SISTEM INFORMASI PENGUKURAN PRODUKTIVITAS HOTEL DI KABUPATEN JEMBER MENGGUNAKAN METODE OMAX *(OBJECTIVE MATRIX)* DAN AHP *(ANALYTICAL HIERARCHY PROCESS)*

(Studi Kasus Hotel Istana)

SKRIPSI

Oleh

Suci Nur Ramadhani

NIM 122410101034

PROGRAM STUDI SISTEM INFORMASI

UNIVERSITAS JEMBER

2017

SISTEM INFORMASI PENGUKURAN PRODUKTIVITAS HOTEL DI KABUPATEN JEMBER MENGGUNAKAN METODE OMAX *(OBJECTIVE MATRIX)* DAN AHP *(ANALYTICAL HIERARCHY PROCESS)*

(Studi Kasus Hotel Istana)

SKRIPSI

diajukan guna melengkapi tugas akhir dan memenuhi salah satu syarat
untuk menyelesaikan Program Studi Sistem Informasi (S1)
dan mencapai gelar Sarjana Komputer

Oleh

Suci Nur Ramadhani

NIM 122410101034

PROGRAM STUDI SISTEM INFORMASI

UNIVERSITAS JEMBER

2017

PERSEMBAHAN

Skripsi ini saya persembahkan untuk :

1. Allah SWT, Tuhan Yang Maha Pengasih lagi Maha Penyayang yang senantiasa memberikan kemudahan dan kelancaran dalam menyelesaikan skripsi ini;
2. Ayahanda Moh. Mustofa Indra Yulianto dan ibunda Amanatus Soleh Hadlanah yang tiada hentinya memberikan doa dan dukungannya;
3. Saudara-saudaraku beserta seluruh keluarga besar;
4. Sahabatku bersama doa dan dukungannya;
5. Guru-guruku sejak taman kanak-kanak hingga perguruan tinggi yang telah memberikan ilmu dan bimbingan;
6. Almamater Program Studi Sistem Informasi Universitas Jember.

MOTO

"Sesungguhnya sesudah kesulitan itu ada kemudahan. Maka apabila kamu telah selesai (dari sesuatu urusan), kerjakanlah dengan sungguh-sungguh (urusan) yang lain. Dan hanya kepada Tuhanmu lah hendaknya kamu berharap."
(Q.S Al Insyirah : 6-8)[1]

"Banyak kegagalan dalam hidup ini dikarenakan orang-orang tidak menyadari betapa dekatnya mereka dengan keberhasilan saat mereka menyerah."[2]

[1] Kementerian Agama Republik Indonesia. 2014. *Ummul Mukminin : Al Qur'an dan Terjemahannya Untuk Wanita.* Jakarta : OASIS Terrace Recident
[2] Confusius

PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Suci Nur Ramadhani

NIM : 122410101034

menyatakan dengan sesungguhnya bahwa karya tulis ilmiah dengan judul "Sistem Informasi Pengukuran Produktivitas Hotel di Kabupaten Jember Menggunakan Metode OMAX *(Objective Matrix)* Dan AHP *(Analytical Hierarchy Process)*" adalah benar-benar hasil karya sendiri, kecuali kutipan yang sudah saya sebutkan sumbernya, belum pernah diajukan pada institusi manapun, dan bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat dengan sebenarnya, tanpa ada tekanan dan paksaan dari pihak manapun serta bersedia mendapat sanksi akademik jika di kemudian hari pernyataan ini tidak benar.

Jember, 31 Maret 2017

Yang menyatakan,

Suci Nur Ramadhani

NIM 122410101034

SKRIPSI

SISTEM INFORMASI PENGUKURAN PRODUKTIVITAS HOTEL DI KABUPATEN JEMBER MENGGUNAKAN METODE OMAX *(OBJECTIVE MATRIX)* DAN AHP *(ANALYTICAL HIERARCHY PROCESS)* (Studi Kasus Hotel Istana)

Oleh

Suci Nur Ramadhani

NIM 122410101034

Pembimbing :

Dosen Pembimbing Utama : Drs. Antonius Cahya P, M.App., Sc., Ph.D

Dosen Pembimbing Pendamping : Nelly Oktavia Adiwijaya S.Si., MT.

PENGESAHAN PEMBIMBING

Skripsi berjudul "Sistem Informasi Pengukuran Produktivitas Hotel di Kabupaten Jember Menggunakan Metode OMAX *(Objective Matrix)* dan AHP *(Analytical Hierarchy Process)*" telah diuji dan disahkan pada :

hari, tanggal : Jumat, 31 Maret 2017

tempat : Program Studi Sistem Informasi Universitas Jember.

Disetujui oleh:

Pembimbing I,	Pembimbing II,
Drs. Antonius Cahya P, M.App., Sc., Ph.D	Nelly Oktavia Adiwijaya S.Si., MT.
NIP. 196909281993021001	NIP. 198410242009122008

PENGESAHAN PENGUJI

Skripsi berjudul "Sistem Informasi Pengukuran Produktivitas Hotel di Kabupaten Jember Menggunakan Metode OMAX *(Objective Matrix)* dan AHP *(Analytical Hierarchy Process)*" telah diuji dan disahkan pada :

hari, tanggal : Jumat, 31 Maret 2017

tempat : Program Studi Sistem Informasi Universitas Jember.

Tim Penguji,

Penguji I,	Penguji II,
Anang Andrianto, ST., MT	Fahrobby Adnan, S.Kom., M.MSI
NIP. 196906151997021002	NIP. 198706192014041001

Mengesahkan

Ketua Program Studi,

Prof. Drs. Slamin, M.Comp.Sc., Ph.D

NIP. 196704201992011001

RINGKASAN

Sistem Informasi Pengukuran Produktivitas Hotel di Kabupaten Jember Menggunakan Metode OMAX *(Objective Matrix)* dan AHP *(Analytical Hierarchy Process)*; Suci Nur Ramadhani, 122410101034; 2017; 252 halaman; Program Studi Sistem Informasi Universitas Jember.

Perkembangan sektor jasa pelayanan perhotelan pada saat ini cukup berkembang pesat. Meningkatnya jumlah hotel dari tahun ke tahun mengakibatkan ketatnya persaingan yang ditimbulkan dalam membangun citra yang positif sehingga dapat menarik pengunjung untuk memilih hotel tertentu. Kondisi persaingan yang ketat memacu setiap pihak manajemen hotel untuk menemukan solusi yang dapat meningkatkan daya saing. Hotel harus mampu melakukan berbagai program peningkatan manajemen agar dapat meningkatkan daya saing yang kompetitif tanpa harus mengurangi pelayanan kepada pelanggan, dimana kualitas pelayanan tetap menjadi prioritas utama. Agar tujuan dapat tercapai, segenap sumber daya organisasi harus diarahkan untuk fokus memberikan pelayanan yang terbaik kepada setiap pengunjung.

Hotel terdiri dari beberapa departemen yang memiliki peran yang berbeda-beda dan saling melengkapi antara satu sama lain. Setiap departemen memiliki tujuan yang sama untuk meningkatkan kualitas hotel, akan tetapi dalam tugasnya setiap departemen dapat mengalami peningkatan maupun penurunan kinerja, sehingga hal tersebut juga berdampak pada produktivitas hotel. Evaluasi sangat diperlukan ketika suatu departement mengalami penurunan terhadap kinerjanya, oleh karena itu untuk mengetahui departemen apa saja yang menyebabkan penurunan kinerja yang berdampak pada penurunan produktivitas diperlukan suatu teknik pengukuran, sehingga memudahkan pihak manajemen hotel dalam mengontrol produktivitas. Hasil

pengukuran produktivitas dapat dijadikan sebagai acuan untuk mengevaluasi departement tersebut agar dapat meningkatkan kinerjanya.

Sistem informasi pengukuran produktivitas hotel di kabupaten Jember ini menggunakan metode *Objective Matrix* dan AHP (*Analytical Hierarchy Process*) merupakan sebuah solusi mengapa sistem ini sangat cocok diterapkan untuk menghitung produktivitas hotel. Pengembangan sistem ini akan didukung dengan metode OMAX dimana metode ini menggunakan gabungkan kriteria-kriteria produktivitas yang terpadu dan berhubungan satu sama lain. Sistem penilaian ini juga menggunakan metode AHP (*Analytical Hierarcy Process*) untuk menentukan nilai bobot pada indikator pengukuran.

PRAKATA

Puji syukur kepada Allah SWT, atas segala limpahan rahmat, hidayat dan karuniaNya maka penulis dapat menyelesaikan skripsi yang berjudul "Sistem Informasi Pengukuran Produktivitas Hotel di Kabupaten Jember Menggunakan Metode OMAX *(Objective Matrix)* dan AHP *(Analytical Hierarchy Process)*". Skripsi ini disusun untuk memenuhi salah satu syarat menyelesaikan pendidikan Strata Satu (S1) pada Program Studi Sistem Informasi Universitas Jember.

Penyusunan skripsi ini tidak lepas dari dukungan dan bantuan berbagai pihak. Oleh karena itu penulis menyampaikan terima kasih kepada:

1. Allah SWT, yang telah memberikan kesempatan untuk hidup menyelesaikan skripsi ini, dan kekuatan yan diberikan sehingga skripsi ini selesai;
2. Prof. Drs. Slamin, M.Comp.Sc.,Ph.D, selaku Ketua Program Studi Sistem Informasi Universitas Jember;
3. Dr. Saiful Bukhori ST., M.Kom selaku Dosen Pembimbing Akademik yang telah memberikan bimbingan selama penulis menjadi mahasiswa;
4. Drs. Antonius Cahya P, M.App., Sc., Ph.D, selaku Dosen Pembimbing Utama, Nelly Oktavia Adiwijaya S.Si., MT, selaku Dosen Pembimbing Anggota yang telah sabar, memberikan banyak arahan, dan bimbingan dalam penyelesaian skripsi ini;
5. Seluruh dosen dan staf karyawan di Program Studi Sistem Informasi Universitas Jember;
6. Ayahanda Moh. Mustofa Indra Yulianto dan ibunda Amanatus Soleh Hadlanah atas segala kesabaran, keikhlasan, limpahan kasih sayang, dan doa serta perjuangan yang tiada henti hingga saya bisa meraih semua ini;
7. Kakak-kakak dan adik-adik saya tercinta, Devinta Ayu Soraya, Daniar Ayu Pratiwi, Wilda Nur Diana, Daffa Arkan Rifnata, Abyan Arkan Nandana yang

telah mendukung dalam hal moril dan materil serta selalu memberi motivasi dan doa yang tanpa putus pada penulis;

8. Teman-teman Formation (SI-2012) tercinta dan seluruh warga Program Studi Sistem Informasi yang telah menjadi keluarga bagi penulis selama menempuh pendidikan di Program Studi Sistem Informasi;
9. Fitriani, Ratna Suryani, Ayu Mardilianah, Tania Rosifrianti, Qomariatul Hasanah, Dwi Puji Febrina, Vivi Damayanti, Faiqotul Hikmah dan Silvia Herlina, selaku teman seperjuangan dan sahabat terbaik yang selalu memberi dukungan dan semangat.
10. Serta semua pihak yang tidak dapat disebutkan satu per satu yang telah membantu baik tenaga maupun pikiran dalam pelaksanaan kegiatan penelitian dan penyusunan skripsi ini.

Dengan harapan bahwa penelitian ini nantinya akan terus berlanjut dan berkembang kelak, penulis juga menerima segala kritik dan saran dari semua pihak demi kesempurnaan skripsi ini. Akhirnya penulis berharap, semoga skripsi ini dapat bermanfaat.

Jember, 31 Maret 2017

Penulis

DAFTAR ISI

Halaman

DAFTAR TABEL

Halaman

DAFTAR GAMBAR

Halaman

BAB I. PENDAHULUAN

Bab ini merupakan bab awal dari laporan tugas akhir. Pada bab ini akan dibahas tentang latar belakang, perumusan masalah, tujuan dan manfaat, batasan masalah, dan sistematika penulisan.

1.1 Latar Belakang

Perkembangan sektor jasa pelayanan perhotelan pada saat ini cukup berkembang pesat. Meningkatnya jumlah hotel dari tahun ke tahun mengakibatkan ketatnya persaingan yang ditimbulkan dalam membangun citra yang positif sehingga dapat menarik pengunjung untuk memilih hotel tertentu. Kondisi persaingan yang ketat memacu setiap pihak manajemen hotel untuk menemukan solusi yang dapat meningkatkan daya saing. Hotel harus mampu melakukan berbagai program peningkatan manajemen agar dapat meningkatkan daya saing yang kompetitif tanpa harus mengurangi pelayanan kepada pelanggan, dimana kualitas pelayanan tetap menjadi prioritas utama. Agar tujuan dapat tercapai, segenap sumber daya organisasi harus diarahkan untuk fokus memberikan pelayanan yang terbaik kepada setiap pengunjung.

Secara umum hotel yang memiliki tingkat produktivitas lebih tinggi dari pesaingnya memiliki tingkat keuntungan yang lebih tinggi pula, oleh karena itu suatu hotel perlu mempertahankan atau meningkatkan produktivitasnya. Hotel dapat mengontrol atau melihat produktivitasnya dengan cara melakukan pengukuran produktivitas dalam setiap periode. Hasil pengukuran ini akan digunakan sebagai pertimbangan untuk mengevaluasi faktor apa saja yang menyebabkan penurunan ataupun peningkatan produktivitas, sehingga program-program perbaikan dapat dilakukan untuk dapat meningkatkan produktivitas diperiode yang akan datang, sehingga dapat meningkatkan daya saing yang kompetitif dengan hotel-hotel lainnya.

Kabupaten Jember merupakan salah satu dari kabupaten di Jawa Timur yang memiliki banyak sektor pelayanan perhotelan, hal tersebut diungkapkan dalam *website* Pemerintah Kabupaten Jember (www.jemberkab.go.id) bahwa tiga sektor utama yang mempengaruhi pertumbuhan ekonomi di kabupaten Jember adalah pertanian 37,46%, sektor perdagangan, hotel dan restoran 25,17%, serta sektor industri pengolahan 10,81%. Berdasarkan data dari Badan Pusat Statistik Jawa Timur jumlah hotel di kabupaten Jember pada tahun 2014 menempati urutan ke 13 dari 38 kabupaten yang ada di Jawa Timur, hal tersebut menciptakan daya saing antara hotel semakin ketat. Hotel harus mampu menarik minat para pengunjung sehingga akan berdampak pada meningkatnya produktivitas dan mampu bertahan di tengah ketatnya persaingan.

Hotel istana yang menjadi objek studi kasus pada penelitian ini merupakan salah satu hotel non bintang yang berada di kabupaten Jember yang memiliki beberapa departement yaitu *Front Office, Housekeeping, Food & Beverage, Engineering, Marketing* dan *Human Resource Development* (HRD). Setiap departement dapat mengalami peningkatan maupun penurunan kinerja, sehingga hal tersebut juga berdampak pada produktivitas hotel. Evaluasi sangat diperlukan ketika suatu departement mengalami penurunan terhadap kinerjanya, oleh karena itu untuk mengetahui departemen apa saja yang menyebabkan penurunan kinerja yang berdampak pada penurunan produktivitas hotel diperlukan suatu teknik pengukuran. Hasil pengukuran produktivitas dapat dijadikan sebagai acuan untuk mengevaluasi departement tersebut agar dapat meningkatkan kinerjanya. Berdasarkan ulasan mengenai produktivitas diatas, diperlukan suatu sistem informasi pengukuran produktivitas yang dapat mengukur produktivitas pada setiap periode dan dapat mengetahui seberapa baik kinerja dari setiap departement yang ada.

Sistem informasi pengukuran produktivitas hotel ini menggunakan metode OMAX (*Objective Matrix*) dan AHP (*Analytical Hierarchy Process*). Metode OMAX (*Objective Matrix*) digunakan untuk mengukur produktivitas, karena pengukuran produktivitas menggunakan OMAX (*Objective Matrix*) menggunakan gabungkan kriteria-kriteria produktivitas yang terpadu dan berhubungan satu sama lain. Metode

ini melibatkan seluruh divisi dalam suatu organisasi, sedangkan AHP (*Analytical Hierarchy Process*) digunakan untuk menentukan bobot pada indikator pengukuran yang akan digunakan pada salah satu tahap dalam melakukan perhitungan produktivitas menggunakan OMAX (*Objective Matrix*). Melalui penelitian dan perancangan aplikasi ini diharapkan dapat membantu manajer hotel dalam mengukur produktivitas pada setiap departement yang ada.

1.2 Rumusan Masalah

Berdasarkan dari beberapa permasalahan yang telah diuraikan diatas, maka dapat diambil rumusan masalah sebagai berikut :

a. Bagaimana cara mengimplementasikan metode *Objective Matrix* (Omax) dan *Analytical Hierarchy Process* (AHP) untuk mengukur produktivitas hotel?
b. Bagaimana merancang dan membangun sistem informasi pengukuran produktivitas hotel di kabupaten Jember menggunakan metode *Objective Matrix* (Omax) dan *Analytical Hierarchy Process* (AHP)?

1.3 Tujuan dan Manfaat

Berikut merupakan tujuan yang ingin dicapai dan manfaat yang ingin diperoleh dalam penelitian ini.

1.3.1 Tujuan

Adapun tujuan yang ingin dicapai dalam penelitian ini adalah:

a. Mengimplementasikan metode *Objective Matrix* (Omax) dan *Analytical Hierarchy Process* (AHP) untuk mengukur produktivitas hotel.
b. Merancang dan membangun sistem informasi produktivitas hotel di kabupaten Jember menggunakan metode *Objective Matrix* (Omax) dan *Analytical Hierarchy Process* (AHP).

1.3.2 Manfaat

Penelitian ini diharapkan dapat memberikan manfaat sebagai berikut:

a. Bagi Instansi

Sebagai masukan bagi instansi tentang layanan yang akan diberikan kepada *customer* dan dapat digunakan sebagai bahan pertimbangan untuk mengevaluasi kinerja dari suatu departemen berdasarkan pengukuran produktivitas.

b. Bagi Akademis

Hasil penelitian ini diharapkan dapat memberikan kontribusi dan masukan bagi siapa saja yang membutuhkan informasi yang berhubungan dengan judul penelitian ini. Selain itu, hasil penelitian ini merupakan suatu upaya untuk menambah varian judul penelitian yang ada di Program Studi Sistem Informasi Universitas Jember.

c. Bagi Peneliti

Mengetahui bagaimana proses penerapan *Objective Matrix* dan AHP pada sistem informasi pengukuran produktivitas hotel.

d. Bagi pihak lain

Penelitian ini dapat dijadikan bahan referensi bagi peneliti lain yang ingin mengembangkan hasil penelitian ini di kemudian hari.

1.4 Batasan Masalah

Terdapat beberapa batasan masalah yang diangkat sebagai parameter pengerjaan penelitian ini diantaranya sebagai berikut :

a. Sistem informasi pengukuran produktivitas hotel diimplementasikan di hotel Istana Jember.
b. Sistem ini menggunakan metode *Objective Matrix* dan AHP.
c. Sistem dibangun berbasis web dan menggunakan framework CI.

1.5 Sistematika Penulisan

Adapun sistematika penulisan skripsi ini adalah sebagai berikut:

a. Bab 1. Pendahuluan

Bab ini memuat uraian tentang latar belakang, rumusan masalah, tujuan, manfaat, batasan masalah, dan sistematika penulisan skripsi yang masing-masing diuraikan dalam sub bab tersendiri.

b. Bab 2. Tinjauan Pustaka

Bab ini memaparkan tinjauan terhadap hasil-hasil penelitian terdahulu berkaitan dengan masalah yang dibahas, landasan materi dan konsep prediksi, dan kajian teori metode analisis data yang berkaitan dengan masalah dalam penelitian.

c. Bab 3. Metodologi Penelitian

Bab ini menguraikan tentang tempat dan waktu penelitian, metode penelitian, metode pengumpulan data, metode analisis data, dan teknik pengembangan sistem yang digunakan dalam penelitian.

d. Bab 4. Pengembangan Sistem

Bab ini berisi uraian tentang proses yang ditempuh dalam menganalisis dan merancang sistem yang hendak dibangun meliputi desain, kode program, dan pengujian sistem.

e. Bab 5. Hasil dan Pembahasan

Bab ini memaparkan secara rinci pemecahan masalah melalui analisis yang disajikan dalam bentuk deskripsi dibantu dengan ilustrasi berupa tabel dan gambar untuk memperjelas hasil penelitian.

f. Bab 6. Penutup

Bab ini terdiri atas kesimpulan atas penelitian yang telah dilakukan dan saran untuk penelitian selanjutnya

BAB 2. TINJAUAN PUSTAKA

Pada bagian ini dipaparkan tinjauan yang berkaitan dengan masalah yang dibahas, kajian teori yang berkaitan dengan masalah, dan juga penelitian-penelitian terdahulu.

2.1 Penelitian Terdahulu

Penelitian terdahulu yang dilakukan untuk mengukur produktivitas pernah dilakukan pada beberapa penelitian, seperti pada penelitian dengan judul *"Pendekatan Model Objective Matrix-AHP untuk Pengambilan Keputusan Penilaian Kinerja Pelayanan pada Kantor Kelurahan"* (Fiati, 2015). Penelitian ini menjelaskan bagaimana suatu organisasi melakukan pengukuran kinerja terhadap organisasi tersebut agar dapat dijadikan sebagai bahan pertimbangan untuk mengetahui faktor apa saja yang menyebabkan peningkatan maupun penurunan kinerja. Penelitian ini bertujuan untuk membantu para pimpinan dalam mengambil langkah untuk meningkatkan kinerja pelayanan masyarakat pada kantor kelurahan. Metode *objective matrix* membantu pimpinan untuk dapat mengetahui faktor yang mengakibatkan peningkatan maupun penurunan kinerja terhadap pelayanan pada kantor kelurahan. *Objective matrix* yaitu penggabungan beberapa kriteria produktivitas ke dalam sebuah matrik. Setiap kriteria kinerja memiliki bobot sesuai dengan tingkat kepentingannya terhadap tujuan organisasi. Pembobotan dilakukan dengan menggunakan metode *Analytical Hierarchy Process* (AHP), dimana setiap kriteria memiliki tingkat kepentingan yang berbeda dan saling dibandingkan dengan kriteria lainnya. Hasil dari penelitian ini yaitu perencanaan perbaikan kinerja dilakukan agar kinerja pelayanan, jam kerja dan karyawan dapat mencapai sasaran yang telah ditetapkan dan hasil perbaikan digunakan sebagai dasar untuk meningkatkan kinerja pada perusahaan. Hasil dari penghitungan diperoleh pelayanan dan karyawan dengan nilai tertinggi sebesar 840, dan meningkat sebanyak 61,54 %. Hal tersebut menunjukkan pengukuran nilai produktifitas kinerja cukup stabil.

Penelitian lain juga pernah dilakukan untuk mengukur produktivitas dengan judul "*Analisis Produktivitas dengan Metode Objective Matrix (OMAX) di PT. X*" (Agustina & Riana, 2011). Penelitian dilakukan dengan menggunakan metode *objective matrix* untuk mengukur produktivitas dan AHP untuk pemberian bobot pada setiap kriteria yang ada. Penelitian tersebut dilakukan untuk mengukur produktivitas dari PT. X yang merupakan pabrik gula yang dimiliki oleh pemerintah. Hasil produksi pabrik pada tahun 2010 mengalami penurunan dibanding tahun-tahun sebelumnya. Oleh karena itu pengukuran produktivitas dilakukan agar dapat dijadikan sebagai bahan pertimbangan untuk mengetahui faktor-faktor apa yang menyebabkan penurunan produktivitas dan usulan rencana apa yang harus dilakukan agar produktivitas perusahaan mengalami peningkatan untuk masa yang akan datang.

2.2 *Objective Matrix* (OMAX)

Objective matrix diciptakan pertama kali oleh James L. Riggs dari Oregen University seorang ahli produktivitas di Amerika Serikat pada tahun 1975. *Objective matrix* merupakan suatu sistem pengukuran produktivitas parsial yang dikembangkan untuk memantau produktivitas pada setiap bagian perusahaan dengan kriteria produktivitas yang sesuai dengan keberadaan bagian tersebut. Metode ini memiliki konsep pengukuran yang menggabungkan beberapa kriteria kinerja kelompok kerja kedalam sebuah matrik. Masing-masing kriteria memiliki bobot sesuai dengan tingkat kepentingannya terhadap suatu organisasi.

Objective matrix pada penelitian ini digunakan untuk menghitung produktivitas hotel pada setiap periode yang telah ditentukan. Menurut (Silalahi, 2014), langkah-langkah implementasi metode *objective matrix* adalah sebagai berikut :

2.2.1 Penentuan Kriteria Kinerja

Penentuan kriteria kinerja merupakan tahap awal untuk dapat melakukan perhitungan produktivitas. Penentuan kriteria ditetapkan berdasarkan hasil analis

terhadap faktor yang dapat mempengaruhi produktivitas sehingga dapat menyebabkan kenaikan maupun penurunan produktivitas.

2.2.2 Perhitungan rasio produktivitas

Kriteria-kriteria yang telah ditentukan sebelumnya diubah kedalam bentuk rasio produktivitas, rasio produktivitas merupakan perbandingan antar kinerja yang telah dilakukan dengan hasil yang diperoleh dari kinerja tersebut. Untuk dapat menghitung nilai rasio maka diperlukan nilai data yang aktual dari kriteria-kriteria yang telah ditentukan yang diperoleh dari instansi bersangkutan selama beberapa periode.

2.2.3 Menghitung kenaikan setiap level

Nilai kenaikan setiap level dalam tabel OMAX dimulai dari level 0 sampai dengan level 10 sehingga terdapat 11 tingkatan untuk setiap kriteria. Level 0 merupakan level terendah, level 3 merupakan dasar atau nilai awal pada saat pengukuran pertama kali dilakukan, sedangkan level 10 merupakan level tertinggi dari hasil pengukuran produktivitas. Nilai kenaikan setiap level di gunakan sebagai acuan untuk pemberian skor yang diperoleh sesuai dengan pencapaian setiap rasio yang dihitung.

2.2.4 Pembobotan

Pemberian bobot dilakukan karena setiap kriteria memiliki tingkat pengaruh yang yang berbeda-beda dalam perhitungan produktivitas. Pada penelitian ini pemberian nilai bobot dilakukan dengan menggunakan metode *Analytic Hierarchy Process* (AHP) Pada penelitian ini pembobotan dilakukan oleh pakar yaitu manager hotel Istana.

2.2.5 Menghitung indeks produktivitas

Indeks produktivitas merupakan hasil perbandingan antara periode yang diukur (periode saat ini) dengan periode sebelumnya. Tujuan dari perhitungan indeks produktivitas adalah untuk mengetahui apakah produktivitas mengalami peningkatan atau penurunan. Untuk menghitung indeks produktifitas yaitu membandingkan antara periode yang diukur dengan periode sebelumnya (Fiati, 2015).

$$IP = \frac{\text{Current} - \text{Previous}}{\text{Previous}} \text{ X } 100\% \quad \ldots\ldots\ldots\ldots [1]$$

2.3 *Analytic Hierarchy Process* (AHP)

Metode AHP dikembangkan oleh Thomas L. Saaty, dari Universitas Pittsburg. Metode AHP dapat memecah masalah kompleks ke dalam kelompok-kelompok yang kemudian diatur menjadi suatu bentuk hirarki sehingga permasalahan akan tampak lebih terstruktur dan sistematis.

AHP pada penelitian ini digunakan untuk menghitung bobot dari setiap kriteria yang menjadi aspek pengukuran. Menurut Juliyanti e*t al* (2011), langkah-langkah implementasi metode AHP adalah sebagai berikut:

2.3.1 Penyusunan Hirarki (*Decomposition*)

Proses penyusunan hirarki merupakan proses dimana permasalahan yang ada akan disusun menjadi model hirarki yaitu dengan memecah suatu persoalan menjadi elemen-elemen yang terpisah. Membuat struktur hierarki yang diawali dengan tujuan utama. Setelah menyusun tujuan utama sebagai level teratas akan disusun level hirarki yang berada di bawahnya yaitu kriteria-kriteria yang cocok untuk mempertimbangkan atau menilai alternatif yang ada. Setiap kriteria mempunyai intensitas yang berbeda-beda.

2.3.2 Penyusunan dan Penetapan Prioritas (*Synthesis of Priority)*

Penyusunan dan penetapan prioritas merupakan tahapan yang dilakukan untuk memperoleh suatu nilai prioritas dari kriteria-kriteria yang digunakan dalam mendukung tercapainya tujuan. Berikut merupakan tahapan-tahapan yang dilakukan dalam menentukan nilai prioritas pada setiap kriteria.

2.3.2.1 Perbandingan Penilaian (*Comparative Judgement)*

Perbandingan Penilaian yaitu membuat suatu penilaian tentang kepentingan antara dua kriteria yang kemudian disajikan dalam bentuk matrik perbandingan berpasangan.

Untuk dapat membuat matrik perbandingan berpasangan maka harus dapat menentukan diantara dua kriteria yang dianggap paling penting, selain itu juga menentukan seberapa penting kriteria satu dengan kriteria lainnya

Seluruh prioritas yang ada dibandingkan satu sama lain secara berpasangan dan diberi bobot berupa skala dari 1 sampai dengan 9. Skala perbandingan tingkat kepentingan pada metode AHP dapat dilihat pada Tabel 2.1

Tabel 2. 1 Skala Perbandingan Tingkat Kepentingan

Perbandingan nilai relative antar kriteria A dan kriteria B	**Definisi Penilaian**	**Keterangan**
1	Sama penting	Dua kriteria (A dan B) memiliki tingkat kepentingan yang sama dalam memenuhi tujuan
3	Sedikit lebih penting	Kriteria A sedikit lebih penting dibandingkan kriteria B dalam memenuhi tujuan
5	Lebih penting	Kriteria A memiliki tingkat kepentingan yang cukup besar dibandingkan kriteria B dalam memenuhi tujuan
7	Sangat penting	Kriteria A memiliki tingkat kepentingan sangat besar dibandingkan kriteria B dalam memenuhi tujuan
9	Jauh lebih penting	Kriteria A memiliki tingkat kepentingan jauh lebih besar dibandingkan kriteria B dalam memenuhi tujuan
2,4,6,8	Nilai antara	Penilaian diantara relatif yang lainnya
Kebalikan	Jika aktivitas A mendapatkan satu angka dibandingkan aktivitas B, maka B memiliki nilai kebalikannya dibandingkan dengan A.	

Sumber: (Saaty, 1994)

Tahapan yang dilakukan setelah matrik perbandingan berpasangan selesai dibuat yaitu menetapkan prioritas yang dilakukan dengan metode *eigen vector* dan *eigen value*. Nilai dari *eigen vector* yang diperoleh dapat ditentukan *local priority* (prioritas

untuk satu level). Kemudian *global priority* diperoleh dengan mengalikan prioritas elemen pada level di atasnya sampai pada level terakhir.

2.3.2.2 Konsistensi Logis *(Logical Consistency*)

Metode AHP memiliki perbedaan asumsi dengan metode pengambilan keputusan yang lain yaitu tidak adanya syarat konsistensi mutlak dimana metode ini menggunakan persepsi manusia sebagai masukan. Keterbatasan manusia tentu saja dapat menyebabkan munculnya nilai yang tidak konsisten dalam menyatakan persepsinya, apalagi jika membandingkan banyak elemen. Konsistensi menunjukkan intensitas relasi antar elemen didasarkan pada suatu kriteria tertentu.

Index konsistensi (CI) dapat diperoleh dari

$$\text{CI} = \frac{\lambda maks - jumlaj\ kriteria\ (n)}{jumlah\ kriteria\ (n-1)} \quad \ldots\ldots\ldots\ldots [2]$$

$\lambda maks : nilai\ eigen\ maksimum\ dari\ bobot\ \ prioritas\ yang\ diperoleh$

Rasio konsistensi (CR) dapat diperoleh dari

$$\text{CR} = \frac{CI}{CR}\ , (RI = Random\ Index) \quad \ldots\ldots\ldots\ldots [3]$$

Daftar indeks random konsistensi (IR) bisa dilihat pada tabel 2.2 berikut ini :

Tabel 2. 2 Daftar indeks random konsistensi (IR)

Ukuran Matriks	Nilai IR
1,2	0.00
3	0.58
4	0.90
5	1.12
6	1.24
7	1.32
8	1.41
9	1.45

Ukuran Matriks	Nilai IR
10	1.49
11	1.51
12	1.48
13	1.56
14	1.57
15	1.59

Apabila nilai konsistensi yang didapat pada perhitungan rasio konsistensi (CR) kurang dari 0,1 maka nilai prioritas dapat dikatakan konsisten, tetapi apabila nilai rasio konsistensi (CR) lebih dari 0,1 maka penilaian dari data perbandingan harus diperbaiki.

BAB 3. METODOLOGI PENELITIAN

Bab ini menjelaskan tentang metode yang digunakan dalam merancang dan membangun sistem menggunakan metode pengembangan.

3.1 Jenis Penelitian

Jenis penelitian yang digunakan pada penelitian ini merupakan jenis penelitian pengembangan, karena tujuan penelitian adalah untuk membangun sebuah sistem informasi pengukuran produktivitas hotel. Penelitian pengembangan bertujuan untuk membuat dan mengembangkan suatu produk yang efektif untuk digunakan. Penelitian ini bukan jenis penelitian yang ditunjukkan untuk menemukan teori atau menguji kebenaran dari suatu teori dalam bentuk eksperimentasi.

3.2 Tempat dan Waktu Penelitian

Penelitian dilakukan di hotel Istana Jember yang beralamat di Jalan Diponegoro No. 43 Jember. Waktu penelitian dilakukan selama 4 (empat) bulan, dimulai pada bulan Mei 2016 sampai dengan bulan September 2016.

3.3 Tahapan Penelitian

Tahapan penelitian pengembangan sistem mengadopsi dari model *waterfall* dengan tahapan-tahapan seperti pada gambar 3.2.

3.3.1 Tahapan Analisis Kebutuhan

Analisis kebutuhan merupakan tahap untuk mengumpulkan data, informasi, serta mencari kebutuhan fungsional dan non fungsional sistem. Pada tahap ini, peneliti mencari permasalahan yang ada untuk dapat dianalisis kebutuhan yang diperlukan, sebagai solusi dari permasalahan yang muncul. Data-data yang telah didapat kemudian dikelompokkan menjadi kebutuhan fungsional dan non-fungsional sistem. Metode yang digunakan adalah sebagai berikut:

a. Metode Pengumpulan Data

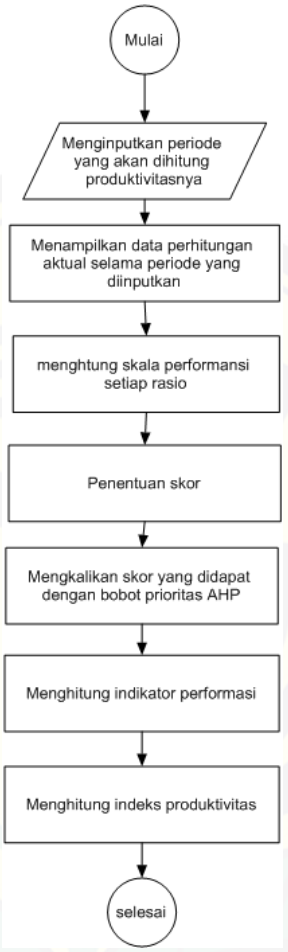
Pengumpulan data dilakukan untuk memperoleh informasi yang dibutuhkan dalam mencapai tujuan penelitian. Pengumpulan data pada penelitian ini dilakukan melalui beberapa teknik pengumpulan data yaitu:

1. Melakukan wawancara kepada manager hotel Istana Jember untuk mendapatkan data dan informasi mengenai pengukuran produktivitas.
2. Melakukan studi literature, jurnal, media, maupun internet tentang pembangunan sistem informasi pengukuran produktivitas hotel.

Jenis data yang digunakan dalam penelitian adalah data primer dan data sekunder. Data primer merupakan data yang diperoleh melalui metode wawancara. Data sekunder merupakan data yang diperoleh dari studi literatur. Pengumpulan data dan informasi yang dilakukan antara lain dimensi kriteria pengukuran produktivitas hotel.

b. Metode Analisis Data

Tahap analisis dimulai dengan menelaah data secara keseluruhan yang telah diperoleh dari tahap pengumpulan data. Data-data tersebut merupakan data kriteria yang memiliki pengaruh terhadap produktivitas hotel. Penerapan metode AHP dan *objective martrix* pada sistem informasi pengukuran produktivitas hotel dapat dilihat pada gambar Gambar 3.1



Gambar 3. 1 Diagram Alir Penerapan Metode *Objective Matrix* dan AHP

3.3.2 Tahapan Desain Sistem

Tahap ini meliputi tahap desain sistem dengan pembuatan diagram menggunakan tools *UML Visual Paradigm*. Diagram-diagram tersebut akan digunakan sebagai acuan pembuatan sistem pada tahap implementasi kebutuhan sistem. Konsep program menggunakan *Object-Oriented Programming* (OOP). Pada bagian desain, pengerjaan yang dilakukan diantaranya adalah:

a. *Business Process*

Business Process dalam perancangan sistem ini digunakan untuk menggambarkan seluruh proses yang dibutuhkan hingga proses goal tercapai.

b. *Use Case Diagram*

Use Case Diagram pada penelitian ini digunakan untuk menggambarkan fitur dan aktor yang mampu mengakses fitur-fitur pada sistem yang akan dibangun. *Use case* ini menggambarkan tugas user dalam sistem yang dirancang berdasarkan hak akses yang dimiliki *user*.

c. *Use Case Scenario*

Use case scenario merupakan penjelasan alur sistem sesuai dengan yang tertera pada *use case diagram*, skenario juga menjelaskan reaksi yang akan terjadi pada sistem setelah menerima perlakuan dari aktor. *Use case scenario* menggambarkan keadaan normal saat setiap aktor mengakses aplikasi dan keadaan alternatif yang terjadi pada kondisi tertentu.

d. *Sequence Diagram*

Sequence diagram merupakan diagram yang digunakan untuk menggambarkan rangkaian interaksi yang dikirim antar object dan interaksi antar object. Interaksi yang terjadi di dalam sistem sesuai dengan urutan dijalankannya sistem tersebut.

e. *Activity Diagram*

Activity diagram menggambarkan alur aktivitas dalam sistem yang sedang dirancang, meliputi awal alur dimulai, *decision* yang terjadi, dan bagaimana alur berakhir yang dapat dilakukan oleh masing-masing aktor. *Activity diagram* juga menggambarkan aktivitas sistem setelah menerima perlakuan dari aktor tersebut.

f. *Class Diagram*

Class diagram merupakan sebuah spesifikasi yang menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class diagram* menggambarkan kelas, *function*, dan atribut yang dibutuhkan oleh seorang programmer dalam membangun sistem.

g. *Entity Relationship Diagram* (ERD)

Entity relationship diagram merupakan suatu model untuk menjelaskan hubungan antar data dalam basis data berdasarkan objek-objek dasar data yang

mempunyai hubungan antar relasi.

3.3.3 Tahapan Implementasi Sistem

Pada tahap ini desain yang telah dibuat akan diimplementasikan menjadi sebuah sistem. hal-hal yang dilakukan pada tahap ini adalah menulis kode program dengan menggunakan bahasa pemrograman PHP, sedangkan manajemen basis data yang digunakan yaitu DBMs My SQL

3.3.4 Tahapan Pengujian Sistem

Pengujian digunakan untuk mengetahui sejauh mana sistem ini dapat berjalan. Testing berfungsi untuk mengetahui apakah sistem ini dapat berfungsi dengan baik sesuai dengan yang diharapkan. Serta untuk mengetahui letak kekurangan yang ada pada sistem ini. Terdapat dua metode yang digunakan untuk pengujian ini yakni :

a. *White box testing*

White box testing merupakan cara pengujian dengan melihat modul untuk yang telah dibuat dengan program-program yang ada. Pengujian ini dilakukan oleh *developer* pembuat program. Jika ada modul yang menghasilkan *output* yang tidak sesuai, maka baris-baris program, variabel dan parameter yang terlibat pada unit tersebut satu persatu akan di cek dan diperbaiki, kemudian di *compile* ulang. (Pressman, 2001)

b. *Black box testing*

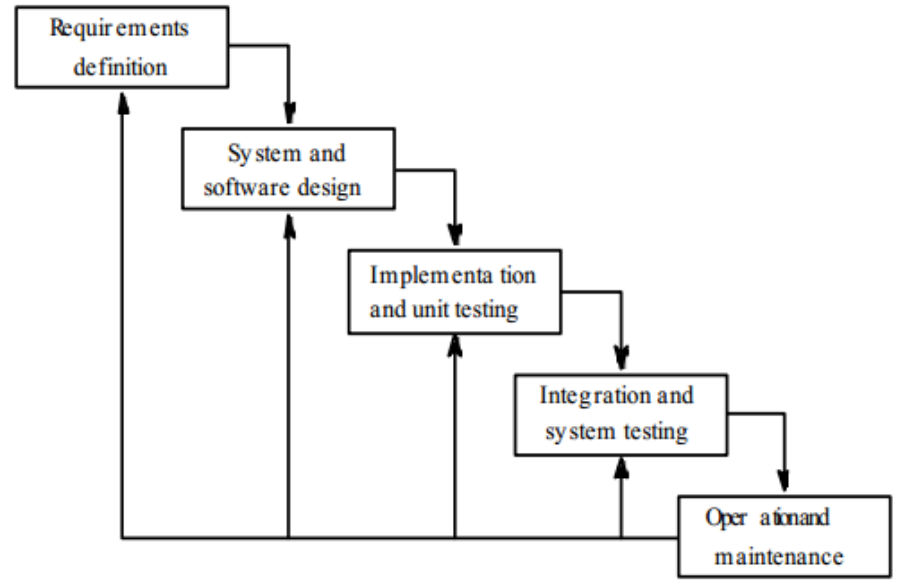
Berbeda dengan *white box testing*, *Black box testing* merupakan metode pengujian perangkat lunak yang memeriksa fungsionalitas dari aplikasi yang berkaitan dengan struktur *internal* atau kerja. Metode ini memfokuskan pada keperluan fungsionalitas dari *software* (Pressman, 2001).

3.4.5 Tahapan Pemeliharaan Sistem

Pemeliharaan merupakan proses perawatan sistem setelah sistem digunakan oleh *user*. Pemeliharaan dilakukan dengan mengecek kinerja sistem secara berkala.

Pengecekan dilakukan apakah kinerja sistem masih berjalan dengan dan melakukan perbaikan apabila terdapat kerusakan.

Pada Gambar 3.2 merupakan gambaran tahapan penelitian pengembangan sistem menggunakan model *waterfall* seperti yang telah dijelaskan diatas.



Gambar 3. 2 Model *Waterfall (Sommerville, 2001)*

BAB 4. PENGEMBANGAN SISTEM

Bab ini menguraikan mengenai analisis kebutuhan, desain, implementasi, dan pengujian sistem yang digunakan dalam proses pengembangan sistem informasi pengukuran produktivitas hotel di kabupaten Jember menggunakan metode OMAX (*objective matrix*) dan AHP (*analytic hierarchy process*). Dimana tahapan analisis hingga pengujian yang dilakukan sesuai dengan metode pengembangan *waterfall*.

4.1 Deskripsi Umum Sistem

Deskripsi umum dari Sistem Informasi Pengukuran Produktivitas Hotel di Kabupaten Jember Menggunakan Metode OMAX (*Objective Matrix*) dan AHP (*analytic hierarchy process*) yang dibangun dalam penelitian ini akan dijelaskan lebih detail pada SOP (*statement of purpose*) sistem dan fungsi sistem.

4.1.1 SOP (*Statement of purpose*)

Sistem informasi pengukuran produktivitas hotel di kabupaten Jember yang akan dibangun dalam penelitian ini adalah sistem yang digunakan untuk mengukur produktivitas hotel. Sistem ini akan menghitung indeks produktivitas yang diperoleh dalam pengukuran yang dilakukan selama periode tertentu, selain itu sistem juga memberikan penilaian terhadap kinerja dari setiap departemen yang ada pada hotel, sehingga dapat diketahui faktor apa yang menyebabkan kenaikan atau penurunan produktivitas berdasarkan kinerja departemen selama periode tersebut. Rasio yang digunakan adalah data yang terdapat pada setiap departemen yang dapat mempengaruhi produktivitas, seperti pada departemen *marketing* yaitu rasio antara jumlah kamar yang dihuni dan total kamar tersedia, departemen HRD yaitu rasio antara jumlah absen karyawan dan jumlah karyawan, departemen *food & beverage* (F&B) yaitu rasio antara jumlah porsi *food & beverage* dan jumlah pengadaan bahan baku, departemen *housekeeping* yaitu rasio antara jumlah kamar yang dibersihkan dan

jumlah hari perawatan kamar, departemen *engineering* yaitu rasio antara jumlah perbaikan oleh *engineering* dan total kerusakan sarana. Sistem ini memiliki tujuh hak akses yaitu manager, HRD, *front office*, *Food & Beverage, housekeeping, engineering* dan *customer*. Hasil pengukuran diperoleh dari data hasil rasio pada setiap departemen yang didapat dalam periode tertentu kemudian diolah menggunakan metode *objective matrix*. Setiap departemen memiliki kriteria yang berbeda untuk dapat dijadikan sebagai indikator pengukuran yang berpengaruh terhadap produktivitas hotel. Setiap kriteria memiliki bobot prioritas yang berbeda-beda sesuai dengan tingkat kepentingan yang telah ditentukan. Kriteria-kriteria tersebut dianalisis menggunakan metode AHP untuk menentukan nilai bobot, sehingga dalam sistem terdapat menu proses AHP. Sesuai dengan tujuan dibangunnya sistem pengukuran produktivitas hotel ini menampilkan hasil pengukuran berupa indeks produktivitas dan melihat faktor yang menyebabkan meningkat maupun menurunnya produktivitas sehingga dapat memudahkan pihak manajemen hotel untuk mengevaluasi kinerja dari setiap departemen.

4.1.2 Fungsi sistem

Fungsi sistem dari sistem yang dibangun dalam penelitian ini terletak pada fitur login yang dapat menentukan hak akses dari setiap pengguna dari sistem ini sendiri. Ketika pengguna melakukan login, maka sistem akan melakukan autentifikasi *username* dan *password* dari pengguna. Selanjutnya sistem akan menyajikan tampilan sistem yang sesuai dengan hak akses dari pengguna yang meliputi :

a. Manager
b. HRD
c. *Front Office*
d. *Food & Beverage* (F&B)
e. *Housekeeping*
f. *Engineering*
g. *Customer*

4.2 Pengumpulan Data

Pengumpulan data pada penelitian ini dilakukan melalui wawancara langsung dengan pihak hotel yakni manager hotel Istana yang bertempat di Jalan Diponegoro No. 43 Jember.

4.3 Analisis Kebutuhan

Tahap analisis kebutuhan sistem merupakan tahapan yang sangat penting dalam pengembangan sebuah sistem informasi. Seluruh kebutuhan penggunaan didefinisikan pada tahap kebutuhan fungsional dan kebutuhan non-fungsional. Hasil analisis tersebut sangat mempengaruhi fungsionalitas sistem yang dibangun untuk dapat digunakan sesuai dengan fungsi dan kebutuhan pengguna.

4.3.1 Kebutuhan Fungsional

Kebutuhan fungsional sistem berisi fitur-fitur inti yang harus dipenuhi dalam sistem agar sistem mampu difungsikan sesuai dengan tujuan dan kebutuhan pengguna terhadap sistem itu sendiri. Kebutuhan fungsional dari sistem informasi pengukuran produktivitas hotel dengan menggunakan metode *objective matrix* dan AHP yaitu:

a. Sistem mampu menghitung dan menyimpan hasil proses perhitungan pengukuran produktivitas hotel
b. Sistem dapat menampilkan hasil pengukuran produktivitas hotel dalam setiap periode
c. Sistem mampu mengelola pembobotan AHP pada data kriteria yang menjadi indikator pengukuran (*update*, *view*)
d. Sistem mampu mengelola data pengguna (view, *input*, edit, *delete*)
e. Sistem mampu mengelola data pegawai (view, *input*, edit, *delete*)
f. Sistem mampu mengelola data kamar (view, *input*, edit, *delete*)
g. Sistem mampu mengelola data komplain (view, *input*)
h. Sistem mampu mengelola data reservasi (view, *input*, edit, *delete*)

4.3.2 Kebutuhan Non-Fungsional

Kebutuhan non-fungsional merupakan hal yang dibutuhkan oleh sistem untuk mendukung aktivitas sistem sesuai dengan kebutuhan fungsional yang telah disusun. Kebutuhan non-fungsional dari sistem informasi pengukuran produktivitas hotel antara lain:

a. Sistem bekerja sesuai dengan fungsinya dan dapat dijalankan bersamaan pada semua komputer dan *browser* yang berbeda.
b. Sistem mudah dimengerti oleh pengguna untuk memberikan kenyamanan pemakaian dan memudahkan pengoperasian.
c. Sistem menggunakan *username* dan *password* untuk autentifikasi akses pengguna terhadap sistem.

4.4 Desain Sistem

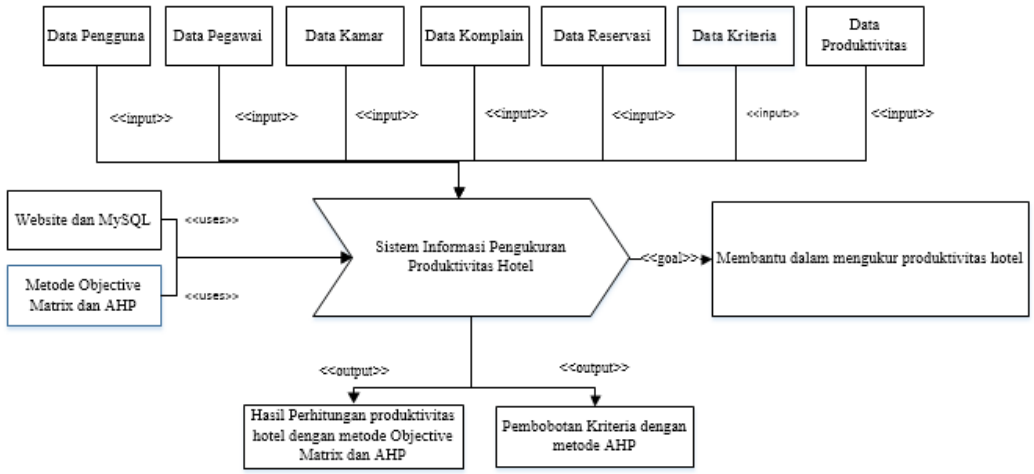
Tahapan yang dilakukan setelah melakukan analisis kebutuhan sistem yaitu tahap perencanaan pembangunan sistem yang dapat digambarkan dengan desain sistem. Desain sistem yang akan dibuat pada sistem informasi pengukuran produktivitas hotel ini meliputi *business process*, *use case diagram*, *scenario*, *activity diagram*, *sequence diagram*, *class diagram*, dan *entity relationship diagram* (ERD).

4.4.1. *Business Process*

Selain dapat dideskripsikan dalam sebuah SOP (*Statement of Purpose*), gambaran umum sistem informasi pengukuran produktivitas hotel menggunakan metode *objective matrix* dan AHP dapat digambarkan melalui sebuah *business process*. Seperti yang dapat kita lihat pada Gambar 4.1 menggambarkan data-data yang digunakan sebagai masukan, data keluaran, *uses* sistem yang dibangun, hingga *goal* dari dibangunnya sistem sendiri.

