

Primero instalamos los paquetes requeridos.

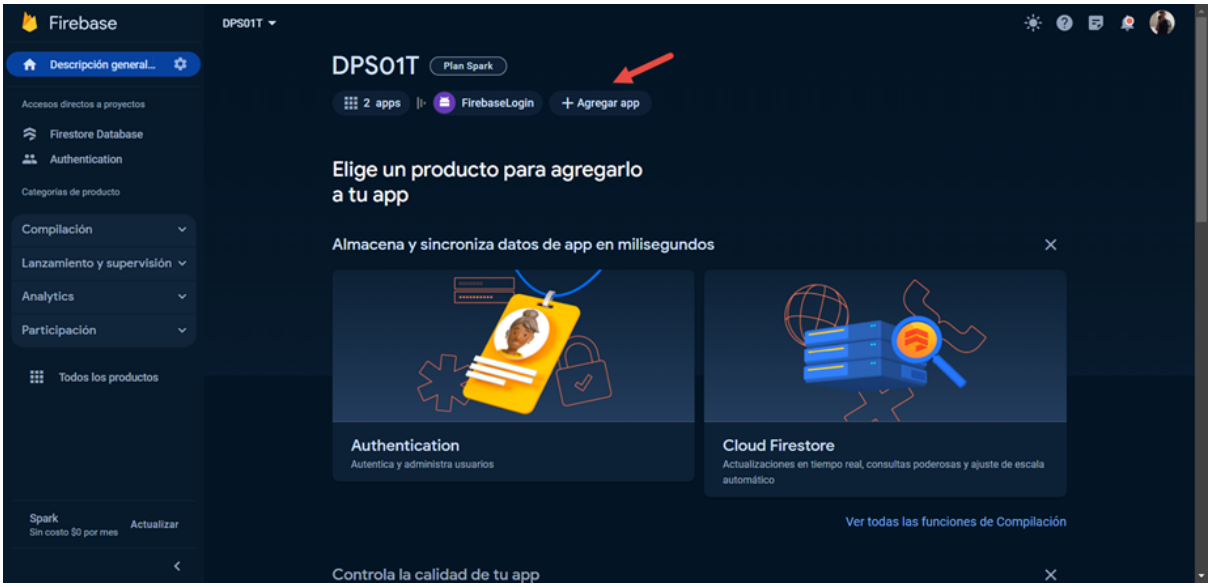
```
# Using npm
npm install --save @react-native-firebase/app

# Using Yarn
yarn add @react-native-firebase/app
```

```
C:\Users\dvarela\Documents\UDB_DSP\fireBaseLogin>npx expo install @react-native-google-signin/google-signin
```

Y así sucesivamente.

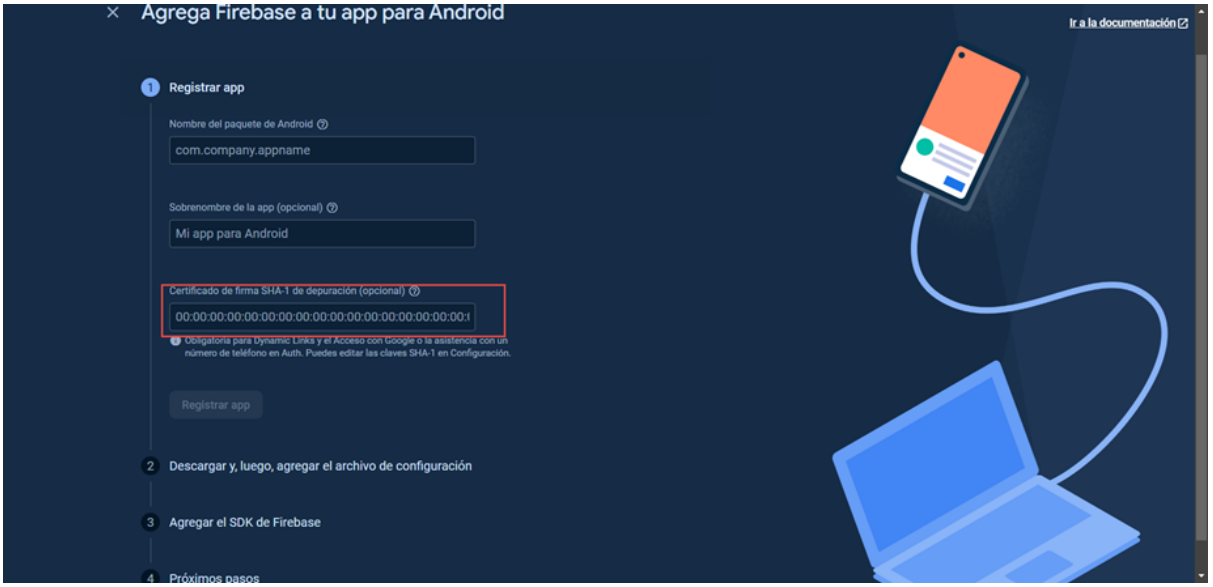
Para poder utilizar los servicios de Firebase para iniciar sesión con Google, es necesario que creemos un proyecto. Podemos hacerlo ingresando al siguiente enlace: https://console.firebase.google.com/.

