ANALISIS PERBANDINGAN PERFORMA *WEB SERVICE* MENGGUNAKAN BAHASA PEMROGRAMAN PYTHON, PHP, DAN PERL PADA *CLIENT* BERBASIS ANDROID

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Achmad Fauzi Harismawan
NIM: 125150207111100

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2017

PENGESAHAN

ANALISIS PERBANDINGAN PERFORMA *WEB SERVICE* MENGGUNAKAN BAHASA PEMROGRAMAN PYTHON, PHP, DAN PERL PADA *CLIENT* BERBASIS ANDROID

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Achmad Fauzi Harismawan
NIM: 125150207111100

Skripsi ini telah diuji dan dinyatakan lulus pada
31 Juli 2017
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Agi Putra Kharisma, S.T, M.T	Tri Afirianto, S.T, M.T
NIK: 201304 860430 1 001	NIK: 201309 851213 1 001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata di dalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 31 Juli 2017

Achmad Fauzi Harismawan

NIM: 125150207111100

KATA PENGANTAR

Puji syukur kehadirat Tuhan YME yang telah melimpahkan rahmat-Nya sehingga laporan skripsi yang berjudul "Analisis Perbandingan Performa *Web Service* Menggunakan Bahasa Pemrograman Python, PHP, dan Perl Pada *Client* Berbasis Android" ini dapat terselesaikan.

Penulis menyadari bahwa skripsi ini tidak akan berhasil tanpa bantuan dari beberapa pihak. Oleh karena itu, penulis ingin menyampaikan rasa hormat dan terima kasih kepada:

1. Agi Putra Kharisma, S.T, M.T dan Tri Afirianto, S.T, M.T selaku dosen pembimbing skripsi yang telah dengan sabar membimbing dan mengarahkan penulis sehingga dapat menyelesaikan skripsi ini.
2. Tri Astoto Kurniawan, S.T, M.T, Ph.D selaku Ketua Jurusan Teknik Informatika.
3. Achmad Basuki, S.T, M.MG, Ph.D selaku dosen penasehat akademik yang selalu memberikan nasehat kepada penulis selama menempuh masa studi.
4. Keluarga S. Heriyanto, SH, serta seluruh keluarga besar atas segala nasehat, kasih sayang, perhatian dan kesabarannya di dalam membesarkan dan mendidik penulis, serta yang senantiasa tiada henti-hentinya memberikan doa dan semangat demi terselesaikannya skripsi ini.
5. Arnanda, Putra, Harisudin, Khabib, Irsyad, Sena, Maulana, Fahmi dan Reza, serta teman – teman Informatika Angkatan 2012 lainnya atas dukungan, masukan dan semangat yang diberikan kepada penulis sehingga terselesaikannya skripsi ini.
6. Seluruh civitas akademika Informatika Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penulis menempuh studi di Informatika Universitas Brawijaya dan selama penyelesaian skripsi ini.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih banyak kekurangan, sehingga saran dan kritik yang membangun sangat penulis harapkan. Akhir kata penulis berharap skripsi ini dapat membawa manfaat bagi semua pihak yang menggunakannya.

Malang, 31 Juli 2017

Penulis
achmadfauziharismawan@gmail.com

ABSTRAK

Dalam pengembangan aplikasi perangkat bergerak harus memperhatikan efisiensi dalam hal penggunaan sumber daya, selain itu aplikasi perangkat bergerak sangatlah sensitif terhadap *delay*. Salah satu faktor yang dapat menyebabkan *delay* adalah proses yang dilakukan pada *web service*. Kecepatan proses pada *web service* bergantung pada bahasa pemrograman yang digunakan, bahasa pemrograman juga dapat mempengaruhi penggunaan CPU (*Central Processing Unit*) dan *memory* pada *server*. Oleh karena itu, perlu dilakukan analisis performa *web service* untuk mengetahui perbedaan penggunaan CPU (*Central Processing Unit*), *memory*, dan kecepatan eksekusi bahasa pemrograman yang digunakan pada *web service*. Pada penelitian ini, bahasa pemrograman yang digunakan adalah Python, PHP, dan Perl dengan menggunakan CGI (*Common Gateway Interface*) pada Apache *web server* diakses melalui perangkat bergerak dengan *platform* Android. Setelah dilakukan penelitian didapatkan hasil bahwa bahasa pemrograman Perl memiliki kecepatan eksekusi paling cepat sedangkan bahasa pemrograman Python memiliki penggunaan CPU (*Central Processing Unit*) dan *memory* paling sedikit.

Kata kunci: *Web Service*, Android, Python, PHP, Perl

ABSTRACT

In the application development of mobile devices must pay attention to the efficiency in terms of resource usage, in addition mobile device applications are very sensitive to delay. One of the factors that can cause delay is the process done on the web service. The speed of the process in the web service depends on the programming language used, the programming language can also affect the use of CPU (Central Processing Unit) and memory on the server. Therefore, it is necessary to analyze the performance of web service to know the difference of CPU usage (Central Processing Unit), memory, and speed of execution of programming language used in web service. In this research, programming languages used are Python, PHP and Perl using CGI (Common Gateway Interface) on Apache web server accessed via mobile device with Android platform. After the experiment, it was found that Perl programming language has the fastest execution speed while the Python programming language has CPU usage (Central Processing Unit) and the least memory.

Keywords: *Web Service,* Android, Python, PHP, Perl

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Dalam pengembangan perangkat lunak harus memperhatikan efisiensi dalam hal penggunaan sumber daya sistem tersebut, terutama pada pengembangan aplikasi perangkat bergerak. Dikarenakan keterbatasan sumberdaya yang ada pada perangkat bergerak, *programmer* harus handal dalam mengelola sumberdaya yang ada.

Selain itu aplikasi perangkat bergerak juga tidak bisa lepas dari koneksi data melalui jaringan *internet*. Koneksi data dibutuhkan pada saat aplikasi perangkat bergerak akan berkomunikasi dengan *web service* pada saat melakukan sebuah operasi. Pada tahap ini sangat rawan terjadi *delay*. Sedangkan aplikasi perangkat bergerak itu sendiri sangat sensitif terhadap *delay*. *Delay* pada sebuah operasi akan mengakibatkan gangguan pada *user-experience* seorang *user* yang menggunakan aplikasi perangkat bergerak tersebut.

Terdapat beberapa parameter *web response time* yang dapat menyebabkan sebuah operasi mengalami *delay*, salah satunya adalah *server processing time* yaitu waktu yang dibutuhkan oleh *server* untuk memproses *request* yang dikirimkan oleh *client*. Hal ini dapat bervariasi secara drastis berdasarkan proses yang dilakukan oleh *server* (Savoia, 2001). Proses yang dilakukan oleh *server* dapat berupa akses *database*, komputasi, verifikasi, dan authentifikasi. Untuk melakukan proses-proses tersebut, perlu dilakukan pengembangan *web service* yang efisien agar dapat meminimalisir *delay* yang terjadi pada *server processing*.

Performa pada *web service* dapat berbeda berdasarkan bahasa pemrograman yang digunakan (Sagayaraj, 2013). Oleh karena itu, dalam melakukan pengembangan *web service* kita perlu memilih bahasa pemrograman yang tepat. Bahasa pemrograman sendiri adalah bahasa buatan yang didesain untuk mengekspresikan komputasi yang dapat dilakukan oleh mesin, umumnya adalah komputer (Oguntunde, 2012). Bahasa pemrograman dapat digunakan untuk mengatur perilaku mesin, untuk mengekspresikan algoritme dengan benar, atau hanya sebagai alat bagi manusia untuk berkomunikasi dengan mesin.

Pada umumnya, komputer tidak dapat mengerti bahasa pemrograman *high-level* secara langsung. Agar komputer dapat mengerti program yang ditulis dengan bahasa pemrograman *high-level* dibutuhkan penerjemah yang digunakan untuk menerjemahkan program ke dalam bahasa mesin. Penerjemah tersebut dinamakan *language processors*. *Language processors* sendiri memiliki tiga kelas, yaitu *assembler, interpreter, dan compiler* (Oguntunde, 2012). Masing-masing *language processors* yang ada pada bahasa pemrograman memiliki perbedaan arsitektur yang digunakan dalam memproses suatu kode program sampai dapat dieksekusi oleh komputer. Hal ini menyebabkan perbedaan penggunaan sumber daya yang ada pada perangkat komputer, seperti jumlah *memory,* dan waktu komputasi CPU (*Central Processing Unit*) yang dibutuhkan oleh bahasa

pemrograman (Oguntunde, 2012). Pada *web service* hal tersebut dapat mempengaruhi *server response time* secara signifikan.

Dari permasalahan tersebut, maka perlu dilakukan penelitian untuk menentukan bahasa pemrograman mana yang lebih efisien dan memiliki performa lebih baik untuk digunakan pada *web service*. Hal tersebut dapat dilihat dari perbedaan penggunaan suberdaya pada komputer, seperti *memory usage,* CPU *usage,* dan *execution time* (Sagayaraj, 2013). Setelah pengembangan *web service* akan diperlukan aplikasi *client* untuk mengakses *web service* tersebut. Aplikasi *client* akan dikembangkan pada sistem operasi berbasis Android. Penulis memilih sistem operasi berbasis Android karena dengan menggunakan sistem operasi berbasis Android akan lebih mudah melakukan kostumisasi dan analisis terhadap paket data yang diterima maupun dikirim melalui aplikasi *client*.

1.2 Rumusan Masalah

Berdasarkan latar belakang tersebut, maka dapat dikaji rumusan masalah sebagai berikut:

1. Bagaimana menganalisis penggunaan CPU (*Central Processing Unit*), *memory,* dan kecepatan eksekusi perintah dari masing-masing bahasa pemrograman yang digunakan sebagai *web service* untuk aplikasi *client* berbasis Android ?
2. Bagaimana hasil dari analisis penggunaan CPU (*Central Processing Unit*), *memory,* dan kecepatan eksekusi perintah dari masing-masing bahasa pemrograman yang digunakan sebagai *web service* untuk aplikasi *client* berbasis Android ?
3. Manakah bahasa pemrograman yang memiliki performa lebih tinggi digunakan pada *web service* untuk aplikasi *client* berbasis Android ?

1.3 Tujuan

Tujuan dari penelitian ini adalah:

1. Menganalisis penggunaan CPU (*Central Processing Unit*), *memory,* dan kecepatan eksekusi perintah dari masing-masing bahasa pemrograman yang digunakan sebagai *web service* untuk aplikasi *client* berbasis Android.
2. Mengetahui hasil analisis penggunaan CPU (*Central Processing Unit*), *memory,* dan kecepatan eksekusi perintah dari bahasa pemrograman Python, PHP, dan Perl pada *web service* untuk aplikasi *client* berbasis Android.
3. Mengetahui bahasa pemrograman manakah yang memiliki performa lebih tinggi digunakan pada *web service* untuk aplikasi *client* berbasis Android.

1.4 Manfaat

Manfaat dari penelitian ini adalah:

1. Mengetahui kelebihan dan kekurangan dari masing-masing bahasa pemrograman yang digunakan pada *web service* untuk aplikasi *client* berbasis Android.

2. Mempermudah pengembang aplikasi perangkat bergerak untuk mengembangkan aplikasi yang bersifat *real-time*.

1.5 Batasan Masalah

Berikut adalah batasan masalah pada penelitian ini:

1. Aplikasi *client* menggunakan *platform* Android 5.1.1 (*Lollipop*).
2. *Web service* menggunakan CGI (*Common Gateway Interface*) pada *server* dengan spesifikasi: CPU 1,8GHz, RAM 6GB, HDD 5.400 RPM, dan berjalan pada Apache *web server* versi 2.4.23 dengan sistem operasi Windows 10.
3. Bahasa pemrograman yang digunakan untuk pengembangan *web server* adalah Python 2.7, PHP 5.6.28, dan Perl 5.24.
4. Parameter pengujian peformansi *web service* yaitu, kecepatan pengiriman data, kecepatan proses, penggunaan RAM, utilitas CPU, dan ukuran data yang diproses dari masing-masing bahasa pemrograman yang digunakan.
5. Pengujian dilakukan dengan menggunakan jaringan lokal yang terdiri dari 2 *host* yang bertugas sebagai *client* dan *server*.
6. Pengujian dilakukan dengan melakukan *string concatenation* pada *web service* dengan masing-masing bahasa pemrograman.

1.6 Sistematika Pembahasan

Untuk mencapai tujuan yang diharapkan, maka sistematika pembahasan yang disusun dalam skripsi ini adalah sebagai berikut:

BAB I PENDAHULUAN

Bab ini menguraikan latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat dari penelitian, dan sistematika penulisan mengenai perlunya penelitian tentang "Analisis Perbandingan Performa *Web Service* Menggunakan Bahasa Pemrograman Python, PHP, dan Perl Pada *Client* Berbasis Android".

BAB II TINJAUAN PUSTAKA

Bab ini berisi teori-teori pendukung dan bahan penelitian mengenai teknologi yang diimplementasikan pada penelitian ini. Teori yang diambil mengenai sistem operasi Android, *Web Service*, Python, PHP, Perl, Java, JSON, dan CGI.

BAB III METODOLOGI PENELITIAN

Bab ini menguraikan tentang metode dan langkah kerja yang dilakukan dalam penulisan skripsi yang terdiri dari studi literatur, metode analisis kebutuhan, perancangan sistem, implementasi, metode dan pengambilan kesimpulan serta metode lain yang relevan dengan penelitian ini.

BAB IV PERANCANGAN

Perancangan pada bab ini terdiri atas tiga bagian, yaitu analisis kebutuhan dan perancangan perangkat lunak baik pada sisi *server* maupun pada sisi *client* serta perancangan lingkungan pengujian sistem.

