M. Bahrul Ulum



Cloud Firestore

**Pengenalan Cloud Firestore**

Baru-baru ini Google merilis sebuah database berbasis dokumen sebagai bagian dari keluarga Firebase bernama Firestore. Firestore merupakan database yang memberikan kemudahan bagi developer untuk menyimpan data, melakukan sinkronisasi, dan mengambil data ke aplikasi mobile mereka dengan dukungan offline yang berkualitas.

Cloud Firestore adalah database yang fleksibel dan skalabel untuk pengembangan seluler, web, dan server di Firebase dan Google Cloud Platform. Seperti Firebase Realtime Database, Cloud Firestore membuat data Anda tetap terhubung di aplikasi klien melalui listener realtime dan menawarkan dukungan secara offline untuk seluler dan web. Dengan begitu, Anda dapat membuat aplikasi yang responsif dan mampu bekerja tanpa harus bergantung pada latensi jaringan atau koneksi Internet. Cloud Firestore juga menawarkan integrasi yang lancar dengan produk Firebase dan Google Cloud Platform lainnya, termasuk Cloud Functions.

Firestore didesain untuk mempermudah developer melakukan sinkronisasi data ke cloud, melakukan query yang kompleks, dan menghapus batas 100.000 perangkat yang terhubung secara bersamaan di Realtime Database.

Firestore ingin mempermudah hidup develoepr agar tak perlu lagi bersusah payah membangun database backend-nya sendiri. Library client yang sudah siap pakai ini sudah dapat melakukan sinkronisasi data otomatis dengan data replikasi lintas benua sehingga data akan semakin aman.

Dengan menggunakan Cloud Firestore, developer dapat dengan mudah menyimpan dan sinkronisasi data aplikasinya untuk skala global. Beberapa fitur dari Cloud Firestore adalah sebagai berikut:

* Dokumen dan pengumpulan dengan *query* yang kuat
* SDK iOS, Android, dan web dengan akses data *offline*
* Sinkronisasi data secara *real-time*
* Replikasi data secara otomatis dan berbagai area dengan konsistensi yang kuat
* SDK server Node, Python, Go, dan Java

Bagi developer yang sudah pernah menggunakan Firebase mungkin sudah tahu bahwa sebetulnya sudah ada layanan *database* dari Firebase yaitu Firebase Realtime Database. Namun dengan Cloud Firestore developer akan dapat menikmati beberapa kelebihannya seperti *query* yang lebih bagus dan data yang lebih terstruktur.

**Kemampuan Utama**

* **Fleksibilitas**

Model data Cloud Firestore mendukung struktur data yang hierarkis dan fleksibel. Simpan data Anda ke dalam dokumen yang tersusun dalam koleksi. Selain subkoleksi, dokumen dapat berisi objek bertingkat yang kompleks.

* **Pembuatan kueri yang ekspresif**

Di Cloud Firestore, Anda dapat menggunakan kueri untuk mengambil masing-masing dokumen tertentu atau semua dokumen dalam koleksi yang sesuai dengan parameter kueri Anda. Kueri Anda dapat meliputi beberapa filter berantai dan menggabungkan penyaringan dan pengurutan. Kueri juga diindeks secara default, sehingga performa kueri sebanding dengan ukuran kumpulan hasil, bukan kumpulan data.

* **Update realtime**

Seperti Realtime Database, Cloud Firestore menggunakan sinkronisasi data untuk mengupdate data pada perangkat yang terhubung. Namun, Cloud Firestore juga dirancang untuk membuat kueri pengambilan 1 kali yang sederhana secara efisien.

* **Dukungan offline**

Cloud Firestore menyimpan data yang digunakan secara aktif oleh aplikasi Anda, sehingga aplikasi dapat menulis, membaca, mendeteksi, dan melakukan kueri data meskipun perangkat sedang offline. Saat perangkat kembali online, Cloud Firestore akan menyinkronkan semua perubahan lokal kembali ke Cloud Firestore.

