IMPLEMENTATIONTASI SISTEM MONITORING PERANGKAT KLIEN PADA *YAMAHA SMART GATEWAY SGX808*

SKRIPSI

KEMINATAN TEKNIK KOMPUTER

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Helmi Nizar
NIM: 125150301111027

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2016

PENGESAHAN

IMPLEMENTASI SISTEM MONITORING PERANGKAT KLIEN
PADA *YAMAHA SMART GATEWAY SGX808*

SKRIPSI

KEMINATAN TEKNIK KOMPUTER

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Helmi Nizar
NIM: 125150301111027

Skripsi ini telah diuji dan dinyatakan lulus pada
11 Agustus 2016
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Sabriansyah Rizqika Akbar, S.T, M.Eng.	Adharul Muttaqin, S.T, M.T.
NIP: 19820809 201212 1 004	NIP: 19820809 201212 1 004

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T., M.T. Ph.D.
NIP: 197205182003121001

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 didalam 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, 29 Juli 2016

Helmi Nizar

NIM: 125150301111027

KATA PENGANTAR

Puji dan syukur penulis panjatkan kepada Tuhan Yang Maha Esa atas selesainya skripsi yang berjudul "Implementasi Sistem Monitoring Perangkat Klien Pada *Yamaha Smart Gateway SGX808*".

Atas bantuan moral dan materiil yang diberikan pada penyusunan skripsi ini, maka penulis mengucapkan banyak terima kasih kepada:

1. Bapak Sabriansyah Rizqika Akbar, S.T, M.Eng. dan Bapak Adharul Muttaqin, S.T, M.T. selaku dosen pembimbing skripsi yang telah dengan sabar membimbing dan mengarahkan penulis sehingga dapat menyelesaikan skripsi ini dengan baik.
2. Bapak Issa Arwani, S.Kom., M.Sc selaku ketua Program Studi Teknik Informatika.
3. Ayahanda dan Ibunda dan seluruh keluarga besar atas segala nasehat, kasih sayang, perhatian dan kesabarannya memberikan semangat kepada penulis, serta senantiasa tiada hentinya memberikan doa dan semangat demi terselesaikannya skripsi ini.
4. Bapak Achmad Basuki, ST., MMG., Ph.D., yang telah membantu untuk mendapatkan kesempatan mengikuti program *internship EBA*.
5. Profesor Jun Murai, Ph.D. selaku Direktur dari *EBA Consortium*.
6. Profesor Achmad Husni Thamrin, Ph.D., selaku *Project Leader* dan Profesor Keiko Okawa, Ph.D., selaku *Course Leader* dari program *internship EBA*, yang telah membimbing selama program *internship* berlangsung.
7. Hidetake Ogino, selaku *supervisor* dari *Network Products Group, Sound & Networking Department*, *Yamaha*, yang telah membimbing selama program *internship EBA* berlangsung di *Yamaha*.
8. Seluruh civitas akademika Teknik Informatika keminatan Teknik Komputer Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penulis menempuh studi di Teknik Informatika keminatan Teknik Komputer Universitas Brawijaya dan selama penyelesaian skripsi ini.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih terdapat 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, 29 Juli 2016

Penulis

Helminizar94@gmail.com

ABSTRAK

Internet of Things (IoT) secara umum merujuk kepada sebuah skenario agar kemampuan konektifitas dan komputerisasi yang dimiliki sebuah jaringan komputer dapat menjangkau objek-objek, sensor-sensor, dan benda-benda yang digunakan sehari-hari yang tidak dikategorikan sebagai komputer, dan memungkinkan perangkat-perangkat ini untuk menghasilkan, menggunakan, serta saling bertukar data dengan sesedikit mungkin campur tangan dari manusia.

Salah satu penerapan dari *IoT* yaitu pada bidang ekonomi, misalnya pada sebuah *convenience store* yang didalamnya terdapat berbagai macam perangkat, misalnya komputer, kamera, mesin kasir, *scanner*, *printer*, *smartphone*, dan lain sebagainya. Pada umumnya, setiap perangkat yang ada tersebut memiliki sistem mereka masing-masing dan tidak terhubung satu sama lain. Sehingga akan merepotkan apabila harus mengecek tiap perangkat untuk mengetahui apakah perangkat tersebut bekerja atau tidak. Untuk itulah perlu dibuat sebuah sistem monitoring seluruh perangkat yang ada pada tempat tersebut untuk memudahkan dalam pengawasan serta perawatannya.

Sistem monitoring perangkat sendiri dapat dilakukan melalui dua jenis aplikasi, yaitu aplikasi *native* (terpasang pada sebuah perangkat) dan aplikasi berbasis *web*. Untuk aplikasi berbasis *web* memiliki kelebihan tersendiri, yaitu menghemat memori karena tidak ada aplikasi yang harus dipasang. Selain itu aplikasi berbasis *web* juga bersifat *open platform*, sehingga dapat dibuka oleh semua perangkat yang memiliki *web browser* dan terhubung ke *internet*. Maka dari itu dalam penelitian ini difokuskan ke aplikasi monitoring berbasis *web*.

Pada penelitian ini menggunakan *Yamaha Smart Gateway SGX808* sebagai server. Protokol yang digunakan untuk melakukan monitoring adalah *ICMP*. Pada aplikasi yang dibuat pada penelitian ini dapat melakukan monitoring perangkat yang sudah terdaftar, menambahkan daftar perangkat secara manual maupun dari tabel *ARP*, konfigurasi *ping interval* dan *monitoring interval*, serta menyimpan data daftar perangkat ke *database* dan memuat data perangkat yang sudah tersimpan dalam *database*.

Kata kunci: *IoT*, monitoring, *server*, aplikasi *web, Yamaha Smart Gateway SGX808*

ABSTRACT

Internet of Things (IoT) generally refers to a scenario that network connectivity and computing capability can be extended to objects, sensors, and everyday items not considered as the computer, allowing this device to generate, to exchange and use data with minimal human intervention.

One of the implementations of IoT is an application for economics, for example at a convenience store which there are a lot of devices, such as personal computer, camera, POS, scanner, printer, smartphone, and so on. Usually, each of the devices has their own system and it is not connected each other. So it will be an inconvenience to check each device whether the device is working or not. Therefore, it will be needed to develop a monitoring system for each device in that place to simplify to monitor and maintenance the device.

Monitoring device system itself can be done by two kinds of application, that is a native application (installed on the monitoring device) and web-based application. For web-based application itself has some advantage, such as for saving memory because there will be no application that should be installed. Furthermore, web-based application is an open platform, so it can be accessed by any device that has web browser and connected to the internet. Therefore, in this research will be focused on the web-based monitoring application.

In this research is using Yamaha Smart Gateway SGX808 as the server. The protocol that is used in this research is ICMP. At the application that developed in this research be able to monitoring for the registered device, adding the entry device manually or from the ARP table, configuring the ping interval and monitoring interval, saving device entry to the database and restoring device entry from the database.

Keywords: IoT, monitoring, server, web application, Yamaha Smart Gateway SGX808

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar belakang

Internet of Things (IoT) secara umum merujuk kepada sebuah skenario agar kemampuan konektifitas dan komputerisasi yang dimiliki sebuah jaringan komputer dapat menjangkau objek-objek, sensor-sensor, dan benda-benda yang digunakan sehari-hari serta tidak dikategorikan sebagai komputer, dan memungkinkan perangkat-perangkat ini untuk menghasilkan, menggunakan, serta saling bertukar data dengan sesedikit mungkin campur tangan dari manusia (Rose, Eldridge, & Chapin, 2015). *IoT* sendiri bertujuan untuk mempermudah manusia dalam melakukan sebuah pekerjaan. Dengan membuat sistem dengan *IoT* maka dapat tujuan dibuatnya sistem akan dapat diselesaikan dengan hanya sedikit atau bahkan tanpa campur tangan manusia. Selain itu dengan kemajuan teknologi saat ini, sangat banyak metode untuk mengakses sistem dengan *IoT* sehingga mempermudah dalam penggunaan sistem tersebut, misalnya dari komputer, laptop, bahkan *smartphone*. Hal inilah yang membuat semakin banyak riset tentang *IoT*.

Penerapan *IoT* sendiri saat ini sudah banyak, namun dalam riset sebagian besar masih terfokus ada pada bidang medis dan lingkungan. Selain itu, sebenarnya masih banyak sektor riset yang bisa dilakukan pemanfaatan *IoT*, salah satunya sektor ekonomi. Karena semakin berkembangnya ekonomi maka tuntutan kebutuhannya pun semakin banyak, termasuk kebutuhan teknologi penunjang termasuk *IoT*. Salah satu objek dalam bidang ekonomi yang dapat dilakukan penelitian *IoT* adalah pada *convenience store*.

Dalam sebuah *convenience store* yang di dalamnya terdapat berbagai macam perangkat, misalnya computer, kamera, mesin kasir, *scanner*, *printer*, *smartphone*, dan lain sebagainya. Pada umumnya, setiap perangkat yang ada tersebut memiliki sistem mereka masing-masing dan tidak terhubung satu sama lain. Sehingga akan merepotkan apabila harus mengecek tiap perangkat untuk mengetahui apakah perangkat tersebut bekerja atau tidak. Untuk itulah perlu dibuat sebuah sistem monitoring seluruh perangkat yang ada pada tempat tersebut untuk memudahkan dalam pengawasan serta perawatannya.

Sistem monitoring perangkat sendiri dapat dilakukan melalui dua jenis aplikasi, yaitu aplikasi *native* (terpasang pada sebuah perangkat) dan aplikasi berbasis *web*. Untuk aplikasi berbasis *web* memiliki kelebihan tersendiri, yaitu menghemat memori karena tidak ada aplikasi yang harus dipasang. Selain itu aplikasi berbasis *web* juga bersifat *open platform*, sehingga dapat dibuka oleh semua perangkat yang memiliki *web browser* dan terhubung ke *internet*. Maka dari itu dalam penelitian ini difokuskan ke aplikasi monitoring berbasis *web*.

Mengetahui kebutuhan tersebut, *Yamaha Corporation* yang salah satu divisinya, *Yamaha Sound & Networking Division*, bergerak di bidang jaringan komputer berusaha memenuhi kebutuhan tersebut. *Yamaha Sound & Networking* mempunyai sebuah produk dengan nama *Yamaha Smart Gateway SGX808*, yang dapat digunakan sebagai solusi untuk permasalahan seperti yang sudah dijelaskan di atas. Hal ini karena selain bertugas sebagai *router*, perangkat ini juga dapat bertugas sebagai *server* yang dapat menyimpan data ke *database* maupun media penyimpanan lain yang terhubung ke perangkat ini. Selain itu pada perangkat ini juga dapat dipasang aplikasi pihak ketiga, sehingga sangat cocok digunakan untuk menyelesaikan masalah pada kasus ini.

Untuk itulah, dengan bekerja sama dengan *Yamaha Sound & Networking Division*, dilakukan penelitian untuk membuat sebuah aplikasi monitoring yang akan dipasang ke dalam perangkat *Yamaha Smart Gateway SGX808*, sehingga akan bisa digunakan untuk menyelesaikan permasalahan seperti yang sudah dijelaskan di atas.

1.2 Rumusan masalah

Berdasarkan latar belakang diatas, maka rumusan masalah yang didapat:

1) Bagaimana cara monitoring seluruh perangkat yang ada dalam sebuah *Local Area Network* (misalnya *convenience store*)?
2) Bagaimana implementasi sistem monitoring perangkat pada *Yamaha Smart Gateway SGX808*?

1.3 Tujuan

Tujuan yang ingin dicapai dari penelitian ini adalah:

1) Dapat merancang sebuah sistem yang dapat digunakan untuk monitoring seluruh perangkat yang ada dalam sebuah *Local Area Network* (misalnya *convenience store*).
2) Dapat mengimplementasikan sistem yang sudah dirancang ke *Yamaha Smart Gateway SGX808*.

1.4 Manfaat

Secara umum, penelitian ini akan memberikan manfaat ke *convenience store*. Dengan penerapan aplikasi ini akan memudahkan monitoring seluruh perangkat dalam *convenience store* tersebut dan akan meningkatkan kinerja dari *convenience store* tersebut. Selain itu akan memberikan kenyamanan kepada pelanggan *convenience store* tersebut.

Secara khusus, penelitian ini akan memberikan manfaat ke *Yamaha Sound & Networking* selaku rekan dalam penelitian ini. Dengan penerapan aplikasi ini, pengguna *Yamaha Smart Gateway SGX808* akan bertambah seiring bertambahnya fungsi yang dimilikinya.

1.5 Batasan masalah

Agar pembahasan lebih terfokus berdasarkan latar belakang dan sistem, maka diberikan batasan masalah sebagai berikut:

1) Perangkat yang digunakan sebagai *server* dalam aplikasi monitoring ini adalah *Yamaha Smart Gateway SGX808*.
2) Perangkat-perangkat yang dapat terhubung hanya perangkat berbasis *IP*.
3) Monitoring dilakukan menggunakan protocol *ICMP*.

1.6 Sistematika pembahasan

Uraian singkat mengenai struktur penulisan pada masing-masing bab dijelaskan adalah sebagai berikut:

BAB I PENDAHULUAN

Pada bab ini menjelaskan tentang hal-hal yang menyangkut pendahuluan penelitian “Implementasi Sistem Monitoring Perangkat Klien pada *Yamaha Smart Gateway SGX808*” seperti latar belakang penelitian, rumusan masalah yang diangkat dalam penelitian, tujuan dari dilakukannya penelitian, manfaat dari hasil penelitian, batasan masalah pada penelitian, dan sistematika pembahasan penelitian.

BAB II LANDASAN KEPUSTAKAAN

Pada bab ini menjelaskan tentang kajian pustaka dari penelitian-penelitian sebelumnya yang memiliki topik yang mirip atau terkait dengan penelitian ini yaitu tentang *Internet of Things* dan yang berhubungan dengan sistem monitoring berbasis *web*. Selain itu pada bab ini juga dijelaskan dasar teori yang berhubungan dengan penelitian ini.

BAB III METODOLOGI

Pada bab metodologi dijelaskan metode-metode penelitian yang diterapkan dalam penelitian ini yang terdiri dari studi literatur, analisis kebutuhan, perancangan sistem, implementasi, dan pengujian.

BAB IV REKAYASA PERSYARATAN

Pada bab rekayasa persyaratan dijelaskan beberapa persyaratan secara detail yang mendukung perancangan sistem ini.

BAB V PERANCANGAN DAN IMPLEMENTASI

Pada bab perancangan dan implementasi dijelaskan langkah-langkah perancangan sistem mulai dari perancangan perangkat keras hingga perangkat lunak dan implementasinya. Selain itu juga dijelaskan spesifikasi perangkat keras dan perangkat lunak beserta batasan implementasi.

BAB VI PENGUJIAN DAN ANALISIS

Bab pengujian dan analisis menjelaskan langkah-langkah pengujian. Setelah itu diuraikan juga hasil dari pengujian serta analisis dari hasil pengujian tersebut.

BAB VII PENUTUP

Pada bab ini berisi tentang saran dan kesimpulan dari penelitian ini.

BAB 2 LANDASAN KEPUSTAKAAN

Dalam penelitian ini, pustaka referensi yang pertama berasal dari paper dari Karen Rose, et al, yang berjudul "*Internet of Things - Overview*" mengenai konsep *Internet of Things (IoT)* secara umum. Selanjutnya adalah pustaka referensi untuk penelitian tentang monitoring berbasis *web* yang sebelumnya sudah pernah dilakukan, yaitu "*A Web Based Temperature Monitoring Sistem*" dari M. Kassim, et al, dan "*Web Based Health Monitoring Sistem*" dari Prof. Y. R. Risodkar, et al. Selain dari penelitian sebelumnya, ada referensi lain yang digunakan pada penelitian ini, yaitu *datasheet* dan dokumentasi dari *Yamaha Smart Gateway SGX808* dari *Yamaha Sound & Networking Division*. Lalu sebagai tambahan, terdapat beberapa referensi tentang dasar teori dari beberapa komponen yang digunakan dalam penelitian ini, seperti *IP*, *ICMP*, *Web Server*, *Database*, *Yamaha Smart Gateway SGX808*.

2.1 Tinjauan Pustaka

Dalam penelitian yang dilakukan oleh Karen Rose, et al, yang dimuat dalam paper berjudul "*Internet of Things - Overview*" pada tahun 2005, secara umum menjelaskan definisi dari *Internet of Things (IoT)*, bentuk komunikasi dalam *IoT*, serta informasi lebih lanjut mengenai permasalahan dalam dunia *IoT*. Untuk definisi *IoT* sendiri sebenarnya belum ada definisi tunggal yang dapat benar-benar diterima secara universal. Definisi yang berbeda-beda digunakan oleh berbagai kelompok untuk menjelaskan sudut pandang dan sifat tertentu dari *IoT*.

Pengertian *IoT* secara umum menurut paper "*Internet of Things - Overview*" adalah sebuah skenario agar kemampuan konektifitas dan komputerisasi yang dimiliki sebuah jaringan komputer dapat menjangkau objek-objek, sensor-sensor, dan benda-benda yang digunakan sehari-hari serta tidak dikategorikan sebagai komputer, dan memungkinkan perangkat-perangkat ini untuk menghasilkan, menggunakan, serta saling bertukar data dengan sesedikit mungkin campur tangan dari manusia.

Selain definisi secara umum, disebutkan juga beberapa definisi yang sesuai dengan penelitian ini. Misalnya definisi *IoT* yang terfokus ke *Internet Protokol* (*IP*) yang dimuat dalam paper "*Architectural Considerations in Smart Object Networking*'' dari *Internet Architecture Board (IAB)*, yang menyebutkan bahwa *IoT* menunjukan sebuah kecenderungan dimana perangkat-perangkat yang bersifat *embedded* menggunakan *Internet Protokol* untuk berkomunikasi dan perangkat-perangkat tersebut tidak dioperasikan langsung oleh manusia.

Selain deskripsi tentang definisi dari *IoT*, dalam paper tersebut juga dijelaskan mengenai bentuk-bentuk komunikasi dalam antar *things* dalam *IoT*. Bentuk-bentuk komunikasi tersebut adalah sebagai berikut.

- *Perangkat-to-Perangkat Communication*, yaitu komunikasi dimana dua atau lebih perangkat saling terhubung dan berkomunikasi satu sama lain tanpa perantara.

- *Perangkat-to-Cloud Communication*, perangkat-perangkat *IoT* terhubung melalui internet ke sebuah layanan *cloud* yang bertugas sebagai penyedia layanan aplikasi untuk bertukar data dan pengontrol lalu lintas data.
- *Perangkat-to-Gateway Model*, perangkat-perangkat *IoT* terhubung ke layanan *cloud* melalui *gateway* lokal terlebih dahulu. Pada *gateway* lokal terdapat sebuah aplikasi yang berjalan, yang bertugas sebagai perantara antara perangkat-perangkat dan layanan *cloud* serta menyediakan layanan lain seperti keamanan dan translasi protokol maupun translasi data.
- *Back-End Data-Sharing Model*, arsitektur komunikasi yang memungkinkan pengguna untuk menggunakan data yang didapat dari sebuah layanan *cloud* yang dikombinasikan dengan data yang berasal dari sumber-sumber yang lain.

Selanjutnya penelitian tentang monitoring berbasis *web* yang sudah dilakukan, diambil dari paper "*A Web Based Temperature Monitoring Sistem*" dari M. Kassim, et al, dan "*Web Based Health Monitoring Sistem*" dari Prof. Y. R. Risodkar, et al.

Pada penelitian "*A Web Based Temperature Monitoring Sistem*", monitoring dilakukan dengan node sensor terhubung ke komputer. Data yang diambil node akan disimpan ke *database* melalui *MS Access 2003* dan akan terhubung ke *Abyss Web Server*. Pengguna akan dapat mengakses data dari *Web* Server melalui *internet.*

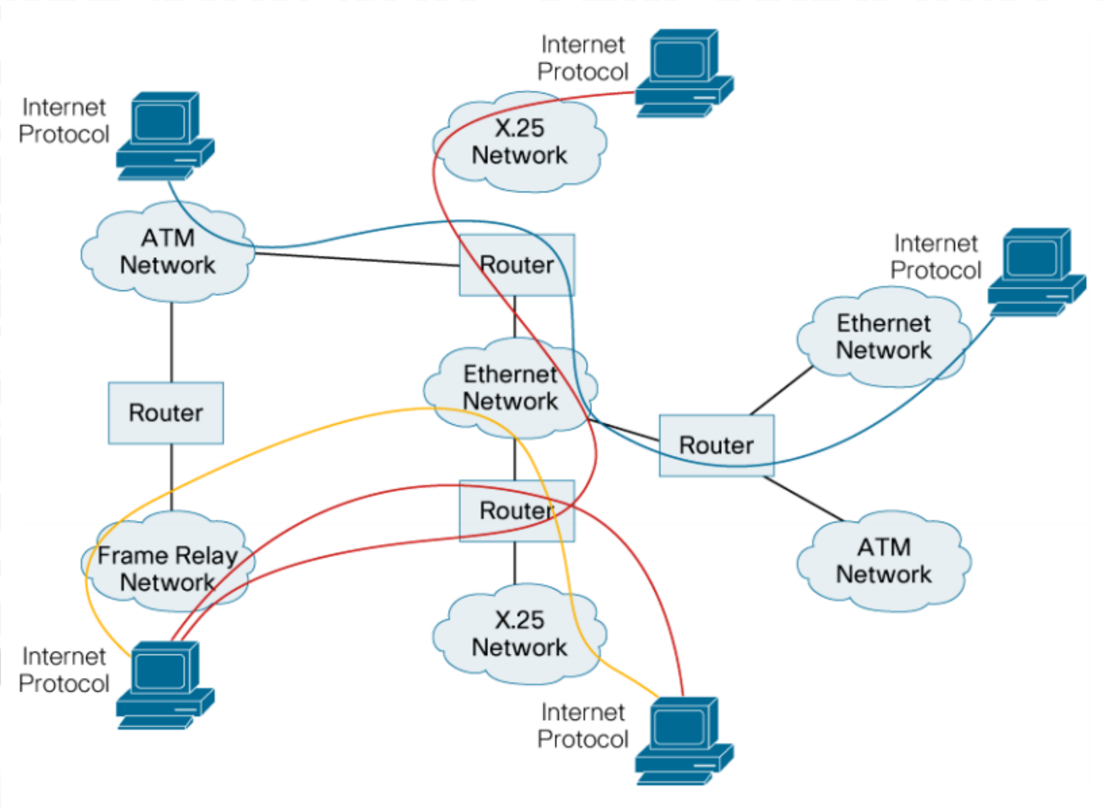
Dalam paper "*Web Based Health Monitoring Sistem*", monitoring dilakukan oleh *node* yang mempunyai modul komunikasi modul *XBEE* sebagai pengirim data. *Node* tersebut akan terhubung ke *Web* Server melalui modul *XBEE* sebagai penerima data, sehingga pengguna dapat mengakses data yang ada pada *Web* Server melalui *internet*.

2.2 Dasar Teori

Pada bagian dasar teori menjelaskan tentang beberapa definisi tentang teori-teori yang berkaitan dengan sistem ini meliputi *Internet Protokol* (*IP*), *Internet Control Message Protokol (ICMP)*, *Web Server*, *Database*, dan *Yamaha Smart Gateway SGX808*.

2.2.1 Internet Protokol (IP)

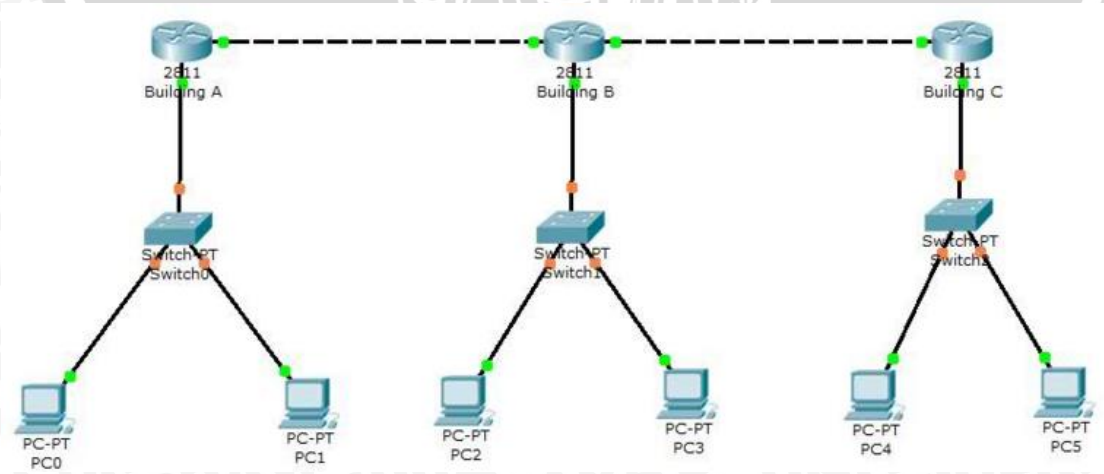
Internet Protokol (IP) merupakan sebuah fungsi yang paling umum digunakan untuk saling bertukar data di internet, dengan bentuk blok data yang disebut "*datagram*", dari sebuah *host* ke *host* yang dituju, dan memungkinkan untuk berkomunikasi pada jaringan yang berbeda (Internet Engineering Task Force, 2011). *IP* menjadi sangat penting karena apapun yang terhubung ke *internet* harus mendapatkan sebuah alamat *IP*. Saat ini, dengan semakin berkembangnya *internet* hampir semua perangkat dapat diberikan alamat *IP* dan terhubung ke *internet*.



Gambar 2.1 Ilustrasi penggunaan alamat *IP*

Sumber: (Cisco, 2009)

Dapat dilihat dari gambar 2.1 struktur *internet* di atas, semua perangkat yang terhubung ke internet harus mempunyai alamat *IP*, apapun media yang digunakan, baik menggunakan kabel, menggunakan jaringan *ATM*, maupun secara nirkabel. Seperti yang disebutkan dalam penjelasan sebelumnya, *IP* dapat digunakan untuk berkomunikasi berbeda jaringan.



Gambar 2.2 Ilustrasi pembagian alamat *IP*

Sumber: (Cisco, 2009)

Pada gambar 2.2 merupakan contoh sebuah topo*logi* jaringan komputer. Jaringan komputer gedung A, gedung B, dan gedung C berada pada jaringan yang berbeda. Yang dimaksud dalam sebuah jaringan yang sama yaitu yang terhubung ke satu antarmuka *router* yang sama dan bisa disebut berada dalam jaringan lokal. Sedangkan jaringan yang menghubungkan antar jaringan lokal dan terhubung ke *internet* disebut jaringan global. Dalam sebuah jaringan lokal, alamat *IP* yang digunakan adalah alamat *IP private*, dengan jangkauan sebagai berikut.

- 10.0.0.0 - 10.255.255.255
- 172.16.0.0 - 172.31.255.255
- 192.168.0.0 - 192.168.255.255

Sedangkan alamat *IP* global menggunakan alamat *IP* lain yang tidak dipakai untuk alamat *IP private* dan tidak dipesan untuk fungsi lain.

2.2.2 Internet Control Message Protokol (ICMP)

Internet control message protokol (ICMP) merupakan sebuah metode yang digunakan oleh *host-host* dan *router-router* untuk saling berkomunikasi satu sama lain pada *layer network* dan bertukar informasi. *ICMP* bisaanya digunakan untuk mengetahui kondisi sebuah koneksi yang dimiliki oleh sebuah *host* atau *router* ke *host* atau *router* yang lain (OMICS, 2014).

Pesan *ICMP* merupakan bagian dari *datagram IP*. Tujuan akhir dari suatu pesan *ICMP* bukan merupakan program atau pengguna melainkan aplikasi *layer network*nya. Ketika pesan *ICMP* hadir aplikasi *ICMP* akan menanganinya. *ICMP* mengijinkan *gateway* untuk mengirim pesan kesalahan ke *gateway* lain atau *host*. *ICMP* menyediakan komunikasi antar perangkat lunak protokol *internet*.

Pada dasarnya terdapat dua macam pesan *ICMP*: *ICMP Error Message* dan *ICMP Query Message*. *ICMP Error Message* digunakan pada saat terjadi kesalahan pada jaringan, sedangkan *ICMP Query Message* adalah jenis pesan yang dihasilkan oleh protokol *ICMP* jika pengirim paket menginginkan informasi tertentu yang berkaitan dengan kondisi jaringan.

Secara teknis *ICMP* adalah mekanisme pelaporan kesalahan untuk *gateway* sehingga dapat memberitahu sumber mengenai kesalahan yang terjadi. Sedangkan untuk koreksinya diserahkan pada program aplikasi yang ada pada pengirim. Contoh aplikasi yang menggunakan protokol *ICMP* adalah: *PING*.

PING merupakan salah satu program yang digunakan untuk mengecek komunikasi antar komputer dalam sebuah jaringan melalui protokol *TCP/IP. PING* akan mengirimkan pesan *ICMP Echo Request* pada alamat *IP* komputer yang dituju dan meminta respon dari komputer tersebut berupa pesan *ICMP Echo Reply* jika koneksi tersedia, dan akan dibalas dengan pesan kesalahan jika koneksi gagal. Beberapa contoh pesan kesalahan dalam *ping* adalah sebagai berikut:

Beberapa pesan yang mungkin muncul jika *ping* tidak berhasil antara lain:

- *TTL Expired in Transit*: artinya jumlah *hop* (*router*) yang dilalui untuk berkomunikasi dengan *server* tersebut telah melebihi *TTL (Time To Live).*

- *Destination Host Unreachable*: artinya paket yang dikirimkan tidak mampu sampai ke tujuan, bisaanya disebabkan oleh tabel *routing* yang tidak tepat di mesin *default gateway* atau *router*/*hop* diatasnya.
- *Request Timed Out*: artinya pesan *reply* tidak dapat diterima kembali dalam waktu yang sudah ditentukan. Biasanya pesan ini muncul karena pemblokiran yang mungkin dilakukan oleh *firewall* (baik disisi *router* maupun di sisi target).
- *Ping request could not find host*: artinya *resolving domain server* tersebut pada komputer tidak dapat menerjemahkan ke alamat *IP*. Hal ini biasanya karena konfigurasi *DNS* klien masih keliru atau komunikasi dengan *DNS server* terganggu/terputus.

2.2.3 *Web Server*

Gambar 2.3 Ilustrasi s*erver*

Source: (esplindia.com, 2015)

Server merupakan sebuah program komputer atau sebuah perangkat yang bertugas memberikan layanan, misalnya berbagi data atau sumber daya dan melakukan komputasi untuk program atau perangkat lain yang disebut klien. *Server* dapat menyediakan data melalui jaringan computer, baik berupa koneksi dengan kabel maupun nirkabel, atau dalam *Local Area Netwok (LAN)* maupun *Wide Area Network (WAN)* melalui *internet* seperti ilustrasi yang ditunjukkan pada gambar 2.3. Ada banyak sekali jenis server yang ada, namun pada penelitian ini akan difokuskan pada *web server*.