BAB V IMPLEMENTASI

Bab ini akan membahas tentang implementasi *web service* menggunakan bahasa pemrograman Python, PHP, dan Perl. Serta dibahas juga implementasi aplikasi *client* berbasis Android menggunakan *web service*.

BAB VI PENGUJIAN DAN ANALISIS HASIL

Bab ini membahas mengenai teknik dan metode pengujian yang dilakukan pada penelitian ini. Setelah dilakukan pengujian selanjutnya akan dilakukan analisis hasil pengujian performa dari masing-masing bahasa pemrograman yang telah diimplementasikan.

BAB VII PENUTUP

Pada bab ini membahas tentang kesimpulan dan saran dari pelaksanaan penelitian "Analisis Perbandingan Performa *Web Service* Menggunakan Bahasa Pemrograman Python, PHP, dan Perl Pada *Client* Berbasis Android" ini.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Landasan Pustaka

Pada tahun 2010 Hatem Hamad, Motaz Saad dan Ramzi Abed melakukan evaluasi performa RESTful *web service* pada perangkat bergerak. Dalam penelitian tersebut mereka mengembangkan RESTful dan SOAP *web service,* mereka melakukan *string concatenation* dan *float number addition* untuk mengetahui metode mana yang lebih bagus. Dari hasil evaluasi yang mereka lakukan menunjukkan bahwa keunggulan meggunakan RESTful *web service* pada perangkat bergerak. Keunggulannya adalah ukuran pesan dan *response time* yang lebih sedikit. Oleh karena itu, RESTful menawarkan solusi yang bagus untuk implementasi secara umum dengan *flexibility* yang tinggi dan *overhead* yang lebih rendah (Hamad, 2010).

Pada tahun 2012 Onlyjob melakukan analisis performa bahasa pemrograman Perl, Python, Ruby, PHP, C, C++, Lua, TCL, Javascript, dan Java dengan melakukan *string concatenation*. Dalam analisis yang telah dilakukan mendapatkan kesimpulan bahwa bahasa pemrograman Perl mempunyai kecepatan eksekusi yang paling cepat dibandingkan dengan bahasa pemrograman yang lain, kemudian dalam penggunaan *memory* bahasa pemrograman C menggunakan *memory* paling sedikit dibandingkan dengan bahasa pemrograman yang lain (Onlyjob, 2012).

Pada tahun 2013 S. Sagayaraj dan M. Santhosh Kumar melakukan evaluasi performa *web service* menggunakan bahasa pemrograman C#, Java, dan PHP pada perhitungan kalkulator. Dalam evaluasi yang mereka lakukan mendapatkan kesimpulan bahwa dengan menggunakan bahasa C# memerlukan 36% CPU yang digunakan, dan waktu proses yang dibutuhkan 0,0037 detik, kemudian *memory* yang digunakan 400MB. Lalu dengan bahasa Java memerlukan 34% CPU yang digunakan, dan waktu proses yang dibutuhkan 0,0034 detik, kemudian *memory* yang digunakan 375MB. Dan terakhir dengan bahasa PHP memerlukan 30% CPU yang digunakan, dan waktu yang dibutuhkan 0,0030 detik, kemudian *memory* yang digunakan 380,2 MB (Sagayaraj, 2013).

Pada tahun 2014 Amarpreet Singh Johal dan Baljit Singh melakukan analisis performa *web service* pada perangkat bergerak berbasis Android. Dalam analisis yang mereka lakukan menggunakan metode SOAP dan REST mereka membandingkan metode mana yang lebih cepat digunakan pada *web service* untuk perangkat bergerak berbasis Android. Mereka menyimpulkan bahwa REST memiliki *response time* yang lebih bagus daripada SOAP. Terdapat beberapa faktor yang mempengaruhi *response time* seperti *server processing capabilities, network bandwidth, payloads,* jarak antara *client* dan *server* serta banyak *client* yang mengakses *web service* tersebut (Johal, 2014).

2.2 Python

Python adalah bahasa pemrograman yang bersifat *open source*. Bahasa pemrograman ini dioptimalisasikan untuk *software quality, developer*

productivity, *program portability*, dan *component integration* (Lutz, 2010). Python telah digunakan untuk mengembangkan berbagai macam perangkat lunak, seperti *internet scripting*, *systems programming*, *user interfaces*, *product customization*, *numberic programming* dll. Python saat ini telah menduduki posisi 4 atau 5 bahasa pemrograman paling sering digunakan di seluruh dunia (Lutz, 2010).

Bahasa pemrograman Python memiliki beberapa fitur yang dapat digunakan oleh pengembang perangkat lunak. Berikut adalah beberapa fitur yang ada pada bahasa pemrograman Python (Lutz, 2010):

1. ***Multi Paradigm Design***

   Bahasa pemrograman Python mendukung beberapa paradigma pemrograman, yaitu OOP (*Object Oriented Programming*), *functional*, dan *modular structures* (Lutz, 2010). Dengan demikian pengembang dapat memilih paradigma pemrograman sesuai dengan kebutuhan.

2. ***Open Source***

   Bahasa pemrograman Python adalah bahasa pemrograman yang bersifat *open source*. *Open source* sendiri adalah perangkat lunak atau sejenisnya yang dapat digunakan secara bebas digunakan, diubah, dan disebarkan (baik termodifikasi maupun tidak) oleh semua orang (OSI, 2016). Dengan demikian pengembang dapat dengan bebas memodifikasi bahasa pemrograman sesuai dengan kebutuhan.

3. ***Simplicity***

   Bahasa pemrograman Python merupakan bahasa pemrograman yang simpel dan mudah dipelajari. Bahasa pemrograman Python sangat mirip dengan dengan *pseudo-code,* sehingga pengembang dapat langsung menggunakan *pseudo-code* sebagai kode program dengan sedikit modifikasi (Swaroopch, 2016). Hal ini dapat mempercepat proses pengembangan perangkat lunak.

4. ***Library Support***

   Terdapat banyak *library* yang dapat digunakan oleh pengembang. *Library* yang ada pada bahasa pemrograman Python bersifat *cross-platform* yang kompatibel pada sistem operasi berbasis UNIX, Windows, maupun Mac OS (tutorialspoint.com (b), 2016).

5. ***Portability***

   Bahasa pemrograman Python bersifat *portability*. Bahasa pemrograman Python dapat berjalan pada berbagai macam *platform hardware* maupun sistem operasi serta mempunyai *interface* yang sama pada semua *platform* (tutorialspoint.com (b), 2016).

6. ***Extendable***

   Pengembang perangkat lunak dapat menambahkan modul-modul bahasa pemrograman *low-level*. Modul-modul tersebut dapat memudahkan pengembang

untuk mengkostumasi perangkat lunak mereka sehingga lebih efisien (tutorialspoint.com (b), 2016).

7. ***Scalability***

Bahasa pemrograman Python menyediakan struktur yang baik, sehingga dapat men-*support* perangkat lunak berskala besar lebih baik dibandingkan dengan bahasa pemrograman *shell scripting* (tutorialspoint.com (b), 2016).

2.3 PHP

PHP adalah bahasa pemrograman yang biasa digunakan untuk *server-side scripting*. PHP merupakan bahasa pemrograman yang simpel namun *powerful* dan tepat untuk digunakan pada *web server* (Tatroe, 2013). PHP dapat berjalan pada semua sistem operasi yang sering digunakan, seperti UNIX, Windows, dan Mac OS.

Bahasa pemrograman PHP memiliki beberapa fitur yang dapat digunakan oleh pengembang perangkat lunak. Berikut adalah beberapa fitur yang ada pada bahasa pemrograman PHP:

1. ***Multi Paradigm Design***

Pada dasarnya bahasa pemrograman PHP menggunakan paradigma pemrograman *structural*, namun PHP juga memiliki kapabilitas untuk mendukung paradigma OOP (*Object Oriented Programming*).

2. ***Open Source***

Bahasa pemrograman PHP adalah bahasa pemrograman yang bersifat *open source*. *Open source* sendiri adalah perangkat lunak atau sejenisnya yang dapat digunakan secara bebas digunakan, diubah, dan disebarkan (baik termodifikasi maupun tidak) oleh semua orang (OSI, 2016). Dengan demikian pengembang dapat dengan bebas memodifikasi bahasa pemrograman sesuai dengan kebutuhan.

3. ***Simplicity***

Bahasa pemrograman PHP memiliki *syntax-syntax* yang simpel dan mudah dipelajari. *Syntax* yang ada pada PHP hampir sama dengan *syntax* yang ada pada bahasa pemrograman yang lain, sehingga pengembang dapat mempelajari PHP dengan cepat tanpa harus mempelajari *syntax-syntax* dari awal.

4. ***Framework Support***

Terdapat banyak pilihan *framework* dapat digunakan pada bahasa pemrograman PHP, seperti CI (*Code Igniter*), Laravel, Symfony, dll. Dengan adanya banyak pilihan *framework* yang ada dapat memudahkan pengembang dalam pengembangan perangkat lunak, karena dengan menggunakan *framework* akan mempercepat proses pengembangan perangkat lunak.

5. ***Library Support***

Terdapat berbagai macam *library* yang dapat digunakan oleh pengembang PHP, seperti ADOdb digunakan untuk membantu abstraksi *database* pada bahasa pemrograman PHP dan masih banyak lagi *library* lain. Dengan menggunakan *library* dapat mempercepat maupun mengoptimalisasi perangkat lunak yang sedang dikembangkan.

6. ***Extendable***

Pengembang perangkat lunak dapat menambahkan modul-modul bahasa pemrograman lain, salah satu contohnya adalah Javascript. Javascript sendiri merupakan bahasa pemrograman turunan Java yang dikhususkan untuk pengembangan perangkat lunak berbasis *scripting*. Dengan menggunakan Javascript dapat membantu pengembang apabila modul yang dibutuhkan tidak tersedia maka pengembang dapat menggunakan modul-modul yang ada pada Javascript.

7. ***Portability***

Bahasa pemrograman PHP bersifat *portability*. Bahasa pemrograman PHP dapat berjalan pada berbagai macam *platform hardware* maupun sistem operasi serta mempunyai *interface* yang sama pada semua *platform* (tutorialspoint.com (b), 2016).

8. ***Familiarity***

Bahasa pemrograman PHP telah digunakan di banyak *website* yang tersebar di *internet*. Banyak pengembang perangkat lunak yang telah menggunakan bahasa pemrograman ini. Hal ini akan memudahkan pengembang untuk mempelajari dan mengembangkannya.

2.4 Perl

Bahasa pemrograman Perl pada awalnya dikembangkan untuk manipulasi teks, namun seiring dengan berjalannya waktu Perl mulai digunakan untuk mengembangkan beberapa macam sistem, seperti sistem administrasi, pengembangan *web*, pemrograman jaringan, pengembangan GUI (*Graphical User Interface*), dll (Siever, 1998). Saat ini Perl merupakan salah satu bahasa pemrograman yang sering digunakan untuk mengembangkan perangkat lunak *web server*.

Bahasa pemrograman Perl memiliki beberapa fitur yang dapat digunakan oleh pengembang perangkat lunak. Berikut adalah beberapa fitur yang ada pada bahasa pemrograman Perl:

1. ***Multi Paradigm Design***

Bahasa pemrograman Perl mendukung beberapa paradigma pemrograman, yaitu fungsional, *imperative*, OOP (*Object Oriented Programming*), *refective*, *procedural*, *event-driven*, maupun paradigma *generic*.

2. *Open Source*

Bahasa pemrograman Perl bersifat *open source*. *Open source* sendiri adalah perangkat lunak atau sejenisnya yang dapat digunakan secara bebas digunakan, diubah, dan disebarkan (baik termodifikasi maupun tidak) oleh semua orang (OSI, 2016). Dengan demikian pengembang dapat dengan bebas memodifikasi bahasa pemrograman sesuai dengan kebutuhan.

3. *Simplicity*

Bahasa pemrograman Perl termasuk dalam bahasa pemrograman yang tergolong simpel namun *powerful* (Siever, 1998). Perl mengadaptasi modul-modul penting yang ada pada bahasa pemrograman lain, seperti C, Awk, Sed, SH, dan BASIC. Sehingga akan memudahkan pengembang dalam mengembangan perangkat lunak.

4. *Framework Support*

Perl mempunyai beberapa pilihan *framework* yang dapat digunakan, seperti catalyst, dancer, dll. Dengan adanya banyak pilihan *framework* yang ada dapat memudahkan pengembang dalam pengembangan perangkat lunak, karena dengan menggunakan *framework* akan mempercepat proses pengembangan perangkat lunak.

5. *Library Support*

Perl mempunyai beberapa pilihan *library* yang dapat digunakan. Lebih dari 20.000 *library* tersedia di CPAN (Perl *Comprehensive* Perl *Archive*) (tutorialspoint.com (a), 2016).

6. *Extendable*

Perl dapat mengakses *library* yang ada pada bahasa pemrograman C/C++ dengan menggunakan XS atau SWIG.