* **Dirancang sesuai paket**

Cloud Firestore menghadirkan infrastruktur Google Cloud Platform yang terbaik kepada Anda: replikasi data multi-region secara otomatis, jaminan konsistensi yang kuat, batch operasi yang bersifat menyeluruh, dan dukungan transaksi nyata. Kami merancang Cloud Firestore untuk menangani beban kerja database yang tersulit dari aplikasi terbesar di dunia.

**Bagaimana cara kerjanya?**

Cloud Firestore adalah database NoSQL yang dihosting di cloud dan dapat diakses langsung melalui SDK asli oleh iOS, Android, dan aplikasi web Anda. Selain REST dan RPC API, Cloud Firestore juga tersedia di Node.js, Java, Python, dan Go SDK yang asli.

Setelah model data NoSQL Cloud Firestore, simpan data Anda dalam dokumen yang berisi pemetaan kolom terhadap nilai. Dokumen ini disimpan dalam koleksi yang berisi container untuk dokumen Anda, yang dapat digunakan untuk mengatur data dan membuat kueri. Dokumen ini mendukung berbagai jenis data, mulai dari string dan angka sederhana, hingga objek yang kompleks dan bertingkat. Anda juga dapat membuat subkoleksi dalam dokumen dan membangun struktur data hierarkis yang berskala sesuai dengan database. Model data Cloud Firestore mendukung struktur data yang paling sesuai untuk aplikasi Anda.

Selain itu, pembuatan kueri di Cloud Firestore bersifat ekspresif, efisien, dan fleksibel. Buatlah kueri dangkal untuk mengambil data pada tingkat dokumen tanpa perlu mengambil keseluruhan koleksi atau subkoleksi bertingkat. Tambahkan pengurutan, penyaringan, dan batasan pada kueri atau cursor untuk memberi nomor pada hasil Anda. Tambahkan listener realtime untuk menjaga data di aplikasi Anda tetap terkini, tanpa harus mengambil keseluruhan database setiap kali ada update. Dengan menambahkan listener realtime ke aplikasi, Anda akan mendapatkan pemberitahuan dengan snapshot data setiap kali data yang dideteksi oleh aplikasi klien Anda berubah. Dengan begitu, hanya perubahan baru yang akan diambil.

Lindungi akses data Anda di Cloud Firestore dengan Firebase Authentication dan Aturan Keamanan Cloud Firestore untuk Android, iOS, dan JavaScript, atau Pengelolaan Akses dan Identitas (IAM) untuk bahasa sisi server.

**Lokasi implementasi**

* Mengintegrasikan Cloud Firestore SDK

Sertakan klien dengan cepat melalui Gradle, CocoaPods, atau script include.

* Melindungi data

Gunakan Aturan Keamanan Cloud Firestore atau Pengelolaan Akses dan Identitas (IAM) guna mengamankan data Anda untuk pengembangan seluler/web dan server.

* Menambahkan Data

Buat dokumen dan koleksi di database Anda.

* Mendapatkan Data

Buat kueri atau gunakan listener realtime untuk mengambil data dari database.

Untuk biaya Cloud Firestore, Google menggunakan beberapa aspek penghitungan seperti jumlah pemrosesan data, kapasitas data yang digunakan, dan network bandiwth yang digunakan.

# Memilih database: Cloud Firestore atau Realtime Database

Firebase menawarkan dua solusi database berbasis cloud yang dapat diakses klien, yang mendukung sinkronisasi data secara real-time:

* **Realtime Database** adalah database asli Firebase. Solusi ini berlatensi rendah dan efisien untuk aplikasi seluler yang membutuhkan status sinkronisasi di seluruh klien secara real-time.
* **Cloud Firestore** adalah database unggulan baru dari Firebase untuk pengembangan aplikasi seluler. Cloud Firestore melanjutkan keberhasilan Realtime Database dengan model data baru yang lebih intuitif. Cloud Firestore juga memiliki fitur kueri yang lebih kaya dan lebih cepat, dan skala yang lebih baik daripada Realtime Database.

