IMPLEMENTASI *WEB SERVICE* PADA SISTEM REKAM MEDIS TERPUSAT

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Bahtyar
NIM : 135150201111107

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

IMPLEMENTASI *WEB SERVICE* PADA SISTEM REKAM MEDIS TERPUSAT

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Bahtyar
NIM: 135150201111107

Skripsi ini telah diuji dan dinyatakan lulus pada
2 Agustus 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Nurudin Santoso, S.T., M.T	Faizatul Amalia, S.Pd.,M.Pd
NIP: 197409162000121001	NIK: 201309 860821 2 001

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 pustaka.

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, 07 Agustus 2018

Bahtyar

NIM: 135150201111107

KATA PENGANTAR

Puji syukur penulis panjatkan atas kehadirat Allah SWT. Karena berkat rahmat dan hidayah-Nya, penulis dapat menyelesaikan penyusunan skripsi dengan judul "Implementasi *Web Service* Pada Sistem Rekam Medis Terpusat" dengan cukup baik.

Dalam Penulisan skripsi ini, penulis banyak mendapatkan bantuan, bimbingan serta dukungan baik materil maupun secara moril dari berbagai pihak. Oleh karena itu dalam kesempatan ini penulis menyampaikan ucapan terimakasih kepada:

1. Orang tua penulis, Bapak Mohammad Baijuri dan Ibu Komariah serta keluarga besar yang selalu memberikan dukungan dan doa untuk kelancaran kuliah penulis.
2. Bapak Nurudin Santoso, S.T., M.T., selaku dosen pembimbing 1 yang selalu meluangkan waktu untuk membimbing, membagi pikiran dan pengalaman untuk penulis sehingga dapat menyelesaikan skripsi ini dengan baik.
3. Ibu Faizatul Amalia, S.Pd.,M.Pd., selaku dosen pembimbing 2 yang selalu meluangkan waktu dan pikiran untuk mengoreksi, mengarahkan dan membimbing penulis sehingga dapat menyelesaikan skripsi ini dengan baik.
4. Bapak Wayan Firdaus Mahmudy, S.Si, M.T, Ph.D., selaku Dekan Fakultas Ilmu Komputer.
5. Bapak Tri Astoto Kurniawan, S.T, M.T, Ph.D., selaku Ketua Jurusan Teknik Informatika.
6. Bapak Agus Wahyu Widodo, S.T, M.Cs., selaku Ketua Program Studi Teknik Informatika.
7. Seluruh dosen dan civitas Program Studi Teknik Informatika, Universitas Brawijaya atas dukungan dan kerjasamanya.
8. Teman-teman grup S.Kom yang telah memberikan semangat dan dukungan secara moril untuk menyelesaikan penulisan skripsi.
9. Sahabat dan teman seperjuangan Fakultas Ilmu Komputer dalam menyelesaikan penulisan skripsi lainnya yang tidak dapat disebutkan satu persatu.

Malang, 8 Agustus 2018

Bahtyar

bahtyar11@gmail.com

ABSTRAK

Bahtyar, Implementasi *Web Service* Pada Rekam Medis Terpusat

Dosen Pembimbing: Nurudin Santoso, S.T., M.T., Faizatul Amalia, S.Pd.,M.Pd.

Pencatatan rekam medis pada umumnya ditulis dan disimpan pada masing-masing unit fasilitas kesehatan untuk menjaga privasi dari informasi kesehatan pasien. Perpindahan berobat pasien pada fasilitas kesehatan mengakibatkan fasilitas kesehatan baru harus melakukan pendataan ulang pasien berobat. Jika dalam keadaan darurat kondisi ini akan mengurangi waktu penanganan darurat pasien. Sistem rekam medis terpusat akan menjembatani keterbatasan informasi ini. Dengan memanfaatkan teknologi *REST web service* sistem ini dapat mengolah dan menyimpan informasi secara terpusat. Sehingga diharapkan informasi catatan riwayat kesehatan pasien dari fasilitas kesehatan sebelumnya dapat diketahui. Dari pengujian *white box* yaitu dengan pengujian unit sistem rekam medis terpusat mendapatkan hasil 87,5%, sedangkan untuk pengujian *black box* dengan pengujian validasi mendapatkan hasil 100%. Pengujian *web service* dilakukan menggunakan Aplikasi Postman untuk menguji *method-method* pada *service* seperti *GET*, *POST*, *PUT* dan *DELETE* dan menghasilkan tingkat keberhasilan 100%. Berdasarkan rata-rata total persentase keberhasilan pengujian fungsional jika dilakukan konversi ke dalam predikat tabel kelayakan *functional suitability* maka sistem yang dikembangkan dinyatakan sangat baik.

Kata kunci: rekam medis, *web service*, sistem rekam medis terpusat, *REST web service*

ABSTRACT

Bahtyar, The Implementation of Web Service In Centralized of Medical Record

Supervisor: Nurudin Santoso, S.T., M.T., Faizatul Amalia, S.Pd.,M.Pd.

The recording of medical records is generally written and stored in each health facility unit to maintain the privacy of the patient's health information. Displacement of patient treatment at health facility resulted in new health facility must be perform patient data rechecking. If in an emergency this condition will reduce the patient's emergency response time. The centralized of medical record system will bridge the limitations of this information. By utilizing REST technology the web service system can process and store information centrally. So it is expected that the records of medical record patient's from previous health facilities can be known. From white box testing that is by unit testing of centralized medical record system get result 87,5%, while for black box testing with validation test get 100% result. Web service testing is done using the Postman Application to test methods on services such as GET, POST, PUT and DELETE and yields a 100% success rate. Based on the average total percentage of successful functional testing if the conversion into predicate table of functional suitability feasibility of the system developed declared as very good.

Keywords*: medical record, web service, the centralized of medical record system, REST web service*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Rekam medis pada umumnya merupakan berkas atau dokumen yang berisikan informasi identitas pasien, pemeriksaan, pengobatan, tindakan dan pelayanan lain yang telah diberikan kepada pasien (Permenkes, 2008). Tujuan diselenggarakannya rekam medis ialah untuk membantu administrasi fasilitas pelayanan kesehatan agar lebih tertib dan meningkatkan pelayanan kesehatan pada tiap fasilitas pelayanan kesehatan. Rekam medis juga digunakan untuk membangun sistem analisis pada dokumen setiap kegiatan pelayanan kesehatan, baik ketika menerima pasien, pencatatan, pengolahan data, penyimpanan serta pengambilan kembali rekam medis dan pelaporan. Kegunaan dari rekam medis diantaranya adalah sebagai alat komunikasi antara dokter dan tenaga kesehatan lainnya yang ikut ambil bagian dalam memberikan pelayanan, pengobatan dan perawatan kepada pasien dan juga sebagai bukti atas segala tindakan pelayanan, perkembangan penyakit dan pengobatan selama pasien berkunjung atau dirawat di fasilitas pelayanan kesehatan.

Perpindahan tempat berobat pasien dari fasilitas pelayanan kesehatan satu ke fasilitas pelayanan kesehatan lain mengakibatkan jejak rekam medis pasien pada fasilitas sebelumnya tidak diketahui, akibatnya dalam situasi darurat ketika pasien dirujuk pada fasilitas kesehatan yang berbeda, untuk mengetahui riwayat kesehatan pasien maka perlu menunggu seseorang yang mengenal pasien untuk memberikan data lengkap (Budiono, 2017). Informasi rekam medis pasien pada saat ini masih disimpan di masing-masing fasilitas pelayanan kesehatan itu sendiri, sedangkan jika sebuah fasilitas pelayanan kesehatan memiliki layanan informasi rekam medis terpusat yang dapat diakses beberapa fasilitas pelayanan kesehatan lain yang terhubung didalamnya, maka akan lebih mudah dengan sistem yang terhubung internet untuk mencari informasi jejak riwayat kesehatan pasien dari tempat berobat sebelumnya dibandingkan dengan menunggu seseorang yang mengenal pasien untuk memberikan informasi.

Web service adalah sebuah *interface* atau antarmuka yang menggambarkan sekumpulan operasi-operasi yang dapat diakses melalui jaringan, misalnya internet dalam bentuk pesan *Extensible Markup Language (XML)* (Kreger, 2001). *Web service* umumnya digunakan sebagai *website* yang melayani aplikasi-aplikasi independen yang mendukung aplikasi tersebut untuk kebutuhan integrasi data atau bertukar informasi. *Web service* menyimpan data informasi dalam format *XML dan JavaScript Object Notation (JSON)*, sehingga data dapat diakses oleh sistem lain walaupun berbeda *platform*, sistem operasi, maupun *compiler*. *Web-service* dapat diimplementasikan sebagai layanan pertukaran informasi rekam medis terpusat, karena dengan adanya *web service* dapat menjadi solusi sebagai jembatan informasi rekam medis antar fasilitas pelayanan kesehatan.

Berdasarkan masalah yang tercantum seperti pada di atas maka dibuatlah Implementasi *Web Service* pada Sistem Rekam Medis Terpusat yang mampu

mengintegrasikan sistem informasi rekam medis pada tiap fasilitas pelayanan kesehatan. Diharapkan dengan adanya layanan seperti ini dapat membantu fasilitas pelayanan kesehatan dalam mencari jejak rekam medis pasien dari fasilitas berobat sebelumnya dan data informasi dapat disimpan secara terpusat sehingga dapat dimanfaatkan sebagai layanan satu atap pada daerah tertentu atau nasional.

1.2 Rumusan Masalah

1. Bagaimana cara untuk mengetahui jejak riwayat berobat pasien dari fasilitas pelayanan kesehatan sebelumnya ketika dalam kondisi darurat?
2. Bagaimana arsitektur komunikasi di dalam layanan sistem informasi rekam medis terpusat?
3. Bagaimana menguji implementasi *web service* sistem informasi rekam medis terpusat?

1.3 Tujuan

Tujuan dari penelitian adalah untuk:

1. Mengetahui jejak riwayat berobat pasien dari dari fasilitas kesehatan sebelumnya ketika dalam kondisi darurat.
2. Mengetahui arsitektur layanan sistem informasi rekam medis terpusat.
3. Mengetahui hasil pengujian dari implementasi *web service* sistem informasi rekam medis terpusat.

1.4 Manfaat

1. Membantu fasilitas pelayanan kesehatan untuk mendapatkan informasi rekam medis pasien dari berbagai fasilitas pelayanan kesehatan dalam keadaan darurat.
2. Memberikan solusi layanan satu atap kepada pemerintah dalam sistem informasi rekam medis terhadap fasilitas-fasilitas pelayanan kesehatan.

1.5 Batasan Masalah

1. Pengembangan sistem menggunakan model *Software Development Lifecycle* (SDLC) *waterfall*.
2. Implementasi sistem hanya sampai menghasilkan *prototype*.
3. Pencatatan data pasien menggunakan standar prosedur Puskesmas Dinoyo dan Puskesmas Kendalsari Kota Malang.

1.6 Sistematika Pembahasan

BAB 1 PENDAHULUAN

Pendahuluan berisi penjelasan mengenai latar belakang masalah, rumusan masalah, tujuan penelitian, manfaat penelitian dan batasan masalah.

BAB 2 LANDASAN KEPUSTAKAAN

Landasan kepustakaan membahas mengenai kajian pustaka dan teori-teori dasar yang digunakan dalam menunjang penelitian.

BAB 3 METODOLOGI

Metode penelitian berisi pembahasan mengenai metode yang digunakan dalam penelitian dan terdiri dari studi pustaka, pengumpulan data, analisis kebutuhan, perancangan sistem, implementasi sistem, pengujian dan analisis dan pengambilan kesimpulan dan saran.

BAB 4 ANALISIS KEBUTUHAN

Analisis kebutuhan berisi elisitasi kebutuhan, identifikasi pemangku kepentingan, teknik elisitasi, spesifikasi kebutuhan, *Diagram use case* dan *use case scenario*.

BAB 5 PERANCANGAN DAN IMPLEMENTASI

Perancangan sistem berisi penjelasan mengenai perancangan sistem secara detail meliputi perancangan arsitektur, perancangan *sequence diagram*, perancangan *class diagram*, perancangan basis data, perancangan algoritme dan perancangan antarmuka. Implementasi berisi penjelasan mengenai implementasi dari perancangan yang sudah dibuat sebelumnya.

BAB 6 PENGUJIAN DAN ANALISIS

Berisi penjelasan mengenai proses dan hasil pengujian terhadap sistem untuk memastikan bahwa sistem yang telah dibangun berjalan sesuai perencanaan dan tujuan yang diharapkan.

BAB 7 PENUTUP

Penutup berisi kesimpulan dari penelitian yang telah dilakukan dan saran bagi penelitian selanjutnya untuk mendapatkan hasil yang lebih optimal.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Tinjauan Pustaka

Pada tinjauan pustaka menjelaskan mengenai pemilihan topik atau judul yang diangkat sebagai penelitian. Tinjauan pustaka berisi tentang penjelasan mengenai topik penelitian yang pernah ada dan berbanding relevan dengan topik yang saat ini dilakukan. Terdapat penelitian sebelumnya mengenai Perancangan Sistem Informasi Rekam Medis Pasien Elektronik Terpusat (Studi Kasus: Kota Madya Denpasar) yang dilakukan oleh I Made Swasta Adiputra dan hasil penelitian Pengembangan Sistem Rekam Medis Pasien Rawat Jalan Pada Puskesmas oleh Edi Setiawan.

Pada penelitian yang dilakukan oleh Adiputra (2012) menjelaskan bahwa setiap pasien yang berobat ke fasilitas kesehatan yang berbeda mengeluhkan pertanyaan perawat dan dokter yang selalu sama setiap pasien datang berkunjung ulang untuk berobat. Diagnosa ulang atau mengulas ulang keluhan pasien ini banyak dikeluhkan oleh pasien yang sebenarnya tenggang waktu berobat pasien tidak cukup lama. Dari masalah tersebut Adiputra melakukan perancangan sistem yang memanfaatkan arsitektur *database* sentralisasi sehingga data pada setiap penyimpanan fasilitas kesehatan akan dipusatkan dalam konteks *database* terdistribusi.

Sedangkan penelitian yang dilakukan oleh Edi Setiawan (2016) meneliti tentang pencatatan data rekam medis pada puskesmas dengan berlatar belakang bahwa tempat penelitian yang dilakukan masih memiliki proses pencatatan dan penyimpanan data rekam medis secara manual. Penelitian yang dilakukan oleh Edi Setiawan mendapatkan hasil akhir berupa implementasi program rekam medis rawat jalan agar petugas lebih mudah melakukan pendataan dan mengurangi kesalahan selama pendataan.

Dari hasil kedua penelitian di atas maka dilakukan percobaan untuk menggabungkan masalah yang dihadapi, sehingga mendapatkan hasil yaitu Implementasi *web service* sistem informasi rekam medis terpusat. Rekam medis terpusat lebih memusatkan fungsi untuk berbagi informasi antar fasilitas kesehatan agar dalam keadaan darurat pencarian data dapat dilakukan dengan lebih baik dan terpercaya. Pengembangan yang dilakukan akan memanfaatkan teknologi *client-server* dengan implementasi *web service* menggunakan bahasa pemrograman *Hypertext Preprocessor* (*PHP*) dan menggunakan format file *JSON* sebagai format standar pengiriman dokumen agar dapat dipakai oleh banyak *platform*. Metode pengembangan yang diterapkan adalah waterfall. Pengembangan menggunakan bahasa *PHP*, *HTML5* dan *CSS* dengan memanfaatkan *MySQL* sebagai media penyimpanan data.

