2nd Seminar Nasional Mahasiswa Fakultas Teknologi Informasi (SENAFTI)
21 Maret 2023 – Jakarta, Indonesia
Volume 2, Nomor 1, April 2023 - ISSN 2962-8628 (*online*)

IMPLEMENTASI RESTFUL API PADA APLIKASI MONITORING PERANGKAT JARINGAN KOMPUTER DI UNIVERSITAS BUDI LUHUR

Ahmad Sopian[1*], Sri Mulyati[2], Mohammad Syafrullah[3], Titin Fatimah[4]

[1,2,3,4] Teknik Informatika, Fakultas Teknologi Informasi, Universitas Budi Luhur, DKI Jakarta, Indonesia

Email: [1*]sopianahmad120@gmail.com, [2]sri.mulyati@budiluhur.ac.id, [3]mohammad.syafrullah@budiluhur.ac.id, [4]titin.fatimah@budiluhur.ac.id
(* : corresponding author)

Abstrak-Universitas Budi Luhur menghadapi masalah jaringan karena tingginya jumlah perangkat yang terhubung, yang membuat jaringan menjadi semakin kompleks dan sulit untuk dikelola. Selain itu, gangguan dan kerusakan jaringan sering terjadi tanpa dapat diprediksi, sehingga diperlukan sistem *monitor* jaringan untuk memudahkan administrator dalam memeriksa kondisi jaringan. Salah satu solusi yang memungkinkan adalah dengan menggunakan protokol *ICMP (Internet Control Message Protocol)* dengan program *ping* untuk memantau perangkat untuk menentukan apakah perangkat tersebut sedang *online* atau *offline*, dengan fokus pada perangkat seperti *mikrotik, modem, dan access point*. Penelitian ini bertujuan untuk mengembangkan sistem *monitoring* jaringan yang menampilkan informasi mengenai status jaringan, pengelolaan *IP*, serta memberikan pemberitahuan melalui aplikasi *Android* dengan menggunakan layanan *FCM*. Sistem ini menggunakan penjadwal untuk memastikan pemantauan secara *real-time*, dengan *administrator* menerima notifikasi secara langsung melalui aplikasi *Android*. Untuk mendukung skalabilitas dan perubahan data yang dinamis, aplikasi web dan seluler memerlukan teknologi *RESTful Web Service*. Namun, keamanan juga menjadi perhatian penting seiring dengan meningkatnya jumlah pengguna yang bertukar data sensitif. Oleh karena itu, algoritma *AES 256* dan protokol otorisasi *OAUTH 2.0* diperlukan untuk memastikan keamanan setiap transaksi data. Setelah melakukan pengujian, ditemukan bahwa sistem pemantauan latar belakang berjalan secara *real-time* dan *administrator* menerima pemberitahuan langsung melalui aplikasi *Android* jika ada perangkat yang terputus. Selain itu, sistem pemantauan dapat melakukan *ping* ke satu *host* atau *IP* dalam waktu kurang dari 5 detik, dan proses validasi mencakup token akses yang sesuai dengan protokol *OAuth 2.0*, enkripsi, dan dekripsi untuk memastikan transaksi data yang aman.

Kata Kunci: Restful API, Aplikasi Monitoring, Monitoring Perangkat Jaringan Komputer.

RESTFUL API IMPLEMENTATION ON COMPUTER NETWORK DEVICE MONITORING APPLICATION AT BUDI LUHUR UNIVERSITY

Abstract- *Budi Luhur University faces network problems due to the high number of connected devices, which makes the network more complex and difficult to manage. In addition, network disruptions and damage often occur unpredictably, so a network monitoring system is needed to make it easier for administrators to check network conditions. One possible solution is to use the ICMP (Internet Control Message Protocol) protocol with the ping program to monitor devices to determine whether they are online or offline, focusing on devices such as proxy, modems, and access points. This research aims to develop a network monitoring system that displays information about network status, IP management, and provides notifications through Android applications using FCM services. The system uses a scheduler to ensure real-time monitoring, with administrators receiving notifications directly through the Android app. To support scalability and dynamic data changes, web and mobile applications require RESTful Web Service technology. However, security is also an important concern as an increasing number of users exchange sensitive data. Therefore, AES 256 algorithm and OAUTH 2.0 authorization protocol are required to ensure the security of each data transaction. After conducting the test, it was found that the background monitoring system runs in real-time and the administrator receives immediate notification through the Android app if any device is disconnected. In addition, the monitoring system can ping a single host or IP in less than 5 seconds, and the validation process includes access tokens corresponding to the OAuth 2.0 protocol, encryption, and decryption to ensure secure data transactions.*

Keywords*: Restful API, Monitoring Application, Monitoring Computer Network Devices.*

1. PENDAHULUAN

Perkembangan teknologi saat ini berkembang pesat, dengan adanya penemuan teknologi baru yang bertujuan untuk membantu manusia dalam kegiatan sehari-hari [1], Sebagian besar institusi dan penduduk di suatu daerah membutuhkan fasilitas telekomunikasi, seperti akses internet atau akses jaringan lokal, untuk memproses data [2].