## Recomendasi

Kami merekomendasikan Cloud Firestore untuk sebagian besar developer yang memulai project baru. Cloud Firestore menawarkan fungsi, performa, dan skalabilitas tambahan pada infrastruktur yang dirancang untuk mendukung fitur yang lebih canggih dalam rilis mendatang. Anda akan melihat jenis kueri baru, aturan keamanan yang lebih kuat, dan peningkatan performa, selain fitur-fitur lanjutan lainnya yang direncanakan untuk Cloud Firestore.

Saat akan memilih solusi database, pertimbangkan perbedaan antara Cloud Firestore dan Realtime Database berikut ini.

### Model data

Baik Realtime Database dan Cloud Firestore adalah Database NoSQL.

|  |  |
| --- | --- |
| **Realtime Database** | **Cloud Firestore** |
| **Menyimpan data sebagai satu hierarki JSON yang besar.**   * Data sederhana sangat mudah disimpan. * Data kompleks dan hierarkis lebih sulit disusun dalam skala besar.. | **Menyimpan data dalam dokumen yang tersusun dalam koleksi.**   * Data sederhana mudah disimpan dalam dokumen, yang sangat mirip dengan JSON. * Data kompleks dan hierarkis lebih mudah disusun dalam skala besar, dengan menggunakan subkoleksi dalam dokumen. * Membutuhkan sedikit denormalisasi dan perataan data. |

### Dukungan Real-Time dan Offline

Keduanya memiliki SDK real-time dan memprioritaskan aplikasi seluler, dan keduanya mendukung penyimpanan data lokal untuk aplikasi siap offline.

|  |  |
| --- | --- |
| **Realtime Database** | **Cloud Firestore** |
| **Dukungan offline untuk klien seluler hanya di iOS dan Android.** | **Dukungan offline untuk klien iOS, Android, dan web.** |

### Membuat Kueri

Mengambil, mengurutkan, dan memfilter data dari salah satu database melalui kueri.

|  |  |
| --- | --- |
| **Realtime Database** | **Cloud Firestore** |
| **Kueri mendalam dengan**[fungsi pengurutan dan pemfilteran](https://firebase.google.com/docs/database/web/lists-of-data?hl=id#sorting_and_filtering_data) **terbatas.**   * Anda hanya dapat mengurutkan atau memfilter properti, bukan mengurutkan dan memfilter properti, dalam satu kueri. * Kueri bersifat mendalam secara default. Kueri selalu menghasilkan keseluruhan subhierarki. | **Kueri terindeks dengan**[pengurutan dan pemfilteran](https://firebase.google.com/docs/firestore/client/query-data?hl=id) **gabungan.**   * Anda dapat merangkai filter dan menggabungkan pemfilteran serta pengurutan pada properti dalam satu kueri. * Menuliskan kueri dangkal untuk subkoleksi: Anda dapat membuat kueri subkoleksi dalam dokumen, dan bukan keseluruhan koleksi, atau bahkan keseluruhan dokumen. * Kueri diindeks secara default: performa kueri sebanding dengan ukuran kumpulan hasil, bukan kumpulan data. |

### Operasi tulis dan transaksi

|  |  |
| --- | --- |
| **Realtime Database** | **Cloud Firestore** |
| **Operasi tulis dan transaksi dasar.**   * [Penulisan data](https://firebase.google.com/docs/database/android/read-and-write?hl=id#basic_write) sebagai operasi individual. * [Transaksi](https://firebase.google.com/docs/database/ios/read-and-write?hl=id#save_data_as_transactions) di SDK asli memerlukan callback penyelesaian. | **Operasi tulis dan transaksi menyeluruh.**   * [Operasi batch](https://firebase.google.com/docs/firestore/manage-data/update-data?hl=id#batch_multiple_write_operations) dan penyelesaian operasi secara menyeluruh. * [Transaksi](https://firebase.google.com/docs/firestore/manage-data/update-data?hl=id#update_data_with_transactions) mengulang secara otomatis hingga operasi selesai. |