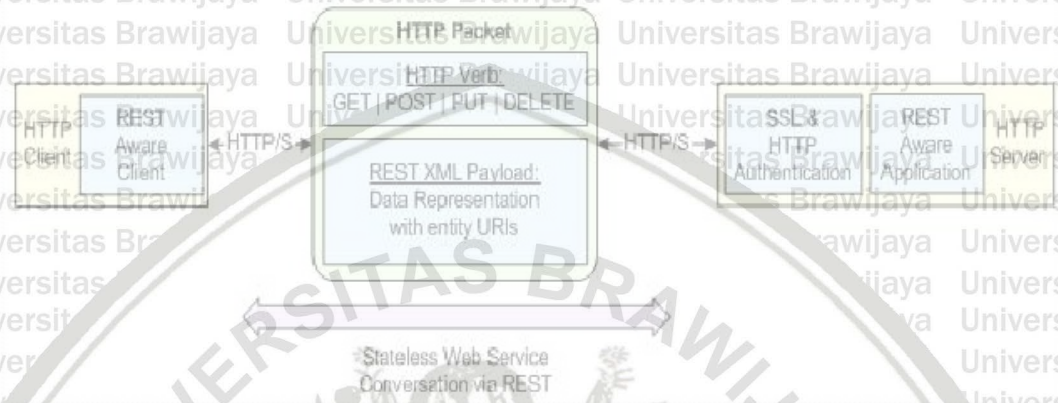
7. *Portability*

Bahasa pemrograman Perl dapat berjalan pada bebagai macam *platform hardware* maupun sistem operasi serta mempunyai kapabilitas untuk digunakan sebagai *embedded system*.

2.5 *Web Service*

Web service adalah sistem perangkat lunak yang didesain untuk membantu interaksi antara mesin dengan mesin pada sebuah jaringan (W3C, 2004). *Web service* pada awalnya menggunakan sebuah *interface* mesin untuk berkomunikasi dengan mesin yang lain (seperti WSDL). Namun seiring berjalannya waktu *web service* kini lebih banyak menggunakan SOAP dan REST *messages*, biasanya pesan akan dikonversi menggunakan HTTP dengan format JSON atau XML sesuai dengan standar *web* yang ada.

Pada penelitian ini akan menggunakan arsitektur *web service* berbasis REST. REST sendiri adalah arsitektur *web service* yang dimodelkan berdasarkan data yang disediakan, diakses, dan dimodifikasi pada *web* (Hamad, 2010). Pada *web service* dengan arsitektur REST, data dan fungsional dianggap sebagai *resourses,* dan *resources* tersebut diakses menggunakan *Uniform Resourses Identifiers* (URIs), biasanya berupa *link* pada *web*. Diagram arsitektur REST *web service* dapat dilihat pada Gambar 2.1.



Gambar 2.1 Arsitektur REST *Web Service*

Sumber: Hamad (2010)

1. **HTTP *Client*.** Merupakan *client* yang akan mengakses *web service*.
2. **HTTP *Packet*.** HTTP *packet request* dapat berupa GET, POST, PUT, atau DELETE. Sedangkan, HTTP *packet response* dapat berupa XML maupun JSON *payload*.
3. **HTTP *Server*.** Merupakan *server* tempat *web service* diimplementasikan.

Terdapat beberapa HTTP *method* yang dapat diimplementasikan pada REST *web service,* diantaranya:

1. **GET.** *Method* yang digunakan untuk menerima *resources*.
2. **POST.** *Method* yang digunakan untuk membuat *resources*.
3. **PUT.** *Method* yang digunakan untuk meng-*update resources*.
4. **DELETE.** *Method* yang digunakan untuk menghapus *resources*.

2.6 CGI (*Common Gateway Interface*)

Common Gateway Interface (CGI) merupakan bagian dari *web server* yang dapat berkomunikasi dengan program lain yang berjalan pada *server* (Kathuria, 2014). Dengan menggunakan CGI, *web server* dapat memanggil program yang ada maupun yang sedang berjalan pada *server*. *Web server* akan mengirimkan *user data* ke program menggunakan CGI, kemudian program akan memproses data dan memberikan respon kembali ke *web server*.

Gambar 2.2 Model Sederhana CGI

Sumber: Kathuria (2014)

Gambar 2.2 menjelaskan model sederhana dari *Common Gateway Interface* (CGI). Pada awalnya *user* akan mengirimkan *form* yang telah di *submit* ke *server*, kemudian *server* akan memanggil program menggunakan CGI, selanjutnya program akan memproses data dan mengirimkan respon kembali ke *server* dan melanjutkannya ke komputer *user*.

Common Gateway Interface (CGI) telah menjadi standar dalam pengembangan aplikasi pada *web server*, banyak *programmer* telah memodifikasi *script* CGI sesuai dengan kebutuhannya (Kathuria, 2014). CGI juga dapat berjalan pada berbagai *platform web server* yang ada di *world wide web* (WWW). Selain itu CGI mempunyai beberapa kekurangan, CGI mempunyai beberapa isu keamanan. Dengan menggunakan CGI akan membuat seseorang dapat menjalankan program pada *server*. Oleh karena itu program yang menggunakan CGI harus dipisahkan pada *directory* tertentu, sehingga seseorang tidak dapat menjalankan program menggunakan CGI diluar *directory* tersebut.

BAB 3 METODOLOGI

Metodologi penelitian menjelaskan metode yang digunakan dalam analisis perbandingan performa *web service* menggunakan bahasa pemrograman Python, PHP, dan Perl pada *client* berbasis Android. Tahapan metodologi penelitian dapat dilihat pada Gambar 3.1.

Gambar 3.1 Gambar Blok Metodologi Penelitian

3.1 Studi Literatur

Mempelajari literatur dari berbagai bidang ilmu yang berhubungan dengan analisis performa *web service* menggunakan bahasa pemrograman Python, PHP, dan Perl pada *client* berbasis Android adalah sebagai berikut:

1. Kajian pustaka dari penelitian sebelumnya
2. Pengembangan *web service*
3. Pengembangan aplikasi perangkat bergerak berbasis Android
4. Bahasa pemrograman Java
5. Bahasa pemrograman Python
6. Bahasa pemrograman PHP
7. Bahasa pemrograman Perl
8. Pengujian perangkat lunak

3.2 Analisis Kebutuhan

Analisis kebutuhan dilakukan untuk mengetahui kebutuhan perangkat lunak yang dibuat, baik kebutuhan dari sistem *web service* maupun pada aplikasi *client*. Kebutuhan sistem berdasarkan pada parameter analisis performa *web service* yang akan dilakukan yaitu, penggunaan CPU (*Central Processing Unit*), penggunaan RAM (*Random Access Memory*), dan kecepatan eksekusi perintah serta berdasarkan pada pengalaman peneliti dan penelitian-penelitian sebelumnya. Kebutuhan tersebut dapat berupa fungsi maupun proses yang ada pada *web service* dan aplikasi *client*.

3.3 Perancangan Sistem

Perancangan sistem dilakukan untuk mengetahui rancangan langkah kerja dari sistem secara menyeluruh, baik dari segi model maupun dari segi arsitektur untuk mempermudah implementasi sistem. Tahap-tahap perancangan sistem dapat dilihat pada Gambar 3.2.

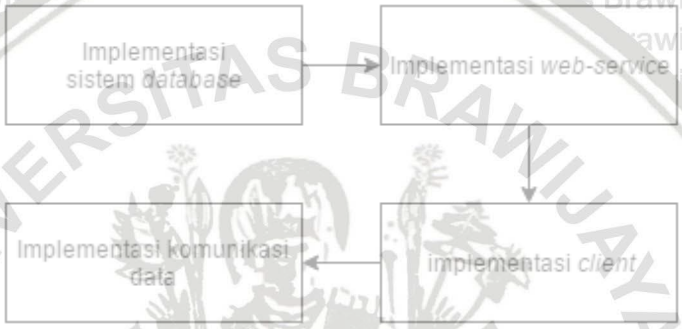
Gambar 3.2 Tahap-tahap Perancangan Sistem

Berikut adalah Jenis-jenis model perancangan yang akan digunakan:

1. **Diagram *Use Case*.** Digunakan untuk merancang interaksi aktor dengan sistem yang akan dibuat.
2. **Diagram *Activity*.** Digunakan untuk merancang aktivitas sistem yang akan dibuat.
3. ***Flowchart*.** Digunakan untuk merancang sistem *web service* dan aplikasi *client*.

3.4 Implementasi Sistem

Implementasi sistem adalah tahap membangun sistem berdasarkan pada perancangan sistem dan menerapkan hal yang telah didapat pada tahap studi literatur. Tahap-tahap implementasi sistem dapat dilihat Gambar 3.3.



Gambar 3.3 Tahap-tahap Implementasi Sistem

1. **Implementasi Sistem *Database*.** Implementasi sistem *database* menggunakan MySQL.
2. **Implementasi *Web Service*.** Implementasi *web service* menggunakan bahasa pemrograman Python, PHP, dan Perl.
3. **Implementasi *Client*.** Implementasi client menggunakan *platform* Android dengan bahasa pemrograman Java.
4. **Implementasi Komunikasi Data.** Implementasi komunikasi data menggunakan protokol HTTP (*Hypertext Transfer Protocol*) dengan format JSON.

3.5 Pengujian Sistem

Pengujian performa *web service* terbagi menjadi dua elemen yaitu kuantitatif dan kualitatif. Kuantitatif meliputi penggunaan CPU (*Central Processing Unit*), Penggunaan RAM (*Random Access Memory*), kecepatan eksekusi perintah, perbandingan SOAP / REST *message*, dan jumlah baris kode program. Sedangkan kualitatif meliputi metodologi, *scalability*, *security*, dan *maintainability* (Sagayaraj, 2013). Pengujian sistem pada penelitian ini akan menggunakan pengujian elemen kuantitatif dengan parameter penggunaan CPU (*Central Processing Unit*), penggunaan RAM (*Random Access Memory*), dan kecepatan eksekusi perintah.

Terdapat dua pengujian yang akan dilakukan, yaitu pengujian dari sisi *client* dan *web service*. Pengujian pada sisi *client* dilakukan dengan cara mengirimkan *request* ke *web service* dan menghitung *response time* dari *web service.* Pada sisi *web service* sendiri juga dilakukan analisis penggunaan CPU (*Central Processing Unit*) dan penggunaan RAM (*Random Access Memory*) pada saat proses *request* sedang berlangsung. Pengujian tersebut dilakukan pada bahasa pemrograman Python, PHP, dan Perl.

3.6 Penarikan Kesimpulan

Setelah tahap-tahap di atas selesai, maka tahap selanjutnya adalah penarikan kesimpulan dan saran. Pada tahap ini akan dilakukan penarikan kesimpulan manakah bahasa pemrograman yang memiliki performa lebih baik di antara Python, PHP, dan Perl dengan parameter penggunaan CPU (*Central Processing Unit*), penggunaan RAM (*Random Access Memory*), dan kecepatan eksekusi perintah. Sedangkan saran dimaksudkan untuk memperbaiki kesalahan dan menyempurnakan penelitian serta untuk menjadi pertimbangan dalam pengembangan selanjutnya.

BAB 4 PERANCANGAN

4.1 Analisis Kebutuhan Sistem

Proses analisis kebutuhan ini diawali dengan penjabaran gambaran umum sistem, identifikasi aktor, penjabaran tentang daftar kebutuhan dan dimodelkan ke dalam diagram *use case*. Analisis kebutuhan ini bertujuan untuk menggambarkan kebutuhan-kebutuhan yang harus disediakan oleh sistem agar dapat memenuhi kebutuhan pengujian sistem.

4.1.1 Gambaran Umum Sistem

Web service merupakan sistem yang dirancang untuk menyajikan data yang dibutuhkan oleh aplikasi *user* sesuai dengan *request* yang telah dikirim. Setelah menerima *request, web service* akan memproses *request* tersebut dan selanjutnya memberikan respon berupa data JSON kepada aplikasi *client*.

Gambar 4.1 Gambaran Umum Sistem

Gambar 4.1 menjelaskan bahwa *client* akan mengirimkan *request* ke *server* melalui HTTP, kemudian *web service* akan mengambil dan memproses data dari *database* sesuai dengan *request* yang dikirimkan oleh *client*. Selanjutnya *web service* akan mengirimkan respon berupa JSON yang selanjutnya akan di proses kepada *client*.

4.1.2 Identifikasi Aktor

Sistem yang akan dibuat hanya memiliki satu aktor yaitu *user*. *User* adalah pengguna aplikasi *client*. *User* akan mengirimkan dan memilih menu sesuai kebutuhan, kemudian aplikasi akan mengirimkan *request* kepada *web service*.

4.1.3 Daftar Kebutuhan

Tabel daftar kebutuhan dapat dilihat pada Tabel 4.1.

Tabel 4.1 Tabel Kebutuhan Fungsional

Nomor SRS	Kebutuhan	Use Case	Sisi
SRS_001_01	Pengguna mengirimkan *request* semua berita.	Meminta data berita	*client*
SRS_001_02	Pengguna mengirimkan *request* berita berdasarkan kategori.	Meminta data berita sesuai kategori	*client*

SRS_001_03	Sistem *web service* mengolah *request* dari *client* kemudian memberikan respon berdasarkan *request* yang dikirim.	Mendapatkan data berita	*web service*

Terdapat satu parameter kebutuhan non-fungsional sistem yaitu parameter *compability* dengan penjelasan bahwa aplikasi *client* hanya dapat berjalan pada sistem operasi Android dengan minimal SDK 15 (*Ice Cream Sandwitch*).

