TUGAS AKHIR – K141502

RANCANG BANGUN SISTEM KONTROL MESIN KEHADIRAN TERDISTRIBUSI BERBASIS WEB DENGAN MENGGUNAKAN WEBSOCKET

Haidar Arya Prasetya
NRP 05111440000051

Dosen Pembimbing
Henning Titi Ciptaningtyas, S.Kom., M.Kom.
Dr.tech. Ir. R.V.Hari Ginardi, M.Sc.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

TUGAS AKHIR – K141502

RANCANG BANGUN SISTEM KONTROL MESIN KEHADIRAN TERDISTRIBUSI BERBASIS WEB DENGAN MENGGUNAKAN WEBSOCKET

Haidar Arya Prasetya
NRP 05111440000051

Dosen Pembimbing
Henning Titi Ciptaningtyas, S.Kom., M.Kom.
Dr.tech. Ir. R.V.Hari Ginardi, M.Sc.

DEPARTEMEN INFORMATIKA
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

[Halaman ini sengaja dikosongkan]

FINAL PROJECT – K141502

DESIGN OF WEB-BASED DISTRIBUTED ATTENDANCE MACHINE CONTROL SYSTEM USING WEBSOCKET

Haidar Arya Prasetya
NRP 05111440000051

Advisors
Henning Titi Ciptaningtyas, S.Kom., M.Kom.
Dr.tech. Ir. R.V.Hari Ginardi, M.Sc.

DEPARTMENT OF INFORMATICS
Faculty of Information Technology and Communication
Sepuluh Nopember Institute of Technology
Surabaya 2018

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

RANCANG BANGUN SISTEM KONTROL MESIN KEHADIRAN TERDISTRIBUSI BERBASIS WEB DENGAN MENGGUNAKAN WEBSOCKET

TUGAS AKHIR

Diajukan Guna Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Arsitektur Jaringan Komputer
Program Studi S-1 Departemen Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh:
HAIDAR ARYA PRASETYA
NRP: 05111440000051

Disetujui oleh Dosen Pembimbing Tugas Akhir:

Henning Titi Ciptaningtyas, S.Kom., M.Kom. .....
NIP: 19840708 201012 2 004 (pembimbing 1)

Dr.tech. Ir. R.V.Hari Ginardi, M.Sc. .....
NIP: 19650518 199203 1 003 (pembimbing 2)

SURABAYA
JUNI 2018

[Halaman ini sengaja dikosongkan]

RANCANG BANGUN SISTEM KONTROL MESIN KEHADIRAN TERDISTRIBUSI BERBASIS WEB DENGAN MENGGUNAKAN WEBSOCKET

Nama Mahasiswa	:	Haidar Arya Prasetya
NRP	:	05111440000051
Departemen	:	Informatika FTIK-ITS
Dosen Pembimbing 1	:	Henning Titi Ciptaningtyas, S.Kom., M.Kom.
Dosen Pembimbing 2	:	Dr.tech. Ir. R.V.Hari Ginardi, M.Sc

ABSTRAKSI

Penerapan sistem kehadiran berbasis digital di lingkungan belajar MMT-ITS Surabaya masih dirasa kurang efisien karena adanya permasalahan seperti pengajar yang tidak bisa secara langsung mengganti subjek atau tempat pertemuan ketika mesin kehadiran didalam ruangan mati, sehingga pengajar harus kembali menggunakan cara manual untuk merekam data kehadiran, atau ketika data yang ada didalam penyimpanan lokal mesin kehadiran terlalu penuh sehingga pegawai harus melakukan pembersihan *database* secara manual.

Penelitian ini bertujuan untuk membangun sistem kontrol mesin kehadiran terdistribusi berbasis web dengan menggunakan protokol WebSocket yang memungkinkan terjadinya komunikasi dua arah secara realtime dan *full-duplex* antara browser dan server [1]. Hasil dari penelitian ini adalah dapat melakukan operasi terhadap mesin kehadiran melalui sistem kontrol, lebih kecilnya waktu yang digunakan dalam proses pengiriman perintah kepada mesin kehadiran dari sistem kontrol, serta lebih mudahnya pengoperasian mesin kehadiran apabila dibandingkan dengan penanganan terhadap mesin kehadiran dengan cara manual.

Kata kunci : WebSocket, Sistem Kontrol Terdistribusi, Sistem Kehadiran.

[Halaman ini sengaja dikosongkan]

DESIGN OF WEB-BASED DISTRIBUTED ATTENDANCE MACHINE CONTROL SYSTEM USING WEBSOCKET

Student Name : Haidar Arya Prasetya
Student ID : 05111440000051
Department : Informatics FTIK-ITS
Advisor 1 : Henning Titi Ciptaningtyas, S.Kom., M.Kom.
Advisor 2 : Dr.tech. Ir. R.V.Hari Ginardi, M.Sc

ABSTRACT

The application of a digital-based attendance system in MMT-ITS Surabaya learning environment is still considered inefficient due to the occurrence of problems, such as teachers who are unable to directly change the subject or meeting place when the attendance machine in the room is inactive, which led the teachers to perform attendance data recording manually, or when the local storage of the attendance machine is full so employees must perform manual cleaning of the database.

The purpose of this research is to build a web-based distributed attendance machine control system using WebSocket protocol, which allows a realtime and full-duplex communication between browsers and servers [1]. *The results of this research are the ability to perform operations to attendance machine through the control system, less time spent in command delivery process to the attendance machines from the control system, as well as easier operation of the attendance machine compared to manual handling of attendance machines.*

Keywords : WebSocket, *Distributed Control System, Attendance System.*

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

Puji syukur kepada Allah SWT atas rahmat dan hidayah-Nya penulis dapat menyelesaikan Penelitian yang berjudul:

RANCANG BANGUN SISTEM KONTROL MESIN KEHADIRAN TERDISTRIBUSI BERBASIS WEB DENGAN MENGGUNAKAN WEBSOCKET

Melalui lembar ini, penulis ingin menyampaikan ucapan terimakasih dan penghormatan yang sebesar-besarnya kepada:

1. Allah SWT dan junjungan Nabi Muhammad SAW, karena atas limpahan rahmat dan karunia-Nya penulis dapat menyelesaikan Penelitian dan juga perkuliahan di Informatika ITS.
2. Ir. Sri Samini dan Ir. Mohammad Muhadi, selaku kedua orang tua penulis, Ibu dan Ayah juara satu sedunia yang tidak pernah lelah dan sangat sabar dalam memberikan dukungan dan meyakinkan penulis selama menjalani masa perkuliahan hingga selesainya proses pengerjaan Penelitian ini.
3. Ghea Aquatica Puteri, S.K.H., selaku orang yang selalu ada untuk penulis, yang tidak pernah bosan menghibur dan menjadi pendengar yang baik selama menjalani masa perkuliahan hingga selesainya Penelitian penulis.
4. Ibu Henning Titi Ciptaningtyas, S.Kom., M.Kom., selaku dosen pembimbing 1 yang telah banyak membantu, membimbing dan senantiasa memberikan motivasi terbaiknya kepada penulis dalam menyelesaikan Penelitian ini.
5. Bapak Dr.tech. Ir. R.V.Hari Ginardi, M.Sc, selaku dosen pembimbing 2 yang telah membimbing dan memberikan

arahan, serta motivasi kepada penulis selama proses penyelesaian Penelitian ini.

6. Muhammad Hilman dan Afif Ridho Kamal Putra, selaku sahabat terdekat penulis yang tidak pernah bosan menemani, membantu, membimbing dan memotivasi penulis untuk menyelesaikan Penelitian ini.
7. Trastian, Faiq, Hamka, Aldi, Dimas, Naufan, Rian Danis, Bayu, Hari, Anandi, Kevin, Akhyar, Hakim, Sani, Aldo, serta kawan-kawan WxJ, TC 14 dan THEREDSTC yang tidak bisa penulis sebutkan satu per satu yang senantiasa ada dan membantu penulis selama masa perkuliahan tanpa kenal lelah dan rasa pamrih.
8. Kawan-kawan administrator Laboratorium Manajemen Informasi yang selalu menemani penulis sejak tahun kedua perkuliahan hingga selesainya Penelitian ini.
9. Kawan-kawan departemen Kaderisasi dan Pemetaan HMTC Optimasi dan Inspirasi, serta rekan-rekan Pengurus Harian HMTC Inspirasi yang telah memberikan banyak pelajaran berharga bagi penulis selama masa perkuliahan.
10. Bapak dan Ibu dosen beserta staff dan karyawan Departemen Informatika ITS lainnya yang telah banyak menyampaikan ilmu dan bimbingan yang tak ternilai harganya bagi penulis.
11. Serta pihak-pihak lain yang namanya tidak dapat penulis sebutkan satu per satu.

Penulis menyadari masih terdapat banyak kekurangan baik dalam pelaksanaan Penelitian maupun penyusunan buku laporan ini, namun penulis berharap buku laporan ini dapat menambah wawasan pembaca dan dapat menjadi sumber referensi. Penulis mengharapkan kritik dan saran yang membangun untuk kesempurnaan penulisan buku Penelitian ini.

Surabaya, Juni 2018

Haidar Arya Prasetya

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

DAFTAR KODE SUMBER

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

Pada bab ini akan dipaparkan mengenai garis besar Penelitian yang meliputi latar belakang, tujuan, rumusan dan batasan permasalahan, metodologi pembuatan Penelitian, dan sistematika penulisan.

1.1. Latar Belakang

Dalam aktivitas belajar mengajar, kehadiran pelajar dalam mengikuti materi pembelajaran merupakan salah satu faktor paling penting, karena selain merupakan pencerminan dari proses belajar demi terserapnya materi dengan baik, kehadiran juga seringkali menjadi salah satu faktor penentu lulus atau tidaknya pelajar atas subjek yang diambil. Di Institut Teknologi Sepuluh Nopember (ITS) Surabaya, kehadiran menentukan seorang mahasiswa berhak atau tidak untuk mengikuti Evaluasi Akhir Semester (EAS) dengan persentase 80% dari 16 pertemuan pada satu semester [2]. Akan tetapi, masih sering kita menjumpai penggunaan kertas untuk mencatat jumlah kehadiran pelajar dalam kegiatan belajar. Sistem pencatatan kehadiran seperti ini dinilai kurang praktis karena kerap kali memunculkan kendala-kendala seperti data yang tidak valid ketika data yang masuk salah, hilang atau rusaknya data yang ada yang akan merugikan pelajar satu kelas dan juga dosen pengajar, kurangnya efisiensi dan efektivitas pada pengolahan data [3].

Dengan semakin berkembangnya teknologi, saat ini telah banyak dibangun sistem kehadiran berbasis digital untuk mengatasi permasalahan yang ditimbulkan oleh sistem kehadiran manual, seperti yang diterapkan pada lingkungan belajar MMT-ITS Surabaya, yang menerapkan penggunaan kartu yang dilengkapi dengan *Near Field Communication* (NFC) beserta *mini-computer* Raspberry Pi. Akan tetapi, penggunaan sistem kehadiran ini masih dirasa kurang efisien karena munculnya beberapa permasalahan baru, seperti pengajar yang tidak bisa secara langsung mengganti subjek atau tempat pertemuan ketika mesin

kehadiran di ruangan mati, sehingga pengajar harus kembali menggunakan cara manual untuk merekam data kehadiran, atau ketika data yang ada didalam penyimpanan lokal mesin kehadiran terlalu penuh sehingga pegawai harus melakukan pembersihan secara manual yang jelas akan membutuhkan waktu lebih lama.

Oleh karena itu, untuk mengatasi permasalahan tersebut diperlukan adanya pengembangan sistem kehadiran berbasis digital yang dapat dikontrol melalui sebuah sistem berbasis web yang terkoneksi dengan server pusat sehingga pengelola sistem kehadiran dapat melakukan kontrol terhadap mesin pencatat kehadiran secara terpusat dan lebih praktis. Sistem kontrol yang dibangun pada Penelitian ini dikembangkan dengan menggunakan protokol WebSocket, karena memungkinkan terjadinya komunikasi *full-duplex* asinkron antara aplikasi berbasis web dan web-server [1]. Solusi yang ditawarkan ini berhubungan dengan pengerjaan Penelitian "Replikasi Sebagian dengan Metode *Horizontal Fragmentation* pada Basis Data Sistem Kehadiran" oleh Muhammad Hilman.

1.2. Tujuan

Tujuan dari pembuatan penelitian ini adalah:

1. Mempermudah fungsi kontrol terhadap mesin pencatat kehadiran yang terdapat pada masing-masing kelas.
2. Mempercepat penanganan terhadap mesin kehadiran yang bermasalah.

1.3. Rumusan Permasalahan

Rumusan masalah yang diangkat dalam penelitian ini antara lain:

1. Bagaimana sistem kontrol mesin kehadiran terdistribusi berbasis web ini dapat mempermudah fungsi kontrol terhadap mesin kehadiran?
2. Bagaimana sistem kontrol mesin kehadiran terdistribusi berbasis web ini dapat mempercepat penanganan masalah terhadap mesin kehadiran?

1.4. Batasan Permasalahan

Permasalahan yang dibahas dalam penelitian ini memiliki beberapa batasan, antara lain:

1. Teknologi utama yang digunakan adalah Raspberry Pi 3 Model B.
2. Jumlah mesin kehadiran yang digunakan menyesuaikan dengan kemampuan server pusat.
3. Kartu presensi yang digunakan merupakan kartu elektronik yang dilengkapi dengan NFC.

1.5. Metodologi

Langkah-langkah yang ditempuh dalam pengerjaan Penelitian ini yaitu:

a. Penyusunan proposal penelitian

Tahapan awal dari pengerjaan Penelitian ini adalah penyusunan Proposal Penelitian yang berisi pendahuluan yang terdiri atas latar belakang pengajuan Penelitian, rumusan masalah yang diangkat dan manfaat dari hasil pembuatan Penelitian ini. Selanjutnya ada deskripsi dan gagasan metode-metode yang diaplikasikan pada Penelitian ini, serta tinjauan pustaka yang digunakan sebagai referensi pembuatan Penelitian.

b. Studi literatur

Pada studi literatur ini, akan dipelajari sejumlah referensi yang diperlukan dalam pembuatan program yaitu mengenai Node.js, WebSocket, Laravel, Socket.io dan beberapa literatur mengenai pembuatan sistem kontrol.

c. Perancangan perangkat lunak

Tahap ini meliputi perancangan sistem kontrol berdasarkan studi literatur yang ada. Tahap ini mendefinisikan alur dari implementasi. Pada tahapan ini akan dibuat *prototype* sistem kontrol, yang merupakan rancangan dasar dari sistem yang dibuat.

d. Implementasi perangkat lunak

Tahapan implementasi perangkat lunak adalah tahap dimana metode-metode pengerjaan Penelitian yang diusulkan pada saat pembuatan proposal Penelitian diaplikasikan dalam proses pengerjaan. Untuk menerapkan metode yang telah direncanakan, maka perlu adanya perangkat lunak dan perangkat keras. Perangkat lunak berada pada mesin kehadiran dan server, sedangkan perangkat keras yang digunakan terdapat pada sisi klien, sedangkan implementasi WebSocket diterapkan pada proses komunikasi antar klien (mesin kehadiran dan halaman operasi sistem kehadiran) dan antara klien (halaman operasi mesin kehadiran) dan server pusat.

e. Pengujian dan evaluasi

Pada tahapan ini dilakukan uji coba terhadap sistem kontrol yang telah dibuat. Pengujian yang dimaksud adalah pengujian fungsionalitas sistem kontrol yang dibangun. Uji coba yang dilakukan meliputi pengoperasian mesin kehadiran melalui sistem kontrol yang telah dibuat dan memantau status klien (mesin kehadiran) dan server pusat melalui halaman sistem kontrol. Tahap ini dimaksudkan juga untuk mengevaluasi jalannya aplikasi, mencari masalah yang mungkin timbul dan mengadakan perbaikan jika terdapat kesalahan.

f. Penyusunan Buku Penelitian

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

1. Pendahuluan
    a. Latar Belakang
    b. Rumusan Masalah

c. Batasan Penelitian
d. Tujuan
e. Metodologi
f. Sistematika Penulisan

2. Tinjauan Pustaka
3. Perancangan Sistem
4. Implementasi
5. Hasil Uji Coba dan Evaluasi
6. Kesimpulan dan Saran
7. Daftar Pustaka

1.6. Sistematika Penulisan

Buku Penelitian ini bertujuan untuk mendapatkan gambaran dari pengerjaan Penelitian ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku Penelitian terdiri atas beberapa bagian seperti berikut ini.

Bab I Pendahuluan

Bab ini menjelaskan tentang latar belakang diperlukannya pembuatan Penelitian, rumusan masalah, batasan masalah, tujuan, manfaat, metodologi, serta sistematika penulisan dari pembuatan Penelitian.

Bab II Tinjauan Pustaka

Bab ini berisi kajian teori dari metode dan algoritma yang digunakan dalam penyusunan Penelitian ini. Secara garis besar, bab ini berisi tentang teknologi, bahasa pemrograman, dan alat yang digunakan berdasarkan literatur.

Bab III Perancangan Sistem

Bab ini berisi pembahasan-pembahasan secara detil dari sistem yang dibuat. Bentuk penjelasan berupa data gambaran dari implementasi seperti kasus penggunaan beserta spesifikasinya.

Bab IV Implementasi

Bab ini menjelaskan implementasi dalam bentuk gambaran sistem yang akan dibuat. Di dalamnya terdapat langkah-langkah pembuatan sistem kontrol mesin kehadiran terdistribusi yang menggunakan WebSocket.

Bab V Hasil Uji Coba dan Evaluasi

Pada bab ini disajikan pemaparan hasil uji coba dari kasus-kasus yang telah diujikan yang berguna sebagai penunjang pengerjaan Penelitian ini sesuai dengan metode yang telah direncanakan sebelumnya.

Bab VI Kesimpulan dan Saran

Berdasarkan hasil yang didapatkan dari proses uji coba kasus yang telah dilakukan akan ditarik kesimpulan yang disampaikan pada bab ini, serta sara-saran untuk kepentingan pengembangan solusi terkait penanganan masalah yang ada pada proses pengerjaan Penelitian ini.

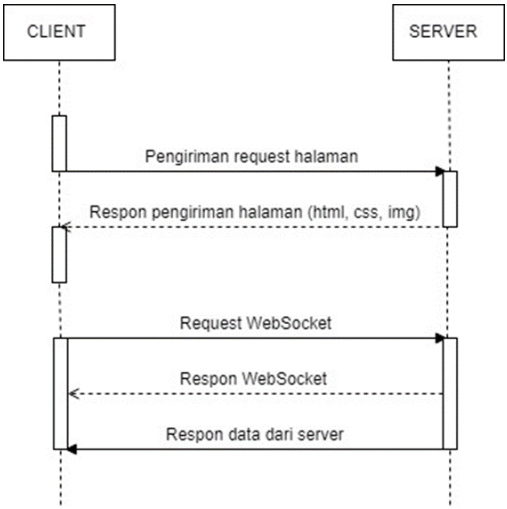
Daftar Pustaka

Bab ini berisi literatur yang berguna sebagai referensi pengerjaan Penelitian yang disajikan didalam daftar pustaka.

BAB II
TINJAUAN PUSTAKA

Pada bab ini dipaparkan dasar-dasar teori yang diaplikasikan pada pengerjaan Penelitian ini, diantaranya adalah penunjang perangkat keras dan perangkat lunak, seperti WebSocket, Socket.io, Node.js, Raspberry Pi 3 Model B, PN-532 dan beberapa teori lainnya yang menunjang pengerjaan Penelitian.

2.1. WebSocket



Gambar 2.1 Diagram WebSocket

WebSocket adalah sebuah protokol komunikasi dua arah yang biasa digunakan sebagai pendukung komunikasi secara *realtime* oleh *browser*. WebSocket dinilai sebagai protokol yang paling cocok untuk diterapkan pada komunikasi dua arah secara *realtime* dan *full-duplex* antara *browser* dan server [1], karena tidak harus menunggu adanya respon balik setelah mengirimkan perintah [1]. Gambar 2.1 menjelaskan

bahwa pihak klien pada WebSocket hanya melakukan *request* pada koneksi awal pada server, kemudian server dapat memberikan tanggapan tanpa menunggu klien mengirimkan *request* terlebih dahulu pada koneksi berikutnya. Oleh karena meningkatnya efisiensi komunikasi antara klien dan server, komunikasi dua arah pada WebSocket menjadi lebih ringan apabila dibandingkan dengan HTTP.

Pengerjaan Penelitian ini menuntut adanya komunikasi dua arah secara *realtime*, sehingga penggunaan WebSocket dinilai paling cocok untuk diterapkan guna membantu menyelesaikan permasalahan yang ada apabila dibandingkan dengan teknik *pooling* yang cenderung membebani server karena adanya *request* oleh klien secara berkala, atau teknik *piggyback* yang tidak memungkinkan adanya perubahan apabila tidak ada *request* dari klien.