Berdasarkan penjelasan yang ada pada buku "*Computer Networking: A Top-Down Approach*" (Kurose & Ross, 2003), *web server* dirancang untuk melayani fungsi permintaan menggunakan protokol *HTTP*. Dimana dalam penerapannya, *HTTP* merupakan komponen yang sangat diperlukan yang berada di sisi server pada arsitektur aplikasi berbasis *web*. Langkah pengolahan dasar yang dilakukan pada *web server* diawali ketika menerima permintaan dari protokol *HTTP*. Kemudian ditentukan apakah file yang diminta tergolong sumber daya file statis atau dinamis. Beberapa contoh file statis seperti video, gambar, atau file *web HTML*. Apabila file tergolong statis, maka *server* akan mencari dan mengambil file untuk kemudian membalas permintaan klien dengan mengirimkan balasan file yang diminta. Sedangkan apabila file yang diminta tergolong file dinamis, yaitu file yang berisi berbagai parameter dari permintaan, seperti waktu, atau informasi tambahan lainnya, maka akan ditangani oleh *container* pada *server*. Beberapa contoh dari *HTTP web server*, terdiri dari *Apache*, *Apache Tomcat*, *Microsoft IIS*, *Nginx*, dan *Google GWS*.

2.2.4 *Database*

Database atau basis data adalah koleksi dari data-data yang terorganisasi dengan cara sedemikian rupa sehingga mudah disimpan, dimanipulasi, diperbaharui, dicari, dan diolah dengan perhitungan-perhitungan tertentu, serta dihapus (Kusrini, 2007). Salah satu dari aplikasi *database server* yaitu *MySQL*.

MySQL adalah sistem manajemen *database SQL (Structured Query Language)* yang populer saat ini. Sistem *database MySQL* mendukung beberapa fitur seperti *multithreaded*, *multi-user*, dan *SQL database managemen sistem (DBMS)*. *Database* ini dibuat untuk keperluan sistem *database* yang cepat, handal dan mudah digunakan. Program *MySQL* bersifat *open source* sehingga dapat digunakan dan dikembangkan dengan mudah tanpa harus membayar lisensi. Pada *MySQL* tersedia *username* dan *password* untuk keamanan serta level hak akses yang berbeda yang dapat diatur oleh penggunanya.

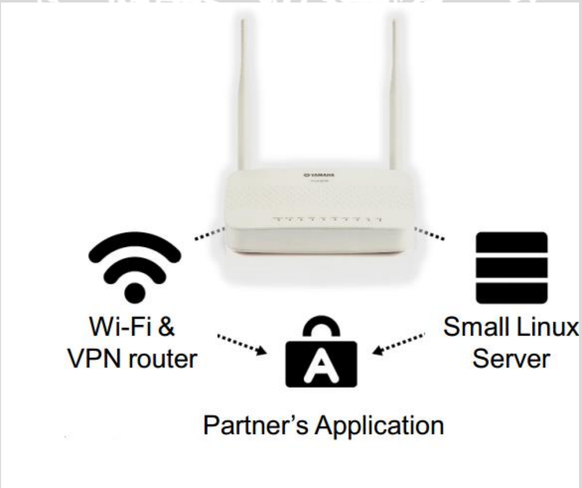
2.2.5 Yamaha Smart Gateway SGX808

Yamaha Smart Gateway SGX808 adalah sebuah *wireless LAN router* yang berbasiskan sistem operasi *Linux* dan mendukung *IEEE802.11b/g/n.* Pada perangkat dilengkapi dengan lingkungan yang mendukung untuk menjalankan perangkat lunak dari pihak ketiga untuk dapat dipasang serta dijalankan pada produk ini.

Gambar 2.4 *Yamaha Smart Gateway SGX808*

Source: (Yamaha Corporation, 2014)

Gambar 2.4 menunjukkan perangkat *Yamaha Smart Gateway SGX808*. Produk ini ditujukan untuk *Small and Medium Bussiness (SMB)*, contohnya seperti sebuah *convenience store*. Untuk memenuhi kebutuhan dari *SMB* maka produk ini memiliki beberapa kelebihan, misalnya *platform* jaringan khusus untuk *SMB* serta lingkungan sistem *LAMP*.



Gambar 2.5 *Platform* jaringan dari *Yamaha Smart Gateway SGX808*

Source: (Yamaha Corporation, 2014)

Gambar 2.5 menunjukkan *platform* jaringan dari *Yamaha Smart Gateway SGX808* untuk *SMB*. Pada perangkat ini dapat dipasang aplikasi pihak ketiga yang akan dapat membantu kinerja dari sebuah *SMB*. Aplikasi akan berjalan pada sistem operasi *Linux* dengan ukuran kecil sehingga akan bekerja secara efisien. Aplikasi dapat juga diakses oleh pengguna melalui *Wifi*, *VPN*, maupun dari *Ethernet* secara langsung.

Pada *Yamaha Smart Gateway SGX808* terpasang *Apache*, *MySQL*, dan *PHP* pada sistem operasi *Linux* dan memungkinkan perangkat ini untuk menjalankan aplikasi berbasis *web*. Aplikasi pihak ketiga dapat dipasang pada perangkat ini untuk menambah fitur yang tidak tersedia pada perangkat ini.

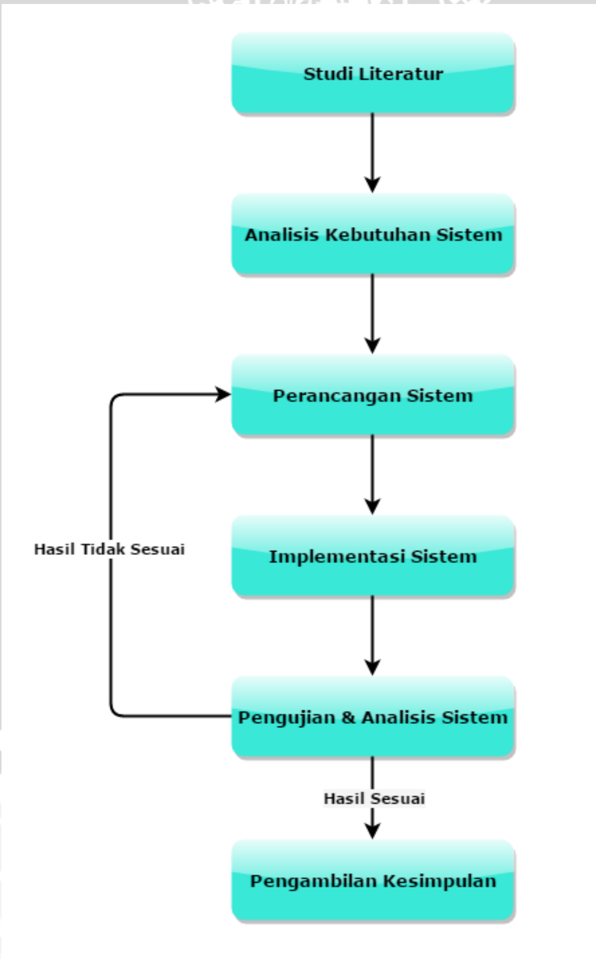
Berikut adalah spesifikasi umum dari *Yamaha Smart Gateway SGX808*.

Tabel 1 Spesifikasi umum *Yamaha Smart Gateway SGX808*

Flash ROM	*256 MB*
DRAM	*256 MB*
WAN port	*1 (10BASE-T/100BASE-TX MDI/MDIX-auto)*
LAN port	*4 (10BASE-T/100BASE-TX MDI/MDIX-auto)*
Wireless LAN	*2.4 GHz Standards:* *IEEE802.11b/g/n (Maximum clients 31)*
USB port	*1 (USB storage, USB modem)*
SD Card slot	*1 (Maximum 32GB)*
Manajemen	*Web Configuration, syslog, YMS-NMA*
VPN	*IPsec (2 destinations)*
Aplikasi	*Linux, Apache, MySQL, PHP*

BAB 3 METODOLOGI

Penelitian yang akan dilakukan ini diawali dengan studi literatur terkait dengan tinjauan pustaka dan dasar teori yang telah dilakukan pada riset-riset terdahulu yang terkait dengan *Internet of Things (IoT)* dan implementasi aplikasi monitoring berbasis *web*. Penelitian yang dilakukan bersifat implementatif dan pada tahap awal akan dilakukan adalah mengidentifikasi masalah yang akan diangkat pada penelitian ini. Lalu pada tahap selanjutnya yakni mempelajari literatur dari penelitian-penelitian sebelumnya yang memiliki keterkaitan tema dan teori-teori yang berhubungan dengan penelitian lalu dilakukan analisa kebutuhan sistem. Dari hasil analisa kebutuhan sistem akan dibuat sebuah perancangan sistem yang selanjutnya akan diimplementasikan. Pengujian dan analisa akan dilakukan terhadap sistem yang dibuat, jika hasil belum sesuai dengan tujuan maka akan dilakukan perubahan pada perancangan dan diimplementasi kembali. Jika sudah sesuai dengan tujuan maka akan ditarik sebuah kesimpulan dari penelitian yang dilakukan. Diagram alur penelitian akan ditunjukan pada gambar 3.1 berikut.



Gambar 3.1 Diagram alir metode penelitian

3.1 Studi Literatur

Studi literatur dilakukan untuk mempelajari referensi dari penelitian-penelitian sebelumnya yang memiliki keterkaitan tema yang digunakan sebagai acuan dalam penelitian yang akan dilakukan. Penelitian-peneilitan yang terkait meliputi penelitian yang memiliki topik *Internet of Things* dan penelitian yang berhubungan dengan sistem monitoring berbasis *web*. Selain itu juga dikaji teori-teori yang dapat dimanfaatkan pada penelitian ini yang meliputi:

1) *Internet Protokol (IP)*
2) *Internet Control Message Protokol (ICMP)*
3) *Web Server*
4) *Database*
5) *Yamaha Smart Gateway SGX808*

3.2 Analisis Kebutuhan

Analisis kebutuhan dilakukan untuk menganalisis kebutuhan yang diperlukan untuk perancangan sistem pada penelitian.

3.2.1 Kebutuhan Pengguna

Pengguna dalam penelitian ini ada dua pihak, yang pertama yaitu *Yamaha Sound & Networking Division* selaku rekan dalan penelitian ini, serta pengguna yang kedua yaitu pengguna produk dari *Yamaha Sound & Networking Division* secara umum, dan khususnya pemilik dari *convenience store* yang merupakan target utama pasar dari *Yamaha Smart Gateway SGX808*.

Kebutuhan pengguna yang pertama yaitu sistem yang dapat meningkatkan fungsionalitas dari produk *Yamaha Sound & Networking Division*, yaitu perangkat *Yamaha Smart Gateway SGX808*, sehingga dapat memenuhi kebutuhan konsumen.

Sedangkan kebutuhan pengguna yang kedua yaitu sebuah sistem yang dapat melakukan monitoring seluruh perangkat berbasis *IP* yang ada pada sebuah *Local Area Network*, atau sebuah *convenience store* untuk lebih spesifik.

3.2.2 Kebutuhan Sistem

Kebutuhan sistem yang diperlukan untuk memenuhi tujuan penelitian ini dibedakan menjadi dua jenis, yaitu kebutuhan fungsional dan kebutuhan non-fungsional.

Kebutuhan fungsional sistem ini meliputi:

- Sistem dapat menambah daftar perangkat yang dimonitoring.
- Sistem dapat melakukan monitoring terhadap perangkat yang terdaftar.

Sedangkan untuk kebutuhan non-fungsional meliputi:

- Sistem dapat menghapus daftar perangkat yang dimonitoring.
- Sistem dapat menambah daftar perangkat dari masukkan dari pengguna secara manual dan maupun masukkan dari tabel *ARP*.
- Sistem dapat menyimpan data daftar perangkat ke *database.*
- Sistem dapat memuat data daftar perangkat yang sudah disimpan di *database*.
- Sistem dapat monitoring dikonfigurasi parameter jeda waktunya.

3.2.3 Kebutuhan Perangkat Keras

Kebutuhan perangkat keras yang diperlukan untuk membangun sistem ini meliputi:

- *Yamaha Smart Gateway SGX808* sebagai *server* untuk sistem monitoring.
- Perangkat-perangkat berbasis *IP* yang akan dimonitoring oleh *server*.

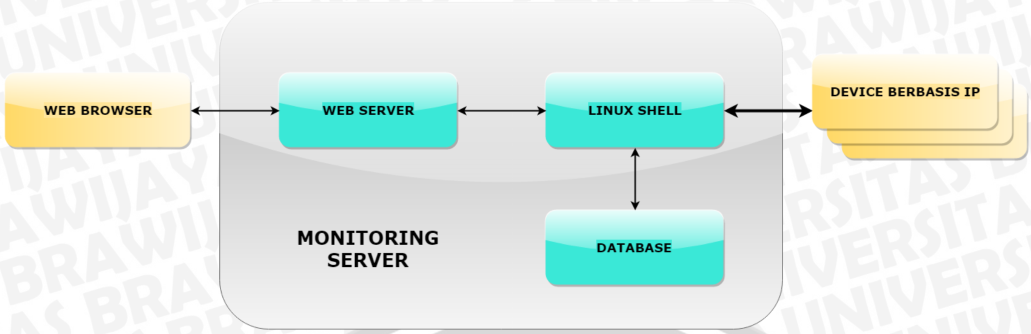
3.2.4 Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak yang diperlukan untuk membangun sistem ini meliputi:

- Sistem operasi *Linux*
- *Apache Web Server*
- *MySQL Database*
- *PHP*
- *Web Browser* yang mendukung *Javascript*

3.3 Perancangan Sistem

Pada tahap perancangan sistem dilakukan langkah-langkah dalam merancang sistem secara sistematis. Perancangan sistem dilakukan apabila seluruh kebutuhan sistem telah terpenuhi. Berdasarkan hasil analisa kebutuhan dari sistem yang sudah dilakukan sebelumnya, maka dapat dibuat sebuah perancangan dari sistem yang akan dibuat. Rancangan dari sistem yang akan dibuat dapat diwujudkan ke dalam sebuah diagram blok dari sistem yang akan ditunjukkan pada gambar 3.2 berikut.



Gambar 3.2 Gambaran umum sistem

Berdasarkan gambar 3.2, sistem akan terbentuk atas 3 bagian utama, berikut adalah penjelasan dari tiap bagian beserta hubungannya dengan bagian lain.

- *Web browser* bertugas menampilkan data dari perangkat yang dimonitoring *server* beserta statusnya. Selain itu *web browser* juga merupakan tempat bagi pengguna untuk memberikan masukkan data ke *server*. Pada *web browser* akan menggunakan *Javascript* untuk fungsi monitoring, serta menggunakan *PHP* dan *Ajax* untuk berkomunikasi dengan *web server*.
- Pada sistem yang dibuat pada penelitian ini, yaitu pada monitoring server, terbentuk atas 3 bagian, yang pertama yaitu *web server* yang bertugas melayani permintaan dari *web browser* dan akan berkomunikasi dengan *linux shell* menggunakan *PHP.* Lalu yang kedua adalah *database* yang dapat digunakan untuk menyimpan data dari perangkat-perangkat yang dimonitoring dan akan berkomunikasi dengan *linux shell* menggunakan *SQL Query.* Yang terakhir adalah *linux shell* yang bertugas untuk mendapatkan data dari perangkat-perangkat yang terhubung ke *monitoring server* dan mengakses data sementara yang dimiliki oleh *monitoring server* yang disimpan pada file .txt. *Monitoring Server* yang digunakan untuk melakukan monitoring perangkat adalah *Yamaha Smart Gateway SGX808*.
- Perangkat-perangkat berbasis *IP* yang akan dimonitoring oleh *monitoring server*. Perangkat-perangkat tadi akan berkomunikasi dengan *server* menggunakan protokol *ICMP* untuk mengetahui status perangkat baik melalui koneksi dengan atau tanpa kabel.

3.4 Implementasi Sistem

Impelementasi sistem merupakan tahap penerapan sistem dari perancangan sistem yang telah dibuat. Pada tahapan ini dibagi menjadi dua tahap, yang pertama yaitu implementasi lingkungan *server* dan selanjutnya adalah implementasi aplikasi.

Pada tahap pertama dilakukan implementasi lingkungan *server*, yaitu pengkondisian lingkungan pada server *Yamaha Smart Gateway SGX808*. Pengkondisian dimulai dengan mengubah *Yamaha Smart Gateway SGX808* ke *Developer Mode*. Dengan menggunakan *Developer Mode* ini akan mempermudah untuk *debugging* aplikasi yang sedang dikembangkan. Selain itu, dalam *Developer Mode* bisa dilakukan akses ke *terminal server* dengan menggunakan *telnet*. Setelah dapat mengakses ke *terminal server*, maka dilakukan pembuatan serta konfigurasi *database* yang akan digunakan oleh sistem monitoring perangkat.

Selanjutnya adalah implementasi aplikasi. Pada tahap ini, desain yang sudah dibuat pada perancangan sebelumnya diimplementasikan menggunakan *PHP* dan fungsi monitoring yang ada pada *Yamaha Smart Gateway SGX808*. Implementasi dimulai dari tampilan halaman *web* aplikasi monitoring perangkat menggunakan *HTML* yang dimasukkan ke file *PHP*. Setelah itu menerapkan fungsi monitoring perangkat, tambah dan hapus daftar perangkat, konfigurasi jeda waktu monitoring, serta simpan dan muat data dari *database* menggunakan *PHP*, *Javascript*, serta *SQL Query* untuk *database*.

3.5 Pengujian Sistem

Pengujian sistem dilakukan dengan pengujian fungsionalitas dari fungsi yang sudah dibuat pada tahap perancangan yang telah diimplementasikan pada *Yamaha Smart Gateway SGX808*. Fungsi-fungsi tersebut meliputi:

- Pengujian terhadap fungsi manajemen daftar perangkat, meliputi tambah daftar perangkat (baik tambah secara manual maupun dari tabel *ARP*), hapus daftar perangkat, dan hapus semua daftar perangkat.
- Pengujian fungsi monitoring terhadap perangkat-perangkat yang terdaftar dalam daftar perangkat.
- Pengujian terhadap fungsi untuk konfigurasi jeda waktu dari aplikasi monitoring.
- Pengujian terhadap fungsi untuk menyimpan data ke *database* serta memuat kembali data yang sudah disimpan ke *database*.

3.6 Pengambilan Kesimpulan

Apabila seluruh pengujian sistem telah dilakukan, maka dapat ditarik kesimpulan dari hasil data yang dihasilkan. Kesimpulan dibuat berdasarkan hasil pengujian secara garis besar.

BAB 4 REKAYASA PERSYARATAN

Pada bab ini berisi penjelasan tentang persyaratan-persyaratan yang harus dipenuhi agar sistem yang dibuat bisa berjalan dengan baik dan berfungsi sesuai dengan tujuan.

4.1 Pendahuluan

4.1.1 Tujuan

Bab ini bertujuan untuk menyajikan penjelasan rinci tentang Implementasi Sistem Monitoring Perangkat Klien pada *Yamaha Smart Gateway SGX808* berdasarkan format *Software Requirement Specification (SRS) IEEE 830*. Bab ini menjelaskan tujuan dan fitur dari sistem, antarmuka sistem, apa yang akan dilakukan sistem, batasan tentang bagaimana sistem akan beroperasi, serta bagaimana sistem akan bereaksi dengan pemicu eksternal atau beberapa faktor eksternal. Bab ini ditujukan untuk *stakeholder* atau pemangku kepentingan dan para pengembang sistem, untuk selanjutnya diusulkan sebagai persetujuan skripsi.

4.1.2 Ruang Lingkup

Tujuan dari sistem ini merupakan penerapan dari konsep *Internet of Things (IoT)* dalam bidang ekonomi, karena itulah dalam penelitian ini dilakukan dengan bekerja sama bersama salah satu perusahaan yang bergerak pada bidang jaringan komputer, yaitu *Yamaha Corporation*, khususnya *Sound & Networking Division*. Dengan memiliki rekan yang sudah mempunyai target dan tujuan yang jelas maka akan mempermudah untuk menemukan produk, target pasar, serta permasalahan yang akan dibahas.

Pada penelitian ini difokuskan ke salah satu produk dari *Yamaha Sound & Networking Division*, yaitu *Yamaha Smart Gateway SGX808*. Untuk target pasar dari perangkat tersebut adalah pelaku bisnis menengah seperti *convenience store.* Permasalahan diangkat adalah sistem melakukan monitoring untuk seluruh perangkat berbasis *IP* yang ada dalam sebuah *Local Area Network*, misalnya *convenience store*.

Sistem monitoring perangkat akan menggunakan *Yamaha Smart Gateway SGX808* sebagai *server*. Sistem yang akan dibuat menggunakan aplikasi berbasis *web*. Sistem akan melakukan monitoring semua perangkat berbasis *IP* yang terdaftar dalam daftar perangkat pada *server*. Daftar perangkat akan disimpan ke dalam sebuah file untuk bisa dikirim ke *web browser* yang mengakses *web* aplikasi monitoring perangkat ini. Selain itu data daftar perangkat juga dapat disimpan ke *database* yang ada dalam perangkat *Yamaha Smart Gateway SGX808*. Monitoring dilakukan dengan memanfaatkan salah satu protokol *ICMP*, yaitu *ping* untuk mengetahui apakah perangkat bekerja atau tidak.

4.1.3 Istilah

Tabel 2 Daftar istilah

Term	Definition
Internet of Things (IoT)	Sebuah skenario agar kemampuan konektifitas dan komputerisasi yang dimiliki sebuah jaringan komputer dapat menjangkau objek-objek, sensor-sensor, dan benda-benda yang digunakan sehari-hari yang tidak dikategorikan sebagai komputer, dan memungkinkan perangkat-perangkat ini untuk menghasilkan, menggunakan, serta saling bertukar data dengan sesedikit mungkin campur tangan dari manusia.
Convenience Store	Sebuah toko yang buka dari pagi hari dan tutup di larut malam, atau bahkan buka 24 jam, yang juga menjual makanan, minuman dan serta kebutuhan sehari-hari.
Server	Sebuah program komputer atau sebuah perangkat yang bertugas memberikan layanan, misalnya berbagi data atau sumber daya dan melakukan komputasi untuk program atau perangkat lain yang disebut klien.
Web	Sebuah halaman yang berisi informasi yang terdapat pada sebuah *server*, yang terhubung ke *internet*, dan terhubung dengan halaman-halaman lainnya, baik dengan halaman lain pada pada *server* itu sendiri maupun dengan halaman pada *server* lain.
Web browser	Sebuah perangkat lunak yang dipasang pada sebuah perangkat dan dapat mengakses halaman *web* yang dimiliki oleh sebuah *server*.
Database	Koleksi dari data-data yang terorganisasi dengan cara sedemikian rupa sehingga mudah disimpan, dimanipulasi, diperbaharui, dicari, dan diolah dengan perhitungan-perhitungan tertentu, serta dihapus.
Software Requirements Specification	Sebuah dokumen yang benar-benar menggambarkan sebuah fungsi dari sistem yang diusulkan dan bagaimana kebutuhan agar sistem dapat beroperasi
Stakeholder	Setiap orang yang berkepentingan dengan sistem atau projek yang bersangkutan, selain peneliti

4.1.4 Referensi

Berikut adalah referensi yang digunakan untuk penulisan rekayasa kebutuhan yang dilakukan pada sistem ini.

IEEE. *IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.* IEEE Computer Society, 1998.

4.1.5 Sistematika

Untuk mempermudah melihat dan mengetahui pembahasan yang ada, maka diperlukan sistematika yang merupakan kerangka dan pedoman penulisan *SRS* ini. Dokumen *SRS* ini dibagi menjadi 3 bagian sebagai berikut.

1. Pendahuluan

   Pada bagian ini terdiri dari tujuan, ruang lingkup, istilah, referensi, dan sistematika penulisan.

2. Deskripsi Umum

   Bagian ini terdiri dari perspektif produk, fungsi produk, karakteristik pengguna, batasan-batasan, asumsi dan ketergantungan.

3. Spesifikasi Kebutuhan

   Bagian ini terdiri dari kebutuhan fungsional, kebutuhan antarmuka eksternal, kebutuhan non-fungsional.

4.2 Deskripsi Umum

4.2.1 Perspektif Produk/Sistem

Sistem akan terpusat pada *server*, *Yamaha Smart Gateway SGX808*, yang akan selalu aktif dan akan melakukan monitoring ke perangkat yang terdaftar ke daftar perangkat. Semua perangkat yang terhubung ke *server* akan dapat didaftarkan ke daftar perangkat oleh pengguna, baik dengan masukkan manual maupun dari masukkan dari tabel *ARP*. Selain menambahkan ke daftar perangkat, pengguna dapat juga menghapus daftar perangkat atau mengosongkan daftar perangkat. Daftar perangkat yang ada dapat disimpan ke *database*, dan juga dapat dimuat dari data yang sudah disimpan dari *database*. Konfigurasi jeda untuk proses dapat dikonfigurasi juga oleh pengguna. Semua fungsi diatas dapat dilakukan oleh pengguna dengan mengakses *web* dari aplikasi monitoring perangkat ini.

4.2.2 Kegunaan

Aplikasi monitoring perangkat berbasis *web* ini dapat digunakan apakah perangkat yang terdaftar pada daftar perangkat bekerja atau tidak. Hal ini akan membuat pengawasan serta perawatan perangkat lebih efisien. Selain itu karena ini adalah aplikasi berbasis *web*, jadi tidak ada aplikasi yang harus dipasang sehingga akan menghemat memori.

4.2.3 Karakteristik Pengguna

Pengguna pada implementasi sistem bertindak sebagai pengguna aplikasi, yang dapat mengakses *web* dari aplikasi monitoring perangkat, sehingga mengetahui apakah sebuah perangkat bekerja atau tidak, menambah atau menghapus daftar perangkat, melakukan konfigurasi jeda dari proses monitoring, serta menyimpan dan memuat data ke dan dari *database*.

4.2.4 Lingkungan Operasi

Persyaratan yang mendukung kebutuhan sistem adalah sebagai berikut.

- Kondisi suhu lingkungan untuk server tidak boleh lebih dari 40 ^{o}C, dan kurang dari 0 ^{o}C
- Kondisi konsentrasi kelembaban lingkungan untuk *server* berkisar antara 15-80%
- Kebutuhan tegangan untuk *server* adalah AC 100 sampai AC 220 *Volt* (50 / 60 *Hertz*), dengan daya maksimal 22 *Watt*
- Sistem operasi *Linux* untuk *server*
- *Web* Browser yang mendukung *Javascript* untuk pengguna aplikasi monitoring perangkat

4.2.5 Batasan Perancangan dan Implementasi

Batasan perancangan dan implementasi dari sistem ini adalah sebagai berikut.

- Sistem akan bekerja apabila *server* mendapat *supply daya* atau tegangan yang sesuai
- Sistem akan bekerja apabila *server* dalam kondisi aktif
- Sistem monitoring hanya akan melakukan monitoring perangkat yang terhubung ke *server* dan terdaftar ke daftar perangkat

4.2.6 Asumsi dan Ketergantungan

Sistem akan menggunakan salah satu penggunaan protokol *ICMP*, yaitu *ping*, untuk mengecek kondisi dari perangkat yang dimonitoring. Sistem berasumsi bahwa jika *ping* sukses, maka perangkat tersebut aktif dan koneksi ke *server* tidak bermasalah. Jika ternyata perangkat bekerja dan *ping* gagal, maka perangkat tetap dianggap bermasalah karena tidak bekerja secara maksimal, dan akan dianggap tidak aktif sehingga mendapat perhatian dari pengguna untuk melakukan pengecekan pada perangkat tersebut.

4.3 Kebutuhan Antarmuka Eksternal

4.3.1 Antarmuka Pengguna

Pengguna menggunaka *web browser* yang mendukung *Javascript* untuk mengakses *web* aplikasi monitoring perangkat, sehingga dapat melihat status perangkat, menambah dan menghapus daftar perangkat, melakukan konfigurasi jeda proses monitoring, serta menyimpan data ke *database* dan memuat data yang sudah disimpan dari *database*.

4.3.2 Antarmuka Perangkat Keras

Perangkat keras yang digunakan sebagai *server* untuk monitoring perangkat adalah *Yamaha Smart Gateway SGX808*. Sedangkan perangkat yang dimonitoring adalah perangkat-perangkat berbasis *IP* yang terhubung ke *server*.

4.3.3 Antarmuka Perangkat Lunak

Perangkat lunak untuk server adalah sebagai berikut.

- Sistem operasi *Linux*
- *Apache Web Server*
- *MySQL Database*
- *PHP*

Sedangkan perangkat lunak untuk pengguna aplikasi adalah *Web Browser* yang mendukung *Javascript*.

4.3.4 Antarmuka Komunikasi

Yamaha Smart Gateway SGX808 selain berfungsi sebagai *server*, perangkat ini juga berfungsi sebagai *router*. Semua perangkat berbasis *IP* dapat terhubung ke *server* secara nirkabel menggunakan frekuensi 2.4 *GHz*. Selain itu bisa juga melalui melalui kabel dengan konektor *RJ-45*.

4.4 Kebutuhan Fungsional

Kebutuhan fungsional merupakan kebutuhan yang harus dipenuhi agar sistem dapat bekerja sesuai dengan tujuan dari sistem, dan kebutuhan fungsional untuk sistem ini adalah sebagai berikut.

4.4.1 Fungsi menambah daftar perangkat

1. Penjelasan dan Prioritas.

   Fitur ini mengharuskan sistem agar dapat menambahkan data perangkat baru ke dalam daftar perangkat yang sudah ada agar perangkat tersebut dapat dimonitoring oleh sistem. Jenis prioritas *high priority*.

2. Stimulus/Respon Sistem

Input data perangkat akan dimasukkan oleh pengguna melalui aplikasi *web*, dan sistem akan membaca masukkan pengguna dan menambahkan ke daftar perangkat

3. Kebutuhan Fungsional

Terdapat dua masukkan dari pengguna, yaitu nama perangkat dan alamat *IP*. Nama perangkat bisa menggunakan karakter apapun selain karakter spasi. Sedangkan data alamat *IP* harus valid. Jika salah satu dari kebutuhan diatas tidak terpenuhi, maka data perangkat baru tidak akan ditambahkan ke daftar perangkat dan akan ada peringatan ke pengguna.

REQ-1: Baca masukkan data perangkat dari pengguna
REQ-2: Validasi masukkan data perangkat dari pengguna
REQ-3: Tambahkan masukkan data perangkat dari pengguna ke daftar perangkat

4.4.2 Fungsi monitoring perangkat yang terdaftar pada daftar perangkat

1. Penjelasan dan Prioritas

Fitur ini mengharuskan sistem agar dapat melakukan monitoring terhadap seluruh perangkat yang terdaftar pada daftar perangkat dan menampilkan ke *web* agar bisa diketahui oleh pengguna. Jenis prioritas *high priority*.

