React Native Introduction
Real Time Database

Chapter 7 - Topic 3

Selamat datang di Chapter 7 Topic 3 online course React Native dari Binar Academy!

Hallo!

Ketemu lagi di Chapter 7. Kalau di topic 2 kamu sudah belajar tentang cara merilis aplikasi iOS di Apple Store, maka di Chapter 7 Topic 3 ini, kamu akan belajar tentang **database aktual (real time database).**

Seperti apa sih yang disebut dengan real time database? Mari kita bahas per bagian.

Semangat semangat, guys!

Detailnya, kita bakal bahas hal-hal berikut ini:

- Mengenali Firebase database dengan real time database.
- Cara melakukan konfigurasi dengan Firebase real time database.
- Cara mengimplementasikan penyimpanan dan pengambilan data dari database.

Menyimpan dan mengambil data yang dibuat oleh pengguna kini umum dilakukan pada aplikasi seluler dan web.

Karena itu, ada berbagai layanan yang memberi pengembang seluler dan web kemampuan untuk menyimpan data.

Di antara layanan tersebut ada yang bernama Google Firebase

Firebase merupakan **BaaS (backend-as-a-service),** yang memungkinkan pengembang web dan seluler untuk melakukan tugas backend umum seperti otentikasi pengguna dan membuat database tanpa perlu pemeliharaan..

Ini lho alasan Firebase banyak digunakan oleh pengembang~

Seperti yang sudah pernah kita pelajari, Firebase telah banyak digunakan dalam aplikasi, karena dianggap memiliki semua fitur yang cukup untuk mengembangkan aplikasi sederhana dan kecil tanpa server.

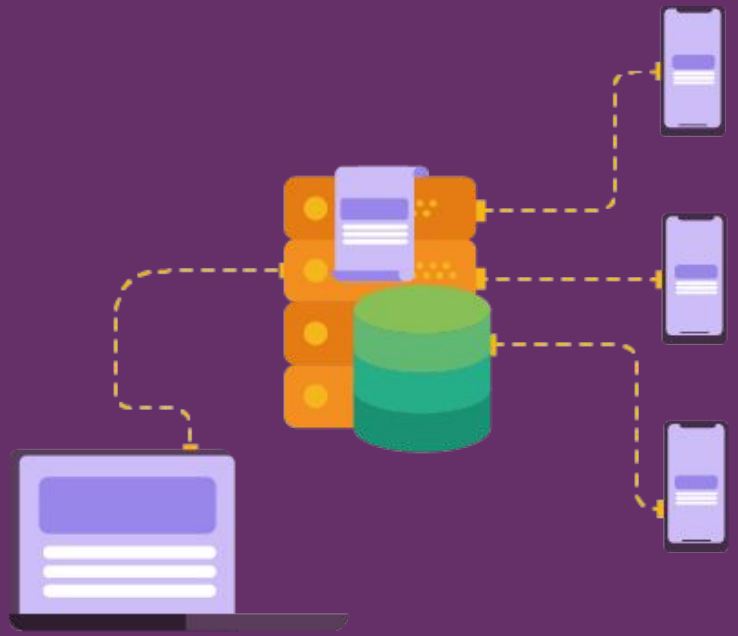
Nah, pada topic kali ini, kamu akan belajar cara menyimpan dan menerima data dari Firebase, serta membuat aplikasi sederhana menggunakan Firebase.

Hmm.. bikin aplikasi sederhana dengan menggunakan Firebase, siapa takut?

Cuss, kita mulai!

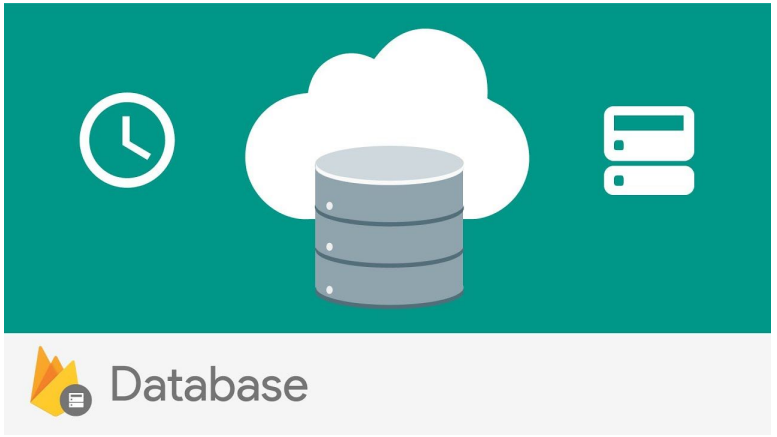
Sebelum benar-benar memulai, mari berkenalan dulu dengan istilah **realtime database.**

Kayak apa sih yang dimaksud dengan realtime database?

