PERANCANGAN SISTEM *MONITORING MULTIPLE NETWORK* MENGGUNAKAN *PLATFORM ELASTIC STACK* (STUDI KASUS: PT. JEDI GLOBAL TEKNOLOGI)

Tugas Akhir

Diajukan untuk Memenuhi Salah Satu Persyaratan Mencapai Derajat Sarjana Informatika

Dibuat Oleh:

Vian Handika

16 07 08975

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ATMA JAYA YOGYAKARTA

2020

HALAMAN PENGESAHAN

Tugas Akhir Berjudul

PERANCANGAN SISTEM MONITORING MULTIPLE NETWORK MENGGUNAKAN PLATFORM ELASTIC STACK (STUDI KASUS: PT. JEDI GLOBAL TEKNOLOGI)

yang disusun oleh

VIAN HANDIKA

160708975

dinyatakan telah memenuhi syarat pada tanggal 08 Juli 2020

		Keterangan
Dosen Pembimbing 1	: Prof. Ir. Suyoto, MSc., PhD	Telah menyetujui
Dosen Pembimbing 2	: Dr.Alb. Joko Santoso, MT.	Telah menyetujui
Tim Penguji		
Penguji 1	: Prof. Ir. Suyoto, MSc., PhD	Telah menyetujui
Penguji 2	: Dr. Pranowo, S.T., M.T.	Telah menyetujui
Penguji 3	: Thomas Adi Purnomo Sidhi, ST., MT.	Telah menyetujui

Yogyakarta, 08 Juli 2020

Universitas Atma Jaya Yogyakarta

Fakultas Teknologi Industri

Dekan

ttd

Dr. A. Teguh Siswantoro, M.Sc

PERNYATAAN ORISINALITAS & PUBLIKASI ILMIAH

Saya yang bertanda tangan di bawah ini:

Nama Lengkap	: Vian Handika
NPM	: 16 07 08975
Program Studi	: Teknik Informatika
Fakultas	: Teknologi Industri
Judul Penelitian	: Perancangan Sistem Monitoring Multiple Network Menggunakan Platform Elastic Stack (Studi Kasus: PT. Jedi Global Teknologi)

Menyatakan dengan ini:

1. Tugas Akhir ini adalah benar tidak merupakan salinan sebagian atau keseluruhan dari karya penelitian lain.
2. Memberikan kepada Universitas Atma Jaya Yogyakarta atas penelitian ini, berupa Hak untuk menyimpan, mengelola, mendistribusikan, dan menampilkan hasil penelitian selama tetap mencantumkan nama penulis.
3. Bersedia menanggung secara pribadi segala bentuk tuntutan hukum atas pelanggaran Hak Cipta dalam pembuatan Tugas Akhir ini.

Demikianlah pernyataan ini dibuat dan dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 08 Juli 2020

Yang menyatakan,

Vian Handika

16 07 08975

PERNYATAAN PERSETUJUAN DARI INSTANSI ASAL PENELITIAN

Saya yang bertanda tangan di bawah ini:

Nama Lengkap Pembimbing	: Rein Rachman Putra
Jabatan	: Team Leader
Departemen	: Product,Solution and Technical Development

Menyatakan dengan ini:

Nama Lengkap	: Vian Handika
NPM	: 16 07 08975
Program Studi	: Teknik Informatika
Fakultas	: Teknologi Industri
Judul Penelitian	: Perancangan Sistem Monitoring Multiple Network Menggunakan Platform Elastic Stack (Studi Kasus: PT. Jedi Global Teknologi)

1. Penelitian telah selesai dilaksanakan pada perusahaan.
2. Perusahaan telah melakukan sidang internal berupa kelayakan penelitian ini dan akan mencantumkan lembar penilaian secara tertutup kepada pihak universitas sebagai bagian dari nilai akhir mahasiswa.
3. Memberikan kepada Instansi Penelitian dan Universitas Atma Jaya Yogyakarta atas penelitian ini, berupa hak untuk menyimpan, mengelola, mendistribusikan, dan menampilkan hasil penelitian selama tetap mencantumkan nama penulis.

Demikianlah pernyataan ini dibuat dan dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 29 Juni 2020

Yang menyatakan,

Rein Rachman Putra

Team Leader

HALAMAN PERSEMBAHAN

Untuk semua orang yang membacanya:

"Tidak ada pencapaian yang dapat diraih dengan mudah, semuanya didapat dari jerih payah yang sudah dilakukan, dan setelah melakukan itu semua…. jangan lupa untuk bersyukur dengan pencapaian yang telah didapat, karena tidak ada kata gagal dalam pencapaian"

Vian 2020

KATA PENGANTAR

Segala puji syukur kehadiratan Allah SWT, Tuhan Yang Maha Esa yang dengan rahmat, hidayah, dan karunia-Nya penulis dapat menyelesaikan pembuatan tugas akhir "Perancangan Sistem *Monitoring Multiple Network* Menggunakan *Platform Elastic Stack*" ini dengan baik. Penyusunan tugas akhir ini bertujuan untuk memenuhi salah satu ketentuan untuk menggapai derajat sarjana Teknik Informatika dari Program Studi Teknik Informatika dari Fakultas Teknologi Industri di Universitas Atma Jaya Yogyakarta. Penulis menyadari jika dalam pembuatan tugas akhir ini penulis sudah memperoleh dukungan, bimbingan, serta dorongan dari banyak pihak. Untuk itu, pada peluang ini penulis ingin mengucapkan terima kasih kepada:

1. Allah SWT yang selalu melimpahkan rahmat serta hidayah-Nya, melancarkan segala urusan dan melindungi penulis.
2. PT. Global Jedi Solution sebagai tempat penulis melakukan penelitian untuk menyelesaikan tugas akhir.
3. Bapak Dr. A. Teguh Siswantoro, M.Sc., selaku Dekan Fakultas Teknologi Industri, Universitas Atma Jaya Yogyakarta.
4. Bapak Prof. Ir. Suyoto, M.Sc., Ph.D., selaku dosen pembimbing I yang telah membimbing dan memberikan masukan serta motivasi kepada penulis untuk menyelesaikan tugas akhir ini.
5. Bapak Dr. Ir. Alb. Joko Santoso, M.T., selaku dosen pembimbing II yang telah membimbing dan memberikan masukan serta motivasi kepada penulis untuk menyelesaikan tugas akhir ini.
6. Dosen pembimbing akademik Ibu Dra. Ch. Suryanti, M.Hum yang telah memberikan dukungan dan motivasi kehidupan selama kuliah.
7. Tak lupa dosen pengajar mata kuliah yang sudah saya tempuh dan tak bisa saya sebutkan satu persatu, terimakasih telah memberikan ilmu dan pengalaman berharga dan berguna untuk saya kedepannya.

8. Kedua orang tua tercinta dan kedua adik tersayang, yang telah memberikan do'a, kasih sayang, dukungan keputusan, serta motivasi dari segi moril untuk menyelesaikan penulisan skripsi ini.
9. Teman-teman magang di PT. Jedi Global Teknologi yaitu Rein, Ricky, Rosa, Mia, Ilfa, Ari yang sudah memberikan dukungan dalam penentuan tema dan penilaian hasil sistem bagi penulis.
10. Sahabat P3L yaitu Dewa dan Edi yang sudah memberikan pengalaman berharga dalam kerja sama tim dan mewarnai keseharian kuliah di Universitas Atma Jaya Yogyakarta.
11. Sahabat Perkulihaan yaitu Aga, Yafet, Novri, Raka, Yudha, Hakeem, Yudho, Yuri, Geo, Loly, Jason, Ari, Wilson dan semua teman yang sudah memberikan banyak motivasi dan bantuan bagi penulis selama masa perkuliahan dan masa penulisan Tugas Akhir ini.
12. Teman-teman keluarga Kelompok Studi Linux (KSL) periode 2016/2017 2017/2018 yang telah berirama bersama dalam masa jabatan dan mewarnai keseharian kuliah di Universitas Atma Jaya Yogyakarta.
13. Teman-teman keluarga Himpunan Mahasiswa Teknik Informatika (HIMAFORKA) periode 2017/2018 yang telah berirama bersama dalam masa jabatan dan mewarnai keseharian kuliah di Universitas Atma Jaya Yogyakarta

Demikian penulisan tugas akhir ini dibuat dan penulis mengucapkan terima kasih kepada semua pihak yang berkontribusi. Semoga laporan ini dapat bermanfaat bagi pembaca.

Yogyakarta, 08 Juli 2020

Vian Handika

16 07 08975

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

INTISARI

PERANCANGAN SISTEM MONITORING MULTIPLE NETWORK MENGGUNAKAN PLATFORM ELASTIC STACK (STUDI KASUS: PT. JEDI GLOBAL TEKNOLOGI)

Vian Handika

16 07 08975

PT. Jedi Global Teknologi (Jedi Solutions) adalah salah satu anak perusahaan / member dari PT. Computrade Technology International (CTI Group) yang berpusat di Jakarta. Perusahaan ini berdiri pada tahun 2018 dan bergerak di bidang *manage service network*, dan berfokus pada skala *businesses* atau *enterphrise*. Sebagai penyedia *manage service network*, Jedi solutions menawarkan layanan penuh seperti pemasangan, pengoperasian, pemeliharaan, *monitoring*, dan menganalisis kinerja jaringan *client* untuk kepentingan kelancaran operasional jaringan *client*. Jedi solutions juga bekerja sama dengan beragam *vendor hardware network* untuk keperluan variasi yang dibutuhkan *client*. Oleh karena layanan *monitoring* jaringan pada *client* yang dilakukan oleh tim *Network Operations Center* (NOC) masih menggunakan *dashboard monitoring* dari masing-masing *vendor hardware* dan *client*, tentunya hal tersebut tidak efektif bagi tim NOC dalam menjalankan layanan *monitoring* dalam deteksi awal masalah jaringan.

