Optimasi Server SIMAK Menggunakan Memcached dan Mirror Server Untuk Meningkatkan Kecepatan Akses Layanan Akademik Universitas Siliwangi

Eka Wahyu Hidayat[1] dan Alam Rahmatulloh [2]

[1] Teknik Informatika, Fakultas Teknik, Universitas Siliwangi
Tasikmalaya, Jawa Barat, Indonesia
ekawahyu@unsil.ac.id

[2] Teknik Informatika, Fakultas Teknik, Universitas Siliwangi
Tasikmalaya, Jawa Barat, Indonesia
alam@unsil.ac.id

Abstrak

Sistem Informasi Akademik Universitas Siliwangi (SIMAK) menjamin transparansi penggunaan sumber daya informasi berbasis internet untuk layanan dan operasional akademik *day-to-day*. Kendala klasik yang sering terjadi dalam penyelenggaraan sistem berbasis internet adalah jumlah pengguna sistem akan mempengaruhi kinerja dan kecepatan sistem dalam memberikan layanan. Peningkatan jumlah mahasiswa di Universitas Siliwangi berimbas kepada banyaknya *user system* yang meng-akses sistem akademik dan menurunnya kecepatan *response* sistem dalam menangani *request* dari pengguna. Kondisi ini dapat ditangani dengan optimasi sistem dengan Memcached dan Mirror Server. Memcached adalah suatu *script* tambahan yang diletakkan dalam suatu Server Mirror sebagai jembatan (*bridge system*) antara server utama dengan *web service*. Sehingga saat penggunaan, setiap kali komunikasi data terjadi, *request* dari pengguna akan di seleksi apabila permintaan tersebut tersedia di Cached Memory pada Mirror Server maka langsung ditampilkan ke antarmuka pengguna tanpa melalui eksekusi ke database sehingga beban sistem utama dan database berkurang. Hasil pengujian Memcached dan Mirror Server menggunakan Jmeter dengan skenario standar adalah terjadi peningkatan *throughput* sekitar 490 transaksi per menit lebih besar dibandingkan sebelum implementasi yaitu 171 transaksi per menit.

Keywords: SIMAK, Memcached, Mirror Server.

1. Pendahuluan

Pengelolaan informasi secara terintegrasi menjadi sangat penting di setiap lembaga, termasuk di Universitas Siliwangi Tasikmalaya. Tantangan Universitas Siliwangi sejak perubahan status dari Perguruan Tinggi Swasta (PTS) menjadi Perguruan Tinggi Negeri Baru (PTNB) adalah tuntutan dalam segala bidang salah satunya dalam layanan informasi yang harus diperhatikan. Pertama yaitu tuntutan kemudahan dalam pelayanan informasi misalnya kemudahan dalam mengakses informasi yang dibutuhkan oleh pengguna. Kedua yaitu tuntutan kecepatan dalam memperoleh informasi misalnya kecepatan dalam pengolahan data dari *database*. Tuntutan tersebut sangat relevan, karena secara teori pengguna Teknologi Informasi dan Komunikasi (TIK) membutuhkan informasi yang akurat, relevan, ekonomis, cepat, tepat, serta mudah mendapatkannya.

Seiring berjalannya waktu, performa Sistem Informasi Akademik (SIMAK) sebagai sistem untuk mengelola kegiatan akademik di Universitas Siliwangi berbasis web menjadi menurun diakibatkan banyaknya *user* yang menggunakan sistem. Selain itu aplikasi web yang dinamis tentunya akan memperlambat kinerja web itu sendiri. Ini terjadi karena semakin dinamis sebuah aplikasi berbasis web, maka semakin banyak juga data yang akan di *load* dari database. Adanya permintaan atau *request* dari pengguna sistem menyebabkan server sibuk, sistem menjadi lambat, dan terjadi antrian *query* pada database sehingga waktu yang diperlukan untuk mengakses SIMAK dan transfer data menjadi lama. Kondisi ini akan menghambat kinerja para pengguna sistem. Dilatar belakangi kasus diatas, diperlukan suatu langkah-langkah untuk meningkatkan kinerja sistem dan server. Solusi untuk mengatasi hambatan ini adalah melakukan optimalisasi di sisi server untuk mempercepat waktu respon dan transfer data.

Sebelumnya pernah dilakukan penelitian mengenai Optimalisasi Sistem Informasi Akademik (SIMAK) di Universitas Siliwangi [1]. Penelitian tersebut

hanya sebatas bagaimana memanfaatkan Memcached dan Mirror Server untuk mengatasi kelambatan *response* dari *request* pengguna terhadap SIMAK dalam jaringan lokal dimana *load* data dari server untuk data-data aktif atau yang sering digunakan dengan kondisi yang paling minimum yaitu tampil di mesin browser dengan tampilan yang paling sederhana dan Memcached belum diimplementasikan terhadap seluruh *script* pemrograman. Hasil penelitian didapat bahwa ada peningkatan sekitar 40% dengan penerapan Memcached dan Mirror Server.

Sehubungan hasil penelitian tersebut, maka diperlukan penelitian lanjutan untuk lebih memaksimalkan penggunaan Memcached dan Mirror Server dalam SIMAK. Dimana penerapan Memcached tidak lagi hanya pada satu aktivitas tapi diterapkan pada semua aktivitas didalam semua Modul layanan pada SIMAK. Penelitian ini sangat penting mengingat beban server SIMAK semakin meningkat. Apabila tidak ditangani sesegera mungkin dapat menyebabkan sistem menjadi *down* dan aktivitas operasional di institusi menjadi terhambat.