Business Process merupakan model atau diagram yang menggambarkan sebuah proses lengkap dengan *resources* dan information yang dibutuhkan, *event* yang mendorong terjadinya proses dan *goal* yang dituju. Pada *business process* terdapat

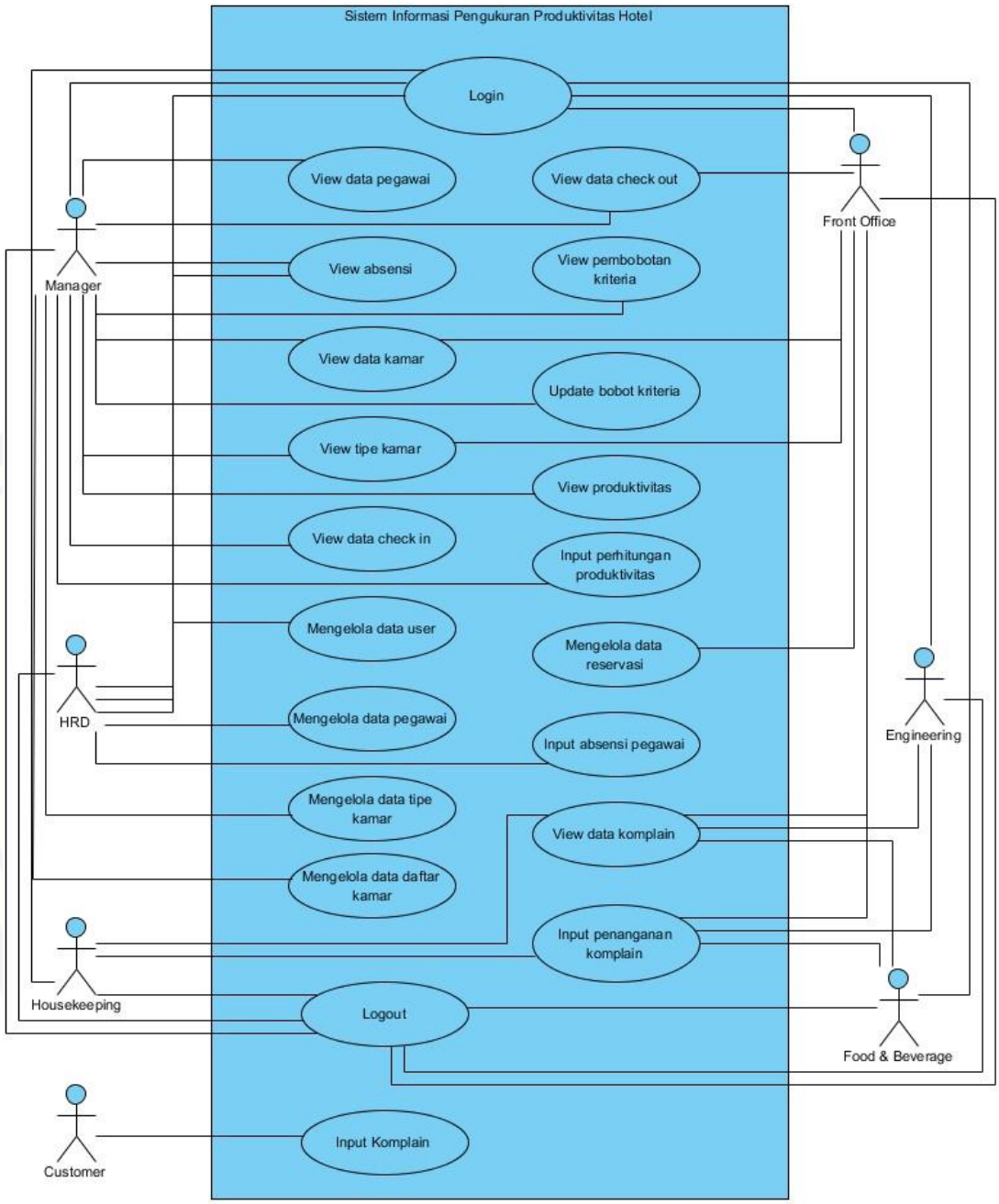
beberapa komponen meliputi masukan (*input*), keluaran (*output*), media yang digunakan (*uses*) dan juga tujuan (*goal*) yang akan dicapai. Pada gambar 4.1 dijelaskan *input*, *output*, *uses*, dan *goal* sistem informasi pengukuran produktivitas hotel.



Gambar 4. 1 *Business Process* Sistem

4.4.2. *Use Case* Diagram

Use case diagram merupakan pemodelan yang dibuat untuk dapat menggambarkan interaksi antara aktor dengan sistem yang akan dibangun. Melalui *use case* diagram dapat diketahui interaksi yang dapat dilakukan aktor terhadap sistem sesuai dengan hak akses yang dimiliki oleh masing masing aktor atau pengguna. Pada Gambar 4.2 digambarkan *use case* diagram sistem informasi pengukuran produktivitas hotel yang terdiri atas dua puluh satu *use case* dan tujuh aktor yaitu manager, HRD, *front office*, *food & beverage*, *housekeeping*, *engineering* dan *customer*.



Gambar 4. 2 *Use Case* Diagram

Berdasarkan use case diagram pada Gambar 4.2 terdapat tujuh aktor atau pengguna, yaitu manager, hrd, *front office, food & beverage, housekeeping,*

engineering dan *customer*. Adapun deskripsi dari masing-masing aktor dapat dilihat pada Tabel 4.1.

Tabel 4. 1 Deskripsi pembagian aktor sistem

No.	Aktor	Deskripsi
1	Manager	Manager merupakan aktor yang memiliki hak akses pada fitur *login*, data pegawai, data kamar, data komplain, data reservasi, data indikator pengukuran, perhitungan produktivitas dan data hasil pengukuran produktivitas.
2	HRD	HRD merupakan aktor yang memiliki hak akses pada fitur *login*, mengelola data pengguna, data pegawai, data absensi.
3	*Front office*	*Front office* merupakan aktor yang memiliki hak akses pada fitur *login*, mengelola data reservasi, melihat data kamar dan data komplain departemen *front office.*
4	*Food & Beverage*	*Food & Beverage* merupakan aktor yang memiliki hak akses pada fitur *login*, dan data komplain departemen *food & beverage.*
5	*Housekeeping*	*Housekeeping* merupakan aktor yang memiliki hak akses pada fitur *login*, dan data komplain departemen *housekeeping.*
6	*Engineering*	*Engineering* merupakan aktor yang memiliki hak akses pada fitur *login*, dan data komplain departemen *engineering.*
7	*Customer*	*Customer* merupakan aktor yang memiliki hak akses pada fitur input komplain

Selain memiliki tujuh aktor, dalam *use case* diagram juga terdapat dua puluh satu *use case*. Deskripsi dari *use case* tersebut dapat dilihat pada Tabel 4.2.

Tabel 4. 2 Deskripsi *use case* sistem

No.	*Use case*	Deskripsi
1	*Login*	*Login* merupakan *use case* yang berfungsi untuk memverifikasi hak akses setiap aktor.
2	*View* data pegawai	*View* data pegawai merupakan *use case* yang terdapat dalam hak akses manager. Fitur tersebut berfungsi untuk menampilkan data pegawai.

3	*View* absensi	*View* absensi merupakan *use case* yang terdapat dalam hak akses manager dan HRD. Fitur tersebut berfungsi untuk menampilkan data absensi pegawai.
4	*View* data kamar	*View* data kamar merupakan *use case* yang terdapat dalam hak akses manager dan *front office*. Fitur tersebut berfungsi untuk menampilkan data tipe kamar.
5	*View* data tipe kamar	*View* data tipe kamar merupakan *use case* yang terdapat dalam hak akses manager dan *front office*. Fitur tersebut berfungsi untuk menampilkan data kamar.
6	*View* data *check in*	*View* data *check in* merupakan *use case* yang terdapat dalam hak akses manager. Fitur tersebut berfungsi untuk menampilkan data *check in*.
7	Mengelola data *user*	Mengelola data *user* merupakan *use case* yang terdapat dalam hak akses HRD yang memiliki tiga fitur, yaitu tambah, *view* dan verifikasi. Fitur tambah berfungsi untuk menambah data *user* sistem. Fitur *view* berfungsi untuk menampilkan data *user*. Fitur verifikasi berfungsi untuk memverifikasi *user* yang baru mendaftar.
8	Mengelola data pegawai	Mengelola data pegawai merupakan *use case* yang terdapat dalam hak akses HRD yang memiliki empat fitur, yaitu tambah, *view* dan *edit* dan *delete*. Fitur tambah berfungsi untuk menambah data pegawai. Fitur *view* berfungsi untuk menampilkan detail data pegawai. Fitur *edit* berfungsi untuk mengubah data pegawai, Fitur *delete* berfungsi untuk menghapus data pegawai.
9	Mengelola data tipe kamar	Mengelola data tipe kamar merupakan *use case* yang terdapat dalam hak akses manager yang memiliki empat fitur, yaitu tambah, *view* dan *edit* dan *delete*. Fitur tambah berfungsi untuk menambah data tipe kamar. Fitur *view* berfungsi untuk menampilkan detail data tipe kamar. Fitur *edit* berfungsi untuk mengubah data tipe kamar. Fitur *delete* berfungsi untuk menghapus data tipe kamar.
10	Mengelola data daftar kamar	Mengelola data kamar merupakan *use case* yang terdapat dalam hak akses manager yang memiliki empat fitur, yaitu tambah, *view* dan *edit* dan *delete*. Fitur tambah berfungsi untuk menambah data kamar. Fitur *view* berfungsi untuk menampilkan detail data kamar. Fitur *edit* berfungsi untuk mengubah data kamar. Fitur *delete* berfungsi untuk menghapus data kamar.
11	*View* data *checkout*	*View* data *checkout* merupakan *use case* yang terdapat dalam hak akses manager dan *front office*. Fitur tersebut berfungsi untuk menampilkan data *checkout*.
12	*View* pembobotan kriteria	*View* data pembobotan kriteria merupakan *use case* yang terdapat dalam hak akses manager. Fitur tersebut berfungsi untuk menampilkan data pembobotan kriteria.

13	*Update* pembobotan kriteria	*Update* pembobotan kriteria merupakan *use case* yang terdapat dalam hak akses manager. Fitur tersebut berfungsi untuk menentukan dan merubah nilai tingkat kepentingan antar kriteria sehingga memperoleh nilai bobot prioritas AHP yang konsisten.
14	*View* produktivitas	*View* produktivitas merupakan *use case* yang terdapat dalam hak akses manager. Fitur tersebut berfungsi untuk menampilkan hasil yang diperoleh dari pengukuran produktivitas dalam beberapa periode.
15	*Input* perhitungan produktivitas	*Input* perhitungan produktivitas merupakan *use case* yang terdapat dalam hak akses manager. Fitur tersebut berfungsi untuk melakukan perhitungan produktivittas pada periode yang tertentu yang dipilih.
16	Mengelola data reservasi	Mengelola data reservasi merupakan *use case* yang terdapat dalam hak akses *front office* yang memiliki empat fitur, yaitu tambah, *view* dan *edit* dan *chek out*. Fitur tambah berfungsi untuk menambah data reservasi. Fitur *view* berfungsi untuk menampilkan detail data reservasi. Fitur *edit* berfungsi untuk mengubah data reservasi. Fitur *delete* berfungsi untuk melakukan proses *check out*.
17	*Input* absensi pegawai	*Input* absensi pegawai merupakan *use case* yang terdapat dalam hak akses HRD. Fitur tersebut berfungsi untuk memasukkan absensi pegawai.
18	*View* data komplain	*View* data komplain merupakan *use case* yang terdapat dalam hak akses manager, *front office, food & beverage, housekeeping* dan *engineering*. Fitur tersebut berfungsi untuk menampilkan data komplain yang diinputkan oleh *customer*.
19	*Input* penanganan komplain	*Input* penanganan komplain merupakan *use case* yang terdapat dalam hak akses *front office, food & beverage, housekeeping* dan *engineering*. Fitur tersebut berfungsi untuk menginputkan penanganan yang telah dilakukan oleh departemen yang bersangkutan.
20	*Input* komplain	*Input* komplain merupakan *use case* yang terdapat dalam hak akses *customer*. Fitur tersebut berfungsi untuk menginputkan komplain atau keluhan kepada departemen yang bersangkutan.
21	*Logout*	*Logout* merupakan *use case* yang berfungsi untuk keluar dari sistem.

4.4.3. *Use Case* Skenario

Use Case Skenario digunakan untuk menjelaskan fitur atau isi yang ada di *use case* diagram. *Use case* Skenario menjelaskan alur sistem dan keadaan yang akan terjadi ketika terjadi suatu *event* tertentu. *Use case* skenario sistem informasi pengukuran

produktivitas hotel di kabupaten Jember menggunakan metode *objective matrix* dan AHP adalah sebagai berikut:

a. Skenario *Login*

Skenario login merupakan penjelasan urutan reaksi aktor dan reaksi sistem pada saat melakukan *login*. Skenario login ini digunakan untuk menggambarkan proses login mulai dari memasukkan username dan password hingga menampilkan halaman home. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario login dapat dilihat pada Lampiran A Tabel A.1 *Use case* skenario *login*.

b. Skenario *View* Data Pegawai

Skenario *view* data pegawai merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan melihat data pegawai. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario *view* data pegawai dapat dilihat pada Lampiran A Tabel A.2 *Use case* skenario *view* data pegawai.

c. Skenario *View* Absensi

Skenario *view* absensi merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan melihat data absensi pegawai. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario *view* absensi dapat dilihat pada Lampiran A Tabel A.3 *Use case* skenario *view* absensi.

d. Skenario *View* Data Kamar

Skenario *view* data kamar merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan melihat data kamar. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario *view* data kamar dapat dilihat pada Lampiran A Tabel A.4 *Use case* skenario *view* data kamar.

e. Skenario *View* Data Tipe Kamar

Skenario *view* data tipe kamar merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan melihat data tipe kamar. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario *view* data tipe kamar dapat dilihat pada Lampiran A Tabel A.5 *Use case* skenario *view* data tipe kamar.

f. Skenario *View* Data *Check in*

Skenario *view* data *check in* merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan melihat data *check in*. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario *view* data *check in* dapat dilihat pada Lampiran A Tabel A.6 *Use case* skenario *view* data *check in*.

g. Skenario Mengelola Data *User*

Skenario mengelola data *user* merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan mengelola data *user* seperti menambah *user* dan melakukan konfirmasi pendaftaran *user*. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario mengelola data *user* dapat dilihat pada Lampiran A Tabel A.7 *Use case* skenario mengelola data *user*.

h. Skenario Mengelola Data Pegawai

Skenario mengelola data pegawai merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan mengelola data pegawai seperti melihat, menambah, mengedit dan menghapus data pegawai. Penjelasan urutan aksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari skenario *use case* mengelola data pegawai dijelaskan pada Tabel 4.3.

Tabel 4. 3 Skenario Mengelola Data Pegawai

Nama *Use Case*	Mengelola Data Pegawai
Aktor	HRD
Pre Kondisi	Aktor akan mengelola data pegawai
Post Kondisi	Aktor telah mengelola data pegawai

Skenario Normal – *Create*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data pegawai	
	2. Mengambil data pegawai yaitu id pegawai, nama, alamat, departemen dan tanggal masuk kerja dari *database*
	3. Menampilkan halaman daftar pegawai dengan tabel data pegawai dan tombol tambah data
4. Pilih tombol tambah data	
	5. Menampilkan form tambah pegawai
6. Mengisi form tambah pegawai seperti : nama, tempat lahir, tanggal lahir, alamat, departement dan tanggal kerja	
7. Pilih tombol simpan Jika batal lanjut ke nomer 7a	
	8. Menyimpan data pegawai ke *database*
	9. Menampilkan halaman daftar pegawai
Skenario Normal – *Edit*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data pegawai	
	2. Mengambil data pegawai yaitu id pegawai, nama, alamat, departemen dan tanggal masuk kerja dari *database*
	3. Menampilkan halaman daftar pegawai dengan tabel data pegawai dan tombol edit pada setiap baris data
4. Pilih tombol edit pada data pegawai yang akan diperbaharui	
	5. Menampilkan form edit data pegawai dengan tombol simpan dan batal
6. Mengisi form edit data pegawai seperti : nama, tempat lahir, tanggal lahir, alamat, departement dan tanggal kerja	
7. Pilih tombol simpan	

Jika batal lanjut ke nomer 7a	
	8. Mengupdate data pegawai pada *database*
	9. Menampilkan halaman daftar pegawai yang berhasil diupdate.
Skenario Normal – *View*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data pegawai	
	2. Mengambil data pegawai yaitu id pegawai, nama, alamat, departemen dan tanggal masuk kerja dari *database*
	3. Menampilkan halaman daftar pegawai dengan tabel data pegawai dan tombol *detail* pada setiap baris data
4. Pilih tombol *detail* pada data pegawai yang dipilih	
	5. Menampilkan halaman *detail* pegawai yang dipilih
Skenario Normal – *Delete*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data pegawai	
	2. Mengambil data pegawai yaitu id pegawai, nama, alamat, departemen dan tanggal masuk kerja dari *database*
	3. Menampilkan halaman daftar pegawai dengan tabel data pegawai dan tombol *delete* pada setiap baris data
4. Pilih tombol *delete* pada data pegawai yang akan dihapus	
	5. Menampilkan peringatan “Anda yakin ingin menghapus data?” dengan pilihan tombol *ok* dan *cancel*.
6. Pilih tombol *ok* Jika cancel lanjut ke nomor 6b.	
	7. Mengupdate status data pegawai yang dipilih menjadi hapus pada *database*
	8. Menampilkan halaman daftar pegawai yang berhasil dihapus
Skenario Alternatif – Batal	
Aksi Aktor	**Reaksi Sistem**

7a. Pilih tombol batal	
	8a. Menampilkan halaman daftar pegawai
Skenario Alternatif – *Cancel*	
Aksi Aktor	**Reaksi Sistem**
6b. Pilih tombol cancel	
	7b. Menampilkan halaman daftar pegawai
Skenario Alternatif (kolom kosong/tidak diisi)	
Aksi Aktor	**Reaksi Sistem**
7c. Pilih tombol simpan	
	8c. Menampilkan pesan "*please fill out this field*" dengan menunjuk pada kolom yang belum diisi

i. Skenario Mengelola Data Tipe Kamar

Skenario mengelola data tipe kamar merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan mengelola data tipe kamar seperti melihat, menambah, mengedit dan menghapus data tipe kamar. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario mengelola data tipe kamar dapat dilihat pada Lampiran A Tabel A.8 *Use case* skenario mengelola data tipe kamar.

j. Skenario Mengelola Data Daftar Kamar

Skenario mengelola data daftar kamar merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan mengelola data daftar kamar seperti melihat, menambah, mengedit dan menghapus data tipe kamar. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario mengelola data daftar kamar dapat dilihat pada Lampiran A Tabel A.9 *Use case* skenario mengelola data daftar kamar.

k. Skenario *View* Data *Checkout*

Skenario *view* data *checkout* merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan melihat data *checkout*. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario

mengelola data *checkout* dapat dilihat pada Lampiran A Tabel A.10 *Use case* skenario *view* data *checkout*.

l. Skenario *View* Pembobotan Kriteria

Skenario *view* pembobotan kriteria merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan melihat pembobotan kriteria. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario *view* pembobotan kriteria dapat dilihat pada Lampiran A Tabel A.11 *Use case* skenario *view* pembobotan kriteria.

m. Skenario *Update* Bobot Kriteria

Skenario *update* data pembobotan kriteria merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan mengupdate bobot kriteria. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario *update* bobot kriteria dapat dilihat pada Lampiran A Tabel A.12 *Use case* skenario *update* bobot kriteria.

n. Skenario *View* Produktivitas

Skenario *view* produktivitas merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan melihat produktivitas. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario *view* produktivitas dapat dilihat pada Lampiran A Tabel A.13 *Use case* skenario *view* produktivitas.

o. Skenario *Input* Perhitungan Produktivitas

Skenario *input* perhitungan produktivitas merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan menginputkan perhitungan produktivitas. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario *input* perhitungan produktivitas dapat dilihat pada Lampiran A Tabel A.14 *Use case* skenario *input* perhitungan produktivitas.

p. Skenario Mengelola Data Reservasi

Skenario mengelola data reservasi merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan mengelola data reservasi seperti melihat, menambah,

mengubah, melakukan proses *checkout*. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario mengelola data reservasi dapat dilihat pada Lampiran A Tabel A.15 *Use case* skenario mengelola data reservasi.

q. Skenario *Input* Absensi Pegawai

Skenario mengelola *input* absensi pegawai merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan menginputkan absensi pegawai. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario *input* absensi pegawai dapat dilihat pada Lampiran A Tabel A.16 *Use case* skenario *input* absensi pegawai.

r. Skenario *View* data Komplain

Skenario *view* data komplain merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan melihat data komplain. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario *view* data komplain dapat dilihat pada Lampiran A Tabel A.17 *Use case* skenario *view* data komplain.

s. Skenario *Input* Penanganan Komplain

Skenario *input* penanganan komplain merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan menginputkan penanganan komplain. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario *input* penanganan komplain dapat dilihat pada Lampiran A Tabel A.18 *Use case* skenario *input* penanganan komplain.

t. Skenario *Input* Komplain

Skenario *input* komplain merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan menginputkan *komplain*. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario *input* komplain dapat dilihat pada Lampiran A Tabel A.19 *Use case* skenario *input* komplain.

u. Skenario *Logout*

Skenario *view* produktivitas merupakan penjelasan urutan reaksi aktor dan reaksi sistem saat akan melakukan *logout*. Penjelasan urutan reaksi aktor dan reaksi sistem pada skenario normal dan skenario alternatif dari *use case* skenario *logout* dapat dilihat pada Lampiran A Tabel A.20 *Use case* skenario *logout*.

4.4.4. *Activity* Diagram

Activity Diagram menggambarkan aliran aktivitas sistem informasi pengukuran produktivitas hotel di kabupaten Jember menggunakan metode *objective matrix* dan AHP yang akan dibangun. Sistem informasi pengukuran produktivitas hotel di kabupaten Jember menggunakan metode *objective matrix* dan AHP ini memiliki dua puluh satu *activity* diagram yaitu sebagai berikut:

a. *Activity* Diagram *Login*

Activity diagram *login* dilakukan oleh manager, HRD, *front office, food & beverage, housekeeping* dan *engineering*. *Activity* diagram *login* menjelaskan tentang bagaimana sistem dapat menjalankan fungsi melakukan autentifikasi hak akses semua aktor dalam menggunakan sistem, proses lengkapnya dapat dilihat pada Lampiran B Gambar B.1 *Activity* diagram *login*.

b. *Activity* Diagram *View* Data Pegawai

Activity diagram *view* data pegawai dapat dilakukan oleh manager. *Activity* diagram *view* data pegawai menjelaskan alur bagaimana manager dapat melihat data pegawai. Alur lengkap dari *activity* diagram *view* data pegawai dapat dilihat pada Lampiran B Gambar B.2 *Activity* diagram *view* data pegawai.

c. *Activity* Diagram *View* Absensi

Activity diagram *view* absensi dapat dilakukan oleh manager dan HRD. *Activity* diagram *view* absensi menjelaskan alur bagaimana manager dan HRD dapat melihat data absensi. Alur lengkap dari *activity* diagram *view* data pegawai dapat dilihat pada Lampiran B Gambar B.3 *Activity* diagram *view* absensi.

d. *Activity* Diagram *View* Data Kamar

Activity diagram *view* data kamar dapat dilakukan oleh manager dan *front office*. *Activity* diagram *view* data kamar menjelaskan alur bagaimana manager dan *front office* dapat melihat data kamar, Alur lengkap dari *activity* diagram *view* data kamar dapat dilihat pada Lampiran B Gambar B.4 *Activity* diagram *view* data kamar.

e. *Activity* Diagram *View* Data Tipe Kamar

Activity diagram *view* data tipe kamar dapat dilakukan oleh manager. *Activity* diagram *view* data tipe kamar menjelaskan alur bagaimana manager dapat melihat data tipe kamar. Alur lengkap dari *activity* diagram *view* data tipe kamar dapat dilihat pada Lampiran B Gambar B.5 *Activity* diagram *view* data tipe kamar.

f. *Activity* Diagram *View* Data *Check in*

Activity diagram *view* data *check in* dapat dilakukan oleh manager dan *front office*. *Activity* diagram *view* data kamar menjelaskan alur bagaimana manager dan *front office* dapat melihat data *check in*. Alur lengkap dari *activity* diagram *view* data *check in* dapat dilihat pada Lampiran B Gambar B.6 *Activity* diagram *view* data *check in*.

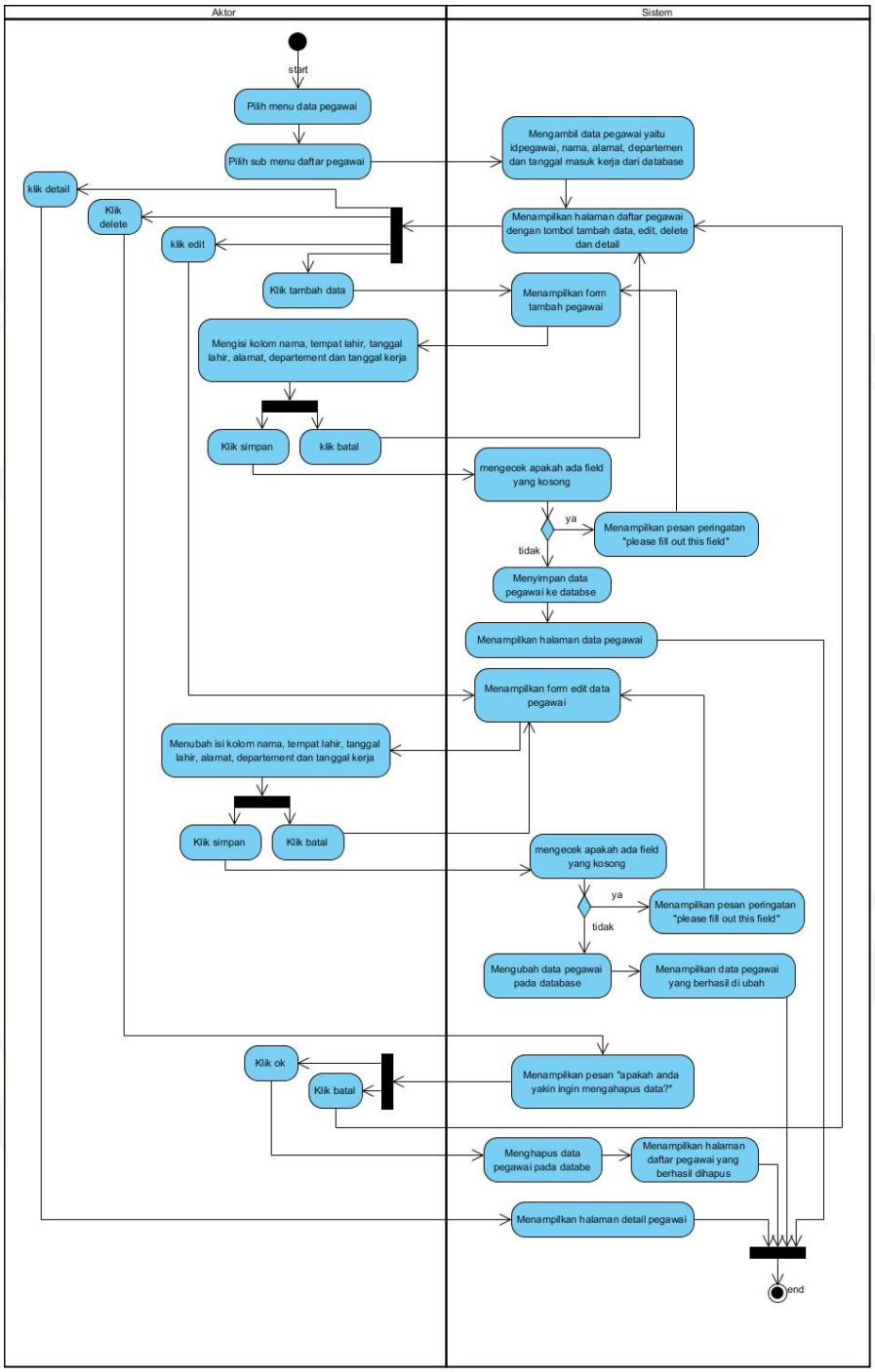
g. *Activity* Diagram Mengelola Data *User*

Activity diagram mengelola data *user* dapat dilakukan oleh HRD. *Activity* diagram mengelola data *user* menjelaskan alur bagaimana HRD dapat mengelola data *user* seperti menambah user dan melakukan konfirmasi pada saat user menginputkan pendaftaran. Alur lengkap dari *activity* diagram mengelola data user dapat dilihat pada Lampiran B Gambar B.7 *Activity* diagram mengelola data *user*.

h. *Activity* Diagram Mengelola Data Pegawai

Activity diagram mengelola data pegawai dapat dilakukan oleh HRD. *Activity* diagram mengelola data pegawai menjelaskan alur bagaimana HRD dapat mengelola data pegawai seperti melihat, menambah, mengedit, dan menghapus data pegawai, alur lengkap dapat dilihat pada Lampiran B Gambar B.X *Activity*

diagram mengelola data pegawai. *Activity* diagram mengelola data pegawai dapat dilihat pada Gambar 4.3.



Gambar 4. 3 *Activity* Diagram Mengelola Data Pegawai

i. *Activity* Diagram Mengelola Data Tipe Kamar

Activity diagram mengelola data tipe kamar dapat dilakukan oleh HRD. *Activity* diagram mengelola data tipe kamar menjelaskan alur bagaimana HRD dapat mengelola data tipe kamar seperti melihat, menambah, mengedit, dan menghapus data tipe kamar. Alur lengkap dari *activity* diagram mengelola data tipe kamar dapat dilihat pada Lampiran B Gambar B.8 *Activity* diagram mengelola data tipe kamar.

j. *Activity* Diagram Mengelola Data Daftar Kamar

Activity diagram mengelola data daftar kamar dapat dilakukan oleh HRD. *Activity* diagram mengelola data daftar kamar menjelaskan alur bagaimana HRD dapat mengelola data daftar kamar seperti melihat, menambah, mengedit, dan menghapus data daftar kamar. Alur lengkap dari *activity* diagram mengelola data daftar kamar dapat dilihat pada Lampiran B Gambar B.9 *Activity* diagram mengelola data daftar kamar.

k. *Activity* Diagram *View* Data *Check out*

Activity diagram *view* data *chek out* dapat dilakukan oleh manager dan *front office*. *Activity* diagram *view* data *checkout* menjelaskan alur bagaimana manager dan *front office* dapat melihat data *checkout*. Alur lengkap dari *activity* diagram *view* data *checkout* dapat dilihat pada Lampiran B Gambar B.10 *Activity* diagram *view* data *checkout*.

l. *Activity* Diagram *View* Pembobotan Kriteria

Activity diagram *view* pembobotan kriteria dapat dilakukan oleh manager. *Activity* diagram *view* pembobotan kriteria menjelaskan alur bagaimana manager dapat melihat pembobotan kriteria. Alur lengkap dari *activity* diagram *view* pembobotan kriteria dapat dilihat pada Lampiran B Gambar B.11 *Activity* diagram *view* pembobotan kriteria.

m. *Activity* Diagram *Update* Bobot Kriteria

Activity diagram *update* pembobotan kriteria menjelaskan tentang bagaimana manager dapat mengubah data nilai kepentingan antar kriteria pada sistem

sehingga menghasilkan nilai bobot prioritas kriteria. Alur lengkap dari *activity* diagram *update* bobot kriteria dapat dilihat pada Lampiran B Gambar B.12 *Activity* diagram *update* bobot kriteria.

n. *Activity* Diagram *View* Produktivitas

Activity diagram *view* produktivitas dapat dilakukan oleh manager. *Activity* diagram *view* produktivitas menjelaskan tentang bagaimana manager dapat melihat data produktivitas yang sudah pernah diukur pada periode tertentu. Alur lengkap dari *activity* diagram *view* produktivitas dapat dilihat pada Lampiran B Gambar B.13 *Activity* diagram *view* produktivitas.

o. *Activity* Diagram *Input* Perhitungan Produktivitas

Activity diagram *input* perhitungan produktivitas dapat dilakukan oleh manager. *Activity* diagram *input* perhitungan produktivitas menjelaskan tentang bagaimana manager dapat menginputkan perhitungan produktivitas pada periode tertentu. Alur lengkap dari *activity* diagram *input* perhitungan produktivitas dapat dilihat pada Lampiran B Gambar B.14 *Activity* diagram perhitungan produktivitas.

p. *Activity* Diagram Mengelola Data Reservasi

Activity diagram mengelola data reservasi dapat dilakukan oleh *front office*. *Activity* diagram mengelola data reservasi menjelaskan alur bagaimana *front office* dapat mengelola data daftar reservasi seperti melihat, menambah, mengedit, dan melakukan *checkout*. Alur lengkap dari *activity* diagram mengelola data reservasi dapat dilihat pada Lampiran B Gambar B.15 *Activity* diagram mengelola data reservasi.

q. *Activity* Diagram *Input* Absensi Pegawai

Activity diagram *input* absensi pegawai dapat dilakukan oleh HRD. *Activity* diagram *input* absensi pegawai menjelaskan tentang bagaimana HRD dapat menginputkan absensi pegawai. Alur lengkap dari *activity* diagram *input* absensi pegawai dapat dilihat pada Lampiran B Gambar B.16 *Activity* diagram *input* absensi pegawai.

r. *Activity* Diagram *View* data Komplain

Activity diagram *view* data komplain dapat dilakukan oleh manager, *front office, food & beverage, housekeeping* dan *engineering*. *Activity* diagram *view* data komplain menjelaskan tentang bagaimana aktor dapat melihat data komplain. Alur lengkap dari *activity* diagram *view* data komplain dapat dilihat pada Lampiran B Gambar B.17 *Activity* diagram *view* data komplain.

s. *Activity* Diagram *Input* Penanganan Komplain

Activity diagram *input* penanganan komplain dapat dilakukan oleh *front office, food & beverage, housekeeping* dan *engineering*. *Activity* diagram *input* penanganan komplain menjelaskan tentang bagaimana aktor dapat menginputkan penanganan komplain yang telah selesai ditangani. Alur lengkap dari *activity* diagram *input* penanganan komplain dapat dilihat pada Lampiran B Gambar B.18 *Activity* diagram *input* penanganan komplain.

t. *Activity* Diagram *Input* Komplain

Activity diagram *input* penanganan komplain dapat dilakukan oleh *customer*. *Activity* diagram *input* komplain menjelaskan tentang bagaimana *customer* dapat menginputkan komplain untuk departemen yang bersangkutan. Alur lengkap dari *activity* diagram *input* komplain dapat dilihat pada Lampiran B Gambar B.19 *Activity* diagram *input* komplain.

u. *Activity* Diagram *Logout*

Activity diagram *logout* menjelaskan tentang bagaimana sistem dapat menjalankan fungsi melakukan proses keluar dari sistem. Alur lengkap dari *activity* diagram *logout* dapat dilihat pada Lampiran B Gambar B.20 *Activity* diagram *logout*.

4.4.5. *Sequence* Diagram

Sequence diagram merupakan dokumentasi dari desain yang berbentuk diagram dan menampilkan urutan interaksi - interaksi antar objek di dalam sistem. *Sequence* diagram digunakan untuk menggambarkan skenario dan memodelkan aliran logika

dalam sistem dengan cara *visual*. *Sequence* diagram dari sistem informasi pengukuran produktivitas hotel di kabupaten Jember menggunakan metode *objective matrix* dan AHP adalah sebagai berikut:

a. *Sequence* Diagram *Login*

Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk melakukan *login* ke dalam sistem. Masing-masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram *login* lebih lengkap dijelaskan pada Lampiran C Gambar C.1 *sequence* diagram *login.*

b. *Sequence* Diagram *View* Data Pegawai

Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk melihat data pegawai. Masing-masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram *view* data pegawai lebih lengkap dijelaskan pada Lampiran C Gambar C.2 *sequence* diagram *view* data pegawai.

c. *Sequence* Diagram *View* Absensi

Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk melihat absensi. Masing-masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram *view* absensi lebih lengkap dijelaskan pada Lampiran C Gambar C.3 *sequence* diagram *view* absensi.

d. *Sequence* Diagram *View* Data Kamar

Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk melihat data kamar. Masing-masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram *view* data kamar lebih lengkap dijelaskan pada Lampiran C Gambar C.4 *sequence* diagram *view* data kamar.

e. *Sequence* Diagram *View* Data Tipe Kamar

Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk melihat data tipe kamar. Masing-masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram *view* data tipe kamar lebih lengkap dijelaskan pada Lampiran C Gambar C.5 *sequence* diagram *view* data tipe kamar.

f. *Sequence* Diagram *View* Data *Check in*

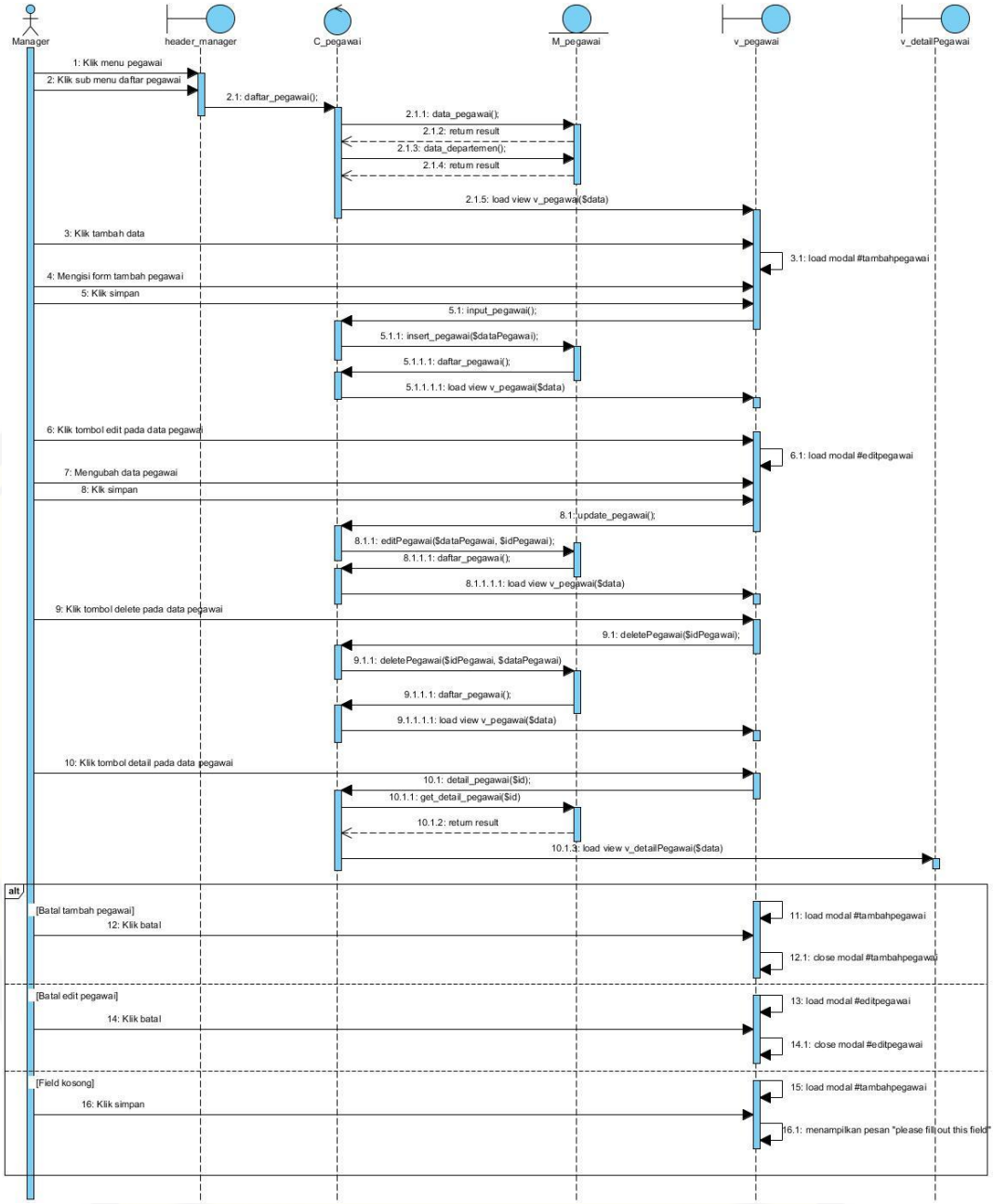
Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk melihat data *check in*. Masing-masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram *view* data *check in* lebih lengkap dijelaskan pada Lampiran C Gambar C.6 *sequence* diagram *view* data *check in*.

g. *Sequence* Diagram Mengelola Data *User*

Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk menambah data *user* dan mengkonfirmasi *user* yang melakukan pendaftaran. Masing-masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram mengelola data *user* lebih lengkap dijelaskan pada Lampiran C Gambar C.7 *sequence* diagram mengelola data *user*.

h. *Sequence* Diagram Mengelola Data Pegawai

Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk melihat, menambah, mengedit, dan menghapus data pegawai. *Sequence* diagram mengelola data pegawai dapat dilihat pada Gambar 4.4.



Gambar 4. 4 *Sequence* Diagram Mengelola Data Pegawai

i. *Sequence* Diagram Mengelola Data Tipe Kamar

Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk melihat,

menambah, mengedit, dan menghapus data tipe kamar. Masing - masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram mengelola data tipe kamar lebih lengkap dijelaskan pada Lampiran C Gambar C.8 *sequence* diagram mengelola data tipe kamar.

j. *Sequence* Diagram Mengelola Data Daftar Kamar

Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk melihat, menambah, mengedit, dan menghapus data daftar kamar. Masing - masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram mengelola data daftar kamar lebih lengkap dijelaskan pada Lampiran C Gambar C.9 *sequence* diagram mengelola data daftar kamar.

k. *Sequence* Diagram *View* Data *Checkout*

Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk melihat data *checkout*. Masing - masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram *view* data *checkout* lebih lengkap dijelaskan pada Lampiran C Gambar C.10 *sequence* diagram *view* data *checkout.*

l. *Sequence* Diagram *View* Pembobotan Kriteria

Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk melihat data pembobotan kriteria. Masing - masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram *view* pembobotan kriteria lebih lengkap dijelaskan pada Lampiran C Gambar C.11 *sequence* diagram *view* pembobotan kriteria.

m. *Sequence* Diagram *Update* Bobot Kriteria

Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk mengubah bobot kriteria. Masing - masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram *update* bobot kriteria lebih lengkap dijelaskan pada Lampiran C Gambar C.12 *sequence* diagram *update* bobot kriteria.

n. *Sequence* Diagram *View* Produktivitas

Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk melihat hasil produktivitas. Masing - masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram *view* produktivitas lebih lengkap dijelaskan pada Lampiran C Gambar C.13 *sequence* diagram *view* produktivitas.

o. *Sequence* Diagram *Input* Perhitungan Produktivitas

Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk menginputkan perhitungan produktivitas pada periode tertentu. Masing - masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram *input* perhitungan produktivitas lebih lengkap dijelaskan pada Lampiran C Gambar C.14 *sequence* diagram *input* perhitungan produktivitas.

p. *Sequence* Diagram Mengelola Data Reservasi

Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk melihat, menambah dan mengedit data reservasi serta melakukan proses *checkout*. Masing - masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram mengelola data reservasi lebih lengkap dijelaskan pada Lampiran C Gambar C.15 *sequence* diagram mengelola data reservasi.

q. *Sequence* Diagram *Input* Absensi Pegawai

Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk menginputkan absensi pegawai. Masing - masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram *input* absensi pegawai lebih lengkap dijelaskan pada Lampiran C Gambar C.16 *sequence* diagram *input* absensi pegawai.

r. *Sequence* Diagram *View* Data Komplain

Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk melihat data

komplain. Masing - masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram *view* data komplain lebih lengkap dijelaskan pada Lampiran C Gambar C.17 *sequence* diagram *view* data komplain.

s. *Sequence* Diagram *Input* Penanganan Komplain

Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk menginputkan penanganan komplain yang dilakukan oleh departemen terkait. Masing - masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram *input* penanganan komplain lebih lengkap dijelaskan pada Lampiran C Gambar C.18 *sequence* diagram *input* penanganan komplain.

t. *Sequence* Diagram *Input* Komplain

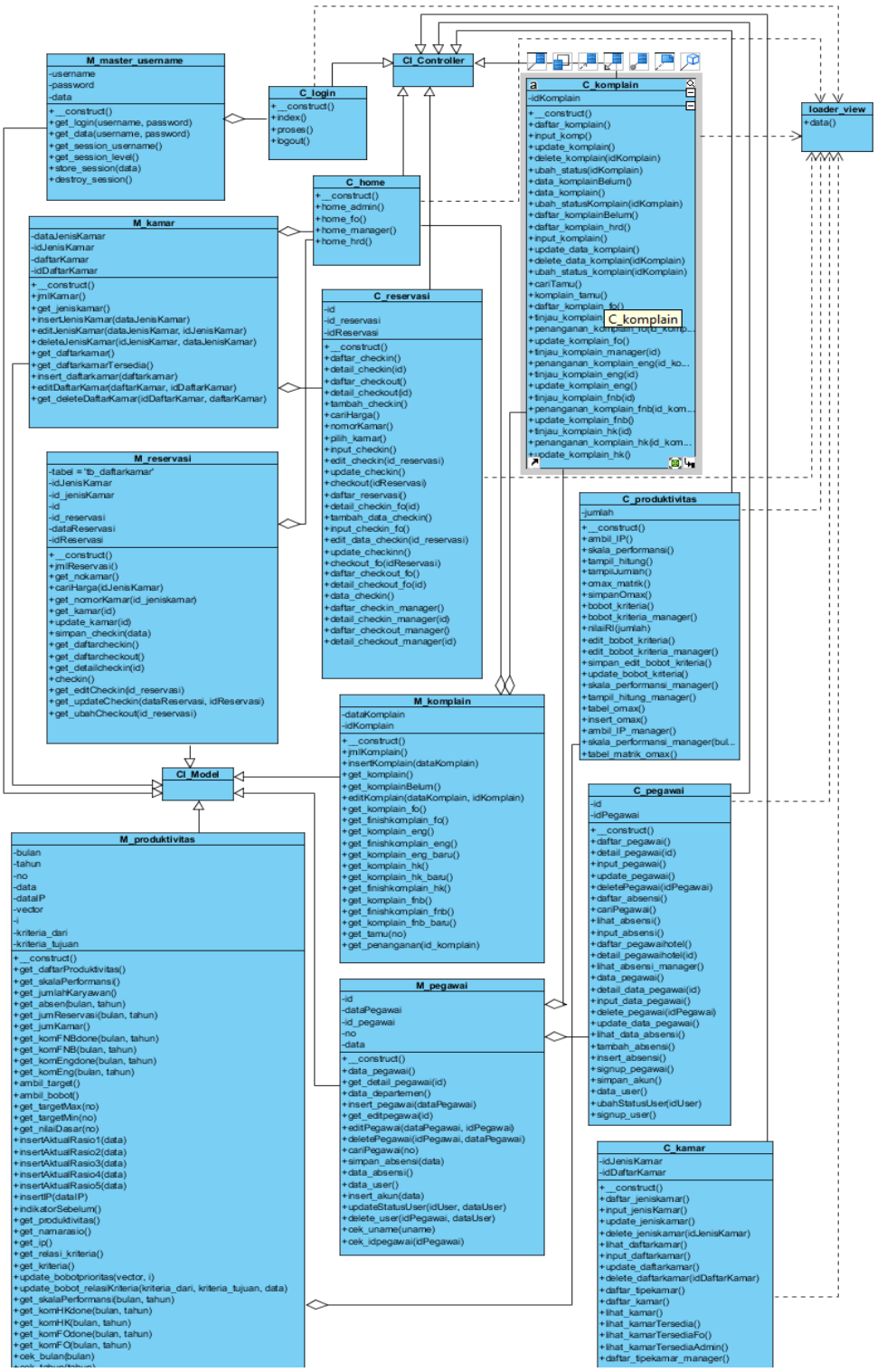
Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk menginputkan komplain yang dilakukan oleh *customer*. Masing - masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram *input* komplain lebih lengkap dijelaskan pada Lampiran C Gambar C.19 *sequence* diagram *input* komplain.

u. *Sequence* Diagram *Logout*

Sequence diagram ini menggambarkan urutan interaksi antara objek di dalam sistem sebagai sebuah respon dari suatu kejadian atau *event* untuk melakukan *logout*. Masing - masing *class* akan ditampilkan secara *visual* dengan gambar. *Sequence* diagram *logout* lebih lengkap dijelaskan pada Lampiran C Gambar C.20 *sequence* diagram *logout*.

4.4.6. *Class* Diagram

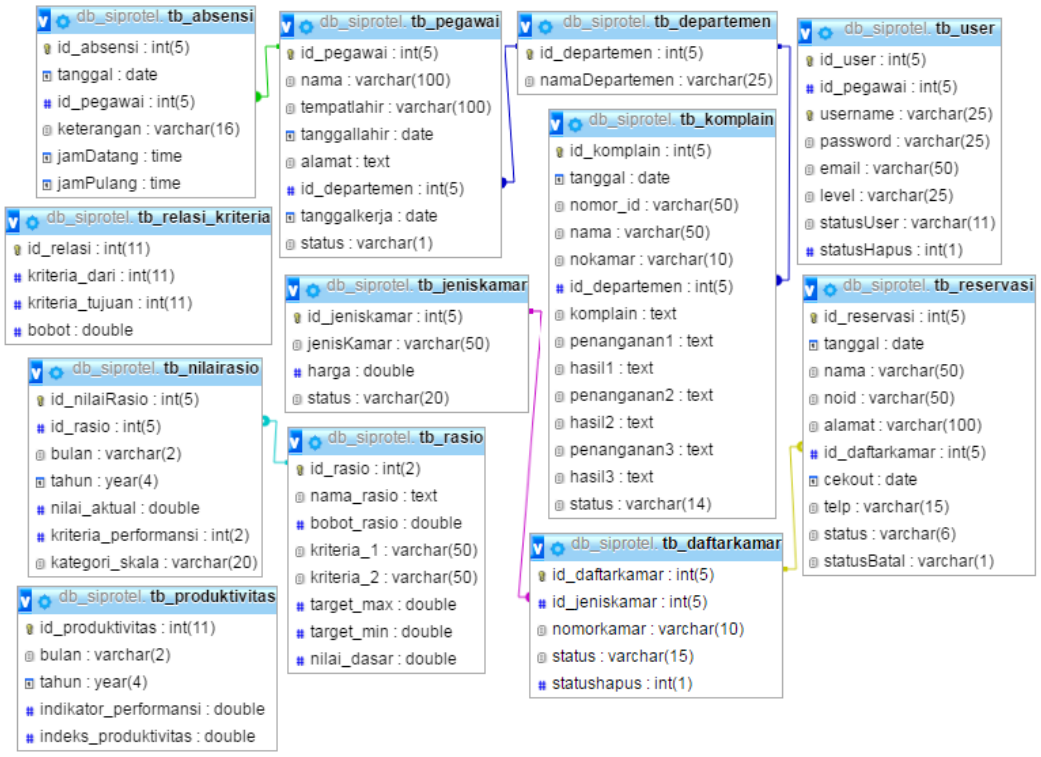
Class diagram merupakan diagram yang menggambarkan hubungan antara kelas yang digunakan untuk membangun suatu sistem. Dalam paradigma OOP (*Object Oriented Programming*) terdapat 3 jenis kelas yaitu *model*, *view* dan *controller*. *Class* diagram sistem informasi pengukuran produktivitas hotel di kabupaten Jember menggunakan metode *objective matrix* dan AHP dapat dilihat pada Gambar 4.5.



Gambar 4. 5 *Class* diagram

4.4.7. *Entity Relationship* Diagram (ERD)

Entity Relationship Diagram (ERD) sistem sistem informasi pengukuran produktivitas hotel di kabupaten Jember menggunakan metode *objective matrix* dan AHP merupakan gambaran komponen dan struktur *database* yang digunakan dalam pembuatan sistem. ERD yang diimplementasikan pada sistem ini terdiri dari X entitas yang dapat dilihat pada Gambar 4.6.