Dengan adanya teknologi, semakin banyak jumlah perangkat jaringan dan kemajuan teknologi, semakin tinggi pula kemungkinan terjadinya gangguan jaringan [3], dimana konektivitas jaringan rentan terhadap

kerusakan dan ketidaksempurnaan fisik dan non-fisik, sehingga pemantauan jaringan secara terus menerus diperlukan untuk memastikan ketersediaan jaringan [4], Memanfaatkan kemajuan teknologi saat ini dalam jaringan komputer, solusi yang paling tepat adalah menerapkan pemantauan jaringan secara terus menerus tanpa pengawasan langsung dan manajemen informasi jaringan [5], kemudian untuk mendukung skalabilitas dan perubahan data yang dinamis, maka memerlukan teknologi *RESTful Web Service* [6]. Namun, keamanan juga menjadi perhatian penting seiring dengan meningkatnya jumlah pengguna yang bertukar data sensitif. Oleh karena itu, algoritma AES 256 dan protokol otorisasi OAUTH 2.0 diperlukan untuk memastikan keamanan setiap transaksi data [7]. Dengan adanya *web service*, memungkinkan aplikasi yang berbeda di berbagai *platform* dan bahasa pemrograman untuk berkomunikasi dan saling bertukar data melalui jaringan internet dengan menggunakan protokol standar seperti *XML* atau *JSON*.

Universitas Budi Luhur menghadapi masalah jaringan karena tingginya jumlah perangkat yang terhubung, yang membuat jaringan menjadi semakin kompleks dan sulit untuk dikelola maka pemantauan jaringan menjadi sangat penting. Dari permasalahan tersebut maka perlu dibangun sistem untuk mempermudah tugas administrasi jaringan, dapat digunakan sistem monitoring jaringan yang menggunakan protokol *ICMP (Internet Control Message Protocol)* untuk memeriksa kondisi jaringan [8], serta teknologi layanan web (*web service*) yang memungkinkan aplikasi dapat diakses melalui berbagai *platform*. Pada penelitian ini menggunakan teknologi *Firebase Cloud Messaging (FCM)* yang berfungsi sebagai *web service* penyedia layanan *push notification* atau pesan notifikasi yang berisikan aktifitas-aktifitas yang terjadi pada router mikrotik [9]. FCM ini juga berperan untuk menjembatani antara *server* dengan perangkat *Android* agar dapat terjadi aktifitas pengiriman pesan *(push notification)*. Dari penjabaran permasalahan di atas, aplikasi dapat menjadi alternatif untuk membantu kesigapan kinerja teknisi/administrator jaringan untuk memberikan pelayanan terbaik saat terjadi permasalahan jaringan internet di lingkungan Universitas Budi Luhur.

2. METODE PENELITIAN

2.1 Pengumpulan Data

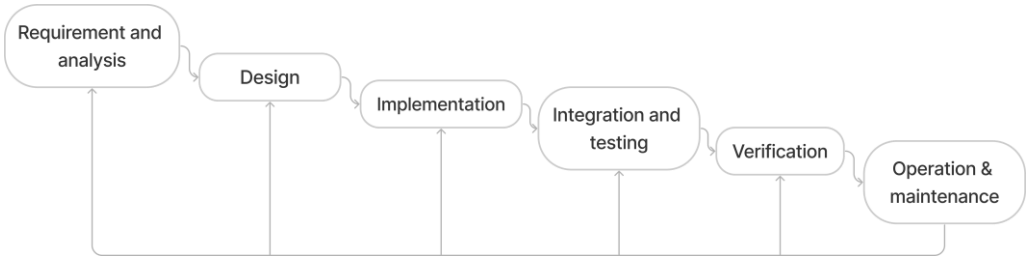
Untuk menyusun penelitian ini, dibutuhkan data dan informasi yang lengkap sebagai referensi untuk mendukung uraian materi dan pembahasan yang sesuai dengan tujuan penelitian. Metode yang digunakan meliputi observasi dan wawancara.

2.2 Studi Literatur

Tahap ini melibatkan penelitian yang terkait dengan topik diskusi, yaitu protokol *ICMP* sebagai program untuk memonitor perangkat dan layanan web yang dilindungi oleh *OAuth 2.0* dan dienkripsi dengan algoritma *AES 256* serta menggunakan layanan *FCM* untuk dapat mengirim pemberitahuan ke aplikasi *android*. Penelitian ini mencakup berbagai sumber referensi seperti buku teks, jurnal, dan karya ilmiah.

2.3 Tahapan Penelitian

Pedoman penelitian yang digunakan bertujuan untuk memastikan bahwa sistem yang dihasilkan tidak menyimpang dari tujuan yang telah ditetapkan sebelumnya, sehingga penelitian dapat dilakukan dengan langkah-langkah yang terarah dan terstruktur. Pada gambar 1 menggambarkan tahapan penerapan metodologi penelitian yang dilakukan dalam penelitian ini.