Una vez creado el proyecto, necesitamos registrar nuestra aplicación.



Elegimos la opción para Android

Llenamos los datos con la información de nuestro proyecto:

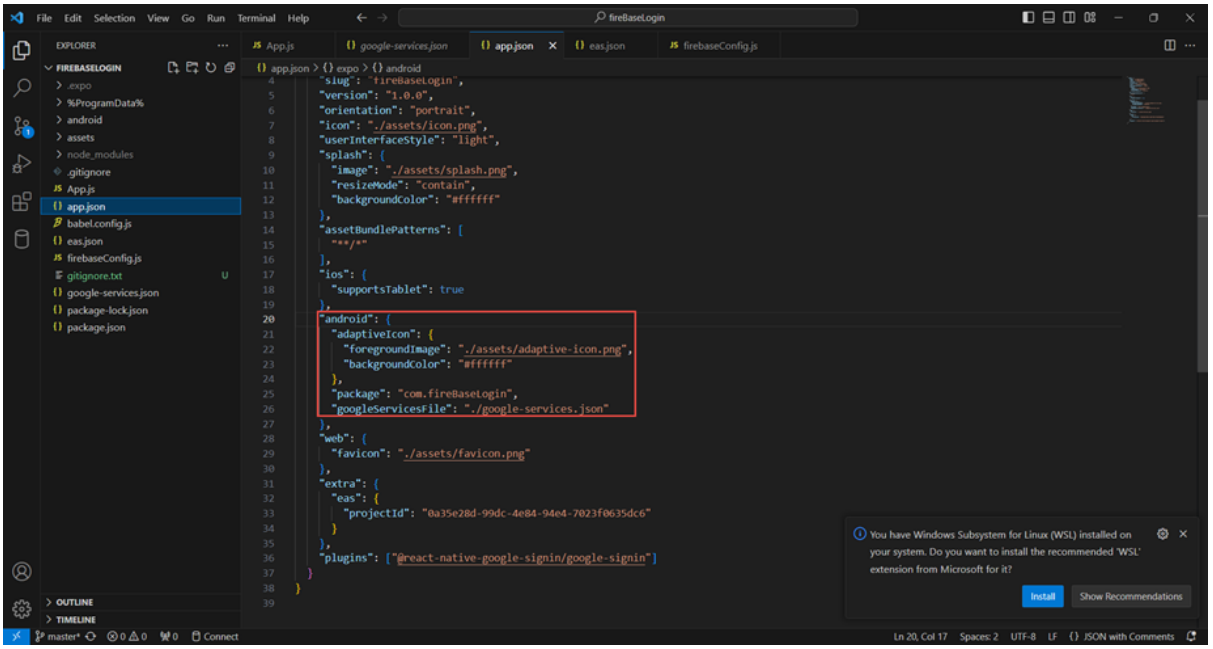


Para obtener el nombre del paquete Android, debemos hacer un prebuild de nuestro proyecto expo utilizando el siguiente comando:

npx expo prebuild --platform android

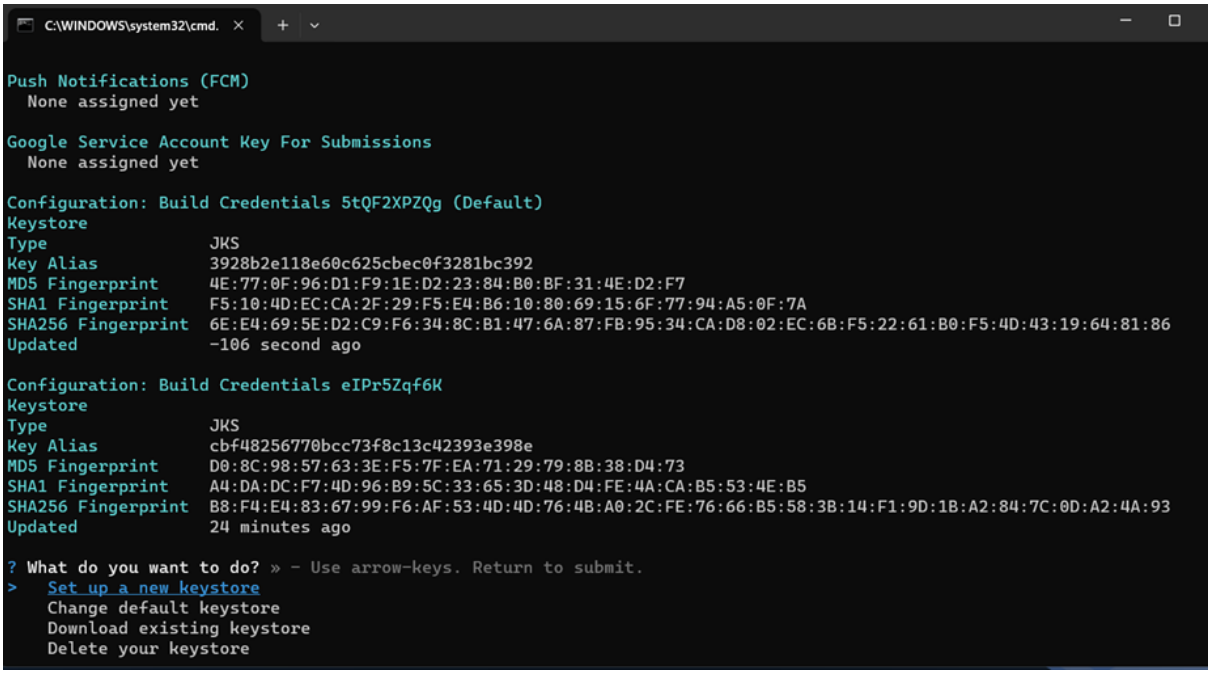
Luego nos pedirá el nombre del paquete, esto generara el Código de app nativa para poder obtener el nombre del paquete.

En nuestro archivo app.json obtenemos el nombre del paquete para luego podes usarlo al registrar nuestra app en firebase.



