PENGEMBANGAN SISTEM INFORMASI PENGELOLAAN KLINIK GIGI BERBASIS *WEBSITE* MENGGUNAKAN PRINSIP *POINT OF SALE*

(STUDI KASUS: KLINIK GIGI SENYUM SEHAT DENTAL CARE)

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Rafiqah Majidah
NIM: 155150200111211

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2019

PENGESAHAN

PENGEMBANGAN SISTEM INFORMASI PENGELOLAAN KLINIK GIGI BERBASIS WEBSITE* MENGGUNAKAN PRINSIP *POINT OF SALE

(STUDI KASUS: KLINIK GIGI SENYUM SEHAT DENTAL CARE)

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Rafiqah Majidah
NIM: 155150200111211

Skripsi ini telah diuji dan dinyatakan lulus pada
28 Mei 2019
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Denny Sagita R., S.Kom, M.Kom	Komang Candra B., S.Kom, M.T, M.Sc
NIP: 19851124 201504 1 001	NIK: 2016078907111001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T., M.T., Ph.D.
NIP: 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar referensi.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 20 Mei 2019

Rafiqah Majidah

NIM: 155150200111211

PRAKATA

Puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat, taufik dan hidayah-Nya sehingga laporan skripsi yang berjudul "Pengembangan Sistem Informasi Pengelolaan Klinik Gigi Berbasis *Website* Menggunakan Prinsip *Point of Sale* (Studi Kasus: Klinik Gigi Senyum Sehat Dental Care)" ini dapat terselesaikan.

Penulis menyadari bahwa skripsi ini tidak akan berhasil tanpa bantuan dari beberapa pihak. Oleh karena itu, penulis ingin menyampaikan rasa hormat dan terima kasih kepada:

1. Ayah saya, Drs. Zul'aidi dan ummi saya, Desfiwati, S.Pd., M.Si., yang selalu memberikan dukungan do'a, moril, materil, dan hal lain yang senantiasa menjadi motivasi yang baik bagi saya sehingga dapat menyelesaikan skripsi ini. Dan tak lupa juga untuk kakak-kakak saya, Nadiya, Faaizah dan Faatin, serta adik-adik saya, Izzuddin dan Faris yang selalu menghadirkan rasa kangen, canda dan tawa dalam masa-masa jauh dari rumah.
2. Bapak Denny Sagita R., S.Kom., M.Kom. selaku pembimbing I, serta Bapak Komang Candra B., S.Kom., M.T., M.Sc. selaku pembimbing II yang telah sabar dan tulus dalam memberikan banyak masukan, saran, arahan dan ilmu selama proses pengerjaan skripsi ini.
3. Bapak Wayan Firdaus Mahmudy, S.Si., M.T., Ph.D. selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya, Bapak Tri Astoto K., S.T., M.T., Ph.D. selaku Ketua Jurusan Teknik Informatika dan Bapak Agus Wahyu Widodo, S.T., M.Sc. selaku Ketua Program Studi Teknik Informatika.
4. Desy Diandra, Fitri Febriyani, dan Tria Melia, serta rekan-rekan seperjuangan yang telah memberi semangat serta motivasi kepada penulis selama penelitian dilakukan.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih banyak kekurangan, sehingga saran dan kritik yang membangun sangat penulis harapkan. Akhir kata penulis berharap skripsi ini dapat membawa manfaat bagi semua pihak yang menggunakannya.

Malang, 20 Mei 2019

Penulis

fiqahmajidah@student.ub.ac.id

ABSTRAK

Rafiqah Majidah, Pengembangan Sistem Informasi Pengelolaan Klinik Gigi Berbasis *Website* Menggunakan Prinsip *Point of Sale* (Studi Kasus: Klinik Gigi Senyum Sehat Dental Care)

Pembimbing: Denny Sagita R., S.Kom., M.Kom. dan Komang Candra B., S.Kom., M.T., M.Sc.

Senyum Sehat Dental Care merupakan klinik gigi yang sudah berjalan selama 3 tahun. Pasien yang terdaftar di klinik gigi ini lebih kurang 500 orang. Sampai saat ini semua proses bisnis pada klinik gigi ini masih dilakukan secara manual. Seiring dengan perkembangan klinik gigi dan bertambahnya jumlah pasien maka timbul berbagai macam masalah, kurangnya efisiensi kerja terkait waktu dan tenaga merupakan salah satu permasalahan yang timbul. Permasalahan selanjutnya yang timbul adalah tidak adanya informasi jumlah ketersediaan bahan yang jelas karena belum ada pencatatan terkait jumlah stok bahan yang tersedia dan yang keluar pada saat transaksi. Disisi lain, pembukuan pendapatan dan pengeluaran klinik masih belum konsisten. Sehingga, sering kali terjadi kekeliruan dalam pembukuan pemasukan dan pengeluaran klinik.

Berdasarkan permasalahan tersebut, salah satu upaya yang dapat dilakukan adalah membangun sebuah sistem yang efisien untuk membantu pemilik klinik gigi menyelesaikan permasalahan-permasalahan terkait efisiensi waktu dan tenaga, serta menanggulangi kesalahan dalam pembukuan klinik. Untuk menguji apakah sistem yang dikembangkan sudah benar sesuai dengan kebutuhan, maka dilakukan pengujian unit, integrasi dan validasi dengan metode *white-box* dan *black-box.* Pengujian non fungsional mengambil aspek *efficiency* dengan menguji *performance* sistem untuk melihat *load time* yang dibutuhkan, kemudian membandingkan waktu yang dibutuhkan sebelum dan sesudah adanya sistem.

Hasil dari penelitian ini berupa sistem informasi pengelolaan klinik gigi berbasis *website* menggunakan prinsip *point of sale* beserta hasil uji sistem tersebut. Tingkat keberhasilan pada pengujian unit, integrasi, dan validasi adalah 100% valid. Hasil dari pengujian *performance* juga bernilai valid. Sistem ini dapat meningkatkan efisiensi waktu 30 kali lebih cepat dibandingkan dengan sebelum menggunakan sistem.

Kata kunci: *point of sale*, klinik gigi, pengembangan sistem informasi, rekam medis.

ABSTRACT

Rafiqah Majidah, Development of Website-based Information System of Dental Clinic Management using Principle of Point of Sale (Case Study: Senyum Sehat Dental Care)

Supervisors: Denny Sagita R., S.Kom., M.Kom. dan Komang Candra B., S.Kom., M.T., M.Sc.

Senyum Sehat Dental Care is a dental clinic which has been operating for 3 years. The number of patients recorded by the dental clinic is numbered approximately at 500. Until now, all the business processes at this dental clinic are done manually. Along with the development and the increasing number of patients of this dental clinic, various problems arise, one of the problems that arise is lack of work efficiency in terms of time and exerted effort. The next problem is lack of information on the amount of dental care ingredients availability because there is no record regarding the amount of dental care ingredients stock available and those that are used during transactions. Besides that, there is also inconsistency of tracking and recording income and expense, which results in unforeseen consequences in regards to recording said transactions.

Based on these problems, it is necessary to develop an efficient system to assist the dental clinic to better save time and exerted efforts and also overcoming bookkeeping inconsistencies. In order to test whether the system that has been developed is correct according to the needs, unit, integration, and validation testing with white-box and black-box method used in this case. Non functional testing takes efficiency aspects by testing the system performance to see the load time, then comparing the time needed before and after using the system.

The results of this study is an website-based information system of dental clinic management using principle of point of sale at Senyum Sehat Dental Care along with the results from the implementation test of the system. The results of unit, integration, and validation testing are 100% valid. The result of performance testing is also valid. This system can increase time efficiency 30 times faster than before using the system.

Keywords: point of sale, dental clinic, information system development, medical record.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi informasi saat ini beriringan dengan perkembangan zaman, sekarang teknologi informasi sudah mencapai tahap kebutuhan hampir di semua lapisan masyarakat, baik perseorangan, instansi pemerintah, dan swasta memanfaatkan teknologi informasi. Pengelolaan teknologi informasi dianggap menjadi suatu hal yang sangat penting dalam sebuah oganisasi maupun bisnis karena dapat meningkatkan kompeksitas dan tugas manajemen, memiliki waktu tanggap (*response time*) yang lebih cepat, serta dapat membantu persaingan bisnis dalam pengaruh ekonomi internasional (Sutarman, 2009). Salah satu conoh penerapa teknologi informasi adalah sistem informasi, peran sistem informasi sangat berpengaruh bagi kemajuan di segala bidang untuk meningkatkan efektifitas dan efisiensi kerja sehingga dapat mempermudah manusia dalam mengerjakan suatu kegiatan. Sistem informasi juga merupakan salah satu aspek yang penting dalam bisnis, karena dapat membantu bisnis untuk meningkatkan produktifitas, baik dalam memperoleh, mengolah, serta menggunakan informasi secara akurat (Cahyodi & Arifin, 2017).

Senyum Sehat Dental Care merupakan klinik gigi yang sudah berjalan selama 3 tahun, klinik gigi ini memiliki 1 orang dokter utama sebagai pemilik klinik gigi dan 1 orang dokter yang bekerjasama dengan klinik gigi. Klink gigi ini juga memiliki 1 orang perawat sebagai resepsionis dan asisten dokter. Pasien yang terdaftar di klinik gigi ini lebih kurang 500 orang. Rata-rata jumlah pasien perawatan setiap bulannya mencapai angka 100-120 orang dengan keluhan yang berbeda-beda. Perkiraan pendapatan kotor dari klinik gigi ini lebih kurang Rp 15.000.000,00 setiap bulannya. Sampai saat ini semua proses bisnis pada Klinik Gigi Senyum Sehat Dental Care masih dilakukan secara manual, baik pendaftaran pasien, rekam medis, atau pun pencatatan pemasukan dan pengeluaran klinik.

Pada awalnya semua proses bisnis yang dilakukan secara manual tersebut tidak menjadi masalah, namun seiring dengan perkembangan klinik gigi dan bertambahnya jumlah pasien maka timbul berbagai macam masalah, menurut pemilik Klinik Gigi Senyum Sehat Dental Care drg. Munadiyah Asy Syahidah, kurangnya efisiensi kerja terkait waktu dan tenaga merupakan salah satu permasalahan yang timbul. Kurangnya efisiensi kerja yang berkaitan dengan tenaga dapat dilihat dari proses bisnis saat ini, dimana perawat harus berulang kali mengantarkan dan mengambil rekam medis pasien dari ruangan dokter ke meja resepsionis atau pun sebaliknya. Sedangkan permasalahan efisiensi waktu yang sering terjadi adalah poses pencarian kartu rekam medis sesuai dengan nama dan

no pasien oleh perawat memakan waktu cukup lama, yaitu sekitar 3-5 menit akibat banyaknya jumlah kartu rekam medis dari pasien terdaftar. Sering bila kartu rekam medis tersebut tidak ditemukan, maka pasien akan dibuatkan kartu rekam medis baru. Sehingga, memungkinkan terjadinya duplikasi data atau pun hilangnya riwayat medis pasien.

Permasalahan selanjutnya yang dirasakan oleh pemilik Klinik Gigi Senyum Sehat Dental Care adalah belum ada pencatatan terkait jumlah stok bahan yang tersedia dan yang keluar pada saat transaksi, sehingga tidak adanya informasi jumlah ketersediaan bahan yang jelas. Sering kali, dokter kekurangan bahan pada saat proses perawatan pasien yang mengakibatkan terhambatnya proses perawatan pasien, karena dokter harus menunggu perawat terlebih dahulu untuk membeli bahan yang dibutuhkan tersebut.

Point of sale merupakan program yang dirancang seperti sistem kasir yang tercatat. Beberapa keuntungan dari penerapan sistem *point of sale* adalah dapat meningkatnya kepercayaan pengguna sistem, mengurangi tingkat kesalahan, meningkatkan efisiensi kerja, meninjau dan melakukan manajemen stock, dan lain sebagainya (Fahrurroji, 2016). Dari permasalahan yang sudah dipaparkan sebelumnya, salah satu upaya yang dapat dilakukan adalah membangun sebuah sistem yang efisien yang dapat membantu pemilik klinik gigi untuk menyelesaikan permasalahan-permasalahan terkait efisiensi waktu dan tenaga. Sistem tersebut adalah sistem informasi pengelolaan klinik gigi berbasis *website* menggunakan prinsip *point of sale*. Didukung dengan penelitian yang dilakukan oleh Takeshi Shimmura, Motoyuki Akamatsu, dan Takeshi Takenaka yang menjelaskan bahwa dengan adanya sistem *point of sale* dapat meningkatkan watu rata-rata pelayanan sehingga menurunkan tingkat keluhan pelanggan, serta sistem ini dapat berkontribusi pada efisiensi dan kepuasan pelanggan (Shimmura, et al., 2009).

Sistem yang di bangun ini tidak hanya terpaku dengan penerapan prinsip *point of sale* saja, namun juga memperluas sistem *point of sale* konvensional agar perawat dan dokter dapat berbagi informasi antrian pasien secara *real-time* menggunakan sistem informasi serta dilengkapi dengan pengelolaan stok bahan. Berdasarkan dari permsalahan yang telah dijabarkan sebelumnya, penulis mengangkat sebuah penelitian dengan judul "Pengembangan Sistem Informasi Pengelolaan Klinik Gigi Berbasis *Website* Menggunakan Prinsip *Point of Sale* (Studi Kasus: Klinik Gigi Senyum Sehat Dental Care)".

1.2 Rumusan Masalah

Berikut adalah rumusan masalah yang akan dibahas dalam penelitian ini:

1. Bagaimana rekayasa kebutuhan sistem informasi pengelolaan klinik gigi berbasis *website* menggunakan prinsip *point of sale*?
2. Bagaimana perancangan dan implementasi sistem informasi pengelolaan klinik gigi berbasis *website* menggunakan prinsip *point of sale*?
3. Bagaimana pengujian fungsional dan non fungsional sistem informasi pengelolaan klinik gigi berbasis *website* menggunakan prinsip *point of sale* agar dapat digunakan untuk meningkatkan efisiensi waktu?

1.3 Tujuan

Tujuan dari penelitian di buat berdasarkan rumusan masalah yang telah dikemukakan sebelumnya, berikut adalah tujuan dari penelitian ini:

1. Mendapatkan hasil rekayasa kebutuhan sistem informasi pengelolaan klinik gigi berbasis *website* menggunakan prinsip *point of sale*.
2. Mendapatkan hasil perancangan dan mengimplementasikan sistem informasi pengelolaan klinik gigi berbasis *website* menggunakan prinsip *point of sale*.
3. Mendapatkan hasil pengujian sistem informasi pengelolaan klinik gigi berbasis *website* menggunakan prinsip *point of sale* agar dapat digunakan untuk meningkatkan efisiensi waktu.

1.4 Manfaat

Berikut adalah manfaat dari penelitian ini:

1. Penelitian ini diharapkan dapat membantu pemilik klinik gigi dalam mengatasi permasalahan-permasalahan yang terjadi sebelumnya, baik dari segi efisiensi waktu dan tenaga maupun pencatatan jumlah stok bahan.
2. Penelitian ini merupakan suatu sarana bagi penulis untuk menerapkan ilmu pengetahuan yang sudah diperoleh selama masa perkuliahan.

1.5 Batasan Masalah

Batasan masalah di buat agar penekanan tujuan dari penelitian ini tepat dalam mencapai sasaran, maka batasan masalah yang akan di bahas antara lain:

1. Hanya bisa diterapkan pada klinik gigi.
2. Data yang digunakan adalah data dari Klinik Gigi Senyum Sehat Dental Care.

3. Harus tersambung dengan internet.

1.6 Sistematika Pembahasan

Sistematika pembahasan laporan penelitian di buat sebagai gambaran dari laporan penelitian secara garis besar yang meliputi beberapa bab, antara lain:

BAB I PENDAHULUAN

Latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, batasan masalah, serta sistematika pembahasan dijelaskan pada bab ini.

BAB II LANDASAN KEPUSTAKAAN

Dasar teori yang akan dijadikan sebagai acuan dalam pengembangan sistem informasi pengelolaan klinik gigi menggunakan prinsip *point of sale* akan diuraikan pada bab ini.

BAB II METODOLOGI PENELITIAN

Metode dan tahapan-tahapan yang digunakan dalam proses pengembangan sistem informasi pengelolaan klinik gigi menggunakan prinsip *point of sale* akan dijabarkan pada bab ini.

BAB IV REKAYASA KEBUTUHAN

Tahap analisis kebutuhan dalam proses pengembangan sistem informasi pengelolaan klinik gigi menggunakan prinsip *point of sale* akan dijabarkan pada bab ini.

BAB V PERANCANGAN DAN IMPLEMENTASI

Tahap perancangan dan implementasi sistem dalam proses pengembangan sistem informasi pengelolaan klinik gigi menggunakan prinsip *point of sale* akan dijabarkan pada bab ini.

BAB VI PENGUJIAN DAN ANALISIS

Tahap pengujian sistem dalam proses pengembangan sistem informasi pengelolaan klinik gigi menggunakan prinsip *point of sale* akan dijabarkan pada bab ini.

BAB VII PENUTUP

Kesimpulan dan saran akan diuraikan pada bab ini. Kesimpulan didapatkan dari hasil penelitian serta saran diberikan untuk pengembangan lebih lanjut dari sistem informasi pengelolaan klinik gigi menggunakan prinsip *point of sale*.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

Dalam peneIitian yang dilakukan oleh Takeshi Shimmura, Motoyuki Akamatsu, dan Takeshi Takenaka pada tahun 2009 yang membahas tentang sistem manajemen restoran terkait proses berbagi informasi pesanan secara *real-time* menjelaskan bahwa, sistem *point of sale* yang dikembangkan merupakan perluasan dari sistem *point of sale* konvensional yang mencatat pesanan setiap meja bukan setiap hidangan. Proses pelayanan restoran terdiri dari penerimaan pesanan, persiapan, dan penyajian. Tugas penting adalah melayani pelanggan dengan waktu yang cepat dan di tempat yang tepat. Tugas tersebut tidak mudah dilakukan ketika restoran dalam keadaan sibuk, pelayan harus menghafalkan banyak pesanan dari meja yang berbeda dan di tuntut untuk menyajikan pesanan dengan cepat. Namun, dengan adanya sistem *point of sale* yang dilengkapi proses berbagi informasi pesanan secara *real-time* diterapkan pada restoran dapat meningkatkan watu rata-rata pelayanan sehingga menurunkan tingkat keluhan pelanggan, serta sistem ini dapat berkontribusi pada efisiensi dan kepuasan pelanggan. Menurut Takeshi Shimmura, Motoyuki Akamatsu, dan Takeshi Takenaka sistem *point of sale* juga harus dapat menawarkan fitur penting seperti transaksi atau pendaftaran pesanan untuk menghindari kelasahan perhitungan serta pemeriksaan pesanan atau ketersediaan pesanan untuk merampingkan waktu persiapan dan mengurangi waktu tunggu pelanggan yang merupakan faktor penting terkait kepuasan pelanggan (Shimmura, et al., 2009).

Selanjutnya penelitian yang dilakukan oleh Silvester Dian Handy Permana dan Faisal pada tahun 2015 tentang analisa dan perancangan aplikasi *point of sale* untuk mendukung manajemen hubungan pelanggan yang bertujuan membantu pengusaha kecil dan menengah dalam pengelolaan data dengan menggunakan sistem layanan pembelian. Menurut Silvester Dian Handy Permana dan Faisal analisa dan perancangan aplikasi *point of sale* yang baik adalah aplikasi yang mecakup semua *scope*/ruang lingkup yang ditentukan, dimana aspek integrasi antar bagian yang baik sangatlah penting sehingga data dalam aplikasi dapat saling berinteraksi untuk meningkatkan kecepatan, akurasi, dan kemudahan. Penerapan aplikasi *point of sale* dapat membantu tugas-tugas seluruh *stakeholder* yang berhubungan langsung dengan aplikasi tersebut serta dapat mengatasi permasalahan dengan menyajikan informasi secara lebih baik dan terkomputerisasi (Permana & Faisal, 2015).

Persamaan dari 2 penelitian yang telah dijabarkan sebelumnya dengan penelitian dilakukan oleh penulis adalah tujuan untuk meningkatkan efisiensi

waktu, kemudahan, serta akurasi perhitungan dalam pengelolaan usaha. Karena jika hanya dilakukan dengan proses bisnis konvensional maka akan sulit untuk mendapatkan tujuan tersebut. Dengan adanya sistem *point of sale* akan mempermudah pemilik klinik dalam mengumpulkan informasi-informasi penting yang dapat digunakan untuk meminimalisir adanya masalah sehingga dapat memajukan usaha.

2.2 Praktik Kedokteran

2.2.1 Praktik Kedokteran Menurut Undang-Undang (UU)

Dokter dan dokter gigi adalah dokter, dokter spesialis, dokter gigi, dan dokter gigi spesialis. Sesuai dengan peraturan perundang-undangan, dokter, dokter spesialis, dokter gigi, dan dokter gigi spesialis yang diakui Pemerintah Republik Indonesia adalah lulusan pendidikan kedokteran atau kedokteran gigi, baik didalam maupun diluar negeri. (Pasal 1, butir 2, UU RI No. 29 tahun 2004 tentang Praktik Kedokteran).

Praktik kedokteran merupakan rangkaian kegiatan yang dilakukan oleh dokter terhadap pasien dalam melaksanakan upaya kesehatan. (Pasal 1, Ayat 1. UU RI No. 29 tahun 2004 tentang Praktik Kedokteran). Praktik kedokteran dilaksanakan berasaskan Pancasila dan didasarkan pada nilai ilmiah, manfaat, keadilan kemanusiaan, keseimbangan, serta perlindungan dan keselamatan pasien. (Pasal 2. UU RI No. 29 tahun 2004 tentang Praktik Kedokteran dan penjelasannya).

2.2.2 Praktik Kedokteran yang Sesuai Standar

Seorang dokter dianggap melaksanakan praktik kedokteran sesuai standar jika sudah memenuhi syarat kelayakan dan kepatutan, serta didukung dengan adanya struktur berupa sumber daya manusia sebagai penunjang, sarana dan prasarana, peralatan, sistem-sistem, serta logistic yang optimal sesuai dengan tingkat dan lokasi praktik. Disisi lain, dokter juga harus memenuhi kewajiban lain yang ditentukan UU menyangkut pemberian asuhan klinik, seperti menjaga kerahasiaan pasien, menjelaskan dengan lengkap dan benar terkait hasil pengobatan pasien kepada pasien atau keluarganya kemudian membuat *informed consent*, membuat rekam medis, dan melakukan kendali mutu dan biaya (Jacobalis, et al., 2016).

Hasil yang diharapkan dari praktik kedokteran yang baik adalah pasien dan dokter sama-sama merasa puas. Kepuasan tersebut mencakup (Jacobalis, et al., 2016):

1. Keselamatan pasien
2. Efisien dalam pemanfaatan sumber daya
3. Asuhan telah berfokus pada pasien

4. Layanan dan asuhan yang tepat waktu
5. Asuhan yang secara klinis efektif
6. Perlakuan yang adil pada pasien

2.3 Konsep Dasar Sistem Informasi

2.3.1 Pengertian Sistem Informasi

Sistem didefinisikan sebagai kumpulan dari sub-sub sistem yang saling berkaitan satu sama lain dan bekerja sama untuk mencapai tujuan yang sama. Informasi adalah hasil dari pengolahan data yang bermanfaat dan berarti. Sehingga, sistem informasi dapat diartikan sebagai kumpulan dari sub-sub sistem yang saling berkaitan satu sama lain dan bekerja sama untuk mencapai tujuan yang sama yaitu mengolah data agar menjadi informasi yang berarti dan berguna (Susanto, 2017). Secara umum, sistem infromasi juga dapat diartikan sebagai suatu sistem penyedia informasi yang digunakan untuk bahan pertimbangan dalam pengambilan keputusan/kebijakan.

2.3.2 Komponen Sistem Informasi

Menurut Susanto, komponen sistem informasi antara lain, komponen masukan, model, keluaran, teknologi, basis data, dan kendali. Keenam komponen tersebut sangat dibutuhkan karena istem informasi tidak akan terwujud apabila salah satu komponen tidak ada. Komponen pertama adalah komponen masukan, komponen masukan merupakan data yang dimasukkan ke dalam sistem sebagai bahan dalam pengelolaan informasi. Kedua komponen model, merupakan gabungan dari model matematika, *procedure*, dan logika yang digunakan sebagai pengolah data masukkan agar dapat memberikan hasil keluaran yang sesuai dengan apa yang diinginkan. Ketiga komponen keluaran, informasi yang berarti dan berguna untuk pengguna merupakan komponen keluaran dari sistem informasi. Keempat komponen teknologi, merupakan alat yang digunakan untuk menerima masukan, menjalankan model, menyimpan dan mengakses data serta memberikan hasil keluaran yang diharapkan. Selanjutnya komponen basis data, basis data merupakan data yang tersimpan pada perangkat keras komputer yang saling berhubungan dan dapat dimanipulasi menggunakan perangkat lunak. Terakhir komponen kendali, merupakan komponen yang dapat menjamin sistem informasi aman dari ganguan (Susanto, 2017).

2.4 Point of Sale

Menurut Oktaviani, *point of sale* merupakan program yang di rancang menggunakan sistem kasir yang tercatat, sehingga dibutuhkan oleh pengusaha

untuk mempermudah transaksi penjualan. Sistem *point of sale* terdiri dari dua modul yaitu administrasi dan kasir. Modul administrasi digunakan untuk berbagai macam transaksi pembelian dan invetarisasi semua transaksi, sedangkan modul kasir dugunakan hanya untuk melayani penjualan (Oktaviani, 2013).

Sistem *point of sale* dibuat untuk memperrmudah kasir dalam mengelola proses penjualan agar lebih praktis dan cepat, sehingga dapat menghindari antrian pelanggan. Beberapa sistem *point of sale* dilengkapi dengan *simple inventory management* yang dapat membantu pengguna dalam melakukan pengelolaan stock barang dengan sederhana. Pada umumnya sistem *point of sale* dilengkapi dengan *hardware* seperti *printer*, *barcode*, *scanner*, dan *cash drawer*. Sistem *point of sale* cocok digunakan jika suatu perusahaan hanya membutuhkan suatu aplikasi kasir dalam mengelola transaksi penjualan dan tidak terlalu membutuhkan aplikasi di bagian perusahaan yang lain seperti akuntansi dan *human resource* (Poluakan, 2017).

2.4.1 Manfaai *Point of Sale*

Menurut Rahman, sistem *point of sale* memiliki beberapa keuntungan antara lain dapat meningkatkan kualitas pelayanan dengan menjalankan proses transaksi secara cepat dan sistematis yang dapat meningkatkan *market interest* dan mendukung orientasi pelayanan usaha terhadap konsumen. *Point of sale* juga diaggap mampu untuk meningkatkan citra usaha karena setiap konsumen akan melihat usaha yang dijalankan sebagai sebuah usaha yang dikelola dengan professional. Selain itu terdapat *competitive advantage,* dengan adanya penerapan teknologi informasi yang mengutamakan efisiensi waktu, perusahaan dapat meningkatkan daya saing perusahaan untuk menghadapi era *global market*. Keuntungan lain dari penggunaan aplikasi *point of sale* adalah adanya kemudahan dalam pengambilan keputusan dan proses *controlling*. Setiap laporan disediakan dengan cepat, sehingga proses *controlling* dapat dengan mudah dilakukan serta mempermudah proses pengambilan keputusan baik secara personal maupun kolektif (Rahman, 2013).

2.5 *System Development Life Cycle* (SDLC)

Menurut Rosa dan Shalahuddin, SDLC merupakan proses mengembangkan atau mengubah suatu perangkat lunak dengan menggunakan metodologi dan model-model yang telah digunakan para pengembang perangkat lunak sebelumnya (S & Shalahuddin, 2013). Secara global SDLC dapat dikatakan sebagai suatu proses dari beberapa tahap yang saling berkesinambungan untuk menciptakan atau merubah suatu sistem. Tahapan SDLC antara lain sebagai berikut:

1. Tahap perencanaan sistem

   Tahap perencanaan sistem merupakan tahap pertama yang dilakukan untuk mendeskrpsikan masalah yang akan diselesaikan dan batasan-batasan dari sistem yang akan dibuat.

2. Tahap analisis sistem

   Pada tahap ini dilakukan proses pengumpulan informasi-informasi yang berkaitan dengan sebab-akibat yang ditimbulkan oleh masalah, sehingga akan menghasilkan definisi mengenai sistem dan kebutuhannya.

3. Tahap desain

   Pada tahap ini dilakukan proses pembuatan rancangan sistem yang akan dibangun. Hasil dari analisis kebutuhan dijadikan sebagai acuan dari rancangan desain yang di buat, mulai dari masukan, proses, dan keluarannya.

4. Tahap implementasi

   Pada tahap ini hasil dari tahap desain akan ditransformasikan ke dalam bahasa komputer untuk menyelesaikan sistem dan mulai menggunakan sistem serta melakukan pengujian untuk memastikan sistem bekerja dengan baik dan sesuai dengan kebutuhan.

5. Tahap pemeliharaan

   Setelah tahap implementasi dilakukan maka tahap selanjutnya adalah pemeliharaan sistem yang meliputi penggunaan sistem, evaluasi dan audit, perbaikan dan peningkatan sistem (Gintoro, et al., 2010).