### Keandalan dan performa

|  |  |
| --- | --- |
| **Realtime Database** | **Cloud Firestore** |
| **Realtime Database adalah solusi regional.**   * Latensi sangat rendah, sehingga ini adalah pilihan tepat untuk sinkronisasi status yang sering dilakukan. * Database terbatas sesuai ketersediaan menurut zona di satu region. | **Cloud Firestore adalah solusi multi-region yang mendukung automatic scaling.**   * Menyimpan data Anda di beberapa pusat data di region yang berbeda, sehingga memastikan skalabilitas global dan keandalan yang kuat. * Menawarkan stabilitas dan keandalan Google Cloud Platform. |

### Skalabilitas

|  |  |
| --- | --- |
| **Realtime Database** | **Cloud Firestore** |
| **Penskalaan membutuhkan sharding.**  Skala ke sekitar 100.000 koneksi serentak dan 1.000 tulis/detik dalam database tunggal. Penskalaan yang lebih besar daripada itu memerlukan sharding data di seluruh database. | **Penskalaan berjalan otomatis.**  Penskalaan sepenuhnya otomatis. Saat ini, batas penskalaan adalah sekitar 1 juta koneksi serentak dan 10.000 tulis/detik. Kami berencana meningkatkan batas ini di masa mendatang. |

### Keamanan

|  |  |
| --- | --- |
| **Realtime Database** | **Cloud Firestore** |
| **Aturan menurun yang memerlukan validasi terpisah.**   * [Aturan Realtime Database](https://firebase.google.com/docs/database/security/?hl=id) adalah satu-satunya opsi keamanan. * Aturan baca dan tulis bersifat menurun * Anda harus [memvalidasi data](https://firebase.google.com/docs/database/security/securing-data?hl=id#validating_data) secara terpisah menggunakan aturan validate. | **Keamanan yang lebih sederhana dan lebih kuat untuk SDK seluler, web, dan server.**   * SDK seluler dan web menggunakan [Aturan Keamanan Cloud Firestore](https://firebase.google.com/docs/firestore/security/get-started?hl=id). SDK Server menggunakan [Pengelolaan Akses dan Identitas (IAM)](https://cloud.google.com/firestore/docs/security/iam?hl=id). * Aturan tidak menurun, kecuali Anda menggunakan karakter pengganti. * Validasi data terjadi secara otomatis. * Aturan dapat membatasi kueri: Jika hasil kueri mungkin berisi data yang tidak dapat diakses oleh pengguna, keseluruhan kueri akan gagal. |

### Harga

Harga kedua solusi tersedia di bagian [paket harga Spark, Flame, dan Blaze](https://firebase.google.com/pricing?hl=id).

|  |  |
| --- | --- |
| **Realtime Database** | **Cloud Firestore** |
| Membebankan biaya hanya untuk **bandwidth dan penyimpanan**, tetapi pada tarif lebih tinggi. | Membebankan biaya terutama pada **operasi yang dilakukan di database Anda (baca, tulis, hapus)**, dan pada tarif, bandwidth serta penyimpanan yang lebih rendah.  Cloud Firestore mendukung [batas pengeluaran](https://cloud.google.com/appengine/docs/standard/python/console/?hl=id#setting_a_spending_limit) harian untuk project Google App Engine, sehingga biayanya dipastikan tidak akan melampaui anggaran Anda. |

## Menggunakan Cloud Firestore dan Realtime Database

Anda dapat menggunakan kedua database ini dalam aplikasi atau project Firebase yang sama. Kedua database NoSQL ini dapat menyimpan jenis data yang sama dan library klien bekerja dengan cara yang sama. Ingatlah perbedaan yang diuraikan di atas jika Anda memutuskan untuk [menggunakan kedua database ini dalam aplikasi Anda](https://firebase.google.com/docs/firestore/firestore-for-rtdb?hl=id).