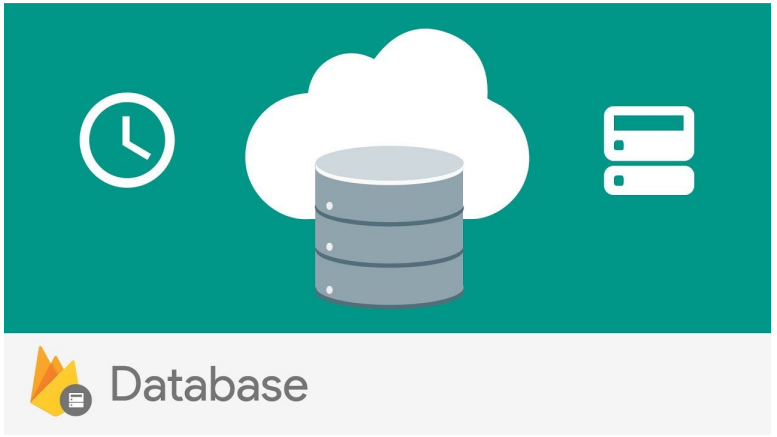


Begini ceritanya guys...

Firebase Realtime Database adalah database yang dihosting di Cloud. Data disimpan sebagai JSON dan **disinkronkan secara aktual alias real time ke setiap klien yang terhubung, saat itu juga.**

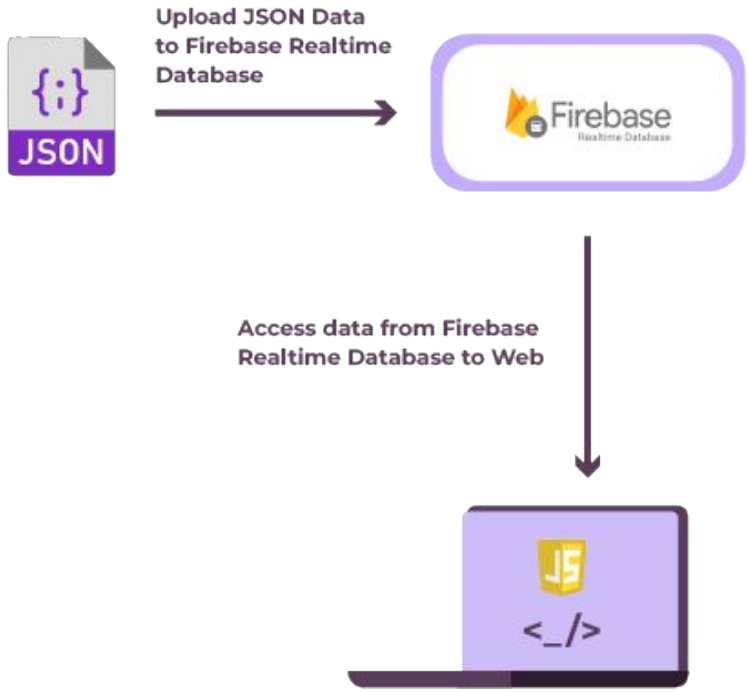


Saat kamu bikin aplikasi lintas platform dengan platform Apple, Android, atau SDK JavaScript, maka semua klienmu akan **berbagi satu instance Realtime Database** dan secara otomatis dapat menerima pembaruan dengan data terbaru.



Mengenal Real Time Database

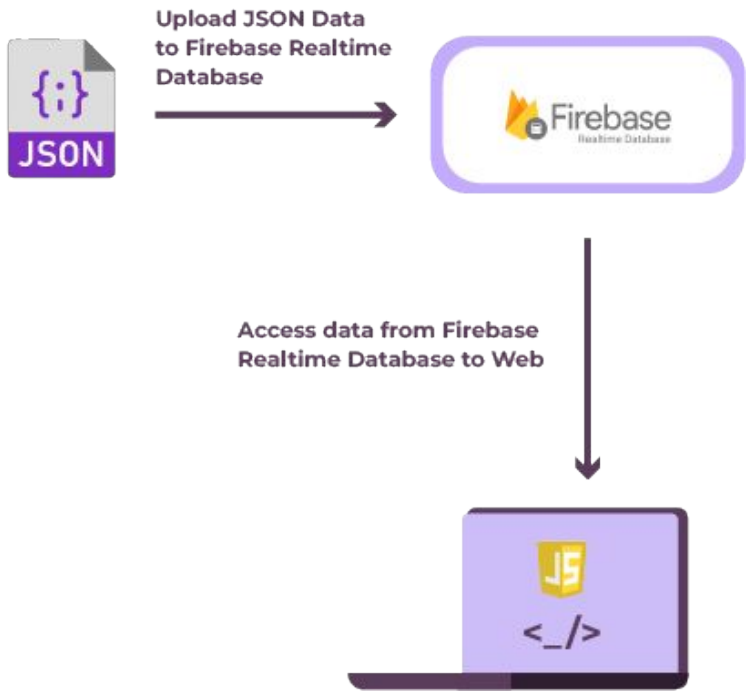
Realtime Database adalah database NoSQL. Dengan demikian **ia memiliki optimasi dan fungsionalitas yang berbeda** dibandingkan dengan database relasional.