Pada pengerjaan Penelitian ini WebSocket digunakan sebagai protokol untuk menyampaikan pesan atau perintah dari klien – sistem kontrol menuju klien – sistem kehadiran melalui server pusat, begitu pula ketika klien – mesin kehadiran memberikan respon kepada klien – sistem kontrol melalui server pusat.

2.2. Node.js

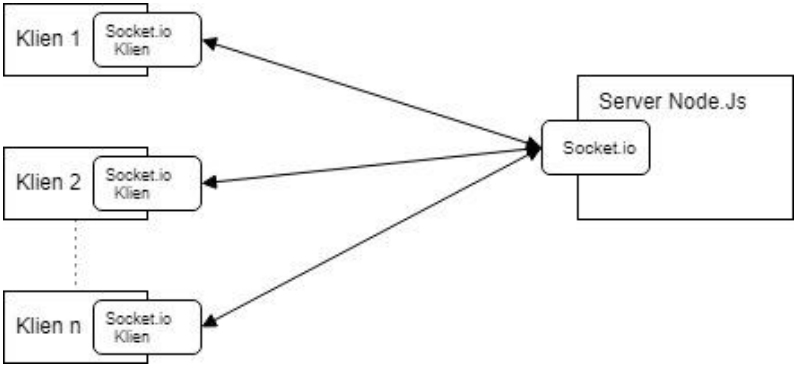
Node.js adalah *runtime* JavaScript yang dibangun diatas Google Chrome V8 Javascript engine. Sebagai *runtime* JavaScript yang berbasis asinkron dan menggunakan teknik *event-driven*, *runtime* memang dirancang untuk membangun aplikasi jaringan yang terukur, sehingga menjadi ringan dan efisien [4] karena tidak harus menunggu selesainya proses data sebelumnya selama data yang dibutuhkan untuk diproses sudah tersedia. Hal ini berbeda dengan model *concurrency* yang sekarang lebih umum digunakan, jaringan berbasis *thread* tersebut relatif tidak efisien dan sangat sulit untuk digunakan.

Sistem dari Node.js adalah asinkron, yaitu tidak ada blok yang menunggu data kemudian proses baru dijalankan. Karena

tidak ada blok, sistem terukur sangat masuk akal untuk dikembangkan di Node [5].

Pada pengerjaan Penelitian ini Node.Js digunakan sebagai WebSocket Server yang bertugas untuk melakukan pengiriman dan meneruskan perintah kepada masing-masing mesin kehadiran dari halaman sistem kontrol mesin kehadiran yang terdistribusi dan melihat daftar mesin kehadiran yang terhubung dengan server. Selain itu, Node.Js digunakan untuk membantu pengecekan koneksi ke server dan penerimaan data dari kerangka kerja Laravel. Untuk dapat menerima data dari kerangka kerja Laravel, Node.Js memerlukan Electron untuk proses pengambilan data, sedangkan untuk proses pengecekan koneksi ke server akan dilakukan dengan melakukan instalasi paket Socket.io yang terdapat didalam Node.Js dengan menggunakan npm.

2.3. Socket.io



Gambar 2.2 Alur Komunikasi Server dan Klien pada Socket.io

Socket.io adalah pustaka untuk platform Node.Js, yang dapat melakukan komunikasi *bi-directional* dan *full-duplex* antara klien dan server secara cepat dan reliabel [6]. Modul Socket.io tersedia didalam npm Node.Js.

Seperti yang dijelaskan pada Gambar 2.2, Socket.io dapat memfasilitasi komunikasi dua arah antara klien dan server secara cepat dan dapat diandalkan. Hasil dari penggunaan Socket.io

adalah data *realtime* pada *browser* sehingga Socket.io sangat cocok untuk mendukung pengerjaan Penelitian.

Pada pengerjaan Penelitian ini Socket.io digunakan sebagai WebSocket Server untuk mengirim dan meneruskan perintah dari sistem kontrol menuju masing-masing mesin kehadiran.

2.4. Kerangka Kerja Laravel

Laravel adalah sebuah kerangka kerja web dari bahasa pemrograman *Hypertext Preprocessor* (PHP) yang dapat memberikan standardisasi atas proses pengembangan dan memproses secara otomatis relasi logika non-bisnis yang memudahkan pengembang untuk berfokus kepada implementasi logika bisnis [7]. Kerangka kerja ini dibuat oleh Taylor Otwell dan menganut pola arsitektur *model-view-controller* (MVC). Beberapa fitur pada Laravel adalah *module package system* yang menjadikan Laravel mudah untuk digunakan hanya dengan melakukan instalasi pada paket [8].

Pada pengerjaan Penelitian ini Laravel digunakan untuk membuat sistem kontrol mesin kehadiran terdistribusi yang memiliki halaman untuk mengirim perintah kepada server Socket.io dalam pengerjaan Penelitian ini.

2.5. Kerangka Kerja Electron

Electron (sebelumnya dikenal sebagai Atom Shell) adalah *open-source framework* yang dibuat oleh Cheng Zhao, dan sekarang dikembangkan oleh GitHub. Electron merupakan kerangka kerja untuk *platform* Node.Js yang dapat membuat *native desktop apps* dari bahasa HTML, CSS dan Javascript yang dapat digunakan pada sistem operasi Windows, Linux dan MacOS. Aplikasi GUI desktop dapat dikembangkan dengan menggunakan komponen *frontend* dan *backend* yang awalnya dikembangkan untuk aplikasi web: *runtime* Node.js untuk *backend* dan Chromium untuk *frontend* [9].

Electron menawarkan kemudahan dalam pembuatan aplikasi desktop dengan menggunakan bahasa pemrograman website dan sudah digunakan sebagai kerangka kerja GUI utama

dalam proyek *open-source* ternama, seperti *code editor* GitHub's Atom dan Microsoft Visual Studio Code.

Dalam pengerjaan Penelitian ini, Electron digunakan untuk menampilkan jadwal perkuliahan yang sedang berlangsung beserta ruangannya oleh klien (mesin kehadiran).

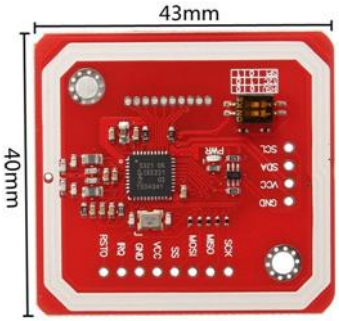
2.6. Raspberry Pi 3 Model B

Gambar 2.3 Raspberry Pi 3 Model B [10]

Gambar 2.3 adalah Raspberry Pi 3 Model B, generasi ketiga Raspberry Pi yang merupakan *Single Board Computer* yang dapat digunakan untuk banyak aplikasi dan menggantikan Raspberry Pi Model B + dan Raspberry Pi 2 B yang asli [11]. Kapabilitas Raspberry Pi adalah yang paling besar jika dibandingkan dengan *mini-computer* lain, sehingga lebih mudah untuk mencari penyelesaian apabila muncul suatu masalah.

Dalam pengembangan sistem kehadiran pada Penelitian ini, semua proses dilakukan pada Raspberry, termasuk pemrosesan data ke *database*, pemindaian kartu dengan menggunakan modul NFC, pengiriman data ke server, hingga menampilkan data ke layar Raspberry dengan menggunakan Electron. Oleh karena itu, Raspberry Pi 3 Model B ini dapat dikatakan sebagai mesin utama penunjang sistem kehadiran ini.

2.7. Modul NFC PN - 532

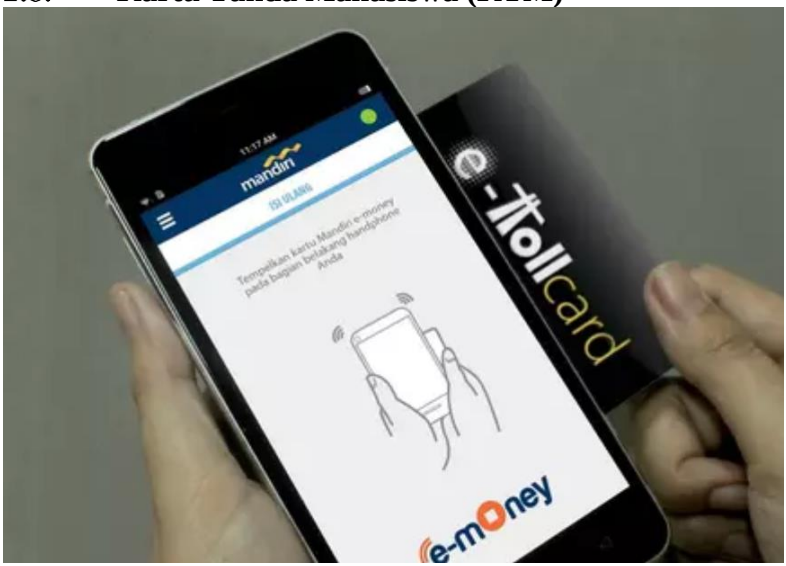


Gambar 2.4 Modul NFC PN532 [12]

Gambar 2.4 menunjukkan modul NFC PN-532, yaitu salah satu modul yang dapat membaca dan menulis tag NFC/RFID. Modul PN-532 memiliki tiga port komunikasi alternatif, yaitu *Serial Peripheral Interface* (SPI), *Inter-Integrated Circuit* (I2C) dan *High Speed UART* (HSU).

PN-532 dapat menangani deteksi *framing* dan *error* ISO / IEC 14443A yang lengkap (Parity & CRC) [13] karena menerapkan *decoder* dan *demodulator* untuk sinyal dari kartu dan transponder kompatibel ISO / IEC 14443A / MIFARE. PN-532 dipilih karena modul ini merupakan modul NFC paling stabil dan paling banyak digunakan serta cocok dengan Raspberry Pi. Pada pengerjaan Penelitian ini modul NFC PN-532 digunakan sebagai pemindai kartu mahasiswa dalam melakukan perekaman data presensi.

2.8. Kartu Tanda Mahasiswa (KTM)



Gambar 2.5 Contoh Kartu yang Dilengkapi NFC [14]

Gambar 2.6 E-KTP Menggunakan Teknoogi NFC [15]

Penggunaan kartu dengan menggunakan *Near Field Communication* seperti yang ditampilkan pada Gambar 2.5 dan Gambar 2.6 sudah banyak diterapkan di Indonesia. Kartu Tanda Mahasiswa (KTM) yang sudah dilengkapi oleh NFC ini yang

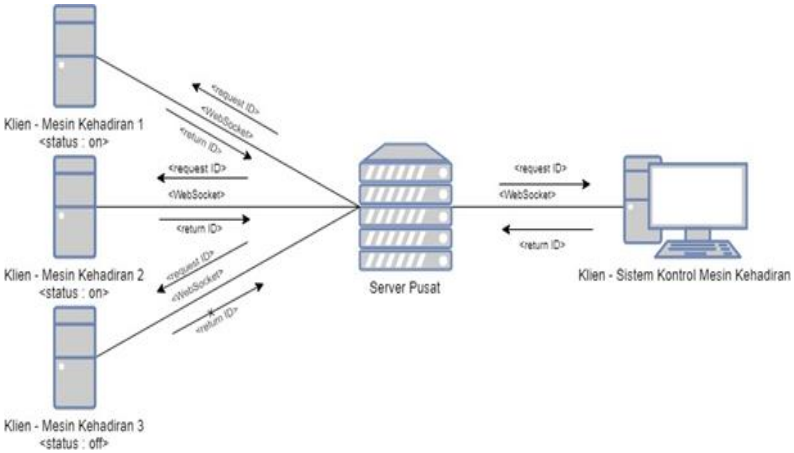
nantinya akan digunakan pada sistem kehadiran. Kartu NFC yang digunakan adalah kartu dengan tipe ISO14443a karena sudah digunakan oleh sebagian besar instansi di Indonesia, seperti pemerintahan dan perbankan. Apabila terjadi kasus tidak membawa atau kehilangan Kartu Tanda Mahasiswa, peserta belajar dapat menggunakan e-KTP yang merupakan kartu dengan tipe yang sama untuk registrasi ulang dengan identitas yang sama, sehingga rekaman data dari peserta belajar tersebut tidak hilang dan dapat melakukan pendataan presensi seperti biasa.

BAB III
PERANCANGAN SISTEM

Bab perancangan sistem menjelaskan tentang perancangan dan pembuatan sistem kontrol mesin kehadiran terdistribusi berbasis web dengan menggunakan WebSocket dengan menjelaskan gambaran umum sistem.

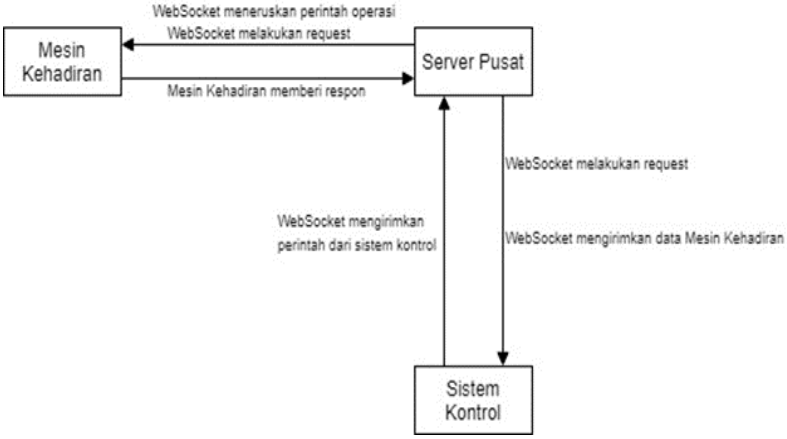
3.1. Arsitektur Sistem

Pada sub-bab arsitektur sistem ini akan dijelaskan rancang bangun sistem kontrol mesin kehadiran terdistribusi berbasis web dengan menggunakan WebSocket secara keseluruhan.



Gambar 3.1 Arsitektur Sistem Kontrol Mesin Kehadiran Terdistribusi Berbasis Web dengan Menggunakan WebSocket

Gambar 3.1 menjelaskan bahwa desain sistem kontrol mesin kehadiran dibagi menjadi 3 bagian, yaitu sisi klien – mesin kehadiran, sisi klien – sistem kontrol mesin kehadiran dan server pusat. Penjelasan lebih dalam akan dipaparkan pada bagian desain sistem kontrol mesin kehadiran terdistribusi.

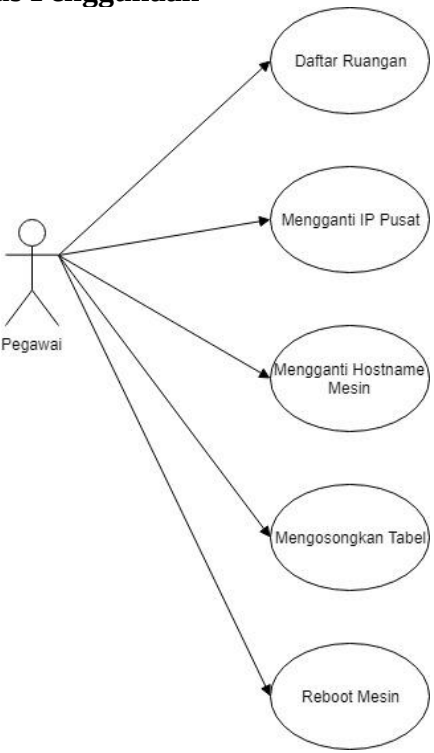


Gambar 3.2 Ilustrasi Umum Sistem Kontrol Mesin Kehadiran

Ilustrasi pada Gambar 3.2 menggambarkan alur yang terjadi secara umum dimulai dari proses awal mesin kehadiran mencoba melakukan koneksi ke server pusat. Permintaan koneksi dari klien – mesin kehadiran akan diteruskan oleh WebSocket menuju server dan apabila berhasil maka akan didapatkan detail mesin kehadiran yang terkoneksi.

Gambar 3.2 juga menggambarkan proses pengiriman perintah yang akan diterima oleh mesin kehadiran dari WebSocket yang dikirimkan oleh bagian klien – sistem kontrol melalui server pusat. Perintah yang dikirimkan akan berisi identitas socket yang dimiliki oleh mesin kehadiran beserta operasi apa yang harus dijalankan oleh mesin kehadiran.

3.2. Kasus Penggunaan



Gambar 3.3 Kasus Penggunaan Sistem Kontrol Mesin Kehadiran Terdistribusi Berbasis Web dengan Menggunakan WebSocket

Gambar 3.3 adalah gambaran kasus penggunaan dari sistem kontrol akan dijelaskan lebih lanjut pada Tabel 3.1.

Tabel 3.1 Daftar Kode Kasus Penggunaan

Kode Kasus Penggunaan	Nama	Aktor	Deskripsi
UC001	Daftar Ruangan	Pegawai	Pegawai dapat melihat informasi daftar mesin diruangan mana saja yang

			terkoneksi dengan server atau tidak
UC002	Mengganti IP Pusat	Pegawai	Pegawai dapat melakukan penggantian IP server pusat pada mesin untuk mengganti tujuan koneksi dari masing-masing mesin
UC003	Mengganti Hostname Mesin	Pegawai	Pegawai dapat melakukan penggantian *hostname* atau nama ruangan pada masing-masing mesin dan data presensi dari mahasiswa akan masuk sesuai dengan *hostname* atau nama ruangan baru yang ditetapkan
UC004	Reboot Mesin	Pegawai	Pegawai dapat menjalankan perintah *reboot* mesin melalui menu *reboot* yang ada pada daftar

			mesin didalam sistem kontrol
UC005	Mengosongkan Tabel	Pegawai	Pegawai dapat menjalankan perintah untuk mengosongkan data presensi pada *hostname* atau nama ruangan yang tercantum pada mesin melalui menu kosongkan tabel yang ada pada daftar mesin didalam sistem kontrol

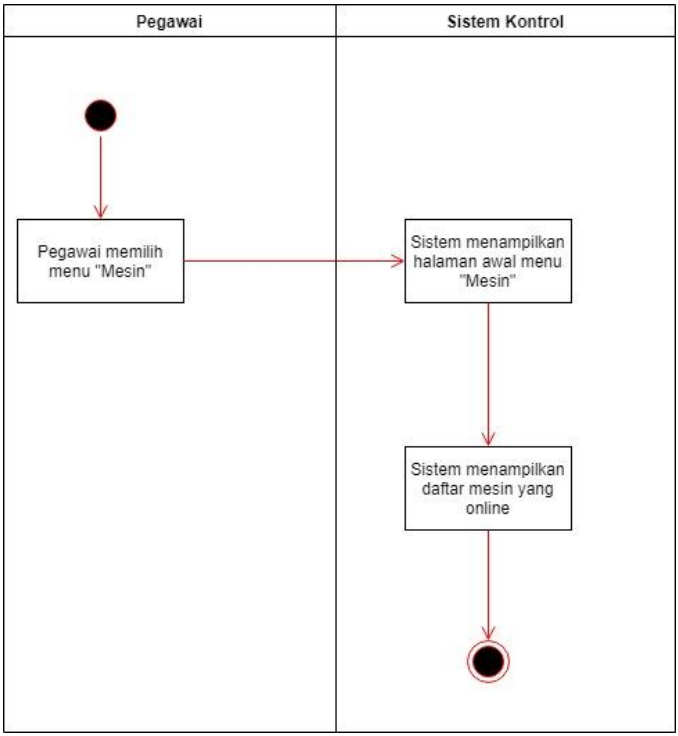
3.2.1. Daftar Ruangan (UC001)

Pada kasus penggunaan ini, pegawai dapat melihat informasi daftar mesin diruangan mana saja yang terkoneksi dengan server atau tidak. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.2 dan diagram aktivitas kasus penggunaan daftar ruangan dapat dilihat pada Gambar 3.4.