**Harga Cloud Firestore**

Dokumen ini menjelaskan tentang detail harga Cloud Firestore. Selain itu, dokumen ini juga memberi tahu cara membatasi dan memantau pengeluaran Anda di Cloud Firestore.

Jika Anda melakukan pembayaran dengan mata uang selain USD, yang berlaku adalah harga yang tercantum dalam mata uang Anda pada [SKU Cloud Platform](https://cloud.google.com/skus?hl=id).

**Ringkasan harga**

Saat menggunakan Cloud Firestore, Anda dikenai biaya untuk hal-hal berikut:

* **Jumlah pembacaan, penulisan, dan penghapusan yang dilakukan**.
* **Jumlah penyimpanan yang digunakan database Anda**, termasuk overhead untuk metadata dan indeks.
* **Jumlah bandwidth jaringan yang digunakan**.

Penggunaan bandwidth dan penyimpanan dihitung dalam gigabyte (GiB), dengan 1 GiB = 230 byte. Semua biaya akan bertambah setiap harinya.

Bagian berikut ini menjelaskan secara detail tentang biaya yang ditagihkan kepada Anda atas penggunaan Cloud Firestore.

**Pembacaan, penulisan, dan penghapusan**

Anda dikenai biaya untuk setiap pembacaan, penulisan, dan penghapusan dokumen yang dilakukan dengan Cloud Firestore:

| **Multi-region** | **Harga di luar kuota gratis** |
| --- | --- |
| Pembacaan dokumen | $0,06 per 100.000 dokumen |
| Penulisan dokumen | $0,18 per 100.000 dokumen |
| Penghapusan dokumen | $0,02 per 100.000 dokumen |

Jika Anda melakukan pembayaran dengan mata uang selain USD, yang berlaku adalah harga yang tercantum dalam mata uang Anda pada [SKU Cloud Platform](https://cloud.google.com/skus?hl=id).

Pengenaan biaya untuk penulisan dan penghapusan dihitung dengan mudah. Namun, harap diperhatikan bahwa pengenaan biaya untuk pembacaan agak sedikit berbeda. Bagian berikut ini akan menjelaskan perbedaan tersebut secara detail.

**Mendeteksi hasil kueri**

Dengan Cloud Firestore, Anda bisa [mendeteksi hasil kueri](https://firebase.google.com/docs/firestore/query-data/listen?hl=id) dan mendapatkan update realtime saat hasil kueri berubah.

Saat mendeteksi hasil kueri, Anda dikenai biaya pembacaan setiap kali terdapat penambahan atau update dokumen di kumpulan hasil. Anda juga dikenai biaya pembacaan ketika dokumen dihapus dari kumpulan hasil karena dokumen telah berubah. (Sebaliknya, saat dokumen dihapus, Anda tidak dikenai biaya pembacaan.)

Selain itu, jika listener tidak terhubung selama lebih dari 30 menit (misalnya, jika pengguna offline), Anda akan dikenai biaya pembacaan karena dianggap telah mengeluarkan kueri baru.

**Mengelola kumpulan hasil yang besar**

Cloud Firestore memiliki beberapa fitur yang dapat membantu Anda mengelola kueri yang menampilkan hasil dalam jumlah besar:

* **Cursor**, membuat Anda dapat melanjutkan kueri yang telah berjalan lama.
* **Token halaman**, membantu Anda memberi nomor pada hasil kueri.
* **Batas**, menentukan berapa banyak hasil yang akan diambil.
* **Offset**, membuat Anda dapat melewati sejumlah dokumen tertentu.