Dari uraian latar belakang diatas, dapat di identifikasi beberapa masalah sebagai berikut:

1. Bagaimana membangun *script* pemrograman sebagai *caching system* untuk meningkatkan kecepatan terhadap akses informasi pada Sistem Informasi Akademik (SIMAK) dengan menggunakan Memcached dan Mirror Server dan untuk mempermudah pengguna dalam mengakses informasi dilihat dari sisi waktu *response* sistem terhadap *request* dari pengguna.
2. Bagaimana cara implementasi algoritma yang tepat untuk proses Memcached dan Mirror Server pada semua aktivitas didalam satu modul layanan Sistem Informasi Akademik (SIMAK).
3. Bagaimana mengetahui keberhasilan hasil implementasi Memcached dan Mirror Server melalui pengukuran kecepatan akses dan *rensponse* untuk meningkatan kecepatan akses terhadap Sistem Informasi Akademik (SIMAK).

Mengingat luasnya cakupan permasalah yang akan dikaji maka pembahasan teknis disesuaikan untuk mencapai hasil akhir yang diharapkan yaitu sebagai berikut:

1. Untuk mendistribusikan beban trafik pada dua atau lebih jalur koneksi secara seimbang, agar trafik dapat berjalan optimal, memaksimalkan throughput, memperkecil waktu tanggap dan menghindari overload pada salah satu jalur koneksi maka akan ditambahkan Loadbalancing.
2. Untuk mempermudah proses pengendalian memcached maka pada Aplikasi SIMAK akan ditambahkan menu dan fasilitas kendali memcached.
3. Pengujian hasil implementasi Memcached dan Mirror Server yang dilakukan terhadap SIMAK menggunakan software online bersifat opensource yaitu Jmeter dengan skenario uji yang bersifat realistis

Adapun tujuan yang ingin dicapai dari penelitian ini adalah:

1. Membangun Mirror Server untuk *Caching System* sebagai jembatan penghubung menggunakan bahasa pemrograman PHP berbasis web yang akan diimplementasikan pada Mirror Server sebagai jembatan terhadap server SIMAK dan Database utama.
2. Menentukan algoritma Memcached yang tepat untuk di *embedded* kedalam *script programming* di setiap file dalam satu modul pada SIMAK.
3. Melakukan pengukuran untuk mengetahui kecepatan akses pada SIMAK sebelum dan sesudah Memcached dan Mirror Server diterapkan pada satu modul sehingga diketahui apakah terjadi peningkatan kecepatan akses.

Sedangkan manfaat dari penelitian ini diharapkan dapat membantu Unit Pelayanan Teknis (UPT) Pusat Layanan Data dan Sistem Informasi Universitas Siliwangi untuk optimalisasi Sistem Informasi Akademik (SIMAK) yang pada akhirnya dapat bermanfaat bagi civitas akademika di Universitas Siliwangi sebagai pengguna sistem.

2. Tinjauan Pustaka

Pada bagian tinjauan pustaka dilakukan kajian terhadap penelitian yang telah dilakukan sebelumnya dan juga membahas mengenai hasil kajian pendukung untuk mencapai tujuan.

2.1 Penelitian Sebelumnya

Seperti telah dijelaskan pada latar belakang, sebelumnya pernah dilakukan penelitian mengenai Optimalisasi Sistem Informasi Akademik (SIMAK). Hasil penelitian didapat bahwa kecepatan eksekusi SQL dan Data View pada browser ada peningkatan sekitar 40% dengan penerapan Memcached dan

Mirror Server. Penelitian tersebut belum mempertimbangkan view data pada kondisi sebenarnya [1].

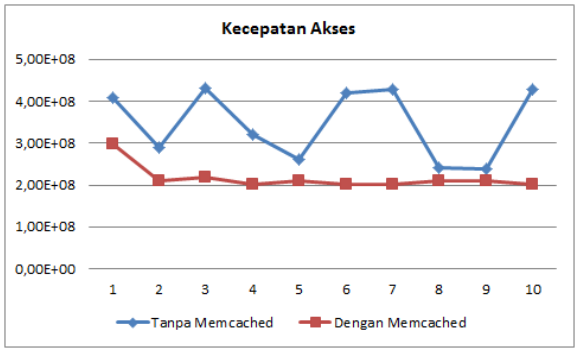
Gambar 1. Server existing.

Hasil pengujian Memcached terhadap SIMAK yang telah dilakukan dalam penelitian sebelumnya dilakukan dengan cara mengukur waktu respon melalui akses dari *client* langsung terhadap server utama tanpa Memcached dan waktu akses dari *client* melalui *mirror server* yang didalamnya terdapat aplikasi Memcached. Keduanya melakukan pengaksesan terhadap data yang sama dengan kriteria bahwa data yang akan di load memiliki *record* terbanyak, paling sering diakses, dan ditampilkan dengan kondisi paling minimum tanpa gambar dan tabel. Pengujian dilakukan dari node yang paling banyak melakukan pengaksesan data. Pengujian ini masih dilakukan pada jaringan lokal mengingat pengaksesan paling banyak dilakukan dalam jaringan lokal selama operasional institusi yang berkaitan dengan SIMAK. Selanjutnya dilakukan perbandingan dari rerata kecepatan waktu *response* dari masing-masing pengukuran. Hasil pengukuran kecepatan waktu respon yang telah dilakukan dapat dilihat pada Tabel 1:

Tabel 1. Hasil pengukuran waktu respon

Task	Time Result Non-Memcached	Time Result With-Memcached
1	4,10E+08	2,98E+08
2	2,91E+08	2,10E+08
3	4,32E+08	2,19E+08
4	3,19E+08	2,00E+08
5	2,62E+08	2,10E+08
6	4,20E+08	2,00E+08
7	4,29E+08	2,00E+08
8	2,41E+08	2,10E+08
9	2,38E+08	2,10E+08
10	4,29E+08	2,00E+08
Avg	**3,47E+08**	**2,16E+08**

Selanjutnya dari data-data tersebut diolah menggunakan Microsoft Excel untuk melihat perbandingan dari kedua cara pengukuran kecepatan waktu *respon* secara visual seperti diperlihatkan Gambar dibawah ini.

