M. Bahrul Ulum, M.Kom





Pemrograman Mobile

Firebase Authentication

**Firebase Authentication**

Sebagian besar aplikasi perlu mengetahui identitas pengguna. Dengan mengetahui identitas pengguna, aplikasi dapat menyimpan data pengguna secara aman di cloud dan memberikan pengalaman personal yang sama di setiap perangkat pengguna.

Firebase Authentication menyediakan layanan backend, SDK yang mudah digunakan, dan library UI yang siap pakai untuk mengautentikasi pengguna ke aplikasi Anda. Firebase Authentication mendukung autentikasi menggunakan sandi, nomor telepon, penyedia identitas gabungan yang populer, seperti Google, Facebook, dan Twitter, dan lain-lain.

Firebase Authentication terintegrasi erat dengan layanan Firebase lainnya dan sistem ini memanfaatkan berbagai standar industri, seperti OAuth 2.0 dan OpenID Connect, sehingga dapat dengan mudah diintegrasikan dengan backend kustom Anda.

**Kemampuan utama**

Anda dapat membuat pengguna login ke aplikasi Firebase dengan menggunakan FirebaseUI sebagai solusi autentikasi drop-in lengkap atau dengan menggunakan Firebase Authentication SDK untuk mengintegrasikan salah satu atau beberapa metode login ke aplikasi Anda secara manual.

|  |
| --- |
| **FirebaseUI Auth**  |
| **Solusi autentikasi drop-in** | Cara yang direkomendasikan untuk menambahkan sistem login yang lengkap ke aplikasi Anda.FirebaseUI menyediakan solusi autentikasi drop-in yang menangani alur UI untuk membuat pengguna login dengan alamat email dan sandi, nomor telepon, dan penyedia identitas gabungan yang populer, seperti Login dengan Google dan Login dengan Facebook. Komponen FirebaseUI Auth mengimplementasikan praktik terbaik untuk proses autentikasi di perangkat seluler dan situs, yang akan memaksimalkan konversi login dan pendaftaran untuk aplikasi Anda. FirebaseUI Auth juga menangani kasus khusus, seperti pemulihan akun dan penyambungan akun, yang dapat dianggap sensitif bagi keamanan dan rawan kesalahan untuk ditangani dengan benar.FirebaseUI dapat disesuaikan dengan mudah agar cocok dengan gaya visual aplikasi Anda. FirebaseUI juga open source, sehingga Anda tidak dibatasi dalam mewujudkan pengalaman pengguna yang Anda inginkan. |
| **Firebase SDK Authentication** |
| **Autentikasi berbasis email dan sandi** | Mengautentikasi pengguna dengan alamat email dan sandi mereka. Firebase Authentication SDK menyediakan metode untuk membuat dan mengelola pengguna yang login menggunakan alamat email dan sandi. Firebase Authentication juga menangani pengiriman email untuk menyetel ulang sandi.[iOS](https://firebase.google.com/docs/auth/ios/password-auth?hl=id) [Android](https://firebase.google.com/docs/auth/android/password-auth?hl=id) [Web](https://firebase.google.com/docs/auth/web/password-auth?hl=id) [C++](https://firebase.google.com/docs/auth/cpp/password-auth?hl=id) [Unity](https://firebase.google.com/docs/auth/unity/password-auth?hl=id)  |
| **Integrasi penyedia identitas tergabung** | Mengautentikasi pengguna dengan mengintegrasikan penyedia identitas gabungan. Firebase Authentication SDK menyediakan metode yang memungkinkan pengguna untuk login dengan akun Google, Facebook, Twitter, dan GitHub mereka.

|  |  |
| --- | --- |
| Google | [iOS](https://firebase.google.com/docs/auth/ios/google-signin?hl=id) [Android](https://firebase.google.com/docs/auth/android/google-signin?hl=id) [Web](https://firebase.google.com/docs/auth/web/google-signin?hl=id) [C++](https://firebase.google.com/docs/auth/cpp/google-signin?hl=id) [Unity](https://firebase.google.com/docs/auth/unity/google-signin?hl=id)  |
| Facebook | [iOS](https://firebase.google.com/docs/auth/ios/facebook-login?hl=id) [Android](https://firebase.google.com/docs/auth/android/facebook-login?hl=id) [Web](https://firebase.google.com/docs/auth/web/facebook-login?hl=id) [C++](https://firebase.google.com/docs/auth/cpp/facebook-login?hl=id) [Unity](https://firebase.google.com/docs/auth/unity/facebook-login?hl=id)  |
| Twitter | [iOS](https://firebase.google.com/docs/auth/ios/twitter-login?hl=id) [Android](https://firebase.google.com/docs/auth/android/twitter-login?hl=id) [Web](https://firebase.google.com/docs/auth/web/twitter-login?hl=id) [C++](https://firebase.google.com/docs/auth/cpp/twitter-login?hl=id) [Unity](https://firebase.google.com/docs/auth/unity/twitter-login?hl=id)  |
| GitHub | [iOS](https://firebase.google.com/docs/auth/ios/github-auth?hl=id) [Android](https://firebase.google.com/docs/auth/android/github-auth?hl=id) [Web](https://firebase.google.com/docs/auth/web/github-auth?hl=id) [C++](https://firebase.google.com/docs/auth/cpp/github-auth?hl=id) [Unity](https://firebase.google.com/docs/auth/unity/github-auth?hl=id)  |