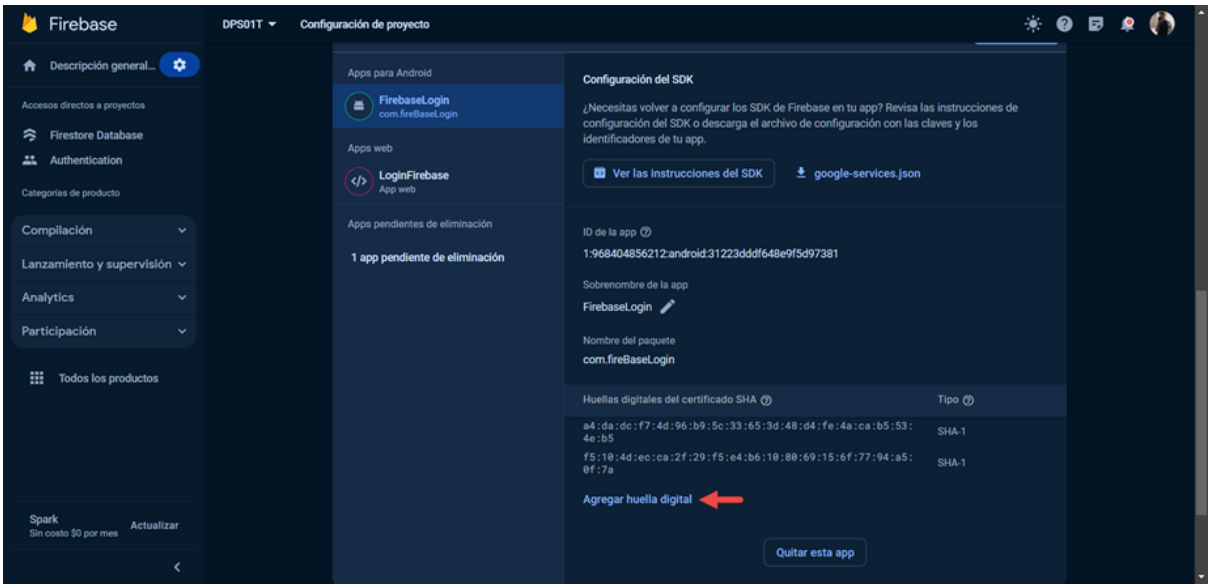
También nos solicita una clave SHA1 para poder registrar la app, para este caso con la ayuda de EAS de expo, creamos una llave para nuestro proyecto con el siguiente comando:

EAS credentials

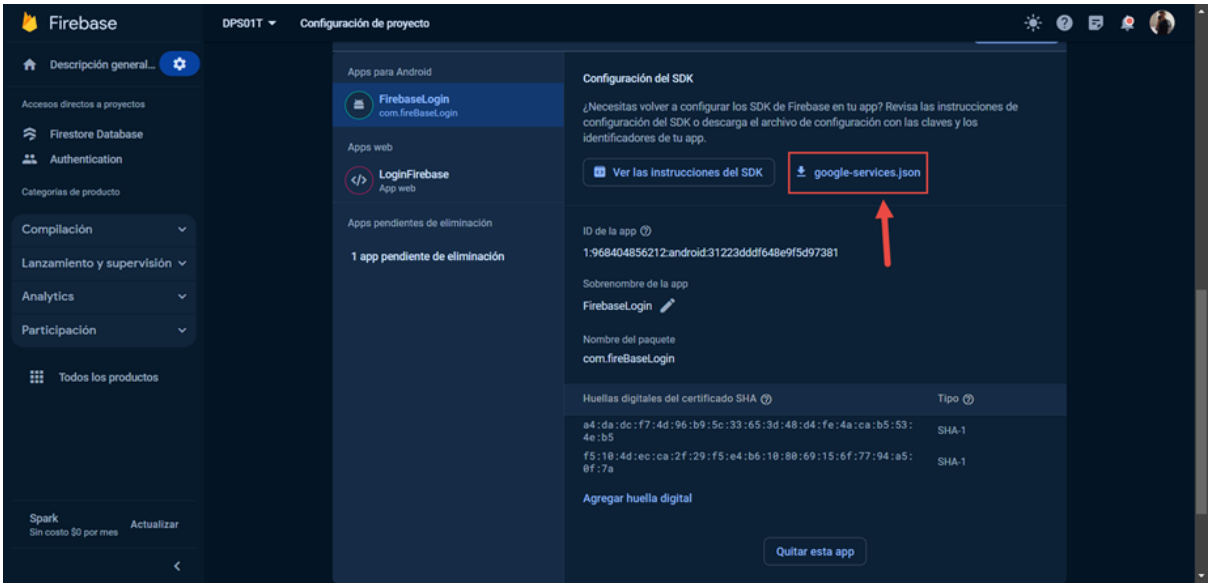


Y copiamos el dato SHA1 Figereprint de el segundo bloque, ese lo usamos para registrar el proyecto FireBase.