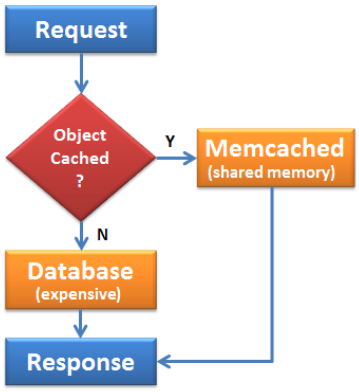


Gambar 2. Grafik hasil pengukuran.

Dari gambar diatas dapat dilihat perbandingan kecepatan akses dimana hasilnya adalah kecepatan akses menggunakan memcached memiliki kecepatan yang stabil setelah sebelumnya membutuhkan waktu yang lebih lama untuk *cache* data kedalam memori server. Pengujian dengan sistem lama tanpa memcached menghasilkan kecepatan yang tidak stabil. Rerata kecepatan akses dengan menggunakan memcached adalah 2,16 detik, sedangkan rerata kecepatan akses tanpa memcached adalah 3,47 detik.

2.2 Memory Cached

Memcached adalah sebuah teknologi yang digunakan sebagai suatu sistem penyimpanan dan pendistribusian data didalam memory server (RAM). Memcached pertama kali dikembangkan oleh Brad Fitzpatrick untuk LiveJournal pada tahun 2003, dan sekarang sudah banyak digunakan oleh website besar selain LiveJournal seperti Wikipedia, Flickr, Facebook, Twitter, Youtube, dan WordPress. Memcached digunakan untuk mempercepat akses database dari website yang bersifat dinamis dengan *caching data* dan objek dalam memory. Ini berarti mengurangi jumlah akses ke sumber data eksternal yang harus dibaca. Tujuan Memcached adalah sebagai sistem *caching* terdistribusi, Memcached digunakan sebagai sistem penyimpanan dan pendistribusian data didalam memory server. Saat ini memcached merupakan project yang open source. Alur memcached dapat dilihat pada gambar berikut:



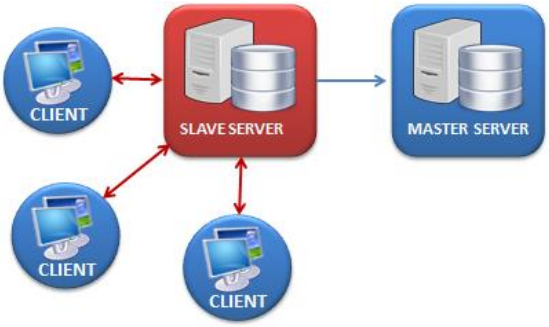
Gambar 3. Alur memcached

2.3 Mirror Server

Mirror server atau disebut juga sinkronisasi server merupakan duplikat dari suatu server yang berbeda dengan server utama. Fungsi mirror adalah untuk mengurangi kemacetan data pada suatu situs. Server utama merupakan server yang paling akurat dan mirror bertugas untuk menyalin isi server utama tersebut.

2.4 Database Replication

Tujuan dari replikasi database adalah membuat suatu backup data dari sebuah database relasional. Replikasi database digunakan untuk menyalin dan mendistribusikan data dari satu database ke database yang lain. Selanjutnya secara otomatis dilakukan sinkronisasi antar database tersebut untuk menjaga konsistensi atau melakukan hal yang sama secara terus menerus. Dengan cara ini data dapat di distribusikan ke lokasi yang berbeda.



Gambar 4. Replikasi database pada mirror server

Replikasi dilakukan pada dua server, yaitu *master server* dan *slave server*. *Master server* merupakan server MySQL untuk menangani transaksi data *request-response* dari pengguna. Sedangkan *slave server* adalah duplikat dari *master server* dengan mekanisme melakukan semua SQL *statement* yang mengubah data di *master server*. Dengan demikian *backup* dapat dilakukan secara periodik pada *slave server*.

2.5 Denormalisasi Database

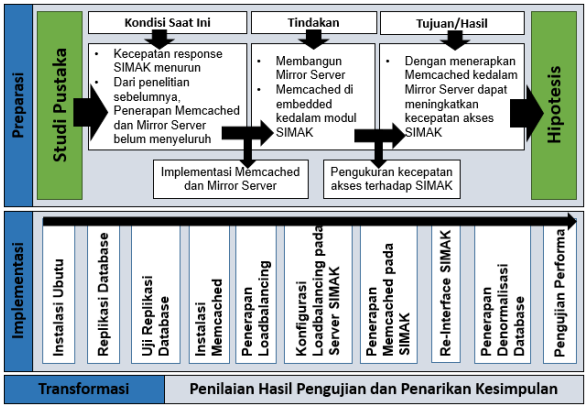
Denormalisasi database adalah pelanggaran aturan normalisasi atau menjabarkan suatu tataan database yang telah normal deng tujuan untuk meningkatkan performa pengaksesan data pada database. Maksud dari database yang telah normal adalah database yang memiliki redundansi data minimum sehingga data yang disimpan tidak mengalami kerancuan dalam proses pengaksesan. Perbedaan antara database normal dan denormal terletak pada kompleksitas *query* yang dimiliki.