2.5.2 Waterfall Model

Waterfall model juga sering disebut alur hidup klasik (*classic life cycle*) atau model sekuensi linier (*sequential linear*) merupakan salah satu model SDLC secara sekuensial atau terurut yang di mulai dari analisis, desain, implementasi, pengujian, dan tahap pendukung. Tahapan-tahapan pada *waterfall model* dijelaskan pada Gambar 2.1.

1. *Requirment Analysis and Definition*

   Layanan sistem, kendala, dan tujuan ditetapkan oleh hasil konsultasi dengan pengguna yang kemudian didefinisikan secara rinci dan berfungsi sebagai spesifikasi sistem.

2. *System and Software Design*

   Tahapan perancangan sistem membentuk arsitektur sistem secara keseluruhan berdasarkan persyaratan yang telah ditetapkan. Perancangan

perangkat lunak melibatkan identifikasi dan penggambaran abstraksi sistem dasar perangkat lunak dan hubungannya.

3. *Implementation and Unit Testing*

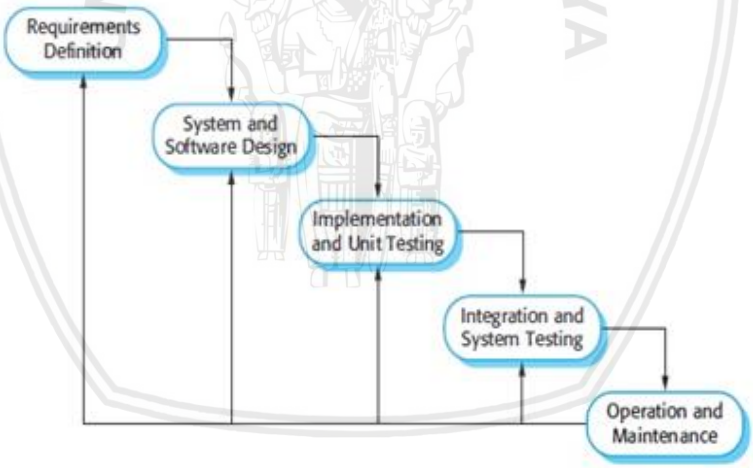
   Hasil dari desain perangkat lunak direalisasikan sebagai serangkaian program atau unit program. Pengujian melibatkan verifikasi bahwa setiap unit memenuhi spesifikasinya.

4. *Integration and System Testing*

   Setiap unit program akan diintegrasikan satu sama lain dan diuji sebagai sebuah sistem lengkap untuk memastikan apakah sesuai dengan kebutuhan perangkat lunak atau tidak. Setelah itu perangkat lunak akan dikirimkan ke pengguna.

5. *Operation and Maintenance*

   Sistem mulai digunakan secara nyata. *Maintenance* melibatkan pembetulan kesalahan yang tidak ditemukan pada tahapan-tahapan sebelumnya, meningkatkan implementasi dari unit sistem, dan meningkatkan layanan sistem sebagai kebutuhan baru (Sommerville, 2011).



Gambar 2.1 *Waterfall Model*

Sumber: (Sommerville, 2011)

Waterfall model sangat cocok digunakan pada pengembangan sistem jika kemungkinan terjadinya perubahan kebutuhan selama pengembangan perangkat lunak kecil dan kebutuhan dari pengguna sudah dipahami dengan jelas. Kelebihan dari *waterfall model* adalah sebuah tahap dilakukan setelah tahap sebelumnya selesai dilakukan, struktur tahap pengembangan sistem jelas, dan dokumentasi dihasilkan dari setiap tahap pengembangan (S & Shalahuddin, 2013).

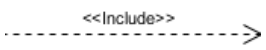
2.6 *Unified Modeling Language* (UML)

Unified Modeling Language (UML) merupakan bahasa yang digunakan untuk memodelkan perangkat lunak. Kebutuhan terhadap pemodelan visual untuk menggambarkan, membangun, merepresentasikan, dan dokumentasi dari perangkat lunak menjadi latar belakang munculnya UML (S & Shalahuddin, 2013)

2.6.1 Use Case Diagram

Menurut Rosa dan Shalahuddin, guna dari *use case diagram* adalah untuk mengetahui fungsi yang terdapat pada sistem dan siapa saja yang berhak untuk menggunakan fungsi-fungsi tersebut (S & Shalahuddin, 2013). Simbol-simbol yang digunakan dalam *use case diagram* dijelaskan pada Tabel 2.1.

Tabel 2.1 Simbol-simbol *Use Case Diagram*

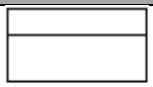
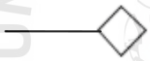
Nama	Simbol	Deskripsi
Aktor		Orang, proses, atau sistem lain yag berinterkasi dengan sistem yang akan dibuat.
Use case		Biasanya dinyatakan dengan menggunakan kata kerja dari fungsionalitas sistem yang saling bertukar pesan dengan aktor.
Generalisasi		Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungs yang lebih umum dibandingkan yang lainnya.
Asosiasi		Menggambarkan hubungan statis antar elemen dan menggambarkan komunikasi antara aktor dan *use case*.
Include		Relasi tambahan antar *usecase* yang berarti suatu *usecase* selau memerlukan *usecase* tersebut untuk bisa menjalankan fungsinya.
Extend		Relasi tambahan antar *usecase* dengan *usecase*. Relasi ini berarti sebuah *usecase* pada suatu waktu membutuhkan perluasan *usecase* yang bersifat opsional.

Sumber: (S & Shalahuddin, 2013)

2.6.2 Class Diagram

Class diagram di buat untuk menggambarkan struktur sistem dari segi pendefinisian kelas-kelas dalam mengembangkan sistem (S & Shalahuddin, 2013). *Class diagram* digunakan sebagai acuan dalam melakukan implementasi agar perangkat lunak dan dokumentasi perancangan sinkron. Simbol-simbol yang digunakan dalam *class diagram* dijelaskan pada Tabel 2.2.

Tabel 2.2 Simbol-simbol *Class Diagram*

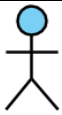
Nama	Simbol	Deskripsi
Class		Kelas pada struktur sistem.
Association		Relasi antar kelas dengan makna general.
Interface		Sama seperti konsep *interface* dalam pemrograman berorientasi objek.
Generalization		Relasi antar kelas dengan makna umum-khusus
Directed association		Relasi antar kelas dengan makna kelas yang satu digunakan untuk kelas yang lain.
Aggregation		Relasi antar kelas dengan makna semua bagian (*whole-part*).
Dependecy		Relasi antar kelas dengan makna kebergantungan antar kelas.

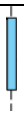
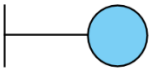
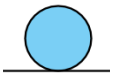
Sumber: (S & Shalahuddin, 2013)

2.6.3 Sequence Diagram

Sequence diagram menunjukkan urutan *event* kejadian dalam suatu waktu. *Sequence diagram* di buat untuk menggambarkan interaksi antar objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima oleh objek (S & Shalahuddin, 2013). Simbol-simbol yang digunakan dalam *sequence diagram* dijelaskan pada Tabel 2.3.

Tabel 2.3 Simbol-simbol *Sequence Diagram*

Nama	Simbol	Deskripsi
Aktor		Merupakan orang/aktor yang sedang berinteraksi dengan sistem
Message		Sebagai pesan yang memuat informasi-informasi tentang aktivitas yang terjadi yang dikirimkan antar objek atau bisa juga dari objek itu sendiri.

A focus of control		Menggambarkan tempat mulai dan berakhirnya sebuah pesan (*message*)
Boundary Class		Mereprentasikan halaman (*interface*) dari sistem.
Control Class		Kelas yang menghubungkan *boundary* dan *entity* class.
Entity Class		Kelas yang memodelkan *behaviour* dari data

Sumber: (S & Shalahuddin, 2013)

2.7 *CodeIgniter* (CI)

CodeIgniter merupakan *framework* PHP bersifat *open source* dengan model MVC (*Model, View, Controller*) untuk membangun aplikasi websie dinamis dengan cepat dan mudah. Membantu developer untuk mengerjakan aplikasi lebih cepat dari pada harus menulis semua code dari awal merupakan tujuan utama pengembangan *CodeIgniter*. *CodeIgniter* dirilis pertama kali pada 28 Februari 2006 (Daqiqil, 2011).

Kelebihan CodeIgniter yaitu (Daqiqil, 2011):

1. Kemudahan dalam mempelajari, memodifikasi, membuat *library* dan *helper*, serta meng-intregasikan *helper* dan *library*.
2. URL *friendly* dimana URL yang dihasilkan pada CodeIgniter seperti $_GET akan diminimalisasi dan digantikan dengan URI.
3. Menggunakan *pattern* MVC sehinngga struktur kode mnejadi lebih terstruktur dan memiliki standar yang jelas.

2.8 Pengujian Perangkat Lunak

Pengujian perangkat lunak dilakukan untuk melihat kesalahan pada sistem yang sedang dikembangkan. Pengujian perangkat lunak merupakan hal penting yang harus dilakukan untuk meningkatkan kualitas sistem serta mencegah pengeluaran biaya dalam perbaikan yang akan terjadi apabila tidak dilakukannya pengujian.

2.8.1 Black-box Testing

Menurut Roger S. Pressman, pengujian kotak hitam atau *black-box testing* berfokus pada spesifikasi fungsional dari perangkat lunak. *Black-box testing* memungkinkan untuk melakukan set kondisi masukan yang sepenuhnya akan

melaksanakan semua persyaratan fungsional untuk suatu program (Pressman, 2010).

1. Kelebihan *black-box testing*
   a. Dapat menemukan cacat pada *software.*
   b. Dapat memilih subset *test* secara efektif dan efisien.
2. Kelemahan *black-box testing*
   a. Penguji tidak pernah sangat yakin apakah perangkat lunak tersebut sadah benar-benar lulus uji.

2.8.2 *White-box Testing*

White-box testing merupakan pengujian perangkat lunak dari segi kode program dan desain, dimana pengujian bertujuan untuk menguji apakah kode program dan desain dapat menghasilkan fungsi-fungsi, masukan, dan keluaran yang sesuai dengan kebutuhan sistem. Contoh dari *white-box testing* misalkan pengujian alur program dengan menelusuri pengulangan (*looping*) pada logika pemrograman (S & Shalahuddin, 2013).

White-box testing dapat dilakukan dengan metode *basis path* dengan tahap-tahap sebagai berikut:

1. Pembuatan Flowgraph

   Flowgraph adalah alur dari logika program. Notasi pada flowgraph terdiri atas lingkaran dan panah. Lingkaran (node) menyatakan statemen procedural. Panah (edge) menyatakan aliran kendali atau alur perjalanan logika.

2. Penghitungan *Cyclomatic Complexity*

   Cyclomatic complexity adalah rumusan perhitungan yang menyediakan ukuran kompleksitas dari suatu program, semakin banyak *independent path* dalam modul maka kompleksitas ujinya semakin tinggi. (Pressman, 2012)

   Cyclomatic Complexity dihitung dengan tiga cara berikut:

   V(G) = E – N + 2 atau,

   V(G) = P + 1

   Keterangan:

   V(G) = Cyclomatic Complexity

   E = jumlah edge pada *flowgraph*

   N = jumlah node pada *flowgraph*

   P = jumlah predicate node pada *flowgraph*

3. *Independent Path*

   Independent path merupakan jalur pada *flowgraph* yang menghubungkan node awal dengan node akhir. *Independent path* minimal harus melewati sebuah edge baru dengan alur yang belum pernah di lalui sebelumnya.

4. *Test Case*

   Menguji atau mengeksekusi alur yang telah ditentukan dilakukan dengan membuat *test case*.

2.8.3 Aspek *Efficiency*

Efficiency merupakan kemampuan sistem dalam memberikan kinerja yang relatif sesuai terhadap sumber daya yang ada dan digunakan saat itu. Terdapat dua sub karakteristik dari aspek *efficieny*, yang diuraikan pada Tabel 2.4.

Tabel 2.4 Sub Karakteristik Aspek *Efficiency*

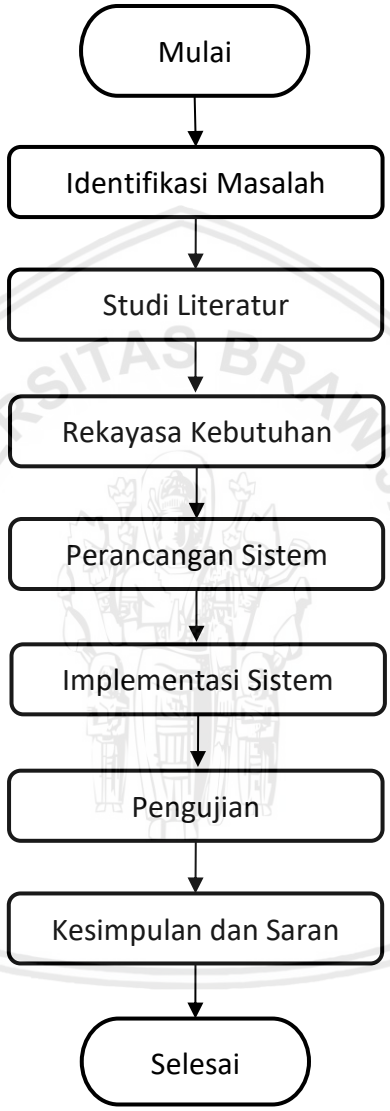
Sub Karakteristik	Deskripsi
Time behavior	Kemampuan sistem dalam memberikan waktu pengolahan dan respon yang sesuai saat melakukan fungsinya.
Resource behavior	Kemampuan sistem dalam menggunakan sumber daya yang ada saat melakukan fungsi

Menurut Nielsen, terdapat tiga batasan waktu dalam mengoptimalkan web berdasarkan persepsi manusia. Batasan pertama adalah 0,1 deik, pada batasan ini pengguna merasakan respon dari web yang cepat. Batasan kedua adalah 1 detik, pada batasan ini pengguna merasakan adanya jeda namun membuat pengguna dapat merasakan adanya proses yang sedang berjalan yang kemudian mengeluarkan hasil sehingga pengguna belum merasa terganggu dengan jeda ini. Batasan ketiga adalah 10 detik, pada batasan ini pengguna mulai terganggu dan berharap agar respon dapat berjalan lebih cepat. Jika melebihi 10 detik maka kebanyakan pengguna akan meninggalkan web, namun jika pengguna masih berada pada web tersebut maka pekerjaan mereka akan terhambat dengan adanya jeda (Nielsen, 2010).

Karakteristik *time behavor* dari pengujian aspek *efficiency* dapat dilakukan dengan menguji *performance* sistem menggunakan *tools* GTmetrix. GTmetrix merupakan *tools* yang dapat digunakan secara bebas untuk menganalisa *speed performance*, *load time*, serta dapat memberikan rekomendasi agar web dapat lebih efisien. GTmetrix merupakan gabungan antara *Google Page Speed Insight* dan *Yahoo Yslow,* sehingga dalam sekali pengujian dapat mendapatkan dua hasil sekaligus.

BAB 3 METODOLOGI

Metodologi merupakan suatu uraian atau pernyataan tentang kerangka konsep pemecahan masalah yang telah diidentifikasi atau dirumuskan untuk mendapatkan sebuah kesimpulan. Maka penulis menyimpulkan suatu alur metodologi seperti pada Gambar 3.1.



Gambar 3.1 Alur Metodologi

Metode pengembangan perangkat lunak yang digunakan dalam penelitian ini adalah *waterfall model*. Inti dari *waterfall model* adalah pengembangan sistem dilakukan secara berurutan atau linear pada setiap tahapnya, sehingga bisa dikatakan suatu tahap tidak bisa dilakukan sebelum tahap sebelumnya selesai dilakukan. *Waterfall model* terdiri dari tahap analisi kebutuhan, perancangan, implementasi, pengujian, dan *maintenance*. Disis lain, pengembangan perangkat lunak dalam penelitian ini menggunakan pendekatan *object-oriented* karena

pendekataan ini sangat natural dan sesuai dengan cara berpikir manusia serta dapat menjaga konsistensi model pada tahap analisis dan desain.

3.1 Identifikasi Masalah

Penelitian ini memfokuskan proses identifikasi masalah terhadap permasalahan yang terjadi pada Klinik Gigi Senyum Sehat Dental Care. Proses identifikasi masalah dilakukan dengan melakukan wawancara kepada pemilik klinik gigi terkait masalah-masalah yang dialami klinik gigi dalam melaksanakan proses bisnisnya. Dari hasil wawancara tersebut diperoleh identifikasi masalah yang akan diberikan solusi dalam penelitian ini.

3.2 Studi Literatur

Studi literatur adalah tahapan mencari teori dan pustaka yang relevan dengan rumusan masalah yang bertujuan untuk memperkuat permasalahan serta sebagai dasar dalam melakukan penelitian.

Teori dan pustaka yang berkaitan dengan penelitian ini meliputi:

1. Praktik Kedokteran
    - Praktik Kedokteran Menurut Undang-Undang (UU)
    - Praktik Kedokteran yang Sesuai Standar
2. Konsep Dasar Sistem Informasi
    - Pegertian Sistem Informasi
    - Komponen Sistem Informasi
3. *Point of Salas*
    - Manfaai *Point of Sale*
4. *System Development Life Cycle* (SDLC)
    - *Waterfall Model*
5. *Unified Modeling Language* (UML)
    - *Use Case Diagram*
    - *Class Diagram*
    - *Sequence Diagram*
6. *CodeIgniter* (CI)
7. Pengujian Perangkat Lunak
    - *Black-box Testing*

- *White-box Testing*
- Aspek *Efficiency*

3.3 Rekayasa Kebutuhan

Analisis kebutuhan merupakan tahap awal dalam melakukan pengembangan perangkat lunak yang memiliki peran sangat penting untuk kelanjutan tahap-tahap berikutnya. Dalam penelitian ini tahap rekayasa kebutuhan di mulai dengan proses elisitasi yang bertujuan untuk mengetahui masalah apa saja yang perlu dipecahkan, mengenali siapa saja pemangku kepentingan, dan mengenali tujuan dari pengembangan sistem. Proses elisitasi dilakukan dengan menggunakan 2 pendekatan, pendekatan pertama adalah observasi, dimana dilakukannya pengamatan terhadap semua proses bisnis yang terjadi pada Klinik Gigi Senyum Sehat Dental Care, baik proses transaksi, proses pengelolaan data pasien, dan proses-proses lainnya. Pendekatan yang kedua adalah dengan melakukan wawancara kepada pemilik Klinik Gigi Senyum Sehat Dental Care, drg. Munadiyah Asy Syahidah terkait masalah-masalah yang dialami oleh pemilik klinik gigi dalam melaksanakan proses bisnisnya.

Setelah proses elisitasi selesai dilakukan maka proses selanjutnya adalah melakukan analisis dan spesifikasi kebutuhan dari hasil proses elisitasi. Proses analisis dan spesifikasi kebutuhan dilakukan untuk mendeskripsikan sistem agar dapat menentukan kebutuhan fungsional dan non fungsional sistem serta siapa saja yang terlibat didalamnya. Proses selanjutnya adalah memodelkan kebutuhan fungsional dan non fungsional sistem ke dalam bentuk *use case diagram* dan *use case scenario*. Pemodelan kebutuhan bertujuan untuk menjelaskan apa yang dibutuhkan oleh pengguna, sebagai dasar perancangan sistem, dan menjadi reverensi dalam melakukan validasi kebutuhan.

3.4 Perancangan Sistem

Tahapan perancangan sistem bertujuan untuk membentuk arsitektur sistem secara keseluruhan yang digambarkan secara abstrak menggunakan diagram UML berdasarkan hasil dari tahap analisis kebutuhan. Tahap perancangan sistem dilakukan dengan pembuatan diagram-diagram perancangan yang sesuai dengan pendekatan *object-oriented*. Diagram-diagram tersebut antara lain, *sequence diagram* dan *class diagram*. *Sequence diagram* berfungsi untuk menggambarkan interaksi objek yang disusun berdasarkan urutan waktu. Pembuatan *sequence diagram* mengacu pada *use case diagram* dan *use case scenario* yang sudah dimodelkan sebelumnya pada tahap analisis kebutuhan untuk menggambarkan pertukaran pesan dalam *sequence diagram*. *Class diagram* merupakan diagram

yang menggambarkan stuktur dan hubungan antar kelas di dalam sistem yang akan dikembangkan. Sedangkan perancangan *database* digambarkan dengan menggunakan *Conceptual Data Model*. Diagram-diagram UML yang dihasilkan dari tahap perancangan ini nantinya akan menjadi acuan dalam tahap implementasi.

3.5 Implementasi Sistem

Tahap implementasi mengacu pada tahap perancangan sistem. Tahap implementasi dilakukan setelah semua tahap sebelumnya selesai dilakukan. Implementasi sistem dalam penelitian ini menggunakan bahasa pemrograman PHP, HTML, CSS, Javascript, serta basis data MySQL dengan menggunakan *framework codeigniter* dan *template* dari *bootstrap*. *Farmework codeigniter* dipilih karena memiliki waktu eksekusi yang lebih cepat dibandingkan *famework* PHP lainnya, disis lain *framework codeigniter* mampu membantu peneliti dalam pengembangan web dengan waktu yang lebih singkat karena penulisan kode yang lebih *simple*. Tahap implementasi dalam penelitian ini dilakukan untuk merealisasikan perancangan sistem ke dalam bentuk kode sebagai serangkaian program. Hasil dari tahap implementasi merupakan sebuah sistem, baik *interface* atau pun fungsi-fungsi yang siap untuk diuji pada tahap pengujian sistem.

3.6 Pengujian

Pengujian dilakukan untuk mendeteksi kesalahan pada program yang sedang dikembangkan. Pengujian dilakukan terhadap fungsional dan non fungsional sistem, antara lain:

1. Pengujian Unit

   Teknik yang digunakan adalah *white-box testing* menggunakan metode *basis path* dimana struktur dan alur logika pada kode program diuji untuk memastikan apakah stuktur dan alur logika tersebut telah benar atau valid.

2. Pengujian Itegrasi

   Teknik yang digunakan adalah *white-box testing* menggunakan metode *basis path* yang dilakukan untuk memastikan hubungan antar method yang dibentuk dapat berjalan dengan baik.

3. Pengujian Validasi

   Teknik yang digunakan adalah *black-box testing. Black-box testing* dilakukan dengan menguji fungsionalitas sistem tanpa harus menguji kode program dengan menggunakan *Test Case* untuk melihat output yang dihasilkan oleh sistem sudah sesuai atau belum.

4. Pengujian *Performance*

   Pengujian aspek *efficiency* pada penelitian ini mengambil sub karakteristik time behavior dengan melakukan pengujian terhadap *performance* sistem menggunakan *tools* GTmetrix. Pengujian *performance* ini dilakukan untuk memeriksa *load time* saat menjalankan sistem.

   Pengujian dilakukan dengan memasukkan URL dari sistem informasi pengelolaan klinik gigi menggunakan *point of sale* pada *website* GTmetrix, kemudian GTmetrix akan melakukan analisis terhadap *speed performance*, *load time*, dan beberapa aspek lainnya. Setelah itu, GTmetrix akan menampilkan hasil dari analisisnya, sehingga dapat diketahui bagaimana *load time* dari sistem ini.

Setelah keempat pengujian tersebut dilakukan, maka langkah selanjutnya adalah melakukan analisis terhadap hasil pengujian. Analisis dilakukan untuk mendapatkan kesimpulan dari pengujian yang telah dilakukan.

3.7 Kesimpulan dan Saran

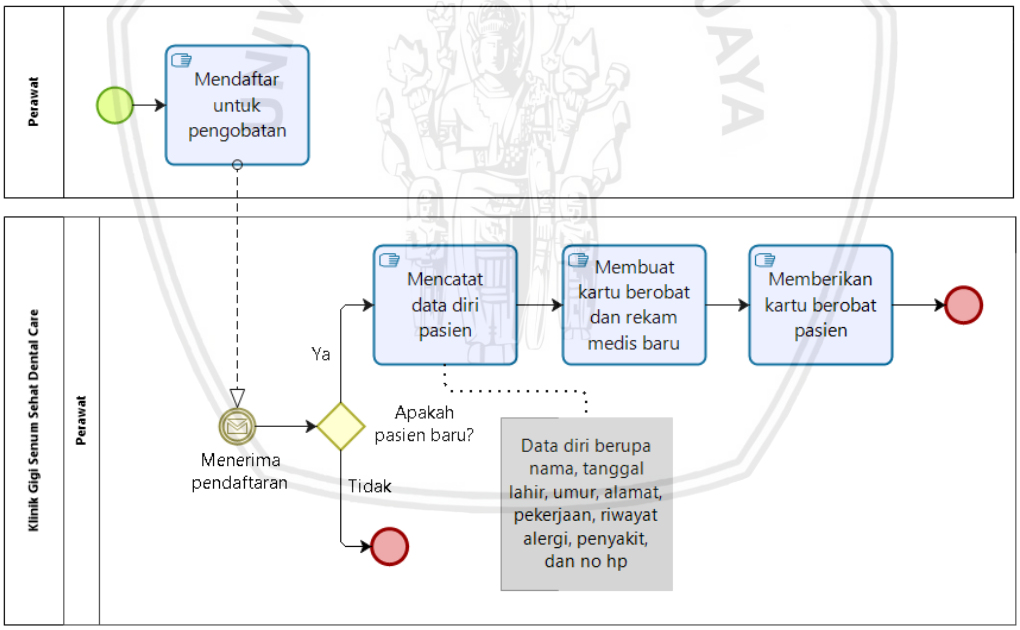
Kesimpulan dan saran diambil setelah semua tahap-tahap sebelumnya selesai dilaksanakan. Kesimpulan dibuat untuk menjawab rumusan masalah yang telah dijabarkan sebelumnya pada bab 1. Sedangkan saran dibuat untuk memperbaiki kekurangan sistem, agar dapat disempurnakan lagi pada penelitian selanjutnya.

BAB 4 REKAYASA KEBUTUHAN

4.1 Elisitasi Kebutuhan

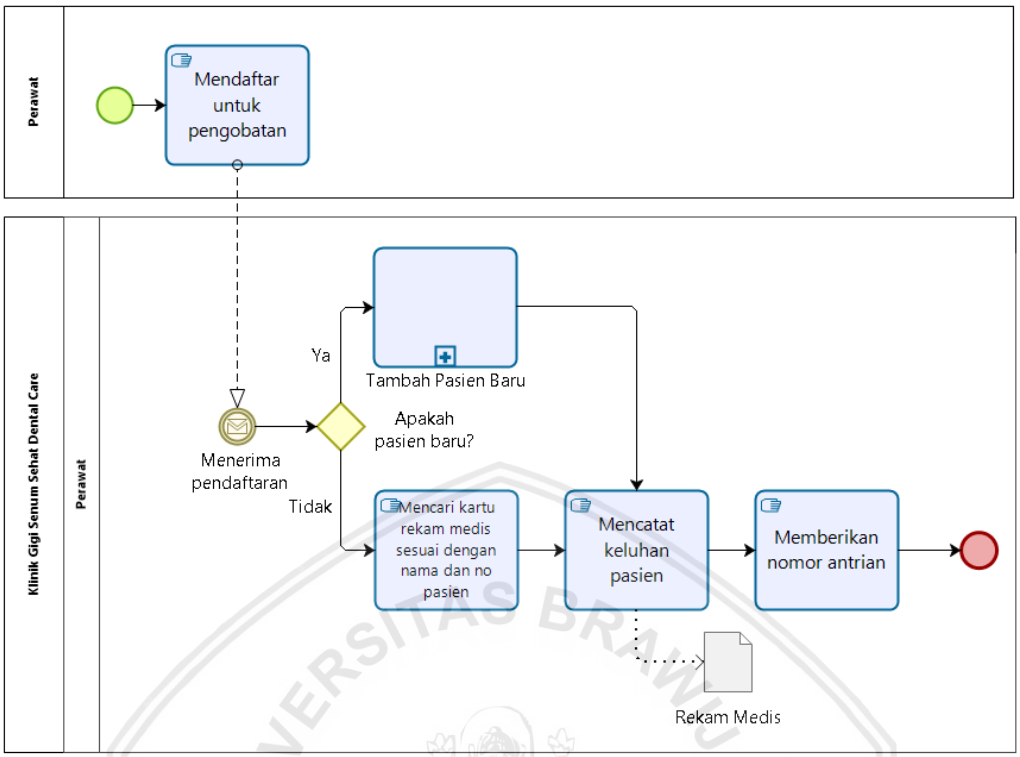
Elisitasi kebutuhan dilakukan dengan 2 pendekatan, yaitu obeservasi dan wawancara. Tahap wawancara dilakukan untuk mengetahui proses bisnis, masalah yang terjadi dan siapa saja yang terlibat. Sedangkan tahap observasi dilakukan dengan cara mengamati secara langsung proses bisnis yang ada. Dari proses elisitasi didapatkan prses bisnis saat ini pada Klinik Gigi Senyum Sehat Dental Care serta beberapa foto dokumen yang berkaitan dengan proses bisnis saat tersebut yang dijelaskan pada Lampiran A.

Pada proses bisnis saat ini, pasien melakukan pendaftaran dengan cara datang langsung ke klinik gigi untuk mendaftar perawatan di meja resepsionis. Pasien yang belum memiliki kartu berobat pasien (pasien baru), perawat akan membuatkan kartu berobat dan kartu rekam medis baru untuk pasien yang bersangkutan seperti yang digambarkan pada Gambar 4.1.