Gambar 4. 6 *Entity Relationship Diagram*

4.5 Implementasi

Desain yang telah dibuat akan di implementasikan ke dalam kode program. Beberapa hal yang dilakukan dalam tahap implementasi antara lain:

a. Penulisan kode program (*coding*) menggunakan bahasa pemrograman *Page Hyper Text Pre-Processor (PHP)* dengan bantuan *framework Code Igniter* (CI).

b. Manajemen basis data menggunakan *DBMS MySQL*.

Kode program perhitungan metode *objective matrix* dan AHP terdapat di *class* m_produktivitas pada *package models*. Perhitungan metode *objective matrix* dan AHP terdapat di *class* c_produktivitas pada *package controllers*. Penulisan kode program perhitungan metode AHP dapat dilihat pada Tabel 4.4, Tabel 4.5, Tabel 4.6 dan perhitungan metode *objective matrix* pada Tabel 4.7, Tabel 4.8, Tabel 4.9, Tabel 4.10, Tabel 4.11 dan Tabel 4.12.

Tabel 4. 4 Kode program *function* bobot_kriteria_manager()

No.	Kode
42	function bobot_kriteria_manager() {
43	$data['daftar_kriteria'] = $this->M_produktivitas->get_kriteria()->result();
44	$data['relasi_kriteria'] = $this->M_produktivitas->get_relasi_kriteria()->result();
45	$data['jumlahKriteria'] = count($data['daftar_kriteria']);
46	
47	//=============Menghitung jumlah dari setiap kolom=============
48	$jumlah = array();
49	foreach ($data['daftar_kriteria'] as $kriteria) {
50	$jumlah1 = 0;
51	foreach ($data['relasi_kriteria'] as $relasi) {
52	if($relasi->kriteria_tujuan == $kriteria->id_rasio){ //filter kriteria perkolom
53	$jumlah1 = $jumlah1 + $relasi->bobot;
54	}
55	}
56	array_push($jumlah, $jumlah1);
57	}
58	$data['jumlah_per_kolom'] = $jumlah;
59	
60	//======Menghitung nilai per cell (total setiap kolom/jumlah kriteria)=====
61	$arrayVector = array();
62	foreach ($data['daftar_kriteria'] as $kriteria) {
63	$nilai = array();
64	$a = 0;
65	foreach ($data['relasi_kriteria'] as $relasi) {
66	if($relasi->kriteria_dari == $kriteria->id_rasio){
67	$nilai1 = $relasi->bobot / $jumlah[$a];
68	array_push($nilai, $nilai1);

```
69              $a++;
70              }
71          }
72
73      //============Menghitung nilai vector==============
74          $vector = 0;
75          for($i = 0; $i < count($nilai); $i++){
76            $vector = $vector + $nilai[$i];
77          }
78
79          $vector = $vector / $data['jumlahKriteria'];
80          $vector = number_format($vector, 3, '.', '');
81          array_push($arrayVector, $vector);
82
83          for ($i=0; $i  < 5 ; $i++) {
84             if(isset($arrayVector[$i])){
85                $this->M_produktivitas->update_bobotprioritas($arrayVector[$i], $i+1);
86             }
87          }
88
89        }
90
91      //==============Menghitung nilai lamda=====================
92          $lamda = 0;
93            for($i = 0; $i < count($arrayVector); $i++){
94              $lamda = $lamda + ($arrayVector[$i] * $jumlah[$i]);
95            }
96
97      //===========Menghitung nilai CI============================
98         $ci = ($lamda - $data['jumlahKriteria']) / ($data['jumlahKriteria'] - 1);
99         $ci = number_format($ci, 3, '.', '');
100
101     //==================Menghitung nilai CR==================
102        $cr = $ci / $this->nilaiRI($data['jumlahKriteria']);
103        $cr = number_format($cr, 3, '.', '');
104
105      $data['nilai_lamda']    = $lamda;
106      $data['nilai_ci']       = $ci;
107      $data['nilai_cr']       = $cr;
108      $data['nilai_vector']   = $arrayVector;
```

```
109     $data['kriteria']=$this->M_produktivitas->get_kriteria();
110
111     $this->load->view('manager/v_kriteria', $data);
112     $this->load->view('manager/header_manager', $data);
113     }
```

Function bobot_kriteria_manager() pada Tabel 4.4 merupakan kode program yang digunakan untuk melakukan perhitungan bobot pada kriteria dengan menggunakan metode AHP, dimana user memasukkan nilai kepentingan antar kriteria dan sistem memberikan respon dengan menghasilkan bobot prioritas. Pada kode program diatas tahap perhitungan metode AHP dimulai dari baris 48-58 yaitu menghitung jumlah setiap kolom dari nilai kepentingan yang telah dimasukkan sebelumnya, kemudian perhitungan dilanjutkan pada baris 61-71 yaitu menghitung nilai per-*cell* dengan pada masing-masing kriteria, kemudian dilanjutkan pada baris 74-89 yaitu menghitung nilai vektor, hasil dari tahap ini yaitu bobot prioritas yang akan disimpan atau diperbaharui, untuk tahap selanjutnya yaitu menghitung kekonsistenan dari nilai kepentingan yang telah dimasukkan, dimulai dari menghitung nilai lamda yang ada pada baris 92-95, nilai CI pada baris 98-99 dan menghitung nilai CR pada baris 102-103. Nilai perbandingan dapat dinyatakan konsisten apabila hasil perhitungan dari CR tidak boleh lebih dari 0,01. Apabila nilai melebihi 0,01 maka dinyatakan tidak konsisten dan harus mengubah nilai kepentingan antar kriteria sampai nilai tersebut dapat dikatakan konsisten

Tahap perhitungan untuk mengubah nilai bobot prioritas yang tidak konsisten terdapat pada *function* edit_bobot_kriteria() dan *function* update _bobot_kriteria dapat dilihat pada Tabel 4.5.

Tabel 4. 5 Kode program *function* edit_bobot_kriteria()

No	Kode
153	function edit_bobot_kriteria_manager(){
154	$data['daftar_kriteria'] = $this->M_produktivitas->get_kriteria()->result();
155	$data['relasi_kriteria'] = $this->M_produktivitas->get_relasi_kriteria()->result();
156	$data['jumlah'] = count($data['daftar_kriteria']);

```
157
158         $this->load->view('manager/v_editKriteria', $data);
159         $this->load->view('manager/header_manager');
160     }
161
162     function update_bobot_kriteria(){
163         for($i = 1; $i <= 6; $i++){ // i adalah id kriteria dari
164             for($j = 1; $j <= 6; $j++){ // j adalah id kriteria tujuan
165                 $data['bobot'] = $this->input->post($i.$j);
166                 $this->M_produktivitas->update_bobot_relasiKriteria($i, $j, $data);
167             }
168         }
169         redirect(site_url().'/C_produktivitas/bobot_kriteria_manager?suksesupdate');
170     }
```

Kode program pada tabel 4.5 digunakan untuk mengubah nilai bobot prioritas dengan cara mengubah nilai kepentingan antar kriteria apabila terjadi nilai yang tidak konsisten atau jika nilai kepentingan hendak diubah, *function* edit_bobot_kriteria() digunakan untuk menampilkan halaman update bobot, sedangkan *function* update_bobot_kriteria() digunakan untuk menyimpan nilai kepentingan yang telah dimasukkan atau diinputkan pada halaman bobot kriteria sehingga diperoleh bobot prioritas yang baru dan konsisten

Tahap perhitungan metode AHP selanjutnya adalah pada *model* m_produktivitas yang terdapat pada *function* md_relasi_kriteria(), md_update_bobot_relasiKriteria(), md_update_bobotprioritas() dapat dilihat pada Tabel 4.6.

Tabel 4. 6 Kode program *function* get_relasi_kriteria()

No	Kode

```
261     function get_relasi_kriteria(){
262         $this->db->select('*');
263         $this->db->from('tb_relasi_kriteria');
264         return $this->db->get();
265     }
266
267     function update_bobotprioritas($vector, $i){
```

```
268         $query = $this->db->query("UPDATE `tb_rasio`
269                                    SET `bobot_rasio` ='$vector'
270                                    where id_rasio = '$i'
271                                    ");
272     return $query;
273     }
274
275     function update_bobot_relasiKriteria($kriteria_dari, $kriteria_tujuan, $data){
276         $this->db->where('kriteria_dari', $kriteria_dari);
277         $this->db->where('kriteria_tujuan', $kriteria_tujuan);
278         $this->db->update('tb_relasi_kriteria', $data);
279     }
```

Kode program pada Tabel 4.6 adalah *function* get_relasi_kriteria() pada baris 261-265 yang digunakan untuk mengambil data relasi kriteria yang sudah tersimpan di *database*. *Function* update_bobotprioritas() pada baris 267-273 digunakan untuk menyimpan bobo prioritas yang telah dihitung dengan cara memasukkan nilai kepentingan antar kriteria, sedangkan *function* update_bobot_relasiKriteria() pada baris 275-279 digunakan untuk menyimpan nilai kepentingan yang telah dilakukan dalam *database.*

Kode program perhitungan metode *objective matrix* terdapat pada *function* tabel_matrik_omax(). Kode program tersebut merupakan kode program yang digunakan untuk perhitungan produktivitas dalam suatu periode tertentu, perhitungan produktivitas menggunakan metode *objective matrix* dapat dilihat pada Tabel 4.7. Penulisan kode program lainnya dapat dilihat pada Lampiran D (Kode Program).

Tabel 4. 7 Kode program menghitung nilai skala performansi

No	Kode

```
189 public function tabel_matrik_omax(){
190         $bulan = $this->input->post('bulan', true);
191         $tahun = $this->input->post('tahun', true);
192         $a1 = $this->input->post('a1', true);
193         $a2 = $this->input->post('a2', true);
194         $a3 = $this->input->post('a3', true);
195         $a31 = $this->input->post('a31', true);
196         $a4 = $this->input->post('a4', true);
197         $a5 = $this->input->post('a5', true);
```

```
198         $a6 = $this->input->post('a6', true);
199         $a7 = $this->input->post('a7', true);
200         $a8 = $this->input->post('a8', true);
201         $a9 = $this->input->post('a9', true);
202         $a10 = $this->input->post('a10', true);
203         $a11 = $this->input->post('a11', true);
204
205         //Menghitung nilai aktual hasil (skala performansi)
206         $r1 = round((($a1*100)/$a2),2);
207
208         if ($a3 == 0 && $a31 == 0) {
209             $r2=100;
210         } else {
211             $r2 = round((($a3*100)/$a31),2);
212         }
213
214         if ($a4 == 0 && $a5 == 0) {
215             $r3=100;
216         } else {
217             $r3 = round((($a4*100)/$a5),2);
218         }
219
220         $r4 = round((($a6*100)/$a7),2);
221
222         if ($a8 == 0 && $a9 == 0) {
223             $r5=100;
224         } else {
225             $r5 = round((($a8*100)/$a9),2);
226         }
227
228         if ($a10 == 0 && $a11 == 0) {
229             $r6=100;
230         } else {
231             $r6 = round((($a10*100)/$a11),2);
232         }
```

Kode program pada Tabel 4.7 merupakan tahap pertama pada metode *objective matrix*. Baris 206-232 pada *function* tabel_matrix_omax() berfungsi untuk menghitung nilai aktual dari data yang didapat sebelumnya, hasil dari perhitungan digunakan sebagai acuan untuk menentukan penilaian terhadap kinerja suatu departemen yang berpengaruh terhadap peningkatan maupun penurunan produktivitas. Untuk perhitungan produktivitas dilanjutkan pada Tabel 4.8

Tabel 4. 8 Kode program menghitung nilai kenaikan matrik

No	Kode
234	//mengisi matrik tiap skala
235	$this->load->model('M_produktivitas');
236	$target = $this->M_produktivitas->ambil_target();
237	$data_bobot = $this->M_produktivitas->ambil_bobot();
238	
239	$jumlahrasio=count($target);
240	
241	for($i = 0; $i < 6; $i++) {
242	$maxRasio = $this->M_produktivitas->get_targetMax($i + 1);
243	$minRasio = $this->M_produktivitas->get_targetMin($i + 1);
244	$nilaiDasar = $this->M_produktivitas->get_nilaiDasar($i + 1);
245	
246	$target_max[$i] = $maxRasio[0]['max_rasio' . ($i + 1)];
247	$target_min[$i] = $minRasio[0]['min_rasio' . ($i + 1)];
248	$nilai_dasar[$i] = $nilaiDasar[0]['dasar' . ($i + 1)];
249	$bobot[$i] = $data_bobot[$i]['bobot_rasio'];
250	}
251	
252	//skala 1-2
253	for ($i = 0; $i < $jumlahrasio; $i++) {
254	$selisih1[$i]= ($nilai_dasar[$i]-$target_min[$i])/3;
255	}
256	
257	//skala 4-9
258	for ($i = 0; $i < $jumlahrasio; $i++) {
259	$selisih2[$i]= ($target_max[$i]-$nilai_dasar[$i])/7;
260	}

Kode program pada Tabel 4.8 merupakan *function* pada tabel_matrix_omax() berfungsi untuk menentukan nilai kenaikan matrik pada setiap skala mulai dari skala paling rendah yaitu 0 sampai skala paling tinggi yaitu 10, jumlah kenaikan tiap skala didapat dari perhitungan data yang telah didapatkan pada saat melakukan pengumpulan data. Perhitungan produktivitas dilanjutkan pada Tabel 4.9.

Tabel 4. 9 Kode program menghitung nilai matrik rasio

No	Kode
263	// menghitung isi metrik rasio
264	//mengisi nilai matrix omax rasio 1-5 indeks ke 0-11

```
// MATRIK RASIO 1
$jumlahskala = 12;
for ($i = 1; $i <= $jumlahskala; $i++) {
   if ($i== 7) { //matrik skala 3 (nilai dasar)
      $rasio[0][0] = $nilai_dasar[0];
      $simpan = $rasio[0][0];
      $rasio[$i][0] = round($simpan,2);
   } else if ($i==0) { // matrik skala 10 (nilai terbaik)
      $rasio[0][0] = $target_max[0];
      $simpan = $rasio[0][0];
      $rasio[$i][0] = round($simpan,2);
   } else if ($i==10) { // matrik skala 1 (nilai terburuk)
      $rasio[0][0] = $target_min[0];
      $simpan = $rasio[0][0];
      $rasio[$i][0] = round($simpan,2);
   }
}
$rasio[0][0] = $nilai_dasar[0]; // matrik skala 1-2
$simpan = $rasio[0][0];
for ($i = 8; $i <= 9; $i++) {
   $simpan = $simpan - $selisih1[0];
   $rasio[$i][0] = round($simpan,2);
}
$rasio[0][0] = $target_max[0]; // matrik skala 4-9
$simpan = $rasio[0][0];
for ($i = 1; $i <= 6; $i++) {
   $simpan = $simpan - $selisih2[0];
   $rasio[$i][0] = round($simpan,2);
}
```

Kode program pada Tabel 4.9 merupakan *function* pada tabel_matrix_omax() yang berfungsi untuk menentukan nilai matrik rasio satu pada setiap skala mulai dari skala paling rendah yaitu 0 sampai skala paling tinggi yaitu 10. Jumlah kriteria atau rasio adalah enam, maka kode diatas juga berlaku untuk menghitung atau mengisi nilai matrik pada rasio dua sampai enam, dengan mengganti rasio sesuai dengan yang dibutuhkan. Perhitungan produktivitas dilanjutkan pada Tabel 4.10.

Tabel 4. 10 Kode program penentuan skor

No	Kode
450	//5. Menentukan skala performansi masuk ke sakal rasio mana
451	//cek r1

```
452         if($r1<=$target_min[0]){
453         $index1=0;
454         }
455         else if($r1>=$target_max[0]){
456         $index1=10;
457         }
458         else{
459         for ($i=0; $i < $jumlahskala; $i++) {
460         $sim2 = round($rasio[$i][0],2);
461         $sim1 = round($rasio[$i+1][0],2);
462
463         $a=round($sim1,2);
464         while($a<=$sim2){
465         if($r1==$a){
466         if(abs($r1-$sim2)>abs($r1-$sim1)){
467            $in1 = $i+1;
468         }
469         if(abs($r1-$sim2)<abs($r1-$sim1)){
470            $in1 = $i;
471         }
472         if(abs($r1-$sim2)==abs($r1-$sim1)){
473            $in1 = $i+1;
474         }
475            switch ($in1) {
476               case 0 : $index1=10;
477                  break;
478               case 1 : $index1=9;
479                  break;
480               case 2 : $index1=8;
481                  break;
482               case 3 : $index1=7;
483                  break;
484               case 4 : $index1=6;
485                  break;
486               case 5 : $index1=5;
487                  break;
488               case 6 : $index1=4;
489                  break;
490               case 7 : $index1=3;
491                  break;
492               case 8 : $index1=2;
493                  break;
494               case 9 : $index1=1;
495                  break;
496               case 10 : $index1=0;
497                  break;
```

```
498             }
499         }
500         $a = round(($a+0.01),2);
501         }
502         }
503         }
```

Kode program pada Tabel 4.10 merupakan *function* pada tabel_matrix_omax() yang berfungsi untuk menentukan skor pada rasio satu. Jumlah kriteria atau rasio adalah enam, maka kode diatas juga berlaku untuk menentukan skor pada rasio dua sampai enam, dengan mengganti rasio sesuai dengan yang dibutuhkan. Perhitungan produktivitas dilanjutkan pada Tabel 4.11.

Tabel 4. 11 Kode program penentuan kategori skala

No	Kode
777	$arr = ["BURUK", "BURUK", "BURUK", "CUKUP BAIK", "CUKUP BAIK", "CUKUP BAIK", "BAIK", "BAIK", "BAIK", "SANGAT BAIK", "SANGAT BAIK"];

Kode program pada Tabel 4.11 merupakan *function* pada tabel_matrix_omax() yang berfungsi untuk menentukan kategori skala yang digunakan sebagai acuan untuk melihat faktor apa saja yang menyebabkan kenaikan atau penurunan produktivitas pada periode tertentu. Perhitungan produktivitas dilanjutkan pada Tabel 4.12.

Tabel 4. 12 Kode program menghitung indeks produktivitas

No	Kode
779	$bobot[$i] = $data_bobot[$i]['bobot_rasio'];
780	$indikatorPerformansi = (($bobot[0]*100)*$index1) + (($bobot[1]*100)*$index2) + (($bobot[2]*100)*$index3) + (($bobot[3]*100)*$index4) + (($bobot[4]*100)*$index5);
781	$indeksProduktivitas=round((($indikatorPerformansi-
782	$ip_sebelum)*100)/$ip_sebelum, 2) ."%";

Kode program pada Tabel 4.12 merupakan *function* pada tabel_matrix_omax(). Pada baris 779-780 berfungsi untuk menghitung indikator performansi, sedangkan pada baris 782 berfungsi untuk menghitung indeks produktivitas. Dari perhitungani Indeks Produktivitas (IP) dapat diketahui suatu produktivitas mengalami kenaikan atau penurunan. IP didapat dari perbandingan indikator performansi yang didapat pada periode saat ini dan periode sebelumnya.

4.6 Pengujian

Tahapan pengujian sistem merupakan suatu tahapan yang dilakukan secara sistematis untuk mengetahui ketidaksesuaian yang telah diimplementasikan dalam kode program dan rancangan sistem. Pengujian dilakukan dengan dua metode yaitu pengujian *White Box* dan *Black Box*. Hasil pengujian sebagai berikut:

4.6.1. Metode *White Box*

Pengujian sistem dengan metode *white box* dilakukan untuk menguji sistem dari segi desain dan kode program. Pengujian bertujuan untuk mengevaluasi apakah sistem mampu menghasilkan fungsi, masukan, dan keluaran yang sesuai dengan spesifikasi dari kebutuhan sistem. Tahapan pengujian metode *white box* meliputi:

a. *Listing* Program

Listing program adalah kumpulan dari baris-baris program yang akan diuji. Setiap langkah dari kode yang ada diberi nomor ketika menjalankan *statement* biasa atau penggunaan kondisi dalam program.

b. Diagram alir

Diagram alir merupakan notasi yang digunakan untuk merepresentasikan aliran kontrol. Aliran kontrol yang digambarkan merupakan hasil penomoran dari listing program. Diagram alir digambarkan dengan *node-node* (simpul) yang dihubungkan dengan *edge-edge* (garis) yang menggambarkan alur jalannya program.

c. Kompleksitas siklomatik (*cyclomatic complexity*)

Kompleksitas siklomatik merupakan metrik perangkat lunak yang menyediakan ukuran kuantitatif dari kompleksitas logis suatu program. Bila digunakan dalam konteks teknik pengujian jalur dasar, nilai yang dihitung untuk kompleksitas siklomatik mendefinisikan jumlah jalur independen dalam basis set suatu program. Perhitungan kompleksitas siklomatik menggunakan rumus sebagai berikut :

$$V(G) = E - N + 2$$

Keterangan :

V(G) : Kompleksitas siklomatik

E : Jumlah *edge*

N : Jumlah *node*

d. Jalur Independen (*Independent Path*)

Jalur independen adalah setiap jalur yang melalui program, menunjukkan setidaknya satu kumpulan pernyataan-pernyataan pemrosesan atau kondisi baru. Bila dinyatakan dalam grafik alir, jalur independen harus bergerak setidaknya sepanjang satu *edge* yang belum dilintasi sebelum jalur tersebut didefinisi

e. Pengujian Basis Set (*Test Case*)

Pada bagian ini diberikan contoh data yang menggambarkan pelaksanaan jalur di basis set. Data yang dieksekusi dimasukkan ke dalam grafik alir apakah sudah melewati basis set yang tersedia. Sistem telah memenuhi syarat kelayakan perangkat lunak jika salah satu jalur yang dieksekusi setidaknya satu kali.

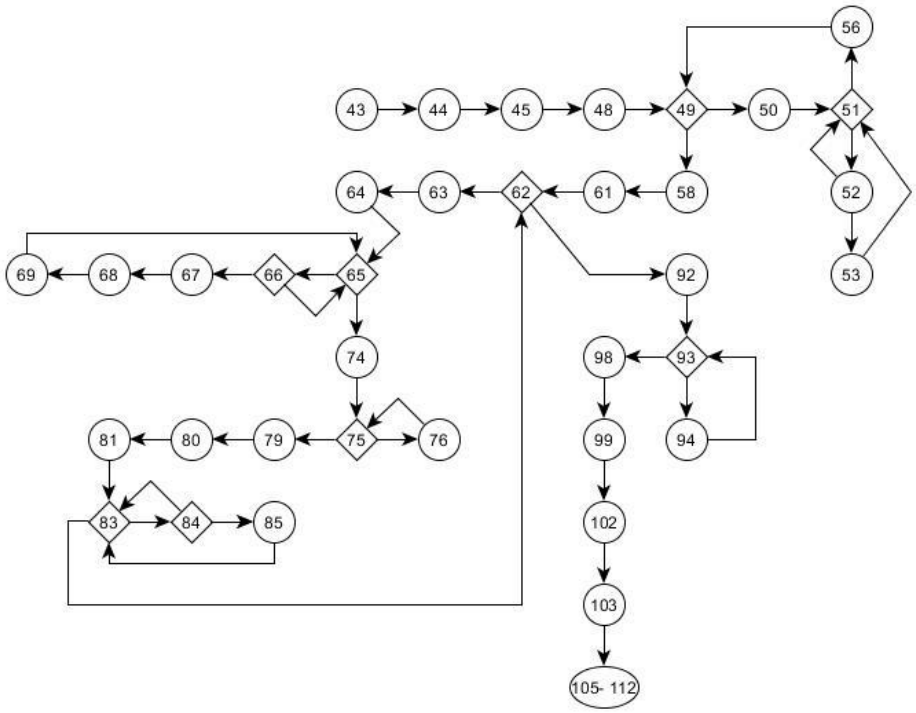
Pengujian sistem informasi pengukuran produktivitas hotel di kabupaten Jember menggunakan metode *objective matrix* dan AHP menggunakan metode *white box* akan diterapkan pada fitur yang dinilai dapat mewakili sistem informasi pengukuran produktivitas hotel sebagai berikut.

1) *Listing* Program Metode AHP

```
function bobot_kriteria_manager() {
    $data['daftar_kriteria']    = $this->M_produktivitas->get_kriteria()->result();
    $data['relasi_kriteria']    = $this->M_produktivitas->get_relasi_kriteria()->result();
    $data['jumlahKriteria']     = count($data['daftar_kriteria']);

 //Menghitung Jumlah dari setiap kolom
    $jumlah = array();
      foreach ($data['daftar_kriteria'] as $kriteria) {
          $jumlah1 = 0;
          foreach ($data['relasi_kriteria'] as $relasi) {
              if($relasi->kriteria_tujuan == $kriteria->id_rasio){
                  $jumlah1 = $jumlah1 + $relasi->bobot;
              }
          }
          array_push($jumlah, $jumlah1);
       }
      $data['jumlah_per_kolom'] = $jumlah;

 // normalisasi matrik
 //Menghitung nilai per cell (nilai cell / jumlah)
    $arrayVector = array();
    foreach ($data['daftar_kriteria'] as $kriteria) {
      $nilai = array();
      $a = 0;
      foreach ($data['relasi_kriteria'] as $relasi) {
          if($relasi->kriteria_dari == $kriteria->id_rasio){
            $nilai1 = $relasi->bobot / $jumlah[$a];
            array_push($nilai, $nilai1);
            $a++;
            }
      }
 // jumlah dari nilai matrik yang dinormalisasi
 //Menghitung nilai vector
      $vector = 0;
      for($i = 0; $i < count($nilai); $i++){
         $vector = $vector + $nilai[$i];
      }

      $vector = $vector / $data['jumlahKriteria'];
      $vector = number_format($vector, 3, '.', '');
      array_push($arrayVector, $vector);

    for ($i=0; $i  < 5 ; $i++) {
        if(isset($arrayVector[$i])){
            $this->M_produktivitas->update_bobotprioritas($arrayVector[$i], $i+1);
        }
    }

  }

//Menghitung nilai lamda
    $lamda = 0;
      for($i = 0; $i < count($arrayVector); $i++){
        $lamda = $lamda + ($arrayVector[$i] * $jumlah[$i]);
      }

//Menghitung Nilai CI
   $ci = ($lamda - $data['jumlahKriteria']) / ($data['jumlahKriteria'] - 1);
   $ci = number_format($ci, 3, '.', '');

//Menghitung Nilai CR
   $cr = $ci / $this->nilaiRI($data['jumlahKriteria']);
   $cr = number_format($cr, 3, '.', '');

 $data['nilai_lamda']    = $lamda;
 $data['nilai_ci']       = $ci;
 $data['nilai_cr']       = $cr;
 $data['nilai_vector']   = $arrayVector;
 $data['kriteria']=$this->M_produktivitas->get_kriteria();
 $this->load->view('manager/v_kriteria', $data);
 $this->load->view('manager/header_manager', $data);
 }
```

Gambar 4. 7 Listing program function bobot_kriteria_manager()

2) Diagram Alir



Gambar 4. 8 Diagram alir *function* bobot_kriteria_manager()

3) Kompleksitas Siklomatik (*Cyclomatic Complexity*)

Function bobot_kriteria()

V(G) = E - N +2

V(G) = 46 – 39 + 2 = 9

4) Pengujian Basis Set (*Test Case*)

Tabel 4. 13 *Test case function* bobot_kriteria_manager()

Test case function bobot_kriteria_manager()	
Jalur 1	
Test Case	Jika berhasil menghitung bobot prioritas masing-masing kriteria, menghitung nilai λmax, CI, CR
Target yang diharapkan	Menampilkan hasil perhitungan bobot prioritas kriteria hingga hasil pengujian konsistensi
Hasil Pengujian	Benar

Path / jalur	43-44-45-48-49-50-51-52-53-51-56-49-58-61-62-63-64-65-66-67-68-69-65-74-75-76-75-79-80-81-83-84-83-62-92-93-94-93-98-99-102-103-(105-112)
Jalur 2	
Test Case	Jika berhasil menghitung bobot prioritas masing-masing kriteria, menghitung nilai λmax, CI, CR.
Target yang diharapkan	Menampilkan hasil perhitungan bobot prioritas kriteria hingga hasil pengujian konsistensi
Hasil Pengujian	Benar
Path / jalur	43-44-45-48-49-50-51-52-53-51-56-49-58-61-62-63-64-65-66-67-68-69-65-74-75-76-75-79-80-81-83-84-83-62-92-93-94-93-98-99-102-103-(105-112)
Jalur 3	
Test Case	Jika berhasil menghitung bobot prioritas masing-masing kriteria, menghitung nilai λmax, CI, CR
Target yang diharapkan	Menampilkan hasil perhitungan bobot prioritas hingga hasil pengujian konsistensi
Hasil Pengujian	Benar
Path / jalur	43-44-45-48-49-50-51-52-53-51-56-49-58-61-62-92-93-94-93-98-99-102-103-(105-112)
Jalur 4	
Test Case	Jika berhasil menghitung bobot prioritas masing-masing kriteria, menghitung nilai λmax, CI, CR
Target yang diharapkan	Menampilkan hasil perhitungan bobot prioritas hingga hasil pengujian konsistensi
Hasil Pengujian	Benar
Path / jalur	43-44-45-48-49-50-51-52-53-51-56-49-58-61-62-92-93-98-99-102-103-(105-112)
Jalur 5	
Test Case	Jika berhasil menghitung bobot prioritas masing-masing kriteria, menghitung nilai λmax, CI, CR
Target yang diharapkan	Menampilkan hasil perhitungan bobot prioritas hingga hasil pengujian konsistensi
Hasil Pengujian	Benar
Path / jalur	43-44-45-48-49-50-51-52-53-51-56-49-58-61-62-63-64-65-66-65-74-75-76-75-79-80-81-83-84-83-62-92-93-94-93-98-99-102-103-(105-112)
Jalur 6	
Test Case	Jika berhasil menghitung bobot prioritas masing-masing kriteria, menghitung nilai λmax, CI, CR
Target yang diharapkan	Menampilkan hasil perhitungan bobot prioritas hingga hasil pengujian konsistensi
Hasil Pengujian	Benar

Path / jalur	43-44-45-48-49-50-51-52-53-51-56-49-58-61-62-63-64-65-74-75-76-75-79-80-81-83-84-83-62-92-93-94-93-98-99-102-103-(105-112)
Jalur 7	
Test Case	Jika berhasil menghitung bobot prioritas masing-masing kriteria, menghitung nilai λmax, CI, CR
Target yang diharapkan	Menampilkan hasil perhitungan bobot prioritas hingga hasil pengujian konsistensi
Path / jalur	43-44-45-48-49-50-51-52-53-51-56-49-58-61-62-63-64-65-66-67-68-69-65-74-79-80-81-83-84-83-62-92-93-94-93-98-99-102-103-(105-112)
Jalur 8	
Test Case	Jika berhasil menghitung bobot prioritas masing-masing kriteria, menghitung nilai λmax, CI, CR
Target yang diharapkan	Menampilkan hasil perhitungan bobot prioritas hingga hasil pengujian konsistensi
Hasil Pengujian	Benar
Path / jalur	43-44-45-48-49-50-51-52-53-51-56-49-58-61-62-63-64-65-66-67-68-69-65-74-75-76-75-79-80-81-83-62-92-93-94-93-98-99-102-103-(105-112)
Jalur 9	
Test Case	Jika berhasil menghitung bobot prioritas masing-masing kriteria, menghitung nilai λmax, CI, CR
Target yang diharapkan	Menampilkan hasil perhitungan bobot prioritas hingga hasil pengujian konsistensi
Hasil Pengujian	Benar
Path / jalur	43-44-45-48-49-50-51-52-53-51-56-49-58-61-62-63-64-65-66-67-68-69-65-74-75-76-75-79-80-81-83-84-83-62-92-93-98-99-102-103-(105-112)

4.6.2. Metode *Black Box*

Black box testing merupakan metode pengujian perangkat lunak yang memeriksa fungsionalitas dari aplikasi yang berkaitan dengan struktur internal atau kerja. Metode ini memfokuskan pada keperluan fungsionalitas dari software. Pengujian *black box* pada informasi pengukuran produktivitas hotel menggunakan metode *objective matrix* dan AHP dilakukan untuk mengetahui apakah masukan dan keluaran dari sistem sesuai dengan kebutuhan fungsional atau tidak. Pengujian dilakukan pada setiap *use case*. Hasil pengujian *black box* dapat dilihat pada Lampiran E (Pengujian *Black Box*).

BAB 6. PENUTUP

Bab ini berisi mengenai kesimpulan dan saran dari peneliti tentang penelitian yang telah dilakukan. Kesimpulan dan saran tersebut diharapkan dapat digunakan sebagai acuan pada penelitian selanjutnya.

6.1 Kesimpulan

Kesimpulan dari penelitian yang telah dilakukan adalah:

a. Penerapan metode *objective matrix* dan AHP yang digunakan pada sistem informasi pengukuran produktivitas hotel digunakan untuk memberikan nilai indeks produktivitas yang digunakan sebagai penentu peningkatan atau penurunan produktivitas. Langkah awal yakni menentukan nilai bobot prioritas setiap kriteria berdasarkan tingkat kepentingannya dengan menggunakan metode AHP. Selanjutnya dilakukan perhitungan produktivitas dengan memilih periode pengukuran, sehingga menghasilkan skala performansi dari setiap kriteria yang ada. Setelah skala performansi didapatkan, maka nilai tersebut akan diberi skor sesuai dengan matrik omax yang telah dibuat sebelumnya, kemudian skor tersebut akan dikalikan dengan bobot prioritas yang dihitung menggunakan metode AHP sehingga diperoleh nilai indikator performansi. Setelah indikator performansi didapat, maka dibandingkan dengan indikator performansi periode sebelumnya, sehingga diperoleh nilai indeks produktivitas. Nilai indeks produktivitas tersebut digunakan untuk mengetahui pada periode tersebut sedang mengalami peningkatan atau penurunan produktivitas. Selain itu juga dapat mengetahui faktor apa yang menyebabkan peningkatan maupun penurunan produktivitas tersebut. Perhitungan produktivitas pada bulan November 2016 menghasilkan nilai indeks produktivitas sebesar 7,8%. Hal tersebut berarti produktivitas hotel mengalami peningkatan sebesar 7,8% dibanding bulan Oktober 2016.

b. Implementasi metode *objective matrix* pada sistem ini menggunakan lima rasio yaitu rasio antara jumlah kamar yang dihuni dan jumlah kamar yang tersedia, rasio antara jumlah kamar yang dibersihkan dengan jumlah hari perawatan kamar, rasio antara jumlah porsi *food & beverage* dan jumlah pengadaan bahan baku, rasio antara jumlah absen karyawan dan jumlah karyawan serta rasio jumlah perbaikan oleh *engineering* dan jumlah kerusakan sarana. Kelima rasio yang telah ditentukan tersebut berdasarkan pengumpulan data dan studi literatur yang telah ditentukan bobotnya berdasarkan skala kepentingan masing-masing rasio dengan menggunakan metode AHP.
c. Sistem informasi pengukuran produktivitas hotel dengan metode *objective matrix* dan AHP ini mampu menghasilkan nilai skala performansi, kategori skala, indikator performansi dan indeks produktivitas pada setiap periode, sehingga dapat membantu pihak manajemen hotel dalam mengontrol produktivitas, serta dapat mengetahui faktor apa yang menyebabkan peningkatan ataupun penurunan produktivitas sehingga evaluasi dapat dilakukan pada faktor yang menyebabkan penurunan produktivitas pada periode tersebut.

6.2 Saran

Adapun saran yang ditujukan untuk memberikan masukan yang lebih baik yaitu :

a. Sistem informasi pengukuran produktivitas hotel yang akan dikembangkan selanjutnya akan lebih baik jika jumlah kriteria bisa ditambahkan melalui sistem agar bisa menyesuaikan dengan kebutuhan pihak hotel.
b. Sistem informasi pengukuran produktivitas hotel yang akan dikembangkan selanjutnya akan lebih baik jika data aktual pencapaian hotel selama periode tertentu dapat diubah melalui sistem sehingga metode *objective matrix* pada sistem ini dapat diterapkan pada hotel manapun.

DAFTAR PUSTAKA

Agustina, F., & Riana, N. A. (2011, Mei 14). Analisis Produktivitas dengan Metode Objective Matrix (OMAX) di PT.X. *Jurnal Teknik dan Manajemen Industri*.

Brojonegoro PS, B. (1992). *AHP (the Analytical Hierarchy Process).* Pusat Antar University – Studi Ekonomi Universitas Indonesia.

Fiati, R. (2015). Pendekatan Model Objective Matrix-AHP untuk Pengambilan Keputusan Penilaian Kinerja Pelayanan pada Kantor Kelurahan. *Jurnal Teknik Informatika UMK*.

Fithri, P., & Firdaus, I. (2014). Analisis Produktivitas Menggunakan Metode Objective Matrix (OMAX) (Studi Kasus: PT. Moradon Berlian Sakti). *Jurnal Optimasi sistem Industri*.

Hamidah, N., & Deoranto, P. (2012). Analisis Produktivitas Menggunakan Metode Objective Matrix (OMAX) (Studi Kasus Pada Bagian Produksi Sari Roti PT Nippon Indosari Corpindo, Tbk. *Jurnal Teknologi Industri*.

Pressman, R. S. (2001). *Software Engineering a practitioner's approach* (5 ed.). New York, America: McGraw-Hill.

Saaty. (1994). *Analytical Hierarchy Process.* Bandung: Alfabeta.

Silalahi, d. (2014). Usulan Strategi Peningkatan Produktivitas Berdasarkan Hasil Analisis Pengukuran Objective Matrix (OMAX) pada Departemen Produksi Transformer. *Jurusan Teknik Industri Itenas*.

Sommerville, I. (2001). *Software Engineering.* Addison Wesley.

LAMPIRAN

LAMPIRAN A. *USE CASE* SKENARIO

A.1 *Use Case* Skenario *Login*

Tabel A. 1 *Use case* skenario *login*

Nama *use case*	*Login*
Aktor	Manager, *front office*, HRD, *housekeeping, engineering, food & beverage*
Pre Kondisi	Aktor akan melakukan *login*
Post Kondisi	Aktor telah melakukan *login*
Skenario Normal	
Aksi Aktor	**Reaksi sistem**
1. Pilih tombol *login*	
	2. Menampilkan halaman *login*
3. Memasukkan *username*	
4. Memasukkan *password*	
5. Pilih tombol *login* Jika batal lanjut ke nomor 5c	
	6. Memeriksa data *login* ke *database*
	7. Menampilkan halaman awal sistem sesuai dengan level aktor masing-masing
Skenario alternatif (*username* atau *password* tidak sesuai)	
Aksi Aktor	**Reaksi sistem**
	7a. Menampilkan pesan *username* atau *password* salah!
Skenario alternatif (*username* atau *password* kosong/tidak diisi)	
Aksi Aktor	**Reaksi sistem**
	7b. Menampilkan pesan "*please fill out this field*" pada pada kolom yang belum diisi
Skenario Alternatif – Batal	
Aksi Aktor	**Reaksi sistem**
5c. Pilih tombol batal	
	6c. Menampilkan halaman utama

A.2 *Use Case* Skenario *View* Data Pegawai

Tabel A. 2 *Use case* skenario *view* data pegawai

Nama *Use Case*	*View* data pegawai
Aktor	Manager
Pre Kondisi	Aktor akan melihat data pegawai
Post Kondisi	Aktor telah melihat data pegawai
Skenario Normal – *View*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data pegawai	
2. Pilih sub menu daftar pegawai	
	3. Mengambil data pegawai dari *database* yaitu id pegawai, nama, alamat, departemen dan tanggal masuk kerja dari *database*
	4. Menampilkan halaman data pegawai dengan tombol detail
5. Klik tombol detail pada data pegawai	
	6. Menampilkan detail pegawai sesuai data yang dipilih

A.3 *Use Case* Skenario *View* Absensi

Tabel A. 3 *Use case* skenario *view* absensi

Nama *Use Case*	*View* absensi
Aktor	Manager, HRD
Pre Kondisi	Aktor akan melihat absensi pegawai
Post Kondisi	Aktor telah melihat absensi pegawai
Skenario Normal – *View*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data pegawai	
2. Pilih sub menu absensi pegawai	
	3. Mengambil data absensi dari *database* yaitu tanggal, nama, keterangan, jam datang dan jam pulang
	4. Menampilkan halaman absensi pegawai

A.4 *Use Case* Skenario *View* Data Kamar

Tabel A. 4 *Use case* skenario *view* data kamar

Nama *Use Case*	*View* data kamar
Aktor	*Front office*
Pre Kondisi	Aktor akan melihat data kamar
Post Kondisi	Aktor telah melihat data kamar
Skenario Normal – *View*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data kamar	
2. Pilih sub menu daftar kamar	
	3. Mengambil data daftar kamar dari *database* yaitu tipe kamar, nomor kamar dan status kamar
	4. Menampilkan halaman daftar kamar

A.5 *Use Case* Skenario *View* Data Tipe Kamar

Tabel A. 5 *Use case* skenario *view* data tipe kamar

Nama *Use Case*	*View* data tipe kamar
Aktor	*Front office*
Pre Kondisi	Aktor akan melihat data tipe kamar
Post Kondisi	Aktor telah melihat data tipe kamar
Skenario Normal – *View*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data kamar	
2. Pilih sub menu tipe kamar	
	3. Mengambil data tipe kamar dari *database* yaitu id tipe kamar, tipe kamar dan harga per malam
	4. Menampilkan halaman tipe kamar

A.6 *Use Case* Skenario *View* Data Komplain

Tabel A. 6 *Use case* skenario *view* data komplain

Nama *Use Case*	*View* data komplain
Aktor	Manager, *front office, housekeeping, engineering, food & beverage*
Pre Kondisi	Aktor akan melihat data komplain

Post Kondisi	Aktor telah melihat data komplain
Skenario Normal – *View* komplain Belum/Tidak Teratasi	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data komplain	
	2. Mengambil data komplain yaitu tanggal, departemen, komplain dan status dari *database*
	3. Menampilkan halaman data komplain
4. Pilih tab belum/tidak teratasi	
	5. Menampilkan data komplain yang belum atau tidak teratasi
Skenario Normal – *View* komplain Teratasi	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data komplain	
	2. Mengambil data komplain yaitu tanggal, departemen, komplain dan status dari *database*
	3. Menampilkan halaman data komplain
4. Pilih tab teratasi	
	5. Menampilkan data komplain yang teratasi
6. Pilih tombol tinjau pada data komplain yang akan ditinjau	
	7. Menampilkan detail komplain sesuai data yang dipilih

A.7 *Use Case* Skenario *View* Data *Check in*

Tabel A. 7 *Use case* skenario *view* data *check in*

Nama *Use Case*	*View* data *check in*
Aktor	Manager
Pre Kondisi	Aktor akan melihat data *check in*
Post Kondisi	Aktor telah melihat data *check in*
Skenario Normal – *View*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data reservasi	
2. Pilih sub menu *check in*	
	3. Mengambil data *check in* yaitu nama, nomor identitas, tanggal *check in*, nomor kamar, tanggal *check out* dari *database*

	4. Menampilkan halaman data *check in*
5. Klik tombol detail pada data *check in*	
	6. Menampilkan halaman detail *chek in* sesuai dengan data yang dipilih

A.8 *Use Case* Skenario *View* Data *Check out*

Tabel A. 8 *Use case* skenario *view* data *check out*

Nama *Use Case*	*View* data *check out*
Aktor	Manager, *front office*
Pre Kondisi	Aktor akan melihat data *check out*
Post Kondisi	Aktor telah melihat data *check out*
Skenario Normal – *View*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data reservasi	
2. Pilih sub menu *check out*	
	3. Mengambil data *check out* yaitu nama, nomor identitas, tanggal *check in*, nomor kamar, tanggal *check out* dari *database*
	4. Menampilkan halaman data *check out*
5. Klik tombol detail pada data *check out*	
	6. Menampilkan halaman detail *chek out* sesuai dengan data yang dipilih

A.9 *Use Case* Skenario *View* Pembobotan Kriteria

Tabel A. 9 *Use case* skenario view pembobotan kriteria

Nama *Use Case*	*View* Pembobotan kriteria
Aktor	Manager
Pre Kondisi	Aktor akan melihat pembobotan kriteria
Post Kondisi	Aktor telah melihat pembobotan kriteria
Skenario Normal – *View*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu indikator pengukuran	

	2. Mengambil data bobot kriteria berupa tabel nilai perbandingan antar kriteria dari *database*
	3. Menampilkan halaman indikator pengukuran

A.10 *Use Case* Skenario *Update* Bobot Kriteria

Tabel A. 10 *Use case* skenario *update* bobot kriteria

Nama *Use Case*	*Update* bobot kriteria
Aktor	Manager
Pre Kondisi	Aktor akan mengupdate bobot kriteria
Post Kondisi	Aktor telah mengupdate bobot kriteria
Skenario Normal – *Update*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu indikator pengukuran	
	2. Menampilkan halaman indikator pengukuran dengan tombol *update*
3. Pilih tombol *update*	
	4. Menampilkan halaman *update* bobot kriteria
5. Mengubah nilai kepentingan antar kriteria di kolom kriteria	
6. Pilih simpan Jika batal lanjut ke nomor 6a.	
	7. Mengupdate bobot kriteria pada *database*
	8. Menampilkan halaman indikator penilaian dengan bobot yang sudah diupdate
Skenario Alternatif – Batal	
Aksi Aktor	**Reaksi Sistem**
6a. Pilih tombol batal	
	7a. Menampilkan halaman indikator Penilaian

A.11 *Use Case* Skenario *View* Produktivitas

Tabel A. 11 *Use case* skenario *view* produktivitas

Nama *Use Case*	*View* produktivitas
Aktor	Manager

Pre Kondisi	Aktor akan melihat produktivitas
Post Kondisi	Aktor telah melihat produktivitas
Skenario Normal – *View*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu produktivitas	
	2. Mengambil data produktivitas yaitu bulan, tahun, indikator performansi dan ideks produktivitas dari *database*
	3. Menampilkan halaman produktivitas dengan tombol detail produktivitas pada setiap baris data
4. Pilih tombol detail produktivitas pada data produktivitas	
	5. Menampilkan halaman detail produktivitas sesuai dengan data yang dipilih

A.12 *Use Case* Skenario *Input* Perhitungan Produktivitas

Tabel A. 12 *Use case* skenario *input* perhitungan produktivitas

Nama *Use Case*	*Input* Perhitungan Produktivitas
Aktor	Manager
Pre Kondisi	Aktor akan menginputkan perhitungan produktivitas
Post Kondisi	Aktor telah menginputkan perhitungan produktivitas
Skenario Normal – *Create*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu produktivitas	
	2. Mengambil data produktivitas yaitu bulan, tahun, indikator performansi dan ideks produktivitas dari *database*
	3. Menampilkan halaman produktivitas dengan tombol tambah data
4. Pilih tombol tambah data	
	5. Menampilkan halaman hitung produktivitas
6. Pilih bulan dan tahun	
7. Pilih tombol tampilkan	

	8. mengambil data jumlah kamar yang dihuni, jumlah kamar tersedia, jumlah hari perawatan kamar, jumlah komplain yang ditangani F&B, jumlah komplain F&B, jumlah absen karyawan, jumlah karyawan, jumlah komplain yang ditangani *engineering*, dan jumlah komplain *engineering* pada *database*
	9. Tetap pada halaman hitung produktivitas dengan menampilkan nilai pada setiap kolom yang ada, Yaitu : jumlah kamar yang dihuni, jumlah kamar tersedia, jumlah hari perawatan kamar, jumlah komplain yang ditangani F&B, jumlah komplain F&B, jumlah absen karyawan, jumlah karyawan, jumlah komplain yang ditangani *engineering*, dan jumlah komplain *engineering*
10. Pilih tombol hitung Jika batal lanjut ke nomor 9a	
	11. Menyimpan data produktivitas ke *database*
	12. Menampilkan halaman produktivitas
Skenario Alternatif – Data Tidak Tersedia	
Aksi Aktor	**Reaksi Sistem**
6a. Pilih bulan dan tahun	
7a. Pilih tombol tampilkan	
	8a. Menampilkan pesan “Data tidak ditemukan, pilih bulan dan tahun yang lain”
Skenario Alternatif – Batal	
Aksi Aktor	**Reaksi Sistem**
9a. Pilih tombol batal	
	10a. Menampilkan halaman produktivitas

A.13 *Use Case* Skenario Mengelola Data *User*

Tabel A. 13 *Use case* skenario mengelola data *user*

Nama *Use Case*	Mengelola Data *User*
Aktor	HRD
Pre Kondisi	Aktor akan mengelola data *user*
Post Kondisi	Aktor telah mengelola data *user*
Skenario Normal – *Create*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu manajemen *user*	
	2. Mengambil data *user* dari *database*
	3. Menampilkan halaman manajemen *user* dan tombol tambah data
4. Pilih tombol tambah data	
	5. Menampilkan halaman tambah *user*
6. Mengisi kolom id pegawai kemudian tekan enter	
	7. Mencari nama pegawai dan departemen pada *database*
	8. Menampilkan isi kolom nama pegawai dan departemen sesuai dengan id pegawai yang dimasukkan
9. Mengisi kolom *username*, *password* dan *e-mail*	
10. Pilih tombol simpan jika batal lanjut ke nomor 10a	
	11. Menyimpan data *user* ke *database*
	12. Menampilkan halaman manajemen *user*
Skenario Normal – *Accept*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu manajemen *user*	
	2. Mengambil data user dari *database*
	3. Menampilkan halaman manajemen user dan tombol centang pada setiap baris data
4. Pilih tombol centang pada data user	

	5. Menampilkan peringatan "Anda yakin ingin mengkonfirmasi user?" dengan pilihan tombol *ok* dan *cancel*
6. Pilih tombol *ok* Jika cancel lanjut ke nomor 6b	
	7. Mengupdate status user menjadi aktif pada *database*
	8. Menampilkan halaman manajemen user
Skenario Alternatif – Batal	
Aksi Aktor	**Reaksi Sistem**
10a. Pilih tombol batal	
	11a. Menampilkan halaman manajemen *user*
Skenario Alternatif – *Cancel*	
Aksi Aktor	**Reaksi Sistem**
6b. Pilih tombol *cancel*	
	7a. Menampilkan halaman tambah user
Skenario Alternatif (kolom kosong/tidak diisi)	
Aksi Aktor	**Reaksi Sistem**
10c. Pilih tombol simpan	
	11c. Menampilkan pesan "*please fill out this field*" dengan menunjuk pada kolom yang belum diisi

A.14 *Use Case* Skenario Mengelola Data Tipe Kamar

Tabel A. 14 *Use case* skenario mengelola data tipe kamar

Nama *Use Case*	Mengelola Data Tipe Kamar
Aktor	Manager
Pre Kondisi	Aktor akan mengelola data tipe kamar
Post Kondisi	Aktor telah mengelola data tipe kamar
Skenario Normal – *Create*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data kamar	
2. Pilih sub menu tipe kamar	
	3. Mengambil data tipe kamar dari *database* yaitu id tipe kamar, tipe kamar dan harga per malam
	4. Menampilkan halaman tipe kamar dan tombol tambah data

