p-ISSN: 2620-3383
e-ISSN: 2528-6544

Penerapan Database Redis Sebagai Optimalisasi Pemrosesan Kueri Data Pengguna Aplikasi SIRESMA Berbasis Laravel

Ilham Nur Ramadhan[1]
Galuh Wilujeng Saraswati[2]

Universitas Dian Nuswantoro
Jl. Imam Bonjol No.207, Pendrikan Kidul, Kec. Semarang Tengah, Kota Semarang, Jawa Tengah 50131, Indonesia
E-mail: ilhamnurramadhan19@gmail.com [1]; galuhwilujengs@dsn.dinus.ac.id [2];

Notifikasi Penulis
30 September 2023
Akhir Revisi
12 Oktober 2023
Terbit
4 Februari 2024

Nur Ramadhan, I., & Saraswati, G. (2023). Implementation of the Redis Database as Optimization of User Data Query Processing for the Laravel-Based SIRESMA Application: Penerapan Database Redis Sebagai Optimalisasi Pemrosesan Kueri Data Pengguna Aplikasi SIRESMA Berbasis Laravel. *Technomedia Journal*, *8*(3 Februari), 394–407.

https://doi.org/10.33050/tmj.v8i3.2152

ABSTRACT

The development of the internet encourages people to access information using smartphones. SIRESMA is an application used by Sambiroto Village, Semarang City, to help manage waste. The application has the disadvantage that the application is insufficiently responsive. This is due to the use of MySQL relational databases without cache which results in heavy processing of queries. So a cache is a necessary tool to temporarily store data. The author implemented redis as a key-value database that functions as a cache. This research evaluates the use of the Redis database in the Laravel-based SIRESMA application to overcome the problem of data query processing delays. The test results show that the implementation of Redis has managed to significantly improve the access speed of most API endpoints, with a total response time improvement of around 6.74%. However, it should be noted that there are certain endpoints that experience performance degradation, which indicates that the use of Redis should be tailored to the characteristics of the data and application usage to get optimal results. Meanwhile, future research can consider more detailed query diagnosis tools and other alternatives to improve application performance.

Keywords: Cache, NoSQL, Redis, Laravel, Optimization

ABSTRAK

Perkembangan internet menunjang masyarakat untuk mengakses informasi menggunakan smartphone. SIRESMA adalah aplikasi yang digunakan oleh Kelurahan Sambiroto, Kota

p-ISSN: 2620-3383
e-ISSN: 2528-6544

Semarang, untuk membantu pengelolaan sampah. Aplikasi tersebut mempunyai kekurangan yaitu aplikasi yang kurang responsif. Hal ini disebabkan penggunaan databsase relasional MySQL tanpa cache yang mengakibatkan pemrosesan kueri yang cukup berat. Sehingga diperlukan cache yang berfungsi menyimpan data sementara. Penulis menerapkan redis sebagai key-value database yang berfungsi sebagai cache. Penelitian ini mengevaluasi penggunaan basis data Redis dalam aplikasi SIRESMA berbasis Laravel untuk mengatasi masalah keterlambatan pemrosesan kueri data. Hasil pengujian menunjukkan bahwa implementasi Redis telah berhasil meningkatkan signifikan kecepatan akses sebagian besar endpoint API, dengan total peningkatan waktu respons sekitar 6,74%. Namun, perlu diperhatikan bahwa ada beberapa endpoint tertentu yang mengalami penurunan performa, yang mengindikasikan bahwa penggunaan Redis harus disesuaikan dengan karakteristik data dan penggunaan aplikasi untuk mendapatkan hasil yang optimal. Sementara itu, penelitian mendatang dapat mempertimbangkan alat diagnosis kueri yang lebih rinci dan alternatif lain untuk meningkatkan performa aplikasi.

Kata kunci : Cache, NoSQL, Redis, Laravel, Optimalisasi.

PENDAHULUAN

Perkembangan Dunia teknologi dan informasi pada era ini berkembang sangat pesat [1]. contohnya yaitu semakin banyaknya pemanfaatan akses internet dalam kalangan masyarakat untuk mendapatkan berbagai informasi [2]. Salah satu inovasi dengan adanya teknologi tersebut adalah akses informasi yang dapat diperoleh di mana saja dan kapan saja dengan menggunakan *smartphone* dan komputer yamg terhubung dengan internet [3]. Salah satu aspek yang mencerminkan perkembangan pesat internet adalah meningkatnya penggunaan perangkat *mobile*, terutama perangkat berbasis *Android* [4]. Aplikasi *mobile* berbasis *Android* telah menjadi bagian *integral* dari kehidupan sehari-hari dengan jutaan aplikasi yang tersedia untuk memenuhi berbagai kebutuhan pengguna [5]. Aplikasi-aplikasi ini mencakup berbagai kategori, mulai dari media sosial, hiburan, hingga produktivitas dan pendidikan [6].

Salah satu aspek yang memanfaatkan kemajuan dari teknologi tersebut adalah pengelolaan sampah [7]. Salah satu kelurahan yang aktif dalam pengelolaan sampah, yaitu Kelurahan Sambiroto [8]. Kelurahan ini melakukan pendataan dalam pengelolaan sampah secara Digital [9]. Yang mana sistem ini sudah terintegrasi dengan bank sampah yang ada di Kelurahan tersebut sehingga data manajemen pengelolaan sampah dapat dikelola secara terpusat [10].

Aplikasi SIRESMA hadir untuk mengintegrasikan data pengelolaan sampah yang ada di Kelurahan Sambiroto [11]. SIRESMA adalah aplikasi Sistem Resik Mandiri yang dimiliki oleh TPS 3R di Kelurahan Sambiroto untuk menghubungkan serta mengelola data dari nasabah maupun bank sampah yang terpusat di aplikasi dan TPS 3R [12]. Di dalam aplikasi ini memiliki beberapa fitur utama [13]. Nasabah dapat melakukan permintaan setoran sampah berdasarkan jenis, melihat proses pengolahan sampah yang sudah disetor, hingga melakukan penarikan dana

p-ISSN: 2620-3383
e-ISSN: 2528-6544

yang didapatkan dari penyetoran sampah tiap bulan [14].