 |
| **Autentikasi nomor telepon** | Mengautentikasi pengguna dengan mengirim pesan SMS ke ponsel mereka.[iOS](https://firebase.google.com/docs/auth/ios/phone-auth?hl=id) [Android](https://firebase.google.com/docs/auth/android/phone-auth?hl=id) [Web](https://firebase.google.com/docs/auth/web/phone-auth?hl=id) [C++](https://firebase.google.com/docs/auth/cpp/phone-auth?hl=id) [Unity](https://firebase.google.com/docs/auth/unity/phone-auth?hl=id)  |
| **Integrasi sistem autentikasi khusus** | Menghubungkan sistem login aplikasi yang ada ke Firebase Authentication SDK, serta memperoleh akses ke Firebase Realtime Database dan layanan Firebase lainnya.[iOS](https://firebase.google.com/docs/auth/ios/custom-auth?hl=id) [Android](https://firebase.google.com/docs/auth/android/custom-auth?hl=id) [Web](https://firebase.google.com/docs/auth/web/custom-auth?hl=id) [C++](https://firebase.google.com/docs/auth/cpp/custom-auth?hl=id) [Unity](https://firebase.google.com/docs/auth/unity/custom-auth?hl=id)  |
| **Autentikasi anonim** | Menggunakan fitur yang memerlukan autentikasi tanpa mewajibkan pengguna untuk login terlebih dahulu dengan membuat akun anonim sementara. Jika pengguna kemudian memilih untuk login, Anda dapat mengupgrade akun anonim menjadi akun reguler agar pengguna dapat melanjutkan pekerjaannya.[iOS](https://firebase.google.com/docs/auth/ios/anonymous-auth?hl=id) [Android](https://firebase.google.com/docs/auth/android/anonymous-auth?hl=id) [Web](https://firebase.google.com/docs/auth/web/anonymous-auth?hl=id) [C++](https://firebase.google.com/docs/auth/cpp/anonymous-auth?hl=id) [Unity](https://firebase.google.com/docs/auth/unity/anonymous-auth?hl=id)  |

**Bagaimana cara kerjanya?**

Untuk membuat pengguna login ke aplikasi, dapatkan kredensial autentikasi dari pengguna terlebih dahulu. Kredensial ini dapat berupa alamat email dan sandi pengguna, atau token OAuth dari penyedia identitas gabungan. Kemudian, teruskan kredensial ini ke Firebase Authentication SDK. Layanan backend kami selanjutnya akan memverifikasi kredensial tersebut dan menampilkan respons ke klien.

Setelah berhasil login, Anda dapat mengakses informasi profil dasar pengguna dan mengontrol akses pengguna ke data yang disimpan di produk Firebase lainnya. Anda juga dapat menggunakan token autentikasi yang disediakan untuk memverifikasi identitas pengguna di layanan backend yang dimiliki.

Catatan: Secara default, pengguna yang terautentikasi dapat membaca dan menulis data ke Firebase Realtime Database dan Cloud Storage. Anda dapat mengontrol akses pengguna tersebut dengan mengubah Firebase Realtime Database dan Aturan Keamanan Cloud Storage.

**Alur implementasi**

|  |
| --- |
| **Menggunakan FirebaseUI Auth** |
|  | **Menyiapkan metode login** | Untuk login dengan alamat email dan sandi, atau nomor telepon, serta semua penyedia identitas gabungan yang ingin Anda dukung, aktifkan semua fungsi tersebut di Firebase console dan selesaikan setiap konfigurasi yang diperlukan oleh penyedia identitas, seperti menetapkan URL pengalihan OAuth. |
|  | **Menyesuaikan UI login** | Anda juga dapat menyesuaikan UI login dengan menyetel opsi FirebaseUI atau jalankan proses fork terhadap kode tersebut di GitHub untuk menyesuaikan pengalaman login lebih jauh. |
|  | **Menggunakan FirebaseUI untuk menjalankan alur login** | Impor library FirebaseUI, tentukan metode login yang ingin Anda dukung, lalu inisialisasi alur login FirebaseUI. |
| **Menggunakan Firebase Authentication SDK** |
|  | **Menyiapkan metode login** | Untuk login dengan alamat email dan sandi, atau nomor telepon, serta semua penyedia identitas gabungan yang ingin Anda dukung, aktifkan semua fungsi tersebut di Firebase console dan selesaikan setiap konfigurasi yang diperlukan oleh penyedia identitas, seperti menetapkan URL pengalihan OAuth. |
|  | **Mengimplementasikan alur UI untuk metode login** | Untuk login alamat email dan sandi, implementasikan alur yang meminta pengguna menulis alamat email dan sandinya. Untuk login dengan nomor telepon, buat alur yang meminta nomor telepon pengguna serta kode dari pesan SMS yang mereka terima. Untuk login tergabung, implementasikan alur yang diperlukan oleh setiap penyedia. |
|  | **Meneruskan kredensial pengguna ke Firebase Authentication SDK** | Teruskan alamat email dan sandi pengguna atau token OAuth yang diperoleh dari penyedia identitas gabungan ke Firebase Authentication SDK. |

# Pengguna dalam Project Firebase

Objek **pengguna** Firebase mewakili akun pengguna yang telah mendaftar untuk suatu aplikasi dalam project Anda. Aplikasi biasanya memiliki banyak pengguna yang terdaftar di dalamnya, dan setiap aplikasi dalam project saling berbagi database pengguna.

Instance pengguna bersifat independen dari instance Firebase Authentication, jadi Anda dapat memiliki beberapa referensi untuk pengguna yang berbeda dalam konteks yang sama dan tetap memanggil metode mereka.

## Properti pengguna

Pengguna Firebase memiliki rangkaian tetap properti dasar, yaitu ID yang unik, alamat email utama, nama dan URL foto—yang disimpan di database pengguna project, yang dapat diperbarui oleh pengguna. Anda tidak dapat menambahkan properti lain ke objek pengguna secara langsung. Namun, Anda dapat menyimpan properti tambahan di layanan penyimpanan lain seperti Google Cloud Firestore.