Mengenal Real Time Database

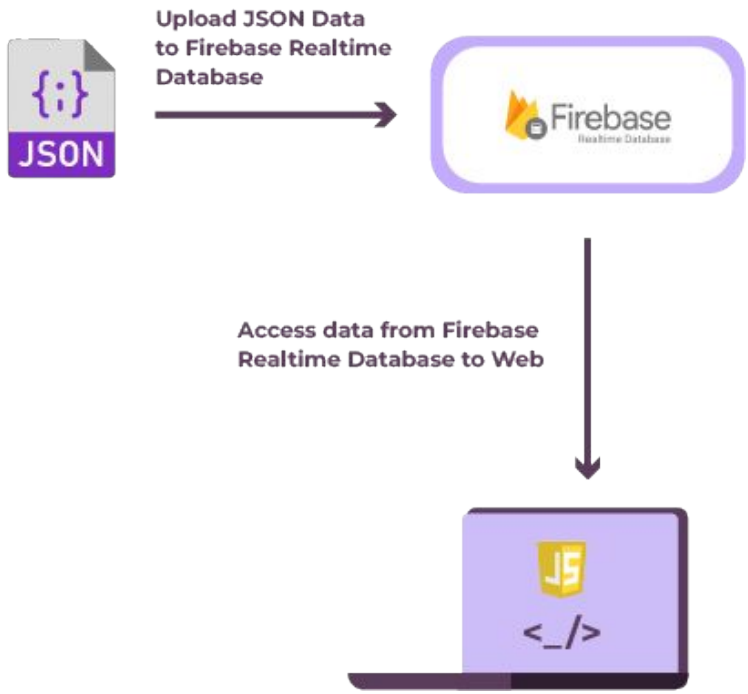
Realtime Database API dirancang hanya untuk **mengizinkan operasi yang dapat dijalankan dengan cepat.**

Hal ini memungkinkanmu membangun pengalaman real time yang luar biasa, dan dapat melayani jutaan pengguna tanpa mengorbankan daya tanggap.



Oke, kita udah bahas poiin penting yang membedakan Firebase dengan database lainnya yaa, yaitu **Firebase merupakan NoSQL database.**

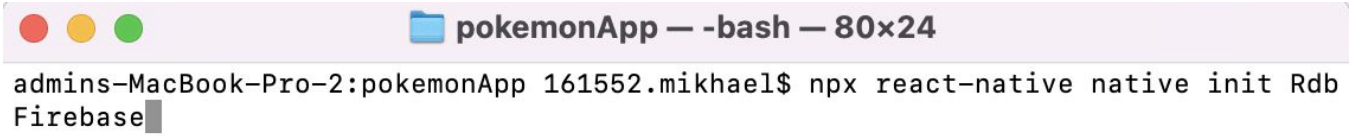
Tapi bagaimana ya contoh NoSQL ini?



Oke let’s go, kita cari tahu!

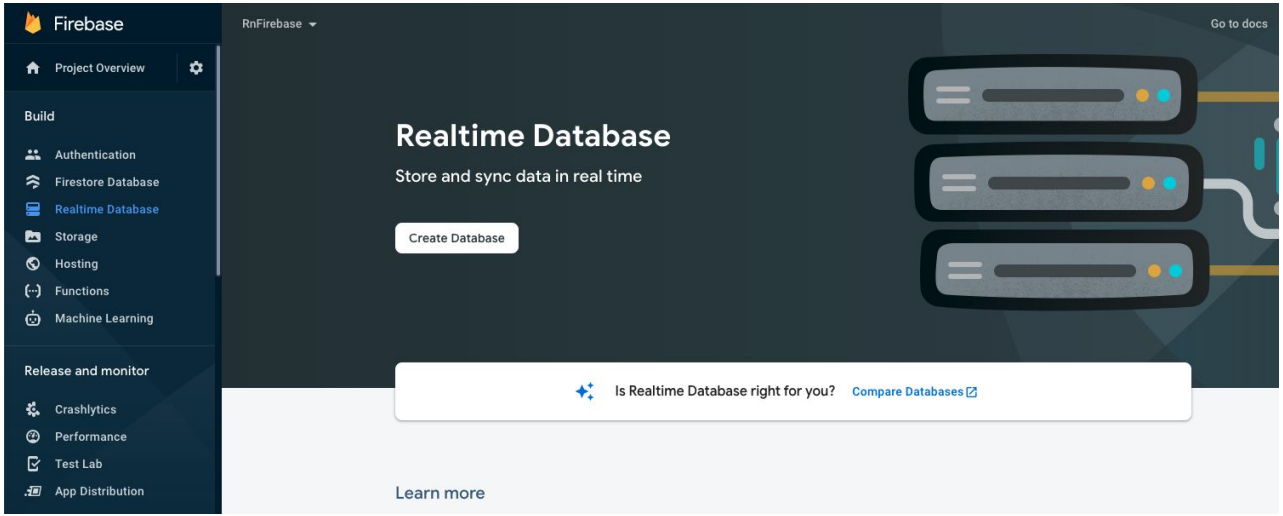
Kita coba melakukan konfigurasi Firebase real time database, yuk!