2.6 Load Balancing

Load Balancing adalah teknik untuk mendistribusikan beban trafik pada dua atau lebih jalur koneksi secara seimbang, agar trafik dapat berjalan optimal, memaksimalkan throughput, memperkecil waktu tanggap dan menghindari overload pada salah satu jalur koneksi.

3. Metode Penelitian

Metode penelitian yang digunakan dalam penelitian ini menggunakan pendekatan bidang komputasi yaitu ***Computer Science***. Pendekatan ini memiliki karakteristik penelitian untuk membahas isu yang berhubungan dengan *core* teknologi dan perbaikan metode. *Content* penelitian meggunakan pendekatan ini lebih fokus ke aspek teknis (*Technical Aspect*) dengan metode riset (*Research Methods*) yang bersifat *Experiment*. Sedangkan *Reseach Objective* dengan pendekatan ini lebih dimaksudkan untuk pengembangan dan pembangunan dari objek penelitian dari metode penelitian bidang komputasi yang digunakan (*Development of Computing Methods*), sedangkan metode analisis (*Analysis Methods*) menggunakan pendekatan *Computing Theories*. Adapun tahapan yang dilakukan diperlihatkan pada gambar berikut ini:



Gambar 5. Metode penelitian

3.1 Objek Penelitian

Objek penelitian dalam penelitian ini adalah Server dan Sistem Informasi Akademik (SIMAK) Universitas Siliwangi yang berada dibawah pengelolaan UPT. Pusat Layanan Data dan Sistem Informasi bertempat di Lantai 2 Gedung Rektorat Universitas Siliwangi.

3.2 Tahapan Penelitian

Metode penelitian yang digunakan menggunakan pendekatan Eksperimental. Pendekatan ini berfokus pada aspek teknis. Ada 3 tahapan yang dilakukan untuk menyelesaikan masalah dalam penelitian ini yaitu:

1. Tahapan Preparasi, adalah tahapan persiapan. Aktifitas utama pada tahapan ini adalah Studi pustaka, Analisis, dan Hipotesis. Untuk mendapatkan data yang relevan dengan masalah penelitian yang akan dicari jawabannya guna menguji kebenaran hipotesis yang telah dirumuskan, maka dilakukan pengumpulan data secara cermat dan sistematis menggunakan pendekatan *computing theories* yang dilakukan dengan tujuan untuk pengumpulan data, mempelajari, dan mengkaji informasi-informasi yang didapat melalui aktifitas pembelajaran sesuai dengan objek penelitian yang akan ditemukan cara/ solusi penyelesaiannya. Aktifitas analisis yang dilakukan bertujuan untuk mengumpulkan artefak-artefak yang nantinya akan dijadikan usulan untuk tahapan implementasi. Kerangka pemikiran dalam analisis ini dipertimbangkan berdasarkan kondisi eksisting tempat penelitian yang akan dilakukan, berdasarkan tindakan yang akan dilakukan untuk mencapai tujuan dari penelitian, dan berdasarkan tujuan / hasil penelitian yang ingin dicapai. Selanjutnya aktivitas yang dilakukan dari setiap pertimbangan tersebut mengarah pada Hipotesis yang akan di buktikan kebenarannya.

2. Tahapan Implementasi, adalah tahapan untuk memperluas aktivitas yang saling menyesuaikan proses interaksi antara tujuan dan tindakan untuk mencapainya secara efektif. Tahapan ini bukan sekedar aktivitas saja, tetapi suatu kegiatan yang terencana dan dilakukan secara sungguh-sungguh berdasarkan acuan norma tertentu untuk mencapai tujuan. Oleh karena itu implementasi tidak berdiri sendiri tetapi dipengaruhi oleh objek berikutnya yaitu untuk membuktikan Hipotesis.

3. Tahapan Transformasi, adalah tahapan untuk berpindah menuju sistem yang dianggap lebih baik dan mendukung. Dalam penelitian ini, tahapan transformasi dimulai dari mengkaji hasil pengujian dari implementasi yang telah dilakukan, penarikan kesimpulan untuk memembuktikan Hipotesis. Dari tahapan ini selanjutnya diberikan rekomendasi apakah, hasil penelitian layak diterapkan kedalam sistem yang menjadi objek penelitian secara keseluruhan

4. Hasil dan Pembahasan

Hasil dan pembahasan secara garis besar dijelaskan dalam tiga bagian sesuai dengan metode penelitian dengan pendekatan eksperimental, yaitu Preparasi, Implementasi, dan Trasformasi.

4.1 Tahap Preparasi

Tahapan preparasi adalah tahapan persiapan. Aktifitas utama pada tahapan ini adalah Studi pustaka, Analisis, dan Hipotesis

4.1.1 Hasil Kajian Pustaka

Hasil kajian pustaka yang dilakukan dalam penelitian ini dijelaskan sebagai berikut:

1. Optimasi Server: “Memanfaatkan Memcached dan Mirror Server untuk **mengatasi kelambatan *response*** dari *request* pengguna [1]”
2. Memory Cached: “Memcached digunakan untuk **mempercepat** akses database dari website yang bersifat dinamis dengan *caching data* dan objek dalam memory [1][2][4-6]”
3. Mirror Server: “Mirror server atau disebut juga sinkronisasi server merupakan duplikat dari suatu server yang berbeda dengan server utama.