Dengan permasalahan efektifitas layanan tersebut yang menjadi latar belakang dari perancangan sistem untuk *monitoring* jaringan dari setiap *client* menjadi *single dashboard* menggunakan platform Elastic Stack agar tim NOC dapat memantau atau mendeteksi masalah awal pada jaringan *client* dengan lebih efektif. Elastic stack adalah sekelompok produk *open source* dari Elastic yang dirancang untuk membantu pengguna mengambil data dari berbagai jenis sumber dengan berbagai format, keperluan mencari serta menganalisis data, dan memvisualisasikan data tersebut secara *real time*.

Perancangan sistem *monitoring* jaringan menggunakan Elastic Stack dalam penulisan ini menggunakan komposisi komponen Heartbeat sebagai pengirim data, Elasticsearch sebagai mesin pencari serta analisis, dan Kibana sebagai visualisasi data. Dengan konsep *end-to-end* yang terpisah dari setiap *service*, hal ini sangat mendukung untuk membuat *single dashboard monitoring* jaringan dari semua *client*. Sehingga dalam menjalankan layanan *monitoring* dapat dilakukan dengan hanya membuka satu panel *dashboard* saja.

Kata Kunci: *network monitoring system, elastic stack, single dashboard*

Dosen Pembimbing I : Prof. Ir. Suyoto, M.Sc., Ph.D.

Dosen Pembimbing II : Dr. Ir. Alb. Joko Santoso, M.T.

Jadwal Sidang Tugas Akhir : 08 Juli 2020

BAB I. PENDAHULUAN

1.1. Latar Belakang

Teknologi informasi (TI) berkembang dengan sangat pesat dan mempengaruhi setiap sektor dalam kehidupan sehari-hari. Pesatnya pertumbuhan TI di sektor bisnis, juga melibatkan berbagai jenis penerapan TI dalam menjalankan operasional bisnisnya dan salah satunya adalah sistem jaringan komputer [1]. Dengan adanya jaringan komputer, perangkat TI bisa saling terkoneksi satu sama lain dan membentuk kolaborasi yang diperlukan untuk mendukung proses jalannya bisnis dalam suatu instansi. Sehingga jaringan komputer memainkan peran yang penting untuk menumbuhkan bisnis, yaitu sebagai platform untuk bisa saling berbagi sumber daya seperti perangkat, data dan informasi [1]. Untuk menjaga peran tersebut berjalan secara optimal, dibutuhkan sebuah sistem yang dapat digunakan untuk *monitoring* jaringan komputer tersebut [2].

Sistem *monitoring* jaringan merupakan sistem yang digunakan untuk membantu para operator jaringan untuk dapat memantau dan mengelola jaringan [3]. Sistem *monitoring* jaringan pada umumnya sudah disediakan oleh setiap *vendor* perangkat jaringan sekaligus sebagai panel untuk konfigurasi perangkat, beberapa vendor seperti *Ruckus*, *Cisco* dan *Versa* sudah mengintegrasikan sistemnya ke *cloud* sehingga bisa dimonitor tanpa harus berada dalam lingkup jaringan. Sistem ini berperan penting dalah hal informasi dan notifikasi ke operator jaringan ketika ada masalah, sehingga operator jaringan bisa cepat tanggap untuk menangani masalah yang terdapat pada jaringan dari deteksi awal sampai solusi yang diperlukan [1]. Kegiatan *monitoring* yang dilakukan oleh operator jaringan merupakan tanggung jawab yang besar karena harus berupaya menjaga kelancaran operasional jaringan yang dapat berdampak pada menurunnya produktivitas dan menurunnya kepercayaan pelanggan terhadap pelayanan yang diberikan pada suatu instansi ketika jaringan mengalami *down* [4]. Beberapa perusahaan mulai menanggapi tantangan dan kebutuhan ini, baik dengan pegawai yang ahli pada bidang ini atau menggunakan jasa dari perusahaan penyedia layanan yang serupa.

Dalam studi kasus ini PT. Jedi Global Teknologi (Jedi Solutions) merupakan salah satu perusahaan yang menyediakan layanan dibidang *manage service network*, dengan mencangkup pilihan layanan seperti pemasangan, pengoperasian, pemeliharaan, dan menganalisis kinerja jaringan *client* untuk kepentingan kelancaran operasional jaringan. Layanan yang dibahas dalam studi kasus ini adalah layanan *monitoring* pada jaringan client yang dilakukan oleh tim *Network Operation Center* (NOC). Kegiatan *monitoring* yang dilakukan masih dengan cara memantau *dashboard* sistem *monitoring* jaringan buatan *vendor* perangkat jaringan yang dipakai oleh setiap client. Banyaknya *client* dan jenis *vendor* perangkat jaringan membuat tim NOC harus membuka lebih dari satu panel *dashboard* untuk melakukan *monitoring* jaringan pada semua *client* yang ada. Hal tersebut menandakan bahwa sistem yang dipakai masih belum komprehensif dengan belum bisa menyediakan *single* panel *view/dashboard* [4]. Dengan kurangnya perangkat *display* yang ada di ruang NOC serta kurangnya SDM yang bekerja, hal tersebut bisa menghambat kesadaran tim NOC untuk mengetahui terjadinya permasalahan jaringan pada *client*, di mana kecepatan tim NOC dalam menangani permasalahan jaringan berhubungan dengan kepercayaan yang diberikan oleh *client* untuk menjaga jalannya operasional jaringan. Dari masalah tersebut, terciptalah ide untuk merancang sistem *monitoring* jaringan menggunakan platform Elastic Stack.

Elastic Stack merupakan kumpulan produk *open source* (*Elasticsearch*, *Logstash*, *Kibana* dan *Beat*) dari *elastic.co* yang memiliki kemampuan untuk mengumpulkan dan mengolah data dari berbagai sumber data, menyimpan data dalam data *center* yang dapat ditingkatkan seiring berkembangnya data, dan menyediakan alat untuk menganalisis serta memvisualisasi data. Elastic Stack memiliki banyak kegunaan, salah satunya adalah digunakan sebagai alat *monitoring* disektor IT [5]. Dalam penerapannya, kumpulan produk yang ada dalam Elastic Stack adalah sebuah perangkat lunak yang dapat berjalan secara terpisah dan dapat saling berkolaborasi dalam sistem operasi komputer. Dengan kemampuan yang bersifat *powerfull*, *scalable*, *separable*, dan *realtime*, Elastic Stack bisa digunakan sebagai sistem *monitoring* network yang dibutuhkan dalam studi kasus ini.

Dalam studi kasus ini, perancangan sistem *monitoring* jaringan dirancang untuk dapat membantu tim NOC pada perusahaan PT. Jedi Global Teknologi menjalakan layanan *monitoring* semua perangkat jaringan *client* menggunakan platform Elastic Stack. Elastic Stack dirancang dengan skema sistem *client-server*, yaitu komponen Heartbeat (Beats) dijalankan disetiap *client* untuk mengumpulkan data serta mengirimnya ke komponen Elasticsearch untuk *indexing* dan menampilkannya di komponen Kibana yang berada di *server*. Untuk *alerting* komponen Elasticsearch akan mengirim data ke *backend* untuk menyimpan data *alert* dan mengirim data ke Firebase untuk notifikasi di aplikasi android. Hasil akhir dari studi kasus ini adalah panel *dashboard* untuk *monitoring* di Kibana, *platform* web untuk mengelola data *client* yang akan dimonitor, dan *platform* Android untuk notifikasi *alert*. Sehingga dalam menjalankan layanan *monitoring* dapat dilakukan dengan hanya membuka satu panel *dashboard* saja.

1.2. Rumusan Masalah

Berdasarkan latar belakang masalah di atas, permasalahan yang muncul diantaranya yaitu:

1. Bagaimana merancang sistem monitoring jaringan dengan menggunakan platform Elastic Stack ?
2. Bagaimana membangun aplikasi berbasis Android yang dapat menerima notifikasi ketika ada permasalahan dari sistem *monitoring* jaringan ?
3. Bagaimana membangun aplikasi berbasis *webiste* untuk mengelola data aset *client* sekaligus menambah informasi pada aplikasi Android ?

1.3. Batasan Masalah

Agar pembahasan lebih terarah maka diperlukan batasan masalah, batasan-batasan masalah dalam penelitian ini yaitu:

1. Fitur yang ada pada sistem *monitoring* jaringan berplatform Elastic Stack yang dibuat adalah fitur *monitoring up/down* pada perangkat jaringan client.

1.4. Tujuan Penelitian

Dari latar belakang dan rumusan masalah di atas,ada tujuan dari penelitian ini yaitu:

1. Merancang sistem *monitoring* jaringan dengan mennggunakan platform Elastic Stack.
2. Membangun aplikasi berbasis Android yang dapat menerima notifikasi ketika ada permasalahan dari sistem monitoring jaringan.
3. Membangun aplikasi berbasis web untuk mengelola data aset *client* sekaligus menambah informasi pada aplikasi Android.