1. Seperti biasa, **buatlah project react-native baru** atau dengan menggunakan yang sudah ada, dengan command npx react-native init project_name



2. Selanjutnya, **lakukanlah konfigurasi Firebase** ke project react-native yang sudah kamu pelajari di Chapter 6.

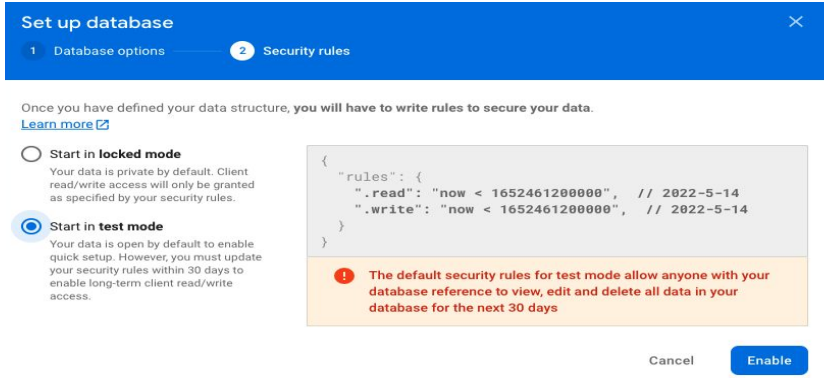
3. Kemudian bukalah Firebase console dan klik database seperti pada gambar di samping, kemudian **klik create database.**



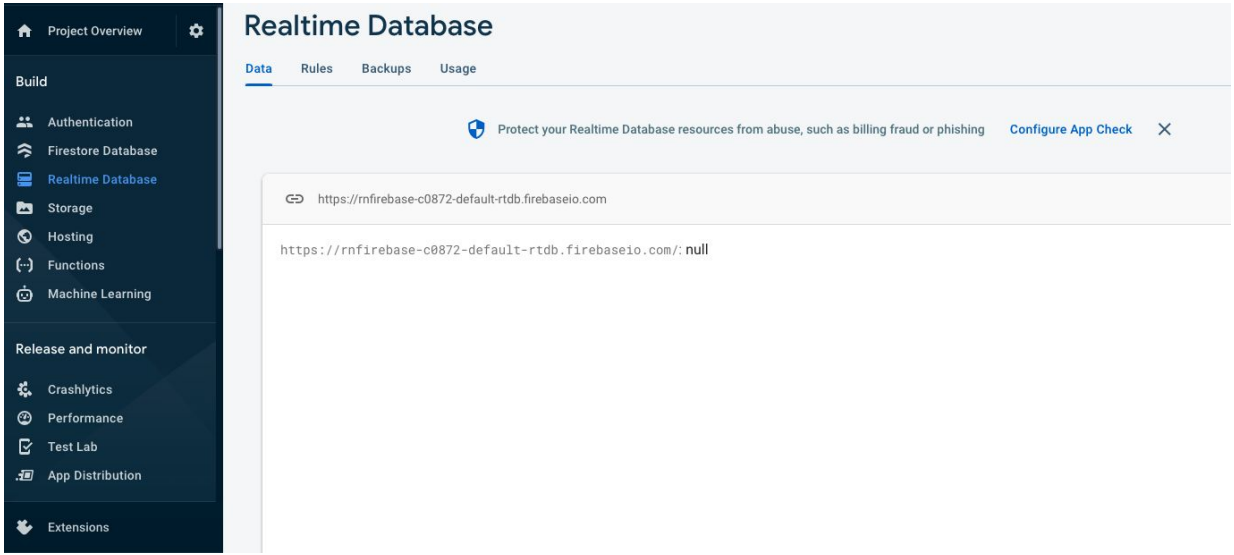
4. Setelah klik, database-mu akan diminta **memasukkan beberapa set up database**. Saat di security rules, kamu bebas memilih test mode atau locked mode. Apa yang membedakan?

   Bedanya test mode akan nggak aktif setelah beberapa hari, sedangkan locked mode sudah versi yang akan kita pakai di production.