Fungsi mirror adalah untuk **mengurangi kemacetan** data pada suatu situs [1]"

4. Replikasi Database: "Replikasi database digunakan untuk menyalin dan mendistribusikan data dari satu database ke database yang lain. Selanjutnya secara otomatis dilakukan **sinkronisasi** antar database tersebut untuk **menjaga konsistensi** atau melakukan hal yang sama **secara terus menerus** [1]"
5. Load Balancing: "Untuk **mendistribusikan beban trafik** pada dua atau lebih jalur koneksi secara seimbang, agar trafik dapat berjalan **optimal**, **memaksimalkan *throughput*, memperkecil waktu tanggap** [4]"
6. MySQL: "MySQL dapat melakukan transaksi dengan **mudah dan efisien**, mampu menangani jutaan pengguna dalam waktu yang bersamaan dan mencukupi untuk kebutuhan database perusahaan-perusahaan skala menengah hingga kecil [2]"
7. Apache: "adalah server web yang **dapat dijalankan di banyak sistem operasi** (Unix, BSD, Linux, Microsoft Windows dan Novell Netware serta platform lainnya) yang berguna untuk melayani dan memfungsikan situs web [2][3]"
8. Programming Javascript, HTML, PHP: "**HTML** adalah sebuah bahasa *markup* yang digunakan untuk membuat sebuah halaman web dan menampilkan berbagai informasi di dalam sebuah *browser* Internet. **Javascript** adalah bahasa yang berbentuk kumpulan skrip yang pada fungsinya berjalan pada suatu dokumen HTML. **PHP** dapat disisipkan pada dokumen HTML dan **bekerja pada lingkungan *server*** [3][7]"
9. Linux Ubuntu "Ubuntu berfokus pada ketersediaan kegunaan pada disfungsi, keamanan dan **stabilitas** [1]"
10. Apache Jmeter "**Untuk mengetahui performa dari suatu web server maka perlu dilakukan pengujian terhadap *web server*** tersebut agar dapat diketahui seberapa tinggi tingkat performansinya. **Jmeter** adalah sebuah tool/alat yang digunakan untuk melakukan *performance test* [2][3]"

4.1.2 Analisis

Kerangka pemikiran dalam analisis ini dipertimbangkan berdasarkan kondisi eksisting tempat penelitian dilakukan, berdasarkan tindakan yang akan dilakukan untuk mencapai tujuan dari penelitian, dan berdasarkan tujuan / hasil penelitian yang ingin dicapai. Hasil analisis setelah melalui studi pustaka dan observasi pada objek penelitian dijelaskan pada tabel berikut:

Tabel 2. Hasil analisis

Kondisi Saat Ini	Tindakan	Tujuan/ Hasil
• Kecepatan response SIMAK menurun. • Dari penelitian sebelumnya, penerapan memcached dan mirror merver pada SIMAK belum menyeluruh	• Membangun mirror server • Memcached di embedded kedalam modul SIMAK	• Dengan menerapkan memcached kedalam mirror server dapat meningkatkan kecepatam akses SIMAK
Gap: untuk meningkatkan kecepatan akses SIMAK maka perlu dilakukan implementasi memcached dan mirror server secara menyeluruh.		**Gap:** untuk membuktikan keberhasilan implementasi memcached dan mirror server pada SIMAK perlu dilakukan pengukuran performa untuk melihat kecepatan akses SIMAK

4.1.3 Hipotesis

Berdasarkan hasil analisa variabel-variabel pada model penelitian ini, maka dapat dibuat hipotesis yang akan di uji dalam penelitian ini yaitu sebagai berikut:

Hipotesis 1:

H0: Mirror Server TIDAK DAPAT dibangun untuk Caching System sebagai jembatan antara SIMAK dan Database

H1: Mirror Server DAPAT dibangun untuk Caching System sebagai jembatan antara SIMAK dan Database

Hipotesis 2:

H0: Memcached TIDAK BISA di *embedded* kedalam *script programming* di setiap file dalam satu modul pada SIMAK

H1: Memcached BISA di *embedded* kedalam *script programming* di setiap file dalam satu modul pada SIMAK

Hipotesis 3:

H0: TIDAK TERBUKTI bahwa penerapan Memcached dan Mirror Server dapat peningkatan kecepatan akses pada SIMAK

H1: TERBUKTI bahwa penerapan Memcached dan Mirror Server dapat peningkatan kecepatan akses pada SIMAK

4.2 Tahap Implementasi

Tahapan ini adalah tahapan untuk memperluas aktivitas yang saling menyesuaikan proses interaksi antara tujuan dan tindakan untuk mencapainya secara efektif yang terencana dan dipengaruhi oleh objek berikutnya yaitu untuk membuktikan Hipotesis.

4.2.1 Instalasi Ubuntu

Tahap pertama dari implementasi Memcached dan Mirror Server adalah menyiapkan server Mirror terlebh dahulu. Server yang akan dibangun menggunakan Ubuntu Server. Ubuntu Server adalah

Ubuntu yang merupakan distro Linux yang didesain untuk di install di server dan untuk membuat server dengan cara mudah. Ubuntu Server dispesialisasikan untuk kebutuhan penggunaan dalam hal *server*. Perbedaan mendasar, di Ubuntu Server tidak tersedia GUI. Jika menggunakan ubuntu server artinya harus bekerja dengan perintah perintah di layar *console* seperti tampilan pada DOS dari Windows OS.