4.1.4 Diagram *Use Case*

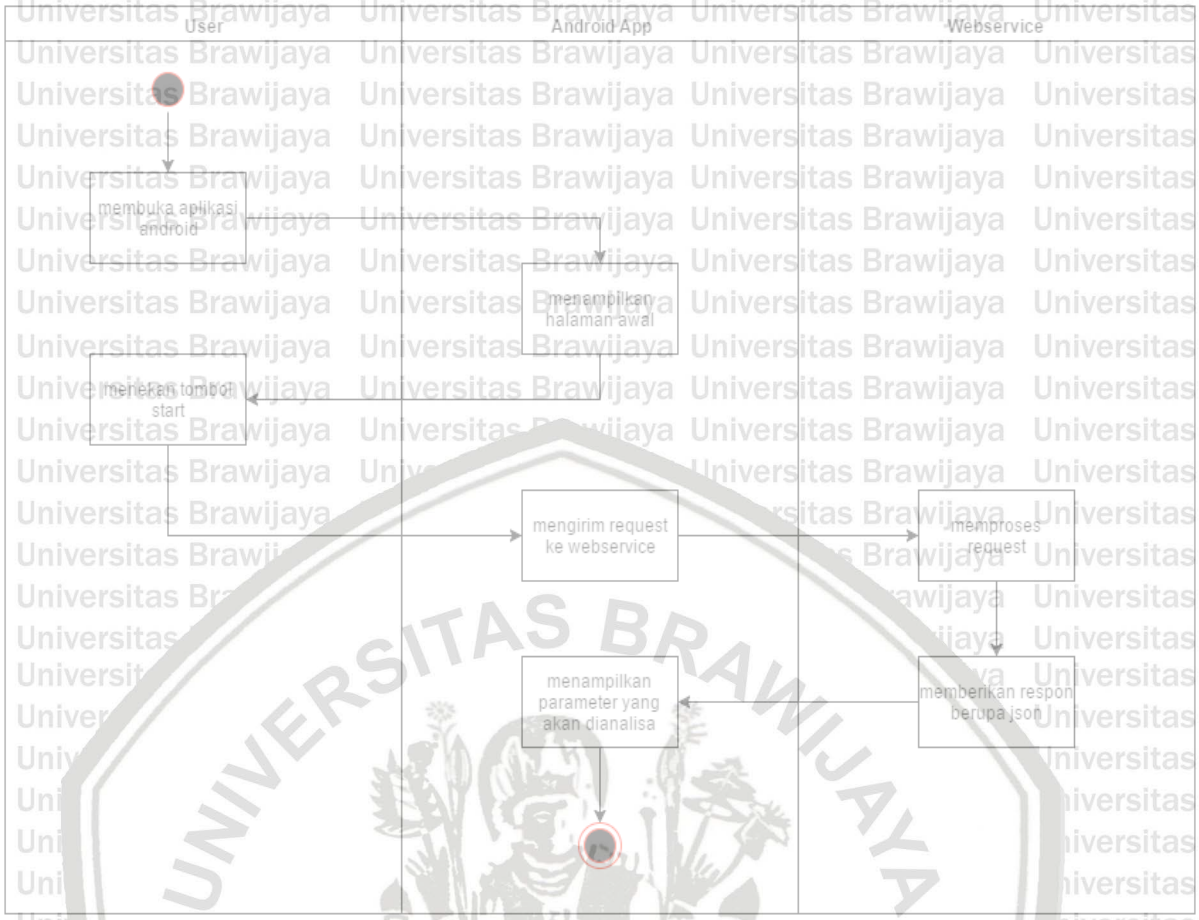
Berikut adalah pemodelan *use case* sistem diperoleh dari kebutuhan fungsional yang digunakan untuk menggambarkan interaksi antara aktor dengan sistem yang dibuat.

Gambar 4.2 *Use Case Diagram*

Gambar 4.2 merupakan *use case* sistem yang akan dibuat. Pada Gambar 4.2 tersebut digambarkan *user* dapat mengirimkan *request* data kemudian *web service* akan memberikan respon data sesuai dengan yang telah di *request*.

4.1.5 Diagram *Activity*

Diagram *Activity* merupakan diagram untuk memodelkan aktivitas. Antara pengguna dengan sistem yang berjalan berdasarkan skenario *use case* yang digambarkan pada Gambar 4.2 dengan diagram *activity,* dapat terlihat aktor yang melakukan tiap proses tiap langkahnya.



Gambar 4.3 Diagram Aktivitas Sistem

Gambar 4.3 adalah diagram aktivitas sistem yang dibuat untuk menjelaskan interaksi pengguna dengan sistem.

4.2 Perancangan Sistem

Proses perancangan sistem memiliki lima tahap, yaitu perancangan arsitektur sistem, perancangan *database*, perancangan aplikasi Android, perancangan antarmuka sistem, dan perancangan *web service.*

4.2.1 Perancangan Arsitektur Sistem

Sistem *web service* akan dibagi menjadi tiga bahasa pemrograman yaitu: Python, PHP, dan Perl dengan menggunakan CGI (*Common Gateway Interface*) sebagai penghubung antara *code interpreter* dengan *web server*. Arsitektur sistem dapat dilihat pada Gambar 4.4.

Gambar 4.4 Arsitekur Sistem

Gambar 4.4 menjelaskan bahwa *client* akan mengirimkan HTTP *request* ke *web server* kemudian *web server* akan memanggil *web service* yang telah diimplementasikan dengan menggunakan bahasa permograman Python, PHP, dan Perl melalui CGI (*Common Gateway Interface*). Setelah selesai memproses data, *web service* akan mengirimkan kembali data tersebut ke *web server* dan mengirimkan respon data tersebut kepada *client* dalam bentuk JSON. Selanjutnya aplikasi *client* akan menampilkan data yang akan digunakan untuk analisis.

4.2.2 Perancangan *Database* Sistem

Database berfungsi sebagai tempat penyimpanan data yang digunakan oleh sistem. Perancangan *database* digunakan untuk merancang *database* yang akan dibuat agar masukan dan keluaran program sesuai dengan apa yang diharapkan. Perancangan *database* dibuat berdasarkan dari proses analisis kebutuhan sistem. Tabel database *news* dapat dilihat pada Tabel 4.2.

Tabel 4.2 Perancangan *Database* Sistem

news	
id	Integer
title	text
Content	Text
date	integer

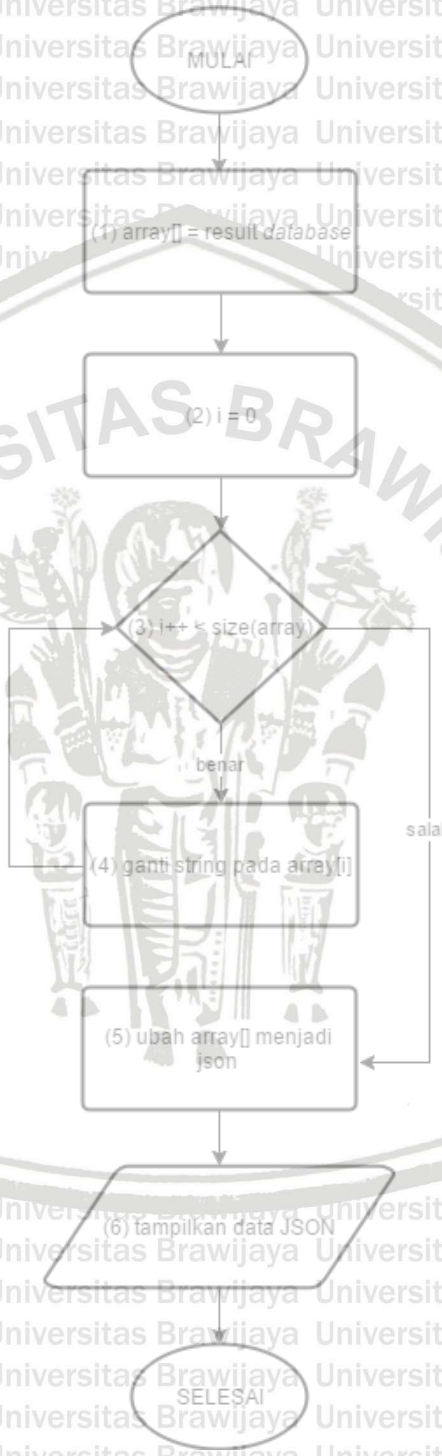
Deskripsi tabel *database* dapat dilihat pada Tabel 4.3.

Tabel 4.3 Deskripsi Tabel *News*

No.	Nama Field	Tipe	Keterangan
1.	id	Integer	Sebagai id berita.
2.	title	Text	Sebagai judul berita.
3.	content	Text	Sebagai isi konten dari berita.
4.	date	Integer	Sebagai tanggal berita diterbitkan.

4.2.3 Perancangan *Web Service*

Algoritme *web service* digunakan untuk menerima *request* dari aplikasi *mobile* kemudian memprosesnya dalam *web service* dan mengirimkan hasilnya ke aplikasi *mobile*. Perancangan algoritme *web service* dapat dilihat pada Gambar 4.5.



Gambar 4.5 Algoritme Fungsi *Web Service*

Sumber: Diadaptasi dari Onlyjob (2012)

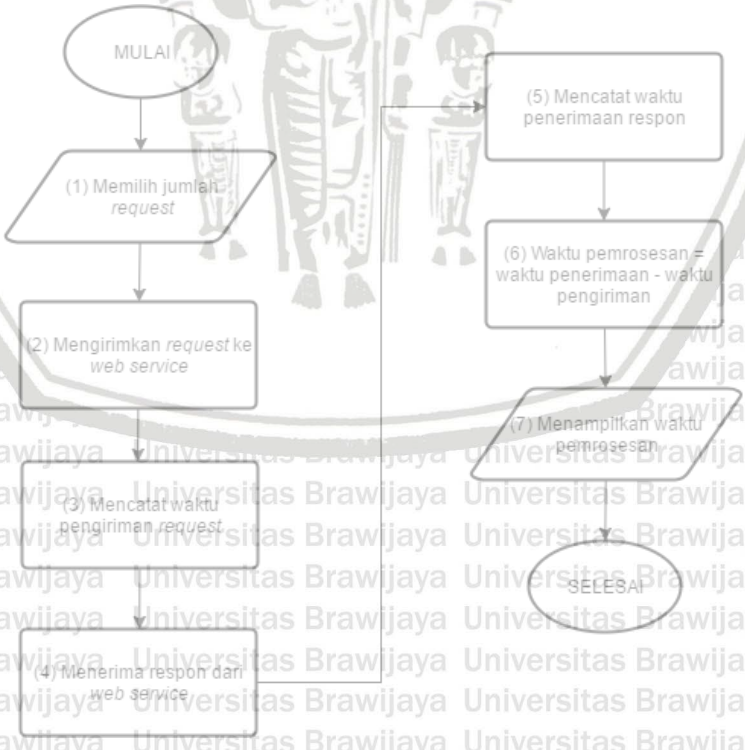
Berikut ini adalah penjelasan dari Gambar 4.5:

1. *Web service* mengambil data dari *database* dan disimpan pada *array*.
2. Inisialisasi variabel *integer* i = 0.
3. Jika nilai dari variabel i++ kurang dari ukuran *array* maka masuk ke kondisi benar, jika tidak maka masuk ke kondisi salah.
4. Jika benar, ubah *string* yang ada pada *array index* ke i.
5. Jika salah, lanjut ke proses ubah data *array* menjadi JSON.
6. Menampilkan data JSON.

Pada Gambar 4.5 dijelaskan bahwa *web service* akan mengambil data dari *database* berupa *array* sesuai dengan *request* yang telah diterima, kemudian *web service* akan melakukan perulangan untuk mengganti *string* yang ada pada *array* dengan *string* baru dan mengkonversikan *array* dari hasil proses tersebut menjadi data JSON.

4.2.4 Perancangan Aplikasi *Mobile Client*

Aplikasi *mobile* digunakan untuk mengirimkan *request* ke *web service* dan menerima respon data hasil pemrosesan *web service*. Perancangan aplikasi *mobile* dapat dilihat pada Gambar 4.6.



Gambar 4.6 *Flowchart* Aplikasi *Mobile*

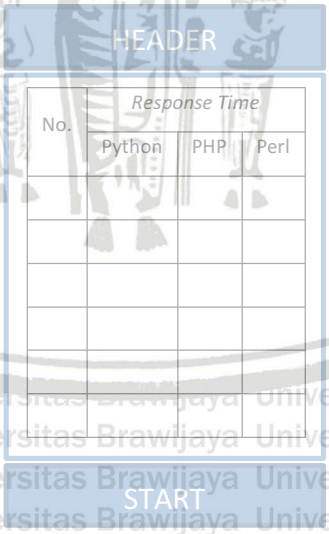
Berikut ini adalah penjelasan dari Gambar 4.6:

1. *User* memilih jumlah *request* yang akan dikirim.
2. Aplikasi *client* mengirimkan *request* ke *web service*.
3. Mencatat waktu pengiriman *request* ke *web service* dan menyimpannya pada *variable* waktu pengiriman.
4. Menerima respon data hasil pemrosesan dari *web service*.
5. Mencatat waktu penerimaan respon dari *web service* dan menyimpannya pada *variable* waktu penerimaan.
6. Mendapatkan waktu pemrosesan dengan cara mengurangi waktu penerimaan dan waktu pengiriman.
7. Menampilkan hasil waktu pemrosesan.

Pada Gambar 4.6 dijelaskan bahwa aplikasi *mobile client* akan mengirimkan *request* ke *web service* dan mencatat waktu pengiriman, setelah *web service* selesai memproses dan mengirimkan respon, aplikasi *mobile client* akan menerima respon tersebut dan mencatat waktu penerimaan respon. Waktu pemrosesan didapatkan dengan cara mengurangkan waktu penerimaan respon dan waktu pengiriman *request.* Setelah waktu pemrosesan akan ditampilkan pada aplikasi *mobile client.*

4.2.5 Perancangan Antarmuka Sistem

Perancangan antarmuka sistem dapat dilihat pada Gambar 4.7.