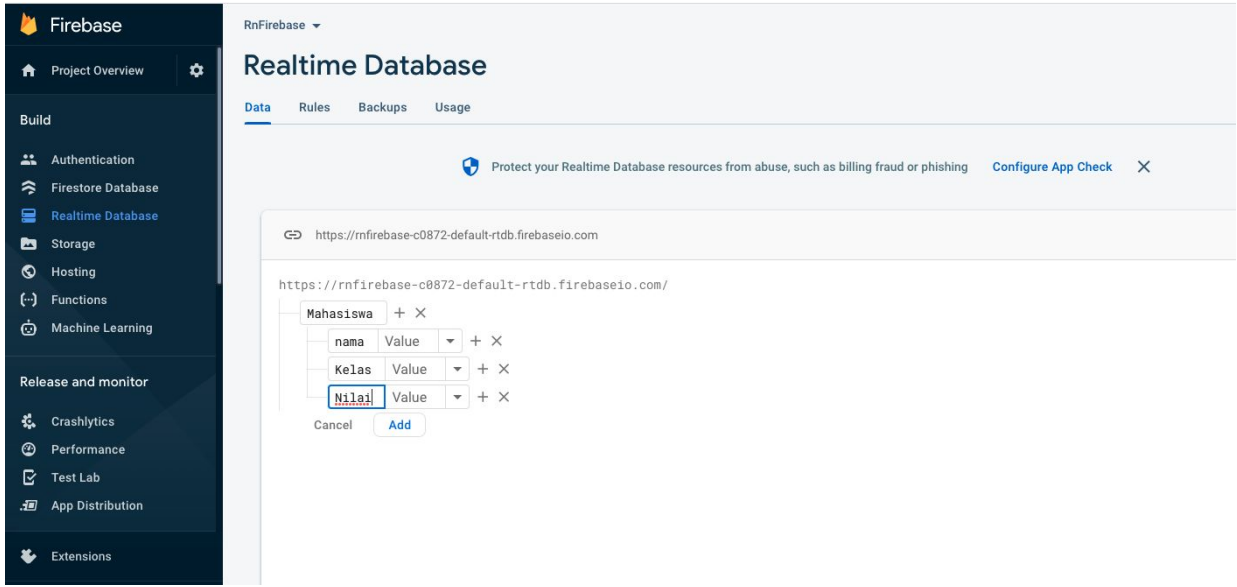
   Untuk sekarang pilihlah test mode karena tujuannya kita hari ini hanya untuk belajar dahulu.



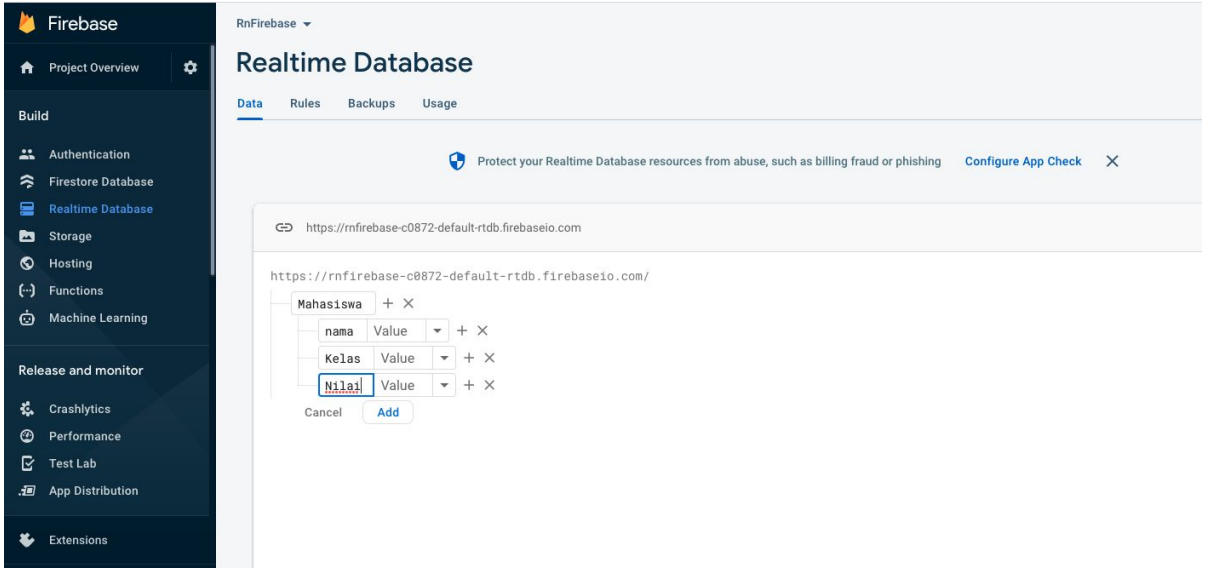
5. Selanjutnya kamu akan dibawa ke halaman seperti pada tampilan di bawah. Kalau kamu perhatikan ini merupakan **dashboard untuk database-mu,** di mana kamu bisa menambah, menghapus, dan mengedit database.



6. Arahkan mouse ke kanan dari link, kemudian kamu akan menemukan **+ button seperti contoh di bawah,** sehingga kamu bisa membuat object Mahasiswa yang memiliki 3 properties nama, kelas, dan nilai.