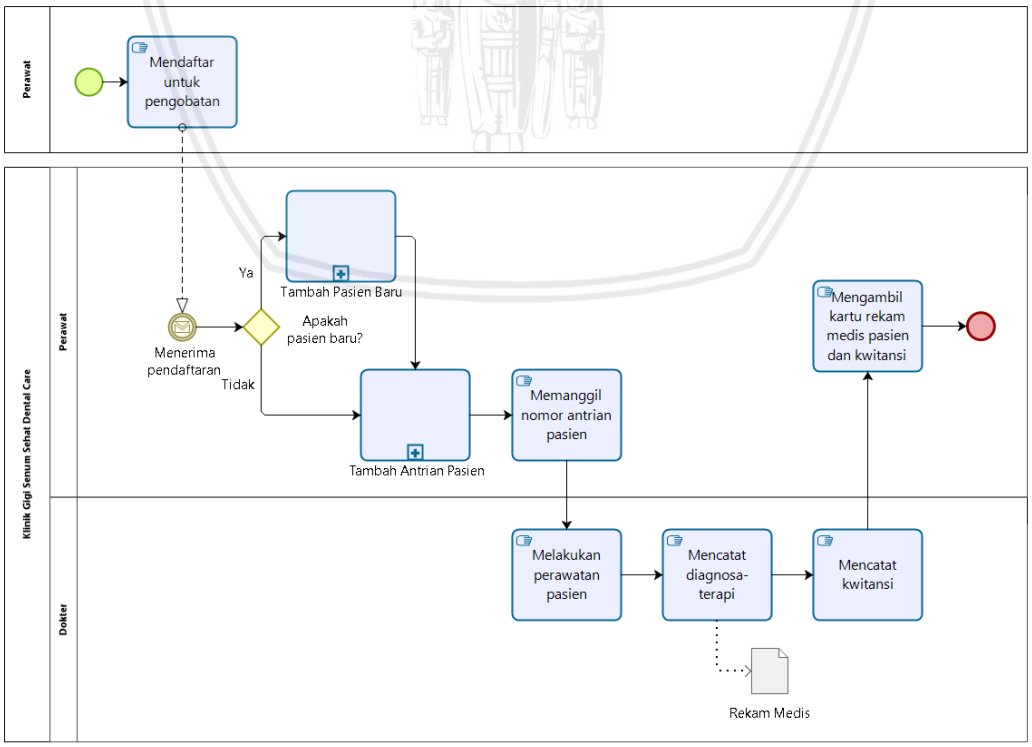
Gambar 4.1 Proses Bisinis *As Is* Tambah Pasien Baru

Selanjutnya, pasien akan diminta untuk menyebutkan keluhan yang dialaminya kemudian perawat akan mencatat keluhahan pasien tersebut pada kartu rekam medis. Perawat akan memberikan nomor antrian kepada pasien dan pasien diminta untuk menunggu hingga nomor antrian di panggil seperti yang digambarkan pada Gambar 4.2.



Gambar 4.2 Proses Bisinis *As Is* Tambah Antrian Pasien

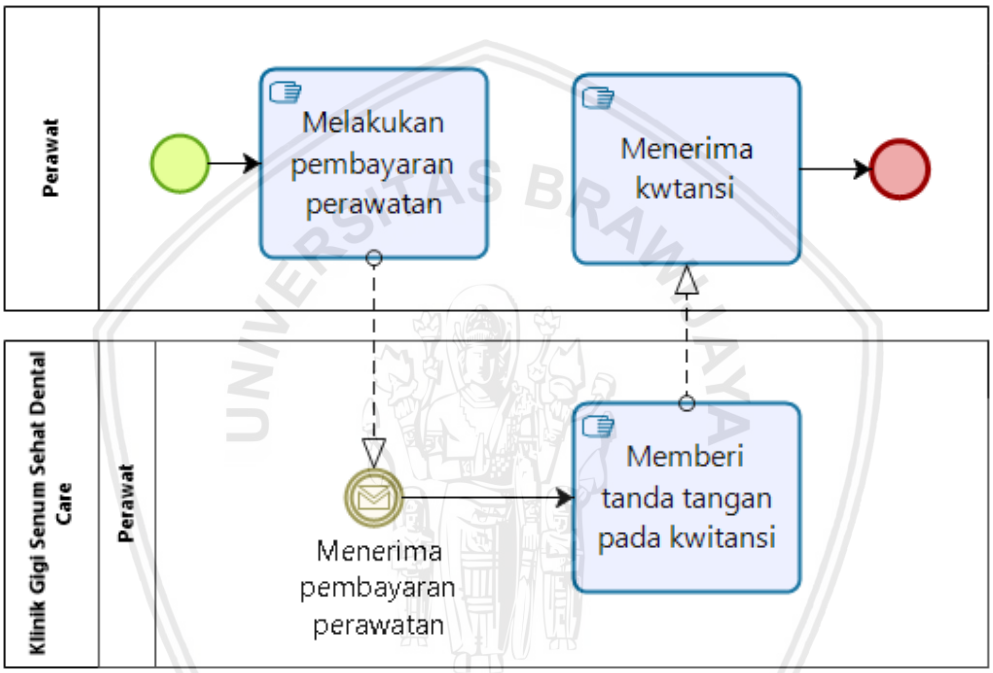
Perawat akan mengantarkan setiap kartu rekam medis yang berisi keterangan keluhan pasien dari meja resepsionis ke ruangan dokter.



Gambar 4.3 Proses Bisinis *As Is* Perawatan Pasien

Selanjutnya pasien akan di panggil sesuai dengan nomor antrian untuk melakukan perawatan pasien seperti yang digambarkan pada Gambar 4.3.

Setelah proses perawatan pasien selesai dilakukan, perawat akan mengambil kartu rekam medis beserta kwitansi perawatan dari ruangan dokter ke meja resepsionis, kemudian pasien akan menuju meja resepsionis untuk melakukan pembayaran. Perawat akan menerima pembayaran dan memberikan kwitansi kepada pasien seperti yang digambarkan pada Gambar 4.4.

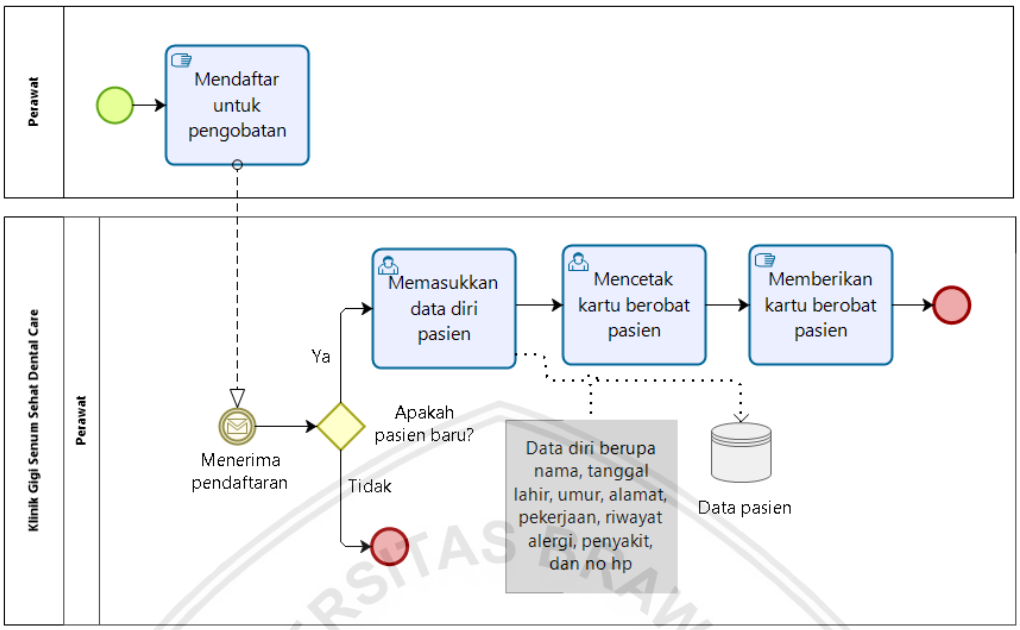


Gambar 4.4 Proses Bisinis *As Is* Pembayaran Perawatan

4.2 Analisis dan Spesifikasi Kebutuhan

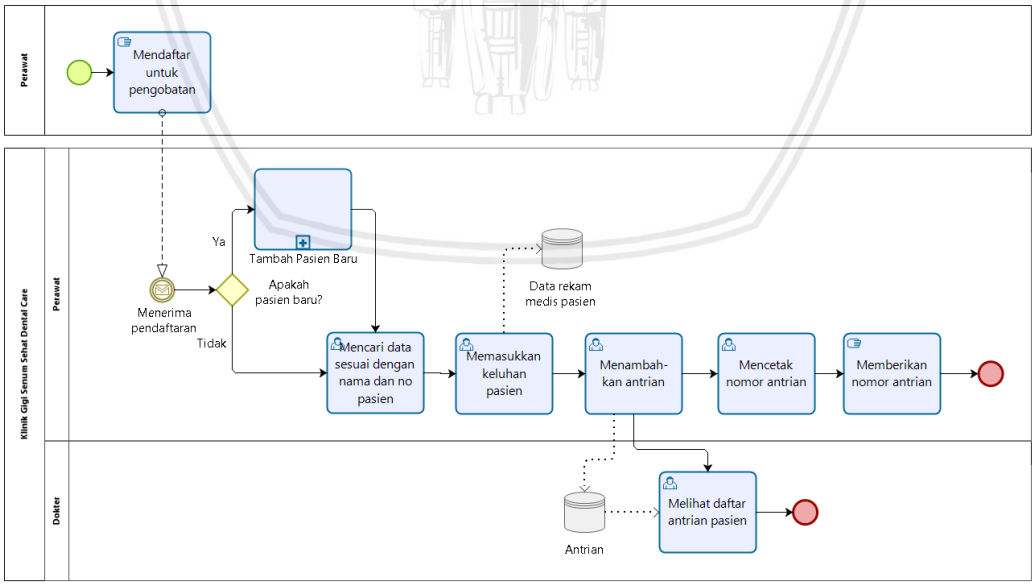
Setelah melakukan analisis permasalahan pada proses bisnis saat ini, maka didapatkan proses bisnis usulan. Proses bisnis usulan di buat dengan memodifikasi proses bisnis saat ini dengan menambahkan sistem informasi pengelolaan klinik gigi menggunakan prinsip *point of sale* yang dapat di gunakan oleh semua *stakeholder* pada Klinik Gigi Senyum Sehat Dental Care sesuai posisinya sebagai solusi pendukung untuk mengatasi permasalahan-permasalahan yang ada pada proses bisnis saat ini.

Proses bisnis usulan untuk melakukan tambah pasien baru digambarkan pada Gambar 4.5. Setelah menambahkan data pasien baru, perawat dapat mencetak kartu berobat pasien.



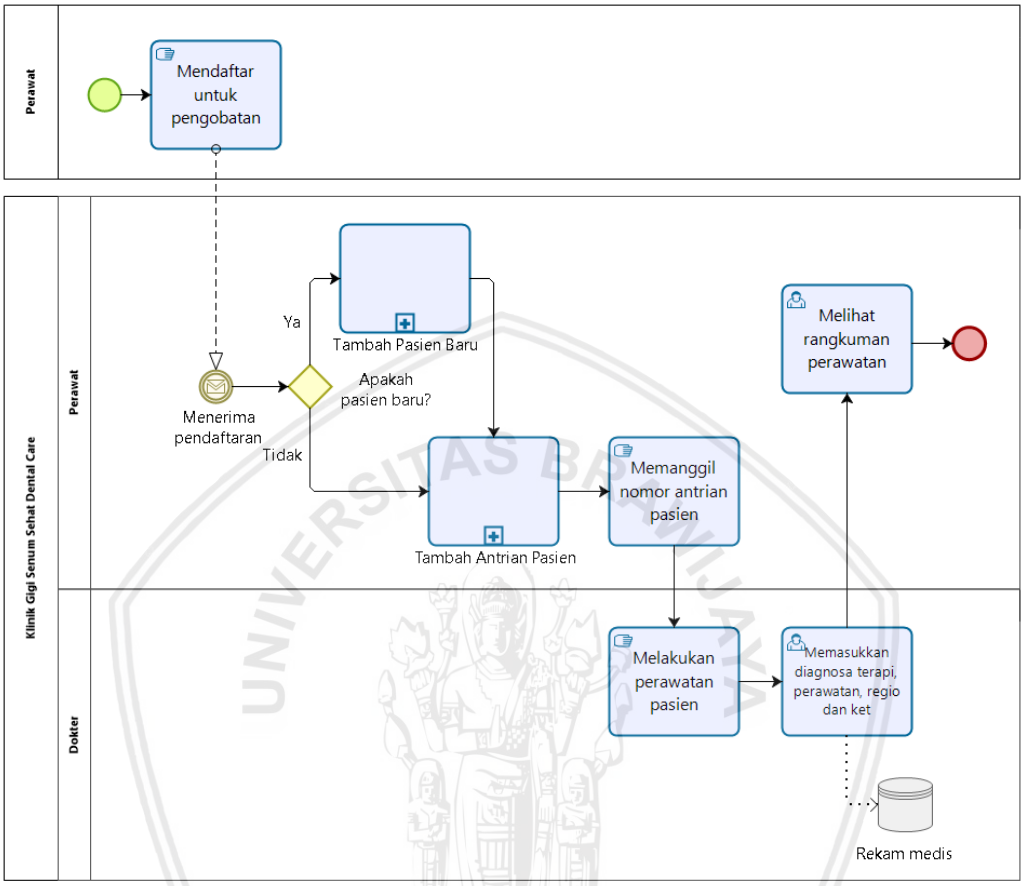
Gambar 4.5 Proses Bisinis *To Be* Tambah Pasien Baru

Proses bisnis usulan untuk melakukan tambah antrian pasien digambarkan pada Gambar 4.6. Data keluhan pasien akan masuk kedalam basis data dan kemudian perawat dapat mencetak nomor antrian pasien. Sedangkan dokter dapat melihat daftar antrian pasien.



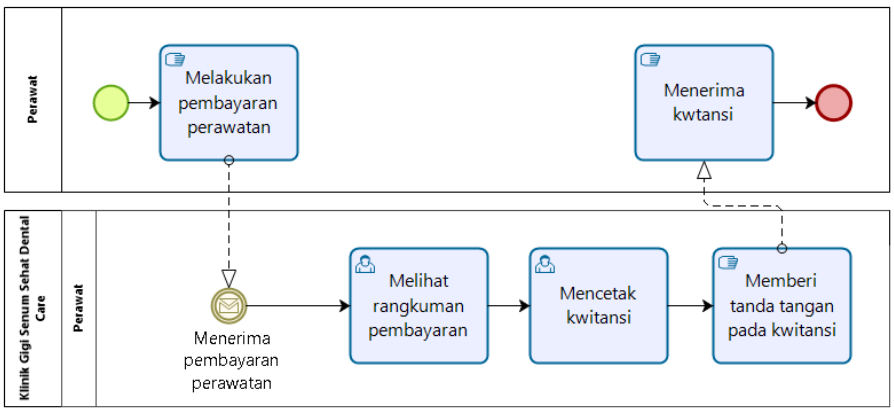
Gambar 4.6 Proses Bisinis *To Be* Tambah Antrian Pasien

Proses bisnis usulan untuk melakukan perawatan pasien digambarkan pada Gambar 4.7. Dokter dapat menambahkan diagnosa-terapi ke dalam basis data dan tidak perlu lagi menuliskan kwitansi.



Gambar 4.7 Proses Bisinis *To Be* Perawatan Pasien

Proses bisnis usulan untuk melakukan pembayaran perawatan digambarkan pada Gambar 4.8. Perawat dapat melihat rangkuman pembayaran dan kemudian mencetak kwitansi pembayaran.



Gambar 4.8 Proses Bisinis *To Be* Pembayaran Perawatan

4.2.1 Identifikasi Aktor

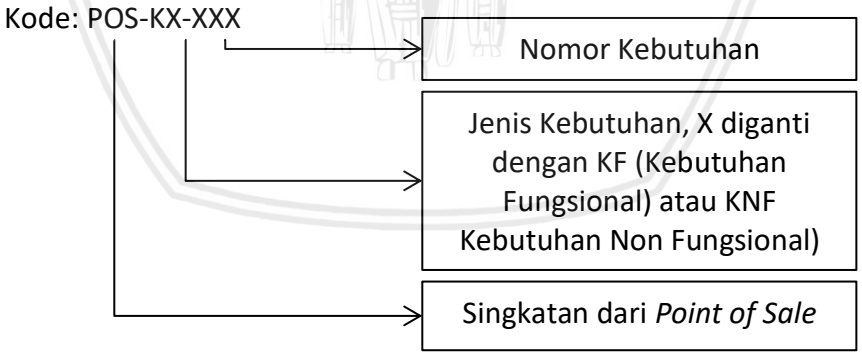
Tahap identifikasi aktor dilakukan untuk mengidentifikasi semua *stakeholder* yang berhubungan langsung dengan sistem agar *stakeholder* tersebut dapat berinteraksi baik dengan sistem. Dari hasil analisis, pengguna sistem terdiri dari 4 kategori, yaitu pengguna, dokter pemilik, dokter, dan perawat. Daftar aktor beserta deskripsinya ditunjukkan pada Tabel 4.1.

Tabel 4.1 Identifikasi Aktor

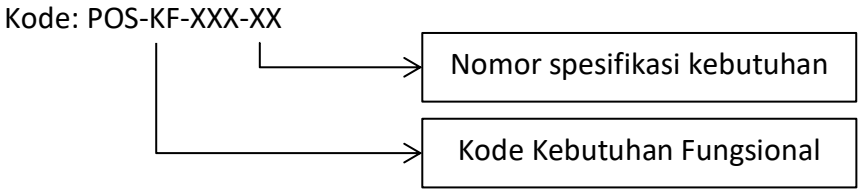
No	Aktor	Deskripsi
1	Pengguna	Aktor belum dapat melakukan akses terhadap sistem hanya sebatas pada tampilan *login*.
2	Dokter Pemilik	Aktor dapat mengelola proses perawatan pasien serta dapat melihat laporan berupa informasi seputar kegiatan klinik gigi.
3	Dokter	Aktor dapat mengelola proses perawatan pasien.
4	Perawat	Aktor dapat melakukan proses pelayanan pasien.

4.2.2 Aturan Penomoran Kode

Aturan penomoran kode dibuat untuk mempermudah penamaan kebutuhan dan spesifikasi kebutuhan. Aturan penomoran kode kebutuhan perangkat lunak yang digunakan dijelaskan pada Gambar 4.3 sedangkan aturan penomoran kode spesifikasi kebutuhan perangkat lunak dijelaskan pada Gambar 4.4.



Gambar 4.9 Aturan Penomoran Kebutuhan



Gambar 4.10 Aturan Penomoran Spesifikasi Kebutuhan

4.2.3 Analisis dan Spesifikasi Kebutuhan Fungsional Sistem

Kebutuhan fungsional sistem merupakan proses-proses yang dapat dilakukan oleh sistem. Dari hasil analisis pada penelitian ini didapatkan 29 kebutuhan fungsional yang berkaitan dengan antrian pasien, pasien, stok bahan, pendapatan, dan pengeluaran. Berikut daftar kebutuhan fungsional beserta spesifikasi kebutuhannya untuk semua aktor yang harus ada pada sistem dijelaskan dalam Tabel 4.2

Tabel 4.2 Kebutuhan Fungsional Sistem

Kode	Aktor	Deskripsi
POS-KF-001	Pengguna	*Login*.
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan halaman untuk *login* pengguna. (POS-KF-001-01) 2. Sistem harus menyediakan *form* untuk memasukkan *username* dan *password*. (POS-KF-001-02) 3. Sistem harus menyediakan tombol aksi *login* untuk masuk ke dalam sistem. (POS-KF-001-03)
POS-KF-002	Dokter pemilik, Dokter, Perawat	*Logout*.
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan tombol aksi untuk *logout* bagi aktor agar dapat keluar dari sistem. (POS-KF-002-01)
POS-KF-003	Dokter pemilik, Dokter, Perawat	Lihat Daftar Pasien
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan halaman untuk menampilkan tabel dengan 10 baris yang berisikan daftar pasien yang sudah tersimpan kepada aktor. (POS-KF-003-01) 2. Daftar pasien yang ditampilkan sistem berupa no pasien, nama, tanggal lahir, riwayat alergi, penyakit, dan no hp. (POS-KF-003-02) 3. Sistem harus menyediakan *field* pencarian agar dapat memudahkan aktor untuk mencari data pasien yang diinginkan. (POS-KF-003-03)
POS-KF-004	Dokter pemilik,	Lihat Daftar Antrian Pasien
		Spesifikasi Kebutuhan:

Kode	Aktor	Deskripsi
	Dokter, Perawat	1. Sistem harus menyediakan halaman untuk menampilkan daftar antrian pasien yang sudah tersimpan. (POS-KF-004-01) 2. Daftar antrian pasien yang ditampilkan sistem berupa, no antrian, no pasien, nama, dan dokter. (POS-KF-004-02) 3. Terdapat empat jenis daftar antrian yang ditampilkan kepada aktor perawat, yaitu menunggu, proses perawatan, pembayaran, dan selesai. (POS-KF-004-03) 4. Terdapat dua jenis daftar antrian yang ditampilkan kepada aktor dokter pemilik dan dokter, yaitu menunggu dan proses perawatan. (POS-KF-004-04)
POS-KF-005	Dokter pemilik, Dokter	Lihat Profil Pasien
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan tombol aksi lihat profil pasien kepada aktor. (POS-KF-005-01) 2. Sistem harus menyediakan halaman untuk menampilkan profil pasien yang sudah tersimpan kepada aktor. (POS-KF-005-02) 3. Profil pasien yang ditampilkan sistem berupa data pribadi dan daftar rekam medis pasien (POS-KF-005-03) 4. Data pribadi pasien berisikan no pasien, nama, tanggal lahir, umur, alamat, pekerjaan, riwayat alergi, penyakit dan no hp. (POS-KF-005-04) 5. Rekam medis pasien ditampilkan dalam bentuk tabel dengan 10 baris dan 3 kolom yang berisikan tanggal, keluhan, dan diagnosa. (POS-KF-005-05) 6. Sistem harus menyediakan *field* pencarian agar dapat memudahkan aktor untuk mencari data rekam medis yang diinginkan aktor. (POS-KF-005-06)
POS-KF-006	Dokter pemilik, Dokter	Selesai Perawatan Pasien
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan tombol aksi selesai perawatan kepada aktor. (POS-KF-006-01)

Kode	Aktor	Deskripsi
		2. Sistem harus menyedian *form* selesai perawatan kepada aktor. (POS-KF-006-02) 3. *Form* selesai perawatan harus memiliki *field* untuk mengisi diagnosa-terapi, pilihan jenis perawatan, regio, dan keterangan. (POS-KF-006-03) 4. Sistem harus menyediakan tombol simpan pada *form* agar masukan yang sudah diisi oleh aktor dapat tersimpan. (POS-KF-006-04)
POS-KF-007	Dokter pemilik, Dokter	Lihat Stok Bahan
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan halaman untuk menampilkan tabel dengan 10 baris yang berisikan daftar stok bahan yang sudah tersimpan kepada aktor. (POS-KF-007-01) 2. Daftar stok bahan yang ditampilkan sistem kepada aktor dokter pemilik berupa kode, nama, stok, satuan, status, harga pokok, harga jual, dan *expired*. (POS-KF-007-02) 3. Daftar stok bahan yang ditampilkan sistem kepada aktor dokter berupa kode, nama, stok, satuan, status, harga jual, dan *expired*. (POS-KF-007-03) 4. Stok bahan yang sudah *expired* akan terhapus secara otomatis. (POS-KF-007-04) 5. Status pada daftar stok bahan merupakan status ketersediaan stok, terdapat 3 jenis status, yaitu tersedia, sedikit, dan habis. (POS-KF-007-05) 6. Sistem harus menyediakan *field* pencarian agar dapat memudahkan aktor untuk mencari data bahan yang diinginkan aktor. (POS-KF-007-06)
POS-KF-008	Dokter pemilik, Dokter	Lihat Pendapatan Pribadi
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan halaman untuk menampilkan tabel dengan 10 baris yang berisikan daftar pendapatan pribadi yang sudah tersimpan. (POS-KF-008-01)

Kode	Aktor	Deskripsi
		2. Daftar pendapatan pribadi yang ditampilkan sistem berupa no faktur, tanggal, perawatan, regio, keterangan, dan total. (POS-KF-008-02) 3. Daftar pendapatan pribadi ditampilkan sesuai dengan aktor serta bulan dan tahun. (POS-KF-008-03) 4. Sistem harus menyediakan *form* dropdown bulan dan tahun serta tombol cari untuk melakukan *filter* data pendapatan pribadi sesuai dengan bulan dan tahun yang diinginkan aktor. (POS-KF-008-04) 5. Sistem harus menyediakan *field* pencarian agar dapat memudahkan aktor untuk mencari pendapatan pribadi yang diinginkan aktor. (POS-KF-008-05)
POS-KF-009	Dokter pemilik, Perawat	Lihat Pengeluaran Bahan
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan halaman untuk menampilkan tabel dengan 10 baris yang berisikan daftar pengeluaran bahan yang sudah tersimpan kepada aktor. (POS-KF-009-01) 2. Daftar pengeluaran bahan yang ditampilkan sistem berupa no, tanggal, kode, nama, jumlah, harga pokok, dan total. (POS-KF-009-02) 3. Sistem harus menyediakan *form* dropdown bulan dan tahun serta tombol cari untuk melakukan *filter* data pengeluaran bahan sesuai dengan bulan dan tahun yang diinginkan aktor. (POS-KF-009-04) 4. Sistem harus menyediakan *field* pencarian agar dapat memudahkan aktor untuk mencari data pengeluaran bahan yang diinginkan aktor. (POS-KF-009-05)
POS-KF-010	Dokter pemilik, Perawat	Tambah Stok Bahan
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan tombol aksi tambah stok bahan kepada aktor. (POS-KF-010-01)

Kode	Aktor	Deskripsi
		2. SIstem harus menyediakan *form* isian untuk melakukan tambah stok bahan kepada aktor. (POS-KF-010-02) 3. *Form* untuk menambahkan stok bahan harus memiliki *field* untuk mengisi pilihan nama bahan, jumlah, dan tanggal. Sedangkan *filed* satuan, harga pokok, dan total akan terisi otomatis sesuai *field* nama bahan yang dipilih. (POS-KF-010-03) 4. Sistem harus menyediakan tombol simpan pada *form* tambah stok bahan agar masukan yang sudah diisi oleh aktor dapat tersimpan. (POS-KF-010-04)
POS-KF-011	Dokter pemilik, Perawat	Tambah Bahan Baru
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan tombol aksi tambah bahan baru kepada aktor. (POS-KF-011-01) 2. SIstem harus menyediakan *form* isian untuk melakukan tambah bahan baru kepada aktor. (POS-KF-011-02) 3. *Form* untuk menambahkan bahan baru harus memiliki *field* untuk mengisi nama bahan, stok minimal, satuan, perawatan, harga pokok, harga jual, *expired*, jumlah, dan tanggal. Sedangkan *filed* total akan terisi otomatis sesuai *field* jumlah dan harga pokok yang telah diisi. (POS-KF-011-03) 4. Sistem harus menyediakan tombol simpan pada *form* tambah stok bahan agar masukan yang sudah diisi oleh aktor dapat tersimpan. (POS-KF-011-04)
POS-KF-012	Dokter pemilik, Perawat	Hapus Pengeluaran Bahan
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan tombol hapus pengeluaran bahan kepada aktor untuk menghapus pengeluaran bahan yang diinginkan. (POS-KF-012-01)

Kode	Aktor	Deskripsi
		2. Sistem harus menampilkan modal konfirmasi untuk menghapus data pengeluaran bahan yang diinginkan. (POS-KF-012-02)
POS-KF-013	Dokter, Perawat	Lihat Gaji
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan halaman untuk menampilkan tabel dengan 10 baris yang berisikan data gaji yang sudah tersimpan kepada aktor. (POS-KF-013-01) 2. Daftar gaji yang ditampilkan sistem berupa tanggal, gaji, dan status. (POS-KF-013-02) 3. Sistem harus menyediakan *form* dropdown bulan dan tahun untuk melakukan *filter* data gaji sesuai dengan bulan dan tahun yang diinginkan aktor. (POS-KF-013-03)
POS-KF-014	Dokter pemilik	*Edit* Stok Bahan
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan tombol *edit* stok bahan kepada aktor untuk mengubah stok bahan yang diinginkan. (POS-KF-014-01) 2. Sistem harus menyediakan *form* isian untuk melakukan *edit* stok bahan yang telah tersimpan kedapa aktor. (POS-KF-014-02) 3. *Form* untuk *edit* stok bahan harus memiliki *field* untuk mengisi nama bahan, stok minimal, satuan, harga pokok, harga jual, perawatan, dan *expired* (POS-KF-014-03) 4. Sistem harus menyediakan tombol *edit* pada *form edit* stok bahan agar masukan yang telah diisi aktor dapat tersimpan. (POS-KF-014-04)
POS-KF-015	Dokter pemilik	Hapus Stok Bahan
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan tombol hapus stok bahan kepada aktor untuk menghapus stok bahan yang diinginkan. (POS-KF-015-01) 2. Sistem harus menampilkan modal konfirmasi untuk menghapus data stok bahan yang diinginkan. (POS-KF-015-02)