Sementara itu, pengelola dapat mengelola data sampah yang masuk pada bank sampah di daerah tempat tinggal pengelola, data yang dapat dikelola antara lain data profil para nasabah di dalam bank sampah tersebut, data setoran sampah dari masing-masing nasabah, hingga transaksi masuk dan keluar para nasabah [15]. Pengelola juga dapat mengunduh data pengelolaan sampah dalam kurun waktu yang telah ditentukan secara praktis dan mudah [16].

Aplikasi ini memiliki jumlah pengguna yang terus bertambah seiring berjalanya waktu [17]. Hal ini mengakibatkan arus data yang berlangsung dalam kurun waktu singkat semakin padat sehingga aplikasi SIRESMA cenderung kurang responsif dalam pengoperasiannya [18]. Masalah tersebut menjadi penghambat yang mengakibatkan aplikasi tersebut kurang efisien digunakan terhadap masyarakat luas [19]. Oleh karena itu, pembenahan dari sistem aplikasi SIRESMA harus dilakukan agar aplikasi tersebut dapat berjalan efisien [20].

PERMASALAHAN

Permasalahan utama yang timbul dalam penggunaan Aplikasi Sistem Resik Mandiri (SIRESMA) adalah proses di dalam aplikasi cenderung lambat. Aplikasi SIRESMA kerap kali digunakan para warga secara serentak pada saat agenda pilah sampah kelurahan [21]. Kali ini Penulis mengaitkan masalah tersebut dengan proses kueri yang ada di dalam sistem *API* SIRESMA [22].

Dari penelitian dan penggunaan database relasional seperti *MySQL* saja untuk aplikasi berskala besar kurang tepat [23]. Meskipun merupakan basis data yang sangat umum digunakan untuk pengelolaan data terstruktur dalam skala besar, metode tersebut kurang mampu memberikan kinerja yang optimal ketika digunakan dalam konteks dengan volume tinggi [24]. Contohnya seperti pengelolaan data otentikasi pengguna dan permintaan transaksi dalam SIRESMA [25].

Oleh karena itu, dalam usulan penelitian ini penulis mengusulkan penggunaan *database* Redis untuk penyimpanan data autentikasi sebagai solusi untuk mengatasi masalah atas tersebut [26]. *Redis* adalah basis data berkinerja tinggi yang secara khusus dirancang untuk menangani data yang memerlukan akses cepat dan dalam jumlah besar [27]. Selain itu, *Redis* memiliki beragam tipe struktur data sehingga kita dapat menggunakan *Redis* sesuai dengan kebutuhan kita [28]. Penggunaan *Redis* akan membantu meningkatkan responsivitas aplikasi dan mengoptimalkan pengelolaan transaksi, sejalan dengan rekomendasi yang terdapat dalam penelitian [29].

Penulis berharap dalam diterapkannya *database Redis* dapat menunjang pemrosesan kueri yang ada di dalam *server* SIRESMA [30]. Dalam penelitian, pengelolaan permintaan transaksi dalam sebuah aplikasi dapat dilakukan dengan responsivitas yang lebih baik sehingga memastikan pengalaman pengguna yang lebih lancar dan efisien [31].

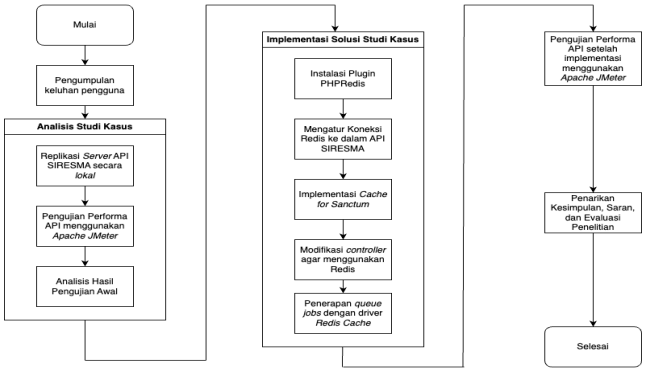
p-ISSN: 2620-3383
e-ISSN: 2528-6544

Berdasarkan Penelitian, *Redis* juga memiliki mekanisme *caching* yang kuat, yang memungkinkan penyimpanan data dalam *cache* untuk mengurangi beban *server* dan mempercepat akses data [32]. Hal ini sangat relevan dalam aplikasi *mobile*, di mana penggunaan *cache* dapat mengoptimalkan kinerja aplikasi dan mengurangi latensi. Dengan demikian, penggunaan *Redis* sebagai basis data penunjang pemrosesan kueri data dalam *server* SIRESMA diharapkan membantu meningkatkan efisiensi, responsivitas, dan pengalaman pengguna secara keseluruhan.

METODOLOGI PENELITIAN

Dalam penelitian ini. Penulis membagi penelitian ini menjadi beberapa tahap. Tahap pertama diawali pengumpulan data melalui survei dalam lapangan. Kumpulan data tersebut ditindaklanjuti di dalam uji coba dan analisis secara teknikal. Penulis kemudian melakukan perancangan pengembangan sistem yang berlandaskan penelitian sebelumnya. pengembangan sistem sesuai dengan penemuan kekurangan dari sistem yang sudah berjalan

Dalam penelitian ini, Penulis memilih metodologi pengembangan perangkat lunak *Extreme Programming.* Salah satu keuntungan utama menggunakan *XP* adalah fleksibilitasnya dalam menghadapi perubahan kebutuhan. Dalam pengembangan *API*, kebutuhan seringkali berubah seiring waktu. Ddengan *XP*, tim pengembang dapat secara cepat menyesuaikan diri dengan perubahan tersebut. Selain itu, metode pengujian dan refaktoring terus menerus dalam *XP* memungkinkan untuk mengidentifikasi dan memperbaiki masalah usabilitas API secara lebih dini dalam proses pengembangan. Dalam Kasus ini, Penulis merancang alur penelitian berbasis *Extreme Programming* Sebagai Berikut :