7. Di sini kamu bisa tahu bahwa NoSql mengizinkanmu melakukan write data tanpa harus menggunakan sql query. Di Firebase realtime database, kamu hanya perlu menggunakan **klik + atau - (untuk menghapus properties) sehingga kamu dapat membuat JSON dan object semaumu.**



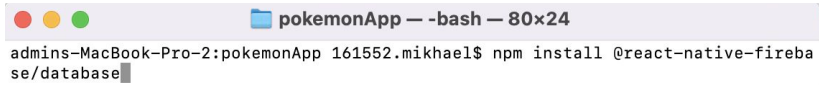
Menghubungkan Project ke Real Time Database

Oke selanjutnya, kamu akan membuat koneksi antara project ke realtime database yang sudah kamu buat di Google Console.

Yuk, langsung saja ikuti langkah-langkah pada link berikut ini

Pertama: Install Firebase Real Time database!

Yaps! Hal pertama yang perlu kamu lakukan adalah **menginstal Firebase Real Time Database Module** dulu dengan command seperti di samping.



Setelahnya, bikin database sederhana dulu yukk~

Buatlah database bebas dan sederhana yang memiliki reference (/pokebag) dan property seperti di samping.

Eh, perhatikan database di samping deh, setiap property name memiliki id unique yang berguna untuk membedakan tiap recordnya.

Baca sinyal pas PDKT? Pentinglah! Sama pentingnya kayak baca data dari database

Hihi, namanya juga pengembang aplikasi, ya harus bisa baca data dari database dong.

Oke, **buatlah satu komponen sederhana di mana kamu akan menampilkan data dari Firebase.** Lalu lakukan import module firebase realtime database.

Import database dari '@react-native-firebase/database';

kemudian buatlah function sederhana seperti di samping.

```
const fetchPokeBagData = () => {
    setLoading(true)
    const reference = database().ref('/pokeBag');
    reference.on('value', snapshot => {
        GetData(snapshot.val())
        setLoading(false)
    });
}
```

```
const GetData = (data) => {
    let keyFirebase = []
    keyFirebase = Object.keys(data)
    setKey(keyFirebase)
    setPokeBag(data)
}
```

Selanjutnya~

Seperti pada gambar di samping, kamu akan **mengakses database dari reference bernama "/pokeBag".**

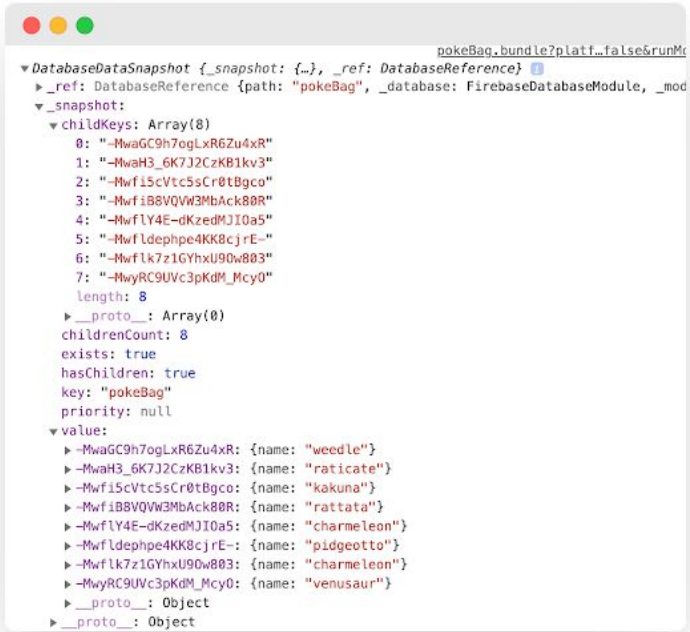
Gunakan function database().ref('refrencename'). kemudian value dari setiap record dapat kita akses dengan variabel "snapshot"

```
const fetchPokeBagData = () => {
    setLoading(true)
    const reference = database().ref('/pokeBag');
    reference.on('value', snapshot => {
        GetData(snapshot.val())
        setLoading(false)
    });
}
```

Membaca Data dari Database

Kalau hasilnya sesuai, coba **tambahkan console.log(snapshot)** untuk memastikan data yang kamu terima sudah berhasil diakses.

Hasil console akan menampilkan data seperti di samping, di mana data memiliki value yang dibungkus dalam objectKeys



Selanjutnya, kamu perlu mengakses setiap data dengan **mengakses object key.**

Javascript memungkinkanmu untuk mengambil key dari object yang tersimpan, dengan function Object.keys(data).