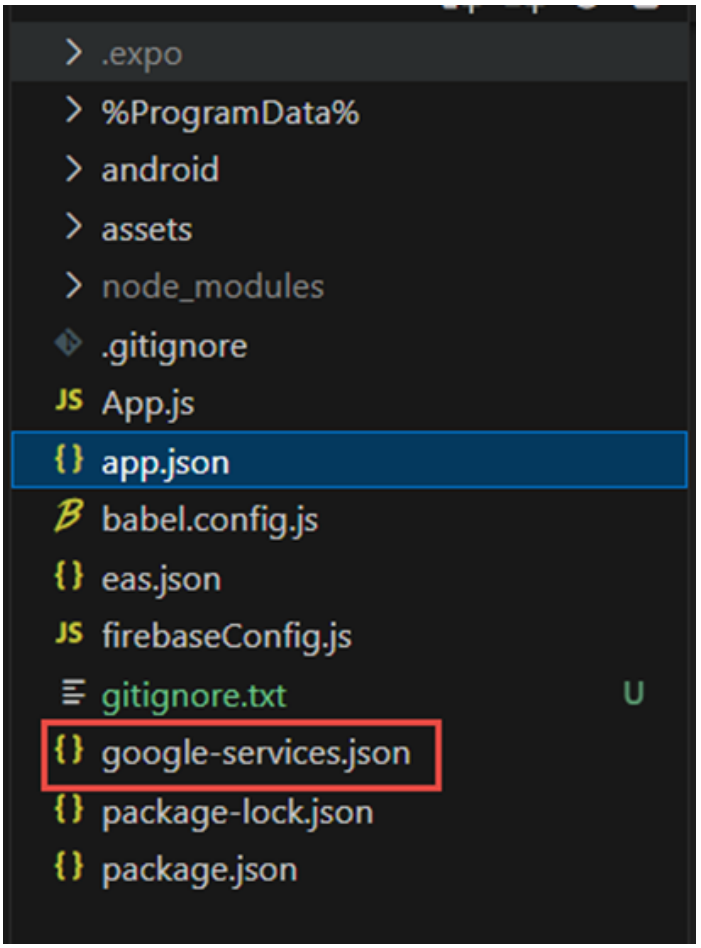
Ahora agregamos la llave (default) SHA1 a nuestro proyecto



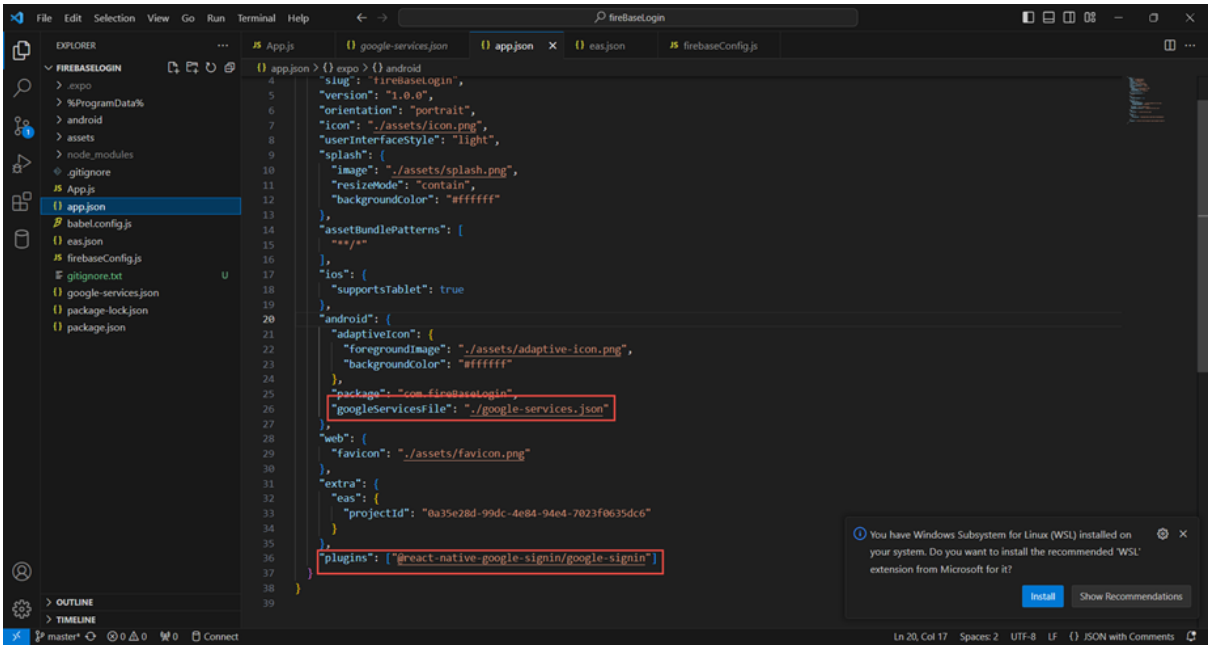
Una vez agregada, descargamos el archivo .json que nos genera firebase.