Tabel 3.2 Daftar Ruangan

Kode	UC001
Nama	Daftar Ruangan
Aktor	Pegawai
Deskripsi	Pegawai dapat melihat informasi daftar mesin diruangan mana saja yang terkoneksi dengan server atau tidak
Kondisi Awal	Pegawai berada pada halaman awal menu "Mesin"
Kondisi akhir	Pegawai mengetahui mesin mana saja yang menyala dan terhubung ke server
Alur Kejadian Normal	

1. Pegawai memilih menu “Mesin” 2. Sistem kontrol akan menampilkan halaman awal menu “Mesin” 3. Sistem menampilkan daftar mesin yang online
Alur Kejadian Alternatif
-
Eksepsi
3.1. Jika tidak ada mesin yang terhuibung ke server, maka sistem kontrol akan menampilkan daftar kosong 3.2. Apabila ada mesin yang koneksinya terputus dari server, maka data mesin tersebut hilang dari daftar mesin yang sedang ditampilkan oleh sistem kontrol



Gambar 3.4 Diagram Aktivitas Daftar Ruangan (UC001)

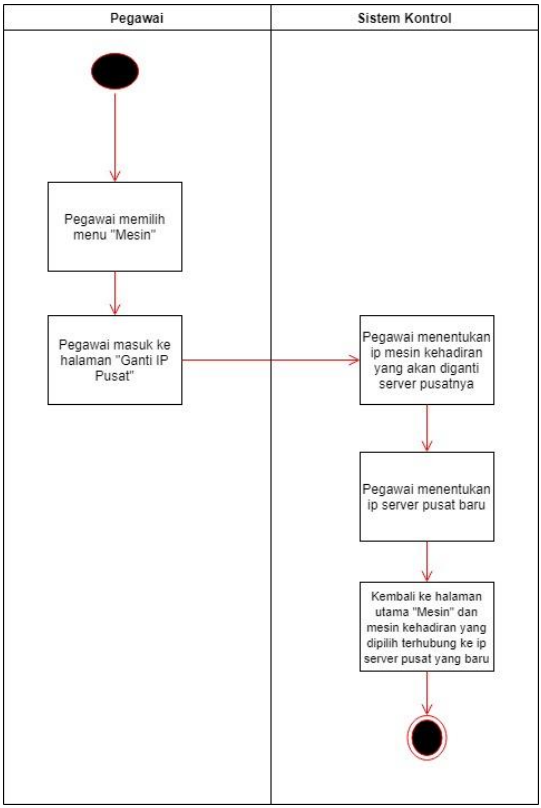
3.2.2. Mengganti IP Pusat (UC002)

Pada kasus penggunaan ini, pegawai dapat melakukan penggantian IP server pusat pada mesin untuk mengganti tujuan koneksi dari masing-masing mesin. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.3 dan diagram aktivitas kasus penggunaan mengganti ip pusat dapat dilihat pada Gambar 3.5.

Tabel 3.3 Mengganti IP Pusat

Kode	UC002
Nama	Mengganti IP Pusat
Aktor	Pegawai
Deskripsi	Pegawai dapat melakukan penggantian IP server pusat pada mesin untuk mengganti tujuan koneksi dari masing-masing mesin
Kondisi Awal	Pegawai masuk ke halaman "Ganti IP Pusat" dari halaman utama "Mesin"
Kondisi akhir	IP server tujuan mesin kehadiran terganti pada mesin yang dipilih
Alur Kejadian Normal	
1. Pegawai memilih menu "Mesin" 2. Pegawai menekan tombol "Ganti IP Pusat" 3. Pegawai masuk ke halaman "Ganti IP Pusat" 4. Pegawai menentukan mesin mana yang akan diganti berdasarkan ip mesin 5. Pegawai menentukan ip server pusat baru 6. Sistem kontrol kembali ke halaman utama "Mesin" dan mesin kehadiran yang dipilih terhubung ke ip server pusat yang baru	
Alur Kejadian Alternatif	
-	
Eksepsi	

4.1. Jika ip mesin kehadiran yang dimasukkan tidak sesuai dengan daftar ip mesin yang sedang online, maka pegawai tidak bisa menentukan ip server pusat yang baru 5.1. Apabila koneksi mesin dengan server terputus ketika melakukan input ip server pust yang baru, maka *kontainer* input ip server pusat baru akan menghilang dan akan muncul pemberitahuan bahwa mesin tidak terkoneksi



Gambar 3.5 Diagram Aktivitas Mengganti IP Pusat (UC002)

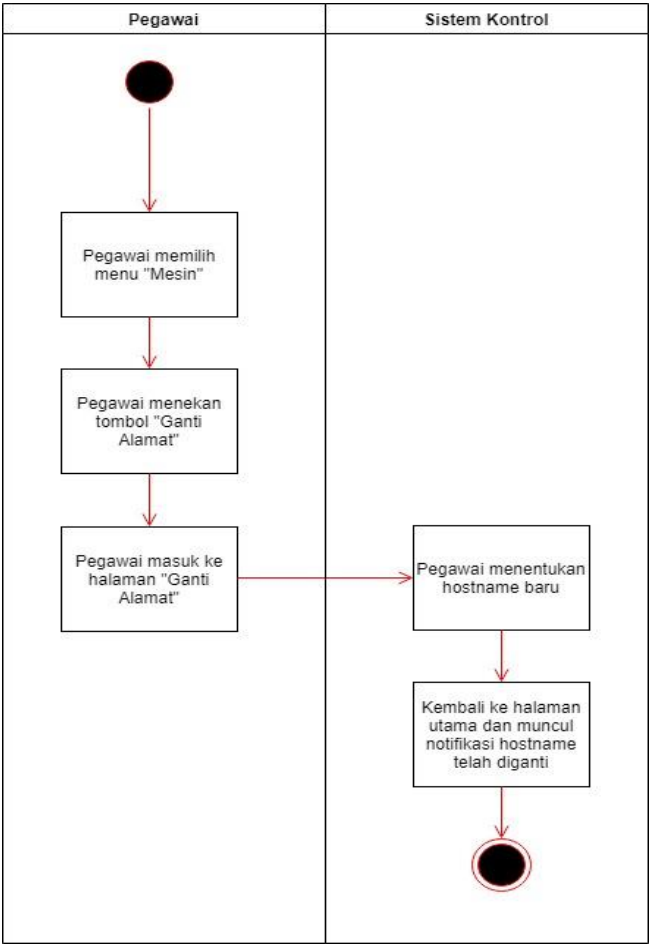
3.2.3. Mengganti Hostname Mesin (UC003)

Pada kasus penggunaan ini, pegawai dapat melakukan penggantian *hostname* atau nama ruangan pada masing-masing mesin dan data presensi dari mahasiswa akan masuk sesuai dengan *hostname* atau nama ruangan baru yang ditetapkan. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.4 dan diagram aktivitas kasus penggunaan mengganti *hostname* mesin dapat dilihat pada Gambar 3.6.

Tabel 3.4 Mengganti Hostname Mesin

Kode	UC003
Nama	Mengganti Hostname Mesin
Aktor	Pegawai
Deskripsi	Pegawai dapat melakukan penggantian *hostname* pada masing-masing mesin dan data presensi dari mahasiswa akan masuk sesuai dengan *hostname* baru yang ditetapkan
Kondisi Awal	Pegawai masuk ke halaman utama "Mesin" dan memilih tombol "Ganti Alamat"
Kondisi akhir	*Hostname* mesin kehadiran yang dipilih berhasil terganti sesuai yang ditentukan oleh pegawai
Alur Kejadian Normal	
1. Pegawai memilih menu "Mesin" 2. Pegawai menekan tombol "Ganti Alamat" 3. Pegawai masuk ke halaman "Ganti Alamat" 4. Pegawai menentukan *hostname* baru 5. Sistem kontrol kembali ke halaman utama "Mesin" dan muncul notifikasi *hostname* telah diganti	
Alur Kejadian Alternatif	
-	
Eksepsi	

2.1	Jika sistem kontrol masih melakukan operasi lain, selain ganti ip server pusat, maka tombol “Ganti Alamat” tidak bisa dipilih



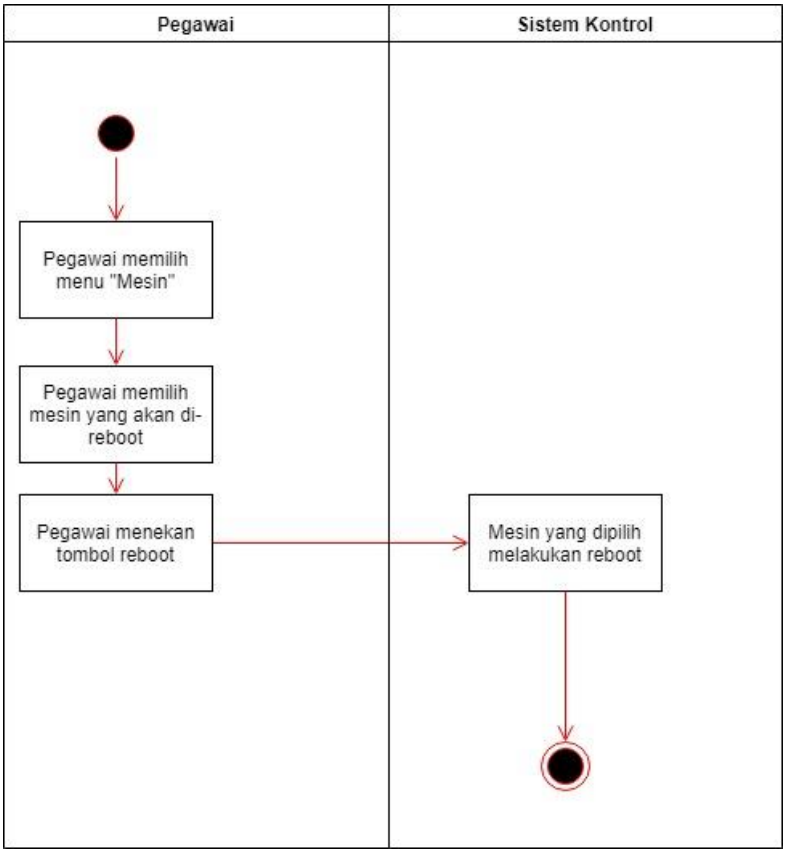
Gambar 3.6 Diagram Aktivitas Mengganti Hostname Mesin (UC003)

3.2.4. Reboot Mesin (UC004)

Pada kasus penggunaan ini, pegawai dapat menjalankan perintah *reboot* mesin melalui menu *reboot* yang ada pada daftar mesin didalam sistem kontrol. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.5 dan diagram aktivitas kasus penggunaan *reboot* mesin dapat dilihat pada Gambar 3.7.

Tabel 3.5 Reboot Mesin

Kode	UC004
Nama	Reboot Mesin
Aktor	Pegawai
Deskripsi	Pegawai dapat menjalankan perintah *reboot* mesin melalui menu *reboot* yang ada pada daftar mesin didalam sistem kontrol
Kondisi Awal	Pegawai masuk ke halaman utama "Mesin"
Kondisi akhir	Mesin kehadiran yang dipilih melakukan *reboot*
Alur Kejadian Normal	
1. Pegawai memilih menu "Mesin" 2. Pegawai memilih mesin yang akan di-*reboot* 3. Pegawai menekan tombol "Reboot" 4. Mesin yang dipilih melakukan *reboot*	
Alur Kejadian Alternatif	
-	
Eksepsi	
1. Jika sistem kontrol masih melakukan operasi lain, selain ganti ip server pusat, maka tombol "Reboot" tidak bisa dipilih	



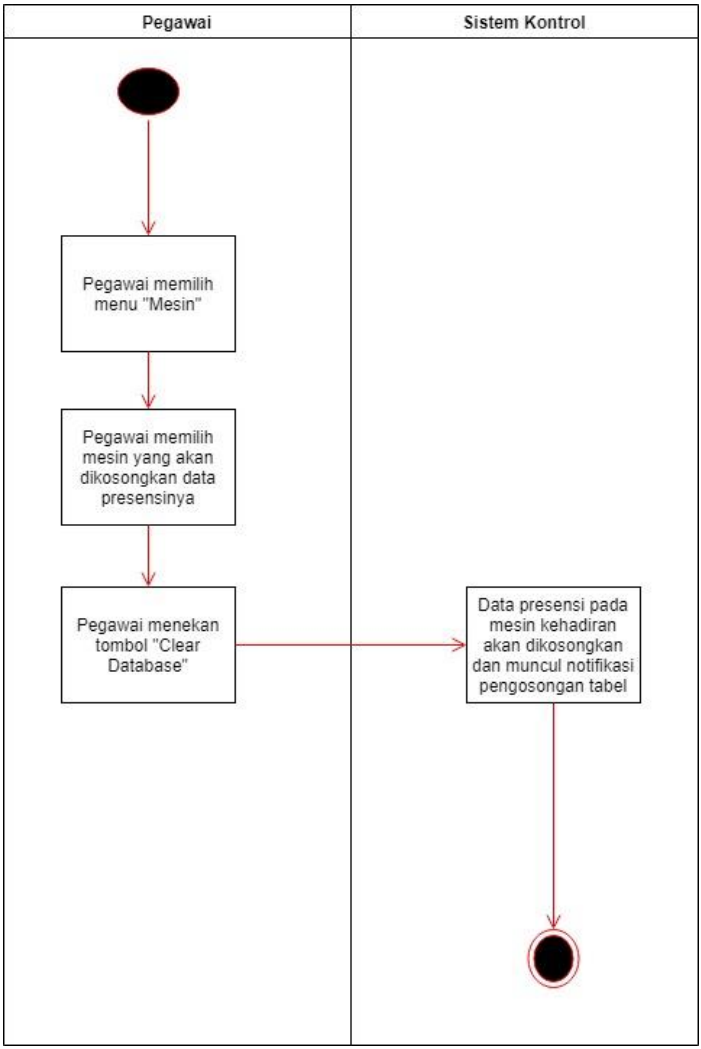
Gambar 3.7 Diagram Aktivitas Reboot Mesin(UC004)

3.2.5. Mengosongkan Tabel (UC005)

Pada kasus penggunaan ini, pegawai dapat menjalankan perintah untuk mengosongkan data presensi pada *hostname* atau nama ruangan yang tercantum pada mesin melalui menu kosongkan tabel yang ada pada daftar mesin didalam sistem kontrol. Spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.6 dan diagram aktivitas kasus penggunaan mengosongkan tabel dapat dilihat pada Gambar 3.8.

Tabel 3.6 Mengosongkan Tabel

Kode	UC005
Nama	Mengosongkan Tabel
Aktor	Pegawai
Deskripsi	Pegawai dapat menjalankan perintah untuk mengosongkan data presensi pada *hostname* yang tercantum pada mesin melalui menu kosongkan tabel yang ada pada daftar mesin didalam sistem kontrol
Kondisi Awal	Pegawai masuk ke halaman utama "Mesin"
Kondisi akhir	Mesin kehadiran yang dipilih mengosongkan data presensi sesuai dengan hostnamenya
Alur Kejadian Normal	
1. Pegawai memilih menu "Mesin" 2. Pegawai memilih *hostname* mesin yang datanya ingin dikosongkan 3. Pegawai menekan tombol "Clear Database" 4. Data pada mesin yang dipilih dikosongkan dan muncul notifikasi pengosongan tabel	
Alur Kejadian Alternatif	
-	
Eksepsi	
2.1 Jika sistem kontrol masih melakukan operasi lain, selain ganti ip server pusat, maka tombol "Clear Database" tidak bisa dipilih	



Gambar 3.8 Diagram Aktivitas Mengosongkan Tabel (UC005)

3.3. Perancangan Antarmuka

Tahap perancangan antarmuka akan menjelaskan tentang perancangan awal ketika sistem kontrol dibuka, tampilan menu dalam tombol hingga alur penggunaan sistem kontrol.

3.3.1. Perancangan Tampilan Awal Sistem Kontrol

Subbab ini menjelaskan bagaimana rancangan antarmuka yang akan berinteraksi secara langsung dengan pegawai sebagai aktor pada saat awal membuka halaman utama sistem kontrol

3.3.1.1. Perancangan Tampilan Awal Aplikasi Dibuka

Pada tampilan awal ketika membuka sistem kontrol pada menu utama “Mesin”, pegawai akan medapati 3 bagian berbeda, yaitu bagian daftar ruangan, riwayat aktivitas dan status mesin di ruangan. Selain itu, pegawai juga akan mendapati adanya tombol “Ganti IP Pusat” yang nantinya akan berguna untuk mengganti alamat server pusat pada mesin, sedangkan bagian riwayat aktivitas akan menampilkan riwayat pengoperasian mesin kehadiran oleh pegawai dan bagian status mesin di ruangan akan menunjukkan mesin kehadiran mana saja yang sedang tidak aktif beserta lokasi kelasnya.

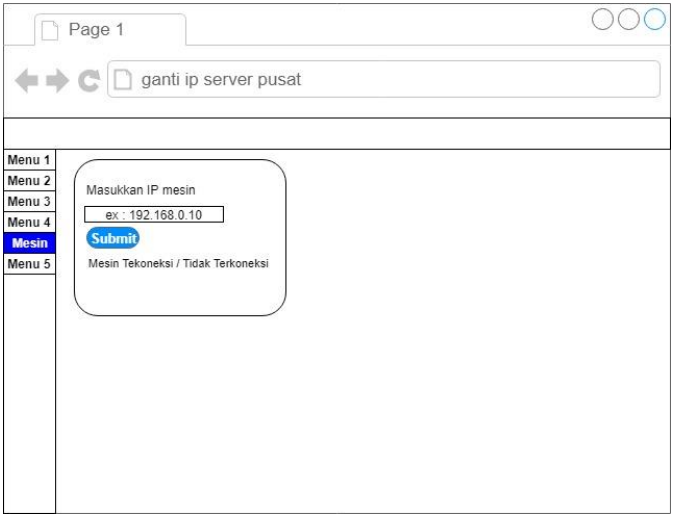
Pada bagian daftar ruangan nantinya akan terdapat pilihan tombol yang akan berguna sebagai proses pengoperasian mesin kehadiran melalui sistem kontrol, mulai dari tombol untuk *reboot* mesin, mengosongkan data presensi dari mesin, hingga mengganti *hostname* mesin. Gambaran dari perancangan tampilan dapat dilihat pada Gambar 3.9.

Gambar 3.9 Rancangan Tampilan Sistem Kontrol Saat Pertama Kali Dibuka

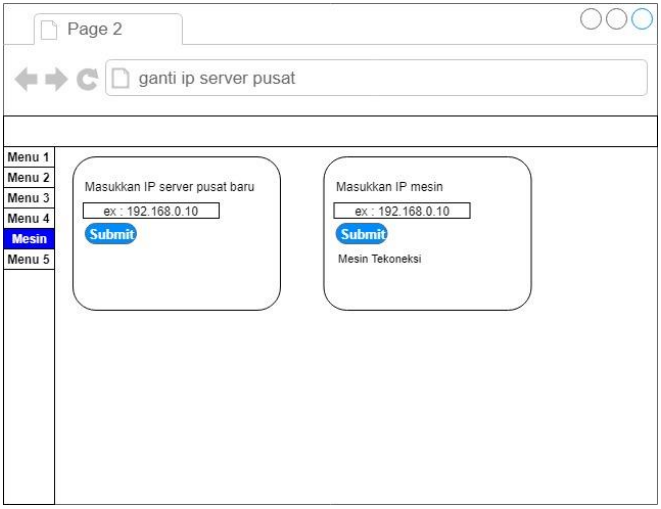
3.3.1.2. Perancangan Tampilan Menu Dalam Tombol

Pada tampilan awal sistem kontrol terdapat dua tombol yang terhubung dengan tampilan halaman yang lain, yaitu tombol "Ganti IP Pusat" yang terdapat diatas bagian daftar ruangan dan tombol "Ganti Alamat" yang ada didalam pilihan tombol masing-masing ruangan didalam *kontainer* daftar ruangan.

Ketika pegawai menekan tombol "Ganti IP Pusat", pegawai akan diarahkan menuju halaman ganti ip server pusat oleh sistem kontrol. Pegawai harus memasukkan ip mesin yang ingin diganti ip server pusatnya yang terkoneksi dengan jaringan yang sama ketika mengakses sistem kontrol, kemudian sistem kontrol akan mengecek apakah mesin terkoneksi atau tidak. Apabila terkoneksi, maka sistem kontrol akan memunculkan satu *kontainer* lagi yang akan berguna untuk menentukan alamat ip server pusat yang baru. Gambaran perancangan antarmuka halaman penggantian ip server pusat dapat dilihat pada Gambar 3.10 dan Gambar 3.11.



