



# React Native Firebase

Chapter 6 - Topic 1

---

**Selamat datang di Chapter 6 Topic 1 online course  
React Native dari Binar Academy!**





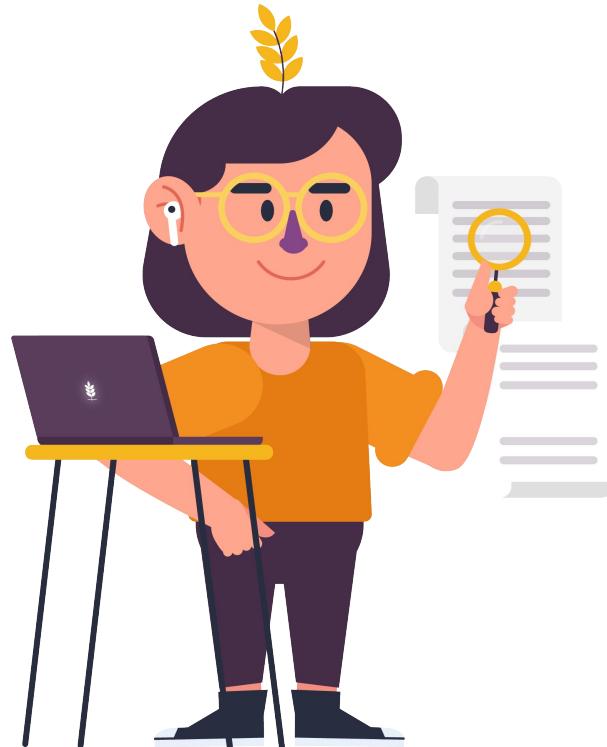
### Halooo temann..

Nggak kerasa ya, kita sudah masuk Chapter 6. Masih semangat belajar nggak nih? Hopefully, yes!

Oke, di Chapter 5 kemarin kamu sudah belajar tentang banyak hal, mulai dari media handling, TDD, Automation Testing, hingga CI/CD. Di Chapter 6 ini, kita akan membuka materi pelajaran dengan **React Native Firebase**.

Apa itu React Native Firebase?

Kuping berdengung, jangan dikipas. Nggak usah bingung, hayok kita bahas.



## Detailnya, kita bakal bahas hal-hal berikut ini:

- Cara mengintegrasikan aplikasi dengan Firebase Crashlytics
- Cara mengintegrasikan aplikasi dengan Firebase Analytics
- Cara mengintegrasikan aplikasi dengan Firebase remote Notification





Di materi ini kamu akan mempelajari salah satu SDK yang lumayan populer dari Google, yaitu **Firebase**.

Apa sih fungsi Firebase?





### Kenalan sama Firebase yuk

Jadi, untuk bikin dan mengembangkan aplikasi yang sukses, kamu akan membutuhkan Firebase. Mengapa?

Karena Firebase menyediakan tools untuk menganalisis, reporting, memperbaiki crash alias kerusakan, melakukan pemasaran, dan eksperimen produk. Lengkap ya?





Sebelum masuk lebih jauh ke Firebase, ada baiknya kalau kamu mengetahui dulu apa itu SDK nih, guys.





## Apa itu SDK?

**SDK (Software Development Kit)** merupakan suatu tools atau service yang dapat membantu kamu dalam mengembangkan aplikasi.

Ada banyak sekali SDK untuk mengembangkan aplikasi mulai dari yang gratis hingga yang berbayar.





Firebase sendiri merupakan SDK buatan Google yang hampir semua layanannya dapat kamu nikmati dan coba dengan gratis.

Apakah semuanya gratis? Tidak, karena Google tetap akan menarik tagihan (billing) jika penggunaanmu melewati batas kapasitas penggunaan firebase.



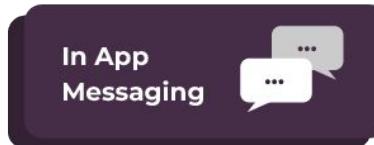
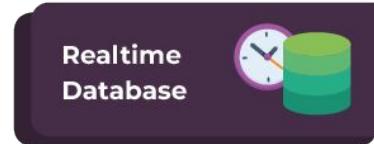
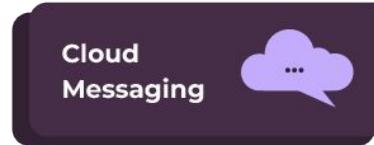
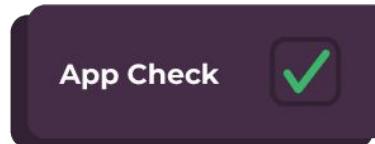


Firebase punya banyak layanan yang bisa kamu coba, di antaranya :

1. Analytics
2. App Check
3. App Distribution
4. Authentication
5. Cloud Firestore
6. Cloud Messaging
7. Crashlytics
8. Realtime Database
9. Dynamic Links
10. in App Messaging



Firebase punya banyak layanan yang bisa kamu coba, di antaranya :

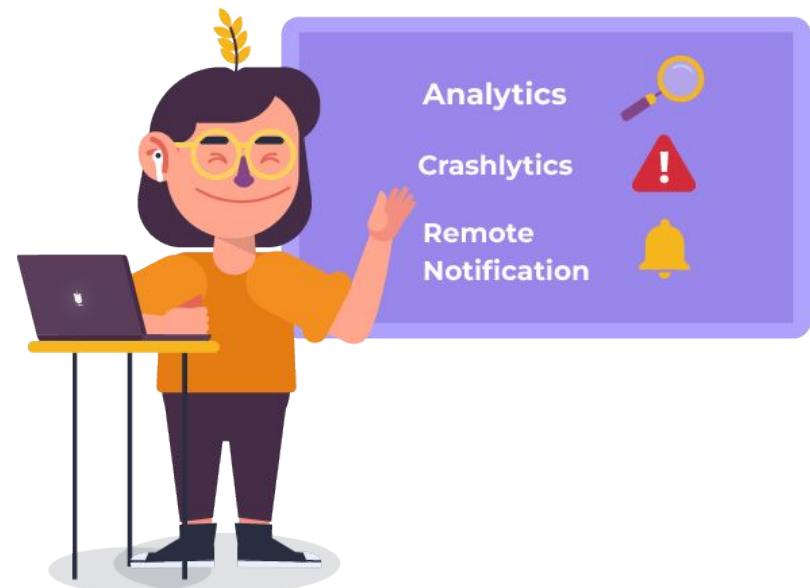




Banyak kan layanan yang dapat kamu pakai dari Google Firebase?

Meski banyak, tapi pada materi ini kamu hanya akan mempelajari cara menggunakan **Firebase Crashlytics** dan **Analytics serta Remote Notification** dulu yaa.

Lho kok cuma itu sih? Iya Sob, ini dulu, supaya kamu bisa tetap fokus dan terarah dalam belajar.





Guys, jangan lupa meluangkan waktu untuk memahami dan membaca dokumentasi yang tersedia secara official [di sini](#) ya!





Jadi, darimana kita akan memulai materi Firebase ini? Jawabannya adalah **Setup Firebase**.

Let's go, kita mulai~





Yaps, agar bisa menggunakan layanan dari Firebase, sebelumnya kamu perlu untuk melakukan setup Firebase dulu untuk mengenali aplikasamu.

Bagaimana caranya setup Firebase? Ikuti langkah pada slide-slide berikut ya.





### 1. Persiapkan project React Native

Silahkan gunakan project yang sudah ada ataupun membuat project baru.

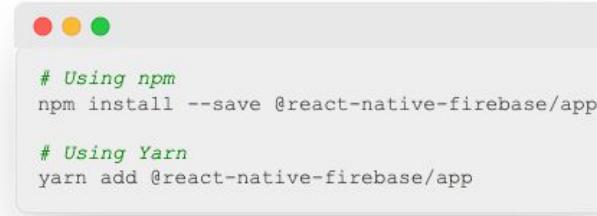


```
npx react-native init rnmedia
```



## 2. Install React Native Firebase “app” module

Kamu perlu menginstall package Firebase dengan menjalankan command seperti pada gambar di samping.