Saat pertama kali pengguna mendaftar ke aplikasi Anda, data profil pengguna akan diisi menggunakan informasi yang tersedia:

* Jika pengguna mendaftar dengan alamat email dan sandi, hanya properti alamat email utama yang akan diisi
* Jika pengguna mendaftar dengan penyedia identitas gabungan, seperti Google atau Facebook, informasi akun yang disediakan oleh penyedia digunakan untuk mengisi profil pengguna
* Jika pengguna mendaftar dengan sistem autentikasi kustom, Anda harus secara eksplisit menambahkan informasi yang diinginkan ke profil pengguna

Setelah akun pengguna dibuat, Anda dapat memuat ulang informasi pengguna tersebut agar menyertakan perubahan yang mungkin telah dibuat oleh si pengguna pada perangkat lain.

## Penyedia proses login

Anda dapat membuat pengguna login ke aplikasi melalui beberapa metode: alamat email dan sandi, penyedia identitas gabungan, dan sistem autentikasi kustom. Anda dapat mengaitkan lebih dari satu metode login dengan suatu pengguna: misalnya, pengguna dapat login ke akun yang sama menggunakan alamat email dan sandi, atau menggunakan Login dengan Google.

Instance pengguna melacak setiap penyedia yang terhubung dengan pengguna. Hal ini memungkinkan Anda untuk memperbarui properti profil kosong menggunakan informasi yang diberikan oleh penyedia.

## Pengguna aktif

Ketika pengguna mendaftar atau login, pengguna tersebut menjadi pengguna aktif dari instance Auth. Instance ini mempertahankan status pengguna, sehingga memuat ulang halaman (pada browser) atau memulai ulang aplikasi tidak akan menghapus informasi pengguna ini.

Ketika pengguna logout, instance Auth berhenti menyimpan referensi ke objek User dan tidak lagi mempertahankan statusnya; tidak ada pengguna yang aktif saat ini. Namun, instance pengguna tetap berfungsi sepenuhnya: jika Anda menyimpan referensi ke sana, Anda masih dapat mengakses dan mengupdate data pengguna.

## Siklus proses pengguna

Cara yang direkomendasikan untuk melacak status instance Auth saat ini adalah dengan menggunakan listener (juga disebut "observer" pada JavaScript). Listener Auth akan dikirimi pemberitahuan setiap kali sesuatu yang relevan terjadi pada objek Auth. Baca bagian Mengelola Pengguna.

Listener Auth akan diberi tahu jika terjadi situasi berikut:

* Objek Auth selesai melakukan inisialisasi dan pengguna sudah login dari sesi sebelumnya atau telah dialihkan dari alur login penyedia identitas
* Pengguna login (pengguna aktif ditetapkan)
* Pengguna logout (pengguna aktif menjadi null)
* Token akses pengguna aktif dimuat ulang. Hal ini dapat terjadi dalam keadaan berikut:
	+ Token akses berakhir: ini yang biasa terjadi. Token refresh digunakan untuk memperoleh sekumpulan token valid yang baru.
	+ Pengguna mengubah sandinya: Firebase menerbitkan akses baru dan token refresh, kemudian token lama dibuat tidak berlaku lagi. Ini secara otomatis membuat token pengguna tidak berlaku dan/atau membuat pengguna logout dari setiap perangkat untuk alasan keamanan.
	+ Pengguna melakukan autentikasi ulang: beberapa tindakan mengharuskan bahwa kredensial pengguna diterbitkan baru-baru ini; tindakan tersebut termasuk menghapus akun, menentukan alamat email utama, dan mengubah sandi. Daripada membuat pengguna logout kemudian login lagi, dapatkan kredensial baru dari pengguna dan teruskan kredensial baru tersebut ke metode autentikasi ulang dari objek pengguna.

## Token Auth

Ketika Anda melakukan autentikasi dengan Firebase, ada tiga jenis token auth yang mungkin Anda temui:

|  |  |
| --- | --- |
| Token ID Firebase | Dibuat oleh Firebase saat pengguna login ke aplikasi. Token ini adalah JWT yang ditandatangani yang mengidentifikasi pengguna dalam project Firebase secara aman. Token ini berisi informasi profil dasar untuk pengguna, termasuk string ID pengguna yang unik untuk project Firebase. Karena integritas token ID dapat diverifikasi, Anda dapat mengirimkannya ke server backend untuk mengidentifikasi pengguna yang login saat ini.  |
| Token penyedia identitas | Dibuat oleh penyedia identitas bersama, seperti Google dan Facebook. Token ini dapat memiliki format yang berbeda, namun sering kali menggunakan token akses OAuth 2.0. Aplikasi Firebase menggunakan token ini untuk memastikan bahwa pengguna telah berhasil diautentikasi dengan penyedia identitas, kemudian mengubahnya menjadi kredensial yang dapat digunakan oleh layanan Firebase.  |
| Token kustom Firebase | Dibuat oleh sistem autentikasi kustom agar pengguna bisa login ke aplikasi menggunakan sistem autentikasi Anda. Token kustom adalah JWT yang ditandatangani menggunakan kunci pribadi akun layanan. Aplikasi menggunakan token ini dengan cara yang sama seperti saat menggunakan token yang diperoleh dari penyedia identitas gabungan.  |

**Menambahkan login secara mudah ke aplikasi Android dengan FirebaseUI**

Firebase UI adalah library yang dibuat sebagai tambahan dari Firebase Authentication SDK yang menyediakan alur UI drop-in untuk digunakan dalam aplikasi Anda. FirebaseUI memberikan manfaat berikut:

* **Beberapa Penyedia** - alur login untuk email, autentikasi telepon, Login dengan Google, Login dengan Facebook, dan Login dengan Twitter.
* **Manajemen Akun** - alur untuk menangani tugas pengelolaan akun, seperti pembuatan akun dan penyetelan ulang sandi.
* **Penautan Akun** - alur untuk menautkan akun pengguna secara aman di seluruh penyedia identitas.
* **Tema Kustom** - menyesuaikan tampilan FirebaseUI agar cocok dengan aplikasi Anda. Selain itu, karena FirebaseUI bersifat open source, Anda bisa menyalin project dan menyesuaikannya sesuai kebutuhan.
* **Smart Lock untuk Sandi** - integrasi otomatis dengan [Smart Lock untuk Sandi](https://developers.google.com/identity/smartlock-passwords/android/?hl=id) untuk login lintas perangkat dengan cepat.