2.2 Landasan Teori

2.2.1 Fasilitas Pelayanan Kesehatan

Pada peraturan pemerintah Republik Indonesia Nomor 47 tahun 2016 tentang fasilitas pelayanan kesehatan menjelaskan bahwa fasilitas pelayanan kesehatan merupakan suatu alat dan/atau tempat yang digunakan untuk menyelenggarakan upaya pelayanan kesehatan, baik promotif, preventif kuratif maupun rehabilitatif yang dilakukan oleh pemerintah pusat, pemerintah daerah, dan/atau masyarakat (RI, 2016). Penyelenggaraan pelayanan pada fasilitas pelayanan kesehatan dapat berupa pelayanan kesahatan perorangan dan/atau pelayanan kesehatan masyarakat. Jenis-jenis fasilitas pelayanan kesehatan diantaranya sebagai berikut.

- Tempat praktik mandiri Tenaga Kesehatan;
- Pusat Kesehatan Masyarakat;
- Klinik;
- Rumah Sakit;
- Apotek;
- Unit Transfusi Darah;
- Laboratorium kesehatan;
- Optikal.

Pada penelitian ini fasilitas yang dipilih adalah Pusat Kesehatan Masyarakat (Puskesmas) yang dapat dijangkau diantaranya yaitu Puskesmas Dinoyo Kota Malang dan Unit Pelaksana Teknis Daerah (UPDT) Puskesmas Kendalsari Kota Malang.

2.2.2 Sistem Informasi (SI)

Suatu sistem yang terdiri dari beberapa komponen untuk bekerja sama membentuk satu kesatuan merupakan definisi dari sistem. Setiap sistem tidak peduli seberapa kecil ukurannya pasti mengandung komponen-komponen atau subsistem. Sedangkan informasi merupakan hasil pengolahan data dalam suatu bentuk yang lebih berguna dan lebih berarti bagi penerimanya yang menggambarkan suatu kejadian-kejadian yang nyata yang digunakan untuk pengambilan keputusan (Aljufri, 2013). Informasi memiliki beberapa kualitas diantaranya sebagai berikut.

1. Akurat

   Akurat berarti informasi harus bebas dari kesalahan dan tidak biasa atau mengelabui. Akurat juga berarti informasi harus jelas mencerminkan maksudnya. Informasi harus akurat karena dari sumber informasi sampai ke penerima informasi kemungkinan banyak terjadi gangguan yang dapat mengubah atau merusak informasi tersebut.

2. Tepat pada waktu

Informasi yang diterima tidak boleh terlibat, harus sesuai dengan waktu yang ditentukan. Informasi yang sudah lama tidak akan mempunyai nilai atau kehilangan *value.* Bila suatu informasi datang terlambat maka pengambilan keputusan akan tidak tepat dan mengakibatkan masalah yang fatal.

3. Relevan

Informasi yang diterima mempunyai manfaat untuk pemakainya, relevansi informasi untuk tiap-tiap orang satu dengan yang lainnya berbeda-beda.

Pada sistem informasi rekam medis, sistem informasi merupakan komponen penting bagi tiap fasilitas kesehatan karena membutuhkan kualitas informasi yang sangat tinggi demi mencapai tujuan dan manfaat.

2.2.2.1 Komponen sistem informasi

Terdapat beberapa komponen dalam sistem informasi diantaranya sebagai berikut (Aljufri, 2013).

1. Perangkat keras *(Hardware)*

Komponen perangkat sistem informasi keras berupa komponen komputer secara fisik yang terdiri dari:

a. Unit peralatan *input* yaitu peralatan yang digunakan untuk menerima *input* atau memasukkan data ke dalam komputer antara *keyboard*, *disk drive* dan disket.

b. Unit peralatan proses yaitu alat yang memiliki beragam instruksi-instruksi program diproses untuk mengolah data yang sudah dimasukkan lewat alat *input* dan hasilnya akan ditampilkan di alat *ouput*.

c. Unit peralatan *ouput* yaitu alat yang digunakan untuk memindahkan atau mentransfer data dari dalam komputer ke dalam bentuk fisik permanen yaitu dengan *printer*.

2. Perangkat lunak *(Software)*

Perangkat lunak atau *software* merupakan sistem yang ditanamkan pada perangkat keras agar dapat membantu operasional atau kegiatan pada perangkat keras untuk mencapai tujuan tertentu. *Software* digunakan untuk melengkapi perangkat keras, bentuk *software* dibagi menjadi tiga yaitu:

a. Program aplikasi,

b. Sistem operasi,

c. Bahasa pemrograman.

2.2.3 Rekam Medis

Rekam medis adalah berkas yang berisikan catatan, dan dokumen tentang identitas pasien, pemeriksaan, pengobatan, tindakan dan pelayanan lain kepada pasien pada sarana pelayanan kesehatan (Pemkes, 2017). Rekam medis adalah kompilasi dari fakta-fakta yang relevan berkaitan dengan riwayat kesehatan pasien dari dulu hingga sekarang, diagnosis, pengobatan dan hasil akhir dari setiap perawatan. Para profesional rekam medis harus memastikan bahwa semua yang diisi relevan dengan fakta yang ada dan bukan rekayasa.

Tujuan utama dari rekam medis adalah untuk memberikan informasi yang akurat mengenai sejarah kesehatan pasien, dimulai dari masa lalu hingga saat ini, pengobatan yang telah diberikan dan kejadian-kejadian pada pasien selama masa perawatan. Informasi yang terkandung di dalam rekam medis memberikan kegunaan tersendiri untuk masing-masing pihak. Adapun nilai rekam medis bagi pihak tersebut adalah:

a. Bagi pasien, menyediakan bukti asuhan keperawatan, merupakan data untuk pengobatan selanjutnya dan memberikan perlindungan hukum dalam kasus-kasus tertentu.
b. Bagi fasilitas layanan kesehatan, memiliki data untuk pekerja tenaga medis, bukti untuk tagihan pembayaran, mengevaluasi sumber daya, mengevaluasi mutu pelayanan, dan membantu dalam membuat perencanaan dan pemasaran.
c. Bagi pemberi pelayanan, menyediakan informasi untuk membantu seluruh tenaga medis, membantu dokter dalam menyediakan data perawatan dan sebagai data untuk penelitian.

Dalam Peraturan Menteri Kesehatan No. 749 a tahun 1989 menyebutkan bahwa Rekam Medis memiliki lima manfaat, yaitu:

1. Sebagai dasar pemeliharaan kesehatan dan pengobatan pasien,
2. Sebagai bahan pembuktian dalam perkara hukum,
3. Bahan untuk kepentingan penelitian,
4. Sebagai dasar pembayaran biaya pelayanan kesehatan,
5. Sebagai bahan untuk menyiapkan statistik kesehatan.

Bagian-bagian dalam penyelenggaraan unit rekam medis dibagi menjadi beberapa bagian diantaranya sebagai berikut (Weningsih, 2016).

a. Pendaftaran pasien

   Proses untuk mengumpulkan data pasien dengan lengkap dan akurat untuk keperluan statistik dan juga metode dalam menghitung semua pertemuan rawat jalan dan rawat inap.

b. Identifikasi pasien

Identifikasi pasien dilakukan untuk menggolongkan pasien ke kategori unik yang memungkinkan setiap pasien mendapatkan perawatan yang berbeda-beda.

c. Indeks utama pasien

Indeks merupakan poin penting pada setiap rumah sakit, klinik dan fasilitas kesehatan primer. Indeks digunakan untuk menentukan lokasi *item* dapat berupa *table*, berkas, atau katalog daftar *item* yang memudahkan untuk akses pada lokasi tiap *item*.

d. Sistem Penomoran

Sistem penomoran rekam medis dimaksudkan untuk mengurutkan nomor pasien yang masuk ke dalam lembaga fasilitas kesehatan, beberapa jenis penomoran dibagi menjadi berikut.

- Sistem nomor seri

  Sistem nomor seri diberikan kepada pasien setiap berkunjung ke lembaga fasilitas kesehatan dan nomor tersebut selalu baru. Pada penomoran sistem nomor seri biasanya digunakan pada fasilitas kesehatan skala kecil dengan tingkat kunjungan kembali yang rendah.

- Sistem nomor Unit

  Sistem nomor unit merupakan penomoran yang digunakan dalam merekam riwayat rawat inap maupun rawat jalan milik pasien. Setiap pasien yang berkunjung hanya akan diberikan satu nomor rekam medis baik dalam rawat inap maupun rawat jalan.

- Sistem nomor seri unit

  Sistem nomor seri unit penggabungan dari nomor unit dan nomor seri, setiap pasien yang berkunjung diberikan nomor baru tetapi nomor rekam medis yang terdahulu digabungkan dan disimpan dengan nomor yang paling baru sehingga tetap satu unit.

Pada hasil observasi dan wawancara di Puskesmas Kendalsari dan Puskesmas Dinoyo, penomoran yang digunakan menggunakan sistem nomor unit dengan mengidentifikasi tempat asal pasien dan nomor urut pendaftaran. Penomoran yang digunakan dijelaskan seperti pada berikut.

XX-YYZZZZ

Keterangan :

XX : Merupakan seri nomor yang menjelaskan lokasi pasien tinggal.

YY : Merupakan seri nomor unit fasilitas kesehatan.

ZZZZ : Merupakan seri nomor pendaftaran pasien.

e. Distribusi catatan kesehatan

Distribusi catatan adalah bagian yang sangat penting dari pegangan rekaman. Hal ini memang tidak cukup untuk menekankan pengambilan yang cepat dari permintaan yang mendesak tanpa menekankan kebutuhan transportasi rekam medis dengan cepat.

f. Sistem pengarsipan

Sistem identifikasi catatan dan pengarsipan harus berjalan dengan bersamaan, sebagai sistem pengarsipan tergantung pada sistem identifikasi yang digunakan. Pengarsipan adalah susunan sistematis catatan dalam urutan sehingga referensi dan pengambilan mudah dan cepat.

g. Sistem penyimpanan

1. Sentralisasi

Catatan pasien yang disimpan dalam satu lokasi. Pasien mungkin memiliki catatan yang berbeda (catatan rawat inap, catatan darurat dan catatan perawatan rawat jalan), tetapi catatan dibawa bersama-sama dalam satu *unit record*, atau setidaknya *field under* nomor yang sama di tempat yang sama. Tujuan utama dari departemen catatan kesehatan adalah untuk mempertahankan catatan medis terus menerus dari pasien, yang tersedia setiap saat. Cara terbaik untuk mencapai tujuan ini adalah untuk membangun sistem unit rekam medis terpusat.

Dalam sistem satuan catatan terpusat, sentralisasi mengacu pada pengajuan rawat inap, rawat jalan dan gawat darurat catatan pasien di satu lokasi. Idealnya, untuk kontrol yang baik, semua informasi medis tentang pasien harus disimpan dalam satu *folder*, dalam satu lokasi atau file. Hal ini membuat pengambilan informasi yang mudah karena diajukan di satu tempat di bawah satu nomor.

2. Desentralisasi

Catatan pasien yang diajukan di beberapa tempat perawatan pasien di bawah unit yang sama nomor atau dengan angka sama sekali tidak terkait. Ini adalah kebijakan yang tepat untuk menjaga ruang penyimpanan di bawah pengawasan ketat oleh profesional informasi kesehatan catatan manajemen / kesehatan.

h. Kodefikasi

Fungsi ICD *(International Classification of Diseases)* sebagai sistem klasifikasi penyakit dan masalah terkait kesehatan digunakan untuk kepentingan informasi statistika morbiditas dan mortalitas. Morbiditas dalam artian sederhana merupakan ukuran peristiwa-peristiwa sakit atau kesakitan. Sedangkan mortalitas merupakan salah satu dari tiga komponen demografi yang dapat memengaruhi perubahan penduduk. Penerapan pengodean sistem ICD digunakan untuk:

- Mengindeks pencatatan penyakit dan tindakan di sarana pelayanan kesehatan,
- Masukan bagi sistem pelaporan diagnosis medis,
- Memudahkan proses penyimpanan dan pengambilan data terkait diagnosis karakteristik pasien dan penyedia layanan,
- Bahan dasar dalam pengelompokan DRGs (diagnosis-related groups) untuk sistem penagihan pembayaran biaya pelayanan,
- Pelaporan nasional dan internasional morbiditas dan mortalitas,
- Tabulasi data pelayanan kesehatan bagi proses evaluasi perencanaaan pelayanan medis,
- Menentukan bentuk pelayanan yang harus direncanakan dan dikembangkan sesuai kebutuhan zaman,
- Analisis pembiayaan pelayanan kesehatan,
- Untuk penelitian epidemiologi dan klinis.

i. Indeks Data Pelayanan Pasien

Pelayanan penting dari departemen rekam medis adalah pengambilan rekam medis khusus yang mungkin dibuat menurut penggunaan daftar data penyakit, operasi, dan indeks dokter. Banyak rumah sakit, mempunyai banyak informasi lengkap mengenai penyakit dan terapi yang dimasukkan dalam rekam medis tetapi jarang dibuka dan hanya disimpan di rak penyimpanan. Kepala departemen rekam medis dapat melakukan banyak untuk mengurangi situasi ini, kepala derpartemen dapat menginformasikan kepada anggota staf medis dan rumah sakit bagian perawatan indeks dalam departemennya dan memperkenalkan kepada staf akan potensi informasi-informasi penting tersebut. Kemudian informasi tersebut dipisahkan sehingga menjadi sebuah indeks agar ketika pencarian informasi rekam medis lebih mudah untuk dilakukan. Banyak dokter dan personel rumah sakit hanya menyadari hal tersebut, namun tak banyak yang memahami.

Dokter atau staf komite medis menggunakan indeks penyakit dan operasi untuk pengambilan rekam medis dengan tujuan:

- Untuk melihat kembali kasus sebelumnya dari penyakit yang diberikan dalam permintaan untuk menyediakan pengetahuan/ wawasan ke dalam pengelolaan masalah kesehatan pasien saat ini,
- Untuk menguji teori dan membandingkan data pada penyakit tertentu dan/atau terapi dalam rangka untuk mempersiapkan karya ilmiah,
- Untuk mendapatkan data tentang pemanfaatan telah dipakai rumah sakit dan untuk membangun keperluan rumah sakit untuk

peralatan baru, tempat tidur dan lainnya dalam berbagai departemen,

- Untuk mengevaluasi kualitas pelayanan di rumah sakit.

Kegunaan lain dari indeks adalah sebagai berikut.

- Menyediakan data pelayanan pasien yang diperlukan untuk peninjauan akreditasi,
- Menempatkan rekaman jika dokter hanya mengingat diagnosis dan/atau operasi dan bukan nama pasien,
- Menyediakan data mengenai praktik pengobatan medis di rumah sakit dalam rangka kualitas untuk terakreditasi magang program dan residen,
- Menyediakan bahan pembelajaran untuk mahasiswa medis, mahasiswa keperawatan dan lainnya.