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

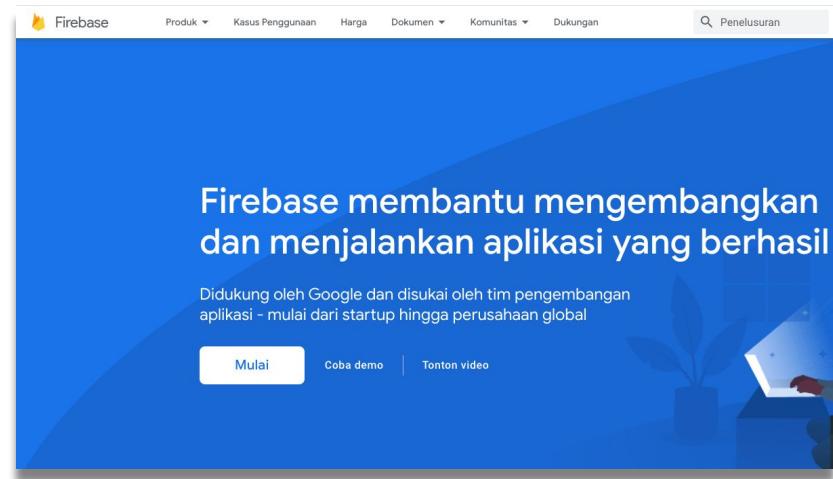
# Using Yarn
yarn add @react-native-firebase/app
```



### 3. Android Setup

Untuk aplikasi Android, kamu bisa terhubung dengan aman ke Firebase Project, configuration file harus di-download dan ditambahkan ke project-mu.

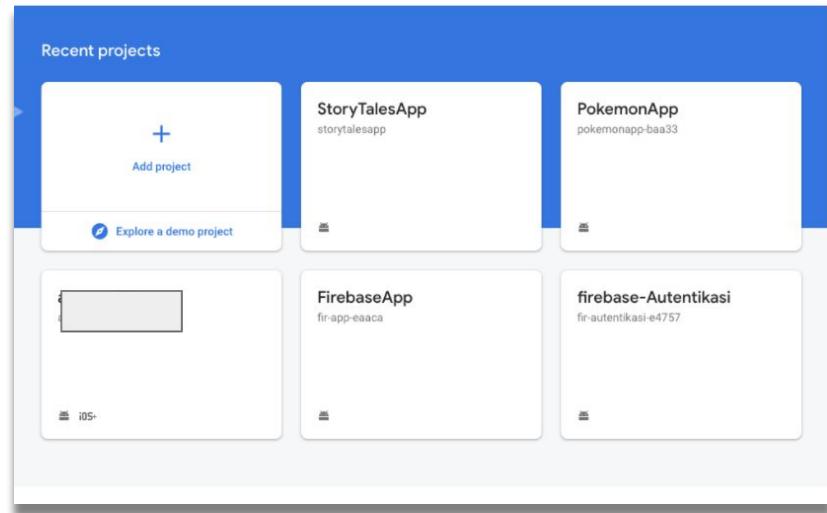
Bukalah [firebase.google.com](https://firebase.google.com) hingga menampilkan tampilan di samping.

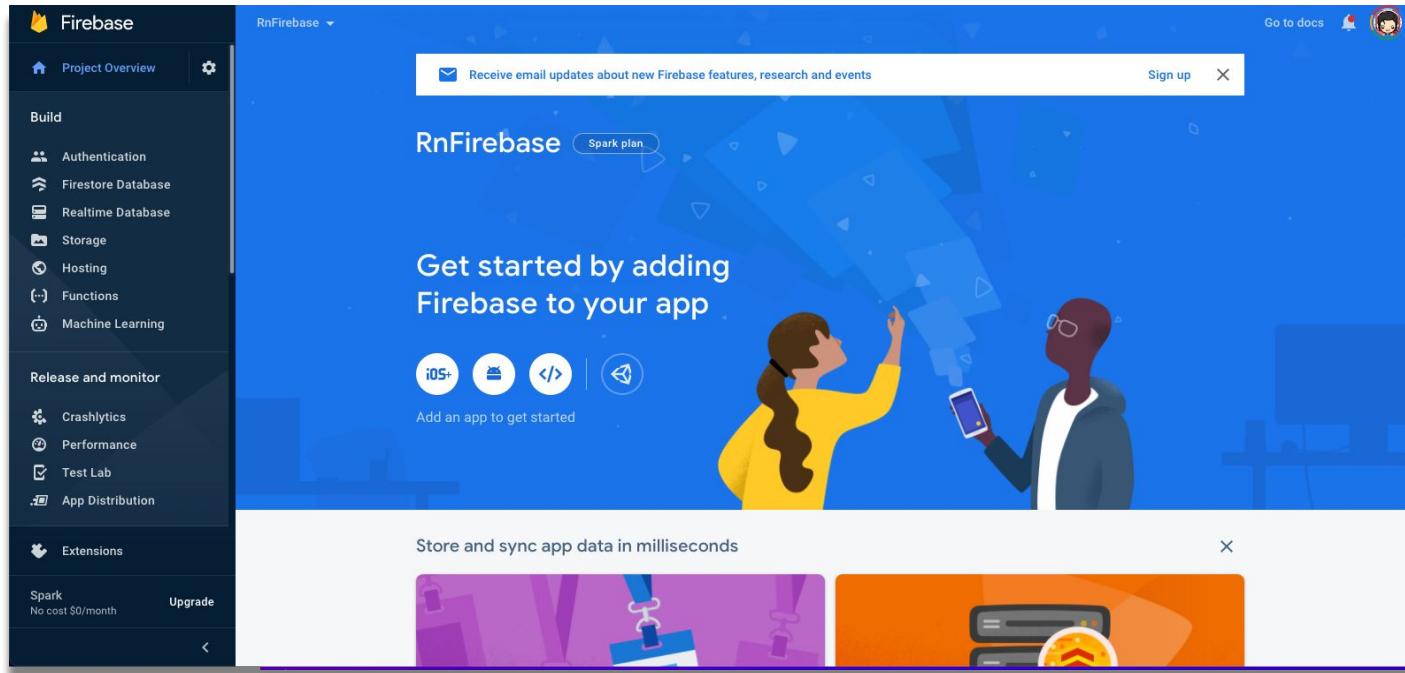




### Selanjutnya..

1. Buat Project Firebase baru kalian dengan menekan tombol **mulai**.
2. Kemudian tekan **Add Project**.
3. Selanjutnya kamu akan diminta memasukan **Project Name**.
4. Lanjutkan sampai kamu menemukan **Create Project**.
5. Tunggu beberapa saat hingga firebase menyelesaikan proses pembuatan project-mu.





6. Berikutnya, kamu akan masuk ke halaman **dashboard Firebase**, seperti yang ditampilkan pada halaman di samping ini.



1 Register app

Android package name ?

App nickname (optional) ?

Debug signing certificate SHA-1 (optional) ?

Required for Dynamic Links, and Google Sign-In or phone number support in Auth. Edit SHA-1s in Settings.

[Register app](#)

2 Download config file

3 Add Firebase SDK

4 Next steps

Go to docs



7. Pada project overview, klik **Add Android**, kemudian kamu akan masuk ke halaman baru..



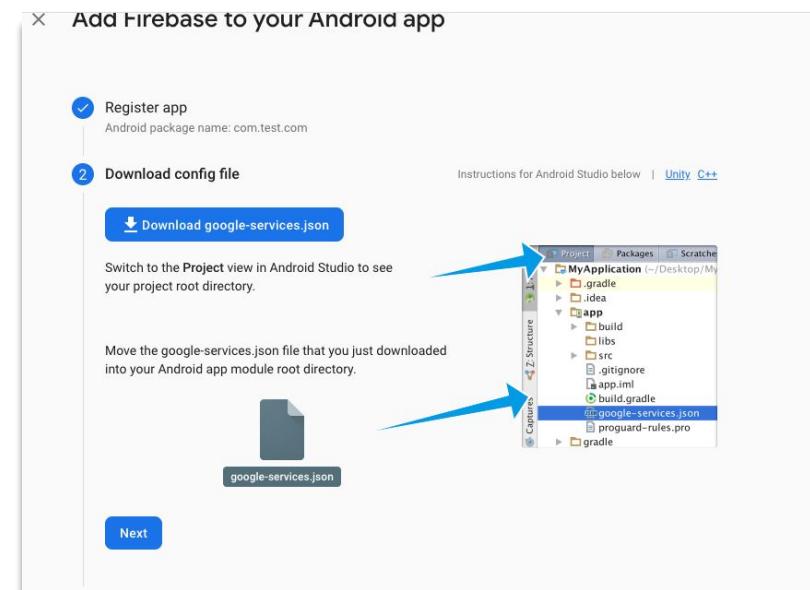
8. Masukan Android package name yang bisa kamu temukan  
**/android/app/src/main/AndroidManifest.xml.**
9. Kemudian masukan **App nickname.**
10. Masukan **debug signing certificate SHA-1** yang kamu bisa dapatkan dengan menjalankan command cd Android, gradlew signing Report.



```
SHA-256: 25:FC:EC:57:CD:39:28:39:5D:AC:FC:73:4C:13:02:DC:0D:AC:F2:CE:CF:71:0D:E6
:31:98:86:33:A5:75:28:9C
Valid until: Friday, March 3, 2051
-----
Variant: debugUnitTest
Config: debug
Store: /Users/161552.mikhael/.android/debug.keystore
Alias: AndroidDebugKey
MD5: E9:05:90:6A:CF:6D:A8:94:95:81:63:45:AC:DF:28:D7
SHA1: A9:FE:DE:37:CE:F4:31:DE:BE:5E:25:33:80:E1:EB:89:8C:7C:0A:AD
SHA-256: 25:FC:EC:57:CD:39:28:39:5D:AC:FC:73:4C:13:02:DC:0D:AC:F2:CE:CF:71:0D:E6
:31:98:86:33:A5:75:28:9C
Valid until: Friday, March 3, 2051
-----
Variant: releaseUnitTest
Config: debug
Store: /Users/161552.mikhael/.android/debug.keystore
Alias: AndroidDebugKey
MD5: E9:05:90:6A:CF:6D:A8:94:95:81:63:45:AC:DF:28:D7
SHA1: A9:FE:DE:37:CE:F4:31:DE:BE:5E:25:33:80:E1:EB:89:8C:7C:0A:AD
SHA-256: 25:FC:EC:57:CD:39:28:39:5D:AC:FC:73:4C:13:02:DC:0D:AC:F2:CE:CF:71:0D:E6
:31:98:86:33:A5:75:28:9C
Valid until: Friday, March 3, 2051
```



11. Lanjutkan dengan mendownload **google-service.json**
12. Letakkan google-service.json di lokasi  
**/android/app/google-services.json**





```
buildscript {
    dependencies {
        // ... other dependencies
        classpath 'com.google.gms:google-services:4.3.10'
        // Add me --- \\
    }
}
```

13. Tambahkan google-service ke **file /android/build.gradle**.