5. Pilih tombol tambah data	
	6. Menampilkan form tambah tipe kamar
7. Mengisi form tambah tipe kamar seperti : tipe kamar dan harga/malam	
8. Pilih tombol simpan Jika batal lanjut ke nomer 8a	
	9. Menyimpan data tipe kamar ke *database*
	10. Menampilkan halaman tipe kamar
Skenario Normal – *Edit*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data kamar	
2. Pilih sub menu tipe kamar	
	3. Mengambil data tipe kamar dari *database* yaitu id tipe kamar, tipe kamar dan harga per malam
	4. Menampilkan halaman tipe kamar dan tombol edit pada setiap baris data
5. Pilih tombol edit pada tipe kamar yang akan diperbaharui	
	6. Menampilkan form edit tipe kamar dengan tombol simpan dan batal
7. Mengubah form tambah tipe kamar seperti : tipe kamar dan harga/malam	
8. Pilih tombol simpan Jika batal lanjut ke nomer 8a	
	9. Mengupdate data tipe kamar pada *database*
	10. Menampilkan halaman tipe kamar yang berhasil diupdate.
Skenario Normal – *View*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data kamar	
2. Pilih sub menu tipe kamar	
	3. Mengambil data tipe kamar dari *database* yaitu id tipe kamar, tipe kamar dan harga per malam
	4. Menampilkan halaman tipe kamar

Skenario Normal – *Delete*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data kamar	
2. Pilih sub menu tipe kamar	
	3. Mengambil data tipe kamar dari *database* yaitu id tipe kamar, tipe kamar dan harga per malam
	4. Menampilkan halaman tipe kamar dan tombol *delete* pada setiap baris data
5. Pilih tombol *delete* pada tipe kamar yang akan dihapus	
	6. Menampilkan peringatan "Anda yakin ingin menghapus data?" dengan pilihan tombol *ok* dan *cancel*
7. Pilih tombol *ok* Jika cancel lanjut ke nomor 7b	
	8. Mengupdate status data tipe kamar yang dipilih menjadi hapus pada *database*
	9. Menampilkan halaman tipe kamar yang berhasil dihapus
Skenario Alternatif – Batal	
Aksi Aktor	**Reaksi Sistem**
8a. Pilih tombol batal	
	9a. Menampilkan halaman tipe kamar
Skenario Alternatif – *Cancel*	
Aksi Aktor	**Reaksi Sistem**
7b. Pilih tombol *cancel*	
	8b. Menampilkan halaman tipe kamar
Skenario Alternatif (kolom kosong/tidak diisi)	
Aksi Aktor	**Reaksi Sistem**
8c. Pilih tombol simpan	
	9c. Menampilkan pesan "*please fill out this field*" dengan menunjuk pada kolom yang belum diisi

A.15 *Use Case* Skenario Mengelola Data Daftar Kamar

Tabel A. 15 *Use case* skenario mengelola data daftar kamar

Nama *Use Case*	Mengelola Data Daftar Kamar

Aktor	Manager
Pre Kondisi	Aktor akan mengelola data daftar kamar
Post Kondisi	Aktor telah mengelola data daftar kamar
Skenario Normal – *Create*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data kamar	
2. Pilih sub menu daftar kamar	
	3. Mengambil data daftar kamar dari *database* yaitu tipe kamar, nomor kamar dan status kamar
	4. Menampilkan halaman daftar kamar dan tombol tambah data
5. Pilih tombol tambah data	
	6. Menampilkan form tambah daftar kamar
7. Mengisi form tambah daftar kamar seperti : tipe kamar dan nomor kamar	
8. Pilih tombol simpan Jika batal lanjut ke nomer 8a	
	9. Menyimpan data daftar kamar ke *database*
	10. Menampilkan halaman daftar kamar
Skenario Normal – *Edit*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data kamar	
2. Pilih sub menu tipe kamar	
	3. Mengambil data daftar kamar dari *database* yaitu tipe kamar, nomor kamar dan status kamar
	4. Menampilkan halaman tipe kamar dengan tombol edit pada setiap baris data
5. Pilih tombol edit pada tipe kamar yang akan diperbaharui	
	6. Menampilkan form edit daftar kamar dengan tombol simpan dan batal
7. Mengubah form tambah tipe kamar seperti : tipe kamar dan nomor kamar	
8. Pilih tombol simpan Jika batal lanjut ke nomer 8a	

	9. Mengupdate data daftar kamar pada *database*
	10. Menampilkan halaman daftar kamar yang berhasil diupdate.
Skenario Normal – *View*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data kamar	
2. Pilih sub menu daftar kamar	
	3. Mengambil data daftar kamar dari *database* yaitu tipe kamar, nomor kamar dan status kamar
	4. Menampilkan halaman daftar kamar
Skenario Normal – *Delete*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data kamar	
2. Pilih sub menu daftar kamar	
	3. Mengambil data daftar kamar dari *database* yaitu tipe kamar, nomor kamar dan status kamar
	4. Menampilkan halaman daftar kamar dengan tombol *delete* pada setiap baris data
5. Pilih tombol *delete* pada daftar kamar yang akan dihapus	
	6. Menampilkan peringatan “Anda yakin ingin menghapus data?” dengan pilihan tombol *ok* dan *cancel*
7. Pilih tombol *ok* Jika cancel lanjut ke nomor 7b	
	8. Mengupdate status data daftar kamar yang dipilih menjadi hapus pada *database*
	9. Menampilkan halaman daftar kamar yang berhasil dihapus
Skenario Alternatif – Batal	
Aksi Aktor	**Reaksi Sistem**
8a. Pilih tombol batal	
	9a. Menampilkan halaman daftar kamar
Skenario Alternatif – *Cancel*	
Aksi Aktor	**Reaksi Sistem**
7b. Pilih tombol *cancel*	

	8b. Menampilkan halaman daftar kamar
Skenario Alternatif (kolom kosong/tidak diisi)	
Aksi Aktor	**Reaksi Sistem**
8c. Pilih tombol simpan	
	9c. Menampilkan pesan "*please fill out this field*" dengan menunjuk pada kolom yang belum diisi

A.16 *Use Case* Skenario Mengelola Data Reservasi

Tabel A. 16 *Use case* skenario mengelola data reservasi

Nama *Use Case*	Mengelola Data Reservasi
Aktor	*Front office*
Pre Kondisi	Aktor akan mengelola data reservasi
Post Kondisi	Aktor telah mengelola data reservasi
Skenario Normal – *Create*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu reservasi	
2. Pilih sub menu *check in*	
	3. Mengambil data *check in* yaitu nama, nomor identitas, tanggal *check in*, nomor kamar, tanggal *check out* dari *database*
	4. Menampilkan halaman daftar *check in* dan tombol tambah data
5. Pilih tombol tambah data	
	6. Menampilkan form *input check in*
7. Mengisi form input check in seperti : nama lengkap, nomor identitas, alamat, nomor telpon. Memilih tipe kamar	
8. Tekan *enter*	
	9. Menampilkan nomor kamar yang tersedia berdasarkan tipe kamar yang dipilih pada kolom nomor kamar dan menampilkan biaya pada kolom biaya
10. Mengisi tanggal *check out* dan jumlah hari	
11. Pilih tombol simpan	

Jika batal lanjut ke nomer 11a	
	12. Menyimpan data *check in* ke *database*
	13. Menampilkan halaman daftar *check in*
Skenario Normal – *Edit*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu reservasi	
2. Pilih sub menu *check in*	
	3. Mengambil data *check in* yaitu nama, nomor identitas, tanggal *check in*, nomor kamar, tanggal *check out* dari *database*
	4. Menampilkan halaman daftar *check in* dan tombol edit pada setiap baris data
5. Pilih tombol edit pada data *check in* yang akan diperbaharui	
	6. Menampilkan form edit *check in* dengan tombol simpan dan batal
7. Mengubah form edit *check in* seperti : nama lengkap, nomor identitas, alamat, nomor telpon, tipe kamar, nomor kamar, harga/malam, tanggal check out dan jumlah hari	
8. Pilih tombol simpan Jika batal lanjut ke nomer 11a	
	9. Mengupdate data *check in* di *database*
	10. Menampilkan halaman daftar *check in* yang berhasil diubah
Skenario Normal – *View*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu reservasi	
2. Pilih sub menu *check in*	
	3. Mengambil data *check in* yaitu nama, nomor identitas, tanggal *check in*, nomor kamar, tanggal *check out* dari *database*
	4. Menampilkan halaman daftar *check in*
5. Pilih tombol detail pada daftar *check in* yang dipilih	

	6. Menampilkan halaman detail *check in* sesuai dengan data yang dipilih
Skenario Normal – *Check out*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu reservasi	
2. Pilih sub menu *check in*	
	3. Mengambil data *check in* yaitu nama, nomor identitas, tanggal *check in*, nomor kamar, tanggal *check out* dari *database*
	4. Menampilkan halaman daftar *check in* dan tombol *check out* pada setiap baris data
5. Pilih tombol *check out* pada data yang dipilih untuk melakukan *check out*	
	6. Menampilkan peringatan "Anda yakin ingin *check out*?" dengan pilihan tombol *ok* dan *cancel*
7. Pilih *ok* Jika *cancel* lanjut ke nomor 7b	
	8. Mengupdate status *check in* menjadi *check out* pada *database*
	9. Menampilkan halaman daftar *check in*
Skenario Alternatif – Batal	
Aksi Aktor	**Reaksi Sistem**
11a. Pilih tombol batal	
	11a. Menampilkan halaman daftar *check in*
Skenario Alternatif – *Cancel*	
Aksi Aktor	**Reaksi Sistem**
7b. Pilih tombol cancel	
	8b. Menampilkan halaman daftar check in
Skenario Alternatif (kolom kosong/tidak diisi)	
Aksi Aktor	**Reaksi Sistem**
8c. Pilih tombol simpan	
	9c. Menampilkan pesan "*please fill out this field*" dengan menunjuk pada kolom yang belum diisi

A.17 *Use Case* Skenario *Input* Absensi Pegawai

Tabel A. 17 *Use case* skenario *input* absensi pegawai

Nama *Use Case*	*Input* Absensi Pegawai
Aktor	HRD
Pre Kondisi	Aktor akan menginputkan absensi pegawai
Post Kondisi	Aktor telah menginputkan absensi pegawai
Skenario Normal – *Create*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu absensi pegawai	
	2. Mengambil data absensi pegawai dari *database*
	3. Menampilkan halaman absensi pegawai dan tombol *input* absensi
4. Pilih tombol *input* absensi	
	5. Menampilkan halaman *input* absensi
6. Mengisi kolom id pegawai	
7. Tekan *enter*	
	8. Mencari nama pegawai pada *database* sesuai dengan id pegawai yang diinputkan
	9. Menampilkan nama pegawai pada kolom nama pegawai
10. Mengisi kolom keterangan, jam datang dan jam pulang	
11. Pilih tombol simpan Jika batal lanjut ke nomor 11a	
	12. Menyimpan data absensi pegawai ke *database*
	13. Menampilkan halaman absensi pegawai
Skenario Alternatif – Batal	
Aksi Aktor	**Reaksi Sistem**
11a. Pilih tombol batal	
	12a. Menampilkan halaman absensi pegawai
Skenario Alternatif (id pegawai tidak ditemukan)	
Aksi Aktor	**Reaksi Sistem**
6b. Mengisi kolom id pegawai	
7b. Tekan enter	
	8b. Menampilkan pesan “data tidak ditemukan”

Skenario Alternatif (kolom kosong/tidak diisi)	
Aksi Aktor	**Reaksi Sistem**
10c. Pilih tombol simpan	
	11c. Menampilkan pesan "*please fill out this field*" dengan menunjuk pada kolom yang belum diisi

A.18 *Use Case* Skenario *Input* Penanganan Komplain

Tabel A. 18 *Use case* skenario *input* penanganan komplain

Nama *Use Case*	*Input* penanganan komplain
Aktor	*front office, housekeeping, engineering, food & beverage*
Pre Kondisi	Aktor akan menginputkan penanganan komplain
Post Kondisi	Aktor telah menginputkan penanganan komplain
Skenario Normal – *Create*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih menu data komplain	
	2. Mengambil data komplain yaitu tanggal, departemen, komplain dan status dari *database*
	3. Menampilkan halaman data komplain
4. Pilih tab belum/tidak teratasi	
	5. Menampilkan data komplain yang belum atau tidak teratasi
6. Pilih tombol penanganan pada komplain yang akan ditangani	
	7. Menampilkan halaman penanganan komplain sesuai dengan data yang dipilih
8. Memilih *radio button* teratasi Jika tidak teratasi lanjut ke nomor 8a	
9. Mengisi kolom penanganan dan hasil	
10. Memilih tombol simpan jika batal lanjut ke nomor 10b	
	11. Mengupdate status komplain menjadi teratasi
	12. Menampilkan halaman data komplain

Skenario Alternatif – Tidak Teratasi	
Aksi Aktor	**Reaksi Sistem**
8a. Memilih *radio button* tidak teratasi	
9a. Pilih tombol simpan	
	10a. Mengupdate status komplain menjadi tidak teratasi
	11a. Menampilkan halaman data komplain
Skenario Alternatif – Batal	
Aksi Aktor	**Reaksi Sistem**
10b. Pilih tombol batal	
	11b. Menampilkan halaman data komplain
Skenario Alternatif (kolom kosong/tidak diisi)	
Aksi Aktor	**Reaksi Sistem**
10c. Pilih tombol simpan	
	11c. Menampilkan pesan "*please fill out this field*" dengan menunjuk pada kolom yang belum diisi

A.19 *Use Case* Skenario *Input* Komplain

Tabel A. 19 *Use case* skenario *input* komplain

Nama *Use Case*	*Input* komplain
Aktor	*Customer*
Pre Kondisi	Aktor akan menginputkan komplain
Post Kondisi	Aktor telah menginputkan komplain
Skenario Normal – *Create*	
Aksi Aktor	**Reaksi Sistem**
1. Pilih tombol komplain	
	2. Menampilkan halaman input komplain
3. Mengisi kolom nomor identitas *customer* saat melakukan *check in*	
4. Tekan *enter*	
	5. Menampilkan nama tamu pada kolom nama tamu dan nomor kamar tamu pada kolom nomor kamar
6. Memilih departemen tujuan komplain dan mengisi isi komplain	
7. Pilih tombol simpan Jika batal lanjut ke nomor 7a	

	8. Menyimpan data komplain ke *database*
	9. Menampilkan halaman awal sistem
Skenario Alternatif – Batal	
Aksi Aktor	**Reaksi Sistem**
7a. Pilih tombol batal	
	8a. Menampilkan halaman utama sistem
Skenario Alternatif (kolom kosong/tidak diisi)	
Aksi Aktor	**Reaksi Sistem**
7b. Pilih tombol simpan	
	8b. Menampilkan pesan "*please fill out this field*" dengan menunjuk pada kolom yang belum diisi

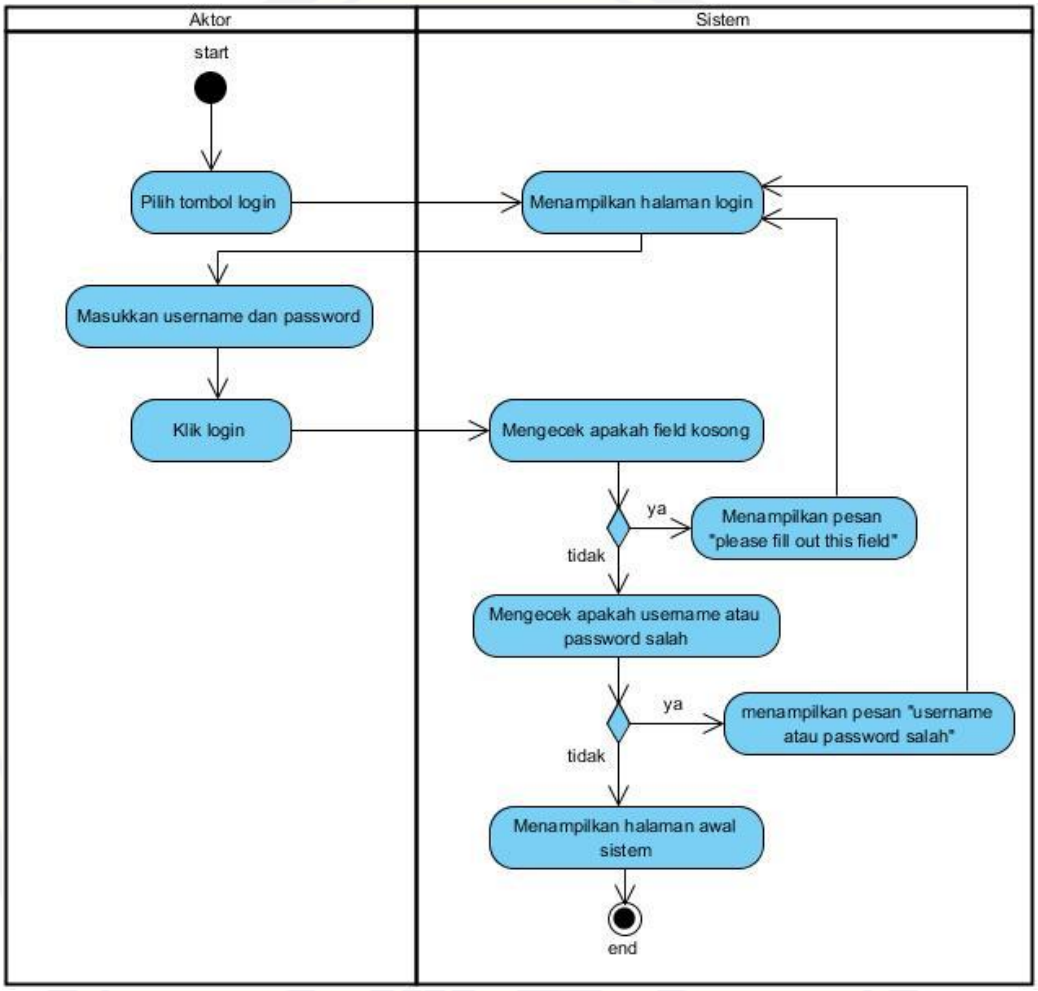
A.20 *Use Case* Skenario *Logout*

Tabel A. 20 *Use case* skenario *logout*

Nama *Use Case*	*Logout*
Aktor	Manager, *front office*, HRD, *housekeeping, engineering, food & beverage*
Pre Kondisi	Aktor akan melakukan *logout*
Post Kondisi	Aktor telah melakukan *logout*
Skenario Normal	
Aksi Aktor	**Reaksi Sistem**
1. Pilih tombol *logout*	
	2. Menampilkan halaman utama sistem

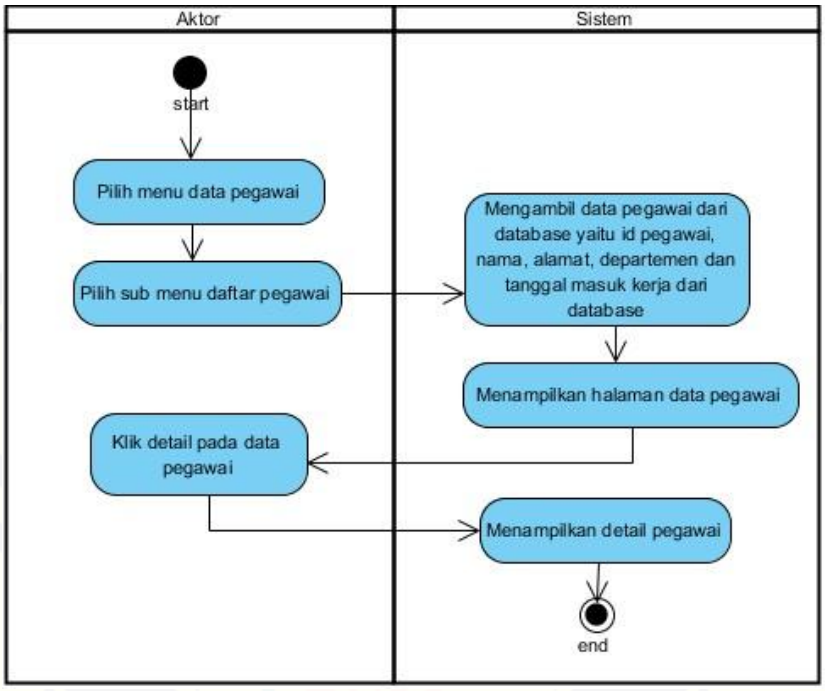
LAMPIRAN B. *ACTIVITY* DIAGRAM

B.1 *Activity* Diagram *Login*



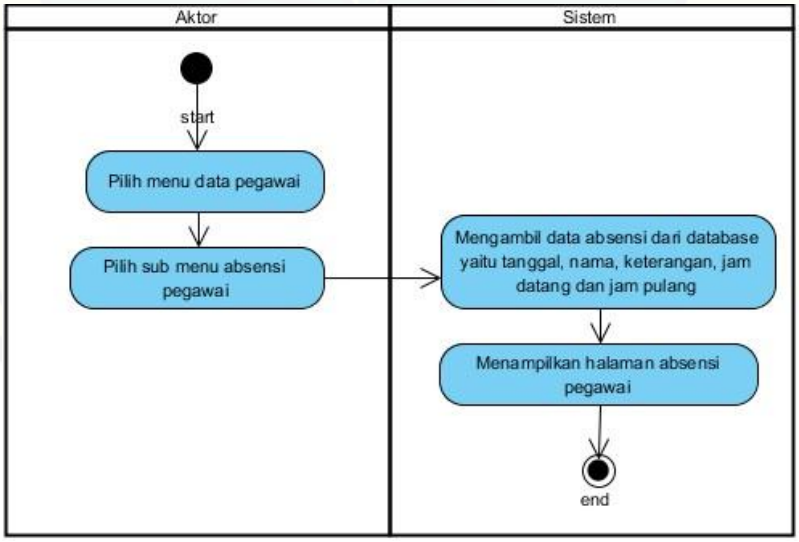
Gambar B. 1 *Activity* Diagram *Login*

B.2 *Activity* Diagram *View* Data Pegawai



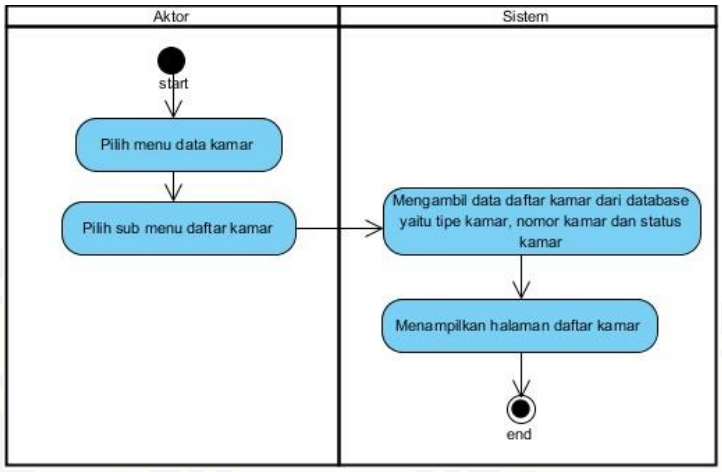
Gambar B. 2 *Activity* Diagram *View* Data Pegawai

B.3 *Activity* Diagram *View* Absensi



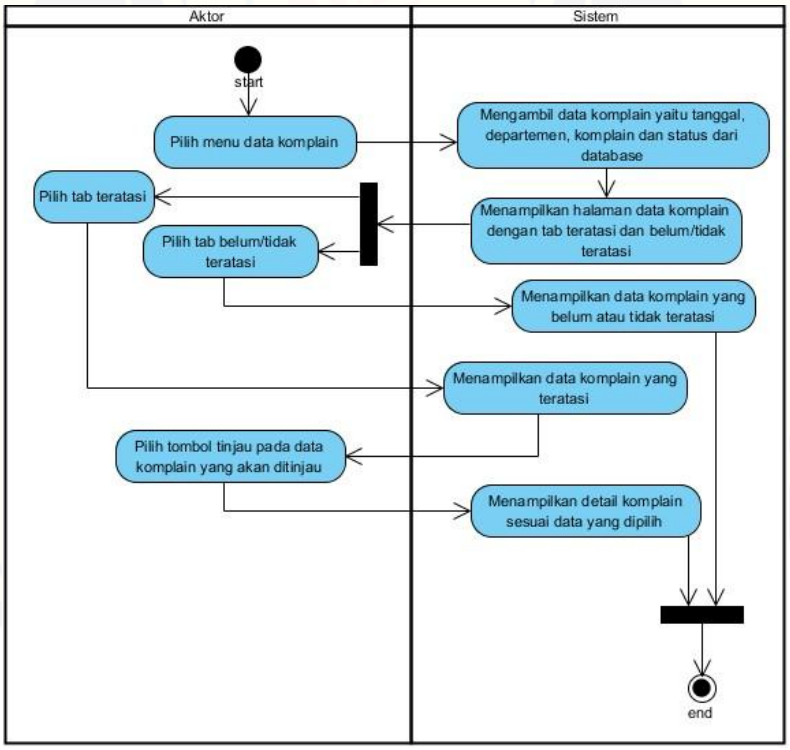
Gambar B. 3 *Activity* Diagram *View* Absensi

B.4 *Activity* Diagram *View* Data Kamar



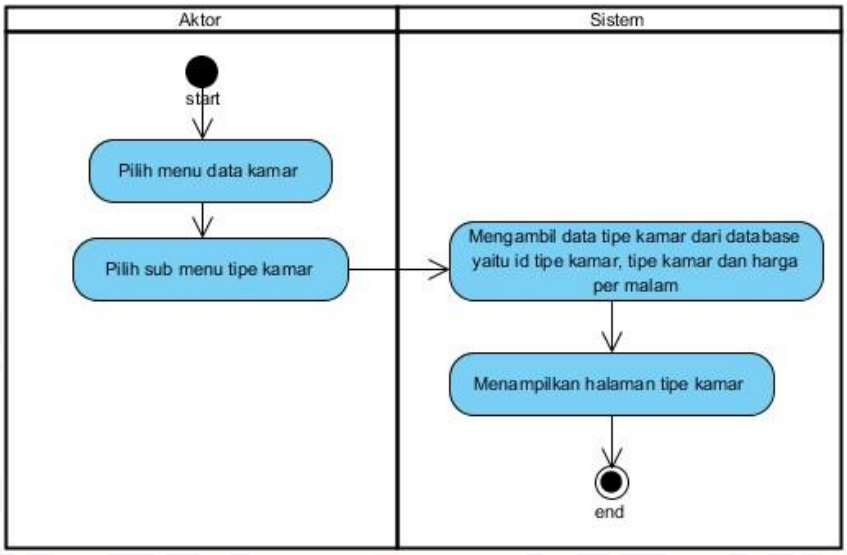
Gambar B. 4 *Activity* Diagram *View* Data Kamar

B.5 *Activity* Diagram *View* Data Komplain



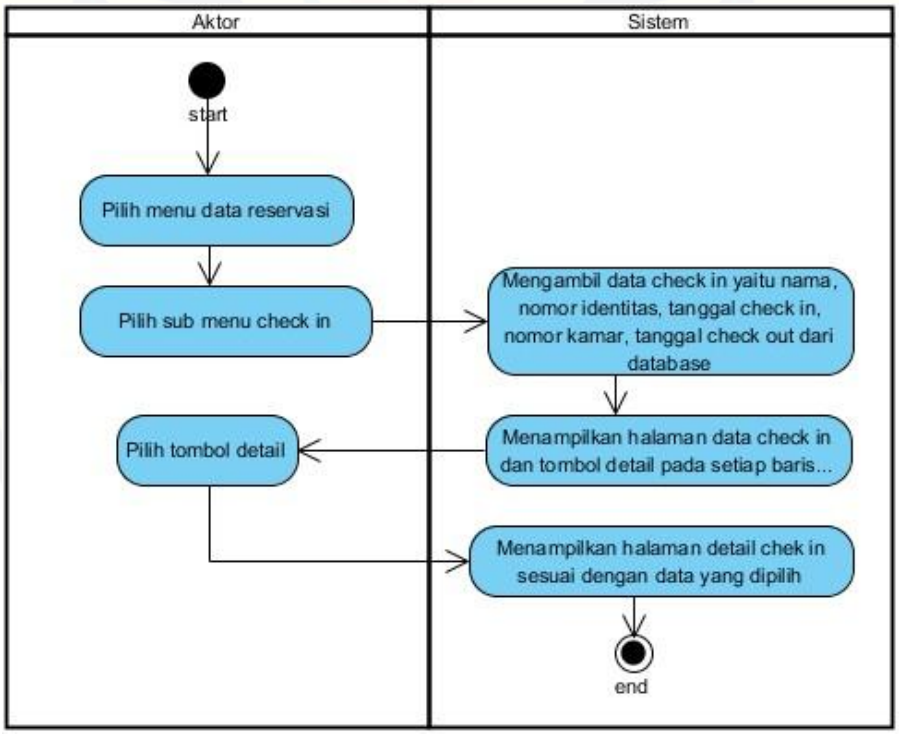
Gambar B. 5 *Activity* Diagram *View* Data Komplain

B.6 *Activity* Diagram *View* Data Tipe Kamar



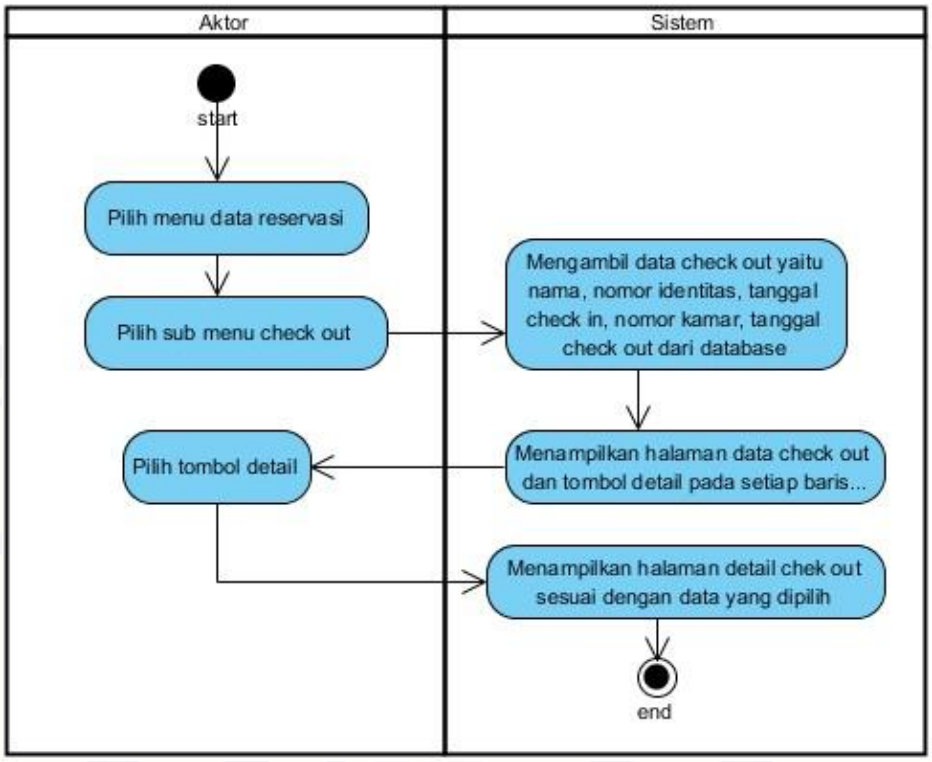
Gambar B. 6 *Activity* Diagram *View* Data Tipe Kamar

B.7 *Activity* Diagram *View* Data *Check in*



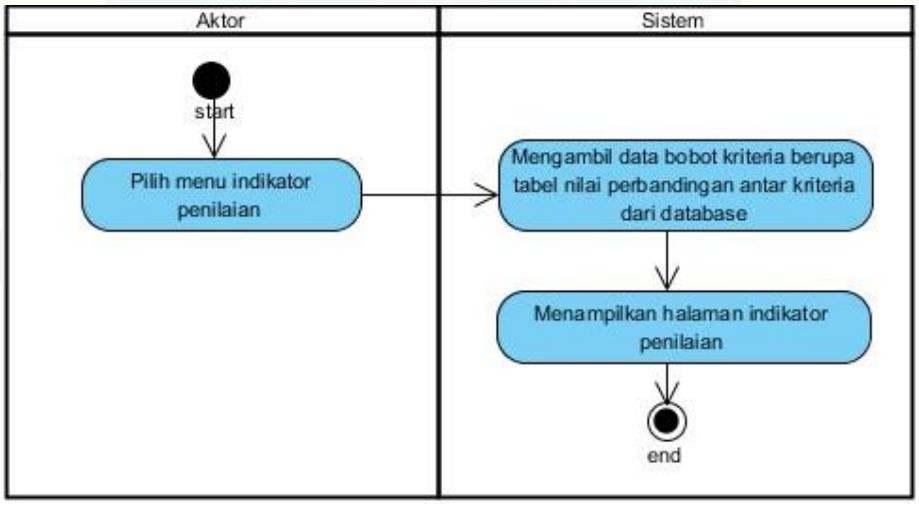
Gambar B. 7 *Activity* Diagram *View* Data *Check in*

B.8 *Activity* Diagram *View* Data *Check out*



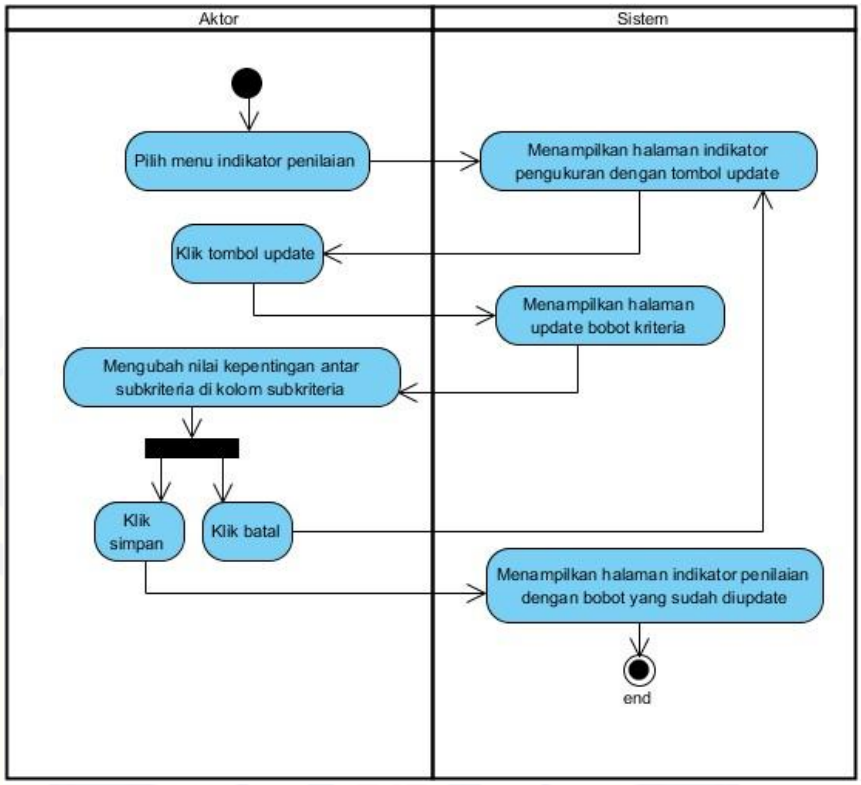
Gambar B. 8 *Activity* Diagram *View* Data *Check out*

B.9 *Activity* Diagram *View* Pembobotan Kriteria



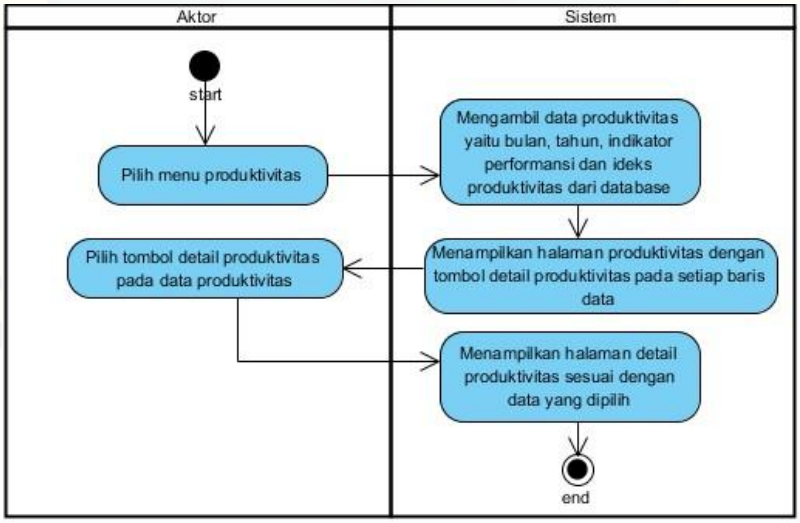
Gambar B. 9 *Activity* Diagram *View* Pembobotan Kriteria

B.10 *Activity* Diagram *Update* Bobot Kriteria



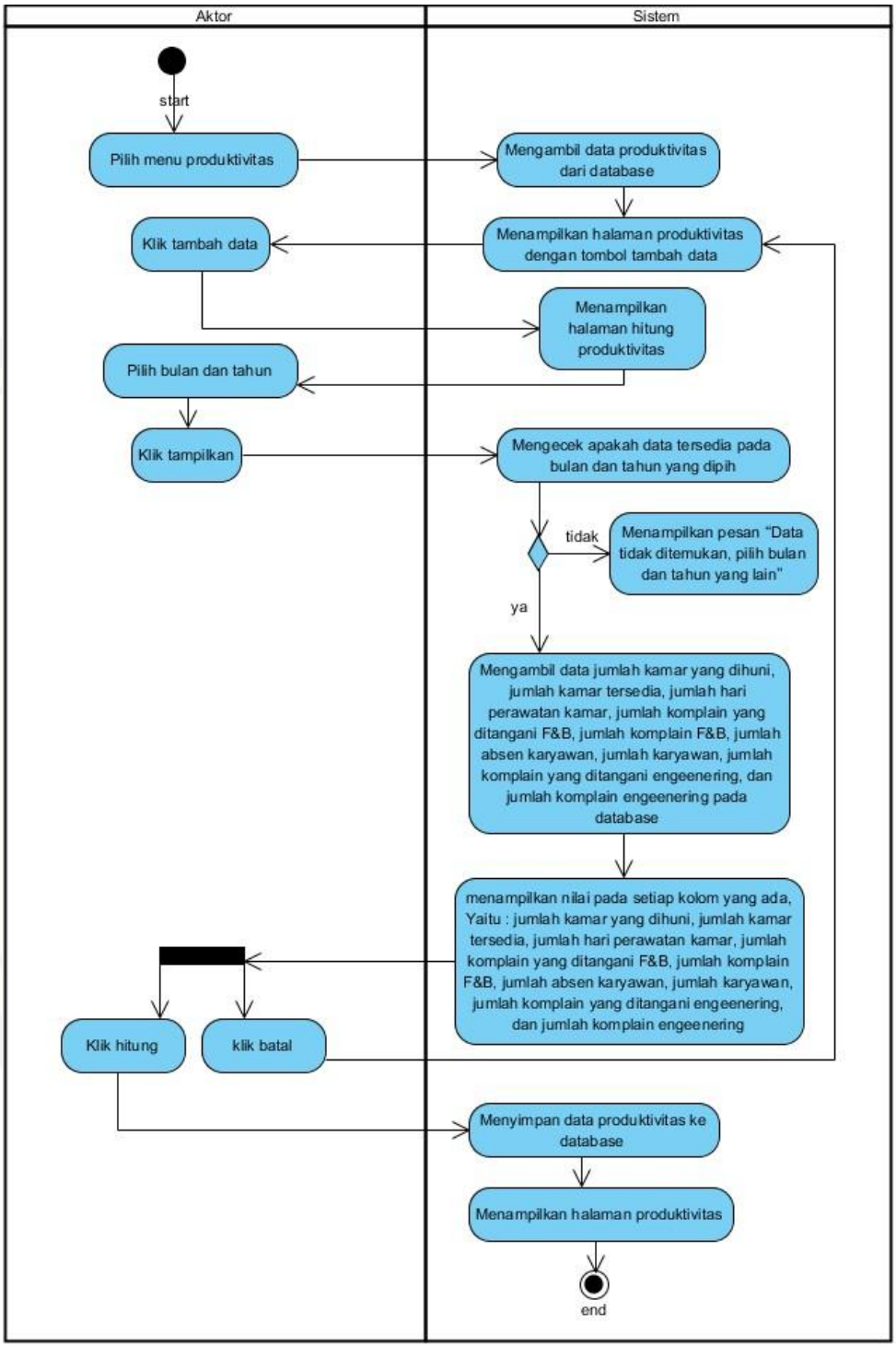
Gambar B. 10 *Activity* Diagram *Update* Bobot Kriteria

B.11 *Activity* Diagram *View* Produktivitas



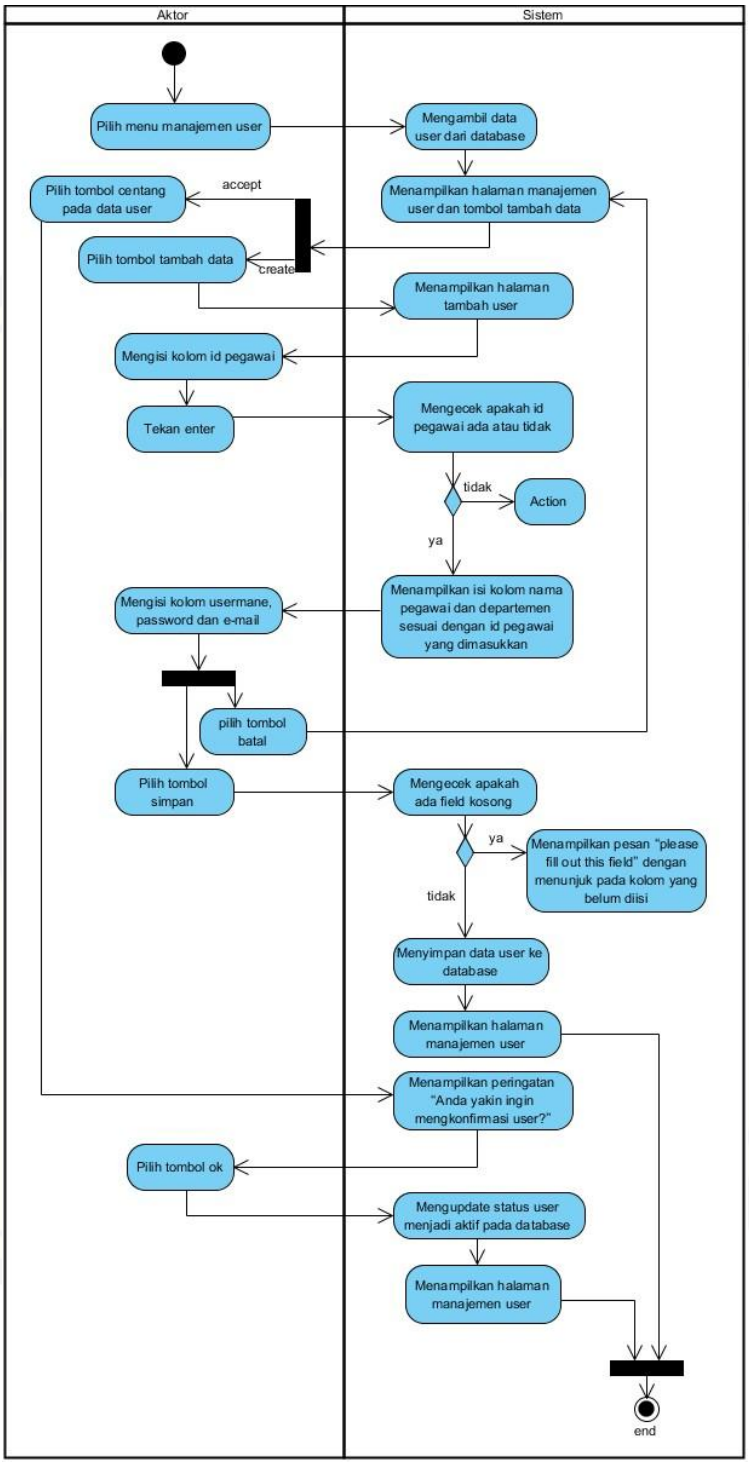
Gambar B. 11 *Activity* Diagram *View* Produktivitas

B.12 *Activity* Diagram *Input* Perhitungan Produktivitas



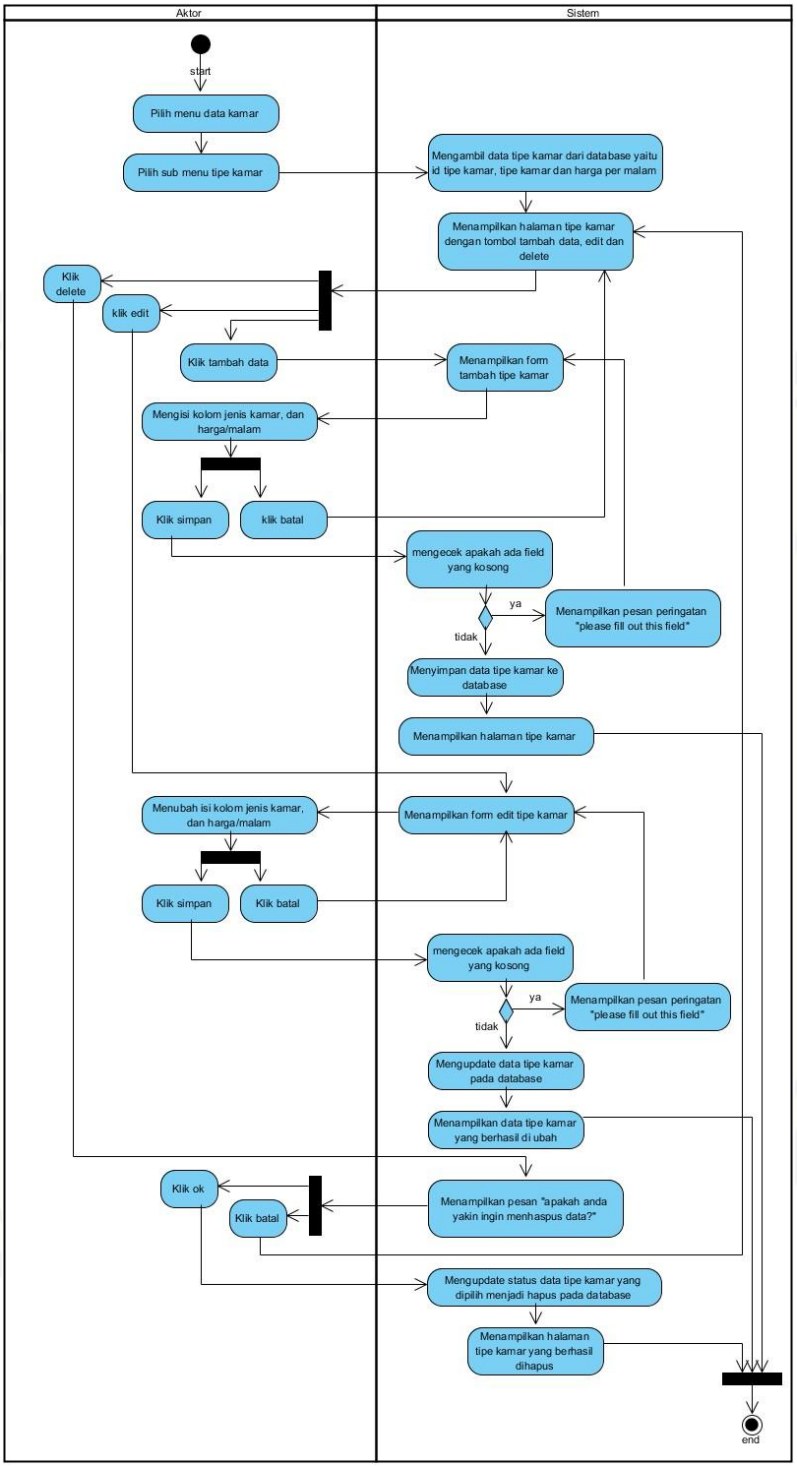
Gambar B. 12 *Activity* Diagram *Input* Perhitungan Produktivitas

B.13 *Activity* Diagram Mengelola Data *User*



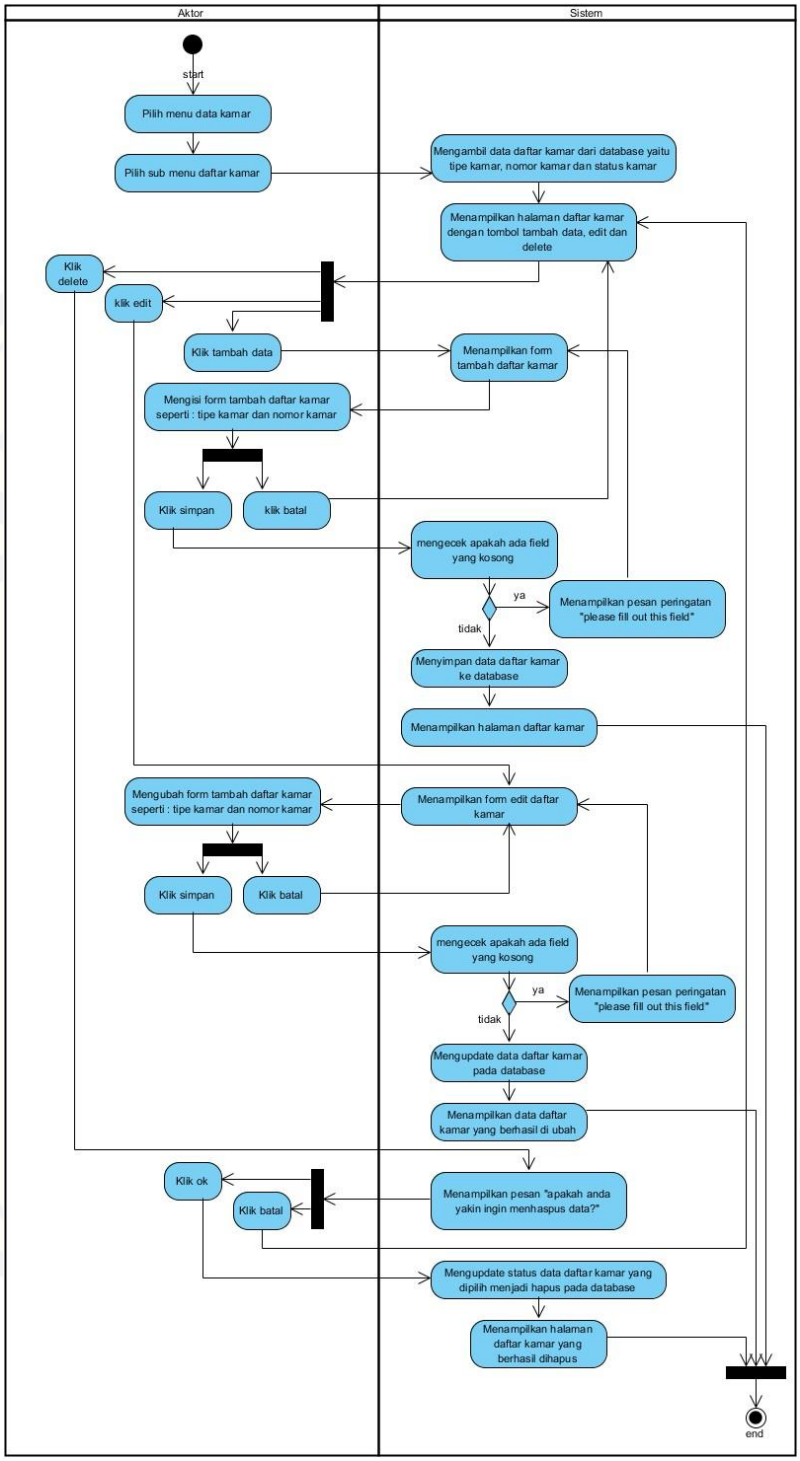
Gambar B. 13 *Activity* Diagram Mengelola Data *User*

