**JURUSAN TEKNIK INFORMATIKA**

**FAKULTAS TEKNOLOGI INFORMASI**

**INSTITUT TEKNLOGI SEPULUH NOPEMBER**

**USULAN TUGAS AKHIR**

# IDENTITAS PENGUSUL

**NAMA : Bahrul Halimi**

**NRP : 5111 100 014**

**DOSEN WALI : Prof.Ir. Handayani Tjandrasa, M.Sc, Ph.D**

**DOSEN PEMBIMBING : 1. Royyana Muslim Ijtihadie, S.Kom, M.Kom, PhD  
 2. Baskoro Adi Pratomo, S.Kom, M.Kom**

# JUDUL TUGAS AKHIR

“**Rancang Bangun Sistem *Load Balancing* Menggunakan Algoritma Berbasis Konten dan Kontrol Ketersediaan Layanan**”

# LATAR BELAKANG

Semakin berkembangnya internet di masyarakat membuat penggunaan aplikasi berbasis web semakin diminati. Pengguna aplikasi berbasis web ini dapat dijumpai diberbagai aktivitas harian masyarakat, diantaranya bank *online*, *e-commerce*, reservasi tempat secara *online*, bahkan pendaftaran peserta didik baru secara *online*. Hal ini membuat penyedia layanan aplikasi berbasis web harus menyediakan servis yang layak sehingga aplikasi tetap berjalan dengan baik walaupun pengguna semakin bertambah.

Terjadinya *bottleneck* (penumpukan permintaan) menjadi tantangan tersendiri ketika pengembang tidak memperhatikan sumber dayanya dan berujung pada gagalnya permintaan pengguna [1]. Muncul gagasan awal dengan penggunaan kelompok server yang akan menangani permintaan ini. Kelompok server ini akan secara bergantian melayani setiap permintaan terhadap aplikasi berbasis web ini. Dengan adanya tugas bergantian ini dibutuhkan sebuah komputer yang bertugas membagi beban kerja kelompok server. Komputer ini biasa disebut pembagi muat atau *load balancer*. Sistem kerja dari *load balancer* ini menggunakan sebuah algoritma yang sudah ditanam untuk kemudian digunakan untuk memilih komputer mana yang harus melayani permintaan pengguna.

Di sisi lain sebuah aplikasi berbasis web memiliki dua jenis halaman yang mungkin di akses. Yang pertama adalah halaman berisi informasi, baik hasil *query* basis data maupun tidak, selanjutnya disebut halaman informasi dan yang kedua adalah halaman yang digunakan untuk mengirimkan data ke server, dalam hal ini berupa form pengisian informasi, selanjutnya disebut halaman daftar.

Dua jenis halaman ini memiliki kebutuhan yang berbeda. Untuk halaman informasi, pengguna mengharapkan akses yang cepat sedangkan untuk halaman daftar, pengguna mengharapkan data yang dimasukkan dapat diproses dengan aman. Padahal di dalam penggunaan algoritma sebelumnya dan dengan teknologi yang ada, *load balancer* tidak dapat memisahkan dua jenis permintaan ini. Algoritma yang ada sebelumnya hanya memisahkan banyak permintaan sesuai dengan ketersediaan server melayani pengguna. Padahal ketika proses pemasukkan data di dalam halaman daftar, seharusnya bisa digunakan untuk melayani permintaan pada halaman informasi.

Muncullah gagasan lain mengenai pengelompokkan permintaan berdasarkan konten yang diinginkan oleh pengguna. Pengelompokkan ini didasarkan pada dua halaman sebelumnya, yakni halaman informasi dan halaman daftar. Tujuannya untuk mengatur penggunaan sumber daya yang digunakan. Dua kelompok server terpisah akan melayani masing-masing permintaan yang berbeda. Dengan permintaan satu tipe dalam satu kelompok server, membuat kerja server menjadi lebih terpusat dan mengurangi beban yang besar.

Berbeda dengan yang terjadi saat ini, sebuah server atau bahkan dalam sebuah kelompok server, harus melayani berbagai bentuk permintaan dari pengguna, sehingga menyebabkan beban kerja server meningkat. Bahkan waktu dalam penyelesaian suatu permintaan tidak dapat diukur dalam satuan waktu yang sama karena berbedanya bentuk permintaan pengguna.