```
apply plugin: 'com.android.application'  
apply plugin: 'com.google.gms.google-services' // <- Add this line
```

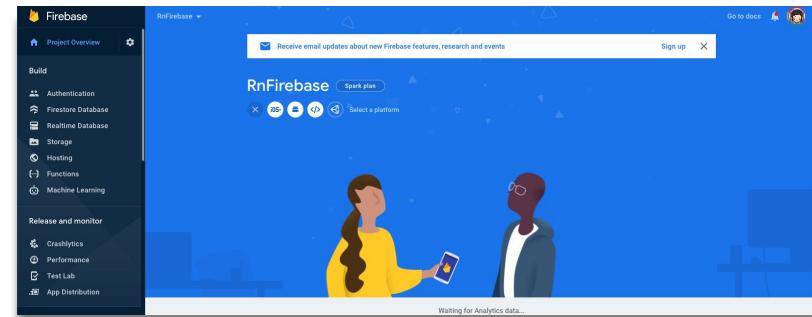
14. Terakhir tambahkan plugin google service ke lokasi **/android/app/build.gradle**
15. Sampai di sini, untuk setup Android sudah selesai. Yeay!



## 4. IOS Setup

Sama halnya dengan Android, kamu perlu melakukan koneksi App iOS-mu ke firebase project. Caranya:

1. Sama kayak Android, untuk iOS, kamu add iOS App pada dashboard Firebase. Pilih Add App kemudian pilih iOS.





2. Jalankan Xcode dan buka React Native project-mu.
3. Di packagename, kamu perlu mengisi bundle ID yang bisa kamu temukan di tab general di Xcode

x Add Firebase to your Apple app

1 Register app

Apple bundle ID ?

A text input field containing "com.company.appname", which is highlighted with a blue border.

App nickname (optional) ?

A text input field containing "My Apple app".

App Store ID (optional) ?

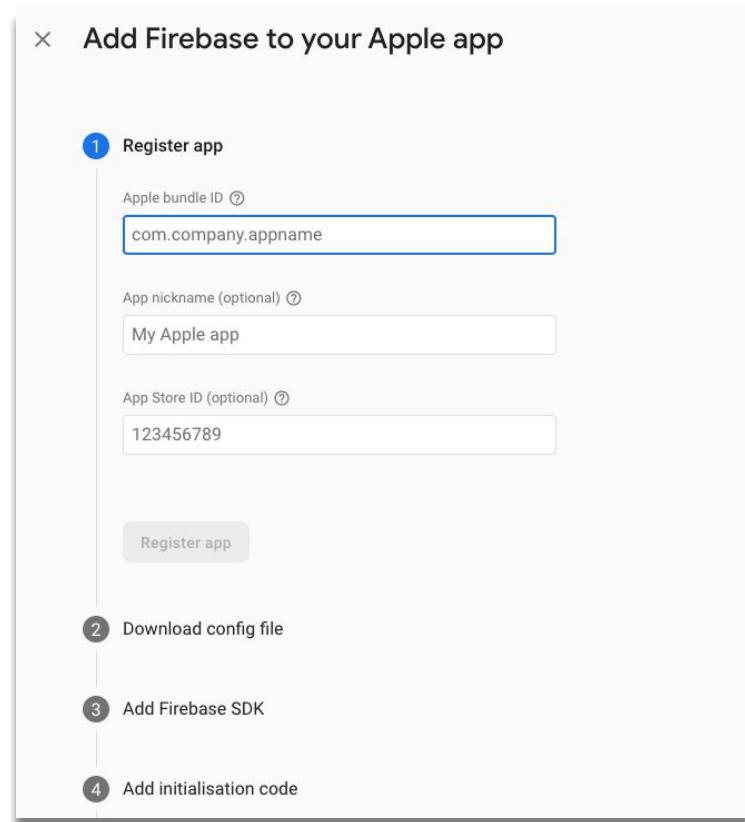
A text input field containing "123456789".

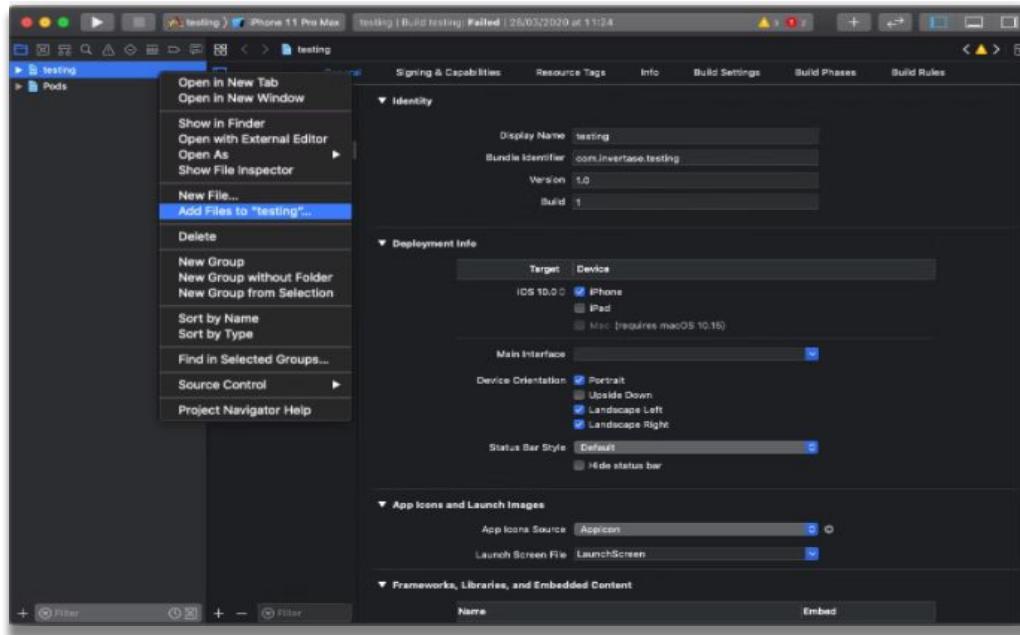
Register app

2 Download config file

3 Add Firebase SDK

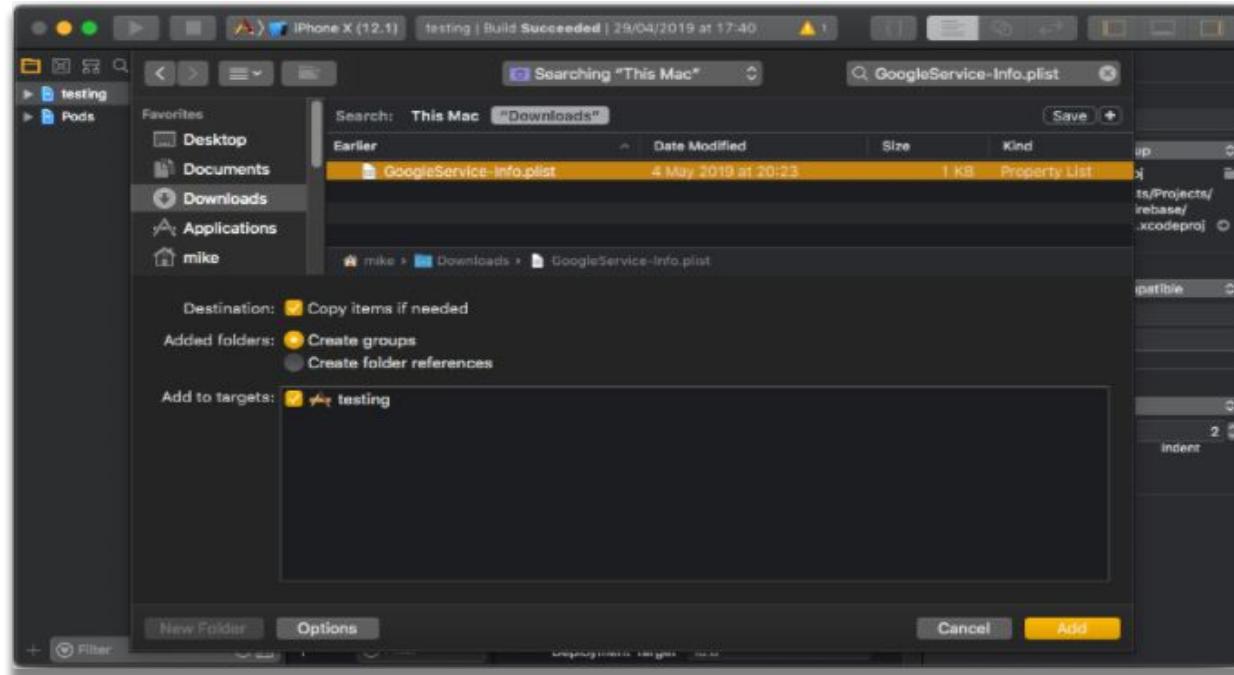
4 Add initialisation code





4. Setelah itu download config file bernama **GoogleService-Info.plist**

5. Pada tampilan Xcode, klik kanan pada Project Name dan add File to “Project Name”, seperti gambar di samping.



6. Pilih GoogleService-Info.plist dari Macbook-mu dan pastikan memiliki copy items if need, nyalakan checkbox-nya.



Selanjutnya kamu perlu melakukan configure firebase dengan iOS credentials. Caranya:

7. Buka file /ios/{projectName}/appDelegate.m di code editor kemudian tambahkan import Firebase SDK, di bagian paling atas file.