**Sebelum memulai**

1. [Tambahkan Firebase ke project Android](https://firebase.google.com/docs/android/setup?hl=id).
2. Tambahkan dependensi untuk FirebaseUI ke file build.gradle tingkat aplikasi. Jika Anda ingin mendukung login dengan Facebook atau Twitter, sertakan juga SDK Facebook dan Twitter:

dependencies {
    // ...

    implementation 'com.firebaseui:firebase-ui-auth:4.1.0'

    // Required only if Facebook login support is required
    // Find the latest Facebook SDK releases here: https://goo.gl/Ce5L94
    implementation 'com.facebook.android:facebook-android-sdk:4.x'

    // Required only if Twitter login support is required
    // Find the latest Twitter SDK releases here: https://goo.gl/E5wZvQ
    implementation 'com.twitter.sdk.android:twitter-core:3.x'
}

 FirebaseUI Auth SDK memiliki ketergantungan transitif pada Firebase SDK dan SDK layanan Google Play.

 Jika Anda belum menghubungkan aplikasi dengan project Firebase, lakukanlah dari [Firebase console](https://console.firebase.google.com/?hl=id).

 Di [Firebase console](https://console.firebase.google.com/?hl=id), buka bagian **Authentication**, lalu aktifkan metode login yang ingin Anda dukung. Beberapa metode login memerlukan informasi tambahan yang biasanya tersedia di konsol developer layanan.

 Jika Anda mendukung Login dengan Google dan belum menentukan sidik jari SHA-1 aplikasi, lakukanlah dari [halaman Setelan](https://console.firebase.google.com/project/_/settings/general/?hl=id) di Firebase console. Baca bagian [Mengautentikasi Klien Anda](https://developers.google.com/android/guides/client-auth?hl=id) untuk mengetahui informasi selengkapnya mengenai cara mendapatkan sidik jari SHA-1 aplikasi Anda.

 Jika Anda mendukung login dengan Facebook atau Twitter, tambahkan resource string ke strings.xml yang menentukan informasi identifikasi yang diperlukan oleh masing-masing penyedia:

<resources>
  <!-- Facebook application ID and custom URL scheme (app ID prefixed by 'fb'). -->
  <string name="facebook\_application\_id" translatable="false">*YOUR\_APP\_ID*</string>
  <string name="facebook\_login\_protocol\_scheme" translatable="false">fb*YOUR\_APP\_ID*</string>
  <!-- Twitter consumer key and secret -->
  <string name="twitter\_consumer\_key" translatable="false">*YOUR\_CONSUMER\_KEY*</string>
  <string name="twitter\_consumer\_secret" translatable="false">*YOUR\_CONSUMER\_SECRET*</string>
</resources>

**Login**

Untuk memulai alur login FirebaseUI, buat intent login dengan metode login yang Anda pilih:

[Java](https://firebase.google.com/docs/auth/android/firebaseui?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/firebaseui?hl=id#kotlin-android)

// Choose authentication providers
List<AuthUI.IdpConfig> providers = Arrays.asList(
        new AuthUI.IdpConfig.EmailBuilder().build(),
        new AuthUI.IdpConfig.PhoneBuilder().build(),
        new AuthUI.IdpConfig.GoogleBuilder().build(),
        new AuthUI.IdpConfig.FacebookBuilder().build(),
        new AuthUI.IdpConfig.TwitterBuilder().build());

// Create and launch sign-in intent
startActivityForResult(
        AuthUI.getInstance()
                .createSignInIntentBuilder()
                .setAvailableProviders(providers)
                .build(),
        RC\_SIGN\_IN);

[FirebaseUIActivity.java](https://github.com/firebase/snippets-android/blob/b8036229e4209f0bab6a49ecdd01b20ad8f7eabb/auth/app/src/main/java/com/google/firebase/quickstart/auth/FirebaseUIActivity.java#L31-L45)

Setelah alur login selesai, Anda akan menerima hasilnya di onActivityResult:

[Java](https://firebase.google.com/docs/auth/android/firebaseui?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/firebaseui?hl=id#kotlin-android)

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);

    if (requestCode == RC\_SIGN\_IN) {
        IdpResponse response = IdpResponse.fromResultIntent(data);

        if (resultCode == RESULT\_OK) {
            // Successfully signed in
            FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();
            // ...
        } else {
            // Sign in failed. If response is null the user canceled the
            // sign-in flow using the back button. Otherwise check
            // response.getError().getErrorCode() and handle the error.
            // ...
        }
    }
}

[FirebaseUIActivity.java](https://github.com/firebase/snippets-android/blob/b8036229e4209f0bab6a49ecdd01b20ad8f7eabb/auth/app/src/main/java/com/google/firebase/quickstart/auth/FirebaseUIActivity.java#L50-L68)

**Logout**

FirebaseUI menyediakan cara mudah untuk logout dari Firebase Authentication dan semua penyedia identitas sosial:

[Java](https://firebase.google.com/docs/auth/android/firebaseui?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/firebaseui?hl=id#kotlin-android)

AuthUI.getInstance()
        .signOut(this)
        .addOnCompleteListener(new OnCompleteListener<Void>() {
            public void onComplete(@NonNull Task<Void> task) {
                // ...
            }
        });

[FirebaseUIActivity.java](https://github.com/firebase/snippets-android/blob/b8036229e4209f0bab6a49ecdd01b20ad8f7eabb/auth/app/src/main/java/com/google/firebase/quickstart/auth/FirebaseUIActivity.java#L73-L79)

Anda juga dapat menghapus akun pengguna sepenuhnya:

[Java](https://firebase.google.com/docs/auth/android/firebaseui?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/firebaseui?hl=id#kotlin-android)

