ANALISIS PERBANDINGAN PERFORMA API METODE REST DAN GRAPHQL DENGAN PHP DAN GO

I Gede Adrian Edy Pratama[1)] **I Putu Satwika**[2)] **I Nyoman Yudi Anggara Wijaya**[3)]
Program Studi Teknik Informatika[1) 2) 3)]
Sekolah Tinggi Manajemen Informatika dan Komputer (STMIK) Primakara, Denpasar, Bali[1) 2) 3)]
adrianedypratama@gmail.com[1)] satwika@primakara.ac.id[2)] inyomanyudi@primakara.ac.id[3)]

ABSTRACT

In the times where technology continues to innovate, many architectures have been created to communicate between technologies using an application programming interface or often called an API, among others are REST and GraphQL. This study will discuss the comparison of REST with GraphQL where GraphQL is a newer technology than REST. The comparison will be made by measuring the performance of the request response time. The comparison will be made with the PHP and using Go programming language as a comparison which is connected to the MongoDB database. The conclusions obtained from this research are expected to help developers who want to know the advantages and disadvantages of the REST and GraphQL architectures and their use cases.
Keywords*: API, REST, GraphQL, PHP, Go, MongoDB.*

ABSTRAK

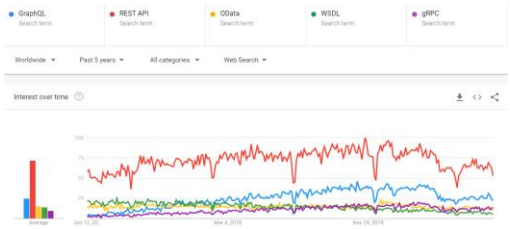
Dalam perkembangan zaman dimana teknologi terus berinovasi, telah banyak arsitektur-arsitektur yang diciptakan untuk melakukan komunikasi antara teknologi-teknologi menggunakan antarmuka pemrograman aplikasi atau sering disebut API diantaranya adalah REST dan GraphQL. Penelitian ini akan membahas perbandingan REST dengan GraphQL dimana GraphQL merupakan teknologi yang lebih baru daripada REST. Perbandingan yang akan dilakukan dengan mengukur performa dari waktu respon permintaan. Perbandingan akan dilakukan dengan bahasa pemrograman PHP dan menggunakan Go sebagai pembanding yang terkoneksi dengan database MongoDB. Kesimpulan yang didapatkan dari penelitian ini diharapkan dapat membantu para pengembang yang lebih ingin mengenal kelebihan dan kekurangan dari arsitektur REST maupun GraphQL dan kasus penggunaannya.
Kata Kunci : API, REST, GraphQL, PHP, Go, MongoDB.

PENDAHULUAN

Di era dimana sudah banyak teknologi yang digunakan masyarakat maka seperti manusia itu sendiri teknologi juga perlu untuk bisa berkomunikasi kepada manusia maupun teknologi lainnya. Maka dari itu Application Programming Interface atau disingkat API dibuat dan diterapkan untuk mempermudah teknologi-teknologi tersebut berkomunikasi antara satu dengan lainnya. [1]

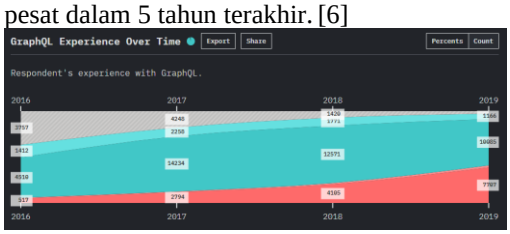
Dalam membangun aplikasi, API menyeder-hanakan pemrograman dengan mengabstraksi implementasi yang mendasarinya dan hanya mengekspos objek atau tindakan yang dibutuhkan pengembang. Sementara dalam segi antarmuka grafis untuk klien dalam penggunaan email adalah dengan memberi pengguna tombol yang melakukan semua langkah untuk mengambil dan menyorot email baru, API untuk *input/output file* akan memberi pengembang fungsi yang menyalin file dari satu lokasi ke lokasi lain tanpa mengharuskan pengembang memahami operasi sistem file yang terjadi di belakang layar. [2]

API dapat diterapkan dalam penggunaan *libraries* dan *framework,* sistem operasi, *remote APIs, Web APIs*. [3] Secara umum API digunakan untuk membuat web service dimana web service tersebut adalah perangkat elektronik yang menawarkan layanan kepada perangkat elektronik lainnya yang berkomunikasi menggunakan World Wide Web. [4] Dalam pengembangan API terdapat 2 teknologi terpopuler berdasarkan Google Trends selama 5 tahun terakhir yang diakses pada 10 Juni 2021 yang digunakan yang akan dibahas di penelitian ini, yaitu REST dan GraphQL.