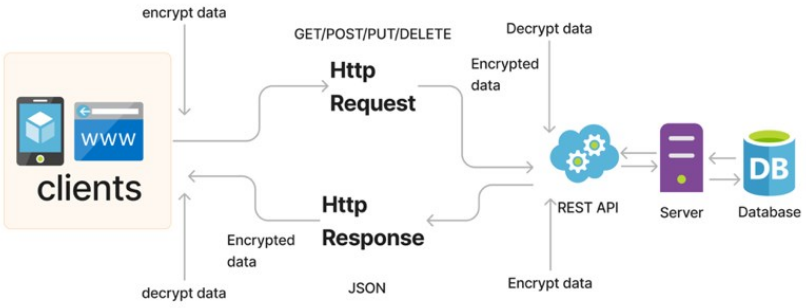
Gambar 1. Tahap Penelitian

2.4 Penerapan Metode

Penelitian ini membangun sistem dengan menggunakan metode *REST*, algoritma *AES 256* sebagai metode keamanan pertukaran data, dan protokol *OAUTH 2.0* sebagai metode otorisasi dan layanan *FCM* untuk mengirim notifikasi ke perangkat *android*. Dari metode tersebut, dirancang beberapa tahapan untuk membangun sistem.

2.4.1 Arsitektur Penerapan *Web Service*

Arsitektur layanan web dibuat untuk menggambarkan layanan yang diperlukan dalam sistem yang diusulkan, dan memberikan gambaran tentang alur komunikasi data antara layanan satu dengan layanan lainnya. Rancangan arsitektur layanan web dapat dilihat pada Gambar 2.



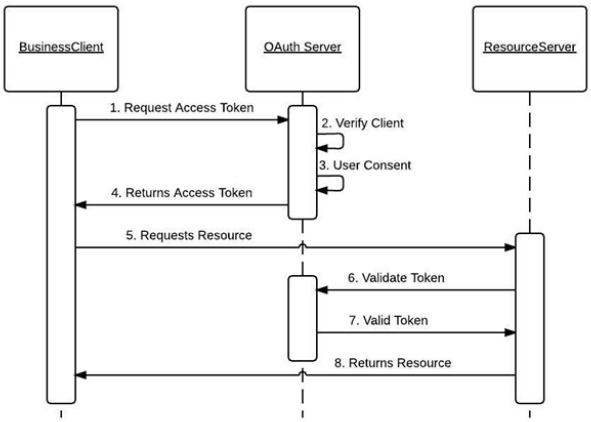
Gambar 2. Arsitektur *Web Service*

Pada gambar 2 menampilkan arsitektur yang menggunakan prinsip-prinsip arsitektur *REST (Representational State Transfer)*, yang memprioritaskan penggunaan protokol *HTTP* untuk pertukaran data dan menyederhanakan arsitektur layanan web, serta melakukan enkripsi data pada layanan web, agar tidak dapat dibaca oleh pihak yang tidak berwenang. Berikut adalah Langkah-langkah mengenai cara kerja REST web service dan proses dari enkripsi dan dekripsi yang terdapat pada sistem yang dibangun.

a. *Client* mengirimkan permintaan (*request*) ke *server* melalui *HTTP*. Permintaan tersebut dapat berupa *GET*, *POST*, *PUT*, atau *DELETE*, tergantung pada operasi yang diinginkan oleh *client* dan berisi data yang akan dikirimkan ke *server*. Sebelum dikirimkan ke *server*, data tersebut dienkripsi menggunakan algoritma *AES 256* agar terjaga kerahasiaannya. Hasil enkripsi berupa *ciphertext* kemudian dikirimkan ke *server*.
b. *Server* menerima permintaan (*request*) dan data yang dikirimkan dalam bentuk *ciphertext*. Kemudian, data tersebut didekripsi menggunakan algoritma *AES-256* sehingga diperoleh data *plaintext*. Selanjutnya, *server* akan memproses permintaan tersebut sesuai dengan yang telah ditentukan oleh layanan *web service*. Proses ini dapat berupa pengambilan data dari basis data (*database*), penyimpanan data ke dalam basis data, atau operasi lainnya.
c. *Server* mengembalikan *response* ke *client* melalui *HTTP*. *Response* tersebut dapat berupa data yang telah dienkripsi oleh *server* sebelumnya, pesan error atau pesan lainnya dalam bentuk *JSON* sesuai dengan yang ditentukan oleh *web service.*
d. *Client* menerima *response* dari *server* dan memproses *response* tersebut sesuai dengan kebutuhan. Proses ini dapat berupa dekripsi data yang telah dienkripsi oleh server sebelumnya menggunakan algoritma enkripsi yang sama.

2.4.2 Otorisasi dengan OAUTH 2.0

Oauth merupakan suatu protokol terbuka yang mengizinkan otorisasi secara aman dengan metode yang sederhana dan standar dari aplikasi web, mobile, dan desktop [10]. Sistem ini terdiri dari dua sisi yaitu *server* dan *client* yang terdiri dari dua *platform* berbeda, yaitu *mobile android* dan web. Interaksi antar sisi dilakukan dengan menggunakan metode *REST* melalui protokol *HTTP*. Pada sisi *client*, akan dilakukan permintaan *HTTP* ke *server*. Setelah itu, *server* akan melakukan otorisasi menggunakan protokol *OAUTH 2.0*. Jika otorisasi berhasil dilakukan, maka *server* akan memberikan respon dalam bentuk data dengan format *JSON*. Data respon tersebut nantinya akan digunakan oleh *client* sesuai dengan kebutuhan. Berikut adalah gambaran mengenai sistem otorisasi yang menggunakan *OAUTH 2.0* pada sistem ini.