2. Stimulus/Respon Sistem

Dalam sistem monitoring ini mempunyai jeda waktu, sehingga saat jeda waktu sudah terlewati maka monitoring akan dilakukan. Sistem akan membaca seluruh perangkat yang ada pada daftar perangkat dan melakukan pengecekan status tiap perangkat. Hasil akan ditampilkan ke halaman *web* aplikasi.

3. Kebutuhan Fungsional

Fungsi ini akan dapat bekerja apabila beberapa persyaratan berikut dipenuhi. Yang pertama yaitu jeda waktu *ping* dan jeda waktu monitoring tidak boleh bernilai 0. Perangkat yang dimonitoring harus terhubung ke *server* dan terdaftar pada daftar perangkat.

REQ-4: Cek apakah jeda waktu sudah terlewati
REQ-5: Baca data perangkat dari daftar perangkat dan cek status tiap perangkat
REQ-6: Tampilkan hasil monitoring ke halaman *web*

BAB 5 PERANCANGAN DAN IMPLEMENTASI

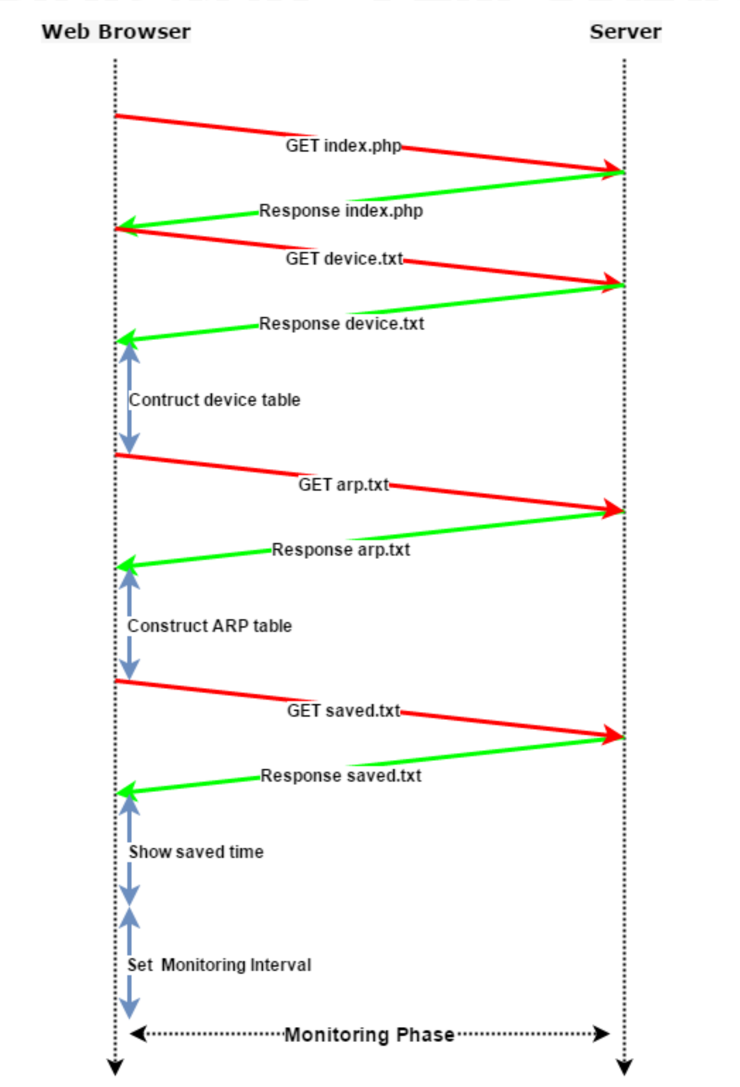
Pada bab ini akan menjelaskan tentang perancangan sistem yang dibuat pada penelitian ini, beserta implementasi sistem yang meliputi perangkat keras, perangkat lunak, serta pendukung sistem lainya.

5.1 Perancangan Sistem

Perancangan sistem dimulai dari penggambaran cara kerja sistem secara umum yang sudah dilakukan pada bab 3 sebelumnya. Selanjutnya perancangan akan dilakukan pada alur kerja dari sistem, perancangan topologi perangkat keras, perancangaan arsitekrur perangkat lunak, serta pendukung sistem lainya.

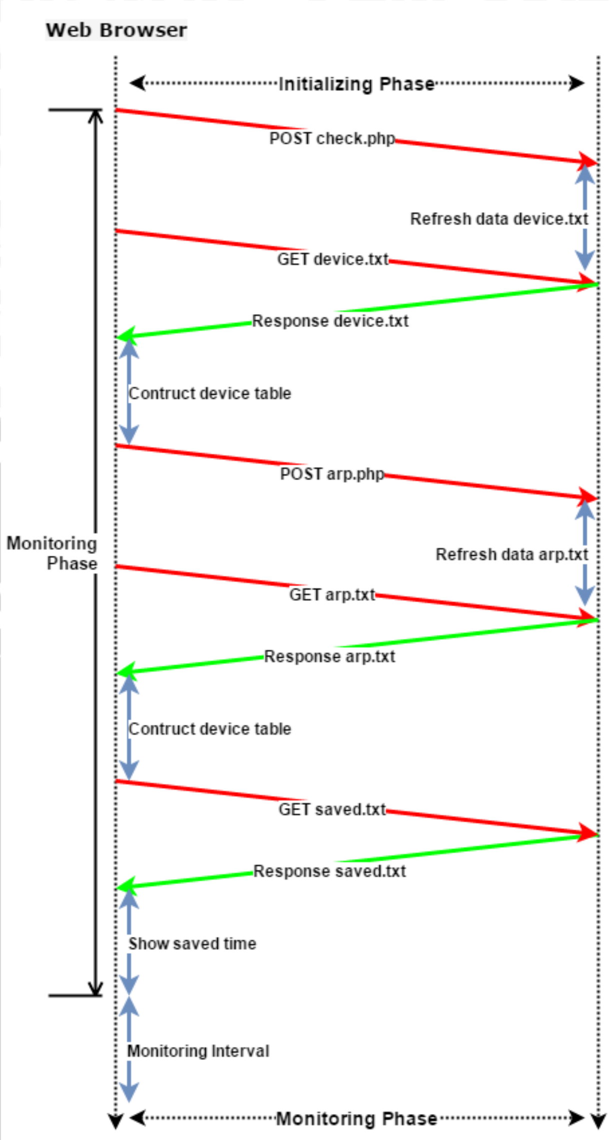
5.1.1 Alur Kerja Sistem

Pada alur kerja sistem ini akan dijelaskan alur kerja dari sistem monitoring perangkat pada saat tidak ada interaksi yang dilakukan oleh pengguna, serta komunikasi antara *web browser* dengan *server*. Proses dalam *web browser* akan menggunakan *Javascript* dan *Ajax*, sedangkan komunikasi antara *web browser* dengan *server* menggunakan *PHP*. Alur kerja sistem ini dibagi menjadi dua tahap, yaitu tahap inisialisasi serta tahap monitoring.



Gambar 5.1 Alur kerja sistem pada tahap inisialisasi

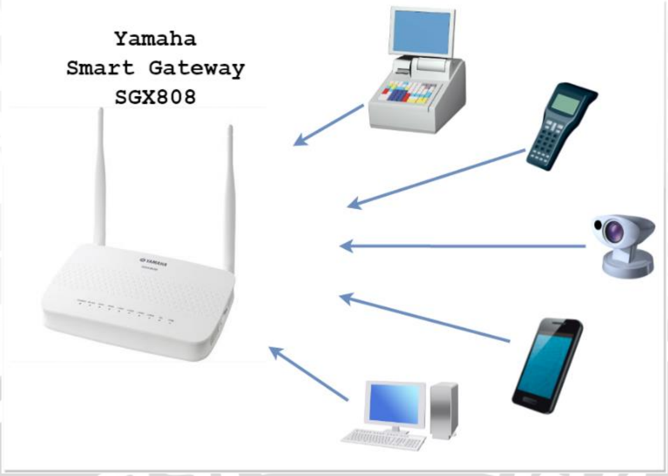
Pada gambar 5.1 merupakan alur kerja sistem saat tahap inisialisasi. Tahap ini berlangsung ketika *web browser* pertama kali melakukan akses ke *server*. Dimulai dengan permintaan halaman *web* ke *server*, lalu melakukan permintaan *HTTP GET* untuk file device.txt yang berisi daftar perangkat yang terdaftar pada *server* dan selanjutnya membuat tampilan untuk tabel daftar perangkat pada *web browser* menggunakan *HTML*. Selanjutnya akan dilakukan hal yang sama untuk arp.txt dan tampilan tabel *ARP*, serta saved.txt dan tampilan untuk waktu penyimpanan daftar perangkat ke *database*. Pada akhir tahap inisialisasi akan dilakukan set *interval monitoring* yang secara *default* bernilai 5 detik. Setelah tahap inisialisasi selesai maka masuk ke tahap berikutnya yaitu tahap monitoring.



Gambar 5.2 Alur kerja sistem pada tahap monitoring

Pada gambar 5.2 merupakan alur kerja sistem pada tahap monitoring. Setelah masuk ke tahap monitoring maka *web browser* akan mengirimkan permintaan *POST* check.php ke *server*, lalu *server* akan menggunakan check.php untuk me-*refresh* data daftar perangkat yang dimiliki beserta status perangkat yang terdaftar. Lalu *web browser* lalu melakukan permintaan *HTTP GET* untuk file device.txt yang baru *server* dan selanjutnya me-*refresh* tampilan daftar perangkat pada *web browser*. Selanjutnya melakukan hal yang sama untuk tabel *ARP* dengan mengirimkan permintaan *POST* arp.php. Sedangkan untuk waktu penyimpanan *database* hanya dilakukan permintaan *GET* saved.txt karena nilai ini tidak berubah setiap saat, namun hanya saat ada *event* tertentu saja. Setelah semua proses selesai maka akan menunggu sampai *interval monitoring* terlewati lagi dan mengulangi tahapan monitoring dari awal lagi dan begitu seterusnya.

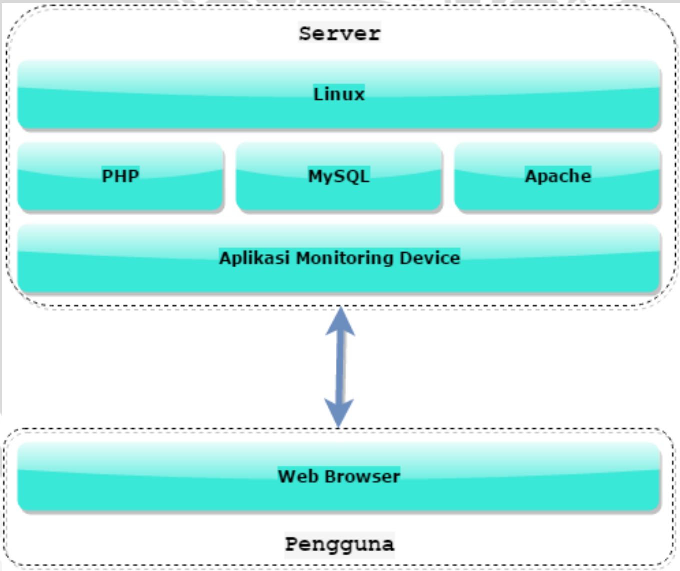
5.1.2 Perancangan Topologi Perangkat Keras



Gambar 5.3 Perancangan topologi perangkat keras

Pada gambar 5.3 merupakan ilustrasi perancangan topologi perangkat keras yang digunakan dalam sistem ini. *Yamaha Smart Gateway SGX808* akan bertindak sebagai *server* monitoring. Sedangkan perangkat yang akan dimonitoring adalah semua perangkat berbasis yang terhubung ke *server* dan terdaftar ke daftar perangkat. Pada gambar diatas merupakan ilustrasi contoh perangkat yang terhubung ke *server*, karena permasalahan dari *convenience store* maka ilustrasi yang dilakukan merupakan perangkat-perangkat yang kemungkinan besar ada dalam sebuah *convenience store*.

5.1.3 Arsitektur Perangkat Lunak

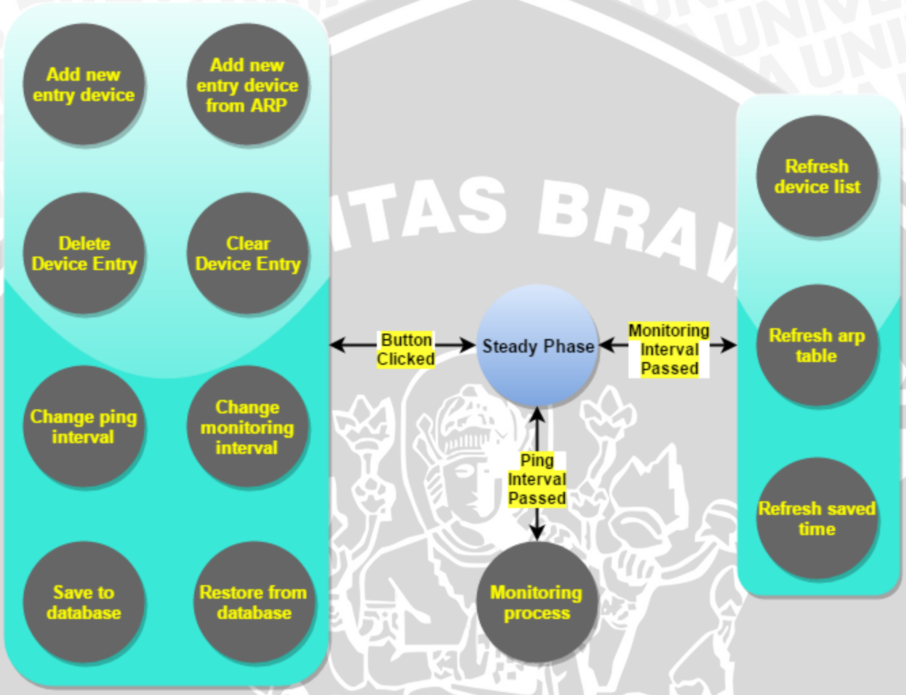


Gambar 5.4 Arsitektur perangkat lunak

Gambar 5.4 merupakan ilustrasi arsitektur perangkat lunak yang digunakan dalam sistem ini. Pengguna akan menggunakan *web browser* untuk mengakses aplikasi monitoring perangkat yang ada pada *server*. Sedangkan pada sisi *server*, aplikasi monitoring perangkat tersebut terhubung dengan aplikasi *PHP*, *MySQL*, serta *Apache* yang sudah terinstal pada *server*. Aplikasi-aplikasi tersebut berada pada sistem operasi *Linux* yang ada pada *server*.

5.1.4 Perancangan Aplikasi Berbasis *Web*

Pada tahap ini akan dijelaskan perancangan aplikasi monitoring perangkat berbasis *web* yang digunakan dalam sistem ini. Berikut adalah ilustrasinya.



Gambar 5.5 Gambaran umum aplikasi monitoring perangkat

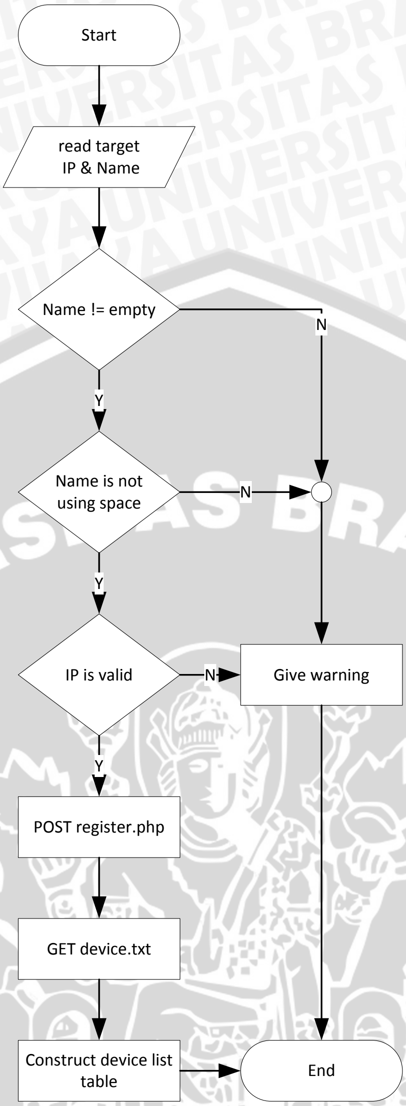
Gambar 5.5 merupakan gambaran umum aplikasi monitoring perangkat. Aplikasi ini secara garis besar dibagi menjadi tiga bagian, yang pertama yaitu pada gambar sebelah kanan ketika *interval monitoring* sudah terlewati. Untuk penjelasan dari bagian ini sudah dijelaskan sebelumnya pada alur kerja sistem.

Sedangkan bagian yang kedua yaitu pada gambar bagian bawah saat *interval ping* sedauh terlewati. Proses monitoring yang dimaksudkan adalah penggunaan fungsi yang dimiliki oleh *Yamaha Smart Gateway SGX808* yaitu fungsi Network Backup yang telah dimodifikasi untuk penelitian ini. Fungsi ini bekerja dengan melakukan pengecekan koneksi dari perangkat yang diinginkan menggunakan *ping* lalu menyimpan hasilnya pada file *log*.

Lalu bagian dari aplikasi yang ketiga, pada gambar bagian kiri, adalah saat ada interaksi dari pengguna aplikasi monitoring perangkat ini. Ada 8 jenis interaksi yang dapat dilakukan oleh pengguna. Berikut adalah penjelasannya.

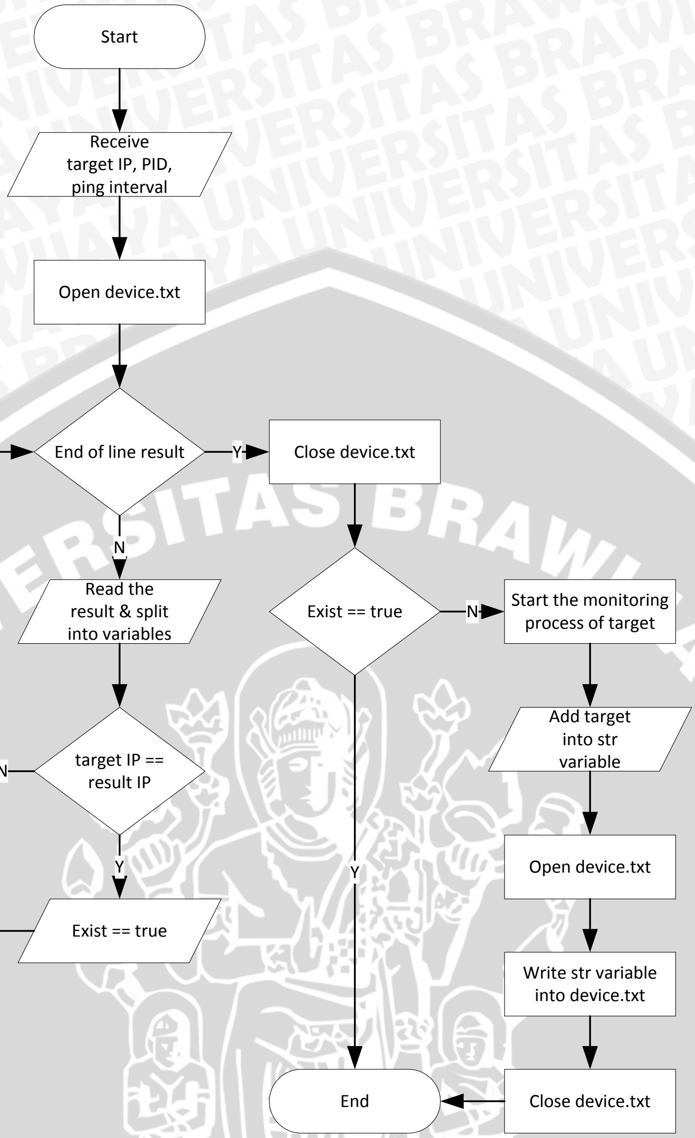
5.1.4.1 Tambah daftar perangkat baru

Pengguna melakukan penambahan daftar perangkat dengan memasukkan data dari perangkat, yaitu alamat *IP* serta nama perangkat secara manual.



Gambar 5.6 Proses penambahan daftar perangkat pada *web browser*

Gambar 5.6 menunjukkan proses penambahan daftar perangkat yang terjadi pada *web browser*. Pertama *web browser* akan membaca masukkan yang dilakukan oleh pengguna, yaitu nama perangkat serta alamat *IP*-nya. Lalu dilakukan pengecekan apakah nama bernilai kosong, nama menggunakan karakter spasi, dan apakah alamat *IP* yang dimasukkan valid atau tidak. Jika tidak memenuhi persyaratan diatas maka akan ditampilkan peringatan ke pengguna. Sedangkan jika memenuhi persyaratan maka *web browser* akan mengirimkan permintaan *POST* register.php ke *server* untuk mendaftarkan masukkan dari pengguna ke *server*. Lalu setelah *server* melakukan *update* daftar perangkat yang dimiliki, *web browser* melakukan permintaan data daftar perangkat yang baru menggunakan *GET* device.txt. Selanjutnya *web browser* akan melakukan *update* daftar daftar perangkat yang dimilikinya sesuai dengan data daftar perangkat paling baru yang dimiliki oleh *server*.

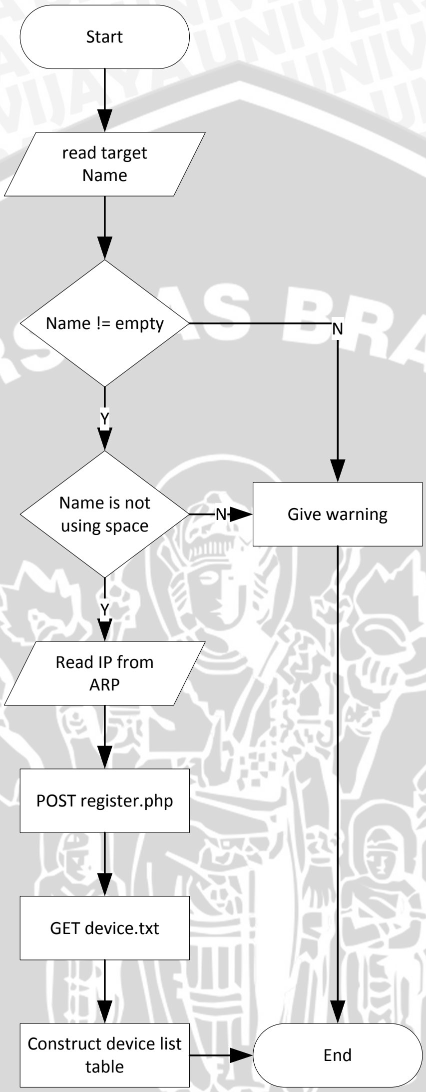


Gambar 5.7 Proses penambahan daftar perangkat pada *server*

Pada gambar 5.7 merupakan ilustrasi untuk penambahan daftar perangkat yang terjadi pada *server*. Setelah menerima permintaan *POST* dari *web browser*, maka *server* akan membaca data perangkat baru yang ditambahkan yaitu nama perangkat serta alamat *IP*. Lalu *server* akan membuka data daftar perangkat yang dimiliki pada device.txt, membaca tiap baris dari file lalu melakukan pengecekan apakah data perangkat baru yang ingin diinputkan sudah ada atau belum. Jika sudah ada maka tidak dilakukan apa-apa, namun jika belum ada maka *server* akan memulai proses monitoring untuk perangkat tersebut lalu menambahkan data perangkat yang baru ke daftar perangkat pada file device.txt

5.1.4.2 Tambah daftar perangkat baru dari tabel *ARP*

Penambahan daftar perangkat dilakukan dengan memilih perangkat yang sudah terhubung ke *server* dan ditampilkan pada tabel *ARP*. Tabel *ARP* ini merupakan tabel yang berisi semua perangkat yang saat ini sedang terhubung dengan *server*.

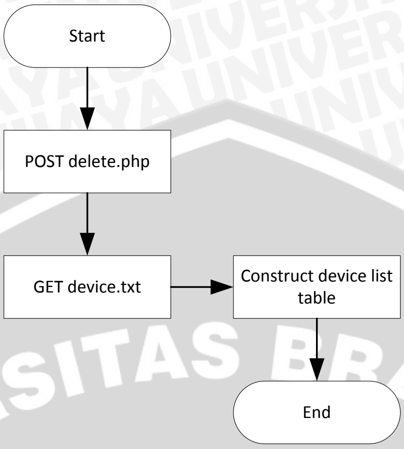


Gambar 5.8 Proses penambahan daftar perangkat dari tabel *ARP* pada *web browser*

Gambar 5.8 merupakan ilustrasi proses penambahan daftar perangkat dari tabel *ARP* pada *web browser*. Proses yang terjadi hampir sama dengan yang terjadi pada proses penambahan daftar perangkat secara manual. Perbedaanya data alamat *IP* dapat diambil dari tabel *ARP* yang dimiliki oleh *server*. Pada proses ini juga masih menggunakan register.php yang sudah dijelaskan pada gambar 5.9 sebelumnya.

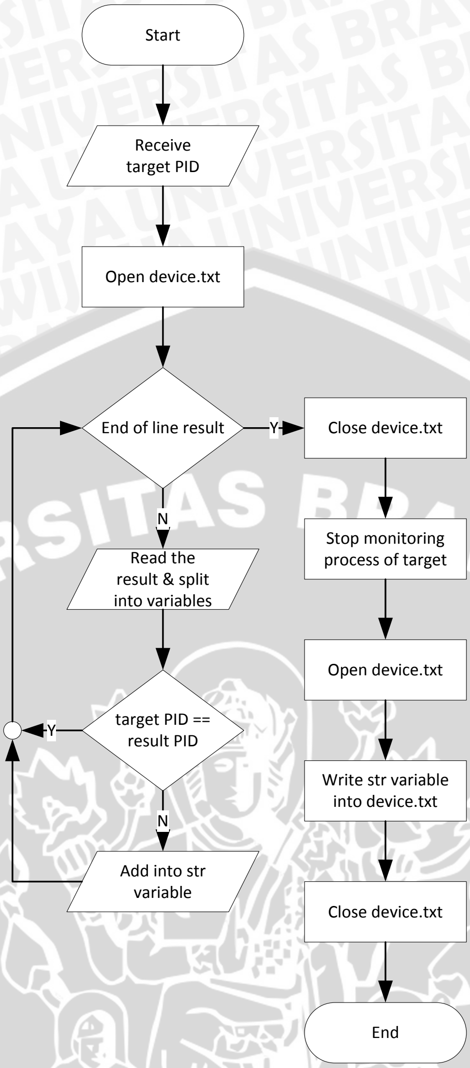
5.1.4.3 Hapus daftar perangkat

Pengguna dapat menghapus satu data perangkat dari daftar perangkat yang ada pada *server* dan menghentikan proses monitoring untuk perangkat tersebut.



Gambar 5.9 Proses penghapusan daftar perangkat pada *web browser*

Gambar 5.9 merupakan proses penghapusan daftar perangkat pada *web browser*. Untuk melakukan penghapusan daftar perangkat, *web browser* melakukan pengiriman permintaan *POST* delete.php beserta PID dari daftar perangkat yang ingin dihapus. Lalu *server* akan melnghapus data daftar perangkat tersebut dan meng*update* data daftar perangkat yang disimpan pada file device.txt. Lalu setelah *server* melakukan *update* daftar perangkat yang dimiliki, *web browser* melakukan permintaan data daftar perangkat yang baru menggunakan *GET* device.txt. Selanjutnya *web browser* akan melakukan *update* daftar daftar perangkat yang dimilikinya sesuai dengan data daftar perangkat paling baru yang dimiliki oleh *server*.

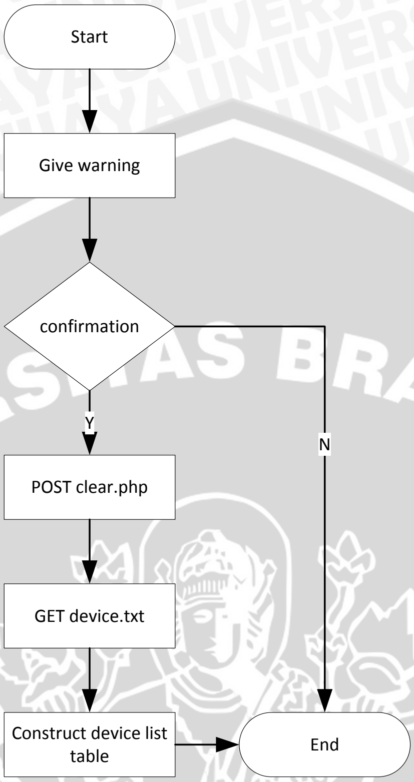


Gambar 5.10 Proses penghapusan daftar perangkat pada *server*

Pada gambar 5.10 merupakan ilustrasi untuk penghapusan daftar perangkat pada *server*. *Server* akan menerima *PID* dari daftar perangkat yang ingin dihapus. Lalu *server* akan membuka file device.txt dan melakukan pengecekan terhadap semua perangkat yang terdaftar. Semua perangkat yang terbaca akan dimasukkan ke variabel sementara, kecuali target yang ingin dihapus. Lalu proses monitoring untuk target yang akan dihapus dihentikan. Selanjutnya data yang ada pada variabel sementara tadi dimasukkan ke file device.txt sebagai data daftar perangkat yang baru.

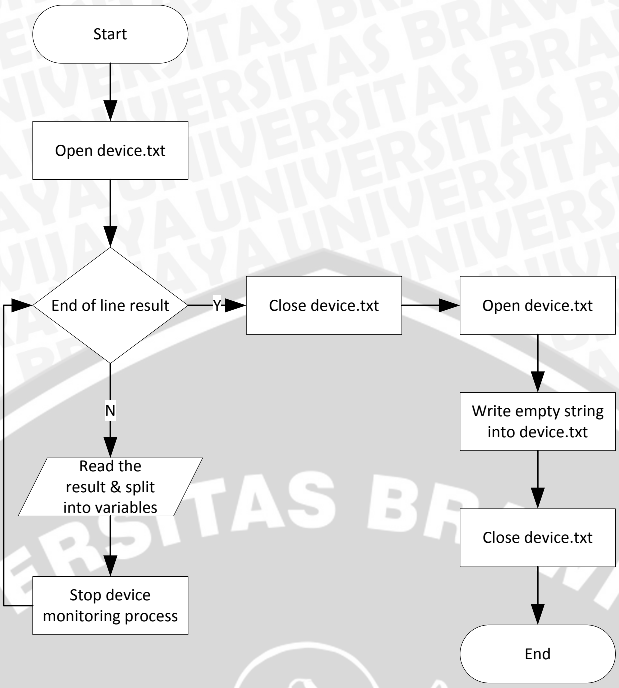
5.1.4.4 Hapus semua daftar perangkat

Fungsi ini digunakan untuk menghapus dan menghentikan proses monitoring semua perangkat yang terdaftar pada daftar perangkat.