Gambar 4.7 Rancangan Antarmuka Aplikasi *Mobile*

Berikut ini adalah penjelasan dari Gambar 4.7:

1. *Header* berisikan nama aplikasi.
2. Tabel berisikan tabel dengan empat kolom yaitu:
   a. Kolom nomor berisi nomor urut percobaan yang telah dilakukan.
   b. Kolom Python berisi waktu respon yang dibutuhkan oleh *web service* menggunakan bahasa pemrograman Python.
   c. Kolom PHP berisi waktu respon yang dibutuhkan oleh *web service* menggunakan bahasa pemrograman PHP.
   d. Kolom Perl berisi waktu respon yang dibutuhkan oleh *web service* menggunakan bahasa pemrograman Perl.
3. Tombol *start* digunakan untuk mengirimkan *request* sekaligus memulai perhitungan waktu.

4.3 Perancangan Pengujian Sistem

Perancangan pengujian sistem dilakukan untuk memberi gambaran pengujian sistem yang akan dilakukan sekaligus dengan perangkat-perangkat yang akan digunakan dalam pengujian. Pengujian nantinya akan dibagi menjadi dua sisi, yaitu pengujian dari sisi *client* dan pengujian dari sisi *server*. Parameter-parameter yang digunakan dalam pengujian dapat dilihat pada Tabel 4.4 (Sagayaraj, 2013).

Tabel 4.4 Parameter Pengujian Sistem

No.	Parameter	Satuan	Keterangan
1.	*Response time (client)*	*Milisecond* (*ms*)	Waktu respon yang dibutuhkan oleh *web service*
2.	CPU *usage (server)*	Persentase (*%*)	Penggunaan CPU pada *server*
3.	RAM *usage (server)*	*Byte* (*B*)	Penggunaan RAM pada *server*

Pengujian pada sisi *client* digunakan untuk menguji kecepatan eksekusi perintah dari masing-masing *web service* yang telah diimplementasikan menggunakan bahasa pemrograman Python, PHP, dan Perl. Aplikasi *client* diimplementasikan pada sistem operasi Android yang berfungsi untuk mencatat waktu pengiriman *request* dan penerimaan *response*, sehingga akan didapatkan total waktu proses yang dibutuhkan oleh *web service*. Sedangkan pengujian pada sisi *server* digunakan untuk menguji penggunaan CPU dan penggunaan RAM saat proses eksekusi perintah pada masing-masing bahasa pemrograman. Pengujian penggunaan CPU dan penggunaan RAM pada *server* akan dilakukan dengan menggunakan *Resource Monitor* yang ada pada sistem operasi Windows.

Pengujian nantinya juga akan dibagi menjadi beberapa tahap, masing-masing tahap terdapat perbedaan jumlah data yang akan diproses, hal ini dilakukan untuk mengetahui perbedaan performa masing-masing bahasa pemrograman dalam memproses data dengan jumlah tertentu. Tahap-tahap tersebut dapat dilihat pada Tabel 4.5.

Tabel 4.5 Tahap-tahap Pengujian Sistem

Tahap	Banyak Data	Ukuran Data	Total Ukuran Data
1.	5.000	1KB	5MB
2.	10.000	1KB	10MB
3.	15.000	1KB	15MB
4.	20.000	1KB	20MB
5.	40.000	1KB	40MB

Pengujian nantinya akan dilakukan pada jaringan nirkabel lokal, dengan hanya menggunakan dua *host* yang berperan sebagai *client* dan *server,* hal itu dilakukan untuk menghindari perbedaan beban pada jaringan maupun *router* pada saat pengujian sedang berlangsung. Rancangan lingkungan pengujian sistem dapat dilihat pada Gambar 4.8.

Gambar 4.8 Rancangan Lingkungan Pengujian

BAB 5 IMPLEMENTASI

Bab ini membahas implementasi sistem berdasarkan perancangan perangkat lunak yang telah dibuat. Implementasi sistem pada pembahasan ini terdiri dari spesifikasi perangkat lunak dan perangkat keras, batasan implementasi, implementasi *database*, dan implementasi sistem.

5.1 Spesifikasi Perangkat Lunak dan Perangkat Keras

Web service diimplemetasikan pada *server* dengan spesifikasi CPU 1,8GHz, RAM 6GB, dan HDD 5.400RPM. *Web service* menggunakan CGI (*Common Gateway Interface*) dan diimplementasikan pada Apache *web server* menggunakan *database* MySQL dengan sistem operasi Windows 10. Aplikasi *client* diimplementasikan pada perangkat bergerak dengan sistem operasi Android 5.1.1 (*Lollipop*). Pengerjaan *web service* dikerjakan menggunakan Netbeans IDE 8.1 sedangkan aplikasi *client* dikerjakan menggunakan Android Studio.

5.2 Batasan Implementasi

Batasan-batasan implementasi sistem adalah sebagai berikut:

1. Aplikasi *client* menggunakan perangkat bergerak dengan sistem operasi Android 5.1.1 (*Lollipop*).
2. Aplikasi *client* diimplementasikan hanya pada satu perangkat.
3. *Web service* diimplementasikan menggunakan bahasa pemrograman Python 2.7, PHP 5.6.28, dan Perl 5.24.
4. *Web service* diimplementasikan pada Apache *web server* dengan versi 2.4.23.
5. *Web service* menggunakan *database interface* (DBI) dari masing-masing bahasa pemrograman (tidak menggunakan *library* tambahan) untuk berkomunikasi dengan *database.*

5.3 Implementasi *Database*

Web service menggunakan *database* sebagai media penyimpanan data yang akan digunakan untuk pengujian performa. Implementasi *database* dapat dilihat pada Gambar 5.1.

```
1 CREATE TABLE `news` (
2   `id` int(11) PRIMARY KEY NOT NULL,
3   `title` text NOT NULL,
4   `content` text NOT NULL,
5   `date` int(11) NOT NULL
6 )
```

Gambar 5.1 Implementasi *Database* Sistem

Tabel 5.1 Penjelasan Implementasi *Database*

No.	Nama Field	Key	Tipe	Keterangan
1.	id	Primary Key	Integer	Digunakan sebagai id berita
2.	title	-	Text	Digunakan sebagai judul berita
3.	content	-	Text	Digunakan sebagai isi konten dari berita
4.	date	-	Integer	Digunakan sebagai tanggal berita diterbitkan

5.4 Implementasi Sistem

Proses implementasi sistem terbagi menjadi empat tahap, yaitu implementasi *web service* Python, implementasi *web service* PHP, implementasi *web service* Perl, dan implementasi aplikasi *mobile client*.

5.4.1 Implementasi *Web Service* Python

```
#!"C:\python27\python.exe"
import MySQLdb
import json
import re

print "Content-type:application/json; charset=utf-8\r\n\r\n"

db = MySQLdb.connect("localhost","root","","webservices")
cursor = db.cursor()

cursor.execute("SELECT * FROM news")
array = list(cursor.fetchall())

for i in xrange(0, len(array)):
    a1 = list(array[i])
    temp = ""

    j = 0
    while (j < 2):
        temp += a1[2]
        temp = re.sub("Lorem ipsum dolor sit amet", "This is real text", temp)
        j += 1

    a1[2] = temp
```

```
25      array[i] = a1
26
27  print json.dumps(array)
28
29  db.close()
```

Gambar 5.2 Implementasi *Web Service* Python

Gambar 5.2 merupakan kode implementasi Algoritme fungsi *web service* menggunakan bahasa pemrograman Python. Penjelasan tiap baris pada kode adalah sebagai berikut:

1. Baris 1 inisialisasi Python CGI (*Common Gateway Interface*).
2. Baris 2-4 *import library* Python yang dibutuhkan.
3. Baris 6 cetak *header* dengan tipe *application/json.*
4. Baris 8 koneksi ke *database* dengan *host*: *localhost, username: root,* dan *database: webservices.*
5. Baris 9 inisialisasi *cursor* pada *database* yang telah terkoneksi.
6. Baris 11 eksekusi *query* "SELECT * FROM news" dan menempatkan hasilnya pada *cursor* yang telah diinisialisasi sebelumnya.
7. Baris 12 mengkonversi data yang ada pada *cursor* ke tipe data *array.*
8. Baris 14 melakukan perulangan dengan *index* i=0 sampai dengan ukuran dari *array.*
9. Baris 15 mengambil data yang ada pada *variable array* pada *index* i dan menempatkannya pada *array* a1.
10. Baris 16 inisialisasi *string* kosong pada *variable* temp.
11. Baris 18 inisialisasi *variable* j=0.
12. Baris 19 melakukan perulangan jika nilai j<2.
13. Baris 20 mengambil data yang ada pada *array* a1 *index* ke 2 dan menambahkannya ke *variable* temp.
14. Baris 21 merubah *text* "Lorem ipsum dolor sit amet" yang ada pada *variable* temp menjadi "This is real text".
15. Baris 22 menambahkan 1 nilai pada *variable* j.
16. Baris 24 mengambil data yang ada pada *variable* temp dan menempatkannya pada *array* a1 *index* ke 2.
17. Baris 25 mengambil data pada *array* a1 dan menempatkannya pada *variable array index* ke i.
18. Baris 27 mengkonversi data pada *variable array* menjadi data JSON dan menampilkannya.
19. Baris 29 menutup koneksi ke *database.*

5.4.2 Implementasi *Web Service* PHP

```
1  #!"C:\xampp\php\php.exe"
2  <?php
3      echo "Content-type:application/json; charset=utf-
   8\r\n\r\n";
4
5      $connect = mysqli_connect('localhost','root','');
6      mysqli_select_db($connect, 'webservices');
7
8      $result = mysqli_query($connect, "SELECT * FROM news");
9      $array = mysqli_fetch_all($result);
10
11     for ($i = 0; $i < sizeof($array); $i++) {
12         $a1 = $array[$i];
13         $temp = "";
14
15         $j = 0;
16         while ($j++ < 2) {
17             $temp .= $a1[2];
               $temp = str_replace("Lorem ipsum dolor sit
   amet", "This is real text", $temp);
18         }
19         $a1[2] = $temp;
20         $array[$i] = $a1;
21     }
22     echo json_encode($array);
23
24     mysqli_close($connect);
25 ?>
26
```

Gambar 5.3 Implementasi *Web Service* PHP

Gambar 5.3 merupakan kode implementasi Algoritme fungsi *web service* menggunakan bahasa pemrograman PHP. Penjelasan tiap baris kode adalah sebagai berikut:

1. Baris 1 inisialisasi PHP *CGI* (*Common Gateway Interface*).
2. Baris 3 cetak *header* dengan tipe *application/json.*
3. Baris 5-6 koneksi ke *database* dengan *host*: *localhost, username: root,* dan *database: webservices.*
4. Baris 8 eksekusi *query* "SELECT * FROM news" dan menempatkan hasilnya pada *variable* result.
5. Baris 9 mengkonversi data hasil *query* ke tipe data *array.*
6. Baris 11 melakukan perulangan dengan *index* i=0 sampai dengan ukuran dari *array.*

7. Baris 12 mengambil data yang ada pada *variable array* pada *index* i dan menempatkannya pada *array* a1.
8. Baris 13 inisialisasi *string* kosong pada *variable* temp.
9. Baris 15 inisialisasi *variable* j=0.
10. Baris 16 melakukan perulangan jika nilai j<2 dan menambahkan nilai j + 1 di setiap perulangan.
11. Baris 17 mengambil data yang ada pada *array* a1 *index* ke 2 dan menambahkannya ke *variable* temp.
12. Baris 18 merubah *text* "Lorem ipsum dolor sit amet" yang ada pada *variable* temp menjadi "This is real text".
13. Baris 20 mengambil data yang ada pada *variable* temp dan menempatkannya pada *array* a1 *index* ke 2.
14. Baris 21 mengambil data pada *array* a1 dan menempatkannya pada *variable array index* ke i.
15. Baris 23 mengkonversi data pada *variable array* menjadi data JSON dan menampilkannya.
16. Baris 25 menutup koneksi ke *database.*

5.4.3 Implementasi *Web Service* Perl

```
1  #!"C:\Strawberry\perl\bin\perl.exe"
2  use DBI;
3  use JSON;
4
5  $|=1;
6  print "Content-type:application/json; charset=utf-
   8\r\n\r\n";
7
8  my $dbh = DBI-
   >connect("DBI:mysql:database=webservices;host=localhost;port
   =3306", "root", "");
9
10 my $result = $dbh->prepare("SELECT * FROM news");
11 $result->execute();
12 my $array = $result->fetchall_arrayref();
13
14 for (my $i = 0; $i < scalar(@{$array}); $i++) {
15     my $a1 = $array->[$i];
16     my $temp = "";
17
18     my $j = 0;
19     while ($j++ < 2) {
20         $temp .= $a1->[2];
```

```
1  #!/usr/bin/perl
2  use CGI;
3  use DBI;
4  print "Content-type: application/json\n\n";
5  my $dbh = DBI->connect('DBI:mysql:webservices;host=localhost','root','');
6  my $sth = $dbh->prepare("SELECT * FROM news");
7  $sth->execute();
8  my $array = $sth->fetchall_arrayref();
9  for my $i (0 .. $#{$array}) {
10 my $a1 = $array->[$i];
11 my $temp = "";
12 for my $j (0 .. $#{$a1}) {
13 if($j<2){
14 $temp = $a1->[$j];
15 $temp =~ s/Lorem ipsum dolor sit amet/This is real
16 text/g;
17 }
18 $a1->[2] = $temp;
19 $array->[$i] = $a1;
20 }
21 print to_json($array);
22
23 $dbh->disconnect();
```

Gambar 5.4 Implementasi *Web Service* Perl

Gambar 5.4 merupakan kode implementasi Algoritme fungsi *web service* menggunakan bahasa pemrograman Perl. Penjelasan tiap baris kode adalah sebagai berikut:

1. Baris 1 inisialisasi Perl CGI (*Common Gateway Interface*).
2. Baris 2-3 *import library* Perl yang dibutuhkan.
3. Baris 6 cetak *header* dengan tipe *application/json.*
4. Baris 8 koneksi ke *database* dengan *host*: *localhost, username: root,* dan *database: webservices.*
5. Baris 9 inisialisasi *cursor* pada *database* yang telah terkoneksi.
6. Baris 10-11 eksekusi *query* "SELECT * FROM news" dan menempatkan hasilnya pada *variable* result.
7. Baris 12 mengkonversi data yang ada pada *cursor* ke tipe data *array.*
8. Baris 14 melakukan perulangan dengan *index* i=0 sampai dengan ukuran dari *array.*
9. Baris 15 mengambil data yang ada pada *variable array* pada *index* i dan menempatkannya pada *array* a1.
10. Baris 16 inisialisasi *string* kosong pada *variable* temp.
11. Baris 18 inisialisasi *variable* j=0.
12. Baris 19 melakukan perulangan jika nilai j<2 dan menambahkan nilai j + 1 di setiap perulangan.
13. Baris 20 mengambil data yang ada pada *array* a1 *index* ke 2 dan menambahkannya ke *variable* temp.
14. Baris 21 merubah *text* "Lorem ipsum dolor sit amet" yang ada pada *variable* temp menjadi "This is real text".
15. Baris 23 mengambil data yang ada pada *variable* temp dan menempatkannya pada *array* a1 *index* ke 2.
16. Baris 24 mengambil data pada *array* a1 dan menempatkannya pada *variable array index* ke i.