Gambar 1. Kerangka Penelitian

Dari Kerangka Kerja di atas, penulis kemudian melakukan penelitian dengan rincian proses kerja dalam penelitian kali ini seperti berikut:

A. Pengumpulan umpan balik pengguna aplikasi SIRESMA

Untuk mengidentifikasi masalah yang terjadi di dalam penggunaan aplikasi SIRESMA, penulis mengadakan survei dengan melakukan wawancara dengan beberapa sampel pengguna SIRESMA pada saat uji coba dalam kegiatan pilah sampah di RT 08, Kelurahan Sambiroto, Kota Semarang. Tujuan dalam penelitian survei adalah untuk mengumpulkan informasi dari suatu sampel dengan menggunakan angket atau wawancara untuk menggambarkan berbagai aspek dari populasi yang diteliti. Hasil temuan dari survei dengan wawancara tersebut adalah

p-ISSN: 2620-3383
e-ISSN: 2528-6544

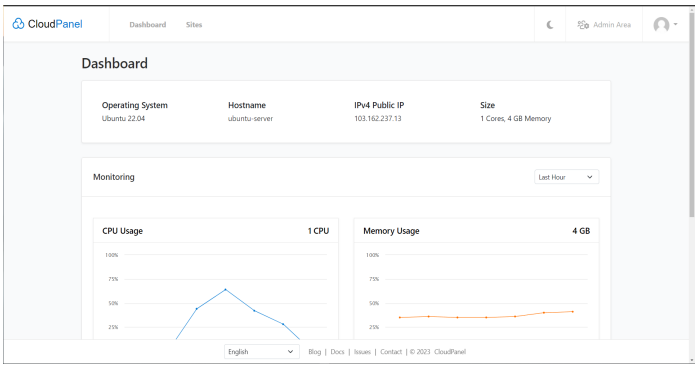
beberapa kelemahan aplikasi tersebut, antara lain:

- Akses Login di dalam aplikasi yang cukup lama saat proses pilah sampah masal
- Proses pindah laman di dalam aplikasi yang kurang responsif (terutama dalam proses memuat daftar transaksi dan setoran sampah)
- Proses penyambungan data setor sampah dan data setor sampah yang seringkali gagal
- Proses memuat data transaksi dalam pengguna yang lama saat data transaksi pengguna telah banyak

A. Analisis hasil umpan balik pengguna aplikasi SIRESMA dalam *server*

Pada proses analisis kali ini, penulis mencoba merepelikasi struktur sistem yang digunakan di dalam server *API* SIRESMA. *Server API* SIRESMA memiliki spesifikasi 1 Inti Prosesor serta 4 GB *RAM*. *Server* ini menggunakan Sistem Operasi *Linux* dengan distribusi *Ubuntu* versi 22.04 dan menggunakan perangkat lunak manajemen *server Cloudpanel*. *API SIRESMA* yang ada di dalam *server* ini menggunakan *framework Laravel* Versi 9 yang berjalan dengan versi *PHP* 8.1.

Setelah mengetahui struktur sistem yang digunakan dalam *server* ini. Penulis menggunakan *VMWare* untuk membuat *virtual machine* yang memiliki spesifikasi yang sama dengan *server API SIRESMA*. *VMWare* memungkinkan kita untuk membuat *Virtual Machine* dengan simulasi jaringan tanpa perangkat keras tambahan sehingga dapat menghemat biaya. Setelah *virtual machine* terbuat. Penulis melakukan instalasi *API SIRESMA* dengan mengkloning sumber kode aplikasi tersebut dari repositori pengembang, memasang *package* yang diperlukan, lalu memigrasi basis data yang telah diatur di dalam *database migration* yang ada di dalam sumber kode tersebut.

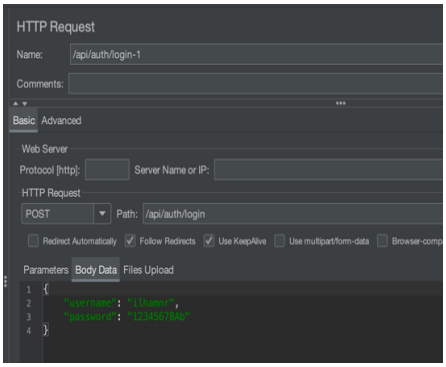


Gambar 2. Tangkapan Layar Server yang Telah direplikasi

Setelah berhasil mereplikasi *server API.* Penulis melakukan ujicoba terhadap Kinerja dari *API* tersebut dengan menggunakan *Apache JMeter.* Alasan penulis menggunakan aplikasi tersebut adalah aplikasi *JMeter* merupakan aplikasi *load testing* yang gratis serta memiliki banyak fitur. Salah satu fitur yang menarik dari aplikasi tersebut adalah dapat membuat laporan pengetesan baik berbentuk tabel maupun grafik.

Penulis membuat *HTTP Request* untuk mengakses *endpoint* dalam *API* tersebut dengan cara memanfaatkan *Recording Controller* yang dapat mendeteksi adanya *HTTP Request* dengan *server proxy*, yang dimana *server proxy* tersebut digunakan ke dalam Aplikasi *Postman* yang berisi tentang segala dokumentasi terhadap *endpoint API SIRESMA* dari pengembang.

p-ISSN: 2620-3383
e-ISSN: 2528-6544



Gambar 3. Tampilan *JMeter* setelah me-*record HTTP Request*

Setelah *endpoint* telah diatur seperti pada Gambar 2, penulis melakukan serangkaian pengujian kinerja dengan menggunakan *Apache JMeter* sebelum mengimplementasikan Redis dalam *server API* SIRESMA di dalam *server* yang sudah direplikasi. Pengujian ini melibatkan simulasi 100 pengguna yang mengakses *endpoint* API secara bersamaan untuk mengidentifikasi potensi masalah kinerja atau penundaan dalam situasi beban tinggi.

