JEPIN
(Jurnal Edukasi dan Penelitian Informatika)
ISSN(e): 2548-9364 / ISSN(p) : 2460-0741

Vol. 10
No. 1
April 2024

LOGAN: *Log Analyzer Simulation* Sebagai Media Simulasi Analisis Log Berbasis Data Generator

Muhammad Koprawi[#1], Wahyu Sukestyastama Putra[#2], Wahid Miftahul Ashari[#3], Jeki Kuswanto[#4], Anggit Ferdita Nugraha[#5]

[#]*Program Studi Teknik Komputer, Fakultas Ilmu Komputer, Universitas AmikomYogyakarta, Indonesia*
Jl. Ring Road Utara, Ngringin, Condongcatur, Kec. Depok, Kabupaten Sleman, Daerah Istimewa Yogyakarta 55281

[1]koprawi@amikom.ac.id
[2]wahyu@amikom.ac.id
[3]wahidashari@amikom.ac.id
[4]jeki@amikom.ac.id
[5]anggitferdita@amikom.ac.id

Abstrak— **Aktivitas pengunjung sebuah *website* akan tersimpan di *server* pada *file* yang bernama *access log*. *Access log* menyimpan berbagai informasi penting seperti IP *address* pengunjung sampai *user agent* yang digunakan (*browser*, sistem operasi, dan sebagainya). Untuk mendapatkan *file access log* setidaknya harus memiliki akses ke *server* sehingga *file* tersebut dapat diambil kemudian dianalisis. Pada penelitian ini, dibuat sebuah sistem yang digunakan sebagai alat untuk membuat *file access log* secara otomatis dan dirancang mirip seperti format *file access log* yang ada pada *server*. Selain sebagi alat untuk membuat *file access log*, sistem ini juga dirancang untuk melakukan analisis informasi terhadap *file access log* tersebut dengan menyajikannya dalam bentuk visualisasi. Metode yang digunakan untuk mengembangkan sistem ini adalah metode pengembangan *Rapid Application Development* (RAD). Metode RAD dipilih salah satunya karena siklus pengembangan yang relatif lebih cepat dan singkat tetapi tetap dapat menghasilkan kualitas sistem yang baik. Untuk pengujian terhadap sistem dilakukan beberapa skenario yaitu membuat *file temporary access log* dengan 1000 baris *log* yang menghasilkan waktu rata-rata 0,791 detik, proses *insert* data ke *database* yang bersumber dari *file temporary access log* menghasilkan waktu rata-rata 0,271 detik, dan skenario terakhir yaitu membuat *log on the fly* kemudian langsung *insert* data ke *database* menghasilkan waktu rata-rata 4,833 detik. Sehingga dari hasil pengujian tersebut didapatkan skenario yang efektif dan tidak membebani sistem terutama bagian *database* yaitu membuat *file temporary access log* terlebih dahulu kemudian *insert* data ke *database* yang bersumber dari *file temporary access log*.**

Kata kunci— ***Log Anaylzer Simulation*, *Access Log*, Data Sintetis, *Log Generator***

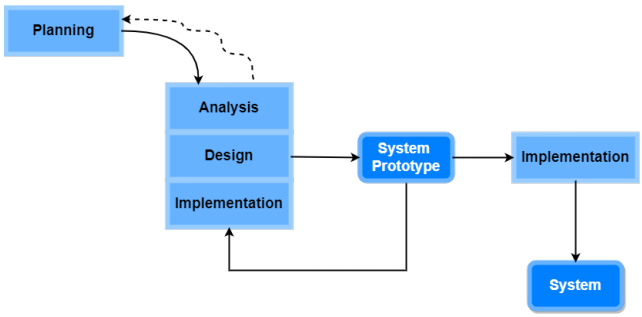
I. PENDAHULUAN

Menurut laporan tahunan hasil monitoring keamanan siber tahun 2021 oleh Badan Siber dan Sandi Negara (BSSN) tercatat lebih dari 1,6 Milyar lebih trafik anomali yang terjadi sepanjang tahun 2021[1]. Salah satu cara mengetahui anomali trafik dan mendeteksi serangan siber [2]–[9] adalah dengan cara melihat dan melakukan analisis terhadap *log* yang ada di *server*. Selain sebagai salah satu cara mendeteksi serangan siber, data *log* dapat digunakan untuk keperluan investigasi forensik[10]. Pada penelitan [11], [12] data *log* juga digunakan untuk memetakan halaman situs web sehingga didapatkan informasi seperti jumlah pengunjung sampai situs web yang sering dikunjungi.

Berdasarkan penelitian sebelumnya, data *log* yang diolah bersumber dari data *log* yang sudah diakuisisi dari *server*, ketika data *log* tersebut tidak tersedia maka analisis terhadap data *log* tersebut tidak dapat dilakukan. Pada penelitian ini, peneliti bermaksud mengusulkan sebuah sistem yang bernama Logan: *Log Analyzer Simulation based on data generator* yang bisa digunakan untuk melakukan simulasi analisis data *log* yang bersumber dari generator *log* yang dibuat mirip sedemikian rupa dengan data *log* yang asli. Selain digunakan sebagai generator data *log*, sistem yang diusulkan juga bisa melakukan visualisasi terhadap data *log* tersebut. Generator *log* yang akan dibuat menggunakan format dari *access log apache web server*.