Sementara itu di dalam kelompok server yang bekerja bergantian melayani permintaan, ada kalanya sebuah server mengalami gangguan dan sama sekali tidak dapat melayani setiap permintaan pengguna. Padahal setiap permintaan yang ada masih diteruskan oleh load balancer pada server tersebut. Tidak adanya mekanisme untuk memindahkan permintaan dari server mati ke server yang masih aktif membuat akses ke sebuah web menjadi tidak maksimal.

Oleh karena itu dibangunlah sistem ini. Dengan adanya sistem *load balancing* menggunakan algoritma berbasis konten yang memisahkan antara halaman informasi dan halaman daftar diharapkan dapat meningkatkan jumlah pengguna suatu halaman web dengan banyaknya bentuk permintaan dari pengguna.

# RUMUSAN MASALAH

Berikut beberapa hal yang menjadi rumusan masalah dalam tugas akhir ini:

1. Bagaimana membagi beban kerja server berdasarkan konten permintaan pengguna ?
2. Bagaimana menentukan pengelompokkan server berdasarkan konten permintaan pengguna ?
3. Bagaimana meningkatkan jumlah pengakses pada halaman informasi dengan terpisahnya akses antara halaman informasi dan halaman daftar ?
4. Bagaimana menjaga pengguna tetap dilayani kelompok server yang tersedia hingga permintaan selesai ?

# BATASAN MASALAH

Dari permasalahan yang telah diuraikan di atas, terdapat beberapa batasan masalah pada tugas akhir ini, yaitu:

1. Konten permintaan pengguna dilihat dari URL yang diakses.
2. Pendefinisian kelompok konten permintaan pengguna dilakukan manual oleh manusia.
3. Sistem pembagi beban kerja diimplementasikan untuk aplikasi berbasis web.
4. Kelompok server yang bekerja dibedakan dengan besar memori yang digunakan.

# TUJUAN PEMBUATAN TUGAS AKHIR

Tugas akhir dibuat dengan beberapa tujuan. Berikut beberapa tujuan dari pembuatan tugas akhir:

1. Mampu mengategorikan permintaan pengguna terhadap suatu web berdasarkan halaman yang diakses pengguna.
2. Mampu melayani banyaknya permintaan pengguna dengan mengandalkan pengelompokan komputer.
3. Mampu meningkatkan jumlah pengakses yang dilayani dengan berhasil oleh aplikasi dibandingkan dengan akses tanpa pemisahan jenis halaman yang diakses.

# MANFAAT TUGAS AKHIR

Dengan dibangunnya *load balancer* ini diharapkan jumlah pengakses yang mampu dilayani oleh kelompok server untuk halaman informasi menjadi lebih banyak dibandingkan dengan penggunaan algoritma dan teknologi *load balancing* yang sudah ada.

# TINJAUAN PUSTAKA

* 1. Algoritma Berbasis Konten

Munculnya algoritma ini didasarkan pada beberapa jenis permintaan pengguna yang mengakses suatu halaman web. Sebuah server melayani berbagai jenis permintaan akan memberikan waktu balasan yang beragam pula. Hal ini akan meningkatkan beban kerja server.

Dengan adanya pemisahan permintaan pengguna berdasarkan konten, kelompok server akan melayani setiap permintaan yang memang ditujukan untuknya server tersebut. Bentuk permintaan akan selalu sama sehingga waktu untuk melayani permintaan menjadi sama dan lebih terkontrol. Beban kerja server akan lebih ringan dengan adanya pembagian beban berdasarkan algoritma ini. [1]

* 1. Node JS

Merupakan sebuah platform yang dibangun di atas Chrome's JavaScript runtime dengan teknologi V8 yang mendukung proses server yang bersifat *long-running*. Tidak seperti platform modern yang mengandalkan multithreading, NodeJS memilih menggunakan asynchronous I/O eventing. Karena inilah NodeJS mampu bekerja dengan konsumsi memori rendah. [2] [3]