```
#import <Firebase.h>
```

8. Kemudian di antara didFinishLaunchingWithOptions method, tambahkan di paling atas.

```
(BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
    // Add me --- \/
    [FIRApp configure];
    // Add me --- \\
    // ...
}
```



Setelah semua langkah-langkah di atas telah selesai, kamu perlu menjalankan autolinking & rebuilding.

Untuk React Native version 0.60+, autolinking sudah otomatis, JADI tidak memerlukan manual linking. Jalankan command seperti pada gambar di samping.



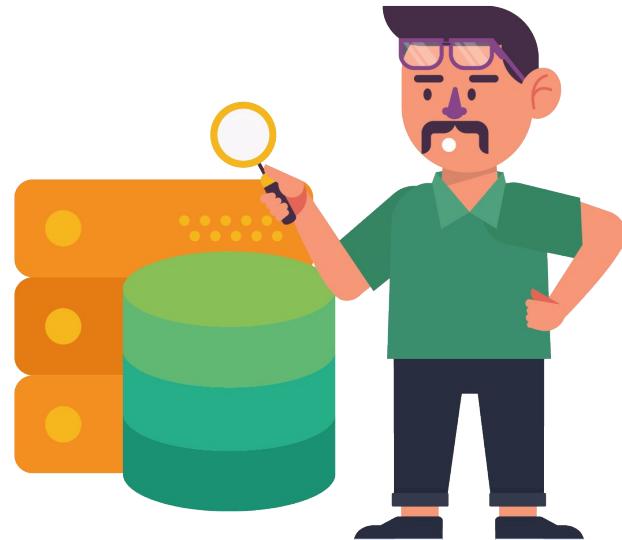
```
# Android apps
npx react-native run-android

# iOS apps
cd ios/
pod install --repo-update
cd ..
npx react-native run-ios
```



Sampai sini, kamu sudah berhasil untuk melakukan setup untuk Android dan iOS yaa. Tujuannya adalah agar kamu dapat menggunakan layanan yang dimiliki Firebase.

Cek official documentation-nya jika kamu terlewat atau ada yang kurang jelas, [di sini](#) ya!





Aplikasi yang kamu buat perlu dimonitoring, guys. Untuk itu, kamu bisa menggunakan **Firebase Crashlytic**.

Kayak apa? Langsung aja kita bahas yuk!





## Ini dia fungsi Firebase Crashlytic

Firebase menyediakan layanan Crashlytic yang berguna untuk memonitoring aplikasimu, apakah mengalami error/crash atau tidak.





## Kalau dipikir-pikir Firebase Crashlytics ini kayak tim quality control di pabrik ya~

Tim quality control adalah mereka yang bertanggung jawab untuk memonitoring hasil kerja karyawan sebuah perusahaan. Nah, hampir sama kan?

Ia dan Firebase Crashlytics sama-sama memastikan apakah pada sebuah produk yang dibuat terdapat kerusakan/kegagalan ataukah sudah sempurna.





Singkatnya Firebase Crashlytic membantumu untuk bisa memonitoring 3 aspek, yaitu:

- **Logs:** Jika aplikasi yang kamu buat error, maka bagian logs yang error akan dikirim bersamaan dengan laporan error.
- **Crash reports:** Semua error akan otomatis masuk ke dalam laporan error dan akan langsung dikirimkan.
- **Stack traces:** Walaupun bagian aplikasimu sudah dipulihkan, Javascript stack trace masih bisa dikirimkan. Yaitu berupa langkah yang mana yang menyebabkan error.





Dalam pembuatan aplikasi, crash adalah hal sangat mungkin terjadi.

Karena itu, penting untuk setiap developer untuk mengetahui **crash attributes** untuk mengetahui cara mengatasinya





### Apa saja yang termasuk crash attributes?

Banyak sekali yang bisa menjadi penyebab aplikasi crash, bukan hanya karena dari error code handling, tapi bisa juga dari fitur yang terlalu berat namun dijalankan di ponsel low spec.

Karena itulah kamu perlu melakukan tracing, yaitu mencari tahu hal apa saja yang menyebabkan crash.





## Bagaimana caranya?

Kita akan coba bikin simulasi untuk membuat aplikasi crash, sehingga memunculkan trace event pada Crashlytics dashboard.

Mari buat code seperti pada dokumentasi firebase di samping lebih dulu.

```
import React, { useEffect } from 'react';
import { View, Button } from 'react-native';
import crashlytics from '@react-native-firebase/crashlytics';

async function onSignIn(user) {
  crashlytics().log('User signed in.');
  await Promise.all([
    crashlytics().setUserId(user.uid),
    crashlytics().setAttribute('credits', String(user.credits)),
    crashlytics().setAttributes({
      role: 'admin',
      followers: '13',
      email: user.email,
      username: user.username,
    }),
  ]);
}

export default function App() {
  useEffect(() => {
    crashlytics().log('App mounted.');
  }, []);
}

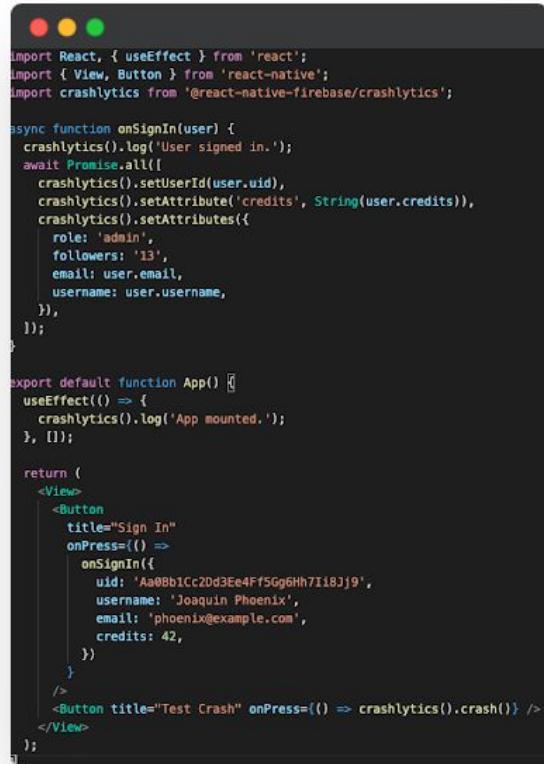
return (
  <View>
    <Button
      title="Sign In"
      onPress={() =>
        onSignIn({
          uid: 'Aa0Bb1Cc2Dd3Ee4Ff5Gg6Hh7Ii8Jj9',
          username: 'Joaquin Phoenix',
          email: 'phoenix@example.com',
          credits: 42,
        })
      }
    />
    <Button title="Test Crash" onPress={() => crashlytics().crash()} />
  </View>
);
```



Setelah itu, akan terlihat seperti pada code di samping.

Sebuah login eksekusi sederhana, di mana kita menambahkan setAttributes untuk mengetahui jika suatu waktu ada kejadian crash pada section login kita.

Button test crash memanggil crashlytics.crash() yang bertujuan agar kita dapat dengan sengaja membuat aplikasi crash.



```
import React, { useEffect } from 'react';
import { View, Button } from 'react-native';
import crashlytics from '@react-native-firebase/crashlytics';