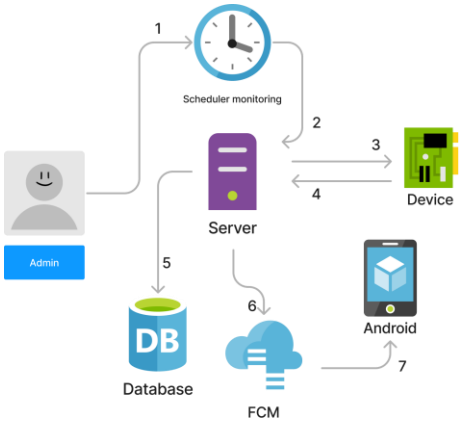


Gambar 3. *OAUTH 2.0 Use Case Flow*

Pada gambar 3 menggambarkan *oauth flow* yang merupakan protokol yang digunakan untuk mengamankan dan melindungi akses ke sumber daya web. Alur ini memungkinkan pengguna untuk memberikan izin akses kepada aplikasi pihak ketiga tanpa harus memberikan informasi login mereka secara langsung kepada aplikasi tersebut dan berikut merupakan penjelasan dari *oauth flow* pada aplikasi.

a. *Client* mengirim *request* ke *server oauth* untuk mendapatkan akses token. *Request* ini harus mencakup informasi tentang aplikasi dan pengguna yang ingin mengakses data. Contoh dari informasi tersebut seperti *client id*, *client secret*, dll.
b. *Server oauth* akan memverifikasi *request* tersebut dan jika valid, akan mengirimkan akses token kembali ke *client.*
c. *Client* menggunakan akses token yang diterimanya untuk mengirim *request* ke *server resource* yang ingin diakses. *client* bisa menggunakan akses token tersebut untuk melakukan *request* dengan menambahkan *header Authorization* : *Bearer* (akses token yang didapat).
d. *Server resource* akan memverifikasi akses token yang di terimanya dan jika valid, akan mengirimkan data yang diminta kembali ke *client.*
e. *Client* menerima data yang diminta dan dapat menampilkan atau menggunakannya sesuai kebutuhan.

2.4.3 Arsitektur Penerapan Sistem Monitoring



Gambar 4. Arsitektur Sistem Monitoring

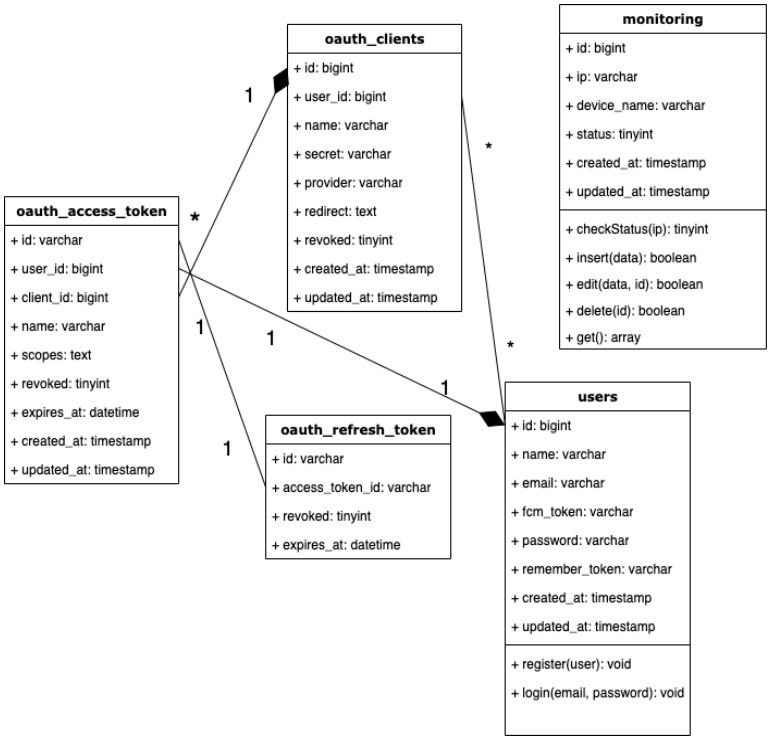
Pada gambar 4 terlihat arsitektur sistem *monitoring*, dan berikut adalah penjelasannya:

a. Admin menjalankan *command scheduler* untuk memulai proses *monitoring.*
b. *Server* melakukan proses *monitoring.*
a. *Server* mengirimkan pesan *echo request* ke alamat *IP* perangkat yang dituju untuk mengetahui apakah perangkat tersebut dalam keadaan terhubung atau tidak terhubung.
b. *Server* menerima respons dari perangkat tersebut.
c. Setelah menerima respons dari perangkat, *server* akan meng*update* status perangkat ke dalam *database*.

d. *Server* mengirimkan pesan berisi status perangkat dengan alamat *IP* tersebut ke pengguna perangkat *Android* melalui *FCM.*
e. Kemudian, *FCM* mengirimkan pesan pemberitahuan (*push notification*) ke perangkat *Android.*

2.5 Rancangan Basis Data

Rancangan basis data pada sistem monitoring ini digambarkan dengan menggunakan *LRS (Logical Record Structure* seperti pada gambar 5.