II. METODE PENELITIAN

Peneltian ini menggunakan pendekatan metode RAD (*Rapid Application Development*). Metode RAD dapat meningkatkan kecepatan dan kualitas dari pengembangan [13]–[17]. Pengembangan sistem pada penelitian ini menggunakan jenis RAD yang disebut dengan *system prototyping* atau purwarupa sistem. Purwarupa sistem ini memiliki 3 fase seperti fase analisis, desain dan implementasi[13]. Fase-fase tersebut bisa dilihat pada Gambar 1. Fase purwarupa tersebut akan terus dilakukan sampai semua kebutuhan selesai.



Gambar 1. *System prototyping*[13]

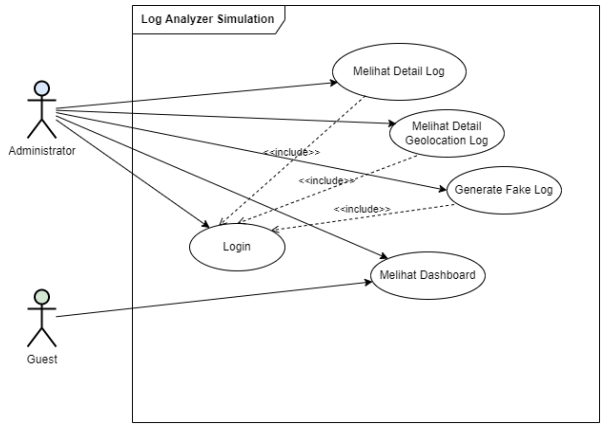
A. *Analisis*

Pada fase analisis ini terbagi menjadi dua bagian yaitu analisis perancangan sistem dan analisis *generate* data *log* sintetis.

- Analisis perancangan sistem
  Perancangan sistem menggunakan pendekatan pemodelan UML (*Unified Modeling Language*).
- Analisis *generate* data *log* sintetis
  Generate data *log* sintetis menggunakan format *log* dari *Apache Web Server* dan menggunakan bantuan *faker* yang dijalankan pada bahasa pemrograman python.

B. *Use Case Diagram*

Gambar 2 menunjukkan *use case diagram* yang ada pada aplikasi Logan: *Log Analyzer Simulation based on data generator*. Pada aplikasi ini terdapat dua aktor yaitu *administrator* dan *guest*. Masing-masing aktor memiliki *role* akses yang berbeda seperti ditunjukkan pada *use case* pada Gambar 2.

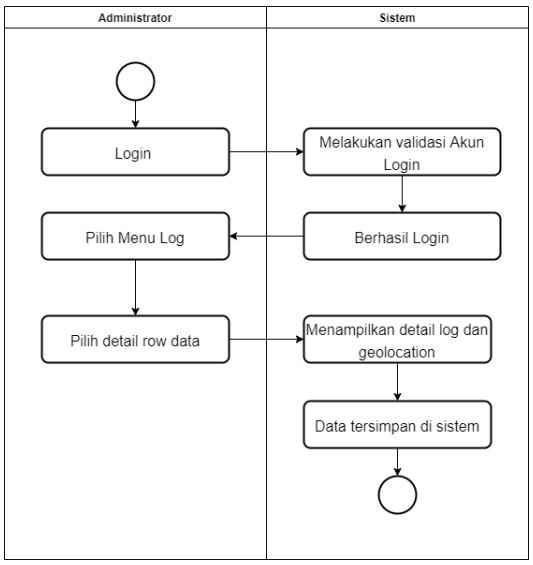


Gambar 2. *use case log analyzer simulation*

C. *Activity Diagram*

Activity diagram utama pada aplikasi Logan: *Log Analyzer Simulation based on data generator* terbagi menjadi 3 bagian yaitu:

1. *Activity diagram Generate* data *log* sintetis

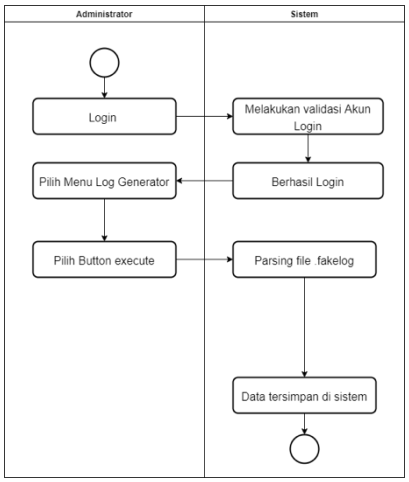


Gambar 3. *Activity diagram generate* data *log* sintetis

Keterangan:

- Login menggunakan *credential administrator*
- Pilih menu Log Generator
- *Input seed* dan jumlah baris yang akan di*generate*
- *Generate file* dengan ekstensi *.fakelog* sesuai dengan *seed* dan jumlah baris yang sudah diatur sebelumnya
- *File log* tersimpan