Kode	Aktor	Deskripsi
POS-KF-016	Dokter pemilik	Lihat Pendapatan Klinik
		Spesifikasi Kebutuhan: 1. Pendapatan klinik yang ditampilkan sistem berupa data pendapatan kotor dan dan pendapatan bersih (POS-KF-016-01) 2. Sistem harus menyediakan halaman untuk menampilkan 2 tabel dengan 10 baris yang berisikan daftar pendapatan kotor dan pendapatan bersih yang sudah tersimpan kepada aktor. (POS-KF-016-02) 3. Daftar pendapatan kotor yang ditampilkan sistem berupa no faktur, tanggal, perawatan, regio, keterangan, dan total. (POS-KF-016-03) 4. Daftar pendapatan bersih yang ditampilkan sistem berupa tanggal, pendapatan kotor, pengeluaran, pendapatan bersih. (POS-KF-016-04) 5. Sistem harus menyediakan *form* dropdown bulan dan tahun untuk melakukan *filter* data pendapatan kotor dan pendapatan bersih sesuai dengan bulan dan tahun yang diinginkan aktor. (POS-KF-016-05) 6. Sistem harus menyediakan *field* pencarian agar dapat memudahkan aktor untuk mencari data pendapatan kotor yang diinginkan aktor. (POS-KF-016-06)
POS-KF-017	Dokter pemilik	Lihat Gaji Pegawai
		Spesifikasi Kebutuhan: 1. Gaji pegawai yang ditampilkan sistem berupa data gaji dokter dan dan gaji perawat (POS-KF-017-01) 2. Sistem harus menyediakan halaman untuk menampilkan 2 tabel dengan 10 baris yang berisikan daftar gaji dokter dan gaji perawat yang sudah tersimpan kepada aktor. (POS-KF-017-02) 3. Daftar gaji dokter dan gaji perawat yang ditampilkan sistem berupa nama dokter/nama

Kode	Aktor	Deskripsi
		perawat, tanggal, gaji, dan status. (POS-KF-017-03) 4. Status gaji "belum dibayarkan" ditampilkan berupa tombol untuk merubah status gaji tersebut. (POS-KF-017-04) 5. Sistem harus menyediakan *form* dropdown bulan dan tahun untuk melakukan *filter* data gaji sesuai dengan bulan dan tahun yang diinginkan aktor. (POS-KF-017-05)
POS-KF-018	Dokter pemilik	Ubah Status Gaji Pegawai
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan tombol aksi untuk mengubah status gaji kepada aktor. (POS-KF-018-01) 2. Ketika tombol "belum dibayarkan" di klik maka status gaji akan berubah menjadi "sudah dibayarkan". (POS-KF-018-02)
POS-KF-019	Perawat	Tambah Pasien Baru
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan tombol aksi untuk tambah pasien baru kepada aktor. (POS-KF-019-01) 2. SIstem harus menyediakan *form* untuk melakukan tambah pasien baru kepada aktor. (POS-KF-019-02) 3. *Form* untuk menambahkan pasien baru harus memiliki *field* untuk mengisi nama, tanggal lahir, alamat, pekerjaan, riwayat alergi, penyakit, dan no hp. (POS-KF-019-03) 4. Sistem harus menyediakan tombol tambah pada *form* agar masukan yang sudah diisi oleh aktor dapat tersimpan. (POS-KF-019-04)
POS-KF-020	Perawat	Tambah Antrian Pasien
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan tombol aksi tambah antrian pasien kepada aktor. (POS-KF-020-01) 2. Sistem harus menyedian *form* untuk tambah antrian kepada aktor. (POS-KF-020-02)

Kode	Aktor	Deskripsi
		3. *Form* untuk menambahkan antrian pasien harus memiliki *field* untuk mengisi keluhan pasien dan pilihan dokter. (POS-KF-020-03) 4. Sistem harus menyediakan tombol simpan pada *form* agar masukan yang sudah diisi oleh aktor dapat tersimpan. (POS-KF-020-04)
POS-KF-021	Perawat	Cetak Kartu Berobat Pasien
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan tombol aksi cetak kartu berobat pasien kepada aktor. (POS-KF-021-01) 2. Kartu berobat pasien ditampilkan dengan *form*at pdf. (POS-KF-021-02)
POS-KF-022	Perawat	Cetak Nomor Antrian Pasien
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan tombol aksi cetak nomor antrian pasien kepada aktor. (POS-KF-022-01) 2. Nomor antrian pasien ditampilkan dengan *form*at pdf. (POS-KF-022-02)
POS-KF-023	Perawat	Pembayaran Perawatan Pasien
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan tombol aksi pembayaran kepada aktor. (POS-KF-023-01) 2. Pembayaran yang ditampilkan sistem berupa rangkuman perawatan, daftar bahan dan dan perawatan (POS-KF-023-02) 3. Sistem harus menyediakan halaman untuk menampilkan rangkuman perawatan untuk pasien yang telah selesai melakukan perawatan kepada aktor. (POS-KF-023-03) 4. Rangkuman perawatan menampilkan nama pasien, perawatan, regio, dan keterangan. (POS-KF-023-04) 5. Sistem harus menyediakan halaman untuk menampilkan tabel dengan 10 baris yang berisikan daftar bahan yang sudah tersimpan kepada aktor. (POS-KF-023-05)

Kode	Aktor	Deskripsi
		6. Daftar bahan yang ditampilkan sistem kepada aktor berupa kode, nama bahan, perawatan, harga jual, dan tombol untuk *add to cart*. (POS-KF-023-06) 7. Sistem harus menyediakan *field* pencarian agar dapat memudahkan aktor untuk mencari data bahan yang diinginkan aktor. (POS-KF-023-07) 8. Sistem harus menyediakan halaman untuk menampilkan tabel yang berisikan data pembayaran perawatan. (POS-KF-023-08) 9. Data pembayaran perawatan yang ditampilkan sistem kepada aktor berupa nama bahan, harga, jumlah, subtotal, total, biaya perawatan, total akhir, dan *field* untuk mengisi jumlah uang yang diberikan pasien serta kembali yang otomatis akan terisi sesuai dengan total akhir dan *field* bayar. (POS-KF-023-09)
POS-KF-024	Perawat	Lihat Rangkuman Pembayaran Perawatan Pasien
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan halaman untuk menampilkan rangkuman pembayaran untuk pasien yang telah selesai melakukan perawatan kepada aktor. (POS-KF-024-01) 2. Rangkuman pembayaran menampilkan no faktur, no pasien, nama, alamat, umur, bahan, jenis perawatan, regio, dan total. (POS-KF-024-02)
POS-KF-025	Perawat	Cetak Kwitansi Perawatan
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan tombol aksi cetak kwitansi perawatan kepada aktor. (POS-KF-025-01) 2. Kwitansi perawatan ditampilkan dengan *form*at pdf. (POS-KF-025-02)
POS-KF-026	Perawat	Cetak Rekam Medis
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan tombol aksi cetak rekam medis kepada aktor. (POS-KF-026-01)

Kode	Aktor	Deskripsi
		2. Rekam medis ditampilkan dengan *form*at pdf. (POS-KF-026-02)
POS-KF-027	Perawat	Lihat Daftar Harga Bahan
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan halaman untuk menampilkan tabel dengan 10 baris yang berisikan daftar harga bahan yang sudah tersimpan kepada aktor. (POS-KF-027-01) 2. Daftar harga bahan yang ditampilkan sistem kepada aktor berupa kode, nama, stok, satuan, status, harga jual, dan *expired*. (POS-KF-027-02) 3. Status pada daftar stok bahan merupakan status ketersediaan stok, terdapat 3 jenis status, yaitu tersedia, sedikit, dan habis. (POS-KF-027-03) 4. Sistem harus menyediakan *field* pencarian agar dapat memudahkan aktor untuk mencari daftar harga bahan yang diinginkan aktor. (POS-KF-027-04)
POS-KF-028	Perawat	Lihat Daftar Biaya Perawatan
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan halaman untuk menampilkan tabel dengan 10 baris yang berisikan daftar biaya perawatan yang sudah tersimpan kepada aktor. (POS-KF-028-01) 2. Daftar biaya perawatan yang ditampilkan sistem kepada aktor berupa no, jenis perawatan, dan biaya. (POS-KF-028-02) 3. Sistem harus menyediakan *field* pencarian agar dapat memudahkan aktor untuk mencari daftar biaya perawatan yang diinginkan aktor. (POS-KF-028-03)
POS-KF-029	Perawat	Atur Ulang Antrian Pasien.
		Spesifikasi Kebutuhan: 1. Sistem harus menyediakan tombol *reset* kepada aktor untuk mengatur ulang antrian pasien. (POS-KF-029-01)

4.2.4 Analisis Kebutuhan Non Fungsional Sistem

Kebutuhan non fungsional bertujuan untuk meningkatkan kualitas dari sistem yang dikembangkan. Kebutuhan ini menentukan bagaimana sistem berjalan semestinya. Pada penelitian ini terdapat 1 kebutuhan non fungsional yaitu, *performance* yang dijelaskan pada Tabel 4.3.

Tabel 4.3 Kebutuhan Non-Fungsional Sistem

Kode Fungsi	Kebutuhan Non Fungsional	Deskripsi
POS-KNF-001	*Performance*	Sistem dapat memberikan load time kurang dari 3 detik.

4.3 Pemodelan Kebutuhan

Tujuan dari tahap pemodelan kebutuhan adalah untuk menjelaskan apa yang dibutuhkan oleh pengguna, sebagai dasar perancangan sistem, dan menjadi referensi dalam melakukan validasi kebutuhan.

4.3.1 Use Case Diagram

Use case diagram dibuat untuk menggambarkan kelakuan sistem yang akan di bangun tampak dari luar. *Use case diagram* pada penelitian ini memiliki 4 aktor yaitu pengguna, dokter pemilik, dokter dan perawat, serta 29 *use case*. 29 *use case* tersebut mecakup perilaku sistem tampak dari luar yang berhubungan dengan antrian pasien, pasien, stok bahan, pendapatan, dan pengeluaran. Pemodelan *use case diagram* digambarkan pada Gambar 4.5.

4.3.2 Use Case Scenario

Use case scenario bertujuan agar gambaran kebutuhan fungsional dapat dijelaskan dengan lebih detail serta menjelaskan bagaimana sistem yang dibangun dapat berinteraksi dengan aktor. *Use case scenario* akan ditampilkan dalam bentuk tabel-tabel. Tabel 4.4 menampilkan *use case scenario login*, *login* merupakan aktivitas yang dapat memungkinkan aktor untuk masuk ke dalam sistem serta dapat menggunakan sistem.

Tabel 4.4 *Use Case Scenario Login*

POS-KF-001	*Login*
Objective	Memungkinkan aktor untuk masuk kedalam sistem dan menggunakan sistem
Actor	Pengguna
Pre-condition	Aktor mengakses sistem
Main flow	1. Sistem menampilkan *form login*

	2. Aktor memasukkan *username* dan *password* 3. Aktor menekan tombol *login* 4. Sistem kemudian melakukan proses *login* dengan melakukan pengecekan *username* dan *password* yang telah dimasukkan oleh aktor berdasarkan hak aksesnya 5. Aktor akan masuk ke halaman utama
Alternative flow	A.3 Pada saat aktor melakukan *login* ada *field* yang belum diisi maka sistem memberi peringatan untuk mengisi *field* yang belum terisi. A.4 Saat aktor *login*, *username* dan *password* yang dimasukkan tidak terdaftar saat di lakukan verifikasi *username* dan *password* maka sistem akan menampilkan pesan kesalahan bahwa "**Gagal *Login*!** Username/ Password yang anda masukkan salah."
Post-condition	*Login* berhasil dilakukan dan sistem akan menampilkan halaman utama aktor sesuai akses.

Tabel 4.5 menampilkan *use case scenario logout*, *logout* merupakan aktivitas yang dapat memungkinkan aktor untuk keluar dari sistem, namun proses *logout* hanya bisa dilakukan oleh aktor yang sudah masuk ke dalam sistem.

Tabel 4.5 *Use Case Scenario Logout*

POS-KF-002	*Logout*
Objective	Memungkinkan aktor untuk keluar dari sistem
Actor	Dokter pemilik, dokter, perawat
Pre-condition	Aktor sudah masuk kedalam sistem
Main flow	1. Aktor pilih menu *logout* 2. Sistem akan menampilkan konfirmasi *logout* kepada aktor 3. Aktor menekan tombol *logout* 4. *Logout* berhasil dan sistem akan menampilkan halaman *login*
Alternative flows	A.3 Aktor menekan tombol cancel maka aktor batal keluar dari sistem.
Post-condition	*Logout* berhasil, aktor berhasil keluar dari sistem dan sistem akan menampilkan halaman *login*.

Tabel 4.6 menampilkan *use case scenario* lihat daftar pasien, lihat daftar pasien merupakan aktivitas yang dapat memungkinkan aktor untuk melihat daftar pasien yang sudah tersimpan di dalam sistem.

Tabel 4.6 *Use Case Scenario* Lihat Daftar Pasien

POS-KF-003	Lihat daftar pasien
Objective	Memungkinkan aktor untuk melihat daftar pasien
Actor	Dokter pemilik, dokter, perawat
Pre-condition	Aktor sudah melakukan *login*
Main flow	1. Aktor memilih menu Pasien 2. Sistem menjalankan proses lihat pasien dan menampilkan daftar berupa tabel dengan 10 baris yang berisikan no pasien, nama, tanggal lahir, riwayat alergi, penyakit, dan no hp
Post-condition	Aktor dapat melihat daftar pasien.

Tabel 4.7 menampilkan *use case scenario* lihat daftar antrian pasien, lihat daftar antrian pasien merupakan aktivitas yang dapat memungkinkan aktor untuk malihat daftar antrian pasien.

Tabel 4.7 *Use Case Scenario* Lihat Daftar Antrian Pasien

POS-KF-004	Lihat daftar antrian pasien
Objective	Memungkinkan aktor untuk melihat daftar antrian pasien
Actor	Dokter pemilik, dokter, perawat
Pre-condition	Aktor sudah melakukan *login*
Main flow	1. Aktor memilih menu Antrian 2. Sistem menjalankan proses lihat antrian pasien dan menampilkan antrian pasien berupa no antrian, no pasien, nama, dan dokter 3. Untuk aktor perawat antrian pasien ditampilkan dalam empat jenis dengan warna berbeda, yaitu menunggu dengan warna merah, proses perawatan dengan warna kuning, pembayaran dengan warna hijau dan selesai dittampilkan dalam sebuah tabel 4. Untuk aktor dokter pemilik dan dokter antrian pasien ditampilkan dalam dua jenis dengan warna berbeda, yaitu menunggu dengan warna merah dan proses perawatan dengan warna kuning
Post-condition	Aktor dapat melihat daftar antrian pasien.

Tabel 4.8 menampilkan *use case scenario* lihat profil pasien, lihat profil pasien merupakan aktivitas yang dapat memungkinkan aktor untuk malihat informasi-informasi pasien yang terdapat pada profil pasien.

Tabel 4.8 *Use Case Scenario* Lihat Profil Pasien

POS-KF-005	Lihat profil pasien
Objective	Memungkinkan aktor untuk melihat profil pasien
Actor	Dokter pemilik, dokter
Pre-condition	Aktor berada di halaman menu Pasien
Main flow	1. Aktor menekan tombol lihat pada daftar pasien yang diinginkan 2. Sistem menjalankan proses lihat profil pasien 3. Profil pasien yang ditampilkan sistem berupa data pribadi dan daftar rekam medis pasien 4. Data pribadi pasien berisikan no, nama, tanggal lahir, umur, alamat, pekerjaan, riwayat alergi, penyakit dan no hp 5. Rekam medis pasien ditampilkan dalam bentuk tabel dengan 10 baris dan 3 kolom yang berisikan tanggal, keluhan, dan diagnosa 6. Sistem menampilkan *field* pencarian agar dapat memudahkan aktor untuk mencari data rekam medis yang diinginkan
Post-condition	Aktor dapat melihat profil pasien yang diinginkan

Tabel 4.9 menampilkan *use case scenario* selesai perawatan pasien, selesai perawatan pasien merupakan aktivitas yang dapat memungkinkan aktor untuk menambahkan diagnosa-terapi, perawatan, regio, dan keterangan dari pasien yang melakukan perawatan.

Tabel 4.9 *Use Case Scenario* Selesai Perawatan Pasien

POS-KF-006	Selesai perawatan pasien
Objective	Memungkinkan aktor untuk menambahkan data setelah selesai melakukan perawatan pasien
Actor	Dokter pemilik, dokter
Pre-condition	Aktor berada pada halaman profil pasien
Main flow	1. Aktor menenekan tombol selesai 2. Sistem menampilkan halaman *form* untuk menambahkan diagnosa-terapi, pilihan perawatan, regio, dan keterangan 3. Aktor memasukkan data sesuai dengan *field* yang tersedia 4. Aktor menekan tombol simpan

	5. Sistem melakukan pengecekan isi *form*, jika berhasil maka sistem menyimpan masukan aktor
Alternative flows	A.5 Pada saat aktor menambah data selesai perawatan dan ada *field* yang belum diisi maka sistem memberi peringatan untuk mengisi *field* yang belum terisi.
Post-condition	Data selesai perawatan pasien berhasil di simpan dan sistem menampilkan halaman antrian pasien.

Tabel 4.10 menampilkan *use case scenario* lihat stok bahan, lihat stok bahan merupakan aktivitas yang dapat memungkinkan aktor untuk melihat daftar stok bahan yang sudah tersimpan dalam sistem.

Tabel 4.10 *Use Case Scenario* Lihat Stok Bahan

POS-KF-007	Lihat stok bahan
Objective	Memungkinkan aktor untuk melihat daftar stok bahan
Actor	Dokter pemilik, dokter
Pre-condition	Aktor sudah melakukan *login*
Main flow	1. Aktor memilih menu Bahan 2. Sistem menjalankan proses lihat stok bahan dan menampilkan daftar berupa tabel dengan 10 baris 3. Untuk aktor dokter pemilik, daftar stok bahan yang ditampilkan sistem berupa kode, nama, stok, satuan, status, harga pokok, harga jual, dan *expired* 4. Untuk aktor dokter, daftar stok bahan yang ditampilkan sistem berupa kode, nama, stok, satuan, status, harga jual, dan *expired* 5. Setiap bahan yang sudah *expired* akan otomatis dihapus dari daftar stok bahan 6. Status pada daftar stok bahan merupakan status ketersediaan stok, terdapat 3 jenis status, yaitu tersedia, sedikit, dan habis 7. Sistem menampilkan *field* pencarian agar dapat memudahkan aktor untuk mencari data bahan yang diinginkan
Post-condition	Aktor dapat melihat daftar stok bahan.

Tabel 4.11 menampilkan *use case scenario* lihat pendapatan pribadi, lihat pendapatan pribadi merupakan aktivitas yang dapat memungkinkan aktor untuk melihat daftar pendapatan pribadi.

Tabel 4.11 *Use Case Scenario* Lihat Pendapatan Pribadi

POS-KF-008	Lihat perndapatan pribadi
Objective	Memungkinkan aktor untuk melihat pendapatan pribadi
Actor	Dokter pemilik, dokter
Pre-condition	Aktor sudah melakukan *login*
Main flow	1. Aktor memilih menu Pendapatan Pribadi 2. Sistem menjalankan proses lihat pendapatan pribadi dan menampilkan daftar berupa tabel dengan 10 baris yang berisikan no faktur, tanggal, perawatan, regio, dan total 3. Daftar pendapatan pribadi ditampilkan sesuai dengan aktor serta bulan dan tahun 4. Sistem menampilkan *form* dropdown bulan dan tahun serta tombol cari untuk melakukan *filter* data pendapatan pribadi sesuai dengan bulan dan tahun yang diinginkan aktor 5. Sistem menampilkan *field* pencarian agar dapat memudahkan aktor untuk mencari data pendapattan pribadi yang diinginkan
Post-condition	Aktor dapat melihat daftar pendapatan pribadi.

Tabel 4.12 menampilkan *use case scenario* lihat pengeluaran bahan, lihat pengeluaran bahan merupakan aktivitas yang dapat memungkinkan aktor untuk melihat daftar pengeluaran klinik.

Tabel 4.12 *Use Case Scenario* Lihat Pengeluaran Bahan

POS-KF-009	Lihat pengeluaran bahan
Objective	Memungkinkan aktor untuk melihat pengeluaran bahan
Actor	Dokter pemilik, perawat
Pre-condition	Aktor sudah melakukan *login*
Main flow	1. Aktor memilih menu Pengeluaran Bahan 2. Sistem menjalankan proses lihat pendapatan pribadi dan menampilkan daftar berupa tabel dengan 10 baris yang berisikan no, tanggal, kode, nama, jumlah, harga pokok, dan total 3. Sistem menampilkan *form* dropdown bulan dan tahun serta tombol cari untuk melakukan *filter* data pengeluaran bahan sesuai dengan bulan dan tahun yang diinginkan aktor

	4. Sistem menampilkan *field* pencarian agar dapat memudahkan aktor untuk mencari data pengeluaran bahan yang diinginkan
Post-condition	Aktor dapat melihat daftar pengeluaran bahan.

Tabel 4.13 menampilkan *use case scenario* tambah stok bahan, tambah stok bahan merupakan aktivitas yang dapat memungkinkan aktor untuk menambahkan pengeluaran klinik ke dalam sistem.

Tabel 4.13 *Use Case Scenario* Tambah Stok Bahan

POS-KF-010	Tambah stok bahan
Objective	Memungkinkan aktor untuk menambahkan stok bahan
Actor	Dokter pemilik, perawat
Pre-condition	Aktor berada di halaman menu Pengeluaran Bahan
Main flow	1. Aktor menekan tombol tambah stok bahan 2. Sistem menampilkan halaman *form* untuk memasukkan data stok bahan 3. Aktor memasukkan data berupa berupa pilihan nama bahan, jumlah, dan tanggal 4. Aktor menekan tombol simpan 5. Sistem melakukan pengecekan isi *form*, jika berhasil maka sistem menyimpan masukan aktor
Alternative flows	A.5 Pada saat aktor menambah stok bahan dan ada *field* yang belum diisi maka sistem memberi peringatan untuk mengisi *field* yang belum terisi.
Post-condition	Data stok bahan berhasil di simpan dan sistem menampilkan daftar pengeluaran bahan.

Tabel 4.14 menampilkan *use case scenario* tambah bahan baru, tambah bahan baru merupakan aktivitas yang dapat memungkinkan aktor untuk menambahkan data stok bahan baru dan pengeluaran ke dalam sistem.

Tabel 4.14 *Use Case Scenario* Tambah Bahan Baru

POS-KF-011	Tambah bahan baru
Objective	Memungkinkan aktor untuk menambahkan bahan baru
Actor	Dokter pemilik, perawat
Pre-condition	Aktor berada di halaman menu Pengeluaran Bahan
Main flow	1. Aktor menekan tombol tambah bahan baru 2. Sistem menampilkan halaman *form* untuk memasukkan data bahan baru

	3. Aktor memasukkan data berupa nama bahan, stok minimal, satuan, perawatan, harga pokok, harga jual, *expired*, jumlah, dan tanggal 4. Aktor menekan tombol simpah 5. Sistem melakukan pengecekan isi *form*, jika berhasil maka sistem menyimpan masukan aktor
Alternative flows	A.5 Pada saat aktor menambah bahan baru dan ada *field* yang belum diisi maka sistem memberi peringatan untuk mengisi *field* yang belum terisi.
Post-condition	Data bahan baru berhasil di simpan dan sistem menampilkan daftar pengeluaran bahan.

Tabel 4.15 menampilkan *use case scenario* hapus pengeluaran bahan, hapus pengeluaran bahan merupakan aktivitas yang dapat memungkinkan aktor untuk menghapus data pengeluaran bahan.

Tabel 4.15 *Use Case Scenario* Hapus Pengeluaran Bahan

POS-KF-012	Hapus pengeluaran bahan
Objective	Memungkinkan aktor untuk menghapus pengeluaran bahan
Actor	Dokter pemilik, perawat
Pre-condition	Aktor berada di halaman menu Pengeluaran Bahan
Main flow	1. Aktor menekan tombol hapus pada pengeluaran bahan yang ingin di hapus 2. Sistem akan menampilkan konfirmasi hapus pengeluaran klinik yang dipilih oleh aktor 3. Aktor menekan tombol ok 4. Sistem akan menghapus data pengeluaran bahan yang di pilih aktor
Alternative flows	A.3 Aktor menekan tombol cancel maka sistem batal menghapus data pengeluaran bahan yang di pilih aktor
Post-condition	Pengeluaran bahan yang dipilih aktor berhasil di hapus dan sistem menampilkan daftar pengeluaran bahan.

Tabel 4.16 menampilkan *use case scenario* lihat gaji, lihat gaji merupakan aktivitas yang dapat memungkinkan aktor untuk melihat daftar gaji yang didapatkan setiap bulannya.

Tabel 4.16 *Use Case Scenario* Lihat Gaji

POS-KF-013	Lihat gaji
Objective	Memungkinkan aktor untuk melihat gaji

Actor	Dokter, perawat
Pre-condition	Aktor sudah melakukan *login*
Main flow	1. Aktor memilih menu Gaji 2. Sistem menjalankan proses lihat gaji dan menampilkan daftar berupa tabel dengan 10 baris yang berisikan tanggal, gaji, dan status 3. Sistem menampilkan *form* dropdown bulan dan tahun untuk melakukan *filter* data gaji sesuai dengan bulan dan tahun yang diinginkan aktor
Post-condition	Aktor dapat melihat daftar gaji.

Tabel 4.17 menampilkan *use case scenario edit* stok bahan, *edit* stok bahan merupakan aktivitas yang dapat memungkinkan aktor untuk mengubah data stok bahan yang sudah tersimpan dalam sistem.

Tabel 4.17 *Use Case Scenario Edit* Stok bahan

POS-KF-014	*Edit* stok bahan
Objective	Memungkinkan aktor untuk meng*edit* stok bahan
Actor	Dokter pemilik
Pre-condition	Aktor berada di halaman menu Bahan
Main flow	1. Aktor menekan tombol *edit* pada stok bahan yang ingin di ubah 2. Sistem menampikan halaman *form* untuk meng*edit* stok bahan 3. Aktor merubah data yang diinginkan dari nama bahan, stok minimal, satuan, harga pokok, harga jual, perawatan, dan *expired* 4. Aktor menekan tombol *edit* 5. Sistem melakukan pengecekan isi *form*, jika berhasil maka sistem menyimpan masukan aktor
Post-condition	Stok bahan berhasil di ubah dan sistem menampilkan daftar stok bahan.

Tabel 4.18 menampilkan *use case scenario* hapus stok bahan, hapus stok bahan merupakan aktivitas yang dapat memungkinkan aktor untuk menghapus data stok bahan yang sudah tersimpan dalam sistem.

Tabel 4.18 *Use Case Scenario* Hapus Stok bahan

POS-KF-015	Hapus stok bahan
Objective	Memungkinkan aktor untuk menghapus stok bahan
Actor	Dokter pemilik

Pre-condition	Aktor berada di halaman menu Bahan
Main flow	1. Aktor menekan tombol hapus pada stok bahan yang ingin di hapus 2. Sistem akan menampilkan konfirmasi hapus stok bahan yang dipilih oleh aktor 3. Aktor menekan tombol ok 4. Sistem akan menghapus data stok bahan yang di pilih aktor
Alternative flows	A.3 Aktor menekan tombol cancel maka sistem batal menghapus data stok bahan yang di pilih aktor
Post-condition	Stok bahan yang dipilih aktor berhasil dihapus dan sistem menampilkan daftar stok bahan.

Tabel 4.19 menampilkan *use case scenario* lihat pendapatan klinik, lihat pendapatan klinik merupakan aktivitas yang dapat memungkinkan aktor untuk melihat daftar pendapatan klinik.

Tabel 4.19 *Use Case Scenario* Lihat Pendapatan Klinik

POS-KF-016	Lihat perndapatan klinik
Objective	Memungkinkan aktor untuk melihat pendapatan klinik
Actor	Dokter pemilik
Pre-condition	Aktor sudah melakukan *login*
Main flow	1. Aktor memilih menu Pendapatan Klinik 2. Sistem menjalankan proses lihat pendapatan pribadi dan menampilkan 2 tabel dengan 10 baris yang berisikan daftar pendapatan kotor dan daftar pendapatan bersih 3. Sistem menampilkan daftar pendapatan kotor berupa no faktur, tanggal, perawatan, regio, keterangan, dan total 4. Sistem menampilkan daftar pendapatan bersih berupa tanggal, pendapatan kotor, pengeluaran, pendapatan bersih 5. Sistem menampilkan *form* dropdown bulan dan tahun untuk melakukan *filter* data pendapatan kotor dan pendapatan bersih sesuai dengan bulan dan tahun yang diinginkan aktor 6. Sistem menampilkan *field* pencarian agar dapat memudahkan aktor untuk mencari data pendapatan kotor yang diinginkan

Post-condition	Aktor dapat melihat daftar pendapatan klinik.