B.14 *Activity* Diagram Mengelola Data Tipe Kamar



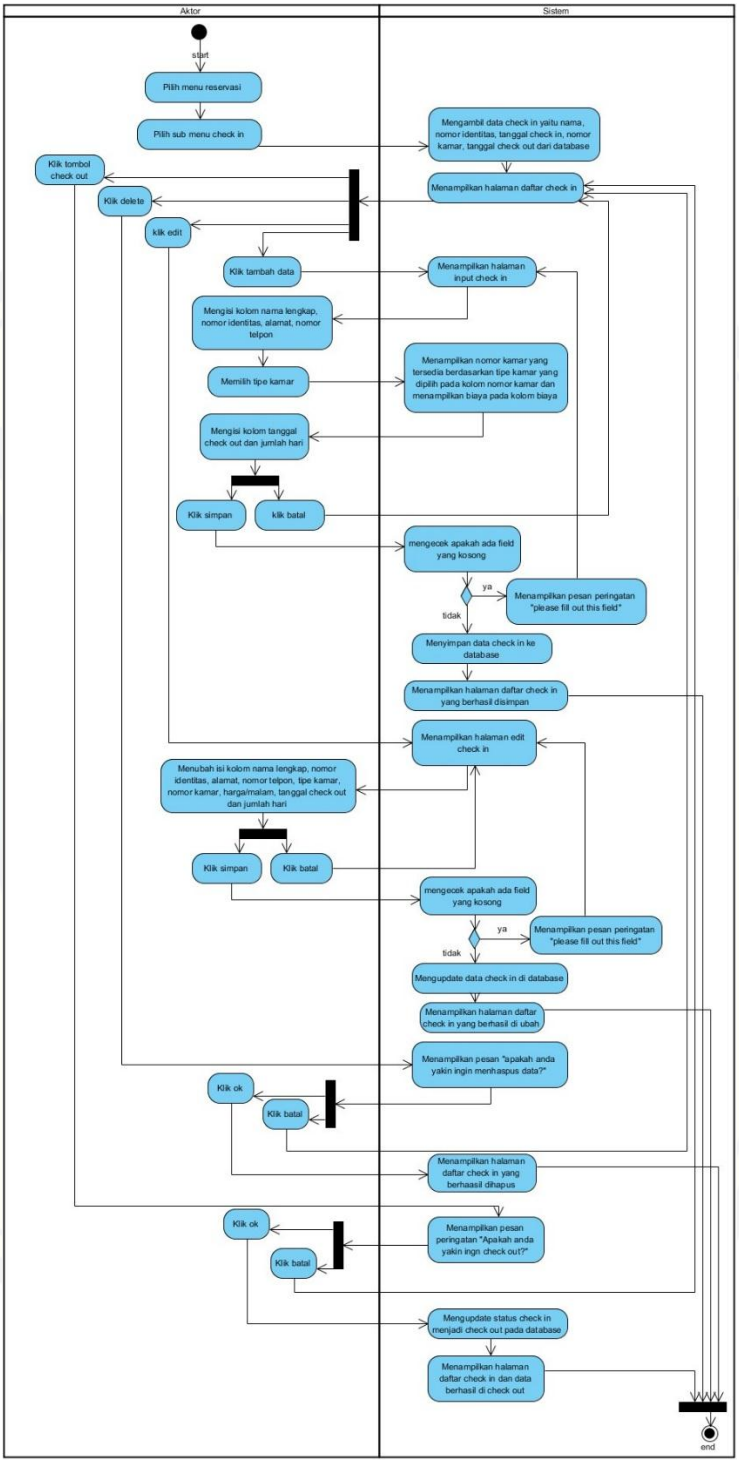
Gambar B. 14 *Activity* Diagram Mengelola Data Tipe Kamar

B.15 *Activity* Diagram Mengelola Data Daftar Kamar



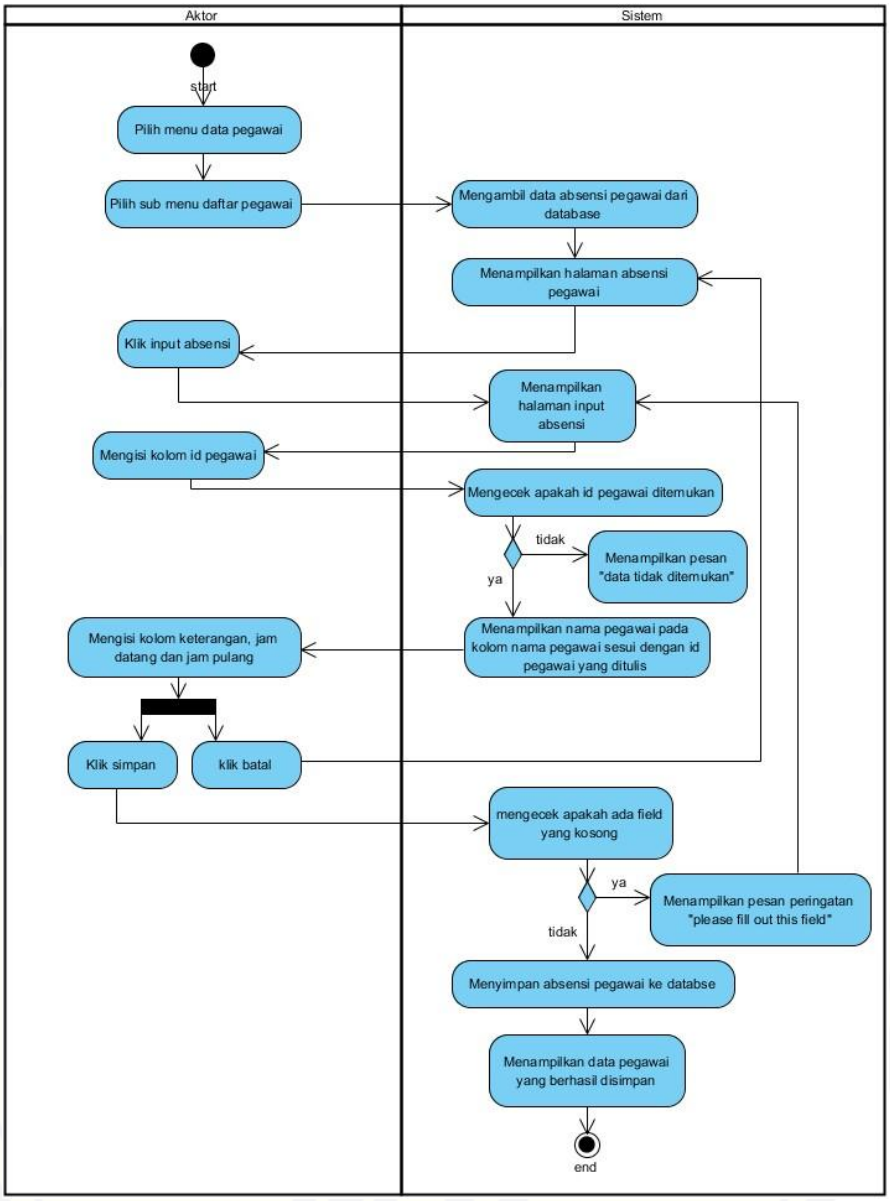
Gambar B. 15 *Activity* Diagram Mengelola Data Daftar Kamar

B.16 *Activity* Diagram Mengelola Data Reservasi



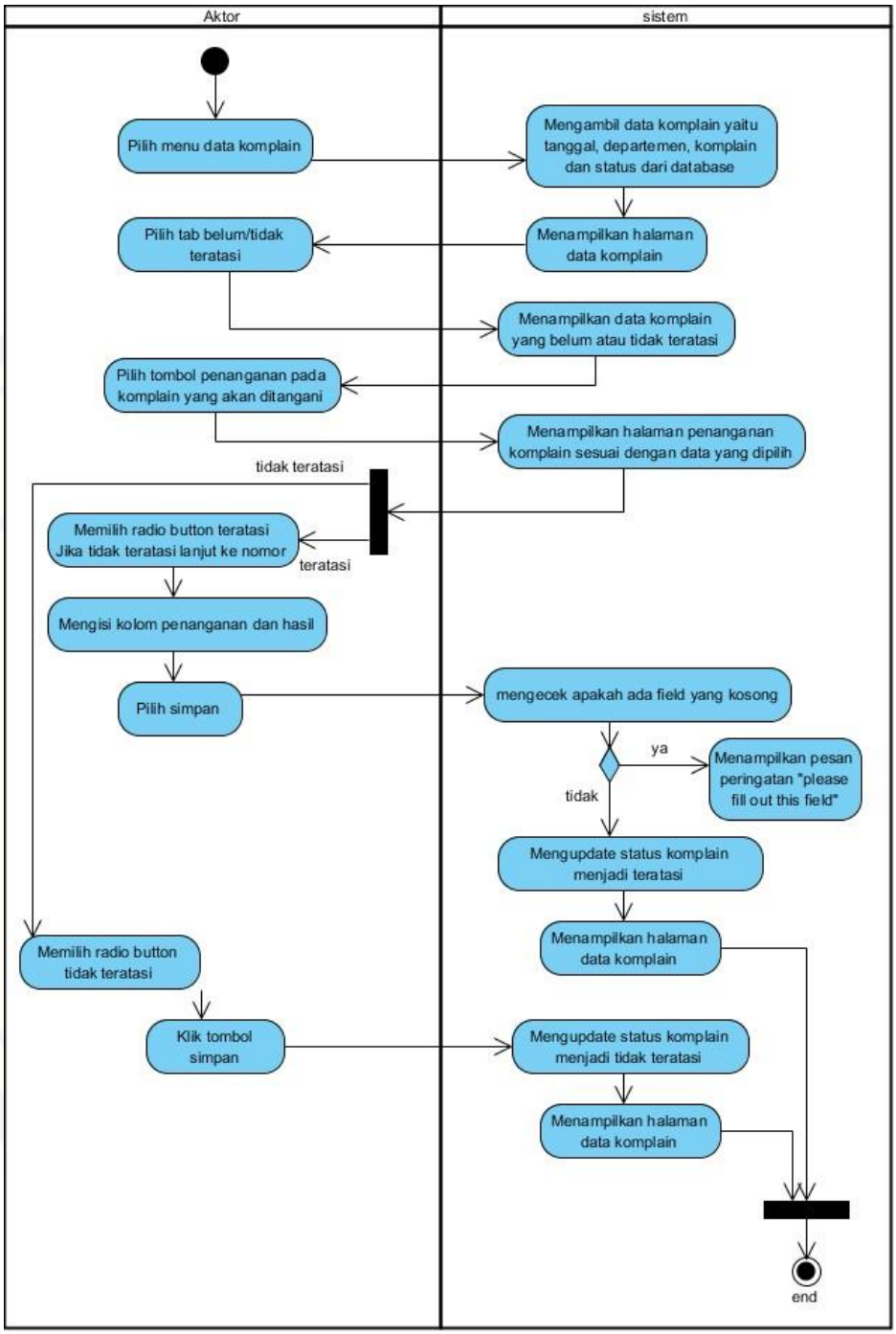
Gambar B. 16 *Activity* Diagram Mengelola Data Reservasi

B.17 *Activity* Diagram *Input* Absensi Pegawai



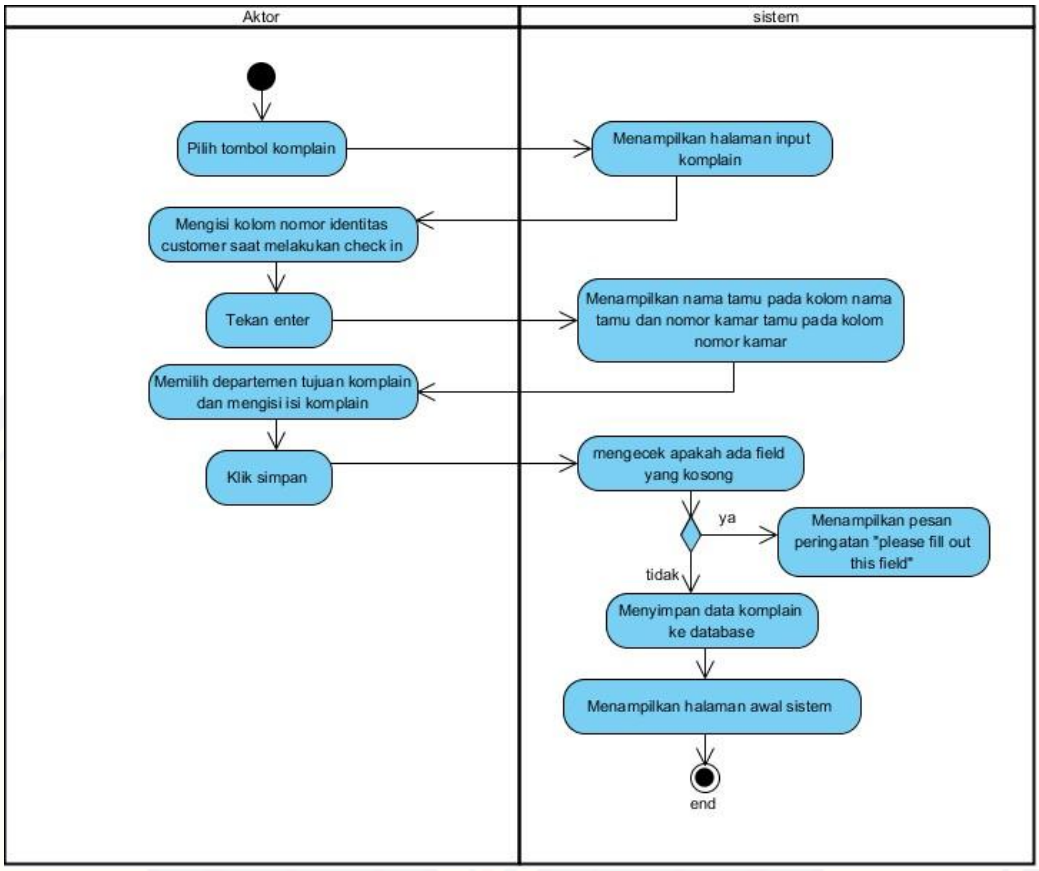
Gambar B. 17 *Activity* Diagram *Input* Absensi Pegawai

B.18 *Activity* Diagram *Input* Penanganan Komplain



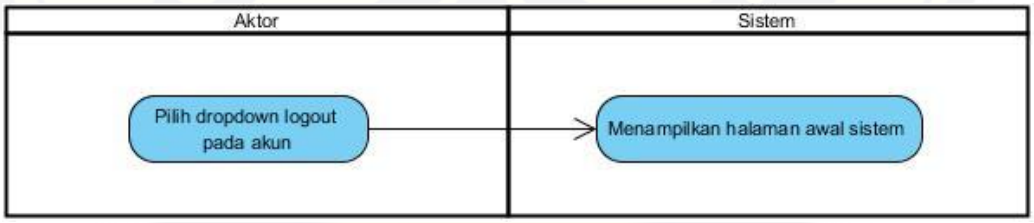
Gambar B. 18 *Activity* Diagram *Input* Penanganan Komplain

B.19 *Activity* Diagram *Input* Komplain



Gambar B. 19 *Activity* Diagram *Input* Komplain

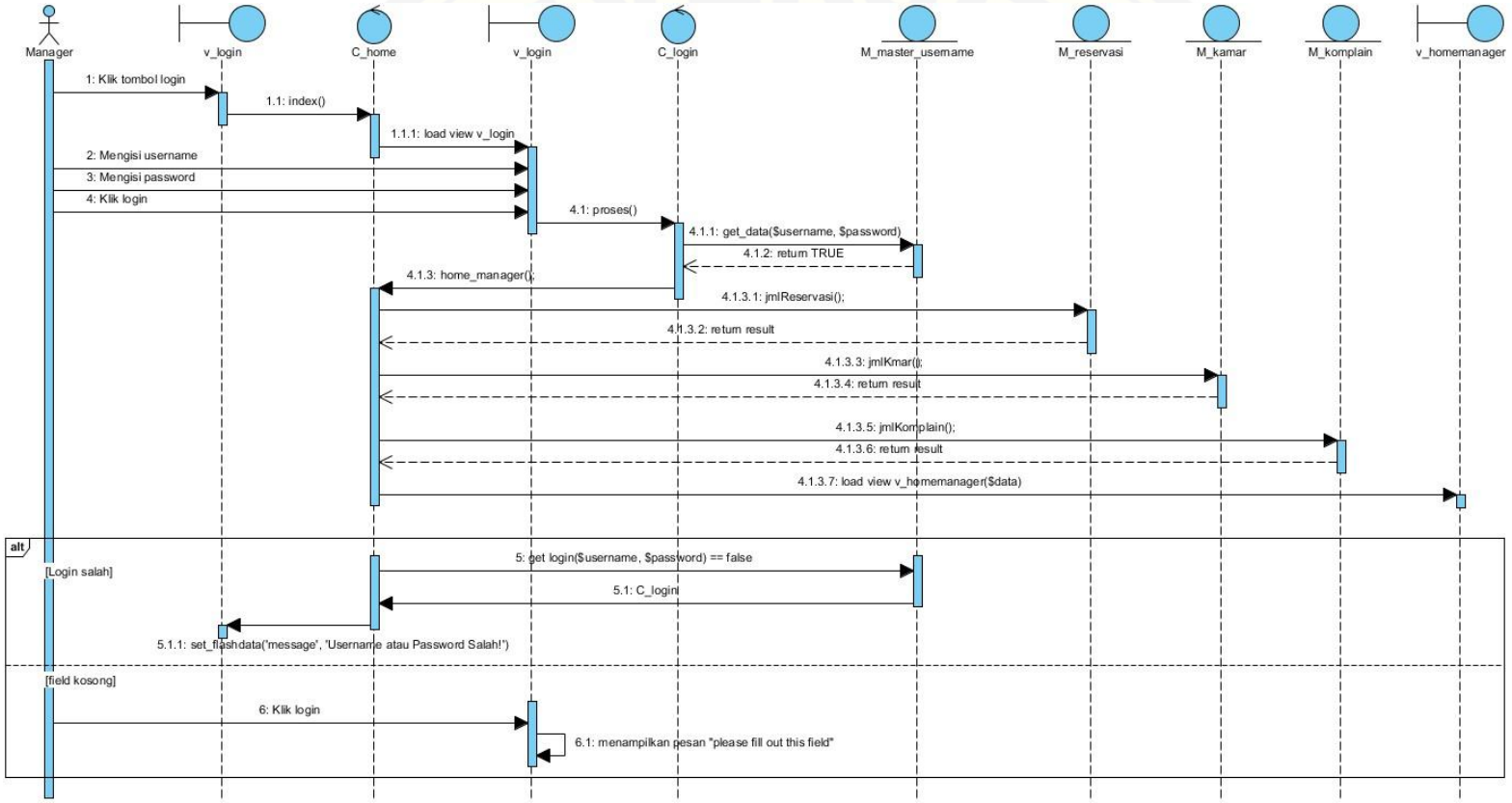
B.20 *Activity* Diagram Skenario *Logout*



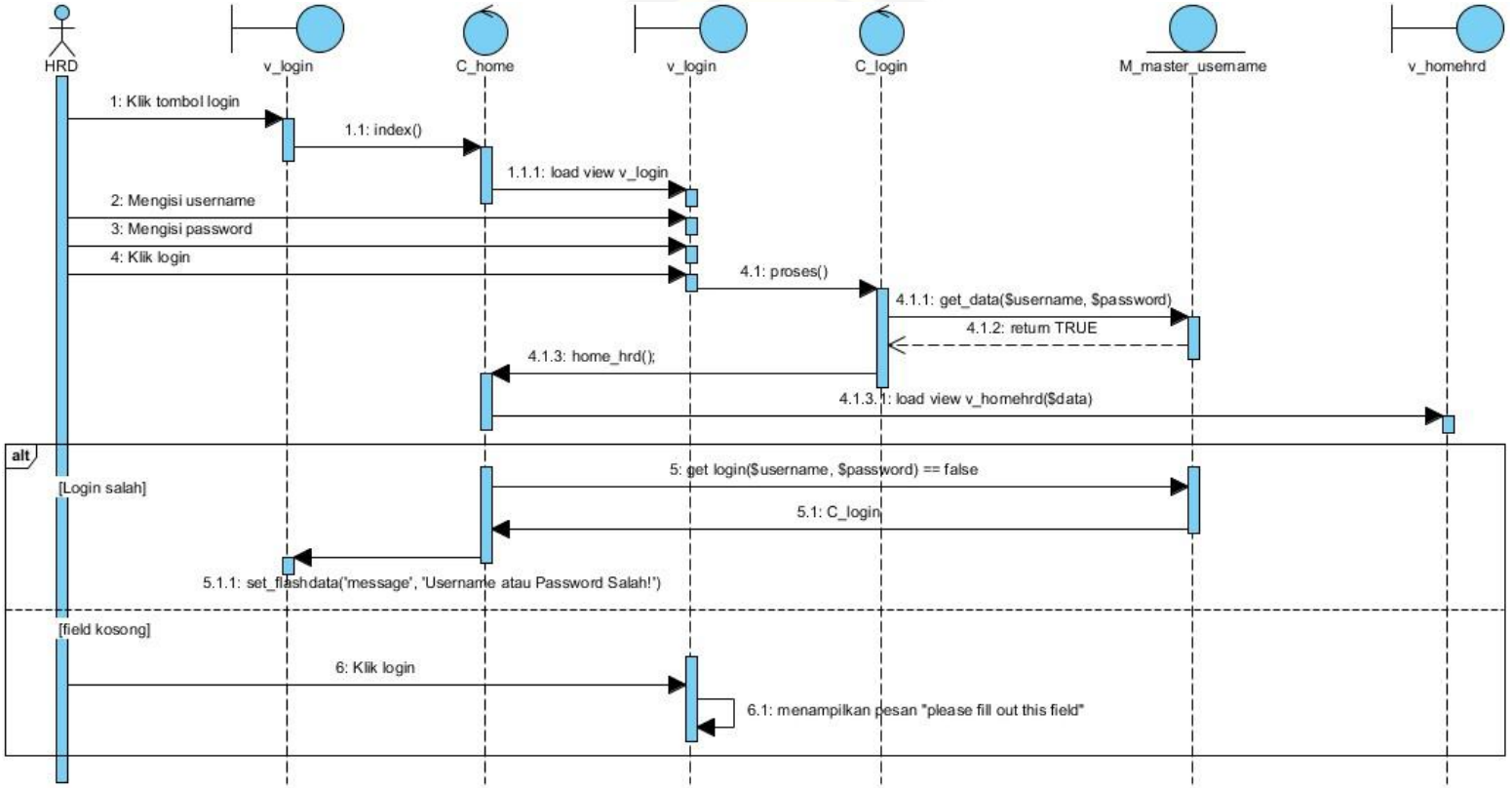
Gambar B. 20 *Activity* Diagram Skenario *Logout*

LAMPIRAN C. *SEQUENCE* DIAGRAM

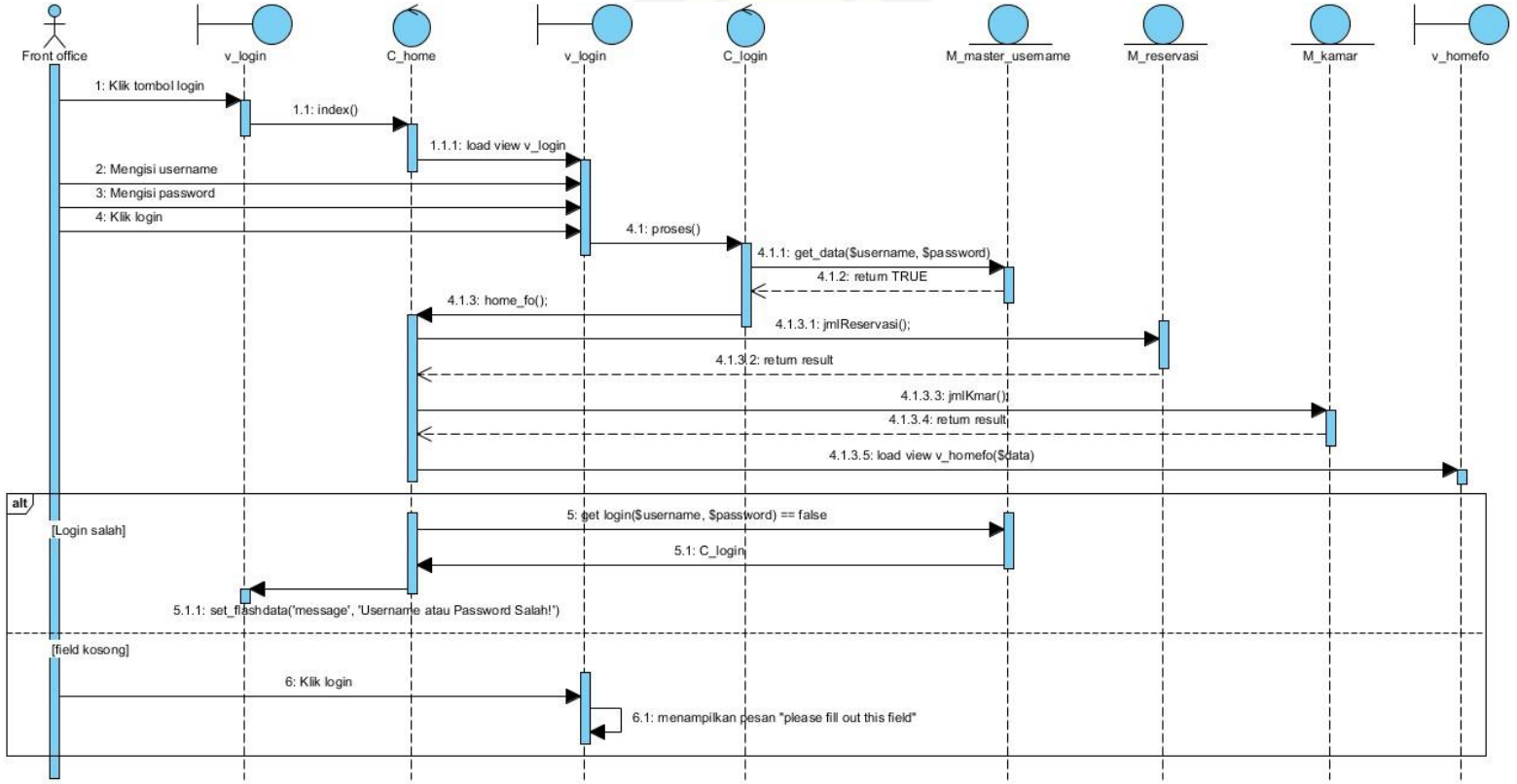
C.1 *Sequence* Diagram *Login*



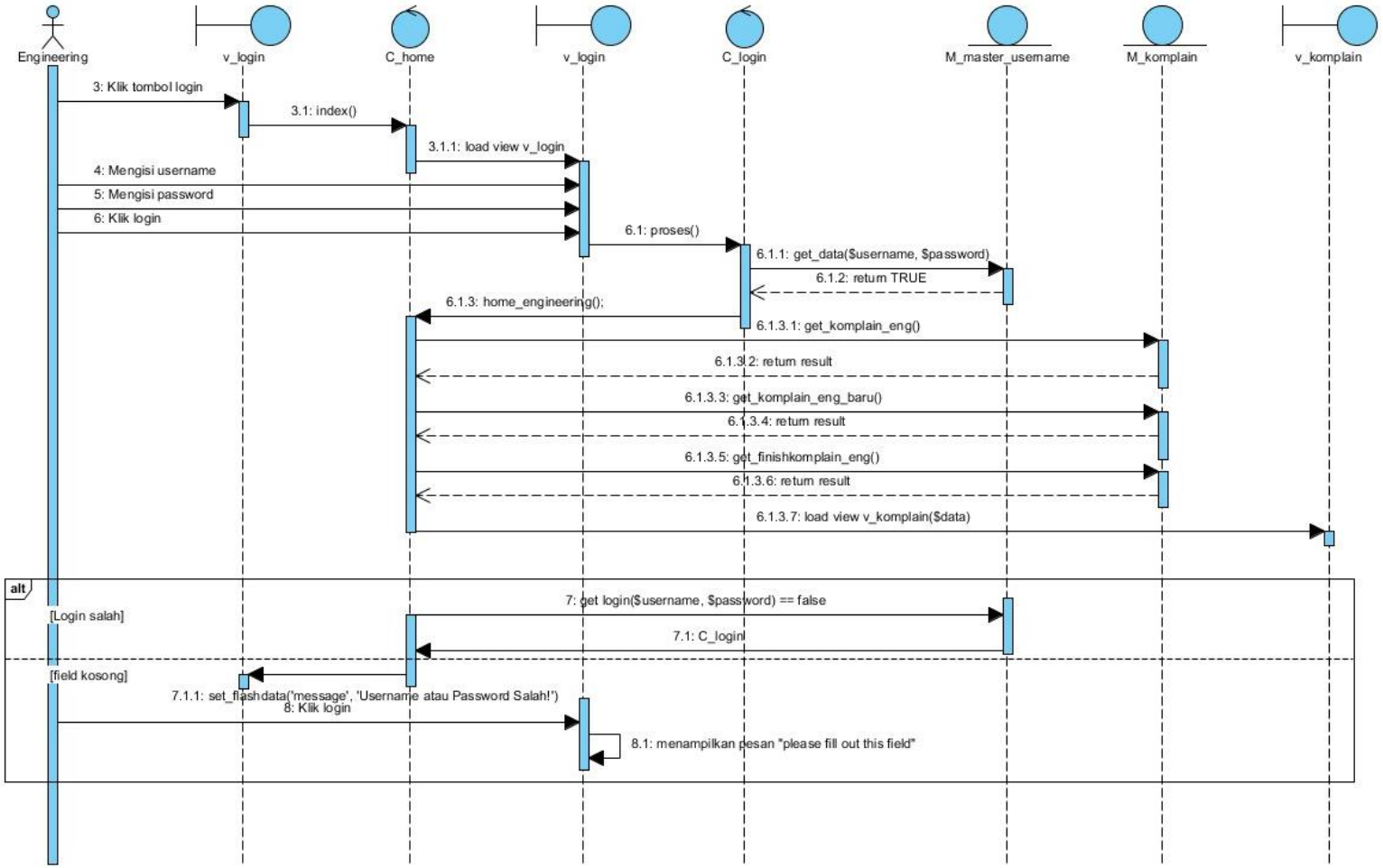
Gambar C. 1 *Sequence* diagram *login* (manager)



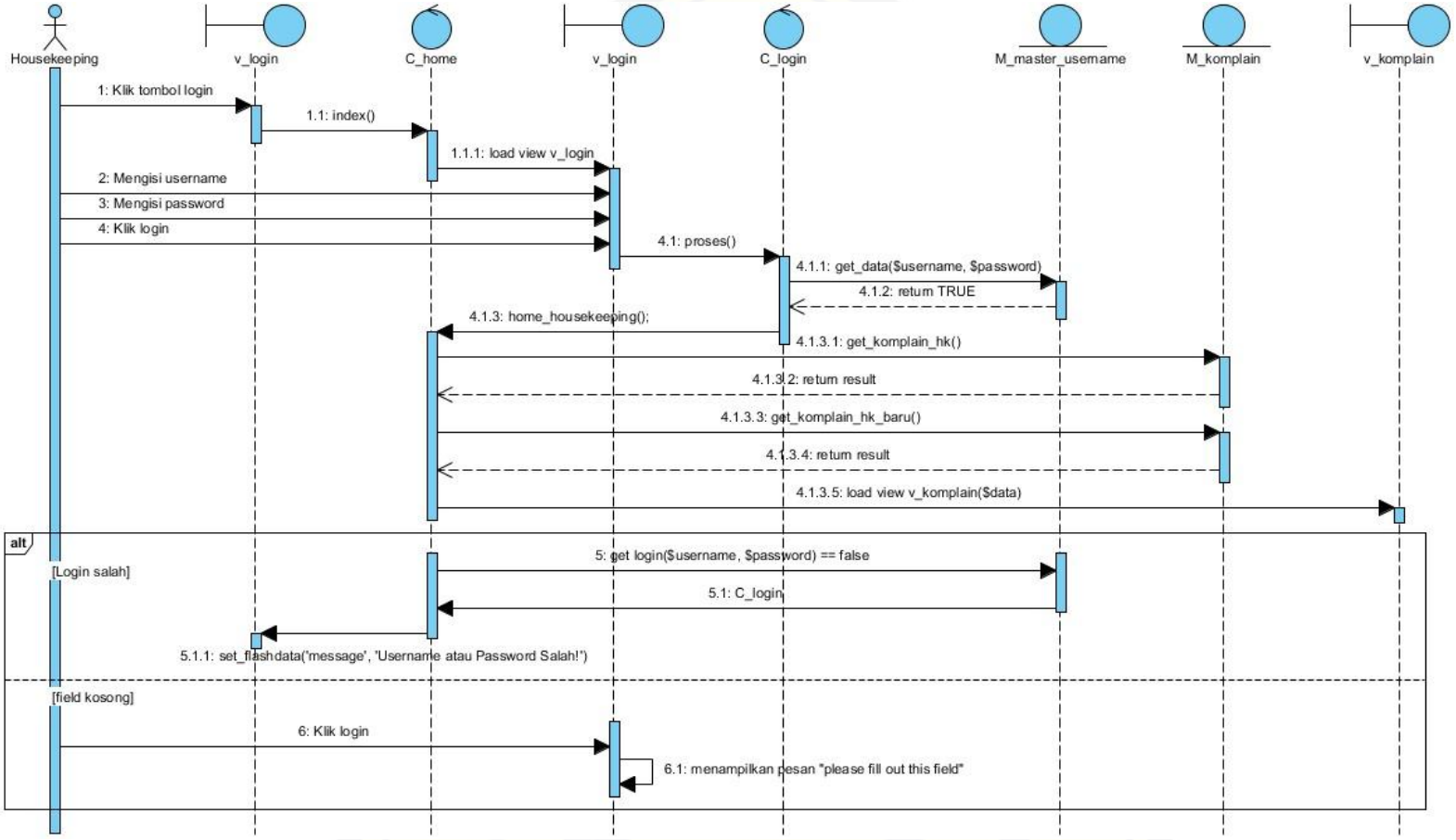
Gambar C. 2 *Sequence* diagram *login* (HRD)



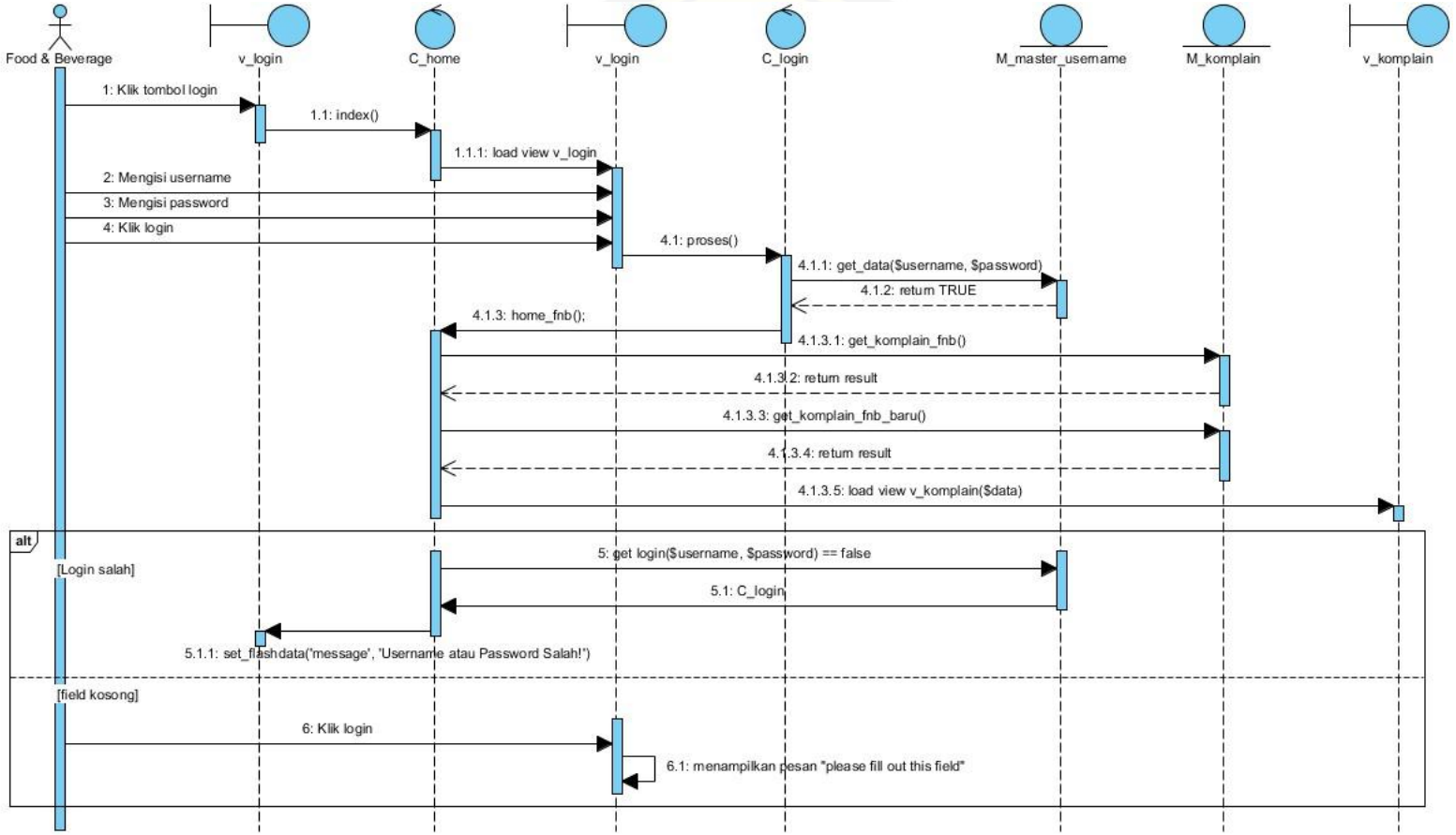
Gambar C. 3 *Sequence* diagram *login* (*front office*)



Gambar C. 4 *Sequence* diagram *login* (*engineering*)

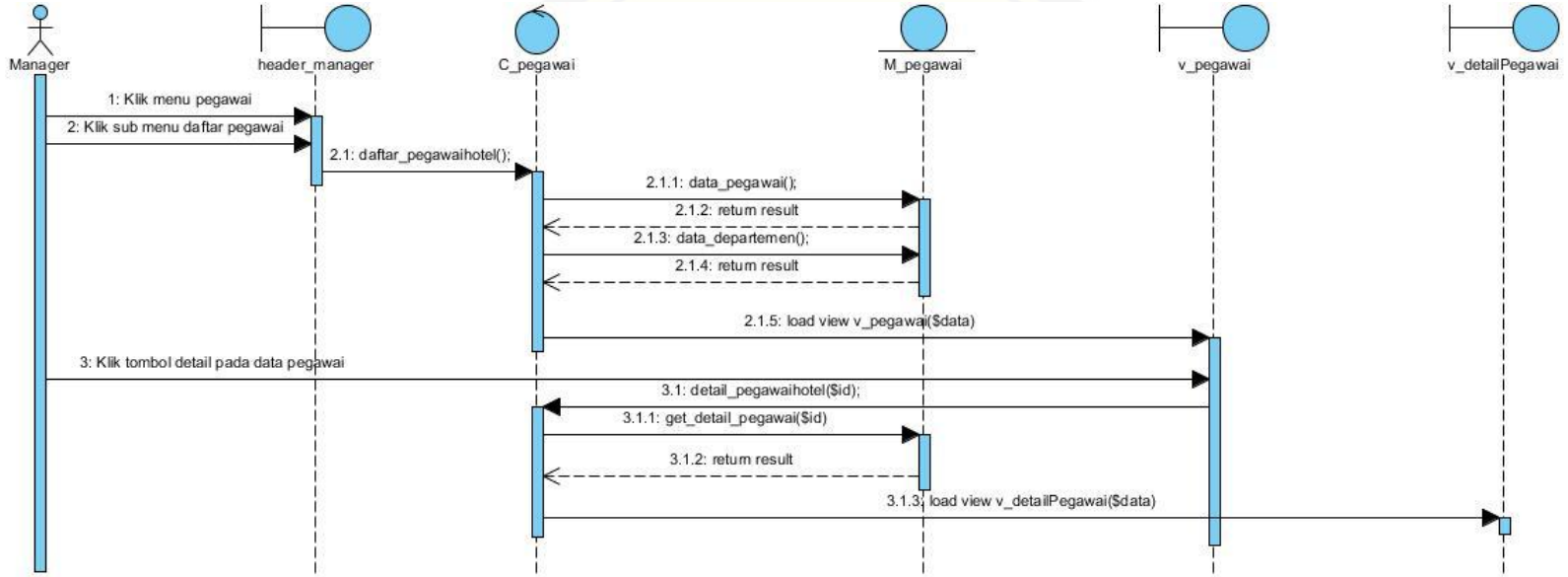


Gambar C. 5 *Sequence* diagram *login* (*housekeeping*)



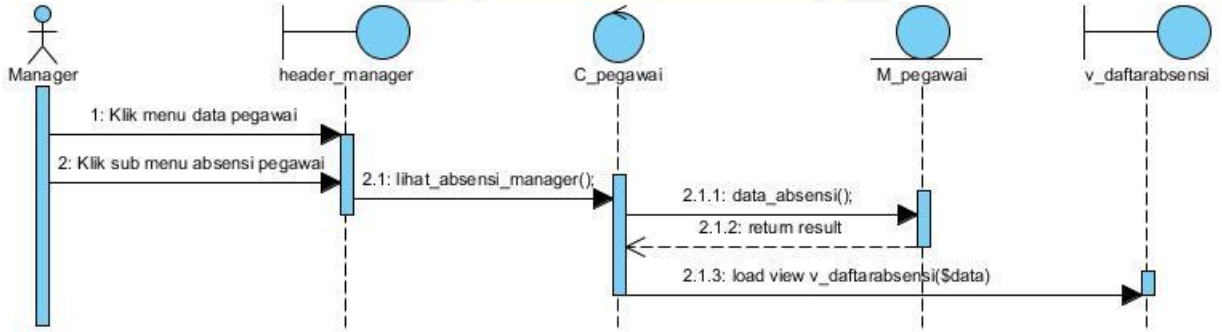
Gambar C. 6 *Sequence* diagram *login* (*food & beverage*)

C.2 *Sequence* Diagram *View* Data Pegawai



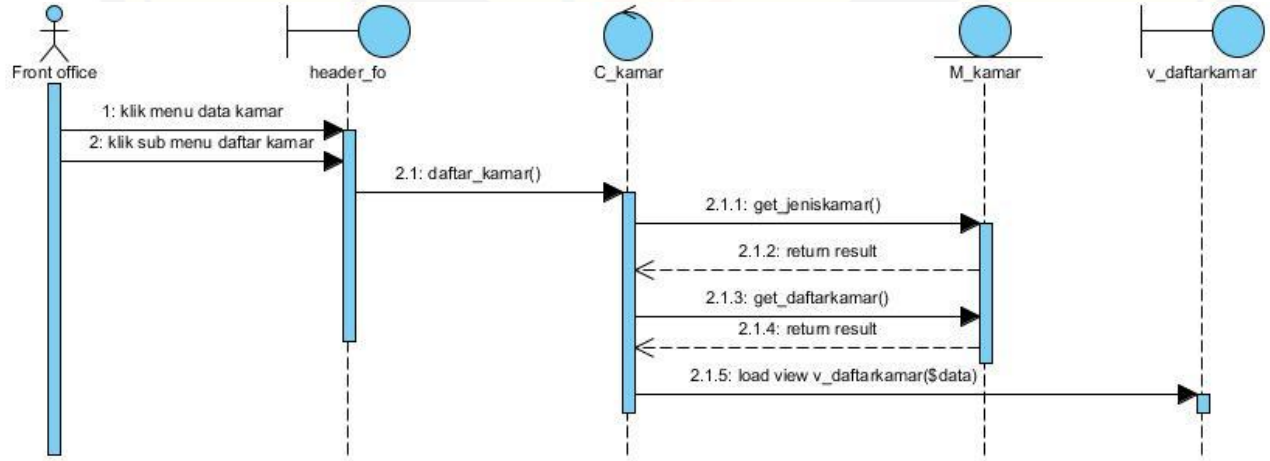
Gambar C. 7 *Sequence* diagram *view* data pegawai (manager)

C.3 *Sequence* Diagram *View* Absensi



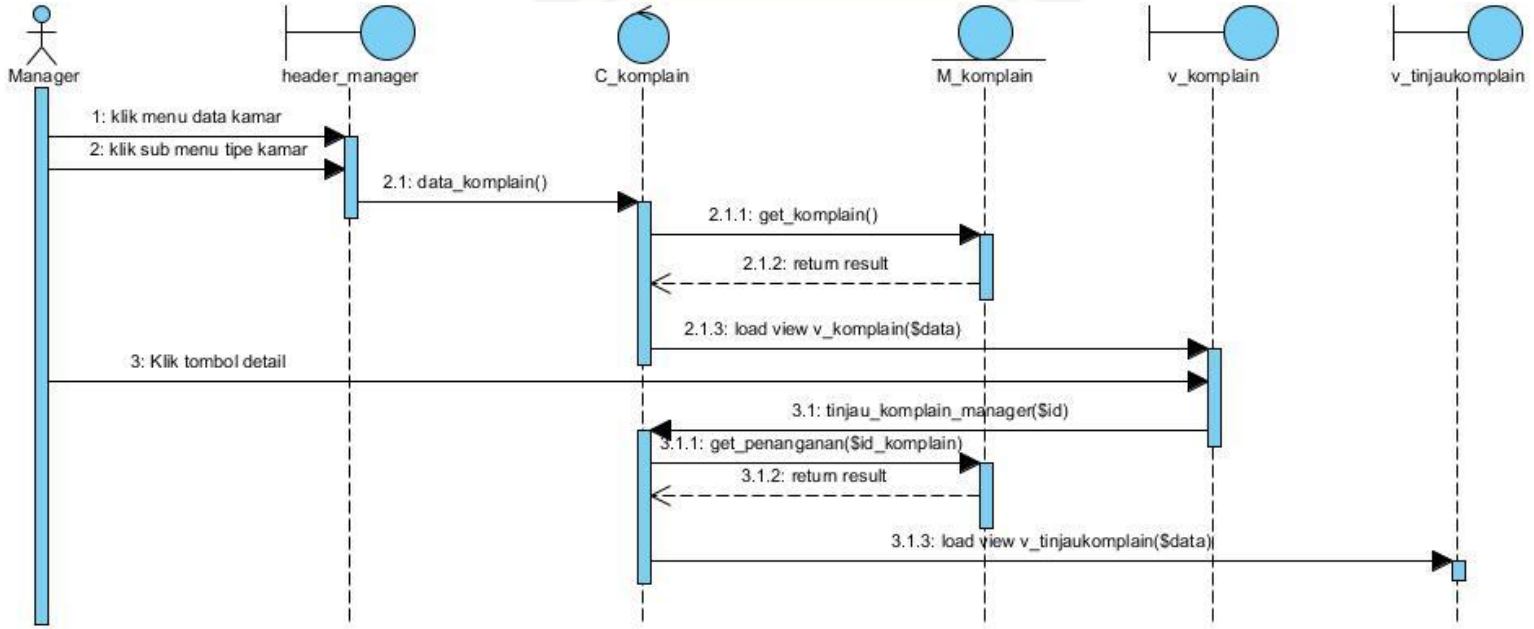
Gambar C. 8 *Sequence* diagram *view* absensi (manager)

C.4 *Sequence* Diagram *View* Data Kamar



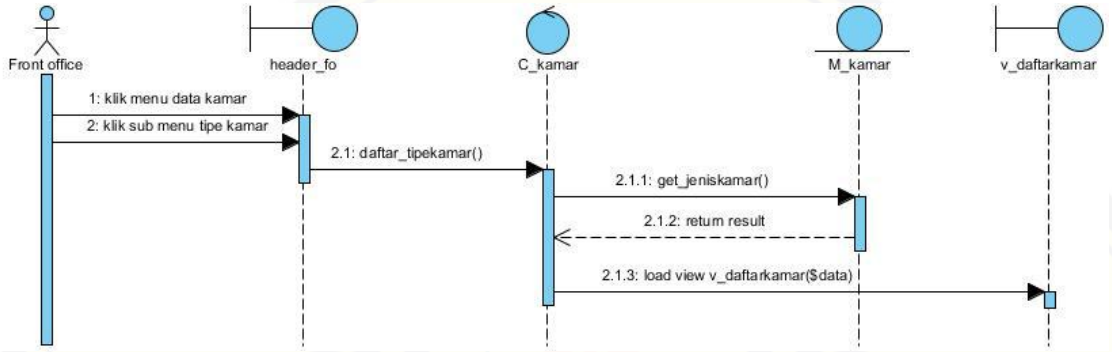
Gambar C. 9 *Sequence* diagram *view* data kamar (*front office*)

C.5 *Sequence* Diagram *View* Data Komplain



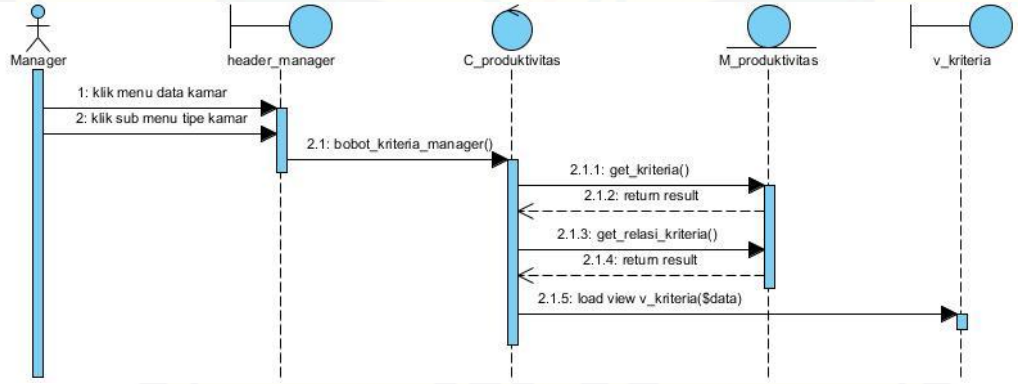
Gambar C. 10 *Sequence* diagram *view* data komplain (manager)

C.6 *Sequence* Diagram *View* Data Tipe Kamar



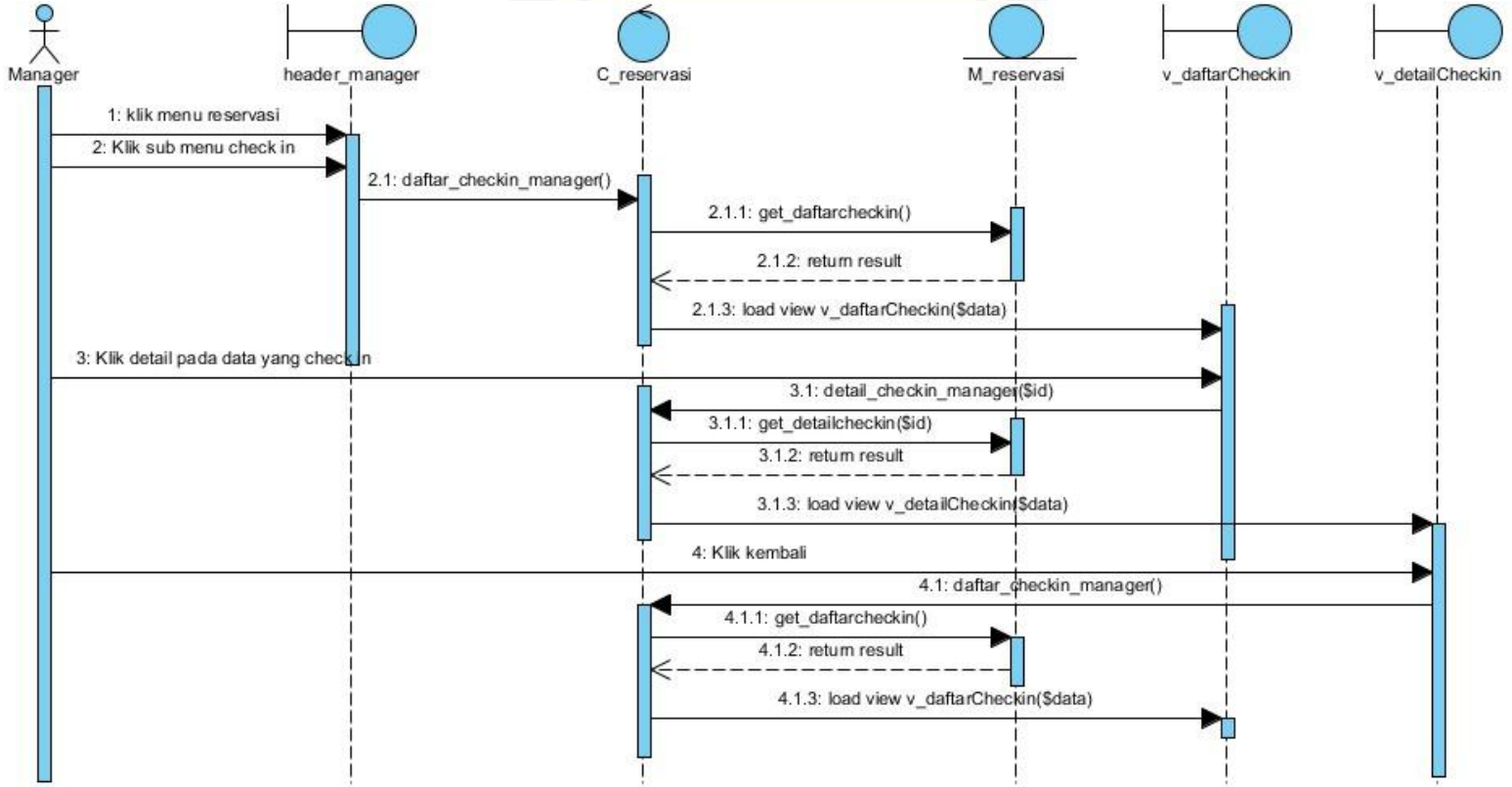
Gambar C. 11 *Sequence* diagram *view* data tipe kamar (*front office*)

