Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN: 2548-964X
Vol. 2, No. 5, Mei 2018, hlm. 2025-2033 http://j-ptiik.ub.ac.id

Analisis Performansi *HTTP Networking Library* pada Android (Studi Kasus: Portal Berita)

Mochammad Syaifullah Ferryansyah[1], Mahardeka Tri Ananta[2], Lutfi Fanani[3]

Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya
Email: [1]syaifullah.ferryansyah@gmail.com, [2]deka@ub.ac.id, [3]lutfifanani@ub.ac.id

Abstrak

Berkembangnya teknologi yang semakin canggih, membuat penyebaran berita semakin cepat. Awalnya berita disajikan dalam bentuk konvensional seperti koran. Perlahan berita tersebut disajikan dalam bentuk aplikasi *mobile*. Dalam implementasinya, para pengembang aplikasi menggunakan berbagai *library* untuk membuat suatu portal berita. *Library* yang sering digunakan diantaranya *HttpURLConnection*, *Asynchronous Http Client*, *Retrofit*, dan *OkHttp*. Namun performansi dari setiap *library* tersebut belum diketahui. Oleh karena itu dibutuhkan suatu penelitian untuk memberikan informasi performansi dari setiap *library*. Pada penelitian ini, *metrics* yang digunakan adalah *response time*, *memory usage*, dan *network usage* sebagai pembanding antar *library*. Pengujian terdiri dari perbandingan antara data teks dan data teks beserta gambar. Hasil pengujian penelitian ini menunjukkan bahwa *response time* tercepat pada *library OkHttp* dengan rata-rata sebesar 148.675 ms. *Memory usage* terkecil pada *library HttpURLConnection* dengan rata-rata sebesar 13906.1 KB. Sedangkan *network usage* terkecil pada *library Asynchronous Http Client* dengan rata-rata sebesar 67196.5 B. Hasil pengujian pada penelitian ini juga menunjukkan terjadinya peningkatan *response time* setiap *library* saat data yang diakses adalah data teks dan gambar. Serta terjadi peningkatan *memory usage* dan *network usage* saat data yang diakses adalah data teks dan gambar.

Kata kunci: *android library, smartphone, komparasi, response time, memory usage, network usage*

Abstract

The development of technology that more sophisticated, making the news spread faster. Initially news is presented in conventional forms such as newspapers. Slowly the news is presented in the form of mobile apps. In its implementation, application developers use various libraries to create a news portal. The frequently used libraries include HttpURLConnection, Asynchronous Http Client, Retrofit, and OkHttp. But the performance of each library is unknown. Therefore, needs a research to provide performance information from each library. In this research, metrics are response time, memory usage, and network usage as a comparison between libraries. Testing consists of a comparison between text data and text data along with images. The results of this study showed that the fastest response time in OkHttp library with an average of 148,675 ms. The smallest usage memory in the HttpURLConnection library with an average of 13906.1 KB. While the smallest network usage on Asynchronous Http Client library with an average of 67196.5 B. The test results in this study also showed an increase in response time of each library when the accessed data is text and image data. And there is an increase in memory usage and network usage memory when the accessed data is text and image data.

Keywords: *android library, smartphone, comparison, response time, memory usage, network usage*

1. PENDAHULUAN

Seiring perkembangan teknologi yang semakin pesat membuat penyebaran berita semakin cepat. Menurut Dr. Willard G. Bleyer, berita adalah segala sesuatu yang hangat dan menarik perhatian sejumlah pembaca. Berita yang terbaik ialah berita yang paling menarik perhatian pembaca dalam jumlah yang sangat besar (Suhandang, 2014). Berita merupakan media informasi yang mengalami perubahan bentuk dari masa ke masa. Para informan menganggap media konvensional sudah tidak praktis dan ekonomis, sehingga perlahan beralih ke media *online* (Irsya, et al., 2013).

Untuk mempermudah dalam pengaksesannya, portal berita tersebut dibuat dalam bentuk aplikasi perangkat bergerak. Adapun beberapa kemudahan mengakses berita melalui perangkat bergerak ialah dapat diakses dimana saja dan kapan saja. Selain itu konten pada aplikasi tidak terbatas hanya pada teks maupun gambar melainkan juga berupa video. Dalam pembuatan portal berita berbasis aplikasi perangkat bergerak, terdapat beberapa *library* yang dapat digunakan. Tujuan penggunaan *library* ialah untuk mempermudah proses pembuatan aplikasi.