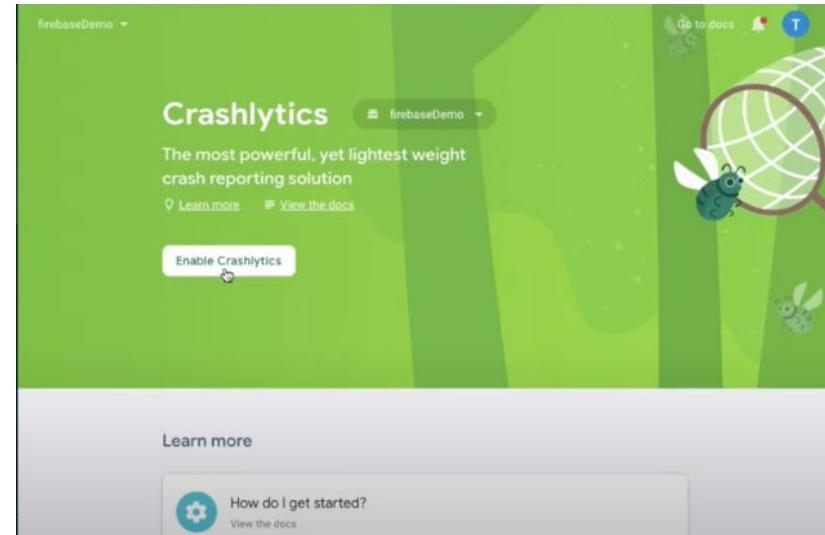
async function onSignIn(user) {
  crashlytics().log('User signed in.');
  await Promise.all([
    crashlytics().setUserId(user.uid),
    crashlytics().setAttribute('credits', String(user.credits)),
    crashlytics().setAttributes({
      role: 'admin',
      followers: '13',
      email: user.email,
      username: user.username,
    }),
  ]);
}

export default function App() {
  useEffect(() => {
    crashlytics().log('App mounted.');
  }, []);
}

return (
  <View>
    <Button
      title="Sign In"
      onPress={() =>
        onSignIn({
          uid: 'Aa0Bb1Cc2Dd3Ee4Ff5Gg6Hh7Ii8Jj9',
          username: 'Joaquin Phoenix',
          email: 'phoenix@example.com',
          credits: 42,
        })
      }
    />
    <Button title="Test Crash" onPress={() => crashlytics().crash()} />
  </View>
);
```



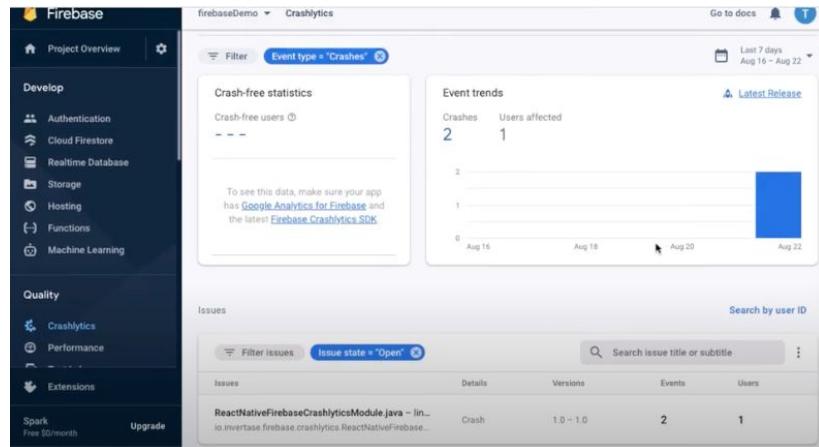
Buka firebase project-mu lalu pilih bagian tab Crashlytics. Klik enable Crashlytics dan tunggu beberapa saat hingga tombol berubah menjadi view dashboard.





Kalau kamu sudah berhasil melakukan crash maka event crash akan tampil di dashboard seperti pada gambar disamping.

Dashboard di samping berguna untuk menganalisa crash pada aplikasi kita.



The screenshot shows the Firebase Crashlytics dashboard for a project named "firebaseDemo". The top navigation bar includes links for "Project Overview", "Crashlytics", "Analytics", "Performance", "Extensions", "Spark", and "Upgrade". The main dashboard features several key metrics:

- Crash-free statistics:** Crash-free users: 2
- Event trends:** Crashes: 2, Users affected: 1
- Issues:** A table showing open issues with the following data:

Issue	Details	Versions	Events	Users
ReactNativeFirebaseCrashlyticsModule.java – lin...	Crash	1.0 ~ 1.0	2	1

Below the dashboard, there is a note: "To see this data, make sure your app has Google Analytics for Firebase and the latest Firebase Crashlytics SDK."



Sampai di sini, berarti kamu sudah berhasil memanfaatkan Crashlytic dari Firebase dengan baik.

By the way, **untuk menilai seberapa penting Crashlytics tergantung dari jumlah user aplikasimu.**

Jika user aplikasi sedikit maka biasanya Crashlytics akan dirasa kurang berguna, sebaliknya kalau **aplikasimu memiliki banyak user**, maka **kamu akan memerlukan Crashlytics** supaya bisa memberikan experience yang baik ke user.





Firebase memiliki fungsi analisa yang dapat membantumu memonitoring aplikasi. FYI, servis ini disediakan oleh Google lho~

Ada yang tahu apa sebutan untuk layanan ini?





## Firebase Analytics

Yaps, Firebase adalah produk layanan yang dibuat oleh Google. Google memberikan service analytics gratisan yang biasa kita kenal sebagai **Google Analytics**, yang fungsinya adalah agar kita bisa **memonitoring aplikasi** kita.



**Google Analytics**  
for Firebase



## Apa sih kegunaan Firebase Analytics?

Firebase Analytics berguna untuk mengumpulkan data kebiasaan dari aplikasi. Dua hal yang utama adalah :

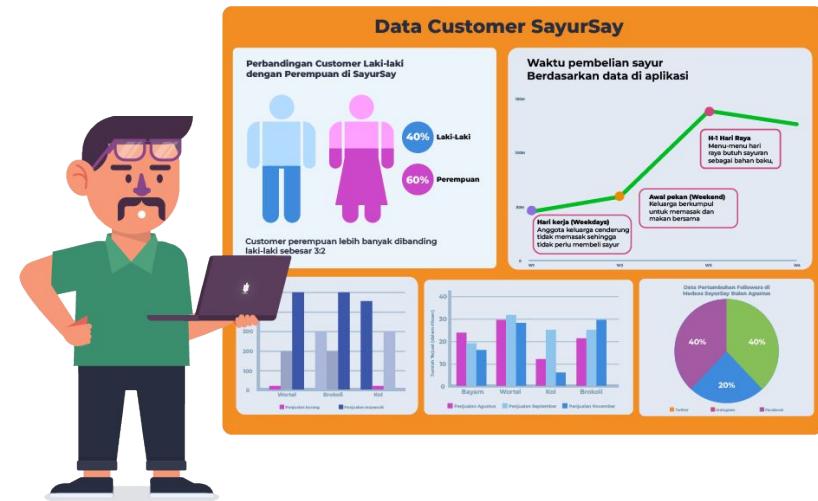
- **Events:** Yaitu segala hal yang terjadi pada aplikasi yang kamu buat, seperti user action, system events ataupun jika terjadi error.
- **User properties:** Sebuah atribut yang harus kamu tentukan untuk menggambarkan segmentasi dari latar belakang pengguna aplikasi, seperti bahasa ataupun lokasi aplikasi digunakan.





Dengan analytics kamu jadi **bisa tahu kebiasaan user ketika menggunakan aplikasimu.**

Misalnya jam berapa mereka menggunakan aplikasi, berapa lama mereka menggunakan aplikasi, screen mana yang sering mereka kunjungi, tombol apa yang mereka klik dan berapa lama, dan masih banyak lagi.





```
# Install & setup the app module
yarn add @react-native-firebase/app

# Install the analytics module
yarn add @react-native-firebase/analytics

# If you're developing your app using iOS, run this command
cd ios/ && pod install
```

## Terus, bagaimana ya caranya melakukan Analytics?

Ya tentu saja, langkah yang harus kamu lakukan adalah memasang (install) Firebase/analytics ke project React Native kamu.

Nah, untuk langkah lengkapnya, lihat arahan pada gambar di atas ya, guysss.



## Selanjutnya, kita akan bahas tentang Custom Events

Salah satu fitur utama yang membuatmu membutuhkan analytics adalah Events.

Events berguna untuk melacak apa saja yang terjadi pada aplikasimu, misalnya ketika user melakukan navigation screen atau melakukan click terhadap button hingga submit form.





## Langkah membuat Custom Events:

- Buatlah komponen sederhana seperti di samping. Tujuannya adalah untuk mengukur seberapa sering function dari button di samping ditekan.
- Tambahkan analytics().logEvent() dengan memberikan nama event untuk setiap function akan kita ukur.

Contoh seperti di samping “Track Events”

```
const App = () => {

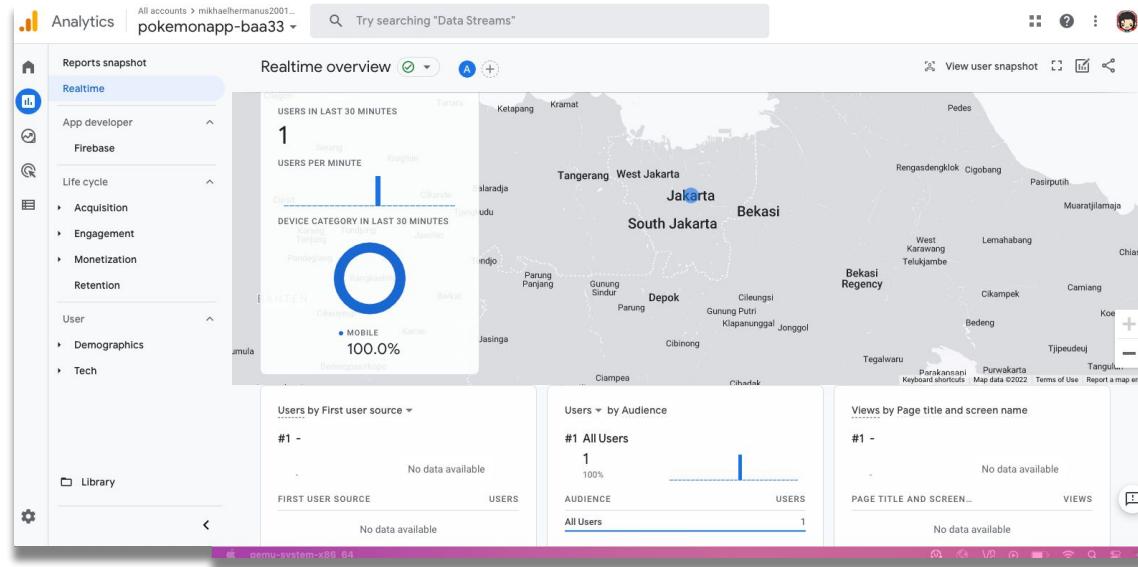
  const onSetupCloudMessaging = async () => {
    const authStatus = await messaging().requestPermission();
    const enabled =
      authStatus === messaging.AuthorizationStatus.AUTHORIZED ||
      authStatus === messaging.AuthorizationStatus.PROVISIONAL;