Perencanaan yang teliti akan hasil dalam sebuah sistem pengindeksan dapat memengaruhi kualitas layanan staf medis dan staf rumah sakit menjadi lebih baik serta dapat menaikkan efisiensi dari pengoperasian pengindeksan itu sendiri.

j. Statistik

Definisi yang digunakan untuk pengumpulan data statistik pemanfaatan rumah sakit bervariasi dari satu fasilitas ke fasilitas lain. Indeks yang digunakan untuk mengkategorikan sebuah rekam medis bervariasi berdasarkan indeks penyakit, indeks unit fasilitas kesehatan, indeks dokter dan lainnya.

k. Laporan

Penting untuk memastikan bahwa laporan ini disusun secara tepat waktu dan akurat karena langsung atau tidak langsung, ini adalah salah satu cara departemen catatan kesehatan dievaluasi. Laporan yang membandingkan data dan indikator yang dipilih selama periode waktu yang berbeda mungkin berguna. Sehingga sebelum melanjutkan untuk mengumpulkan atau menghitung informasi statistik apapun, catatan kesehatan profesional harus mencari tahu apa yang dibutuhkan, bagaimana dan kapan itu akan digunakan.

Laporan yang dihasilkan juga sangat penting dan digunakan sebagai alat komunikasi. Semua presentasi harus sederhana dan mudah dibaca dengan fokus pada fakta-fakta penting. Meskipun sebagian besar laporan akan dalam bentuk tabel, tabel tersebut akan lebih mudah untuk dibaca jika alat bantu visual seperti grafik, grafik batang dan *diagram pie* yang digunakan untuk menggambarkan dengan jelas bentuk angka-angka.

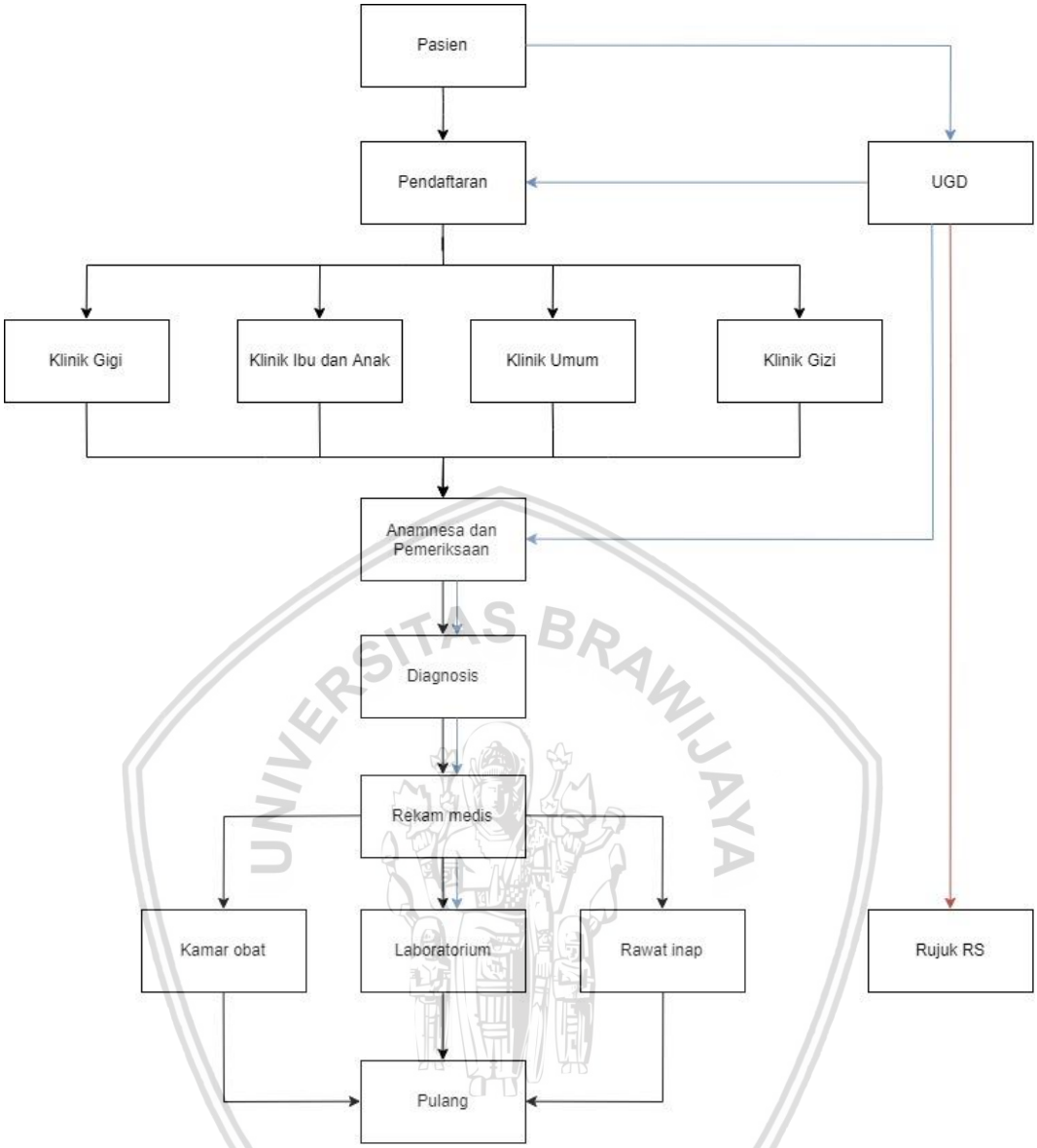
l. Retensi Rekam Kesehatan

Retensi berarti transfer catatan dari aktif untuk penyimpanan tidak aktif, untuk mikrofilm, atau akhirnya kehancuran. Hal ini berlaku umum bahwa catatan kesehatan harus disimpan selama catatan sedang digunakan untuk perawatan pasien, dan tujuan penelitian dan pengajaran medikolegal (berkaitan dengan ilmu kesehatan dan hukum). Jika ruang tidak tersedia, keputusan harus dibuat untuk menentukan panjang catatan waktu harus disimpan dalam file aktif. Pada studi kasus yang diambil dari Puskesmas Dinoyo batas lama retensi rekam medis adalah selama dua tahun, setelah pasien tidak pernah berobat ke puskesmas kembali, maka nomor rekam medis milik pasien akan masuk ke bank nomor, dan staf rekam medis akan menunggu publikasi surat dari kepala departemen untuk memusnahkan rekam medis milik pasien yang sudah tidak aktif.

2.2.3.1 Blok *Diagram* Alur Pasien Berobat

Blok *diagram* adalah alur yang memiliki beberapa tahapan dan proses dari pasien awal mendaftar hingga menjadi data rekam medis. Berdasarkan hasil observasi yang dilakukan telah dilakukan di Puskesmas Dinoyo didapatkan alur informasi rekam medis adalah sebagai berikut (Budiono, 2017).

1. Pasien mendaftar pada loket pendaftaran fasilitas pelayanan kesehatan,
2. Jika pasien sudah terdaftar maka petugas akan melakukan pencarian kartu rekam medis pasien,
3. Pasien akan menerima kartu rekam medis dan menunggu antrian di klinik tujuan pasien berobat,
4. Dokter pada setiap klinik akan memanggil pasien berdasarkan urutan antrian,
5. Dokter akan melakukan anamnesa kepada pasien dan melakukan pemeriksaan,
6. Dokter akan menuliskan diagnosis pasien pada kartu rekam medis,
7. Pasien akan diberikan resep obat oleh dokter,
8. Kartu rekam medis akan disimpan kembali oleh petugas rekam medis.



Gambar 2.1 Blok diagram alur rekam medis pasien (Sumber: Budiono, 2017)

Pada gambar 2.1 diawali dari pasien melakukan pendaftaran, kemudian setelah melakukan pendaftaran pasien meunggu di klinik tujuan pasien akan berobat. Setelah masuk ke dalam klinik maka dokter akan melakukan anamnesa dan pemeriksaan. Anamnesa merupakan kegiatan wawancara terhadap pasien atau keluarga pasien untuk memperoleh keterangan keluhan yang diderita oleh pasien. Kemudian dokter akan menetapkan diagnosis kepada pasien dan akan ditulis dalam kartu rekam medis pasien. Jika pasien dalam kondisi yang sangat kurang baik maka pasien akan dirawat inap, namun jika kondisi pasien masih cukup baik maka dokter memberikan resep obat dan pasien akan mengambil obat. Setelah mengambil obat pasien dapat pulang ke rumah. Hasil dari rekam medis akan dibawa ke laboratorium dan disimpan. Namun jika pasien dalam keadaan

darurat dan puskesmas tidak mampu untuk menanganinya maka langkah alternatif pasien akan segera dirujuk ke rumah sakit.

2.2.3.2 *Work Breakdown Structure* (WBS) Rekam Medis Terpusat

Work breakdown structure atau WBS merupakan pengelompokan dalam orientasi produk dari elemen kerja proyek yang mengatur dan mendefinisikan ruang lingkup total proyek (Devi, 2012). WBS berupa kerangka *multi-level* untuk menampilkan elemen yang mewakilkan tiap pekerjaan yang harus diselesaikan secara grafis.

Pada pengembangan sistem rekam medis pusat yang akan dikembangkan, didapatkan beberapa data untuk merekam informasi rekam medis yang dapat dibagi kepada setiap fasilitas kesehatan. Beberapa *scope project* digambarkan pada WBS untuk melihat fungsi-fungsi pada sistem yang akan dikerjakan diantaranya adalah:

1. Daftar berobat

Daftar berobat merupakan tahap awal pasien mendaftar berobat pada fasilitas kesehatan. Setiap pasien yang belum pernah berobat sebelumnya diwajibkan untuk mendaftarkan identitas lengkap untuk memberikan informasi diri dan juga dapat menentukan penomoran pada rekam medis. Data diantaranya adalah identitas pribadi pasien. Jika pasien pernah mendaftarkan diri dan atau pernah berobat sebelumnya, maka pencarian informasi dapat dilakukan pada sistem rekam medis.

2. Mencatat rekam medis

Pencatatan dilakukan untuk mendokumentasikan setiap riwayat data kesehatan yang diderita oleh pasien dan untuk memastikan diagnosis pasien serta pelayanan yang akan diberikan. Dalam pencatatan rekam medis terdapat tiga bagian *scope* lainnya di dalamnya yaitu daftar rekam medis, data dan resume.

a. Daftar rekam medis, berisikan daftar rekam medis pasien berupa *list* agar dapat dilakukan pencarian rekam medis ketika keadaan darurat. Terdapat fungsi detail informasi untuk melihat informasi lengkap rekam medis yang dibutuhkan.
b. Data, catatan rekam medis memiliki data atau *record* yang harus atau perlu diisi untuk melengkapi informasi kesehatan yang diderita oleh pasien diantaranya adalah anamnesa, pemeriksaan umum, pemeriksaan fisik, pernafasan, kardiovaskuler, persyarafan, perkemihan, pencernaan, muskuloskleletal, sistem reproduksi, dpss (data psikologi, sosiologi dan spiritual), pemeriksaan lab dan data petugas yang merawat pasien.
c. Resume, merupakan catatan akhir atau hasil pemeriksaan pasien yang telah dijalani selama perawatan pasien atau terapi. Catatan resume pada umumnya lebih dari satu ketika melakukan rawat jalan untuk meneruskan informasi perkembangan kesehatan pasien selama melakukan terapi. Dalam *scope* resume terdapat beberapa komponen yaitu daftar resume

untuk melihat daftar terapi yang pernah dilakukan dan detail resume untuk melihat informasi lengkap catatan resume.

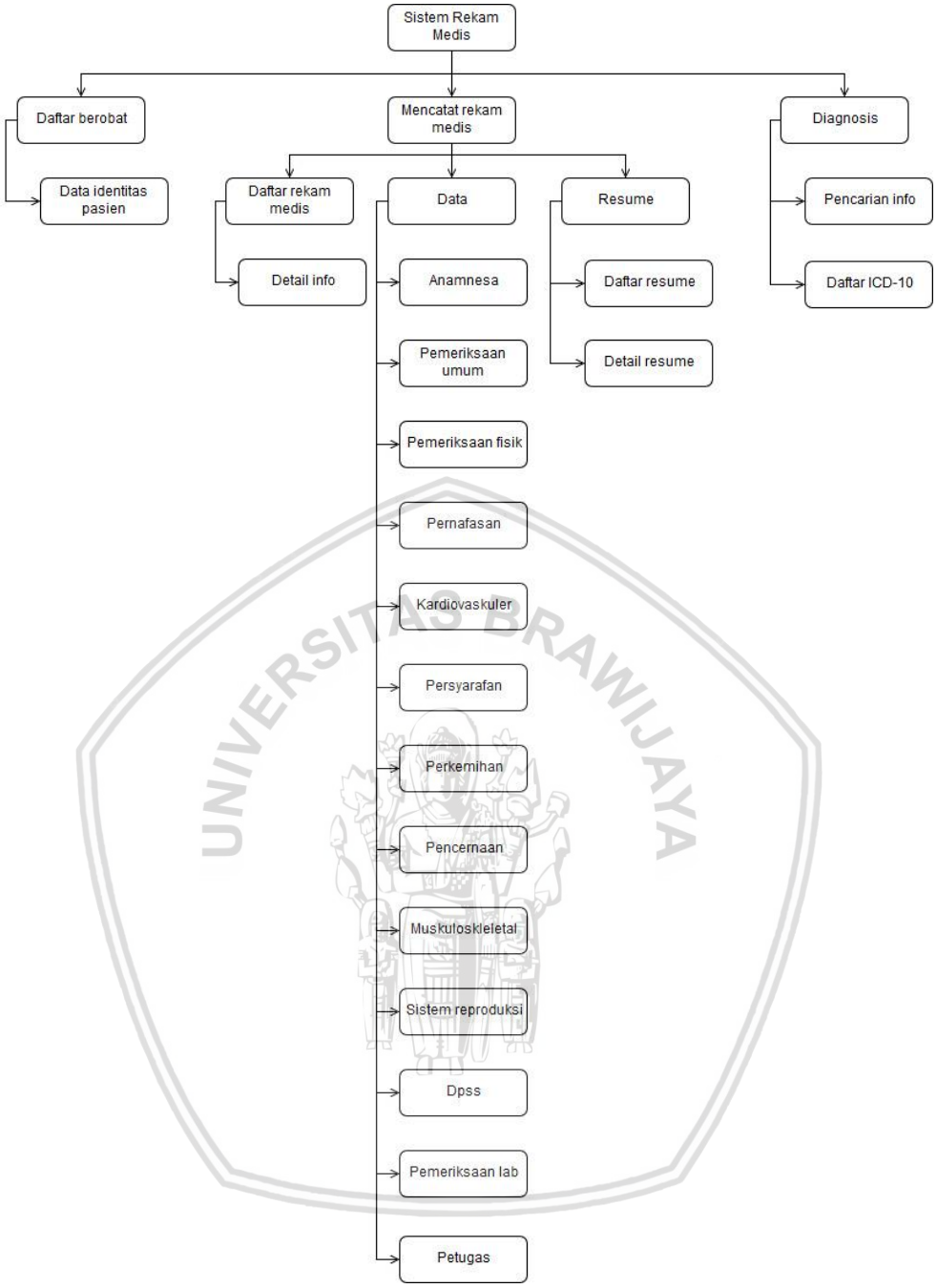
3. Diagnosis

Diagnosis berisi mengenai informasi daftar lengkap diagnosis dengan standar ICD-10. ICD atau *International Statistical Classification of Diseases* adalah buku mengenai pengkodean atas penyakit dan tanda-tanda, gejala, temuan-temuan yang abnormal, keluhan, keadaan sosial dan eksternal menyebabkan cedera atau penyakit, seperti yang diklasifikasikan oleh *World Health Organization* (WHO). Terdapat beberapa komponen dalam *scope* ini diantaranya adalah pencarian info untuk mencari informasi data diagnosis berdasarkan kategori penyakit standar ICD dan daftar ICD untuk menampilkan beberapa data dalam bentuk daftar. Selengkapnya dapat dilihat pada gambar 2.2.