AuthUI.getInstance()
        .delete(this)
        .addOnCompleteListener(new OnCompleteListener<Void>() {
            @Override
            public void onComplete(@NonNull Task<Void> task) {
                // ...
            }
        });

[FirebaseUIActivity.java](https://github.com/firebase/snippets-android/blob/b8036229e4209f0bab6a49ecdd01b20ad8f7eabb/auth/app/src/main/java/com/google/firebase/quickstart/auth/FirebaseUIActivity.java#L85-L92)

**Penyesuaian**

Secara default, FirebaseUI menggunakan AppCompat untuk penentuan tema yang berarti ini akan secara alami menggunakan skema warna aplikasi Anda. Jika Anda memerlukan penyesuaian lebih lanjut, Anda dapat meneruskan tema dan logo ke builder Intent login:

[Java](https://firebase.google.com/docs/auth/android/firebaseui?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/firebaseui?hl=id#kotlin-android)

startActivityForResult(
        AuthUI.getInstance()
                .createSignInIntentBuilder()
                .setAvailableProviders(providers)
                .setLogo(R.drawable.my\_great\_logo)      // Set logo drawable
                .setTheme(R.style.MySuperAppTheme)      // Set theme
                .build(),
        RC\_SIGN\_IN);

[FirebaseUIActivity.java](https://github.com/firebase/snippets-android/blob/b8036229e4209f0bab6a49ecdd01b20ad8f7eabb/auth/app/src/main/java/com/google/firebase/quickstart/auth/FirebaseUIActivity.java#L100-L107)

Anda juga dapat menetapkan kebijakan privasi dan persyaratan layanan khusus:

[Java](https://firebase.google.com/docs/auth/android/firebaseui?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/firebaseui?hl=id#kotlin-android)

startActivityForResult(
        AuthUI.getInstance()
                .createSignInIntentBuilder()
                .setAvailableProviders(providers)
                .setTosAndPrivacyPolicyUrls(
                        "https://example.com/terms.html",
                        "https://example.com/privacy.html")
                .build(),
        RC\_SIGN\_IN);

**Memulai Firebase Authentication di Android**

**Menghubungkan aplikasi ke Firebase**

Tambahkan Firebase ke project Android jika Anda belum melakukannya.

**Menambahkan Firebase Authentication ke aplikasi Anda**

Tambahkan dependensi untuk library Android Firebase Authentication ke file Gradle modul (level aplikasi) Anda (biasanya berupa app/build.gradle):

implementation 'com.google.firebase:firebase-auth:16.2.1'

Jika ingin menggunakan suatu penyedia otentikasi, Anda harus mengaktifkannya di [Firebase console](https://console.firebase.google.com/?hl=id). Buka halaman Metode Login di bagian Firebase Authentication untuk mengaktifkan Login dengan Email/Sandi dan penyedia identitas lain mana pun yang Anda inginkan untuk aplikasi Anda.

**Memeriksa status autentikasi saat ini**

1. Nyatakan instance FirebaseAuth.

[Java](https://firebase.google.com/docs/auth/android/start?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/start?hl=id#kotlin-android)

private FirebaseAuth mAuth;

[EmailPasswordActivity.java](https://github.com/firebase/quickstart-android/blob/25754213f8ceae8e72644ecfb4d7f8ac93dfa0b5/auth/app/src/main/java/com/google/firebase/quickstart/auth/java/EmailPasswordActivity.java#L46-L46)

  Pada metode onCreate(), inisialisasi instance FirebaseAuth.

[Java](https://firebase.google.com/docs/auth/android/start?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/start?hl=id#kotlin-android)

// Initialize Firebase Auth
mAuth = FirebaseAuth.getInstance();

[EmailPasswordActivity.java](https://github.com/firebase/quickstart-android/blob/25754213f8ceae8e72644ecfb4d7f8ac93dfa0b5/auth/app/src/main/java/com/google/firebase/quickstart/auth/java/EmailPasswordActivity.java#L67-L68)

  Saat menginisialisasi Aktivitas Anda, periksa apakah pengguna saat ini sedang login.

[Java](https://firebase.google.com/docs/auth/android/start?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/start?hl=id#kotlin-android)

@Override
public void onStart() {
    super.onStart();
    // Check if user is signed in (non-null) and update UI accordingly.
    FirebaseUser currentUser = mAuth.getCurrentUser();
    updateUI(currentUser);
}

[EmailPasswordActivity.java](https://github.com/firebase/quickstart-android/blob/25754213f8ceae8e72644ecfb4d7f8ac93dfa0b5/auth/app/src/main/java/com/google/firebase/quickstart/auth/java/EmailPasswordActivity.java#L73-L79)

**Mendaftarkan pengguna baru**

Buat metode createAccount baru yang menggunakan alamat email dan sandi, jalankan validasi, lalu buat pengguna baru dengan metode [createUserWithEmailAndPassword] (/docs/reference/android/com/google/firebase/auth/FirebaseAuth.html#createUserWithEmailAndPassword(java.lang.String, java.lang.String).

[Java](https://firebase.google.com/docs/auth/android/start?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/start?hl=id#kotlin-android)

mAuth.createUserWithEmailAndPassword(email, password)
        .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful()) {
                    // Sign in success, update UI with the signed-in user's information
                    Log.d(TAG, "createUserWithEmail:success");
                    FirebaseUser user = mAuth.getCurrentUser();
                    updateUI(user);
                } else {
                    // If sign in fails, display a message to the user.
                    Log.w(TAG, "createUserWithEmail:failure", task.getException());
                    Toast.makeText(EmailPasswordActivity.this, "Authentication failed.",
                            Toast.LENGTH\_SHORT).show();
                    updateUI(null);
                }

                // ...
            }
        });

[EmailPasswordActivity.java](https://github.com/firebase/quickstart-android/blob/25754213f8ceae8e72644ecfb4d7f8ac93dfa0b5/auth/app/src/main/java/com/google/firebase/quickstart/auth/java/EmailPasswordActivity.java#L91-L112)

Tambahkan formulir untuk mendaftarkan pengguna baru dengan email dan sandi mereka, dan panggil metode baru ini saat dikirim. Anda dapat melihat contohnya pada [sampel quickstart](https://github.com/firebase/quickstart-android/tree/master/auth/app/src/main) kami.