Library merupakan kumpulan *resource non-volatile* yang digunakan oleh program komputer dan sering digunakan untuk mengembangkan *software*. *Library* tersebut dapat berisi konfigurasi data, dokumentasi, data penunjang, maupun *class* (Chen, 2015). Salah satu jenis *library* ialah *library http* yaitu *library* yang digunakan untuk koneksi klien dan *server*. Dengan adanya *library* tersebut maka proses koneksi antar klien dan *server* dapat diimplementasikan dengan lebih mudah. *Library HTTP* yang banyak digunakan yaitu *OkHttp* dengan persentase 6,30% dari keseluruhan aplikasi yang ada pada *Google Playstore*. Kemudian *Retrofit* sebanyak 2,56%, dan *Asynchronous Http Client* sebanyak 2,43% (AppBrain, 2016). Dengan adanya *library* tersebut, *developer* tidak perlu membuat ulang suatu modul untuk koneksi klien dan *server*. *Developer* cukup memanggil fungsi yang terdapat pada *library* tersebut. Di samping kemudahan tersebut, terdapat permasalah yaitu kualitas layanan atau *Quality of Sevices (QoS)*.

Salah satu dari *metrics* kualitas layanan yaitu *perfomance*. Parameter pada *performance* berupa *response time*. Semakin cepat *response time* maka dapat semakin baik *performance* yang didapat (Ladan, 2011). Sehingga *user* tidak perlu menunggu proses terlalu lama hanya untuk melihat suatu berita. Namun *response time* belum cukup untuk dijadikan sebagai standar kehandalan suatu *library*. Adapun parameter lain *performance* yaitu *resource utilization* yang dijelaskan pada ISO/IEC 25010 (ISO, 2016). Beberapa bagian dari *resource utilization* yaitu *memory usage* dan *network usage*. Diperlukan pengujian pada *memory usage* karena karakteristik pada perangkat android ialah tidak memiliki *swap partition*. Sehingga jika sistem kehabisan *memory* maka *kernel* akan menghentikan suatu *process* untuk mendapatkan *memory*. Hal ini dapat mengganggu jika *user* ingin menggunakan *process* tersebut dalam waktu dekat. *User* dapat membuka ulang aplikasi namun hal tersebut dapat menyebabkan baterai terkuras (Wei, 2014). Meskipun perkembangan *random access memory* (RAM) pada perangkat meningkat, namun permintaan pasar negara berkembang berada pada *mid-to-low end smartphone* (Egham, 2016). Untuk membandingkan *memory usage*, peneliti menggunakan *private dirty RAM*. *Private dirty RAM* ialah *memory* yang hanya digunakan oleh suatu *process* tanpa dipengaruhi oleh *process* lainnya. Banyaknya *memory* itulah yang dapat diambil oleh sistem jika proses tersebut telah dihentikan (Developers, 2016).

Sedangkan *network usage* ialah banyaknya data yang dikirim dan diterima dari suatu *interface* jaringan. Untuk mendapatkan nilai *network usage*, peneliti menggunakan *TrafficStats*. *TrafficStats* ialah suatu *class* pada android yang menyediakan statistik trafik jaringan. Statistik tersebut termasuk *bytes* maupun paket jaringan yang dikirim dan diterima pada seluruh *interface*, *interface* perangkat, maupun berbasis per-*UID* (Developers, 2016). Diperlukan pengujian pada *network usage* untuk mengetahui *library* yang memiliki penggunaan data terkecil. Sehingga pemakaian kuota internet lebih efektif.

Untuk menghasilkan berita yang menarik minat pembaca bukan hanya dari tulisan yang baik dan mudah dipahami melainkan melalui gambar. Pembaca lebih suka melihat gambar daripada membaca tulisan berita yang panjang. Sehingga diperlukan juga pengujian untuk mengetahui pengaruh data berupa gambar terhadap *library* yang diuji.

Tujuan yang diharapkan dari penelitian ini untuk memberikan informasi bagi *developer* aplikasi dalam menentukan library http yang sesuai dengan kebutuhan dan menjadi acuan dalam pengembangan aplikasi khususnya pada *platform* android.

2. LANDASAN KEPUSTAKAAN

2.1 HTTP

Protokol *HTTP* merupakan protokol jaringan pada level aplikasi yang ringan dan cepat dibutuhkan untuk sistem informasi terdistribusi, kolaboratif, dan hipermedia. *HTTP* telah digunakan di jaringan seluruh dunia sejak 1990 (Berners-Lee, et al., 1996). Pada penelitian ini *protocol HTTP* digunakan untuk aplikasi

mobile dan *web administrator* sebagai protokol yang dapat mendistribusikan data yang bersumber dari *database server*.