2.2.3.3 Sistem Informasi Rekam Medis Terpusat

Sistem rekam medis terpusat merupakan sarana penyimpanan data rekam medis secara sentralisasi. Data rekam medis dapat dijangkau oleh aplikasi yang dikembangkan oleh berbagai *platform* karena dalam pengembangannya diimplementasikan menggunakan teknologi *web service*. Sehingga memungkinkan fasilitas kesehatan dapat saling berbagi informasi data rekam medis pasien guna memberi tindakan dalam keadaan darurat yang dialami pasien.

Dari hasil observasi atau pengamatan yang telah dilakukan didapatkan informasi mengenai belum adanya teknologi atau layanan yang mampu menghubungkan informasi data rekam medis dari fasilitas kesehatan satu ke fasilitas kesehatan lain, sehingga jejak kesehatan pasien tidak mudah untuk diketahui sebelum meminta surat keterangan dari fasilitas berobat sebelumnya (Nurul, 2017). Dengan tujuan untuk memberikan jembatan informasi guna berbagi data rekam medis pasien maka diusulkan sistem informasi rekam medis terpusat dengan memanfaatkan teknologi *web service* dan basis *website* sebagai *platform* pengembangannya. Fungsi-fungsi yang dimiliki oleh sistem tersebut dapat dilihat pada tabel 2.1.



Gambar 2.2 WBS rekam medis terpusat

Tabel 2.1 Fungsi-fungsi sistem informasi rekam medis terpusat

No	Aktor	Fungsi	Penjelasan
1	Pengguna	Daftar Berobat	Melihat daftar pasien berobat
2		*Edit* Data Berobat	*Edit* data profil pasien
3		Tambah Daftar Pasien Berobat	Menambahkan daftar berobat pasien baru

Tabel 2.1 Fungsi-fungsi sistem informasi rekam medis terpusat (lanjutan)

No	Aktor	Fungsi	Penjelasan
4	Pengguna	Lihat RM Pasien	Melihat data rekam medis pasien berdasarkan daftar berobat
5		Tambah RM	Menambahkan data rekam medis baru
6		*Edit* RM	*Edit* data rekam medis pasien
7		Tambah Anamnesa	Menambah data anamnesa
8		*Edit* Anamnesa	Mengubah data anamnesa
9		Tambah Pemeriksaan Umum	Menambahkan data pemeriksaan umum
10		*Edit* Pemeriksaan Umum	Mengubah data pemeriksaan umum
11		Tambah Pemeriksaan Fisik	Menambahkan data pemeriksaan fisik
12		*Edit* Pemeriksaan Fisik	Mengubah data pemeriksaan fisik
13		Tambah Pernafasan	Menambahkan data pernafasan
14		*Edit* Pernafasan	Mengubah data pernafasan
15		Tambah Kardiovaskuler	Menambahkan data kardiovaskuler
16		*Edit* Kardiovaskuler	Mengubah data kardiovaskuler
17		Tambah Sistem Persyarafan	Menambahkan data sistem persyarafan
18		*Edit* Sistem Persyarafan	Mengubah data persyarafan
19		Tambah Perkemihan	Menambah data perkemihan
20		*Edit* perkemihan	Mengubah data perkemihan
21		Tambah Sistem Pencernaan	Menambah data sistem pencernaan
22		*Edit* Sistem Pencernaan	Mengubah data sistem pencernaan
23		Tambah Sistem Moskuloskleletal	Menambahkan data mukuloskleletal

Tabel 2.1 Fungsi-fungsi sistem informasi rekam medis terpusat (lanjutan)

No	Aktor	Fungsi	Penjelasan
24	Pengguna	*Edit* Muskuloskleletal	Mengubah data muskuloskleletal
25		Tambah Sistem Reproduksi	Menambahkan data sistem reproduksi
26		*Edit* Reproduksi	Mengubah data reproduksi
27		Tambah Data Psikologi, Sosiologi dan Spiritual	Menambahkan data kondisi psikologi, sosiologi dan spiritual pasien
28		*Edit* Data Psikologi, Sosiologi dan spiritual	Mengubah data psikologi, sosiologi dan spiritual
29		Tambah Data Pemeriksaan Penunjang	Menambahkan data hasil pemeriksaan oleh penunjang
30		*Edit* Pemeriksaan Penunjang	Mengubah data pemeriksaan penunjang
31		Tambah Petugas	Menambahkan data petugas yang mengurus pasien
32		*Edit* Petugas	Mengubah data petugas
33		Resume	Melihat data resume medis pasien
34		Tambah Resume	Menambahkan data resume medis
35		*Edit* Resume	Mengubah data resume
36		Hapus Data	Menghapus seluruh *record* data pasien
37		Cari Diagnosis	Mencari informasi data diagnosis berdasarkan kategori ICD-10
38		Tambah Diagnosis	Menambahkan data diagnosis baru
39		*Edit* Diagnosis	Mengubah data diagnosis
40		Hapus Diagnosis	Menghapus data diagnosis
41		*Upload* berkas	Mengunggah berkas data rekam medis pasien ekstensi .csv/.pdf/.doc
42		*Download* berkas	Mengunduh berkas data rekam medis pasien

Pada tabel 2.1 di atas tedapat fungsi-fungsi yang telah diberikan penjelasan masing-masing perannya, fungsi tersebut juga memiliki atribut yang mewakili data-data. Data-data pada atribut tersebut didapatkan dari hasil wawancara dan formulir rekam medis di salah satu fasilitas kesehatan yang bersedia memberikan formulir tersebut. Berikut deskripsi sebagian fungsi dengan atribut yang dimiliki diantaranya:

1. Atribut fungsi Daftar Berobat

   Atribut fungsi daftar berobat dapat dilihat pada tabel 2.2 berikut.

Tabel 2.2 Atribut fungsi Daftar Berobat

No	Nama Atribut	Keterangan
1	Nama lengkap	Nama lengkap pasien
2	NIK/KK	Nomor NIK atau KK pasien
3	Jenis Kelamin	Jenis kelamin pasien
4	Tempat Lahir	Tempat asal lahir pasien
5	Tanggal Lahir	Tanggal lahir pasien
6	Alamat	Alamat rumah tetap pasien
7	Status	Status hubungan pasien
8	Pekerjaan	Pekerjaan pasien
9	Jabatan	Jabatan yang dimiliki pasien
10	Lama Kerja	Lama pekerjaan pasien
11	Agama	Agama pasien
12	Suku	Suku asal pasien, biasanya memiliki keunikan pada jumlah gigi
13	Telepon	Nomor telepon pasien yang dapat dihubungi
14	Jenis Pasien	Jenis penunjang berobat pasien (misalkan BPJS atau jaminan kesehatan lain)
15	Rumah sakit	Nama rumah sakit tempat pasien mendaftar berobat pertama

2. Atribut fungsi Rekam Medis

   Atribut yang dimiliki pada tabel Rekam Medis dapat dilihat pada tabel 2.3 berikut.

Tabel 2.3 Atribut fungsi Rekam Medis

No	Nama Atribut	Keterangan
1	Nomor RM	Nomor rekam medis milik pasien

Tabel 2.3 Atribut fungsi Rekam Medis (lanjutan)

No	Nama Atribut	Keterangan
2	Nama	Nama pasien
3	Jenis kelamin	Jenis kelamin pasien
4	Umur	Umur pasien
5	Ruang	Ruangan pasien dirawat
6	Alamat	Alamat Paisen
7	Nama RS	Nama rumah sakit pasien dirawat
8	Tanggal MRS	Tanggal pasien masuk rumah sakit
9	Jam	Menunjukkan waktu pasien mrs
10	Poli	Nama poli tempat pasien melakukan pemeriksaan (khususnya rawat jalan)

3. Atribut fungsi Anamnesa
   Atribut yang dimiliki pada tabel anamnesa dapat dilihat pada tabel 2.4 berikut.

Tabel 2.4 Atribut fungsi Anamnesa

No	Nama Atribut	Keterangan
1	Keluhan utama	Merupakan keluhan yang diderita pasien sehingga menyebabkan kondisi kesehatan pasien memburuk
2	Riwayat penyakit terdahulu	Merupakan kondisi penyakit terdahulu yang pernah diderita pasien
3	Riwayat penyakit dari keluarga	Merupakan kondisi penyakit yang diturunkan dari garis keturunan
4	Riwayat pekerjaan	Merupakan kondisi pekerjaan yang pernah dilakukan oleh pasien
5	Riwayat alergi	Merupakan kondisi alergi yang dimiliki pasien

4. Atribut fungsi Pemeriksaan Umum
   Atribut yang dimiliki pada tabel pemeriksaan umum dapat dilihat pada tabel 2.5 berikut.

Tabel 2.5 Atribut fungsi Pemeriksaan Umum

No	Nama Atribut	Keterangan
1	Keadaan umum	Merupakan kondisi keadaan pasien tampak dari luar

Tabel 2.5 Atribut fungsi Pemeriksaan Umum (lanjutan)

No	Nama Atribut	Keterangan
2	Pemeriksaan	Merupakan pemeriksaan yang dilakukan kepada pasien
3	Pemeriksaan penunjang sebelumnya	Penunjang pemeriksaan sebelumnya yang pernah dilakukan
4	Pemeriksaan penunjang saat ini	Penunjang pemeriksaan saat ini
5	Diagnosis kerja	Penetapan diagnosis awal yang belum diikuti dengan pemeriksaan yang lebih mendalam
6	Diagnosis banding	Sejumlah diagnosis (lebih dari 1) dengan kemungkinan yang ada untuk menetapkan diagnosis lebih lanjut
7	Perencanaan pelayanan	Tindakan yang akan diberikan kepada pasien

5. Atribut fungsi Pemeriksaan Fisik
   Atribut yang dimiliki pada tabel pemeriksaan fisik dapat dilihat pada tabel 2.6 berikut.

Tabel 2.6 Atribut fungsi Pemeriksaan Fisik

No	Nama Atribut	Keterangan
1	Kesadaran	Kondisi kesadaran pasien
2	GCS (*Eye*)	Nilai tingkat kesadaran terdapat tiga parameter (*Eye*)
3	GCS (*Verbal*)	Nilai tingkat kesadaran terdapat tiga parameter (*Verbal*)
4	GCS (*Motor*)	Nilai tingkat kesadaran terdapat tiga parameter (*Motor*)
5	Suhu	Suhu tubuh pasien
6	Respirasi	Pernapasan pasien
7	Nadi	Banyak denyut nadi pasien per menit
8	Tekanan darah	Tekanan darah pasien

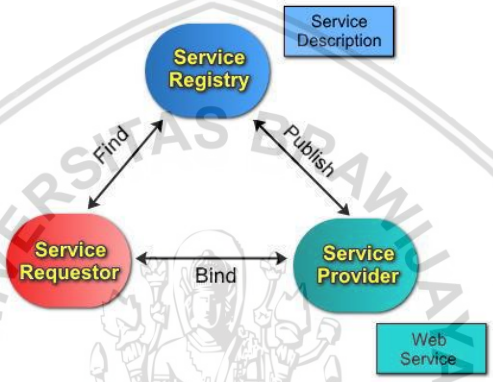
2.2.4 *Web Service*

Web service merupakan sebuah antarmuka yang menggambarkan sekumpulan *method* atau operasi yang dapat diakses melalui pesan standar *XML* (Kreger, 2001). *Web service* didefinisikan dengan format standar *XML* dan *JSON* yang mencakup seluruh layanan untuk berinteraksi dengan layanannya, termasuk format pesan, protokol *transport* dan lokasi. *Web service* adalah sistem perangkat

lunak yang dirancang untuk mendukung interaksi antara mesin dan mesin melalui jaringan (Suyanto, 2007). Sistem lain berinteraksi dengan layanan *web* dengan cara yang ditentukan oleh deskripsinya menggunakan pesan *SOAP (Simple Object Access Protocol)*, biasanya disampaikan menggunakan *HTTP (Hypertext Transfer Protocol)* dengan serialisasi *XML* bersamaan dengan standar *web* lainnya. Namun pada implementasi yang akan dilakukan menggunakan standar pesan *JSON* dikarenakan pengiriman data lebih cepat dan ringan dibandingkan dengan format *XML*.

2.2.4.1 Arsitektur *Web Service*

Arsitektur *web service* didasarkan pada interaksi tiga peran yaitu penyedia layanan, layanan registrasi dan peminta layanan (Kreger, 2001). Interaksi



Gambar 2.3 Arsitektur *web service*

melibatkan mempublikasikan, menemukan dan mengikat operasi. Bersamaan peran dan operasi ini bertindak berdasarkan artefak dari *web service* yaitu modul perangkat lunak *web service* dan deskripsinya.

Pada gambar 2.3 (Kurniawan, 2014) terdapat tiga komponen dalam arsitektur *web service* diantaranya adalah:

- *Service provider*, merupakan *provider* atau yang menyediakan layanan dalam sudut pandang bisnis. Sedangkan dalam sudut pandang perspektif arsitektural, penyedia layanan (*service*) merupakan sebuah *platform* yang menjadi tuan rumah untuk akses sebuah layanan.
- *Service requestor (client)*, merupakan sisi yang membutuhkan dan menggunakan sebuah layanan dalam sudut pandang bisnis atau bisa disebut juga klien dari *web service*. Sedangkan di sudut pandang arsitektural, *service requestor* merupakan yang melakukan pencarian dan melakukan permohonan untuk memulai interaksi dengan sebuah layanan. Peran peminta layanan dapat dimainkan oleh *browser* yang digerakkan oleh seseorang atau program tanpa antarmuka pengguna, misalnya layanan *web* lain.
- *Service registry*, merupakan tempat *service provider* mempublikasikan layanannya. Pada arsitektur *web service*, *Service registry* bersifat opsional.

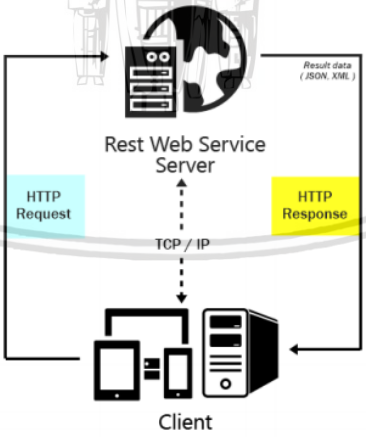
Teknologi *web service* memungkinkan pengembang dapat menghubungkan macam-macam aplikasi dengan *platform* dan sistem operasi yang berbeda.