Gambar 1. Grafik Popularitas Gaya Arsitektur API

Alasan penelitian ini hanya membandingkan REST dengan GraphQL selain dikarenakan merupakan 2 teknologi yang paling trending, juga dikarenakan 70% dari API umum menggunakan arsitektur REST [5] dan GraphQL merupakan teknologi yang mengalami perkembangan pengguna yang pesat dalam 5 tahun terakhir. [6]

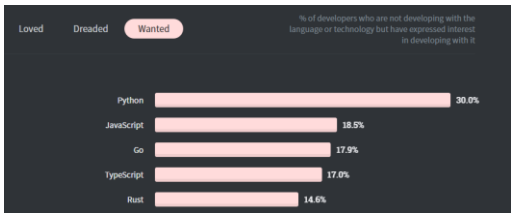


Gambar 2. Grafik Popularitas GraphQL

Dengan pesatnya perkembangan teknologi maka pertimbangan yang diperlukan untuk memilih teknologi semakin banyak dan kompleks, maka dari itu skripsi ini akan menjabarkan hasil penelitiannya berupa perbedaan dan perbandingan pada kedua teknologi tersebut dalam aspek performa dari waktu respon permintaan, menggunakan bahasa pemrograman PHP dan menggunakan Go sebagai pembanding yang terkoneksi dengan database MongoDB.

Digunakannya bahasa pemrograman PHP dikarenakan PHP adalah bahasa pemrograman yang paling banyak digunakan pada web yaitu 79.2% digunakan sebagai *server-side websites* yang diketahui pada 10 Juni 2021. [7]

Sedangkan bahasa pemrograman Go sebagai pembanding dikarenakan Go sendiri adalah salah 1 dari banyak bahasa pemrograman modern yang populer dan dicari menurut survei dari Stack Overflow pada tahun 2020. [8]



Gambar 3. Grafik Survei Bahasa Pemrograman Oleh Stack Overflow

Penggunaan database MongoDB sendiri digunakan dalam penelitian ini dikarenakan kemudahannya untuk dipelajari dan dimulai, selain itu juga sangat populer. [9]

Dengan bertambah banyaknya teknologi baru yang dapat digunakan untuk membuat API maka pilihan bagi pengembang pun bertambah banyak. Dalam pemilihan teknologi tersebut salah satu parameter yang digunakan untuk memilih adalah waktu respon. Waktu respon adalah waktu dari sesaat sebelum mengirim permintaan hingga tepat setelah respons terakhir diterima. [10]

Pada penelitian ini akan melakukan testing pada aplikasi sederhana yang dapat melakukan CRUD sederhana (*Create, Read, Update, Delete*). CRUD sendiri dalam pemrograman komputer adalah empat operasi dasar penyimpanan persisten. [11] CRUD juga terkadang digunakan untuk menggambarkan konvensi antarmuka pengguna yang memfasilitasi tampilan, pencarian, dan perubahan informasi menggunakan formulir dan laporan berbasis komputer. Selain itu testing juga akan dilakukan pada operasi *read* dengan menguji permasalahan *over-fetching* pada REST yang akan diselesaikan permasalahannya dengan *single query* dari GraphQL.

Tolak ukur kesuksesan penelitian ini ialah penelitian ini dapat memaparkan hasil peforma waktu respon pada teknologi yang diterapkan dan skema yang dilakukan dan memberikan kesimpulan atas hasil tersebut.

TINJAUAN PUSTAKA

API (*Application Programming Interface*)

Dalam komputasi, sebuah API merupakan antarmuka yang mendefinisikan interaksi diantara aplikasi perangkat lunak atau campuran perangkat keras dan lunak campuran.

API menetapkan jenis-jenis panggilan atau permintaan yang dapat dilakukan, bagaimana cara membuatnya, format data yang harus digunakan, konvensi yang harus dilakukan dan lain-lain. [12] Pada umumnya sekarang API direferensikan ke Web API [4] yang merupakan API yang akan dibahas pada penelitian ini, selain itu ada juga API untuk bahasa pemrograman, *library* untuk perangkat lunak, sistem operasi komputer dan perangkat keras komputer.

REST

Representational state transfer (REST) adalah ragam arsitektur perangkat lunak yang dibuat untuk memandu desain dan pengembangan arsitektur untuk *World Wide Web*. REST mendefinisikan satu set batasan tentang bagaimana arsitektur sistem hypermedia berskala internet terdistribusi, contohnya adalah bagaimana Web harus berperilaku. Gaya arsitektur REST menekankan skalabilitas interaksi antar komponen, antarmuka yang beraturan, penyebaran komponen secara independen, dan pembuatan arsitektur berlapis untuk memfasilitasi *caching* komponen untuk mengurangi latensi yang dirasakan pengguna, menegakkan keamanan, dan merangkum sistem warisan. [13] REST telah digunakan di seluruh industri perangkat lunak dan merupakan seperangkat pedoman yang diterima secara luas untuk membuat layanan web yang *stateless* dan dapat diandal.