2.2 Library

Library merupakan kumpulan *resource non-volatile* yang digunakan oleh program komputer dan sering digunakan untuk mengembangkan *software*. *Library* tersebut dapat berisi konfigurasi data, dokumentasi, data penunjang, maupun *class* (Chen, 2015). Salah satu jenis *library* ialah *library* http yaitu *library* yang digunakan untuk koneksi client dan server. Beberapa contoh *library* http ialah :

2.2.1 HttpURLConnection

HttpURLConnection merupakan suatu *class* turunan dari *URLConnection* yang mendukung fitur *HTTP*. Penggunaan dari *class* ini mengikuti pola sebagai berikut (Developers, 2016):

1. Membuat *HttpURLConnection* baru dengan cara memanggil *method URL.openConnection()* dan *casting* hasilnya ke *HttpURLConnection.*
2. Mempersiapkan *request.* Bagian utama dari setiap *request* ialah URI (*Uniform Resource Identifier*). *Request header* juga termasuk *metadata* seperti kredensial, konten yang diinginkan, dan *session cookies.*
3. Secara opsional mengunggah *request body*. Instansiasi harus dikonfigurasi dengan *setDoOutput(true)* jika termasuk *request body*. Mentransmisikan data dengan menulis melalui *stream* yang dikembalikan oleh *getOutputStream().*
4. Membaca *response*. *Response header* biasanya berisi *metadata* seperti tipe konten, dan panjang *response body*, tanggal dimodifikasi dan *session cookies*. *Response body* dapat dibaca dari *stream* yang dikembalikan oleh *getInputStream()*.Jika suatu *response* tidak memiliki *body* maka *method* akan mengembalikan *stream* kosong.
5. *Disconnect.* Setelah *response body* telah dibaca, *HttpURLConnection* harus ditutup dengan memanggil *method disconnect().* Sehingga *resource* yang dimiliki suatu koneksi dapat ditutup atau digunakan kembali.

2.2.2 Asynchronous Http Client

Suatu *http client* berbasis *asynchronous callback* yang dibangun dari *library Apache HttpClient*. Beberapa fitur dari *library* tersebut adalah (Smith, 2013):

1. Dukungan *API* 23 ke atas.
2. *Request HTTP* secara *asynchronous*.
3. *Request HTTP* terjadi di luar *thread UI*.
4. Integrasi dengan *Jackson JSON*, *Gson*, atau *library* deserialisasi *JSON* lainnya menggunakan *BaseJsonHttpResponseHandler*.
5. Mendukung *encoding* bahasa dan konten, bukan hanya *UTF-8.*

2.2.3 Retrofit

Retrofit merupakan *REST client library* yang aman untuk *android* dan *java*. Dibuat oleh Square. *Retrofit* menyediakan cara yang aman untuk autentikasi dan interaksi dengan berbagai API lainnya. Sehingga memungkinkan pengiriman permintaan jaringan dengan *OkHttp*. *Retrofit* mengambil data *JSON* atau *XML* dari *web API* dan saat data diterima akan langsung diubah ke *Plain Old Java Object (POJO).* Sehingga harus ditentukan setiap *class* yang akan dipakai saat *response* diterima (Codepath, 2015).

Retrofit juga bekerja dengan *REST API* menggunakan implementasi *java interface*, yang dapat dihasilkan dengan bantuan *RestAdapter*. Implementasi dalam hal ini bertindak sebagai *local instance* dari layanan dan setiap panggilan sesuai dengan permintaan *HTTP* (Maskov, 2015).

2.2.4 OkHttp

OkHttp merupakan suatu *HTTP client* untuk android dan aplikasi *java* buatan Square. *OkHttp* siap ketika jaringan terdapat gangguan. Secara perlahan *OkHttp* akan pulih dari masalah koneksi. Jika suatu layanan terdapat beberapa *IP address*, *OkHttp* akan memilih alamat alternatif jika koneksi pertama gagal. Ini dibutuhkan untuk *IPv4 + IPv6* dan untuk layanan dengan *data center* yang besar. *OkHttp* memulai dengan fitur *TLS* yang baru (*SNI*, *ALPN*) dan jika gagal akan kembali ke *TLS* 1.0. *Request/response API* telah dirancang dengan kuat sekaligus mendukung *synchronous blocking call* dan *async call with callback* (OkHttp, 2016).

2.3 JSON

JSON (JavaScript Object Notation) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer. Format ini dibuat berdasarkan bagian dari Bahasa Pemrograman *JavaScript*, Standar

ECMA-262 Edisi ke-3 Desember 1999 (JSON, 2016).