Gambar 5.11 Proses penghapusan semua daftar perangkat pada *web browser*

Gambar 5.11 merupakan ilustrasi proses penghapusan semua daftar perangkat yang terjadi pada *web browser*. Karena penghapusan semua perangkat merupakan sesuatu yang bersifat kritis, maka akan ada peringatan terlebih dahulu. Jika dikonfirmasi oleh pengguna maka *web browser* akan mengirimkan permintaan *POST* clear.php ke *server*. Lalu setelah *server* melakukan update daftar perangkat yang dimiliki, *web browser* melakukan permintaan data daftar perangkat yang baru menggunakan *GET* device.txt. Selanjutnya *web browser* akan melakukan *update* daftar daftar perangkat yang dimilikinya sesuai dengan data daftar perangkat paling baru yang dimiliki oleh *server*.

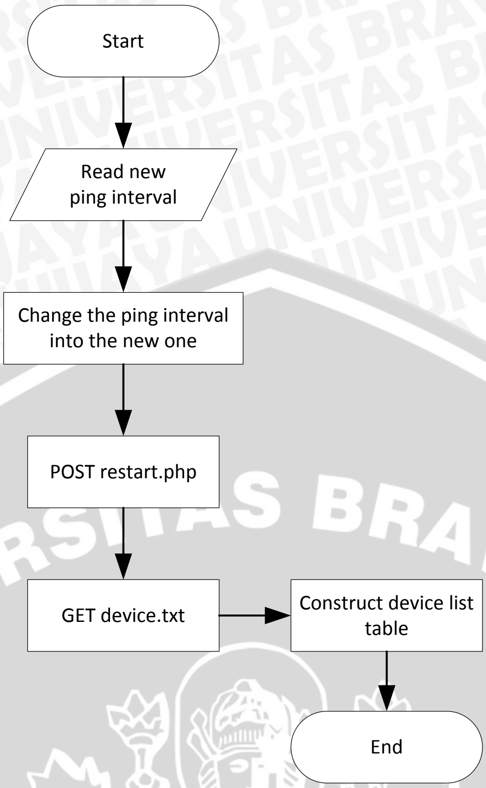


Gambar 5.12 Proses penghapusan semua daftar perangkat pada *server*

Gambar 5.12 merupakan ilustrasi penghapusan semua daftar perangkat yang terjadi pada *server*. Setelah menerima permintaan *POST* clear.php maka *server* akan membuka device.txt lalu membaca data semua perangkat yang ada didalamnya dan menghentikan proses monitoring pada semua perangkat yang terdaftar. Lalu menghapus semua data pada device.txt dengan menuliskan *string* bernilai kosong didalamnya.

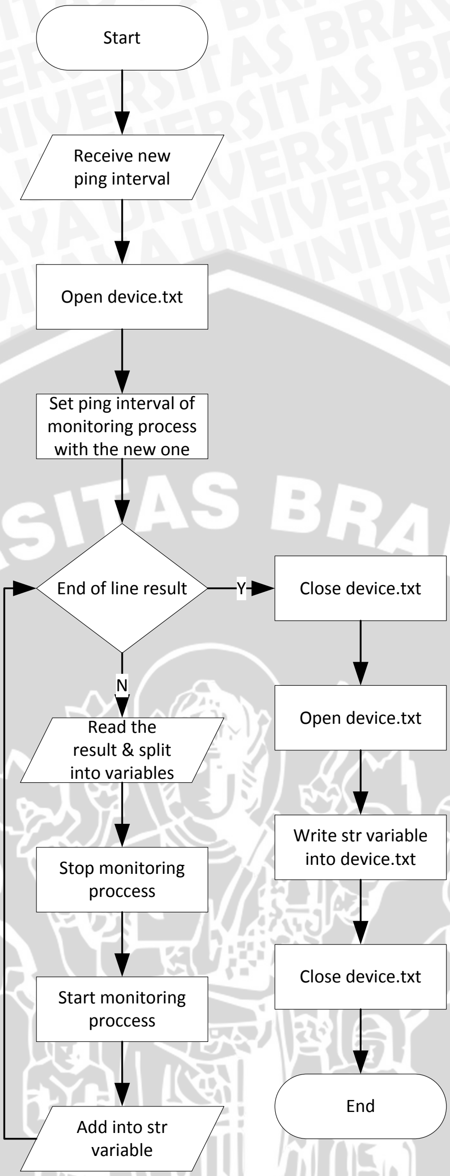
5.1.4.5 Mengubah *interval ping*

Pada fungsi ini memungkinkan pengguna untuk mengubah *interval ping* yang digunakan oleh proses monitoring untuk mengoptimalkan kinerja sistem berdasarkan banyaknya perangkat yang terhubung ke *server*.



Gambar 5.13 Proses mengubah *interval ping* pada *web browser*

Gambar 5.13 merupakan ilustrasi proses mengubah *interval ping* yang terjadi pada *web browser*. Dimulai dengan *web browser* yang akan membaca inputan *interval ping* yang baru dari masukkan pengguna. Lalu setelah *server* melakukan *update* daftar perangkat yang dimiliki, *web browser* melakukan permintaan data daftar perangkat yang baru menggunakan GET device.txt. Selanjutnya *web browser* akan melakukan *update* daftar daftar perangkat yang dimilikinya sesuai dengan data daftar perangkat paling baru yang dimiliki oleh *server*.

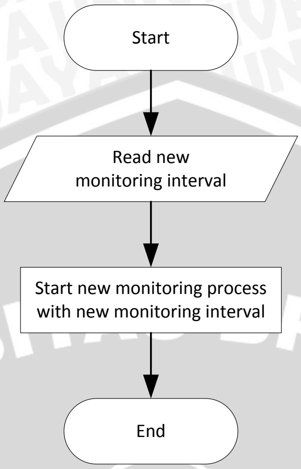


Gambar 5.14 Proses mengubah *interval ping* pada *server*

Gambar 5.14 merupakan ilustrasi mengubah *interval ping* yang terjadi pada *server*. Setelah menerima permintaan *POST* restart.php maka *server* akan membaca nilai *interval ping* yang baru, lalu mengubah parameter *interval ping* pada proses monitoring dengan nilai *interval ping* yang baru. Agar nilai dapat diterapkan pada tiap proses monitoring, maka setiap proses harus di *restart* terlebih dahulu. Maka *server* akan membuka device.txt lalu membaca data semua perangkat yang ada didalamnya dan menghentikan proses monitoring pada semua perangkat yang terdaftar lalu menjalankan kembali proses monitoring dengan nilai *interval ping* yang sudah dirubah. Selanjutnya menyimpan data daftar perangkat yang baru ke file device.txt. Data daftar perangkat perlu dirubah karena setelah mengalami proses *restart*, maka *PID* dari proses monitoring akan berubah sehingga data daftar perangkat perlu di *update*.

5.1.4.6 Mengubah *interval monitoring*

Pada fungsi ini pengguna dapat melakukan konfigurasi *interval monitoring* untuk mengoptimalkan sistem dengan menyesuaikan dengan jumlah perangkat yang dimonitoring.

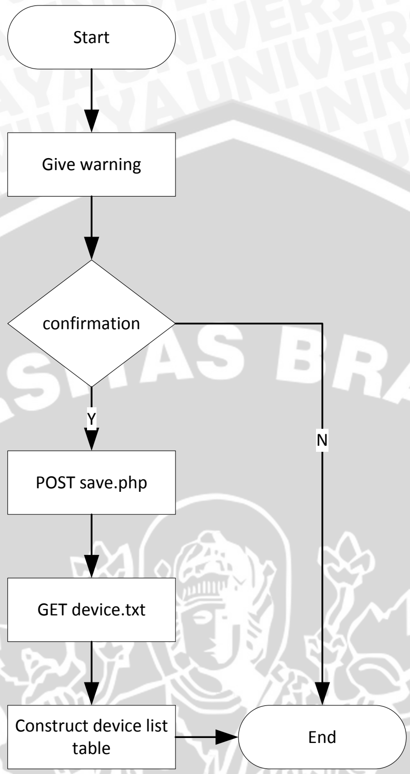


Gambar 5.15 Proses mengubah *interval monitoring*

Gambar 5.15 merupakan ilustrasi proses mengubah *interval monitoring*. Pertama *web browser* akan membaca nilai *interval monitoring* yang baru dari inputan pengguna. Karena *interval monitoring* ini hanya berlaku pada *browser* dan tidak ke *server*, maka yang dilakukan selanjutnya adalah menjalankan proses monitoring yang baru dengan *interval monitoring* yang sudah ditentukan. Namun sebelum menjalankan proses monitoring yang baru, maka proses monitoring yang lama harus dihentikan terlebih dahulu agar tidak terjadi dua proses yang sama.

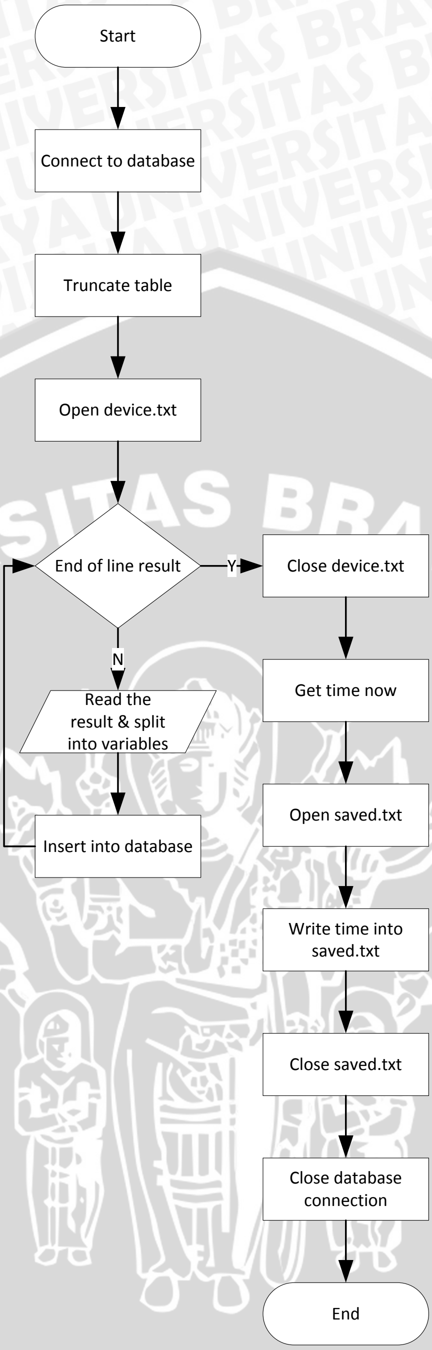
5.1.4.7 Simpan data daftar perangkat ke *database*

Pengguna dapat melakukan penyimpanan data daftar perangkat yang sekarang ada ke *database* sehingga akan dapat di muat lagi nantinya.



Gambar 5.16 Proses penyimpanan daftar perangkat ke *database* pada *web browser*

Gambar 5.16 merupakan ilustrasi proses penyimpanan daftar perangkat ke *database* yang terjadi pada *web browser*. Karena sebelum penyimpanan daftar perangkat ke *database* dilakukan penghapusan semua daftar perangkat yang sebelumnya ada pada *database*, maka fungsi ini bersifat kritis, maka akan ada peringatan terlebih dahulu. Jika dikonfirmasi maka *web browser* akan mengirimkan permintaan *POST* save.php ke *server*. Lalu setelah *server* melakukan *update* daftar perangkat yang dimiliki, *web browser* melakukan permintaan data daftar perangkat yang baru menggunakan *GET* device.txt. Selanjutnya *web browser* akan melakukan *update* daftar daftar perangkat yang dimilikinya sesuai dengan data daftar perangkat paling baru yang dimiliki oleh *server*.

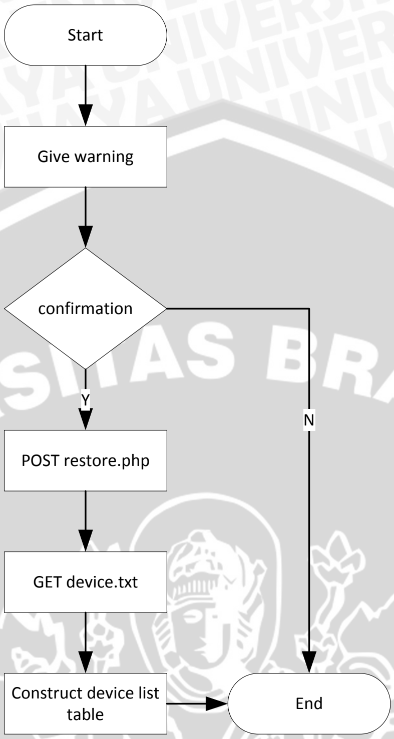


Gambar 5.17 Proses penyimpanan daftar perangkat ke *database* pada *web browser*

Gambar 5.17 merupakan ilustrasi proses penyimpanan daftar perangkat ke *database* yang terjadi pada *web browser*. Ketika *server* menerima permintaan *POST* save.php maka *server* akan melakukan koneksi ke *database* dan menghapus semua data yang ada pada tabel yang digunakan. Lalu *server* akan membuka device.txt dan membaca data dari semua perangkat yang terdaftar dan melakukan *insert* ke *database* untuk setiap perangkat yang terdaftar. Selanjutnya *server* akan membaca waktu saat itu dan menyimpannya ke file saved.txt. Dan yang terakhir *server* akan menutup koneksi yang sudah dibuat ke *database*.

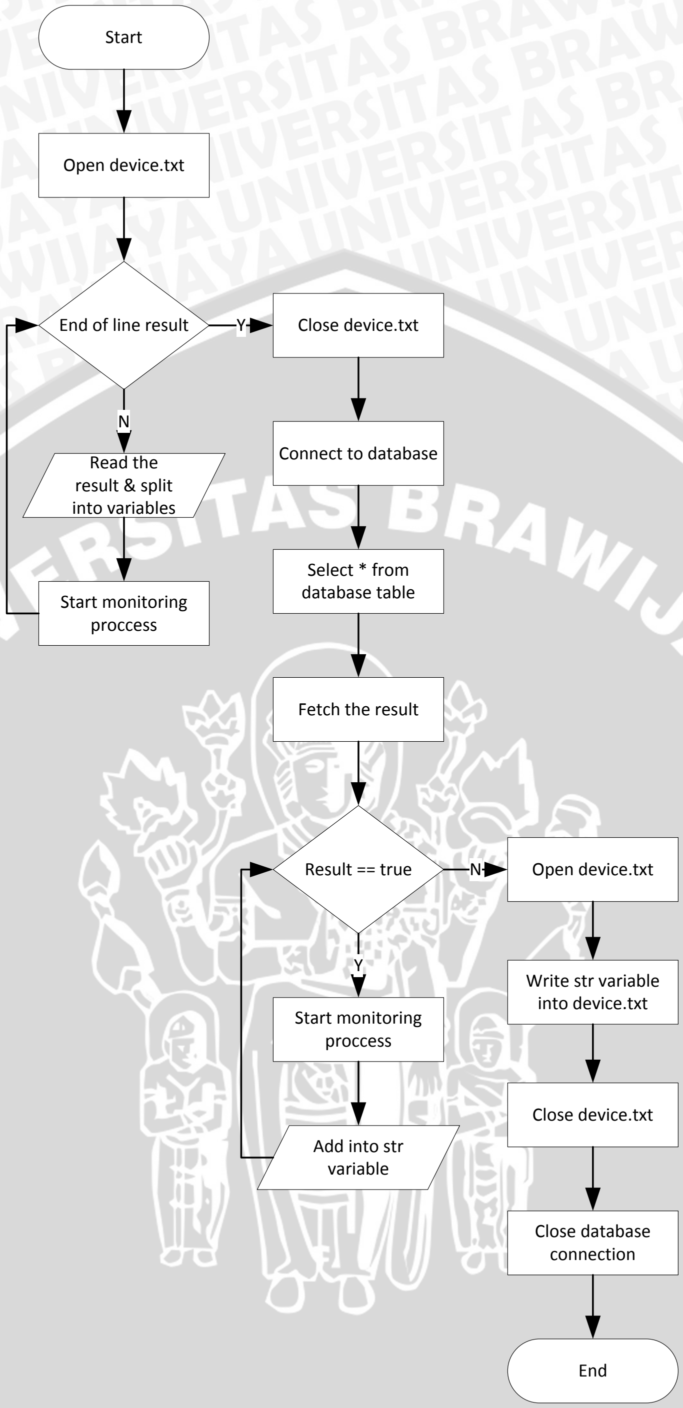
5.1.4.8 Muat data daftar perangkat dari *database*

Pengguna dapat memuat kembali data daftar perangkat yang sebelumnya sudah disimpan ke *database* ke data daftar perangkat yang digunakan sekarang.



Gambar 5.18 Proses memuat kembali data daftar perangkat dari *database* pada *web browser*

Gambar 5.18 merupakan ilustrasi proses memuat kembali data daftar perangkat dari *database* yang terjadi pada *web browser*. Karena memuat kembali daftar perangkat dari *database* akan menghapus semua daftar perangkat yang saat ini ada, maka fungsi ini bersifat kritis, maka akan ada peringatan terlebih dahulu. Jika dikonfirmasi maka *web browser* akan mengirimkan permintaan *POST* restore.php ke *server*. Lalu setelah *server* melakukan *update* daftar perangkat yang dimiliki, *web browser* melakukan permintaan data daftar perangkat yang baru menggunakan *GET* device.txt. Selanjutnya *web browser* akan melakukan *update* daftar daftar perangkat yang dimilikinya sesuai dengan data daftar perangkat paling baru yang dimiliki oleh *server*.



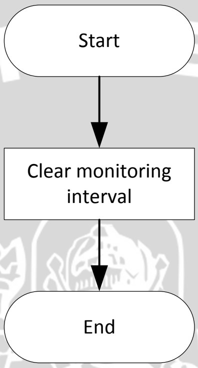
Gambar 5.19 Proses memuat kembali data daftar perangkat dari *database* pada *server*

Pada gambar 5.19 merupakan ilustrasi memuat kembali data daftar perangkat dari *database* yang terjadi pada *server*. Ketika *server* menerima permintaan *POST* restore.php maka *server* akan membuka file device.txt dan membaca seluruh data daftar perangkat yang ada lalu menghentikan proses monitoring untuk setiap perangkat. Selanjutnya *server* akan membuat koneksi ke *database* dan membaca seluruh daftar perangkat yang ada pada tabel yang digunakan. Untuk setiap daftar perangkat yang ada pada *database*, maka *server* akan menjalankan proses monitoring untuk setiap perangkat lalu menyimpan daftar perangkat yang baru ini ke file device.txt. Dan yang terakhir *server* akan menutup koneksi *database* yang telah dibuat.

Selain 8 fungsi diatas dapat terdapat beberapa fungsi lain yang dibutuhkan oleh sistem, berikut adalah penjelasanya.

1. Menghentikan proses monitoring

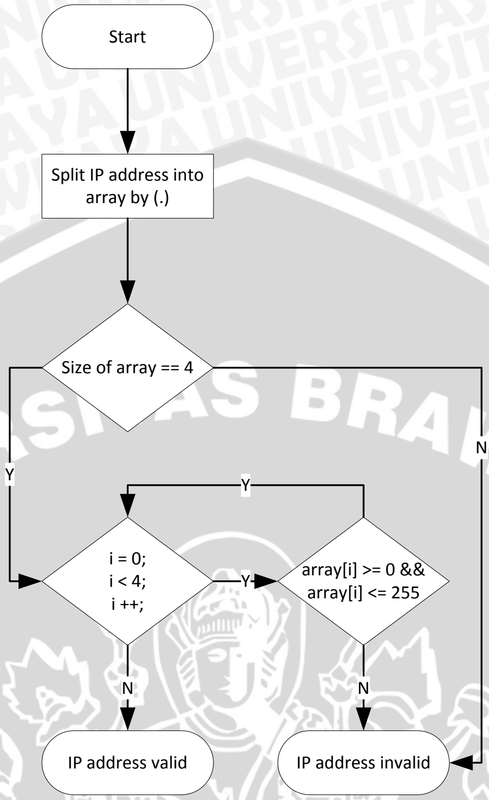
Walaupun saat dalam kondisi tidak ada perangkat yang dimonitoring, *web browser* tetap akan menjalankan fungsi monitoring jika *interval monitoring* sudah terlewati. Hal ini merupakan pemborosan pada sumber daya yang dimiliki. Untuk itu ada fungsi untuk menghentikan fungsi monitoring, berikut ilustrasinya.



Gambar 5.20 Proses untuk menghentikan fungsi monitoring pada *web browser*

Gambar 5.20 merupakan ilustrasi proses untuk menghentikan fungsi monitoring pada *web browser*. Untuk menghentikannya cukup dengan menghapus *interval monitoring* yang sudah ada sebelumnya. Selain itu fungsi ini juga dibutuhkan untuk fungsi mengubah *interval monitoring* yang diharuskan untuk menghentikan proses monitoring yang sebelumnya sudah berjalan.

2. Fungsi untuk validasi alamat *IP*



Gambar 5.21 Ilustrasi fungsi untuk melakukan validasi alamat *IP*

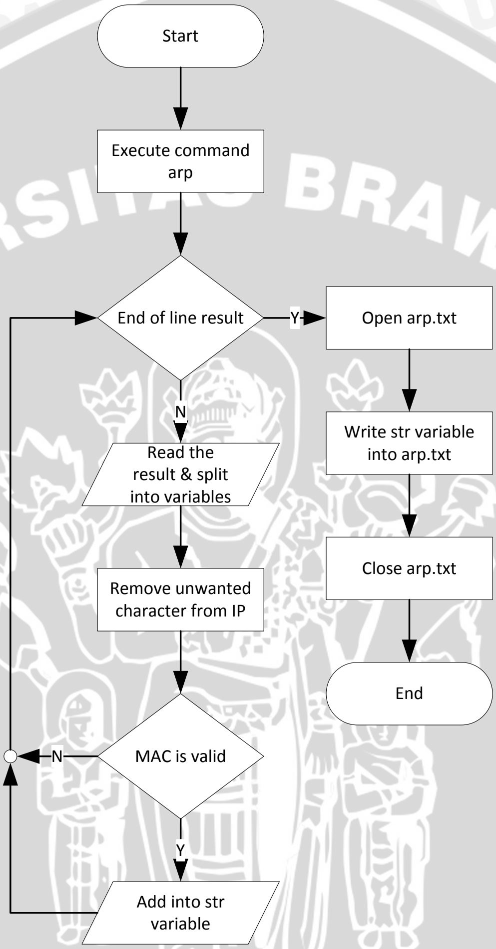
Gambar 5.21 menunjukkan proses untuk validasi alamat *IP*. Berikut adalah persyaratan alamat *IP* dianggap valid dalam penelitian ini.

Contoh alamat *IP* yang valid: 192.168.100.1

- Terdiri dari 4 *oktet* (bagian) yang dipisahkan oleh karakter (.)
- Angka pada tiap bagian harus lebih dari atau sama dengan 0 dan kurang dari atau sama dengan 255.

3. Fungsi untuk me*refresh* data tabel *ARP*

Tabel *ARP* berisi semua perangkat yang terhubung ke *server*. Tabel ini perlu ditampilkan agar memudahkan pengguna untuk menambahkan daftar perangkat. Karena kadang sulit untuk mengetahui alamat *IP* pada beberapa perangkat, maka tabel *ARP* ini akan memudahkan pengguna untuk menambahkan perangkat tersebut.

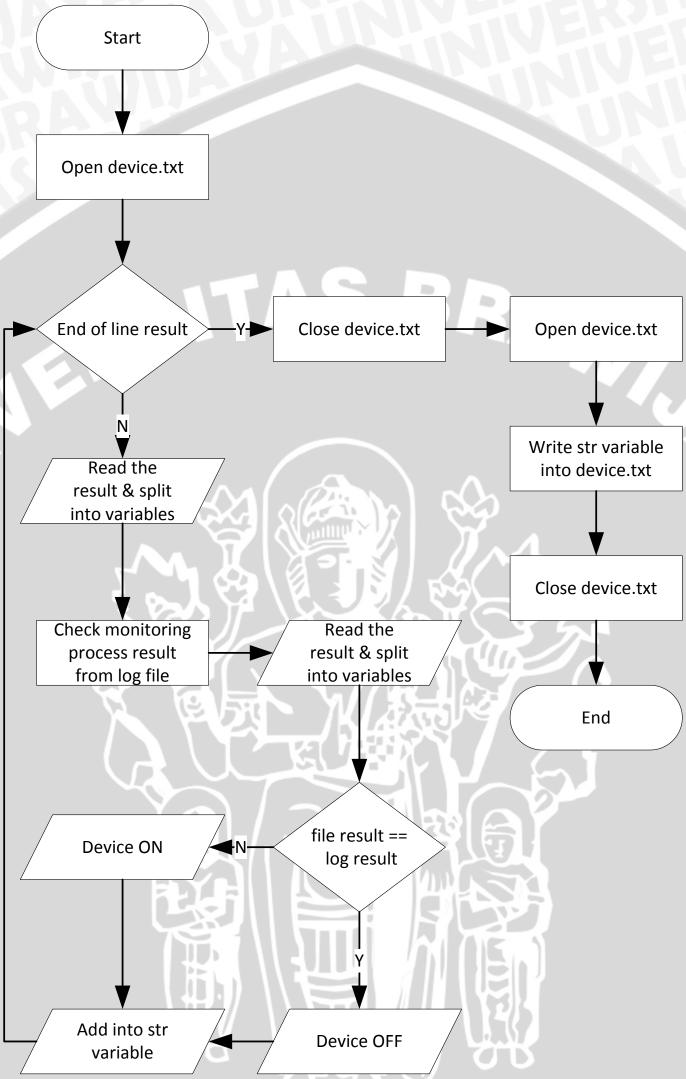


Gambar 5.22 Ilustrasi fungsi untuk me*refresh* data tabel *ARP*

Gambar 5.22 merupakan ilustrasi fungsi untuk me*refresh* data pada tabel *ARP* yang ditampilkan di *web browser*. Saat fungsi ini dipanggil maka *server* akan menjalankan perintah "*arp*" yang akan menampilkan tabel *ARP* dari *server*. Lalu *server* akan membaca tiap baris hasil dari perintah "*arp*" dan menyimpannya pada beberapa variabel. Selanjutnya menghilangkan karakter yang tidak diperlukan pada karakter *IP*. Setelah itu variabel untuk alamat *MAC* dicek apakah valid atau tidak. Jika valid data akan ditambahkan ke variabel sementara. Setelah semua baris hasil dari perintah "*arp*" dicek, maka data valid yang disimpan di variabel sementara tadi akan disimpan pada file arp.txt. Fungsi ini akan dipanggil saat tahap monitoring tengah berlangsung.

4. Fungsi untuk pengecekan status dari perangkat yang terdaftar

Fungsi ini digunakan untuk mengetahui apakah perangkat yang terdaftar pada daftar perangkat dalam keadaan bekerja atau tidak.



Gambar 5.23 Ilustrasi fungsi untuk pengecekan status dari perangkat yang terdaftar

Gambar 5.23 merupakan ilustrasi fungsi untuk pengecekan status dari perangkat yang terdaftar pada daftar perangkat. Fungsi ini membandingkan data perangkat pada file device.txt dengan file hasil proses monitoring yang disimpan pada file *log*. Pertama, *server* akan membuka file device.txt dan membaca data tiap perangkat yang ada lalu menyimpannya pada variabel sementara. Lalu hasil pembacaan dari file device.txt tadi dibandingkan dengan file *log* hasil proses monitoring. Jika hasilnya berbeda maka artinya perangkat berada dalam kondisi bekerja, jika hasilnya sama maka perangkat berada dalam kondisi tidak bekerja. Hasil pengecekan tadi akan ditambahkan sebagai informasi tambahan dari perangkat yang disimpan pada file device.txt.

5.2 Implementasi Sistem

5.2.1 Spesifikasi Perangkat Keras

Implementasi perangkat keras dalam sistem ini dibagi menjadi dua, yang pertama yaitu perangkat yang akan dimonitoring oleh *server*. Perangkat ini tidak memiliki spesifikasi khusus, melainkan beberapa spesifikasi umum seperti sebagai berikut.

- Berbasis *IP*.
- Mampu terhubung ke *server* sesuai dengan spesifikasinya, baik dengan menggunakan kabel maupun nirkabel.

Sedangkan implementasi perangkat keras yang kedua adalah pada *server*, yang akan diimplmentasikan pada *Yamaha Smart Gateway SGX808*. Berikut adalah spesifikasinya.

Tabel 3 Spesifikasi *Yamaha Smart Gateway SGX808*

Dimensi eksternal	176 mm x 32 mm x 117 mm
Berat	250 g
Power supply	AC100 sampai AC240 V (50/60 Hz)
Kunsumsi Daya	Maksimal 22 W
Lingkungan operasi	Suhu 0 – 40^{o} C Kelembaban 15 – 80 %
Lingkungan penyimpanan	Suhu -20 – 50^{o} C Kelembaban 10 – 90 %
Flash ROM	256 MB
DRAM	256 MB
WAN port	1 (10BASE-T/100BASE-TX MDI/MDIX-auto)
LAN port	4 (10BASE-T/100BASE-TX MDI/MDIX-auto)
Wireless LAN	2.4 GHz Standards: IEEE802.11b/g/n (Maximum clients 31)
USB port	1 (USB storage, USB modem)
SD Card slot	1 (Maximum 32GB)
Manajemen	*Web* Configuration, sys*log*, YMS-NMA
VPN	*IP*sec (2 destinations)
Aplikasi	Linux, Apache, MySQL, *PHP*

5.2.2 Spesifikasi Perangkat Lunak

1. ***Server***
   - *Linux*
   - *Apache*
   - *MySql*
   - *PHP*
2. **Aplikasi Monitoring**
   - *PHP*
   - *Web Browser* yang mendukung *Javascript*

5.2.3 Batasan Implementasi

Beberapa batasan yang dimiliki sistem pada tahap implementasi sistem ini adalah sebagai berikut.