Contoh penggunaan *endpoint* untuk mengambil data pengguna adalah sebagai berikut:

Endpoint:

GET /api/users/420

Response:

```
{
    "username": "Mr. T",
    "avatar":
"http://example.com/users/420/pic.
jpg",
    "catchphrase": "I pity the
fool",
    "favorite_dog": "beagle"
}
```

GraphQL

GraphQL adalah *open-source data query* dan bahasa manipulasi untuk API, dan waktu proses untuk memenuhi kueri dengan data yang ada. [14] GraphQL memberikan pendekatan untuk mengembangkan web API yang memungkinkan klien untuk mendefinisikan struktur data yang diperlukan, dan struktur data yang sama dikembalikan dari server, memungkinkan untuk mencegah pengembalian data dalam jumlah besar secara berlebihan. [15] Dengan menggunakan GraphQL data yang akan diterima akan lebih fleksibel hanya dengan menggunakan 1 *endpoint*, berikut contoh *query* GraphQL yang perlu dikirim untuk mendapatkan data *username*, *avatar*, dan *favorite_dog* dari data pengguna:

```
query {
    users(id: "420") {
        username,
        avatar,
        favorite_dog
    }
}
```

Bahasa Pemrograman

Bahasa pemrograman merupakan notasi untuk menulis program, digunakan untuk spesifikasi komputasi atau algoritma. [16] Pemrograman komputer itu sendiri penting saat ini karena begitu banyak dunia kita yang otomatis. Manusia harus mampu mengontrol interaksi antara manusia dan mesin. Karena komputer dan mesin dapat melakukan berbagai hal dengan sangat efisien dan akurat, kita dapat menggunakan pemrograman komputer untuk memanfaatkan kekuatan komputasi itu. [17] Dalam ilmu komputer, bahasa pemrograman dibagi menjadi 2 yaitu Bahasa pemrograman tingkat tinggi (*high-level*) dan tingkat rendah (*low level*). Bahasa pemrograman tingkat tinggi adalah bahasa pemrograman dengan abstraksi yang kuat dari detail komputer. Berbeda dengan bahasa pemrograman tingkat rendah, Bahasa pemrograman tingkat tinggi menggunakan elemen bahasa alami, lebih mudah digunakan, atau bahkan mengotomatisasi (atau bahkan menyembunyikan seluruhnya) area signifikan dari sistem komputasi (misalnya manajemen memori), membuat proses pengembangan program lebih sederhana dan lebih mudah dipahami daripada ketika menggunakan bahasa

tingkat rendah. Jumlah abstraksi yang diberikan menentukan seberapa "tingkat tinggi" bahasa pemrograman itu. [18] Dikarenakan kemudahan dan popularitasnya dua bahasa pemrograman tingkat tinggi akan digunakan sebagai variabel penelitian ini yaitu PHP dan Go.

PHP

PHP adalah bahasa *scripting* dengan tujuan umum yang secara khusus cocok untuk pengembangan web. [19] Ini awalnya dibuat oleh programmer Denmark-Kanada Rasmus Lerdorf pada tahun 1994. [20] Penggunaan acuan PHP sekarang dimiliki oleh The PHP Group. [21] PHP pada mulanya merupakan singkatan dari *Personal Home Page*, [20] tetapi sekarang merupakan singkatan dari inisialisasi rekursif *PHP: Hypertext Preprocessor*. [22] Kelebihan dari PHP itu sendiri antara lain kesederhanaan dan kefleksibelan sintaksnya sehingga mudah untuk dipelajari dan dukungan komunitas yang besar yang menyediakan solusi untuk masalah teknikal, *library* dan *framework*, sedangkan kekurangannya antara lain PHP tidak berbagi sumber daya antar proses, kualitas kode yang buruk dikarenakan terlalu fleksibel, terlalu bergantung pada *extention*, dan berjalan lebih lambat dari bahasa pemrograman lainnya. [23]

Go Programming Language

Go programming language atau biasa disebut go adalah bahasa pemrograman yang dikompilasi dan bertipe statis yang dirancang di Google [24] oleh Robert Griesemer, Rob Pike dan Ken Thompson. [25] Go secara sintaks mirip dengan C tetapi dengan keamanan memori, pengumpulan sampah, pengetikan struktural [25], dan konkurensi bergaya *communicating sequential processes* (CSP). [26] Go dirancang di Google pada tahun 2007 untuk meningkatkan produktivitas pemrograman di era *multi core*, mesin yang berjaringan dan basis kode yang besar. [27] Para desainer ingin mengatasi kritik terhadap bahasa lain yang digunakan di Google, tetapi tetap mempertahankan karakteristik mereka yang berguna [28] yaitu tipe yang statis dan waktu menjalankan yang efisien (seperti C), dapat dibaca dan mudah digunakan (seperti Python atau JavaScript), [25] berperforma tinggi pada jaringan dan *multiprocessing*. Kelebihan dari go itu sendiri terdapat pada kemudahannya untuk dipakai, standar *library* cerdas, keamanan kuat bawaan, dukungan dari Google, dan dokumentasi cerdas, sedangkan kekurangannya antara lain terlalu sederhana karena mengorbankan fungsi tingkat tinggi, bahasanya masih terlalu muda, kekurangan *virtual machine*, belum menemukan pasar spesialnya, dan belum memiliki *GUI library*. [29]