2.4 Web Service

Web service menyediakan standar operasi antara aplikasi perangkat lunak yang berbeda dan berjalan pada *platform* atau *framework* yang berbeda (Lee, 2014). *Web service* adalah suatu standar yang mengintregasikan aplikasi berbasis *web* dengan menghubungkan dan berbagi proses bisnis di seluruh jaringan yang memiliki aplikasi berbeda vendor, bahasa, dan *platform* (Al-Fedaghi, 2011) . Pada awalnya implementasi *web service* menggunakan *SOAP (Simple Object Access Protocol)* dan *WSDL (Web Services Description Language). SOAP* adalah suatu standar protokol yang saling beroperasi untuk menghubungkan aplikasi. Sedangkan *WSDL* adalah suatu standar *interface* yang digunakan untuk mengakses aplikasi. Dalam beberapa tahun terakhir, terdapat suatu pendekatan alternatif menggunakan *REST (REpresentational State Transfer). REST* telah banyak digunakan dalam aplikasi berskala internet (Ramanathan, 2014).

3. METODOLOGI

Metodologi menjelaskan tentang tahapan yang akan dilakukan dalam penelitian Analisis Performansi *HTTP Networking Library* pada Android. Diagram alir pengerjaan penelitian ini ditunjukkan dalam pada Gambar 1.

3.1 Identifikasi Masalah

Identifikasi masalah merupakan suatu tahapan yang dilakukan peneliti untuk mendapatkan permasalahan yang terjadi di masyarakat. Dari permasalahan tersebut akan dirumuskan menjadi permasalahan yang lebih khusus. Sehingga peneliti dapat mempelajari dengan baik permasalahan yang terjadi dan dapat menemukan solusi yang terbaik untuk menyelesaikannya.

3.2 Analisis Kebutuhan

Analisis kebutuhan bertujuan untuk mengidentifikasi kebutuhan-kebutuhan yang terdapat pada sistem. Kebutuhan tersebut meliputi kebutuhan fungsional dan kebutuhan non fungsional.

Gambar 1. Tahapan Metodologi Penelitian

3.3 Perancangan Basis Data dan Web Service

Perancangan basis data bertujuan untuk merancang tabel yang akan digunakan dalam pembuatan aplikasi. Hasil pada perancangan basis data akan diimplementasikan menggunakan aplikasi *phpMyAdmin* karena basis data yang digunakan pada penelitian ini adalah *MySQL*.

Setelah melakukan perancangan basis data maka dilakukan perancangan *web service*. Dilakukan perancangan *web service* agar *client* dan *server* dapat berkomunikasi melalui suatu *API (Application Program Interface).* Perancangan dimulai dari konfigurasi *username, password,* dan basis data yang digunakan. Setelah itu membuat suatu *controller* yang akan diakses oleh klien dan mengirimkan *response* balik.

3.4 Perancangan Penggunaan Library

Perancangan penggunaan *library* bertujuan untuk merancang urutan tahapan yang digunakan pada implementasi sistem. Pada tahap ini terdapat tiga tahapan utama yaitu *input*, proses,dan *output*.

Pada tahapan *input*, *input* berupa suatu alamat *IP* yang telah didefinisikan sebelumnya. Kemudian sistem akan melakukan *request* data ke *server*. Sistem akan memanggil fungsi dari

library untuk membuat koneksi dengan *server* melalui *protocol http*. *Server* akan akan memberikan *response* berupa data berformat *JSON*.

Pada tahapan proses, data yang diterima tadi akan di-*parsing* untuk dimasukkan ke dalam *layout* pada aplikasi. Sedangkan data yang berupa gambar, sistem akan melakukan *request* menggunakan *URL* dari gambar dan mengunduhnya menggunakan metode yang terdapat pada setiap *library*.

Pada tahapan *output*, sistem akan menampilkan portal berita beserta *response time, memory usage,* dan *network usage*.

3.5 Perancangan Antarmuka

Perancangan antarmuka bertujuan untuk merancang antarmuka aplikasi yang menarik dan mudah digunakan. Untuk membantu dalam penelitian ini, maka dibutuhkan antarmuka aplikasi yang dapat menampilkan *response time, memory usage,* dan *network.*

3.6 Implementasi

Berdasarkan perancangan yang telah dijelaskan sebelumnya maka implementasi dilakukan dengan membuat purwarupa berupa aplikasi *native* android. Pendekatan aplikasi *native* dilakukan untuk mendapatkan kinerja aplikasi yang optimal.

Library yang akan diimplementasikan pada sistem yaitu *library asynchronous http client, retrofit, okhttp,* dan *HttpURLConnection*. Untuk menunjang penelitian ini, peneliti menggunakan *PC* berjenis *notebook* dan *IDE Android Studio*. Serta *smartphone* dengan sistem operasi android untuk menjalankan aplikasinya.