- Penggunaan *ping* oleh aplikasi lain tidak diperbolehkan oleh *Yamaha Smart Gateway SGX808*, sehingga menggunakan fungsi *Network Backup* yang sudah dimodifikasi pada perangkat tersebut.
- *Firmware* yang digunakan pada penelitian ini merupakan *firmware* yang telah dimodifikasi khusus untuk penelitian ini.
- Data daftar perangkat, tabel *ARP*, serta waktu penyimpanan ke *database* yang ditampilkan pada aplikasi monitoring disimpan ke file .txt.
- Implementasi sistem dapat bekerja dimulai dari menjalankan *server Yamaha Smart Gateway SGX808*

5.2.4 Implementasi Perangkat Keras

Dalam implementasi perangkat keras ini berdasarkan pada perancangan perangkat keras yang sudah dilakukan sebelumnya. *Server* yang digunakan adalah *Yamaha Smart Gateway SGX808*. Sedangkan perangkat yang akan terhubung ke *server* ada 2, yaitu laptop yang terhubung menggunakan *ethernet* serta *smartphone* yang terhubung melalui *wifi*.

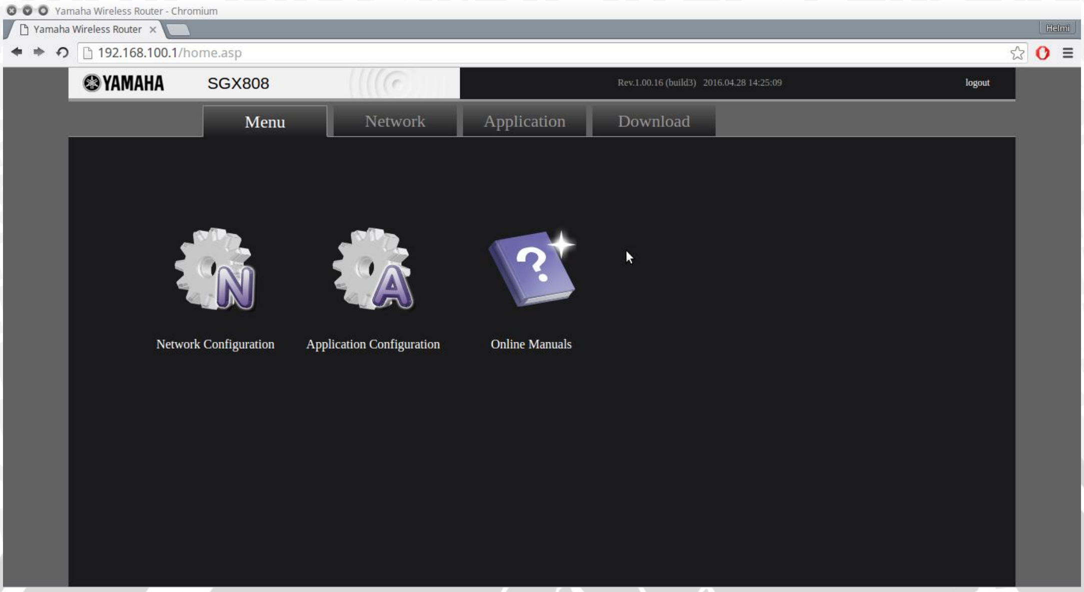
Gambar 5.24 Gambar implementasi perangkat keras

Gambar 5.24 merupakan implementasi perangkat keras yang digunakan dalam penelitian ini. Perangkat sebelah kiri adalah *server Yamaha Smart Gateway SGX808*, sedangkan perangkat yang tengah adalah *smartphone* yang akan dimonitoring serta pada sebelah kanan adalah laptop yang bertindak selain sebagai perangkat yang dimonitoring juga bertindak sebagai media yang digunakan untuk mengakses aplikasi monitoring perangkat.

5.2.5 Implementasi Lingkungan *Server*

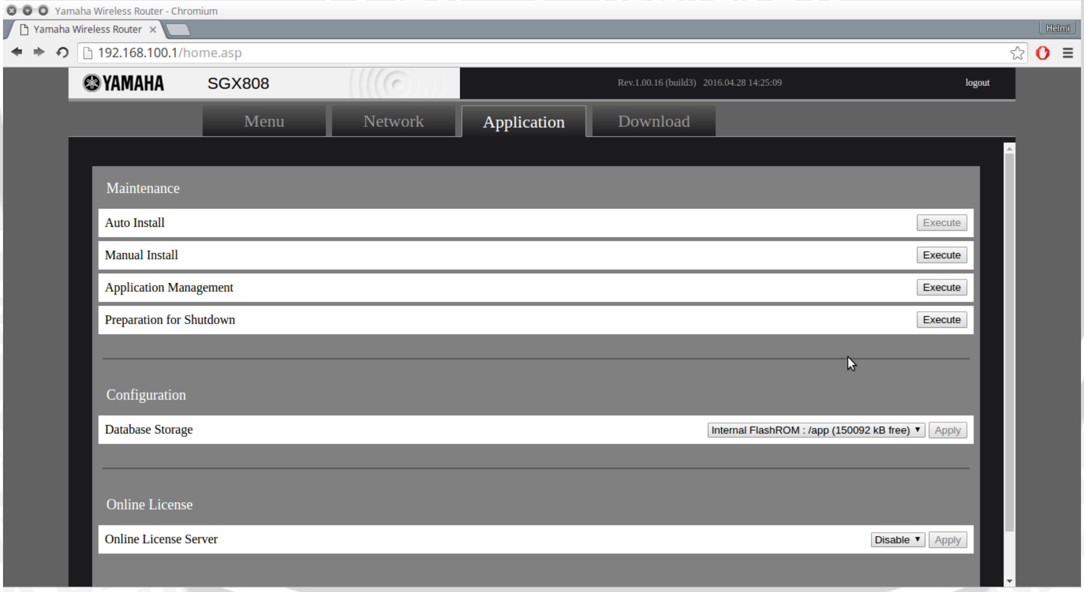
Pada implementasi lingkungan *server* dilakukan beberapa konfigurasi pada lingkungan *server* agar *server* dapat menjalankan sistem. Dimulai dengan mengubah *server Yamaha Smart Gateway SGX808* menjadi ke *Developer Mode*. Dengan menggunakan *Developer Mode* ini akan mempermudah untuk *debugging* aplikasi yang sedang dikembangkan. Normalnya, aplikasi harus diinstal terlebih dahulu agar dapat dijalankan pada *Yamaha Smart Gateway SGX808*, sehingga jika ada perubahan pada sistem maka harus dilakukan instalasi kembali pada *Yamaha Smart Gateway SGX808*.

Untuk mengaktifkan *Developer Mode*, bisa dilakukan instalasi aplikasi yang dapat diinstal melalui halaman *web* dari *Yamaha Smart Gateway SGX808*. Pada halaman *web* ini dapat dilakukan instalasi aplikasi, manajemen aplikasi, dan konfigurasi-konfigurasi lain dari perangkat ini. Berikut adalah tampilan awal dari halaman *web* tersebut.



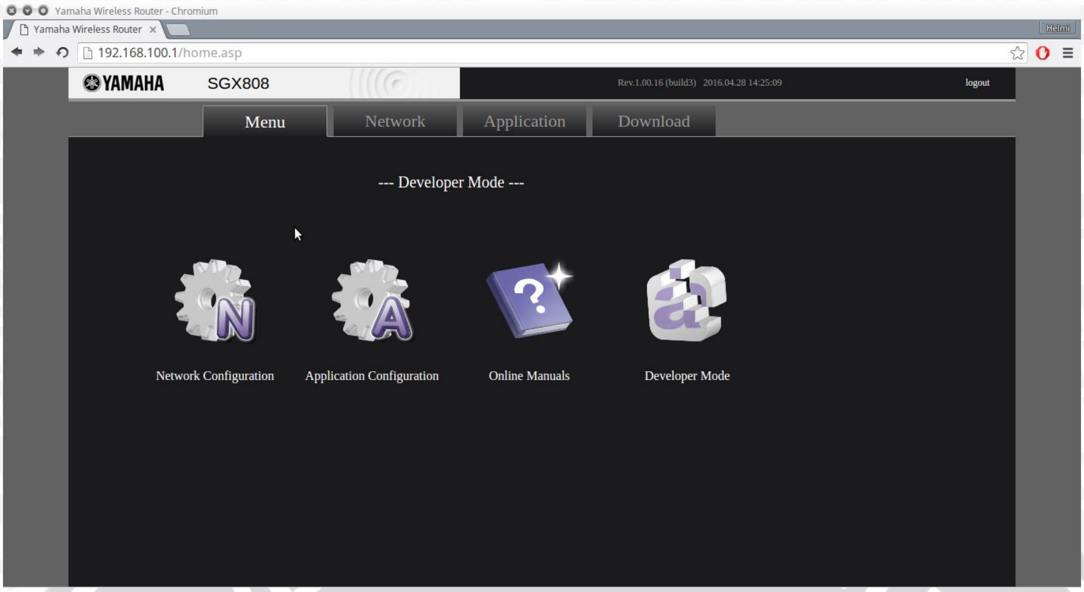
Gambar 5.25 Tampilan awal halaman *web Yamaha Smart Gateway SGX808*

Dari gambar 5.25 bisa dilihat beberapa fungsi *default* dari halaman *web Yamaha Smart Gateway SGX808*. Untuk dapat menggunakan *Developer Mode*, maka harus dilakukan instalasi aplikasi yang dapat dilakukan pada tab *Application*.



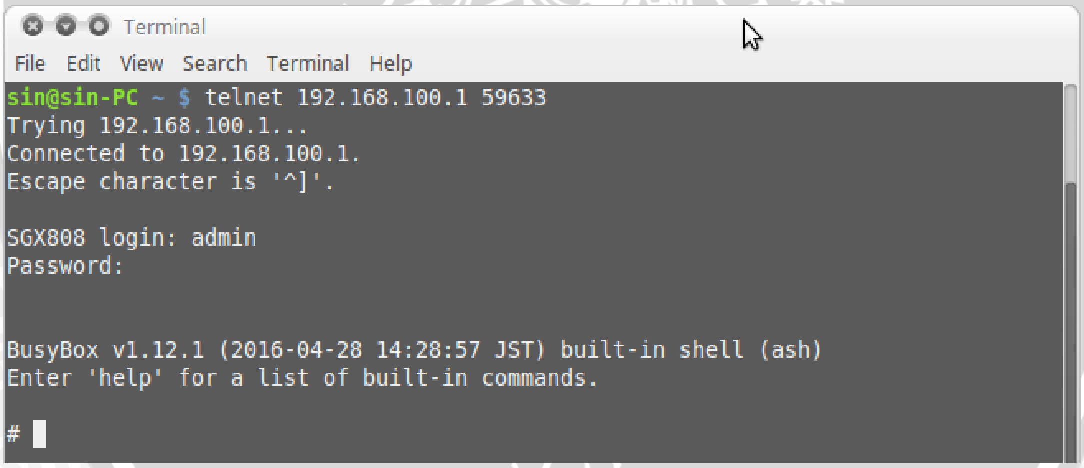
Gambar 5.26 Tampilan menu *Application* pada halaman *web*

Gambar 5.26 merupakan tampilan untuk tab *Application*. Pada bagian ini dapat dilakukan instalasi aplikasi baik secara *online* maupun *offline*, manajemen aplikasi, dan fungsi lainya. Dalam penelitian ini, *Developer Mode* diinstal secara *offline* menggunakan *manual installation*. Setelah diinstal maka aplikasi *Developer Mode* akan mucul pada halaman awal *web* dan aplikasi dapat diaktifkan.



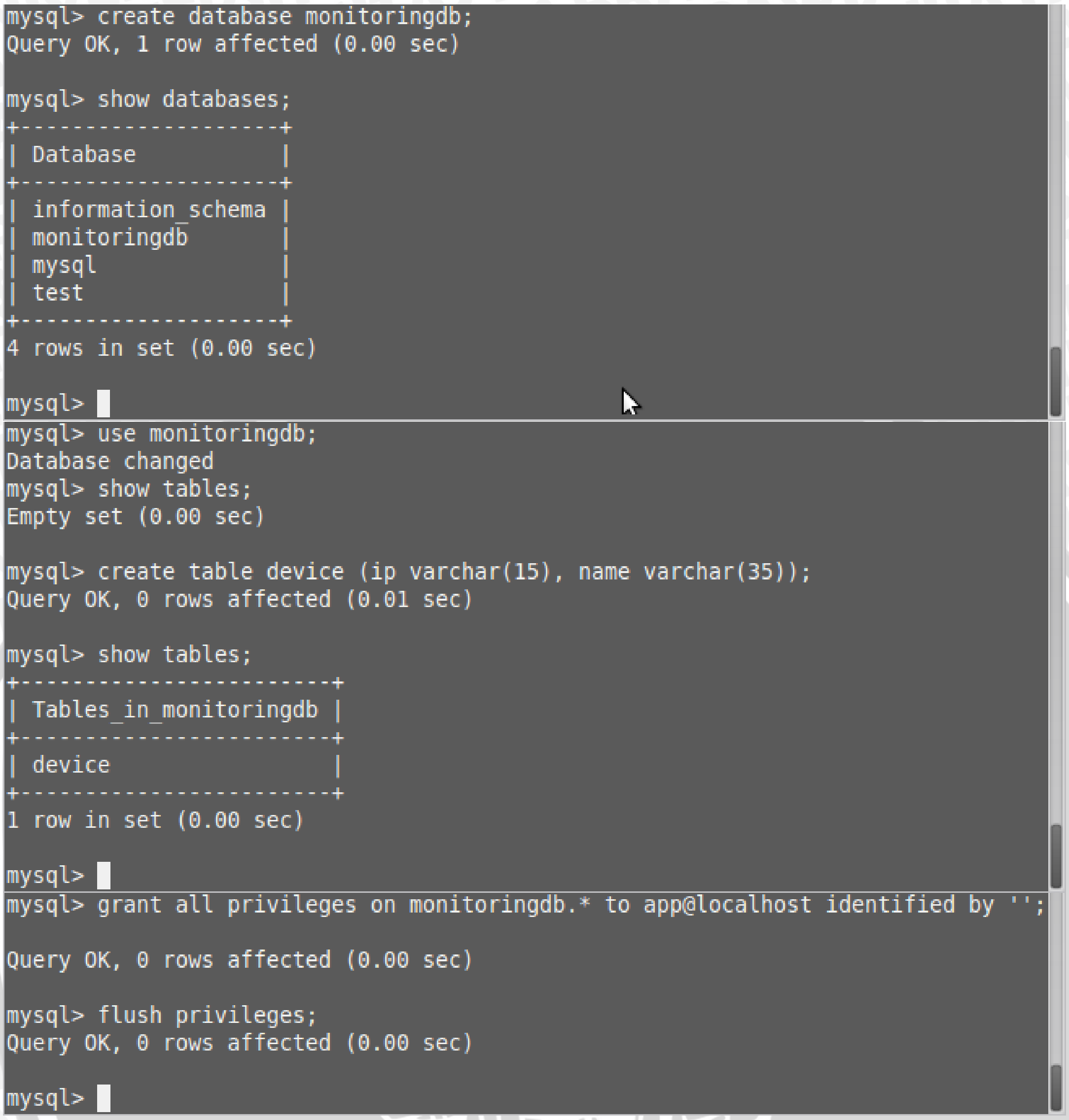
Gambar 5.27 Tampilan halaman awal setelah *Developer Mode* diaktifkan

Gambar 5.27 merupakan tampilan halaman awal setelah aplikasi *Developer Mode* sukses diinstall dan diaktifkan. Selain kemudahan *debugging*, pada *Developer Mode* juga memungkinkan untuk *terminal server* agar dapat diakses secara *remote* menggunakan *telnet*, sehingga dapat dilakukan konfigurasi selanjutnya.



Gambar 5.28 Tampilan mengakses *terminal server* menggunakan *telnet*

Pada gambar 5.28 dapat dilihat pengaksesan *terminal server* dari komputer lain menggunakan *telnet*. Setelah berhasil mengakses *terminal server*, maka selanjutnya dilakukan pembuatan *database* yang akan digunakan pada sistem ini.



```
mysql> create database monitoringdb;
Query OK, 1 row affected (0.00 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| monitoringdb       |
| mysql              |
| test               |
+--------------------+
4 rows in set (0.00 sec)

mysql> 
mysql> use monitoringdb;
Database changed
mysql> show tables;
Empty set (0.00 sec)

mysql> create table device (ip varchar(15), name varchar(35));
Query OK, 0 rows affected (0.01 sec)

mysql> show tables;
+------------------------+
| Tables_in_monitoringdb |
+------------------------+
| device                 |
+------------------------+
1 row in set (0.00 sec)

mysql> 
mysql> grant all privileges on monitoringdb.* to app@localhost identified by '';

Query OK, 0 rows affected (0.00 sec)

mysql> flush privileges;
Query OK, 0 rows affected (0.00 sec)

mysql> 
```

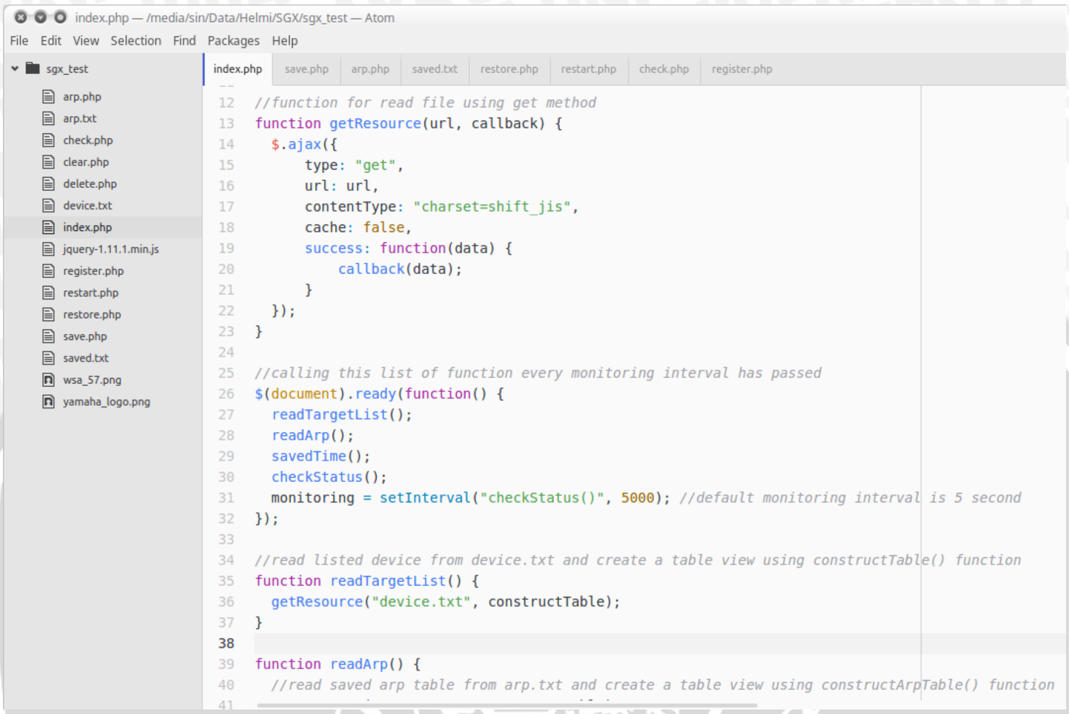
Gambar 5.29 Tampilan konfigurasi *database* yang akan digunakan

Gambar 5.29 merupakan proses pembuatan serta konfigurasi *database* yang akan digunakan. Berikut adalah penjelasanya.

- Dimulai dengan pembuatan *database* baru yaitu *monitoringdb*
- Pengecekan *database* yang baru dibuat
- Pada *database monitoringdb* dibuat tabel baru yaitu tabel perangkat yang terdiri dari dua kolom, yaitu kolom *ip* dan *nama*
- Pemberian *permission* agar aplikasi yang dibuat dapat mengakses dan menggunakan *database monitoringdb*

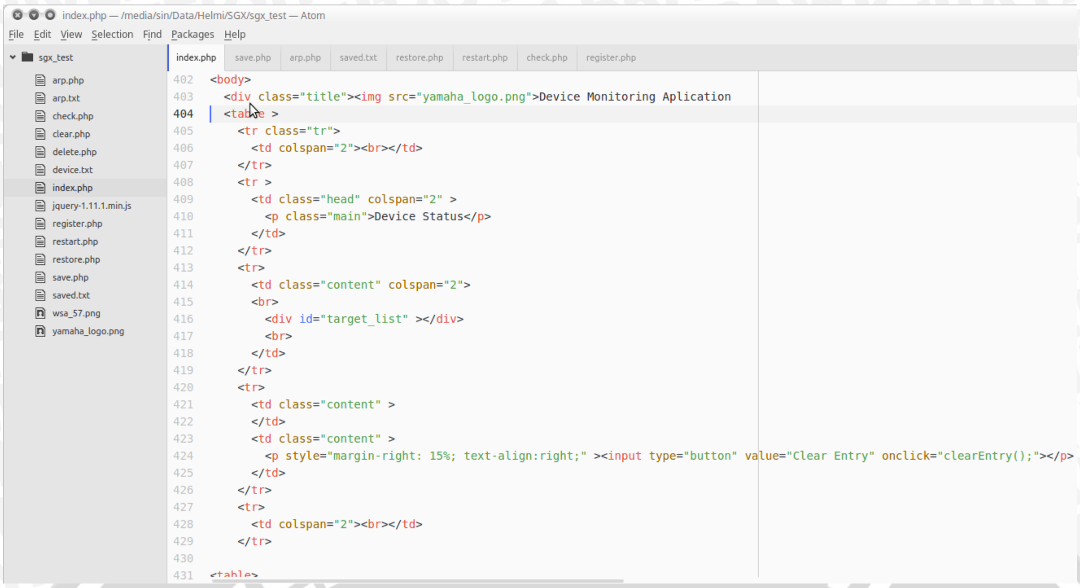
5.2.6 Implementasi Aplikasi

Tahap implementasi dilakukan dengan menerapkan perancangan yang sudah dilakukan pada tahapan sebelumnya. Dalam sistem ini penerapan diawali dengan pembuatan *source code* yang dilakukan di luar *server*. *Source code* dibuat menggunakan *Atom IDE*. Berikut adalah beberapa contohnya.



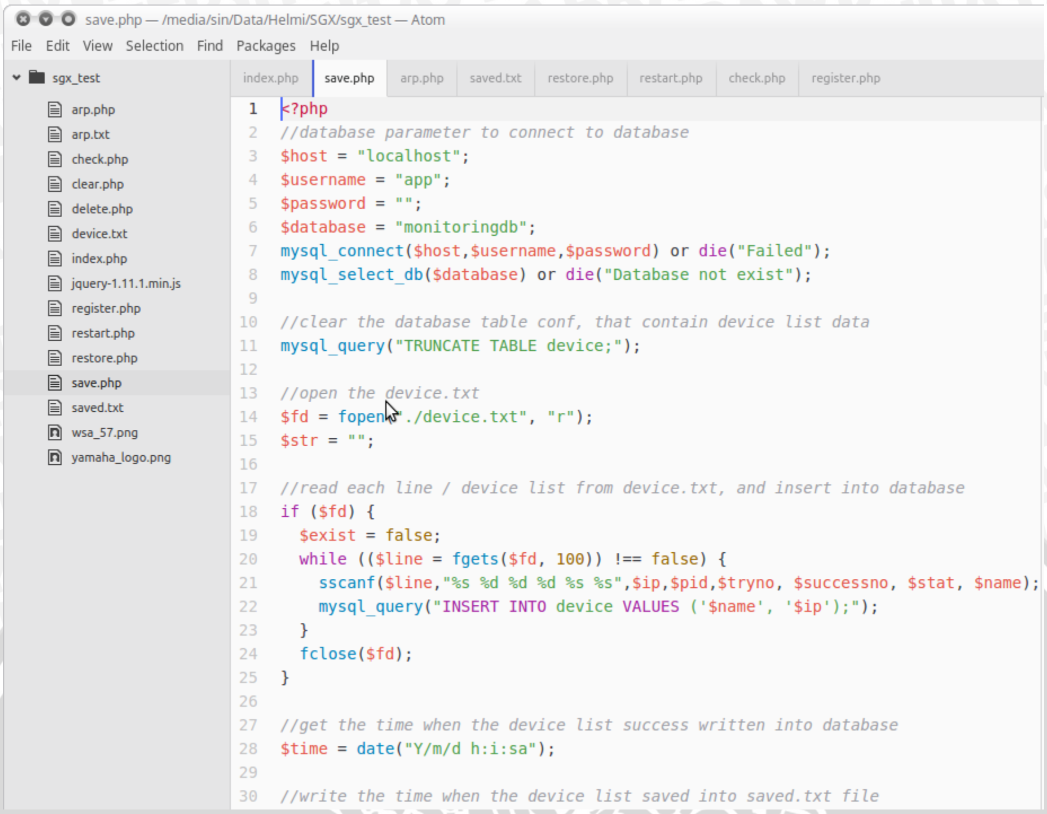
Gambar 5.30 Implementasi *Ajax* dan *Javascript* pada *source code*

Gambar 5.30 merupakan contoh implementasi *Ajax* dan *Javascript* pada sistem. *Javascript* digunakan untuk membuat fungsi yang akan digunakan oleh *web browser*. Sedangkan *Ajax* bertugas untuk menyambungkan fungsi yang dimiliki *Javascript* ke *server*. Baik *Ajax* maupun *Javascript* hanya diimplementasikan pada index.php saja.



Gambar 5.31 Implementasi *HTML* untuk tampilan halaman *web*

Gambar 5.31 merupakan implementasi *HTML* untuk tampilan halaman *web* dari aplikasi monitoning perangkat yang dibuat pada penelitian ini. Tampilan halaman *web* menggunakan *HTML* digunakan untuk desain *layout* dari aplikasi sehingga bersifat statis. Selain itu ada pula bagian *HTML* yang bersifat dinamis dengan bantuan *Javascript*, digunakan untuk menampilkan tabel dari daftar perangkat dan tabel *ARP* sehingga bersifat dinamis.

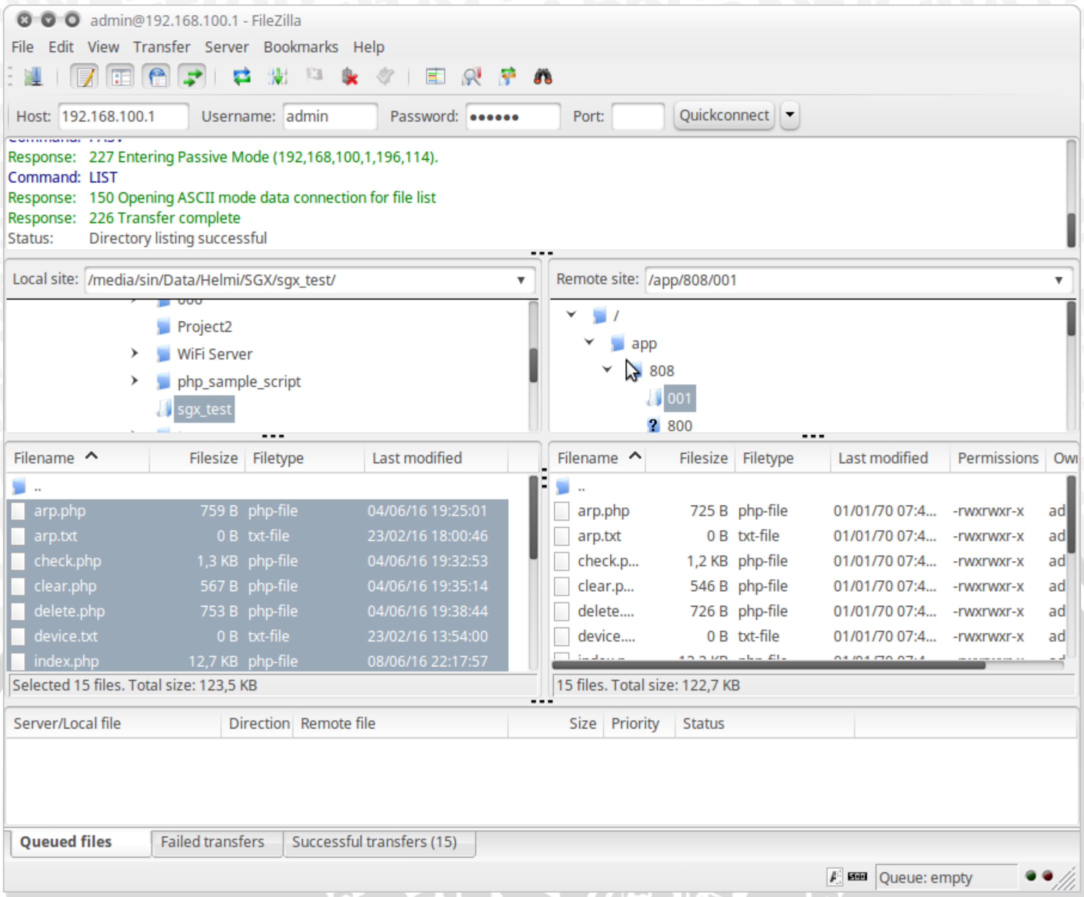



```php
<?php
//database parameter to connect to database
$host = "localhost";
$username = "app";
$password = "";
$database = "monitoringdb";
mysql_connect($host,$username,$password) or die("Failed");
mysql_select_db($database) or die("Database not exist");

//clear the database table conf, that contain device list data
mysql_query("TRUNCATE TABLE device;");

//open the device.txt
$fd = fopen("./device.txt", "r");
$str = "";

//read each line / device list from device.txt, and insert into database
if ($fd) {
  $exist = false;
  while (($line = fgets($fd, 100)) !== false) {
    sscanf($line,"%s %d %d %d %s %s",$ip,$pid,$tryno, $successno, $stat, $name);
    mysql_query("INSERT INTO device VALUES ('$name', '$ip');");
  }
  fclose($fd);
}

//get the time when the device list success written into database
$time = date("Y/m/d h:i:sa");

//write the time when the device list saved into saved.txt file
```

Gambar 5.32 Implementasi koneksi ke *database* dan *SQL Query*

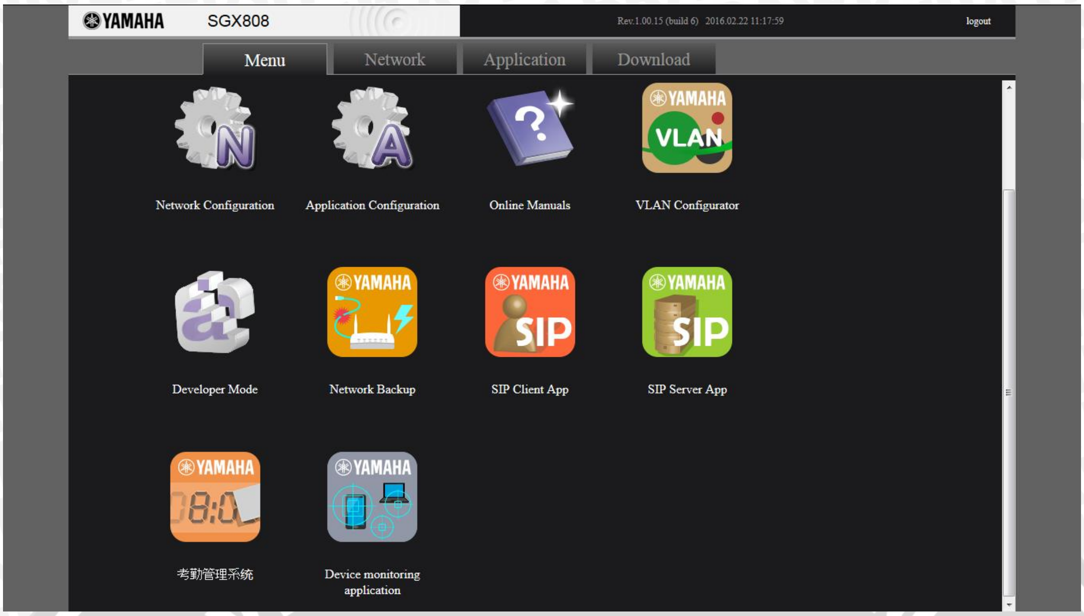
Pada gambar 5.32 menunjukkan implementasi dari koneksi ke *database* yang digunakan dan implementasi dari *SQL Query* yang digunakan untuk mengakses *database* yang digabungkan dengan *PHP*.