Basis Data

Database adalah koleksi terorganisir dari informasi, atau data yang terstruktur, umumnta disimpan secara elektronik dalam sistem komputer. Sebuah database umumnya dikendalikan oleh *database management system* (DBMS). Data dan DBMS, beserta dengan aplikasi yang berhubungan dengannya, dirujuk sebagai sistem basis data, kerap disingkat menjadi basis data saja.

Data dalam tipe database paling umum yang beroperasi saat ini secara khusus dimodelkan dengan baris dan kolom dalam jajaran tabel untuk melakukan pemrosesan dan kueri data yang efisien. Setelah itu data dapat dengan mudahnya diakses dan diatur. Umumnya database memakai *structured query language* (SQL) untuk menulis dan meminta data.

Selain SQL terdapat juga database NoSQL atau database non relasional, memungkinkan data tidak terstruktur dan semi-terstruktur untuk disimpan dan dimanipulasi (berbeda dengan database relasional, yang mendefinisikan bagaimana semua data yang dimasukkan ke dalam database harus disusun). Basis data NoSQL semakin disukai karena aplikasi web menjadi lebih ramai dan lebih kompleks. [30]

MongoDB

MongoDB adalah basis data berorientasi dokumen yang *source-available* dan *cross-platform*. Diklasifikasikan sebagai program basis data NoSQL, MongoDB menggunakan dokumen seperti JSON dengan skema yang opsional. MongoDB dikembang-kan oleh MongoDB Inc. Dan dilisensikan di bawah

Server Side Public License (SSPL). [31] Digunakannya MongoDB untuk penelitian ini dikarenakan laju pengembangan dengan database NoSQL bisa jauh lebih cepat daripada dengan database SQL. [32]

JSON

JSON (*JavaScript Object Notation*) adalah format file standar terbuka dan format pertukaran data yang memanfaatkan teks yang dapat dibaca manusia untuk menyimpan objek data yang tersusun dari pasangan *attribute-value* dan *array* . Ini adalah format data yang sangat umum, dengan beragam aplikasi, salah satu contohnya adalah aplikasi web yang berkomunikasi dengan server. [33]

Penelitian Sebelumnya

Penelitian mengenai perbandingan REST dan GraphQL sebelumnya sudah pernah diteliti oleh Mr.Sayan Guha and Mrs.Shreyasi Majumder (2020) dengan judul penelitian "A Comparative Study Between Graph-QL & Restful Services In Api Management Of Stateless Architectures" dimana penelitian tersebut berfokus pada potensi keuntungan GraphQL daripada REST dalam desain arsitektur *stateless* API. [34]

Dalam penelitian "GraphQL: The API Design Revolution" oleh Aleksi Ritsilä (2017) juga melakukan perbandingan antara REST dan GraphQL yang memaparkan hal-hal apa saja yang dapat lebih mudah dilakukan dengan GraphQL dibandingkan REST. [35]

Anugerah Christian Rompis (2018) juga melakukan penelitian dengan judul "Perbandingan Performa Kinerja Node.js, PHP, dan Python dalam Aplikasi REST" yang meneliti waktu respons dari setiap bahasa pemrograman yang dipaparkan melalui REST. [36]

Achmad Fauzi Harismawan, Agi Putra Kharisma dan Tri Afirianto (2018) dengan judul "Analisis Perbandingan Performa Web Service Menggunakan Bahasa Pemrograman Python, PHP, dan Perl pada klien Berbasis Android". Penelitian ini bertujuan untuk menghasilkan data perbandingan terhadap faktor yang digunakan yakni waktu tanggapan, penggunaan CPU dan RAM agar pembaca dapat meningkatkan efisiensi dalam hal penggunaan sumber daya, dan delay. [37]

M Gilvy Langgawan Putra dan M Ihsan Alfani Putera (2019) melakukan penelitian dengan judul "Analisis Perbandingan Metode SOAP dan REST yang Digunakan Pada Framework FLASK untuk Membangun Web Service" yang bertujuan untuk mengetahui kinerja kedua metode tersebut dari segi permintaan dan tanggapan API dari REST dan SOAP yang berjalan di framework Flask. [38]

Kontribusi baru penelitian ini adalah dimana membandingkan REST dan GraphQL dengan pemrograman PHP yang dibandingkan dengan bahasa pemrograman Go dan terkoneksi dengan database MongoDB untuk mengetahui apakah metode yang berbeda akan memiliki performa yang berbeda jika dijalan dengan bahasa pemrograman yang berbeda yang merupakan bahasa pemrograman yang popular saat penelitian ini dilaksanakan.