3.7 Pengujian dan Analisis Library

Proses pengujian dilakukan dengan pengambilan data pada *server*. Data tersebut memiliki bentuk *JSON*. Segala data berada pada *server localhost* agar kondisi jaringan tidak terganggu jaringan luar dan kondisi dapat terukur.

Pengujian dilakukan dengan cara membuka aplikasi dan memilih *library* yang ingin diuji kemudian memilih tipe data teks ataupun data teks dan gambar. *Response time, memory usage,* dan *network* usage akan ditampilkan untuk kepentingan proses analisis. Pengujian dilakukan sebanyak 20 kali untuk mendapatkan variasi nilai. Data akan ditampilkan dalam bentuk tabel maupun grafik.

3.8 Pengambilan Kesimpulan

Pengambilan kesimpulan dilakukan setelah semua tahapan selesai dilakukan. Mulai dari tahapan identifikasi masalah, perancangan, implementasi, pengujian, hingga analisis. Kesimpulan dibuat untuk memberikan hasil perbandingan performansi dari setiap *library*. Selain kesimpulan, saran juga dibuat untuk memberikan pertimbangan maupun acuan terhadap penelitian selanjutnya khususnya pada *platform* android.

4. REKAYASA PERSYARATAN/KEBUTUHAN

4.1 Kebutuhan Fungsional

Kebutuhan fungsional berisi tentang proses yang dilakukan oleh sistem dan menjelaskan informasi yang dihasilkan oleh sistem. Kebutuhan fungsional pada sistem ini terdiri dari:

1. Sistem dapat menampilkan berita sesuai dengan tipe data yang dipilih pengguna, yaitu berupa data teks maupun data teks dan gambar.
2. Sistem dapat menampilkan nilai *response time, memory usage,* dan *network usage*.

4.2 Kebutuhan Non Fungsional

1. *Response time* : Aplikasi mampu dibuka dalam waktu <10 detik.
2. *User-friendly* : Tampilan aplikasi dibuat agar mudah dioperasikan oleh pengguna.
3. *Memory* : Aplikasi dibuat dengan kapasitas <20MB.
4. *Supportability* : Aplikasi dapat dijalankan pada sistem android minimal android JB 4.3.

5. PERANCANGAN DAN IMPLEMENTASI

5.1 Perancangan Sistem

Perancangan pada bab ini dibagi dalam 8 bagian yaitu perancangan basis data, perancangan *web service,* perancangan *HttpURLConnection,* perancangan *library Asynchronous Http Client*, perancangan *library Retrofit*, perancangan *library OkHttp*, perancangan antar muka, dan perancangan skenario pengujian.

5.2 Implementasi

1. Implementasi *Response Time.*

Implementasi kode *response time* pada *HttpURLConnection, Asynchronous Http Client, Retrofit,* dan *OkHttp* ditunjukkan pada Tabel 1.

Tabel 1. Implementasi Kode *Response Time*

```
startTime =
   System.currentTimeMillis();
long
   elapsedTime=System.currentTimeM
   illis()-startTime;
```

2. Implementasi Memory Usage.

Implementasi kode *memory usage* pada *HttpURLConnection, Asynchronous Http Client, Retrofit,* dan *OkHttp* ditunjukkan pada Tabel 2.

Tabel 2. Implementasi Kode *Memory Usage*

```
MI = new Debug.MemoryInfo();
android.os.Debug.getMemoryInfo(MI)
   ;
totalPrivateDirty =
   MI.getTotalPrivateDirty();
```

3. Implementasi Network Usage.

Implementasi kode *network usage pada HttpURLConnection, Asynchronous Http Client, Retrofit,* dan *OkHttp* ditunjukkan pada Tabel 3.

Tabel 3. Implementasi Kode *Network Usage Usage*

```
 public void getNetworkUsage(){

if (txBytes ==
   TrafficStats.UNSUPPORTED ||
   rxBytes ==
   TrafficStats.UNSUPPORTED) {

   Toast.makeText(MainActivityRe
   trofit.this, "not supported
   monitoring",
   Toast.LENGTH_LONG).show();
}
else {
 int UID =
   android.os.Process.myUid();
 resultTx =
   TrafficStats.getUidTxBytes(UI
   D) - txBytes;
 resultRx =
   TrafficStats.getUidRxBytes(UI
   D) - rxBytes;
 txText.setText(resultTx +"B");
 rxText.setText(resultRx+ "B");
 }
}
```

6. PENGUJIAN DAN ANALISIS

6.1 Skenario Pengujian

Pengujian dilakukan dengan percobaan pengambilan data dengan format *JSON*. Terdapat dua data berbeda yaitu data yang berisi hanya teks dan data yang berisi teks dan gambar. Data tersebut akan diparsing agar dapat ditampilkan dengan baik pada antarmuka aplikasi. Pengujian dilakukan sebanyak 20 kali pada masing-masing data dari tiap *library* untuk mendapatkan variasi nilai.