Selanjutnya setelah *source code* dibuat maka *source code* harus dapat dikirim ke *server* agar dapat dijalankan. Pengiriman ke *server* dalam penelitian ini dilakukan menggunakan perangkat lunak *FileZilla*.



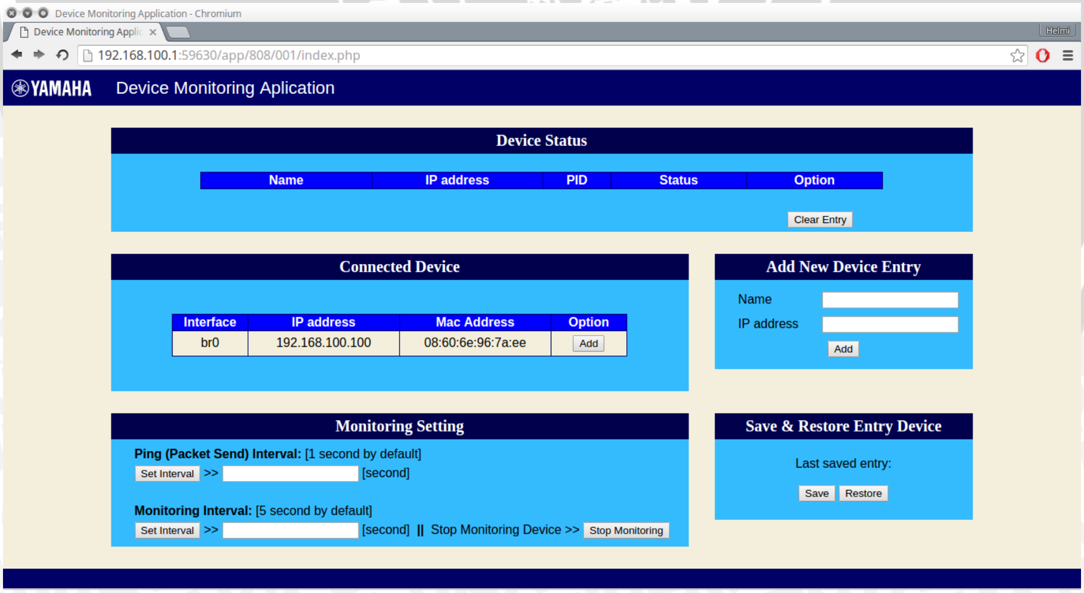
Gambar 5.33 Pengiriman *source code* ke *server*

Gambar 5.33 menunjukkan pengiriman *source code* ke *server* yang dilakukan menggunakan *FileZilla*. *Server* yang diakses sama seperti yang digunakan ketika mengakses menggunakan *telnet*, berikut juga untuk *username* serta *password*. *Source code* diletakkan dibawah folder app dan berada dibawah folder 808.



Gambar 5.34 Tampilan dari aplikasi perangkat monitoring pada halaman *web Yamaha Smart Gateway SGX808*

Gambar 5.34 merupakan tampilan dari *icon shortcut* dari aplikasi yang sudah dibuat, yaitu *Device monitoring application* yang terletak pada gambar paling bawah sebelah kanan. Untuk dapat menampilkan *shortcut* pada halaman awal seperti ini, aplikasi harus sudah diinstal pada *Yamaha Smart Gateway SGX808*. Saat *shortcut* diakses maka akan langsung menuju halaman dari aplikasi monitoring perangkat yang sudah dibuat sebelumnya.



Gambar 5.35 Tampilan halaman awal dari aplikasi monitoring perangkat

Gambar 5.35 merupakan tampilan halaman awal dari aplikasi monitoring perangkat. Dapat dilihat pada *address bar*, halaman yang diakses berada dibawah folder app dan dibawah folder 808. Pada halaman *web* aplikasi dibagi menjadi 5 bagian utama yang merupakan representasi dari tiap fungsi yang dimiliki. Berikut adalah penjelasan dari tiap bagian pada aplikasi.

- *Device Status*

  Pada bagian ini digunakan untuk menampilkan perangkat-perangkat yang terdaftar pada daftar perangkat, serta menunjukkan status dari perangkat yang terdaftar. Selain itu pada bagian ini juga terdapat fungsi untuk menghapus daftar perangkat maupun menghapus semua daftar perangkat.

- *Connected Device*

  Pada bagian ini ditampilkan informasi yang diambil dari tabel *ARP* yang dimiliki oleh *server* dan terdapat fungsi untuk menambahkan daftar perangkat dari tabel *ARP* ini.

- *Add New Device Entry*

  Pada bagian ini dapat dilakukan masukkan daftar perangkat baru yang dapat dilakukan oleh pengguna. Masukkan tersebut berupa nama dari perangkat serta alamat *IP* dari perangkat yang ingin ditambahkan ke daftar perangkat.

- *Monitoring Setting*

  Pada bagian ini terdapat fungsi untuk mengatur *interval ping* dan *interval monitoring* dari aplikasi monitoring perangkat ini. Selain itu juga terdapat fungsi untuk menghentikan aplikasi monitoring perangkat ini.

- *Save & Restore Entry Device*

  Pada bagian ini terdapat fitur yang dapat digunakan untuk menyimpan data daftar perangkat ke *database* serta memuat kembali data daftar perangkat yang sudah tersimpan ke *database*. Jika sudah ada data yang pernah disimpan ke *database*, waktu terakhir penyimpanan ke *database* akan ditampilkan pada bagian ini.

BAB 6 PENGUJIAN DAN ANALISIS

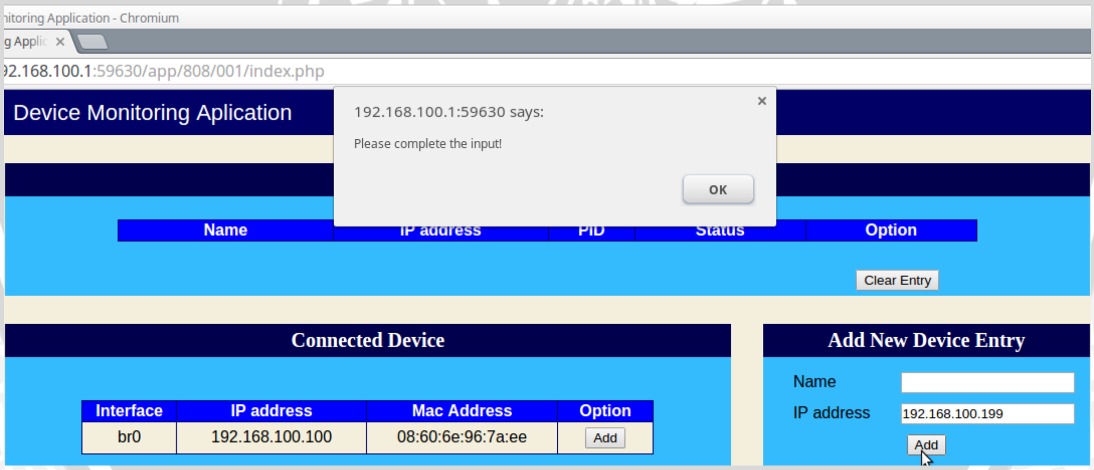
Pada bab ini akan dijelaskan mengenai proses pengujian dari sistem yang sudah melalui tahap perancangan dan implementasi. Pengujian dilakukan terhadap fungsionalitas yang dimiliki oleh sistem, untuk mengetahui bahwa sistem dapat memenuhi tujuan yang ingin dicapai dalam penelitian ini serta fungsi-fungsi dari sistem yang sudah dibuat pada tahap perancangan. Dari hasil pengujian yang dilakukan akan dilakukan analisis yang akan menghasilkan kesimpulan yang bisa diambil dari penelitian yan sudah dilakukan ini.

6.1 Pengujian terhadap fungsi manajemen daftar perangkat

Pengujian terhadap fungsi manajemen daftar perangkat ini meliputi tambah daftar perangkat (baik tambah secara manual maupun dari table *ARP*), hapus daftar perangkat, dan hapus semua daftar perangkat.

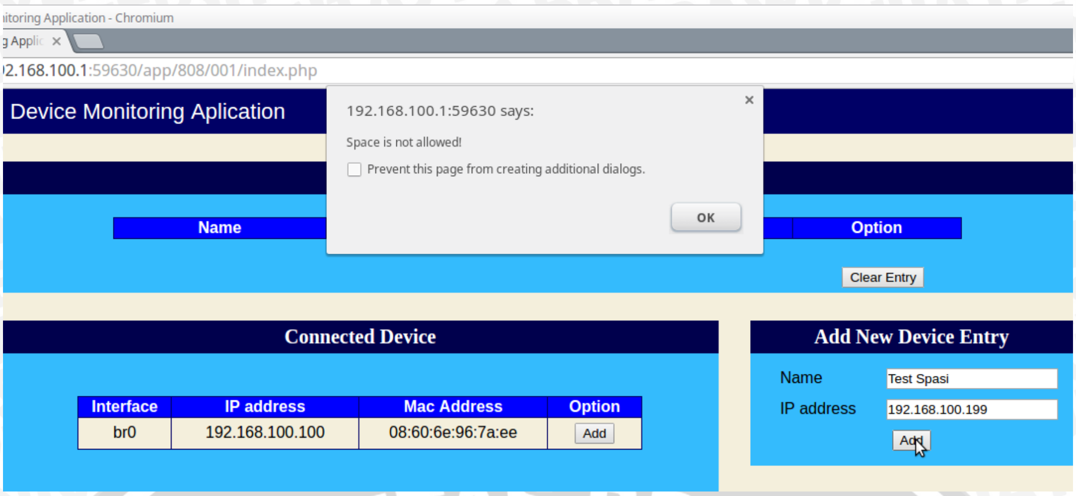
6.1.1 Tambah daftar perangkat

Pada fitur ini bertujuan agar sistem dapat melakukan penambahan daftar perangkat, yang disimpan pada file device.txt pada *server*, berdasarkan masukkan dari pengguna. Pada fitur ini juga dilengkapi beberapa proses untuk pengecekan masukkan dari pengguna agar tidak mengganggu kinerja dari sistem. Berikut adalah hasil pengujian dari fitur tambah daftar perangkat.



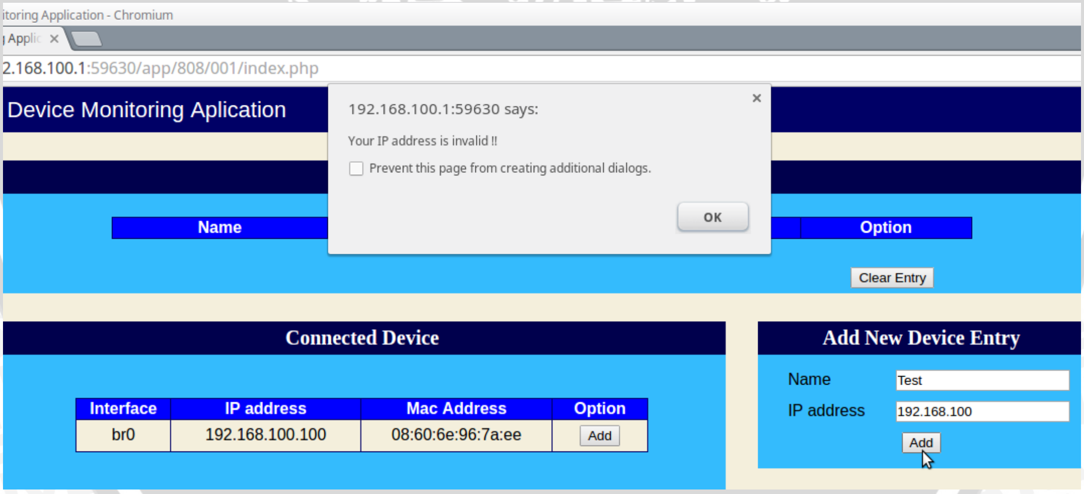
Gambar 6.1 Tampilan peringatan pada aplikasi jika masukkan nama bernilai kosong

Gambar 6.1 merupakan tampilan peringatan pada aplikasi jika masukkan nama perangkat bernilai kosong. Nama perangkat tidak boleh bernilai kosong karena data daftar perangkat akan disimpan dalam bentuk *plain text* dan diakses sesuai urutan. Jika nama perangkat kosong, maka akan mengganggu dalam pembacaan data perangkat pada daftar perangkat.



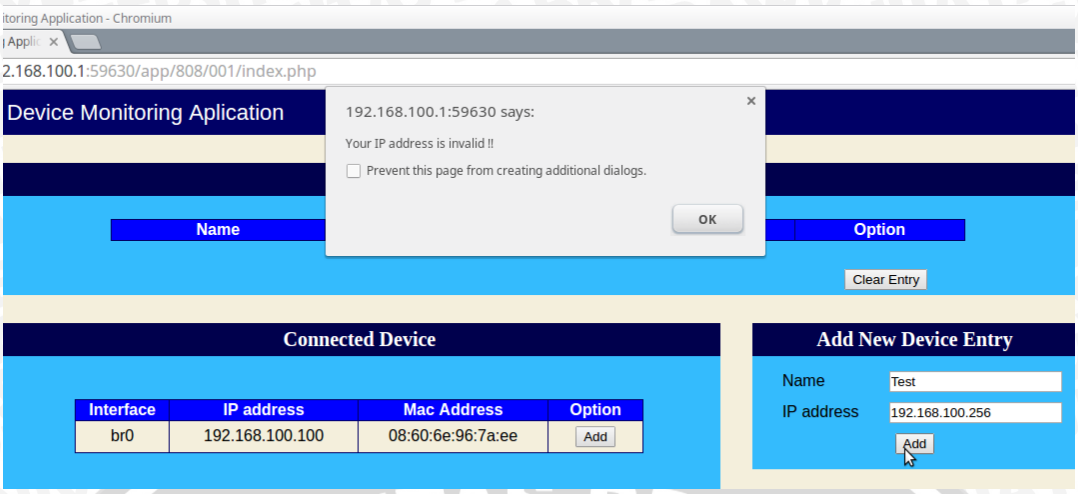
Gambar 6.2 Tampilan peringatan pada aplikasi jika masukkan nama menggunakan spasi

Gambar 6.2 merupakan tampilan peringatan jika masukkan nama perangkat menggunakan karakter spasi. Nama perangkat tidak boleh menggunakan spasi karena data daftar perangkat disimpan pada file device.txt menggunakan pemisah karakter spasi untuk setiap informasinya. Jika ada informasi dengan karakter spasi maka akan mengganggu proses pembacaan informasi perangkat pada daftar perangkat.



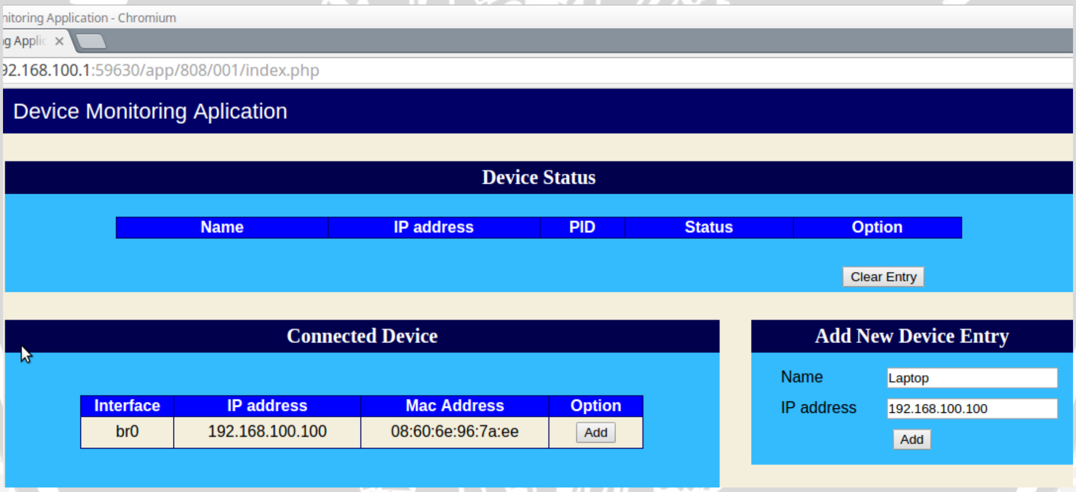
Gambar 6.3 Tampilan peringatan pada aplikasi jika masukkan *IP* tidak valid

Gambar 6.3 merupakan tampilan peringatan jika masukkan alamat *IP* address tidak valid karena alamat *IP* hanya terdiri dari *3 oktet*. Alamat *IP* akan bernilai valid apabila terdiri dari *4 oktet*.



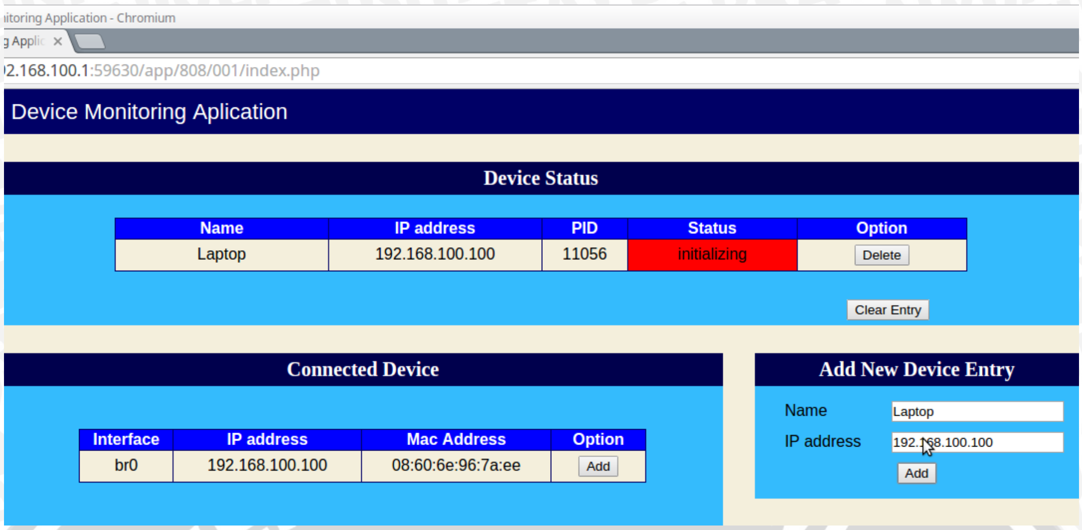
Gambar 6.4 Tampilan peringatan pada aplikasi jika masukkan *IP* tidak valid (2)

Gambar 6.4 merupakan tampilan peringatan jika masukkan alamat *IP* tidak valid karena jangkauan alamat melebihi jumlah maksimal. Pada alamat *IP* dianggap valid apabila berada pada jangkauan 0 – 255.



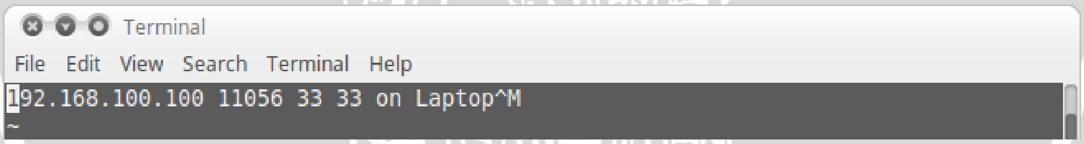
Gambar 6.5 Tampilan aplikasi saat input daftar perangkat baru

Gambar 6.5 merupakan tampilan saat melakukan masukkan daftar perangkat baru secara manual dari bagian *Add New Device Entry*. Perangkat yang didaftarkan ini adalah Laptop yang terhubung ke *server* menggunakan *ethernet*. Pada form masukkan daftar perangkat dapat dilihat daftar perangkat baru dengan nama perangkat Laptop serta alamat *IP* 192.168.100.100.



Gambar 6.6 Tampilan aplikasi saat daftar perangkat baru berhasil ditambahkan

Gambar 6.6 merupakan tampilan saat daftar perangkat baru berhasil ditambahkan ke *server*. Pada bagian Device Status akan ada informasi perangkat baru yaitu Laptop pada alamat *IP* 192.168.100.100. Pada tabel ini juga ditampilkan informasi dari *PID* proses monitoring untuk perangkat tersebut agar apabila aplikasi ini mengalami masalah pada proses monitoring tetap dapat dimatikan secara manual menggunakan *PID*. Selain itu ada informasi status dari perangkat, pada tahap awal akan dilakukan inisialisasi sebelum perangkat ditentukan dalam kedaan aktif atau tidak.



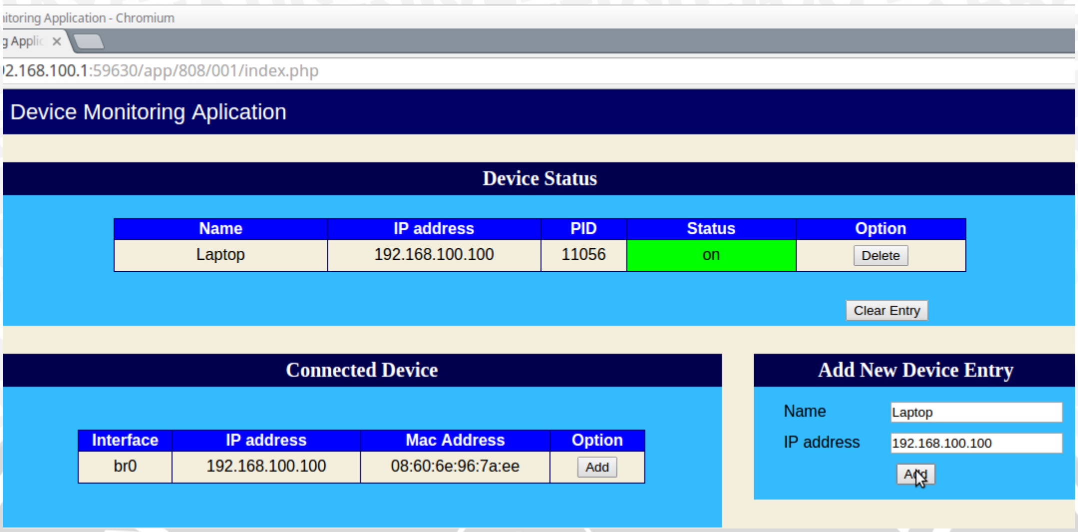
Gambar 6.7 Isi file device.txt saat daftar perangkat berhasil ditambahkan

Gambar 6.7 merupakan isi dari file device.txt saat daftar perangkat Laptop berhasil ditambahkan. Data disimpan dengan format sebagai berikut.

[**Alamat*IP***] [**PID**] [**PingTryNo**] [**PingSuccessNo**] [**Status**] [**NamaPerangkat**]

- **Alamat*IP***: Alamat *IP* dari perangkat
- **PID**: *PID* dari proses monitoring untuk perangkat
- **PingTryNo**: Jumlah total *ping* yang sudah dikirim ke alamat *IP* perangkat
- **PingSuccessNo**: Jumlah total *ping* yang sukses dikirim ke alamat *IP* perangkat
- **Status**: Status dari perangkat
- **NamaPerangkat**: Nama dari perangkat sesuai dengan inputan pengguna

Pemisah untuk tiap informasi dari perangkat yang terdaftar menggunakan karakter spasi.

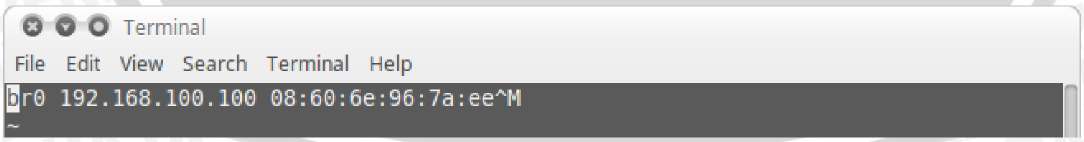


Gambar 6.8 Tampilan aplikasi saat perangkat yang sudah terdaftar didaftarkan kembali

Gambar 6.8 merupakan tampilan saat ada perangkat yang sudah terdaftar pada daftar perangkat didaftarkan kembali. Pada saat hal tersebut maka tidak akan terjadi apapun, karena pada saat penambahan perangkat akan dicek terlebih dahulu apakah perangkat yang akan didaftarkan sudah ada atau belum. Pengecekan perangkat sudah terdaftar atau belum dilakukan menggunakan alamat *IP*, karena proses monitoring dijalankan berdasarkan alamat *IP* dari perangkat. Hal ini dilakukan agar tidak ada proses monitoring yang berjalan bersamaan untuk perangkat yang sama, agar tidak saling mengganggu.

6.1.2 Tambah daftar perangkat dari tabel *ARP*

Pada fitur ini bertujuan agar pengguna dapat memasukkan data daftar perangkat baru dari tabel *ARP*. Hal ini akan memudahkan pengguna untuk memasukkan data karena terkadang sulit untuk mengetahui alamat *IP* dari perangkat tersebut.



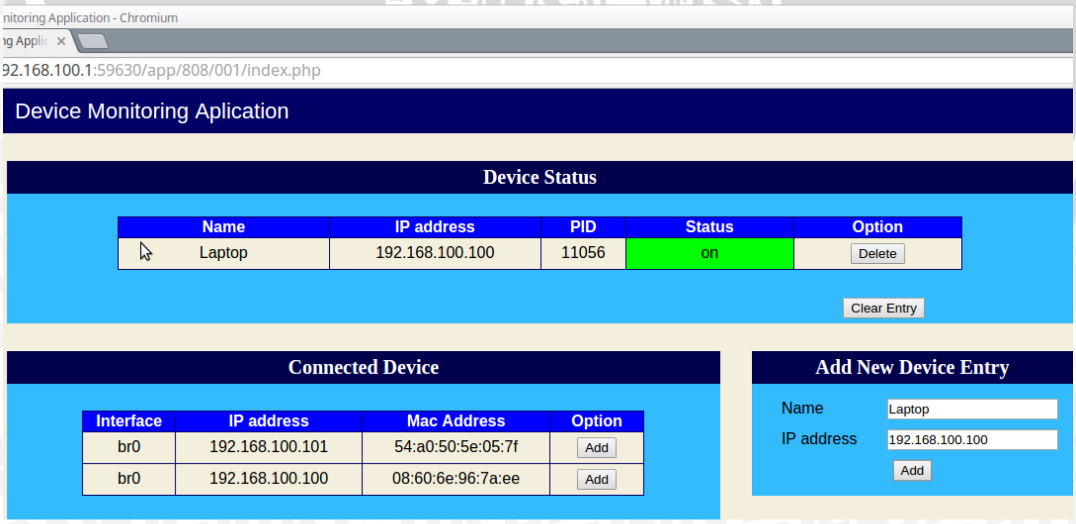
Gambar 6.9 Isi file arp.txt saat hanya laptop yang terhubung

Gambar 6.9 merupakan tampilan file arp.txt saat hanya ada satu perangkat, yaitu Laptop, yang terhubung ke *server*. Data pada arp.txt ini disimpan dengan format sebagai berikut.

[**interface**] [**alamat*IP***] [**alamatMAC**]

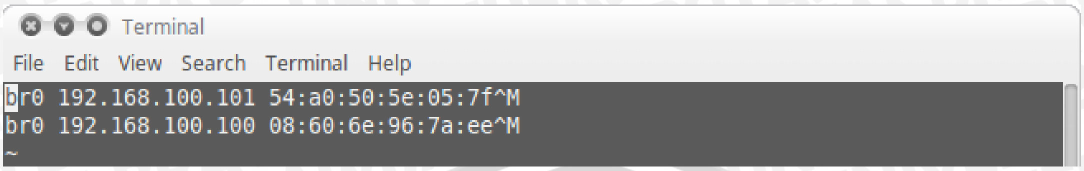
Gambar 6.10 Tampilan informasi perangkat yang akan dihubungkan ke *server*

Selanjutnya untuk pengujian penambahan daftar perangkat dari tabel *ARP* akan digunakan *smartphone* yang terhubung ke *server* menggunakan *wifi*. Gambar 6.10 diatas merupakan tampilan informasi dari *smartphone* yang sudah terhubung ke *server* dengan mendapatkan alamat *IP* 192.168.100.101 serta dengan alamat *MAC* 54:a0:50:5e:05:7f.