1.5. Metode Penelitian

Metode penelitian yang digunakan dalam mengembangkan aplikasi ini antara lain sebagai berikut:

a. Studi Literatur

Pada tahap ini, penulis mencari dan mempelajari literatur yang berhubungan dengan topik Elastic Stack yang akan digunakan. Literatur atau referensi dapat diperoleh dari artikel, jurnal, buku, serta dari halaman situs yang dapat dipercaya. Literatur yang diperoleh digunakan sebagai media pembelajaran dalam membangun sistem baik secara praktis dan teoritis dengan tema yang sesuai diangkat penulis.

b. Analisis

Pada tahap ini, penulis menganalisis kebutuhan perangkat lunak yang dibutuhkan untuk membangun sistem dengan didukung dengan literatur yang sudah dipelajari dan sesuai dengan kebutuhan perusahaan. Dalam tahap ini ada beberapa hal yang dianalisis seperti kebutuhan perangkat keras dan perangkat lunak, konfigurasi yang dibutuhkan, bahasa pemrograman yang digunakan, database beserta relasinya, antarmuka, serta proses kerja sistem. Tahapan ini dapat menghasilkan gambaran mengenai arsitektur sistem yang akan dibuat.

c. Perancangan Perangkat Lunak

Pada tahap ini, penulis merancang perangkat lunak berdasarkan proses kerja dan arsitektur sistem yang dikaji dari tahap analisis. Tahap ini digunakan untuk membuat desain arsitektur sistem, desain relasi database, desain antarmuka dan *use case* dari alur proses kerja sistem yang akan dibuat.

d. Implementasi Perangkat Lunak

Pada tahap ini, penulis mengimplementasikan hasil dari tahap perancangan. Penulis menerapkan konfigurasi yang diperlukan untuk menjalankan Elastic stack serta menggunakan *query* Lucene untuk menampilkan data pada *dashboard* sistem monitoring jaringan yang dibuat di Kibana. Pembuatan aplikasi berbasis web yang digunakan untuk mengelola data asset client, dibangun dengan menggunakan *framework* Laravel (PHP) untuk *backend* dengan konsep API, dan menggunakan *framework* Vuejs-Vuetify (Javascript) untuk *frontend*. Pembuatan aplikasi Android yang berguna untuk menampilkan data *alert* dan menerima notifikasi, dibangun menggunakan bahasa pemrograman java dengan konsep API.

e. Pengujian Perangkat Lunak

Pada tahap ini, pengujian dilakukan setelah tahap implementasi perangkat lunak selesai dilakukan. Tahap ini akan menguji aplikasi dengan beberapa skenario secara *black box* dengan tujuan mencari kesalahan yang ada di dalam sistem tersebut dan melihat apakah sistem sudah sesuai dan layak untuk digunakan.

1.6. Sistematika Penulisan

Untuk mempermudah dalam memahami laporan ini, maka materi-materi yang tertera dalam laporan ini dikelompokkan menjadi beberapa sub-bab dengan sistematika penulisan sebagai berikut:

BAB I: Pedahuluan

Bab ini berisikan proses yang memiliki keterkaitan dalam penelitian ini seperti latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, alat dan bahan, metodologi penelitian, dan sistematika penulisan.

BAB II: Tinjauan Pustaka

Bab ini berisikan penjelasan penelitian terdahulu dengan keterikatan yang mirip dengan penelitian ini. Bab ini menggambarkan perbandingan antara penelitian ini dan penelitian terdahulu dalam bentuk tabel.

BAB III: Landasan Teori

Bab ini berisikan dasar teori yang digunakan dan mendukung dalam implementasi aplikasi beserta proses yang berhubungan dengan pembangunan aplikasi.

BAB IV: Analisis dan Perancangan Sistem

Bab ini berisikan penjelasan serta analisis perancangan sistem dan implementasi sistem yang telah dibuat.

BAB V: Implementasi Dan Pengujian Sistem

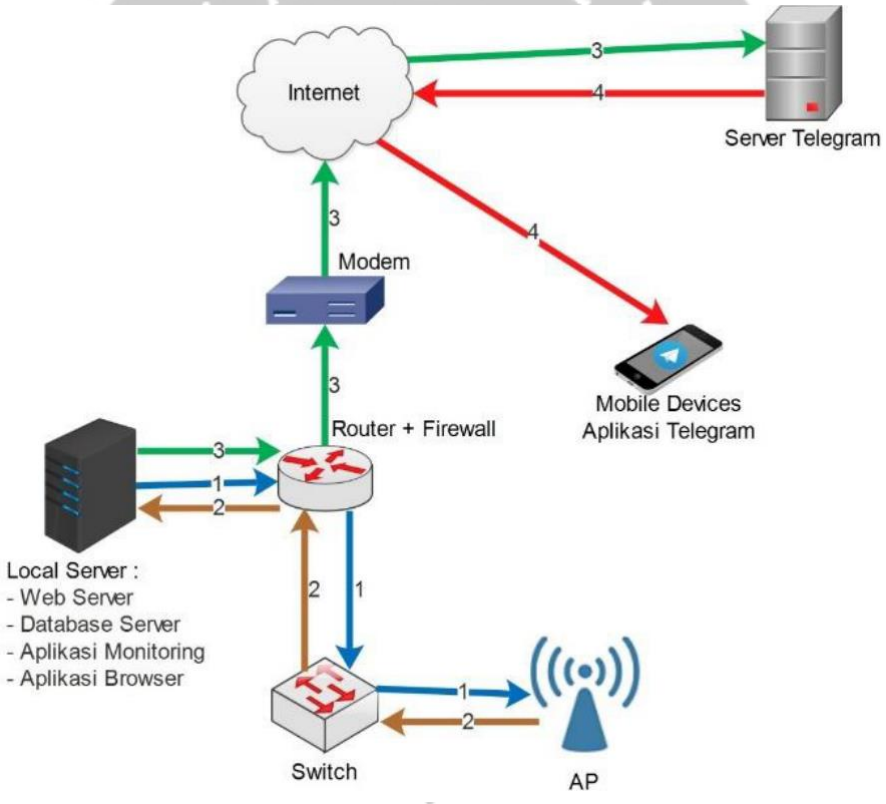
Bab ini berisi penjelasan dari implemntasi serta hasil dari pengujian sistem beserta penilaian dari pengujian pengguna.

BAB VI: Penutup

Bab ini berisikan kesimpulan dari penelitian yang sudah melalui tahap pengujian, serta saran yang berguna bagi pengembangan lebih lanjut.

BAB II. TINJAUAN PUSTAKA

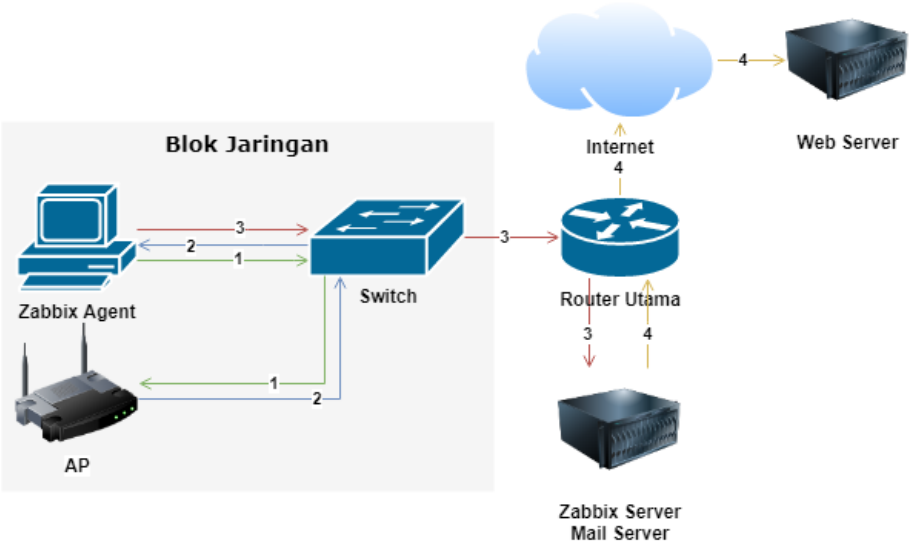
Dalam melakukan penelitian diperlukan dukungan dari beberapa penelitian yang sudah ada dari pihak lain dan berakitan dengan penelitian tersebut. Bagian ini akan menjelaskan dan menganalisis beberapa referensi penelitian yang memiliki hubungan dengan topik sistem *monitoring* jaringan dan penggunaan Elastic stack. Selain itu, informasi dari referensi tersebut digunakan sebagai landasan teori dalam penulisan dan pembuatan program atau sistem di penelitian ini beserta pembanding pada Tabel 2.1.



Gambar 2.1 Arsitektur sistem buatan Petrus [6]

Gambar 2.1 menunjukkan arsitektur sistem penelitian Petrus Sokibi, membahas tentang pemanfaatan *Internet Control Message Protocol* (ICMP) untuk merancang sistem monitoring perangkat jaringan SMK NU Kaplongan beserta *bot* Telegram untuk mengirim notifikasi berupa pesan ketika ada perangkat yang terputus. Protokol ICMP dapat memberikan dua jenis pesan yaitu *query message* yang berupa informasi suatu kondisi dari jaringan komputer yang berhasil memberikan respon sesuai permintaan dan

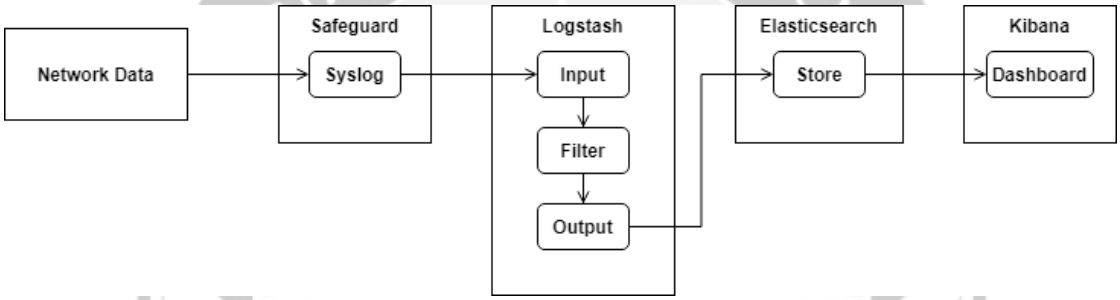
error message yang berupa informasi kesalahan (error) yang terjadi dalam jaringan komputer yang gagal memberikan respon. Petrus menjelaskan bahwa gangguan pada jaringan tidak akan teratasi meskipun sudah memanfaatkan ICMP sebagai acuan monitoring jaringan, akan tetapi setidaknya dapat memberikan peringatan kepada *Network Administrator* untuk dapat segera melakukan tindakan terhadap gangguan tersebut. Untuk implementasinya Petrus membuat aplikasi berplatform web sebagai antar muka sistem *monitoring* dan untuk mengelola data tempat blok jaringan beserta perangkat jaringan yang dimonitor secara lokal di server sekolah tersebut. Dalam pengujian waktu respon pengiriman notifikasi ke telegram pada sistem buatan Petrus, tercatat lima kali pengujian dengan rata-rata membutuhkan waktu 16 detik untuk pesan notifikasi masuk ke telegram setelah status perangkat berubah menjadi *disconnect*. Petrus menjelaskan bahwa ada beberapa faktor yang mempengaruhi waktu respon pengiriman notifikasi antara lain kualitas koneksi internet, dan kondisi *server* lokal di sekolah tersebut maupun *server* Telegram [6].