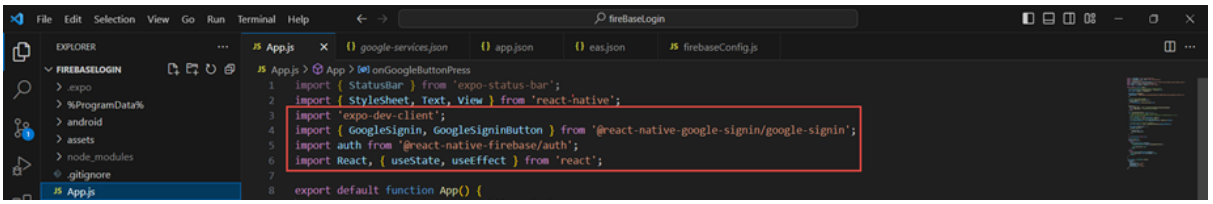
Este lo colocamos en la raíz de nuestro proyecto expo



En nuestro archivo app.json agregamos las siguientes líneas.



En el archivo app.js agregamos los siguientes imports



Ahora implementamos los métodos de firebase para registrarnos, junto con el botón de inicio de sesión de Google que ya posee la librería.

```
export default function App() {
  // Set an initializing state whilst Firebase connects
  const [initializing, setInitializing] = useState(true);
  const [user, setUser] = useState();

  GoogleSignin.configure({
    webClientId: '968404856212-hc24pcd9878vd9mnfqcjih0g33t9sakg.apps.googleusercontent.com',
  });

  // Handle user state changes
  function onAuthStateChanged(user) {
    setUser(user);
    if (initializing) setInitializing(false);
  }

  useEffect(() => {
    const subscriber = auth().onAuthStateChanged(onAuthStateChanged);
    return subscriber; // unsubscribe on unmount
  }, []);

  const onGoogleButtonPress = async () => {
    // Check if your device supports Google Play
    //await GoogleSignin.hasPlayServices({ showPlayServicesUpdateDialog: true });
    // Get the users ID token
    const { idToken } = await GoogleSignin.signIn();
    // Create a Google credential with the token
    const googleCredential = auth.GoogleAuthProvider.credential(idToken);

    // Sign-in the user with the credential
    const user_sign_in = auth().signInWithCredential(googleCredential);

    user_sign_in.then((user) => {
      console.log(user);
    })
      .catch((error) => {
        console.log(error);
      });
  }
```

```
  if (initializing) return null;

  console.log(user);
  if (user == null) {
    return (
      <View style={styles.container}>
        <GoogleSigninButton onPress={onGoogleButtonPress} />

        <Text>Bienvenido</Text>
      </View>
    );
  } else {
    return (<View style={styles.container}>
      <Text>Bienvenido {user.displayName}</Text>
    </View>);
  }
}

const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: '#fff',
    alignItems: 'center',
    justifyContent: 'center',
  },
});
```

Enlaces

Video explicativo
https://drive.google.com/drive/folders/1CHwzEUbIA9b9DznBJqISE4AM0-pSmlcE

Bibliografía

Firebase Authentication. (s. f.). Firebase.

https://firebase.google.com/docs/auth?hl=es-419

Cómo configurar el inicio de sesión de Google en React Native y Firebase:

https://www.freecodecamp.org/espanol/news/como-configurar-el-inicio-de-sesion-de-google-en-react-native-y-firebase/

Autentica mediante el Acceso con Facebook y JavaScript

https://firebase.google.com/docs/auth/web/facebook-login?hl=es

React Native GitHub Login with Firebase Authentication

Social Authentication