    if (enabled) {
      console.log('Authorization status:', authStatus);
    }
  }

  useEffect(()=>{
    onSetupCloudMessaging()
  }, [])
}
```



- Buka Google Console-mu dan lihatlah ke bagian analytics. Perhatikan, ada bagian perubahan yang telah terlacak di dashboard analytics. Ini menandakan ada user aktif di 30 menit lalu. Kamu bisa menggunakan UI Google Analytics dengan menekan button View More in Google Analytics.



- Kemudian untuk mendapatkan lebih banyak informasi, pergilah ke bagian Real Time di tab analytics ataupun melalui Google Analytics View. Di sana kamu bisa mendapatkan informasi di mana lokasi aplikasimu digunakan.



## Selanjutnya, Screen Tracking

Fitur lain yang nggak kalah penting adalah Screen Tracking. Ini berguna untuk mengetahui kebiasaan user, misalnya untuk mengetahui screen apa yang sering mereka kunjungi, dan sebagainya.





## Bagaimana membuat Screen Tracking?

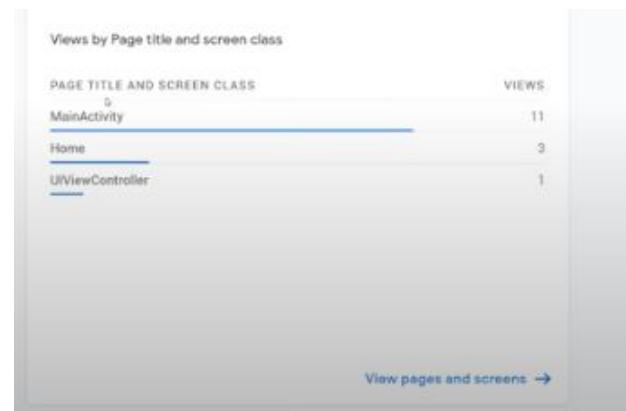
- Buatlah code sederhana di samping untuk mengetahui kebiasaan user, di component screen yang kamu telah kamu buat.

```
const onLogScreenView = async () =>{
  try {
    await analytics().logScreenView({
      screen_name: 'Home',
      screen_class: 'Home',
    })
  } catch (error) {
    console.log(error)
  }
}

useEffect(()=>{
  onLogScreenView()
})
```



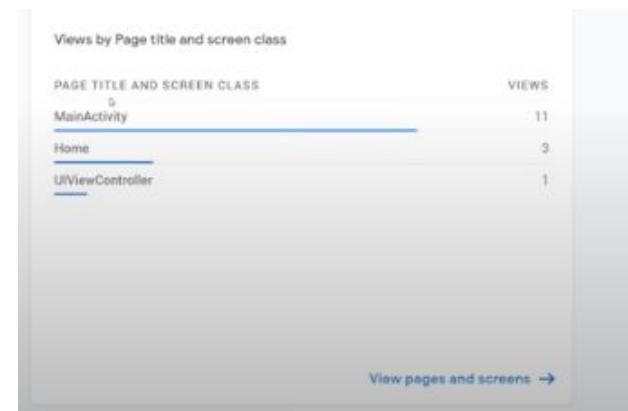
- Lihatlah hasil pada dashboard screen tracking pada tab Real time pada bagian page Title dan screen name.





By the way, gak perlu risau kalau apa yang kamu lakukan nggak terlacak di dashboard, karena menurut dokumentasi, bahkan Google disebut membutuhkan 12-24 jam untuk bisa melakukan update data –atau bisa lebih cepat jika traffic rate yang banyak.

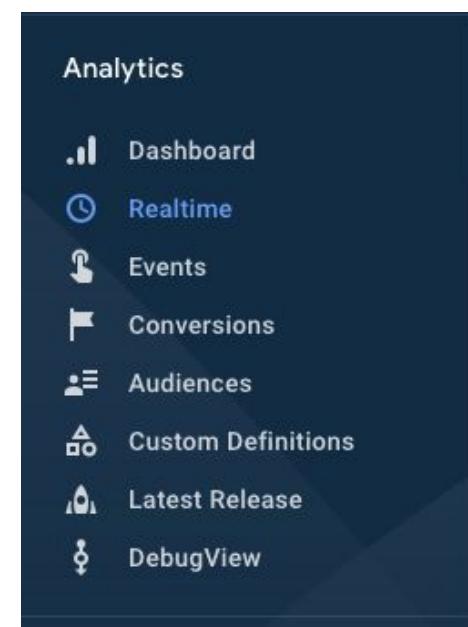
Dalam kasus ini karena kita hanya menggunakan satu aplikasi per satu user, maka update akan lebih lama.





Nah, itulah beberapa hal tentang analytics yang bisa kamu pelajari.

Sebenarnya masih banyak lagi yang bisa kamu telusuri, misalnya menu-menu di samping, namun tentu akan membutuhkan waktu yang juga. Selain itu, masing-masing menu akan lebih kelihatan fungsinya kalau kamu sudah punya application user yang banyak.





**Selanjutnya, kita masuk ke materi Remote Notifications.**

**Apa sih Remote Notifications itu? Apakah berbeda dengan sistem notifikasi pada umumnya?**





tring.. tring.. eh ada notifikasi chat  
ayang 😍

Karena E-mail, Instagram, Whatsapp, semua aplikasi yang ada di ponselmu pasti mengirimkan notifikasi kan? Jadi pasti kamu sudah nggak asing lagi dengan yang namanya **notifikasi atau pemberitahuan.**





### Ini Ihoo Remote Notifications

Yang hebat dari Firebase adalah ia memiliki fitur notifikasi yang memungkinkanmu untuk melakukan pengiriman notifikasi ke aplikasimu.

Eh bagaimana tuh ya maksudnya? Yuk kita pahami dengan penggunaan contoh.





## Hal pertama dalam Remote Notification adalah Permission for Notification

Tanpa panjang lebar, ayo kita coba lakukan remote notification dari Firebase di samping.

Hal pertama yang perlu kamu lakukan adalah **memberikan izin (permission)** untuk Android sehingga kita nggak perlu menambahkan command apa-apa lagi.

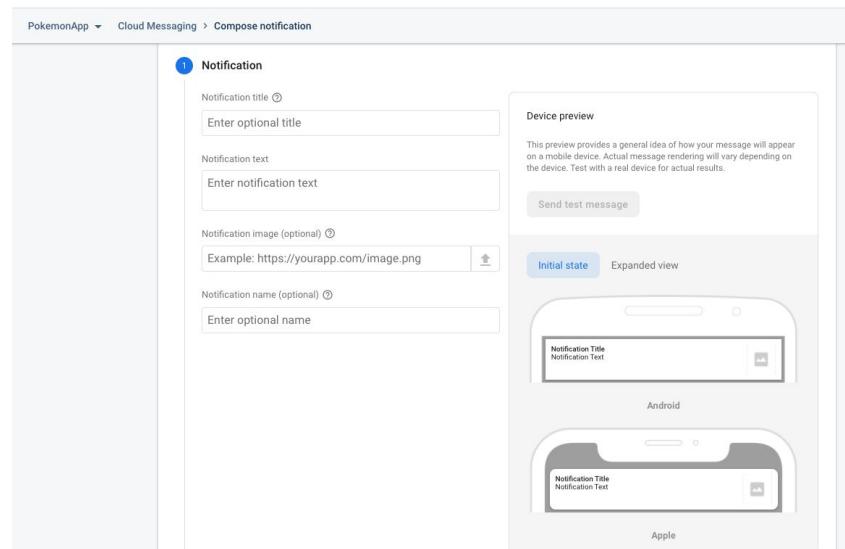
Sementara itu, untuk iOS kita perlu menambahkan code berikut:

```
const App = () => {  
  
  const onSetupCloudMessaging = async () => {  
    const authStatus = await messaging().requestPermission();  
    const enabled =  
      authStatus === messaging.AuthorizationStatus.AUTHORIZED ||  
      authStatus === messaging.AuthorizationStatus.PROVISIONAL;  
  
    if (enabled) {  
      console.log('Authorization status:', authStatus);  
    }  
  }  
  
  useEffect(()=>{  
    onSetupCloudMessaging()  
  }, [])
```



## Receiving Notification

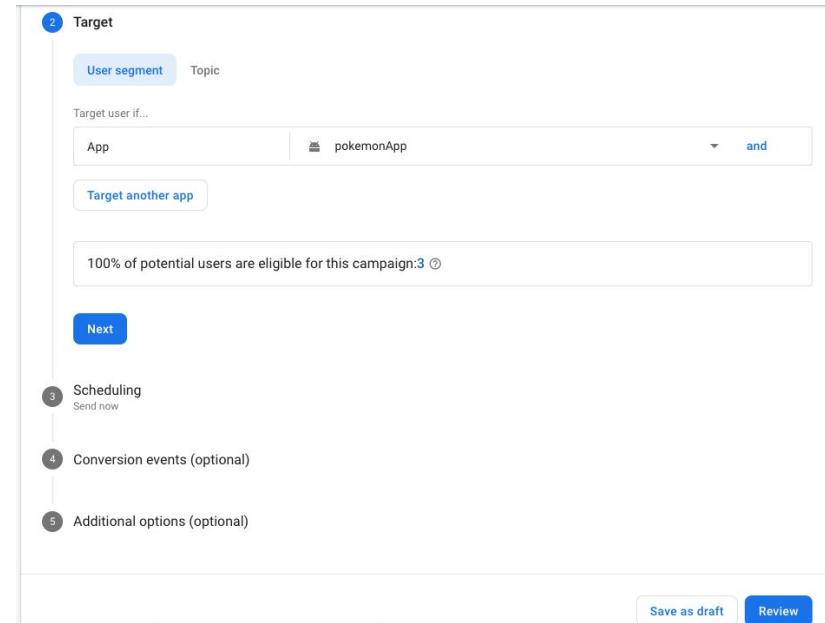
- Buka Firebase console kemudian klik bagian cloud messaging. Klik start your first messaging, hingga kalian menemukan seperti tampilan di samping



The screenshot shows the 'Compose notification' interface in the Firebase Cloud Messaging section of the Firebase console. The main area is titled 'Notification' and contains four input fields: 'Notification title', 'Notification text', 'Notification image (optional)', and 'Notification name (optional)'. To the right, there's a 'Device preview' section with a 'Send test message' button. Below it, two mobile device mockups show the notification layout: 'Initial state' and 'Expanded view' for both 'Android' and 'Apple' platforms. The Android preview shows a standard notification bar with 'Notification Title' and 'Notification Text'. The Apple preview shows a similar notification with rounded corners.

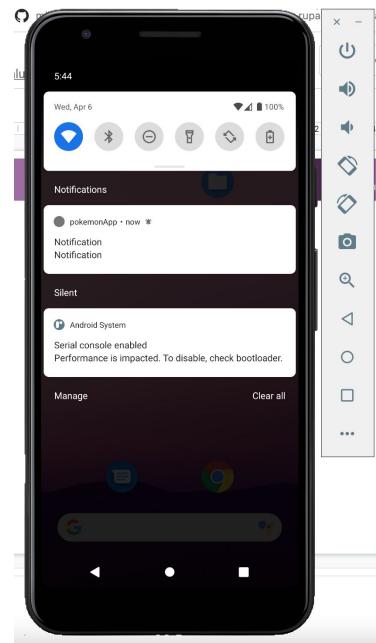


- Langkah selanjutnya, pilihlah target aplikasi untuk dikirimkan notification. Untuk saat ini pilihlah hanya packagename dari aplikasamu, kemudian select scheduling untuk dikirim sekarang. Jika semua setup sudah sesuai klik review dan send message..





- Jika sudah, maka kamu akan mendapatkan notifikasi yang akan tampil di emulator/device kalian, seperti gambar di samping.

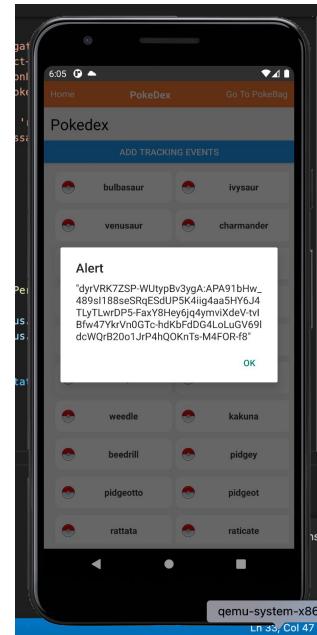




Misalnya kamu dalam kondisi staging, masih uji coba dan kamu nggak ingin notifikasi dikirimkan ke semua user yang punya aplikasi, maka kamu bisa melakukan **notification testing ke specify device dengan menggunakan token**.

```
const getToken = async() =>{
  const token = await messaging().getToken()
  alert(JSON.stringify(token))
}

useEffect(()=>{
  onSetupCloudMessaging()
  getToken()
}, [])
```





### Umm.. bagaimana ya cara mengirimkan notifikasi dengan token?

Nah jadi, pada tampilan New Notification, isi Notification Title dan Notification Text kemudian pada bagian Device Preview klik Send Test Message.

Di sini kamu akan diminta mengisi FCM token yg kalian dapatkan dari Function Get Token yang kamu buat. Jika sudah, hajar dengan klik Test

#### Test on device

You can test this campaign by entering or selecting the [FCM registration tokens](#) of your development device below.

Add an FCM registration token

dyrVRK7ZSP-WUtypBv3ygA:APA91bHw\_489sI188seSRqESdUP5K4iig4...

Cancel

Test



Oke, yang tadi kamu lakukan adalah cara mengirimkan notification via Firebase console.

Lalu bagaimana kalau kamu memerlukan back-end atau API dalam mengirimkan notifikasi?

Dokumentasi Firebase menyediakan cara agar backend atau API dapat berkomunikasi melalui server, ini disebut Via Rest.

### Via REST

If you are unable to use a [Firebase Admin SDK](#), Firebase also provides support for sending messages to devices via a POST request:

```
POST https://fcm.googleapis.com/v1/projects/myproject-b5ae1/messages:send HTTP/1.1
Content-Type: application/json
Authorization: Bearer ya29.ElgKBGN2Ri_Uz...HnS_uNreA