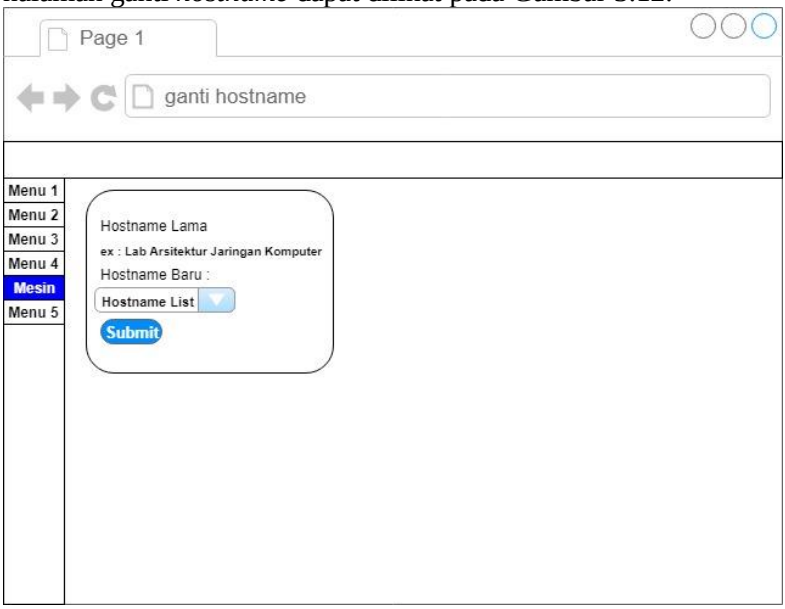
Gambar 3.10 Rancangan Tampilan Menu Dalam Tombol Ganti IP Pusat – Pencarian Alamat Mesin



Gambar 3.11 Rancangan Tampilan Dalam Tombol Ganti IP Pusat - Ganti IP Server Baru

Selain tampilan dalam tombol “Ganti IP Pusat”, sistem kontrol mesin kehadiran juga memiliki satu tampilan dalam tombol

lain pada tombol “Ganti Alamat” yang terletak pada pilihan tombol yang ada pada masing-masing daftar mesin yang ditampilkan didalam *kontainer* daftar ruangan pada halaman utama “Mesin”. Ketika pegawai menekan tombol “Ganti Alamat”, pegawai akan diarahkan menuju halaman ganti *hostname* oleh sistem kontrol. Halaman ganti *hostname* berisi *kontainer* untuk memilih *hostname* atau nama ruangan baru yang nantinya akan ditampilkan pada antarmuka mesin kehadiran. Gambaran rancangan antarmuka halaman ganti *hostname* dapat dilihat pada Gambar 3.12.

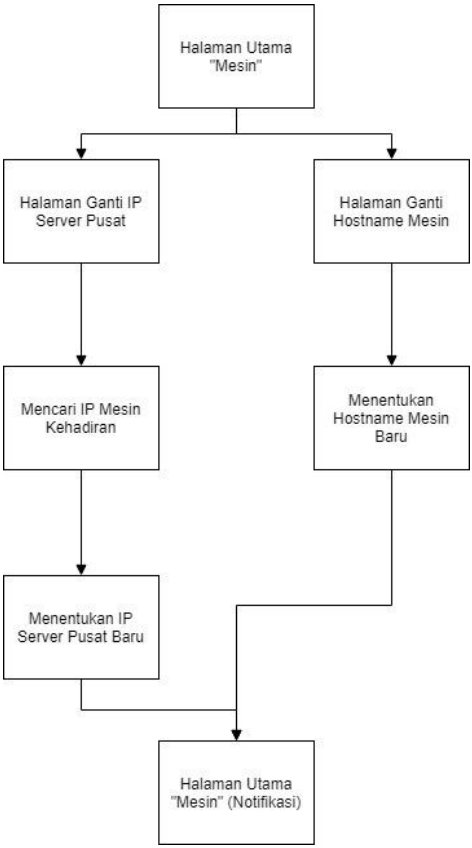


Gambar 3.12 Rancangan Tampilan Menu Dalam Tombol Ganti Alamat

3.3.2. Perancangan Alur Proses Penggunaan Sistem Kontrol Mesin Kehadiran

Pada tahap ini akan dijelaskan mengenai rancangan alur proses penggunaan aplikasi yang digunakan sebagai acuan dalam pembangunan sistem kontrol mesin kehadiran. Alur proses ini membantu memperlihatkan langkah demi langkah yang akan

dilakukan pengguna. Alur penggunaan aplikasi akan dijelaskan pada Gambar 3.13.



Gambar 3.13 Diagram Alur Penggunaan Sistem Kontrol Mesin Kehadiran

[Halaman ini sengaja dikosongkan]

BAB IV
IMPLEMENTASI

Bab ini membahas implementasi perancangan sistem yang telah dibahas pada Bab III. Namun dalam penerapannya, rancangan tersebut dapat mengalami perubahan minor sewaktu-waktu apabila dibutuhkan.

4.1. Lingkungan Implementasi

Dalam implementasinya, lingkungan yang digunakan menggunakan beberapa perangkat pendukung seperti yang ditunjukkan pada Tabel 4.1.

Tabel 4.1 Lingkungan Implementasi Penelitian

Perangkat	Jenis Perangkat	Spesifikasi
Perangkat Keras Mesin Kehadiran	Mini Komputer	Raspberry Pi 3 Model B
	Modul NFC	PN-532
	SD Card	SanDisk Ultra 16GB
	Monitor	KeDei 3.5 inch 480x320
	Adaptor	Part Resmi Raspberry Pi
Perangkat Lunak Mesin Kehadiran	Sistem Operasi	Raspbian
	Perangkat Pengembang	Socket.io 2.0
		Node.js v8.9.1
Perangkat Keras Server	Prosesor	Intel Core i5 – 6267U
	Memori	8 GB

Perangkat Lunak Server	Sistem Operasi	MacOS 10.13.4
	Perangkat Pengembang	Socket.IO 2.0
		Node.js v8.9.1
		Laravel 5.4
Perangkat Keras Sistem Kontrol	Prosesor	Intel(R) Core(TM) i5-7200U
	Memori	8 GB
Perangkat Lunak Sistem Kontrol	Sistem Operasi	Linux Ubuntu 16.04
	Perangkat Pengembang	Sublime Text sebagai *editor* kode

4.2. Implementasi Perangkat Lunak

Bab implementasi perangkat lunak akan menjelaskan tentang pembangunan perangkat lunak seperti yang telah direncanakan pada Bab III. Pada pengerjaan Penelitian ini implementasi perangkat lunak dibagi menjadi implementasi pada server, implementasi pada mesin kehadiran dan implementasi pada sistem kontrol.

4.2.1. Implementasi pada Server

Pada subbab implementasi pada server ini akan dijelaskan bagaimana pembangunan perangkat lunak yang digunakan pada server pusat secara detail serta kode sumber yang digunakan untuk pembuatan Penelitian ini.

4.2.1.1. Implementasi Reboot Mesin Kehadiran

Bagian ini adalah implementasi dari kasus penggunaan Reboot Mesin (UC004). Implementasi *reboot* mesin kehadiran diawali dengan aktivitas pegawai dengan menekan tombol “Reboot” pada daftar mesin yang ditampilkan pada sistem kontrol mesin kehadiran. Ketika pegawai menekan tombol “Reboot”, sistem kontrol akan mengirimkan identitas mesin kehadiran yang

dipilih menuju ke server pusat yang nantinya akan meneruskan perintah menuju mesin kehadiran.

```
1.  CALL rebootMesin(socket,msg)
2.       restart ← socket.to(msg)(‘restartMesin’)
```

Kode Sumber 4.1 Implementasi Reboot Mesin Kehadiran pada Server

Kode Sumber 4.1 menjelaskan bahwa server akan memanggil fungsi “rebootMesin” yang berisikan pesan yang nantinya akan dibawa oleh socket menuju mesin kehadiran yang dituju untuk menjalankan fungsi “restartMesin” yang akan dijelaskan kemudian pada subbab 4.2.2.1. Dari fungsi “restartMesin” yang dijalankan pada mesin kehadiran inilah akan dilakukan operasi *reboot* mesin sesuai dengan perintah yang dikirimkan oleh server.

4.2.1.2. Implementasi Membersihkan Data Presensi pada Tabel dalam Mesin Kehadiran

Bagian ini adalah implementasi dari kasus penggunaan Mengosongkan Tabel (UC005). Implementasi membersihkan data presensi pada tabel dalam mesin kehadiran diawali dengan aktivitas pegawai dengan menekan tombol “Clear Database” pada pilihan tombol yang terdapat di daftar mesin yang ditampilkan pada sistem kontrol mesin kehadiran. Ketika pegawai menekan tombol “Clear Database”, sistem kontrol akan mengirimkan identitas mesin kehadiran yang dipilih menuju ke server pusat yang nantinya akan meneruskan perintah menuju mesin kehadiran.

```
1.  CALL clearTable(socket,msg)
2.       kosongkan ← socket.to(msg)(‘bersihkanTabel’)
```

Kode Sumber 4.2 Implementasi Membersihkan Data Presensi pada Tabel dalam Mesin Kehadiran – Mengirim Perintah pada Mesin Kehadiran

Kode Sumber 4.2 menjelaskan bahwa server akan memanggil fungsi “clearTable” yang berisikan pesan yang

nantinya akan dibawa oleh socket menuju mesin kehadiran yang dituju untuk menjalankan fungsi "bersihkanTabel" yang akan dijelaskan kemudian pada subbab 4.2.2.2. Setelah mesin kehadiran menerima perintah untuk membersihkan data dari server dan selesai menjalankan fungsi "bersihkanTabel", server akan menerima info dari mesin kehadiran bahwa data telah bersih yang dipastikan dengan proses *acknowledgement* pada server seperti yang digambarkan pada Kode Sumber 4.3. Kemudaian server akan mengirimkan rekaman menuju ke *database* untuk ditampilkan di riwayat aktivitas pada sistem kontrol mesin kehadiran.

```
1.  CALL bersihInfo(socket,socket.id)
2.       ACK()
3.       insertLogDB(tipe, ruangan, waktu, socket_id,
4.       ip_mesin)
5.       sendSistemKontrol(log)
```

Kode Sumber 4.3 Implementasi Membersihkan Data Presensi pada Tabel dalam Mesin Kehadiran – Menerima Informasi dari Mesin Kehadiran

4.2.1.3. Implementasi Mengganti Hostname Mesin Kehadiran

Bagian ini adalah implementasi dari kasus penggunaan Mengganti Hostname Mesin (UC003). Implementasi mengganti *hostname* mesin kehadiran diawali dengan aktivitas pegawai dengan memilih *hostname* atau nama ruangan baru untuk mesin dan kemudian dipastikan dengan menekan tombol "Submit". Setelah pegawai menentukan nama *hostname* atau nama ruangan mesin kehadiran yang baru dan kemudian melakukan *submit*, maka sistem kontrol akan mengirimkan pesan menuju server pusat berupa *socket id* dari mesin yang dimaksud beserta *hostname* atau nama ruangan baru yang dipilih.

```
1.  CALL gantiAlamat(socket,msg)
```

```
2.       ganti ← socket.to(msg.id)('gantiHostname')
```

Kode Sumber 4.4 Implementasi Mengganti Hostname Mesin – Mengirim Perintah pada Mesin Kehadiran

Kode Sumber 4.4 menjelaskan bahwa server akan memanggil fungsi "gantiAlamat" yang berisikan pesan yang nantinya akan dibawa oleh *socket* menuju mesin kehadiran yang dimaksud untuk menjalankan fungsi "gantiHostname" yang akan dijelaskan kemudian pada subbab 4.2.2.3. Setelah mesin kehadiran menerima perintah untuk mengganti *hostname* atau nama ruangan dari server dan berhasil menjalankan fungsi "gantiHostname", server akan menerima info dari mesin kehadiran bahwa *hostname* atau nama ruangan mesin kehadiran telah diganti yang dipastikan dengan proses *acknowledgement* pada server seperti yang dijelaskan pada Kode Sumber 4.5. Kemudaian server akan mengirimkan rekaman menuju ke database untuk ditampilkan di riwayat aktivitas pada sistem kontrol mesin kehadiran.

```
1.  CALL gantiHostInfo(socket,msg)
2.       ACK()
3.       insertLogDB(tipe, ruangan, waktu, socket_id,
4.       ip_mesin, hostname_baru)
5.       sendSistemKontrol(log)
```

Kode Sumber 4.5 Implementasi Membersihkan Data Presensi pada Tabel dalam Mesin Kehadiran – Menerima Informasi dari Mesin Kehadiran

4.2.1.4. Implementasi Menampilkan Daftar Ruangan

Bagian ini adalah implementasi dari kasus penggunaan Daftar Ruangan (UC001). Implementasi menampilkan daftar ruangan diawali dengan pesan berupa ip mesin kehadiran dan *hostname* mesin yang dikirimkan oleh mesin kehadiran melalui pustaka *Socket.io server* menuju server pusat seperti yang dijelaskan pada Kode Sumber 4.11.

```
1.  VAR { ruanganlist }
2.  CALL mesinConnect(socket,msg)
```

```
3.       ip_mesin[socket.id] ← msg[‘ip’]
4.       client[socket.id] ← msg[‘ruangan’]
5.       IF ruanganlist[‘ruangan’] != empty THEN
6.             push(socket.id)
7.       ELSE CREATE ruanganlist[‘ruangan’]=[socket.id]
8.       ENDIF
9.       io.emit(‘klien’,{ruanganlist,ip_mesin})
```

Kode Sumber 4.6 Implementasi Menampilkan Daftar Ruangan

Kode Sumber 4.6 menjelaskan bahwa server akan memanggil fungsi “mesinConnect” setelah menerima pesan berupa ip mesin dan *socket id* dari sistem kontrol mesin kehadiran. Kemudian server akan melakukan pengecekan terhadap variabel “ruanganlist” yang sebelumnya sudah dideklarasikan. Variabel “ruanganlist” merupkan variabel yang berisi *hostname* atau nama ruangan mesin, jadi apabila dalam proses pengecekan sudah ada *socket id* mesin kehadiran yang terhubung pada *hostname* atau nama ruangan tersebut, maka server akan menambahakan *socket id* yang baru kedalam *array* yang ada pada *hostname* yang sama karena satu *hostname* memang memungkinkan untuk diisi oleh lebih dari satu *socket id*. Akan tetapi, apabila pesan yang masuk mengirimkan *hostname* atau nama ruangan yang sebelumnya belum terekam di server pada proses ini, maka server akan membuat *hostname* atau nama ruangan baru yang berisi satu *socket id* mesin kehadiran yang baru terhubung tersebut. Setelah selesai melakukan pemrosesan terhadap variabel “ruanganlist”, maka server akan mengirimkan pesan berupa hasil pemrosesan variabel “ruanganlist” beserta ip masing-masing mesin kehadiran yang terhubung ke server menuju ke sistem kontrol untuk disajikan dalam bentuk tabel daftar mesin dan status mesin dalam masing-masing ruangan yang nantinya akan dijelaskan lebih lanjut pada subbab 4.2.3.

4.2.2. Implementasi pada Mesin Kehadiran

Pada subbab implementasi pada mesin kehadiran ini akan dijelaskan bagaimana pembangunan perangkat lunak yang

digunakan pada mesin kehadiran secara detail serta kode sumber yang digunakan untuk pembuatan Penelitian ini.

4.2.2.1. Implementasi Reboot Mesin Kehadiran

Bagian ini adalah implementasi dari kasus penggunaan Reboot Mesin (UC004). Pada proses yang ditunjukkan pada Kode Sumber 4.7 ini mesin kehadiran akan menjalankan fungsi "restartMesin" sesuai yang diperintahkan oleh server pusat seperti yang dijelaskan pada Kode Sumber 4.1 yang kemudian membuat mesin kehadiran melakukan *reboot*.

```
1.  CALL restartMesin(socket)
2.       reboot()
```

Kode Sumber 4.7 Implementasi Reboot Mesin Kehadiran pada Mesin Kehadiran

4.2.2.2. Implementasi Membersihkan Data Presensi pada Tabel dalam Mesin Kehadiran

Bagian ini adalah implementasi dari kasus penggunaan Mengosongkan Tabel (UC005). Dalam proses yang digambarkan pada Kode Sumber 4.8 dijelaskan bahwa mesin kehadiran akan menjalankan fungsi "bersihkanTabel" setelah menerima pesan berupa *socket id* dari mesin yang dituju seperti yang dijelaskan pada Kode Sumber 4.2. Setelah melakukan proses pengosongan tabel, socket pada mesin akan mengirimkan info kepada server pusat bahwa proses pembersihan data pada mesin telah selesai.

```
1.  CALL bersihkanTabel(socket, socket.id)
2.       mengosongkanTabel()
3.       bersih ← socket.emit('bersihInfo')
```

Kode Sumber 4.8 Implementasi Membersihkan Data Presensi pada Tabel dalam Mesin Kehadiran

4.2.2.3. Implementasi Mengganti Hostname Mesin Kehadiran

Bagian ini adalah implementasi dari kasus penggunaan Mengganti Hostname Mesin (UC003). Dalam proses yang digambarkan pada Kode Sumber 4.9 dijelaskan bahwa mesin kehadiran akan menjalankan fungsi "gantiHostname" setelah menerima pesan berupa *socket id* dan *hostname* atau nama ruangan mesin yang baru seperti yang tertera pada Kode Sumber 4.4 dan kemudian mengirimkan informasi hasil proses ganti *hostname* kepada server pusat.

```
1.  CALL gantiHostname(socket,msg)
2.       ubahHost ← socket.emit('gantiHostInfo')
```

Kode Sumber 4.9 Implementasi Mengganti Hostname Mesin Kehadiran – Mengirim Perintah ke Server Pusat

Kode Sumber 4.10 menjelaskan ketika mesin kehadiran menerima pesan untuk melakukan operasi ganti *hostname* atau nama ruangan. Mesin kehadiran akan memeriksa apakah perintah sudah diterima atau belum. Apabila sudah diterima, maka *hostname* atau nama ruangan pada mesin kehadiran akan diubah dan mesin akan melakukan *reboot*.

```
1.  IF ubahHost == received THEN
2.       gantiHostname()
3.       reboot()
4.  ENDIF
```

Kode Sumber 4.10 Implementasi Mengganti Hostname Mesin Kehadiran – Melakukan Pengecekan Perintah

4.2.2.4. Implementasi Menampilkan Daftar Ruangan

Bagian ini adalah implementasi dari kasus penggunaan Daftar Ruangan (UC001). Implementasi menampilkan daftar ruangan diawali dengan aktivitas mesin yang mencoba melakukan koneksi kepada server pusat.

```
1.  CALL connectOn(socket)
2.       koneksi ← socket.emit('mesinConnect',ip)
```

Kode Sumber 4.11 Implementasi Menampilkan Daftar Ruangan

Kode Sumber 4.11 menggambarkan aktivitas mesin kehadiran yang mencoba melakukan koneksi terhadap server pusat. Apabila fungsi "connectOn" berhasil dilakukan, maka mesin akan mengirimkan pesan kepada server pusat berupa ip mesin kehadiran. Setelah pesan dari mesin kehadiran diterima oleh server pusat, maka server pusat akan menjalankan fungsi "mesinConnect" seperti yang telah dijelaskan pada Kode Sumber 4.6.

4.2.2.5. Implementasi Mengganti IP Server Pusat pada Mesin Kehadiran

Bagian ini adalah implementasi dari kasus penggunaan Mengganti IP Pusat (UC002). Pada proses yang ditunjukkan pada Kode Sumber 4.12, mesin kehadiran akan menjalankan fungsi "gantiIP". Pada proses ini mesin kehadiran tidak menerima pesan atau perintah dari server pusat karena pada kasus ini mesin kehadiran bertindak sebagai server karena mesin kehadiran sudah memiliki ip mesin sehingga pegawai harus mencari tahu ip mesin tersebut sebelum mencoba mengirimkan request melalui sistem kontrol. Ip server pusat yang baru nantinya akan ditulis pada file "ip.txt" dan kemudian mesin kehadiran akan melakukan *reboot* agar ip server pusat yang baru dapat dibaca.

```
1.  CALL gantiIP(msg)
2.       writeToFile('/ip.txt' , 'msg')
3.       reboot()
```

Kode Sumber 4.12 Implementasi Reboot Mesin Kehadiran pada Mesin Kehadiran

4.2.3. Implementasi pada Sistem Kontrol Mesin Kehadiran

Pada subbab implementasi pada sistem kontrol mesin kehadiran ini akan dijelaskan bagaimana pembangunan perangkat

lunak yang digunakan pada sistem kontrol secara detail serta kode sumber yang digunakan untuk pembuatan Penelitian ini.