2. *Activity diagram execute* data *log* sintetis

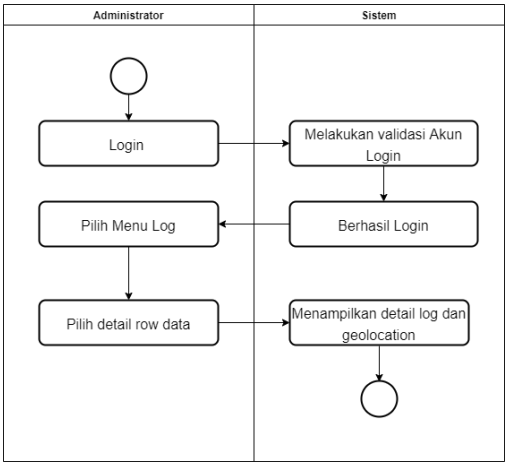


Gambar 4. *Activity diagram execute* data *log* sintetis

Keterangan:

- *Login* menggunakan *credential administrator*
- Pilih menu *Log Generator*
- Pilih *button execute*
- Semua data yang ada di dalam *file log* akan di*parsing* ke *database*
- Data tersimpan

3. *Activity Diagram Detail Geolocation Log*



Gambar 5. *Activity diagram* detail *geolocation log*

Keterangan:

- *Login* menggunakan *credential administrator*
- Pilih menu *Log*
- Pilih detail *row* data
- Sistem akan menampilkan detail *log* dan informasi *geolocation* berdasarkan *ip address*

D. Perancangan Tabel

Informasi terkait data *log* akan disimpan ke dalam sebuah tabel dengan *nama tabel logger* dengan atribut seperti pada Gambar 6.

```
CREATE TABLE logger (
        id INTEGER NOT NULL,
        ipv4 VARCHAR(20) NOT NULL,
        req_method VARCHAR(6) NOT NULL,
        uri VARCHAR(256) NOT NULL,
        status_code INTEGER NOT NULL,
        referer VARCHAR(256) NOT NULL,
        user_agent VARCHAR(256) NOT NULL,
        date_time DATETIME NOT NULL,
        protocol VARCHAR(10) NOT NULL,
        size INTEGER NOT NULL,
        PRIMARY KEY (id)
)
```

Gambar 6. Atribut pada *tabel logger*

Pendefinisian atribut berdasarkan pada format *log* yang digunakan pada *apache*[18]–[20].

E. Fungsi Visualisasi

Berikut adalah rancangan fungsi visualisasi yang akan dibuat pada aplikasi Logan: *Log Analyzer Simulation based on data generator* pada Tabel 1.

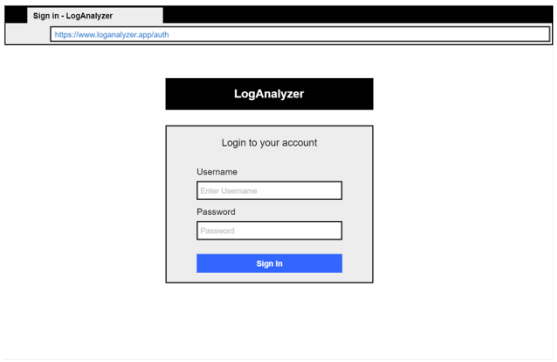
TABEL I
RANCANGAN FUNGSI VISUALISASI

Fungsi	Keterangan
Most IP Hit	*IP Address* yang paling banyak akses ke sebuah *website*
Not Found URL	Akses URL dengan status *not found*
HTTP Status Codes	Jumlah *HTTP Status Code* pada *log*
Browser	*Browser* yang digunakan ketika akses ke sebuah *website*
Operating System	Sistem Operasi yang digunakan untuk akses ke sebuah *website*
Detail Geolocation Log	Informasi *geo* lokasi berdasarkan *IP Address*

F. Design

Berikut rancangan antarmuka dalam bentuk *wireframe* pada aplikasi Logan: *Log Analyzer Simulation based on data generator*:

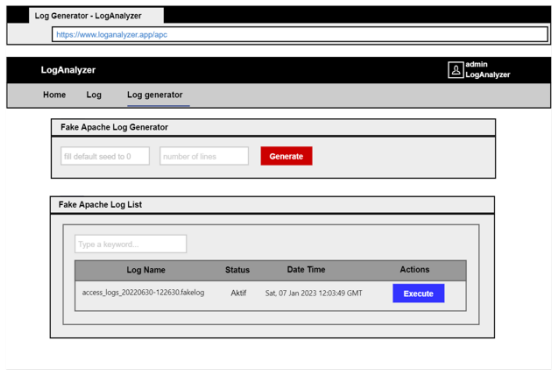
1. Rancangan Halaman *Login*



Gambar 7. Desain antarmuka halaman *login*

Gambar 7 merupakan rancangan antarmuka halaman *login* yang terdiri dari dua *field* yaitu *username* dan *password*.

2. Rancangan *Log Generator*

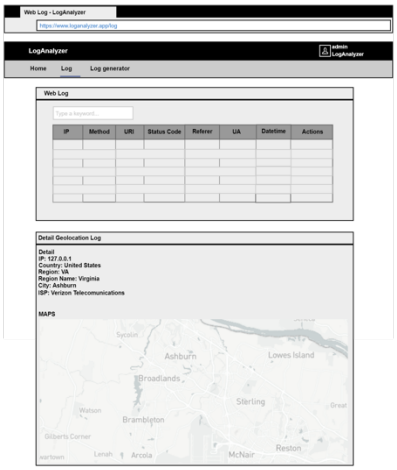


Gambar 8. Desain antarmuka *log generator*

Gambar 8 merupakan rancangan antarmuka pada halaman *log generator*, pada halaman ini digunakan untuk membuat atau membangkitkan data *log* dengan baris *log*

yang bisa diatur jumlahnya serta digunakan untuk mengeksekusi *file* data *log* yang sudah dibuat sebelumnya.