{
  "message": {
    "token": "bk3RNwTe3H0:CI2k_HHwgIpoDKCIZvvDMExUdFQ3P1...",
    "data": {},
    "notification": {
      "body": "This is an FCM notification message!",
      "title": "FCM Message"
    }
  }
}
```

To learn more about the REST API, view the [Firebase documentation](#), and select the "REST" tab under the code examples.



## Handling Interaction

Sekarang kamu akan berkenalan dengan handling interaction dalam notification, di antaranya 3 jenis yaitu:

1. Foreground
2. Background
- 3 .Quit

### Message handlers

The device state and message contents determines which handler will be called:

	Foreground	Background	Quit
<b>Notification</b>	<code>onMessage</code>	<code>setBackgroundMessageHandler</code>	<code>setBackgroundMessageHandler</code>
<b>Notification + Data</b>	<code>onMessage</code>	<code>setBackgroundMessageHandler</code>	<code>setBackgroundMessageHandler</code>
<b>Data</b>	<code>onMessage</code>	<code>setBackgroundMessageHandler (see below)</code>	<code>setBackgroundMessageHandler (see below)</code>



## Foreground State Message

Untuk mengetahui apakah kamu akan berinteraksi dengan notification di saat app dalam keadaan foreground, yaitu saat aplikasi sedang dalam keadaan aktif, dengan menggunakan function `onMessage()`

### Message handlers

The device state and message contents determines which handler will be called:

	Foreground	Background	Quit
<b>Notification</b>	<code>onMessage</code>	<code>setBackgroundMessageHandler</code>	<code>setBackgroundMessageHandler</code>
<b>Notification + Data</b>	<code>onMessage</code>	<code>setBackgroundMessageHandler</code>	<code>setBackgroundMessageHandler</code>
<b>Data</b>	<code>onMessage</code>	<code>setBackgroundMessageHandler (see below)</code>	<code>setBackgroundMessageHandler (see below)</code>



## Background & Quit state message

Background adalah keadaan ketika kamu menerima notification ketika posisi sedang minimize.

Sedangkan untuk quit adalah keadaan ketika aplikasi sedang dalam kondisi kill application.

### Message handlers

The device state and message contents determines which handler will be called:

	Foreground	Background	Quit
<b>Notification</b>	<code>onMessage</code>	<code>setBackgroundMessageHandler</code>	<code>setBackgroundMessageHandler</code>
<b>Notification + Data</b>	<code>onMessage</code>	<code>setBackgroundMessageHandler</code>	<code>setBackgroundMessageHandler</code>
<b>Data</b>	<code>onMessage</code>	<code>setBackgroundMessageHandler (see below)</code>	<code>setBackgroundMessageHandler (see below)</code>

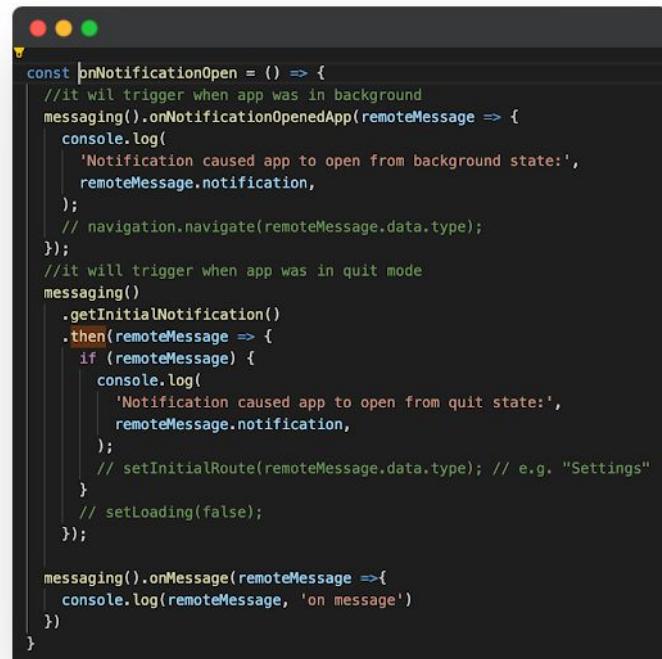


### Lanjutan Background & Quit~

Untuk lebih jelasnya, yuk coba buat file di app.js dengan menerapkan handling interaction seperti di samping.

Berdasarkan dokumentasi, kamu bisa mengelola data yang kamu terima dari payload remote Message.

Perhatikan komen di samping, setiap function yang akan trigger berdasarkan state application.



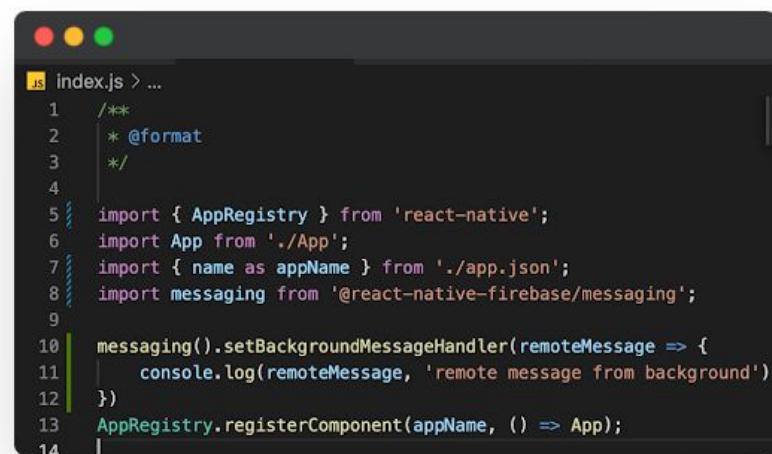
```
const [onNotificationOpen = () => {}] = useState(() => {
  // it will trigger when app was in background
  messaging().onNotificationOpenedApp(remoteMessage => {
    console.log(
      'Notification caused app to open from background state:',
      remoteMessage.notification,
    );
    // navigation.navigate(remoteMessage.data.type);
  });
  // it will trigger when app was in quit mode
  messaging()
    .getInitialNotification()
    .then(remoteMessage => {
      if (remoteMessage) {
        console.log(
          'Notification caused app to open from quit state:',
          remoteMessage.notification,
        );
        // setInitialRoute(remoteMessage.data.type); // e.g. "Settings"
      }
      // setLoading(false);
    });
  messaging().onMessage(remoteMessage => {
    console.log(remoteMessage, 'on message')
  })
})
```



Untuk background state, kamu akan menuliskan code di index.js.

Kenapa di situ ya?

Karena, file yang **pertama kali dijalankan** oleh React Native adalah file index.js. bukan app.js

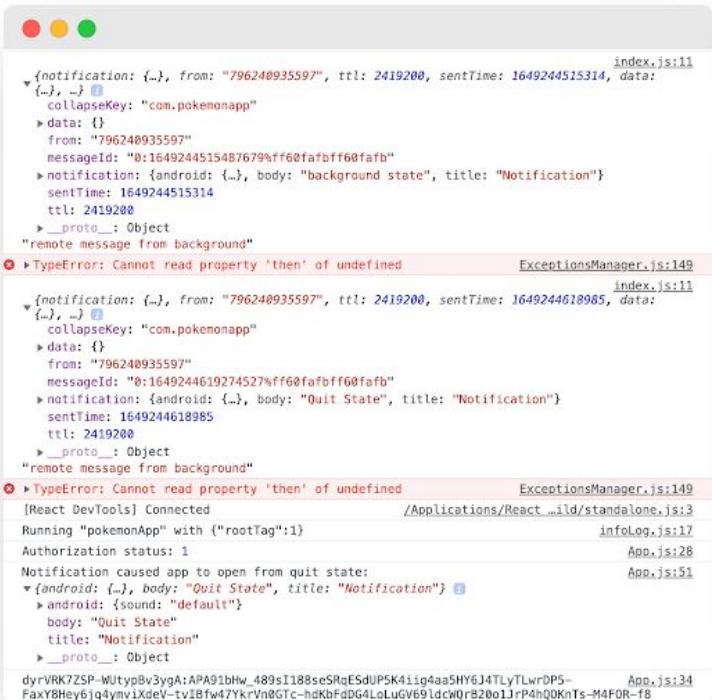


```
index.js > ...
1  /**
2   * @format
3   */
4
5  import { AppRegistry } from 'react-native';
6  import App from './App';
7  import { name as appName } from './app.json';
8  import messaging from '@react-native-firebase/messaging';
9
10 messaging().setBackgroundMessageHandler(remoteMessage => {
11   console.log(remoteMessage, 'remote message from background')
12 })
13 AppRegistry.registerComponent(appName, () => App);
14
```



Selanjutnya, lakukanlah send message melalui Firebase console sesuai dengan state masing-masing untuk mencoba function trigger.

Untuk mengetahui apakah tiap function berhasil ter-trigger, kamu dapat mengeceknya dari console di browser yang tampil.



```
index.js:11
{
  notification: {...}, from: "796240935597", ttl: 2419200, sentTime: 1649244515314, data: {...}, ...
    collapseKey: "com.pokemonapp"
  > data: {}
  > from: "796240935597"
  > messageId: "0:1649244515487679%ff6fafbff6fafb"
  > notification: {android: {}, body: "background state", title: "Notification"}
  > sentTime: 1649244515314
  > ttl: 2419200
  > __proto__: Object
"remote message from background"
✖ TypeError: Cannot read property 'then' of undefined ExceptionsManager.js:149
index.js:11
{
  notification: {...}, from: "796240935597", ttl: 2419200, sentTime: 1649244618985, data: {...}, ...
    collapseKey: "com.pokemonapp"
  > data: {}
  > from: "796240935597"
  > messageId: "0:1649244619274527%ff6fafbff6fafb"
  > notification: {android: {}, body: "Quit State", title: "Notification"}
  > sentTime: 1649244618985
  > ttl: 2419200
  > __proto__: Object
"remote message from background"
✖ TypeError: Cannot read property 'then' of undefined ExceptionsManager.js:149
[React DevTools] Connected /Applications/React .../standalone.js:3
Running "pokemonApp" with {"rootTag":1}
infoLog.js:17
Authorization status: 1 App.js:28
App.js:51
Notification caused app to open from quit state:
  > {android: {}, body: "Quit State", title: "Notification"} 
    > android: {sound: "default"}
      > body: "Quit State"
        > title: "Notification"
      > __proto__: Object
d4vrVK7ZP-WUtypBv3ygA:APA91bHw_489sI188seSRqESdUPSK4iig4aa5HY6J4TLwvDP5- App.js:34
FaxY8Hey6jq4ymviXdeV-tvIBfw47YkrVn0Gtc-hdKbfDg4LoLuGV69ldcWQrB20o1JrP4hQ0KnTs-M4FDR-f8
App.js:28
```



```
const onNotificationOpen = () => [
  // it will trigger when notification open
  messaging().onNotificationOpenedApp(remoteMessage => {
    console.log(
      'Notification caused app to open from background state:',
      remoteMessage.notification,
    );
    // navigation.navigate(remoteMessage.data.type);
  });
];
```

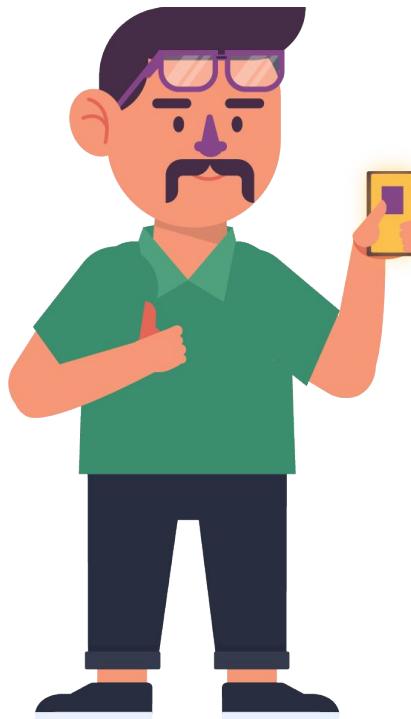
## Handling onNotificationOpenApp()

Misalnya kamu mau melakukan suatu perintah saat aplikasi masuk berdasarkan notification, contoh kita ingin berpindah ke spesifik screen pada saat klik notification, maka kamu perlu menambahkan `onNotificationOpen()` function. Kamu bisa melakukan apapun ketika notifikasi di buka.



Nah, sekarang kamu sudah berhasil melakukan remote notification. Kamu juga sudah bisa melakukan handling interactive dan melakukan trigger berdasarkan state aplikasi.

Dengan adanya remote notification, diharapkan kamu dapat membuat aplikasi jadi lebih menarik lagi, terutama untuk aplikasi yg memerlukan notification.



# Terima Kasih!



Next Topic

loading...