Gambar 2.2 Ilustrasi arsitektur sistem buatan Dwi [3]

Gambar 2.2 menunjukkan arsitektur sistem penelitian Dwi Wijonarko, membahas tentang pemanfaatan aplikasi Zabbix sebagai perangkat monitoring jaringan di SKPD Kota Malang dengan notifikasi email dan sms ketika ada gangguan pada jaringan. Zabbix itu sendiri merupakan aplikasi *open source* yang berfokus dibidang jaringan dengan fitur grafis lengkap seperti topologi jaringan, statistik, *screen monitoring*, dan notifikasi. Pada

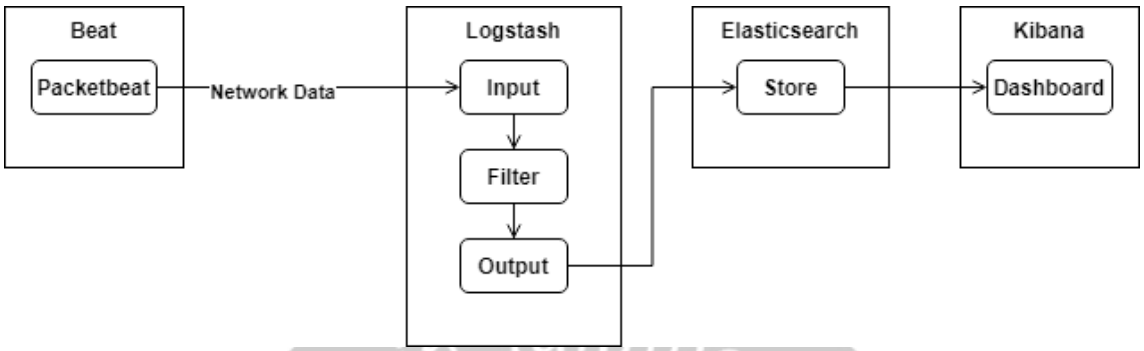
penelitiannya Dwi menerapkan konsep client/agent–server dengan memasang *agent* dari Zabbix di komputer pada masing-masing blok jaringan client untuk mengambil data dan mengirimnya ke sisi server Zabbix. Untuk metode pengambilan data pada penelitiannya menggunakan *Simple Network Management Protocol* (SNMP) sehingga setiap perangkat jaringan di sisi client harus disetting bagian SNMP *poll* agar bisa dikenali oleh Zabbix *agent*. Pada implementasinya Dwi membuat aplikasi berplatform *web* untuk menampilkan data-data dari Zabbix *server* sehingga pengguna bisa membukanya tanpa harus membukanya di Zabbix *server*. Pada akhir penelitiannya Dwi menyarankan dan menjelaskan bahwa menata pengalokasian alamat IP perangkat jaringan merupakan hal yang penting agar mempermudah dari segi *monitoring* perangkat tersebut [3].



Gambar 2.3 Workflow Elastic stack buatan Ibrahim et al [7]

Gambar 2.3 menunjukkan *workflow* Elastic Stack penelitian Ibrahim, Potdar, dan Prashant, membahas tentang penggunaan ELK stack untuk meningkatkan keamanan jaringan melalui analisis *log*. Ibrahim et al menemukan masalah tentang adanya *vulnerabilities* pada suatu perangkat *safeguard* (*IDS/IPS, firewall, anti-spam, web filtering*) komersil yang notabene perangkat tersebut mengontrol semua koneksi baik yang masuk maupun yang keluar dari jaringan. Beberapa *vulnerabilities* memang bisa tidak terdeteksi selain dengan melakukan analisis *log* untuk memeriksa ulang celah yang ditemukan dan yang membahayakan jaringan. Untuk implementasinya Ibrahim et al menggunakan ELK stack (*Elasticsearch, Logstash, Kibana*) yang diinstall pada komputer dalam jaringan tersebut. Logstash dikonfigurasi untuk menerima data *stream syslog* secara *real-time* dari perangkat *safeguard*, mengolahnya keformat sesuai filter yang diterapkan, dan mengirim log tersebut ke Elasticsearch untuk disimpan sebelum ditampilkan di *custom dashboard* Kibana yang informatif dan mudah dimengerti. Hasil

dari penelitian Ibrahim et al adalah *custom dashboard* di Kibana untuk keperluan analisis data *syslog* dari *safeguard* yang sudah difilter menggunakan Logstash [7].



Gambar 2.4 Workflow Elastic stack buatan Bayana et al [8]

Gambar 2.4 menunjukkan penelitian Bayana, Ravindranath, dan Jayanag, membahas tentang penggunaan Packetbeat untuk monitoring dan analisis aktifitas yang mencurigakan pada jaringan. Packetbeat merupakan salah satu *agent* dari Beat yang berfungsi sebagai pengirim data ke Elasticsearch. Bayana et al menjelaskan bahwa Packetbeat bisa diintegrasikan dengan komponen ELK stack, dengan tugas sebagai pengumpul data atau *packet* jaringan dari sistem client dengan merekam *network traffic* dari beberapa protokol seperti DNS, HTTP, ICMP, dll lengkap dengan *source-destination* pada *traffic* tersebut dan secara *real-time*. Permasalahan yang ada pada penelitian Bayana et al hampir sama dengan penelitian Ibrahim et al [7] yaitu tentang kesadaran akan *vulnerabilities* yang ada pada jaringan, hal yang membedakan dari kedua penelitian ini adalah sumber dan jenis data yang digunakan untuk tahap analisis. Pada implementasinya Bayana et al menginstal Packetbeat pada komputer client yang akan dimonitor, Packetbeat akan hanya merekam *network traffic* baik yang keluar maupun yang masuk dari komputer itu saja. Selanjutnya Packetbeat akan mengirim data yang didapat ke Logstash yang sudah dikonfigurasi untuk menerima *input* dari Packetbeat, kemudian masuk proses *filter* dan hasil *output* akan dikirim ke Elasticsearch untuk di simpan dan ditampilkan di Kibana dalam bentuk *dashboard* ataupun *log stream*. Hasil dari penelitian Bayanan et al adalah sebuah sistem Elastic Stack untuk keperluan analisis data *packet* jaringan dari suatu *network traffic* komputer menggunakan Packetbeat lengkap dengan *custom dashboard* di Kibana [8].

Tabel 2.1 Tabel Perbandingan

Item Pembanding	**Petrus Sokibi** [6]	**Dwi Wijonarko** [3]	**Ibrahim et al** [7]
Judul Penelitian	Perancangan Sistem Monitoring Perangkat Jaringan Berbasis ICMP dengan Notifikasi Telegram	Zabbix Network Monitoring Sebagai Perangkat Monitoring Jaringan Di SKPD Kota Malang	*Network Security Enhancement through Effective Log Analysis Using ELK*
Tujuan Penelitian	Merancang sistem *monitoring* jaringan untuk membantu Network Administrator dalam me-monitoring jaringan di SMK NU Kaplongan	Merancang sitem *monitoring* jaringan untuk mengatasi permasalahan kurangnya tenaga ahli yang *standby* untuk mengawasi jaringan setiap SKPD Kota Malang	Merancang sistem *monitoring log* untuk dapat dianalisis dalam mencari dan mencegah *vulnerabilities* yang ada pada perangkat *safeguard* komersil
Metode	Menggunakan data ICMP	Menggunakan Zabbix dengan data dari SNMP	Menggunakan Elastic Stack dengan data *syslog* dari perangkat *safeguard*
Hasil Penelitian	- Sistem monitoring bebasis web lokal dengan notifikasi telegram	- Sistem monitoring berbasis web yang bisa diakses secara online dengan notifikasi email dan SMS	- Sistem monitoring log berupa *dashboard* di Kibana untuk kebutuhan analisis log

Item Pembanding	**Bayana et al** [8]	**Penulis**
Judul Penelitian	*Monitoring and Analysing Anomaly Activities in a Network using Packetbeat*	Perancangan Sistem Monitoring Multiple Network Menggunakan Platform Elastic Stack (Studi Kasus: PT. Jedi Global Teknologi)
Tujuan Penelitian	Merancang sistem *monitoring network traffic* untuk mencegah serangan jaringan pada sistem *client* dengan cara mendeteksi *network traffic* yang mencurigakan	Merancang sitem *monitoring* perangkat jaringan untuk membantu tim NOC pada PT. Jedi Global Teknologi untuk mempermudah melakukan layanan *monitoring* terhadap *client* dan mengelola asset *client* yang dimonitor
Metode	Menggunakan Elastic stack dengan data dari Packetbeat berupa *network traffic*	Menggunakan Elastic stack dengan data dari Heartbeat berupa data ICMP
Hasil Penelitian	- Sistem monitoring *network traffic* dengan *dashboard* di Kibana untuk keperluan analisis *network traffic*	- Sistem monitoring jaringan berupa *dashboard* di Kibana dengan notifikasi pada aplikasi Android - Aplikasi Android untuk menerima serta menampilkan notifikasi dan informasi *alert* - Aplikasi berbasis web untuk mengelola data aset client

Tabel 2.1 menunjukkan perbedaan pada penilitan ini dengan penelitian penelitian terdahu.

BAB III. LANDASAN TEORI

3.1. Jaringan Komputer

Jaringan komputer adalah sebuah sarana interkoneksi komputer agar bisa saling terhubung untuk saling berkomunikasi, bertukar data, dan berbagi sumber daya [9]. Dalam jaringan komputer terdapat *node* yang mewakili sebuah komputer atau peripheral lain yang terhubung, dan pada umumnya jaringan komputer dapat terdiri dari dua *node* atau bahkan lebih. Setiap komunikasi yang terjadi antar *node* pada jaringan komputer memiliki tujuan yang sama yaitu membawa informasi dengan tepat dan tanpa ada kesalahan baik dari sisi pengirim (*transmitter*) dan sisi penerima (*receiver*) melalui media komunikasi berupa kabel maupun nirkabel.