Pada bagian ini dijelaskan implementasi dari kasus penggunaan Daftar Ruangan (UC001). Implementasi menampilkan daftar ruangan pada sistem kontrol dilakukan setelah sistem kontrol sebagai klien menerima perintah untuk menjalankan fungsi “klien” beserta pesan berupa isi dari variabel “ruanganlist” yang sudah dijelaskan sebelumnya pada Kode Sumber 4.6.

```
1.  clearTable()
2.  CALL klien(data)
3.       FOREACH item in data[ruanganlist]
4.              FOREACH mesin in item :
5.                      tableRowAdd(socket.id,hostname)
6.                      drawTable()
7.              ENDFOR
8.       ENDFOR
```

Kode Sumber 4.13 Implementasi Menampilkan Daftar Ruangan – Membuat Tabel Daftar Mesin Kehadiran

Kode Sumber 4.13 menerangkan proses pemanggilan fungsi klien oleh sistem kontrol yang berperan untuk mengambil data yang dikirimkan dari server pusat melalui variabel “ruanganlist” yang telah diproses dan kemudian disajikan dalam bentuk tabel. Tabel awal akan dihapus dan diganti dengan tabel baru setiap ada mesin kehadiran baru yang berhasil melakukan koneksi atau kehilangan koneksi sehingga daftar mesin kehadiran yang ditampilkan hanya mesin yang masih terkoneksi dengan server pusat.

```
1.  FOREACH hostname
2.       IF data[‘ruanganlist’] hasOwnProperty THEN
3.             status == green
4.       ELSE status == red
5.       ENDIF
```

```
6.  ENDFOR
```

Kode Sumber 4.14 Implementasi Menampilkan Daftar Ruangan – Menampilkan Status Mesin dalam Ruangan

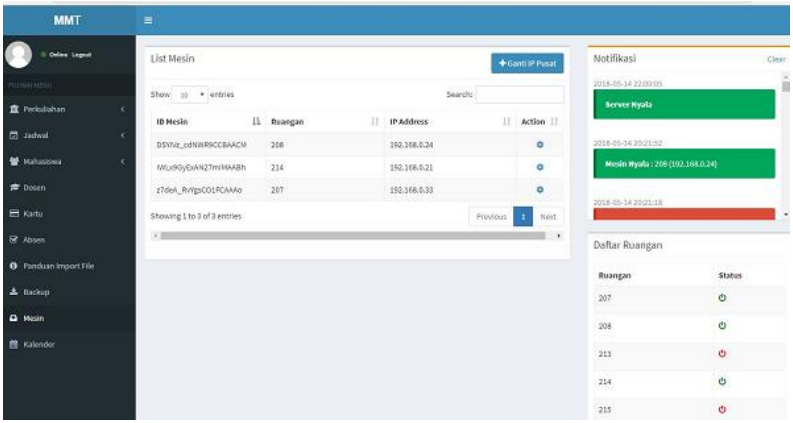
Kode Sumber 4.14 menjelaskan proses penampilan status mesin dalam ruang kelas. Apabila *hostname* atau nama ruangan yang didapat dari ruanganlist tidak kosong atau terdapat id mesin yang terkoneksi atas *hostname* atau nama ruangan tersebut, dalam kode sumber dituliskan sebagai "hasOwnProperty", maka tampilan status ruangan yang sesuai dengan *hostname* tersebut akan berubah menjadi warna hijau. Namun jika *hostname* tersebut tidak memiliki mesin yang terhubung, maka status yang ditampilkan berubah menjadi warna merah.

4.2.4. Implementasi Antarmuka

Subbab implementasi antarmuka menjelaskan tampilan antarmuka sistem kontrol pada halaman utama dan halaman dalam tombol seperti yang telah dijelaskan pada bagian perancangan sistem.

4.2.4.1. Implementasi Halaman Utama Sistem Kontrol

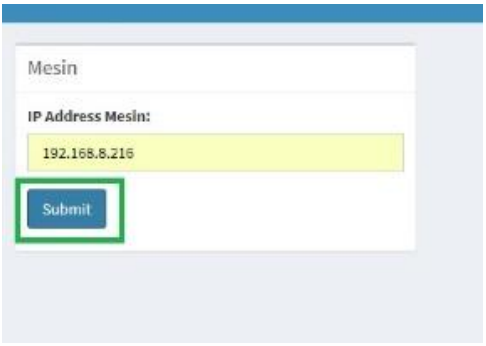
Berdasarkan rancangan tampilan antarmuka halaman utama yang telah dibuat pada Gambar 3.9, halaman sistem kontrol mesin kehadiran dibuat menjadi seperti pada Gambar 4.1. Tampilan antarmuka ketika membuka halaman awal akan dibagi menjadi empat bagian, yaitu daftar menu utama disebelah kiri, dimana menu utama "Mesin" juga terdapat didalamnya, kemudian bagian daftar mesin yang terhubung pada server pusat sistem kontrol, lalu bagian notifikasi atau riwayat aktivitas, serta bagian daftar status mesin pada masing-masing ruang kelas.



Gambar 4.1 Implementasi Halaman Utama "Mesin" pada Sistem Kontrol Mesin Kehadiran

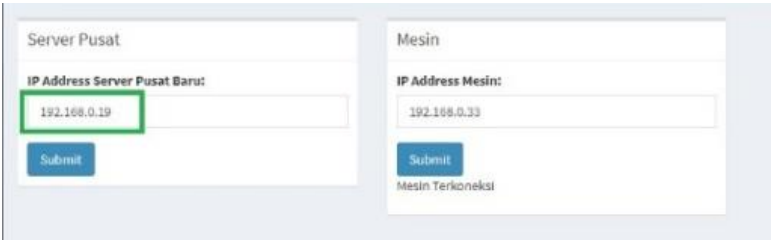
4.2.4.2. Implementasi Halaman dalam Tombol – Ganti IP Pusat

Berdasarkan rancangan tampilan dalam tombol "Ganti IP Pusat" yang sebelumnya sudah dijelaskan pada Gambar 3.10, tampilan antarmuka awal ketika membuka halaman dalam tombol "Ganti IP Pusat" dalam sistem kontrol dibuat seperti yang ditunjukkan pada Gambar 4.2.



Gambar 4.2 Implementasi Halaman dalam Tombol - Ganti IP Pusat

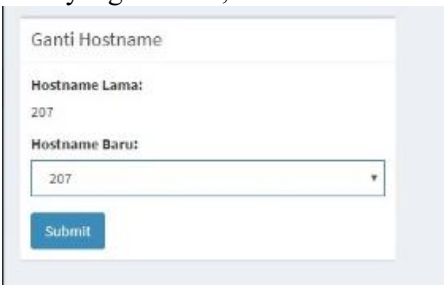
Apabila ip mesin kehadiran yang dimasukkan oleh pegawai terdeteksi oleh sistem, maka sistem kontrol akan menampilkan satu *kontainer* lain yang berfungsi untuk memasukkan alamat ip server pusat yang baru pada mesin kehadiran, seperti yang ditunjukkan pada Gambar 4.3 yang sesuai dengan rancangan pada Gambar 3.11.



Gambar 4.3 Implementasi Halaman dalam Tombol ketika IP Mesin Terdeteksi - Ganti IP Pusat

4.2.4.3. Implementasi Implementasi Halaman dalam Tombol – Ganti Alamat

Berdasarkan rancangan tampilan dalam tombol "Ganti Alamat" yang sebelumnya sudah dijelaskan pada Gambar 3.12, tampilan antarmuka halaman dalam tombol "Ganti Alamat" dalam sistem kontrol dibuat seperti yang ditunjukkan pada Gambar 4.4. Pada halaman dalam tombol "Ganti Alamat" terdapat satu buah *kontainer* yang berisi *hostname* atau nama ruangan lama mesin kehadiran yang dipilih, satu kotak daftar pilihan *hostname* atau nama ruangan baru yang tersedia, serta tombol "Submit".



Gambar 4.4 Implementasi Halaman dalam Tombol - Ganti Alamat

[Halaman ini sengaja dikosongkan]

BAB V
PENGUJIAN DAN EVALUASI

Bab ini membahas pengujian dan evaluasi pada aplikasi yang dikembangkan pada Penelitian ini. Pengujian yang dilakukan adalah pengujian terhadap kebutuhan fungsionalitas sistem yang telah dijabarkan pada Bab III dan terhadap tujuan dibuatnya aplikasi ini, yakni mempermudah fungsi kontrol serta mempercepat penanganan terhadap mesin kehadiran yang bermasalah.

5.1. Lingkungan Pengujian

Lingkungan pengujian menggunakan dua buah komputer yang berfungsi sebagai server dan sistem kontrol. Selain itu pengujian juga menggunakan empat Raspberry Pi 3 Model B yang telah dirancang menjadi mesin kehadiran lengkap dengan sensor PN532.

Spesifikasi lingkungan pengujian yang berfungsi sebagai server dapat dilihat pada Tabel 5.1.

Tabel 5.1 Spesifikasi Lingkungan Pengujian - Server

Perangkat	**Jenis Perangkat**	**Spesifikasi**
Perangkat Keras	Prosesor	Intel Core i5 – 6267U
	Memori	8 GB
Perangkat Lunak	Sistem Operasi	MacOS 10.13.4

Spesifikasi lingkungan pengujian yang berfungsi sebagai sistem kontrol dapat dilihat pada Tabel 5.2.

Tabel 5.2 Spesifikasi Lingkungan Pengujian - Sistem Kontrol

Perangkat	**Jenis Perangkat**	**Spesifikasi**
Perangkat Keras	Prosesor	Intel Core i5 – 7200U

	Memori	8 GB
Perangkat Lunak	Sistem Operasi	Ubuntu 16.04 LTS

Spesifikasi lingkungan pengujian yang berfungsi sebagai mesin kehadiran dapat dilihat pada Tabel 5.3.

Tabel 5.3 Spesifikasi Lingkungan Pengujian – Mesin Kehadiran

Perangkat	**Jenis Perangkat**	**Spesifikasi**
Perangkat Keras	Prosesor	Quad Core 1.2GHz Broadcom BCM2837 64bit CPU
	Wireless LAN	BCM43438 wireless LAN
	Ruang Penyimpanan	16 GB SD Card 98MB/s
	Memori	1 GB
Perangkat Lunak	Sistem Operasi	Ubuntu 16.04.4

5.2. Skenario Pengujian

Pada bagian ini akan dijelaskan tentang skenario pengujian yang dilakukan. Uji coba dilakukan untuk setiap kasus penggunaan, mulai dari “Daftar Ruangan”, “Mengganti IP Pusat”, “Mengganti Hostname Mesin”, “Mengosongkan Tabel”, hingga “Reboot Mesin”. Uji Coba penelitian ini dilakukan dalam dua tahap, yaitu tahap uji coba fungsionalitas dan uji performa.

5.2.1. Uji Fungsionalitas

Pengujian kebutuhan fungsional dilakukan secara mandiri dengan mengacu pada kasus penggunaan yang

sebelumnya telah dijelaskan pada Bab III. Setiap kasus penggunaan memiliki skenarionya masing-masing, antara lain :

1. Skenario 1 – Daftar Ruangan:
    - Mesin kehadiran dalam kondisi awal sudah menyala dan terhubung dengan alamat ip server pusat yang sama dengan sistem kontrol.
    - Menyalakan server pusat sistem kontrol.
    - WebSocket klien dari mesin kehadiran akan mengirimkan data menuju server berupa *socket id* mesin dan alamat ip mesin.
    - WebSocket server menerima pesan dari mesin dan server akan melakukan proses pembuatan *array* dalam variabel "ruanganlist" yang berisi data *socket id* mesin dan alamat ip mesin kehadiran.
    - WebSocket server mengirimkan pesan dari server pusat menuju sistem kontrol mesin kehadiran berupa variabel "ruanganlist".
    - Sistem kontrol akan menggambar tabel baru di halaman utama mesin bagian "List Mesin" yang berisi data mesin kehadiran yang dikirim oleh server.
    - Sistem kontrol akan menggambar tabel baru lagi setiap ada mesin kehadiran yang baru terhubung, begitu pula apabila ada mesin kehadiran yang koneksinya terputus dari server.
    - Apabila mesin kehadiran yang terhubung dengan server memiliki *hostname* atau nama ruangan yang sesuai dengan daftar ruangan, maka ruangan dengan nama yang sama dengan *hostname* mesin kehadiran statusnya akan berubah menjadi warna hijau.
    - Nama ruangan akan tetap memiliki status berwarna merah apabila tidak ada mesin kehadiran yang terkoneksi dengan *hostname* yang sama dengan nama ruangan tersebut.
2. Skenario 2 – Mengganti IP Pusat:
    - Masuk ke halaman dalam tombol "Ganti IP Pusat"
    - Memasukkan alamat ip mesin kehadiran yang akan diganti ip server pusatnya.

- Mesin kehadiran harus dalam kondisi menyala dan berada pada satu jaringan yang sama dengan sistem kontrol ketika melakukan proses pencarian alamat ip mesin kehadiran melalui sistem kontrol.
- WebSocket klien akan melakukan *request* terhadap mesin kehadiran yang kali ini bertindak sebagai server.
- Apabila alamat mesin kehadiran yang dicari berhasil ditemukan, maka sistem kontrol akan memunculkan *kontainer* baru untuk memasukkan ip server pusat yang baru.
- Apabila alamat mesin kehadiran yang dicari tidak berhasil ditemukan, maka sistem kontrol tidak akan memunculkan *kontainer* baru dan akan memberikan keterangan bahwa mesin tidak terkoneksi.
- Apabila mesin kehadiran terputus dari jaringan yang sama dengan sistem kontrol pada tengah-tengah proses memasukkan ip server pusat baru pada *kontainer* yang kedua, maka secara otomatis *kontainer* kedua akan hilang dan sistem kontrol hanya menampilkan *kontainer* awal dengan status mesin tidak terkoneksi.
- Setelah berhasil memasukkan ip server pusat yang baru, mesin kehadiran akan melakukan *reboot* dan kemudian akan terhubung dengan ip server pusat yang baru.
- Sistem kontrol akan menampilkan data mesin kehadiran pada bagian "List Mesin" apabila alamat ip server pusat mesin kehadiran yang baru sesuai dengan alamat server pusat sistem kontrol.

3. Skenario 3 – Mengganti Hostname Mesin:
    - Memilih mesin yang *hostname* atau nama ruanganya akan diganti pada halaman utama "Mesin" pada bagian "List Mesin".
    - Masuk ke halaman dalam tombol "Ganti Alamat".
    - Memilih *hostname* atau nama ruangan baru untuk mesin.

- WebSocket server akan mengirimkan *socket id* dan *hostname* baru dari mesin kehadiran yang *hostname* atau nama ruangannya akan diganti.
- Mesin menerima pesan dari server dan mengganti *hostname*, kemudian melakukan *reboot*.
- WebSocket klien akan mengirimkan info ke server pusat bahwa *hostname* atau nama ruangan mesin kehadiran telah diganti.
- Sistem kontrol kembali ke halaman utama dan muncul notifikasi bahwa mesin mati dan terjadi penggantian *hostname* / nama ruangan.
- Setelah mesin kehadiran kembali menyala dan terhubung ke server, sistem kontrol akan menampilkan tabel yang berisi data mesin dan menampilkan notifikasi bahwa mesin menyala dengan identitas *hostname* atau nama ruangan yang baru.

4. Skenario 4 – Reboot Mesin:
    - Memilih mesin yang akan di-*reboot* pada halaman utama "Mesin" pada bagian "List Mesin".
    - Menekan tombol "Reboot"
    - WebSocket klien akan membawa pesan berupa *socket id* dan perintah untuk melakukan *restart* dari sistem kontrol menuju server pusat yang nanti juga akan diteruskan oleh WebSocket server menuju mesin kehadiran.
    - Mesin kehadiran menerima perintah untuk melakukan *reboot*, kemudian melakukan *reboot*.
    - Tabel mesin kehadiran yang sedang melakukan *reboot* akan hilang ketika mesin mati dan sistem kontrol akan menampilkan pemberitahuan bahwa mesin mati.
    - Ketika mesin kehadiran kembali menyala dan terhubung dengan server, sistem kontrol akan kembali menampilkan tabel yang berisi data mesin tersebut dan menampilkan notifikasi bahwa mesin menyala.
5. Skenario 5 – Mengosongkan Tabel:

- Memilih mesin yang datanya akan dikosongkan pada halaman utama “Mesin” pada bagian “List Mesin”.
- Menekan tombol “Clear Database”.
- WebSocket klien akan membawa pesan berupa *socket id* dan perintah untuk melakukan pengosongan data presensi pada mesin dari sistem kontrol menuju server pusat yang nanti juga akan diteruskan oleh WebSocket server menuju mesin kehadiran.
- Mesin kehadiran menerima perintah untuk mengosongkan data, kemudian melakukan pengosongan *database*.
- Websocket klien akan mengirimkan info ke server pusat bahwa pembersihan *database* pada mesin kehadiran telah selesai.
- Websocket server meneruskan pesan dari mesin kehadiran menuju sistem kontrol.
- Sistem kontrol menerima informasi pembersihan selesai, kemudian memunculkan notifikasi bahwa data presensi pada mesin kehadiran telah dikosongkan.

5.2.2. Uji Performa

Uji performa pada Penelitian ini dilakukan pada fitur “Mengganti IP Pusat”, “Mengganti Hostname Mesin”, “Reboot Mesin” dan “Mengosongkan Tabel”. Uji performa pada penelitian ini dilakukan untuk mengetahui waktu yang diperlukan oleh WebSocket dalam mengirimkan pesan dari klien – sistem kontrol menuju klien – mesin kehadiran melalui server pusat. Waktu pengiriman pesan didapat dari selisih antara *timestamp* ketika pesan dari klien- sistem kontrol tersampaikan ke klien – mesin kehadiran dengan *timestamp* ketika pesan mulai dikirimkan.

Uji performa akan dilakukan dengan melakukan pengambilan data waktu yang diperlukan oleh WebSocket dalam menyampaikan pesan pada masing-masing fitur yang telah disebutkan sebelumnya sejumlah 30 kali percobaan. Performa akan diukur dari rata-rata waktu proses penyampaian pesan yang dilakukan oleh WebSocket pada masing-masing fitur. Selain itu pengujian performa juga dilakukan dengan mengukur waktu yang

dibutuhkan untuk mengirim perintah pada mesin kehadiran dengan cara manual.

5.3. Evaluasi Pengujian

Pada subbab ini akan dipaparkan hasil evaluasi dari pengujian-pengujian yang telah dilakukan pada Penelitian ini. Evaluasi yang dipaparkan adalah evaluasi atas pengujian kebutuhan fungsionalitas dan pengujian performa pada Penelitian ini.

5.3.1. Evaluasi Uji Fungsionalitas

Hasil dari pengujian kebutuhan fungsionalitas pada sistem kontrol mesin kehadiran secara keseluruhan dapat dilihat pada Tabel 5.4, sedangkan pemaparan proses skenario pengujian kebutuhan fungsionalitas akan dijelaskan pada subbab 5.3.1.1 sampai dengan subbab 5.3.1.5.

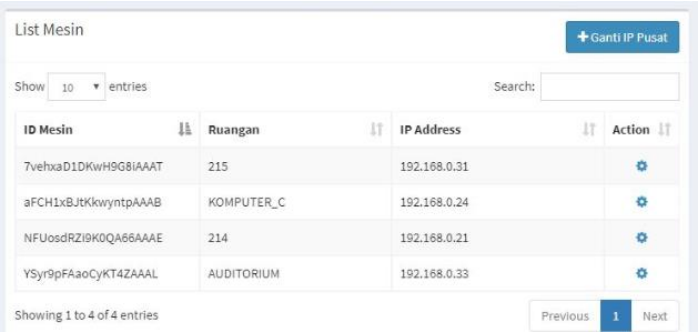
Data yang disajikan pada Tabel 5.4 menjelaskan bahwa semua skenario pengujian kebutuhan fungsionalitas berhasil dan program berjalan dengan baik, sehingga dapat ditarik kesimpulan bahwa kebutuhan fungsionalitas dari sistem kontrol mesin kehadiran bekerja sesuai dengan yang diharapkan.

Tabel 5.4 Evaluasi Pengujian Kebutuhan Fungsionalitas

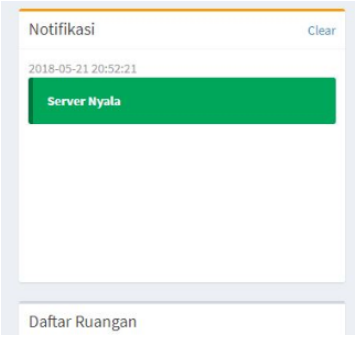
Kode Kasus Penggunaan	Kasus Penggunaan	Hasil
UC001	Daftar Ruangan	Berhasil
UC002	Mengganti IP Pusat	Berhasil
UC003	Mengganti Hostname Mesin	Berhasil
UC004	Reboot Mesin	Berhasil
UC005	Mengosongkan Tabel	Berhasil

5.3.1.1. Evaluasi Skenario 1 – Daftar Ruangan

Gambar 5.1 dan Gambar 5.2 adalah tampilan awal sistem kontrol mesin kehadiran dalam kondisi server menyala dengan 4 mesin kehadiran terhubung dengan server pusat.