* 1. Angular JS

Angular JS membantu dalam pembangunan halaman HTML menjadi lebih dinamis. Merupakan sebuah kumpulan alat bantu yang mampu bekerja baik dengan pustaka lainnya. Setiap fitur dapat dimodifikasi sesuai dengan kebutuhan aplikasi. Menjadi salah satu kerangka kerja yang memfasilitasi pembangunan aplikasi kompleks yang terorganisir dan mudah dirawat. [4] [5]

* 1. MongoDB

Merupakan salah satu NoSQL (*Not only* SQL) terkenal yang dirancang untuk mengelola polimorfik, obyek, dan struktur data yang terus berkembang. MongoDB adalah basis data *open-source* yang memungkinan mengubah skema dengan cepat sementara fungsi yang diharapkan dari basis data tradisional masih berjalan. [6] [7]

* 1. Apache JMeter

Menjadi salah satu alat bantu untuk melakukan tes muat dan mengukur performa aplikasi, salah satunya berbasis web. Mampu melakukan pengujian pada berbagai protokol diantaranya Web, FTP, basis data, Mail (SMTP, POP3, IMAP), serta MongoDB. [8]

# RINGKASAN ISI TUGAS AKHIR

Pendekatan yang digunakan sistem ini adalah adanya pemisah antara permintaan pada halaman informasi dan permintaan pada halaman daftar. Pemisahan dua jenis permintaan pengguna ini didasarkan pada URL yang diakses dan hal ini dilakukan oleh *load balancer*. Setiap permintaan pengguna akan melewati *load balancer* yang ditanami algoritma berbasis konten untuk membagi kerja kelompok komputer yang ada.



Gambar 1 Alur kerja load balancer

Pada Gambar 1 ditunjukkan bahwa setiap permintaan pengguna masuk ke dalam *load balancer*. Selanjutnya *load balancer* akan menentukan ke kelompok komputer mana permintaan akan diteruskan. Penentuan ini didasarkan pada data yang ada di dalam basis data. Basis data ini berisi daftar URL valid yang dapat diakses pengguna dan jenis permintaan pengguna.

Basis data yang berisi URL dan jenis permintaan pengguna diisi oleh penyedia layanan web. Terdapat sebuah antarmuka yang nantinya akan memudahkan pengisian data di dalam basis data. Untuk pengelompokkan server yang digunakan, juga dilakukan oleh pengembang.

Akan ada sistem monitoring dimana setiap server yang bekerja akan ditampilkan statusnya. Ketika ada satu server yang mengalami masalah dalam melayani permintaan, *load balancer* akan otomatis memindahkan setiap permintaan dari server tersebut ke server lain yang masih aktif. Hal ini dilakukan otomatis oleh *load balancer*. Dari sisi penyedia layanan akan tampil server mana yang gagal memberi layanan. Dan ketika server kembali dapat digunakan, *load balancer* akan melanjutkan pengiriman permintaan ke server ini.

# METODOLOGI

## Penyusunan proposal tugas akhir

Proposal tugas akhir ini berisikan pembangunan *load balancer* dengan algoritma berbasis konten yang akan memisahkan permintaan pengguna berdasarkan URL yang diakses. Setiap URL dicocokkan dengan data di dalam basis data kemudian permintaan diteruskan ke kelompok server yang sesuai.

## Studi literatur

Tugas akhir ini menggunakan literatur *paper* beserta artikel dari internet. *Paper* yang menjadi inspirasi dalam pengerjaan tugas akhir ini adalah *“A content-based load balancing algorithm with admission control for cluster web servers”* dan *“Node.js: Using JavaScript to Build High-Performance Network Programs”*. *Paper* tersebut menjelaskan mengenai penggunaan algoritma *content-based* untuk *load balancing* dan penggunaan Node.js dalam aplikasi berbasis web.

## Analisis dan Desain Perangkat Lunak

*Load balancer* berjalan pada sebuah server yang terhubung dengan dua kelompok komputer. Setiap permintaan dari pengguna akan diarahkan pada alamat *load balancer*. Akan ada dua kelompok server yang nantinya akan melayani permintaan yang berbeda sesuai dengan jenis permintaan pengguna. Setiap komputer yang ada di masing-masing kelompok server menjalankan aplikasi berbasis web yang sama.