Simpanlah key / id ke state bertipe array.

```
const GetData = (data) => {
    let keyFirebase = []
    keyFirebase = Object.keys(data)
    setKey(keyFirebase)
    setPokeBag(data)
}
```

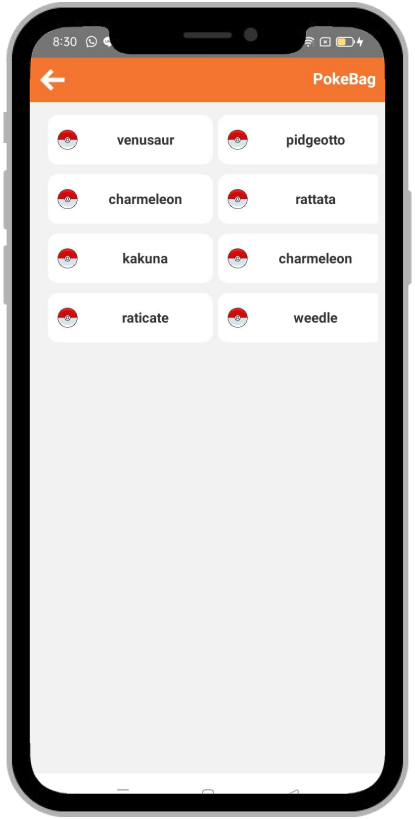
Kemudian **lakukanlah console.log(key)** untuk melihat apakah key yang kamu simpan sudah sesuai seperti gambar di atas atau belum. Selanjutnya kamu dapat mengakses properties name dengan **melakukan looping key[‘name’]** dengan menggunakan JavaScript notation.

Buatlah tampilan untuk merender data dari Firebase seperti pada code di atas.
Buatlah sekreatif mungkin pada contoh yang sederhana.

```
loading ?
    <ActivityIndicator color={colors.orangeQonstanta} />
    :
    <FlatList
        style={{ margin: 8 }}
        numColumns={2}
        data={key}
        renderItem={({ item, index }) => <PokemonCard name={pokeBag[item]?.name} item={item} pokeBag rightAction={()
        columnWrapperStyle={{
            margin: 5
        }}
        listKey={(item, index) => `_key${index.toString()}`}
    />
```

Setelah kamu render maka akan muncul tampilan sederhana seperti di samping.

Naaah kalau sudah sampai di sini berarti kamu sudah berhasil membaca suatu data dari Firebase ke dalam aplikasimu. Congratz, guys!



Membaca data sudah bisa, sekarang saatnya menulis data~

Selanjutnya kamu akan mencoba melakukan penambahan data Firebase.

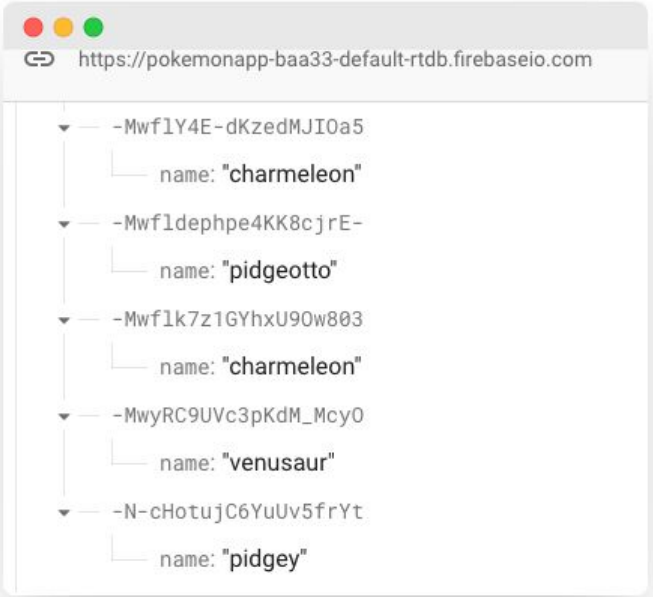
Sudah siap?

Menulis Data dari Database

Sama kayak pada saat membaca data, kamu harus **menambahkan reference database dulu**, baru kemudian dengan **function .push(), kamu dapat menambahkan object** ke dalam database, seperti di samping

```
const savePokemon = () => {
    const reference = database().ref('/pokeBag');
    try {
        reference.push({ name: 'pidgey' })
    } catch (error) {
        alert(error)
    }
```

Kalau sudah berhasil, maka data akan berhasil ditambahkan ke dalam Firebase, seperti gambar di samping.



Menghapus data dari database

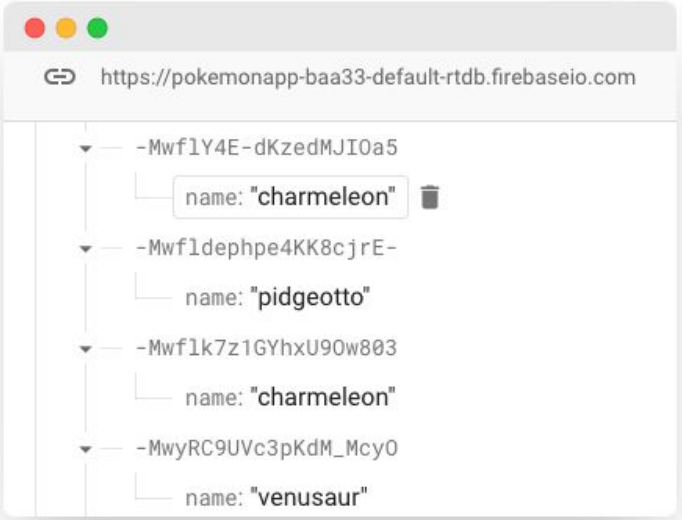
Selanjutnya, kamu akan belajar **menghapus** ~~kenangan~~ **data berdasarkan id** yang terdapat pada database.