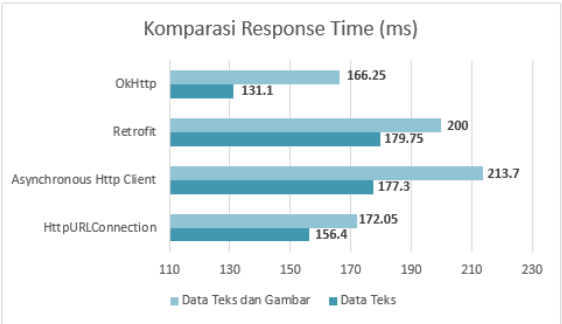
6.2 Pengujian *Response Time*

Komparasi *response time* berdasarkan pengujian yang telah dilakukan sebelumnya. Terdapat hasil pengujian dari dua skenario yaitu hasil pengujian dengan data teks, serta data teks dan gambar yang ditunjukkan pada Tabel 4.

Tabel 4. Komparasi *Response Time*

Tipe data	Hanya Teks	Teks dan gambar	Rata-rata
HttpURLConnection	156.4ms	172.05ms	164.225ms
Asynchronous Http Client	177.3ms	213.7ms	195.5ms
Retrofit	179.75ms	200ms	189.875ms
OkHttp	131.1ms	166.25ms	148.675ms

Komparasi pada Tabel 4 menunjukkan bahwa terjadi penurunan kecepatan *response time* dari setiap *library*. Kecepatan rata-rata response time pada *HttpURLConnection* sebesar 164.225 ms, *Asynchronous Http Client* sebesar 195.5 ms, *Retrofit* sebesar 189.875 ms, dan *OkHttp* sebesar 148.675 ms. Grafik komparasi *response time* ditunjukkan pada Gambar 2.



Gambar 2. Komparasi *Response Time*

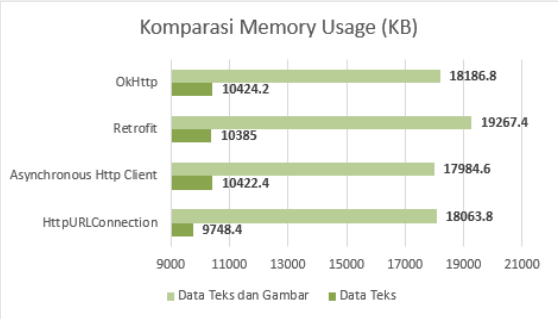
6.3 Pengujian *Memory Usage*

Komparasi *memory usage* berdasarkan pengujian yang telah dilakukan sebelumnya. Terdapat hasil pengujian dari dua skenario yaitu hasil pengujian dengan data teks, serta data teks dan gambar yang ditunjukkan Tabel 5.

Tabel 5. Komparasi *Memory Usage*

Tipe data	Hanya Teks	Teks dan gambar	Rata-rata
HttpURLConnection	9748.4KB	18063.8KB	13906.1KB
Asynchronous Http Client	10422.4KB	17984.6KB	14203.5KB
Retrofit	10385KB	19267.4KB	14826.2KB
OkHttp	10424.2KB	18186.8KB	14305.5KB

Komparasi pada Tabel 5 menunjukkan bahwa terjadi peningkatan *memory usage* dari setiap *library*. Nilai rata-rata *memory usage* pada *HttpURLConnection* sebesar 13906.1 KB, *Asynchronous Http Client* sebesar 14203.5 KB, *Retrofit* sebesar 14826.2 KB, dan *OkHttp* sebesar 14305.5 KB. Grafik komparasi *memory usage* ditunjukkan pada Gambar 3.



Gambar 3. Komparasi Response Time

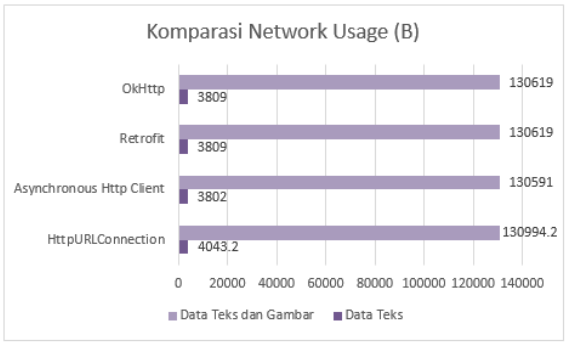
6.4 Pengujian *Network Usage*

Komparasi *network usage* berdasarkan pengujian yang telah dilakukan sebelumnya. Terdapat hasil pengujian dari dua skenario yaitu hasil pengujian dengan data teks, serta data teks dan gambar yang ditunjukkan Tabel 6.