**Membuat pengguna yang ada agar login**

Buat metode signIn baru yang akan menerima alamat email dan sandi, memvalidasinya, lalu membuat pengguna login dengan metode [signInWithEmailAndPassword](https://firebase.google.com/docs/reference/android/com/google/firebase/auth/FirebaseAuth.html?hl=id#signInWithEmailAndPassword(java.lang.String,%20java.lang.String)).

[Java](https://firebase.google.com/docs/auth/android/start?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/start?hl=id#kotlin-android)

mAuth.signInWithEmailAndPassword(email, password)
        .addOnCompleteListener(this, new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if (task.isSuccessful()) {
                    // Sign in success, update UI with the signed-in user's information
                    Log.d(TAG, "signInWithEmail:success");
                    FirebaseUser user = mAuth.getCurrentUser();
                    updateUI(user);
                } else {
                    // If sign in fails, display a message to the user.
                    Log.w(TAG, "signInWithEmail:failure", task.getException());
                    Toast.makeText(EmailPasswordActivity.this, "Authentication failed.",
                            Toast.LENGTH\_SHORT).show();
                    updateUI(null);
                }

                // ...
            }
        });

[EmailPasswordActivity.java](https://github.com/firebase/quickstart-android/blob/25754213f8ceae8e72644ecfb4d7f8ac93dfa0b5/auth/app/src/main/java/com/google/firebase/quickstart/auth/java/EmailPasswordActivity.java#L125-L149)

Tambahkan formulir untuk membuat pengguna login dengan email dan sandi mereka, dan panggil metode baru ini saat dikirim. Anda dapat melihat contohnya pada [sampel quickstart](https://github.com/firebase/quickstart-android/tree/master/auth/app/src/main) kami.

**Mengakses informasi pengguna**

Jika pengguna berhasil login, Anda bisa mendapatkan data akun mereka pada tahap mana saja dengan metode getCurrentUser.

[Java](https://firebase.google.com/docs/auth/android/start?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/start?hl=id#kotlin-android)

FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();
if (user != null) {
    // Name, email address, and profile photo Url
    String name = user.getDisplayName();
    String email = user.getEmail();
    Uri photoUrl = user.getPhotoUrl();

    // Check if user's email is verified
    boolean emailVerified = user.isEmailVerified();

    // The user's ID, unique to the Firebase project. Do NOT use this value to
    // authenticate with your backend server, if you have one. Use
    // FirebaseUser.getIdToken() instead.
    String uid = user.getUid();
}

# Mengelola Pengguna di Firebase

## Membuat pengguna

Anda dapat membuat pengguna baru dalam project Firebase dengan memanggil metode [createUserWithEmailAndPassword](https://firebase.google.com/docs/auth/android/password-auth?hl=id#create_a_password-based_account) atau dengan membuat pengguna login untuk pertama kalinya menggunakan penyedia identitas gabungan, seperti [Login dengan Google](https://firebase.google.com/docs/auth/android/google-signin?hl=id) atau [Login dengan Facebook](https://firebase.google.com/docs/auth/android/facebook-login?hl=id).

Anda juga dapat membuat pengguna baru yang diautentikasi dengan sandi dari bagian Authentication pada [Firebase console](https://console.firebase.google.com/?hl=id), di halaman Pengguna.

## Mengetahui pengguna yang sedang login

Cara yang direkomendasikan untuk mengetahui pengguna saat ini adalah dengan memanggil metode getCurrentUser. Jika tidak ada pengguna yang login, getCurrentUser akan menampilkan null:

[Java](https://firebase.google.com/docs/auth/android/manage-users?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/manage-users?hl=id#kotlin-android)

FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();
if (user != null) {
    // User is signed in
} else {
    // No user is signed in
}

[MainActivity.java](https://github.com/firebase/snippets-android/blob/b8036229e4209f0bab6a49ecdd01b20ad8f7eabb/auth/app/src/main/java/com/google/firebase/quickstart/auth/MainActivity.java#L65-L70)

Ada beberapa kasus di mana getCurrentUser akan menampilkan FirebaseUser non-null, tapi token utama tidak valid. Hal ini dapat terjadi, misalnya, jika pengguna dihapus pada perangkat lain dan token lokal belum diperbarui. Dalam kasus ini, Anda mungkin akan mendapatkan getCurrentUser pengguna yang valid, namun panggilan berikutnya ke resource yang diautentikasi akan gagal.

getCurrentUser mungkin juga akan menampilkan null karena proses inisialisasi pada objek auth belum selesai.

Jika Anda memasang [AuthStateListener](https://firebase.google.com/docs/reference/android/com/google/firebase/auth/FirebaseAuth.AuthStateListener?hl=id), Anda akan mendapatkan callback setiap kali status token utama berubah. Hal ini bisa berguna untuk bereaksi terhadap kasus berisiko seperti yang disebutkan di atas.