Setelah mengujicoba *API* yang ada di dalam aplikasi tersebut. Penulis mendapatkan hasil seperti berikut :

Tabel 1. Waktu Respons *API* SIRESMA awal

Request	Waktu Respons (milidetik)		
	Rata-rata	Minimum	Maksimum
/api/admin/nasabah	8601.31	1012	16269
/api/admin/nasabah/details	1001.49	955	1034
/api/admin/nasabah/details/transactions	997.08	956	1019
/api/admin/transactions/incoming	3376.95	1018	5687
/api/admin/transactions/outcoming	5638.45	5634	5684
/api/auth/login	2804.20	109	5509
/api/auth/logout	3258.73	951	5589
/api/bank-sampah/list	3192.52	988	5383
/api/home	1001.74	962	1025
/api/iot/connect	8096.38	2560	13658
/api/iot/store	1051.17	992	1075
/api/myprofile/details	1006.68	970	1039
/api/myprofile/update	1073.97	1021	1125
/api/myprofile/update/password	3799.29	1152	6245
/api/transaction/list	11347.17	6324	16089
/api/transaction/withdraw	16060.52	15857	16430
/api/trash/category	1760.18	992	2545
/api/trash/list	7512.08	1179	13829
/api/trash/store	1031.29	979	1072
Total	**4347.96**	**109**	**16430**

Hasil pengujian kinerja dengan menggunakan *Apache JMeter* pada berbagai *endpoint API* dalam aplikasi SIRESMA menunjukkan variasi dalam waktu respons. Rata-rata waktu respons

p-ISSN: 2620-3383
e-ISSN: 2528-6544

berkisar antara 1001 hingga 16.060 milidetik, dengan beberapa *endpoint* seperti *"/api/transaction/withdraw"* mencapai waktu respons tertinggi. Hal ini mengindikasikan bahwa beberapa bagian dari aplikasi mungkin mengalami penundaan dalam situasi beban tinggi dan perlu diperbaiki untuk meningkatkan responsivitasnya. Berdasarkan hasil pengujian diatas. Penulis kemudian menganalisis algoritma dan kueri yang berjalan di dalam *API* aplikasi tersebut. Penulis menggunakan *laravel telescope* sebagai alat pengecek kueri yang berjalan. *Laravel Telescope* berfungsi untuk merekam dan melihat segala proses yang berjalan di dalam aplikasi yang berbasis *laravel.* Berdasarkan analisis tersebut, penulis menyimpulkan beberapa poin *Requirement* pengguna untuk memperbaiki *API* SIRESMA, antara lain sebagai berikut:

Tabel 2. Analisis *Requirement* Pengguna SIRESMA

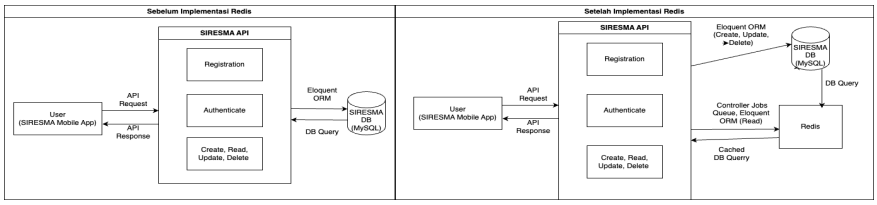
Identifikasi Masalah	Akibat	Penyebab	Alternatif Solusi
Akses *Login* di dalam aplikasi yang cukup lama saat proses pilah sampah masal	Pengguna seringkali mengira proses *login* gagal. sehingga selalu meminta *reset password*.	Penggunaan *Authentication Sanctum* yang memerlukan pembuatan & pembacaan *entity* baru di tabel 'personal_access_token' yang diakses masal secara simultan	Memanfaatkan *cache Redis* dengan *Cache for Sanctum*
Proses pindah laman di dalam aplikasi yang kurang responsif (terutama dalam proses memuat data)	Proses pemuatan data yang selalu membaca data yang ada di dalam aplikasi *MySQL*	Pembacaan *entity* pada tiap controller yang mengharuskan akses ke dalam *database MySQL* berulang-ulang	Menggunakan *cache redis* untuk menyimpan data yang tidak selalu berubah
Proses penyambungan data setor sampah dan data setor sampah yang seringkali gagal	Algoritma yang ada di dalam fungsi penyambungan sampah yang cukup kompleks dan selalu di akses secara simultan di waktu yang sama	*controller* untuk menyambung data timbangan dan setoran sampah memiliki algoritma yang cukup kompleks sehingga membutukan waktu proses yang tidak singkat	Penerapan *queue jobs* agar controller tersebut dapat berjalan di latar belakang
Proses memuat data transaksi dalam pengguna yang lama saat data transaksi pengguna telah banyak	Para pengelola tidak dapat mengelola data pengguna secara cepat dan efisien	Data yang di beri dari *controller* pemuat data transaksi memuat seluruh data dari pengguna tersebut	Penerapan *server-side model pagination*

B. Penerapan Redis ke dalam aplikasi berbasis *Laravel*

Setelah mengetahui bagaimana hasil dari *API* sebelum menerapkan Redis. Penulis mulai menerapkan redis ke dalam *API* SIRESMA yang menggunakan *framework Laravel.* Penulis melakukan tahapan mempersiapkan *redis* ke dalam *virtual machine* yang telah dibuat, menghubungkan redis ke dalam *API* SIRESMA, dan mengimplementasikan beberapa metode

p-ISSN: 2620-3383
e-ISSN: 2528-6544

redis ke dalam *API* SIRESMA. Perbandingan alur sistem di dalam *API* SIRESMA sebelum dan sesudah menerapkan *redis* adalah sebagai berikut:

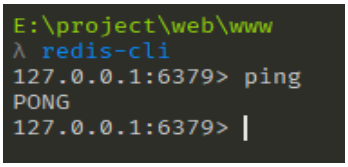


Gambar 4. Perbandingan Sistem *API* di dalam SIRESMA.