Tabel 6. Komparasi *Network Usage*

Tipe data	Hanya Teks	Teks dan gambar	Rata-rata
HttpURLConnection	4043.2 B	130994.2 B	67518.7 B
Asynchronous Http Client	3802B	130591B	67196.5 B
Retrofit	3809B	130619B	67214B
OkHttp	3809KB	130619B	67214B

Komparasi pada Tabel 6 menunjukkan bahwa terjadi peningkatan *network usage* dari setiap *library*. Nilai rata-rata *network usage* pada *HttpURLConnection* sebesar 67518.7 B, *Asynchronous Http Client* sebesar 67196.5 B, *Retrofit* sebesar 67214 B, dan *OkHttp* sebesar 67214 B. Grafik komparasi *network usage* ditunjukkan pada Gambar 4.



Gambar 4. Komparasi *Network Usage*

6.5 Analisis Kesimpulan

Analisis kesimpulan merupakan sebuah tahap akhir dalam penelitian analisis performansi *http networking library* pada *android* (studi kasus: portal berita). Analisis kesimpulan bertujuan untuk menganalisis keseluruhan penelitian meliputi analisis *response time*, *memory usage*, dan *network usage*. Hasil dari analisis adalah sebagai berikut :

1. *Library OkHttp* lebih baik dalam segi kecepatan *response time* daripada *library* lainnya dalam kasus pengujian data teks maupun data teks serta data teks dan gambar. *Response time OkHttp* memiliki rata-rata sebesar 148.675 ms. *Library OkHttp* memiliki *response time* yang lebih cepat dibanding *library* lainnya karena pada *library OkHttp* memiliki suatu mekanisme *response caching* dan menggunakan *library Okio*. *Library* tersebut lebih efisien dalam *read write* data daripada standar *Java I/O* seperti *InputStream* dan *OutputStream*.
2. *Library Asynchronous Http Client* dalam kasus pengujian data teks maupun data teks serta data teks dan gambar memiliki *response time* yang lebih lambat daripada *library* lainnya dengan rata-rata *response time* sebesar 195.5 ms. *Library Asynchronous Http Client* memiliki *response time* yang lambat dikarenakan *library* tersebut tidak memiliki mekanisme *response caching*. Inisialisasi *handler JsonHttpReponseHandler* yang mengubah *response* ke dalam bentuk *JSON* juga membuat waktu *response* time lebih lama.
3. *HttpURLConnection* dalam segi *memory usage* lebih baik daripada *library* lainnya

dengan rata-rata *memory usage* sebesar 13906.1 KB. *Memory usage* yang kecil dikarenakan *HttpURLConnection* merupakan *class* turunan *URLConnection* dan tidak tergantung pada *library* lainnya.

4. *Library Retrofit* dalam segi *memory usage* membutuhkan *resource memory* lebih banyak daripada *library* lainnya dengan rata-rata *memory usage* sebesar 14826.2 KB. *Resource memory* yang banyak dikarenakan *library Retrofit* merupakan *library* yang dikembangkan dari *library OkHttp* untuk membuat koneksi *http*. Serta *response* yang diterima akan diubah ke dalam bentuk *Plain Old Java Object (POJO)* sehingga membutuhkan *resource memory* yang lebih.
5. *Library Asynchronous Http Client* memiliki *network usage* yang lebih sedikit dibanding *library* lainnya dengan rata-rata sebesar 67196.5 KB. *Network usage* yang sedikit dikarenakan data yang ditransmisikan (*request*) ke *server* lebih sedikit dibandingkan *library* yang lainnya.
6. *HttpURLConnection* memiliki *network usage* yang lebih banyak daripada *library* lainnya dengan rata-rata *network usage* sebesar 67518.7 KB. *Network usage* yang banyak dikarenakan data yang ditransmisikan (*request*) ke *server* lebih banyak dibandingkan *library* yang lainnya.

7. KESIMPULAN

Komparasi antar *library* dengan cara melakukan pengujian pada data berupa teks sebanyak 20 kali pada masing-masing *library*. Sehingga didapatkan nilai rata-rata pada tiap parameter yaitu *response time*, *memory usage*, dan *network usage*. Dilakukan juga pengujian yang sama pada data berupa teks dan gambar sehingga didapatkan nilai rata-rata tiap parameternya. Dari kedua nilai tersebut akan diambil rata-ratanya untuk mendapatkan hasil akhir pada tiap parameter. Nilai tersebut digunakan sebagai perbandingan performansi tiap *library*.