4.2.2 Replikasi Database

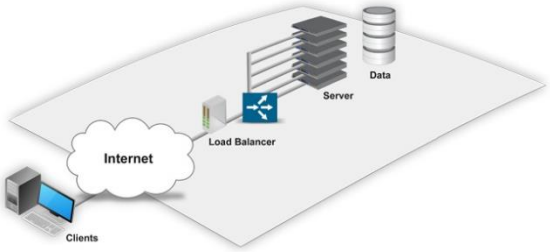
Replikasi database dilakukan untuk mereplikasi Database MySQL Master to Master. Syarat konfigurasi adalah kedua server bisa saling terhubung dan port 3306 di kedua server tersebut dibuka, MySQL pada kedua server direkomendasikan mempunyai versi yang sama, seting Server A menggunakan IP 11.11.1.14 dan server B menggunakan IP 11.11.1.15 serta menggunakan database untuk SIMAK. Secara teknis pada tahapan ini dilakukan proses menjadikan server B sebagai server slave, menjadikan server A sebagai server slave, dan melakukan pengujian replikasi database master to master berfungsi dengan baik atau tidak. Indikator keberhasilan tahapan ini adalah apabila ada perubahan atau penambahan data di server A maka pada server B akan terjadi hal yang sama dan sebaliknya. Selanjutnya dilakukan seting nilai AUTO_INCREMENT antara kedua server untuk menghindarinya tabrakan sehingga nilai AUTO_INCREMENT pada server A akan bernilai 1,3,5,7,9,dst sedangkan pada server B AUTO_INCREMENT akan bernilai 2,4,6,8,10,dst.

4.2.3 Instalasi Memcached

Tahapan ini adalah melakukan instalasi Memcached pada server yang telah siap digunakan. Secara teknis aktifitas yang dilakukan adalah melakukan instalasi memcached pada Ubuntu Server yang telah disiapkan sebelumnya.

4.2.4 Penerapan Load Balancing

Tahapan selanjutnya adalah menerapkan Load Balancing pada server yang telah disiapkan. Penerapan Load Balancing ini menggunakan Load Balancer dengan arsitektur seperti diperlihatkan pada gambar sebagai berikut:



Gambar 6. Arsitektur load balancer server

Aplikasi yang digunakan untuk load balancing pada web server menggunakan modul balancer dari apache yaitu apache balancer yang telah otomatis terinstal, diantaranya adalah aplikasi apache balancer. Untuk menggunakan modul balancer ini, tahapan terpenting adalah memastikan bahwa 3 modul balancer yang berada di file httpd.conf tidak di disable. Kemudian memastikan bahwa fitur proxy yang berada di file httpd.conf didisable. Tahap terakhir adalah melakukan konfigurasi inti dari apache load balancer yang diletakkan di bagian akhir dari file httpd.conf.

4.2.5 Konfigurasi Load Balancing pada Server SIMAK

Tahapan selanjutnya adalah melakukan konfigurasi Load Balancig pada Server SIMAK yang telah disiapkan sebelumnya. Adapaun konfigurasi tersebut menggunakan IP 11.11.1.14, IP 11.11.1.15, dan IP 11.11.1.16, dilanjutkan dengan konfigurasi pada module /etc/apache2/httpd.conf, konfigurasi proxy /etc/apache2/mod-enabled/proxy.conf, dan terakhir melakukan konfigurasi pada Proxy Balancer /etc/apache2/conf.d/proxy-balancer.

4.2.6 Penerapan Memcached pada SIMAK

Tahap selanjutnya adalah menerapkan Memcached pada server SIMAK. Tahapan pertama adalah membuat koneksi Memcached dan menerapkan Memcached pada beberapa modul pada Modul Keuangan di SIMAK seperti Cache untuk data tahun akademik, cache data matakuliah praktikum dan transitoris, dan cache data daftar peserta P2SPT.

4.2.7 Re-Interface SIMAK

Tahap selanjutnya adalah melakukan Re-Interface pada SIMAK atau membuat antarmuka pada SIMAK. Tahapan ini memberikan intervensi terhadap antarmuka SIMAK yang telah mapan untuk disesuaikan dengan kebutuhan guna implementasi Memcached pada modul SIMAK. Ada dua tahap yang dilakukan, pertama menerapkan Memcached pada menu-menu yang ada pada

SIMAK dengan tujuan agar Memcached yang di terapkan dapat dengan mudah diatur melalui antarmuka pada modul Menu SIMAK, kedua membuat fasilitas menu tambahan pada SIMAK.

4.2.8 Penerapan Denormalisasi Database

Tahap selanjutnya adalah menerapkan denormalisasi pada database. Sebelum dilakukan denormalisasi data, terdapat kode SQL di dalam manajemen keuangan yang mengakses empat tabel yaitu tabel bipotmhsw, mhsw, bayartrans, dan bayarmhsw dengan kode SQL-nya adalah sebagai berikut:

```
$s = "select b.MhswID, m.Nama as NamaMhsw,
m.Kelas, m.ProgramID, b.BIPOTMhswID,
b.DiajukanFakultas as DF,
b.DiajukanKeuangan as DK,
b.DicairkanYayasan as DIY,
b.DicairkanKeuangan as DIK from bipotmhsw b
left outer join mhsw m on m.MhswID =
b.MhswID left outer join bayartrans bt on
bt.BIPOTMhswID = b.BIPOTMhswID left outer
join bayarmhsw bm on bm.BayarMhswID =
bt.BayarMhswID where m.ProdiID='$ProdiID'
AND b.TahunID='$TahunID' $jenis $whr_prg
and (select sum(b.Jumlah*b.Besar)-
b.Dibayar) = 0 and b.NA = 'N' order by
bm.Tanggal";
```

Kemudian dilakukan denormalisasi data dengan cara menggabungkan field-field yang diperlukan seperti pada kode SQL di atas, ke dalam sebuah tabel sehingga struktur tabel tersebut menjadi seperti pada gambar berikut:

Column Name	Datatype
BIPOTMhswID	BIGINT(20)
MhswID	VARCHAR(10)
Nama	VARCHAR(100)
Kelas	VARCHAR(2)
ProdiID	VARCHAR(4)
ProgramID	VARCHAR(20)
TambahanNama	VARCHAR(100)
DiajukanFakultas	DATETIME
DiajukanKeuangan	DATETIME
DicairkanYayasan	DATETIME
DicairkanKeuangan	DATETIME
BayarMhswID	VARCHAR(30)
TahunID	VARCHAR(5)
Jumlah	INT(5)
Besar	BIGINT(10)
Dibayar	BIGINT(10)
Tanggal	DATE

Gambar 7. Struktur tabel keuangan

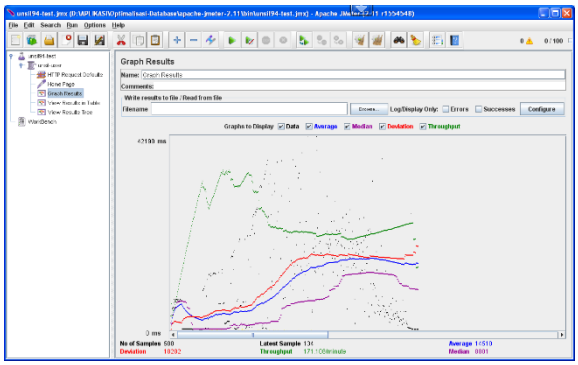
Setelah dilakukan denormalisasi data maka kode SQL dapat disederhanakan sebagai berikut:

```
$s = "select b.*,Nama as
NamaMhsw,b.DiajukanFakultas as DF,
b.DiajukanKeuangan as DK,
b.DicairkanYayasan as DIY,
b.DicairkanKeuangan as DIK,
DATE_FORMAT(Tanggal,'%d-%m-%Y') as Tanggals
from keuangan b where b.ProdiID='$ProdiID'
AND b.TahunID='$TahunID' $jenis $whr_prgb";
```

Dengan denormalisasi ini maka kecepatan akses data bertambah dikarenakan kode SQL hanya mengambil data dari satu tabel saja tanpa harus mengambil dari empat tabel seperti sebelumnya.

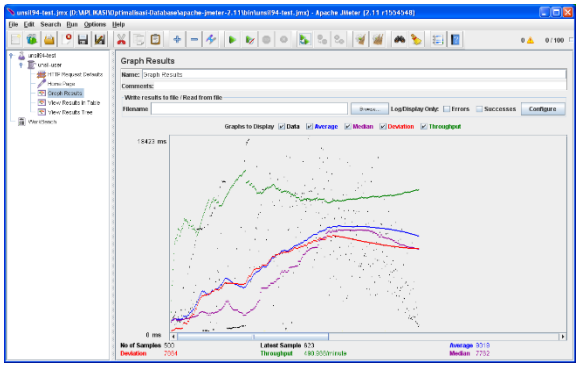
4.2.9 Pengujian Performa

Untuk melihat apakah penerapan memcached dan mirror server berhasil meningkatkan kecepatan akses SIMAK maka tahap selanjutnya adalah melakukan pengujian performa menggunakan Jmeter. Berikut ini adalah pengujian performa menggunakan Apache Jmeter pada file us-unsil/baa/transkrip.php. Pengujian menggunakan skenario yaitu jumlah user yang mengakses adalah 100 orang dengan periode 1 detik dan diulang sebanyak 5 kali. Waktu yang diperlukan untuk mengakses file us-unsil/baa/transkrip.php sebelum dilakukan optimalisasi adalah sebagai berikut:



Gambar 8. Sebelum optimalisasi

Sedangkan waktu yang diperlukan untuk mengakses file us-unsil/baa/transkrip.php setelah dilakukan optimalisasi adalah sebagai berikut:



Gambar 9. Setelah optimalisasi

4.3 Tahap Transformasi

Tahapan ini adalah tahapan untuk berpindah menuju sistem yang dianggap lebih baik dan mendukung. Dimulai dari mengkaji hasil pengujian dari implementasi yang telah dilakukan, penarikan kesimpulan untuk membuktikan Hipotesis, dan rekomendasi apakah hasil penelitian layak diterapkan kedalam sistem yang menjadi objek penelitian secara keseluruhan.

4.3.1 Penilaian Hasil Pengujian

Dari hasil pengujian memcached terhadap file us-unsil/baa/transkrip.php menggunakan Jmeter sesuai skenario pengujian didapat data sebagai berikut:

Tabel 2. Hasil pengujian

No	Kasus Uji	Sebelum	Sesudah
1.	Data	500	500
2.	Average	14150	9019
3.	Median	8801	7762
4.	Deviation	18282	7864
5.	Throughput	171.106/minute	490.966/minute

Dari hasil pengujian tersebut didapat data 1 user yang melakukan 500 kali *request* di setiap *thread group*. Hasil percobaan ini menunjukkan bahwa server memiliki kecepatan yang lebih stabil, hal ini bisa dilihat dari nilai *standard Deviation*, *Average*, *Median* yang semakin kecil nilainya menunjukkan bahwa variasi kecepatan semakin sedikit (sampel yang ada semakin mendekati nilai rata-rata/average, tidak banyak yang melenceng jauh). Selain itu, transaksi per detik (TPS) atau *throughput* yang lebih besar setelah penerapan memcached dibandingkan dengan sebelum penerapan memcached. Hasil ini berarti bahwa penerapan Memcached pada Mirror Server dapat meningkatkan kecepatan akses dan waktu *response* dari SIMAK.