Penulis berencana merombak alur sistem yang ada di *API* SIRESMA sesuai pada alur di atas. Penulis akan mengimplementasikan *Redis* menjadi berbagai fungsi, yaitu sebagai penyimpan *session* otentikasi pengguna, penyimpan data pada kueri yang tidak berubah, hingga penyimpan *queue jobs* untuk beberapa *controller.* Penulis menjabar beberapa proses tersebut sebagai berikut:

a. Instalasi Redis & PHPRedis ke dalam Server Ubuntu

Tahap pertama adalah menginstal *Redis* dan *PHPRedis* di *server Ubuntu*. Instalasi *Redis* dapat dilakukan dengan cara meng-*import* kunci *GPG Redis* yang tersedia di dalam dokumentasi tersebut dan menjalankan perintah *"sudo apt-get install redis"* [23]. Lalu penulis menguji coba basis data *Redis* tersebut dengan menjalankan perintah *"redis-cli"* dan menjalankan perintah ping seperti ada di Gambar 4.



Gambar 4. Contoh basis data *Redis* yang sudah berjalan di dalam *server.*

Setelah basis data *Redis* telah bekerja secara baik, Langkah selanjutnya adalah memasang ekstensi *PHPRedis* ke dalam *PHP* yang terpasang di dalam *server* dengan perintah *"install php8.1-dev && pickle install redis"* ekstensi tersebut dipasang agar aplikasi berbasis dapat berinteraksi dengan basis data *Redis*.

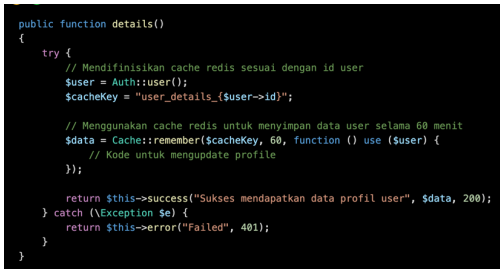
b. Mengatur agar autentikasi *sanctum* menggunakan *cache Redis*

API SIRESMA ini menggunakan autentikasi *Laravel Sanctum. Laravel Sanctum* bertanggung jawab dalam menyediakan data autentikasi pada pengguna yang telah melakukan proses *log in*. Setiap *endpoint* dari *API SIRESMA* memerlukan *token* autentikasi melalui *middleware Sanctum* yang tersimpan di dalam tabel "*personal_access_token*". Sehingga tabel tersebut selalu diakses setiap *endpoint* terpanggil. Oleh karena itu, kita dapat mengoptimalisasi proses tersebut dengan memanfaatkan *cache* di dalam basis data agar data *personal_access_token* dapat diakses secara cepat dan tidak memerlukan transaksi dari basis data *MySQL* secara terus-menerus dengan cara memasang "*package cache for Laravel sanctum"*. *Package* tersebut dapat dipasang dengan cara melakukan perintah *"composer require abe/cache-for-laravel-sanctum"* dan mengubah variabel di dalam *environment variables* menjadi *"SESSION_DRIVER=redis"* dan *"CACHE_DRIVER=redis"*.

c. Memodifikasi *controller* penampil data menggunakan *cache*

p-ISSN: 2620-3383
e-ISSN: 2528-6544

Pada tahap ini, penulis memodifikasi beberapa *controller,* yaitu *controller* data profil nasabah, lokasi bank sampah, serta daftar bank sampah. Penulis hanya mengubah tiga *controller* tersebut agar menggunakan *cache* dikarenakan data di dalam *controller* tersebut merupakan data yang jarang diperbaharui sehingga tidak memerlukan akses ke basis data utama secara langsung.



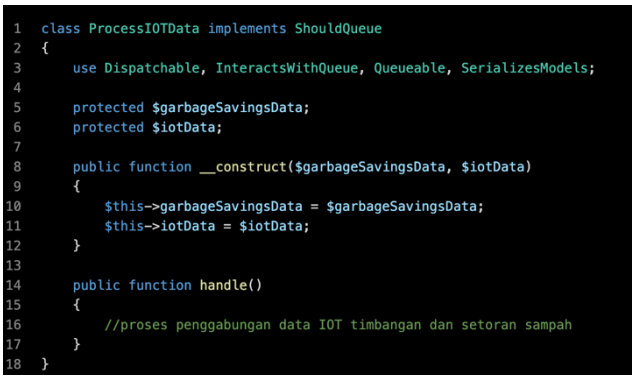
```
public function details()
{
    try {
        // Mendifinisikan cache redis sesuai dengan id user
        $user = Auth::user();
        $cacheKey = "user_details_{$user->id}";

        // Menggunakan cache redis untuk menyimpan data user selama 60 menit
        $data = Cache::remember($cacheKey, 60, function () use ($user) {
            // Kode untuk mengupdate profile
        });

        return $this->success("Sukses mendapatkan data profil user", $data, 200);
    } catch (\Exception $e) {
        return $this->error("Failed", 401);
    }
}
```

Gambar 5. Kode *Detail* Profil yang telah diberi model *Cache*

Setiap pengambilan data dilakukan dengan cara yang serupa seperti kode di atas. Pertama, suatu kunci cache didefinisikan berdasarkan kebutuhan (misal : *ID user*). Kemudian, metode "*Cache::remember()*" digunakan untuk mencoba mengambil data dari cache Redis. Jika data tidak ada dalam cache, maka data akan diambil dari database dan disimpan dalam cache untuk penggunaan berikutnya dengan waktu kedaluwarsa tertentu (dalam contoh di atas, 60 menit). Cara ini dapat membantu mengurangi beban *server* dikarenakan tidak terlalu sering membaca data di dalam basis data *MySQL*.

d. Penerapan *queue jobs* ke dalam *controller* penyambung data sampah dengan timbangan sampah.