C.7 *Sequence* Diagram *View* Pembobotan Kriteria

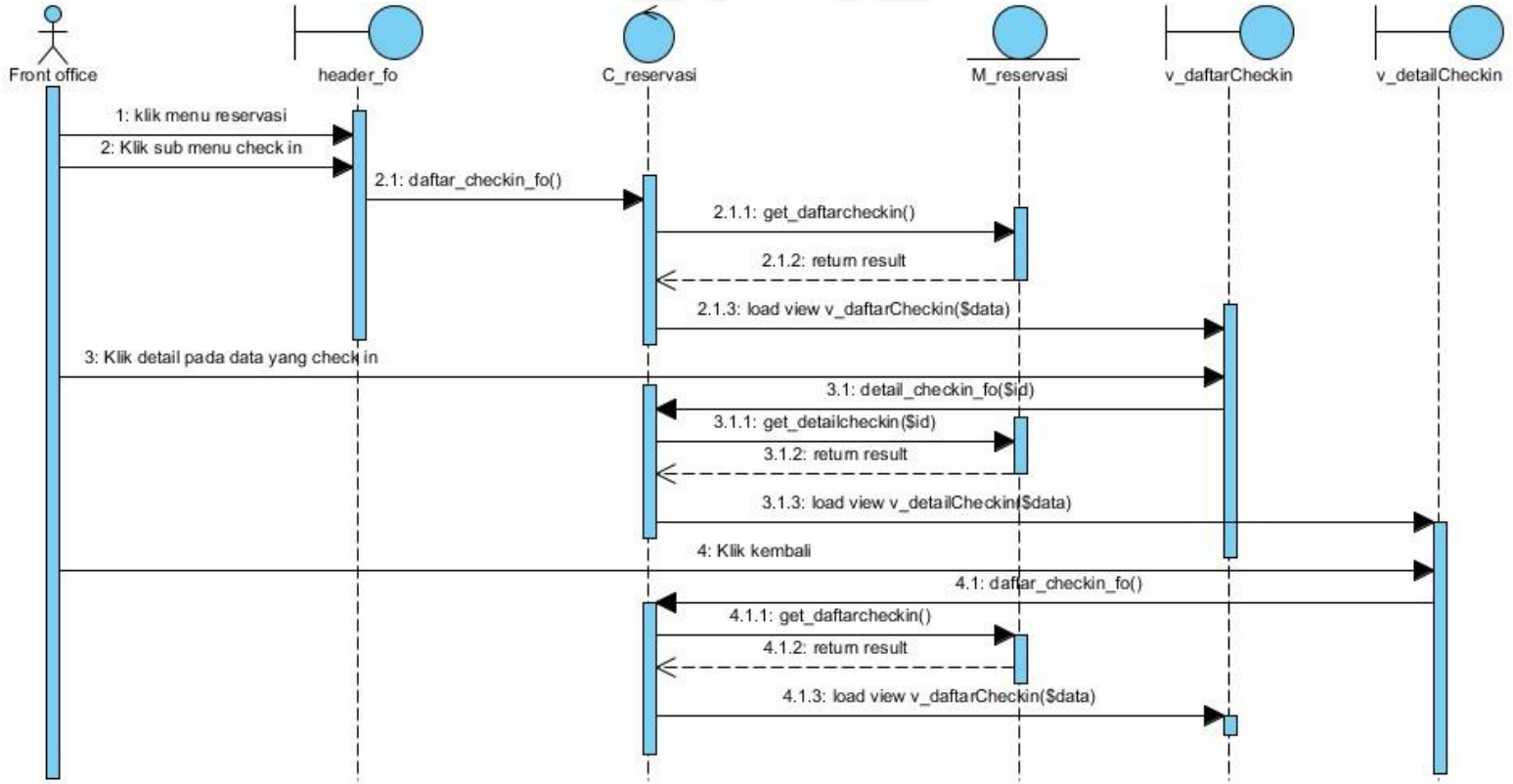


Gambar C. 12 *Sequence* diagram *view* pembobotan kriteria (manager)

C.8 *Sequence* Diagram *View* Data *Check in*

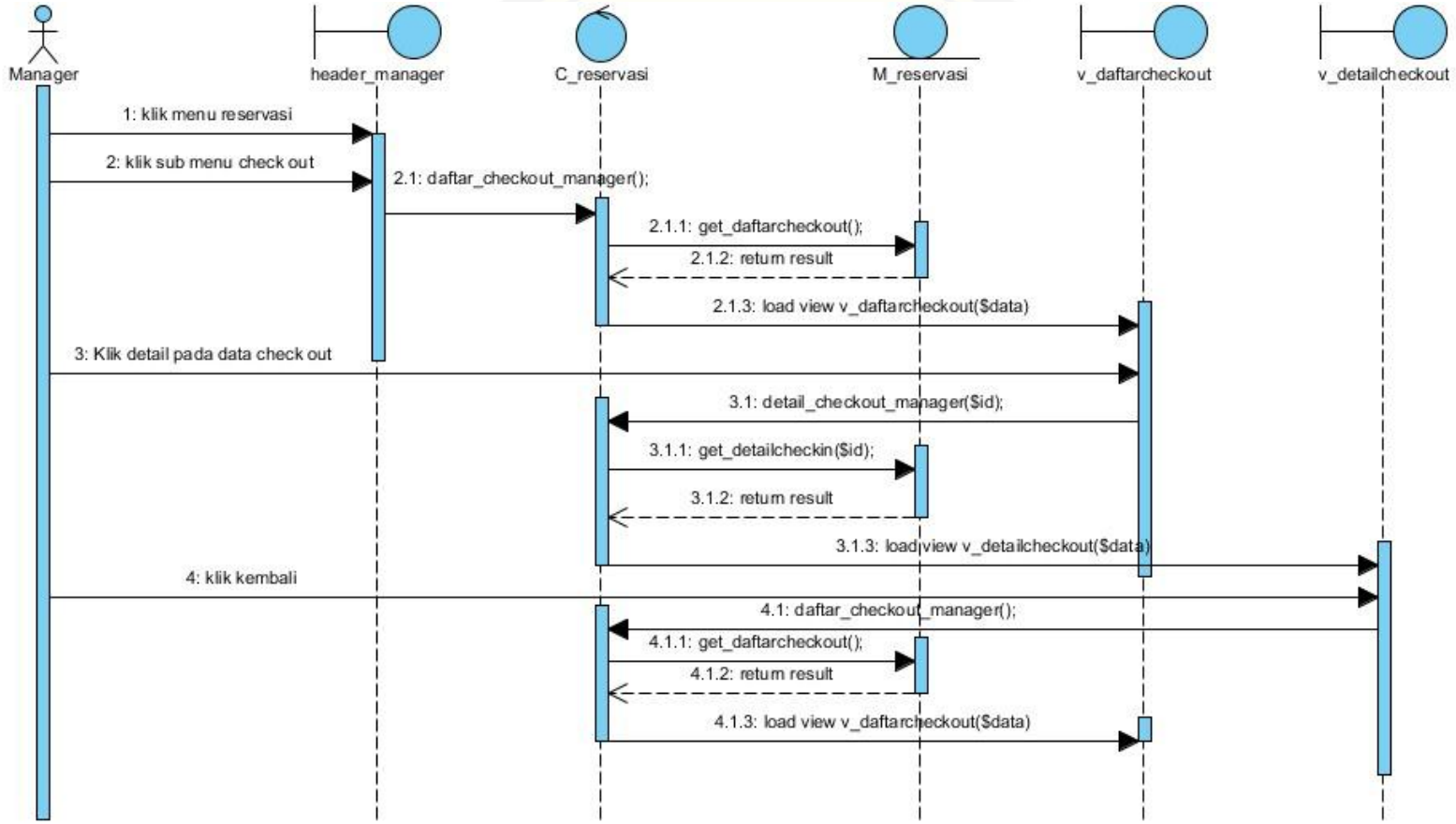


Gambar C. 13 *Sequence* diagram *view* data *check in* (manager)

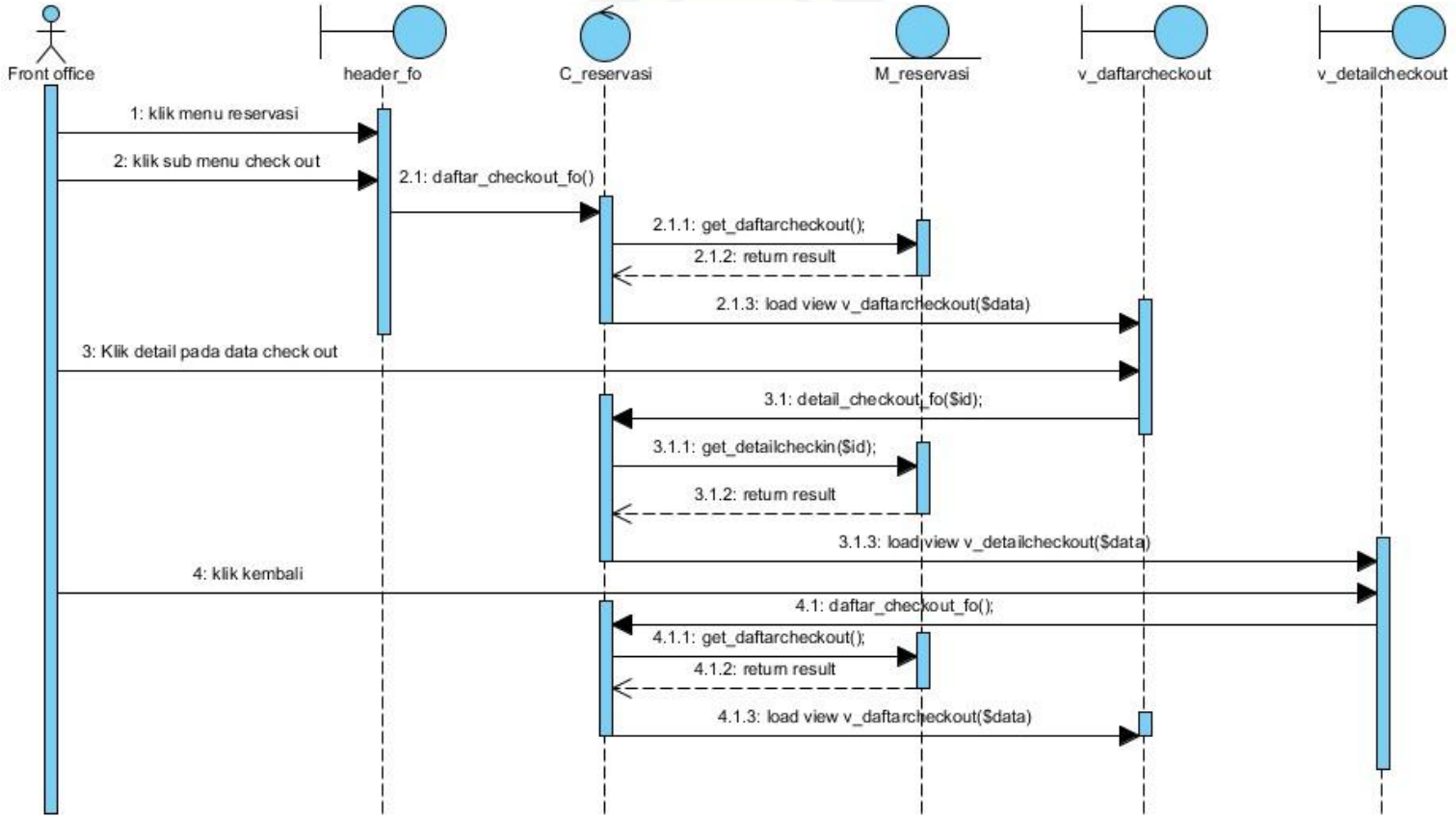


Gambar C. 14 *Sequence* diagram *view* data *check in* (*front office*)

C.9 *Sequence* Diagram *View* Data *Check out*

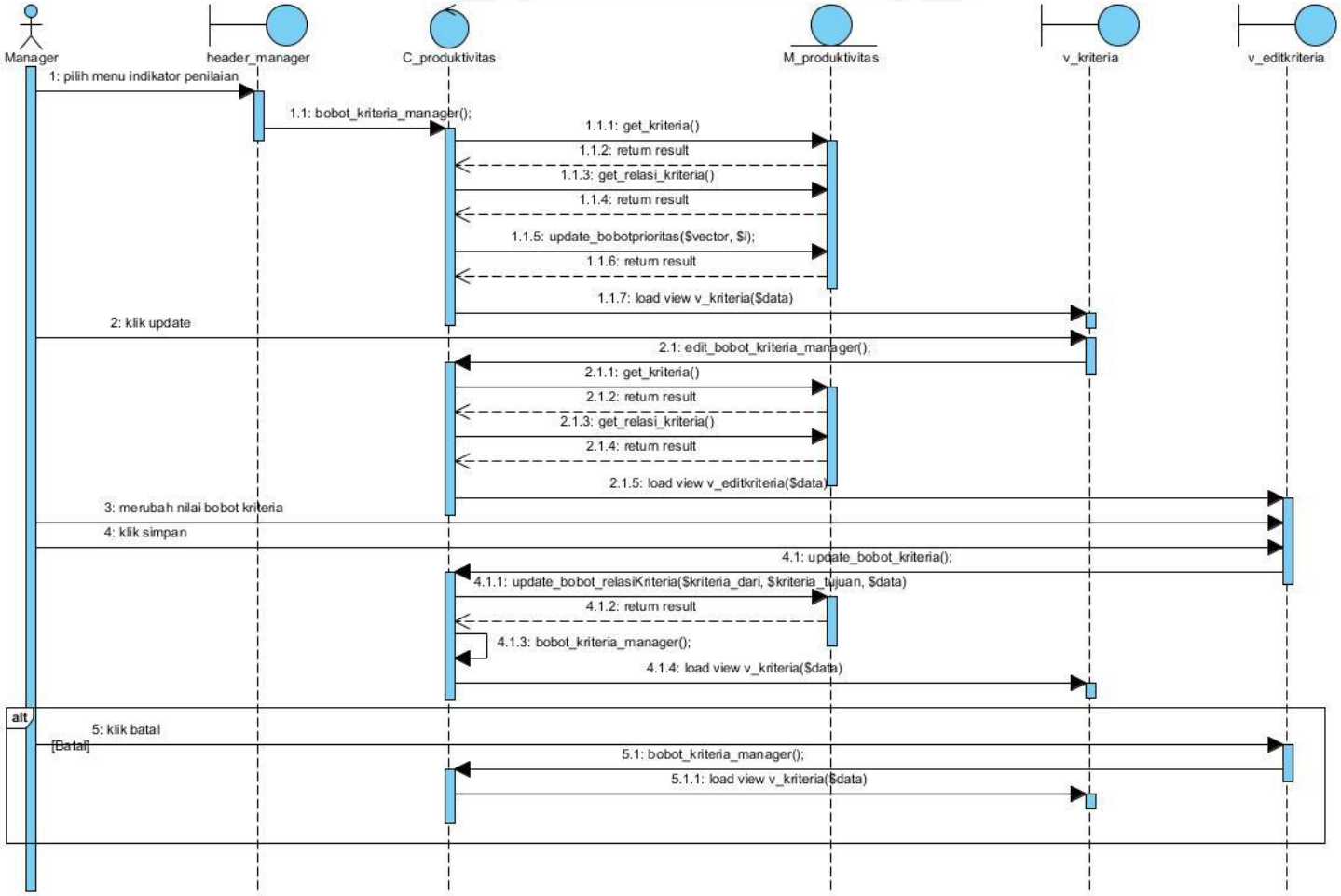


Gambar C. 15 *Sequence* diagram *view* data *check out* (manager)



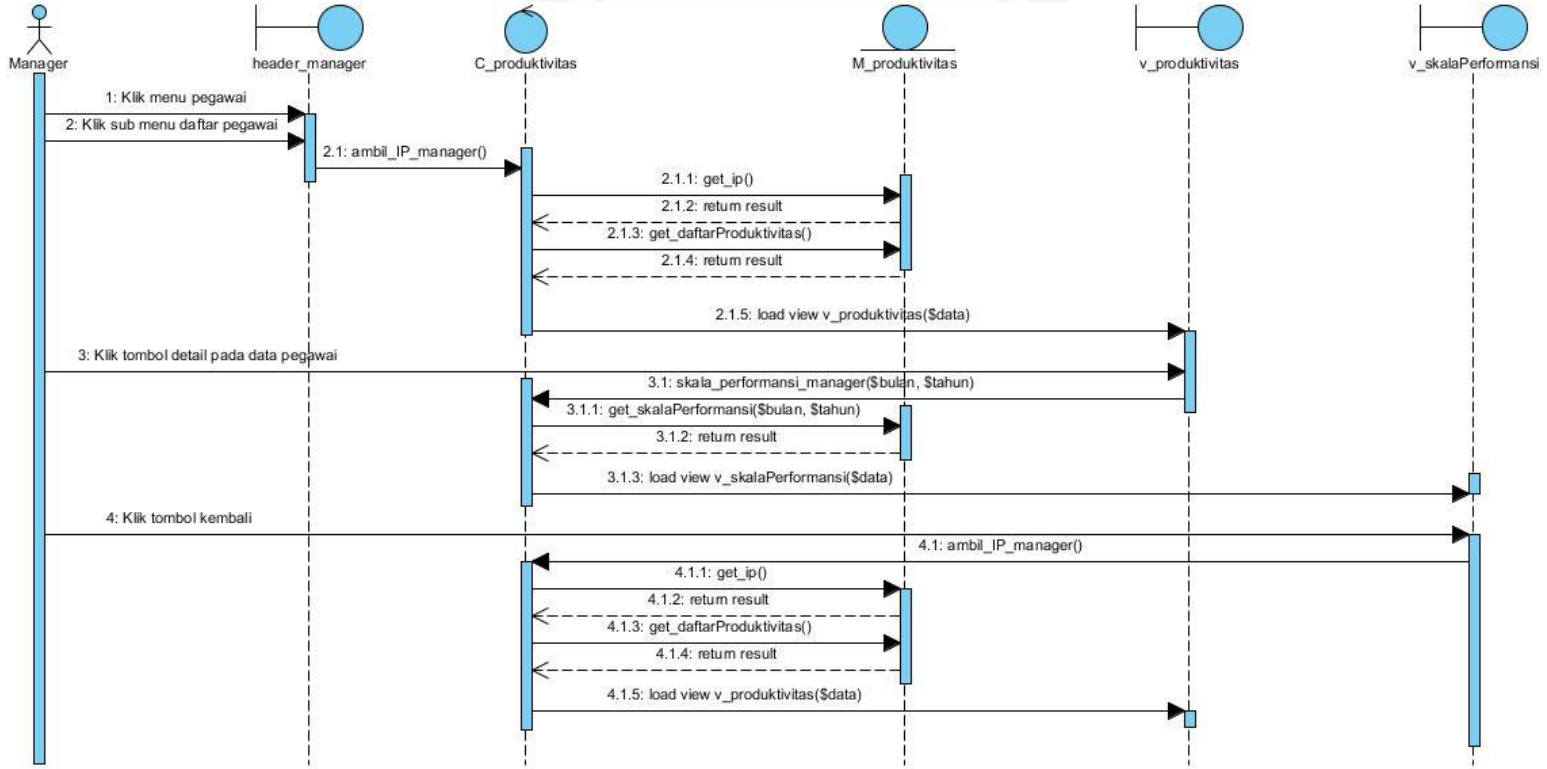
Gambar C. 16 *Sequence* diagram *view* data *check out (front office)*

C.10 *Sequence* Diagram *Update* Bobot Kriteria



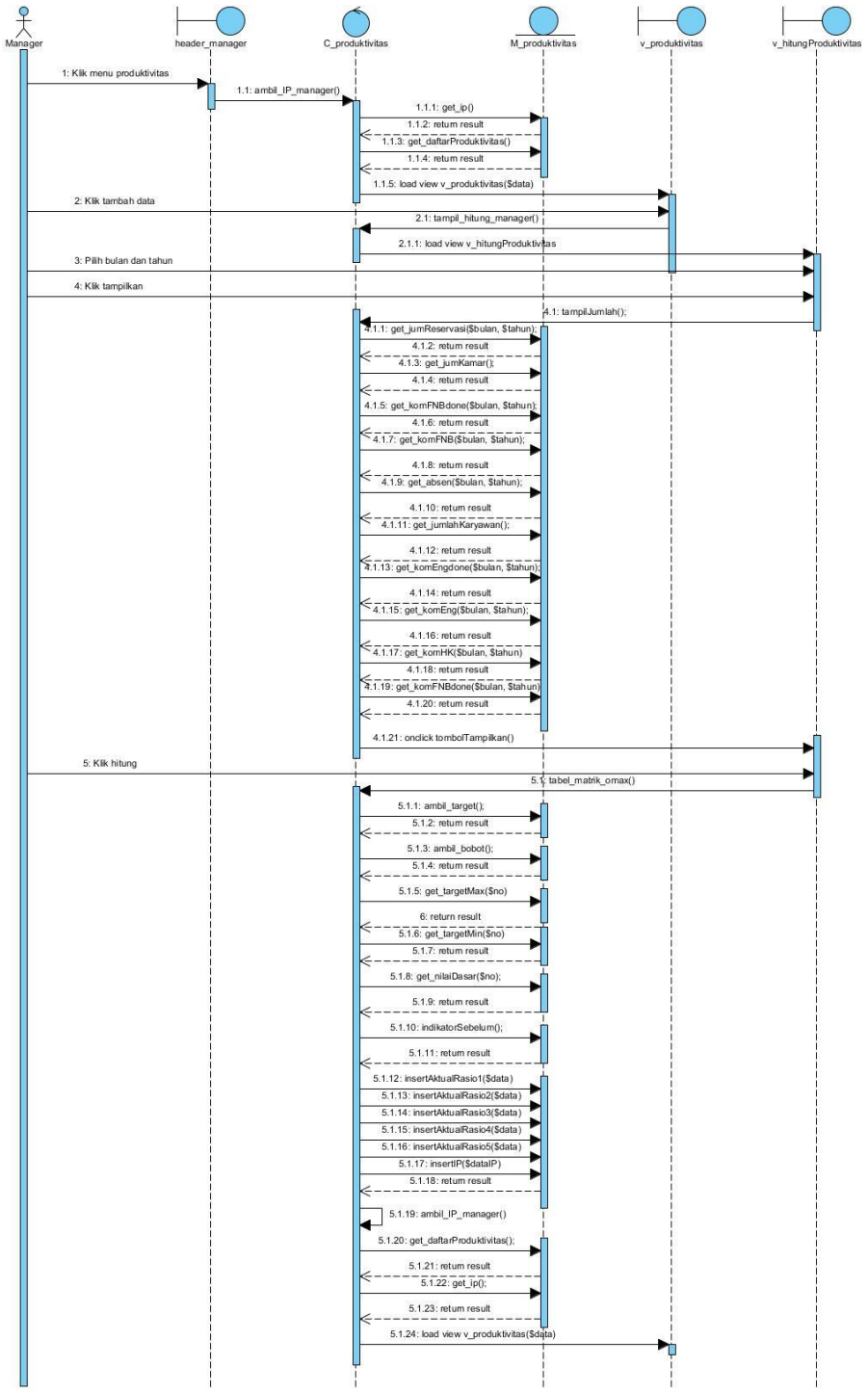
Gambar C. 17 *Sequence* diagram *update* bobot kriteria (manager)

C.11 *Sequence* Diagram *View* Produktivitas



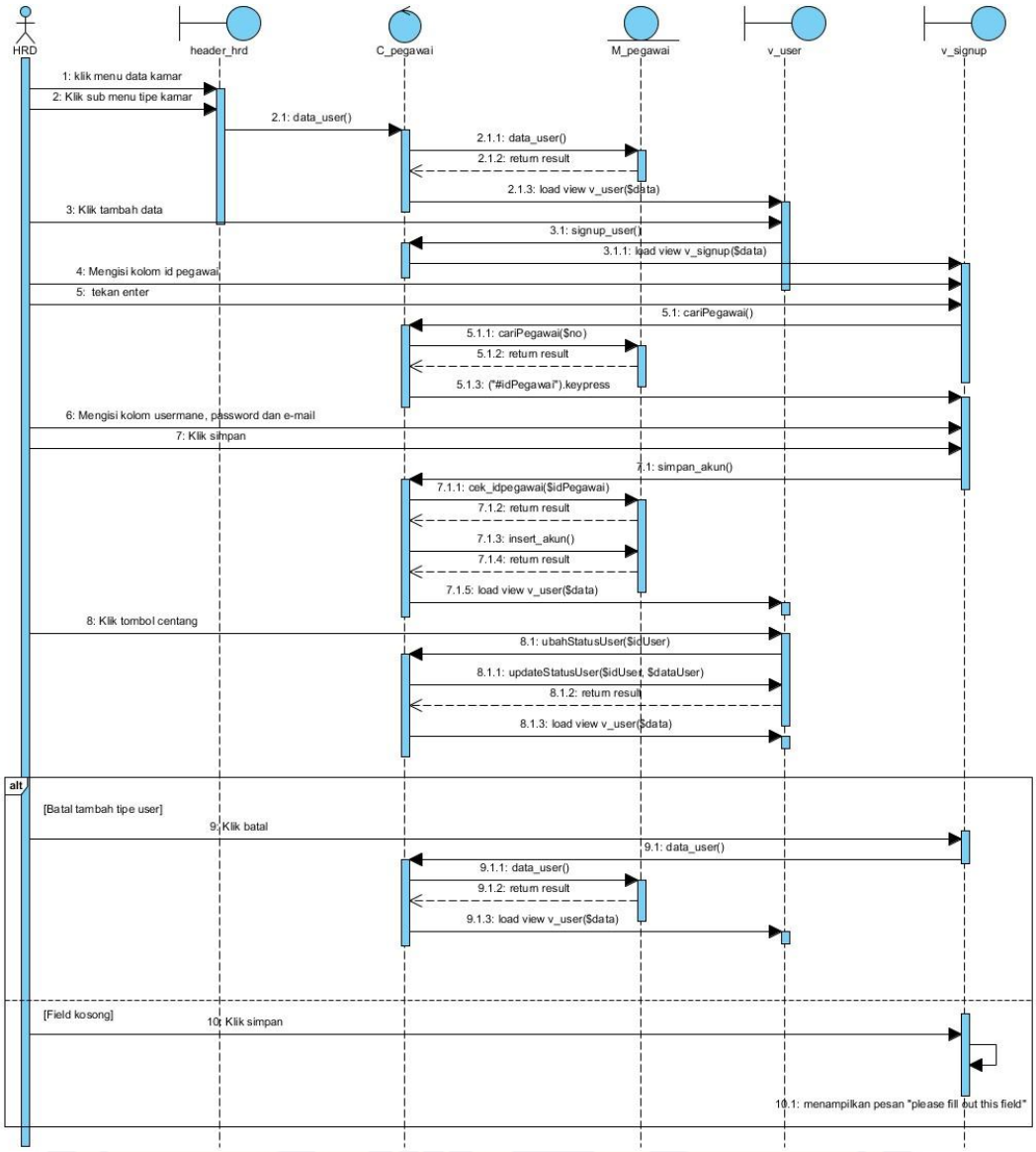
Gambar C. 18 *Sequence* diagram *view* produktivitas (manager)

C.12 *Sequence* Diagram *Input* Perhitungan Produktivitas



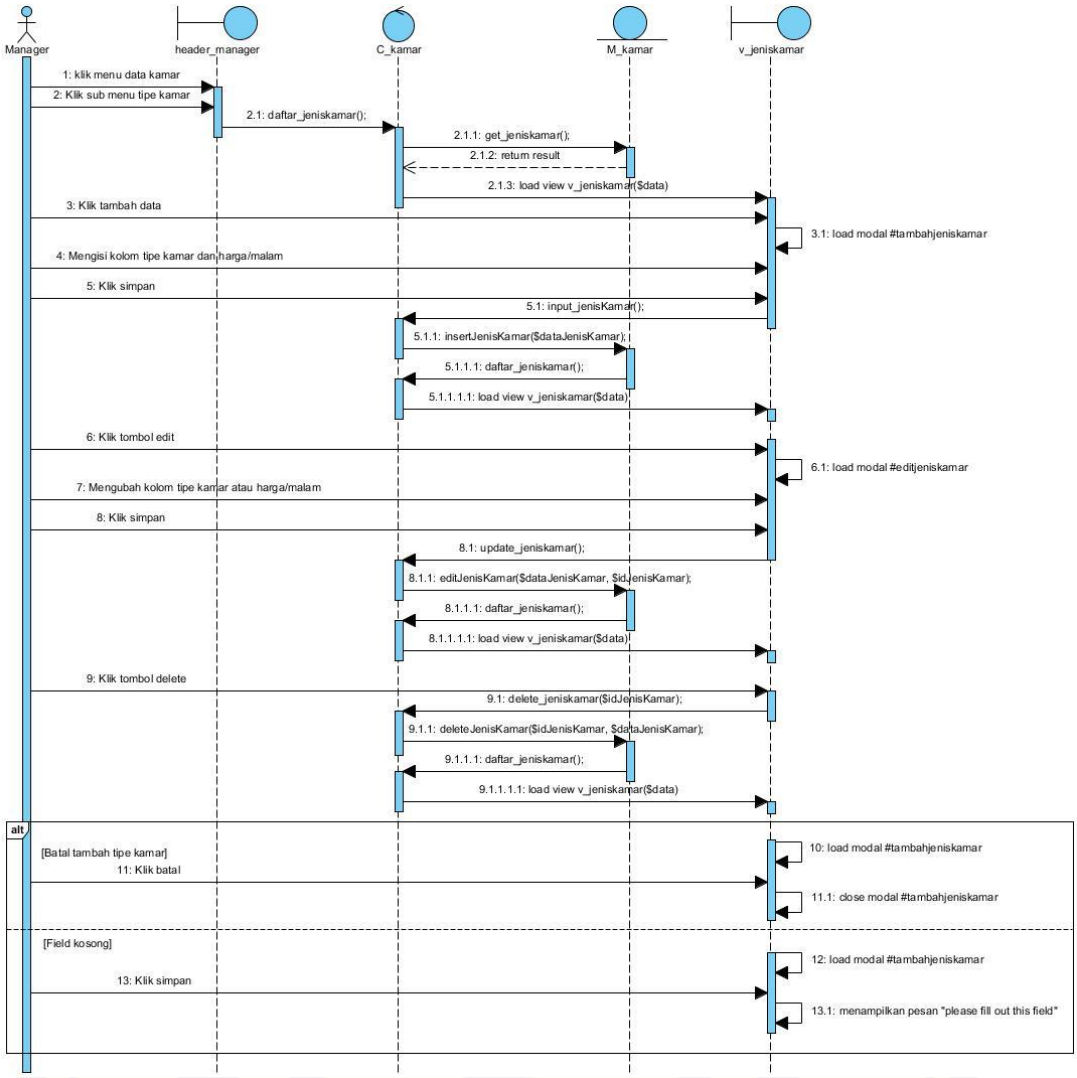
Gambar C. 19 *Sequence* diagram *input* perhitungan produktivitas (manager)

C.13 *Sequence* Diagram Mengelola Data *User*



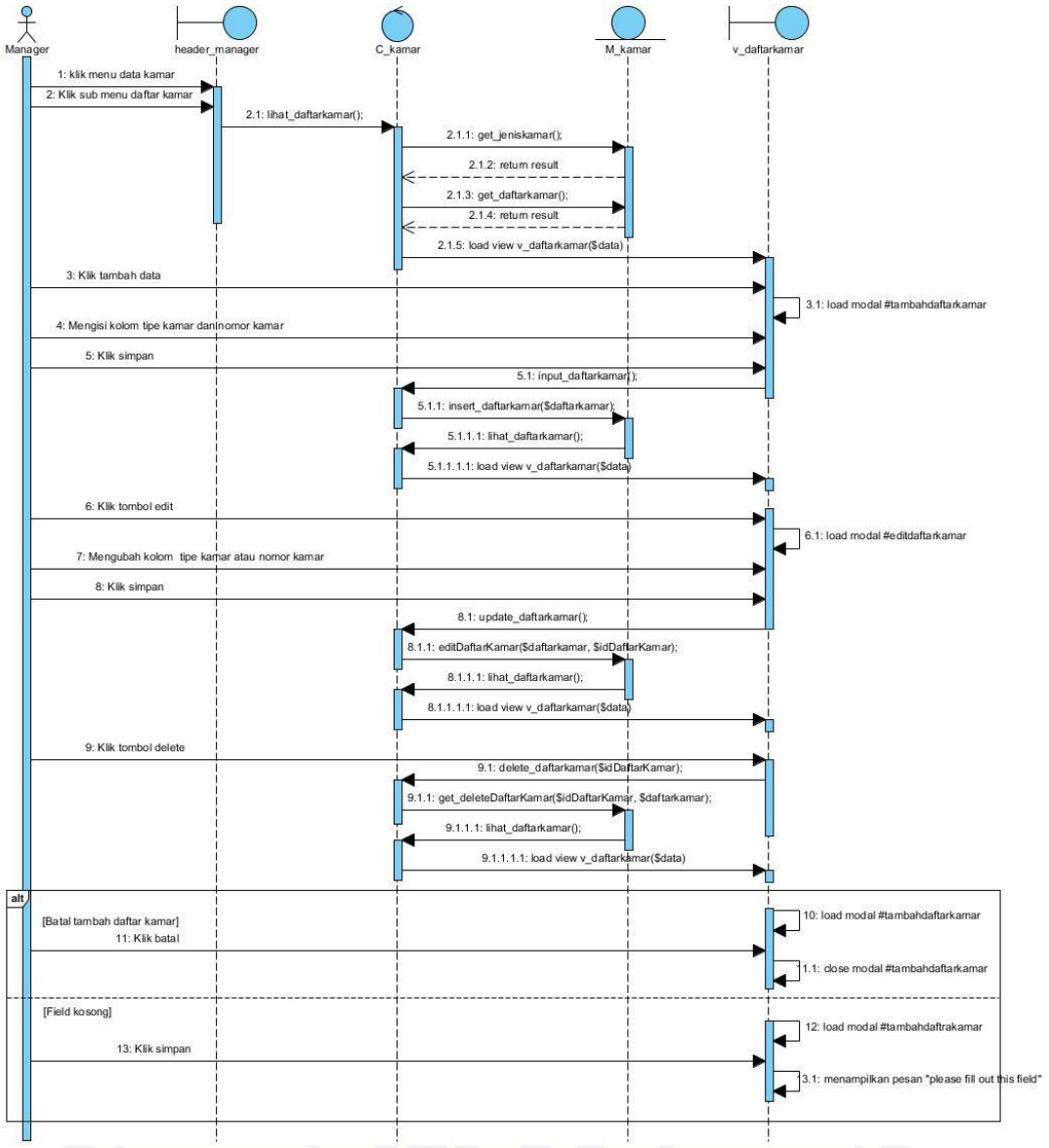
Gambar C. 20 *Sequence* diagram mengelola data *user* (HRD)

C.14 *Sequence* Diagram Mengelola Data Tipe Kamar



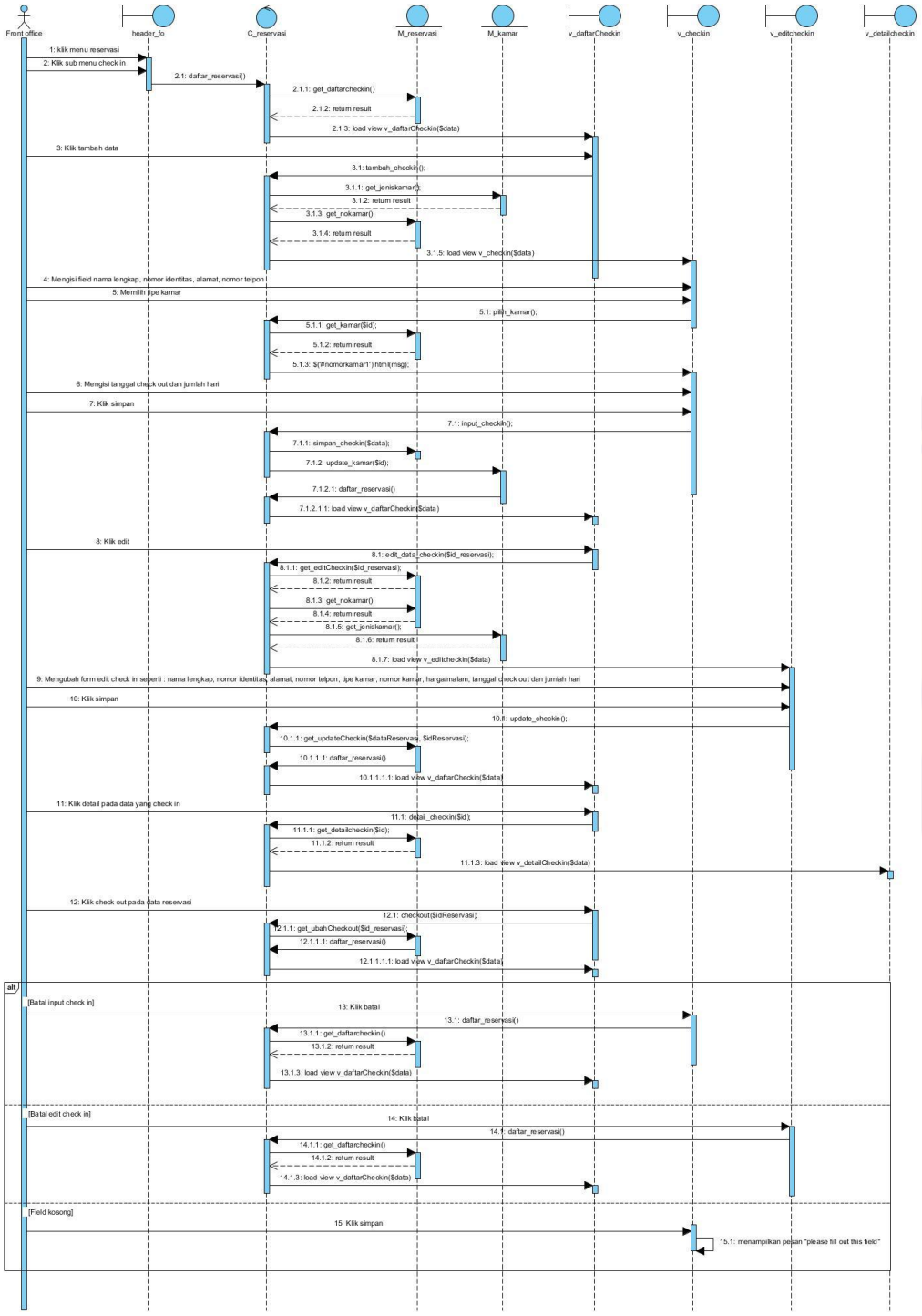
Gambar C. 21 *Sequence* diagram mengelola data tipe kamar (manager)

C.15 *Sequence* Diagram Mengelola Data Daftar Kamar



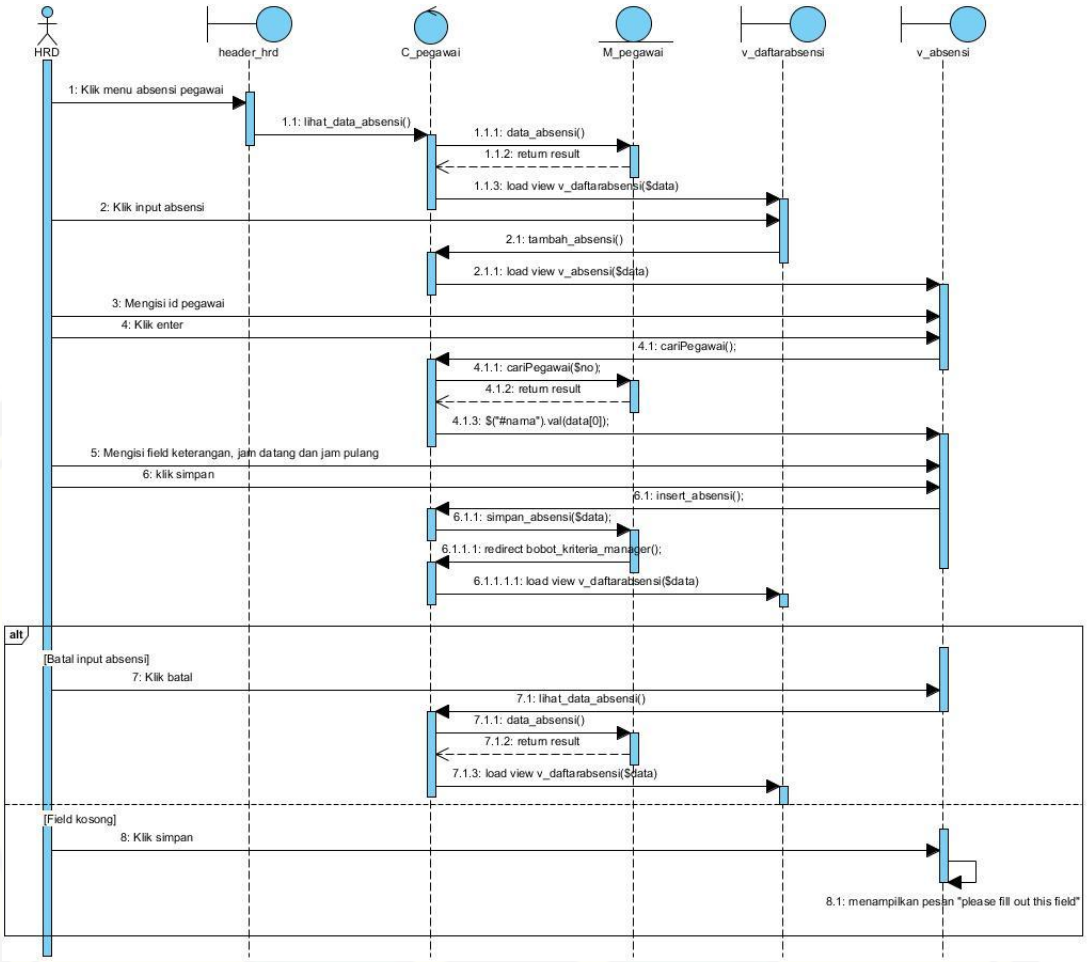
Gambar C. 22 *Sequence* diagram mengelola data daftar kamar (manager)

C.16 *Sequence* Diagram Mengelola Data Reservasi



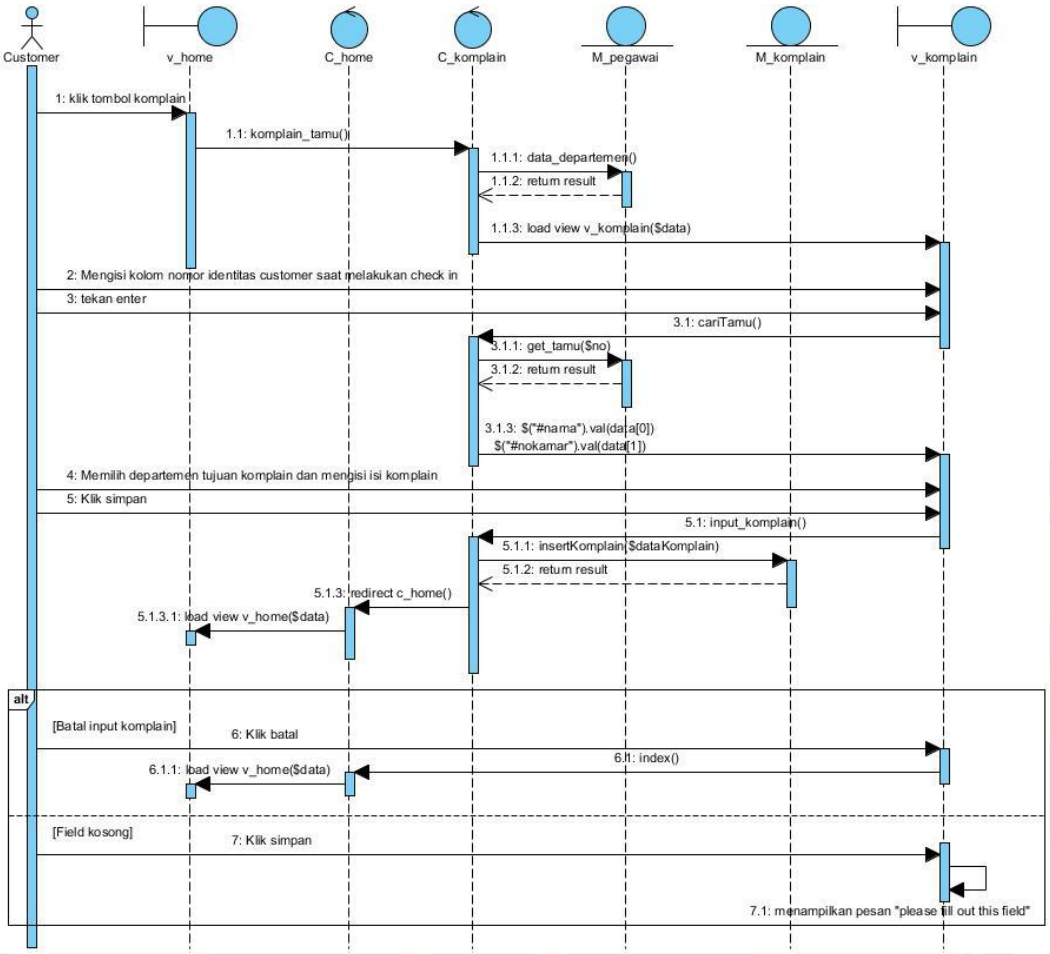
Gambar C. 23 *Sequence* diagram mengelola data reservasi (*front office)*

C.17 *Sequence* Diagram *Input* Absensi Pegawai



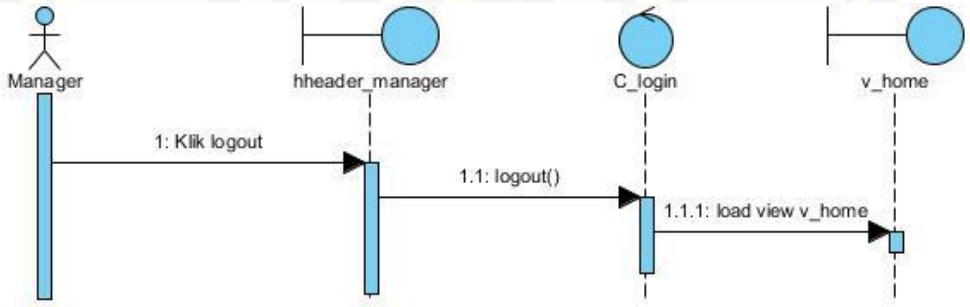
Gambar C. 24 *Sequence* diagram input absensi pegawai (HRD)

C.18 *Sequence* Diagram *Input* Komplain



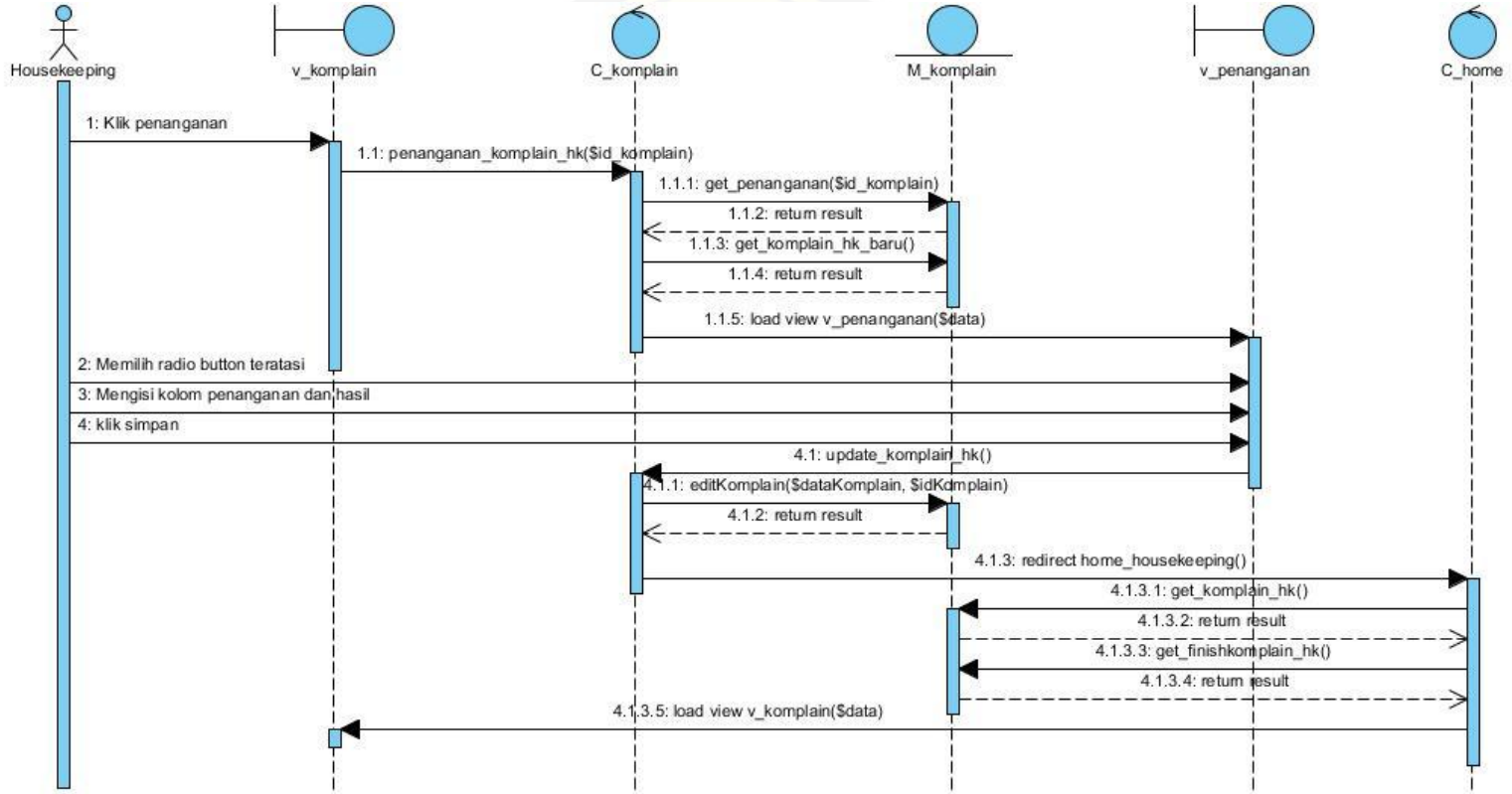
Gambar C. 25 *Sequence* diagram input komplain (*customer*)