2.2.4.2 REST (Representational State Transfer)

Representational State Transfer (REST) adalah gaya arsitektur perangkat lunak untuk sistem *hypermedia* terdistribusi seperti *world wide web*. *REST* sering digunakan untuk menguraikan penghubung sederhana yang mengirimkan data spesifik domain di atas *HTTP* tanpa lapisan *messaging* tambahan seperti *SOAP. SOAP* dan *REST* menyediakan fungsionalitas yang sama tetapi *request* dan *response* dari *web service* berbasis *SOAP* harus ditulis dalam format *SOAP* dan kemudian dikemas dalam pesan *HTTP*, sedangkan *REST web service* tidak menggunakan format *SOAP* dan hanya menggunakan *HTTP* sebagai protokol layer aplikasi (Susanto, 2017). Metode *REST* dilandasi oleh empat prinsip dasar utama, yaitu:

1. URI (*Resource Indentifier through Uniform Resource Identifier*),
2. *Uniform Interface*, sumber daya yang dimanipulasi adalah *CRUD* (*Create*, *Read*, *Update*, *DELETE*) dengan menggunakan operasi standar *HTTP* yaitu *PUT*, *GET*, *POST*, *DELETE*.
3. *Selft-descriptive message*, sumber daya informasi yang tidak terkait, sehingga dapat mengakses berbagai format konten (*HTML*, *PDF*, *JPEG*, *XML*, *Plain Text* dan lainnya).
4. *Stateful interaction though hyperlink*, setiap interaksi dengan suatu sumber daya bersifat *stateless*, yaitu *request message* tergantung jenis kontennya.

Interaksi *REST web service* dapat dilihat pada gambar 2.4 berikut (Susanto, 2017).



Gambar 2.4 Interaksi *REST Web Service*

Gambar 2.4 di atas menjelaskan bagaimana *REST web service* berkomunikasi dengaan *client* mengirim sebuah *request* kemudian *request* tersebut dikirim melalui *HTTP Request* dan sampai pada *REST server*. Kemudian *server* membalasnya dengan mengirimkan hasil dari permintaan *client* melalui *HTTP response* berupa *JSON*.

Pada penelitian yang dilakukan oleh Damar Riyadi (2013) dengan topik rancang bangun *rest web service* untuk perbandingan harga pengiriman dengan

metode *web scrapping* dan pemanfaatan API menunjukkan penggunaan teknologi *rest web service* pada implementasinya. *Service* yang digunakan untuk mendapatkan data ongkos kirim dan layanan kurir sehingga dapat membandingkan harga pengiriman dari TIKI, JNE dan POS Indonesia secara terpadu. Pada pembangunan sistemnya komunikasi yang digunakan pada *client* memanfaatkan *PHP*/*cURL* atau *client URL Request Library*. *cURL* merupakan pustaka yang juga disediakan oleh banyak pengembang pada *framework CodeIgniter* untuk membantu pengembangan komunikasi antara *client* dan *server* pada *rest web service*.

Kemudian untuk melakukan pengujian *rest web service* dapat menggunakan *tool,* yaitu Aplikasi Postman seperti penelitian yang dilakukan oleh Dian Setiana (2016) untuk mengetahui fungsi-fungsi atau *method* pada *web service* berjalan dengan seharusnya. Postman merupakan aplikasi yang memiliki fungsi sebagai *rest client* untuk melakukan pengujian pada *Rest* API yang telah dibuat (Postman, 2018). Pada pengujian yang dilakukan oleh Dian menggunakan postman terdapat *method-method* yang diuji pada *web service* seperti *method GET*, *POST*, *PUT* dan *DELETE*.

2.2.5 *System Development Life Cycle (SDLC)*

System Development Life Cycle (SDLC) atau siklus hidup sistem dalam rekayasa perangkat lunak merupakan sebuah alur proses atau metodologi yang digunakan untuk mengembangkan sistem (Bender, 2003). Dalam SDLC terdapat banyak tipe diantaranya seperti *Waterfall*, *Rapid*, *Prototype* dan lainnya.

Dalam rekayasa sistem rekam medis terpusat SDLC yang digunakan adalah *Waterfall*. *Waterfall* merupakan metode pengembangan model statis dalam pengembangan sistem secara linear dan berurutan, dengan menyelesaikan satu aktivitas sebelum melakukan aktivitas lainnya (Adenowo, 2013). Tahap pertama yang dilakukan adalah analisis kebutuhan dengan mengumpulkan data dan melakukan analisis hingga menemukan sebuah kebutuhan untuk pembangunan sistem rekam medis terpusat. Tahap kedua yaitu mulai merancang dari hasil kebutuhan yang telah didapatkan. Tahap ketiga yaitu implementasi dari hasil perancangan. Setelah melakukan implementasi kemudian melakukan pengujian terhadap imlementasi yang dihasilkan.

2.2.6 Pengujian

2.2.6.1 Pengujian White Box

Pengujian *white box* merupakan metode pengujian atau teknik pengujian dengan memeriksa kode program atau jalur logika kode dalam modul setidaknya dilewati sekali untuk memastikan tidak terdapat masalah (Pressman dan Maxim, 2015). Dalam pengujian *white box* terdapat *path coverage* atau *basis path testing* yang merupakan salah satu jenis pengujian yang sering digunakan untuk mendapatkan jalur independen kode program supaya tiap jalur logika dapat diperiksa dan untuk mengetahui tidak terdapat masalah pada tiap jalur.

Pada sistem rekam medis terpusat dengan implementasi *web service* pengujian ini digunakan untuk melihat apakah fungsi sistem benar-benar memanggil fungsi pada layanan *web service* dan tidak memiliki masalah.

2.2.6.2 Pengujian Black Box

Pengujian *black box* merupakan teknik pengujian untuk melihat fungsi berjalan dengan baik dengan melihat kondisi *input* atau kondisi luar yang diterima dan hasil yang dilakukan oleh sistem (Pressman dan Maxim, 2015). Dalam pengujian *black box* terdapat pengujian validasi yaitu untuk memastikan bahwa sistem berjalan sesuai dengan kebutuhan fungsional. Jika terdapat kesalahan pada saat pengujian maka dapat dipastikan dengan pengujian *white box* untuk melihat kesalahan kode program.

Pada sistem rekam medis terpusat dengan implementsi *web service* pengijan validasi pada pengujian *black box* mengandalkan kasus uji dengan melihat kondisi *input-output* sesuai dengan skenario *use case*.

2.2.6.3 Functional Suitabilty

Functional suitabilty merupakan salah satu komponen faktor penentuan kualitas perangkat lunak berdasarkan hasil pengujian fungsional sistem dengan standar ISO 25010 (Sulaen, 2017). Analisis data pada faktor *functional suitabilty* dilakukan dengan cara menghitung hasil pengujian menggunakan persentase keberhasilan, kemudian dilakukan konversi ke dalam bentuk predikat (Guritno *et all*, 2011).

$$Persentase\ Keberhasilan = \frac{i}{r} \times 100\% \quad \textbf{(2.1)}$$

Keterangan:

i = Jumlah kebutuhan fungsional yang berhasil diimplementasikan

r = Jumlah total pengujian kebutuhan fungsional yang dilakukan

Tabel predikat digunakan untuk mengukur kelayakan aplikasi yang dibangun (Putra, 2016). Terdapat lima kategori yang dimasukkan ke dalam predikat kelayakan sebagai berkut.

Tabel 2.7 Predikat kelayakan aplikasi

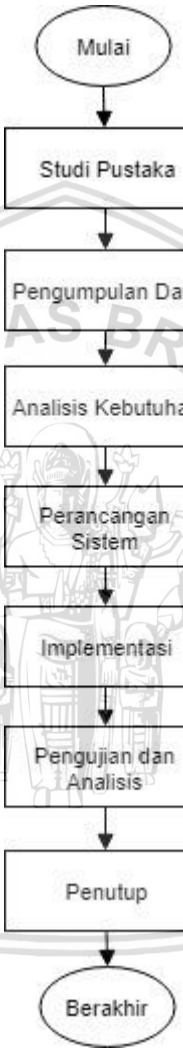
Persentase	Predikat
0%-20%	Sangat Buruk
21%-40%	Buruk
41%-60%	Cukup
61%-80%	Baik
81%-100%	Sangat Baik

Pada tabel 2.7 merupakan predikat kelayakan aplikasi berdasarkan hasil total pengujian fungsional yang telah dikonversikan ke dalam persentase

keberhasilan. Pada penelitian ini *functional suitability* digunakan untuk mengukur hasil pengujian fungsional pada rata-rata persentase keberhasilan implementasi *web service* pada sistem rekam medis terpusat.

BAB 3 METODOLOGI PENELITIAN

Pada metodologi penelitian ini membahas mengenai tahap-tahap yang akan dilakukan dalam melakukan penelitian "Implementasi *Web Service* Pada Sistem Rekam Medis Terpusat" yang akan dibuat. Tahapan-tahapan yang dilakukan diilustrasikan dengan *diagram* alir pada gambar berikut.



Gambar 3.1 *Diagram* alir metodologi penelitian

Pada gambar 3.1 mengilustrasikan *diagram* alir dari penelitian yang dilakukan dalam menyelesaikan masalah.

Tipe penelitian yang digunakan pada penelitian ini adalah implementatif pengembagan (*development*). Tipe penelitian ini menghasilkan produk/artefak baru yang sebelumnya belum pernah diterapkan pada objek penelitian sehingga

dapat digunakan untuk menyelesaikan permasalahan yang dialami pada objek penelitian.

3.1 Studi Pustaka

Studi pustaka menjelaskan kajian-kajian teori atau landasan pustaka yang digunakan dalam menunjang penelitian dalam menyelesaikan masalah. Teori-teori tersebut didapatkan berdasarkan penelitian dan data yang telah dilakukan oleh sumber yang dapat dipercaya. Beberapa teori yang mendukung penelitian ini diantaranya:

- Fasilitas Pelayanan Kesehatan;
- Sistem Informasi;
- Rekam Medis;
- *Web Service*;
- *System Development Life Cycle (SDLC);*
- *Pengujian.*

3.2 Pengumpulan Data

Pada tahap ini pengambilan data menggunakan tipe pengumpulan data kualitatif. Data kualitatif digunakan dalam aspek pemahaman secara mendalam mengenai suatu masalah dalam penelitian lingkup generalisasi. Data yang dikumpulkan adalah informasi alur pencatatan rekam medis dan format catatan rekam medis pasien. Data yang dikumpulkan untuk mendukung penelitian terdapat dua jenis yaitu data primer dan sekunder. Data primer diperoleh dari hasil observasi dan rekaman wawancara dari narasumber yang berkaitan dengan topik penelitian yaitu petugas rekam medis dan petugas pendaftaran berobat pada Puskesmas Dinoyo dan Puskesmas Kendalsari Kota Malang. Sedangkan data sekunder berupa dokumen yang berkaitan dengan topik penelitian yaitu berkas rekam medis rawat jalan dan rawat inap serta berkas pendaftaran berobat pasien.

3.3 Analisis Kebutuhan Sistem

Pada tahap analisis kebutuhan, kebutuhan-kebutuhan pengguna digali untuk mendukung perancangan sistem yang akan dibuat. Pengguna didefinisikan dengan baik agar interaksi antar pengguna dan sistem dapat berjalan dengan baik. Menjabarkan kebutuhan-kebutuhan secara fungsional agar mendukung perancangan sistem dapat dilakukan dengan benar.

3.3.1 Tahap Analisis Kebutuhan Sistem

Pada penelitian ini, upaya dalam menggali kebutuhan pengguna dapat dilakukan dengan menggunakan strategi sebagai berikut.

1. Mengidentifikasi pengguna dan ruang lingkup yang akan menggunakan sistem,

2. Melakukan wawancara kepada calon pengguna (khususnya kepada pemangku kepentingan yang mengambil data rekam medis) mengenai sistem yang akan dibuat,
3. Mendefinisikan kebutuhan berdasarkan kebutuhan fungsional,
4. Membuat spesifikasi dari kebutuhan yang telah didefinisikan,
5. Mendefinisikan perilaku pengguna dan sistem atau interaksi pengguna sistem dengan membuat *use case diagram* dan *use case scenario*.

3.4 Perancangan Sistem

Pada tahap perancangan sistem, kebutuhan-kebutuhan yang telah didefinisikan dengan baik pada tahap analisis kebutuhan dibuatkan model untuk menunjang implementasi pembangunan sistem. Model-model tersebut menggambarkan bagaimana sistem dan pengguna dapat berinteraksi dan bertukar informasi, sehingga pada tahap implementasi tidak mengalami kekurangan atau kecacatan pada sistem yang menimbulkan kesalahan informasi pada sistem yang akan diberikan kepada pengguna.

3.4.1 Tahap perancangan sistem

Upaya dalam melakukan perancangan sistem adalah sebagai berikut.

1. Merancang arsitektur komunikasi klien dan *server web service*.
2. Merancang *diagram* kelas sebagai *model* untuk mendefinisi atribut pengguna dan fungsi-fungsi yang harus dimiliki sistem.
3. Merancang *diagram sequence* sebagai *model* untuk menunjukkan interaksi pengguna dan respon sistem.
4. Merancang *diagram* relasi sebagai *model* hubungan yang dimiliki objek-objek pada sistem.
5. Merancang tampilan antarmuka sistem.

3.5 Implementasi

Pada tahap implementasi, menunjukkan bagaimana struktur sistem yang dibuat berdasarkan perancangan pada tahap sebelumnya. Alur implementasi sistem sebagai berikut.

1. Implementasi arsitektur sistem,
2. Implementasi *class*,
3. Implementasi basis data,
4. Implementasi kode program,
5. Implementasi tampilan antarmuka.

3.6 Pengujian dan Analisis

Pada tahap pengujian, menunjukkan bahwa apakah sistem yang dibuat telah berhasil seperti apa yang telah didefinisikan dari tahap analisis kebutuhan, perancangan sistem dan implementasi sistem. Pengujian yang dilakukan yaitu melakukan tes secara fungsional dan *non* fungsional kepada sistem. Pendekatan yang dipakai dalam pengujian yaitu pengujian *white box* dan *black box*.

Pada pengujian *white box* memanfaatkan tipe pengujian *unit testing*. *Unit testing* digunakan untuk menguji tiap komponen *unit* objek pada kode program untuk mengetahui apakah sudah berjalan sesuai dengan fungsinya. Sedangkan pengujian *blackbox* memanfaatkan pengujian validasi atau kasus uji sebagai salah satu bentuk pengujian untuk melihat apakah sistem berjalan dengan baik dilihat dari proses *input-output,* sehingga dapat dipastikan hasil pengujian berupa status *valid* dan *invalid*.

3.7 Kesimpulan dan Saran

Kesimpulan dilakukan setelah semua tahapan analisis kebutuhan, perancangan, implementasi, dan pengujian metode yang diterapkan sudah selesai dilakukan. Kesimpulan diambil dari hasil akhir sesuai dengan masalah yang didefinisikan dalam rumusan masalah. Sedangkan saran yaitu membahas harapan akan dikembangkannya penelitian oleh pihak pengembang selanjutnya dan tidak kalah penting juga untuk penyempurnaan penulisan laporan.

BAB 4 ANALISIS KEBUTUHAN

Proses analisis kebutuhan merupakan tahap awal yang dilakukan dalam pengembangan suatu sistem. Rekayasa kebutuhan umumnya digunakan untuk menentukan kebutuhan-kebutuhan yang harus tersedia pada implementasi *web service* sistem informasi rekam medis terpusat. Dalam tahap ini juga menjelaskan siapa saja yang dapat berinteraksi dengan sistem.

4.1 Elisitasi Kebutuhan

Elisitasi kebutuhan merupakan proses untuk mengumpulkan informasi-informasi yang dibutuhkan untuk menemukan kebutuhan suatu sistem. Elisitasi kebutuhan pada sistem informasi rekam medis terpusat dilakukan dengan interaksi berupa komunikasi terhadap pemangku kepentingan guna mendapatkan sebuah kebutuhan. Kemudian dilanjutkan dengan mengidentifikasi kebutuhan dan digunakan sebagai solusi atas permasalahan yang ada.