Pada tahap ini, penulis memodifikasi *controller* yang selalu berubah datanya namun tidak memerlukan perubahan data secara *real-time* ke dalam *queue jobs. Queue Jobs* merupakan sebuah sistem antrean *controller* di dalam *framework Laravel.* Sistem antrean ini dapat diatur agar berjalan di dalam basis data *Redis* sebagai pencatat antrean *controller.* Isi dari kode *jobs* adalah seperti di bawah ini :



```
class ProcessIOTData implements ShouldQueue
{
    use Dispatchable, InteractsWithQueue, Queueable, SerializesModels;

    protected $garbageSavingsData;
    protected $iotData;

    public function __construct($garbageSavingsData, $iotData)
    {
        $this->garbageSavingsData = $garbageSavingsData;
        $this->iotData = $iotData;
    }

    public function handle()
    {
        //proses penggabungan data IOT timbangan dan setoran sampah
    }
}
```

Gambar 6. Contoh Kode *Jobs* untuk menghubungkan berat timbangan

Setelah membuat kode *jobs* diatas, kita harus memanggil kode *jobs* tersebut ke dalam *controller* utama dengan kode "*ProcessIOTData::dispatch($garbage_savings_data, $iot_data);*", dan kita perlu menambah perintah "*php artisan queue:work*" ke dalam *CRON Job* di *server* agar *worker* dari *jobs* tersebut selalu berjalan di *server.* dengan menggunakan sistem *queue*

p-ISSN: 2620-3383
e-ISSN: 2528-6544

jobs, controller dapat berjalan di dalam latar belakang sistem *API* tersebut.

C. **Uji Coba *API* yang telah terimplementasi Redis dengan menggunakan *JMeter***.

Setelah Penulis mengimplementasikan *Redis*. Penulis melakukan uji coba ulang *endpoint API* tersebut dengan menggunakan JMeter dengan konfigurasi yang digunakan (100 pengguna secara bersamaan). Ketika menguji coba *endpoint API* sebelum melakukan perubahan. Data yang diperoleh penulis adalah sebagai berikut.

Tabel 2. Waktu Respons *API* SIRESMA setelah implementasi Redis

Request	Waktu Respons (milidetik)		
	Rata-rata	Minimum	Maksimum
/api/admin/nasabah	8189.00	864	15467
/api/admin/nasabah/details	861.87	828	906
/api/admin/nasabah/details/transactions	866.78	830	906
/api/admin/transactions/incoming	3175.01	882	5408
/api/admin/transactions/outcoming	5368.25	5343	5408
/api/auth/login	2821.89	93	5521
/api/auth/logout	3054.24	794	5325
/api/bank-sampah/list	3214.67	1022	5415
/api/home	863.93	694	1019
/api/iot/connect	7129.83	1146	13167
/api/iot/store	963.17	913	1005
/api/myprofile/details	1819.03	1365	2059
/api/myprofile/update	1381.77	973	1864
/api/myprofile/update/password	3215.37	1027	5383
/api/transaction/list	9996.20	5480	14495
/api/transaction/withdraw	15010.80	14498	15611
/api/trash/category	1094.08	967	1353
/api/trash/list	7207.17	1082	13293
/api/trash/store	806.54	694	930
Total	**4054.72**	**93**	**15611**

HASIL DAN PEMBAHASAN

Dari Penelitian di atas, penulis mendapat data analisis Waktu Respons *API* SIRESMA sebelum dan sesudah implementasi *Redis* sebagai berikut:

Tabel 3. Perbandingan Waktu Respons *API* Sebelum dan Sesudah Implementasi Redis

Request	Perbandingan Rata-Rata		
	Sebelum (milidetik)	Setelah (milidetik)	Perbandingan (%)
/api/admin/nasabah	8601.31	8189.00	4.78
/api/admin/nasabah/details	1001.49	861.87	13.94
/api/admin/nasabah/details/transactions	997.08	866.78	13.08
/api/admin/transactions/incoming	3376.95	3175.01	5.97
/api/admin/transactions/outcoming	5638.45	5368.25	4.77
/api/auth/login	2804.20	2821.89	-0.63

p-ISSN: 2620-3383
e-ISSN: 2528-6544

/api/auth/logout	3258.73	3054.24	6.26
/api/bank-sampah/list	3192.52	3214.67	-0.69
/api/home	1001.74	863.93	13.75
/api/iot/connect	8096.38	7129.83	11.9
/api/iot/store	1051.17	963.17	8.37
/api/myprofile/details	1006.68	1819.03	-80.86
/api/myprofile/update	1073.97	1381.77	-28.58
/api/myprofile/update/password	3799.29	3215.37	15.34
/api/transaction/list	11347.17	9996.20	11.90
/api/transaction/withdraw	16060.52	15010.80	6.52
/api/trash/category	1760.18	1094.08	37.80
/api/trash/list	7512.08	7207.17	4.06
/api/trash/store	1031.29	806.54	21.76
Total	**4347.96**	**4054.72**	6.74

Hasil uji coba tersebut menunjukkan bahwa penerapan *Redis* dalam aplikasi SIRESMA menggunakan framework Laravel telah memberikan perbaikan yang signifikan terhadap kecepatan akses sebagian besar endpoint API, dengan total peningkatan waktu respons sekitar 6.74%. Terdapat penurunan yang mencolok pada beberapa *endpoint*, seperti "*/api/admin/nasabah/details*" dan "*/api/trash/category*" yang mengalami penurunan waktu respons yang signifikan sebesar 13.94% dan 37.80% secara berturut-turut. Namun, perlu diperhatikan bahwa pada *endpoin*t "*/api/myprofile/update*" dan "*/api/myprofile/details*" terjadi penurunan waktu respons yang cukup besar sebesar -28.58% dan bahkan -80.86. Ini mengindikasikan bahwa meskipun *Redis* berhasil meningkatkan kecepatan akses sebagian besar data, ada kekurangan pada endpoint tertentu seperti "*/api/myprofile/update*" yang dimana *endpoint* tersebut menyimpan data profil ke dalam *cache Redis*.