17. Baris 26 mengkonversi data pada *variable array* menjadi data JSON dan menampilkannya.
18. Baris 28 menutup koneksi ke *database*.

5.4.4 Implementasi Aplikasi *Mobile Client*

```
1  package net.gumcode.webresponsetimer;
2
3  import android.app.AlertDialog;
4  import android.app.ProgressDialog;
5  import android.os.Bundle;
6  import android.support.v7.app.AppCompatActivity;
7  import android.support.v7.widget.Toolbar;
8  import android.view.View;
9  import android.widget.AdapterView;
10 import android.widget.ArrayAdapter;
11 import android.widget.Button;
12 import android.widget.ListView;
13 import android.widget.TextView;
14
15 import
   net.gumcode.webresponsetimer.adapter.ColumnHeaderAdapter;
16 import
   net.gumcode.webresponsetimer.adapter.TableContentAdapter;
17 import net.gumcode.webresponsetimer.config.Config;
18 import net.gumcode.webresponsetimer.model.Item;
19 import net.gumcode.webresponsetimer.utilities.HTTPHelper;
20
21 import java.io.InputStream;
22 import java.util.ArrayList;
23 import java.util.List;
24
25 import de.codecrafters.tableview.SortableTableView;
26
27
28 public class MainActivity extends AppCompatActivity {
29
30     private Thread t1, t2, t3;
31
32     private SortableTableView tableView;
33     private List<Item> items = new ArrayList<>();
34     private long python, php, perl;
```

```
35      private boolean a, b, c;
        private String[] phps = {Config.PHP_GET_ALL_URL +
"_500.php", Config.PHP_GET_ALL_URL + "_1000.php",
Config.PHP_GET_ALL_URL + "_2000.php", Config.PHP_GET_ALL_URL
+ "_5.000.php", Config.PHP_GET_ALL_URL + "_10.000.php"};
        private String[] pythons = {Config.PYTHON_GET_ALL_URL +
"_500.py", Config.PYTHON_GET_ALL_URL + "_1000.py",
Config.PYTHON_GET_ALL_URL + "_2000.py",
Config.PYTHON_GET_ALL_URL + "_5.000.py",
37 Config.PYTHON_GET_ALL_URL + "_10.000.py"};
        private String[] perls = {Config.PERL_GET_ALL_URL +
"_500.pl", Config.PERL_GET_ALL_URL + "_1000.pl",
Config.PERL_GET_ALL_URL + "_2000.pl",
Config.PERL_GET_ALL_URL + "_5.000.pl",
38 Config.PERL_GET_ALL_URL + "_10.000.pl"};
39      private AlertDialog dialog;
40      private ProgressDialog pd;
41      private int selected = -1;
42
43      @Override
44      protected void onCreate(Bundle savedInstanceState) {
45          super.onCreate(savedInstanceState);
46          setContentView(R.layout.activity_main);
            Toolbar toolbar = (Toolbar)
47  findViewById(R.id.toolbar);
48          setSupportActionBar(toolbar);
49
            tableView = (SortableTableView)
50  findViewById(R.id.table);
            tableView.setHeaderAdapter(new
ColumnHeaderAdapter(this,
51  getResources().getStringArray(R.array.table_header)));
52
            Button start = (Button)
53  findViewById(R.id.btn_start);
            start.setOnClickListener(new View.OnClickListener()
54  {
55              @Override
56              public void onClick(View view) {
57                  showDialog();
58              }
59          });
60      }
61
62      private void showDialog() {
            View v =
63  getLayoutInflater().inflate(R.layout.dialog_header, null);
```

```
        TextView title = (TextView)
64  v.findViewById(R.id.title);
65          title.setText("JUMLAH DATA");
66
            View content =
67  getLayoutInflater().inflate(R.layout.dialog_content, null);
            ListView list = (ListView)
68  content.findViewById(R.id.list);
69          ArrayList<String> items = new ArrayList<>();
70          items.add("500");
71          items.add("1000");
72          items.add("2000");
73          items.add("5.000");
74          items.add("10.000");
            list.setAdapter(new ArrayAdapter<>(this,
75  android.R.layout.simple_list_item_1, items));
            list.setOnItemClickListener(new
76  AdapterView.OnItemClickListener() {
77              @Override
                public void onItemClick(AdapterView<?>
78  adapterView, View view, int i, long l) {
79                  dialog.cancel();
80                  selected = i;
81                  start();
82              }
83          });
84
            AlertDialog.Builder builder = new
85  AlertDialog.Builder(this);
86          builder.setCustomTitle(v);
87          builder.setView(content);
88          dialog = builder.create();
89          dialog.show();
90      }
91
92      private void initTable() {
            tableView.setDataAdapter(new
93  TableContentAdapter(this, items));
94      }
95
96      private void start() {
97          php = 0;
98          python = 0;
99          perl = 0;
100         a = false;
```

```
101             b = false;
102             c = false;
103
104             t1 = new Thread(new Runnable() {
105                 @Override
106                 public void run() {
107                     //request start time
108                     long startTime = System.currentTimeMillis();
109
110                     InputStream response =
HTTPHelper.sendGETRequest(phps[selected]);
111
112                     //request end time
113                     php = System.currentTimeMillis() -
startTime;
114
115                     a = true;
116                 }
117             });
118
119             t2 = new Thread(new Runnable() {
120                 @Override
121                 public void run() {
122
123                     //request start time
124                     long startTime = System.currentTimeMillis();
125
126                     InputStream response =
HTTPHelper.sendGETRequest(perls[selected]);
127
128                     //request end time
129                     perl = System.currentTimeMillis() -
startTime;
130
131                     b = true;
132                 }
133             });
134
135             t3 = new Thread(new Runnable() {
136                 @Override
137                 public void run() {
138                     //request start time
```

```
139                 long startTime = System.currentTimeMillis();
140
141                 InputStream response =
    HTTPHelper.sendGETRequest(pythons[selected]);
142
143                 //request end time
144                 python = System.currentTimeMillis() -
    startTime;
145
146                 c = true;
147             }
148         });
149
150         t1.start();
151         t2.start();
152         t3.start();
153
154         boolean loop = true;
155         while (loop) {
156             if (a & b & c) {
157                 Item item = new Item();
158                 item.python = Long.toString(python);
159                 item.php = Long.toString(php);
160                 item.perl = Long.toString(perl);
161                 items.add(item);
162
163                 initTable();
164
165                 loop = false;
166             }
167         }
168     }
169 }
```

Gambar 5.5 Implementasi Aplikasi *Mobile Client*

Gambar 5.5 merupakan kode implementasi aplikasi *mobile cilent*. Penjelasan tiap baris kode adalah sebagai berikut:

1. Baris 3-25 *import library* yang dibutuhkan.
2. Baris 30-41 inisialisasi *variable* global.
3. Baris 46 *bounding activity* dengan *layout*.
4. Baris 47-48 inisialisasi dan menampilkan *toolbar*.

5. Baris 50 inisialisasi *TableView*.
6. Baris 51 inisialisasi *header TableView* dengan *string* yang ada pada *variable table_header*.
7. Baris 53 inisialisasi *button start*.
8. Baris 54-60 inisialisasi aksi *button start* untuk memanggil *method start()*.
9. Baris 62 inisialisasi *method showDialog()* yang berfungsi untuk menampilkan *dialog* pilihan jumlah *request*.
10. Baris 63 inisialisasi *view dialog header*.
11. Baris 64 inisialisasi *TextView dialog header title*.
12. Baris 65 menampilkan teks "JUMLAH DATA" pada *TextView*.
13. Baris 67 inisialisasi *view content dialog*.
14. Baris 68 inisialisasi *ListView* pilihan jumlah data.
15. Baris 69 inisialisasi *ArrayList items* untuk menampung pilihan *request*.
16. Baris 70-74 menambahkan data pada *ArrayList items*.
17. Baris 75 inisialisasi *Adapter* dengan *item* pada *ArrayList items* dan menampilkannya pada *ListView*.
18. Baris 76-83 memberikan *OnItemClickListener* pada *ListView* yang mana akan mengirimkan *request* sesuai dengan jumlah *request* yang telah dipilih.
19. Baris 85-89 inisialisasi dan menampilkan *AlertDialog* dengan *header* dan *content* yang telah diinisialisasi.
20. Baris 92-94 inisialisasi *method initTable()* yang berfungsi untuk mengisi data ke *TableView*.
21. Baris 96 inisialisasi *method start()*.
22. Baris 97-102 inisialisasi *variable*.
23. Baris 104 inisialisasi *thread* pertama yang digunakan untuk mengirimkan *request* ke *web service* PHP.
24. Baris 108 mencatat waktu sekarang sebagai waktu awal *request*.
25. Baris 110 mengirimkan *request* ke *web service* PHP.
26. Baris 113 mencatat waktu sekarang sebagai waktu akhir *request*.
27. Baris 115 memberi nilai *true* pada *variable* a.
28. Baris 119 inisialisasi *thread* kedua yang digunakan untuk mengirimkan *request* ke *web service* Perl.
29. Baris 124 mencatat waktu sekarang sebagai waktu awal *request*.
30. Baris 126 mengirimkan *request* ke *web service* Perl.
31. Baris 129 mencatat waktu sekarang sebagai waktu akhir *request*.

32. Baris 131 memberi nilai *true* pada *variable* b.
33. Baris 135 inisialisasi *thread* ketiga yang digunakan untuk mengirimkan *request* ke *web service* Python.
34. Baris 139 mencatat waktu sekarang sebagai waktu awal *request.*
35. Baris 141 mengirimkan *request* ke *web service* Python.
36. Baris 144 mencatat waktu sekarang sebagai waktu akhir *request.*
37. Baris 146 memberi nilai *true* pada *variable* c.
38. Baris 150-152 menjalankan semua *thread* secara bersamaan.
39. Baris 154 memberi nilai *true* pada *variable loop.*
40. Baris 155 melakukan perulangan jika nilai *loop = true.*
41. Baris 156 jika nilai *variable* a, b, dan c = *true* maka masuk kondisi *true*, jika kondisi *false* maka akan melanjutkan perulangan.
42. Baris 157-161 mengambil nilai dari *variable* Python, PHP, dan Perl yang merupakan waktu eksekusi *request* dari masing-masing bahasa pemrograman dan memasukkannya kedalam *array list* untuk ditampilkan pada *TableView.*
43. Baris 163 memanggil *method initTable().*
44. Baris 165 memberikan nilai *false* pada *variable loop.*

5.5 Implementasi Antarmuka Sistem

```
1  <? xml version="1.0" encoding="utf-8"?>
   <LinearLayout
2  xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:table="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      android:orientation="vertical"
8  tools:context="net.gumcode.webresponsetimer.MainActivity">
9
10     <android.support.v7.widget.Toolbar
11         android:id="@+id/toolbar"
12         android:layout_width="match_parent"
13         android:layout_height="?attr/actionBarSize"
14         android:theme="@style/Base.Theme.AppCompat"
15         android:background="@color/colorPrimary" />
16
17     <de.codecrafters.tableview.SortableTableView
18         android:id="@+id/table"
19         android:layout_width="match_parent"
```

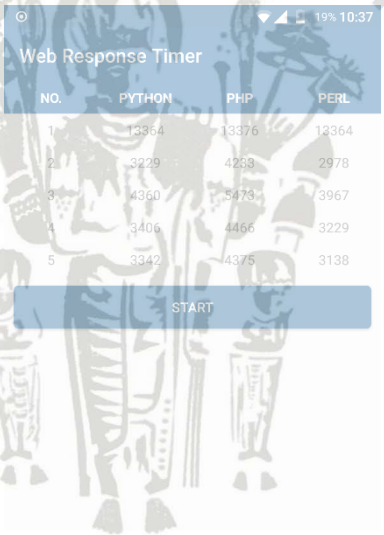
```
20          android:layout_height="wrap_content"
21          table:columnCount="4"
22          table:headerColor="@color/colorPrimary"/>
23
24      <Button
25          android:id="@+id/btn_start"
26          android:layout_width="match_parent"
27          android:layout_height="45dp"
28          android:layout_margin="10dp"
29          android:background="@drawable/rounded_background"
30          android:text="START"
31          android:textColor="@android:color/white" />
32
33  </LinearLayout>
```