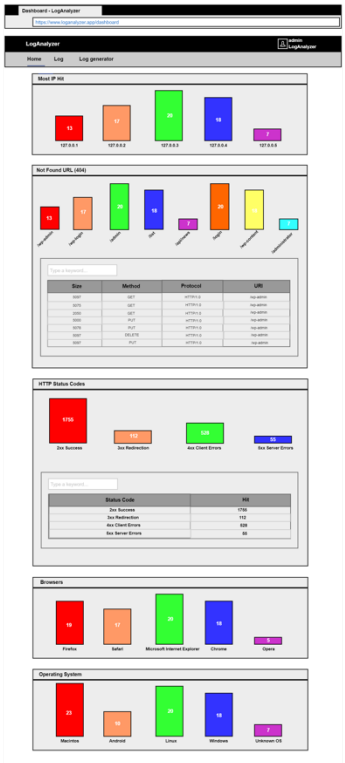
3. Rancangan *Geolocation Log*



Gambar 9. Desain antarmuka *geolocation log*

Gambar 9 merupakan rancangan antarmuka untuk *geolocation log*, pada halaman ini merupakan halam detail informasi berbasis *geolocation*. *IP address* yang tercatat di *log* bisa dilakukan *tracing* seperti negara, kota, ISP yang digunakan oleh *IP address* tersebut serta halaman ini menampilkan lokasi berbasi peta berdasarkan *IP address*.

4. Rancangan *Dashboard*



Gambar 10. Desain antarmuka *dashboard*

Gambar 10 merupakan rancangan antarmuka untuk halaman *dashboard*. Halaman *dashboard* menyajikan informasi terkait *Most IP Hit, Not Found URL, HTTP Status Codes, Browser, Operating System* dan *Detail Geolocation Log*. Untuk lebih jelasnya bisa dilihat kembali pada Tabel 1. Halaman *dashboard* ini bisa diakses oleh user dengan *role guest* dan *administrator*.

G. Data Generator

Untuk membuat data *log* sintetis salah satunya bisa menggunakan *tool* berikut[21]. *Log* yang dibuat menggunakan format *log* dari *Apache Web Server*. Format *log* dari *Apache* terdiri dari dua format yaitu CLF *(Common Log Format)* dan ELF *(Combined/Extended Log Format)*[18]–[20]. Lebih jelasnya format dari CLF dan ELF bisa diihat pada Tabel 2.

TABEL II
FORMAT *LOG APACHE*

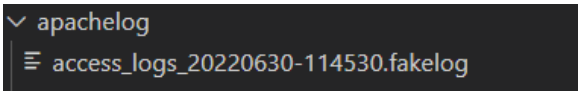
Format Log	Format String
CLF	"%h %l %u %t \"%r\" %>s %b"
ELF	"%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-agent}i\""

Untuk lebih jelasnya deskripsi terkait format *string* pada format *log apache* dijelaskan pada Tabel 3.

TABEL III
DESKRIPSI FORMAT *STRING*[19]

Format String	Keterangan
%h	*IP Address* dari klien, bisa dalam format IPv4 atau IPv6
%l	*Remote logname,* pada penelitian ini akan diisi dengan tanda "-"
%u	*Remote username,* pada penelitian ini akan diisi dengan tanda "-"
%t	Waktu request, secara default menggunakan format [day/month/year:hour:minute:second zone]
%r	Berisi baris *request* dari klien, biasanya berisi *request method, path, query string* dan protokol
%>s	Kode status *request*
%b	Ukuran respon dalam bentuk *bytes*
%{Referer}i	Rujukan URL dari *request*
%{User-agent}i	*User agent* dari *request* seperti jenis *browser,* sistem operasi dan sebagainya.

Pada penelitian ini format *log* yang akan dipakai adalah format *log* ELF. Format *log* ELF informasi yang dapat diambil dan diolah lebih detail dibandingkan format CLF[18], [19], [22]. Kemudian untuk membedakan *log* yang akan dibuat oleh aplikasi Logan*: Log Analyzer Simulation based on data generator* maka disini peneliti mebuat ekstensi baru dengan nama *.fakelog*.
Berikut contoh file log yang dibuat melalui aplikasi Logan: *Log Analyzer Simulation based on data generator* bisa dilihat pada Gambar 11.