Gambar 5. Rancangan Basis Data

Berdasarkan pada gambar 5, *LRS* di atas akan menghasilkan tabel basis data *monitoring, oauth_clients, oauth_access_token, oauth_refresh_token, users.*

3. HASIL DAN PEMBAHASAN

3.1 Lingkungan Percobaan (Spesifikasi *Hardware & Software*)

3.1.1 Spesifikasi *Hardware*

Berikut merupakan spesifikasi *hardware* yang digunakan:
a. Macbook pro-2021, Apple M1 Pro, 16gb, 512gb
b. Modem ZTE F606
c. Realme 9A, Helio G25, 3gb, 32gb

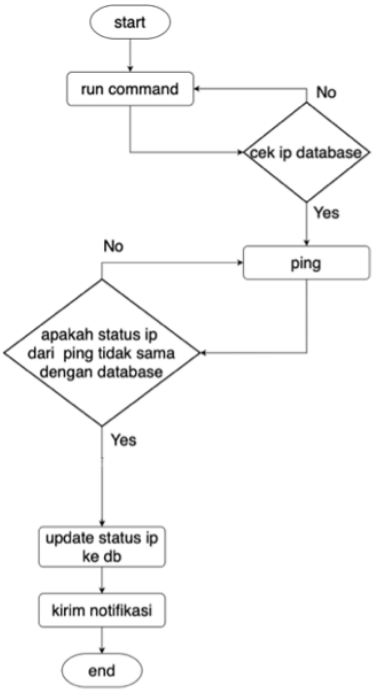
3.1.2 Spesifikasi *Software*

Berikut merupakan spesifikasi *software* yang digunakan:
a. Sistem operasi macOs ventura 13.1
b. Google chrome
c. Xampp control panel
d. Postman
e. PhpStorm
f. Android studio
g. DataGrip
h. Figma

3.2 *Flowchart*

3.2.1 *Flowchart Monitoring*

Pada gambar 6 merupakan alur proses *monitoring* pada aplikasi.

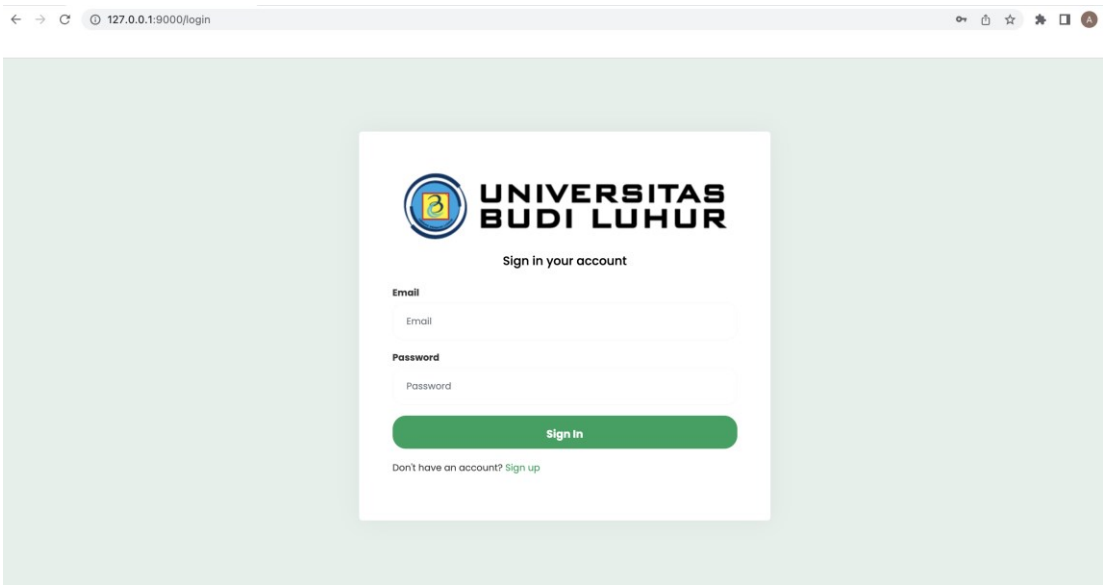


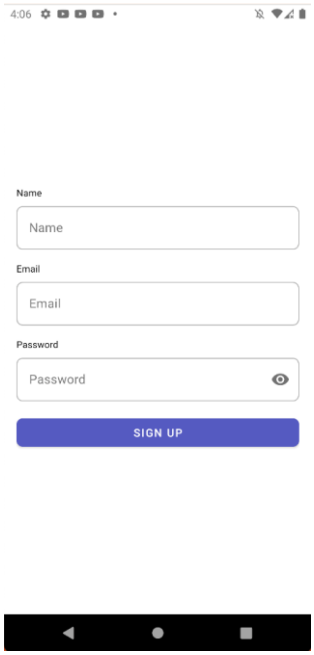
Gambar 6. *Flowchart Monitoring*

3.3 Tampilan Layar

3.3.1 Tampilan Layar Login

Pada gambar 7 merupakan tampilan layar *login* yang digunakan untuk memvalidasi pengguna sebelum masuk ke halaman utama.