METODE PENELITIAN

Metode Penelitian

Aplikasi yang akan digunakan sebagai objek penelitian adalah sebuah aplikasi penyimpanan data sebuah film pada yang dibuat berdasarkan dataset sampel dari MongoDB sendiri yang bernama "*Sample Mflix Dataset*". Dipilihnya dataset ini dikarenakan dataset ini menyediakan koleksi *movies* dan *comments* dimana koleksi *movies* memiliki struktur yang cukup kompleks untuk menguji operasi *read* dalam kondisi *over-fecthing* oleh REST dan koleksi *comments* yang cukup sederhana untuk melakukan operasi *create*, *update* dan *delete*.

Aplikasi ini menyediakan layanan berdasarkan URL dengan data yang akan disajikan dalam bentuk *Javascript Object Notation* (JSON).

Penelitian untuk performa REST dan GraphQL ini akan menggunakan *cloud virtual machine* berbasis Linux dengan sistem operasi Ubuntu dari Digital Ocean dengan 1 vCPU Intel campuran prosesor Intel Xeon Scalable generasi pertama dan prosesor Intel Xeon lama dengan memory 1GB dan SSD 25GB. Menggunakan Nginx *server* untuk mengatur *server blocks*.

Pada pengembangan aplikasi GraphQL program yang dibangun akan dibantu dengan *package* yang direko-mendasikan oleh website

resmi dari GraphQL yaitu, *package* webonyx/graphql-php untuk aplikasi yang menggunakan PHP dan *package* graphql-go/graphql untuk aplikasi yang menggunakan Go. Untuk database MongoDB juga dibantu dengan *driver* yang direkomendasikan untuk masing-masing bahasa pemrograman.

Pada penelitian ini yang akan diuji adalah peforma respon dari API dengan permasalahan *over-fetching* pada REST yang akan diselesaikan permasalahannya dengan *single query* dari GraphQL, jumlah waktu respon yang didapatkan dari suatu permintaan data yang sama akan dibandingkan diantara REST dan GraphQL.

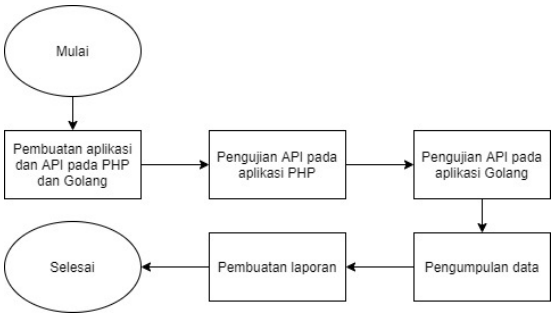
Alat yang akan digunakan untuk membantu pengujian ini adalah JMeter. JMeter merupakan perangkat lunak *open source* yang dibuat menggunakan aplikasi Java murni 100% yang didesain untuk memuat tindakan tes fungsional dan mengukur performa. Program ini mulanya didesain untuk menguji aplikasi web tetapi setelah itu diperluas ke fungsi pengetesan lainnya. [39] Digunakannya alat ini adalah untuk mempermudah pengiriman permintaan yang automatis dan mendapatkan pelaporan yang kuat dan efektif yang dapat divisualisasikan untuk diteliti lebih lanjut.

Untuk versi-versi perangkat lunak yang akan digunakan adalah versi stabil terbaru dari perangkat lunak tersebut saat penelitian dilaksanakan, yaitu sebagai berikut:

- PHP: 8.0.10
- Go: 1.17
- webonyx/graphql-php: 14.9.0
- graphql-go/graphql: 0.8.0
- MongoDB: 5.0.2
- MongoDB PHP Driver: ext 1.10 + lib 1.9
- MongoDB Go Driver: 1.7
- Ubuntu: 20.04.3 LTS
- Nginx: 1.18.0
- JMeter: 5.4.1

Alur Penelitian

Untuk penelitian mengenai performa metode API yang digunakan alurnya adalah sebagai berikut:



Gambar 4. Diagram Alur Penelitian

Untuk menjawab pertanyaan performa dari waktu respon permintaan dari kedua metode yang diuji, penelitian yang menunjukkan statistik dari waktu respon akan diberikan saat melakukan CRUD dari aplikasi yang digunakan untuk menguji.

Berikut adalah daftar fungsi yang dapat dijalankan:

- Membuat komentar
- Mengubah komentar
- Menghapus komentar
- Daftar film

Koleksi akan diperbaharui ke keadaan sebelum diubah setelah percobaan pada satu aplikasi agar jumlah data pada setiap percobaan diawali dengan keadaan yang sama. Id pada permintaan kasus komentar akan disesuaikan agar tidak mengubah/menghapus data yang sama. Pada fungsi daftar film akan dilakukan testing mengenai kasus *over-fetching* dimana klien mendapatkan data yang berlebihan dari suatu *endpoint* dari REST.