Gambar 5.6 Kode Implementasi Antarmuka Sistem

Gambar 5.6 merupakan kode implementasi antarmuka sistem. Penjelasan tiap baris kode adalah sebagai berikut:

1. Baris 1 inisialisasi versi XML.
2. Baris 2-4 inisialisasi *LinearLayout* dan skema XML yang digunakan.
3. Baris 5 mengatur lebar dari *view LinearLayout* menyesuaikan dengan ukuran layar.
4. Baris 6 mengatur tinggi dari *view LinearLayout* menyesuaikan dengan ukuran layar.
5. Baris 7 mengatur orientasi *view LinearLayout* menjadi *vertical.*
6. Baris 8 menggunakan "net.gumcode.webresponsetimer.MainActivity" sebagai *context* pada XML.
7. Baris 10 inisialisasi *toolbar.*
8. Baris 11 memberikan id pada *toolbar* dengan nama "@+id/toolbar".
9. Baris 12 mengatur lebar dari *toolbar* menyesuaikan dengan ukuran layar.
10. Baris 13 mengatur tinggi dari *toolbar* sesuai dengan ukuran *default.*
11. Baris 14 memberikan *style* pada *toolbar.*
12. Baris 15 memberikan *background* pada *toolbar* dengan warna yang ada pada *variable* "@color/colorPrimary".
13. Baris 17 inisialisasi *TableView.*
14. Baris 18 memberikan id pada *TableView* dengan nama "@+id/table".
15. Baris 19 mengatur lebar dari *TableView* menyesuaikan dengan ukuran layar.
16. Baris 20 mengatur tinggi dari *TableView* menyesuaikan dengan isi *content* dari *TableView.*

17. Baris 21 mengatur jumlah kolom pada *TableView* sejumlah 4 kolom.
18. Baris 22 memberikan *background* pada *TableView header* dengan warna yang ada pada *variable* "@color/colorPrimary".
19. Baris 24 inisialisasi *button.*
20. Baris 25 meberikan id pada *button* dengan nama "@+id/btn_start".
21. Baris 26 mengatur lebar dari *button* menyesuaikan dengan ukuran layar.
22. Baris 27 mengatur tinggi dari *button* dengan ukuran 45dp.
23. Baris 28 mengatur *margin* dari *button* sebesar 10dp.
24. Baris 29 memberikan *background* pada *button* dengan warna yang ada pada *variable* "@drawable/rounded_background".
25. Baris 30 memberikan teks *"START"* pada *button.*
26. Baris 31 memberikan warna pada teks dengan warna yang ada pada *variable* "@android:color/white".
27. Baris 33 menutup *LinearLayout.*



Gambar 5.7 Implementasi Antarmuka Sistem

Gambar 5.7 merupakan implementasi antarmuka sistem. Kolom NO. menunjukkan nomor urut *request* yang dikirimkan ke *server*, kolom PYTHON, PHP, dan PERL menunjukkan waktu proses *request* dan *response* dari masing-masing bahasa pemrograman.

BAB 6 PENGUJIAN DAN ANALISIS HASIL

Pada bab ini dibahas mengenai pengujian dan analisis yang telah diimplementasikan sebelumnya. Proses pengujian dilakukan sesuai skenario pengujian pada bab 4.

6.1 Pengujian

Dalam pengujian yang akan dilakukan pada sistem adalah pengujian penggunaan CPU (*Central Processing Unit*), RAM (*Random Access Memory*), dan kecepatan eksekusi perintah dari masing-masing bahasa pemrograman yang digunakan sebagai *web service* untuk aplikasi *client* berbasis Android. Kemudian dilihat hasil analisisnya menggunakan bahasa program manakah yang memiliki perfoma *web service* yang tinggi untuk digunakan pada aplikasi *client* berbasis Android.

6.2 Hasil Pengujian

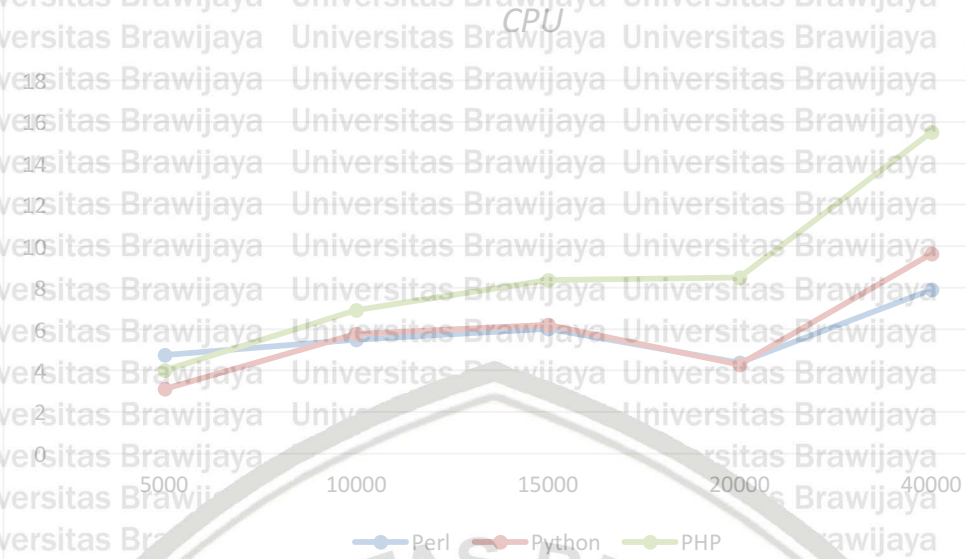
Berdasarkan pada skenario pengujian pada bab 4 pengujian akan dilakukan dengan melihat penggunaan CPU (*Central Processing Unit*), RAM (*Random Access Memory*), dan kecepatan eksekusi perintah masing-masing bahasa pemrograman dengan menggunakan data sebesar 5.000, 10.000, 15.000, 20.000, dan 40.000 data. Dengan harapan untuk mengetahui pengaruh dari setiap bahasa pemrograman dalam penggunaan CPU (*Central Processing Unit*), RAM (*Random Access Memory*), dan kecepatan eksekusi perintah masing masing bahasa pemrograman. Maka hasil pengujian dari sistem yang telah dibuat mengeluarkan hasil seperti berikut.

6.2.1 Pengujian Penggunaan CPU pada *Server*

Pengujian penggunaan CPU pada *server* ini dibagi menjadi beberapa tahap, yaitu pengujian dengan 5.000, 10.000, 15.000, 20.000, dan 40.000 data pada masing-masing bahasa pemrograman. Setiap data yang diuji berukuran 1KB sehingga total data yang diproses adalah 5MB, 10MB, 15MB, 20MB, dan 40MB. Hasil pengujian dapat dilihat pada Tabel 6.1.

Tabel 6.1 Hasil Pengujian Penggunaan CPU pada *Server*

Banyak data	Perl	Python	PHP
5.000	4,76 %	3,11 %	4,01 %
10.000	5,49 %	5,76 %	6,9 %
15.000	6,01 %	6,2 %	8,37 %
20.000	4,37 %	4,3 %	8,48 %
40.000	7,89 %	9,62 %	15,49 %



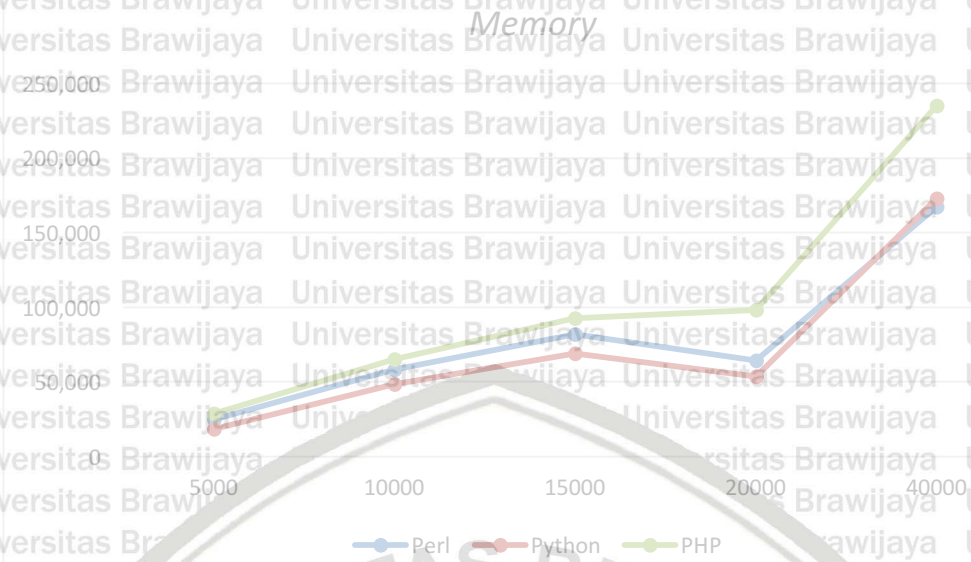
Gambar 6.1 Grafik Penggunaan CPU pada *Server*

6.2.2 Pengujian Penggunaan *Memory* pada *Server*

Pengujian penggunaan *memory* pada *server* ini dibagi menjadi beberapa tahap, yaitu pengujian dengan 5.000, 10.000, 15.000, 20.000, dan 40.000 data pada masing-masing bahasa pemrograman. Setiap data yang diuji berukuran 1KB sehingga total data yang diproses adalah 5MB, 10MB, 15MB, 20MB, dan 40MB. Hasil pengujian dapat dilihat pada Tabel 6.2.

Tabel 6.2 Hasil Pengujian Penggunaan *Memory* pada *Server*

Banyak data	Perl	Python	PHP
5.000	24.880 KB	18.148 KB	28.600 KB
10.000	58.388 KB	48.568 KB	65.264 KB
15.000	82.028 KB	69.156 KB	92.412 KB
20.000	64.092 KB	53.584 KB	98.024 KB
40.000	167.080 KB	172.976 KB	234.824 KB



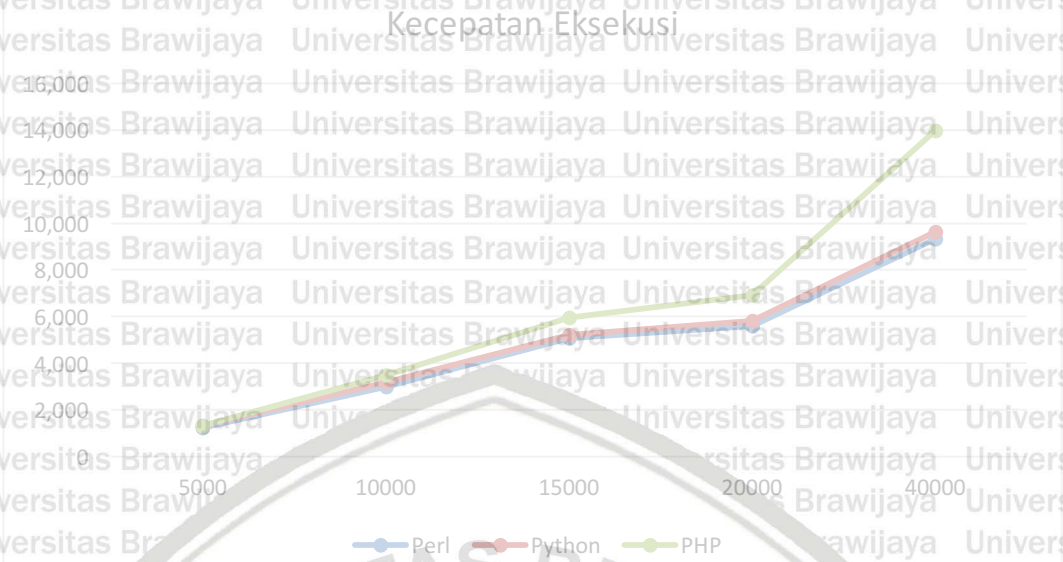
Gambar 6.2 Grafik Penggunaan *Memory* pada *Server*

6.2.3 Pengujian Kecepatan Eksekusi pada Aplikasi *Mobile Client*

Pengujian kecepatan eksekusi pada aplikasi *mobile client* ini dibagi menjadi beberapa tahap, yaitu pengujian dengan 5.000, 10.000, 15.000, 20.000, dan 40.000 data pada masing-masing bahasa pemrograman. Setiap data yang diuji berukuran 1KB sehingga total data yang diproses adalah 5MB, 10MB, 15MB, 20MB, dan 40MB. Hasil pengujian dapat dilihat pada Tabel 6.3.

Tabel 6.3 Hasil Pengujian Kecepatan Eksekusi pada Aplikasi *Mobile Client*

Banyak data	Perl	Python	PHP
5.000	1.228 ms	1.316 ms	1.312 ms
10.000	2.987 ms	3.179 ms	3.479 ms
15.000	5.062 ms	5.210 ms	5.965 ms
20.000	5.585 ms	5.808 ms	6.911 ms
40.000	9.337 ms	9.644 ms	13.966 ms



Gambar 6.3 Grafik Kecepatan Eksekusi pada Aplikasi *Mobile Client*

6.3 Analisis Hasil Pengujian

Berdasarkan pada pengujian yang telah dilakukan dengan melihat penggunaan CPU (*Central Processing Unit*), RAM (*Random Access Memory*), dan kecepatan eksekusi perintah masing-masing bahasa pemrograman dengan menggunakan data sebesar 5.000, 10.000, 15.000, 20.000, dan 40.000 data. Untuk mengetahui bahasa pemrograman mana yang memiliki performa lebih baik untuk digunakan pada aplikasi *client* berbasis Android. Maka perlu dilakukan analisis dari hasil pengujian yang telah dilakukan seperti berikut ini.