Tabel 4.20 menampilkan *use case scenario* lihat gaji pegawai, lihat gaji pegawai merupakan aktivitas yang dapat memungkinkan aktor untuk melihat daftar gaji dari setiap dokter dan perawat yang didapatkan setiap bulannya.

Tabel 4.20 *Use Case Scenario* Lihat Gaji Pegawai

POS-KF-017	Lihat gaji pegawai
Objective	Memungkinkan aktor untuk melihat gaji pegawai
Actor	Dokter pemilik
Pre-condition	Aktor sudah melakukan *login*
Main flow	1. Aktor memilih menu Gaji Pegawai 7. Sistem menjalankan proses lihat gaji pegawai dan menampilkan 2 tabel dengan 10 baris yang berisikan daftar gaji dokter dan daftar gaji perawat 2. Sistem menampilkan daftar gaji dokter dan gaji perawat berupa nama dokter/nama perawat, tanggal, gaji, dan status 3. Status gaji "belum dibayarkan" ditampilkan berupa tombol 4. Sistem menampilkan *form* dropdown bulan dan tahun untuk melakukan *filter* data gaji dokter dan gaji perawat sesuai dengan bulan dan tahun yang diinginkan aktor
Post-condition	Aktor dapat melihat daftar gaji pegawai.

Tabel 4.21 menampilkan *use case scenario* ubah status gaji pegawai, ubah status gaji pegawai merupakan aktivitas yang dapat memungkinkan aktor untuk melihat daftar gaji dari setiap dokter dan perawat yang didapatkan setiap bulannya.

Tabel 4.21 *Use Case Scenario* Ubah Status Gaji Pegawai

POS-KF-018	Lihat gaji pegawai
Objective	Memungkinkan aktor untuk mengubah status gaji pegawai
Actor	Dokter pemilik
Pre-condition	Aktor sudah melakukan *login*
Main flow	1. Aktor memilih menu Gaji Pegawai 2. Sistem menjalankan proses lihat gaji pegawai dan menampilkan 2 tabel dengan 10 baris yang berisikan daftar gaji dokter dan daftar gaji perawat

	3. Sistem menampilkan daftar gaji dokter dan gaji perawat berupa nama dokter/nama perawat, tanggal, gaji, dan status 4. Status gaji "belum dibayarkan" ditampilkan berupa tombol 5. Aktor menekan tombol belum dibayarkan pada data yang diinginkan 6. Sistem akan menampilkan status gaji menjadi "sudah dibayarkan"
Post-condition	Aktor berhasil mengubah status gaji pegawai dan status gaji pegawai menjadi "sudah dibayarkan".

Tabel 4.22 menampilkan *use case scenario* tambah pasien baru, tambah pasien baru merupakan aktivitas yang dapat memungkinkan aktor untuk menambahkan data pasien baru dan menyimpannya ke dalam sistem.

Tabel 4.22 *Use Case Scenario* Tambah Pasien Baru

POS-KF-019	Tambah pasien baru
Objective	Memungkinkan aktor untuk menambahkan pasien baru
Actor	Perawat
Pre-condition	Aktor berada di halaman menu Pasien
Main flow	1. Aktor menekan tombol tambah pasien baru 2. Sistem menampilkan halaman *form* untuk memasukkan data pasien baru 3. Aktor memasukkan data berupa nama, tanggal lahir, alamat, pekerjaan, riwayat alergi, penyakit, dan no hp 4. Aktor menekan tombol simpan 5. Sistem melakukan pengecekan isi *form*, jika berhasil maka sistem menyimpan masukan aktor
Alternative flows	A.5 Pada saat aktor menambah pasien baru dan ada *field* yang belum diisi maka sistem memberi peringatan untuk mengisi *field* yang belum terisi.
Post-condition	Data pasien baru berhasil di simpan dan sistem menampilkan daftar pasien.

Tabel 4.23 menampilkan *use case scenario* tambah antrian pasien, tambah antrian pasien merupakan aktivitas yang dapat memungkinkan aktor untuk menambahkan daftar antrian pasien.

Tabel 4.23 *Use Case Scenario* Tambah Antrian Pasien

POS-KF-020	Tambah antrian pasien

Objective	Memungkinkan aktor untuk menambahkan antrian pasien
Actor	Perawat
Pre-condition	Aktor berada di halaman menu Pasien
Main flow	1. Aktor menekan tombol tambah antrian pasien 2. Sistem menampilkan halaman *form* untuk menambahkan keluhan dan pilihan dokter 3. Aktor memasukkan data berupa keluhan pasien dan pilihan dokter 4. Aktor menekan tombol simpan 5. Sistem melakukan pengecekan isi *form*, jika berhasil maka sistem menyimpan masukan aktor
Alternative flows	A.5 Pada saat aktor menambah antrian pasien dan ada *field* yang belum diisi maka sistem memberi peringatan untuk mengisi *field* yang belum terisi.
Post-condition	Antrian pasien berhasil ditambahkan dan sistem menampilkan halaman antrian pasien.

Tabel 4.24 menampilkan *use case scenario* cetak kartu berobat pasien, cetak kartu berobat pasien merupakan aktivitas yang dapat memungkinkan aktor untuk mencetak kartu berobat pasien.

Tabel 4.24 *Use Case Scenario* Cetak Kartu Berobat Pasien

POS-KF-021	Cetak kartu berobat pasien
Objective	Memungkinkan aktor untuk mencetak kartu berobat pasien
Actor	Perawat
Pre-condition	Aktor berada di halaman menu Pasien
Main flow	1. Aktor menekan tombol cetak pada daftar pasien yang ingin di cetak 2. Sistem akan menampilkan data pasien yang di pilih aktor dalam *form*at pdf 3. Aktor mencetak kartu berobat pasien
Post-condition	Aktor berhasil mencetak kartu berobat pasien yang dipilih.

Tabel 4.25 menampilkan *use case scenario* cetak nomor antrian pasien, cetak nomor antrian pasien merupakan aktivitas yang dapat memungkinkan aktor untuk mencetak nomor antrian pasien.

Tabel 4.25 *Use Case Scenario* Cetak Nomor Antrian Pasien

POS-KF-022	Cetak nomor antrian pasien
Objective	Memungkinkan aktor untuk mencetak nomor antrian pasien
Actor	Perawat
Pre-condition	Aktor berada di halaman menu Antrian
Main flow	1. Aktor menekan tombol cetak pada antrian yang ingin di cetak 2. Sistem akan menampilkan nomor antrian pasien yang di pilih aktor dalam *form*at pdf 3. Aktor mencetak nomor antrian pasien
Post-condition	Aktor berhasil mencetak nomor antrian pasien.

Tabel 4.26 menampilkan *use case scenario* pembayaran perawatan pasien, pembayaran perawatan pasien merupakan aktivitas yang dapat memungkinkan aktor untuk menambahkan pembayaran perawatan bagi pasien yang telah selesai melakukan perawatan.

Tabel 4.26 *Use Case Scenario* Pembayaran Perawatan Pasien

POS-KF-023	Pembayaran perawatan pasien
Objective	Memungkinkan aktor untuk menambahkan pembayaran perawatan pasien
Actor	Perawat
Pre-condition	Aktor berada di halaman menu Antrian
Main flow	1. Aktor menekan tombol pembayaran pada antrian pasien 2. Sistem akan menampilkan rangkuman perawatan, daftar bahan dan dan perawatan 3. Rangkuman perawatan ang ditampilkan berupa nama pasien, perawatan, regio, dan keterangan 4. Dafar bahan ditampilkan berupa tabel dengan 10 baris yang berisikan kode, nama bahan, perawatan, harga jual, dan tombol untuk *add to cart* 5. Sistem menampilkan *field* pencarian agar dapat memudahkan aktor untuk mencari data bahan yang diinginkan 6. Data pembayaran perawatan yang ditampilkan sistem kepada aktor berupa nama bahan, harga, jumlah, subtotal, total, biaya perawatan, total akhir. 7. Aktor mengisi *field* jumlah uang yang diberikan pasien

	8. Aktor menekan tombol pembayaran 9. Sistem menyimpan data pembayaran perawatan pasien
Post-condition	Pembayaran perawatan pasien berhasil disimpan dan sistem menampilkan halaman rangkuman pembayaran pasien.

Tabel 4.27 menampilkan *use case scenario* lihat rangkuman pembayaran, lihat rangkuman pembayaran merupakan aktivitas yang dapat memungkinkan aktor untuk melihat rangkuman pembayaran dari antrian pasien selesai.

Tabel 4.27 *Use Case Scenario* Lihat Rangkuman Pembayaran

POS-KF-024	Lihat rangkuman pembayaran
Objective	Memungkinkan aktor untuk melihat rangkuman pembayaran
Actor	Perawat
Pre-condition	Aktor berada di halaman menu Antrian
Main flow	1. Aktor menekan tombol rangkuman pembayaran 2. Sistem akan menampilkan halaman rangkuman pembayaran untuk pasien yang telah selesai melakukan perawatan kepada aktor yang no faktur, no pasien, nama, alamat, umur, bahan, jenis perawatan, regio, dan total
Post-condition	Aktor dapat melihat data rangkuman pembayaran pasien yang diinginkan.

Tabel 4.28 menampilkan *use case scenario* cetak kwitansi, cetak kwitansi merupakan aktivitas yang dapat memungkinkan aktor untuk mencetak kwitansi yang diinginkan.

Tabel 4.28 *Use Case Scenario* Cetak Kwitansi

POS-KF-025	Cetak kwitansi
Objective	Memungkinkan aktor untuk mencetak kwitansi
Actor	Perawat
Pre-condition	Aktor berada di halaman menu Antrian
Main flow	1. Aktor menekan tombol cetak 2. Sistem akan menampilkan data kwitansi yang di pilih aktor dalam *form*at pdf 3. Aktor mencetak kwitansi
Post-condition	Aktor berhasil mencetak kwitansi yang dipilih.

Tabel 4.29 menampilkan *use case scenario* cetak rekam medis, cetak rekam medis merupakan aktivitas yang dapat memungkinkan aktor untuk mencetak rekam medis pasien yang diinginkan.

Tabel 4.29 *Use Case Scenario* Cetak Rekam Medis

POS-KF-026	Cetak rekam medis
Objective	Memungkinkan aktor untuk mencetak kwitansi
Actor	Perawat
Pre-condition	Aktor berada di halaman menu Pasien
Main flow	1. Aktor menekan tombol cetak rekam medis 2. Sistem akan menampilkan data rekam medis pasien yang di pilih aktor dalam *form*at pdf 3. Aktor mencetak rekam medis
Post-condition	Aktor berhasil mencetak rekam medis yang dipilih.

Tabel 4.30 menampilkan *use case scenario* lihat daftar harga bahan, lihat stok bahan merupakan aktivitas yang dapat memungkinkan aktor untuk melihat daftar daftar harga bahan yang sudah tersimpan dalam sistem.

Tabel 4.30 *Use Case Scenario* Lihat Daftar Harga Bahan

POS-KF-027	Lihat daftar harga bahan
Objective	Memungkinkan aktor untuk melihat daftar daftar harga bahan
Actor	Perawat
Pre-condition	Aktor sudah melakukan *login*
Main flow	1. Aktor memilih menu Bahan 2. Sistem menjalankan proses lihat stok bahan dan menampilkan daftar berupa tabel dengan 10 baris 3. Daftar daftar harga bahan yang ditampilkan sistem berupa kode, nama, stok, satuan, status, harga jual, dan *expired* 4. Status pada daftar stok bahan merupakan status ketersediaan stok, terdapat 3 jenis status, yaitu tersedia, sedikit, dan habis 5. Sistem menampilkan *field* pencarian agar dapat memudahkan aktor untuk mencari data harga bahan yang diinginkan
Post-condition	Aktor dapat melihat daftar harga bahan.

Tabel 4.31 menampilkan *use case scenario* lihat daftar biaya perawatan, lihat daftar biaya perawatan merupakan aktivitas yang dapat memungkinkan aktor untuk melihat daftar biaya perawatan yang sudah tersimpan dalam sistem.

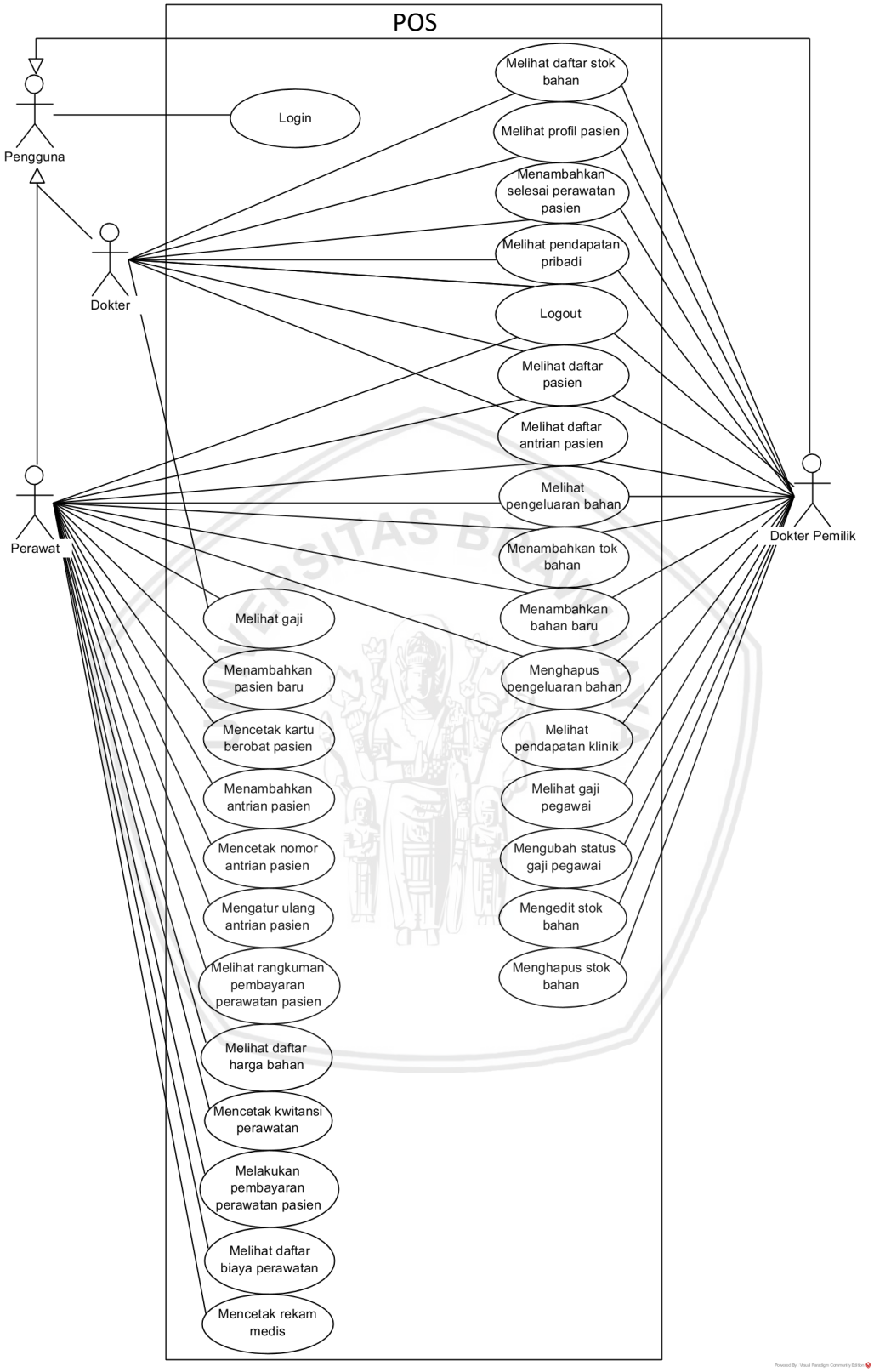
Tabel 4.31 *Use Case Scenario* Lihat Daftar Biaya Perawatan

POS-KF-028	Lihat daftar biaya perawatan
Objective	Memungkinkan aktor untuk melihat daftar biaya perawatan
Actor	Perawat
Pre-condition	Aktor sudah melakukan *login*
Main flow	1. Aktor memilih menu Perawatan 2. Sistem menjalankan proses lihat daftar biaya perawatan dan menampilkan daftar berupa tabel dengan 10 baris 3. Daftar biaya perawatan yang diampilkan berupa no, jenis perawatan, dan biaya 4. Sistem menampilkan *field* pencarian agar dapat memudahkan aktor untuk mencari data biaya perawatan yang diinginkan
Post-condition	Aktor dapat melihat daftar biaya perawatan.

Tabel 4.32 menampilkan *use case scenario* atur ulang antrian pasien, atur ulang antrian pasien merupakan aktivitas yang dapat memungkinkan aktor untuk mengatur ulang antrian pasien agar kembali ke urutan satu.

Tabel 4.32 *Use Case Scenario* Atur Ulang Antrian Pasien

POS-KF-029	Atur ulang antrian pasien
Objective	Memungkinkan aktor untuk mengatur ulang antrian pasien
Actor	Perawat
Pre-condition	Aktor berada di halaman menu Antrian
Main flow	1. Aktor menekan tombol reset 2. Sistem akan menghapus semua antrian yang ada
Post-condition	Aktor berhasil mengatur ulang antrian pasien dan sistem akan menampilkan halaman antrian pasien.



Gambar 4.11 *Use Case Diagram*

BAB 5 PERANCANGAN DAN IMPLEMENTASI

5.1 Perancangan Sistem

Pada tahap perancangan sistem hasil dari tahap rekayasa kebutuhan akan digambarkan kedalam beberapa diagram untuk nantinya dapat memudahkan proses implementasi sistem. Diagram yang akan ditampilkan antara lain *sequence diagram* dan *class diagram* yang digunakan sebagai perancangan pada sistem pengelolaan klinik gigi menggunakan *point of sale*.

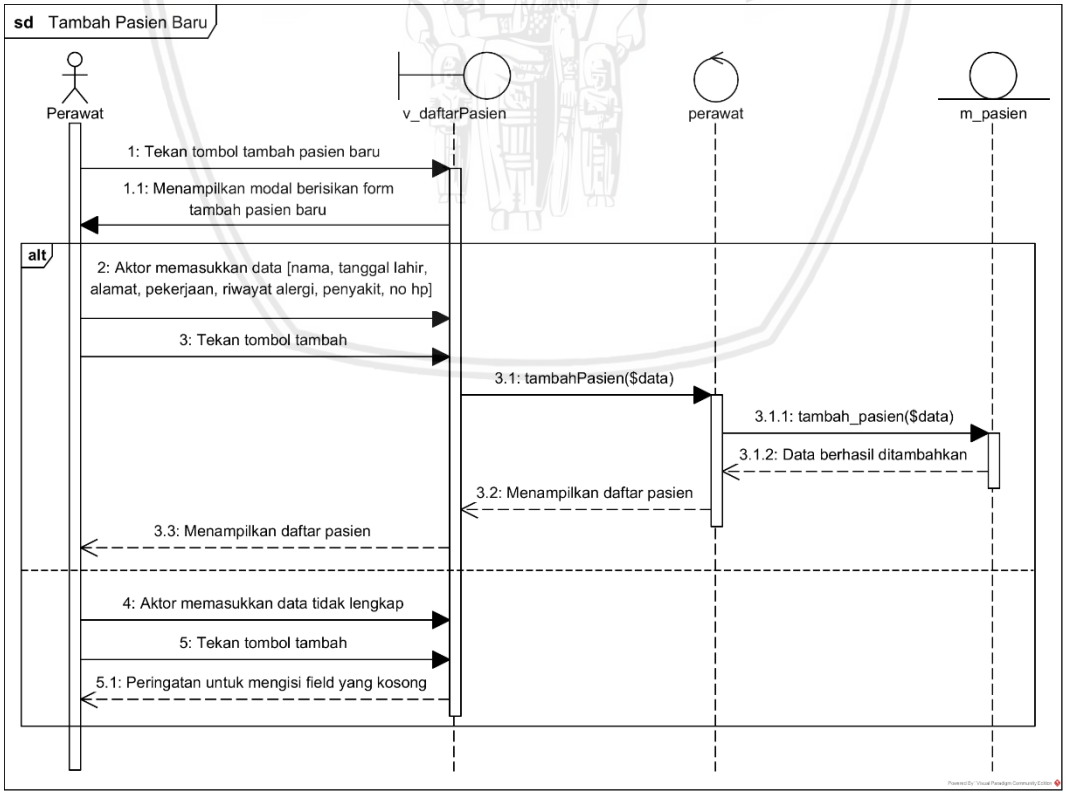
5.1.1 Sequence Diagram

Sequence diagram merupakan diagram yang menampilkan interaksi antar objek dengan menunjukkan pesan-pesan yang ditukar oleh objek-objek tersebut saat melakukan aksi tertentu.

5.1.1.1 Sequence Diagram Tambah Pasien Baru

Aktor perawat dapat melakukan tambah pasien baru. Gambar 5.1 merupakan *sequence diagram* yang menggambarkan interaksi objek dalam fungsi tambah tambah pasien baru.

Kode: POS-KF-019

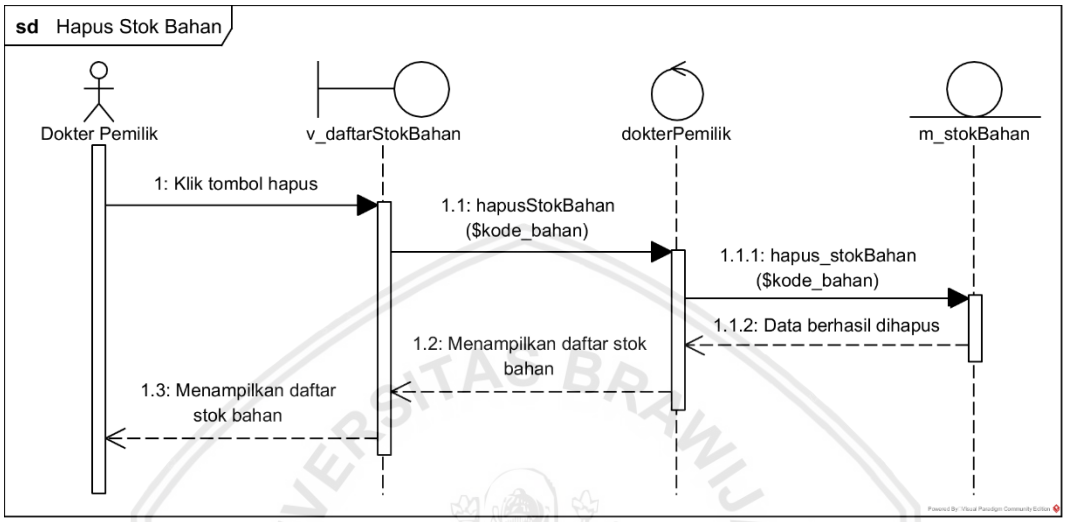


Gambar 5.1 *Sequence Diagram* Tambah Pasien Baru

5.1.1.2 Sequence Diagram Hapus Stok Bahan

Aktor perawat dapat melakukan hapus stok bahan. Gambar 5.2 merupakan *sequence diagram* yang menggambarkan interaksi objek di fungsi hapus stok bahan.

Kode: POS-KF-015

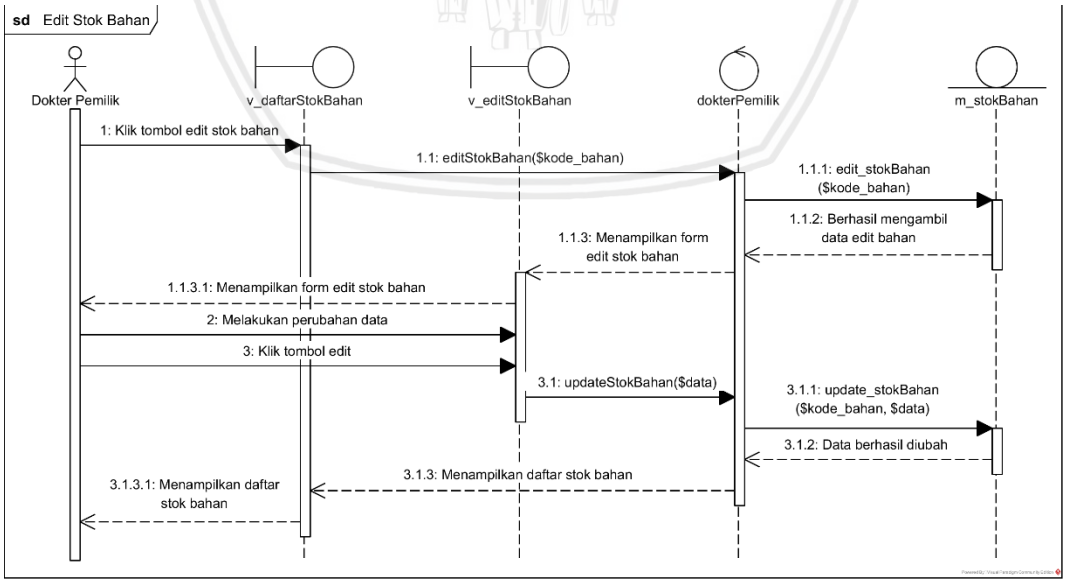


Gambar 5.2 *Sequence Diagram* Hapus Stok Bahan

5.1.1.3 Sequence Diagram Edit Stok Bahan

Aktor perawat dapat melakukan *edit* stok bahan. Gambar 5.3 merupakan *sequence diagram* yang menggambarkan interaksi objek di fungsi *edit* stok bahan.

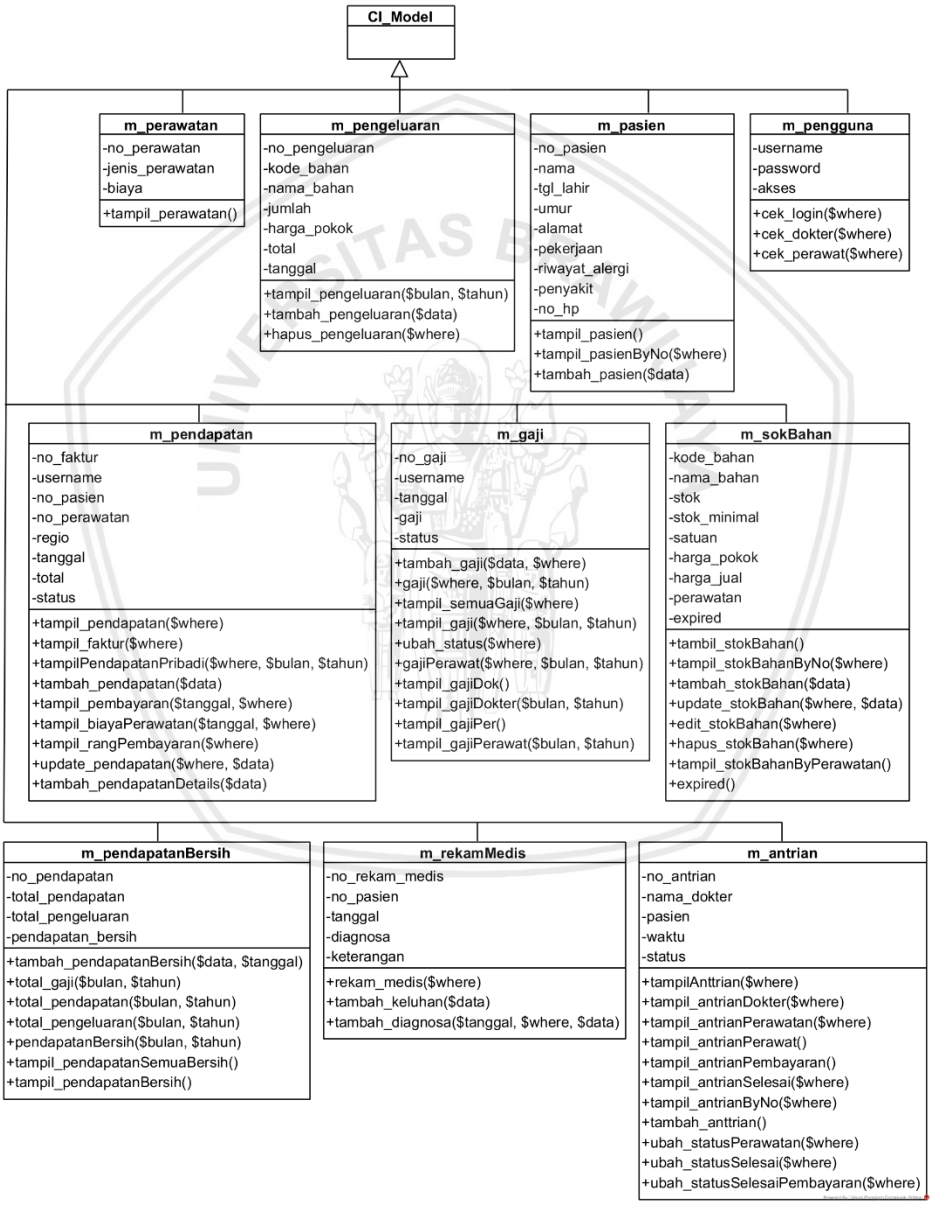
Kode: POS-KF-014



Gambar 5.3 *Sequence Diagram Edit* Stok Bahan

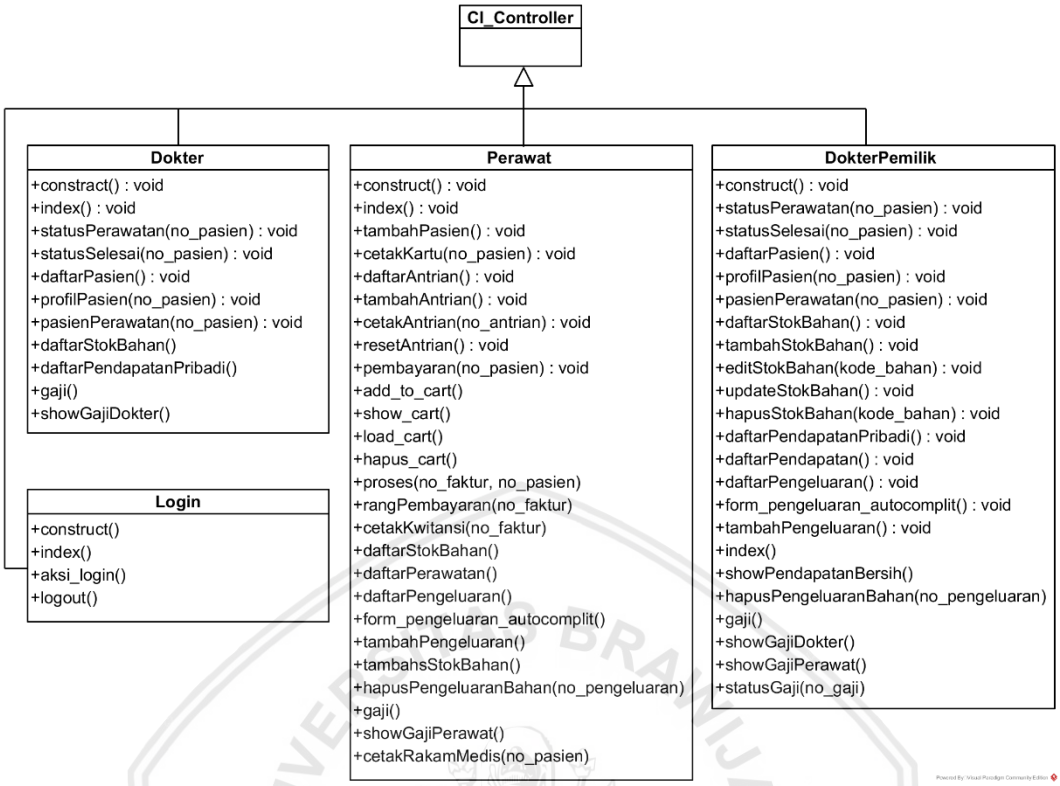
5.1.2 Class Diagram

Class diagram adalah sebuah diagram yang berfungsi menggambarkan suatu hubungan antar kelas dalam suatu sistem. *Class diagram* bertujuan untuk menunjukkan objek yang ada pada sistem yang di buat beserta relasi yang terjadi antara objek-objek tersebut. Pada aplikasi ini, *class diagram* yang di rancang memiliki *view*, *controller* dan *model* atau biasa disebut MVC. Gambar 5.4 menjelaskan *class diagram* untuk *model,* terlihat bahwa sistem memiliki 10 kelas pada *model*. Kelas *model* bertujuan untuk mengelola data pada sistem.