## Memperoleh profil pengguna

Untuk memperoleh informasi profil pengguna, gunakan metode assessor dari instance FirebaseUser. Misalnya:

[Java](https://firebase.google.com/docs/auth/android/manage-users?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/manage-users?hl=id#kotlin-android)

FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();
if (user != null) {
    // Name, email address, and profile photo Url
    String name = user.getDisplayName();
    String email = user.getEmail();
    Uri photoUrl = user.getPhotoUrl();

    // Check if user's email is verified
    boolean emailVerified = user.isEmailVerified();

    // The user's ID, unique to the Firebase project. Do NOT use this value to
    // authenticate with your backend server, if you have one. Use
    // FirebaseUser.getIdToken() instead.
    String uid = user.getUid();
}

[MainActivity.java](https://github.com/firebase/snippets-android/blob/b8036229e4209f0bab6a49ecdd01b20ad8f7eabb/auth/app/src/main/java/com/google/firebase/quickstart/auth/MainActivity.java#L76-L90)

## Memperoleh informasi profil khusus penyedia pengguna

Untuk memperoleh informasi profil yang diambil dari penyedia login yang ditautkan ke pengguna, gunakan metode getProviderData. Misalnya:

[Java](https://firebase.google.com/docs/auth/android/manage-users?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/manage-users?hl=id#kotlin-android)

FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();
if (user != null) {
    for (UserInfo profile : user.getProviderData()) {
        // Id of the provider (ex: google.com)
        String providerId = profile.getProviderId();

        // UID specific to the provider
        String uid = profile.getUid();

        // Name, email address, and profile photo Url
        String name = profile.getDisplayName();
        String email = profile.getEmail();
        Uri photoUrl = profile.getPhotoUrl();
    }
}

[MainActivity.java](https://github.com/firebase/snippets-android/blob/b8036229e4209f0bab6a49ecdd01b20ad8f7eabb/auth/app/src/main/java/com/google/firebase/quickstart/auth/MainActivity.java#L96-L110)

## Mengupdate profil pengguna

Anda dapat mengupdate informasi profil dasar pengguna—yaitu nama tampilan dan URL foto profil pengguna—dengan metode updateProfile. Misalnya:

[Java](https://firebase.google.com/docs/auth/android/manage-users?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/manage-users?hl=id#kotlin-android)

FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();

UserProfileChangeRequest profileUpdates = new UserProfileChangeRequest.Builder()
        .setDisplayName("Jane Q. User")
        .setPhotoUri(Uri.parse("https://example.com/jane-q-user/profile.jpg"))
        .build();

user.updateProfile(profileUpdates)
        .addOnCompleteListener(new OnCompleteListener<Void>() {
            @Override
            public void onComplete(@NonNull Task<Void> task) {
                if (task.isSuccessful()) {
                    Log.d(TAG, "User profile updated.");
                }
            }
        });

[MainActivity.java](https://github.com/firebase/snippets-android/blob/b8036229e4209f0bab6a49ecdd01b20ad8f7eabb/auth/app/src/main/java/com/google/firebase/quickstart/auth/MainActivity.java#L116-L131)

## Menyetel alamat email pengguna

Anda dapat menyetel alamat email pengguna dengan metode updateEmail. Misalnya:

[Java](https://firebase.google.com/docs/auth/android/manage-users?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/manage-users?hl=id#kotlin-android)

FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();

user.updateEmail("user@example.com")
        .addOnCompleteListener(new OnCompleteListener<Void>() {
            @Override
            public void onComplete(@NonNull Task<Void> task) {
                if (task.isSuccessful()) {
                    Log.d(TAG, "User email address updated.");
                }
            }
        });

[MainActivity.java](https://github.com/firebase/snippets-android/blob/b8036229e4209f0bab6a49ecdd01b20ad8f7eabb/auth/app/src/main/java/com/google/firebase/quickstart/auth/MainActivity.java#L137-L147)

**Penting:** Untuk menetapkan alamat email pengguna, pengguna tersebut harus pernah login baru-baru ini. Lihat [Mengautentikasi ulang pengguna](https://firebase.google.com/docs/auth/android/manage-users?hl=id#re-authenticate_a_user).

## Mengirim email verifikasi kepada pengguna

Anda dapat mengirim email verifikasi alamat kepada pengguna dengan metode sendEmailVerification. Misalnya:

[Java](https://firebase.google.com/docs/auth/android/manage-users?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/manage-users?hl=id#kotlin-android)

FirebaseAuth auth = FirebaseAuth.getInstance();
FirebaseUser user = auth.getCurrentUser();

user.sendEmailVerification()
        .addOnCompleteListener(new OnCompleteListener<Void>() {
            @Override
            public void onComplete(@NonNull Task<Void> task) {
                if (task.isSuccessful()) {
                    Log.d(TAG, "Email sent.");
                }
            }
        });

[MainActivity.java](https://github.com/firebase/snippets-android/blob/b8036229e4209f0bab6a49ecdd01b20ad8f7eabb/auth/app/src/main/java/com/google/firebase/quickstart/auth/MainActivity.java#L170-L181)

Anda dapat menyesuaikan template email yang digunakan di bagian Autentikasi dari [Firebase console](https://console.firebase.google.com/?hl=id) di halaman Template Email. Lihat [Template Email](https://support.google.com/firebase/answer/7000714?hl=id) di Pusat Bantuan Firebase.

Anda juga dapat meneruskan status melalui [continue URL](https://firebase.google.com/docs/auth/android/passing-state-in-email-actions?hl=id) untuk mengalihkannya kembali ke aplikasi tersebut ketika mengirim email verifikasi.

Selain itu, Anda dapat melokalkan email verifikasi dengan memperbarui kode bahasa pada instance Auth sebelum mengirim email. Misalnya:

[Java](https://firebase.google.com/docs/auth/android/manage-users?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/manage-users?hl=id#kotlin-android)

auth.setLanguageCode("fr");
// To apply the default app language instead of explicitly setting it.
// auth.useAppLanguage();

[MainActivity.java](https://github.com/firebase/snippets-android/blob/b8036229e4209f0bab6a49ecdd01b20ad8f7eabb/auth/app/src/main/java/com/google/firebase/quickstart/auth/MainActivity.java#L210-L212)

## Menyetel sandi pengguna

Anda dapat menyetel sandi pengguna dengan metode updatePassword. Misalnya:

[Java](https://firebase.google.com/docs/auth/android/manage-users?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/manage-users?hl=id#kotlin-android)

FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();
String newPassword = "SOME-SECURE-PASSWORD";

user.updatePassword(newPassword)
        .addOnCompleteListener(new OnCompleteListener<Void>() {
            @Override
            public void onComplete(@NonNull Task<Void> task) {
                if (task.isSuccessful()) {
                    Log.d(TAG, "User password updated.");
                }
            }
        });

[MainActivity.java](https://github.com/firebase/snippets-android/blob/b8036229e4209f0bab6a49ecdd01b20ad8f7eabb/auth/app/src/main/java/com/google/firebase/quickstart/auth/MainActivity.java#L153-L164)

**Penting:** Untuk menetapkan sandi pengguna, pengguna tersebut harus pernah login baru-baru ini.Lihat [Mengautentikasi ulang pengguna](https://firebase.google.com/docs/auth/android/manage-users?hl=id#re-authenticate_a_user).