4.3.2 Penarikan Kesimpulan

Dari tahapan implementasi yang telah dilakukan sebelumnya mulai dari tahapan persiapan implementasi hingga pengukuran maka dapat dibuktikan dari Hipotesis yang diajukan bahwa:

1. Hipotesis 1 : H1 TERBUKTI.
2. Hipotesis 2 : H1 TERBUKTI.
3. Hipotesis 3 : H1 TERBUKTI.

5. Kesimpulan

Sebagaimana tujuan dari penelitian ini, dari hasil kajian dan implementasi hingga pengujian dapat ditarik kesimpulan sebagai berikut:

1. Berhasil membuktikan Hipotesis 1 dimana H1 TERBUKTI yaitu Mirror Server DAPAT dibangun untuk Caching System sebagai jembatan antara SIMAK dan Database. Ini dibuktikan dengan berhasil membangun Mirror Server untuk *Caching System* sebagai jembatan penghubung menggunakan bahasa pemrograman PHP berbasis web yang akan diimplementasikan pada Mirror Server sebagai jembatan terhadap server SIMAK dan Database utama.
2. Berhasil membuktikan Hipotesis 2 dimana H1 TERBUKTI yaitu Memcached BISA di *embedded* kedalam *script programming* di setiap file dalam satu modul pada SIMAK. Ini dibuktikan bahwa algoritma Memcached yang tepat dapat ditentukan berdasarkan penelitian eksperimental untuk di *embedded* kedalam *script programming* di dalam satu modul pada SIMAK.
3. Berhasil membuktikan Hipotesis 3 dimana H1 TERBUKTI bahwa penerapan Memcached dan Mirror Server dapat meningkatan kecepatan akses pada SIMAK dimana terjadi peningkatan *throughput* sekitar 490 transaksi per menit lebih besar dibandingkan sebelum implementasi yaitu 171 transaksi per menit. Hal ini dibuktikan melalui pengukuran untuk mengetahui kecepatan akses dan performansi pada SIMAK sebelum dan sesudah Memcached dan Mirror Server diterapkan pada modul sehingga diketahui apakah terjadi peningkatan kecepatan akses menggunakan *tools stress tester* menggunakan Jmeter dengan hasil bahwa terjadi peningkatan kecepatan akses dari sistem sesuai skenario uji yang dilakukan.

Dari kesimpulan diatas, mengingat Modul yang ada pada SIMAK sangat banyak dan kompleks maka saran yang dapat diberikan untuk penelitian ini adalah:

1. Server yang dibangun dalam penelitian ini menggunakan perangkat dengan spesifikasi yang minimum, sebaiknya penerapan *mirror server* dan *load balancing* dilakukan pada *server* dengan spesifikasi yang sama dengan *server* yang digunakan untuk akses SIMAK sebagai *master server* sehingga *mirror server* atau *slave server* memiliki spesifikasi yang dentik dengan *master server*.
2. Pengujian yang dilakukan baru sebatas melihat performasi dari SIMAK, maka perlu dilakukan pengujian lainnya dengan menggunakan perangkat pegujian performansi selain Jmeter. Termasuk juga perlu dilakukan pengujian dari sisi keamanan atau *Security* dimana penerapan

memcached tidak menimbulkan dampak dari sisi keamanan data.

3. Dihasilkan rekomendasi yang ditujukan kepada PUSDASI sebagai pengelola SIMAK di Universitas Siliwangi untuk menerapkan memcached secara menyeluruh. Saat ini belum semua modul disisipkan *memcached*, baru dilakukan pada modul SIMAK yang dianggap banyak di akses untuk pelayanan data. Perlu dilakukan penerapan *memcached* untuk semua modul utama pada SIMAK agar kecepatan akses dapat merata di seluruh aplikasi SIMAK.
4. Server yang digunakan dalam penelitian ini masih menggunakan perangkat yang dibangun dari CPU biasa. Usulan yang dapat diberikan kepada PUSDASI adalah membangun Server baru sesuai dengan karakteristik server yang digunakan saat ini untuk dijadikan *mirror server* atau *server slave* bagi server utama SIMAK.

Referensi

[1] E. W. Hidayat, dan A. Rahmatulloh, " Optimalisasi Kinerja Sistem Informasi Akademik Universitas Siliwangi Menggunakan Memcached Dan Mirror Server", dalam Konferensi Nasional Sistem Informasi, 2014, KNSI2014-46, pp. 240-244.
[2] P. Galbraith, Developing Web Applications with Apache, MySQL, Memcached, and Perl, Wiley Publishing Inc, 2009.
[3] J.C. Meloni, Sams Teach Yourself PHP, MySQL and Apache All in One, Pearson Education Inc. 2012.
[4] MySQL AB, Designing and Implementing Scalable Applications with memcached and MySQL, MySQL White Papper, 2008.
[5] http://memcached.org
[6] http://www.danga.com/memcached
[7] http://php.net/docs.php

Eka Wahyu Hidayat, Menyelesaikan pendidikan S2 di Magister Informatika Institut Teknologi Bandung. Makalah ini sebagai diseminasi hasil Hibah Penelitian Dosen Pemula tahun 2014 yang diselenggarakan oleh LPPM Universitas Siliwangi. Peneliti adalah Dosen di Jurusan Teknik Informatika dan sebagai Kepala Bagian Perencanaan dan Pengembangan di UPT. Pusat Layanan Data dan Sistem Informasi (PUSDASI) Universitas Siliwangi.

Alam Rahmatulloh, Sedang melanjutkan studi di Magister Informatika Institut Teknologi Bandung. Peneliti adalah Dosen di Jurusan Teknik Informatika dan sebagai Programmer di UPT. Pusat Layanan Data dan Sistem Informasi (PUSDASI) Universitas Siliwangi.