Gambar 5.4 *Model Class Diagram*

Gambar 5.5 menjelaskan *class diagram* untuk *controller,* terlihat bahwa sistem memiliki 4 kelas pada *controller* yaitu, dokter, dokter pemilik, perawat, dan *login*. Kelas *controller* bertujuan untuk menjembatani kelas *model* dengan kelas *view*.



Gambar 5.5 *Controller Class Diagram*

5.2 Perancangan Komponen

Perancangan komponen bertujuan untuk menjelaskan proses yang terjadi di beberapa bagian sub-sistem yang digunakan untuk menjalankan fungsionalitas sistem. Perancangan komponen dalam penelitian ini dilakukan dengan 3 sampel yaitu, fungsi tambah pasien baru, hapus stok bahan, dan *edit* stok bahan.

5.2.1 Tambah Pasien Baru

Pada proses pengelolaan pasien terdapat fungsi tambah pasien baru yang memungkinkan aktor dapat menambahkan pasien baru. Tabel 5.1 menjelaskan komponen-komponen yang terdapat pada fungsi tambah pasien baru.

Kode: POS-KF-019

Tabel 5.1 Algoritme Fungsi Tambah Pasien Baru

`model`

`procedure tambah_pasien ($data)`
`  load db <- pasien ->insert ($data)`
`end tambah_pasien ()`

`controller`

`procedure tambahPasien ()`
`  $today <- date ('Y-m-d')`
`  post <- nama`
`  post <- tgl_lahir`
`  post <- alamat`

```
      post <- pekerjaan
      post <- riwayat_alergi
      post <- penyakit
      post <- no_hp
      umur <- $today - tgl_lahir
      array data = pasien <- data (nama, tgl_lahir, umur, alamat,
   pekerjaan, riwayat_alergi, penyakit, no_hp)
      load model -> tambah_pasien ($data)
end tambahPasien ()
```

5.2.2 Hapus Stok Bahan

Pada proses pengelolaan stok bahan terdapat fungsi hapus stok bahan yang memungkinkan aktor dapat menghapus data stok bahan. Tabel 5.2 menjelaskan komponen-komponen yang terdapat pada fungsi hapus stok.

Kode: POS-KF-015

Tabel 5.2 Algoritme Fungsi Hapus Stok Bahan

```
model
procedure hapus_stokBahan($where)
      load db <- stok_bahan -> where ($where)
      load db <- stok_bahan -> delete
end hapus_stokBahan ()
controller
procedure hapusStokBahan($kode_bahan)
      array where = kode_bahan <- data ($kode_bahan)
      load model -> hapus_stokBahan($where)
end hapusStokBahan ()
```

5.2.3 *Edit* Stok Bahan

Pada proses pengelolaan stok bahan terdapat fungsi *edit* stok bahan yang memungkinkan aktor dapat mengubah data stok bahan. Tabel 5.3 menjelaskan komponen-komponen yang terdapat pada fungsi *edit* stok bahan.

Kode: POS-KF-014

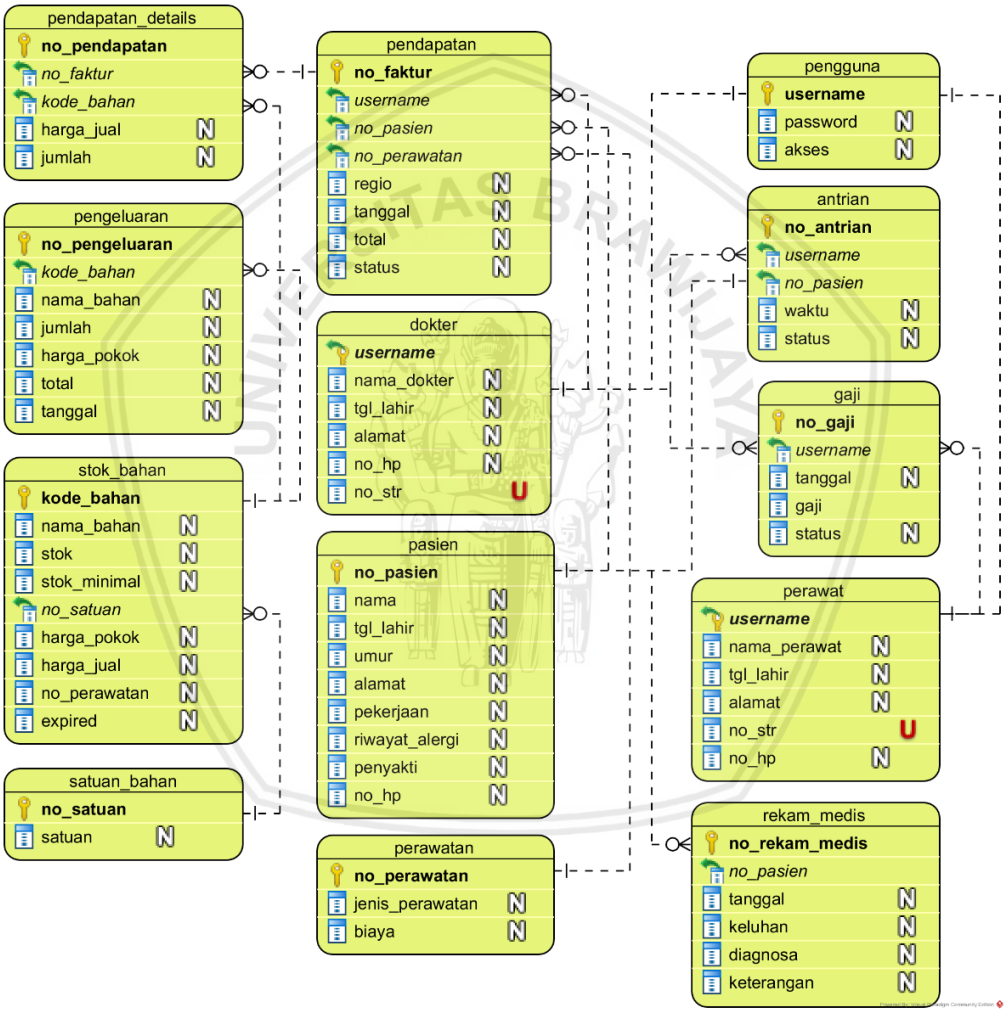
Tabel 5.3 Algoritme Fungsi *Edit* Stok Bahan

```
model
procedure update_stokBahan ($where, $data)
      load db <- stok_bahan -> where ($where)
      load db <- stok_bahan -> update
end update_stokBahan ()
controller
procedure updateStokBahan ()
      post <- kode_bahan
      post <- nama_bahan
      post <- stok
      post <- stok_minimal
      post <- no_satuan
      post <- harga_pokok
      post <- harga_jual
      post <- no_perawatan
      post <- expired
```

```
        array data = stok_bahan <- data (nama_bahan, stok,
    stok_minimal,      no_satuan,      harga_pokok,      harga_jual,
    no_perawatan, expired)
        array where = kode_bahan <- data (kode_bahan)
        load model -> update_stokBahan ($where, $data)
end tambahPasien ()
```

5.3 Perancangan Database

Perancangan *database* sistem pengelolaan klinik gigi menggunakan prinsip *point of sale* dalam penelitian ini digambarkan pada Gambar 5.6 dalam bentuk *conceptual data model*.

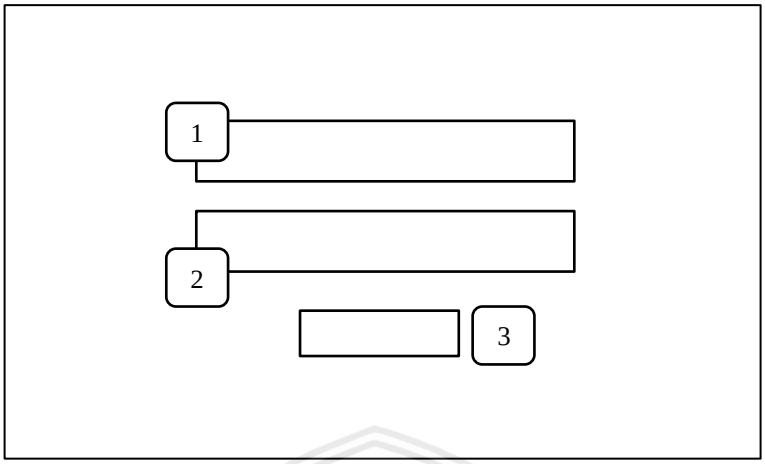


Gambar 5.6 *Conceptual Data Model*

5.4 Perancangan Antarmuka

Pada perancangan antarmuka akan dibuat rancangan yang menampilkan gambaran tampak sistem. Perancangan antarmuka yang dibuat hanya mengambil beberapa sampel saja pada fungsi untuk menu *login*, menu pasien, menu *edit* stok bahan, dan menu gaji.

5.4.1 Halaman *Login*

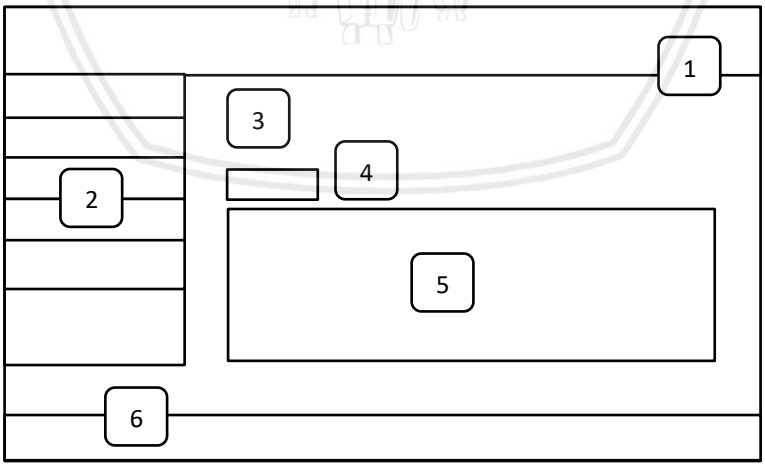


Gambar 5.7 Rancangan Antarmuka Halaman *Login*

Semua pengguna dapat mengakses halaman *login*, namun hanya pengguna yang terdaftar yang dapat masuk kedalam sistem sesuai dengan hak akses masing-masing. Rancangan antarmuka *login* dijelaskan pada Gambar 5.7.

1) *Field username*, yaitu *field* yang digunakan untuk memasukkan *username*.
2) *Field password*, yaitu *field* yang digunakan untuk memasukkan *password*.
3) Tombol login, merupakan tombol untuk masuk kedalam sistem, jika *username* dan *password* yang dimasukkan oleh pengguna benar, maka sistem akan menampilkan halaman awal dari akses masing-masing.

5.4.2 Halaman Menu Pasien

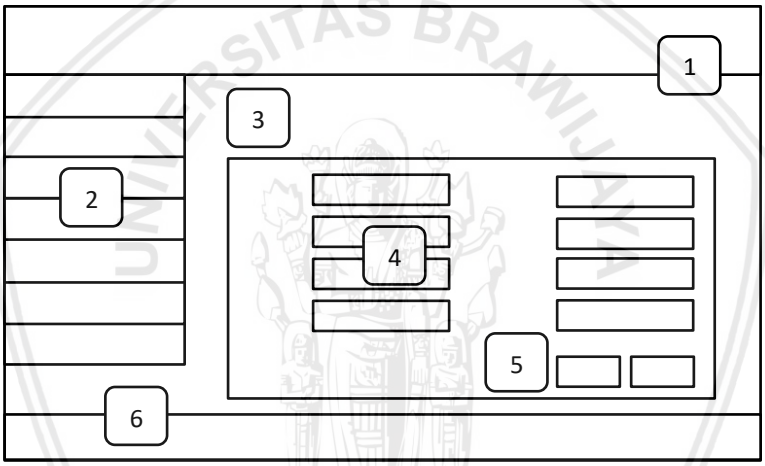


Gambar 5.8 Rancangan Antarmuka Halaman Menu Pasien

Gambar 5.8 menjelaskan rancangan antarmuka halaman menu pasien untuk aktor perawat. Halaman menu pasien bertujuan untuk menampilkan daftar pasien dan pengelolaan data pasien.

1) *Header*, merupakan kepala dari sebuah halaman web yang terletak pada bagian atas. *Header* berisi nama dari klinik gigi Senyum Sehat Dental Care.
2) *Side nav*, terdapat beberapa menu pada halaman yang dapat diakses oleh perawat.
3) *Label*, digunakan untuk menampilkan judul dari halaman menu pasien.
4) Tombol tambah pasien baru, merupakan tombol yang digunakan untuk menambahkan data pasien baru.
5) Tabel, berisikan informasi dari data pasien yang terdaftar.
6) *Footer,* merupakan bagian bawah dari sebuah halaman yang berisi *copyright* dari sistem.

5.4.3 Halaman *Edit* Stok Bahan



Gambar 5.9 Rancangan Antarmuka Halaman *Edit* Stok Bahan

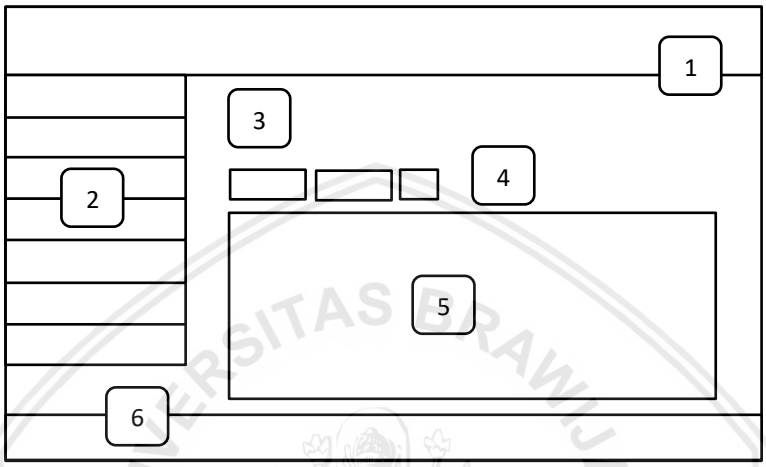
Gambar 5.9 menjelaskan rancangan antarmuka halaman *edit* stok bahan untuk aktor dokter pemilik. Halaman *edit* stok bahan bertujuan untuk menampilkan *form* yang berguna untuk merubah data stok bahan.

1) *Header*, merupakan kepala dari sebuah halaman web yang terletak pada bagian atas. *Header* berisi nama dari klinik gigi Senyum Sehat Dental Care.
2) *Side nav*, terdapat beberapa menu pada halaman yang dapat diakses oleh dokter pemilik.
3) *Label*, digunakan untuk menampilkan judul dari halaman *edit* stok bahan.
4) *Form*, digunakan untuk merubah data-data stok bahan yang sebelumna sudah tersimpan di dalam *database.*

5) Tombol *edit* dan *cancel*, tombol edit digunakan untuk menyimpan perubahan data, sedangkan tombol *cancel* digunakan untuk kembali ke halaman stok bahan.

6) *Footer,* merupakan bagian bawah dari sebuah halaman yang berisi *copyright* dari sistem.

5.4.4 Halaman Menu Gaji



Gambar 5.10 Rancangan Antarmuka Halaman Menu Gaji

Gambar 5.10 menjelaskan rancangan antarmuka halaman *menu gaji* untuk aktor dokter. Halaman menu gaji bertujuan untuk menampilkan daftar gaji dari aktor yang dapat di *filter* sesuai bulan dan tahun.

1) *Header*, merupakan kepala dari sebuah halaman web yang terletak pada bagian atas. *Header* berisi nama dari klinik gigi Senyum Sehat Dental Care.

2) *Side nav*, terdapat beberapa menu pada halaman yang dapat diakses oleh dokter pemilik.

3) *Label*, digunakan untuk menampilkan judul dari halaman menu gaji.

4) *Dropdown,* digunakan untuk menampilkan pilihan bulan dan tahun untuk melakukan *filter* data.

5) Tabel, berisikan informasi dari data gaji aktor.

6) *Footer,* merupakan bagian bawah dari sebuah halaman yang berisi *copyright* dari sistem.

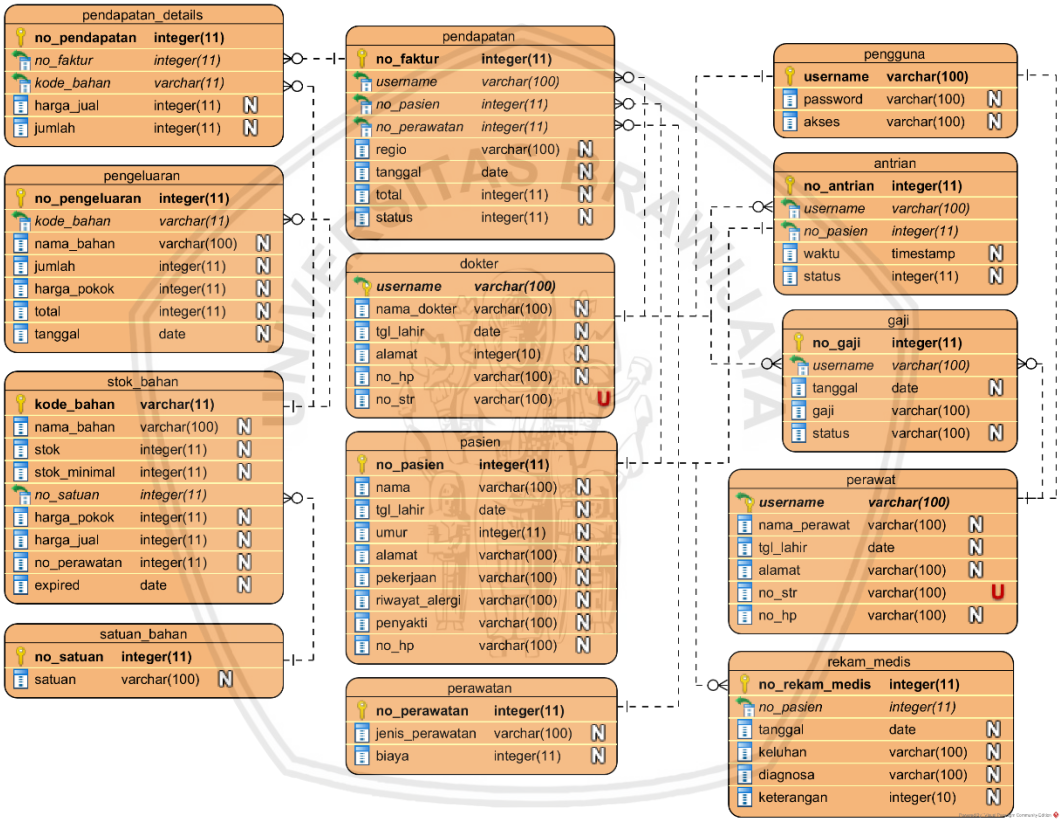
5.5 Implementasi Sistem

Tahap implementasi dalam penelitian ini dilakukan untuk merealisasikan perancangan sistem ke dalam bentuk kode sebagai serangkaian program. Hasil dari tahap implementasi merupakan sebuah sistem, baik *interface* atau pun

fungsi-fungsi yang siap untuk diuji pada tahap pengujian sistem. Pada tahap implementasi ini, penulis melakukan implementasi terhadap tabel database serta implementasi dari sistem pengelolaan klinik gigi menggunakan prinsip *point of sale*.

5.5.1 Implementasi Basis Data

Perancangan *database* sistem pengelolaan klinik gigi menggunakan prinsip *point of sale* dalam penelitian ini digambarkan pada Gambar 5.6 dalam bentuk *physical data model*. Pada *physical data model* terdapat 13 entitas yang mencakup pengelolaan pasien, bahan, pendapatan dan pengeluaran.



Gambar 5.11 P*hysical Data Model*

5.5.2 Imeplementasi Kode

Tabel 5.4 menjelaskan implementasi kode tambah pasien baru, dimana pada kelas *controller* terdapat fungsi tambahPasien () yang berguna sebagai jembatan antara kelas *view* dan *model*. Sedangkan pada kelas *model* terdapat fungsi tambah_pasien () yang digunakan untuk melakukan *insert* data pada *database*.

Kode: POS-KF-019

Tabel 5.4 Implementasi Kode Tambah Pasien Baru

perawat.php
function tambahPasien($no_pasien)

```
{
	$nama = $this->input->post('nama');
	$tgl_lahir = $this->input->post('tgl_lahir');
	$umur = $this->input->post('umur');
	$alamat = $this->input->post('alamat');
	$pekerjaan = $this->input->post('pekerjaan');
	$riwayat_alergi = $this->input->post('riwayat_alergi');
	$penyakit = $this->input->post('penyakit');
	$no_hp = $this->input->post('no_hp');
	$data = array (
		'nama' => $nama,
		'tgl_lahir' => $tgl_lahir,
		'umur' => $umur,
		'alamat' => $alamat,
		'pekerjaan' => $pekerjaan,
		'riwayat_alergi' => $riwayat_alergi,
		'penyakit' => $penyakit,
		'no_hp' => $no_hp,
		);
	$this->m_pasien->tambah_pasien($data);
	redirect(base_url('index.php/perawat/'));
}
```

m_pasien.php

```
function tambah_pasien($data)
{
	$this->db->insert ('pasien', $data);
}
```

Tabel 5.5 menjelaskan implementasi kode *edit* stok bahan, dimana pada kelas *controller* terdapat fungsi updateStokBahan () yang berguna sebagai jembatan antara kelas *view* dan *model*. Sedangkan pada kelas *model* terdapat fungsi update_stokBahan () yang digunakan untuk melakukan *update* data pada *database*.

Kode: POS-KF-014

Tabel 5.5 Implementasi *Edit* Stok bahan

dokterPemilik.php

```
function updateStokBahan ()
{
	$kode_bahan = $this->input->post('kode_bahan');
	$nama_bahan = $this->input->post('nama_bahan');
	$stok = $this->input->post('stok');
	$stok_minimal = $this->input->post('stok_minimal');
	$no_satuan = $this->input->post('no_satuan');
	$harga_pokok = $this->input->post('harga_pokok');
	$harga_jual = $this->input->post('harga_jual');
	$no_perawatan = $this->input->post('no_perawatan');
	$expired = $this->input->post('expired');
	$data = array (
		'nama_bahan' => $nama_bahan,
		'stok' => $stok,
		'stok_minimal' => $stok_minimal,
		'no_satuan' => $no_satuan,
		'harga_pokok' => $harga_pokok,
		'harga_jual' => $harga_jual,
```

```
            'no_perawatan' => $no_perawatan,
            'expired' => $expired,
            );
      $where = array ('kode_bahan' => $kode_bahan);
      $this->m_stokBahan->update_stokBahan ($where, $data);
      redirect(base_url('index.php/dokterPemilik/daftarStokBahan
'));
}
```

m_stokBahan.php

```
function update_stokBahan ($where, $data)
{
      $this->db->where($where);
      $this->db->update ('stok_bahan', $data);
}
```

Tabel 5.6 menjelaskan implementasi kode hapus *s*stok bahan, dimana pada kelas *controller* terdapat fungsi hapusStokBahan () yang berguna sebagai jembatan antara kelas *view* dan *model*. Sedangkan pada kelas *model* terdapat fungsi hapus_stokBahan () yang digunakan untuk melakukan *delete* data pada *database*.

Kode: POS-KF-015

Tabel 5.6 Implementasi Hapus Stok Bahan

dokterPemilik.php

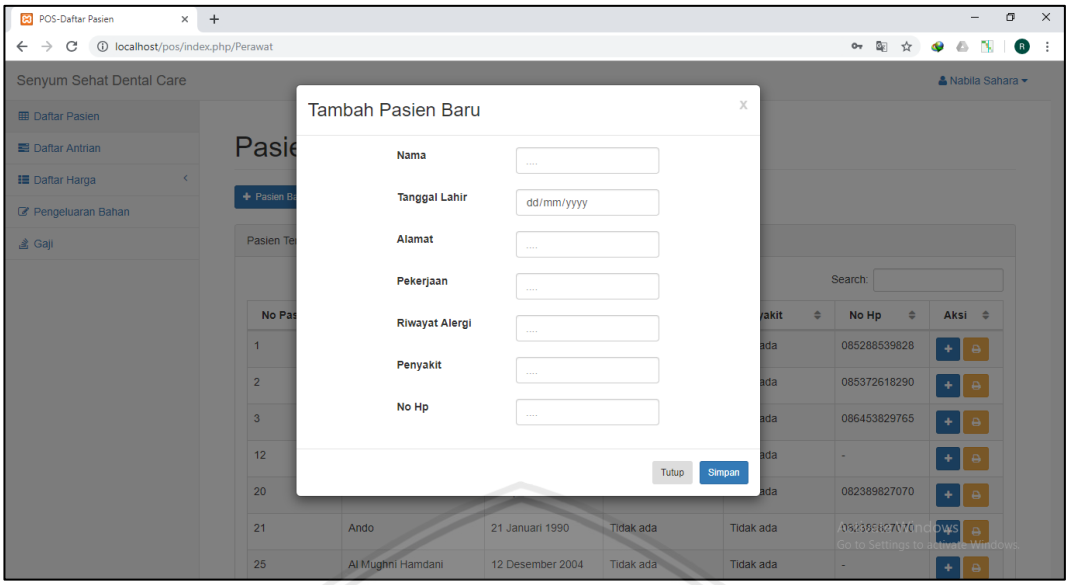
```
function hapusStokBahan($kode_bahan)
{
      $where = array ('kode_bahan' => $kode_bahan);
      $this->m_stokBahan->hapus_stokBahan($where);
      redirect(base_url('index.php/dokterPemilik/daftarStokBahan
'));
}
```

m_stokBahan.php

```
function hapus_stokBahan($where)
{
      $this->db->where($where);
      $this->db->delete('stok_bahan');
}
```

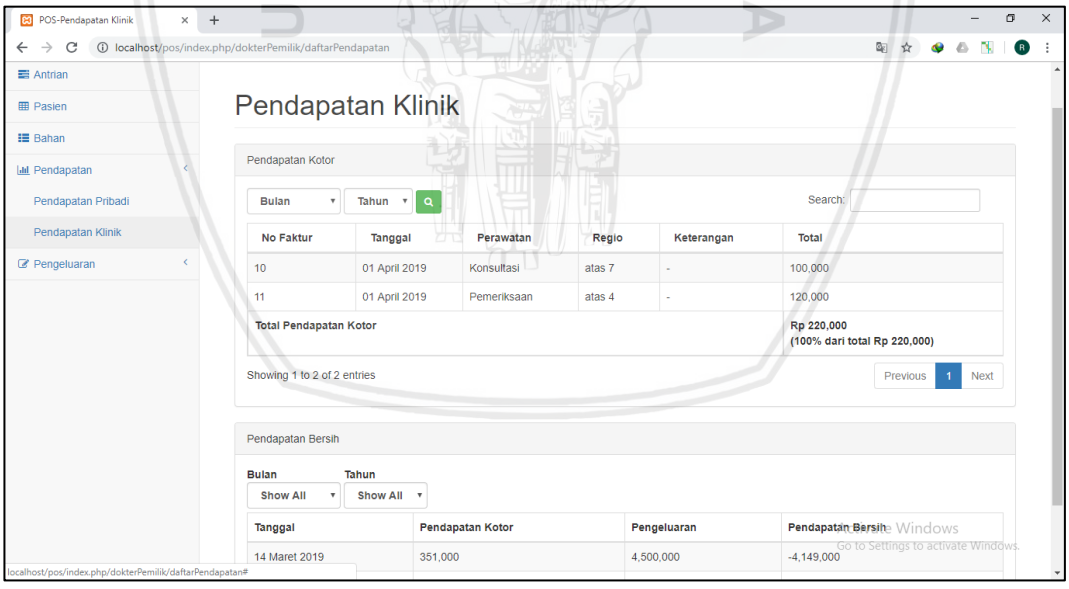
5.5.3 Impelementasi Antarmuka

Implementasi sistem merupakan implementasi bagaimana sistem ini dioperasikan berdasarkan tampilan antarmuka. Gambar 5.12 merupakan implementasi antarmuka dari halaman tambah pasien baru. Terdapat *modal* yang berisikan *form* untuk memasukkan data pasien serta terdapat 2 buah tombol yang digunakan untuk menyimpan masukkan data dan untuk menutup *modal*.