KESIMPULAN

Dalam analisis di atas, terlihat jelas bahwa pemanfaatan Redis telah memberikan dampak positif dalam meningkatkan efisiensi pemrosesan kueri data di berbagai endpoint aplikasi SIRESMA. Redis, sebagai sistem penyimpanan data berkinerja tinggi berbasis in-memory, membuktikan keunggulannya dengan mempercepat akses dan pengambilan data, yang secara langsung berkontribusi pada responsifitas aplikasi. Endpoint-endpoint kunci dalam aplikasi SIRESMA, seperti yang telah dibahas sebelumnya, menunjukkan peningkatan signifikan dalam waktu tanggapan dan performa keseluruhan. Dengan kata lain, Redis telah berhasil mengoptimalkan pengelolaan data dalam aplikasi, memberikan pengguna kemampuan untuk lebih efektif dan efisien dalam manajemen sampah. Keberhasilan implementasi Redis ini juga memberikan dampak positif terhadap pengalaman pengguna (user experience) dalam menggunakan aplikasi SIRESMA. Responsifitas yang ditingkatkan tidak hanya meningkatkan efisiensi pengelolaan sampah bagi masyarakat, tetapi juga memberikan pengalaman pengguna yang lebih baik dan memuaskan. Selain itu, implementasi Redis dalam server aplikasi SIRESMA memberikan fondasi yang kuat untuk skala aplikasi yang lebih besar di masa depan. Dengan manfaat pengolahan data yang lebih cepat dan efisien, aplikasi ini siap mengatasi peningkatan jumlah pengguna dan volume data tanpa mengorbankan kinerja. Pentingnya pengelolaan sampah dalam masyarakat membuat responsifitas aplikasi SIRESMA menjadi kritis. Redis, dengan keunggulan dalam pemrosesan data real-time, telah membuktikan dirinya

p-ISSN: 2620-3383
e-ISSN: 2528-6544

sebagai pilihan yang tepat untuk meningkatkan performa aplikasi ini. Dengan demikian, dapat disimpulkan bahwa penerapan Redis dalam SIRESMA bukan hanya sekadar pilihan teknologi yang efektif, tetapi juga langkah strategis untuk menciptakan solusi yang responsif dan efisien dalam mengelola sampah secara optimal.

SARAN

Saran penulis untuk penelitian selanjutnya adalah meningkatkan kecepatan kueri dalam aplikasi SIRESMA dengan lebih banyak eksplorasi metode optimalisasi Lainnya. Selain itu, penulis juga berharap dapat menguji hasil penelitian ini di masyarakat yang menggunakan aplikasi ini untuk mengukur dampak perbaikan kinerja setelah penelitian ini diterapkan di dalam *server* utama SIRESMA. Langkah tersebut ini akan mendukung perbaikan aplikasi SIRESMA maupun aplikasi sejenis sehingga dapat berkontribusi pada pelestarian lingkungan yang berkelanjutan.

DAFTAR PUSTAKA

[1] A. W. Ramadhan, A. Susanto, and G. W. Saraswati, “Implementasi Digital Payment Gateway Midtrans Pada Sistem Agribisnis Di Temanggung (SIADIT),” *J-SAKTI (Jurnal Sains Komputer dan Informatika)*, vol. 7, no. 1, pp. 95–107, 2023.

[2] H. Matallah, G. Belalem, and K. Bouamrane, “Comparative study between the MySQL relational database and the MongoDB NoSQL database,” *International Journal of Software Science and Computational Intelligence (IJSSCI)*, vol. 13, no. 3, pp. 38–63, 2021.

[3] D. S. S. Wuisan and T. Mariyanti, “Analisa Peran Triple Helik dalam Mengatasi Tantangan Pendidikan di Era Industri 4.0,” *Jurnal MENTARI: Manajemen, Pendidikan dan Teknologi Informasi*, vol. 1, no. 2, pp. 123–132, Jan. 2023, doi: 10.34306/mentari.v1i2.258.

[4] W. A. Nurasniar, “Employee Performance Improvement Through Competence and Organizational Culture with Work Motivation as A Mediation Variable,” *APTISI Transactions on Management (ATM)*, vol. 6, no. 2, pp. 121–131, Nov. 2021, doi: 10.33050/atm.v6i2.1743.

[5] Anggy Giri Prawiyogi and Aang Solahudin Anwar, “Perkembangan Internet of Things (IoT) pada Sektor Energi : Sistematik Literatur Review,” *Jurnal MENTARI: Manajemen, Pendidikan dan Teknologi Informasi*, vol. 1, no. 2, pp. 187–197, Jan. 2023, doi: 10.34306/mentari.v1i2.254.

[6] I. Hidayat and P. O. Sutria, “Influence of Determined Tax Load, Tax Planning, and Profitability in Profit Management in The Company Manufacturing The Mining Sector, The Coal Sub Sector Listed on The Indonesia Stock Exchange Year,” *APTISI Transactions on Management (ATM)*, vol. 7, no. 1, pp. 79–85, Feb. 2022, doi: 10.33050/atm.v7i1.1833.

[7] Zulham, Z. Lubis, M. Zarlis, and M. R. Aulia, “Performance Analysis of Oil Palm Companies Based on Barcode System through Fit Viability Approach: Long Work as A Moderator Variable,” *Aptisi Transactions on Technopreneurship (ATT)*, vol. 5, no. 1, pp. 40–52, Jan. 2023, doi: 10.34306/att.v5i1.288.

[8] N. Lutfiani, P. A. Sunarya, S. Millah, and S. Aulia Anjani, “Penerapan Gamifikasi

p-ISSN: 2620-3383
e-ISSN: 2528-6544

Blockchain dalam Pendidikan iLearning," *Technomedia Journal*, vol. 7, no. 3, pp. 399–407, Dec. 2022, doi: 10.33050/tmj.v7i3.1958.

[9] A. Ajredini, "Database Cashing in-memory with Redis NoSQL Databases".