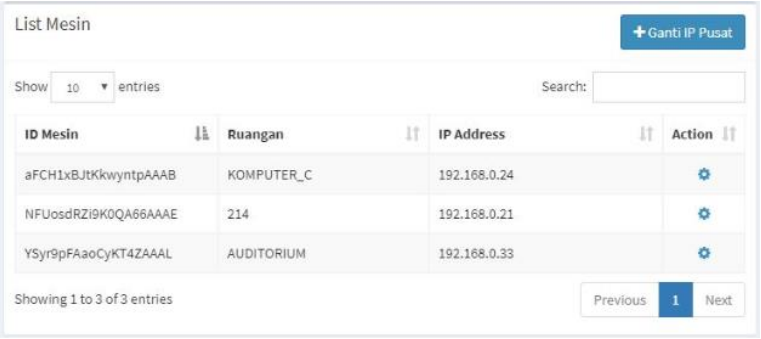


Gambar 5.1 Evaluasi Skenario Daftar Ruangan Langkah 1 - Bagian Kiri

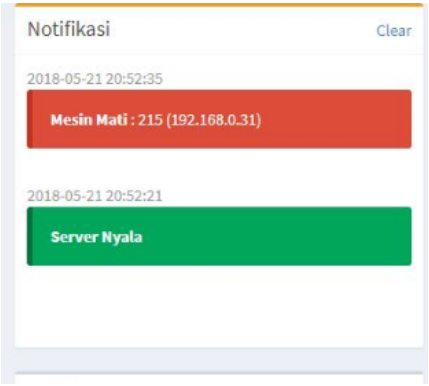


Gambar 5.2 Evaluasi Skenario Daftar Ruangan Langkah 1 - Bagian Kanan

Gambar 5.3 adalah tampilan sistem kontrol mesin kehadiran dalam kondisi server pusat sedang menyala dan satu buah mesin sedang tidak terhubung dengan server pusat, sehingga tabel yang berisi mesin yang koneksinya terputus akan hilang dan akan muncul pemberitahuan bahwa mesin sedang mati, seperti yang ditunjukkan pada Gambar 5.4.



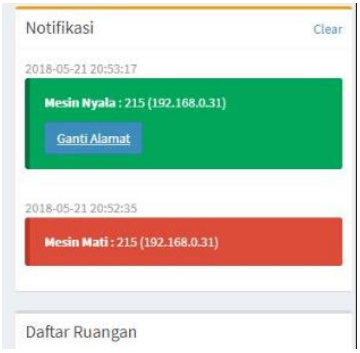
Gambar 5.3 Evaluasi Skenario Daftar Ruangan Langkah 2 – Bagian Kiri



Gambar 5.4 Evaluasi Skenario Daftar Ruangan Langkah 2 – Bagian Kanan

Gambar 5.5 adalah tampilan ketika mesin yang sebelumnya mati, seperti yang ditunjukkan pada Gambar 5.4, terhubung kembali dengan server pusat. Sistem kontrol mesin kehadiran akan menggambar tabel baru yang berisi data dari mesin yang baru saja terhubung dengan server dan menampilkan notifikasi bahwa mesin menyala seperti yang ditunjukkan pada Gambar 5.6.

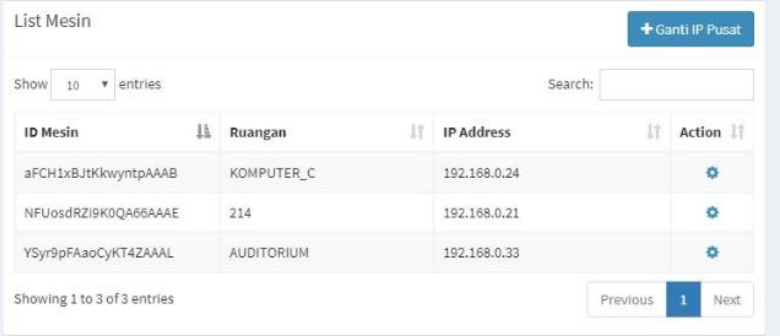
Gambar 5.5 Evaluasi Skenario Daftar Ruangan Langkah 3 – Bagian Kiri



Gambar 5.6 Evaluasi Skenario Daftar Ruangan Langkah 3 - Bagian Kanan

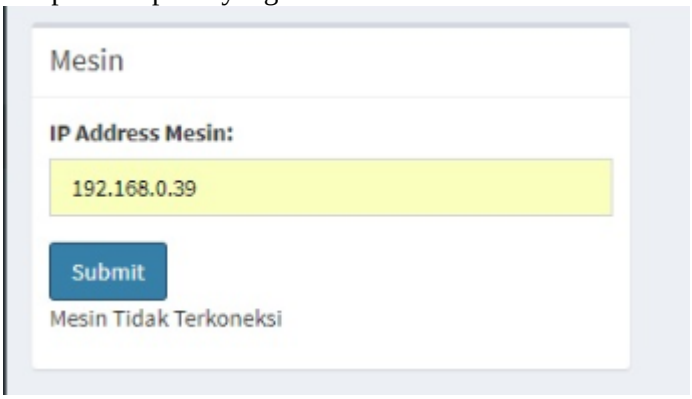
5.3.1.2. Evaluasi Skenario 2 – Mengganti IP Pusat

Gambar 5.7 adalah tampilan awal sistem kontrol mesin kehadiran dengan kondisi server sedang menyala dan 3 mesin kehadiran yang sedang terhubung ke server pusat. Pada pengujian kebutuhan fungsionalitas ini dilakukan penggantian alamat ip server pusat dari mesin kehadiran dengan alamat ip mesin 192.168.0.31 yang sebelumnya belum terhubung pada server.



Gambar 5.7 Evaluasi Skenario Mengganti IP Pusat - Langkah 1

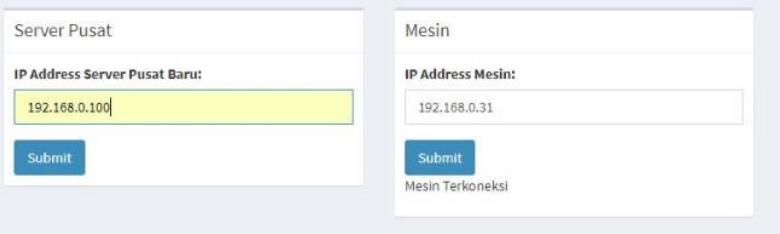
Gambar 5.8 adalah tampilan skenario uji coba mengganti ip pusat dengan kondisi pencarian alamat ip mesin kehadiran yang gagal karena alamat ip mesin kehadiran 192.168.0.39 tidak ada pada satu jaringan yang sama sehingga sistem kontrol tidak memunculkan *kontainer* baru untuk melakukan penggantian alamat ip server pusat yang baru.



Gambar 5.8 Evaluasi Skenario Mengganti IP Pusat - Langkah 2

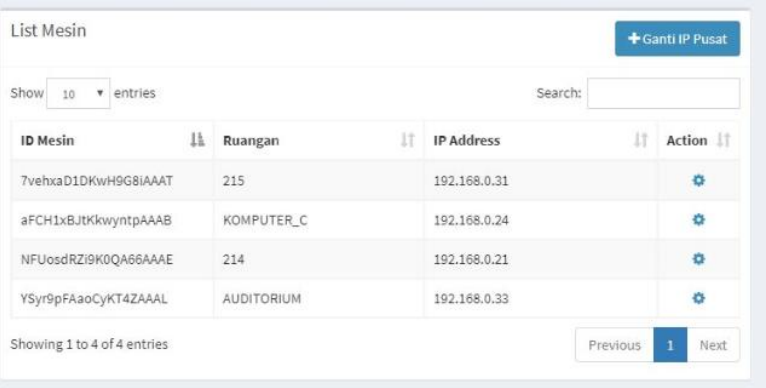
Gambar 5.9 adalah tampilan skenario uji coba mengganti ip pusat dengan kondisi perncarian alamat ip mesin kehadiran yang berhasil, karena mesin kehadiran dengan alamat ip 192.168.0.31 terhubung dalam satu jaringan yang sama dengan sistem kontrol. Seperti yang ditunjukkan pada Gambar 5.9, sistem kontrol akan memunculkan satu *kontainer* baru untuk memasukkan alamat ip

server pusat yang baru pada mesin kehadiran dengan alamat ip 192.168.0.31.

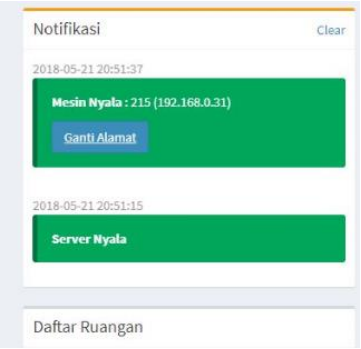


Gambar 5.9 Evaluasi Skenario Mengganti IP Pusat - Langkah 3

Setelah mesin kehadiran dengan alamat ip 192.168.0.31 dihubungkan dengan alamat ip server pusat yang sama dengan sistem kontrol, yaitu 192.168.0.100, maka mesin kehadiran akan melakukan *reboot* dan ketika mesin sudah kembali menyala, mesin akan masuk ke tampilan sistem kontrol karena sudah memiliki alamat server pusat yang sama, seperti yang ditunjukkan pada Gambar 5.10 dan Gambar 5.11.



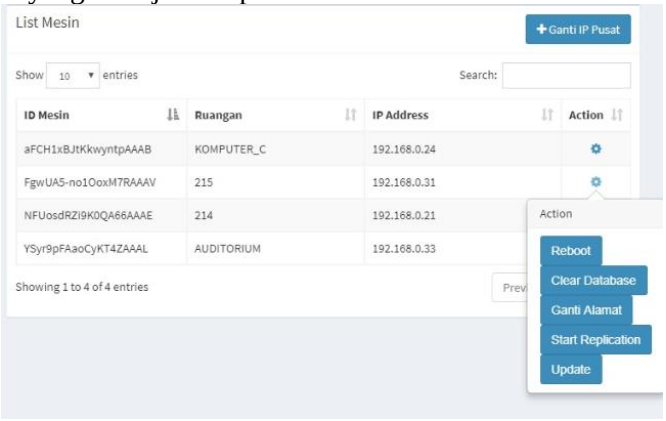
Gambar 5.10 Evaluasi Skenario Mengganti IP Pusat - Langkah 4 – Bagian Kiri



Gambar 5.11 Evaluasi Skenario Mengganti IP Pusat - Langkah 4 - Bagian Kanan

5.3.1.3. Evaluasi Skenario 3 – Mengganti Hostname Mesin

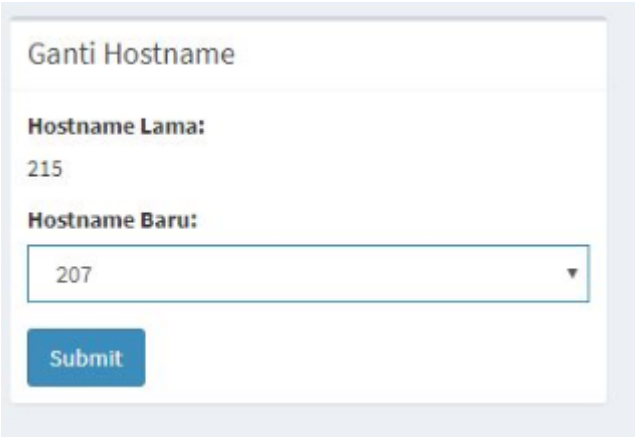
Gambar 5.12 adalah tampilan awal pada sistem kontrol ketika akan melakukan pengujian kebutuhan fungsionalitas mengganti *hostname* atau nama ruangan mesin. Hal yang harus dilakukan pertama kali adalah menekan tombol "Ganti Alamat" pada pilihan tombol atau pada bagian notifikasi mesin menyala seperti yang ditunjukkan pada Gambar 5.13.



Gambar 5.12 Evaluasi Skenario Mengganti Hostname Mesin - Langkah 1 – Bagian Kiri

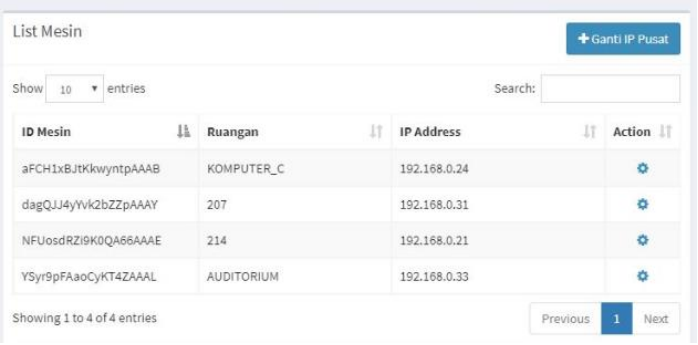
Gambar 5.13 Evaluasi Skenario Mengganti Hostname Mesin - Langkah 1 - Bagian Kanan

Gambar 5.14 menunjukkan tampilan sistem kontrol setelah masuk kedalam tombol "Ganti Alamat". Pada pengujian kali ini dilakukan penggantian *hostname* atau nama ruangan dengan alamat ip mesin 192.168.0.31 dengan *hostname* atau nama ruangan lama "215" dan menggantinya dengan *hostname* baru "207".

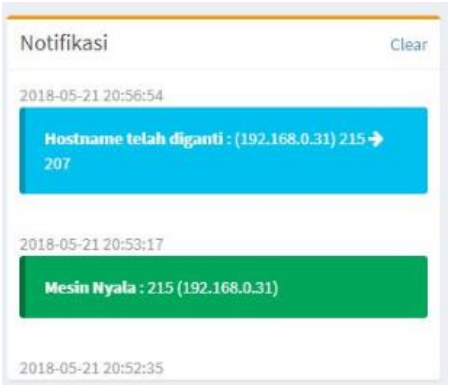


Gambar 5.14 Evaluasi Skenario Mengganti Hostname Mesin - Langkah 2

Gambar 5.16 menunjukkan adanya pemberitahuan bahwa mesin dengan *hostname* (nama ruangan) “215” telah berhasil diganti menjadi “207”, dan nama ruangan “215” sudah tidak ada lagi pada tabel daftar mesin dan digantikan dengan ruangan “207” seperti yang terlihat pada Gambar 5.15.



Gambar 5.15 Evaluasi Skenario Mengganti Hostname Mesin - Langkah 3 – Bagian Kiri

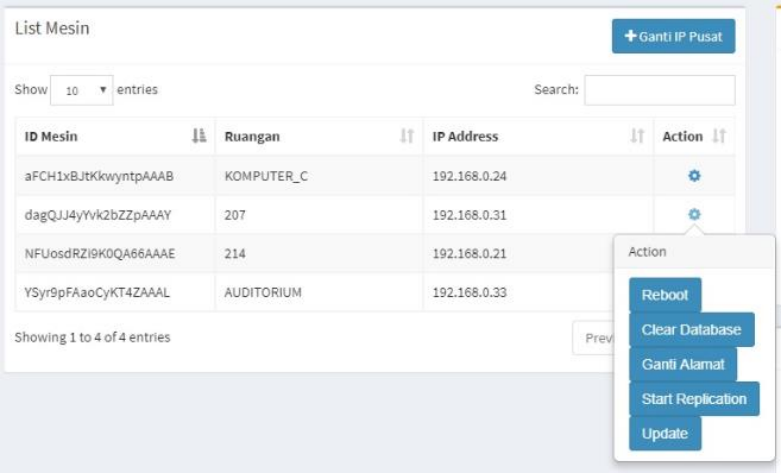


Gambar 5.16 Evaluasi Skenario Mengganti Hostname Mesin - Langkah 3 - Bagian Kanan

5.3.1.4. Evaluasi Skenario 4 – Reboot Mesin

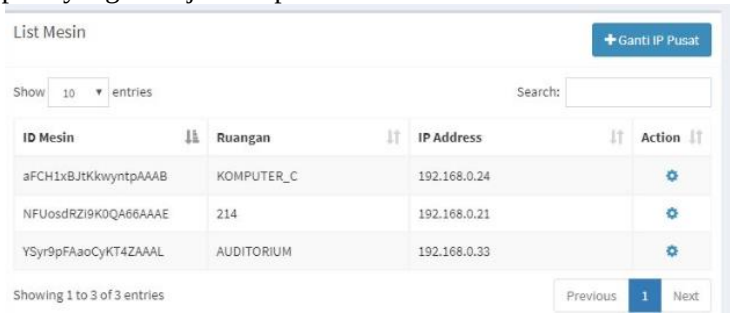
Gambar 5.17 adalah tampilan awal ketika akan melakukan proses *reboot* mesin. Pada percobaan kali ini, mesin

yang akan di-*reboot* adalah mesin dengan alamat ip 192.168.0.31 dan *hostname* (nama ruangan) “207”.

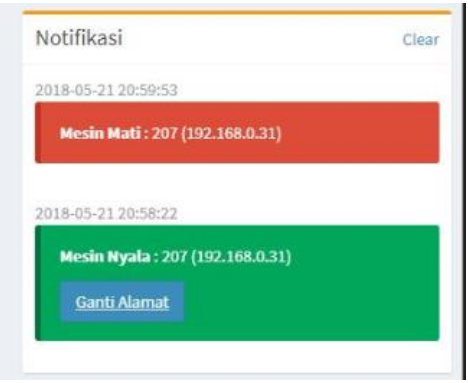


Gambar 5.17 Evaluasi Skenario Reboot Mesin – Langkah 1

Setelah menekan tombol “Reboot” seperti yang ditunjukkan pada Gambar 5.17, mesin kehadiran dengan alamat ip 192.168.0.31 akan melakukan proses *reboot* dan terputus dari server, sehingga sistem kontrol akan memunculkan notifikasi bahwa mesin sedang mati, seperti yang ditunjukkan pada Gambar 5.19. Mesin kehadiran juga akan hilang dari tabel daftar mesin seperti yang ditunjukkan pada Gambar 5.18.

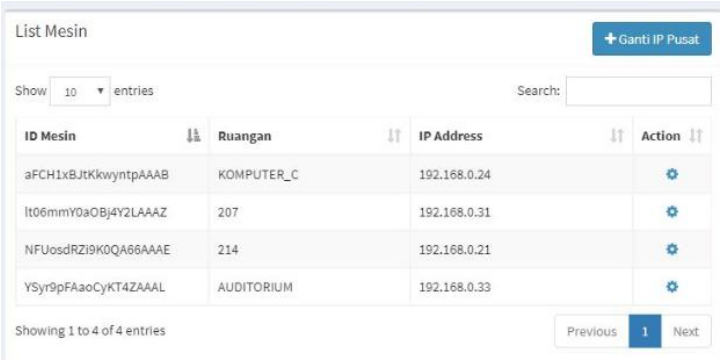


Gambar 5.18 Evaluasi Skenario Reboot Mesin – Langkah 2 – Bagian Kiri



Gambar 5.19 Evaluasi Skenario Reboot Mesin - Langkah 2 - Bagian Kanan

Setelah mesin kehadiran kembali menyala dan terhubung ke server, maka sistem kontrol akan memunculkan notifikasi bahwa mesin kehadiran telah menyala, seperti yang ditunjukkan pada Gambar 5.21. Mesin kehadiran juga akan muncul kembali pada tabel daftar mesin seperti yang ditunjukkan pada Gambar 5.20.

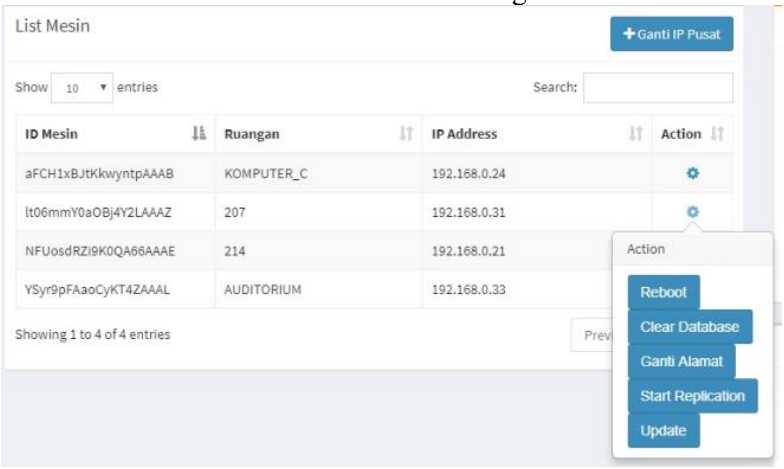


Gambar 5.20 Evaluasi Skenario Reboot Mesin – Langkah 3 – Bagian Kiri

Gambar 5.21 Evaluasi Skenario Reboot Mesin – Langkah 3 - Bagian Kanan

5.3.1.5. Evaluasi Skenario 5 – Mengosongkan Tabel

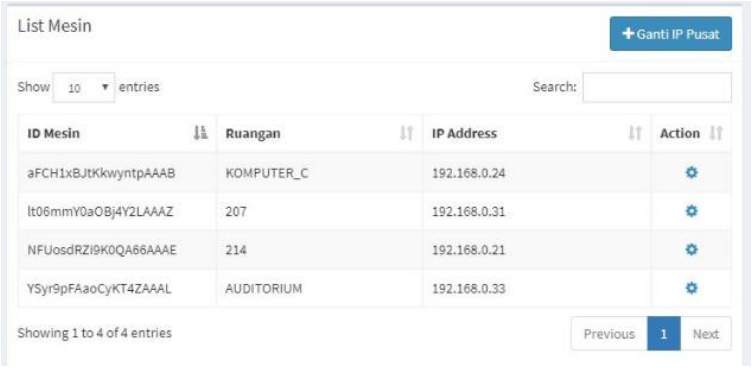
Gambar 5.22 adalah tampilan awal sistem kontrol mesin kehadiran ketika akan melakukan pengosongan *database* pada mesin kehadiran. Pada uji coba fungsionalitas kali ini, mesin yang akan dibersihkan datanya adalah mesin dengan alamat ip 192.168.0.31 dan *hostname* atau nama ruangan "207".