Gambar 11. Log dengan ekstensi *.fakelog*

Kemudian contoh isi dari *file log* yang sudah dibuat bisa dilihat pada Gambar 12.

```
171.174.170.81 - - [05/Jun/2022:12:53:20
+0700] "GET /wp-admin HTTP/1.0" 200
4943 "https://hull-gallegos.info/" "Mozilla/5.0
(Linux; Android 8.0.0) AppleWebKit/533.2
(KHTML, like Gecko) Chrome/56.0.842.0
Safari/533.2"
```

Gambar 12. Isi *file log*

Untuk mengukur waktu rata-rata yang diperlukan ketika data *log* dibuat menggunakan formula berikut:

$$\bar{x} = \frac{\sum x}{n} \quad (1)$$

Keterangan:

$\bar{x}$ = waktu rata-rata (detik)

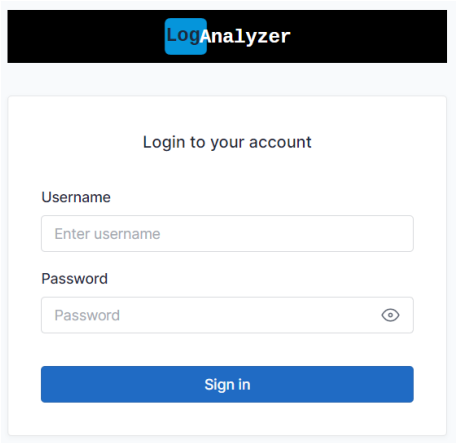
$\sum x$ = jumlah waktu(detik)

n = jumlah *file access log*

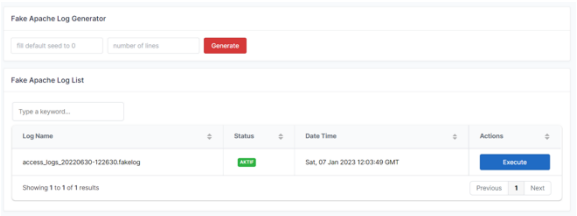
III. HASIL DAN PEMBAHASAN

Berdasarkan rancangan antarmuka sebelumnya dilanjutkan dengan implementasi antarmuka tersebut seperti halaman *login, log generator, geolocation log* dan *dashboard*. Untuk implementasi menggunakan bahasa pemrograman python dan menggunakan *framework* flask[23].

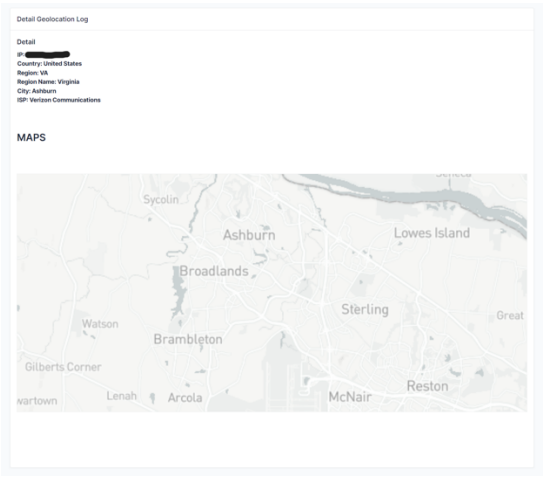
Berikut implementasi halaman *login* pada Gambar 13.



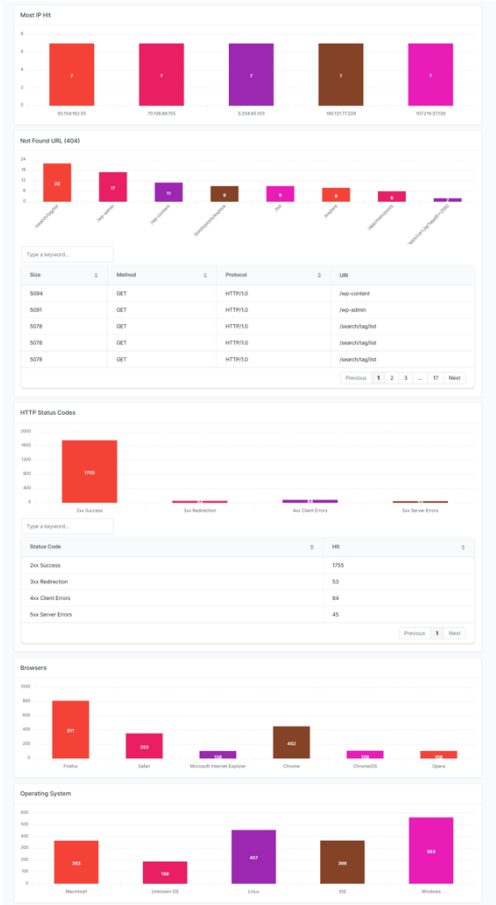
Gambar 13. Halaman *login*



Gambar 14. Halaman *log generator*



Gambar 15. Halaman *geolocation log*



Gambar 16. Halaman *dashboard*

A. Pengujian

Skenario pengujian sistem terbagi menjadi tiga, yaitu pengujian terhadap data, pengujian terhadap visualisasi data dan pengujian terhadap fungsi.