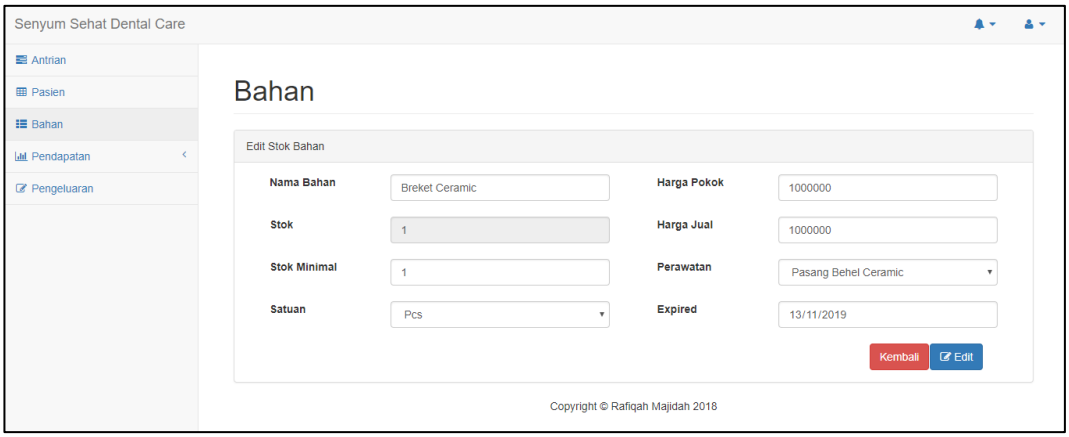
Gambar 5.12 Implementasi Antarmuka Tambah Pasien Baru

Gambar 5.13 merupakan implementasi antarmuka dari halaman lihat pendapatan klinik. Terdapat tabel yang berisikan data daftar pendapatan bersih dan pendapatan kotor serta terdapat dropdown untuk memilih bulan dan tahun yang dapat digunakan untuk melakukan *filter* data pendapatan klinik.



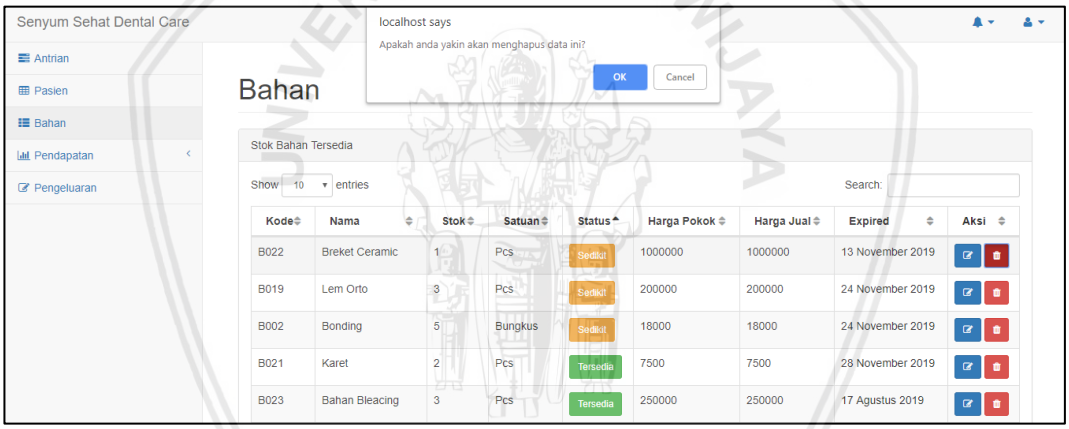
Gambar 5.13 Implementasi Antarmuka Lihat Pendapatan Klinik

Gambar 5.14 merupakan implementasi antarmuka dari halaman *edit* stok bahan. Terdapat *form edit* stok bahan yang dapat digunakan untuk merubah data stok bahan yang sudah ada serta terdapat 2 buah tombol yang digunakan untuk menimpan masukkan data dan untuk menutup kembali ke halaman stok bahan.



Gambar 5.14 Implementasi Antarmuka *Edit* Stok bahan

Gambar 5.15 merupakan implementasi antarmuka dari hapus stok bahan. Terdapat tombol aksi yang dapat digunakan untuk menghapus stok bahan yang sudah ada. Kemudian akan menampilkan konfirmasi untuk melanjutkan proses hapus stok bahan. Terdapat 2 tombol, yaitu *ok* dan *cancel*.



Gambar 5.15 Implementasi Antarmuka Hapus Stok Bahan

BAB 6 PENGUJIAN DAN ANALISIS

6.1 Pengujian Fungsional

Pengujian perangkat lunak merupakan suatu elemen yang penting untuk menjamin kualitas perangkat lunak. Pengujian dilakukan untuk mengurangi kesalahan sedini mungkin serta meningkatkan kualitas dan mempertahankan kepuasan pengguna. Pada penelitian ini dilakukan pengujian fungsional dan non-fungsional. Pengujian fungsional dilakukan metode *white-box testing* dan *black-box testing* dengan menggukan strategi pengujian unit, pengujian integrasi dan pengujian validasi, sedangkan untuk pengujian non-fungsional akan dilakukan *performance testing*.

6.1.1 Pengujian Unit

Pengujian unit dilakukan dengan cara melihat ke dalam modul dengan meneliti *psudocode* program, kemudian akan dilakukan analisis apakah masukan dan keluaran dari unit program telah tepat dan sesuai harapan. Skenario pengujian unit yang dilakukan pada laporan penelitian ini mengambil dua unit, antara lain tampil_pendapatan () dan tambah_pendapatanBersih ().

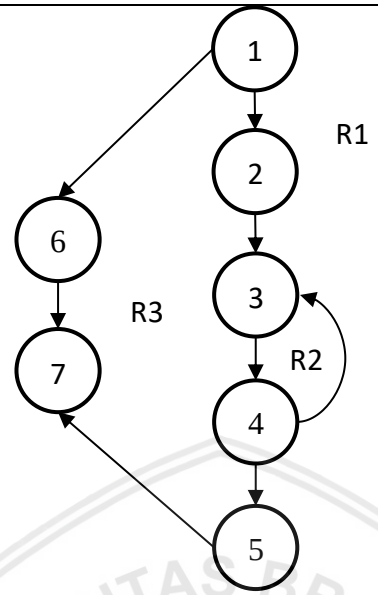
6.1.1.1 Unit tampil_pendapatan ()

Fungsi lihat pendapatan klinik merupakan fungsi yang memungkinkan pengguna untuk melihat pendapatan klinik. tampil_pendapatan () merupakan salah satu unit yang membangun fungsi lihat pendapatan klinik. Tabel 6.1 menampilkan algoritme dari unit tampil_pendapatan ().

Tabel 6.1 Algoritme tampil_pendapatan ()

```
m_pendapatan.php
start
   load db <- pendapatan -> select
   load db <- pendapatan -> join (perawatan)
   load db <- where ('status' = 1)          1
   query <- load db -> get
   if query -> num row  > 0     2
      foreach query -> result () -> data        3
            pendapatan [] <- data        4
      end foreach
      return pendapatan        5
    else              6
      return 0
    end if          7
```

```
end
```



Gambar 6.1 *Flow Graph* Algoritme tampil_pendapatan ()

Pada algoritme tampil_pendapatan () yang dijelaskan dalam Tabel 6.1 terdapat angka-angka yang menunjukkan bahwa algoritme tersebut memiliki node-node sebanyak angka, yaitu terdapat 7 node yang akan di ubah ke dalam bentuk *flow graph*. Gambar 6.1 merupakan *flow graph* dari algoritme tampil_pendapatan () yang akan dilakukan perhitungan *cyclometic complexity*. Berikut perhitungan *cyclometic complexity* pada *flow graph* algoritme tampil_pendapatan ().

V(G) = E – N + 2

= 8 – 7 + 2

= 3

V(G) = P + 1

= 2 + 1

= 3

V(G) = Jumlah region

= 3

Dari perhitungan *cyclometic complexity* didapatkan sebanyak 3 jalur independen, antara lain:

1. 1 – 6 – 7
2. 1 – 2 – 3 – 4 – 5 – 7
3. 1 – 2 – 3 – 4 – 3 – 4 – 5 – 7

Dari jalur independen yang sudah dibuat selanjutnya dilakukan pengujian melalui *Test Case* atau kasus uji di masing-masing jalurnya. *Test Case* pada tiap-tiap jalur independen dijelaskan pada Tabel 6.2.

Tabel 6.2 *Test Case* Algoritme tampil_pendapatan ()

No	Jalur	Data Input	Hasil yang diharapkan	Hasil yang diperoleh	Status
1	1 – 6 – 7	where = (bulan = 01 tahun = 1999)	Tidak ada data di dalam *database* dan mengembalikan nilai 0.	Tidak ada data di dalam *database.*	Valid
2	1 – 2 – 3 – 4 – 5 – 7	where = (bulan = 03 tahun = 2019)	Ada data di dalam *database* dan mengembalikan nilai data pendapatan.	Ada data di dalam *database* dan mengembalikan nilai data pendapatan.	Valid
3	1 – 2 – 3 – 4 – 3 – 4 – 5 – 7	where = (bulan = 04 tahun = 2019)	Ada data di dalam *database* dan mengembalikan nilai data pendapatan.	Ada data di dalam *database* dan mengembalikan nilai data pendapatan.	Valid

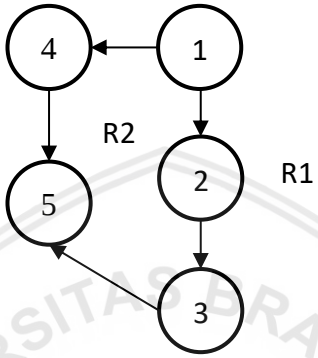
6.1.1.2 Unit tambah_pendapatanBersih ()

Fungsi lihat pendapatan bersih merupakan fungsi yang memungkinkan pengguna untuk melihat pendapatan bersih. tambah_pendapatanBersih () merupakan salah satu unit yang membangun fungsi lihat pendapatan bersih. Tabel 6.3 menampilkan algoritme dari unit tambah_pendapatanBersih ().

Tabel 6.3 Algoritme tambah_pendapatanBersih ()

```
m_pendapatanBersih.php
start
   load db <- pendapatan_bersih -> select
   load db <- where ('tanggal =', tanggal)    1
   $query <- load db -> get
   if query -> num row == 0    2
      load db <- pendapatan_bersih -> insert (data)    3
    else                    4
      return 0
    end if          5
end
```

Pada algoritme tambah_pendapatanBersih () yang dijelaskan dalam Tabel 6.3 terdapat angka-angka yang menunjukkan bahwa algoritme tersebut memiliki node-node sebanyak angka, yaitu terdapat 7 node yang akan di ubah ke dalam bentuk *flow graph*. Gambar 6.2 merupakan *flow graph* dari algoritme tambah_pendapatanBersih () yang akan dilakukan perhitungan *cyclometic complexity*. Berikut perhitungan *cyclometic complexity* pada *flow graph* algoritme tambah_pendapatanBersih ().



Gambar 6.2 *Flow Graph* Algoritme tambah_pendapatanBersih ()

V(G) = E – N + 2

= 5 – 5 + 2

= 2

V(G) = P + 1

= 1 + 1

= 2

V(G) = Jumlah region

= 2

Dari perhitungan *cyclometic complexity* didapatkan sebanyak 2 jalur independen, antara lain:

1. 1 – 4 – 5
2. 1 – 2 – 3 – 5

Dari jalur independen yang sudah dibuat selanjutnya dilakukan pengujian melalui *Test Case* atau kasus uji di masing-masing jalurnya. *Test Case* pada tiap-tiap jalur independen dijelaskan pada Tabel 6.4.

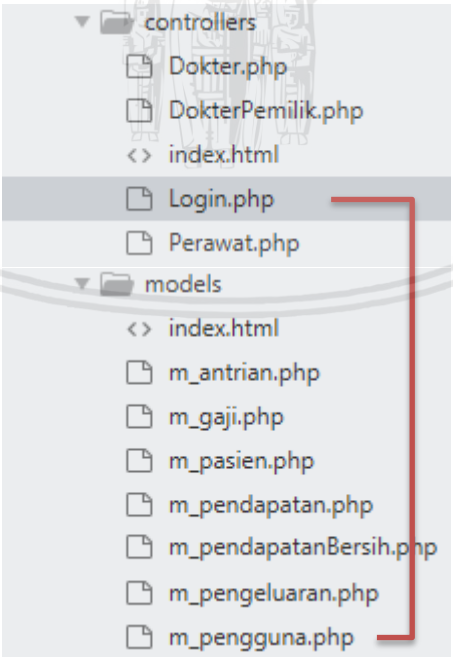
Tabel 6.4 *Test Case* Algoritme tampil_pendapatan ()

No	Jalur	Data Input	Hasil yang diharapkan	Hasil yang diperoleh	Status
1	1 – 4 – 5	Tanggal = 2019-03-04	Ada data di dalam database dan tidak bisa menambahkan data baru.	Gagal menambahkan data baru.	Valid
2	1 – 2 – 3 – 5	Tanggal = 2019-05-01	Tidak ada data di dalam *database* dan berhasil menambahkan data baru.	Berhasil menambahkan data baru.	Valid

6.1.2 Pengujian Integrasi

6.1.2.1 Login

Fungsi *login* merupakan fungsi yang memungkinkan pengguna untuk masuk kedalam sistem. Operasi aksi_login () terdapat pada kelas *Login* yang memanggil 3 operasi lain dari kelas m_pengguna, yaitu cek_login (), cek_dokter (), dan cek_perawat () yang digambarkan pada Gambar 6.3.



Gambar 6.3 Arsitekur Komponen Uji

Tabel 6.5 menampilkan algoritme dari operasi aksi_login () merupakan salah satu operasi pada fungsi *login*.

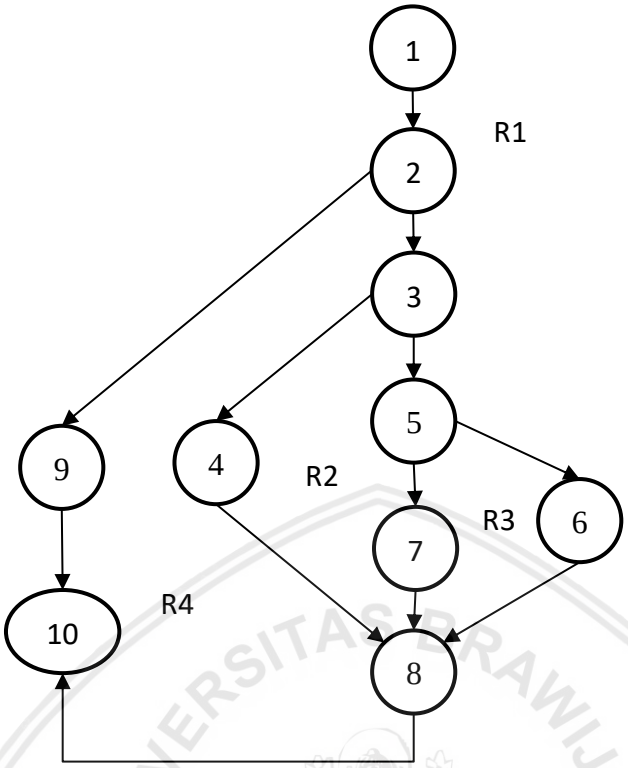
Tabel 6.5 Algoritme aksi_login ()

```
Login.php
start
   username <- post('username')
   password <- post('password')                    1
   where <- array('username','password')
   cek <- load model -> cek_login ('pengguna', where)

   if cek > 0                                      2
      data <- cek -> array
      this <- set userdata('login')
      if data['akses'] = 'dokter_pemilik'          3
           data session <- array <- load model -> cek_dokter
('username', 'name', 'akses', 'status')
           this <- set userdata                    4
           redirect 'DokterPemilik'
      elseif data['akses'] = 'dokter'              5
           data session <- array <- load model -> cek_dokter
('username', 'name', 'akses', 'status')
           this <- set userdata
           redirect 'Dokter'                       6


      elseif data['akses'] = 'perawat'
           data session <- array <- load model -> cek_perawat
('username', 'name', 'akses', 'status')
           this <- set userdata                    7
           redirect 'Perawat'
      end if                                       8
   else                                            9
       load view
       echo "Gagal Login Username / Password yang anda masukkan
salah."
       redirect 'Login'                            10
   end if
end
```

Pada algoritme aksi_login () yang dijelaskan dalam Tabel 6.5 terdapat angka-angka yang menunjukkan bahwa algoritme tersebut memiliki node-node sebanyak angka, yaitu terdapat 10 node yang akan di ubah ke dalam bentuk *flow graph*.



Gambar 6.4 *Flow Graph* Algoritme *Login*

Gambar 6.4 merupakan *flow graph* dari algoritme aksi_login () yang akan dilakukan perhitungan *cyclometic complexity*. Berikut perhitungan *cyclometic complexity* pada *flow graph* algoritme aksi_login ().

V(G) = E – N + 2

= 12 – 10 + 2

= 4

V(G) = P + 1

= 3 + 1

= 4

V(G) = Jumlah region

= 4

Dari perhitungan *cyclometic complexity* didapatkan sebanyak 4 jalur independen, antara lain:

1. 1 – 2 – 9 – 10
2. 1 – 2 – 3 – 4 – 8 – 10
3. 1 – 2 – 3 – 5 – 6 – 8 – 10

4. 1 – 2 – 3 – 5 – 7 – 8 – 10

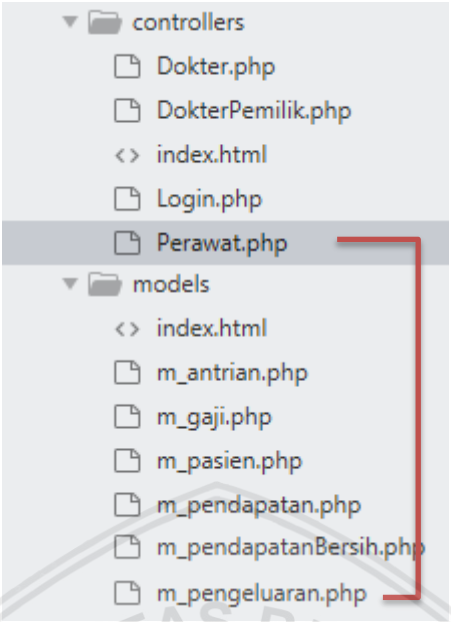
Dari jalur independen yang sudah dibuat selanjutnya dilakukan pengujian melalui *Test Case* atau kasus uji di masing-masing jalurnya. *Test Case* pada tiap-tiap jalur independen dijelaskan pada Tabel 6.6.

Tabel 6.6 *Test Case* Algoritme *Login*

No	Jalur	Data Input	Hasil yang diharapkan	Hasil yang diperoleh	Status
1	1 – 2 – 9 – 10	Username = test Password = test	Tidak dapat masuk kedalam sistem dan sistem menampilkan peringatan bahwa username dan password salah	Tampil halaman login dan muncul peringatan "**Gagal *Login*!** Username / Password yang anda masukkan salah."	Valid
2	1 – 2 – 3 – 4 – 8 – 10	Username = munadiyah Password = dokterpemilik	Pengguna dapat masuk ke dalam halaman dokterPemilik	Berhasil masuk ke dalam halaman dokterPemilik	Valid
3	1 – 2 – 3 – 5 – 6 – 8 – 10	Username = nadia Password = dokter	Pengguna dapat masuk ke dalam halaman dokter	Berhasil masuk ke dalam halaman dokter	Valid
4	1 – 2 – 3 – 5 – 7 – 8 – 10	Username = perawat Password = perawat	Pengguna dapat masuk ke dalam halaman perawat	Berhasil masuk ke dalam halaman perawat	Valid

6.1.2.2 Lihat Pengeluaran Bahan

Fungsi lihat pengeluaran bahan merupakan fungsi yang memungkinkan pengguna untuk melihat daftar pengeluaran bahan. Operasi tampil_pengeluaran () terdapat pada kelas Perawat yang memanggil 1 operasi lain dari kelas m_pengeluaran, yaitu tampil_pengeluaran () yang digambarkan pada Gambar 6.5.



Gambar 6.5 Arsitekur Komponen Uji

Tabel 6.7 menampilkan algoritme dari operasi daftarPengeluaran () merupakan salah satu operasi pada fungsi lihat pengeluaran.

Tabel 6.7 Algoritme daftarPengeluaran ()

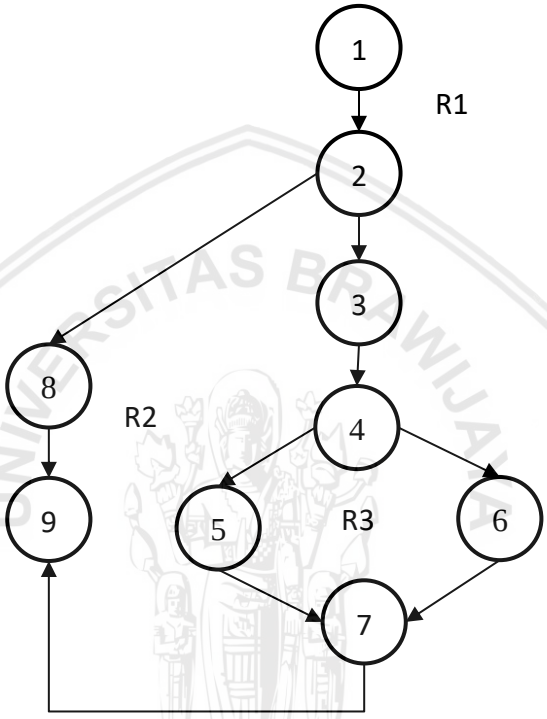
```
Perawat.php
start
   month <- date('m')
   year <- date('Y')

   if get['bulan'] and get['tahun']
      bulan <- get['bulan']
      tahun <- get['tahun']

      if bulan == 0 && tahun == 0
         data <- load model -> tampil_pengeluaran (month, year)
      else
         data <- load model -> tampil_pengeluaran (bulan, ta
      end if
   else
      data <- load model-> tampil_pengeluaran (month, year)
      load view -> v_pengeluaran(data)
   end if
end
```

1 2 3 4 5 6 7 8 9

Pada algoritme lihat pengeluaran bahan yang dijelaskan dalam Tabel 6.7 terdapat angka-angka yang menunjukkan bahwa algoritme tersebut memiliki node-node sebanyak angka, yaitu terdapat 9 node yang akan di ubah ke dalam bentuk *flow graph*. Gambar 6.6 merupakan *flow graph* dari algoritme daftarPengeluaran () yang akan dilakukan perhitungan *cyclometic complexity*. Berikut perhitungan *cyclometic complexity* pada *flow graph* algoritme daftarPengeluaran ().



Gambar 6.6 *Flow Graph* Algoritme daftarPengeluaran ()

V(G) = E – N + 2

= 10 – 9 + 2

= 3

V(G) = P + 1

= 2 + 1

= 3

V(G) = Jumlah region

= 3

Dari perhitungan *cyclometic complexity* didapatkan sebanyak 3 jalur independen, antara lain:

1. 1 – 2 – 8 – 9

2. 1 – 2 – 3 – 4 – 5 – 7 – 9

3. 1 – 2 – 3 – 4 – 6 – 7 – 9

Dari jalur independen yang sudah dibuat selanjutnya dilakukan pengujian melalui *Test Case* atau kasus uji di masing-masing jalurnya. *Test Case* pada tiap-tiap jalur independen dijelaskan pada Tabel 6.8.

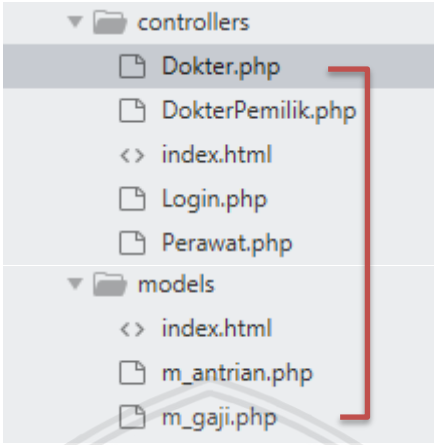
Tabel 6.8 *Test Case* Algoritme Lihat Pengeluaran Bahan

No	Jalur	Data Input	Hasil yang diharapkan	Hasil yang diperoleh	Status
1	1 – 2 – 8 – 9	month = 04 year = 2019	Sistem menampilkan daftar pengeluaran bahan sesuai dengan bulan dan tahun saat ini.	Tampil halaman daftar pengeluaran bahan dengan data pengeluaran bahan sesuai dengan bulan dan tahun saat ini.	Valid
2	1 – 2 – 3 – 4 – 5 – 7 – 9	bulan = 0 tahun = 0	Sistem menampilkan daftar pengeluaran bahan sesuai dengan bulan dan tahun saat ini.	Tampil halaman daftar pengeluaran bahan dengan data pengeluaran bahan sesuai dengan bulan dan tahun saat ini.	Valid
3	1 – 2 – 3 – 4 – 6 – 7 – 9	bulan = 06 tahun = 2018	Sistem menampilkan daftar pengeluaran bahan sesuai dengan bulan Juni dan tahun 2018.	Tampil halaman daftar pengeluaran bahan dengan data pengeluaran bahan sesuai dengan bulan Juni dan tahun 2018.	Valid

6.1.2.3 Lihat Gaji

Fungsi lihat gaji merupakan fungsi yang memungkinkan aktor untuk melihat daftar gaji. Operasi showGajiDokter () terdapat pada kelas Dokter yang memanggil

2 operasi lain dari kelas m_gaji, yaitu tampil_semuaGaji () dan tampil_gaji () ang digambarkan seperti Gambar 6.7.



Gambar 6.7 Arsitekur Komponen Uji

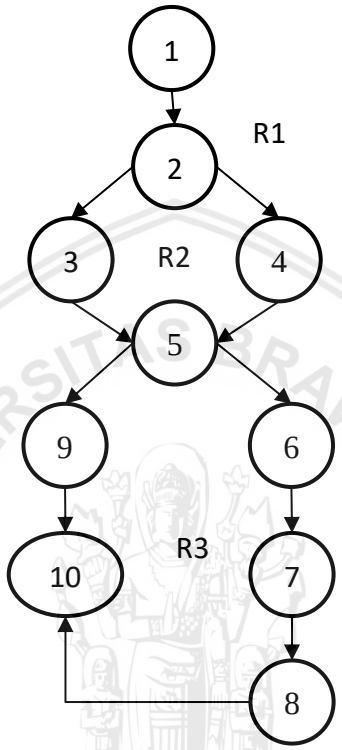
Tabel 6.9 menampilkan algoritme dari operasi showGajiDokter () yang merupakan salah satu operasi pada fungsi lihat gaji.

Tabel 6.9 Algoritme showGajiDokter ()

```
Dokter.php
start
   where <- userdata ('username')
   bulan <- get['bulan']                    1
   tahun <- get['tahun']

   if bulan == 0 && tahun == 0              2
      data <- load model -> tampil_semuaGaji(where)    3
   else
4     data <- load model -> tampil_gaji (where, bulan, tahun)
   end if  5
   if not empty (data)                      6
      foreach (data -> row)
         echo row->tanggal
         echo row->gaji                     7
         echo row->status
 8    end foreach
   else                                     9
      echo 'Tidak ada data'
   end if                                   10
end
```

Pada algoritme showGajiDokter () yang dijelaskan dalam Tabel 6.9 terdapat angka-angka yang menunjukkan bahwa algoritme tersebut memiliki node-node sebanyak angka, yaitu terdapat 10 node yang akan di ubah ke dalam bentuk *flow graph*. Gambar 6.8 merupakan *flow graph* dari algoritme showGajiDokter () yang akan dilakukan perhitungan *cyclometic*. Berikut perhitungan *cyclometic complexity* pada *flow graph* algoritme showGajiDokter ().