4.2 Identifikasi Pemangku Kepentingan

Pada tahap ini hal yang dilakukan adalah mengidentifikasi dan menganalisis pemangku kepentingan yang berhubungan dengan sistem dengan menjelaskan tipe pemangku kepentingan pada sistem yang akan dibuat. Identifikasi pemangku kepentingan diantaranya sebagai berikut.

Tabel 4.1 Identifikasi pemangku kepentingan

No	Tipe pemangku kepentingan	Deskripsi Pemangku Kepentingan	Pemangku kepentingan yang relevan
1	Pengguna	Setiap orang yang menggunakan sistem informasi rekam medis terpusat	Petugas loket pendaftaran dan petugas rekam medis
2	Pengembang	Individual atau sekelompok orang yang bertanggung jawab atas pengembangan aplikasi sistem informasi rekam medis terpusat	Mahasiswa

Pada tabel 4.1 di atas terdapat dua tipe pemangku kepentingan yaitu pengguna dan pengembang. Pengguna merupakan setiap orang yang menggunakan dan mengoperasikan sistem informasi rekam medis terpusat untuk memenuhi proses bisnis, pemangku kepentingan yang relevan diantaranya adalah petugas loket dan petugas rekam medis. Sedangkan tipe pemangku kepentingan pengembang merupakan setiap orang atau individual atau kelompok yang bertanggung jawab atas berjalannya sistem informasi rekam medis terpusat yaitu mahasiswa.

4.3 Teknik Elisitasi

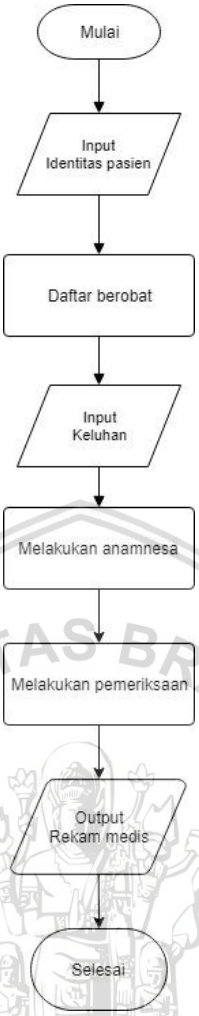
Pada tahap ini teknik yang digunakan untuk menggali informasi menggunakan metode wawancara dan observasi. Wawancara dilakukan kepada pemangku kepentingan seperti petugas loket dan petugas rekam medis dengan menggali informasi menangani alur pencatatan rekam medis dari melakukan registrasi di loket hingga pasien selesai berobat. Kemudian observasi dilakukan pada objek penelitian untuk melihat alur kegiatan pencatatan rekam medis. Dari hasil elisitasi dan observasi yang telah dilakukan, maka didapatkan beberapa kebutuhan diantaranya adalah sebagai berikut.

Tabel 4.2 Temuan kebutuhan

No.	Temuan kebutuhan
1	Petugas loket pendaftaran harus dapat mengolah data pasien daftar berobat.
2	Petugas rekam medis harus dapat mengolah data rekam medis pasien berobat.
3	Petugas rekam medis harus dapat mencari jejak riwayat berobat pasien berupa data atau berkas rekam medis dari fasilitas kesehatan lain.

4.3.1 Alur Pencatatan Rekam Medis Terpusat

Pada gambar 2.1 telah dijelaskan mengenai alur rekam medis yang didapat dari hasil observasi dan wawancara pada Puskesmas Dinoyo Kota Malang dan tidak jauh berbeda dengan alur yang terdapat pada Puskesmas Kendalsari Kota Malang. Sehingga dapat disimpulkan alur pencatatan rekam medis jika diterapkan dengan sistem terpusat dapat dilihat pada gambar 4.1.



Gambar 4.1 Alur pencatatan rekam medis terpusat

Pada gambar 4.1 menerangkan bahwa alur proses pencatatan rekam medis ke dalam sistem lebih dilakukan hanya oleh dua peran yaitu petugas loket dan petugas rekam medis. Petugas loket berperan hanya untuk mengolah data pendaftaran pasien berobat, sedangkan petugas rekam medis berperan penting untuk memasukkan data rekam medis pasien ke dalam sistem. Ini dilakukan agar tidak mengganggu proses kerja tenaga perawat atau dokter dalam mencatat rekam medis dengan standar prosedur masing-masing fasilitas kesehatan.

4.4 Spesifikasi Kebutuhan

Spesifikasi kebutuhan untuk sistem informasi rekam medis terpusat dideskripsikan dengan kebutuhan sesuai dengan solusi permasalahan yang telah diidentifikasi pada tahap sebelumnya. Kemudian daftar dari spesifikasi kebutuhan yang telah diidentifikasi akan diurutkan sesuai dengan prioritas. Skala prioritas dapat dinilai bobotnya ketika suatu kebutuhan tersebut menjawab permasalahan utama yang telah dideskripsikan pada tahap elisitasi. Sedangkan skala prioritas medium dapat dinilai ketika kebutuhan tersebut mendukung kebutuhan utama

dan skala prioritas rendah dapat dinilai jika suatu kebutuhan hanya mendukung sistem dan tidak berkaitan dengan kebutuhan utama.

Daftar spesifikasi kebutuhan fungsional untuk sistem informasi rekam medis terpusat dapat dilihat pada Tabel 4.3 berikut.

Tabel 4.3 Kebutuhan fungsional aktor pengguna

No SRS	Kebutuhan	*Use case*	Aktor	Prioritas
SRS_F_1_01	Sistem dapat menampilkan seluruh data daftar berobat pasien	Daftar Berobat	Pengguna	*Medium*
SRS_F_1_02	Sistem harus mampu mengubah data pasien berobat	*Edit* daftar	Pengguna	*Medium*
SRS_F_1_03	Sistem harus menyediakan fungsi hapus data pasien	Hapus data	Pengguna	*High*
SRS_F_1_04	Sistem harus mampu menambahkan data Daftar Berobat pasien baru	Tambah daftar	Pengguna	*High*
SRS_F_1_05	Sistem harus mampu menampilkan data rekam medis pasien berdasarkan Daftar Berobat	Lihat RM	Pengguna	*Medium*
SRS_F_1_06	Sistem harus mampu menambahkan data rekam medis baru	Tambah RM	Pengguna	*High*
SRS_F_1_07	Sistem menyediakan tombol *Edit* pada rekam medis pasien.	*Edit* RM	Pengguna	*Medium*
SRS_F_1_08	Sistem harus mampu menambahkan data anamnesa pasien	Tambah Anamnesa	Pengguna	*Medium*
SRS_F_1_09	Sistem harus mampu mengubah data anamnesa	*Edit* Anamnesa	Pengguna	*Medium*

Tabel 4.3 Kebutuhan fungsional aktor pengguna (lanjutan)

No SRS	Kebutuhan	*Use case*	Aktor	Prioritas
SRS_F_1_10	Sistem harus mampu menambah data pemeriksaan umum	Tambah Pemum	Pengguna	*Medium*
SRS_F_1_11	Sistem harus mampu mengubah data pemeriksaan umum	*Edit* Pemum	Pengguna	*Medium*
SRS_F_1_12	Sistem harus mampu menambah data pemeriksaan fisik	Tambah Fisik	Pengguna	*Medium*
SRS_F_1_13	Sistem harus mampu mengubah data pemeriksaan fisik	*Edit* Fisik	Pengguna	*Medium*
SRS_F_1_14	Sistem harus mampu menambah data data sistem pernafasan	Tambah pernafasan	Pengguna	*Medium*
SRS_F_1_15	Sistem harus mampu mengubah data sistem pernafasan	*Edit* Pernafasan	Pengguna	*Medium*
SRS_F_1_16	Sistem harus mampu menambah data kardiovaskuler	Tambah Kardiovaskuler	Pengguna	*Medium*
SRS_F_1_17	Sistem harus mampu mengubah data kardiovaskuler	*Edit* Kardiovaskuler	Pengguna	*Medium*
SRS_F_1_18	Sistem harus mampu menambah data sistem persyarafan	Tambah Syaraf	Pengguna	*Medium*
SRS_F_1_19	Sistem harus mampu mengubah data persyarafan	*Edit* Syaraf	Pengguna	*Medium*

Tabel 4.3 Kebutuhan fungsional aktor pengguna (lanjutan)

No SRS	Kebutuhan	*Use case*	Aktor	Prioritas
SRS_F_1_20	Sistem harus mampu menambah data sistem perkemihan	Tambah Perkemihan	Pengguna	*Medium*
SRS_F_1_21	Sistem harus mampu mengubah data sistem Perkemihan	*Edit* perkemihan	Pengguna	*Medium*
SRS_F_1_22	Sistem harus mampu menambah data sistem pencernaan	Tambah Pencernaan	Pengguna	*Medium*
SRS_F_1_23	Sistem harus mampu mengubah data pencernaan	*Edit* Pencernaan	Pengguna	*Medium*
SRS_F_1_24	Sistem harus mampu menambah data muskuloskleletal	Tambah Muskuloskleletal	Pengguna	*Medium*
SRS_F_1_25	Sistem harus mampu mengubah data muskuloskleletal	*Edit* Muskuloskleletal	Pengguna	*Medium*
SRS_F_1_26	Sistem harus mampu menambah data sistem reproduksi	Tambah Reproduksi	Pengguna	*Medium*
SRS_F_1_27	Sistem harus mampu mengubah data reproduksi	*Edit* Reproduksi	Pengguna	*Medium*
SRS_F_1_28	Sistem harus dapat menambah data psikologi, sosiologi dan spiritual	Tambah Dpss	Pengguna	*Medium*
SRS_F_1_29	Sistem harus mampu mengubah data psikologi, sosiologi dan spiritual	*Edit* Dpss	Pengguna	*Medium*

Tabel 4.3 Kebutuhan fungsional aktor pengguna (lanjutan)

No SRS	Kebutuhan	*Use case*	Aktor	Prioritas
SRS_F_1_30	Sistem harus mampu menambahkan data pemeriksaan penunjang	Tambah Datalab	Pengguna	*Medium*
SRS_F_1_31	Sistem harus mampu mengubah data pemeriksaan penunjang	*Edit* Datalab	Pengguna	*Medium*
SRS_F_1_32	Sistem harus mampu menambahkan data petugas	Tambah Petugas	Pengguna	*Medium*
SRS_F_1_33	Sistem harus mampu mengubah data petugas	*Edit* Petugas	Pengguna	*Medium*
SRS_F_1_34	Sistem harus mampu menampilkan data resume medis pasien	Resume	Pengguna	*Medium*
SRS_F_1_35	Sistem harus mampu menambahkan data resume baru	Tambah Resume	Pengguna	*High*
SRS_F_1_36	Sistem menyediakan tombol *Edit* pada Resume Medis pasien	*Edit* Resume	Pengguna	*Medium*
SRS_F_1_37	Sistem mampu melakukan pencarian daftar diagnosis berdasarkan *category*	Cari Diagnosis	Pengguna	*Medium*
SRS_F_1_38	Sistem mampu menambahkan data diagnosis baru	Tambah Diagnosis	Pengguna	*Low*
SRS_F_1_39	Sistem menyediakan tombol *Edit* pada Daftar Diagnosis	*Edit* Diagnosis	Pengguna	*Medium*

Tabel 4.3 Kebutuhan fungsional aktor pengguna (lanjutan)

No SRS	Kebutuhan	*Use case*	Aktor	Prioritas
SRS_F_1_40	Sistem menyediakan tombol hapus pada Daftar Diagnosis	Hapus Diagnosis	Pengguna	*Low*
SRS_F_1_41	Sistem menyediakan fungsi untuk mengunggah berkas rekam medis	*Upload* Berkas	Pengguna	*High*
SRS_F_1_42	Sistem menyediakan fungsi untunk mengunduh berkas rekam medis pasien	*Download* Berkas	Pengguna	*High*
SRS_F_1_43	Sistem menyediakan akses *logout* agar pengguna dapat keluar dari sistem	*Logout*	Pengguna	*Low*

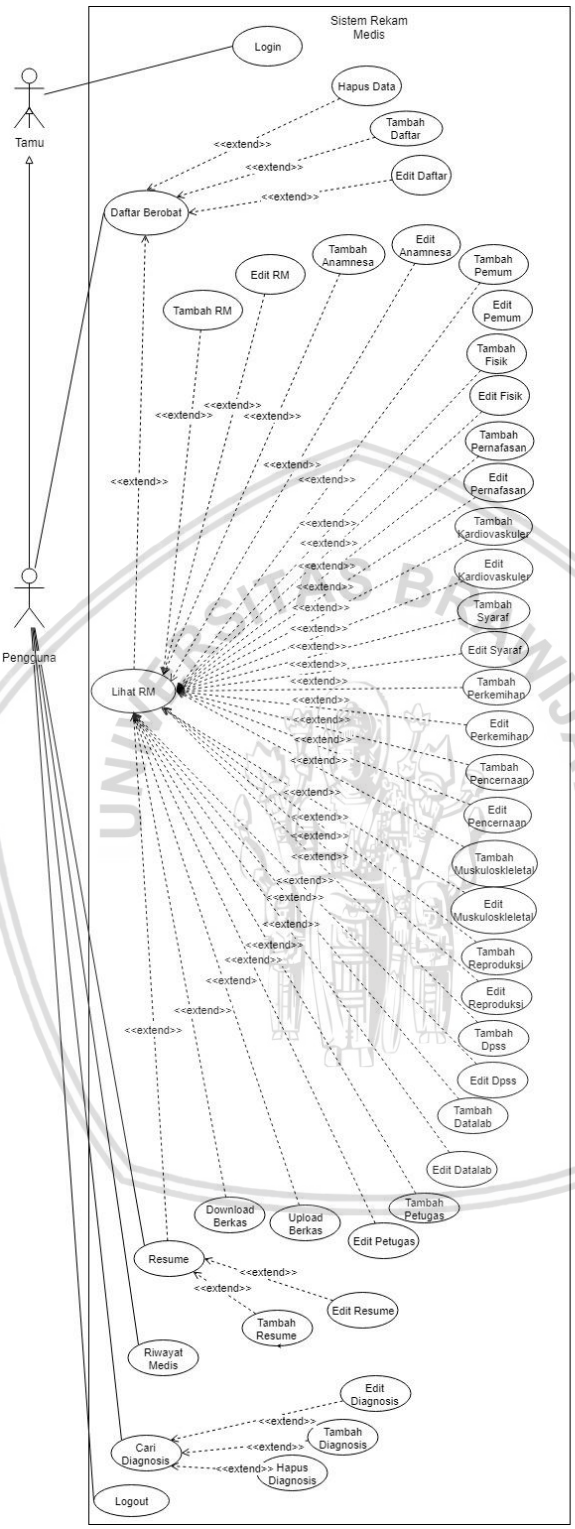
Tabel 4.4 Kebutuhan fungsional aktor tamu

No SRS	Kebutuhan	*Use case*	Aktor	Prioritas
SRS_F_2_01	Sistem harus menyediakan autentikasi agar pengguna dapat menggunakan perangkat lunak.	*Login*	Tamu	*High*

Tabel 4.5 Kebutuhan Non-Fungsional

No SRS	Kebutuhan	Prioritas
SRS_NF_1_01	Layanan *web service* harus diberi keamanan berupa authentikasi	*High*

4.5 *Diagram Use case*



Gambar 4.2 *Use case diagram* rekam medis terpusat

Pada gambar 4.2 di atas memperlihatkan *diagram use case* sistem rekam medis.