Buatlah kode seperti di samping. Kemudian, fungsi remove() bisa kamu pakai untuk menghapus data dalam Firebase, dengan memberikan **`ref('/refrenceName/id').remove()`**

```
const removePokemon = async () => {
    try {
        await database().ref(`/pokeBag/${id}`).remove();
        fetchPokeBagData()
        setModalVisible(false)
        Alert.alert('Succes', `Berhasil Menghapus ${pokeBag[id].name}`)
    } catch (error) {
        Alert.alert('Oops', error)
    }
}
```

Jeng jeng!!

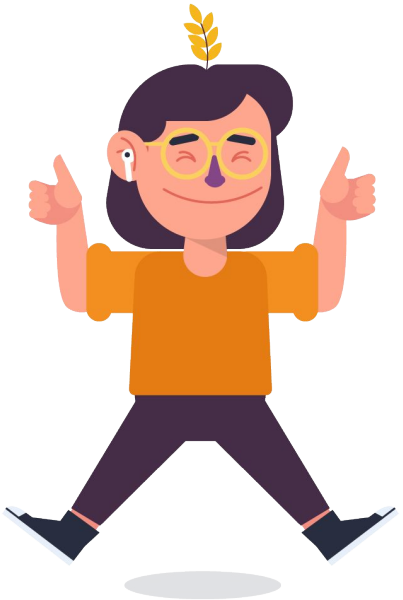
Data yang terdapat pada Firebase akan terhapus deh. Misalnya tadi kamu menambahkan pidgey, maka pada gambar di samping kamu juga menghapus pidgey tersebut.



Saatnya review!

Sampai di sini, kamu telah berhasil mempelajari bagaimana menggunakan real time database dengan Firebase.

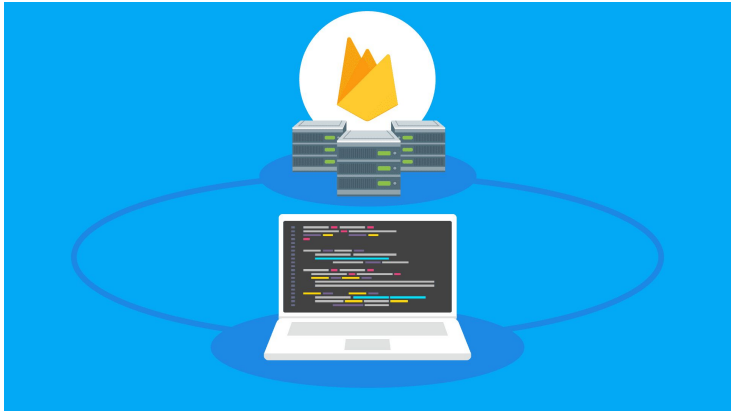
Selanjutnya kalau kamu membuat web/mobile yang bentuk datanya nggak terlalu rumit, maka kamu sudah bisa memakai Firebase untuk membantu pekerjaanmu deh! :)



Review

Realtime Database dikirim dengan SDK seluler dan web, sehingga kamu dapat melakukan build aplikasi tanpa server.

Kamu juga dapat mengeksekusi kode backend yang merespons peristiwa yang dipicu oleh databasemu **menggunakan** Cloud Functions for Firebase yaaa.



Saatnya kita

Quiz!

BINAR

1. Apa yang membedakan firebase real time database, dengan database lainnya?

A. Security firebase lebih baik dibanding database SQL

B. Firebase merupakan sdk yang hanya support untuk memwadahi mobile application

C. Firebase merupakan penyimpanan NoSql berbeda dengan database Seperti SQL yang masih menggunakan query

BINAR

2. Function berikut yang berguna untuk mengakses data dari firebase adalah...

A. database().ref().
B. database().ref().read
C. database().ref().write

BINAR

3. Apa kegunaan setiap row data memiliki id/key ?

A. untuk menjaga keamanan data

B. untuk menandakan bahwa data tersebut berasal dari firebase

C. untuk membedakan satu data dengan yang lainnya.

BINAR

4. Apa bentuk data dari hasil read data dari firebase?

A. Array Object

B. JSON Object

C. ArrayList

BINAR

5. Apa keuntungan dari Realtime Database API?

A. Memungkinkan pengguna mengirim data secara cepat dengan ukuran aplikasi yang tidak besar.

B. Memungkinkan pengguna mendapatkan data secara realtime tanpa mengorbankan daya tanggap.

C. Memungkinkan untuk melakukan operasi yang dilakukan dengan cepat ataupun yang lebih lama.

Terima Kasih!

Next Topic

loading...

BINAR