Jaringan komputer dapat memberikan fasilitas kolaborasi yang tinggi sehingga banyak diterapkan baik untuk individu maupun untuk organisasi atau perusahaan dalam menjalankan kegiatan operasionalnya. Hal ini dikarenakan jaringan komputer memiliki beberapa manfaat lainnya, antara lain [10]:

- Mempermudah menyebarkan informasi, sehingga infromasi dapat tersampaikan dengan lebih cepat.
- Menjaga keamanan data dari pihak luar, sehingga data lebih aman. Dengan adanya jaringan komputer membuat data hanya bisa diakses jika sudah terhubung dengan jaringan tersebut.
- Mempermudah untuk melakukan komunikasi, sehingga bisa menjamin keharmonisan dan kestabilan dalam suatu kolaborasi. Dengan adanya jaringan komputer permasalahan yang terjadi akibat komunikasi terhambat bisa diminimalisir.
- Mempermudah mengakses data dari data *center* atau *sever*, sehingga kebutuhan data cepat terpenuhi. Dengan adanya jaringan komputer *server* bisa terhubung dan diakses dengan komputer user.
- Menghemat biaya dari segi transportasi data yang tergantikan oleh jaringan komputer.

3.2. Network Monitoring System

NMS (*Network Monitoring System*) merupakan sebuah sistem yang digunakan oleh operator jaringan untuk memantau kondisi jaringan secara *real-time* [3]. NMS juga berperan untuk memberikan notifikasi kepada operator jikalau ada permasalahan yang terjadi pada jaringan agar dapat segera ditangani [1]. Operator jaringan memiliki tanggung jawab dan berperan penting dalam jaringan, dengan menggunakan NMS operator jaringan dapat menjaga peran jaringan tetap berjalan secara optimal [11]. Terdapat dua jenis sistem yang bisa di terapkan pada NMS yaitu :

- *Connection Monitoring*

  Sistem yang berfokus pada deteksi awal terjadinya gangguan jaringan yang berhubungan dengan konektivitas antar perangkat jaringan. Sistem akan memberikan informasi berupa status *up* atau *down* pada perangkat jaringan sesuai kondisi yang terjadi.
- *Traffic Monitoring*

  Sistem yang berfokus pada data statistik yang bisa membantu keputusan untuk optimasi jaringan berdasarkan lalu lintas transaksi data yang dilakukan antar perangkat jaringan.

Dalam menjalankan fungsi sistem, NMS mendapatkan data dari *network protocol* yang digunakan oleh perangkat jaringan. *Network protocol* merupakan sekumpulan peraturan yang memungkinkan perangkat di jaringan saling berkomunikasi. Berikut adalah contoh *network protocol* dasar atau standar [12]:

- *Simple Network Management Protocol*

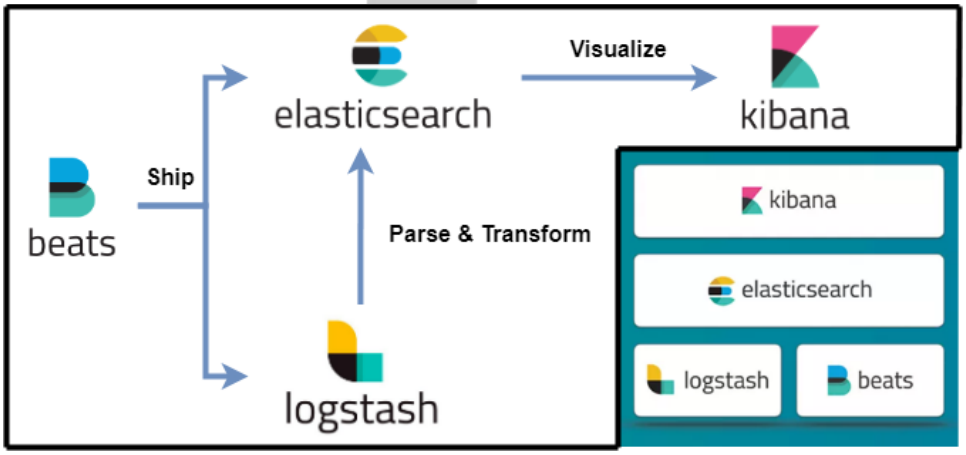
  SNMP (*Simple Network Management Protocol*) merupakan standar protokol untuk mengambil data objek dari perangkat jaringan yang terkoneksi dengan jaringan. Data yang diambil dapat digunakan untuk mengembangkan informasi dalam memantau kinerja jaringan berdasarkan status *interface*, penggunaan CPU, penggunaan *bandwidth*, *network latency*, dan lain-lain.

- *Internet Control Message Point*

  ICMP (*Internet Control Message Point*) merupakan protokol yang dirancang khusus untuk *error reporting*. Perangkat jaringan memanfaatkan ICMP untuk mengirim pesan error ketika *host* atau *client* tidak bisa dihubungi atau informasi yang diminta tidak tersedia dan ICMP tidak terlibat dalam pertukaran data.

3.3. Elastic Stack

Elastic stack merupakan istilah dari ELK stack yang berubah setelah adanya penambahan komponen Beat didalam *stack*. Elastic stack merupakan *rich ecosystem* akan komponen *open source* yang berfungsi penuh sebagai *search* dan *analytic stack* [13]. Gambar 3.1 menjelaskan tentang komponen utama yang terdapat pada Elastic stack adalah Elasticsearch, Kibana, Logstash dan Beat. Elasticsearch merupakan komponen inti dari Elastic stack, dengan menyediakan fitur seperti penyimpanan, pencarian, dan analitik. Kibana merupakan komponen dengan sebutan sebagai 'jendela' bagi Elastic stack, hal ini dikarenakan Kibana memberikan visualisasi dan antarmuka pengguna yang bagus untuk Elastic stack. Logstash dan Beat berada diposisi yang sama dan merupakan komponen yang membantu untuk mendapatkan data yang akan di pakai di Elastic stack.



Gambar 3.1 Arsitektur standar dan komponen dari Elastic Stack

Dalam penerapannya Elastic stack dapat memenuhi kebutuhan bisnis yang berkembang dengan sangat efisien karena memiliki komponen yang saling terintegrasi dan dirancang untuk memberikan *actionable insights* secara *real-time* dari kumpulan pencarian data yang besar. Selain itu komunitas dari Elastic stack sangatlah aktif dalam mengembangkan Elastic stack kedepannya. Berikut beberapa poin penting tentang keunggulan Elastic stack :

- Elastic stack dapat dengan aman mengambil, menganalisis, dan memvisualisasikan data secara real-time dari berbagai sumber dan format.
- Elastic stack dapat melakukan *logging* secara terpusat untuk membantu mengidentifikasi permasalahan yang terjadi pada beberapa server sekaligus.
- Elastic stack disiapkan untuk menangani *big data* dan memberikan *business insights* yang penting untuk dapat dimanfaatkan.
- Elastic stack dibuat untuk dapat beradaptasi seiring bertumbuhnya kebutuhan data.
- Elastic stack mudah digunakan, dikonfigurasi, dan *user friendly*.
- Sebagai program open-source, Elastic stack sangat hemat biaya.

3.3.1. Elasticsearch

Elasticsearch adalah inti dari Elastic Stack dan memainkan peran yang penting sebagai mesin pencarian dan analitik [13]. Elasticsearch merupakan produk *open source* yang dikembangkan dengan bahasa pemrograman Java dan termasuk dalam database NoSQL. Elasticsearch bisa bekerja dengan data yang besar dan mampu untuk menyimpan dan melakukan pencarian didalamnya. Elasticsearch berbasis teknologi Apache Lucene dan menyediakan API untuk mempermudah penggunaan. Terdapat beberapa bagian penting yang ada di Elasticsearch, antara lain yaitu [13]:

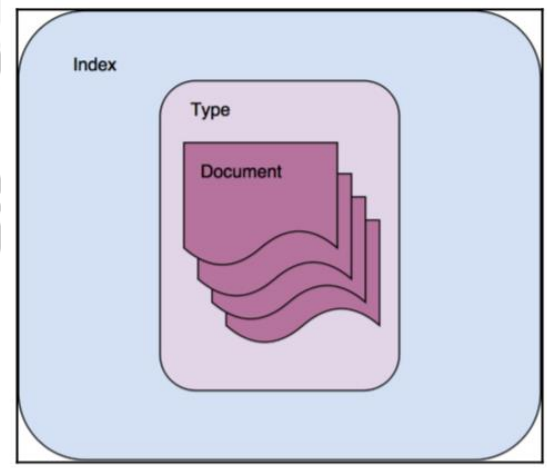
- *Document*

  Document terdiri dari beberapa *field* dan merupakan bentuk dasar informasi yang tersimpan di Elasticsearch. Setiap *field* yang terdapat pada *document* berbentuk JSON dan dilihat sebagai pasangan *key* dan *value* yang melambangkan nama kolom dan nilainya kalau dalam database relasional.

- *Type*

  Type membantu untuk mengelompokkan atau mengatur jenis dari *document* yang sama dalam suatu *index*. Hal ini dikarenakan Elasticsearch bersifat *schemaless*, yang dapat menyimpan JSON *document* dalam bentuk apapun, sehingga perlu menghindari percampuran *document* dengan entitas yang berbeda. Seperti contoh menyimpan data pelanggan dan produk, hal ini lebih masuk akal apabila menyimpannya dengan *type* yang berbeda dengan *index* yang terpisah.

- *Index*



Gambar 3.2 Susunan index di Elasticsearch

Index merupakan wadah yang menyimpan dan mengelola *document* dari satu jenis *type* dalam Elasticsearch. Dapat dilihat dari Gambar 3.2 *index* hanya terdiri dari satu jenis *type* dan istilah *index* digunakan ketika melakukan operasi *indexing, search, update,* dan *delete* pada *document* didalamnya.