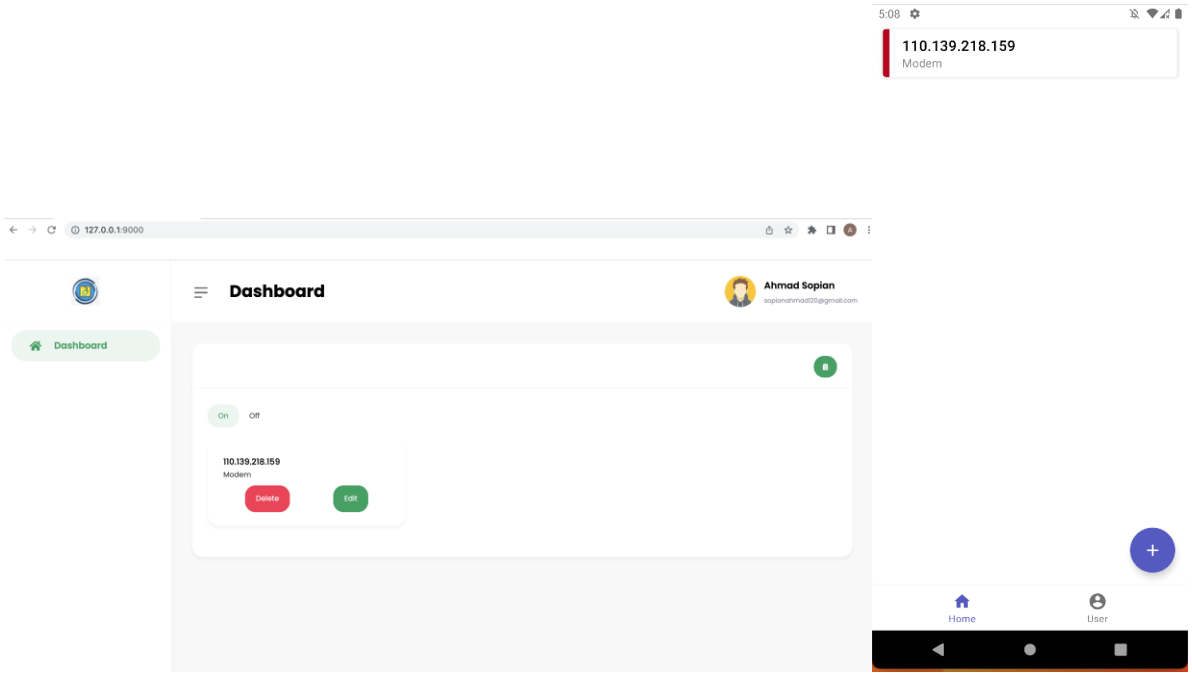




Gambar 7. Tampilan Layar *Login* Pada *Platform* web dan *mobile*

3.3.2 Tampilan Layar *Dashboard*

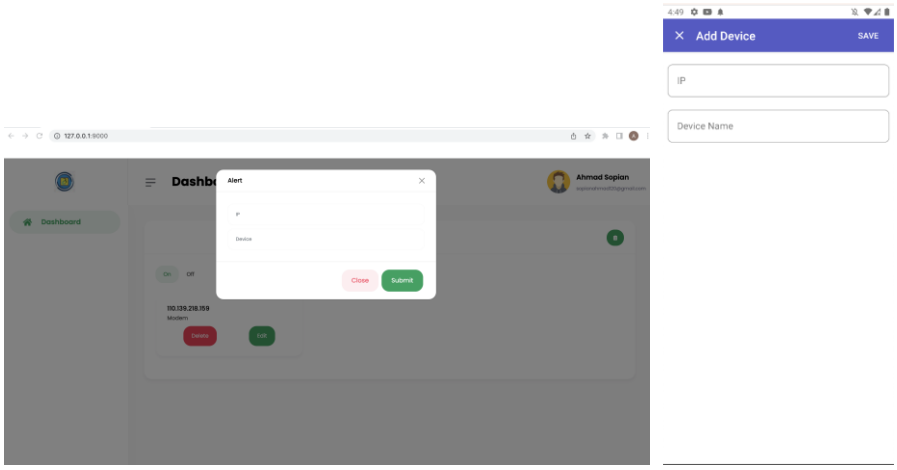
Pada gambar 8 merupakan tampilan layar dashboard yang digunakan untuk memantau perangkat dan mengelola data perangkat.



Gambar 8. Tampilan Layar *Dashboard* Pada *Platform* web dan *mobile*

3.3.3 Tampilan Layar Tambah Data Perangkat

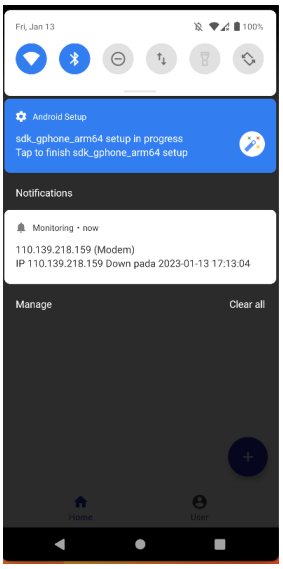
Pada gambar 9 merupakan tampilan layar tambah data perangkat yang digunakan untuk menambahkan data perangkat seperti ip dan nama perangkat.