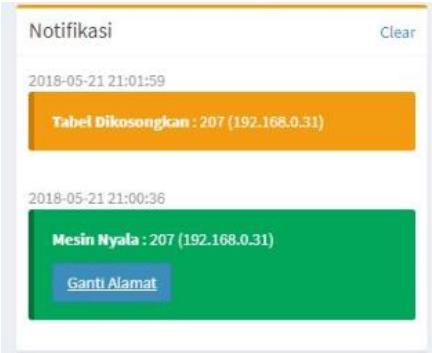
Gambar 5.22 Evaluasi Skenario Mengosongkan Tabel - Langkah 1

Gambar 5.23 dan Gambar 5.24 menunjukkan tampilan sistem kontrol mesin kehadiran setelah melakukan pembersihan

database pada mesin kehadiran "207" dengan alamat ip 192.168.0.31. Sistem kontrol akan memunculkan notifikasi bahwa data presensi pada mesin kehadiran dengan alamat ip 192.168.0.31 telah dibersihkan.



Gambar 5.23 Evaluasi Skenario Mengosongkan Tabel - Langkah 2 – Bagian Kiri



Gambar 5.24 Evaluasi Skenario Mengosongkan Tabel - Langkah 2 - Bagian Kanan

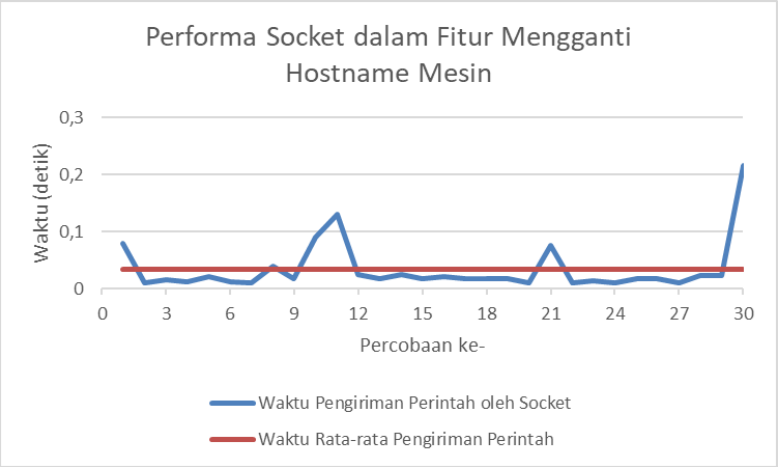
5.3.2. Evaluasi Uji Performa

Pengujian performa pada Penelitian ini dilakukan pada fitur "Mengganti IP Pusat", "Mengganti Hostname Mesin",

"Reboot Mesin" dan "Mengosongkan Tabel" dan akan dijelaskan lebih lanjut pada subbab 5.3.2.1 hingga 5.3.2.4.

5.3.2.1. Uji Performa Mengganti Hostname Mesin

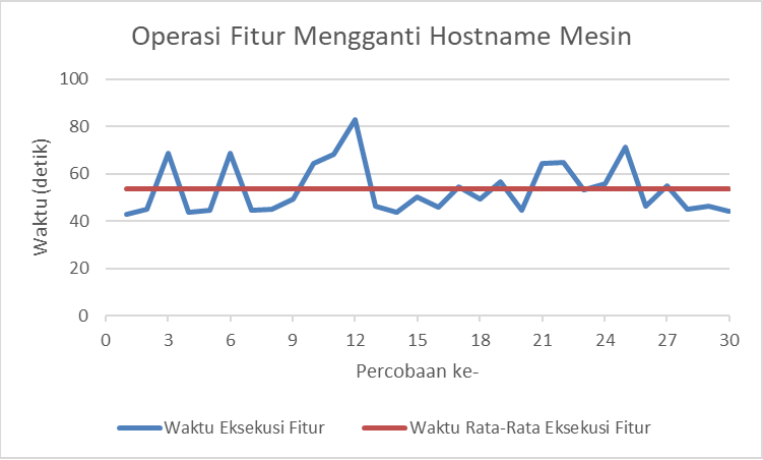
Pengujian performa pada fitur "Mengganti Hostname Mesin" dilakukan dengan mengukur waktu yang dibutuhkan socket untuk mengirimkan pesan perintah mengubah *hostname* atau nama ruangan dari klien – sistem kontrol melalui server pusat hingga pesan tersampaikan pada klien – mesin kehadiran, serta mengukur waktu yang diperlukan selama melakukan operasi penggantian *hostname* hingga muncul pemberitahuan pada sistem kontrol bahwa *hostname* (nama ruangan) mesin telah diganti. Selain itu, pengujian performa pada fitur mengganti *hostname* mesin kehadiran juga dilakukan dengan mengukur waktu yang diperlukan untuk melakukan pengoperasian secara manual dalam memberikan perintah untuk mengganti *hostname* atau nama ruangan mesin.



Gambar 5.25 Grafik Hasil Pengujian Performa Socket dalam Mengirimkan Perintah Ganti Hostname

Gambar 5.25 menunjukkan grafik hasil percobaan mengukur waktu yang dibutuhkan oleh socket dalam mengirimkan perintah untuk mengubah *hostname* (nama ruangan) dari sistem kontrol kepada mesin kehadiran. Dari percobaan yang dilakukan

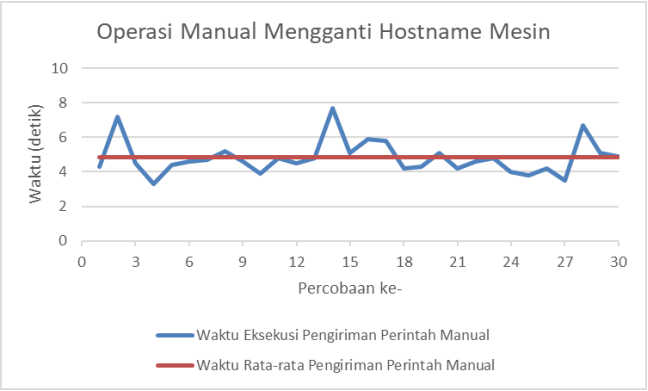
sebanyak 30 kali, rata-rata waktu pengiriman perintah yang didapatkan adalah 0,034 detik.



Gambar 5.26 Grafik Hasil Pengujian Performa Operasi Ganti Hostname Mesin

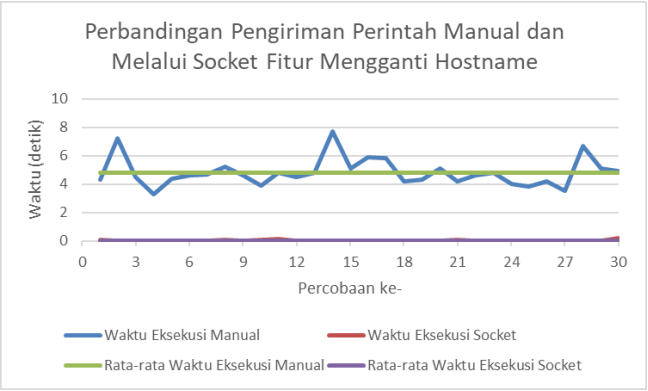
Gambar 5.26 menunjukkan grafik hasil percobaan mengukur waktu yang diperlukan dalam pengoperasian fitur mengganti *hostname* (nama ruangan) mesin kehadiran melalui sistem kontrol mulai dari ketika aktor menekan tombol "Ganti Alamat" hingga muncul pemberitahuan pada sistem kontrol apabila *hostname* (nama ruangan) mesin telah diubah.

Dari 30 kali percobaan yang dilakukan, didapatkan waktu rata-rata dalam pengoperasian fitur mengganti *hostname* mesin sebesar 53,591 detik. Angka tersebut terbilang relatif tinggi mengingat rata-rata waktu yang diperlukan untuk pengiriman perintah mengganti *hostname* hanya 0,034 detik. Hal ini disebabkan oleh adanya proses *reboot* dan pengambilan data yang diperlukan pada mesin kehadiran apabila *hostname* (nama ruangan) yang dituju memiliki jadwal perkuliahan. Proses pengambilan data tersebut merupakan implementasi dari fitur Replikasi Database pada penelitian "Replikasi Sebagian dengan Metode Horizontal Fragmentation pada Basis Data Sistem Kehadiran" oleh Muhammad Hilman.



Gambar 5.27 Grafik Hasil Pengujian Operasi Manual Mengganti Hostname Mesin

Gambar 5.27 menunjukkan grafik hasil percobaan mengukur waktu yang diperlukan dalam pengoperasian fitur mengganti *hostname* (nama ruangan) mesin kehadiran secara manual. Dari 30 kali percobaan yang telah dilakukan, waktu rata-rata yang didapatkan adalah 4,823 detik. Waktu yang didapatkan menunjukkan adanya perbedaan yang signifikan apabila dibandingkan dengan proses pengiriman perintah mengganti *hostname* melalui sistem kontrol yang hanya memakan rata-rata waktu sebesar 0,034 detik.

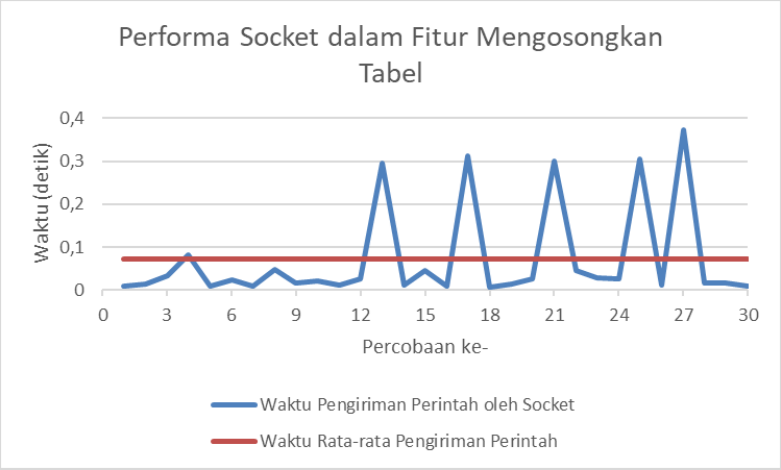


Gambar 5.28 Grafik Perbandingan Pengiriman Perintah Manual dan Socket Fitur Mengganti Hostname

Gambar 5.28 adalah grafik perbandingan antara waktu yang diperlukan dalam pengiriman perintah mengganti *hostname* (nama ruangan) kepada mesin kehadiran secara manual dan pengiriman perintah melalui socket dari sistem kontrol. Pada Gambar 5.28 dapat dilihat bahwa waktu yang diperlukan untuk melakukan pengiriman perintah kepada mesin kehadiran secara manual jauh lebih besar daripada waktu pengiriman perintah kepada mesin kehadiran melalui socket.

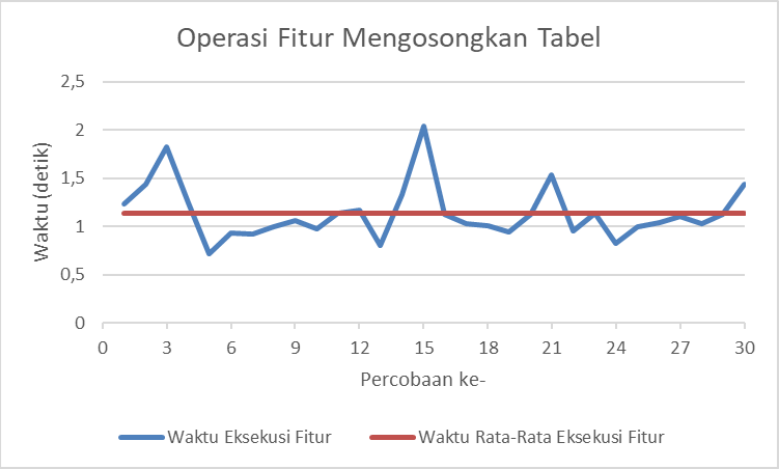
5.3.2.2. Uji Performa Mengosongkan Tabel

Pengujian performa pada fitur "Mengosongkan Tabel" dilakukan dengan mengukur waktu yang dibutuhkan socket untuk mengirimkan pesan berupa perintah untuk membersihkan *database* pada mesin kehadiran dari sistem kontrol, serta mengukur waktu yang diperlukan selama melakukan operasi membersihkan *database* mesin kehadiran hingga muncul pemberitahuan bahwa data pada mesin kehadiran telah dikosongkan. Selain itu, pengujian performa pada fitur mengosongkan tabel juga dilakukan dengan mengukur waktu yang diperlukan untuk melakukan pengoperasian secara manual dalam memberikan perintah untuk mengosongkan tabel data pada mesin kehadiran.



Gambar 5.29 Grafik Hasil Pengujian Performa Socket dalam Mengirimkan Perintah Membersihkan Data pada Mesin Kehadiran

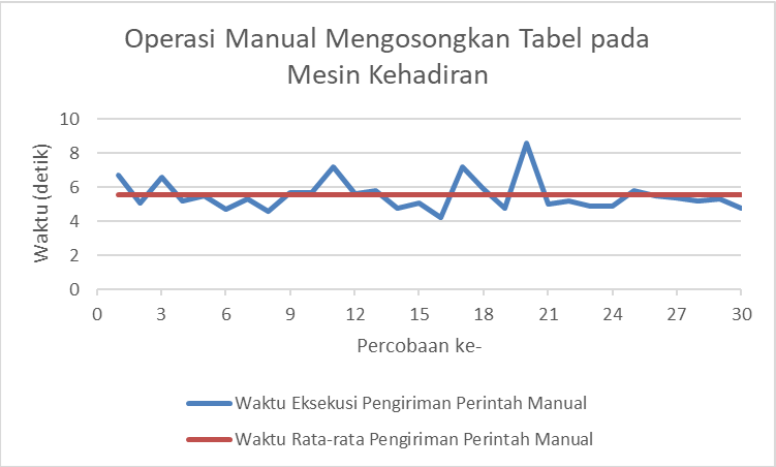
Gambar 5.29 menunjukkan grafik hasil percobaan mengukur waktu yang dibutuhkan oleh socket dalam mengirimkan perintah untuk membersihkan *database* mesin kehadiran dari sistem kontrol kepada mesin kehadiran. Dari percobaan yang dilakukan sebanyak 30 kali, rata-rata waktu pengiriman perintah yang didapatkan adalah 0,072 detik.



Gambar 5.30 Grafik Hasil Pengujian Performa Operasi Membersihkan Database Mesin Kehadiran

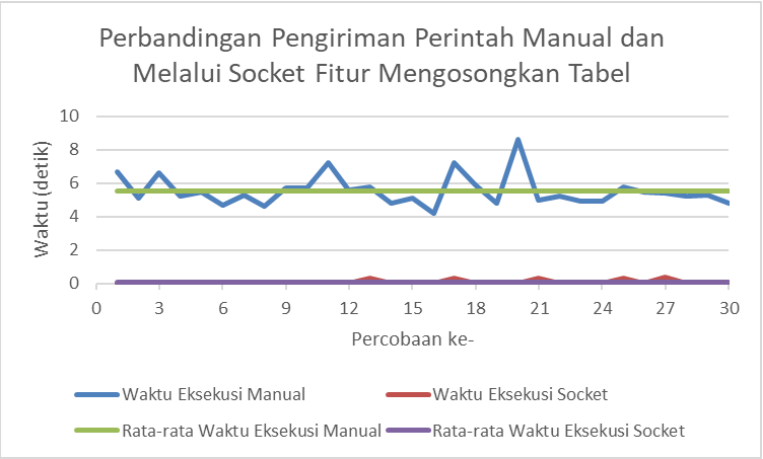
Gambar 5.30 menunjukkan grafik hasil percobaan mengukur waktu yang diperlukan dalam pengoperasian fitur membersihkan *database* mesin kehadiran melalui sistem kontrol mulai dari ketika aktor menekan tombol "Clear Database" hingga muncul pemberitahuan pada sistem kontrol bahwa tabel pada mesin kehadiran telah dikosongkan.

Dari 30 kali percobaan yang dilakukan, didapatkan waktu rata-rata dalam pengoperasian fitur membersihkan *database* mesin kehadiran sebesar 1,139 detik. Angka tersebut jauh lebih kecil apabila dibandingkan dengan rata-rata waktu pengoperasian fitur mengganti *hostname* (nama ruangan) mesin karena pengoperasian fitur membersihkan *database* mesin kehadiran tidak memerlukan adanya aktivitas *reboot* oleh mesin kehadiran sehingga waktu pengoperasian yang diperlukan jauh lebih kecil.



Gambar 5.31 Grafik Hasil Pengujian Operasi Manual Mengosongkan Tabel pada Mesin Kehadiran

Gambar 5.31 menunjukkan grafik hasil percobaan mengukur waktu yang diperlukan dalam pengoperasian fitur mengosongkan *database* pada mesin kehadiran secara manual. Dari 30 kali percobaan yang telah dilakukan, waktu rata-rata yang didapatkan adalah 5,543 detik. Waktu yang didapatkan menunjukkan adanya perbedaan yang signifikan apabila dibandingkan dengan proses pengiriman perintah untuk membersihkan *database* melalui sistem kontrol yang hanya memakan rata-rata waktu sebesar 0,072 detik.

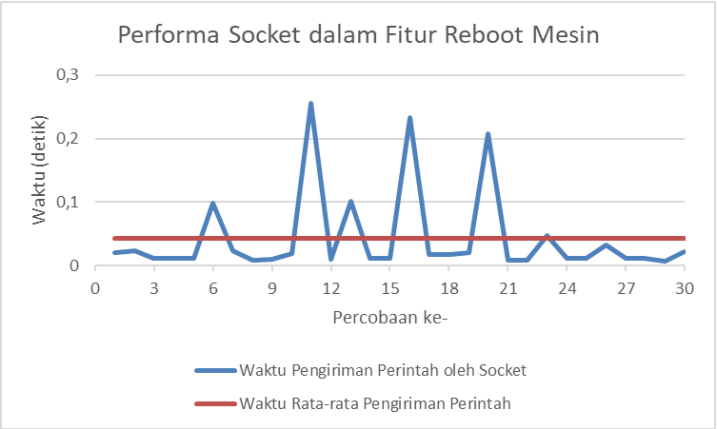


Gambar 5.32 Grafik Perbandingan Pengiriman Perintah Manual dan Socket Fitur Mengosongkan Tabel

Gambar 5.32 adalah grafik perbandingan antara waktu yang diperlukan dalam pengiriman perintah mengosongkan tabel (membersihkan *database*) kepada mesin kehadiran secara manual dan pengiriman perintah melalui socket dari sistem kontrol. Pada Gambar 5.32 dapat dilihat bahwa waktu yang diperlukan untuk melakukan pengiriman perintah kepada mesin kehadiran secara manual jauh lebih besar daripada waktu pengiriman perintah kepada mesin kehadiran melalui socket.

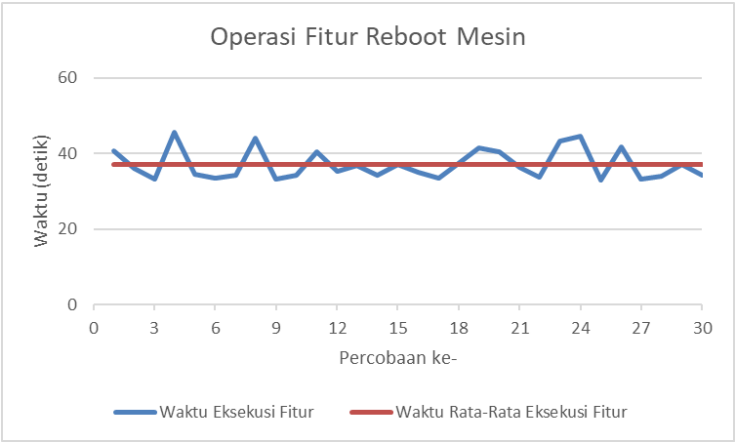
5.3.2.3. Uji Performa Reboot Mesin

Pengujian performa pada fitur "Reboot Mesin" dilakukan dengan mengukur waktu yang dibutuhkan socket untuk mengirimkan pesan berupa perintah untuk *reboot* pada mesin kehadiran dari sistem kontrol, serta mengukur waktu yang diperlukan selama melakukan operasi fitur *reboot* mesin kehadiran hingga muncul pemberitahuan bahwa mesin kembali terhubung dengan sistem kontrol. Selain itu, pengujian performa pada fitur reboot mesin kehadiran juga dilakukan dengan mengukur waktu yang diperlukan untuk melakukan pengoperasian secara manual dalam memberikan perintah untuk *reboot* mesin.