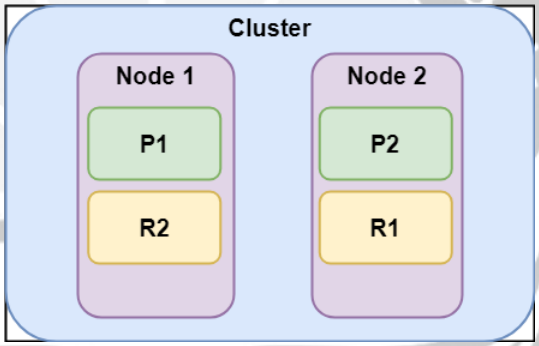
- *Node*

  Node mewakili sebuah server pada Elasticsearch yang dapat menjadi bagian dari *cluster node* yang lebih besar. Node berpartisipasi dalam *indexing*, *searching* dan operasi lainnya yang ada pada Elasticsearch. Didalam *node* terdapat *shards* dan *replicas* yang mewakili *index*.

- *Cluster*

  Cluster terdiri dari satu atau lebih *node* dan menyediakan operasi *indexing*, *searching*, dan agregasi menggunakan *node* yang ada. Elasticsearch *node* selalu menjadi bagian dari cluster, penambahan *node* secara *default* akan bergabung otomatis dengan *cluster* default dari Elasticsearch yang bernama 'Elasticsearch'.

- *Shards and Replicas*



Gambar 3.3 Contoh susunan shards dan replicas di Elasticsearch

Shards merupakan *cluster utility* yang bertugas untuk membagi data pada suatu *index*. Proses tersebut disebut *sharding*, dengan tujuan untuk dapat menghindari limit dan memanfaatkan *storage*, *memory*, dan kapasitas *processing* pada *node* yang berbeda agar dapat dimanfaatkan dengan lebih optimal. Sedangkan *replicas* merupakan salinan *shard* tambahan dari *primary shard* dan dapat memberikan nilai *availability* yang tinggi terhadap data. Gambar 3.3 dapat menjelaskan bahwa Elasticsearch sebagai sistem yang terdistribusi dapat tetap berjalan ketika terjadi *hardware failure* pada salah satu *node*.

Dari segi penggunaan, berikut manfaat dalam menggunakan Elasticsearch sebagai tempat penyimpanan data, antara lain [13]:

- Elasticsearch tidak memiliki struktur data yang ketat sehingga pengguna bisa menyimpan dokumen JSON dalam bentuk apapun.
- Elasticsearch mendukung *full-text search* yang dapat memberikan hasil pencarian yang sangat relevan.
- Elasticsearch mendukung beragam jenis agregasi untuk analitik.
- Elasticsearch mendukung *libraries* yang dapat digunakan di berbagai bahasa pemrograman seperti Java, C#, Python, JavaScript, PHP, Perl, Ruby, dan lain-lain.
- Elasticsearch menyediakan REST API yang bekerja di protokol HTTP dengan dokumentasi yang bagus dan komprehensif.
- Elasticsearch dapat berjalan dalam satu node, dan mudah untuk *scale up* tanpa harus mengubah konfigurasi. Serta memberikan kemampuan *Horizontal scalability* atau meningkatkan kemampuan sistem dengan cara menambahkan beberapa jenis sistem yang sama, hal ini lebih baik dari pada hanya membuat satu sistem *powerful* saja.
- Elasticsearch memberikan respon data hampir secara *real-time*, data dapat tersedia setelah sepersekian detik data tersimpan. Elasticsearch bisa menyimpan hingga ratusan ribu dokumen per detik dan dapat segera tersedia untuk keperluan *searching*.
- Elasticsearch secara *default* mengindeks setiap field yang ada pada dokumen. Hal ini mendukung penggunaan teknologi yang dipkai Elasticsearch yaitu Apache Lucene yang dapat mencari data dari semua *field* yang ada dokumen.
- Elasticsearch toleran terhadap *hardware failure* dan *network failure*. Jika terjadi *hardware failure* pada *node*, Elasticsearch bisa menyalin semua data yang ada pada *node* yang bermasalah dan dikirim ke *node* lain yang ada dalam *cluster*. Jika terjadi *network failure* pada *node*, Elasticsearch bisa memilih salinan replika node tersebut yang ada di node lain untuk menjadi master data sehingga data tetap aman.

3.3.2. Kibana

Kibana merupakan *tool open source* berbasis web untuk visualisasi dari Elastic stack yang dapat memberikan wawasan secara luas tentang data yang ada di Elasticsearch. Kibana menawarkan beragam jenis visualisasi data termasuk *histogram*, *time series*, *chart*, *graphs*, *maps*, *table*, dan lain-lain. Pengguna dapat membuat dashboard dengan mengkombinasikan beberapa jenis visualisasi yang berbeda dengan hasil yang interaktif dan berkualitas [13]. Kibana juga memiliki *tool* untuk keperluan manajemen dan pengembangan. Didalamnya bisa digunakan untuk manajemen beberapa fitur yang digunakan Elastic stack dan juga digunakan pengembang untuk membuat dan menguji *request* REST API.

Dalam penggunaanya Kibana digunakan untuk mencari, menampilkan, menganalisis, dan berinteraksi secara *real-time* dengan data yang sudah tersimpan di *cluster* Elasticsearch. Kibana secara *built in* dapat mendukung beberapa kasus penggunaan seperti *logging* dan *log analytics*, *infrastructure metrics* dan *container monitoring*, *geospatial data analysis* dan *visualization* , *security analytics*, dan *bussines analytic*s.

3.3.3. Logstash

Logstash merupakan mesin pengumpul data *open source* dengan kemampuan *pipelining* secara realtime. *Pipelining* merupakan proses yang memiliki tiga tahap yaitu *input*, *filter*, dan *output*. Logstash memiliki sekitar 200 lebih *plugin* yang dapat dipakai pada setiap tahap (*input*, *filter*, dan *output*) dan akan terus bertambah. Logstash termasuk komponen yang berada di sisi server, perannya adalah memusatkan kumpulan data dari berbagai *input*, mentransformasikan dalam berbagai format, dan mengirim data ke berbagai tujuan *output* [13].

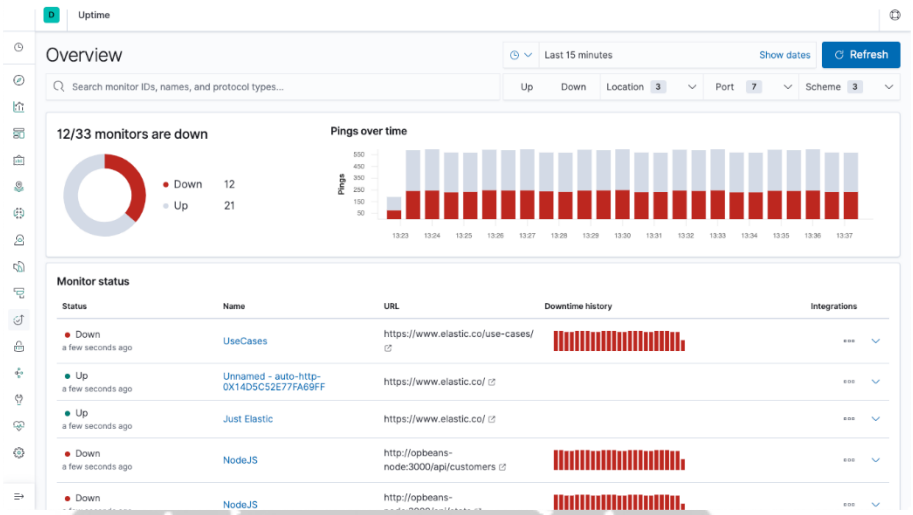
3.3.4. Beats

Beats merupakan *platform open source* pengirim *lightweight data* [13]. Perannya hampir sama seperti Logstash yaitu pengumpul dan pengirim data, yang membedakan adalah Logstash merupakan komponen yang berada di sisi server, sedangkan Beats berada di sisi *client*. Dalam penggunaanya Beats dapat mengirim data ke Logstash atau langsung ke Elastcisearch. Beats memiliki beragam jenis, masing-masing bertujuan untuk mengumpulkan data dari sumber yang spesifik. Saat ini sudah ada 95 Beats buatan komunitas dan tujuh buatan resmi dari tim Elastic, dan akan terus bertambah. Beats yang dibuat oleh tim Elastic adalah Filebeat, Metricbeat, Packetbeat, Winlogbeat, Auditbeat, Heartbeat, dan Functionbeat.

3.3.4.1 Heartbeat

Heartbeat merupakan salah satu Beats yang digunakan untuk memeriksa apakah *service* sedang up atau down, dan memeriksa apakah *service* masih dapat diakses [13]. Heartbeat dipasang di sisi *client* dan secara berkala akan memeriksa status dari *service* atau *host* yang dipantau. Cara kerja Heartbeat sangatlah sederhana yaitu menanyakan "apakah kamu hidup?" kepada suatu *service* atau *host*, hampir sama dengan melakukan *ping* di komputer. Untuk melakukannya Heartbeat menggunakan beberapa protokol seperti ICMP, TCP, dan HTTP.

Dalam penggunaanya, Heartbeat dapat melakukan ping terhadap *host* dengan menggunakan *hostname* (www.contoh.com) atau menggunakan IP address. Heartbeat dapat dikonfigurasi untuk melakukan ping setiap berapa detik/menit/jam atau pada rentan waktu tertentu (07.00-09.00). Disetiap konfigurasi *host* yang akan di monitor, bisa ditambahkan informasi seperti *geo metadata* yang berisi nama lokasi dan lokasi yang berupa *latitude* dan *longtitude* untuk kepentingan visualisasi. Heartbeat juga memiliki *dashboard* yang sudah *built in* di Kibana yang bernama Uptime seperti pada Gambar 3.4.