Berikut adalah kasus-kasus dimana *over-fetching* dapat terjadi yang akan penelitian ini uji:

- Kasus 1: Klien membutuhkan semua detail dari koleksi film.
- Kasus 2: Klien hanya membutuhkan data id, plot, genres, num_mflix_comments, poster, title, rated, lastupdated, year, dan type dari koleksi film.
- Kasus 3: Klien hanya membutuhkan id dan nama dari koleksi film.

Selain itu filter juga ditambahkan untuk menambah kompleksitas dari permintaan data terdapat 3 pengujian tambahan dengan filter dan batas maksimal data 1000, filter data yang digunakan adalah sebagi berikut:

- Filter 1: Data dengan countries yang mengandung “USA”.
- Filter 2: Data dengan countries yang mengandung “USA” dan rated = “PASSED”.
- Filter 3: Data dengan countries yang mengandung “USA”, rated = “PASSED” dan languages mengandung “English”.

Untuk setiap fungsi yang dijalankan akan dikirim sebanyak 10 kali, tiap permintaan akan dikirim setelah permintaan sebelumnya selesai atau setelah 100 detik permintaan sebelumnya terkirim tanpa di *cache*, dikirim pada rentang waktu 00:00 – 03:00 UTC+8 dan rata-rata waktu responnya akan dijadikan sebagai hasil penelitian, untuk fungsi daftar film akan ditambahkan variabel jumlah data yang diterima di setiap permintaan, banyak dari datanya adalah sebagai berikut:

Tahap	Banyak Data
1	10
2	100
3	1000
4	10000

Tabel 1 Tabel Banyaknya Data Tiap Permintaan

HASIL DAN PEMBAHASAN

Berikut adalah hasil rata-rata waktu respon dalam ms (*milisecond*) pada fungsi *create, update, delete* pada masing-masing aplikasi:

	PHP		Go	
	REST	GraphQL	REST	GraphQL
Create	42 ms	45 ms	42 ms	42 ms
Update	81 ms	81 ms	92 ms	79 ms
Delete	82 ms	79 ms	77 ms	81 ms

Tabel 2. Tabel Hasil Pengujian 1

Berikut adalah hasil waktu respon rata-rata (avg) dalam ms (*milisecond*) pada aplikasi PHP fungsi *read* pada metode REST dan GraphQL dengan GraphQL *query* yang menyesuaikan kasus yang diuji:

Data	REST	GraphQL 1	GraphQL 2	GraphQL 3
10	55 ms	53 ms	50 ms	45 ms
100	398 ms	156 ms	136 ms	52 ms
1000	1969 ms	831 ms	334 ms	135 ms
10000	8655 ms	9808 ms	3126 ms	892 ms

Tabel 3. Tabel Hasil Pengujian 2

Berikut adalah hasil waktu respon rata-rata (avg) dalam ms (*milisecond*) pada aplikasi Go fungsi *read* pada metode REST dan GraphQL dengan GraphQL *query* yang menyesuaikan kasus yang diuji :

Data	REST	GraphQL 1	GraphQL 2	GraphQL 3
10	55 ms	52 ms	56 ms	46 ms
100	273 ms	177 ms	194 ms	59 ms
1000	773 ms	1080 ms	556 ms	191 ms
10000	6412 ms	13905 ms	5021 ms	1015 ms

Tabel 4. Tabel Hasil Pengujian 3

Berikut adalah hasil rata-rata waktu respon dalam ms (*milisecond*) pada menggunakan kasus filter pada masing-masing aplikasi:

	PHP		Go	
	REST	GraphQL	REST	GraphQL
Filter 1	7512 ms	14375 ms	914 ms	2065 ms
Filter 2	509 ms	1466 ms	271 ms	1255 ms
Filter 3	499 ms	843 ms	268 ms	634 ms

Tabel 5. Tabel Hasil Pengujian 4

Dari hasil yang dilampirkan diatas dapat diambil kesimpulan bahwa penggunaan API REST dan GraphQL pada fungsi *create, update,* dan *delete* menggunakan data pengujian tidak memiliki perbedaan signifikan terhadap waktu responnya. Tetapi saat diuji pada fungsi *read* dimana GraphQL memberikan solusi untuk *over-fetching* pada REST maka perubahan waktu respon yang signifikan pada tiap kasus *query* yang dikirim melalui GraphQL dimana semakin sediti data yang diminta oleh *query* maka semakin cepat waktu respon yang didapatkan. Tetapi pada kasus pertama dari *query* GraphQL terdapat lonjakan waktu respon dibandingkan yang lain, ini dikarenakan data yang diambil dari basis data harus disesuaikan kepada tipe GraphQL terlebih dahulu agar pemanggilannya dapat melalui *query* pada *endpoint* GraphQL, lonjakan waktu respon tersebut sangat tinggi pada aplikasi Go, itu dikarenakan selain harus menyesuaikan tipe untuk GraphQL, Go yang bertipe statis mewajibkan data yang akan diterima *library* GraphQL memiliki tipe yang dibuat pada aplikasi Go terlebih dahulu sebelum diubah menjadi tipe GraphQL. Dari

kasus diatas maka pada aplikasi PHP terdapat kompleksitas O(N) pada kodenya dimana pengulangan terjadi sebanyak data/N yang dibutuhkan sedangkan pada aplikasi Go terdapat kompleksitas O(2N) pada kodenya dimana pengulangan terjadi sebanyak 2 kali dari data/N yang dibutuhkan.