Gambar 9. Tampilan Layar Tambah Data Perangkat Pada *Platform* web dan *mobile*

3.3.4 Tampilan Layar Notifikasi

Pada gambar 10 merupakan tampilan layar notifikasi ketika perangkat *down.*



Gambar 10. Tampilan Layar Notifikasi Pada *Platform mobile*

3.4 Pengujian

Setelah selesai membuat aplikasi, dilakukan pengujian untuk mengevaluasi kinerja dan fungsi aplikasi dengan menguji *test case* yang telah ditentukan pada saat perancangan. Data yang dihasilkan dari pengujian digunakan untuk memastikan kebenaran algoritma dan proses sistem aplikasi, serta untuk memeriksa kesalahan dan kesesuaian antarmuka dengan kebutuhan pengguna. Metode pengujian yang digunakan adalah *Black Box Testing*, yaitu suatu metode pengujian yang menggunakan struktur rancangan prosedural untuk menguji dan memahami bagaimana perangkat lunak bekerja secara *internal*.

3.4.1 Pengujian Layanan *Web Service*

Pada tabel 1 merupakan pengujian layanan *web service* yang dilakukan menggunakan tools postman yang bertujuan untuk memeriksa fitur-fitur seperti menampilkan token setelah login, menampilkan data *list device*, dan lain-lain. Untuk memeriksa fitur tersebut dilakukan pengujian fungsional dengan cara mengirimkan permintaan ke layanan *web service* dan memeriksa apakah fitur tersebut bekerja dengan baik atau tidak, kemudian dilakukan pengujian performa pada layanan *web service* untuk memastikan bahwa layanan tersebut dapat menangani beban pengguna dengan baik. Pengujian performa dilakukan dengan mengirimkan permintaan ke layanan *web service* secara bersamaan oleh beberapa pengguna untuk menguji kemampuan layanan dalam menangani beban yang besar.

Tabel 1. Pengujian Layanan *Web Service*

No	Skenario Pengujian	*Test Case*	Hasil
1	Menampilkan token setelah *login*	*method*: *POST* *endpoint*: *oauth/token* *parameter*: *client_id, client_secret, grant_type, scope, username, password*	Berhasil
2	Menampilkan data *list device*	*method: GET* *endpoint: device* *header: Authorization: Bearer token*	Berhasil
3	Mengubah data *device*	*method: PUT* *input: ip, device_name, id* *endpoint : device* *header : Authorization : Bearer token*	Berhasil
4	Menambah data *device*	*method: POST* *input: ip, device_name* *endpoint : device* *header : Authorization : Bearer token*	Berhasil
5	Menghapus data *device*	*method: DELETE* *input: id* *endpoint : device* *header : Authorization : Bearer token*	Berhasil
6	Registrasi pengguna	*method: POST* *input: name, email, password* *endpoint : device* *header : Authorization : Bearer token*	Berhasil
7	Memastikan *access* token sudah *revoked* ketika *logout*	*method: POST* *input: is_mobile* *endpoint : auth/logout* *header : Authorization : Bearer token*	Berhasil
8	menambahkan *fcm* token ke *server*	*method: POST* *input: fcm_token, email* *endpoint : notification/register* *header : Authorization : Bearer token*	Berhasil

3.4.2 Pengujian Enkripsi dan Dekripsi

Pada tabel 2 merupakan pengujian yang dilakukan untuk menguji keamanan pada layanan *web service* untuk memastikan bahwa layanan tersebut aman dari serangan *hacker* dan tidak ada celah keamanan yang dapat dieksploitasi.

Tabel 2. Pengujian Enkripsi dan Dekripsi

No	Skenario Pengujian	*Test Case*	Hasil
1	Mengenkripsi data *input* yang akan dikirim ke *server*	Mengenkripsi data *input* menggunakan *function aesEncrypt* untuk enkripsi	Berhasil
2	Mendekripsi data *input* yang telah diterima oleh *server*	Mendekripsi data *input* menggunakan *function aesDecrypt* untuk dekripsi	Berhasil
3	Mengenkripsi data *response* dari *server* yang akan dikembalikan ke *client*	Mengenkripsi data *response* menggunakan *function aesEncrypt* untuk enkripsi	Berhasil
4	Mendekripsi data *response* yang telah diterima oleh *client*	Mendekripsi data *response* menggunakan *function aesDecrypt* untuk dekripsi	Berhasil

3.4.3 Pengujian Monitoring Perangkat.

Pada tabel 3 merupakan pengujian terhadap layanan FCM dengan cara mengirimkan permintaan notifikasi melalui layanan FCM ke perangkat Android yang akan diuji. Tujuan dari pengujian ini adalah untuk memastikan bahwa perangkat Android dapat menerima notifikasi dari layanan FCM secara real-time. Selanjutnya dilakukan pengujian terhadap fitur monitoring pada aplikasi monitoring yang terpasang pada perangkat Android. Fitur tersebut meliputi pemantauan kondisi perangkat apakah dalam keadaan hidup atau mati dan lain-lain. Pengujian

dilakukan dengan cara memeriksa apakah informasi yang ditampilkan pada aplikasi monitoring akurat dan up-to-date. Terakhir, dilakukan pengujian performa pada aplikasi monitoring untuk memastikan bahwa aplikasi tersebut dapat menangani beban monitoring dengan baik dan tidak terjadi delay dalam memberikan notifikasi. Pengujian performa dilakukan dengan mengirimkan beban monitoring yang besar pada aplikasi dan memeriksa apakah aplikasi tersebut tetap berjalan dengan baik dan memberikan notifikasi dengan waktu respons yang cepat. Dengan melakukan pengujian-pengujian tersebut, diharapkan aplikasi monitoring perangkat jaringan berbasis Android yang menggunakan layanan FCM dapat berfungsi dengan baik dan dapat memberikan notifikasi secara real-time jika terjadi masalah pada jaringan.