1) Skenario Pengujian data: Skenario pengujian data yang dilakukan adalah untuk menguji waktu rata-rata yang diperlukan untuk membuat *log* sampai menyimpan *log* ke dalam *database*. Jumlah baris yang dibuat dalam satu *file* temporari *access log* adalah sebanyak 1.000 baris. Adapun skenario pengujiannya sebagai berikut:

- *Generate file temporary access log*
- *Insert* data ke *database* berdasarkan *file temporary access log*
- *Generate log on the fly* kemudian langsung *insert* data ke *database*

TABEL IV
PERHITUNGAN WAKTU *GENERATE FILE TEMPORARY ACCESS LOG*

Nama log	Waktu (detik)
access_logs_20220705-125146.fakelog	0,844
access_logs_20220705-125150.fakelog	0,797
access_logs_20220705-125153.fakelog	0,766
access_logs_20220705-125157.fakelog	0,766
access_logs_20220705-125201.fakelog	0,828
access_logs_20220705-125205.fakelog	0,781
access_logs_20220705-125208.fakelog	0,844
access_logs_20220705-125212.fakelog	0,828
access_logs_20220705-125216.fakelog	0,797
access_logs_20220705-125220.fakelog	0,828
access_logs_20220705-125223.fakelog	0,812
access_logs_20220705-125227.fakelog	0,797
access_logs_20220705-125231.fakelog	0,828
access_logs_20220705-125235.fakelog	0,719
access_logs_20220705-125238.fakelog	0,719
access_logs_20220705-125242.fakelog	0,781
access_logs_20220705-125246.fakelog	0,766
access_logs_20220705-125250.fakelog	0,75
access_logs_20220705-125253.fakelog	0,75
access_logs_20220705-125257.fakelog	0,828
Rata-rata: 0,791	

Pada Tabel 4 menunjukkan waktu rata-rata yang diperlukan untuk *generate file temporary access log* dengan jumlah file uji sebanyak 20 buah. Waktu rata-rata yang dihasilkan adalah 0,791 detik.

TABEL V
PERHITUNGAN WAKTU *INSERT* DATA KE *DATABASE* BERDASARKAN *FILE TEMPORARY ACCESS LOG*

Nama log	Waktu (detik)
access_logs_20220705-125146.fakelog	0,25
access_logs_20220705-125150.fakelog	0,234
access_logs_20220705-125153.fakelog	0,203
access_logs_20220705-125157.fakelog	0,234
access_logs_20220705-125201.fakelog	0,219
access_logs_20220705-125205.fakelog	0,219
access_logs_20220705-125208.fakelog	0,203
access_logs_20220705-125212.fakelog	0,234
access_logs_20220705-125216.fakelog	0,297
access_logs_20220705-125220.fakelog	0,328
access_logs_20220705-125223.fakelog	0,312
access_logs_20220705-125227.fakelog	0,281
access_logs_20220705-125231.fakelog	0,297
access_logs_20220705-125235.fakelog	0,297
access_logs_20220705-125238.fakelog	0,297
access_logs_20220705-125242.fakelog	0,281
access_logs_20220705-125246.fakelog	0,312
access_logs_20220705-125250.fakelog	0,297
access_logs_20220705-125253.fakelog	0,312
access_logs_20220705-125257.fakelog	0,312
Rata-rata: 0,271	

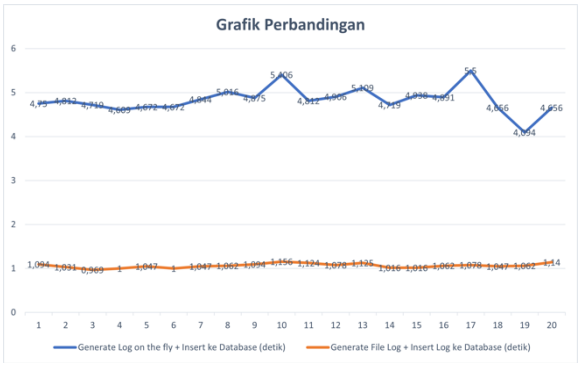
Pada Tabel 5 menunjukkan waktu rata-rata yang diperlukan untuk *insert* data ke *database* berdasarkan *file temporary access log* yang dibuat pada tahapan sebelumnya. Waktu rata-rata yang dihasilkan adalah 0,271 detik.

TABEL VI
PERHITUNGAN WAKTU *GENERATE LOG ON THE FLY* KEMUDIAN LANGSUNG *INSERT* DATA KE *DATABASE*

Iterasi	Waktu (detik)
1	4,75
2	4,812
3	4,719
4	4,609
5	4,672
6	4,672
7	4,844
8	5,016
9	4,875
10	5,406
11	4,812
12	4,906
13	5,109
14	4,719
15	4,938
16	4,891
17	5,5
18	4,656
19	4,094
20	4,656
Rata-rata: 4,833	

Pada Tabel 6 menunjukkan waktu rata-rata yang diperlukan untuk *generate log on the fly* kemudian *insert* data ke *database*. Waktu rata-rata yang dihasilkan adalah 4,833 detik.

Dari pengujian di atas menghasilkan grafik perbanding seperti Gambar 17.



Gambar 17. Grafik perbandingan

Berdasarkan Gambar 17 dapat ditentukan skenario yang efektif untuk membuat *log* dan kemudian *insert* ke *database* adalah dengan cara membuat *file log* terlebih dahulu kemudian *file log* tersebut dieksekusi sehingga data *log* masuk ke *database*.

2) Skenario pengujian visualisasi data: Skenario pengujian visualisasi data berdasarkan pada rancangan antarmuka dan rancangan fungsi yang sudah dijelaskan sebelumnya. Hasil pengujian dituliskan pada Tabel 7.