SIMPULAN

Berdasarkan hasil dari penelitian mengenai perbandingan performa API metode REST dan GraphQL dengan PHP dan Go, dapat ditarik kesimpulan sebagai berikut:

1. Pada fungsi *create, update, delete* yang diteliti tidak terdapat perbedaan performa yang signifikan pada penggunaan metode API maupun bahasa pemrogram annya.
2. Perubahan yang signifikan pada metode REST dan GraphQL terjadi pada fungsi *read* dengan permintaan data sebanyak 100 dan diatasnya, itu dikarenakan *over-fetching* yang terjadi pada metode REST.
3. Pada fungsi *read*, metode GraphQL mengalami peningkatan peforma waktu respon pada setiap kasusnya.
4. Pada fungsi *read* metode REST, bahasa pemrograman Go lebih unggul daripada PHP.
5. Pada fungsi *read* metode GraphQL, bahasa pemrogra man PHP lebih unggul daripada Go.

Saat menggunakan filter aplikasi PHP maupun Go melakukan performa lebih baik melalui REST dibandingkan GraphQL.

Dari kesimpulan diatas, penulis menyarankan untuk menggunakan kedua metode tersebut untuk saling melengkapi kebutuhan pada suatu aplikasi. Untuk aplikasi yang sederhana, pakailah metode REST yang penggunaannya lebih universal, tetapi jika pada aplikasi menerima permintaan yang membutuhkan data yang besar dan dapat difleksibelkan, tambahkanlah GraphQL yang dapat menanganinya lebih baik, tetapi untuk GraphQL penurunan peforma dapat sangat terasa tergantung dari bahasa pemrograman yang digunakan dan cara data yang dibutuhkan diolah.

DAFTAR PUSTAKA

[1] "What is an API," HubSpire, [Online]. Available: https://www.hubspire.com/resources/general/application-programming-interface. [Diakses 13 September 2021].

[2] S. Clarke, "Measuring API Usability," Dr.Dobb's, 1 Mei 2004. [Online]. Available: https://www.drdobbs.com/windows/measuring-api-usability/184405654. [Diakses 13 September 2021].

[3] "Usage," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/API#Usage. [Diakses 13 September 2021].

[4] K. Lane, "Intro to APIs: History of APIs," Postman, 10 Oktober 2019. [Online]. Available: https://blog.postman.com/intro-to-apis-history-of-apis. [Diakses 13 September 2021].

[5] R. Pinkham, "[Infographic] REST 101: An Introduction to RESTful APIs," Smartbear, 8 Agustus 2016. [Online]. Available: https://smartbear.com/blog/an-introduction-to-restful-apis-infographic/. [Diakses 13 09 2021].

[6] "GraphQL," State of JS, 2019. [Online]. Available: https://2019.stateofjs.com/data-layer/graphql/. [Diakses 13 September 2021].

[7] "Usage statistics of PHP for websites," W3Techs, [Online]. Available: https://w3techs.com/technologies/details/pl-php. [Diakses 13 Agustus 2021].

[8] StackOverflow, "Most Loved, Dreaded, and Wanted Languages," StackOverflow, 2020. [Online]. Available:

https://insights.stackoverflow.co/survey/2020#technology-most-loved-dreaded-and-wanted-languages-wanted. [Diakses 13 September 2021].

[9] M. Asay, "Why MongoDB Is Popular," MongoDB, `1 Juli 2013. [Online]. Available: https://www.mongodb.com/blog/post/why-mongodb-popular. [Diakses 13 September 2021].

[10] Apache Software Foundation, "Glossary," Apache Software Foundation, [Online]. Available: https://jmeter.apache.org/usermanual/glossary.html. [Diakses 13 September 2021].

[11] J. Martin, "What are actions?," dalam *Managing the data-base environment*, Englewood Cliffs, Nj, Englewood Cliffs, N.J. : Prentice-Hall, 1983, p. 381.

[12] S. Fisher, dalam *OS/2 EE to Get 3270 Interface Early*, Google Books, 1989.

[13] R. Fielding dan J. Reschke, "HTTP/1.1 Semantics and Content," Internet Engineering Task Force (IETF), Juni 2014. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc7231#section-4. [Diakses 13 September 2021].