Tabel 3. Pengujian Enkripsi dan Dekripsi

No	Skenario Pengujian	*Test Case*	Hasil
1	*Monitoring device* dan memastikan notifikasi muncul pada aplikasi *mobile android*	Menjalankan *command scheduler* untuk melakukan *monitoring device* dan pastikan notifikasi muncul berisi pesan status dari *ip* perangkat	Berhasil

4. KESIMPULAN

Setelah melakukan perancangan, implementasi, dan serangkaian uji coba pada aplikasi berbasis web dan *mobile* yang bertujuan untuk membantu proses *monitoring* perangkat di jaringan komputer pada Universitas Budi Luhur, maka dapat disimpulkan bahwa aplikasi yang telah dibangun dapat me*monitoring node/ip* dengan hasil 100% berhasil. Keamanan pada sistem ini dijamin melalui penggunaan otorisasi *OAUTH 2.0* dan algoritma *AES-256*. Uji coba *blackbox testing* dilakukan berdasarkan layanan *web service*, dan hasilnya sesuai dengan yang diharapkan. Semua layanan *web service* yang telah dibuat dapat diimplementasikan dengan baik pada platform web dan *mobile*. Setiap kali terjadi pemutusan koneksi pada suatu perangkat, sistem akan segera memberitahukan pengguna melalui notifikasi. Untuk dapat mengirim notifikasi melalui layanan *Firebase Messaging Service (FCM)*, penting bagi komputer *server* dan aplikasi *mobile* untuk selalu terhubung dengan internet. Proses pengiriman notifikasi hanya akan berhasil dilakukan jika kedua sisi tersebut terhubung dengan internet.

Pada penelitian selanjutnya sebaiknya dilakukan pengembangan sistem monitoring dengan menggunakan SNMP *(Simple Network Management Protocol)*.

DAFTAR PUSTAKA

[1] S. M. Lim, M. Y. Tuga, dan E. A. Lisangan, "Prototype Aplikasi Pengawasan Masyarakat Menggunakan Smart Camera Dalam Mendeteksi COVID-19," J. Fokus Elektroda Energi List. Telekomun. Komputer, Elektron. dan Kendali., vol. 5, pp. 15-19, 2020.

[2] P. R. Utami, "Analisis Perbandingan Quality of Service Jaringan Internet Berbasis Wireless Pada Layanan Internet Service Provider (Isp) Indihome Dan First Media," Jurnal Ilmiah Teknologi dan Rekayasa., vol. 25, pp. 125-137, Jan. 2020.

[3] R. M. Febriana, "Implementasi Sistem Monitoring Menggunakan Prometheus Dan Grafana," Semin. Nas. Telekomun dan Inform., vol. 13, pp. 164-169, 2020.

[4] P. K. Prayogi, M. Orisa, dan F. X. Ariwibisono, "Rancang Bangun Sistem Monitoring Jaringan Access Point Menggunakan Simple Network Management Protocol (SNMP) Berbasis Web," JATI (Jurnal Mahasiswa Teknik Informatika)., vol. 4, pp. 192-197, 2020.

[5] M. Syani, dan B. Saputro, "Implementasi Remote Monitoring Pada Virtual Private Server Berbasis Telegram Bot Api (Studi Kasus Politeknik Tedc Bandung)," Jurnal SISKOM-KB (Sistem Komputer dan Kecerdasan Buatan)., vol. 4, pp. 94-111, 2021.

[6] T. Hartanto dan Painem, "Implementasi Web Service Berbasis REST Menggunakan Algoritma AES 128 dan Affine Cipher Fitur Bluacademic Aplikasi Blucampus" SKANIKA., vol. 1, pp. 1130-1136, Jul. 2018.

[7] I. Kusuma, A. Susanto, dan I. U. W. Mulyono, "Implementasi RESTful Web Services dengan Otorisasi OAUTH 2.0 pada Sistem Pembayaran Parkir," Jurnal SIMETRIS., vol. 10, pp. 391-404, Apr. 2019.

[8] M. F. Qomarudin, dan A. Amrullah, "Sistem Monitoring Jaringan Realtime Berbasis Internet Control Message Protocol" JINTECH: Journal of Information Technology., vol. 3, pp. 67-80, Agu. 2022.

[9] D. B. Prasetyo,R. I. Mifftah, and R. I. Perwira, "Implementasi Network Notification System Dengan Menggunakan Teknologi Firebase Cloud Messaging (FCM) Berbasis Android," TELEMATIKA., vol. 16, pp. 62-72, Okt. 2019.

[10] (2022) OAuth Community Site. [Online]. Available: https://oauth.net.