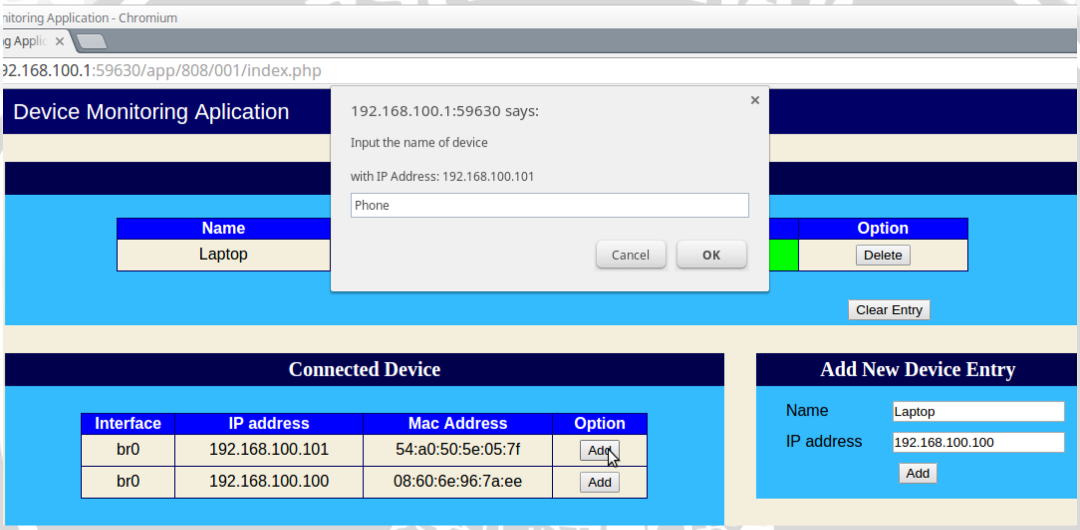
Gambar 6.11 Tampilan aplikasi saat ada perangkat baru yang terhubung

Setelah *smartphone* terhubung ke *server*, tampilan tabel *ARP* pada aplikasi akan mengalami *update* dengan munculnya informasi dari *smartphone* yang baru terhubung seperti pada gambar 6.11. Dapat dilihat jika informasi yang dimiliki sama dengan informasi yang ada pada *smartphone*, yaitu alamat *IP* 192.168.100.101 serta dengan alamat *MAC* 54:a0:50:5e:05:7f



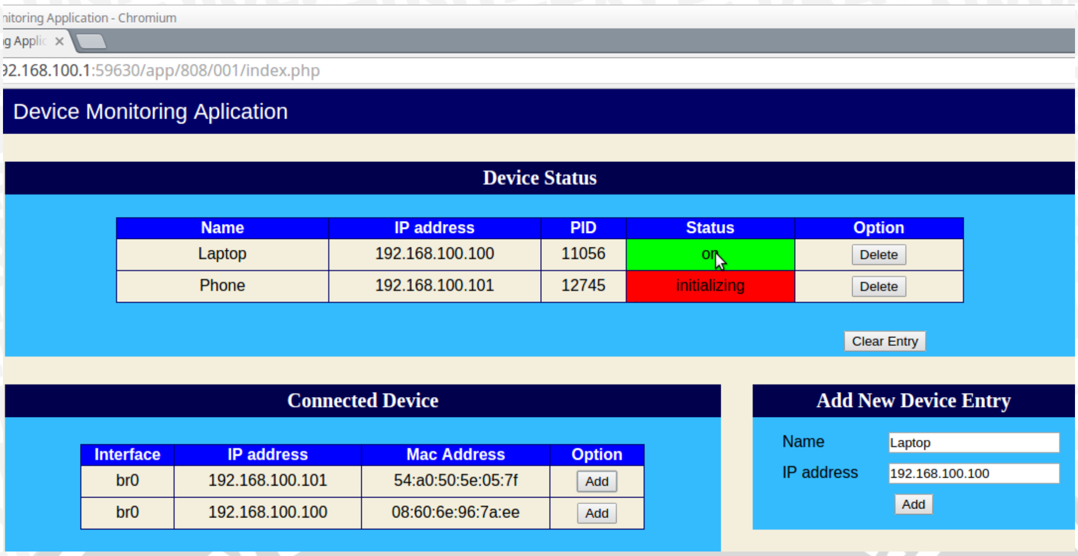
Gambar 6.12 Isi file arp.txt saat ada perangkat baru yang terhubung

Tampilan tabel *ARP* akan berubah karena data *ARP* yang disimpan di *server* akan ter*update* setelah ada koneksi baru. Dapat dilihat dari gamber 6.12 yang merupakan isi dari file arp.txt setelah *smartphone* yang baru terhubung informasinya ditambahakan pada file teresebut.



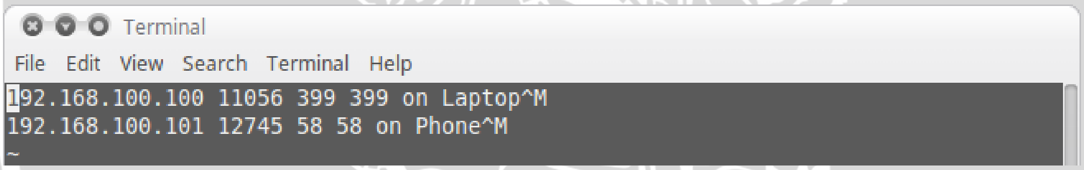
Gambar 6.13 Tampilan aplikasi saat menambahkan daftar perangkat baru dari tabel *ARP*

Pada informasi tabel *ARP* terdapat pilihan untuk menambahkan perangkat ke dalam daftar perangkat. Dari gambar 6.13 menunjukkan tampilan saat menambahkan daftar perangkat baru dari tabel *ARP*. Jika menggunakan pilihan ini pengguna hanya tinggal memasukkan nama perangkat untuk menambahkan perangkat tersebut ke daftar perangkat.



Gambar 6.14 Tampilan aplikasi saat daftar perangkat baru berhasil ditambahkan

Gambar 6.14 menunjukkan tampilan dari aplikasi saat daftar perangkat baru berhasil ditambahkan. Status perangkat akan menunjukkan status inisialisasi, karena status dari perangkat ini belum dapat diketahui karena baru ditambahkan.

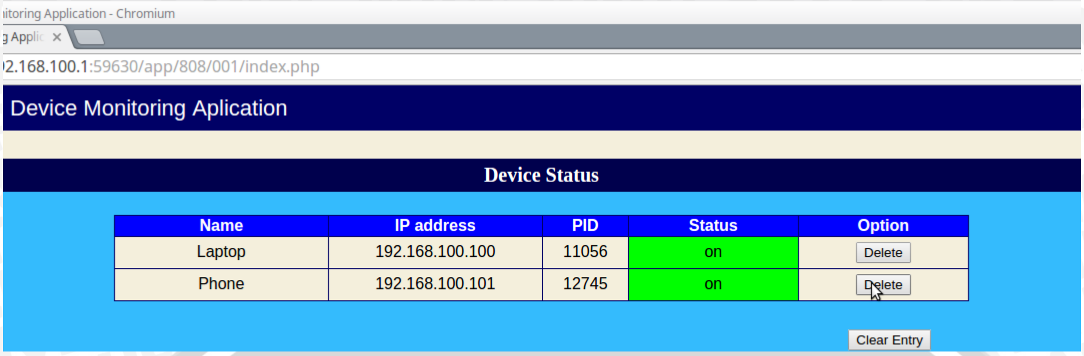


Gambar 6.15 Isi file device.txt saat daftar perangkat baru ditambahkan

Daftar perangkat yang dimasukkan tentunya akan mengupdate data dari file device.txt yang merupakan tempat untuk menyimpan informasi perangkat yang terdaftar ke *server*. dapat dilihat dari gambar 6.15 data untuk *smartphone* berhasil ditambahkan ke file device.txt

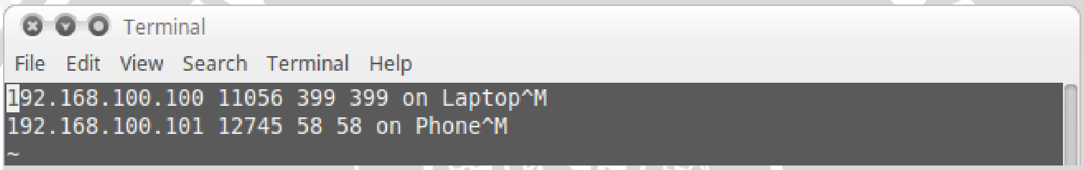
6.1.3 Hapus daftar perangkat

Pada fitur ini bertujuan untuk menghapus data dari sebuah perangkat dari daftar perangkat. Hal ini dilakukan agar proses monitoring dari perangkat yang sudah tidak digunakan dapat dihilangkan untuk menghemat memori yang digunakan untuk proses monitoring.



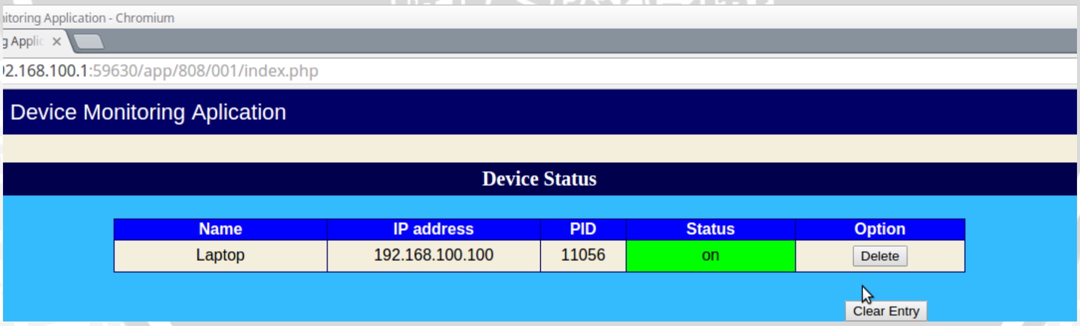
Gambar 6.16 Tampilan aplikasi daftar perangkat yang akan dihapus

Gambar 6.16 menunjukkan tampilan perangkat yang akan dihapus. Pada pengujian ini data perangkat yang akan dihapus adalah data dari *smartphone*.



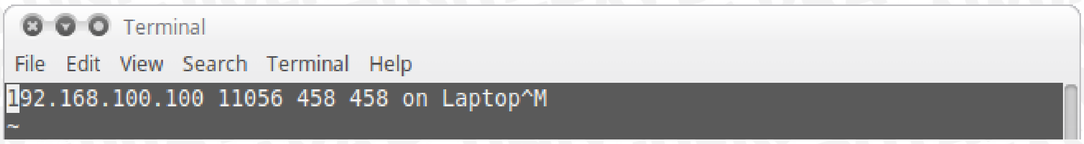
Gambar 6.17 Isi file device.txt sebelum perangkat dihapus

Dalam penghapusan data dari perangkat tentu akan mengubah informasi dari file device.txt. Gambar 6.17 merupakan tampilan file device.txt sebelum data *smartphone* dihapus.



Gambar 6.18 Tampilan aplikasi setelah daftar perangkat *smartphone* dihapus

Setelah data daftar perangkat dihapus maka informasi dari perangkat yang dihapus tidak akan ditampilkan lagi pada bagian perangkat status. Gambar 6.18 menunjukkan bahwa informasi dari perangkat *smartphone* tidak ditampilkan karena sudah dihapus.

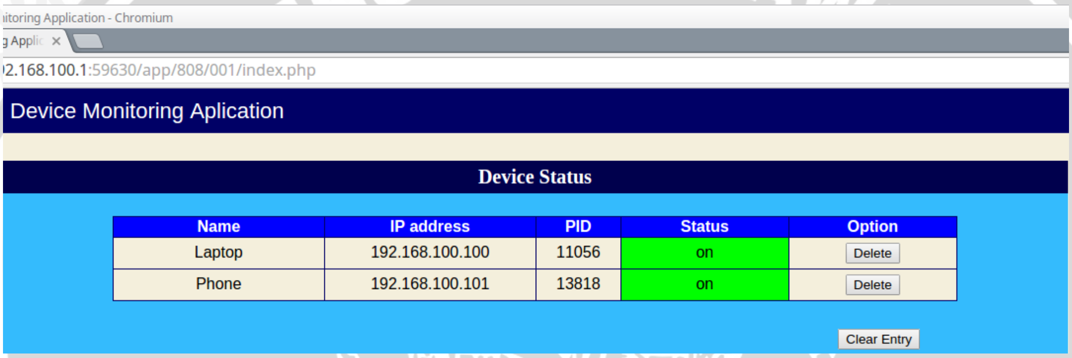


Gambar 6.19 Isi file device.txt setelah daftar perangkat dihapus

Setelah data dihapus maka file device.txt juga akan ikut ter-*update*. Dapat dilihat dari gambar 6.19 dimana informasi dari perangkat *smartphone* sudah tidak ada.

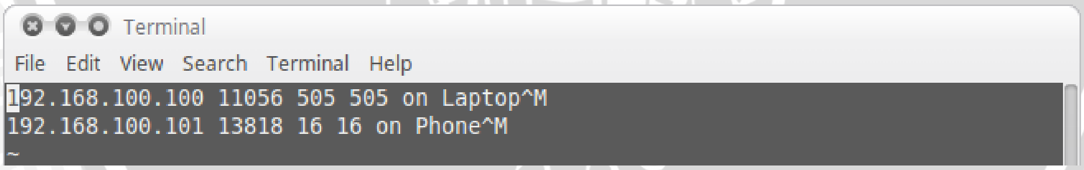
6.1.4 Hapus semua daftar perangkat

Pada fitur ini bertujuan untuk menghentikan semua proses monitoring dari semua perangkat yang terdaftar serta menghapus semua informasi dari perangkat dari daftar perangkat.



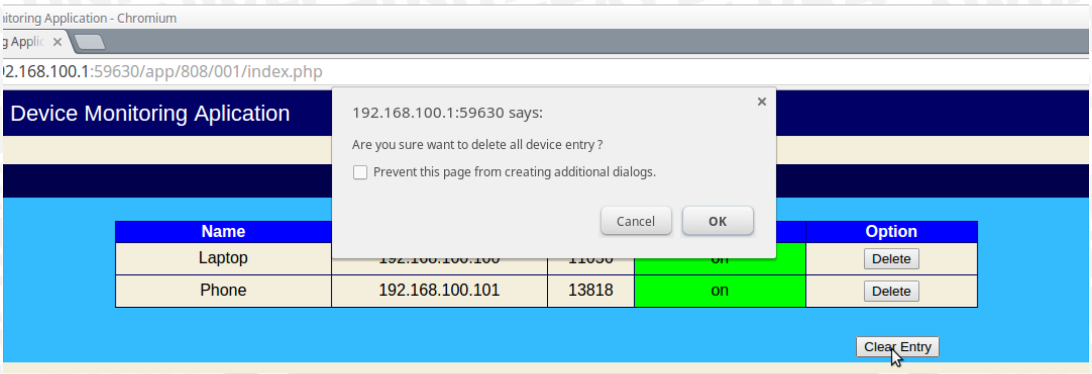
Gambar 6.20 Tampilan aplikasi daftar perangkat sebelum dihapus

Gambar 6.20 merupakan tampilan dari daftar perangkat sebelum data akan dihapus semua menggunakan fitur ini.



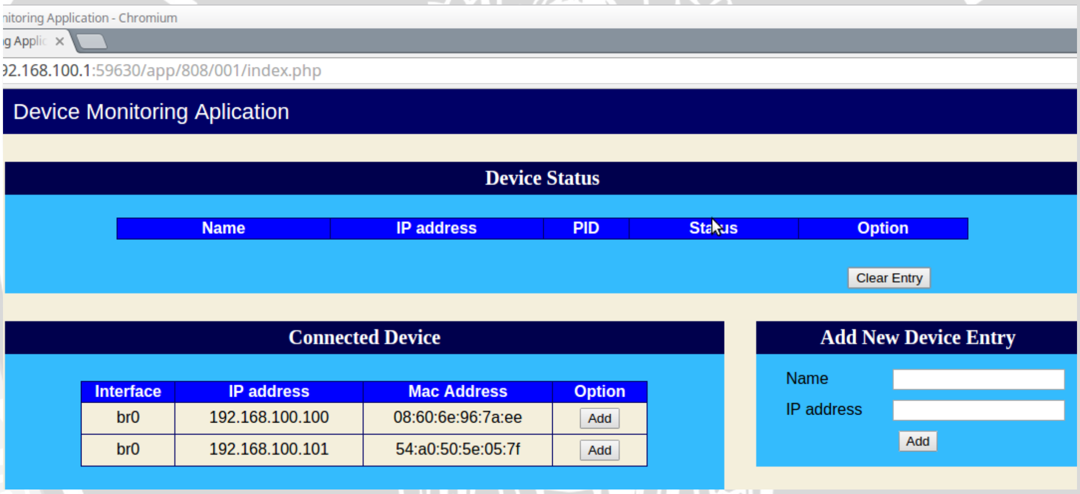
Gambar 6.21 Isi file device.txt sebelum semua entr perangkat dihapus

Gambar 6.21 merupakan tampilan informasi perangkat yang terdapat pada file device.txt yang akan dihapus semua.



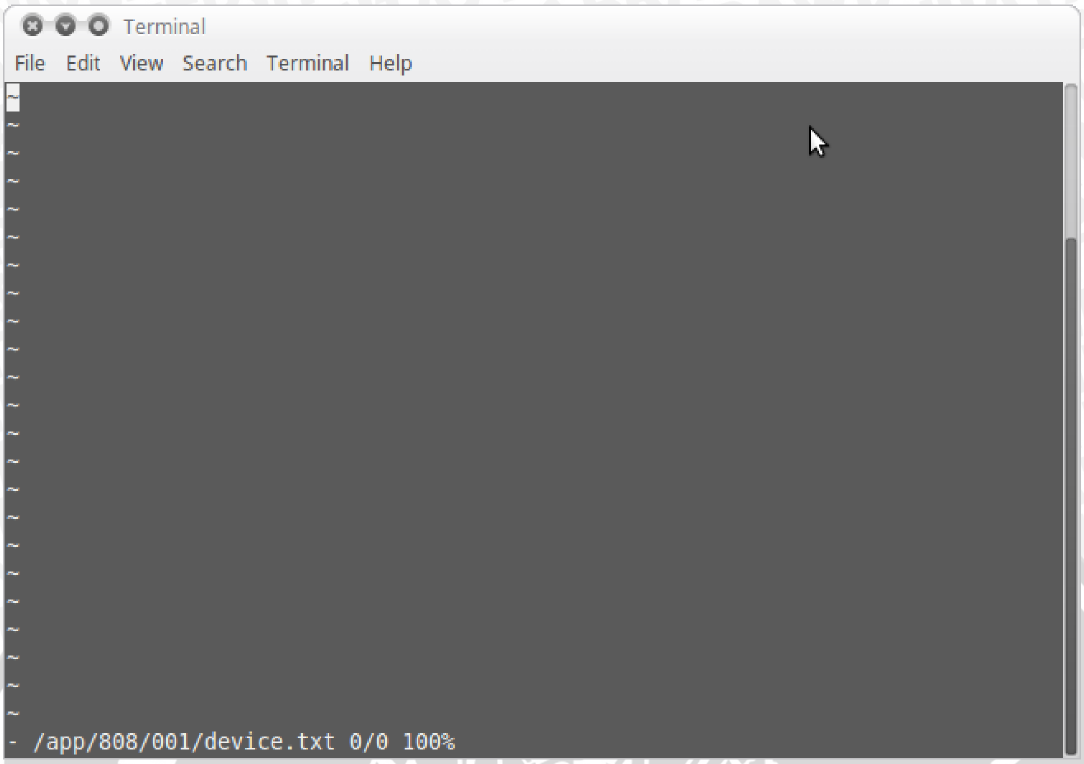
Gambar 6.22 Tampilan konfirmasi pada aplikasi saat daftar semua perangkat akan dihapus

Pada fitur untuk menghapus semua informasi perangkat ini bersifat kritis, maka dari itu konfirmasi dari pengguna diperlukan pada fitur ini. Pada gambar 6.22 menunjukkan tampilan konfirmasi saat fitur hapus semua perangkat ini akan digunakan.



Gambar 6.23 Tampilan aplikasi setelah semua daftar perangkat terhapus

Jika dikonfirmasi maka data dari semua perangkat akan dihapus sehingga berikut juga proses monitoring untuk setiap perangkat. Gambar 6.23 menunjukkan tampilan aplikasi setelah semua data perangkat dihapus.

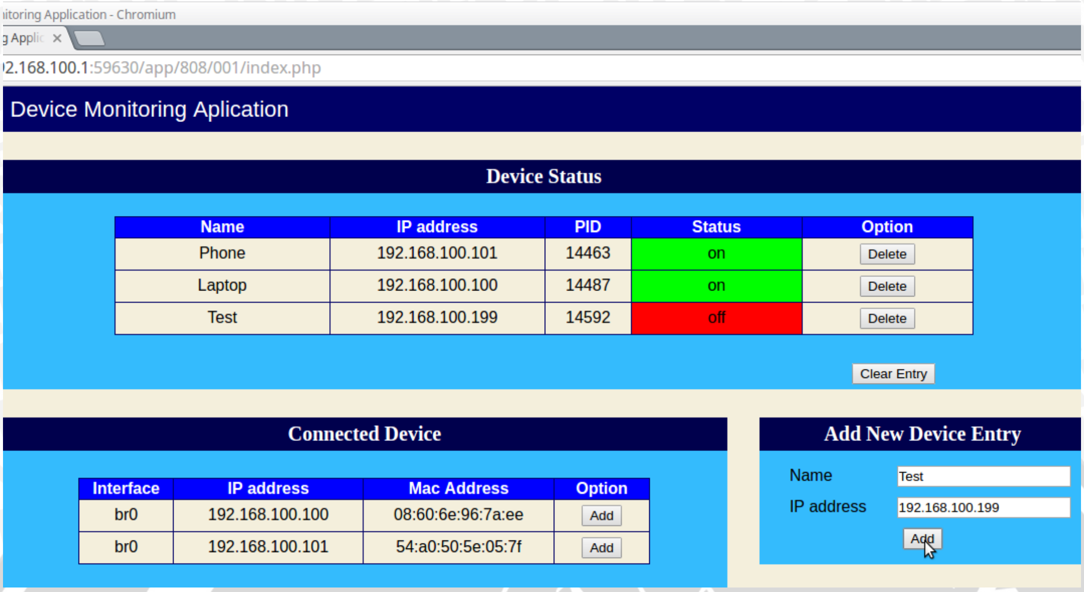


Gambar 6.24 Isi file device.txt setelah semua daftar perangkat terhapus

Saat menghapus semua daftar perangkat maka data pada file device.txt akan dihapus juga. Pada gambar 6.24 dapat dilihat tampilan dari file device.txt yang kosong setelah data daftar perangkat dihapus semua.

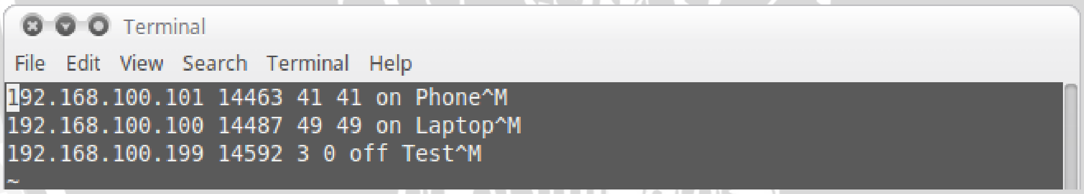
6.2 Pengujian fungsi monitoring terhadap perangkat-perangkat yang terdaftar dalam daftar perangkat

Fungsi monitoring terhadap perangkat-perangkat yang terdaftar bertujuan untuk menunjukkan status dari perangkat tersebut, apakah perangkat tersebut bekerja ataupun tidak. Pada tampilan informasi status perangkat akan ditampilkan dengan menggunakan warna yang berbeda agar lebih terlihat. Jika perangkat aktif maka akan ditampilkan dengan warna hijau, sedangkan jika tidak maka akan ditampilkan dengan warna merah.



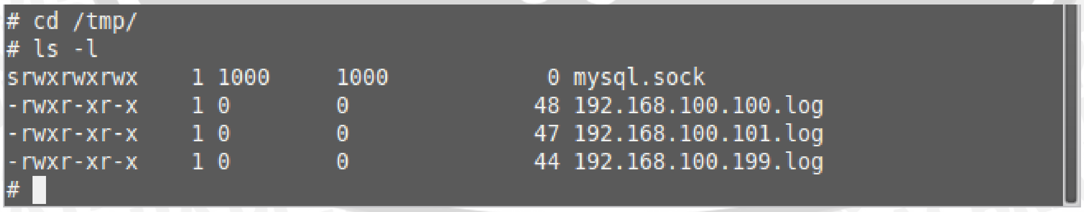
Gambar 6.25 Tampilan aplikasi perangkat yang terdaftar ke server

Gambar 6.25 menunjukkan tampilan dari perangkat yang terdaftar ke *server*. Pada pengujian ini ditambahkan informasi perangkat baru yaitu Test, menggunakan alamat *IP* yang tidak aktif sebagai contoh untuk monitoring terhadap perangkat yang tidak aktif. Sedangkan dua perangkat lainnya berstatus aktif, yaitu Laptop yang terhubung menggunakan *ethernet* serta Phone yang terhubung menggunakan *wifi*.



Gambar 6.26 Isi file device.txt untuk perangkat yang terdaftar

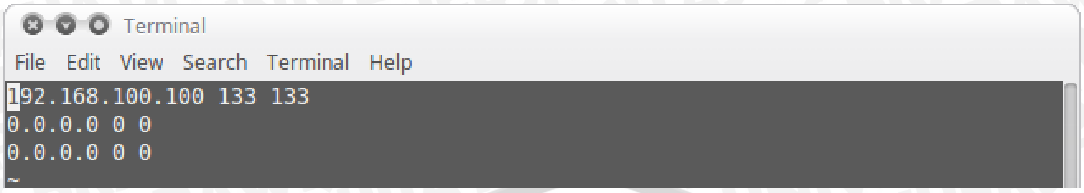
Gambar 6.26 menunjukkan informasi yang tersimpan pada file device.txt untuk perangkat-perangkat yang digunakan dalam pengujian ini.



Gambar 6.27 Tampilan daftar file *log* dari proses monitoring

Untuk setiap perangkat yang terdaftar pada daftar perangkat maka akan dilakukan proses monitoring untuk setiap perangkat tersebut. Hasil dari proses monitoring akan disimpan pada file *log* yang terletak dibawah folder /tmp dengan nama **alamat*IP.log***. Pada gambar 6.27 menunjukkan tampilan ini dari folder /tmp.

Dalam folder tersebut terdapat tiga buah file *log* dari proses monitoring, sesuai dengan perangkat yang terdaftar sebelumnya.

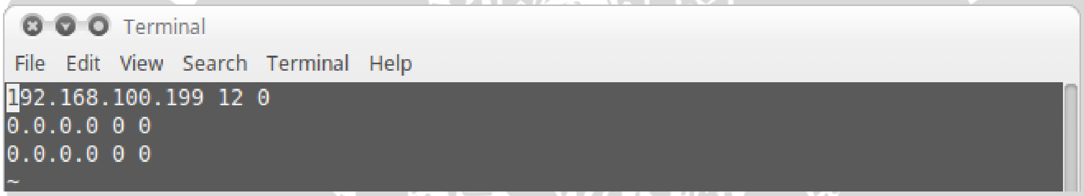


Gambar 6.28 Tampilan file *log* dari proses monitoring untuk perangkat yang aktif

Gambar 6.28 merupakan tampilan dari file *log* dari perangkat yang aktif, yaitu Laptop yang berada pada alamat *IP* 192.168.100.100. Pada file *log* ini data disimpan dengan format sebagai berikut.

[**Alamat*IP***] [**PingTryNo**] [**PingSuccessNo**]

Untuk perangkat yang aktif maka jika **PingTryNo** nilainya bertambah maka **PingSuccessNo** nilainya akan bertambah pula. Pada gambar 6.28 dapat dilihat bahwa dari 133 percobaan *ping*, semuanya berhasil artinya perangkat tersebut selalu dalam kondisi aktif.



Gambar 6.29 Tampilan file *log* dari proses monitoring untuk perangkat yang tidak aktif

Gambar 6.29 merupakan tampilan file *log* dari perangkat yang tidak aktif, yaitu perangkat Test dengan alamat *IP* 192.168.100.199. Dapat dilihat dari percobaan *ping* 12 kali, tidak ada yang berhasil. Hal ini menunjukkan bahwa perangkat tidak pernah aktif sebelumnya.

6.3 Pengujian terhadap fungsi untuk konfigurasi *interval* dari aplikasi monitoring

Pada fitur ini bertujuan agar pengguna dapat melakukan konfirgurasi *interval ping* dan *interval monitoring* yang dimiliki oleh sistem sesuai dengan kebutuhan dari sistem pada suatu saat terentu.

6.3.1 Konfigurasi *interval ping*

Interval ping merupakan jeda waktu yang diperlukan untuk sistem untuk melakukan proses monitoring terhadap sebuah perangkat dengan mengirimkan paket *ping*.

```
# watch -n 1 -t cat /tmp/192.168.100.100.log
```

Gambar 6.30 Perintah yang digunakan untuk monitoring file *log* proses monitoring

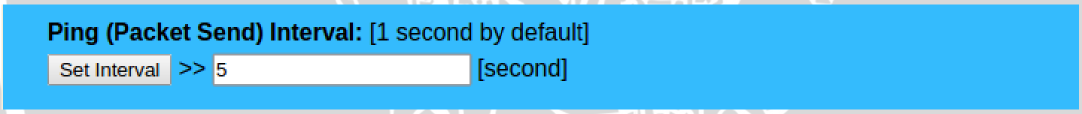
Pada gambar 6.30 menunjukkan perintah yang akan digunakan untuk melakukan pengecekan terhadap *interval ping* dengan melihat data pada file *log* sebuah perangkat. Perintah **watch** akan menjalankan perintah ***cat* /tmp/192.168.100.100.*log*** yang akan menampilkan isi dari file *log* tersebut, lalu perintah diatas akan dipanggil setiap 1 detik sesuai dengan parameter -n 1.

```
192.168.100.100 275 275
0.0.0.0 0 0
0.0.0.0 0 0
```

```
192.168.100.100 280 280
0.0.0.0 0 0
0.0.0.0 0 0
```

Gambar 6.31 Tampilan file *log* proses monitoring dengan interval *ping* default 1 detik setelah 5 detik

Gambar 6.31 menunjukkan tampilan file *log* dari proses monitoring dengan *interval ping default*, yaitu selama 1 detik. Gambar pada bagian atas menunjukkan percobaan *ping* yang sudah dilakukan ada 275 kali, lalu pada gambar bagian bawah adalah tampilan setelah 5 detik kemudian yaitu percobaan *ping* yang sudah dilakukan sebanyak 280 kali. Hal ini dapat terjadi kare *interval ping* yang digunakan adalah 1 detik, jadi selama 5 detik maka percobaan *ping* sudah dilakukan 5 kali.