Penggunaan cursor, token halaman, dan batas tidak dikenai biaya tambahan. Tidak hanya itu, semua fitur tersebut dapat membantu menghemat uang hanya dengan membaca dokumen yang Anda butuhkan.

Namun, jika terdapat offset dalam kueri yang dikirim, Anda dikenai biaya pembacaan untuk setiap dokumen yang dilewati. Misalnya, jika kueri menggunakan offset 10, dan kueri tersebut menampilkan 1 dokumen, Anda akan dikenai biaya untuk 11 pembacaan. Karena biaya tambahan ini, Anda harus menggunakan cursor dan bukan offset, jika memungkinkan.

**Kueri selain pembacaan dokumen**

Untuk kueri selain pembacaan dokumen, seperti permintaan daftar ID koleksi, Anda dikenai biaya untuk pembacaan 1 dokumen. Jika mengambil kumpulan hasil yang lengkap memerlukan lebih dari 1 permintaan (misalnya, jika menggunakan penomoran halaman), Anda dikenai biaya sebanyak 1 kali per permintaan.

**Biaya minimum untuk kueri**

Terdapat biaya minimum pada 1 pembacaan dokumen untuk setiap kueri yang Anda lakukan, meskipun kueri tidak menampilkan hasil apa pun.

**Aturan Keamanan Cloud Firestore**

Untuk library klien seluler dan web, jika [Aturan Keamanan Cloud Firestore](https://firebase.google.com/docs/firestore/security/get-started?hl=id) Anda menggunakan exists(), get(), atau getAfter() untuk membaca 1 atau beberapa dokumen dari database, Anda akan dikenai biaya pembacaan tambahan sebagai berikut:

* Anda dikenai biaya pembacaan yang dibutuhkan untuk mengevaluasi Aturan Keamanan Cloud Firestore.

Misalnya, jika aturan Anda mengacu pada 3 dokumen, tetapi Cloud Firestore hanya perlu membaca 2 dokumen untuk mengevaluasi aturan, Anda akan dikenai biaya untuk 2 pembacaan tambahan terhadap dokumen tertanggung.

* Anda hanya dikenai biaya untuk 1 kali pembacaan per dokumen tertanggung, bahkan jika dokumen tersebut dirujuk lebih dari 1 kali oleh aturan Anda.
* Anda hanya dikenai biaya untuk evaluasi aturan sebanyak 1 kali per permintaan.

Oleh karena itu, biaya pembacaan banyak dokumen menjadi lebih murah daripada membaca dokumen 1 per 1, karena permintaan yang dibutuhkan lebih sedikit untuk membaca beberapa dokumen sekaligus.

* Saat mendeteksi hasil kueri, Anda dikenai biaya untuk evaluasi aturan dalam semua kasus berikut:
  + Saat Anda mengeluarkan kueri.
  + Setiap kali hasil kueri diupdate.
  + Kapan pun perangkat pengguna offline, lalu kembali online.
  + Kapan pun Anda mengupdate aturan.
  + Kapan pun Anda mengupdate dokumen tertanggung dalam aturan.

**Ukuran penyimpanan**

Anda dikenai biaya untuk jumlah data yang disimpan di Cloud Firestore, termasuk overhead penyimpanan. Jumlah overhead meliputi metadata, indeks otomatis, dan indeks gabungan. Setiap dokumen yang tersimpan di Cloud Firestore membutuhkan metadata berikut:

* ID dokumen, termasuk ID koleksi dan nama dokumen.
* Nama dan nilai setiap kolom. Karena Cloud Firestore tidak memiliki skema, nama setiap kolom dalam dokumen harus disimpan dengan nilai kolom.
* Indeks gabungan dan indeks 1 kolom yang mengacu pada dokumen. Setiap entri indeks akan berisi ID koleksi; sejumlah nilai kolom, bergantung pada definisi indeks; dan nama dokumen.