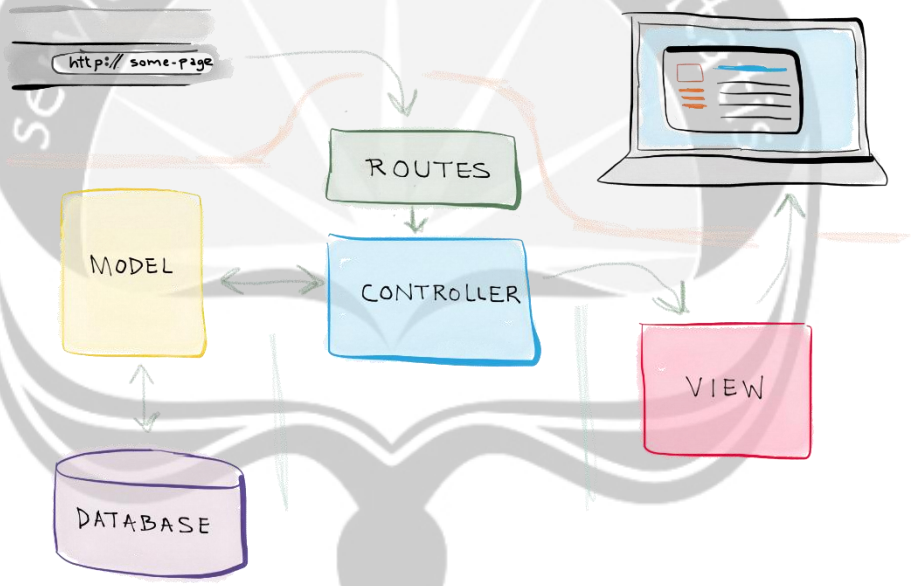
Gambar 3.4 Dashboard Uptime di Kibana

3.4. Laravel

Laravel merupakan framework PHP *open source* yang didesain untuk dapat membangun aplikasi web dengan mudah dan cepat menggunakan fitur-fitur *built in* yang ada didalamnya [14]. Laravel termasuk framework yang berada di sisi server, sehingga memungkinkan untuk membangun REST API *service* atau secara *back-end*, maupun membangun aplikasi secara *full-stack*. Hal ini dikarenakan Laravel menggunakan pola MVC (*Model-View-Controller*) yang dapat digunakan sesuai kebutuhan dan bertujuan untuk memisah komponen menjadi tiga bagian dengan tugasnya masing-masing yaitu model merepresentasikan data, view merepresentasikan antarmuka, dan controller merepresentasikan operasi terhadap data. Pada Gambar 3.5 menjelaskan penerapannya pola MVC pada Laravel didukung dengan komponen *router* yang bertugas untuk mengarahkan *request* untuk mendapat *response* sesuai dengan konfigurasi bisa berupa *view* atau laman pada web maupun data yang berbentuk JSON [15]. Terdapat beberapa fitur yang membuat Laravel unggul, antara lain :

- Memiliki sistem *bundles* atau sistem pengemasan modular yang mempermudah pengguna untuk menambahkan fungsionalitas ke projek Laravel tanpa menulisnya dari awal.

- Menerapkan Eloquent ORM (*Object-Relational Mapping*) yang menyajikan tabel basis data sebagai kelas objek untuk mempermudah akses dan manipulasi data.
- Memiliki CLI (Command-Line Interface) yang memiliki banyak perintah *built in*.
- Memiliki sistem *authentication built in* untuk keperluan *login* dan *session* pada projek.
- Memiliki unit testing yang dapat digunakan mendeteksi dan mencegah kesalahan pada projek.



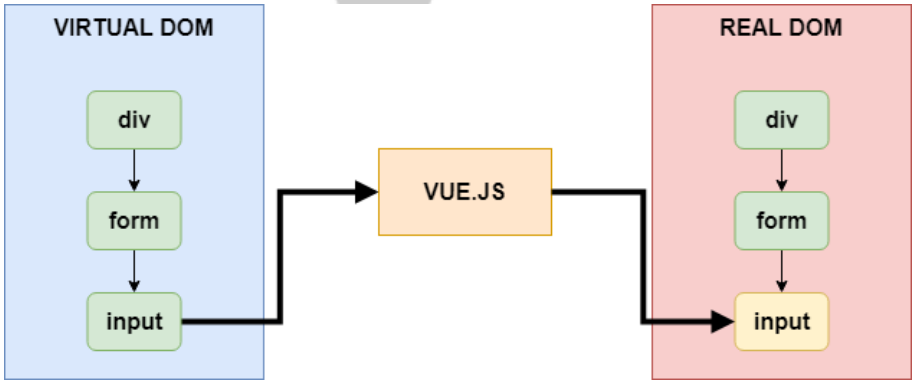
Gambar 3.5 Konsep MVC pada Laravel

3.5. Vue.Js

Vue.Js merupakan framework bahasa pemrograman JavaScript yang progresif untuk keperluan membangun antarmuka pengguna [14]. Framework ini dirancang dengan *core* yang terlihat memiliki fitur yang minimalis, tetapi framework ini menawarkan beberapa *library* yang dapat dipakai untuk memenuhi kebutuhan penggunanya sehingga menjadi framework yang lengkap. Bagian core pada Vue.Js hanya berfokus pada *view layer* saja dan

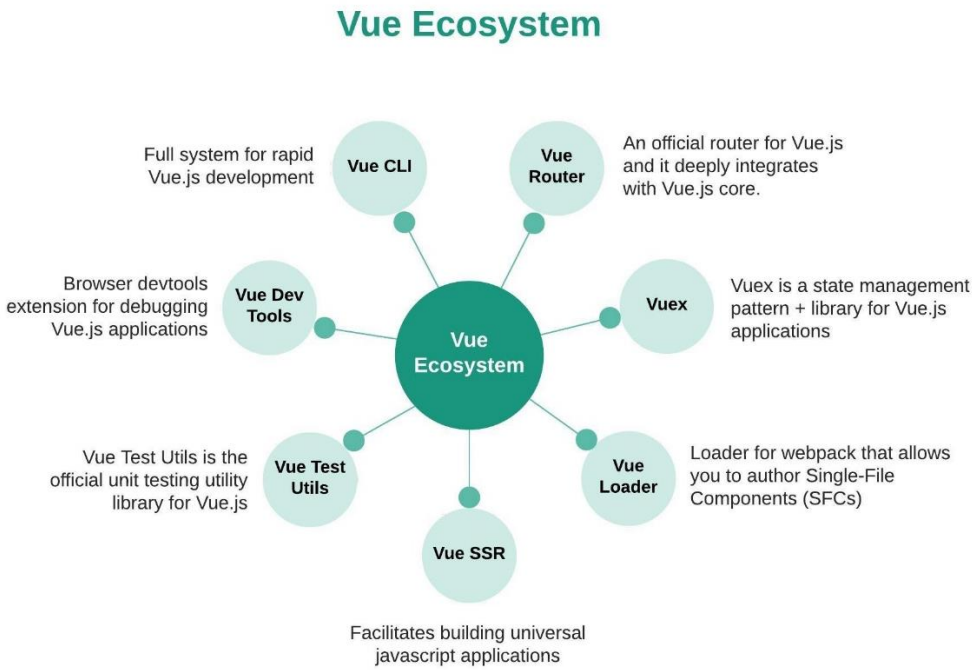
ekosistem pendukung seperti *library* akan membantu pengguna dalam mengatasi kompleksitas dalam membangun SPA (*Single-Page Aplications*) [16]. SPA merupakan aplikasi yang bekerja didalam *browser* dan tidak membutuhkan *reload* halaman untuk menggunakan semua fitur yang ada, ini hanya satu halaman *webite* yang memuat semua konten pada *website* tersebut dan dimuat sesuai keperluan.

Pada bagian core, Vue.Js menggunakan data reaktif dan *dependency tracking* yang cerdas untuk mencari tahu bagian mana dari antarmuka pengguna yang perlu untuk diperbarui. Vue.Js juga menggunakan teknik *virtual* DOM (*Document Object Model*) sehingga Vue.Js secara efisien dapat menciptakan sejumlah perubahan pada antarmuka pengguna untuk diterapkan pada *real* DOM dan kemudian dapat memperbarui antarmuka pengguna dengan sangat cepat [16]. DOM itu sendiri merupakan cara JavaScript untuk melihat objek yang mewakili element-element HTML (*div*, *body*, *head*, dll) yang digunakan pada suatu halaman website. Seperti pada Gambar 3.6 penggunaan *virtual* DOM dapat meminimalisir biaya proses untuk memperbarui *real* DOM yang notabene memerlukan biaya mahal secara komputasi. *Real* DOM akan secara penuh merender data yang sama ketika *virtual* DOM hanya mempertahankan *state* data tanpa merender ulang. Hal inilah yang membuat Vue.Js dapat secara efisien dan cepat dalam menciptakan perubahan beserta pembaruan pada antarmuka pengguna.



Gambar 3.6 Konsep Virtual DOM pada Vue.Js

Pada Gambar 3.7 menjelaskan bawha Vue.Js memiliki ekosistem yang kaya, baik dari *library* dan dari framework yang terkait dengan Vue.Js seperti Vuetify, BoostrapVue, dan lain-lain. Terdapat 7 *library* pada Vue ekosistem yang dikelola oleh tim Vue yaitu Vue Router, Vuex, dan Vue Server Renderer termasuk *core library* dan Vue Loader, Vue Test Utils, Vue Dev-Tools, dan Vue CLI termasuk *tool library*. Dengan library yang lengkap Vue.Js memungkinkan pengembang untuk menulis *code* seluruh aplikasi *front-end* dalam JavaScript dengan arsitektur berbasis komponen.