## Mengirim email setel ulang sandi

Anda dapat mengirim email setel ulang sandi kepada pengguna dengan metode sendPasswordResetEmail. Misalnya:

[Java](https://firebase.google.com/docs/auth/android/manage-users?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/manage-users?hl=id#kotlin-android)

FirebaseAuth auth = FirebaseAuth.getInstance();
String emailAddress = "user@example.com";

auth.sendPasswordResetEmail(emailAddress)
        .addOnCompleteListener(new OnCompleteListener<Void>() {
            @Override
            public void onComplete(@NonNull Task<Void> task) {
                if (task.isSuccessful()) {
                    Log.d(TAG, "Email sent.");
                }
            }
        });

[MainActivity.java](https://github.com/firebase/snippets-android/blob/b8036229e4209f0bab6a49ecdd01b20ad8f7eabb/auth/app/src/main/java/com/google/firebase/quickstart/auth/MainActivity.java#L218-L229)

Anda dapat menyesuaikan template email yang digunakan di bagian Autentikasi dari [Firebase console](https://console.firebase.google.com/?hl=id) di halaman Template Email. Lihat [Template Email](https://support.google.com/firebase/answer/7000714?hl=id) di Pusat Bantuan Firebase.

Anda juga bisa meneruskan status melalui [continue URL](https://firebase.google.com/docs/auth/android/passing-state-in-email-actions?hl=id) untuk mengalihkan kembali ke aplikasi tersebut ketika mengirim email setel ulang sandi.

Selain itu, Anda dapat melokalkan email setel ulang sandi dengan memperbarui kode bahasa pada instance Auth sebelum mengirim email. Misalnya:

[Java](https://firebase.google.com/docs/auth/android/manage-users?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/manage-users?hl=id#kotlin-android)

auth.setLanguageCode("fr");
// To apply the default app language instead of explicitly setting it.
// auth.useAppLanguage();

[MainActivity.java](https://github.com/firebase/snippets-android/blob/b8036229e4209f0bab6a49ecdd01b20ad8f7eabb/auth/app/src/main/java/com/google/firebase/quickstart/auth/MainActivity.java#L210-L212)

Anda juga bisa mengirimkan email setel ulang sandi dari Firebase console.

## Menghapus pengguna

Anda dapat menghapus akun pengguna dengan metode delete. Misalnya:

[Java](https://firebase.google.com/docs/auth/android/manage-users?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/manage-users?hl=id#kotlin-android)

FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();

user.delete()
        .addOnCompleteListener(new OnCompleteListener<Void>() {
            @Override
            public void onComplete(@NonNull Task<Void> task) {
                if (task.isSuccessful()) {
                    Log.d(TAG, "User account deleted.");
                }
            }
        });

[MainActivity.java](https://github.com/firebase/snippets-android/blob/b8036229e4209f0bab6a49ecdd01b20ad8f7eabb/auth/app/src/main/java/com/google/firebase/quickstart/auth/MainActivity.java#L235-L245)

**Penting:** Untuk menghapus pengguna, pengguna tersebut harus pernah login baru-baru ini. Lihat [Mengautentikasi ulang pengguna](https://firebase.google.com/docs/auth/android/manage-users?hl=id#re-authenticate_a_user).

Anda juga bisa menghapus pengguna dari bagian Autentikasi pada [Firebase console](https://console.firebase.google.com/?hl=id), di halaman Pengguna.

## Mengautentikasi ulang pengguna

Beberapa tindakan yang rentan terhadap ancaman keamanan, seperti [menghapus akun](https://firebase.google.com/docs/auth/android/manage-users?hl=id#delete_a_user), [menetapkan alamat email utama](https://firebase.google.com/docs/auth/android/manage-users?hl=id#set_a_users_email_address), dan [mengubah sandi](https://firebase.google.com/docs/auth/android/manage-users?hl=id#set_a_users_password), mengharuskan pengguna untuk pernah login baru-baru ini. Jika Anda melakukan salah satu tindakan tersebut dan pengguna login pada waktu yang sudah lama berlalu, tindakan akan gagal dengan error [FirebaseAuthRecentLoginRequiredException](https://firebase.google.com/docs/reference/android/com/google/firebase/auth/FirebaseAuthRecentLoginRequiredException?hl=id). Jika ini terjadi, autentikasi ulang pengguna tersebut dengan meminta kredensial login baru dari pengguna dan meneruskannya ke reauthenticate. Misalnya:

[Java](https://firebase.google.com/docs/auth/android/manage-users?hl=id#java-android)[Kotlin](https://firebase.google.com/docs/auth/android/manage-users?hl=id#kotlin-android)

FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();

// Get auth credentials from the user for re-authentication. The example below shows
// email and password credentials but there are multiple possible providers,
// such as GoogleAuthProvider or FacebookAuthProvider.
AuthCredential credential = EmailAuthProvider
        .getCredential("user@example.com", "password1234");

// Prompt the user to re-provide their sign-in credentials
user.reauthenticate(credential)
        .addOnCompleteListener(new OnCompleteListener<Void>() {
            @Override
            public void onComplete(@NonNull Task<Void> task) {
                Log.d(TAG, "User re-authenticated.");
            }
        });