4.6 Use case Scenario

Use case scenario merupakan alur dari jalannya sebuah proses *use case* dari sisi aktor dan sistem. *Use case scenario* dari sistem rekam medis akan dideskripsikan pada bagian berikut.

4.6.1 *Use case scenario* Daftar Berobat

Use case scenario Daftar Berobat dapat dilihat deskripsi lengkapnya pada Tabel 4.6 berikut.

Tabel 4.6 Skenario Daftar Berobat

Actor	Pengguna
Objective	Fungsi untuk menampilkan data pasien berobat
Pre-Condition	Aktor telah masuk ke dalam sistem rekam medis
Main Flow	1. Aktor memilih menu Daftar Berobat 2. Sistem menampilkan daftar pasien berobat
Alternative Flow	-
Post-Condition	Sistem menampilkan halaman daftar pasien berobat

4.6.2 *Use case scenario Edit* Daftar

Use case scenario Edit Daftar dapat dilihat lengkap pada Tabel 4.7 berikut.

Tabel 4.7 Skenario *Edit* Daftar

Actor	Pengguna
Objective	Fungsi untuk mengubah data pasien berobat
Pre-Condition	Aktor melihat daftar pasien berobat
Main Flow	1. Aktor menekan tombol *Edit* pada tabel data daftar pasien berobat 2. Sistem menampilkan menu *pop-up* dengan data pasien yang akan diubah 3. Aktor mengubah data yang akan diubah 4. Aktor menekan tombol Simpan Perubahan 5. Sistem akan mengirim data perubahan ke *server* 6. Jika perubahan berhasil maka sistem menampilkan pesan "Sukses! Data berhasil diubah"
Alternative Flow	1. Jika tidak berhasil menyimpan data sistem akan menampilkan "*Error*! Data gagal diubah"
Post-Condition	Sistem menampilkan halaman daftar pasien berobat

4.6.3 *Use case scenario* Hapus Data

Use case scenario Hapus Data dapat dilihat lengkap pada Tabel 4.8 berikut.

Tabel 4.8 Skenario Hapus Data

Actor	Pengguna
Objective	Fungsi untuk menghapus seluruh data milik pasien dari awal berobat
Pre-Condition	Aktor melihat daftar pasien berobat
Main Flow	1. Aktor menekan tombol Hapus 2. Sistem menampilkan pesan "Anda yakin ingin menghapus seluruh data pasien ini?" 3. Jika aktor menekan tombol *Yes* maka sistem akan mengirim id daftar pasien yang akan dihapus ke *server* 4. Jika id daftar ditemukan maka data pasien seluruhnya akan dihapus 5. Jika hapus berhasil maka menampilkan pesan "Sukses! Data berhasil dihapus"
Alternative Flow	1. Jika aktor menekan tombol *Cancel* maka sistem tidak akan melakukan eksekusi hapus data 2. Jika id daftar tidak ditemukan maka sistem akan menampilkan pesan "*Error*! Data gagal dihapus"
Post-Condition	Sistem menampilkan halaman daftar pasien berobat

4.6.4 *Use case scenario* Tambah Daftar

Use case scenario Tambah Daftar dapat dilihat lengkap pada Tabel 4.9 berikut.

Tabel 4.9 Skenario Tambah Daftar

Actor	Pengguna
Objective	Fungsi untuk menambahkan data pasien berobat baru
Pre-Condition	Aktor melihat daftar pasien berobat
Main Flow	1. Aktor menekan tombol Tambah 2. Sistem akan menampilkan halaman *form* tambah daftar pasien berobat 3. Aktor mengisi data pasien pada kolom *input* 4. Setelah selesai aktor menekan tombol *Save*

Tabel 4.9 Skenario Tambah Daftar (lanjutan)

Main Flow	5. Jika berhasil menyimpan data sistem akan menampilkan pesan "Sukses! Data berhasil disimpan"
Alternative Flow	1. Jika data tidak berhasil disimpan maka menampilkan pesan "*Error*! Data gagal disimpan"
Post-Condition	Sistem menampilkan halaman daftar pasien berobat

4.6.5 *Use case scenario* Lihat RM

Use case scenario Lihat RM dapat dilihat lengkap pada Tabel 4.10 berikut.

Tabel 4.10 Skenario Lihat RM

Actor	Pengguna
Objective	Fungsi untuk melihat data rekam medis pasien
Pre-Condition	Aktor melihat daftar pasien berobat
Main Flow	1. Aktor menekan *link* pada nomor NIK pasien 2. Sistem menampilkan *pop-up* detail data pasien 3. Aktor menekan tombol *Lihat RM* 4. Jika data rekam medis tidak kosong maka sistem akan menampilkan halaman rekam medis pasien
Alternative Flow	1. Jika data rekam medis kosong maka sistem menampilkan halaman data rekam medis kosong
Post-Condition	Sistem menampilkan halaman data rekam medis pasien

4.6.6 *Use case scenario* Tambah RM

Use case scenario Tambah RM dapat dilihat lengkap pada Tabel 4.11 berikut.

Tabel 4.11 Skenario Tambah RM

Actor	Pengguna
Objective	Fungsi untuk menambahkan data rekam medis pasien
Pre-Condition	Aktor melihat halaman kosong data rekam medis pasien
Main Flow	1. Aktor menekan tombol Tambah RM 2. Sistem menampilkan halaman *form* masukkan data rekam medis pasien 3. Setelah mengisi kolom *input* aktor menekan tombol *Save*

Tabel 4.11 Skenario Tambah RM (lanjutan)

Main Flow	4. Jika berhasil menyimpan data sistem akan menampilkan data di halaman rekam medis pasien
Alternative Flow	1. Jika data tidak berhasil disimpan maka menampilkan pesan "*Error*! Data gagal disimpan"
Post-Condition	Sistem menampilkan halaman data rekam medis pasien

4.6.7 *Use case scenario Edit* RM

Use case scenario Edit RM dapat dilihat lengkap pada Tabel 4.12 berikut.

Tabel 4.12 Skenario *Edit* RM

Actor	Pengguna
Objective	Fungsi untuk mengubah data rekam medis pasien
Pre-Condition	Aktor melihat halaman data rekam medis pasien
Main Flow	1. Aktor menekan tombol *Edit* pada kolom data rekam medis 2. Sistem menampilkan *pop-up form* untuk mengubah data rekam medis 3. Setelah mengubah data aktor menekan tombol Simpan Perubahan 4. Jika berhasil menyimpan data sistem akan menampilkan pesan "Sukses! Data berhasil diubah"
Alternative Flow	1. Jika data tidak berhasil disimpan maka menampilkan pesan "*Error*! Data gagal diubah"
Post-Condition	Sistem menampilkan halaman data rekam medis pasien

4.6.8 *Use case scenario* Tambah Anamnesa

Use case scenario Tambah Anamnesa dapat dilihat lengkap pada Tabel 4.13 berikut.

Tabel 4.13 Skenario Tambah Anamnesa

Actor	Pengguna
Objective	Fungsi untuk menambahkan data anamnesa pada halaman rekam medis pasien
Pre-Condition	Aktor melihat data anamnesa di halaman data rekam medis pasien
Main Flow	1. Aktor memilih menu info anamnesa 2. Aktor menekan tombol Tambah pada kolom data anamnesa 3. Sistem akan menampilkan *popuop form* untuk memasukkan data anamnesa

Tabel 4.13 Skenario Tambah Anamnesa (lanjutan)

Main Flow	4. Setelah memasukkan data anamnesa maka aktor menekan tombol Simpan 5. Jika data berhasil disimpan maka akan menampilkan pesan "Sukses! Data berhasil disimpan"
Alternative Flow	1. Jika data tidak berhasil disimpan maka menampilkan pesan "*Error*! Data gagal disimpan"
Post-Condition	Sistem menampilkan halaman data rekam medis pasien

Pada penjelasan skenario tabel 4.12 di atas digunakan oleh beberapa fungsi lain untuk menambahkan info data rekam medis di satu halaman, yang berbeda ialah hanya objek data yang ingin ditambahkan namun skenario yang digunakan hampir serupa yang diantaranya adalah skenario Tambah Pemum, Tambah Fisik, Tambah Kardiovaskuler, Tambah Syaraf, Tambah Perkemihan, Tambah Pencernaan, Tambah Muskuloskleletal, Tambah Reproduksi, Tambah Dpss, Tambah Datalab dan Tambah Petugas.

4.6.9 *Use case scenario Edit* Anamnesa

Use case scenario Edit Anamnesa dapat dilihat lengkap pada Tabel 4.14 berikut.

Tabel 4.14 Skenario *Edit* Anamnesa

Actor	Pengguna
Objective	Fungsi untuk mengubah data anamnesa pada halaman rekam medis pasien
Pre-Condition	Aktor melihat data anamnesa di halaman data rekam medis pasien
Main Flow	1. Aktor memilih menu info anamnesa 2. Aktor menekan tombol *Edit* pada kolom data anamnesa 3. Sistem akan menampilkan *popuop form* dengan data anamnesa pasien untuk mengubah data anamnesa 4. Setelah mengubah data anamnesa maka aktor menekan tombol Simpan Perubahan 5. Jika data berhasil disimpan maka akan menampilkan pesan "Sukses! Data berhasil diubah"
Alternative Flow	1. Jika data tidak berhasil disimpan maka menampilkan pesan "*Error*! Data gagal diubah"
Post-Condition	Sistem menampilkan halaman data rekam medis pasien

Pada penjelasan skenario di atas digunakan oleh beberapa fungsi lain untuk menambahkan info data rekam medis di satu halaman, yang berbeda ialah

hanya menu objek data yang ingin ditambahkan namun skenario yang digunakan hampir serupa yang diantaranya adalah skenario *Edit* Pemum, *Edit* Fisik, *Edit* Kardiovaskuler, *Edit* Syaraf, *Edit* Perkemihan, *Edit* Pencernaan, *Edit* Muskuloskleletal, *Edit* Reproduksi, *Edit* Dpss, *Edit* Datalab dan *Edit* Petugas.

4.6.10 *Use case scenario* Resume

Use case scenario Resume dapat dilihat lengkap pada Tabel 4.15 berikut.

Tabel 4.15 Skenario Resume

Actor	Pengguna
Objective	Fungsi untuk melihat data resume pasien
Pre-Condition	Aktor melihat data rekam medis di halaman data rekam medis pasien
Main Flow	1. Aktor menekan tombol Resume di kolom data rekam medis pasien 2. Sistem akan menampilkan halaman resume medis pasien jika terdapat data resume
Alternative Flow	1. Jika data resume kosong maka akan menampilkan halaman resume kosong
Post-Condition	Sistem menampilkan halaman data resume pasien

4.6.11 *Use case scenario* Tambah Resume

Use case scenario Tambah Resume dapat dilihat lengkap pada Tabel 4.16 berikut.

Tabel 4.16 Skenario Tambah Resume

Actor	Pengguna
Objective	Fungsi untuk menambah data resume pasien
Pre-Condition	Aktor melihat data resume pasien di halaaman resume medis pasien
Main Flow	1. Aktor menekan tombol Tambah pada halaman resume 2. Sistem akan menampilkan *form* pada halaman lain untuk memasukkan data resume pasien 3. Setelah memasukkan data resume aktor memilih tombol *Save* untuk menyimpan data 4. Sistem akan mengirim data ke *server* untuk menyimpan data 5. Jika data berhasil disimpan maka akan menampilkan pesan “Sukses! Data berhasil disimpan”

Tabel 4.16 Skenario Tambah Resume (Lanjutan)

Alternative Flow	1. Jika data tidak berhasil disimpan maka menampilkan pesan "*Error*! Data gagal disimpan"
Post-Condition	Sistem menampilkan data resume di halaman resume pasien

4.6.12 *Use case scenario Edit* Resume

Use case scenario Edit Resume dapat dilihat lengkap pada Tabel 4.17 berikut.

Tabel 4.17 Skenario *Edit* Resume

Actor	Pengguna
Objective	Fungsi untuk mengubah data resume pasien
Pre-Condition	Aktor melihat data resume pasien di halaaman resume medis pasien
Main Flow	1. Aktor menekan tombol Detail pada halaman resume 2. Sistem akan menampilkan data lengkap resume pasien 3. Pada halaman detail resume aktor memilih tombol *Edit* 4. Sistem akan menampilkan *pop-up form* dengan data resume untuk diubah 5. Setelah mengubah data resume aktor menekan tombol Simpan Perubahan 6. Sistem akan menyimpan data perubahan resume pasien 7. Jika data berhasil disimpan maka akan menampilkan pesan "Sukses! Data berhasil diubah"
Alternative Flow	1. Jika data tidak berhasil disimpan maka menampilkan pesan "*Error*! Data gagal diubah"
Post-Condition	Sistem menampilkan data resume di halaman resume pasien

4.6.13 *Use case scenario* Cari Diagnosis

Use case scenario Cari Diagnosis dapat dilihat lengkap pada Tabel 4.18 berikut.

Tabel 4.18 Skenario Cari Diagnosis

Actor	Pengguna
Objective	Fungsi untuk mencari daftar diagnosis berdasarkan *category* ICD-10
Pre-Condition	Aktor memilih menu cari diagnosis
Main Flow	1. Aktor memasukkan nomor *category* diagnosis di kolom pencarian diagnosis kemudian menekan *enter*

Tabel 4.18 Skenario Cari Diagnosis (lanjutan)

Main Flow	2. Sistem akan melakukan pencarian berdasarkan nomor *category* diagnosis 3. Setelah data ditemukan sistem menampilkan daftar diagnosis berdasarkan nomor *category*
Alternative Flow	-
Post-Condition	Sistem menampilkan daftar diagnosis berdasarkan nomor *category*

4.6.14 *Use case scenario* Tambah Diagnosis

Use case scenario Tambah Diagnosis dapat dilihat lengkap pada Tabel 4.19 berikut.

Tabel 4.19 Skenario Tambah Diagnosis

Actor	Pengguna
Objective	Fungsi untuk menambah data diagnosis
Pre-Condition	Aktor memilih menu cari diagnosis
Main Flow	1. Aktor memilih tombol Tambah pada halaman pencarian diagnosis 2. Sistem akan menampilkan *form* pada halaman lain untuk memasukkan data diagnosis baru 3. Setelah menambahkan data aktor memilih tombol *Save* untuk menyimpan data 4. Jika data berhasil disimpan maka akan menampilkan pesan "Sukses! Data berhasil disimpan"
Alternative Flow	1. Jika data tidak berhasil disimpan maka menampilkan pesan "*Error*! Data gagal disimpan"
Post-Condition	Sistem menampilkan pesan pada halaman pencarian diagnosis

4.6.15 *Use case scenario Edit* Diagnosis

Use case scenario Edit Diagnosis dapat dilihat lengkap pada Tabel 4.20 berikut.