Gambar 2 Arsitektur Jaringan

## Implementasi perangkat lunak

Dalam pembuatan *load balancer*, digunakan beberapa teknologi untuk dapat mengaplikasikan rancangan yang sudah ada, diantaranya:

1. Bahasa Pemrograman Aplikasi

*Load balancer* dibangun dengan menggunakan bahasa pemrograman Javascript karena NodeJS yang digunakan sudah menggunakan bahasa pemrograman ini.

1. IDE

Pengembangan aplikasi ini menggunakan Sublime 2 sebagai IDE.

1. Modeling Tools

Beberapa *modeling tools* yang digunakan untuk mengembangkan aplikasi ini Power Designer 15.00, StarUML, Microsoft Visio 2013.

## Pengujian dan Evaluasi

## Pengujian dari Rancang Bangun Sistem *Load Balancing* Menggunakan Algoritma Berbasis Konten dan Kontrol Ketersediaan Layanan akan diujikan pada Laboraturium Arsitektur dan Jaringan Komputer pada Teknik Informatika ITS dan yang akan diujikan antara lain sebagai berikut :

## Melakukan ujicoba pengelompokkan permintaan pengguna.

## Melakukan ujicoba dengan jumlah pengakses yang semakin meningkat, baik pada halaman informasi maupun halaman daftar.

## Membandingkan jumlah permintaan terlayani antara sistem baru dan sistem yang sudah ada sebelumnya.

## Penyusunan Buku Tugas Akhir

Pada tahap ini dilakukan penyusunan laporan yang menjelaskan dasar teori dan metode yang digunakan dalam tugas akhir ini serta hasil dari implementasi aplikasi perangkat lunak yang telah dibuat. Sistematika penulisan buku tugas akhir secara garis besar antara lain:

1. Pendahuluan
   1. Latar Belakang
   2. Rumusan Masalah
   3. Batasan Tugas Akhir
   4. Tujuan
   5. Metodologi
   6. Sistematika Penulisan
2. Tinjauan Pustaka
3. Desain dan Implementasi
4. Pengujian dan Evaluasi
5. Kesimpulan dan Saran
6. Daftar Pustaka

# JADWAL KEGIATAN

# Tabel 1 merupakan jadwal kegiatan dari pengerjaan tugas akhir Rancang Bangun Sistem *Load Balancing* Menggunakan Algoritma Berbasis Konten dan Kontrol Ketersediaan Layanan.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Tahapan | 2014 | | | | | | | | | | | | | | | | | | | | |
| Maret | | | | April | | | | Mei | | | | Juni | | | | | Juli | | | |
| Penyusunan Proposal |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Studi Literatur |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Perancangan sistem |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Implementasi |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Pengujian dan evaluasi |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Penyusunan buku |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Tabel 1. Jadwal Kegiatan Tugas Akhir

# DAFTAR PUSTAKA

|  |  |
| --- | --- |
| [1] | S. Sharifian, S. A. Motamedi and M. K. Akbari, "A content-based load balancing algorithm with admission control for cluster web servers," *Future Generation Computer Systems,* vol. 24, pp. 775-787, 2008. |
| [2] | I. Joyent, "NodeJS," Joyent, Inc, [Online]. Available: https://nodejs.org/. [Accessed 01 04 2015]. |
| [3] | S. Tilkov and S. Vinoski, "Node.js: Using JavaScript to Build High-Performance Network Programs," *IEEE Computer Society Issue,* vol. 6, pp. 80-83, 2010. |
| [4] | J. D. Walker and S. C. Chapra, "A client-side web application for interactive environmental simulation modeling," *Environmental Modelling & Software,* vol. 55, pp. 49-60, 2014. |
| [5] | Google, "AngularJS," Google, [Online]. Available: https://angularjs.org/. [Accessed 07 04 2015]. |
| [6] | M. Zhang, J. Zhang, W. Zheng, F. Hu and G. Zhuang, "A self-description data framework for Tokamak control system design," *Fusion Engineering and Design,* p. 5, 2015. |
| [7] | I. MongoDB, "mongoDB," MongoDB, Inc., [Online]. Available: http://www.mongodb.org/. [Accessed 08 04 2015]. |
| [8] | A. S. Foundation, "Apache JMeter," Apache Software Foundation, [Online]. Available: http://jmeter.apache.org/. [Accessed 08 04 2015]. |