Gambar 6.8 *Flow Graph* Algoritme Lihat Gaji

V(G) = E – N + 2

= 11 – 10 + 2

= 3

V(G) = P + 1

= 2 + 1

= 3

V(G) = Jumlah region

= 3

Dari hasil perhiungan *cyclometic complexity* didapatkan sebanyak 3 jalur independen, antara lain:

1. 1 – 2 – 3 – 5 – 9 – 10

2. 1 – 2 – 4 – 5 – 6 – 7 – 8 – 10

3. 1 – 2 – 4 – 5 – 9 – 10

Dari jalur independen yang sudah dibuat selanjutnya dilakukan pengujian melalui *Test Case* atau kasus uji di masing-masing jalurnya. *Test Case* pada tiap-tiap jalur independen dijelaskan pada Tabel 6.10.

Tabel 6.10 *Test Case* Algoritme Lihat Gaji

No	Jalur	Data Input	Hasil yang diharapkan	Hasil yang diperoleh	Status
1	1 – 2 – 3 – 5 – 9 – 10	bulan = 0 tahun = 0 where = nadia	Sistem menampilkan daftar gaji sesuai dengan username nadia.	Tampil halaman daftar gaji dengan data gaji sesuai dengan username nadia.	Valid
2	1 – 2 – 4 – 5 – 6 – 7 – 8 – 10	bulan = 04 tahun = 2019 where = nadia	Sistem menampilkan daftar gaji sesuai dengan bulan April dan tahun 2019 serta sesuai dengan username nadia.	Tampil halaman daftar gaji dengan data gaji sesuai dengan bulan April dan tahun 2019 serta sesuai dengan username nadia.	Valid
3	1 – 2 – 4 – 5 – 9 – 10	bulan = 04 tahun = 1998 where = nadia	Sitsem menampilkan peringatan bahwa tidak ada data yang sesuai dengan bulan April tahun 1998 pada username nadia	Tampil halaman daftar gaji dan muncul peringatan “Tidak ada data.”	Valid

6.1.3 Pengujian Validasi

Pengujian validasi mengacu pada aktivitas untuk memastikan bahwa fungsional sistem memenuhi kebutuhan yang disebutkan dalam spesifikasi menggunakan skenario *test case*. Pengujian validasi pada penelitian ini

menggunakan teknik *requirement test* untuk melakukan pengujian *black-box testing.* Skenario pengujian yang dilakukan menggunakan *test case* normal dan *test case* alternatif. Hasil dari pengujian dengan menggunakan *test case* dari fungsional sistem dijabarkan pada Tabel 6.11.

Tabel 6.11 *Test Case* Fungsional Sistem

***Test Case Login* (POS-KF-001)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor memasukkan *username* dan *password* pada form login yang telah disediakan pada halaman *login*.	*Login* berhasil dilakukan dan sistem akan menampilkan halaman utama aktor sesuai hak akses.	*Login* berhasil dan tampil halaman utama aktor sesuai dengan akses aktor.	Valid
Kasus dan Hasil Uji (Kondisi Alternatif)			
Ada *field* atau semua *field* tidak di isi.	Sistem akan menampilkan peringatan bahwa *field* harus di isi.	Tampil peringatan pada *field* yang belum di isi bahwa *field* harus di isi.	Valid
Username dan *password* yang dimasukkan tidak terdaftar di database.	Sistem akan menampilkan pesan kesalahan "**Gagal *Login*!** Username/ Password yang anda masukkan salah".	Tampil peringatan bahwa kombinasi username dan password salah.	Valid
***Test Case Logout* (POS-KF-002)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada di dalam sistem memilih menu *logout,* kemudian melakukan konfirmasi *logout* dengan menekan tombol *logout*.	Aktor sudah berada di dalam sistem berhasil kerluar dari sistem dan sistem akan menampilkan halaman *login*	*Logout* berhasil, aktor berhasil keluar dari sistem dan sistem menampilkan halaman *login*.	Valid
Kasus dan Hasil Uji (Kondisi Alternatif)			

Test Case Logout (POS-KF-002)			
Aktor sudah berada di dalam sistem memilih menu *logout,* kemudian menekan tombol *cancel* pada modal konfirmasi *logout*.	Aktor batal keluar dari sisem.	Aktor batal keluar dari sisem dan sistem menampilkan halaman terakhir yang diakses oleh aktor.	Valid
***Test Case* Lihat Daftar Pasien (POS-KF-003)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada di dalam sistem memilih menu pasien.	Sistem menampilkan halaman daftar pasien.	Tampil halaman daftar pasien.	Valid
***Test Case* Lihat Daftar Antrian (POS-KF-004)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada di dalam sistem memilih menu antrian.	Sistem menampilkan halaman daftar antrian.	Tampil halaman daftar antrian.	Valid
***Test Case* Lihat Profil Pasien (POS-KF-005)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada di halaman pasien dan menekan tombol lihat pada daftar pasien yang diinginkan.	Sistem menampilkan halaman profil pasien.	Tampil halaman profil pasien.	Valid
***Test Case* Selesai Perawatan Pasien (POS-KF-006)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada di halaman profil pasien dan menekan tombol selesai, kemudian mengisi form untuk	Data selesai perawatan pasien berhasil di simpan dan sistem menampilkan	Tampil halaman halaman antrian pasien.	Valid

Test Case Selesai Perawatan Pasien (POS-KF-006)			
menambahkan diagnosa-terapi, pilihan perawatan, regio, dan keterangan.	halaman antrian pasien.		
Kasus dan Hasil Uji (Kondisi Alternatif)			
Ada *field* atau semua *field* tidak di isi.	Sistem akan menampilkan peringatan bahwa *field* harus di isi.	Tampil peringatan pada *field* yang belum di isi bahwa *field* harus di isi.	Valid
***Test Case* Lihat Stok Bahan (POS-KF-007)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada di dalam sistem memilih menu bahan.	Sistem menampilkan halaman daftar stok bahan.	Tampil halaman daftar stok bahan.	Valid
***Test Case* Lihat Pendapatan Pribadi (POS-KF-008)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada di dalam sistem memilih menu perndapatan pribadi.	Sistem menampilkan halaman daftar perndapatan pribadi.	Tampil halaman daftar perndapatan pribadi.	Valid
***Test Case* Lihat Pengeluaran Bahan (POS-KF-009)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada di dalam sistem memilih menu pengeluaran bahan.	Sistem menampilkan halaman daftar pengeluaran bahan.	Tampil halaman daftar pengeluaran bahan.	Valid
***Test Case* Tambah Stok Bahan (POS-KF-010)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada pada halaman pengeluaran bahan	Data stok bahan berhasil di simpan dan sistem	Tampil halaman daftar pengeluaran bahan dan tampil	Valid

Test Case Tambah Stok Bahan (POS-KF-010)			
dan menekan tombol tambah stok bahan, kemudian mengisi form untuk menambahkan stok bahan dengan memasukkan nama bahan, jumlah, dan tanggal.	menampilkan daftar pengeluaran bahan.	data stok bahan yang baru ditambahkan.	
Kasus dan Hasil Uji (Kondisi Alternatif)			
Ada *field* atau semua *field* tidak di isi.	Sistem akan menampilkan peringatan bahwa *field* harus di isi.	Tampil peringatan pada *field* yang belum di isi bahwa *field* harus di isi.	Valid
***Test Case* Tambah Bahan Baru (POS-KF-011)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada pada halaman pengeluaran bahan dan menekan tombol tambah bahan baru, kemudian mengisi form untuk menambahkan bahan baru dengan memasukkan nama bahan, stok minimal, satuan, perawatan, harga pokok, harga jual, *expired*, jumlah, dan tanggal.	Data bahan baru berhasil di simpan dan sistem menampilkan daftar pengeluaran bahan.	Tampil halaman daftar pengeluaran bahan dan tampil data bahan baru yang baru ditambahkan.	Valid
Kasus dan Hasil Uji (Kondisi Alternatif)			
Ada *field* atau semua *field* tidak di isi.	Sistem akan menampilkan peringatan bahwa *field* harus di isi.	Tampil peringatan pada *field* yang belum di isi bahwa *field* harus di isi.	Valid
***Test Case* Hapus Pengeluaran Bahan (POS-KF-012)**			
Kasus dan Hasil Uji (Kondisi Normal)			

Test Case Hapus Pengeluaran Bahan (POS-KF-012)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada pada halaman pengeluaran bahan dan menekan tombol hapus, kemudian melakukan konfirmasi hapus dengan menekan tombol *ok*.	Pengeluaran bahan yang dipilih aktor berhasil di hapus dan sistem menampilkan daftar pengeluaran bahan.	Tampil halaman daftar pengeluaran bahan tanpa data pengeluaran bahan yang sudah di hapus.	Valid
Kasus dan Hasil Uji (Kondisi Alternatif)			
Aktor sudah berada pada halaman pengeluaran bahan dan menekan tombol hapus, kemudian menekan tombol *cancel* pada modal konfirmasi hapus.	Sistem batal menghapus data pengeluaran bahan yang di pilih aktor.	Data batal di hapus dan tampil halaman daftar pengeluaran bahan.	Valid
***Test Case* Lihat Gaji (POS-KF-013)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada di dalam sistem memilih menu gaji.	Sistem menampilkan halaman daftar gaji.	Tampil halaman daftar gaji.	Valid
***Test Case* Tambah *Edit* Stok Bahan (POS-KF-014)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada pada halaman bahan dan menekan tombol *edit* pada stok bahan yang ingin di ubah, kemudian mengubah data stok bahan yang ingin di ubah pada	Data stok bahan berhasil di ubah dan sistem menampilkan daftar stok bahan.	Tampil halaman daftar stok bahan dan tampil data stok bahan yang di ubah ditambahkan.	Valid

Test Case Tambah *Edit* Stok Bahan (POS-KF-014)			
form edit stok bahan.			
***Test Case* Hapus Stok Bahan (POS-KF-015)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada pada halaman bahan dan menekan tombol hapus, kemudian melakukan konfirmasi hapus dengan menekan tombol *ok*.	Stok bahan yang dipilih aktor berhasil di hapus dan sistem menampilkan daftar stok bahan.	Tampil halaman daftar stok bahan tanpa data stok bahan yang sudah di hapus.	Valid
Kasus dan Hasil Uji (Kondisi Alternatif)			
Aktor sudah berada pada halaman stok bahan dan menekan tombol hapus, kemudian menekan tombol *cancel* pada modal konfirmasi hapus.	Sistem batal menghapus data stok bahan yang di pilih aktor.	Data batal di hapus dan tampil halaman daftar stok bahan.	Valid
***Test Case* Lihat Pendapatan Klinik (POS-KF-016)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada di dalam sistem memilih menu perndapatan klinik.	Sistem menampilkan halaman daftar perndapatan klinik.	Tampil halaman daftar perndapatan klinik.	Valid
***Test Case* Lihat Gaji Pegawai (POS-KF-017)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada di dalam sistem memilih menu gaji pegawai.	Sistem menampilkan halaman daftar gaji pegawai.	Tampil halaman daftar gaji pegawai.	Valid
***Test Case* Lihat Ubah Status Gaji Pegawai (POS-KF-018)**			
Kasus dan Hasil Uji (Kondisi Normal)			

Test Case Lihat Ubah Status Gaji Pegawai (POS-KF-018)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada pada halaman gaji pegawai, kemudian menekan tombol belum dibayarkan pada status gaji.	Berhasil mengubah status gaji pegawai dan status gaji pegawai menjadi "sudah dibayarkan".	Tampil halaman daftar gaji pegawai dan status gaji pegawai menjadi "sudah dibayarkan".	Valid
Test Case Tambah Pasien Baru (POS-KF-019)			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada pada halaman pasien dan menekan tombol tambah pasien baru, kemudian mengisi form untuk menambahkan pasien baru dengan memasukkan nama, tanggal lahir, alamat, pekerjaan, riwayat alergi, penyakit, dan no hp.	Data pasien baru berhasil di simpan dan sistem menampilkan daftar pasien.	Tampil halaman daftar pasien dan tampil data pasien baru yang baru ditambahkan.	Valid
Kasus dan Hasil Uji (Kondisi Alternatif)			
Ada *field* atau semua *field* tidak di isi.	Sistem akan menampilkan peringatan bahwa *field* harus di isi.	Tampil peringatan pada *field* yang belum di isi bahwa *field* harus di isi.	Valid
Test Case Tambah Antrian Pasien (POS-KF-020)			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada pada halaman pasien dan menekan tombol tambah antrian pasien, kemudian mengisi form untuk	Data antrian pasien berhasil di simpan dan sistem halaman antrian pasien.	Tampil halaman antrian pasien dan tampil data antrian pasien yang baru ditambahkan.	Valid

Test Case Tambah Antrian Pasien (POS-KF-020)			
menambahkan antrian pasien dengan memasukkan keluhan pasien dan pilihan dokter.			
Kasus dan Hasil Uji (Kondisi Alternatif)			
Ada *field* atau semua *field* tidak di isi.	Sistem akan menampilkan peringatan bahwa *field* harus di isi.	Tampil peringatan pada *field* yang belum di isi bahwa *field* harus di isi.	Valid

Test Case Cetak Kartu Berobat Pasien (POS-KF-021)			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada pada halaman pasien dan menekan tombol cetak pada daftar pasien yang ingin di cetak.	Aktor berhasil mencetak kartu berobat pasien yang dipilih.	Tampil halaman kartu berobat pasien dalam format pdf dan berhasil di cetak.	Valid

Test Case Cetak Nomor Antrian Pasien (POS-KF-022)			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada pada halaman antrian dan menekan tombol cetak pada antrian yang ingin di cetak.	Aktor berhasil mencetak nomor antrian pasien yang dipilih.	Tampil halaman nomor antrian pasien dalam format pdf dan berhasil di cetak.	Valid

Test Case Pembayaran Perawatan Pasien (POS-KF-023)			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada pada halaman antrian dan menekan tombol pembayaran, kemudian menambahkan	Data pembayaran perawatan pasien berhasil disimpan dan sistem menampilkan halaman rangkuman	Tampil halaman rangkuman pembayaran pasien.	Valid

Test Case Pembayaran Perawatan Pasien (POS-KF-023)			
bahan pada *cart* dan mengisi jumlah uang yang diberikan pasien pada *field* jumlah.	pembayaran pasien.		
***Test Case* Lihat Rangkuman Pembayaran (POS-KF-024)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada pada halaman pembayaran dan menekan tombol pembayaran.	Sistem menampilkan halaman rangkuman pembayaran.	Tampil halaman rangkuman pembayaran.	Valid
***Test Case* Cetak Kwintansi (POS-KF-025)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada pada halaman antrian dan menekan tombol cetak kwitansi pada pasien yang ingin di cetak.	Aktor berhasil mencetak kwitansi yang di pilih.	Tampil halaman kwitansi dalam format pdf dan berhasil di cetak.	Valid
***Test Case* Cetak Rekam Medis (POS-KF-026)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada pada halaman pasien dan menekan tombol cetak rekam medis pada pasien yang ingin di cetak.	Aktor berhasil mencetak rekam medis yang di pilih.	Tampil halaman rekam medis dalam format pdf dan berhasil di cetak.	Valid
***Test Case* Lihat Daftar Harga Bahan (POS-KF-027)**			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada di dalam sistem	Sistem menampilkan	Tampil halaman daftar harga bahan.	Valid

Test Case Lihat Daftar Harga Bahan (POS-KF-027)			
memilih menu bahan.	halaman daftar harga bahan.		

Test Case Lihat Daftar Biaya Perawatan (POS-KF-028)			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada di dalam sistem memilih menu perawatan.	Sistem menampilkan halaman daftar biaya perawatan.	Tampil halaman daftar biaya perawatan.	Valid

Test Case Atur Ulang Antrian Pasien (POS-KF-029)			
Kasus dan Hasil Uji (Kondisi Normal)			
Prosedur pengujian	**Hasil yang diharapkan**	**Hasil pengamatan**	**Kesimpulan**
Aktor sudah berada pada halaman antrian dan menekan tombol reset.	Sistem akan menghapus semua antrian yang ada.	Tampil halaman antrian pasien dan daftar antrian kosong.	Valid

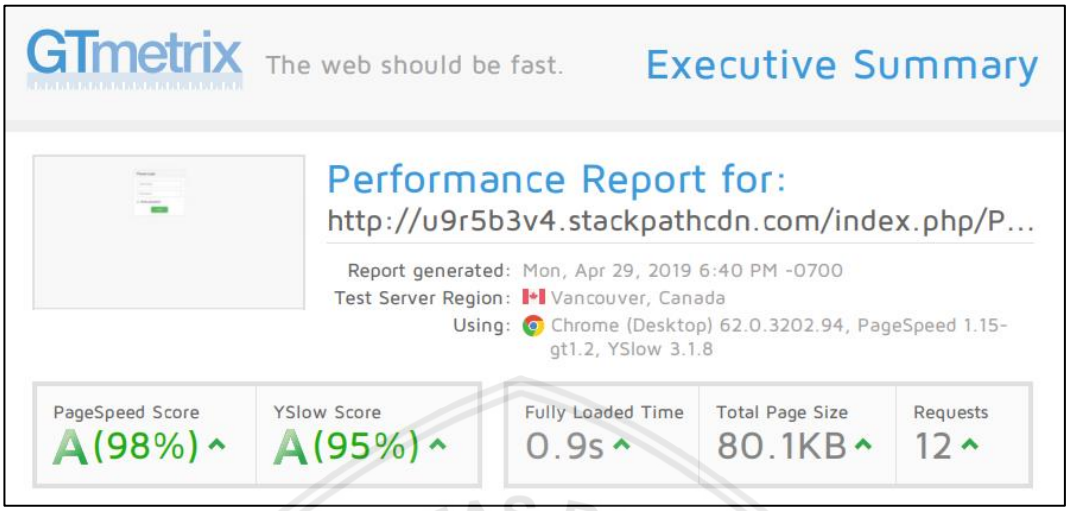
6.2 Pengujian Non Fungsional

Pengujian non fungsional dalam penelitian ini dilakukan dengan menguji *performance* sistem. Pengujian *performance* dilakukan menggunakan *tools* GTmetrix untuk melihat *performance* score dan rata-rata *load time* dari sistem informasi pengelolaan klinik gigi menggunakan *point of sale*. Lingkungan dari pengujian ini adalah pengujian dilakukan dengan *test server region* dari Vancouver, Canada menggunakan *browser* Chrome 62.0.3202.94 serta menggunakan *rules* dari Yslow 3.1.8 dan PageSpeed 1.15-gt1.2.

Pengujian *performance* dalam penelitian ini dilakukan dengan mengakses *website* Gtmetix mernggunakan Wi-Fi dengan kecepatan *download* 30.1 Mbps dan kecepatan *upload* 7.67 Mbps, kemudian memasukkan URL halaman perawat dari sistem informasi pengelolaan klinik gigi menggunakan *point of sale* pada *website* GTmetrix. GTmetrix akan melakukan analisis terhadap *speed performance*, *load time*, dan beberapa aspek lainnya, setelah analisis selesai GTmetrix akan menampilkan hasil dari pengujian.

Gambar 6.6 menjelaskan hasil analisis *performance* sistem menggunakan *tools* GTmetrix. Hasil dari pengujian menjelaskan bahwa *performance score* dari sistem informasi pengelolaan klinik gigi menggunakan *point of sale* menurut PageSpeed adalah 98% (A) serta menurut YSlow adalah 95% (A). Sedangkan rata-rata *load*

time yang didapatkan adalah 0.9 detik dengan *total page size* 80.1 KB dan 12 *requests*.



Gambar 6.9 Hasil Analisis *Performance* Menggunakan GTmetrix

Sehingga dapat disimpulkan bahwa pengujian *performance* pada sistem informasi pengelolaan klinik gigi menggunakan *point of sale* bernilai valid seperti yang dijelaskan pada Tabel 6.12.

Tabel 6.12 Hasil Pengujian *Performance*

***Performance* (POS-KNF-001)**		
Kasus dan Hasil Uji		
Hasil yang diharapkan	**Hasil pengamatan**	**Kesimpulan**
Sistem dapat memberikan *load time* kurang dari 3 detik.	Rata-rata load time yang didapatkan adalah 0,9 detik	Valid

6.3 Analisis Pengujian

Setelah proses pengujian selesai dilakukan, maka langkah selanjutnya adalah melakukan analisis pengujian. Adapun hasil pengujian yan akan di analisis yaitu, pengujian unit, pengujuan integrasi, pengujian validasi dan pengujian *performance*. Pengujian unit dilakukan dengan menguji 2 unit, yaitu tampil_pendapatan () dan tambah_pendapatanBersih (). Dari algoritme yang di buat pada keduanya menghasilkan total jalur independen yang berbeda. Unit tampil_pendapatan () memiliki jalur independen sebanyak 3 sedangkan pada unit tambah_pendapatanBersih () memiliki jalur independen sebanyak 2. Jalur independen tertinggi yang ditemukan pada pengujian unit yaitu 3, yang menandakan bahwa kompleksitas pada unit- unit tersebut tidak terlalu kompleks. Pada pengujian unit terdapat total 5 *test case,* yang mana semua *test case* tersebut bernilai valid.

Pengujian integrasi dilakukan dengan menguji 3 operasi, yaitu aksi_login () dan tampil_pengeluaran () dan showGajiDokter (). Dari algoritme yang di buat pada ketiganya menghasilkan total jalur independen yang berbeda. Operasi aksi_login () memiliki jalur independen sebanyak 4, sedangkan pada operasi tampil_pengeluaran () dan showGajiDokter () memiliki jalur independen sebanyak 3. Jalur independen tertinggi yang ditemukan pada pengujian integrasi yaitu 4, yang menandakan bahwa kompleksitas pada operasi-operasi tersebut tidak terlalu kompleks. Pada pengujian integrasi terdapat total 10 *test case,* yang mana semua *test case* tersebut bernilai valid.

Pengujian validasi dilakukan pada 29 fungsional sistem menggunakan *test case*. Semua hasil pengujian validasi pada sistem informasi pengelolaan klinik gigi menggunakan prinsip *point of sale* bernilai valid. Hal tersebut membuktikan bahwa sistem informasi pengelolaan klinik gigi menggunakan prinsip *point of sale* sudah siap untuk digunakan.

Pengujian *performance* yang dilakukan menggunakan *tools* GTmetrix mendapatkan hasil *performance score* 98% dan 95% dengan *grade* A dengan rata-rata *load time* sebesar 0.9 detik. Hasil tersebut menandakan bahwa tingkat *performance* dari sistem informasi pengelolaan klinik gigi menggunakan *point of sale* ini sangat baik serta dapat dikatakan bahwa sistem ini efisien, karena mampu meningkatkan rata-rata waktu yang dibutuhkan untuk menjalankan suatu aktivitas dari proses bisnis sebelum menggunakan sistem, salah satu contoh perbandingan waktu yang dibutuhkan pada proses bisnis sebelum dan sesudah menggunakan sistem dijelaskan pada Tabel 6.13.

Tabel 6.13 Perbandingan Waktu

Akivitas yang dikerjakan		**Rata-rata waktu yang dibutuhkan**		**Perbandingan**
As is	**To be**	**As is**	**To be**	
Melakukan pencarian kartu rekam medis pasien terdaftar sesuai dengan nama dan no pasien	Melakukan pencarian pasien terdaftar sesuai dengan nama dan no pasien	3 menit	6 detik	1:30

BAB 7 PENUTUP

7.1 Kesimpulan

Setelah melalui beberapa tahap pengembangan sistem, berakhir pada penarikan kesimpulan. Hasil dari penelitian ini diharapkan dapan menjawab semua rumusan masalah yang sudah dibahas pada bab 1. Kesimpulan yang dapat di ambil dari penelitian ini, antara lain:

1. Tahap rekyasa kebutuhan sistem informasi pengelolaan klinik gigi menggunakan prinsip *point of* menghasilkan 29 kebutuhan fungsional dan 1 kebutuhan non-fungsional yang mencakup fungsi untuk mengelola pasien, antrian, bahan, pendapatan dan pengeluaran. Kebutuhan fungsional sistem yang telah didapatkan, dimodelkan ke dalam bentuk *use case diagram* dan *use case scenario*.
2. Tahap perancangan kebutuhan sistem informasi pengelolaan klinik gigi menggunakan prinsip *point of* menghasilkan rancangan-rancangan sistem dalam bentuk *sequence diagram*, *class diagram*, *phsycal data model*, serta rancangan komponen dan antarmuka sistem. Sedangkan tahap implementasi sistem informasi pengelolaan klinik gigi menggunakan prinsip *point of* menghasilkan hasil dari implementasi kode program yang mengacu pada perancangan komponen, hasil dari implementasi *database* yang mengacu pada perancangan basis data, dan hasil implementasi antarmuka yang mengacu pada perancangan antarmuka.
3. Tahap pengujian kebutuhan sistem informasi pengelolaan klinik gigi menggunakan prinsip *point of* menghasilkan hasil uji dari pengujian unit, integrasi, validasi dan *performance*. Tingkat keberhasilan pada ketiga jenis pengujian tersebut adalah 100% valid. Sistem ini dapat meningkatkan efisiensi waktu 30 kali lebih cepat dibandingkan dengan sebelum menggunakan sistem pada aktivitas pencarian data pasien.

7.2 Saran

Berikut merupakan saran yang dapat diambil oleh pembaca jika ingin melanjutkan penelitian ini, antara lain:

1. Melakukan pengembangan lebih lanjut menggunakan platform android sehingga dapat dijalankan menggunakan *smartphone* atau *tablet* agar dapat menghemat biaya pembelian komputer.
2. Melakukan pengembangan lebih lanjut dengan memperbaiki fitur rekam medis agar sesuai dengan Standar Nasional Rekam Medik Gigi.

DAFTAR PUSTAKA

Cahyodi, S. C. & Arifin, R. W., 2017. Sistem Informasi Point Of Sales Berbasis Web Pada Colony Amaranta Bekasi. *Information System For Educators And Professionals,* 1(2), pp. 189-204.

Daqiqil, I., 2011. *Amar Bank.* [Online] Available at: https://www.amarbank.co.id/ [Accessed 1 Oktober 2018].

Fahrurroji, A., 2016. *Keuntungan Menggunakan Point-of-Sale Bagi Bisnis Anda.* [Online] Available at: https://afahrurroji.net/keuntungan-menggunakan-point-of-sale-bagi-bisnis-anda/
[Accessed 26 August 2018].

Gintoro, Andreyus, Emilia & William, R., 2010. *Analisis Dan Perancangan Sistem Pemesanan Tiket Dengan Teknologi Mobile.* Yogyakarta, Seminar Nasional Aplikasi Teknologi Informasi.

Jacobalis, S. et al., 2016. *Pedoman Praktik Dokter dan Dokter Gigi di Indonesia.* 1st ed. Jakarta Selatan: Konsil Kedokteran Indonesia.

Nielsen, J., 2010. *Website Response Times.* [Online] Available at: www.nngroup.com/articles/website-response-times/
[Accessed 20 April 2019].

Oktaviani, D., 2013. *P.O.S (Point of Sale).* [Online] Available at: https://pradiptadevie.wordpress.com/2012/05/16/p-o-s-point-of-sale/
[Accessed 21 August 2018].

Permana, S. D. H. & Faisal, 2015. Analisa dan Perancangan Aplikasi Point Of Sale (POS) untuk Mendukung Manajemen Hubungan Pelanggan. *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK),* 2(1), pp. 20-28.

Poluakan, K., 2017. *Perbedaan Sistem POS (Point of Sales) dan Sistem ERP (Enterprise Resource Planning).* [Online] Available at: https://ukirama.com/blogs/perbedaan-sistem-pos-point-of-sales-dan-sistem-erp-enterprise-resource-planning
[Accessed 15 October 2018].

Pressman, R. S., 2010. *Software Engineering: A Practitioner's Approach.* 7th ed. London: McGraw-Hill Education.

Pressman, R. S., 2012. *Rekayasa Perangkat Lunak (Pendekatan Praktisi).* 7 ed. Yogyakarta: Andi.

Rahman, H., 2013. *Mengenal Program Point Of Sales (POS).* [Online] Available at: http://blognyapuye.blogspot.com/2013/02/mengenal-program-point-of-sales-pos.html
[Accessed 21 August 2018].

Shimmura, T., Takenaka, T. & Akamatsu, M., 2009. *Real-Time Process Management System in a Restaurant by Sharing Food Order Information.* Malacca, Malaysia, 4-7 Dec. 2009, IEEE.

Sommerville, I., 2011. *Software Engineering.* 9th ed. United State: Pearson Education.

S, R. A. & Shalahuddin, M., 2013. *Rekayasa Perangkat Lunak Terstruktur Dan Berorientasi Objek.* Bandung: Informatika.

Susanto, A., 2017. *Sistem Informasi Manajemen Konsep dan Pengembangan Secara Terpadu.* Bandung: Lingga Jaya.