Gambar 5.33 Grafik Hasil Pengujian Performa Socket dalam Mengirimkan Perintah Reboot pada Mesin Kehadiran

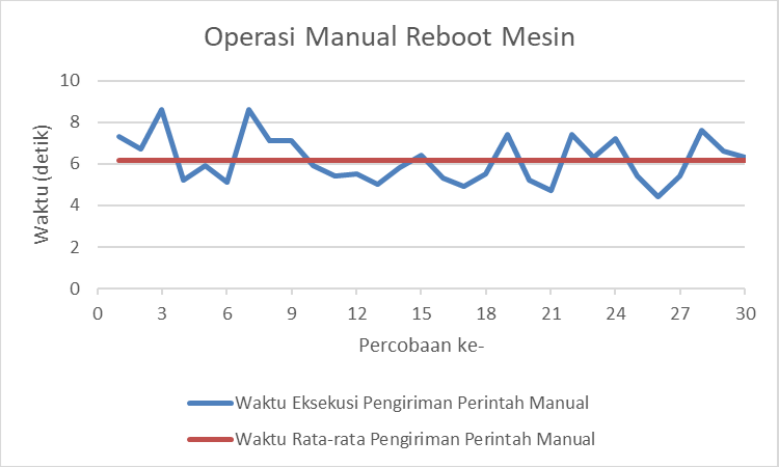
Gambar 5.33 menunjukkan grafik hasil percobaan mengukur waktu yang dibutuhkan oleh socket dalam mengirimkan perintah untuk *reboot* mesin kehadiran dari sistem kontrol kepada mesin kehadiran. Dari percobaan yang dilakukan sebanyak 30 kali, rata-rata waktu pengiriman perintah yang didapatkan adalah 0,0432 detik.



Gambar 5.34 Grafik Hasil Pengujian Performa Operasi Reboot Mesin Kehadiran

Gambar 5.34 menunjukkan grafik hasil percobaan mengukur waktu yang diperlukan dalam pengoperasian fitur *reboot* mesin kehadiran melalui sistem kontrol mulai dari ketika aktor menekan tombol "Reboot" hingga muncul pemberitahuan bahwa mesin kehadiran telah terhubung kembali dengan sistem kontrol.

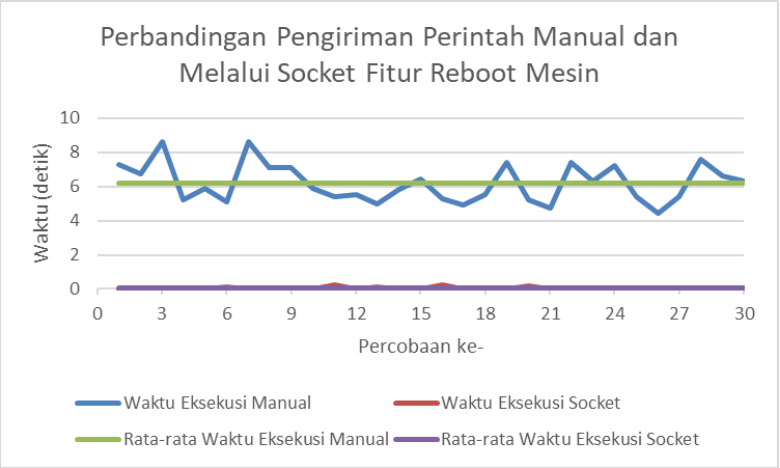
Dari 30 kali percobaan yang dilakukan, didapatkan waktu rata-rata dalam pengoperasian fitur *reboot* mesin sebesar 37,093 detik. Angka tersebut lebih rendah apabila dibandingkan dengan rata-rata waktu pengoperasian fitur mengganti *hostname* (nama ruangan) mesin karena pada pengoperasian fitur *reboot* mesin tidak memerlukan adanya aktivitas replikasi *database* seperti yang ada pada fitur mengganti *hostname* (nama ruangan) mesin, akan tetapi tetap membutuhkan waktu untuk proses *reboot* mesin sampai mesin terhubung kembali dengan sistem kontrol.



Gambar 5.35 Grafik Hasil Pengujian Operasi Manual Reboot Mesin Kehadiran

Gambar 5.35 menunjukkan grafik hasil percobaan mengukur waktu yang diperlukan dalam pengoperasian fitur *reboot* mesin kehadiran secara manual. Dari 30 kali percobaan yang telah dilakukan, waktu rata-rata yang didapatkan adalah 6,173 detik. Waktu yang didapatkan menunjukkan adanya perbedaan

yang signifikan apabila dibandingkan dengan proses pengiriman perintah untuk melakukan *reboot* melalui sistem kontrol yang hanya memakan rata-rata waktu sebesar 0,0432 detik.



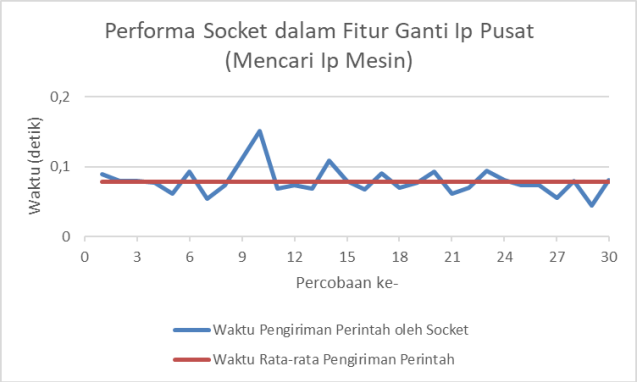
Gambar 5.36 Grafik Perbandingan Pengiriman Perintah Manual dan Socket Fitur Reboot Mesin

Gambar 5.36 adalah grafik perbandingan antara waktu yang diperlukan dalam pengiriman perintah *reboot* kepada mesin kehadiran secara manual dan pengiriman perintah melalui socket dari sistem kontrol. Pada Gambar 5.36 dapat dilihat bahwa waktu yang diperlukan untuk melakukan pengiriman perintah kepada mesin kehadiran secara manual jauh lebih besar daripada waktu pengiriman perintah kepada mesin kehadiran melalui socket.

5.3.2.4. Uji Performa Ganti Ip Pusat

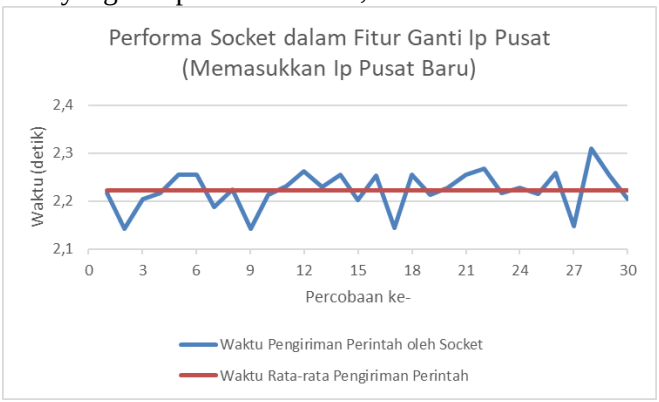
Pengujian performa pada fitur "Ganti Ip Pusat" dilakukan dengan mengukur waktu yang dibutuhkan socket untuk mencari ip mesin kehadiran yang server pusatnya akan diganti, mengirimkan pesan berupa perintah untuk melakukan penggantian ip server pusat pada mesin kehadiran dari sistem kontrol, serta mengukur waktu yang diperlukan selama melakukan operasi fitur mengganti ip server pusat mesin kehadiran hingga muncul pemberitahuan bahwa mesin kembali terhubung dengan sistem kontrol. Selain itu, pengujian performa pada fitur mengganti ip server pusat pada

mesin kehadiran juga dilakukan dengan mengukur waktu yang diperlukan untuk melakukan pengoperasian secara manual dalam memberikan perintah untuk mengganti alamat ip server pusat pada mesin kehadiran.



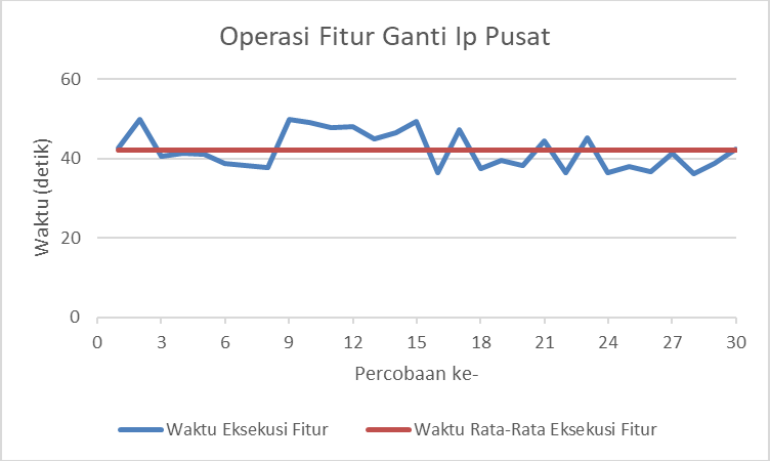
Gambar 5.37 Grafik Hasil Pengujian Performa Socket dalam Mencari Ip Mesin Kehadiran dalam Fitur Ganti Ip Pusat

Gambar 5.37 menunjukkan grafik hasil percobaan mengukur waktu yang dibutuhkan oleh socket dalam mencari alamat ip mesin kehadiran yang server pusatnya akan diganti. Dari percobaan yang dilakukan sebanyak 30 kali, rata-rata waktu pencarian yang didapatkan adalah 0,079 detik.



Gambar 5.38 Grafik Hasil Pengujian Performa Socket dalam Mengirim Perintah untuk Mengganti Ip Server Pusat

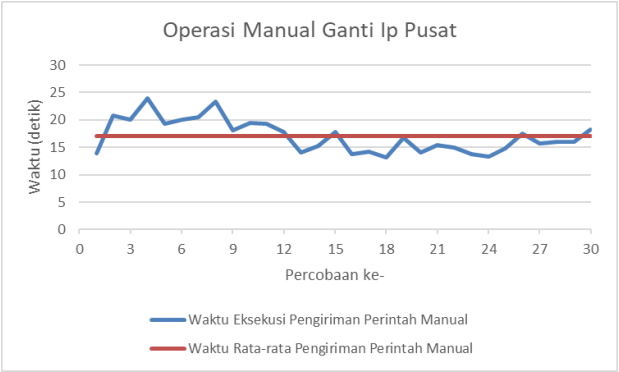
Gambar 5.38 menunjukkan grafik hasil percobaan mengukur waktu yang dibutuhkan oleh socket dalam mencari alamat ip mesin kehadiran yang server pusatnya akan diganti. Dari percobaan yang dilakukan sebanyak 30 kali, rata-rata waktu pencarian yang didapatkan adalah 2,223 detik.



Gambar 5.39 Grafik Hasil Pengujian Performa Operasi Mengganti Ip Server Pusat

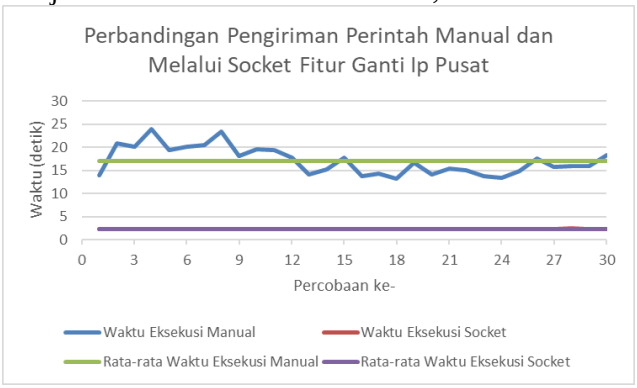
Gambar 5.39 menunjukkan grafik hasil percobaan mengukur waktu yang diperlukan dalam pengoperasian fitur mengganti ip server pusat pada mesin kehadiran melalui sistem kontrol mulai dari ketika aktor menekan tombol “ganti IP Pusat” hingga muncul pemberitahuan bahwa mesin kehadiran telah terhubung kembali dengan sistem kontrol.

Dari 30 kali percobaan yang dilakukan, didapatkan waktu rata-rata dalam pengoperasian fitur mengganti ip server pusat sebesar 42,039 detik.



Gambar 5.40 Grafik Hasil Pengujian Operasi Manual Mengganti Ip Server Pusat pada Mesin

Gambar 5.40 menunjukkan grafik hasil percobaan mengukur waktu yang diperlukan dalam pengoperasian fitur mengganti alamat ip server pusat pada mesin kehadiran secara manual. Dari 30 kali percobaan yang telah dilakukan, waktu rata-rata yang didapatkan adalah 17,04 detik. Waktu yang didapatkan menunjukkan adanya perbedaan yang signifikan apabila dibandingkan dengan jumlah rata-rata waktu yang diperlukan untuk proses pencarian alamat ip mesin dan pengiriman perintah mengganti ip server pusat melalui sistem kontrol yang hanya memakan jumlah rata-rata waktu sebesar 2,307 detik.



Gambar 5.41 Grafik Perbandingan Pengiriman Perintah Manual dan Socket Fitur Ganti Ip Pusat

Gambar 5.41 adalah grafik perbandingan antara waktu yang diperlukan dalam pengiriman perintah mengganti ip pusat kepada mesin kehadiran secara manual dan pengiriman perintah melalui socket dari sistem kontrol. Pada Gambar 5.41 dapat dilihat bahwa waktu yang diperlukan untuk melakukan pengiriman perintah kepada mesin kehadiran secara manual jauh lebih besar daripada waktu pengiriman perintah kepada mesin kehadiran melalui socket.

[Halaman ini sengaja dikosongkan]

BAB VI
KESIMPULAN DAN SARAN

Pada bab ini akan diberikan kesimpulan yang diperoleh selama pengerjaan Penelitian dan saran mengenai pengembangan yang dapat dilakukan terhadap Penelitian ini pada masa yang akan datang.

6.1. Kesimpulan

Dari hasil pengamatan yang dilakukan selama proses perancangan, implementasi dan pengujian yang dilakukan, dapat diambil kesimpulan sebagai berikut:

1. Dari uji coba fungsionalitas pada penelitian ini, yang menunjukkan bahwa semua fitur pada sistem kontrol berfungsi dengan baik, sehingga mempermudah pengelolaan mesin kehadiran.
2. Terdapat perbedaan yang sangat signifikan antara waktu yang dibutuhkan untuk mengirimkan perintah kepada mesin kehadiran dengan cara manual dan dengan melalui socket dari sistem kontrol. Hal ini ditunjukkan dengan adanya selisih dari rata-rata waktu yang diperlukan untuk pengiriman perintah sebesar 4,789 detik pada fitur mengganti hostname mesin, 5,471 detik pada fitur mengosongkan tabel, 6,13 detik pada fitur *reboot* mesin dan 14,733 detik pada fitur mengganti ip pusat. Oleh karena itu, dapat ditarik kesimpulan bahwa adanya sistem kontrol mempercepat penanganan terhadap mesin kehadiran.

6.2. Saran

Berikut merupakan beberapa saran untuk pengembangan sistem pada masa yang akan datang. Saran-saran ini didasarkan pada hasil perancangan, implementasi dan pengujian yang telah dilakukan:

1. Apabila ada perintah yang memerlukan input dari pengguna, maka sebaiknya dilakukan filter terlebih

dahulu, seperti filter terhadap tanda titik koma (;), petik (“ “), dan sebagainya.

2. Menambahkan autentikasi untuk pengoperasian yang memerlukan peranan WebSocket pada sistem untuk menghindari adanya penyalahgunaan sistem dari pihak lain.
3. Menambahkan satu identitas unik pada masing-masing mesin kehadiran yang tidak dapat diubah untuk menghindari terjadinya kebingungan oleh pengguna yang disebabkan oleh perubahan alamat ip mesin kehadiran, nama ruangan (*hostname*), maupun *socket id.*
4. Akan lebih baik apabila dapat ditambahkan metode otomasi pada fitur mengganti *hostname* atau nama ruangan mesin kehadiran dengan memperhatikan jarak antar ruangan, kondisi ruangan yang sedang digunakan atau tidak, serta pencegahan supaya nama ruangan lain tidak ikut terganti apabila ada mesin yang berubah nama ruangannya karena menggantikan peran mesin kehadiran lainnya yang sedang mati.

DAFTAR PUSTAKA

[1] Y. Furukawa, "WEB BASED CONTROL APPLICATION USING WEBSOCKET," *2014 International Conference on Computer Science and Electronic Technology (ICCSET 2014),* Vol. %1 dari %2SPring-8/JASRI, Kouto, Sayo-cho Hyogo, 679-5198, Japan, 2014.

[2] R. V. H. G. d. A. S. A. Muhammad Yusuf, "Rancang Bangun Aplikasi Absensi Perkuliahan Mahasiswa dengan Pengenalan Wajah," *JURNAL TEKNIK ITS* , Vol. %1 dari %2 Vol. 5, No. 2, , no. ISSN: 2337-3539 (2301-9271 Print) , pp. A766-A770, 2016.

[3] S. W. A. H. Adiba Kamalia Putri, "SISTEM PENCATATAN KEHADIRAN MAHASISWA DAN DOSEN SERTA PERHITUNGAN KOMPENSASI DI PROGRAM STUDI TEKNIK TELEKOMUNIKASI POLITEKNIK NEGERI SEMARANG MENGGUNAKAN RFID DAN DIKIRIM MELALUI SMS GATEWAY," *Techno, ISSN 1410 - 8607,* vol. Volume 17 No. 1, pp. 048 - 055, 2016.

[4] Node.js Foundation, "Node.js," [Online]. Available: https://nodejs.org/en/. [Diakses 17 Desember 2017].

[5] F. Laurita, "Building Scalable, Distributed Job Queues with Redis and Ruby," [Online]. Available: https://slideshare.net/francescolaurita/italians-coderjune13redisqueue. [Diakses Desember 2017].

[6] R. Rai, "Socket.IO Real-time Web Application Development," PACKT, Birmingham, 2013.

[7] H. R. Yu, "Design and implementation of web based on Laravel framework," *Proceedings of*

ICALEPCS2011, Vol. %1 dari %2NANKAI UNIVERSITY BINHAI COLLEGE, 300270, 2011.

[8] T. Otwell, "Introduction," 2017. [Online]. Available: https://laravel.com/docs/4.2/introduction. [Diakses 17 Desember 2017].

[9] CABOT, "Using Electron for Cross Platform Desktop Application Development: An Introduction," 16 11 2017. [Online]. Available: https://www.cabotsolutions.com/2017/11/using-electron-for-cross-platform-desktop-application-development-an-introduction. [Diakses 12 12 2017].

[10] R. P. FOUNDATION, "RASPBERRY PI 3 MODEL B," [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/.

[11] Raspberry Pi, "Raspberry Pi 3 Model B," RS Components.

[12] Elechouse, "PN532 NFC RFID moduke V3 kits -- NFC with Android phone," November 2013. [Online]. Available: https://www.elechouse.com/elechouse/index.php?main_page=product_info&cPath=90_93&products_id=2242. [Diakses February 2018].

[13] NXP Semiconductors, "PN532/C1 Near Field Communication (NFC) Controller Data Sheet," 2012 September 20. [Online]. Available: https://www.nxp.com/docs/en/data-sheet/PN532_C1_SDS.pdf. [Diakses 7 November 2012].

[14] A. S. Wardani, "m.liputan6.com," 08 September 2017. [Online]. Available: https://m.liputan6.com/tekno/read/3086054/isi-

kartu-e-toll-bisa-pakai-smartphone-polytron-prime-7. [Diakses April 2018].

[15] Tempo.co, “Pakar: e-KTP Rawan Diretas,” 22 Mei 2013. [Online]. Available: https://tekno.tempo.co/read/482368/pakar-e-ktp-rawan-diretas. [Diakses April 2018].

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS

Haidar Arya Prasetya, lahir pada 5 Juni 1996 di Surabaya. Setelah lulus dari SMAN 5 Surabaya, penulis menempuh pendidikan perguruan tinggi di Institut Teknologi Sepuluh Nopember Surabaya di Departemen Informatika Fakultas Teknologi Informasi dan Komunikasi pada tahun 2014. Penulis memiliki pengalaman menjadi administrator Laboratorium Manajemen Informasi di Departemen Informatika FTIK – ITS. Penulis juga terlibat aktif dalam organisasi kemahasiswaan dan kepanitiaan selama berkuliah, antara lain Himpunan Mahasiswa Teknik Computer-Informatika ITS pada tahun 2015-2018, staff kepanitiaan Schematics pada tahun 2015 dan 2016, serta staff ahli pada kepanitian FTIf Festival tahun 2016. Dalam melakukan pengerjaan Penelitian ini penulis memiliki ketertarikan terhadap bidang Arsitektur Jaringan Komputer (AJK). Untuk menghubungi penulis, dapat menghubungi melalui *email*: haidararya@yahoo.com.