C.19 *Sequence* Diagram *Logout*



Gambar C. 26 *Sequence* Diagram *Logout* (semua aktor)

C.20 *Sequence* Diagram *Input* Penanganan Komplain



Gambar C. 27 *Sequence* diagram *input* penanganan komplain (*housekeeping)*

LAMPIRAN D. KODE PROGRAM

D.1. Kode program *class controller*/c_login

```
<?php

if (!defined('BASEPATH'))
    exit('No direct script access allowed');

class C_login extends CI_Controller {

    public function __construct() {
        parent::__construct();
        $this->load->model('M_master_username');
    }

    function index() {
        $session = $this->M_master_username->get_session_level();
        if (!$session['session_level']) {
            $this->load->view('v_login');
        } else {
            if ($session['session_level'] == 'Manager') {
                redirect('C_home/home_manager');
            } else if ($session['session_level'] == 'Engineering') {
                redirect('C_home/home_engineering');
            } else if ($session['session_level'] == 'Human Resource Developmen') {
                redirect('C_home/home_hrd');
            } else if ($session['session_level'] == 'Front Office') {
                redirect('C_home/home_fo');
            } else if ($session['session_level'] == 'Housekeeping') {
                redirect('C_home/home_housekeeping');
            } else if ($session['session_level'] == 'Food & Beverage') {
                redirect('C_home/home_fnb');
            } else {
                redirect('C_login');
            }
        }
    }

    function proses() {
        $username = $this->input->post('username');
        $password = $this->input->post('password');
```

```
        $this->load->model('M_master_username');
        $logic = $this->M_master_username->get_login($username, $password);
        if ($logic == false) {
            $this->session->set_flashdata('message', 'Username atau Password Salah!');
            redirect('C_login');
        } else {
            $coba = $this->M_master_username->get_data($username, $password);
            foreach ($coba->result_array() as $d) {
                $data = array(
                    'username' => $d['username'],
                    'level' => $d['level'],
                );
            }
            $this->M_master_username->store_session($data);
            redirect('C_login');
        }
    }

    function logout() {
        $this->M_master_username->destroy_session();
        redirect('C_home');
    }

}

?>
```

D.2. Kode program *class controller*/c_home

```
<?php

if (!defined('BASEPATH'))
    exit('No direct script access allowed');

class C_home extends CI_Controller {

    function __construct() {
        parent::__construct();
        $this->load->model('M_kamar');
        $this->load->model('M_reservasi');
        $this->load->model('M_komplain');
    }

    public function index() {
        $this->load->view('v_home');
```

```
}

//menampilkan halaman home front office
function home_fo(){
    $data['jmlReservasi'] = $this->M_reservasi->jmlReservasi();
    $data['jmlKamar'] = $this->M_kamar->jmlKamar();
    $this->load->view('frontoffice/header_fo', $data);
    $this->load->view('frontoffice/v_homefo', $data);
}

//menampilkan halaman home manager
function home_manager(){
    $data['jmlReservasi'] = $this->M_reservasi->jmlReservasi();
    $data['jmlKamar'] = $this->M_kamar->jmlKamar();
    $data['jmlKomplain'] = $this->M_komplain->jmlKomplain();
    $this->load->view('manager/header_manager', $data);
    $this->load->view('manager/v_homemanager', $data);
}

//menampilkan halaman home hrd
function home_hrd(){
    $data['jmlReservasi'] = $this->M_reservasi->jmlReservasi();
    $data['jmlKamar'] = $this->M_kamar->jmlKamar();
    $data['jmlKomplain'] = $this->M_komplain->jmlKomplain();
    $this->load->view('hrd/header_hrd', $data);
    $this->load->view('hrd/v_homehrd', $data);
}


// menampilkan halaman komplain (admin)
function home_engineering(){
    $data=array(
        'title'=>'Data Komplain',
        'get_komplain_eng'=>$this->M_komplain->get_komplain_eng(),
        'get_finishkomplain_eng'=>$this->M_komplain->get_finishkomplain_eng(),
    );
    $data['get_komplain_eng_baru'] = $this->M_komplain->get_komplain_eng_baru();
    $this->load->view('engineering/header_engineering', $data);
    $this->load->view('engineering/v_komplain', $data);
}

// menampilkan halaman komplain (admin)
function home_housekeeping(){
    $data=array(
        'title'=>'Data Komplain',
        'get_komplain_hk'=>$this->M_komplain->get_komplain_hk(),
```

```
            'get_finishkomplain_hk'=>$this->M_komplain->get_finishkomplain_hk(),
        );
        $data['get_komplain_hk_baru'] = $this->M_komplain->get_komplain_hk_baru();
        $this->load->view('housekeeping/header_hk', $data);
        $this->load->view('housekeeping/v_komplain', $data);
    }

    // menampilkan halaman komplain (admin)
    function home_fnb(){
        $data=array(
            'title'=>'Data Komplain',
            'get_komplain_fnb'=>$this->M_komplain->get_komplain_fnb(),
            'get_finishkomplain_fnb'=>$this->M_komplain->get_finishkomplain_fnb(),
        );
        $data['get_komplain_fnb_baru'] = $this->M_komplain->get_komplain_fnb_baru();
        $this->load->view('fnb/header_fnb', $data);
        $this->load->view('fnb/v_komplain', $data);
    }
}

?>
```

D.3. Kode program *class controller*/c_kamar

```
<?php

if (!defined('BASEPATH'))
    exit('No direct script access allowed');

class C_kamar extends CI_Controller {

    public function __construct() {
        parent::__construct();
        $this->load->model('M_kamar');
    }

     //menampilkan halaman tipe kamar (hrd)
    function daftar_jeniskamar(){
        $data=array(
            'title'=>'Data Jeniskamar',
            'get_jeniskamar'=>$this->M_kamar->get_jeniskamar(),
        );
        $this->load->view('manager/header_manager',$data);
        $this->load->view('manager/v_jeniskamar',$data);
    }
```

```
//menyimpan data tipe kamar ke database (hrd)
function input_jenisKamar(){
    $dataJenisKamar=array(
        'id_jeniskamar'=>'',
        'jenisKamar'=>'',
        'harga'=>'',
        'status'=> 'ada',
    );

    $dataJenisKamar['jenisKamar']         = $this->input->post('jenis');
    $dataJenisKamar['harga']              = $this->input->post('harga');
    $this->M_kamar->insertJenisKamar($dataJenisKamar);
    redirect(site_url('C_kamar/daftar_jeniskamar'));
}

//mengubah data tipe kamar (hrd)
function update_jeniskamar(){
    $dataJenisKamar=array(
        'id_jeniskamar'=>'',
        'jenisKamar'=>'',
        'harga'=>'',
        'status'=>'ada',

    );
    $idJenisKamar                       = $this->input->post('iidjenis');
    $dataJenisKamar['id_jeniskamar']       = $this->input->post('iidjenis');
    $dataJenisKamar['jenisKamar']         = $this->input->post('inamajenis');
    $dataJenisKamar['harga']              = $this->input->post('iharga');
    $this->M_kamar->editJenisKamar($dataJenisKamar, $idJenisKamar);
    redirect(site_url('C_kamar/daftar_jeniskamar'));
}

//menghapus data tipe kamar (hrd)
function delete_jeniskamar($idJenisKamar){
    $dataJenisKamar=array(
        'status'=>'hapus',
    );
    $this->M_kamar->deleteJenisKamar($idJenisKamar, $dataJenisKamar);
    redirect(site_url('C_kamar/daftar_jeniskamar'));
}

//menampilkan halaman daftar kamar (hrd)
function lihat_daftarkamar(){
    $data=array(
        'title'=>'Daftar Kamar',
```

```
            'get_jeniskamar'=>$this->M_kamar->get_jeniskamar(),
            'get_daftarkamar'=>$this->M_kamar->get_daftarkamar(),
        );
        $this->load->view('manager/header_manager', $data);
        $this->load->view('manager/v_daftarkamar', $data);
    }

    //menyimpan data daftar kamar ke database(hrd)
    function input_daftarkamar(){
        $daftarkamar=array(
            'id_daftarkamar'=>'',
            'id_jeniskamar'=>'',
            'nomorkamar'=>'',
            'status'=>'TERSEDIA',
            'statushapus'=> '1',
        );

        $daftarkamar['id_jeniskamar']           = $this->input->post('jeniskamar');
        $daftarkamar['nomorkamar']              = $this->input->post('nomorkamar');

        $this->M_kamar->insert_daftarkamar($daftarkamar);

        redirect(site_url('C_kamar/lihat_daftarkamar'));
    }

    //mengubah data daftar kamar (hrd)
    function update_daftarkamar(){
        $daftarkamar=array(
            'id_daftarkamar'=>'',
            'id_jeniskamar'=>'',
            'nomorkamar'=>'',

        );
        $idDaftarKamar                      = $this->input->post('ikamar');
        $daftarkamar['id_daftarkamar']          = $this->input->post('ikamar');
        $daftarkamar['id_jeniskamar']           = $this->input->post('ijeniskamar');
        $daftarkamar['nomorkamar']              = $this->input->post('inomorkamar');
        $this->M_kamar->editDaftarKamar($daftarkamar, $idDaftarKamar);
        redirect(site_url('C_kamar/lihat_daftarkamar'));
    }

    //menghapus data daftar kamar dari database (hrd)
    function delete_daftarkamar($idDaftarKamar){
        $daftarkamar=array(
            'statushapus'=>'0',
        );
```

```
        $this->M_kamar->get_deleteDaftarKamar($idDaftarKamar, $daftarkamar);
        redirect(site_url('C_kamar/lihat_daftarkamar'));
    }

    //menampilkan halaman tipe kamar (FO)
    function daftar_tipekamar(){
        $data=array(
            'title'=>'Data Jeniskamar',
            'get_jeniskamar'=>$this->M_kamar->get_jeniskamar(),
        );
        $this->load->view('frontoffice/header_fo',$data);
        $this->load->view('frontoffice/v_jeniskamar',$data);
    }

    //menampilkan halaman daftar kamar (FO)
    function daftar_kamar(){
        $data=array(
            'title'=>'Daftar Kamar',
            'get_jeniskamar'=>$this->M_kamar->get_jeniskamar(),
            'get_daftarkamar'=>$this->M_kamar->get_daftarkamar(),
        );
        $this->load->view('frontoffice/header_fo', $data);
        $this->load->view('frontoffice/v_daftarkamar', $data);
    }

    //menampilkan data kamar yang tersedia (FO)
    function lihat_kamarTersediaFo(){
        $data=array(
            'title'=>'Daftar Kamar',
            'get_daftarkamar'=>$this->M_kamar->get_daftarkamarTersedia(),
        );
        $this->load->view('frontoffice/header_fo', $data);
        $this->load->view('frontoffice/v_daftarkamar', $data);
    }

    //menampilkan data kamar yang tersedia (manager)
    function lihat_kamarTersedia(){
        $data=array(
            'title'=>'Daftar Kamar',
            'get_daftarkamar'=>$this->M_kamar->get_daftarkamarTersedia(),
        );
        $this->load->view('manager/header_manager', $data);
        $this->load->view('manager/v_daftarkamar', $data);
    }

}
```

```
?>
```

D.4. Kode program *class controller*/c_komplain

```
<?php

if (!defined('BASEPATH'))
    exit('No direct script access allowed');

class C_komplain extends CI_Controller {

    public function __construct() {
        parent::__construct();
        $this->load->model('M_pegawai');
        $this->load->model('M_komplain');
    }

    // menampilkan nama tamu setelah mengisi no id tamu saat checkin  untuk
menginputkan komplain (customer)
    function cariTamu(){
        $no=$this->input->post('no');
        $tamu=$this->M_komplain->get_tamu($no);
        if($tamu->num_rows()>0){
            $tamu=$tamu->row_array();
            echo $tamu['nama']."|".$tamu['nomorkamar'];
        }
    }

    // menampilkan halaman komplain (Customer)
    function komplain_tamu(){
        $data=array(
            'title'=>'Komplain Tamu',
            'data_departemen'=>$this->M_pegawai->data_departemen(),
        );
        $data['tanggalsekarang']=date('Y-m-d');
        $this->load->view('v_komplain', $data);
    }

    //menginputkan komplain ke database (customer)
    function input_komplain(){
        $data=array(
            'id_komplain'   =>",
            'tanggal'       =>$this->input->post('tanggalSekarang'),
            'nomor_id'      =>$this->input->post('no'),
```

```
            'nama'          =>$this->input->post('nama'),
            'nokamar'       =>$this->input->post('nokamar'),
            'id_departemen' =>$this->input->post('departemen'),
            'komplain'      =>$this->input->post('komplain'),
            'status'        =>'BELUM TERATASI',
        );
        $this->M_komplain->insertKomplain($data);
        redirect(site_url('C_home'));
    }

    // menampilkan halaman komplain (fo)
    function daftar_komplain_fo(){
            $data=array(
            'title'=>'Data Komplain',
            'get_komplain_fo'=>$this->M_komplain->get_komplain_fo(),
            'get_finishkomplain_fo'=>$this->M_komplain->get_finishkomplain_fo(),

        );
        $this->load->view('frontoffice/header_fo', $data);
        $this->load->view('frontoffice/v_komplain', $data);
    }

    // menampilkan halaman detail komplain yang telah teratasi (fo)
    function tinjau_komplain_fo($id){
        $data['get_detailkomplain']=$this->M_komplain->get_penanganan($id);
        $this->load->view('frontoffice/header_fo',$data);
        $this->load->view('frontoffice/v_tinjaukomplain', $data);
    }

    // menampilkan halaman penanganan komplain (fo)
    function penanganan_komplain_fo($id_komplain) {
        $data= array(
            'get_penanganan'=> $this->M_komplain->get_penanganan($id_komplain),
        );
        $this->load->view('frontoffice/v_penanganan', $data);
        $this->load->view('frontoffice/header_fo', $data);
    }

    // menberikan penanganan atas komplain dari customer (fo)
    function update_komplain_fo(){
        $dataKomplain=array(
            'status'=>'',
            'penanganan1'=>'',
            'hasil1'=>'',
            'penanganan2'=>'',
            'hasil2'=>'',
```

```
            'penanganan3'=>'',
            'hasil3'=>'',
        );
        $idKomplain                 = $this->input->post('id');
        $dataKomplain['status']         = $this->input->post('status');
        $dataKomplain['penanganan1']    = $this->input->post('penanganan1');
        $dataKomplain['hasil1']         = $this->input->post('hasil1');
        $dataKomplain['penanganan2']    = $this->input->post('penanganan2');
        $dataKomplain['hasil2']         = $this->input->post('hasil2');
        $dataKomplain['penanganan3']    = $this->input->post('penanganan3');
        $dataKomplain['hasil3']         = $this->input->post('hasil3');

        $this->M_komplain->editKomplain($dataKomplain, $idKomplain);
        redirect(site_url('C_komplain/daftar_komplain_fo'));
    }

    // menampilkan halaman data komplain (manager)
    function data_komplain(){
        $data=array(
            'title'=>'Data Komplain',
            'get_komplain'=>$this->M_komplain->get_komplain(),
        );
        $this->load->view('manager/header_manager', $data);
        $this->load->view('manager/v_komplain', $data);
    }

    // menampilkan data komplain yang belum teratasi (manager)
    function daftar_komplainBelum(){
        $data=array(
            'title'=>'Data Komplain',
            'get_komplain'=>$this->M_komplain->get_komplainBelum(),
        );
        $data['tanggalsekarang']=date('Y-m-d');
        $this->load->view('manager/header_manager', $data);
        $this->load->view('manager/v_komplain', $data);
    }

    // menampilkan halaman detail komplain yang telah teratasi (manager)
    function tinjau_komplain_manager($id){
        $data['get_detailkomplain']=$this->M_komplain->get_penanganan($id);
        $this->load->view('manager/header_manager',$data);
        $this->load->view('manager/v_tinjaukomplain', $data);
    }

    // menampilkan halaman penanganan komplain (eng)
    function penanganan_komplain_eng($id_komplain) {
```

```
    $data= array(
        'get_penanganan'=> $this->M_komplain->get_penanganan($id_komplain),
    );
    $data['get_komplain_eng_baru'] = $this->M_komplain->get_komplain_eng_baru();
    $this->load->view('engineering/v_penanganan', $data);
    $this->load->view('engineering/header_engineering', $data);
}

// menampilkan halaman detail komplain yang telah teratasi (eng)
function tinjau_komplain_eng($id){
    $data['get_detailkomplain']=$this->M_komplain->get_penanganan($id);
    $data['get_komplain_eng_baru'] = $this->M_komplain->get_komplain_eng_baru();
    $this->load->view('engineering/header_engineering',$data);
    $this->load->view('engineering/v_tinjaukomplain', $data);
}

// memberikan penanganan atas komplain dari customer (eng)
function update_komplain_eng(){
    $dataKomplain=array(
        'status'=>'',
        'penanganan1'=>'',
        'hasil1'=>'',
        'penanganan2'=>'',
        'hasil2'=>'',
        'penanganan3'=>'',
        'hasil3'=>'',
    );
    $idKomplain                 = $this->input->post('id');
    $dataKomplain['status']         = $this->input->post('status');
    $dataKomplain['penanganan1']    = $this->input->post('penanganan1');
    $dataKomplain['hasil1']         = $this->input->post('hasil1');
    $dataKomplain['penanganan2']    = $this->input->post('penanganan2');
    $dataKomplain['hasil2']         = $this->input->post('hasil2');
    $dataKomplain['penanganan3']    = $this->input->post('penanganan3');
    $dataKomplain['hasil3']         = $this->input->post('hasil3');

    $this->M_komplain->editKomplain($dataKomplain, $idKomplain);

    redirect(site_url('C_home/home_engineering'));
}


// menampilkan halaman detail komplain yang telah teratasi (fnb)
function tinjau_komplain_fnb($id){
    $data['get_detailkomplain']=$this->M_komplain->get_penanganan($id);
    $data['get_komplain_fnb_baru'] = $this->M_komplain->get_komplain_fnb_baru();
```

```
    $this->load->view('fnb/header_fnb',$data);
    $this->load->view('fnb/v_tinjaukomplain', $data);
}

// menampilkan halaman penanganan komplain (fnb)
function penanganan_komplain_fnb($id_komplain) {
    $data= array(
        'get_penanganan'=> $this->M_komplain->get_penanganan($id_komplain),
    );
    $data['get_komplain_fnb_baru'] = $this->M_komplain->get_komplain_fnb_baru();
    $this->load->view('fnb/v_penanganan', $data);
    $this->load->view('fnb/header_fnb', $data);
}

// memberikan penanganan atas komplain dari customer (fnb)
function update_komplain_fnb(){
    $dataKomplain=array(
        'status'=>'',
        'penanganan1'=>'',
        'hasil1'=>'',
        'penanganan2'=>'',
        'hasil2'=>'',
        'penanganan3'=>'',
        'hasil3'=>'',
    );
    $idKomplain                 = $this->input->post('id');
    $dataKomplain['status']         = $this->input->post('status');
    $dataKomplain['penanganan1']    = $this->input->post('penanganan1');
    $dataKomplain['hasil1']         = $this->input->post('hasil1');
    $dataKomplain['penanganan2']    = $this->input->post('penanganan2');
    $dataKomplain['hasil2']         = $this->input->post('hasil2');
    $dataKomplain['penanganan3']    = $this->input->post('penanganan3');
    $dataKomplain['hasil3']         = $this->input->post('hasil3');

    $this->M_komplain->editKomplain($dataKomplain, $idKomplain);

    redirect(site_url('C_home/home_fnb'));
}

// menampilkan halaman detail komplain yang telah teratasi (hk)
function tinjau_komplain_hk($id){
    $data['get_detailkomplain']=$this->M_komplain->get_penanganan($id);
    $data['get_komplain_hk_baru'] = $this->M_komplain->get_komplain_hk_baru();
    $this->load->view('housekeeping/header_hk',$data);
    $this->load->view('housekeeping/v_tinjaukomplain', $data);
}
```

```
    // menampilkan halaman penanganan komplain (hk)
    function penanganan_komplain_hk($id_komplain) {
        $data= array(
            'get_penanganan'=> $this->M_komplain->get_penanganan($id_komplain),
        );
        $data['get_komplain_hk_baru'] = $this->M_komplain->get_komplain_hk_baru();
        $this->load->view('housekeeping/v_penanganan', $data);
        $this->load->view('housekeeping/header_hk', $data);
    }

    // memberikan penanganan atas komplain dari customer (hk)
    function update_komplain_hk(){
        $dataKomplain=array(
            'status'=>'',
            'penanganan1'=>'',
            'hasil1'=>'',
            'penanganan2'=>'',
            'hasil2'=>'',
            'penanganan3'=>'',
            'hasil3'=>'',
        );
        $idKomplain                 = $this->input->post('id');
        $dataKomplain['status']         = $this->input->post('status');
        $dataKomplain['penanganan1']    = $this->input->post('penanganan1');
        $dataKomplain['hasil1']         = $this->input->post('hasil1');
        $dataKomplain['penanganan2']    = $this->input->post('penanganan2');
        $dataKomplain['hasil2']         = $this->input->post('hasil2');
        $dataKomplain['penanganan3']    = $this->input->post('penanganan3');
        $dataKomplain['hasil3']         = $this->input->post('hasil3');

        $this->M_komplain->editKomplain($dataKomplain, $idKomplain);

        redirect(site_url('C_home/home_housekeeping'));
    }

}

?>
```

D.5. Kode program *class controller*/c_pegawai

```
<?php

if (!defined('BASEPATH'))
    exit('No direct script access allowed');

class C_pegawai extends CI_Controller {

    public function __construct() {
        parent::__construct();
        $this->load->model('M_pegawai');
    }

    // menampilkan nama pegawai setelah mengisi id pegawai
    function cariPegawai(){
        $no=$this->input->post('no');
        $pegawai=$this->M_pegawai->cariPegawai($no);
        if($pegawai->num_rows()>0){
            $pegawai=$pegawai->row_array();
            echo $pegawai['nama']."|".$pegawai['namaDepartemen'];
        }
    }

    // menampilkan halaman data pegawai (manager)
    function daftar_pegawaihotel(){
        $data=array(
            'title'=>'Data Pegawai',
            'data_pegawai'=>$this->M_pegawai->data_pegawai(),
            'data_departemen'=>$this->M_pegawai->data_departemen(),
        );
        $this->load->view('manager/header_manager',$data);
        $this->load->view('manager/v_pegawai',$data);
    }

    //menampilkan halaman detail data pegawai (manager)
    function detail_pegawaihotel($id){
        $data['data']=$this->M_pegawai->get_detail_pegawai($id);
        $this->load->view('manager/header_manager',$data);
        $this->load->view('manager/v_detailPegawai', $data);
    }

    //menampilkan halaman data absensi (manager)
    function lihat_absensi_manager(){
        $data=array(
            'title'=>'Data Absensi',
```

```
            'data_absensi'=>$this->M_pegawai->data_absensi(),
        );
        $this->load->view('manager/header_manager',$data);
        $this->load->view('manager/v_daftarabsensi',$data);
    }

    //menampilkan halaman daftar pegawai (hrd)
    function data_pegawai(){
        $data=array(
            'title'=>'Data Pegawai',
            'data_pegawai'=>$this->M_pegawai->data_pegawai(),
            'data_departemen'=>$this->M_pegawai->data_departemen(),
        );
        $this->load->view('hrd/header_hrd',$data);
        $this->load->view('hrd/v_pegawai',$data);
    }

    //menampilkan halaman detail data pegawai (hrd)
    function detail_data_pegawai($id){
        $data['data']=$this->M_pegawai->get_detail_pegawai($id);
        $this->load->view('hrd/header_hrd',$data);
        $this->load->view('hrd/v_detailPegawai', $data);
    }

    //menginputkan data pegawai ke database (hrd)
    function input_data_pegawai(){
        $dataPegawai=array(
            'id_pegawai'=>'',
            'nama'=>'',
            'tempatlahir'=>'',
            'tanggallahir'=>'',
            'alamat'=>'',
            'id_departemen'=>'',
            'tanggalkerja'=>'',
            'status'=> '1',
        );
        $dataPegawai['nama']                = $this->input->post('nama');
        $dataPegawai['tempatlahir']           = $this->input->post('tempatlahir');
        $dataPegawai['tanggallahir']          = $this->input->post('tanggallahir');
        $dataPegawai['alamat']              = $this->input->post('alamat');
        $dataPegawai['id_departemen']           = $this->input->post('departemen');
        $dataPegawai['tanggalkerja']          = $this->input->post('tanggalkerja');

        $this->M_pegawai->insert_pegawai($dataPegawai);
        redirect(site_url('C_pegawai/data_pegawai'));
    }
```

```
// menghapus data pegawai (hrd)
function delete_pegawai($idPegawai){
    $dataPegawai=array(
        'status'=>'0',
    );
    $dataUser=array(
        'statusHapus'=>'0',
    );
    $this->M_pegawai->deletePegawai($idPegawai, $dataPegawai);
    $this->M_pegawai->delete_user($idPegawai, $dataUser);
    redirect(site_url('C_pegawai/data_pegawai'));
}

//mengubah data pegawai (hrd)
function update_data_pegawai(){
    $dataPegawai=array(
        'id_pegawai'=>'',
        'nama'=>'',
        'tempatlahir'=>'',
        'tanggallahir'=>'',
        'alamat'=>'',
        'id_departemen'=>'',
        'tanggalkerja'=>'',
        'status'=> '1',
    );
    $idPegawai                      = $this->input->post('iidpegawai');
    $dataPegawai['id_pegawai']          = $this->input->post('iidpegawai');
    $dataPegawai['nama']                = $this->input->post('inama');
    $dataPegawai['tempatlahir']         = $this->input->post('itempatlahir');
    $dataPegawai['tanggallahir']        = $this->input->post('itanggallahir');
    $dataPegawai['alamat']              = $this->input->post('ialamat');
    $dataPegawai['id_departemen']       = $this->input->post('idepartemen');
    $dataPegawai['tanggalkerja']        = $this->input->post('itanggalkerja');

    $this->M_pegawai->editPegawai($dataPegawai, $idPegawai);
    redirect(site_url('C_pegawai/data_pegawai'));
}

//melihat data absensi pegawai (hrd)
function lihat_data_absensi(){
    $data=array(
        'title'=>'Data Absensi',
        'data_absensi'=>$this->M_pegawai->data_absensi(),
    );
    $this->load->view('hrd/header_hrd',$data);
```

```
    $this->load->view('hrd/v_daftarabsensi',$data);
}

// menampilkan halaman input absensi (hrd)
function tambah_absensi() {
    $data['title']="absensi";
    $data['tanggalsekarang']=date('Y-m-d');
    $this->load->view('hrd/v_absensi', $data);
    $this->load->view('hrd/header_hrd', $data);
}

// menginputkan absensi pegawai ke database (hrd)
function insert_absensi(){
    $data=array(
        'tanggal'=>$this->input->post('tanggalsekarang'),
        'id_pegawai'=>$this->input->post('no'),
        'keterangan'=>$this->input->post('keterangan'),
        'jamDatang'=>$this->input->post('datang'),
        'jamPulang'=>$this->input->post('pulang'),
    );
    $this->M_pegawai->simpan_absensi($data);
    redirect('C_pegawai/lihat_data_absensi');
}

// halaman sign up
function signup_pegawai(){
    $this->load->view('v_signup');
}

// menyimpan akun
function simpan_akun(){
    $u = $this->input->post('username');
    $id = $this->input->post('no');
    $cek = $this->M_pegawai->cek_uname($u);
    $cek_id = $this->M_pegawai->cek_idpegawai($id);
    if ($cek_id != 1) {
        $this->session->set_flashdata('message', 'ID Pegawai sudah terdaftar!');
        redirect('C_pegawai/signup_pegawai');
    } else if ($cek != 1) {
        $this->session->set_flashdata('message', 'Username sudah terpakai!');
        redirect('C_pegawai/signup_pegawai');
    } else {
        $data=array(
        'id_pegawai'=>$this->input->post('no'),
        'username'=>$this->input->post('username'),
        'password'=>$this->input->post('password'),
```

```
            'email'=>$this->input->post('email'),
            'level'=>$this->input->post('level'),
            'statusUser'=>'BELUM AKTIF',
            'statusHapus'=>'1',
        );
        $this->M_pegawai->insert_akun($data);
        redirect('C_login/logout');
        }
    }

    // menampilkan data user
    function data_user(){
        $data=array(
            'title'=>'Data User',
            'data_user'=>$this->M_pegawai->data_user(),
        );
        $this->load->view('hrd/header_hrd',$data);
        $this->load->view('hrd/v_user',$data);
    }

    // mengganti status user
    function ubahStatusUser($idUser){
        $dataUser=array(
            'statusUser'=>'AKTIF',
        );
        $this->M_pegawai->updateStatusUser($idUser, $dataUser);
        redirect(site_url('C_pegawai/data_user'));
    }

    // halaman sign up
    function signup_user(){
        $this->load->view('hrd/header_hrd');
        $this->load->view('hrd/v_signup');
    }



}

?>
```

D.6. Kode program *class controller*/c_reservasi

```
<?php

if (!defined('BASEPATH'))
    exit('No direct script access allowed');

class C_reservasi extends CI_Controller {

    public function __construct() {
        parent::__construct();
        $this->load->model('M_kamar');
        $this->load->model('M_reservasi');
    }

    // menampilkan harga/malam
    function cariHarga(){
        $idJenisKamar=$this->input->post('idJenisKamar');
        $hari=$this->input->post('hari');
        $harga=$this->M_reservasi->cariHarga($idJenisKamar);
        if($harga->num_rows()>0){
            $harga=$harga->row_array();
            echo $harga['harga'];
        }
    }

    // menampilakan nomor kamar yang tersedia pada saat memilih tipe kamar
    function nomorKamar() {
        $idJenisKamar = $this->input->post('idJenisKamar');
        $nomorKamar = $this->M_reservasi->get_nomorKamar($idJenisKamar);
        $data = "<option value="">--Pilih--</option>";
        foreach ($nomorKamar as $no) {
            $data = "<option value='$no[id_daftarkamar]'>$no[nomorkamar]</option>\n";
        }
         echo $data;
    }

    // menampilkan kombo box daftar kamar
    function pilih_kamar() {
        $id = $this->input->post('idJenisKamar');
        $data['get_kamar'] = $this->M_reservasi->get_kamar($id);
        $this->load->view('frontoffice/v_comboKamar',$data);
    }

    //manampilkan halaman daftar check in (FO)
    function daftar_reservasi(){
```

```
        $data=array(
            'title'=>'Data Reservasi',
            'get_daftarcheckin'=>$this->M_reservasi->get_daftarcheckin(),
        );
        $this->load->view('frontoffice/header_fo',$data);
        $this->load->view('frontoffice/v_daftarcheckin',$data);
    }

    // menampilkan halaman detail check out (FO)
    function detail_checkin_fo($id){
        $data['data']=$this->M_reservasi->get_detailcheckin($id);
        $this->load->view('frontoffice/header_fo',$data);
        $this->load->view('frontoffice/v_detailCheckin', $data);
    }

    // menampilkan halaman input check in (FO)
    function tambah_data_checkin(){
        $data=array(
            'title'=>'Data checkin',
            'get_jeniskamar'=>$this->M_kamar->get_jeniskamar(),
            'get_nokamar'=>$this->M_reservasi->get_nokamar(),
        );
        $data['tanggalsekarang']=date('Y-m-d');
        $data['tanggalkembali'] = strtotime('+7 day',strtotime($data['tanggalsekarang']));
        $data['tanggalkembali'] = date('Y-m-d', $data['tanggalkembali']);
        $this->load->view('frontoffice/header_fo', $data);
        $this->load->view('frontoffice/v_checkin', $data);
    }

    // menginputkan data reservasi ke database (FO)
    function input_checkin_fo(){
        $id = $this->input->post('nomorkamar');
        $data=array(
            'tanggal'=>$this->input->post('tanggalsekarang'),
            'nama'=>$this->input->post('nama'),
            'noid'=>$this->input->post('noid'),
            'alamat'=>$this->input->post('alamat'),
            'id_daftarkamar'=>$this->input->post('nomorkamar'),
            'cekout'=>$this->input->post('cekout'),
            'telp'=>$this->input->post('telp'),
            'status'=>'IN',
            'statusBatal'=>'T',
        );
        $this->M_reservasi->simpan_checkin($data);
        $this->M_reservasi->update_kamar($id);
        redirect(site_url('C_reservasi/daftar_reservasi'));
```

```
}

// menampilkan halaman edit check in (FO)
function edit_data_checkin($id_reservasi) {
    $data= array(
        'get_daftarcheckin'=> $this->M_reservasi->get_editCheckin($id_reservasi),
        'get_jeniskamar'=>$this->M_kamar->get_jeniskamar(),
        'get_nokamar'=>$this->M_reservasi->get_nokamar(),
    );
    $this->load->view('frontoffice/v_editcheckin', $data);
    $this->load->view('frontoffice/header_fo', $data);
}

// mengubah data check in (FO)
function update_checkinn(){
    $dataPegawai=array(
        'id_reservasi'=>'',
        'tanggal'=>'',
        'nama'=>'',
        'noid'=>'',
        'alamat'=>'',
        'telp'=>'',
        'id_daftarkamar'=>'',
        'cekout'=> '',
    );
    $idReservasi                    = $this->input->post('id_reservasi');
    $dataReservasi['id_reservasi']          = $this->input->post('id_reservasi');
    $dataReservasi['tanggal']               = $this->input->post('tanggal');
    $dataReservasi['nama']          = $this->input->post('nama');
    $dataReservasi['noid']          = $this->input->post('noid');
    $dataReservasi['alamat']                = $this->input->post('alamat');
    $dataReservasi['telp']          = $this->input->post('telp');
    $dataReservasi['id_daftarkamar']            = $this->input->post('nomorkamar');
    $dataReservasi['cekout']            = $this->input->post('cekout');

    $this->M_reservasi->get_updateCheckin($dataReservasi, $idReservasi);
    redirect(site_url('C_reservasi/daftar_reservasi'));
}

// mengubah status check in menjadi check out (FO)
function checkout_fo($idReservasi){
    $data['data']=$this->M_reservasi->get_ubahCheckout($idReservasi);
    redirect(site_url('C_reservasi/daftar_reservasi'));
}

// menampilkan halaman daftar check out (FO)
```

```
function daftar_checkout_fo(){
    $data=array(
        'title'=>'Data Checkout',
        'get_daftarcheckout'=>$this->M_reservasi->get_daftarcheckout(),
    );
    $this->load->view('frontoffice/header_fo',$data);
    $this->load->view('frontoffice/v_daftarcheckout',$data);
}

// menampilkan detail check out (FO)
function detail_checkout_fo($id){
    $data['data']=$this->M_reservasi->get_detailcheckin($id);
    $this->load->view('frontoffice/header_fo',$data);
    $this->load->view('frontoffice/v_detailCheckout', $data);
}

//manampilkan halaman daftar check in (FO)
function data_checkin(){
    $data=array(
        'title'=>'Data Reservasi',
        'get_daftarcheckin'=>$this->M_reservasi->get_daftarcheckin(),
    );
    $this->load->view('manager/header_manager',$data);
    $this->load->view('manager/v_daftarcheckin',$data);
}

//menampilkan halaman daftar check in (manager)
function daftar_checkin_manager(){
    $data=array(
        'title'=>'Data Checkin',
        'get_daftarcheckin'=>$this->M_reservasi->get_daftarcheckin(),
    );
    $this->load->view('manager/header_manager',$data);
    $this->load->view('manager/v_daftarCheckin',$data);
}

// menampilkan halaman detail check in (manager)
function detail_checkin_manager($id){
    $data['data']=$this->M_reservasi->get_detailcheckin($id);
    $this->load->view('manager/header_manager',$data);
    $this->load->view('manager/v_detailCheckin', $data);
}

// menampilkan halaman daftar check out (manager)
function daftar_checkout_manager(){
    $data=array(
```

```
            'title'=>'Data Checkout',
            'get_daftarcheckout'=>$this->M_reservasi->get_daftarcheckout(),
        );
        $this->load->view('manager/header_manager',$data);
        $this->load->view('manager/v_daftarCheckout',$data);
    }

    // menampilkan halaman daftar check out (manager)
    function detail_checkout_manager($id){
        $data['data']=$this->M_reservasi->get_detailcheckin($id);
        $this->load->view('manager/header_manager',$data);
        $this->load->view('manager/v_detailCheckout', $data);
    }
}

?>
```

D.7. Kode program *class controller*/c_produktivitas

```
<?php
error_reporting(0);

if (!defined('BASEPATH'))
    exit('No direct script access allowed');

class C_produktivitas extends CI_Controller {

    public function __construct() {
        parent::__construct();
        $this->load->model('M_produktivitas');
    }

    function tampilJumlah(){
        $bulan=$this->input->post('bulan');
        $tahun=$this->input->post('tahun');
        if ($bulan == $this->M_produktivitas->cek_bulan($bulan) && $tahun == $this-
>M_produktivitas->cek_tahun($tahun)) {
            redirect(base_url(), 'refresh');
        } else if($jumReservasi=$this->M_produktivitas->get_jumReservasi($bulan,
$tahun)==0) {
            redirect(base_url(), 'refresh');
        } else {
            $jumReservasi=$this->M_produktivitas->get_jumReservasi($bulan, $tahun);
            $jumKam=$this->M_produktivitas->get_jumKamar();
            $tanggal=date('t', mktime(0, 0, 0, $bulan+1, 0, $tahun));
```

```
            $jumKamarTersedia=$jumKam*$tanggal;
            $jumAb=$this->M_produktivitas->get_absen($bulan, $tahun);
            $jumKar=$this->M_produktivitas->get_jumlahKaryawan();
            $komEngdone=$this->M_produktivitas->get_komEngdone($bulan, $tahun);
            $komEng=$this->M_produktivitas->get_komEng($bulan, $tahun);
            $komFOdone=$this->M_produktivitas->get_komFOdone($bulan, $tahun);
            $komFO=$this->M_produktivitas->get_komFO($bulan, $tahun);

            echo
$jumReservasi."|".$jumKamarTersedia."|".$jumReservasi."|".$tanggal."|".$komFNBdone."
|".$komFNB."|".$jumAb."|".$jumKar."|".$komEngdone."|".$komEng;
        }
    }

    //menampilkan halaman bobot kriteria
    function bobot_kriteria_manager() {
        $data['daftar_kriteria']    = $this->M_produktivitas->get_kriteria()->result();
        $data['relasi_kriteria']    = $this->M_produktivitas->get_relasi_kriteria()->result();
        $data['jumlahKriteria']     = count($data['daftar_kriteria']);

  //Menghitung Jumlah dari setiap kolom
      $jumlah = array();
        foreach ($data['daftar_kriteria'] as $kriteria) {
            $jumlah1 = 0;
            foreach ($data['relasi_kriteria'] as $relasi) {
                if($relasi->kriteria_tujuan == $kriteria->id_rasio){
                    $jumlah1 = $jumlah1 + $relasi->bobot;
                }
            }
            array_push($jumlah, $jumlah1);
         }
        $data['jumlah_per_kolom'] = $jumlah;

  // normalisasi matrik
  //Menghitung nilai per cell (nilai cell / jumlah)
      $arrayVector = array();
      foreach ($data['daftar_kriteria'] as $kriteria) {
        $nilai = array();
        $a = 0;
        foreach ($data['relasi_kriteria'] as $relasi) {
            if($relasi->kriteria_dari == $kriteria->id_rasio){
              $nilai1 = $relasi->bobot / $jumlah[$a];
              array_push($nilai, $nilai1);
              $a++;
              }
        }
```

```
// jumlah dari nilai matrik yang dinormalisasi
//Menghitung nilai vector
      $vector = 0;
      for($i = 0; $i < count($nilai); $i++){
        $vector = $vector + $nilai[$i];
      }

      $vector = $vector / $data['jumlahKriteria'];
      $vector = number_format($vector, 3, '.', '');
      array_push($arrayVector, $vector);

      for ($i=0; $i  < 5 ; $i++) {
         if(isset($arrayVector[$i])){
            $this->M_produktivitas->update_bobotprioritas($arrayVector[$i], $i+1);
         }
      }

    }

//Menghitung nilai lamda
      $lamda = 0;
        for($i = 0; $i < count($arrayVector); $i++){
          $lamda = $lamda + ($arrayVector[$i] * $jumlah[$i]);
        }

//Menghitung Nilai CI
    $ci = ($lamda - $data['jumlahKriteria']) / ($data['jumlahKriteria'] - 1);
    $ci = number_format($ci, 3, '.', '');

//Menghitung Nilai CR
    $cr = $ci / $this->nilaiRI($data['jumlahKriteria']);
    $cr = number_format($cr, 3, '.', '');

  $data['nilai_lamda']    = $lamda;
  $data['nilai_ci']       = $ci;
  $data['nilai_cr']       = $cr;
  $data['nilai_vector']   = $arrayVector;
  $data['kriteria']=$this->M_produktivitas->get_kriteria();
  $this->load->view('manager/v_kriteria', $data);
  $this->load->view('manager/header_manager', $data);
  }


  //=========KETETAPAN NILAI RI============
  public function nilaiRI($jumlah){
     $nilai2;
```

```
        if($jumlah == '1'){
          $nilai2 = 0;
        } else if($jumlah == '2'){
            $nilai2 = 0;
        } else if($jumlah == '3'){
            $nilai2 = 0.58;
        } else if($jumlah == '4'){
            $nilai2 = 0.9;
        } else if($jumlah == '5'){
            $nilai2 = 1.12;
        } else if($jumlah == '6'){
            $nilai2 = 1.24;
        } else if($jumlah == '7'){
            $nilai2 = 1.32;
        } else if($jumlah == '8'){
            $nilai2 = 1.41;
        } else if($jumlah == '9'){
            $nilai2 = 1.45;
        } else if($jumlah == '10'){
            $nilai2 = 1.49;
        } else if($jumlah == '11'){
            $nilai2 = 1.51;
        } else if($jumlah == '12'){
            $nilai2 = 1.48;
        } else if($jumlah == '13'){
            $nilai2 = 1.56;
        } else if($jumlah == '14'){
            $nilai2 = 1.57;
        } else if($jumlah == '15'){
            $nilai2 = 1.59;
        }
        return $nilai2;
    }

    function edit_bobot_kriteria_manager(){
        $data['daftar_kriteria'] = $this->M_produktivitas->get_kriteria()->result();
        $data['relasi_kriteria'] = $this->M_produktivitas->get_relasi_kriteria()->result();
        $data['jumlah'] = count($data['daftar_kriteria']);

        $this->load->view('manager/v_editKriteria', $data);
        $this->load->view('manager/header_manager');
    }

    function update_bobot_kriteria(){
        for($i = 1; $i <= 5; $i++){ // i adalah id kriteria dari
            for($j = 1; $j <= 5; $j++){ // j adalah id kriteria tujuan
```

```
            $data['bobot'] = $this->input->post($i.$j);
            $this->M_produktivitas->update_bobot_relasiKriteria($i, $j, $data);
        }
    }
    redirect(site_url().'/C_produktivitas/bobot_kriteria_manager?suksesupdate');
}

//menampilkan skala performansi admin
public function skala_performansi_manager($bulan, $tahun){
    $data=array(
        'title'=>'Skala Performansi',
        'get_skalaPerformansi'=>$this->M_produktivitas->get_skalaPerformansi($bulan,
$tahun),
        );
    $this->load->view('manager/header_manager',$data);
    $this->load->view('manager/v_skalaPerformansi',$data);
}

public function tampil_hitung_manager(){ //menampilkan halaman hitung produktivitas
    $data['bulansekarang']=date('F');
    $data['tahunsekarang']=date('Y');
    $this->load->view('manager/v_hitungProduktivitas', $data);
    $this->load->view('manager/header_manager', $data);
}

public function tabel_matrik_omax(){ //menampilkan halaman hitung produktivitas
    $bulan = $this->input->post('bulan', true);
    $tahun = $this->input->post('tahun', true);
    $a1 = $this->input->post('a1', true);
    $a2 = $this->input->post('a2', true);
    $a3 = $this->input->post('a3', true);
    $a31 = $this->input->post('a31', true);
    $a4 = $this->input->post('a4', true);
    $a5 = $this->input->post('a5', true);
    $a6 = $this->input->post('a6', true);
    $a7 = $this->input->post('a7', true);
    $a8 = $this->input->post('a8', true);
    $a9 = $this->input->post('a9', true);

    //Menghitung nilai aktual hasil
    $r1 = round((($a1*100)/$a2),2); //round digunakan untuk pembulatan desimal

    if ($a3 == 0 && $a31 == 0) {
        $r2=100;
    } else {
        $r2 = round((($a3*100)/$a31),2);
```

```
}

if ($a4 == 0 && $a5 == 0) {
   $r3=100;
} else {
   $r3 = round((($a4*100)/$a5),2);
}

$r4 = round((($a6*100)/$a7),2);

if ($a8 == 0 && $a9 == 0) {
   $r5=100;
} else {
   $r5 = round((($a8*100)/$a9),2);
}

//mengisi matrik tiap skala
$this->load->model('M_produktivitas');
$target = $this->M_produktivitas->ambil_target();
$data_bobot = $this->M_produktivitas->ambil_bobot();

$jumlahrasio=count($target);

for($i = 0; $i < 6; $i++) {
   $maxRasio = $this->M_produktivitas->get_targetMax($i + 1);
   $minRasio = $this->M_produktivitas->get_targetMin($i + 1);
   $nilaiDasar = $this->M_produktivitas->get_nilaiDasar($i + 1);

   $target_max[$i] = $maxRasio[0]['max_rasio' . ($i + 1)];
   $target_min[$i] = $minRasio[0]['min_rasio' . ($i + 1)];
   $nilai_dasar[$i] = $nilaiDasar[0]['dasar' . ($i + 1)];
   $bobot[$i] = $data_bobot[$i]['bobot_rasio'];
}

//skala 1-2
for ($i = 0; $i < $jumlahrasio; $i++) {
  $selisih1[$i]= ($nilai_dasar[$i]-$target_min[$i])/3;
}

//skala 4-9
for ($i = 0; $i < $jumlahrasio; $i++) {
  $selisih2[$i]= ($target_max[$i]-$nilai_dasar[$i])/7;
}


// menghitung isi metrik rasio
```

```
// mengisi nilai matrix omax rasio 1-5indeks ke 0-11
// METRIK RASIO 1
$jumlahskala = 12;
for ($i = 1; $i <= $jumlahskala; $i++) {
   if ($i== 7) { //matrik skala 3 (nilai dasar)
      $rasio[0][0] = $nilai_dasar[0];
      $simpan = $rasio[0][0];
      $rasio[$i][0] = round($simpan,2);
   } else if ($i==0) { // matrik skala 10 (nilai terbaik)
      $rasio[0][0] = $target_max[0];
      $simpan = $rasio[0][0];
      $rasio[$i][0] = round($simpan,2);
   } else if ($i==10) { // matrik skala 1 (nilai terburuk)
      $rasio[0][0] = $target_min[0];
      $simpan = $rasio[0][0];
      $rasio[$i][0] = round($simpan,2);
   }
}
$rasio[0][0] = $nilai_dasar[0]; // matrik skala 1-2
$simpan = $rasio[0][0];
for ($i = 8; $i <= 9; $i++) {
   $simpan = $simpan - $selisih1[0];
   $rasio[$i][0] = round($simpan,2);
}
$rasio[0][0] = $target_max[0]; // matrik skala 4-9
$simpan = $rasio[0][0];
for ($i = 1; $i <= 6; $i++) {
   $simpan = $simpan - $selisih2[0];
   $rasio[$i][0] = round($simpan,2);
}

// MATRIK RASIO 2
for ($i = 1; $i <= $jumlahskala; $i++) {
   if ($i== 7) {
      $rasio[0][1] = $nilai_dasar[1];
      $simpan = $rasio[0][1];
      $rasio[$i][1] = round($simpan,2);
   } else if ($i==0) {
      $rasio[0][1] = $target_max[1];
      $simpan = $rasio[0][1];
      $rasio[$i][1] = round($simpan,2);
   } else if ($i==10) {
      $rasio[0][1] = $target_min[1];
      $simpan = $rasio[0][1];
      $rasio[$i][1] = round($simpan,2);
   }
```

```
}
$rasio[0][1] = $nilai_dasar[1];
$simpan = $rasio[0][1];
for ($i = 8; $i <= 9; $i++) {
   $simpan = $simpan - $selisih1[1];
   $rasio[$i][1] = round($simpan,2);
}
$rasio[0][1] = $target_max[1];
$simpan = $rasio[0][1];
for ($i = 1; $i <= 6; $i++) {
   $simpan = $simpan - $selisih2[1];
   $rasio[$i][1] = round($simpan,2);
}

// MATRIK RASIO 3
for ($i = 1; $i <= $jumlahskala; $i++) {
   if ($i== 7) {
      $rasio[0][2] = $nilai_dasar[2];
      $simpan = $rasio[0][2];
      $rasio[$i][2] = round($simpan,2);
   } else if ($i==0) {
      $rasio[0][2] = $target_max[2];
      $simpan = $rasio[0][2];
      $rasio[$i][2] = round($simpan,2);
   } else if ($i==10) {
      $rasio[0][2] = $target_min[2];
      $simpan = $rasio[0][2];
      $rasio[$i][2] = round($simpan,2);
   }
}
$rasio[0][2] = $nilai_dasar[2];
$simpan = $rasio[0][2];
for ($i = 8; $i <= 9; $i++) {
   $simpan = $simpan - $selisih1[2];
   $rasio[$i][2] = round($simpan,2);
}
$rasio[0][2] = $target_max[2];
$simpan = $rasio[0][2];
for ($i = 1; $i <= 6; $i++) {
   $simpan = $simpan - $selisih2[2];
   $rasio[$i][2] = round($simpan,2);
}

// MATRIK RASIO 4
for ($i = 1; $i <= $jumlahskala; $i++) {
   if ($i== 7) {
```

```
            $rasio[0][3] = $nilai_dasar[3];
            $simpan = $rasio[0][3];
            $rasio[$i][3] = round($simpan,2);
        } else if ($i==0) {
            $rasio[0][3] = $target_max[3];
            $simpan = $rasio[0][3];
            $rasio[$i][3] = round($simpan,2);
        } else if ($i==10) {
            $rasio[0][3] = $target_min[3];
            $simpan = $rasio[0][3];
            $rasio[$i][3] = round($simpan,2);
        }
    }
    $rasio[0][3] = $nilai_dasar[3];
    $simpan = $rasio[0][3];
    for ($i = 8; $i <= 9; $i++) {
        $simpan = $simpan - $selisih1[3];
        $rasio[$i][3] = round($simpan,2);
    }
    $rasio[0][3] = $target_max[3];
    $simpan = $rasio[0][3];
    for ($i = 1; $i <= 6; $i++) {
        $simpan = $simpan - $selisih2[3];
        $rasio[$i][3] = round($simpan,2);
    }

    // MATRIK RASIO 5
    for ($i = 1; $i <= $jumlahskala; $i++) {
        if ($i== 7) {
            $rasio[0][4] = $nilai_dasar[4];
            $simpan = $rasio[0][4];
            $rasio[$i][4] = round($simpan,2);
        } else if ($i==0) {
            $rasio[0][4] = $target_max[4];
            $simpan = $rasio[0][4];
            $rasio[$i][4] = round($simpan,2);
        } else if ($i==10) {
            $rasio[0][4] = $target_min[4];
            $simpan = $rasio[0][4];
            $rasio[$i][4] = round($simpan,2);
        } else {
            $rasio[0][4] = $target_max[4];
            $simpan = $rasio[0][4];
            $simpan = $simpan - $selisih2[4];
            $rasio[$i][4] = round($simpan,2);
        }
```

```
}
$rasio[0][4] = $nilai_dasar[4];
$simpan = $rasio[0][4];
for ($i = 8; $i <= 9; $i++) {
   $simpan = $simpan - $selisih1[4];
   $rasio[$i][4] = round($simpan,2);
}
$rasio[0][4] = $target_max[4];
$simpan = $rasio[0][4];
for ($i = 1; $i <= 6; $i++) {
   $simpan = $simpan - $selisih2[4];
   $rasio[$i][4] = round($simpan,2);
}

 //5. memilih nilai masuk skala yg mana
//cek r1
if($r1<=$target_min[0]){
$index1=0;
}
else if($r1>=$target_max[0]){
$index1=10;
}
else{
for ($i=0; $i < $jumlahskala; $i++) {
$sim2 = round($rasio[$i][0],2);
$sim1 = round($rasio[$i+1][0],2);

$a=round($sim1,2);
while($a<=$sim2){
if($r1==$a){
if(abs($r1-$sim2)>abs($r1-$sim1)){
   $in1 = $i+1;
}
if(abs($r1-$sim2)<abs($r1-$sim1)){
   $in1 = $i;
}
if(abs($r1-$sim2)==abs($r1-$sim1)){
   $in1 = $i+1;
}
   switch ($in1) {
      case 0 : $index1=10;
         break;
      case 1 : $index1=9;
         break;
      case 2 : $index1=8;
         break;
```

```
            case 3 : $index1=7;
                break;
            case 4 : $index1=6;
                break;
            case 5 : $index1=5;
                break;
            case 6 : $index1=4;
                break;
            case 7 : $index1=3;
                break;
            case 8 : $index1=2;
                break;
            case 9 : $index1=1;
                break;
            case 10 : $index1=0;
                break;
        }
    }
    $a = round(($a+0.01),2);
    }
    }
    }

    //cek r2
    if($r2<=$target_min[1]){
    $index2=0;
    }
    else if($r2>=$target_max[1]){
    $index2=10;
    }
    else{
    for ($i=0; $i < $jumlahskala; $i++) {
    $sim2 = round($rasio[$i][1],2);
    $sim1 = round($rasio[$i+1][1],2);

    $a=round($sim1,2);
    while($a<=$sim2){
    if($r2==$a){
    if(abs($r2-$sim2)>abs($r2-$sim1)){
    $in2 = $i+1;
    }
    if(abs($r2-$sim2)<abs($r2-$sim1)){
    $in2 = $i;
    }
    if(abs($r2-$sim2)==abs($r2-$sim1)){
    $in2 = $i+1;
```

```
}
   switch ($in2) {
      case 0 : $index2=10;
         break;
      case 1 : $index2=9;
         break;
      case 2 : $index2=8;
         break;
      case 3 : $index2=7;
         break;
      case 4 : $index2=6;
         break;
      case 5 : $index2=5;
         break;
      case 6 : $index2=4;
         break;
      case 7 : $index2=3;
         break;
      case 8 : $index2=2;
         break;
      case 9 : $index2=1;
         break;
      case 10 : $index2=0;
         break;
   }
}
$a = round(($a+0.01),2);
}
}
}

//cek r3
if($r3<=$target_min[2]){
$index3=0;
}
else if($r3>=$target_max[2]){
$index3=10;
}
else{
for ($i=0; $i < $jumlahskala; $i++) {
$sim2 = round($rasio[$i][2],2);
$sim1 = round($rasio[$i+1][2],2);

$a=round($sim1,2);
while($a<=$sim2){
if($r3==$a){
```

```
if(abs($r3-$sim2)>abs($r3-$sim1)){
$in3 = $i+1;
}
if(abs($r3-$sim2)<abs($r3-$sim1)){
$in3 = $i;
}
if(abs($r3-$sim2)==abs($r3-$sim1)){
$in3 = $i+1;
}
   switch ($in3) {
      case 0 : $index3=10;
         break;
      case 1 : $index3=9;
         break;
      case 2 : $index3=8;
         break;
      case 3 : $index3=7;
         break;
      case 4 : $index3=6;
         break;
      case 5 : $index3=5;
         break;
      case 6 : $index3=4;
         break;
      case 7 : $index3=3;
         break;
      case 8 : $index3=2;
         break;
      case 9 : $index3=1;
         break;
      case 10 : $index3=0;
         break;
   }
}
$a = round(($a+0.01),2);
}
}
}

//cek r4
if($r4>=$target_min[3]){
$index4=0;
}
else if($r4<=$target_max[3]){
$index4=10;
}
```

```
else{
for ($i=0; $i < $jumlahskala; $i++) {
$sim1 = round($rasio[$i][3],2);
$sim2 = round($rasio[$i+1][3],2);

$a=round($sim1,2);
while($a<=$sim2){
if($r4==$a){
if(abs($r4-$sim2)>abs($r4-$sim1)){
$in4 = $i+1;
}
if(abs($r4-$sim2)<abs($r4-$sim1)){
$in4 = $i;
}
if(abs($r4-$sim2)==abs($r4-$sim1)){
$in4 = $i+1;
}
   switch ($in4) {
      case 0 : $index4=10;
         break;
      case 1 : $index4=9;
         break;
      case 2 : $index4=8;
         break;
      case 3 : $index4=7;
         break;
      case 4 : $index4=6;
         break;
      case 5 : $index4=5;
         break;
      case 6 : $index4=4;
         break;
      case 7 : $index4=3;
         break;
      case 8 : $index4=2;
         break;
      case 9 : $index4=1;
         break;
      case 10 : $index4=0;
         break;
   }
}
$a = round(($a+0.01),2);
}
}
}
```

```
//cek r5
if($r5<=$target_min[4]){
$index5=0;
}
else if($r5>=$target_max[4]){
$index5=10;
}
else{
for ($i=0; $i < $jumlahskala; $i++) {
$sim2 = round($rasio[$i][4],2);
$sim1 = round($rasio[$i+1][4],2);

$a=round($sim1,2);
while($a<=$sim2){
if($r5==$a){
if(abs($r5-$sim2)>abs($r5-$sim1)){
$in5 = $i+1;
}
if(abs($r5-$sim2)<abs($r5-$sim1)){
$in5 = $i;
}
if(abs($r5-$sim2)==abs($r5-$sim1)){
$in5 = $i+1;
}
   switch ($in5) {
      case 0 : $index5=10;
         break;
      case 1 : $index5=9;
         break;
      case 2 : $index5=8;
         break;
      case 3 : $index5=7;
         break;
      case 4 : $index5=6;
         break;
      case 5 : $index5=5;
         break;
      case 6 : $index5=4;
         break;
      case 7 : $index5=3;
         break;
      case 8 : $index5=2;
         break;
      case 9 : $index5=1;
         break;
```

```
            case 10 : $index5=0;
                break;
          }
        }
        $a = round(($a+0.01),2);
        }
        }
        }

        $arr = ["BURUK", "BURUK", "BURUK", "CUKUP BAIK", "CUKUP BAIK",
"CUKUP BAIK", "BAIK", "BAIK", "BAIK", "SANGAT BAIK", "SANGAT BAIK"];

        $bobot[$i] = $data_bobot[$i]['bobot_rasio'];
        $indikatorPerformansi = (($bobot[0]*100)*$index1) + (($bobot[1]*100)*$index2) +
(($bobot[2]*100)*$index3) + (($bobot[3]*100)*$index4) + (($bobot[4]*100)*$index5) +
(($bobot[5]*100)*$index6);

        $ip_sebelum = $this->M_produktivitas->indikatorSebelum();
        $indeksProduktivitas=round((($indikatorPerformansi-
$ip_sebelum)*100)/$ip_sebelum, 2) ."%";

        $data['bulan'] = $bulan;
        $data['tahun'] = $tahun;
        $data = array(
          'bulan'=> $bulan,
          'tahun' => $tahun,
          'nilaiAktualR1' => $r1,
          'nilaiAktualR2' => $r2,
          'nilaiAktualR3' => $r3,
          'nilaiAktualR4' => $r4,
          'nilaiAktualR5' => $r5,
          'kriteria_performansi1' => $index1,
          'kriteria_performansi2' => $index2,
          'kriteria_performansi3' => $index3,
          'kriteria_performansi4' => $index4,
          'kriteria_performansi5' => $index5,
          'kat_rasio1' => $arr[$index1],
          'kat_rasio2' => $arr[$index2],
          'kat_rasio3' => $arr[$index3],
          'kat_rasio4' => $arr[$index4],
          'kat_rasio5' => $arr[$index5]
          );

        $dataIP = array(
          'bulan'=> $bulan,
          'tahun' => $tahun,
```

```
        'indikator_performansi' => $indikatorPerformansi,
        'indeks_produktivitas' => $indeksProduktivitas
        );

    $this->M_produktivitas->insertAktualRasio1($data);
    $this->M_produktivitas->insertAktualRasio2($data);
    $this->M_produktivitas->insertAktualRasio3($data);
    $this->M_produktivitas->insertAktualRasio4($data);
    $this->M_produktivitas->insertAktualRasio5($data);
    $this->M_produktivitas->insertIP($dataIP);
    redirect('C_produktivitas/ambil_IP_manager');
}


function insert_omax(){
    $bulan = $this->input->post('bulan');
    $tahun = $this->input->post('tahun');
    $nilaiAktualR1 = $this->input->post('r1');
    $nilaiAktualR2 = $this->input->post('r2');
    $nilaiAktualR3 = $this->input->post('r3');
    $nilaiAktualR4 = $this->input->post('r4');
    $nilaiAktualR5 = $this->input->post('r5');
    $nilaiPerformansiR1 = $this->input->post('rasio1');
    $nilaiPerformansiR2 = $this->input->post('rasio2');
    $nilaiPerformansiR3 = $this->input->post('rasio3');
    $nilaiPerformansiR4 = $this->input->post('rasio4');
    $nilaiPerformansiR5 = $this->input->post('rasio5');
    $kritRasio1 = $this->input->post('Katrasio1');
    $kritRasio2 = $this->input->post('Katrasio2');
    $kritRasio3 = $this->input->post('Katrasio3');
    $kritRasio4 = $this->input->post('Katrasio4');
    $kritRasio5 = $this->input->post('Katrasio5');
    $indikator_performansi = $this->input->post('nilaiIndikator');
    $indeksProduktivitas = $this->input->post('nilaiProduktivitas');

    $data = array(
        'bulan'=> $bulan,
        'tahun' => $tahun,
        'nilaiAktualR1' => $nilaiAktualR1,
        'nilaiAktualR2' => $nilaiAktualR2,
        'nilaiAktualR3' => $nilaiAktualR3,
        'nilaiAktualR4' => $nilaiAktualR4,
        'nilaiAktualR5' => $nilaiAktualR5,
        'kriteria_performansi1' => $nilaiPerformansiR1,
        'kriteria_performansi2' => $nilaiPerformansiR2,
        'kriteria_performansi3' => $nilaiPerformansiR3,
```

```
            'kriteria_performansi4' => $nilaiPerformansiR4,
            'kriteria_performansi5' => $nilaiPerformansiR5,
            'kat_rasio1' => $kritRasio1,
            'kat_rasio2' => $kritRasio2,
            'kat_rasio3' => $kritRasio3,
            'kat_rasio4' => $kritRasio4,
            'kat_rasio5' => $kritRasio5
            );
          $dataIP = array(
            'bulan'=> $bulan,
            'tahun' => $tahun,
            'indikator_performansi' => $indikator_performansi,
            'indeks_produktivitas' => $indeksProduktivitas
            );

          $this->load->model('M_produktivitas');
          $this->M_produktivitas->insertAktualRasio1($data);
          $this->M_produktivitas->insertAktualRasio2($data);
          $this->M_produktivitas->insertAktualRasio3($data);
          $this->M_produktivitas->insertAktualRasio4($data);
          $this->M_produktivitas->insertAktualRasio5($data);
          $this->M_produktivitas->insertIP($dataIP);
          redirect('C_produktivitas/ambil_IP_manager');
        }

    public function ambil_IP_manager(){ //menampilkan daftar jenis kamar untuk
dropdown
        $data=array(
            'title'=>'Data Produktivitas',
            'get_daftarProduktivitas'=>$this->M_produktivitas->get_daftarProduktivitas(),
        );
        $data['cobalagi']=$this->M_produktivitas->get_ip();
        // $data['data_omax']=$this->M_produktivitas->get_produktivitas();
        // $data['data_departemen'] = $this->M_produktivitas->get_namarasio();
        $this->load->view('manager/header_manager',$data);
        $this->load->view('manager/v_produktivitas',$data);
    }

}

?>
```