TABEL VII
HASIL PENGUJIAN VISUALISASI DATA

Skenario Pengujian	Hasil Pengujian
Sistem menampilkan visualisasi data *Most IP Hit*	Sesuai
Sistem menampilkan visualisasi data *Not Found URL*	Sesuai
Sistem menampilkan visualisasi data *HTTP Status Code*	Sesuai
Sistem menampilkan visualisasi data *Browser*	Sesuai
Sistem menampilkan visualisasi data *Operating System*	Sesuai
Sistem menampilkan visualisasi data *Detail Geo Location*	Sesuai

3) Skenario pengujian fungsi: Skenario pengujian fungsi dibagi menjadi pengujian fungsi *login*, pengujian fungsi *generate log* dan pengujian eksekusi *log*.

- Pengujian Terhadap Fungsi *Login*

TABEL VIII
HASIL PENGUJIAN FUNGSI *LOGIN*

Skenario Pengujian	Hasil yang diharapkan	Hasil Pengujian
Input *username* dan *password* kosong	Sistem menampilkan pesan *error* dan menolak proses *login*	Sesuai
Hanya mengisi inputan *username* dan mengosongkan inputan *password*	Sistem menampilkan pesan *error* dan menolak proses *login*	Sesuai
Input *username* dan *password* yang tidak terdaftar di *database*	Sistem menampilkan pesan *error* dan menolak proses *login*	Sesuai
Input *username* dan *password* yang terdaftar di *database*	Sistem menerima proses *login* dan masuk ke halaman *dashboard*	Sesuai

- PengujianTerhadap Fungsi *Generate Log*

TABEL IX
HASIL PENGUJIAN FUNGSI *GENERATE LOG*

Skenario Pengujian	Hasil yang diharapkan	Hasil Pengujian
Input *field seed* dan *number of lines* kosong	Sistem menampilkan pesan *error* dan menolak proses *generate*	Sesuai
Hanya mengisi inputan *field seed* dan mengosongkan inputan *field number of lines*	Sistem menampilkan pesan error dan menolak proses *generate*	Sesuai
Hanya mengisi inputan *field number of lines* dan mengosongkan inputan *field seed*	Sistem menampilkan pesan *error* dan menolak proses *generate*	Sesuai
Input *field seed* dan *field number of lines*	Sistem melakukan proses *generate*	Sesuai

- Pengujian Terhadap Fungsi Eksekusi *Log*

TABEL X
HASIL PENGUJIAN FUNGSI EKSEKUSI *LOG*

Skenario Pengujian	Hasil yang diharapkan	Hasil Pengujian
Eksekusi *File Log* menggunakan *button execute*	Sistem melakukan proses eksekusi *file log*	Sesuai

IV. KESIMPULAN

Logan: *Log Analyzer Simulation based on data generator* sebagai media analisis *log* telah berhasil dikembangkan. Logan: *Log Analyzer Simulation based on data generator* tidak hanya dirancang sebagai media analisis menggunakan visualisasi *log* seperti (*Most IP Hit,*

Not Found URL, HTTP Status Codes, Browser, Operating System, Detail Geolocation Log) tetapi juga sebagai sistem yang bisa melakukan *generate* data *access log* yang mirip dengan format *log* aslinya. Berdasarkan hasil pengujian terhadap sistem telah dilakukan beberapa skenario yaitu membuat *file temporary access log* dengan 1000 baris *log* yang menghasilkan waktu rata-rata 0,791 detik, proses *insert* data ke *database* yang bersumber dari *file temporary access log* menghasilkan waktu rata-rata 0,271 detik, dan skenario terakhir yaitu membuat *log on the fly* kemudian langsung *insert* data ke *database* menghasilkan waktu rata-rata 4,833 detik. Sehingga dari hasil pengujian tersebut didapatkan skenario yang efektif dan tidak membebani sistem terutama bagian *database* yaitu membuat *file temporary access log* terlebih dahulu kemudian *insert* data ke *database* yang bersumber dari *file temporary access log*.

UCAPAN TERIMAKASIH

Terima kasih kepada Universitas Amikom Yogyakarta melalui Lembaga Penelitian Universitas Amikom Yogyakarta atas dukungannya sehingga penelitian ini selesai dilaksanakan.

REFERENSI

[1] BSSN, "Cloud BSSN." https://cloud.bssn.go.id/s/Lyw8E4LxwNiJoNw (accessed May 08, 2022).

[2] T. F. Yen *et al.*, "Beehive: Large-scale log analysis for detecting suspicious activity in enterprise networks," *ACM International Conference Proceeding Series*, pp. 199–208, 2013, doi: 10.1145/2523649.2523670.

[3] K. Singh, P. Singh, and K. Kumar, "User behavior analytics-based classification of application layer HTTP-GET flood attacks," *Journal of Network and Computer Applications*, vol. 112, pp. 97–114, Jun. 2018, doi: 10.1016/J.JNCA.2018.03.030.

[4] M. Baş Seyyar, F. Ö. Çatak, and E. Gül, "Detection of attack-targeted scans from the Apache HTTP Server access logs," *Applied Computing and Informatics*, vol. 14, no. 1, pp. 28–36, Jan. 2018, doi: 10.1016/J.ACI.2017.04.002.

[5] C. You, Q. Wang, and C. Sun, "sBiLSAN: Stacked Bidirectional Self-attention LSTM Network for Anomaly Detection and Diagnosis from System Logs," *Lecture Notes in Networks and Systems*, vol. 296, pp. 777–793, 2022, doi: 10.1007/978-3-030-82199-9_52/COVER.