6.3.1 Analisis Hasil Pengujian Menggunakan 5.000 Data

Analisis hasil pengujian penggunaan CPU (*Central Processing Unit*), RAM (*Random Access Memory*), dan kecepatan eksekusi perintah pada masing-masing bahasa pemrograman dengan menggunakan 5.000 data dapat dilihat pada Tabel 6.4.

Tabel 6.4 Tabel Analisis Hasil Pengujian Menggunakan 5.000 Data

Parameter	Hasil		
	Perl	Python	PHP
CPU	4,76 %	3,11 %	4,01 %
	-	✓	-
Memory	24.880 KB	18.148 KB	28.600 KB
	-	✓	-
Kecepatan Eksekusi	1.228 ms	1.316 ms	1.312 ms
	✓	-	-

Keterangan: tanda ✓ menunjukkan yang terbaik.

Pada Tabel 6.4 menunjukkan dengan melakukan pengujian menggunakan 5.000 data, bahasa pemrograman Python menggunakan CPU paling sedikit yaitu sebesar 3,11%, disusul dengan PHP sebesar 4,01%, dan Perl sebesar 4,76%. Dari parameter penggunaan *memory* bahasa pemrograman Python menempati urutan pertama dengan penggunaan *memory* sebesar 18.149 KB, disusul dengan Perl menggunakan *memory* sebesar 24.880 KB, dan PHP sebesar 28.600 KB. Selanjutnya dari parameter kecepatan eksekusi bahasa pemrograman Perl menempati urutan pertama dengan kecepatan eksekusi sebesar 1.228 ms, disusul dengan PHP sebesar 1.312 ms, dan Python sebesar 1.316 ms.

6.3.2 Analisis Hasil Pengujian Menggunakan 10.000 Data

Analisis hasil pengujian penggunaan CPU (*Central Processing Unit*), RAM (*Random Access Memory*), dan kecepatan eksekusi perintah pada masing-masing bahasa pemrograman dengan menggunakan 10.000 data dapat dilihat pada Tabel 6.5.

Tabel 6.5 Tabel Analisis Hasil Pengujian Menggunakan 10.000 Data

Parameter	Hasil		
	Perl	Python	PHP
CPU	5,49 %	5,76 %	6,9 %
	✓	-	-
Memory	58.388 KB	48.568 KB	65.264 KB
	-	✓	-
Kecepatan Eksekusi	2.987 ms	3.179 ms	3.479 ms
	✓	-	-

Keterangan: tanda ✓ menunjukkan yang terbaik.

Pada Tabel 6.5 menunjukkan dengan melakukan pengujian menggunakan 10.000 data, bahasa pemrograman Perl menggunakan CPU paling sedikit yaitu sebesar 5,49%, disusul dengan Python sebesar 5,76%, dan PHP sebesar 6,9%. Dari parameter penggunaan *memory* bahasa pemrograman Python menempati urutan pertama dengan penggunaan *memory* sebesar 48.567 KB, disusul dengan Perl menggunakan *memory* sebesar 56.388 KB, dan PHP sebesar 65.264 KB. Selanjutnya dari parameter kecepatan eksekusi bahasa pemrograman Perl menempati urutan pertama dengan kecepatan eksekusi sebesar 2.987 ms, disusul dengan Python sebesar 3.179 ms, dan PHP sebesar 3.479 ms.

6.3.3 Analisis Hasil Pengujian Menggunakan 15.000 Data

Analisis hasil pengujian penggunaan CPU (*Central Processing Unit*), RAM (*Random Access Memory*), dan kecepatan eksekusi perintah pada masing-masing bahasa pemrograman dengan menggunakan 15.000 data dapat dilihat pada Tabel 6.6.

Tabel 6.6 Tabel Analisis Hasil Pengujian Menggunakan 15.000 Data

Parameter	Hasil		
	Perl	Python	PHP
CPU	6,01 %	6,2 %	8,37 %
	✓	-	-
Memory	82.028 KB	69.156 KB	92.412 KB
	-	✓	-
Kecepatan Eksekusi	5.062 ms	5.210 ms	5.965 ms
	✓	-	-

Keterangan: tanda ✓ menunjukkan yang terbaik.

Pada Tabel 6.6 menunjukkan dengan melakukan pengujian menggunakan 15.000 data, bahasa pemrograman Perl menggunakan CPU paling sedikit yaitu sebesar 6,01%, disusul dengan Python sebesar 6,2%, dan PHP sebesar 8,37%. Dari parameter penggunaan *memory* bahasa pemrograman Python menempati urutan pertama dengan penggunaan *memory* sebesar 69.156 KB, disusul dengan Perl menggunakan *memory* sebesar 82.028 KB, dan PHP sebesar 92.412 KB. Selanjutnya dari parameter kecepatan eksekusi bahasa pemrograman Perl menempati urutan pertama dengan kecepatan eksekusi sebesar 5.062 ms, disusul dengan Python sebesar 5.210 ms, dan PHP sebesar 5.965 ms.

6.3.4 Analisis Hasil Pengujian Menggunakan 20.000 Data

Analisis hasil pengujian penggunaan CPU (*Central Processing Unit*), RAM (*Random Access Memory*), dan kecepatan eksekusi perintah pada masing-masing bahasa pemrograman dengan menggunakan 20.000 data dapat dilihat pada Tabel 6.7.

Tabel 6.7 Tabel Analisis Hasil Pengujian Menggunakan 20.000 Data

Parameter	Hasil		
	Perl	Python	PHP
CPU	4,37 %	4,3 %	8,48 %
	-	✓	-
Memory	64.092 KB	53.584 KB	98.024 KB
	-	✓	-
Kecepatan Eksekusi	5.585 ms	5.808 ms	6.911 ms
	✓	-	-

Keterangan: tanda ✓ menunjukkan yang terbaik.

Pada Tabel 6.7 menunjukkan dengan melakukan pengujian menggunakan 20.000 data, bahasa pemrograman Python menggunakan CPU paling sedikit yaitu sebesar 4,3%, disusul dengan Perl sebesar 4,37%, dan PHP sebesar 8,48%. Dari parameter penggunaan *memory* bahasa pemrograman Python menempati urutan pertama dengan penggunaan *memory* sebesar 53.584 KB, disusul dengan Perl

menggunakan *memory* sebesar 64.092 KB, dan PHP sebesar 97.024 KB. Selanjutnya dari parameter kecepatan eksekusi bahasa pemrograman Perl menempati urutan pertama dengan kecepatan eksekusi sebesar 5.585 ms, disusul dengan Python sebesar 5.808 ms, dan PHP sebesar 6.911 ms.

6.3.5 Analisis Hasil Pengujian Menggunakan 40.000 Data

Analisis hasil pengujian penggunaan CPU (*Central Processing Unit*), RAM (*Random Access Memory*), dan kecepatan eksekusi perintah pada masing-masing bahasa pemrograman dengan menggunakan 40.000 data dapat dilihat pada Tabel 6.8.

Tabel 6.8 Tabel Analisis Hasil Pengujian Menggunakan 40.000 Data

Parameter	Hasil		
	Perl	Python	PHP
CPU	7,89 %	9,62 %	15,49 %
	✓	-	-
Memory	167.080 KB	172.976 KB	234.824 KB
	✓	-	-
Kecepatan Eksekusi	9.337 ms	9.644 ms	13.966 ms
	✓	-	-

Keterangan: tanda ✓ menunjukkan yang terbaik.

Pada Tabel 6.8 menunjukkan dengan melakukan pengujian menggunakan 40.000 data, bahasa pemrograman Perl menggunakan CPU paling sedikit yaitu sebesar 7,89%, disusul dengan Python sebesar 9,62%, dan PHP sebesar 15,49%. Dari parameter penggunaan *memory* bahasa pemrograman Perl menempati urutan pertama dengan penggunaan *memory* sebesar 167.080 KB, disusul dengan Python menggunakan *memory* sebesar 172.976 KB, dan PHP sebesar 234.824 KB. Selanjutnya dari parameter kecepatan eksekusi bahasa pemrograman Perl menempati urutan pertama dengan kecepatan eksekusi sebesar 9.337 ms, disusul dengan Python sebesar 9.644 ms, dan PHP sebesar 13.966 ms.

BAB 7 PENUTUP

7.1 Kesimpulan

Berdasarkan pada pengujian dan analisis hasil dari penelitian ini, maka dapat disimpulkan bahwa:

1. Analisis dilakukan dengan cara menjalankan operasi *string concatenation* pada bahasa pemrograman Python, PHP, dan Perl. Operasi tersebut diimplementasikan pada *web service* di jaringan lokal dan diakses menggunakan aplikasi *client* berbasis Android. Dari operasi tersebut didapatkan hasil penggunaan CPU dan *memory* pada sisi *server*, serta kecepatan eksekusi perintah pada sisi *client*.
2. Pada pengujian dengan menggunakan 5.000 dan 20.000 data, bahasa pemrograman Perl memiliki kecepatan eksekusi paling cepat, sedangkan bahasa pemrograman Python memiliki penggunaan CPU dan *memory* paling sedikit. Pada pengujian dengan menggunakan 10.000 dan 15.000 data, bahasa pemrograman Perl memiliki kecepatan eksekusi paling cepat dan penggunaan CPU paling sedikit, sedangkan bahasa pemrograman Python memiliki penggunaan *memory* paling sedikit. Pada pengujian dengan menggunakan 40.000 data, bahasa pemrograman Perl memiliki kecepatan eksekusi paling cepat dan penggunaan CPU serta *memory* paling sedikit.
3. Bahasa pemrograman Perl mempunyai rata-rata kecepatan eksekusi paling cepat, sedangkan bahasa pemrograman Python mempunyai rata-rata penggunaan *memory* dan CPU paling sedikit dibandingkan dengan bahasa pemrograman lainnya.

7.2 Saran

Berikut adalah saran yang dapat digunakan untuk penelitian selanjutnya:

1. Melakukan penelitian dengan menggunakan bahasa pemrograman lain seperti Java, C++, dsb, sehingga dapat mengetahui peformansi dari bahasa pemrograman tersebut.
2. Melakukan penelitian pada *platform* selain Android seperti IOS, sehingga dapat diketahui apakah hasil yang didapatkan akan berbeda atau sama.
3. Melakukan penelitian menggunakan sistem *database* selain MySQL seperti DB2, sehingga dapat diketahui apakah hasil yang didapatkan akan berbeda atau sama.
4. Menggunakan sistem *interface* selain CGI (*Common Gateway Interface*) seperti FCGI (*Fast Common Gateway Interface*), sehingga dapat diketahui apakah hasil yang didapatkan akan berbeda atau sama.
5. Melakukan penelitian dengan menggunakan data kurang dari 5.000 atau lebih dari 40.000 data, sehingga dapat diketahui apakah hasil yang didapatkan akan berbeda atau sama.

DAFTAR PUSTAKA

Hamad, H., Saad, M. & Abed, R., 2010. Performance Evaluation of RESTful Web Services for Mobile Devices. *International Arab Journal of e-Technology,* 1(3), pp. 72-78.

Johal, A. S. & Singh, B., 2014. Performance Analysis of Web Services for Android based Devices. *International Journal of Computer Applications,* 92(11), pp. 43-45.

Kathuria, K., Kapoor, C. & Adlakha, A., 2014. Common Gateway Interface. *International Journal of Science and Research (IJSR),* 3(10), pp. 1733-1735.

Kreger, H., 2001. Web Services Conceptual Architecture.

Lutz, M., 2010. *Programming Python.* Fourth Edition ed. Sebastopo: O'Reilly Media, Inc.

Oguntunde, B. O., 2012. Comparative Analisys of Some Programming Languages. *Transnational Journal of Science and Technology,* Volume 2, pp. 107-118.

Onlyjob, 2012. *Perl, Python, Ruby, PHP, C, C++, Lua, tcl, javascript and Java comparison.*[Online] Tersedia di: <https://raid6.com.au/~onlyjob/posts/arena/> [Diakses 14 Juni 2017].

OSI, 2016. *(OSI) Open Source Initiative.* [Online] Tersedia di: <https://opensource.org/> [Diakses 24 Februari 2016].

Sagayaraj, S. & Kumar, M. S., 2013. Performance Evaluation of Web Services in C#, JAVA, and PHP. *International Journal of Computer Science and Business Informatics,* 7(1).

Savoia, A., 2001. *Web Page Response Time 101.* s.l.:STQE.

Siever, E., Spainhour, S. & Patwardhan, N., 1998. *Perl in a Nutshell.* First Edition ed. Sebastopol: O'Reilly & Associates, Inc..

Swaroopch, 2016. *Features of Python.* [Online] Tersedia di: <https://www.ibiblio.org/swaroopch/byteofpython/read/features-of-python.html> [Diakses 24 Februari 2016].

Tatroe, K., MacIntyre, P. & Lerdorf, R., 2013. *Programming PHP.* Third Edition ed. Sebastopol: O'Reilly Media, Inc.

Tutorialspoint.com (a), 2016. *Tutorial Point.* [Online] Tersedia di: <http://www.tutorialspoint.com/perl/perl_introduction.htm> [Diakses 25 Februari 2016].

Tutorialspoint.com (b), 2016. *Tutorials Point.* [Online] Tersedia di: <http://www.tutorialspoint.com/python/python_overview.htm> [Diakses 24 Februari 2016].

W3C, 2004. Web Services Architecture. *W3C Working Group Note.*

W3techs.com, 2016. *W3Techs*. [Online] Tersedia di: <http://w3techs.com/technologies/details/pl-php/5/all> [Diakses 25 Februari 2016].