Gambar 3.7 Ekosistem Library dari Vue.Js

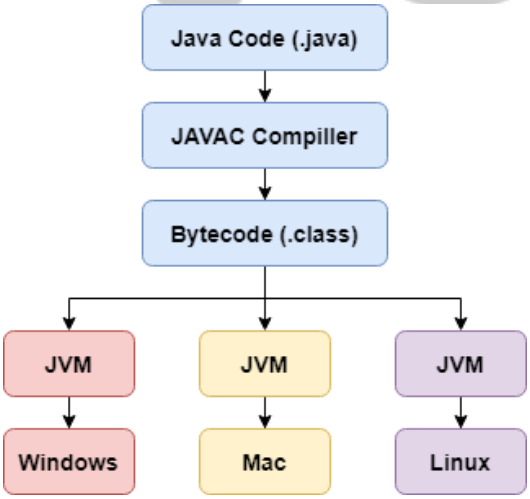
3.5.1. Vuetify

Vuetify merupakan framework dari komponen UI (*User Interface*) Vue.Js yang dibangun sesuai dengan *Material Design* sehingga memungkinkan pengguna dapat membangun tampilan *front-end* pada *website* dengan bagus dan cepat. *Material Design* itu sendiri merupakan bahasa visual yang mencampurkan prinsip klasik dengan inovasi teknologi dan sains. Vuetify merupakan salah satu ekosistem Vue.Js dengan komunitas yang aktif dengan

fitur yang selalu *update*. Pada penggunaanya Vuetify mendukung pada penggunaan di berbagai macam *browser* dan penggunaan yang mudah dengan didukung dokumentasi yang lengkap.

3.6.Java

Java merupakan bahasa pemrograman yang berorientasi objek atau biasa disebut OOP (Object Oriented Programming) yang *multiplatform* [17]. Disebut *multiplatform* dikarenakan Java merupakan platform bersifat netral yang berarti tidak terikat dengan sistem operasi tertentu dan dapat berjalan tanpa modifikasi *code* pada sistem operasi yang berbeda. Java memegang teguh slogan "*Write Once, Run Anywhere*" dengan tujuan utama adalah membuat bahasa pemrograman yang sangat handal, portabel, dan sederhana. Hal ini bisa terjadi karena peran JVM (*Java Virtual Machine*) pada platform yang ingin menjalanakan aplikasi berbasis Java seperti yang dijelaskan pada Gambar 3.8. Awalnya file source code java di *compile* menggunakan *compiller* buatan Java yaitu Javac dan menghasilkan file berekstensi *.class* yang berisi *bytecode* atau set intruksi untuk JVM. Selanjutnya JVM akan mengubah *bytecode* tersebut ke bahasa mesin yang sesuai arsitektur mesin komputer yang menjalankan [18]. Pada penggunaanya Java biasa dipakai pada pengembangan aplikasi *mobile* seperti Android, aplikasi *desktop*, dan juga *web service*.



Gambar 3.8 Cara kerja Java dapat berjalan di multiplatform

3.7.MySQL

MySQL merupakan sistem manajemen basis data relasional atau biasa disebut RDBMS (*Realtional Database Management System*) [19]. MySQL termasuk sistem yang *open source* dan mengimplementasikan SQL (*Structured Query Language*) yang merupakan bahasa *query* yang dirancang untuk mengelola data yang disimpan di *database*. Disebut database relasional karena MySQL menyimpan data dalam tabel yang terpisah dan terstruktur daripada menempatkan semua data dalam satu penyimpanan besar, hal ini sangat berdampak pada kecepatan mengambil data. MySQL *server* dapat berjalan di beragam sistem operasi seperti Linux, Windows, maupun Mac dan juga menawarkan konektivitas, kecepatan, dan keamanan sehingga cocok jika digunakan sebagai server *database* di internet.

BAB VI. PENUTUP

6.1. Kesimpulan

Berdasarkan hasil dari perancangan, pembahasan, dan pengujian Jedi NMS, maka dapat disimpulkan beberapa hal antara lain:

1. Jedi NMS mempunyai tampilan *user interface* yang mudah digunakan dan dipahami.
2. Jedi NMS berhasil membantu perusahaan PT. Jedi Global Teknologi untuk melakukan layanan *monitoring* jaringan *client* berupa *custom dashboard* pada Elastic Stack dan mengelola data *asset* dari *client* yang dimonitor pada NMS Asset.
3. Jedi NMS memberikan fasilitas *mobile* yang dapat memberikan informasi dan notifikasi dari layanan *monitoring* jaringan *client* pada NMS Mobile.
4. Salah satu teknologi Jedi NMS yang digunakan yaitu Elastic Stack memiliki nilai *scalability* dan *availability* yang tinggi sehingga dapat memenuhi kebutuhan perusahaan PT. Jedi Global Teknologi yang akan terus berkembang.
5. Jedi NMS berhasil membantu perusahaan PT. Jedi Global Teknologi dalam menjalankan layanan *monitoring* jaringan *client* yang sebelumnya harus membuka lebih dari satu panel *dashboard*, sekarang hanya cukup membuka satu panel *dashboard*.

6.2. Saran

Berikut adalah saran dari hasil pengembangan dan pengujian Jedi NMS:

1. Penelitian selanjutnya diharapkan dapat mengintregasikan lebih, baik dari fitur dan manajemen di *platform* Elastic Stack pada *platform* lainnya seperti NMS Asset dan NMS Mobile.
2. Penelitian selanjutnya diharapkan dapat memberikan fitur untuk menerima atau tidak notifikasi dari server sehingga pengguna yang merasa mendapatkan spam notifikasi bisa menonaktifkan fitur notifikasi pada NMS Mobile.

DAFTAR PUSTAKA

[1] M. Ali, M. Alam, and M. Mustafa, "Network Device Monitoring System with SMS Alert," *Br. J. Appl. Sci. Technol.*, vol. 17, no. 6, pp. 1–7, 2016.

[2] R. Rinaldo, "Implementasi Sistem Monitoring Jaringan Menggunakan Mikrotik Router Os Di Universitas Islam Batik Surakarta," *J. Emit.*, vol. 16, no. 02, 2016.

[3] D. Wijonarko, "Zabbix Network Monitoring Sebagai Perangkat Monitoring Jaringan Di Skpd Kota Malang," *J. ELTEK*, vol. 12, no. 1, pp. 27–38, 2017.

[4] I. Ghafir, V. Prenosil, J. Svoboda, and M. Hammoudeh, "A survey on network security monitoring systems," *Proc. - 2016 4th Int. Conf. Futur. Internet Things Cloud Work. W-FiCloud 2016*, pp. 77–82, 2016.

[5] D. Berman, "The Complete Guide to the ELK Stack," *Logz.io*, 2019. [Online]. Available: https://logz.io/learn/complete-guide-elk-stack/. [Accessed: 21-May-2020].

[6] P. Sokibi, "Perancangan Sistem Monitoring Perangkat Jaringan Berbasis ICMP dengan Notifikasi Telegram," *Inf. Technol. Eng. Journals*, vol. 02, no. 02, 2017.

[7] I. Y. M. Al-Mahbashi, M. B. Potdar, and P. Chauhan, "Network security enhancement through effective log analysis using ELK," *Proc. Int. Conf. Comput. Methodol. Commun. ICCMC 2017*, vol. 2018-Janua, no. Iccmc, pp. 566–570, 2018.

[8] B. Raja, K. Ravindranath, and B. Jayanag, "Monitoring and analysing anomaly activities in a network using packetbeat," *Int. J. Innov. Technol. Explor. Eng.*, vol. 8, no. 6, pp. 45–49, 2019.

[9] D. Permana, "Jenis dan Manfaat Jaringan Komputer yang Harus Diketahui," *Tedas.id*, 2019. [Online]. Available: https://tedas.id/teknologi/komputer/manfaat-jaringan-komputer/. [Accessed:

26-May-2020].

[10] I. Lawal, “Computer Networking Introduction for Beginners,” *Gleekflare*, 2019. [Online]. Available: https://geekflare.com/computer-networking-basics/. [Accessed: 26-May-2020].

[11] B. P. Winasis and B. Sugiantoro, “Design and Implementation of Network Monitoring System on Local Area Network with Social Media Twitter Notification,” *IJID (International J. Informatics Dev.*, vol. 6, no. 2, p. 1, 2018.

[12] liveaction, “Types of Network Monitoring Protocols,” *liveaction*, 2018. [Online]. Available: https://www.liveaction.com/blog/types-of-network-monitoring-protocols/. [Accessed: 27-May-2020].

[13] P. Shukla and S. Kumar, *Learning Elastic Stack 6.0*, vol. 53, no. 9. Birmingham: Packt Publishing, 2017.

[14] I. K. Aditya, H. Putra, D. Pramana, N. Luh, and P. Srinadi, “Sistem Manajemen Arsip Menggunakan Framework Laravel dan Vue . Js (Studi Kasus : BPKAD Provinsi Bali),” *Sist. dan Inform.*, vol. 13, no. 2, pp. 97–104, 2019.

[15] E. Simanjuntak, “Pengenalan Laravel Framework,” *Medium*, 2019. [Online]. Available: https://medium.com/easyread/pengenalan-laravel-framework-1c829b8164af. [Accessed: 31-May-2020].

[16] S. Nigam, “Thinking in components with Vue.js,” *Medium*, 2019. [Online]. Available: https://medium.com/@_shirish/thinking-in-components-with-vue-js-a35b5af12df. [Accessed: 30-May-2020].

[17] A. Abdullah and E. Utami, “Analisis dan Perancangan Sistem Informasi SKB Kab Kubu Raya Menggunakan Konsep MVC Dalam Bahasa Pemrograman Java,” *Cybernetics*, vol. 1, no. 01, pp. 51–57, 2017.

[18] R. Gour, “Introduction to Java Programming Language,” *Medium*, 2018. [Online]. Available: https://medium.com/@rinu.gour123/introduction-to-java-programming-language-389033edb91f. [Accessed: 31-May-2020].

[19] M. Muslihudin and A. Larasati, “Perancangan Sistem Aplikasi Penerimaan Mahasiswa Baru Di Stmik Pringsewu Menggunakan Php Dan Mysql,” *J. TAM*, vol. 3, pp. 32–39, 2014.