Library Asynchronous Http Client membutuhkan *network usage* lebih sedikit daripada *library* lainnya dengan rata-rata *network usage* sebesar 67196.5 B. *Library Retrofit* memiliki rata-rata *memory usage* lebih banyak daripada *library* lainnya dengan rata-rata *memory usage* sebesar 14826.2 KB. *Library OkHttp* memiliki rata-rata *response time* lebih cepat daripada *library* lainnya dengan rata-rata *response time* sebesar 148.675 ms. Sedangkan *HttpURLConnection* memiliki *memory usage* yang lebih kecil daripada *library* lainnya dengan rata-rata *memory usage* sebesar 13906.1 KB.

Pengaruh adanya data berupa gambar ialah terjadi penurunan kecepatan *response time* pada seluruh *library*. Pada *library HttpURLConnection* terjadi penurunan sebesar 15.65 ms. Pada *library Asynchronous* terjadi penurunan sebesar 36.4 ms. Pada *library Retrofit* terjadi penurunan sebesar 24.95 ms. Pada *library OkHttp* terjadi penurunan sebesar 35.15 ms. Terjadi peningkatan *memory usage* pada seluruh *library* karena gambar dikonversi ke dalam bentuk *bitmap* yang juga memakan *resource memory*. Pada *library HttpURLConnection* terjadi peningkatan sebesar 8315.4 KB. Pada *library Asynchronous* terjadi peningkatan sebesar 7562.2 KB. Pada *library Retrofit* terjadi peningkatan sebesar 8882.4 KB. Pada *library OkHttp* terjadi peningkatan sebesar 7762.6 KB. Terjadi peningkatan pada *network usage* pada seluruh *library* dikarenakan aspek ini juga dipengaruhi oleh ukuran dari gambar. Pada *library HttpURLConnection* terjadi peningkatan sebesar 126951 B. Pada *library Asynchronous* terjadi peningkatan sebesar 126788 B. Pada *library Retrofit* terjadi peningkatan sebesar 126832 B. Pada *library OkHttp* terjadi peningkatan sebesar 126832 B.

DAFTAR PUSTAKA

Al-Fedaghi, S., 2011. Developing web application, s.l.: International journal of software engineering and its applications.

Berners-Lee, T., Fielding, R. & Frystyk, H., 1996. Hypertext Transfer Protocol -- HTTP/1.0, s.l.: RFC 1945.

Chen, H., 2015. Management of contextual information for data, s.l.: s.n.

Codepath, 2015. Consuming APIs with Retrofit. [Online]
Available at: https://guides.codepath.com/android/Consuming-APIs-with-Retrofit
[Accessed 20 September 2016].

Developers, A., 2016. HttpURLConnection. [Online]
Available at: https://developer.android.com/reference/java/net/HttpURLConnection.html
[Accessed 5 December 2016].

Developers, A., 2016. TrafficStats. [Online] Available at: https://developer.android.com/reference/android/net/TrafficStats.html [Accessed 10 December 2016].

Egham, 2016. Gartner Says Five of Top 10 Worldwide Mobile Phone Vendors Increased Sales in Second Quarter of 2016. [Online] Available at: http://www.gartner.com/newsroom/id/3415117 [Accessed 10 Decemver 2016].

Irsya, E., Santosa, H. P. & Luqman, Y., 2013. Memahami Fenomena Komunikasi Hiperpersonal Menggunakan Anonymous Username dalam Portal Berita Online, s.l.: s.n.

ISO, 2016. ISO/IEC 25010:2011. [Online] Available at: https://www.iso.org/obp/ui/#iso:std:iso-iec:25010:ed-1:v1:en [Accessed 10 December 2016].

JSON, 2016. Introducing JSON. [Online] Available at: http://www.json.org/ [Accessed 20 September 2016].

Ladan, M. I., 2011. Web Services Metrics: A survey and A Classification, s.l.: International Conference on Network and Electronics Engineering.

Maskov, V., 2015. Implementing REST Client for Android, s.l.: Helsinki Metropolia University of Applied Sciences.

OkHttp, 2016. OkHttp : An HTTP & HTTP/2 client for Android and Java applications. [Online] Available at: http://square.github.io/okhttp/ [Accessed 20 September 2016].

Ramanathan, R., 2014. Software service architecture to access weather data using restful web services, s.l.: IEEE.

Smith, J., 2013. Android Asynchronous Http Client. [Online] Available at: http://loopj.com/android-async-http/ [Accessed 20 September 2016].

Suhandang, K., 2014. Pengantar Jurnalistik: Seputar Organisasi, Produk, & Kode Etik, Bandung: Nuansa.

Wei, Z., 2014. LazyTainter : Memory-Efficient Taint Tracking in Managed Runtimes, s.l.: University of Toronto.