[6] M. Cinque, D. Cotroneo, and A. Pecchia, "Mining Dependability Properties from System Logs: What We Learned in the Last 40 Years," *Springer Series in Reliability Engineering*, pp. 221–238, 2023, doi: 10.1007/978-3-031-02063-6_12/COVER.

[7] X. D. Hoang, "Detecting Common Web Attacks Based on Machine Learning Using Web Log," *Lecture Notes in Networks and Systems*, vol. 178, pp. 311–318, 2021, doi: 10.1007/978-3-030-64719-3_35/COVER.

[8] W. Wan, X. Shi, J. Wei, J. Zhao, and C. Long, "ELSV: An Effective Anomaly Detection System from Web Access Logs," in *2021 IEEE International Performance, Computing, and Communications Conference (IPCCC)*, 2021, pp. 1–6. doi: 10.1109/IPCCC51483.2021.9679413.

[9] M. Majd, P. Najafi, S. A. Alhosseini, F. Cheng, and C. Meinel, "A Comprehensive Review of Anomaly Detection in Web Logs," in *2022 IEEE/ACM International Conference on Big Data Computing, Applications and Technologies (BDCAT)*, 2022, pp. 158–165. doi: 10.1109/BDCAT56447.2022.00027.

[10] H. Studiawan, F. Sohel, and C. Payne, "A survey on forensic investigation of operating system logs," *Digit Investig*, vol. 29, pp. 1–20, Jun. 2019, doi: 10.1016/J.DIIN.2019.02.005.

[11] D. Librado and W. Wagito, "PEMETAAN AKSES HALAMAN SITUS WEB BERBASIS LOG-ACCESS (Log-Access Based Web Site Access Mapping Page)," *JIKO (Jurnal Informatika dan Komputer)*, vol. 3, no. 1, pp. 19–26, Feb. 2018, doi: 10.26798/JIKO.V3I1.81.

[12] D. Librado, J. Raya Janti, and K. Jambe, "PEMETAAN AKSES HALAMAN SITUS WEB BERBASIS LOG-ACCESS," *Jurnal SAINTEKOM*, vol. 9, no. 2, pp. 95–106, Oct. 2019, doi: 10.33020/SAINTEKOM.V9I2.78.

[13] A. Dennis, B. Hallex Wixom, and R. M.Roth, *Systems Analysis & Design*, Fifth Edition. John Wiley & Sons, Inc., 2012.

[14] R. Trimahardhika and E. Sutinah, "Penggunaan Metode Rapid Application Development Dalam Perancangan Sistem Informasi Perpustakaan," *Jurnal Informatika*, vol. 4, no. 2, Sep. 2017, doi: 10.31294/JI.V4I2.2226.

[15] D. Murdiani and H. Hermawan, "Perbandingan Metode Waterfall dan RAD (Rapid Application Development) Pada Pengembangan Sistem Informasi," *(JurTI) Jurnal Teknologi Informasi*, vol. 6, no. 1, pp. 14–23, Jun. 2022, doi: 10.36294/JURTI.V6I1.2544.

[16] V. Y. P. Ardhana, M. Sapi'i, H. Hasbullah, and E. A. M. Sampetoding, "Web-Based Library Information System Using Rapid Application Development (RAD) Method at Qamarul Huda University," *The IJICS (International Journal of Informatics and Computer Science)*, vol. 6, no. 1, pp. 43–50, Mar. 2022, doi: 10.30865/IJICS.V6I1.4031.

[17] P. Bidang Komputer Sains dan Pendidikan Informatika, D. Aryani, and H. Dewi Ariessanti, "Application of Rapid Application Development (RAD) in Designing Tracer Study Application an Android-Based," *Jurnal Edik Informatika Penelitian Bidang Komputer Sains dan Pendidikan Informatika*, vol. 7, no. 1, pp. 111–122, Nov. 2020, doi: 10.22202/EI.2020.V7I1.4368.

[18] Apache Software Foundation, "Log Files - Apache HTTP Server Version 2.4." https://httpd.apache.org/docs/2.4/logs.html (accessed Jun. 25, 2022).

[19] Apache Software Foundation, "mod_log_config - Apache HTTP Server Version 2.4." https://httpd.apache.org/docs/current/mod/mod_log_config.html#customlog (accessed Jun. 25, 2022).

[20] D. S. Sisodia, V. Khandal, and R. Singhal, "Fast prediction of web user browsing behaviours using most interesting patterns:," *https://doi.org/10.1177/0165551516673293*, vol. 44, no. 1, pp. 74–90, Oct. 2016, doi: 10.1177/0165551516673293.

[21] D. Faraglia, "Faker Package Documentation." https://faker.readthedocs.io/en/master/ (accessed Jun. 25, 2022).

[22] P. M. Hallam-Bakerm and B. Behlendorf, "Extended Log File Format." https://www.w3.org/TR/WD-logfile.html (accessed Jun. 25, 2022).

[23] Pallets, "Welcome to Flask — Flask Documentation (2.1.x)." https://flask.palletsprojects.com/en/2.1.x/ (accessed May 26, 2022).