[14] "GraphQL: A query language for APIs," GraphQL, [Online]. Available: https://graphql.org/. [Diakses 13 September 2021].

[15] Phil, "GraphQL vs REST: Overview," Google, Inc, 24 Januari 2017. [Online]. Available: https://phil.tech/2017/graphql-vs-rest-overview. [Diakses 13 September 2021].

[16] A. Aaby, "Introduction to Programming Languages," 2004. [Online]. Available: https://web.archive.org/web/20121108043216/http://www.emu.edu.tr/aelci/Courses/D-318/D-318-Files/plbook/intro.htm. [Diakses 1 September 2021].

[17] Grand Canyon University, "Why Is Programming Important?," Grand Canyon University, 25 September 2020. [Online]. Available: https://www.gcu.edu/blog/engineering-technology/computer-programming-importance. [Diakses 13 September 2021].

[18] "RD Glosarry," [Online]. Available: https://web.archive.org/web/20070826224349/http://www.ittc.ku.edu/hybridthreads/glossary/index.php. [Diakses 13 September 2021].

[19] "PHP," The PHP Group, [Online]. Available: https://www.php.net/. [Diakses 13 September 2021].

[20] "History of PHP," The PHP Group, [Online]. Available: https://www.php.net/manual/en/history.php.php. [Diakses 13 September 2021].

[21] "History of PHP and Related Projects," The PHP Group, [Online]. Available: https://www.php.net/history. [Diakses 13 September 2021].

[22] "Preface," The PHP Group, [Online]. Available: https://www.php.net/manual/en/preface.php. [Diakses 13 September 2021].

[23] D. Pham, "PHP – Strengths and Weaknesses," Ryadel, 13 Juni 2020. [Online]. Available: https://www.ryadel.com/en/php-programming-language-strengths-weaknesses. [Diakses 13 September 2021].

[24] J. Kincaid, "Google's Go: A New Programming Language That's Python Meets C++," TechCrunch, 11 November 2009. [Online]. Available:

https://techcrunch.com/2009/11/10/google-go-language. [Diakses 13 September 2021].

[25] "Frequently Asked Questions (FAQ)," Google, Inc., [Online]. Available: https://golang.org/doc/faq#history. [Diakses 13 September 2021].

[26] C. Metz, "Google Go boldly goes where no code has gone before," The Register, 5 Mei 2011. [Online]. Available: https://www.theregister.com/2011/05/05/google_go. [Diakses 13 September 2021].

[27] R. Pike, "Go at Google: Language Design in the Service of Software Engineering," Google, Inc, 2012 Oktober 2012. [Online]. Available: https://talks.golang.org/2012/splash.article. [Diakses 13 September 2021].

[28] R. Pike, "Another Go at Language Design," Google, Inc, 28 April 2010. [Online]. Available: https://web.stanford.edu/class/ee380/Abstracts/100428.html. [Diakses 13 September 2021].

[29] I. Sidorenko, "Should I Go? The Pros and Cons of Using Go Programming Language," HackerNoon, 26 Februari 2019. [Online]. Available: https://hackernoon.com/should-i-go-the-pros-and-cons-of-using-go-programming-language-8c1daf711e46. [Diakses 13 September 2021].

[30] Oracle, "What Is a Database?," Oracle, [Online]. Available: https://www.oracle.com/database/what-is-database. [Diakses 13 September 2021].

[31] "MongoDB," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/MongoDB. [Diakses 13 September 2021].

[32] mongoDB, "When to Use a NoSQL Database," mongoDB, [Online]. Available: https://www.mongodb.com/nosql-explained/when-to-use-nosql. [Diakses 13 September 2021].

[33] ISO/IEC, "ISO/IEC 21778:2017 Information technology — The JSON data interchange syntax," 2017.

[34] S. Guha, "A Comparative Study Between Graph-QL & Restful Services in API Management of Stateless Architectures," *International Journal on Web Service Computing (IJWSC), Vol.11, No.2, June 2020,* 2020.

[35] A. Ritsilä, "GraphQL: The API Design Revolution," 2018.

[36] A. Rompis, "Perbandingan Performa Kinerja Node.js, PHP, dan Python dalam Aplikasi REST," *CogITo Smart Journal,* 2018.

[37] A. F. Harismawan, A. P. Kharisma dan T. Afirianto, "Analisis Perbandingan Performa Web Service Menggunakan Bahasa Pemrograman Python, PHP, dan Perl pada Client Berbasis Android," 2018.

[38] G. Langgawan dan M. I. A. Putera, "ANALISIS PERBANDINGAN METODE SOAP DAN REST YANG DIGUNAKAN PADA FRAMEWORK FLASK UNTUK MEMBANGUN WEB SERVICE," *SCAN - Jurnal Teknologi Informasi dan Komunikasi,* 2019.

[39] Apache Software Foundation, "Apache JMeter™," Apache Software Foundation, [Online]. Available: https://jmeter.apache.org. [Diakses 13 September 2021].