D.8. Kode program *class model*/m_master_username

```
<?php

if (!defined('BASEPATH'))
    exit('No direct script access allowed');

Class M_master_username extends CI_Model {

    function __construct() {
        parent::__construct();
    }

    function get_login($username, $password){
        $result = $this->db->query("SELECT username, level FROM tb_user WHERE
username ='$username' AND password='$password' AND statusUser='AKTIF'");
        $rows = $result->num_rows();
        if($rows == 1){
            return true;
        }else{
            return false;
        }
    }

    function get_data($username, $password) {
        return $this->db->query("SELECT username, level FROM tb_user WHERE
username ='$username' AND password='$password' AND statusUser='AKTIF'");
    }

    function get_session_username() {
        $data['session_username'] = $this->session->userdata('session_username');
        return $data;
    }
    function get_session_level() {
        $data['session_level'] = $this->session->userdata('session_level');
        return $data;
    }

    function store_session($data) {
        $this->session->set_userdata('session_username', $data['username']);
        $this->session->set_userdata('session_level', $data['level']);
    }

    function destroy_session() {
        $this->session->unset_userdata('session_username');
        $this->session->unset_userdata('session_level');
```

```
    }

}

?>
```

D.9. Kode program *class model*/m_kamar

```
<?php

if (!defined('BASEPATH'))
    exit('No direct script access allowed');

Class M_kamar extends CI_Model {

    function __construct() {
        parent::__construct();
    }

    function jmlKamar() {
        return $this->db->query("SELECT * FROM tb_daftarkamar WHERE
status='TERSEDIA' and statushapus='1'")->num_rows();
    }

    function get_jeniskamar() {
        return $this->db->query("SELECT * FROM tb_jeniskamar where status='ada'")-
>result_array();
    }

    function insertJenisKamar($dataJenisKamar){
        $this->db->insert('tb_jeniskamar', $dataJenisKamar);
    }

    function editJenisKamar($dataJenisKamar, $idJenisKamar){
        $this->db->where('id_jeniskamar', $idJenisKamar);
        $this->db->update('tb_jeniskamar', $dataJenisKamar);
    }

    function deleteJenisKamar($idJenisKamar, $dataJenisKamar){
        $this->db->where('id_jeniskamar', $idJenisKamar);
        $this->db->update('tb_jeniskamar', $dataJenisKamar);
    }

    function get_daftarkamar() {
```

```
        return $this->db->query("SELECT dk.id_daftarkamar, jk.jenisKamar,
dk.nomorkamar, dk.status FROM tb_daftarkamar dk join tb_jeniskamar jk on
(dk.id_jeniskamar=jk.id_jeniskamar) where dk.statushapus='1'")->result_array();
    }

    function get_daftarkamarTersedia() {
        return $this->db->query("SELECT dk.id_daftarkamar, jk.jenisKamar,
dk.nomorkamar, dk.status FROM tb_daftarkamar dk join tb_jeniskamar jk on
(dk.id_jeniskamar=jk.id_jeniskamar) where dk.statushapus='1' and
dk.status='TERSEDIA'")->result_array();
    }

    function insert_daftarkamar($daftarkamar){
        $this->db->insert('tb_daftarkamar', $daftarkamar);
    }

    function editDaftarKamar($daftarKamar, $idDaftarKamar){
        $this->db->where('id_daftarkamar', $idDaftarKamar);
        $this->db->update('tb_daftarkamar', $daftarKamar);
    }

    function get_deleteDaftarKamar($idDaftarKamar, $daftarKamar){
        $this->db->where('id_daftarkamar', $idDaftarKamar);
        $this->db->update('tb_daftarkamar', $daftarKamar);
    }
}
?>
```

D.10. Kode program *class model*/m_komplain

```
<?php

if (!defined('BASEPATH'))
    exit('No direct script access allowed');

Class M_komplain extends CI_Model {

    function __construct() {
        parent::__construct();
    }

    function jmlKomplain() {
        return $this->db->query("SELECT * FROM tb_komplain WHERE
status='BELUM'")->num_rows();
    }
```

```
    function insertKomplain($dataKomplain){
        $this->db->insert('tb_komplain', $dataKomplain);
    }

    function get_komplain() {
        return $this->db->query("SELECT * FROM tb_komplain k join tb_departemen d
on (d.id_departemen = k.id_departemen) order by id_komplain desc")->result_array();
    }

    function get_komplain_fo() {
        return $this->db->query("SELECT * FROM tb_komplain k join tb_departemen d
on (d.id_departemen = k.id_departemen) where d.namaDepartemen='Front Office' and
k.status!='TERATASI' order by id_komplain desc")->result_array();
    }

    function get_finishkomplain_fo() {
        return $this->db->query("SELECT * FROM tb_komplain k join tb_departemen d
on (d.id_departemen = k.id_departemen) where d.namaDepartemen='Front Office' and
k.status='TERATASI' order by id_komplain desc")->result_array();
    }

    function get_komplain_eng() {
        return $this->db->query("SELECT * FROM tb_komplain k join tb_departemen d
on (d.id_departemen = k.id_departemen) where d.namaDepartemen='Engineering' and
k.status!='TERATASI' order by id_komplain desc")->result_array();
    }

    function get_finishkomplain_eng() {
        return $this->db->query("SELECT * FROM tb_komplain k join tb_departemen d
on (d.id_departemen = k.id_departemen) where d.namaDepartemen='Engineering' and
k.status='TERATASI' order by id_komplain desc")->result_array();
    }

    function get_komplain_eng_baru() {
        return $this->db->query("SELECT * FROM tb_komplain k join tb_departemen d
on (d.id_departemen = k.id_departemen) where d.namaDepartemen='Engineering' and
k.status='BELUM TERATASI' order by id_komplain desc")->num_rows();
    }

    function get_komplain_hk() {
        return $this->db->query("SELECT * FROM tb_komplain k join tb_departemen d
on (d.id_departemen = k.id_departemen) where d.namaDepartemen='Housekeeping'
and k.status!='TERATASI' order by id_komplain desc")->result_array();
    }
```

```
    function get_komplain_hk_baru() {
        return $this->db->query("SELECT * FROM tb_komplain k join tb_departemen d
on (d.id_departemen = k.id_departemen) where d.namaDepartemen='Housekeeping'
and k.status='BELUM TERATASI' order by id_komplain desc")->num_rows();
    }

    function get_finishkomplain_hk() {
        return $this->db->query("SELECT * FROM tb_komplain k join tb_departemen d
on (d.id_departemen = k.id_departemen) where d.namaDepartemen='Housekeeping'
and k.status='TERATASI' order by id_komplain desc")->result_array();
    }

    function get_komplain_fnb() {
        return $this->db->query("SELECT * FROM tb_komplain k join tb_departemen d
on (d.id_departemen = k.id_departemen) where d.namaDepartemen='Food &
Beverage' and k.status!='TERATASI' order by id_komplain desc")->result_array();
    }

    function get_finishkomplain_fnb() {
        return $this->db->query("SELECT * FROM tb_komplain k join tb_departemen d
on (d.id_departemen = k.id_departemen) where d.namaDepartemen='Food &
Beverage' and k.status='TERATASI' order by id_komplain desc")->result_array();
    }

    function get_komplain_fnb_baru() {
        return $this->db->query("SELECT * FROM tb_komplain k join tb_departemen d
on (d.id_departemen = k.id_departemen) where d.namaDepartemen='Food &
Beverage' and k.status='BELUM TERATASI' order by id_komplain desc")-
>num_rows();
    }

    function get_komplainBelum() {
        return $this->db->query("SELECT * FROM tb_komplain k join tb_departemen d
on (d.id_departemen = k.id_departemen) where k.status='BELUM TERATASI' order
by id_komplain desc")->result_array();
    }

    function editKomplain($dataKomplain, $idKomplain){
        $this->db->where('id_komplain', $idKomplain);
        $this->db->update('tb_komplain', $dataKomplain);
    }

    function get_tamu($no){
        $query=$this->db->query("SELECT r.nama, d.nomorkamar FROM tb_reservasi r
join tb_daftarkamar d on (r.id_daftarkamar=d.id_daftarkamar) WHERE r.status='IN'
AND r.noid='$no'");
```

```
        return $query;
    }

    function get_penanganan($id_komplain) {
        return $this->db->query("SELECT * FROM tb_komplain k join tb_departemen d
on (d.id_departemen = k.id_departemen) where k.id_komplain='$id_komplain'")-
>result_array();
    }

}

?>
```

D.11. Kode program *class model*/m_pegawai

```
<?php

if (!defined('BASEPATH'))
    exit('No direct script access allowed');

Class M_pegawai extends CI_Model {

    function __construct() {
        parent::__construct();
    }

    function data_pegawai() {
        return $this->db->query("SELECT * FROM tb_pegawai p join tb_departemen d
on (p.id_departemen = d.id_departemen) where p.status='1' order by id_pegawai
desc")->result_array();
    }

    function data_user() {
        return $this->db->query("SELECT * FROM tb_user u join tb_pegawai p on
(u.id_pegawai = p.id_pegawai) where u.statusHapus='1'")->result_array();
    }

    function get_detail_pegawai($id) {
        $query = $this->db->query("SELECT * FROM tb_pegawai p join tb_departemen
d on (p.id_departemen = d.id_departemen) where p.id_pegawai = '$id'");
        return $query->result_array();
    }

    function data_departemen() {
        return $this->db->query("SELECT * FROM tb_departemen")->result_array();
```

```
    }

    function insert_pegawai($dataPegawai){
        $this->db->insert('tb_pegawai', $dataPegawai);
    }

    function get_editpegawai($id) {
        $query = $this->db->query("SELECT * FROM tb_pegawai where id_pegawai =
'$id'");
        foreach ($query->result_array() as $row)
        return $row;
    }

    function editPegawai($dataPegawai, $idPegawai){
        $this->db->where('id_pegawai', $idPegawai);
        $this->db->update('tb_pegawai', $dataPegawai);
    }

    function deletePegawai($idPegawai, $dataPegawai){
        $this->db->where('id_pegawai', $idPegawai);
        $this->db->update('tb_pegawai', $dataPegawai);
    }

    function cariPegawai($no){
        $query=$this->db->query("SELECT p.nama, d.namaDepartemen FROM
tb_pegawai p join tb_departemen d on (p.id_departemen=d.id_departemen) where
p.id_pegawai='$no'");
        return $query;
    }

    function simpan_absensi($data){
        $this->db->insert("tb_absensi",$data);
    }

    function data_absensi() {
        return $this->db->query("SELECT * FROM tb_absensi a join tb_pegawai p on
(a.id_pegawai = p.id_pegawai) order by id_absensi desc")->result_array();
    }

    function insert_akun($data){
        $this->db->insert("tb_user",$data);
    }

    function updateStatusUser($idUser, $dataUser){
        $this->db->where('id_user', $idUser);
        $this->db->update('tb_user', $dataUser);
```

```
    }

    function delete_user($idPegawai, $dataUser){
        $this->db->where('id_pegawai', $idPegawai);
        $this->db->update('tb_user', $dataUser);
    }

    function cek_uname($uname) {
        return $this->db->query("SELECT count(id_user) FROM tb_user where
username='$uname'")->result_array();
    }

    function cek_idpegawai($idPegawai) {
        return $this->db->query("SELECT count(id_user) FROM tb_user where
id_pegawai='$idPegawai'")->result_array();
    }

}

?>
```

D.12. Kode program *class model*/m_reservasi

```
<?php

if (!defined('BASEPATH'))
    exit('No direct script access allowed');

Class M_reservasi extends CI_Model {

    var $tabel = 'tb_daftarkamar';

    function __construct() {
        parent::__construct();
    }

    function jmlReservasi() {
        return $this->db->query("SELECT * FROM tb_reservasi WHERE status='IN'")-
>num_rows();
    }

    function get_nokamar() {
        return $this->db->query("SELECT * FROM tb_daftarkamar where
status='tersedia' and statushapus='1'")->result_array();
    }
```

```
    function cariHarga($idJenisKamar){
        $query=$this->db->query("SELECT harga from tb_jeniskamar where
status='ada' and id_jeniskamar='$idJenisKamar'");
        return $query;
    }

    function get_nomorKamar($id_jeniskamar) {
        $this->db->where('id_jeniskamar', $id_jeniskamar);
        $result = $this->db->get('nomorkamar');
        if ($result->num_rows() > 0) {
            return $result->result_array();
        }
        else {
            return array();
        }
    }

    function get_kamar($id) {
        $query = $this->db->query("SELECT * from tb_jeniskamar jk join
tb_daftarkamar dk on jk.id_jeniskamar=dk.id_jeniskamar where
dk.id_jeniskamar='$id' and dk.status='tersedia' and dk.statushapus='1'");
        return $query;
    }

    function update_kamar($id){
        $query= $this->db->query("UPDATE tb_daftarkamar SET status='TIDAK
TERSEDIA' where id_daftarkamar='$id'");
        return $query;
    }


    function simpan_checkin($data){
        $this->db->insert("tb_reservasi",$data);
    }

    function get_daftarcheckin() {
        return $this->db->query("SELECT * from tb_reservasi r join tb_daftarkamar dk
on (r.id_daftarkamar = dk.id_daftarkamar)  where r.status='IN' order by r.cekout asc")-
>result_array();
    }

    function get_daftarcheckout() {
        return $this->db->query("SELECT * from tb_reservasi r join tb_daftarkamar dk
on (r.id_daftarkamar = dk.id_daftarkamar)  where r.status='OUT' order by r.cekout
desc")->result_array();
```

```
    }

    function get_detailcheckin($id) {
        return $this->db->query("SELECT * from tb_reservasi r join tb_daftarkamar dk
on (r.id_daftarkamar = dk.id_daftarkamar)  where id_reservasi='$id'")->result_array();
    }

    function checkin() {
        return $this->db->query("SELECT * from tb_reservasi r join tb_daftarkamar dk
on (r.id_daftarkamar = dk.id_daftarkamar)")->result_array();
    }

    function get_editCheckin($id_reservasi) {
        return $this->db->query("SELECT * from tb_jeniskamar jk join tb_daftarkamar
dk on (jk.id_jeniskamar = dk.id_jeniskamar) join tb_reservasi r on (r.id_daftarkamar =
dk.id_daftarkamar) where r.id_reservasi='$id_reservasi'")->result_array();
    }

    function get_updateCheckin($dataReservasi, $idReservasi){
        $this->db->where('id_reservasi', $idReservasi);
        $this->db->update('tb_reservasi', $dataReservasi);
    }

    function get_ubahCheckout($id_reservasi){
        $query=$this->db->query("UPDATE tb_daftarkamar dk join tb_reservasi r on
(r.id_daftarkamar = dk.id_daftarkamar) set dk.status='TERSEDIA', r.status='OUT'
where r.id_reservasi='$id_reservasi'");
        return $query;
    }
}
?>
```

D.13. Kode program *class model*/m_produktivitas

```
<?php

if (!defined('BASEPATH'))
    exit('No direct script access allowed');

Class M_produktivitas extends CI_Model {

    function __construct() {
        parent::__construct();
```

```
    }

    function get_daftarProduktivitas() {
        return $this->db->query("SELECT * FROM tb_produktivitas order by
id_produktivitas desc")->result_array();
    }

    function get_skalaPerformansi($bulan, $tahun) {
        return $this->db->query("SELECT * FROM tb_nilairasio tr join tb_rasio r on
tr.id_rasio=r.id_rasio where bulan='$bulan' and tahun='$tahun' order by id_nilaiRasio
desc")->result_array();
    }

    function get_jumlahKaryawan() {
        $query=$this->db->query("SELECT count(id_pegawai) as jml from tb_pegawai
where status = '1'");
        return $query->row()->jml;
    }

    function get_absen($bulan, $tahun) {
        $query=$this->db->query("SELECT count(id_absensi) as jml from tb_absensi where
keterangan != 'Hadir' and MONTH(tanggal)='$bulan' and YEAR(tanggal)='$tahun'");
        return $query->row()->jml;
    }

    function get_jumReservasi($bulan, $tahun) {
            $query=$this->db->query("SELECT count(id_reservasi) as jml from tb_reservasi
where MONTH(tanggal)='$bulan' and YEAR(tanggal)='$tahun'");
            return $query->row()->jml;
    }

    function get_jumKamar() {
            $query=$this->db->query("SELECT count(id_daftarkamar) as jml from
tb_daftarkamar where statushapus = '1'");
            return $query->row()->jml;
    }

    function get_komHKdone($bulan, $tahun) {
            $query=$this->db->query("SELECT count(id_komplain) as jml from tb_komplain
k join tb_departemen d on (d.id_departemen=k.id_departemen) where
MONTH(k.tanggal)='$bulan' and YEAR(k.tanggal)='$tahun' and d.namaDepartemen =
'housekeeping' and k.status='TERATASI'");
            return $query->row()->jml;
    }

    function get_komHK($bulan, $tahun) {
```

```
        $query=$this->db->query("SELECT count(id_komplain) as jml from tb_komplain
k join tb_departemen d on (d.id_departemen=k.id_departemen) where
MONTH(k.tanggal)='$bulan' and YEAR(k.tanggal)='$tahun' and d.namaDepartemen =
'housekeeping'");
        return $query->row()->jml;
    }

    function get_komFNBdone($bulan, $tahun) {
        $query=$this->db->query("SELECT count(id_komplain) as jml from tb_komplain
k join tb_departemen d on (d.id_departemen=k.id_departemen) where MONTH
(k.tanggal)='$bulan' and YEAR(k.tanggal)='$tahun' and d.namaDepartemen = 'Food &
Beverage' and k.status='TERATASI'");
        return $query->row()->jml;
    }

    function get_komFNB($bulan, $tahun) {
        $query=$this->db->query("SELECT count(id_komplain) as jml from tb_komplain
k join tb_departemen d on (d.id_departemen=k.id_departemen) where
MONTH(k.tanggal)='$bulan' and YEAR(k.tanggal)='$tahun' and d.namaDepartemen =
'Food & Beverage'");
        return $query->row()->jml;
    }

    function get_komEngdone($bulan, $tahun) {
        $query=$this->db->query("SELECT count(id_komplain) as jml from tb_komplain
k join tb_departemen d on (d.id_departemen=k.id_departemen) where
MONTH(k.tanggal)='$bulan' and YEAR(k.tanggal)='$tahun' and d.namaDepartemen =
'Engineering' and k.status='TERATASI'");
        return $query->row()->jml;
    }

    function get_komEng($bulan, $tahun) {
        $query=$this->db->query("SELECT count(id_komplain) as jml from tb_komplain
k join tb_departemen d on (d.id_departemen=k.id_departemen) where
MONTH(k.tanggal)='$bulan' and YEAR(k.tanggal)='$tahun' and d.namaDepartemen =
'Engineering'");
        return $query->row()->jml;
    }

    function get_komFOdone($bulan, $tahun) {
        $query=$this->db->query("SELECT count(id_komplain) as jml from tb_komplain
k join tb_departemen d on (d.id_departemen=k.id_departemen) where
MONTH(k.tanggal)='$bulan' and YEAR(k.tanggal)='$tahun' and d.namaDepartemen =
'front office' and k.status='TERATASI'");
        return $query->row()->jml;
    }
```

```
    function get_komFO($bulan, $tahun) {
        $query=$this->db->query("SELECT count(id_komplain) as jml from tb_komplain
k join tb_departemen d on (d.id_departemen=k.id_departemen) where
MONTH(k.tanggal)='$bulan' and YEAR(k.tanggal)='$tahun' and d.namaDepartemen =
'front office'");
        return $query->row()->jml;
    }

    function cek_bulan($bulan) {
        $query=$this->db->query("SELECT bulan as bln from tb_produktivitas where
bulan='$bulan'");
        return $query->row()->bln;
    }

    function cek_tahun($tahun) {
        $query=$this->db->query("SELECT tahun as thn from tb_produktivitas where
tahun='$tahun'");
        return $query->row()->thn;
    }

    function ambil_target(){
        $query=$this->db->query("SELECT
id_rasio,nama_rasio,target_max,target_min,bobot_rasio from tb_rasio order by id_rasio");
        return $query->result();
    }

    function ambil_bobot(){
        $query=$this->db->query("SELECT * from tb_rasio order by id_rasio");
        return $query->result_array();
    }

    function get_targetMax($no){
        $query = NULL;
        if ($no == 1) {
            $query=$this->db->query("SELECT target_max AS max_rasio1 from tb_rasio
where id_rasio='1'");
        } else if ($no == 2) {
            $query=$this->db->query("SELECT target_max AS max_rasio2 from tb_rasio
where id_rasio='2'");
        } else if ($no == 3) {
            $query=$this->db->query("SELECT target_max AS max_rasio3 from tb_rasio
where id_rasio='3'");
        } else if ($no == 4) {
            $query=$this->db->query("SELECT target_max AS max_rasio4 from tb_rasio
where id_rasio='4'");
```

```
        } else if ($no == 5) {
            $query=$this->db->query("SELECT target_max AS max_rasio5 from tb_rasio
where id_rasio='5'");
        } else if ($no == 6) {
            $query=$this->db->query("SELECT target_max AS max_rasio6 from tb_rasio
where id_rasio='6'");
        }
        return $query->result_array();
    }

    function get_targetMin($no){
        $query = NULL;
        if ($no == 1) {
            $query=$this->db->query("SELECT target_min AS min_rasio1 from tb_rasio
where id_rasio='1'");
        } else if ($no == 2) {
            $query=$this->db->query("SELECT target_min AS min_rasio2 from tb_rasio
where id_rasio='2'");
        } else if ($no == 3) {
            $query=$this->db->query("SELECT target_min AS min_rasio3 from tb_rasio
where id_rasio='3'");
        } else if ($no == 4) {
            $query=$this->db->query("SELECT target_min AS min_rasio4 from tb_rasio
where id_rasio='4'");
        } else if ($no == 5) {
            $query=$this->db->query("SELECT target_min AS min_rasio5 from tb_rasio
where id_rasio='5'");
        } else if ($no == 6) {
            $query=$this->db->query("SELECT target_min AS min_rasio6 from tb_rasio
where id_rasio='6'");
        }
        return $query->result_array();
    }


    function get_nilaiDasar($no){
        $query = NULL;
        if ($no == 1) {
            $query=$this->db->query("SELECT nilai_dasar AS dasar1 from tb_rasio where
id_rasio='1'");
        } else if ($no == 2) {
            $query=$this->db->query("SELECT nilai_dasar AS dasar2 from tb_rasio where
id_rasio='2'");
        } else if ($no == 3) {
            $query=$this->db->query("SELECT nilai_dasar AS dasar3 from tb_rasio where
id_rasio='3'");
```

```
        } else if ($no == 4) {
            $query=$this->db->query("SELECT nilai_dasar AS dasar4 from tb_rasio where
id_rasio='4'");
        } else if ($no == 5) {
            $query=$this->db->query("SELECT nilai_dasar AS dasar5 from tb_rasio where
id_rasio='5'");
        } else if ($no == 6) {
            $query=$this->db->query("SELECT nilai_dasar AS dasar6 from tb_rasio where
id_rasio='6'");
        }
        return $query->result_array();
    }


    function insertAktualRasio1($data){
        $this->db->query("insert tb_nilairasio
                set bulan             ='$data[bulan]',
                tahun                 ='$data[tahun]',
                id_rasio              ='1',
                nilai_aktual          ='$data[nilaiAktualR1]',
                kriteria_performansi  ='$data[kriteria_performansi1]',
                kategori_skala        ='$data[kat_rasio1]'
                ");
    }

    function insertAktualRasio2($data){
        $this->db->query("insert tb_nilairasio
                set bulan             ='$data[bulan]',
                tahun                 ='$data[tahun]',
                id_rasio              ='2',
                nilai_aktual          ='$data[nilaiAktualR2]',
                kriteria_performansi  ='$data[kriteria_performansi2]',
                kategori_skala        ='$data[kat_rasio2]'
                ");
    }

    function insertAktualRasio3($data){
        $this->db->query("insert tb_nilairasio
                set bulan             ='$data[bulan]',
                tahun                 ='$data[tahun]',
                id_rasio              ='3',
                nilai_aktual          ='$data[nilaiAktualR3]',
                kriteria_performansi  ='$data[kriteria_performansi3]',
                kategori_skala        ='$data[kat_rasio3]'
                ");
    }
```

```
function insertAktualRasio4($data){
   $this->db->query("insert tb_nilairasio
         set bulan              ='$data[bulan]',
         tahun                 ='$data[tahun]',
         id_rasio              ='4',
         nilai_aktual           ='$data[nilaiAktualR4]',
         kriteria_performansi    ='$data[kriteria_performansi4]',
         kategori_skala          ='$data[kat_rasio4]'
         ");
}

function insertAktualRasio5($data){
   $this->db->query("insert tb_nilairasio
         set bulan              ='$data[bulan]',
         tahun                 ='$data[tahun]',
         id_rasio              ='5',
         nilai_aktual           ='$data[nilaiAktualR5]',
         kriteria_performansi    ='$data[kriteria_performansi5]',
         kategori_skala          ='$data[kat_rasio5]'
         ");
}

function insertAktualRasio6($data){
   $this->db->query("insert tb_nilairasio
         set bulan              ='$data[bulan]',
         tahun                 ='$data[tahun]',
         id_rasio              ='6',
         nilai_aktual           ='$data[nilaiAktualR6]',
         kriteria_performansi    ='$data[kriteria_performansi6]',
         kategori_skala          ='$data[kat_rasio6]'
         ");
}


function insertIP($dataIP){
   $this->db->query("insert tb_produktivitas
         set bulan              ='$dataIP[bulan]',
         tahun                 ='$dataIP[tahun]',
         indikator_performansi   ='$dataIP[indikator_performansi]',
         indeks_produktivitas    ='$dataIP[indeks_produktivitas]'
         ");
}

function indikatorSebelum(){
```

```
        $aa = $this->db->query('SELECT * FROM tb_produktivitas ORDER BY
id_produktivitas desc');
        $bb = $aa->result_array();
        print_r($bb);
        return ((count($bb) > 0) ? $bb[0]['indikator_performansi'] : 0);
    }

    function get_produktivitas(){
        $query=$this->db->query("SELECT * FROM tb_nilairasio nr join tb_rasio r on
nr.id_rasio=r.id_rasio");
        return $query->result();
    }


    function get_namarasio(){
        return $this->db->query("SELECT nama_rasio FROM tb_nilairasio nr join tb_rasio r
on nr.id_rasio=r.id_rasio group by nama_rasio")->result();
    }

    function get_ip(){
        return $this->db->query("SELECT indeks_produktivitas FROM tb_produktivitas")-
>result();
    }

    function get_kriteria() {
        $this->db->select('*');
        $this->db->from('tb_rasio');
        return $this->db->get();
    }


    //===============PEMBOBOTAN KRITERIA===========
    function get_relasi_kriteria(){
        $this->db->select('*');
        $this->db->from('tb_relasi_kriteria');
        return $this->db->get();
    }

    function update_bobotprioritas($vector, $i){
        $query = $this->db->query("UPDATE `tb_rasio`
                            SET `bobot_rasio` ='$vector'
                            where id_rasio = '$i'
                            ");
    return $query;
    }
```

```
    function update_bobot_relasiKriteria($kriteria_dari, $kriteria_tujuan, $data){
        $this->db->where('kriteria_dari', $kriteria_dari);
        $this->db->where('kriteria_tujuan', $kriteria_tujuan);
        $this->db->update('tb_relasi_kriteria', $data);
    }

}

?>
```

LAMPIRAN E. PENGUJIAN *BLACK BOX*

No.	Fitur	Aksi	Hasil	Kesimpulan
1.	*Login*	Mengisi kolom *username* dan *password* lalu klik tombol *login*	*Login* berhasil dan menampilkan halaman awal masing-masing aktor	[√] Berhasil [ ] Gagal
		Kolom *username* atau *password* tidak sesuai	Menampilkan pesan "*username* atau *password* salah"	[√] Berhasil [ ] Gagal
		Salah satu kolom *username* atau *password* kosong	Menampilkan pesan "*please fill out this field*" dengan menunjuk pada kolom yang belum diisi	[√] Berhasil [ ] Gagal
2.	*View* Data Pegawai	Klik menu data pegawai dan klik sub menu daftar pegawai	Menampilkan halaman data pegawai	[√] Berhasil [ ] Gagal
		Klik tombol detail pada salah satu data pegawai	Menampilkan halaman detail pegawai yang dipilih	[√] Berhasil [ ] Gagal
3.	*View* Absensi	Klik menu data pegawai dan klik sub menu absensi pegawai	Menampilkan halaman absensi pegawai	[√] Berhasil [ ] Gagal

4.	*View* Data Kamar	Klik menu data kamar dan klik sub menu daftar kamar	Menampilkan halaman daftar kamar	[√] Berhasil [ ] Gagal
5.	*View* Data Tipe Kamar	Klik menu data kamar dan klik sub menu tipe kamar	Menampilkan halaman tipe kamar	[√] Berhasil [ ] Gagal
6.	*View* Data Komplain	Klik menu data komplain	Menampilkan halaman data komplain	[√] Berhasil [ ] Gagal
		Klik tab menu belum/tidak teratasi	Menampilkan data komplain yang belum atau tidak teratasi	[√] Berhasil [ ] Gagal
		Klik tab menu teratasi	Menampilkan data komplain yang teratasi	[√] Berhasil [ ] Gagal
7.	*View* Data *check in*	Klik menu data reservasi dan klik sub menu *check in*	Menampilkan halaman data *check in*	[√] Berhasil [ ] Gagal
		Klik tombol detail pada salah satu data *check in*	Menampilkan detail *check in* yang dipilih	[√] Berhasil [ ] Gagal
		Klik tombol kembali	Menampilkan halaman data *check in*	[√] Berhasil [ ] Gagal
8.	Mengelola data pegawai	Klik menu data pegawai	Menampilkan halaman data pegawai	[√] Berhasil [ ] Gagal

		Klik tombol tambah data dan mengisi kolom serta klik tombol simpan	Menampilkan halaman data pegawai dengan data pegawai yang berhasil diinputkan	[√] Berhasil [ ] Gagal
		Kolom kosong tidak diisi	Menampilkan pesan "please fill out this field" dengan menunjuk pada kolom yang belum diisi	[√] Berhasil [ ] Gagal
		Klik tombol batal	Menampilkan halaman daftar kamar	[√] Berhasil [ ] Gagal
		Klik tombol edit dan mengubah data pegawai serta klik simpan	Menampilkan halaman data pegawai dengan data pegawai yang berhasil diubah	[√] Berhasil [ ] Gagal
		Klik tombol delete klik tombol "ok"	Menampilkan pesan notifikasi "Apakah anda yakin ingin menghapus data?" kemudian menampilkan halaman data pegawai dengan data yang berhasil dihapus	[√] Berhasil [ ] Gagal
		Klik tombol "cancel"	Tidak menghapus data pegawai dan menampilkan halaman data pegawai	[√] Berhasil [ ] Gagal
9.	*View* Data *check out*	Klik menu data reservasi dan klik sub menu *check out*	Menampilkan halaman data *check out*	[√] Berhasil [ ] Gagal

		Klik tombol detail pada salah satu data *check in*	Menampilkan detail *check out* yang dipilih	[√] Berhasil [ ] Gagal
		Klik tombol kembali	Menampilkan halaman data *check out*	[√] Berhasil [ ] Gagal
10.	*View* Pembobotan Kriteria	Klik menu indikator pengukuran	Menampilkan indikator pengukuran	[√] Berhasil [ ] Gagal
11.	*Update* bobot kriteria	Klik menu indikator pengukuran	Menampilkan halaman indikator pengukuran	[√] Berhasil [ ] Gagal
		Klik tombol *update* dan mengubah data nilai kepentingan antar kriteria lalu klik tombol simpan	Menyimpan data nilai kepentingan antar kriteria dan nilai bobot prioritas di masing-masing kriteria hasil perhitungan metode AHP dan menampilkan halaman indikator pengukuran	[√] Berhasil [ ] Gagal
		Klik tombol batal	Menampilkan halaman indikator pengukuran	[√] Berhasil [ ] Gagal
12.	*View* Produktivitas	Klik menu produktivitas	Menampilkan halaman produktivitas	[√] Berhasil
		Klik tombol detail produktivitas pada salah satu data produktivitas	Menampilkan halaman detail produktivitas yang dipilih	[√] Berhasil [ ] Gagal
		Klik tombol kembali	Menampilkan halaman produktivitas	[√] Berhasil

13.	*Input* Perhitungan Produktivitas	Klik menu produktivitas	Menampilkan halaman produktivitas	[√] Berhasil [ ] Gagal
		Klik tombol tambah data dan memilih bulan dan tahun serta klik tombol tampilkan	Menampilkan nilai pada masing-masing kolom sesuai dengan data *real* yang didapat selama periode perhitungan	[√] Berhasil [ ] Gagal
		Klik tombol simpan	Menampilkan halaman produktivitas dengan nilai perhitungan yang berhasil diinputkan	[√] Berhasil [ ] Gagal
		Klik tombol batal	Menampilkan halaman produktivitas	[√] Berhasil [ ] Gagal
14.	Mengelola Data User	Klik menu manajemen *user*	Menampilkan halaman manajemen *user*	[√] Berhasil [ ] Gagal
		Klik menu tambah data dan mengisi id pegawai serta tekan enter	Menampilkan halaman tambah *user* dan menampilkan isi kolom nama pegawai dan departemen sesuai dengan id pegawai	[√] Berhasil [ ] Gagal
		Klik tombol simpan	Menampilkan halaman manajemen *user* sesuai dengan data *user* yang berhasil diinputkan	[√] Berhasil [ ] Gagal
		Klik tombol batal	Menampilkan halaman manajemen *user*	[√] Berhasil [ ] Gagal

		Klik tombol centang pada salah satu data user dan klik "ok"	Menampilkan pesan notifikasi "Apakah anda yakin ingin mengkonfirmasi *user*?" dan menampilkan halaman manajemen *user*	[√] Berhasil [ ] Gagal
15.	Mengelola data tipe kamar	Klik menu data kamar dan klik sub menu tipe kamar	Menampilkan halaman tipe kamar	[√] Berhasil [ ] Gagal
		Klik tombol tambah data dan mengisi kolom serta klik tombol simpan	Menampilkan halaman tipe kamar dengan data tipe kamar yang berhasil diinputkan	[√] Berhasil [ ] Gagal
		Kolom kosong tidak diisi	Menampilkan pesan "please fill out this field" dengan menunjuk pada kolom yang belum diisi	[√] Berhasil [ ] Gagal
		Klik tombol batal	Menampilkan halaman tipe kamar	[√] Berhasil [ ] Gagal
		Klik tombol edit dan mengubah data tipe kamar serta klik simpan	Menampilkan halaman tipe kamar dengan data tipe kamar yang berhasil diubah	[√] Berhasil [ ] Gagal
		Klik tombol delete klik tombol "ok"	Menampilkan pesan notifikasi "Apakah anda yakin ingin menghapus data?" kemudian menampilkan halaman tipe kamar dengan data yang berhasil dihapus	[√] Berhasil [ ] Gagal

		Klik tombol “cancel”	Tidak menghapus data tipe kamar dan menampilkan halaman tipe kamar	[√] Berhasil [ ] Gagal
16.	Mengelola data daftar kamar	Klik menu data kamar dan klik sub menu daftar kamar	Menampilkan halaman daftar kamar	[√] Berhasil [ ] Gagal
		Klik tombol tambah data dan mengisi kolom serta klik tombol simpan	Menampilkan halaman daftar kamar dengan data daftar kamar yang berhasil diinputkan	[√] Berhasil [ ] Gagal
		Kolom kosong tidak diisi	Menampilkan pesan “please fill out this field” dengan menunjuk pada kolom yang belum diisi	[√] Berhasil [ ] Gagal
		Klik tombol batal	Menampilkan halaman daftar kamar	[√] Berhasil [ ] Gagal
		Klik tombol edit dan mengubah data daftar kamar serta klik simpan	Menampilkan halaman daftar kamar dengan data daftar kamar yang berhasil diubah	[√] Berhasil [ ] Gagal
		Klik tombol delete klik tombol “ok”	Menampilkan pesan notifikasi “Apakah anda yakin ingin menghapus data?” kemudian menampilkan halaman daftar kamar dengan data yang berhasil dihapus	[√] Berhasil [ ] Gagal
		Klik tombol “cancel”	Tidak menghapus data daftar kamar dan menampilkan halaman daftar kamar	[√] Berhasil [ ] Gagal

17.	Mengelola data reservasi	Klik menu data reservasi dan klik sub menu *check in*	Menampilkan halaman daftar *check in*	[√] Berhasil [ ] Gagal
		Klik tombol tambah data dan mengisi kolom serta klik tombol simpan	Menampilkan halaman daftar *check in* dengan data *check in* yang berhasil diinputkan	[√] Berhasil [ ] Gagal
		Kolom kosong tidak diisi	Menampilkan pesan "please fill out this field" dengan menunjuk pada kolom yang belum diisi	[√] Berhasil [ ] Gagal
		Klik tombol batal	Menampilkan halaman daftar *check in*	[√] Berhasil [ ] Gagal
		Klik tombol edit dan mengubah data *check in* serta klik simpan	Menampilkan halaman daftar *check in* dengan data *check in* yang berhasil diubah	[√] Berhasil [ ] Gagal
		Klik tombol delete klik tombol "ok"	Menampilkan pesan notifikasi "Apakah anda yakin ingin menghapus data?" kemudian menampilkan halaman daftar *check in* dengan data yang berhasil dihapus	[√] Berhasil [ ] Gagal
		Klik tombol "cancel"	Tidak menghapus data *check in* dan menampilkan halaman daftar *check in*	[√] Berhasil [ ] Gagal
18.	*Input* Absensi Pegawai	Klik menu absensi pegawai	Menampilkan halaman absensi pegawai	[√] Berhasil [ ] Gagal

		Klik tombol *input* absensi dan mengisi id pegawai kemudian tekan *enter*	Menampilkan halaman *input* absensi dengan kolom nama pegawai terisi sesuai dengan id pegawai	[√] Berhasil [ ] Gagal
		Mengisi *form* input absensi dan klik simpan	Menampilkan halaman absensi pegawai dengan data absensi yang berhasil diinputkan	[√] Berhasil [ ] Gagal
		Klik batal	Menampilkan halaman absensi pegawai	[√] Berhasil
19.	*Input* Penanganan Komplain	Klik menu data komplain	Menampilkan halaman data komplain	[√] Berhasil [ ] Gagal
		Pilih tab belum/tidak teratasi dan klik tombol penanganan pada data komplain	Menampilkan halaman data komplain	[√] Berhasil [ ] Gagal
		Mengisi *form* penanganan komplain dan klik simpan	Menampilkan halaman data komplain dengan data penanganan yang berhasil diinputkan	[√] Berhasil [ ] Gagal
20.	*Input* Komplain	Klik tombol komplain	Menampilkan halaman *input* komplain	[√] Berhasil [ ] Gagal
		Mengisi kolom nomor identitas kemudian tekan *enter*	Menampilkan nama *customer* sesuai dengan nomor identitas	[√] Berhasil [ ] Gagal
		Mengisi *form* input komplain dan klik simpan	Menampilkan utama dengan komplain yang berhasil diinputkan	[√] Berhasil

21.	*Logout*	Klik tombol *logout*	Berhasil keluar sistem dan menampilkan halaman awal	[√]Berhasil [ ] Gagal