Gambar 6.32 Tampilan aplikasi saat *interval ping* dirubah

Gambar 6.32 menunjukkan cara untuk mengubah *interval ping* dari sistem. Dapat dilihat bahwa *interval ping* yang baru adalah sebesar 5 detik.

```
192.168.100.100 333 333
0.0.0.0 0 0
0.0.0.0 0 0
```

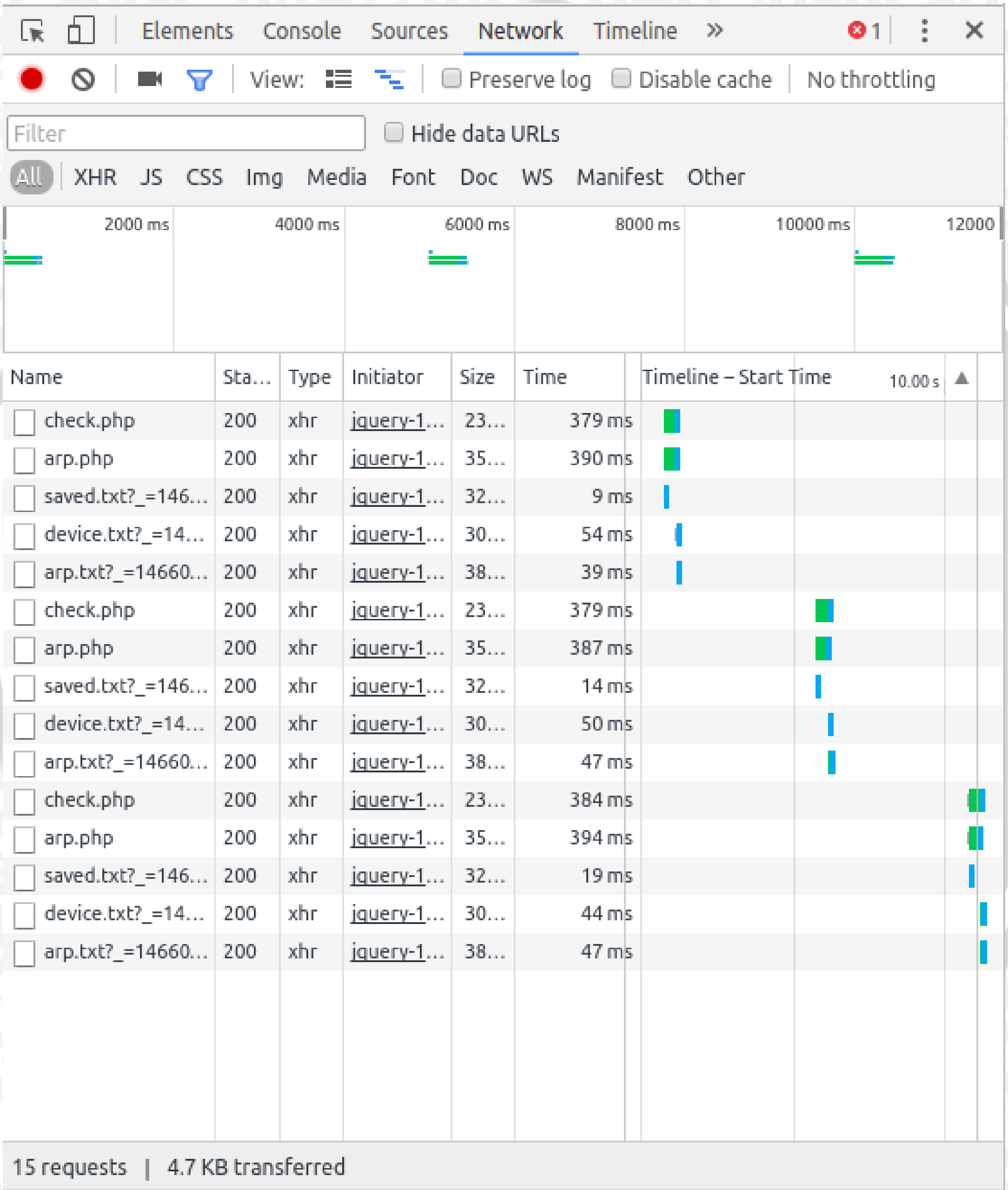
```
192.168.100.100 334 334
0.0.0.0 0 0
0.0.0.0 0 0
```

Gambar 6.33 Tampilan file *log* proses monitoring dengan interval *ping* 5 detik setelah 5 detik

Gambar 6.33 menunjukkan tampilan file *log* dari proses monitoring dengan *interval ping* yang sudah dirubah menjadi 5 detik. Gambar pada bagian atas menunjukkan percobaan *ping* yang sudah dilakukan ada 333 kali, lalu pada gambar bagian bawah adalah tampilan setelah 5 detik kemudian yaitu percobaan *ping* yang sudah dilakukan sebanyak 334 kali. Hal ini dapat terjadi kare *interval ping* yang digunakan adalah 5 detik, jadi selama 5 detik maka hanya ada satu tambahan percobaan *ping* yang dilakukan.

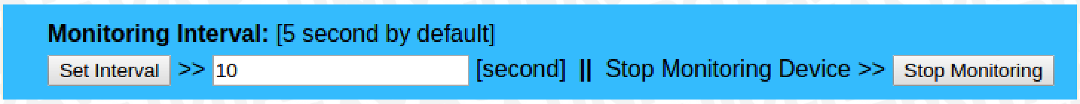
6.3.2 Konfigurasi *interval monitoring*

Interval monitoring adalah jeda waktu yang diperlukan *web browser* untuk melakukan permintaan terhadap *server* untuk me-*refresh* data yang dimiliki lalu meminta agar *server* mengirimkan data tersebut ke *web browser*.



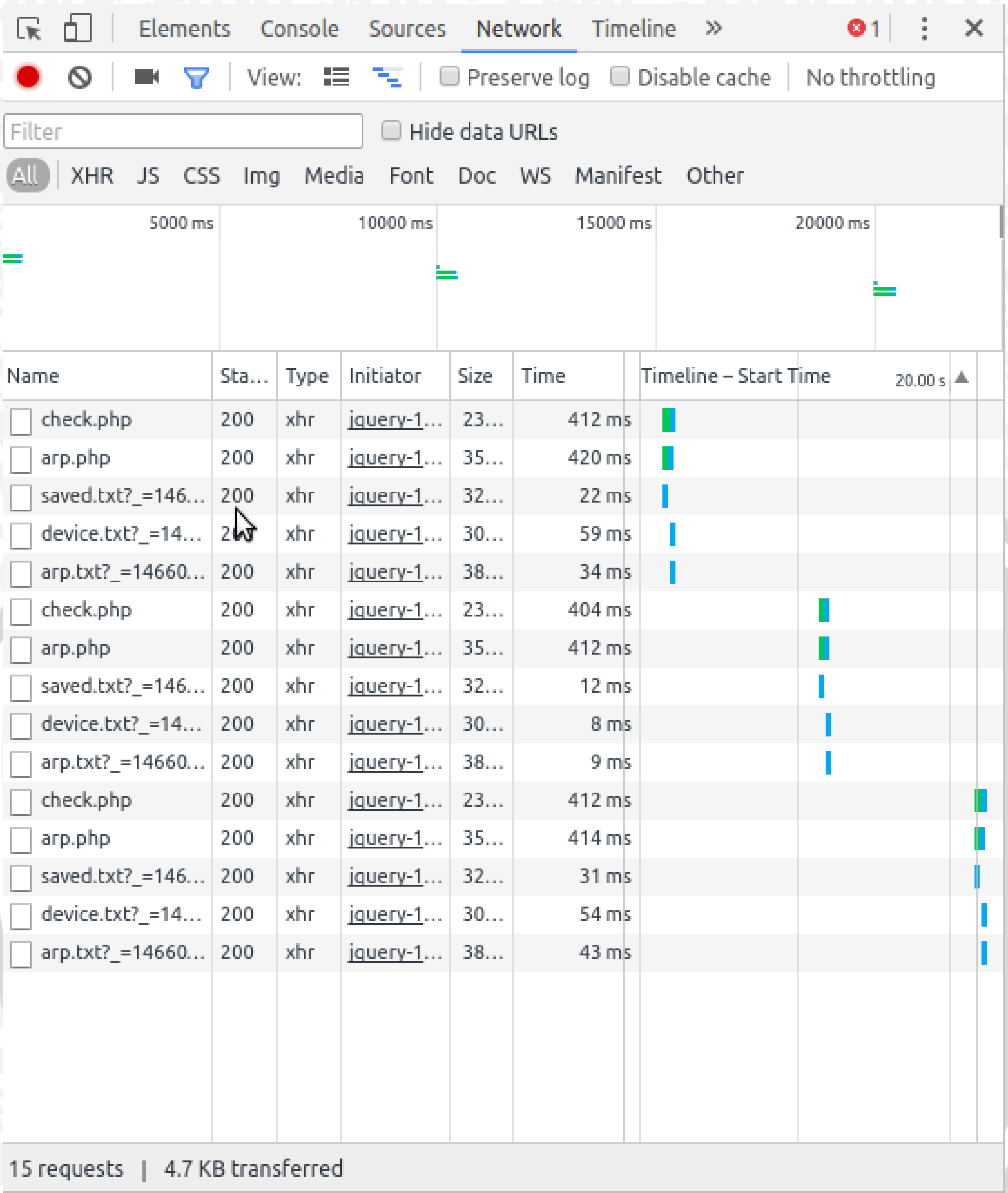
Gambar 6.34 Tampilan informasi jaringan dari *web browser* saat *interval monitoring default* 5 detik

Pada gambar 6.34 merupakan tampilan dari informasi jaringan yang dimiliki *web browser* yang mengakses aplikasi monitoring perangkat ini. Dapat dilihat bahwa proses monitoring akan dilakukan setiap 5 detik sesuai dengan konfigurasi *interval* *monitoring* secara *default*. Jadi setiap 5 detik *web browser* akan melakukan permintaan *POST* untuk check.php dan arp.php, lalu melakukan permintaan *GET* untuk file saved.txt, device.txt, serta arp.txt.



Gambar 6.35 Tampilan aplikasi saat *interval monitoring* dirubah ke 10 detik

Gambar 6.35 menunjukkan cara untuk mengubah *interval monitoring* menjadi 10 detik. Namun sebelum mengubahnya maka proses monitoring harus dihentikan terlebih dahulu agar tidak terjadi dua proses yang sama dan berjalan secara bersamaan.



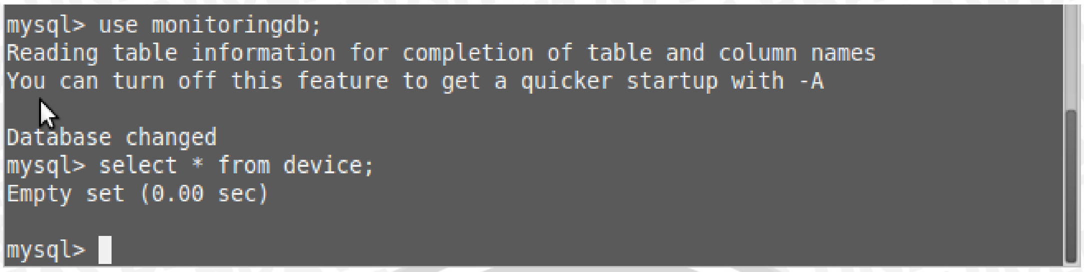
Gambar 6.36 Tampilan informasi jaringan dari *web browser* setelah *interval monitoring* dirubah ke 10 detik

Gambar 6.36 menunjukkan informasi jaringan yang dimiliki *web browser* setelah *interval monitoring* dirubah. Dapat dilihat bahwa *web browser* akan melakukan permintaan ke *server* setiap 10 detik, sesuai dengan *interval monitoring* yang baru.

6.4 Pengujian terhadap fungsi untuk *database*

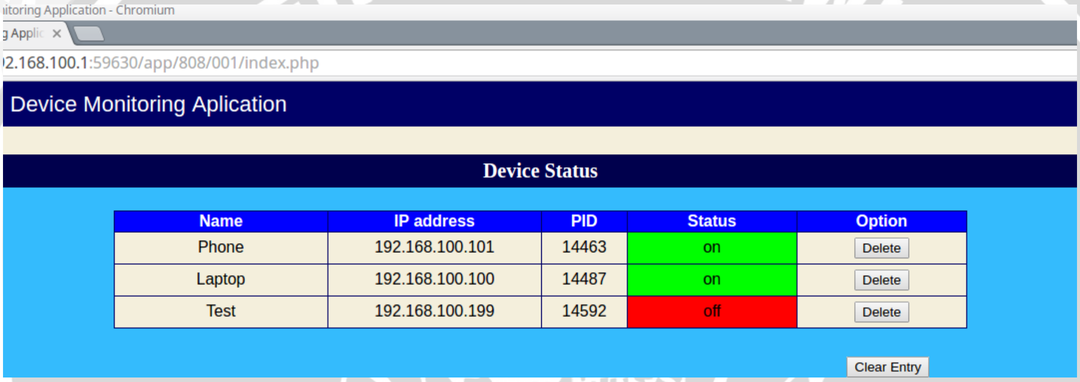
Pada fitur ini bertujuan agar sistem dapat menyimpan data daftar perangkat yang dimiliki ke *database*, ataupu memuat data daftar perangkat yang sudah disimpan di *database*.

6.4.1 Simpan daftar perangkat ke *database*



Gambar 6.37 Tampilan tabel *database* yang digunakan sebelum ada data disimpan

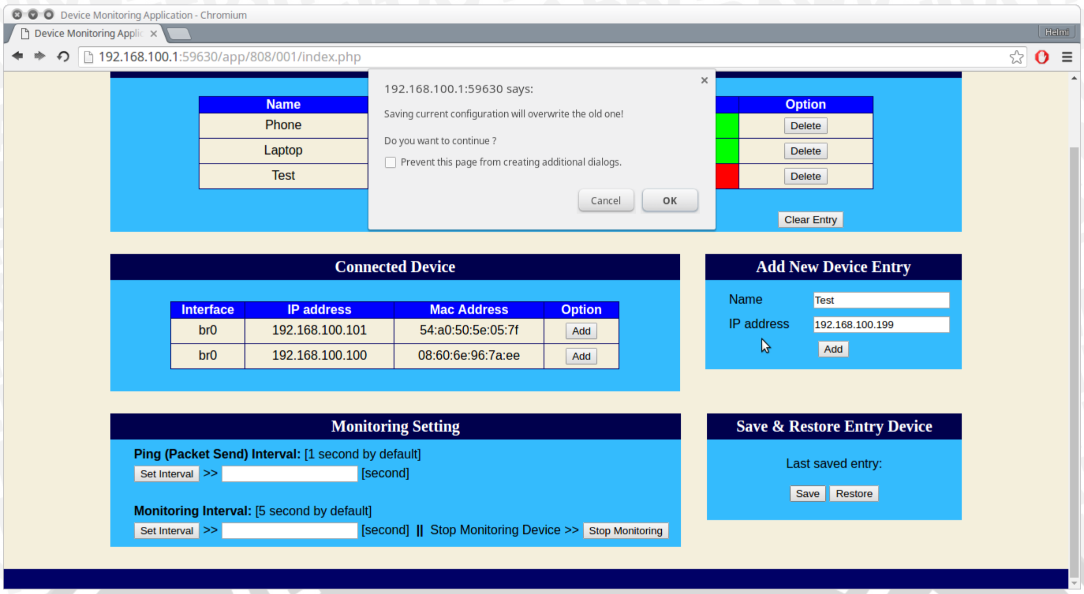
Gambar 6.37 menunjukkan tampilan dari tabel *database* yang digunakan pada sistem ini. Dapat dilihat bahwa tabel masih kosong karena belum ada data yang disimpan ke *database*.



Name	IP address	PID	Status	Option
Phone	192.168.100.101	14463	on	Delete
Laptop	192.168.100.100	14487	on	Delete
Test	192.168.100.199	14592	off	Delete

Gambar 6.38 Tampilan data daftar perangkat pada aplikasi yang akan disimpan ke *database*

Gambar 6.38 menunjukkan tampilan dari data daftar perangkat pada aplikasi yang akan disimpan ke *database*.



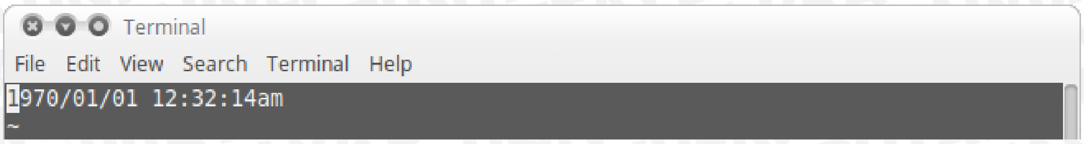
Gambar 6.39 Tampilan konfirmasi pada aplikasi untuk penyimpanan data daftar perangkat ke *database*

Sebelum penyimpanan ke *database* dilakukan, maka seluruh data yang ada pada tabel *database* yang digunakan akan dihapus terlebih dahulu, sehingga perlu dilakukan konfirmasi karena fitur ini bersifat kritis. Gambar 6.39 menunjukkan konfirmasi yang ditampilkan oleh aplikasi saat fitur ini akan digunakan.

```
mysql> select * from device;
+--------+-----------------+
| name   | ip              |
+--------+-----------------+
| Phone  | 192.168.100.101 |
| Laptop | 192.168.100.100 |
| Test   | 192.168.100.199 |
+--------+-----------------+
3 rows in set (0.00 sec)

mysql> 
```

Gambar 6.40 Tampilan tabel *database* yang digunakan setelah ada data yang disimpan

Setelah konfirmasi dilakukan maka data daftar perangkat tadi sudah tersimpan ke *database*. Gambar 6.40 menunjukkan isi dari tabel *database* yang sekarang sudah terdapat informasi dari perangkat yang disimpan tadi. Informasi yang disimpan adalah nama perangkat serta alamat *IP* dari perangkat tersebut.



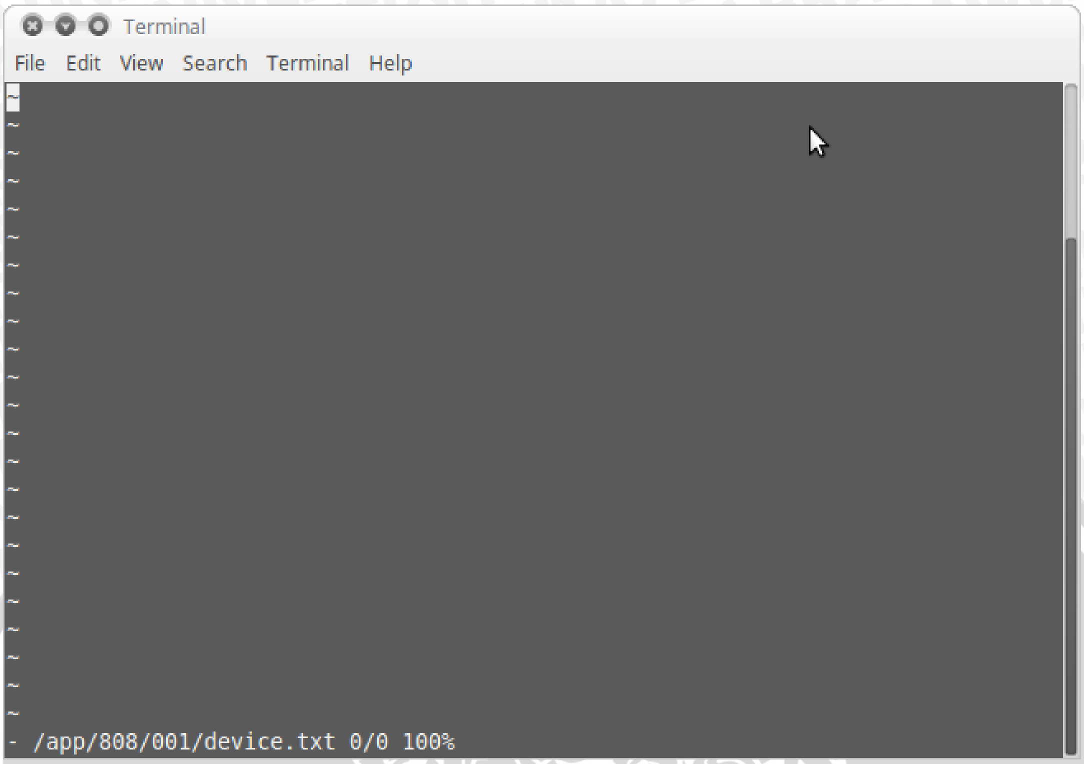
Gambar 6.41 Isi file daved.txt setelah data daftar perangkat disimpan ke *database*

Pada saat data daftar perangkat disimpan di *database*, maka sistem akan menyimpan waktu saat penyimpanannya. 6.41 menunjukkan tampilan dari waktu yang disimpan ke file saved.txt.

Gambar 6.42 Tampilan pada aplikasi waktu penyimpanan terakhir ke *database*

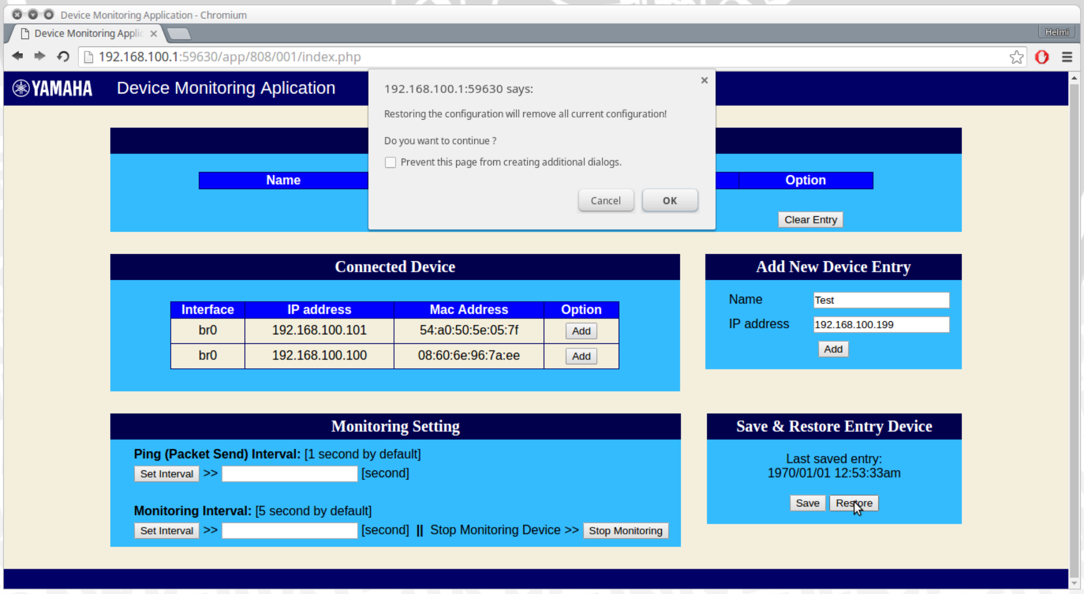
Data dari waktu penyimpanan tadi akan ditampilkan ke pengguna. Gambar 6.42 menunjukkan tampilan dari waktu terakhir penyimpanan ke *database*. Hal ini diperlukan agar pengguna dapat mengetahui bahwa data berhasi disimpan ke *database*.

6.4.2 Muat daftar perangkat dari *database*



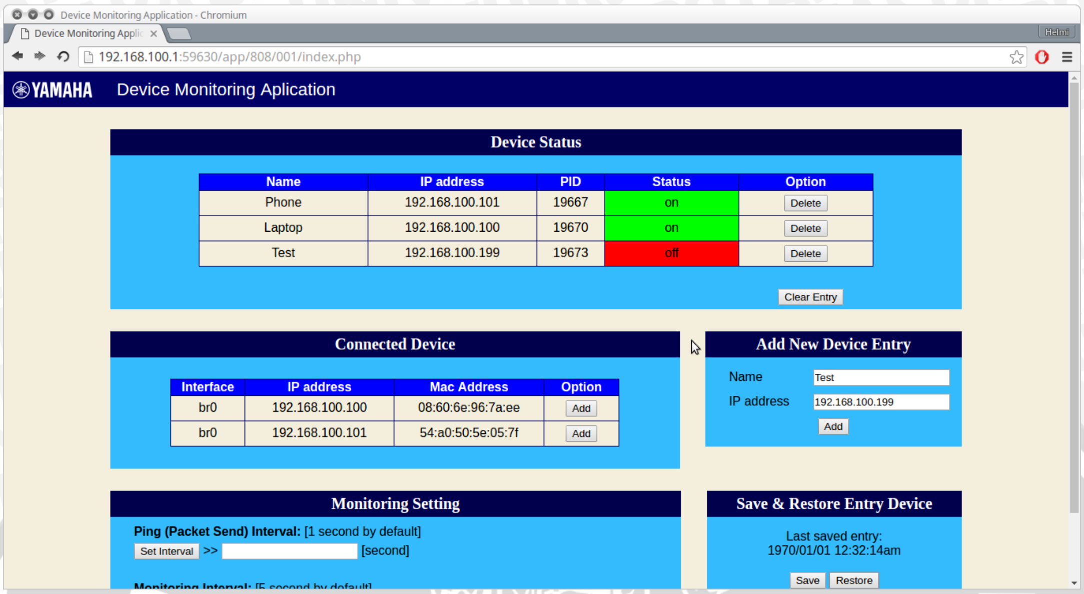
Gambar 6.43 Isi file device.txt saat belum ada daftar perangkat

Pada pengujian untuk memuat data dari *database* ini menggunakan daftar perangkat yang masih kosong. Gambar 6.43 menunjukkan isi dari file device.txt yang artinya belum ada daftar perangkat yang disimpan.



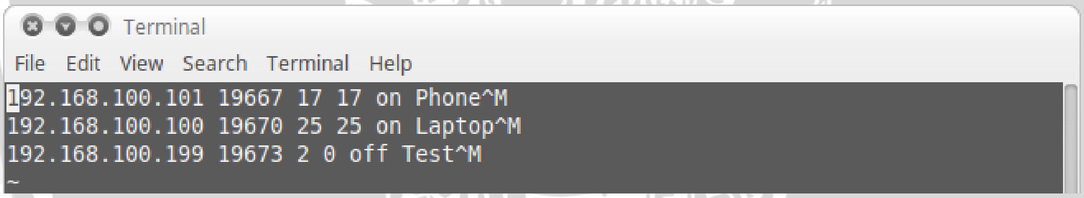
Gambar 6.44 Tampilan konfirmasi pada aplikasi saat data dari *database* akan dimuat

Sebelum data dari *database* dimuat, maka seluruh data yang ada pada daftar perangkat akan dihapus serta proses monitoring untuk setiap perangkat juga akan dihentikan terlebih dahulu, sehingga perlu dilakukan konfirmasi karena fitur ini bersifat kritis. Gambar 6.44 menunjukkan konfirmasi yang ditampilkan oleh aplikasi saat fitur ini akan digunakan.



Gambar 6.45 Tampilan aplikasi setelah data dari *database* berhasil dimuat

Gambar 6.45 menunjukkan tampilan dari daftar perangkat yang sudah di muat kembali dari *database* pada aplikasi. Saat data di muat kembali, maka proses monitoring untuk perangkat tersebut juga akan dijalankan.



Gambar 6.46 Isi file device.txt setelah data dari *database* berhasil dimuat

Data yang dimuat dari *database* juga akan ditambahkan ke file device.txt. Gambar 6.46 menunjukkan daftar perangkat yang berisi informasi dari perangkat yang sudah dimuat dari *database*.

BAB 7 PENUTUP

Berdasarkan perancangan, implementasi, pengujian, dan analisis hasil yang telah dilakukan, maka dapat diambil kesimpulan sebagai berikut.

7.1 Kesimpulan

- Monitoring perangkat berbasis *IP* yang terdapat pada sebuah *Local Area Network* dapat dilakukan oleh sistem yang sudah dibuat.
- Manajemen daftar perangkat pada sistem ini dapat dilakukan, meliputi tambah daftar perangkat (baik tambah secara manual maupun dari *ARP* tabel), hapus daftar perangkat, dan hapus semua daftar perangkat.
- Perangkat dapat terhubung ke *server* baik secara *wired* maupun *wireless*.
- Sistem berhasil melakukan monitoring terhadap perangkat-perangkat yang terdaftar ke daftar perangkat.
- Konfigurasi *interval ping* dan *interval monitoring* pada sistem ini dapat dilakukan.
- Untuk mengubah *interval* monitoring maka proses monitoring yang sebelumnya harus dihentikan terlebih dahulu, karena pengubahan *interval monitoring* pada sistem ini artinya menjalankan proses monitoring dengan *interval monitoring* yang baru.
- Penyimpanan data daftar perangkat ke *database* dan pemuatan kembali data yang sudah disimpan di *database* dapat dilakukan pada sistem ini.

7.2 Saran

- Untuk pengembangan sistem ini masih dapat dilakukan, diantaranya yaitu dengan membuat antarmuka pengguna yang lebih baik. Lalu perlu juga diterapkan metode pengamanan yang lebih, karena pada sistem ini data-data masih disimpan dengan bentuk *plain text*. Selain itu dapat pula diterapkan sebuah algoritma tertentu untuk nilai dari *interval ping* dan *interval monitoring* agar dapat bekerja secara efektif tergantung dari jumlah perangkat yang dimonitoring.
- Untuk penelitian-penelitian yang akan dilakukan selanjutnya, penelitian ini dapat menjadi salah satu metode yang dipertimbangkan, yaitu dengan bekerja sama dengan pihak tertentu. Hal ini akan lebih mudah karena dengan bekerja sama dengan kemudahan akan mendapatkan beberapa kemudahan, misalnya tujuan penelitian, target dari penelitian, termasuk juga infrastruktur yang digunakan dalam sebuah penelitian.

DAFTAR PUSTAKA

Abdullah, A., Ismael, A., Rashid, A., Abou-ElNour, A., & Tarique, M. (2015). *Real Time Wireless Health Monitoring Application Using Mobile Device*.

Cisco. (2009). *Internet Protocol and Radio Frequency Networks:*. Dipetik July 22, 2016, dari Cisco: http://www.cisco.com/c/dam/en_us/solutions/industries/docs/gov/c11-504998_military_network_wp.pdf

Gubbi, J., Buyya, R., Marusic, S., & Palaniswami, M. (2013). *Internet of Things (IoT): A vision, architectural elements, and future directions*.

Internet Engineering Task Force (IETF). (2011). *RFC Base*. Retrieved July 22, 2016, from Security Assessment of the Internet Protocol Version 4: http://www.rfc-base.org/txt/rfc-6274.txt

Internet Engineering Task Force. (2011). *Security Assessment of the Internet Protocol Version 4*. Dipetik July 22, 2016, dari RFC Base: http://www.rfc-base.org/txt/rfc-6274.txt

Kassim, M., Ismail, M., & Che Ku Yahaya, C. (2011). *A Web Based Temperature Monitoring System*.

Kurose, J. F., & Ross, K. W. (2003). *Computer Networking: A Top-Down Approach.* New Jersey: Pearson Education, Inc.

Kusrini. (2007). *Strategi Perancangan dan Pengelolaan Basis Data.* Yogyakarta: Andi.

OMICS. (2014). *Internet Control Protocol*. Dipetik Agustus 16, 2016, dari OMICS International: http://research.omicsgroup.org/index.php/Internet_Control_Message_Protocol

Risodkar, Y. R., Sangole, M. K., Vankhede, A. R., Medhe, R. S., & Shirsat, J. K. (2015). *Web Based Health Monitoring System*.

Rose, K., Eldridge, S., & Chapin, L. (2015). *The Internet of Things (IoT): An Overview.* Dipetik July 21, 2016, dari Internet Society: https://www.internetsociety.org/sites/default/files/ISOC-IoT-Overview-20151014_0.pdf

Yamaha Corporation. (2014). *Smart Gateway SGX808*. Dipetik Agustus 16, 2016, dari Yamaha Corporation: http://www.yamaha.com/products/zh/network/ja/routers/sgx808/?mode=model