[10] A. Maulana *et al.*, *PEMROGRAMAN WEB 101: MEMAHAMI DASAR-DASAR UNTUK MENGEMBANGKAN SITUS WEB*. Get Press Indonesia, 2023.

[11] Y. Sawitri, I. A. Yannaty, S. I. Widyastika, T. D. Harumsih, and H. F. Musyarofah, "Dampak penggunaan smartphone terhadap perkembangan anak usia dini," in *Prosiding Seminar Nasional LPPM UMP*, 2019, pp. 691–697.

[12] A. Fagerstrøm, N. Eriksson, and V. Sigurdsson, "Investigating the impact of Internet of Things services from a smartphone app on grocery shopping," *Journal of Retailing and Consumer Services*, vol. 52, p. 101927, 2020.

[13] A. R. Kurniawan, "Tantangan pengembangan pariwisata berbasis masyarakat pada era digital di Indonesia (Studi Kasus Pengembangan Pariwisata Berbasis Masyarakat di Pangalengan)," *Tornare: Journal of Sustainable and Research*, vol. 2, no. 2, p. 10, 2020.

[14] M. D. Permatasari *et al.*, "PENGEMBANGAN APLIKASI SISTEM INFORMASI RESIK BECIK (SIKECIK) BERBASIS WEB PADA RUMAH SAMPAH RESIK BECIK KELURAHAN KROBOKAN SEMARANG," *BUDIMAS: JURNAL PENGABDIAN MASYARAKAT*, vol. 4, no. 2, pp. 595–599, 2022.

[15] Q. Liu and H. Yuan, "A High Performance Memory Key-Value Database Based on Redis.," *J. Comput.*, vol. 14, no. 3, pp. 170–183, 2019.

[16] J. Jansson, A. Vukosavljevic, and I. Catovic, "Performance comparison between multi-model, key-value and documental NoSQL database management systems." 2021.

[17] Q. Meng, K. Zhang, H. Pan, M. Yuan, and B. Ma, "Design and Implementation of Key-Value Database for Ship Virtual Test Platform Based on Distributed System," in *International Conference of Pioneering Computer Scientists, Engineers and Educators*, Springer, 2023, pp. 109–123.

[18] A. I. Sanka, M. H. Chowdhury, and R. C. C. Cheung, "Efficient high-performance FPGA-Redis hybrid NoSQL caching system for blockchain scalability," *Comput Commun*, vol. 169, pp. 81–91, 2021.

[19] R. Ceresnak, M. Kvet, and K. Matiasko, "Improved method of selecting data in a nonrelational database," in *2021 International Conference on Information and Digital Technologies (IDT)*, IEEE, 2021, pp. 59–64.

[20] R. Patel, "Data+ Education. Redis Is a Cache or More?," *EasyChair Preprint*, vol. 88, 2021.

[21] H. M. Elmatsani, "Desain Metode PrefetchCache untuk Peningkatan Kinerja Aplikasi Web," *Techno. Com*, vol. 19, no. 2, pp. 147–156, 2020.

[22] W. M. Sari, A. Amran, and H. O. L. Wijaya, "Penerapan E-Commerce Menggunakan Metode Extreme Programming Pada Umkm Kabupaten Muratara," *Jusikom: Jurnal Sistem Komputer Musirawas*, vol. 5, no. 2, pp. 136–144, 2020.

[23] D. Satria, *PENGANTAR TEKNIK KOMPUTER: Konsep dan Prinsip Dasar*. PT. Sonpedia Publishing Indonesia, 2023.

[24] M. Maidiana, "Penelitian survey," *ALACRITY: Journal of Education*, pp. 20–29, 2021.

[25] K. McGlade, K. Hakkarainen, J. Jaaskelainen, and R. Roulston, "Implementing a digital authoring system in finnish medical and dental schools," in *EDULEARN21 Proceedings*,

p-ISSN: 2620-3383
e-ISSN: 2528-6544

IATED, 2021, pp. 7969–7974.

[26] R. Chopade and V. Pachghare, "A data recovery technique for Redis using internal dictionary structure," *Forensic Science International: Digital Investigation*, vol. 38, p. 301218, 2021.

[27] O. M. A. AL-atraqchi, "A Proposed Model for Build a Secure Restful API to Connect between Server Side and Mobile Application Using Laravel Framework with Flutter Toolkits," *Cihan University-Erbil Scientific Journal*, vol. 6, no. 2, pp. 28–35, 2022.

[28] H. Samosir, T. A. Prasetyo, S. Lumbantobing, D. O. Naibaho, and C. R. T. Situmorang, "Website Development with Laravel and Scrum Method: A Study case of Stasiun Mebel Jepara Store Case," in *2021 17th International Conference on Quality in Research (QIR): International Symposium on Electrical and Computer Engineering*, IEEE, 2021, pp. 60–65.

[29] F. P. PURWANTORO, "Pengembangan Sistem Presensi Untuk Work From Home (Wfh) Dan Work From Office (Wfo) Selama Pandemi Covid-19," 2021.

[30] V. Gopalakrishnan and C. Ramaswamy, "Patient opinion mining to analyze drugs satisfaction using supervised learning," *Journal of Applied Research and Technology*, vol. 15, no. 4, pp. 311–319, Aug. 2017, doi: 10.1016/j.jart.2017.02.005.

[31] V. Gopalakrishnan and C. Ramaswamy, "Patient opinion mining to analyze drugs satisfaction using supervised learning," *Journal of Applied Research and Technology*, vol. 15, no. 4, pp. 311–319, Aug. 2017, doi: 10.1016/j.jart.2017.02.005.

[32] U. Rahardja, M. A. Ngadi, R. Budiarto, Q. Aini, M. Hardini, and F. P. Oganda, "Education Exchange Storage Protocol: Transformation Into Decentralized Learning Platform," *Front Educ (Lausanne)*, vol. 6, Dec. 2021, doi: 10.3389/feduc.2021.782969.