Pelajari cara menghitung penyimpanan Cloud Firestore di [Perhitungan Ukuran Penyimpanan](https://firebase.google.com/docs/firestore/storage-size?hl=id). Anda dapat melihat jumlah penyimpanan yang digunakan database dengan mengunjungi halaman [Detail Kuota App Engine](https://console.cloud.google.com/appengine/quotadetails?hl=id)di Konsol API.

| **Multi-region** | **Harga di luar kuota gratis** |
| --- | --- |
| Data tersimpan | $0,18/GiB/bulan |

Jika Anda melakukan pembayaran dengan mata uang selain USD, yang berlaku adalah harga yang tercantum dalam mata uang Anda pada [SKU Cloud Platform](https://cloud.google.com/skus?hl=id).

**Bandwidth jaringan**

Anda dikenai biaya untuk bandwidth jaringan yang digunakan oleh permintaan Cloud Firestore, seperti ditunjukkan pada bagian berikut. Untuk mengetahui berapa banyak bandwidth jaringan yang telah digunakan, Anda dapat menggunakan Konsol API untuk [mengekspor data penagihan ke file](https://support.google.com/cloud/answer/6293835?hl=id).

**Harga jaringan umum**

Untuk permintaan yang berasal dari Google Cloud Platform (misalnya, dari aplikasi yang berjalan di Google Compute Engine), Anda dikenai biaya sebagai berikut:

| **Jenis traffic** | **Harga** |
| --- | --- |
| Masuk | Gratis |
| Keluar dalam suatu region | Gratis |
| Keluar antar-region di multi-region yang sama | Gratis |
| Keluar antar-region di AS (per GiB) | $0,01 (gratis 10 GiB pertama per bulan) |
| Keluar antar-region, tidak termasuk traffic antar region AS | [Tarif jalur keluar internet untuk Google Cloud Platform](https://cloud.google.com/firestore/pricing?hl=id#internet-egress) |

Jika Anda melakukan pembayaran dengan mata uang selain USD, yang berlaku adalah harga yang tercantum dalam mata uang Anda pada [SKU Cloud Platform](https://cloud.google.com/skus?hl=id).

Cloud Firestore juga dikenai biaya jalur keluar internet tambahan untuk hal berikut:

* Permintaan Google Cloud Platform antar-region, tidak termasuk traffic antar-region AS
* Permintaan dari luar Google Cloud Platform (misalnya, dari perangkat seluler pengguna)

**Data tersimpan**

* Kuota gratis: 1 GiB
* Data tambahan: $0,18 \* 99 GiB = $17,82

**Mengelola pengeluaran**

Untuk membantu menghindari biaya tak terduga pada tagihan Anda, tetapkan [batasan pengeluaran harian](https://firebase.google.com/docs/firestore/usage?hl=id#limits)menggunakan Google App Engine atau [pemberitahuan dan anggaran bulanan](https://firebase.google.com/docs/firestore/usage?hl=id#budget) menggunakan konsol penagihan Google Cloud Platform.

**Contoh**

# Sampel web untuk Cloud Firestore

## Quickstart - Friendly Eats

Aplikasi sampel ini menunjukkan pembuatan layanan rekomendasi restoran sederhana menggunakan Cloud Firestore. Aplikasi ini memperlihatkan berbagai fitur, seperti kueri gabungan, transaksi sisi klien, subkoleksi, dan persistensi offline.

FriendlyEats adalah aplikasi rekomendasi restoran yang dibangun di Cloud Firestore. Untuk informasi lebih lanjut tentang Firestore, kunjungi dokumen.

Proyek ini adalah titik awal untuk Cloud Firestore Web Codelab, yang akan menunjukkan kepada Anda bagaimana membangun aplikasi langkah demi langkah. Jika Anda ingin menjalankan hasil yang sudah selesai, lihat aplikasi mulai cepat.