Tabel 4.20 Skenario *Edit* Diagnosis

Actor	Pengguna
Objective	Fungsi untuk mengubah data diagnosis
Pre-Condition	Aktor melihat daftar diagnosis

Tabel 4.20 Skenario *Edit* Diagnosis (lanjutan)

Main Flow	1. Aktor memilih tombol *Edit* untuk melakukan perubahan data 2. Sistem akan menampilkan *form* dengan data diagnosis di halaman lain untuk diubah 3. Setelah aktor mengubah data maka aktor memilih tombol *Save* untuk menyimpan data 4. Jika data berhasil disimpan maka akan menampilkan pesan "Sukses! Data berhasil diubah"
Alternative Flow	1. Jika data tidak berhasil disimpan maka menampilkan pesan "*Error*! Data gagal diubah"
Post-Condition	Sistem menampilkan pesan pada halaman pencarian diagnosis

4.6.16 *Use case scenario* Hapus Diagnosis

Use case scenario Hapus Diagnosis dapat dilihat lengkap pada Tabel 4.21 berikut.

Tabel 4.21 Skenario Hapus Diagnosis

Actor	Pengguna
Objective	Fungsi untuk menghapus data diagnosis
Pre-Condition	Aktor melihat daftar diagnosis
Main Flow	1. Aktor memilih tombol Hapus untuk menghapus data diagnosis 2. Sistem akan menampilkan pesan peringatan apakah ingin menghapus data 3. Jika iya maka sistem akan mengirim permintaan hapus data ke *server* 4. Jika data berhasil dihapus maka akan menampilkan pesan "Sukses! Data berhasil dihapus"
Alternative Flow	1. Jika tidak ingin menghapus data maka sistem akan membatalkan perintah hapus 2. Jika data tidak berhasil disimpan maka menampilkan pesan "*Error*! Data gagal dihapus"
Post-Condition	Sistem menampilkan pesan pada halaman pencarian diagnosis

4.6.17 *Use case scenario Upload* Berkas

Use case scenario Upload Berkas dapat dilihat lengkap pada Tabel 4.22 berikut.

Tabel 4.22 Skenario *Upload* Berkas

Actor	Pengguna
Objective	Fungsi untuk mengunggah berkas rekam medis pasien
Pre-Condition	Aktor masuk ke halaman rekam medis pasien
Main Flow	1. Aktor memilih tombol *unggah file* 2. Sistem akan menampilkan *pop-up* untuk melampirkan berkas 3. Berkas yang dilampirkan berekstensi .csv, .pdf dan .doc 4. Jika berhasil mengunggah berkas maka akan muncul pesan berhasil dan jika tidak maka sistem tidak akan melakukan eksekusi
Alternative Flow	-
Post-Condition	Sistem menampilkan pesan berhasil dan kembali ke halaman daftar rekam medis

4.6.18 *Use case scenario Download* Berkas

Use case scenario Download Berkas dapat dilihat lengkap pada Tabel 4.23 berikut.

Tabel 4.23 Skenario *Download* Berkas

Actor	Pengguna
Objective	Fungsi untuk mengunduh berkas rekam medis pasien
Pre-Condition	Aktor masuk ke halaman rekam medis pasien
Main Flow	1. Aktor memilih tombol *Unduh file* 2. Sistem akan menampilkan pesan apakah yakin pengguna ingin mengunduh berkas 3. Jika ya sistem akan mulai melakukan proses *download* dan jika tidak sistem tidak melakukan proses *download*
Alternative Flow	-
Post-Condition	Sistem melakukan *download* berkas

4.6.19 *Use case scenario* Riwayat Medis

Use case scenario Riwayat Medis dapat dilihat lengkap pada Tabel 4.24 berikut.

Tabel 4.24 Skenario Riwayat Medis

Actor	Pengguna
Objective	Fungsi untuk melihat data riwayat medis pasien sebelumnya
Pre-Condition	Aktor berada di dalam sistem rekam medis
Main Flow	1. Aktor memilih menu Riwayat Medis 2. Sistem akan menampilkan daftar riwayat medis pasien 3. Aktor mencari data pasien di kolom pencarian dan memilih tombol detail 4. Sistem akan menampilkan informasi riwayat medis pasien yang dicari aktor
Alternative Flow	-
Post-Condition	Sistem menampilkan informasi riwayat medis pasien

4.6.20 *Use case scenario Logout*

Use case scenario Logout dapat dilihat lengkap pada Tabel 4.25 berikut.

Tabel 4.25 Skenario *Logout*

Actor	Pengguna
Objective	Fungsi untuk keluar dari sistem
Pre-Condition	Aktor memilih menu *logout*
Main Flow	1. Aktor memilih fitur *logout* pada navigasi 2. Sistem akan menampilkan peringatan apakah aktor yakin ingin keluar dari sistem 3. Jika ya maka sistem akan mematikan *session* dan aktor keluar dari sistem
Alternative Flow	-
Post-Condition	Aktor keluar sistem

4.6.21 *Use case scenario Login*

Use case scenario Login dapat dilihat lengkap pada Tabel 4.26 berikut.

Tabel 4.26 Skenario *Login*

Actor	Pengguna
Objective	Fungsi untuk masuk ke dalam sistem

Tabel 4.26 Skenario *Login* (lanjutan)

Pre-Condition	Aktor berada pada halaman *login* rekam medis
Main Flow	1. Aktor memasukkan *username* dan *password* dan memilih tombol *login* 2. Sistem akan melakukan pengecekan data 3. Jika data ditemukan maka sistem akan menyimpan *session* dan menampilkan halaman *dashboard*
Alternative Flow	Jika data yang dimasukkan salah maka menampilkan pesan "*Error*! *Username* dan *password* salah"
Post-Condition	Aktor masuk ke dalam sistem

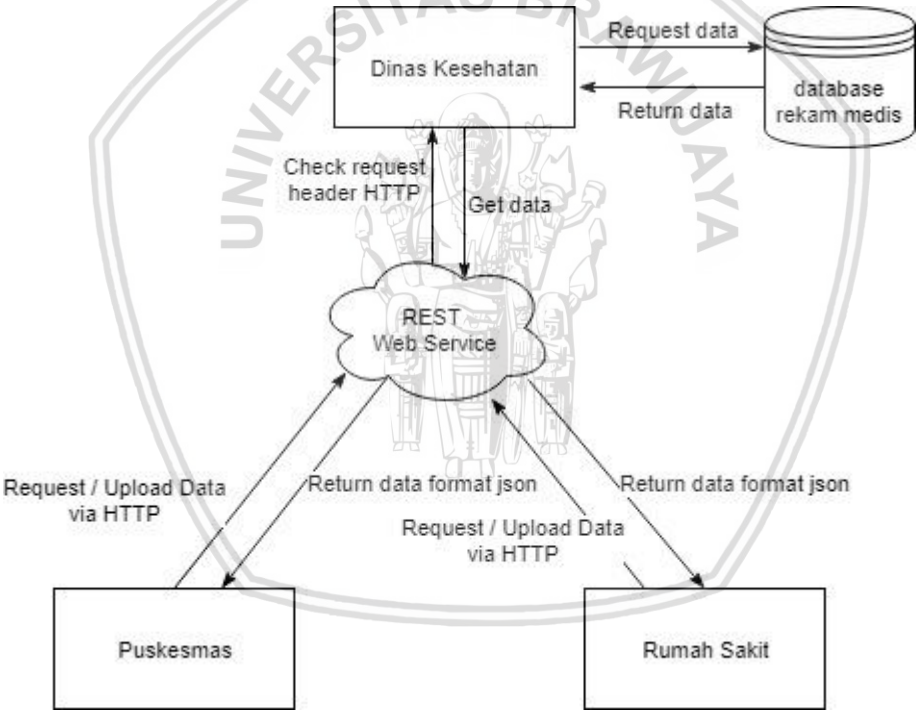
BAB 5 PERANCANGAN DAN IMPLEMENTASI

5.1 Perancangan

Setelah tahap analisis kebutuhan, tahap selanjutnya yaitu perancangan sistem. Perancangan yang akan dilakukan diantaranya meliputi perancangan arsitektur sistem, pemodelan *class diagram,* pemodelan *sequence diagram*, *entity* relation *diagram*, perancangan algoritme dan perancangan antarmuka. Pembahasan tahap ini akan dijelaskan pada sub-bab perancangan sistem diantaranya sebagai berikut.

5.1.1 Perancangan Arsitektur Sistem

Pada bagian ini akan menjelaskan bagaimana komunikasi yang dilakukan oleh aplikasi sistem rekam medis dengan *server* rekam medis terpusat sesuai dengan arsitektur *REST web service*. Arsitektur sistem yang digunakan dapat dilihat pada gambar 5.1 berikut.



Gambar 5.1 Arsitektur Sistem Informasi Rekam Medis Terpusat

Pada gambar 5.1 dapat dilihat bahwa terdapat tiga peran yang berinteraksi diantaranya adalah Puskesmas, Rumah Sakit dan Dinas Kesehatan. Puskesmas dan Rumah Sakit menggunakan layanan untuk mendapatkan informasi, mengolah dan mendistribusikan informasi rekam medis melalui *web service* yang disediakan oleh penyedia layanan yaitu Dinas Kesehatan. Dinas Kesehatan menyediakan layanan kepada Puskesmas atau Rumah Sakit untuk mengolah data rekam medis melalui *service* dan basis data hanya disimpan oleh Dinas Kesehatan sehingga layanan ini dapat disebut sebagai penyimpanan terpusat. Untuk mengirim permintaan kepada penyedia layanan Puskesmas dan Rumah Sakit mengirim data atau

parameter *value* melalui HTTP *request.* Kemudian *request* tersebut akan diperiksa dengan membaca *header* HTTP melalui *library* REST *web service*. *Request* yang berhasil dibaca kemudian dilanjutkan ke *database* untuk mencari informasi yang diminta. Setelah mendapatkan informasi, informasi tersebut kemudian dikembalikan kepada Puskesmas dan Rumah Sakit melalui *web service* dengan bentuk pesan menggunakan format *json*.

5.1.2 Pemodelan *Class Diagram*

Pemodelan *class diagram* dapat digunakan dalam penerapan *Object Oriented Programing (OOP).* Pada pemodelan implementasi sistem rekam medis terpusat memiliki beberapa *class* objek diantaranya *controller*, *model* dan *view*. Pe*model*an *class diagram* dapat dilihat pada gambar 5.2.

Pada gambar 5.2 merupakan rancangan penerapan sistem rekam medis dengan memanfaatkan *REST Web Service*. Pada pengembangan ini *REST* diimplementasikan menggunakan *framework CodeIgniter* sehingga *controller REST_Controller* menurunkan sifat dari *CI_Controller*. Kelas yang menurunkan sifat dari *REST_Controller* merupakan kelas pada *web service* diantaranya adalah Pasien, Rm, Resume dan Diagnosis. Tiap kelas yang menurunkan sifat dari *REST_Controller* masing-masing memiliki *method GET*, *POST*, *PUT* dan *DELETE*.

Dari seluruh total *class diagram* beberapa *class* akan dijelaskan detail fungsi yang ada didalamnya diantaranya:

1. *Class Rest_Controller*

   Detail dari *class Rest_Controller* dilihat pada tabel 5.1 berikut.

Tabel 5.1 *Class Rest_controller*

Nama Operasi	**Visibility**	**Tipe**
__construct($config = 'rest')	Public	Void
get_local_config($config_file)	Private	Void
function __destruct()	Public	Void
preflight_checks()	Protected	Void
_remap($object_called, $arguments = [])	Public	Void
response($data=NULL,$http_code=NULL, $continue=FALSE)	Public	Void
set_response($data=NULL,$http_code=NULL)	Public	Void
_detect_input_format()	Protected	Void
_get_default_output_format()	Protected	Void
_detect_output_format()	Protected	Void
_detect_method()	Protected	Void
_detect_api_key()	Protected	Void

Tabel 5.1 *Class Rest_controller* (lanjutan)

Nama Operasi	Visibility	Tipe
_detect_lang()	Protected	Void
_log_request($authorized=FALSE)	Protected	Void
_check_limit($controller_method)	Protected	Void
_auth_override_check()	Protected	Void
_parse_get()	protected	Void
_parse_post()	Protected	Void
_parse_put()	Protected	Void
_parse_head()	Protected	Void
_parse_options()	Protected	Void
_parse_patch()	Protected	Void
_parse_delete()	Protected	Void
_parse_query()	Protected	Void
get($key=NULL,$xss_clean=NULL)	Public	Void
options($key=NULL,$xss_clean=NULL)	Public	Void
head($key=NULL, $xss_clean=NULL)	Public	Void
post($key=NULL,$xss_clean=NULL)	Public	Void
put($key=NULL, $xss_clean=NULL)	Public	Void
delete($key=NULL, $xss_clean=NULL)	Public	Void
patch($key=NULL, $xss_clean=NULL)	Public	Void
query($key=NULL, $xss_clean=NULL)	Public	Void
_xss_clean($value, $xss_clean)	Protected	Void
validation_errors()	Public	Void
_perform_ldap_auth($username='',$password =NULL)	Protected	Void
_perform_library_auth($username='',$password = NULL)	Protected	Void
_check_login($username=NULL,$password= FALSE)	Protected	Void
_check_php_session()	Protected	Void
_prepare_basic_auth()	Protected	Void
_prepare_digest_auth()	Protected	Void
_check_blacklist_auth()	Protected	Void

Tabel 5.1 *Class Rest_controller* (lanjutan)

Nama Operasi	Visibility	Tipe
_check_whitelist_auth()	Protected	Void
_force_login($nonce = '')	Protected	Void
_log_access_time()	Protected	Void
_log_response_code($http_code)	Protected	Void
_check_access()	Protected	Void
_check_cors()	Protected	Void

Pada tabel 5.1 di atas merupakan detail dari fungsi yang ada pada *class Rest_Controller*. *Rest_Controller* merupakan objek *libraries* yang telah disediakan oleh pengembang *open source* untuk digunakan pada pengembangan *web service* dengan *framework CodeIgniter*.

2. *Class* Daftar

Detail dari *class* daftar dilihat pada tabel 5.2 berikut.

Tabel 5.2 *Class* Daftar

Nama Operasi	Visibility	Tipe
__construct($config = 'rest')	Public	Void
index_get()	Public	Void
index_post()	Public	Void
index_put()	Public	Void
index_delete()	Public	Void

Pada tabel 5.2 di atas merupakan fungsi yang terdapat pada *class* daftar. *Class* daftar merupakan objek untuk mengolah data pendaftaran pasien yang terdiri dari identitas lengkap pasien. Objek daftar menurunkan sifat induknya yaitu *Rest_Controller* untuk memanggil fungsi yang digunakan dalam mengolah data sehingga dapat menjalankan fungsi seperti *GET*, *POST*, *PUT* dan *DELETE* dari *libraries REST.* Objek lain yang memiliki fungsi sama diantaranya adalah Anamnesa, Datalab, Diagnosis, Dpss, Fisik, Kardio, Mukuloskleletal, Pemum, Pencernaan, Perkemihan, Pernafasan, Persyarafan, Petugas, Reproduksi, Resume dan Rm. Fungsi operasi yang dimiliki memiliki nama yang sama, namun memiliki data yang berbeda. Pada objek tertentu memiliki hubungan agregasi dengan objek lain dikarenakan terdapat atribut yang harus dimiliki oleh objek lain ketika objek tersebut akan dibuat.

3. *Class Login*

Detail dari *class login* dapat dilihat pada tabel 5.3 berikut.

Tabel 5.3 *Class login*

Nama Operasi	Visibility	Tipe
__construct()	public	void
index()	public	void
do_login()	public	Void
logout()	public	void

4. *Class* Pasien

Detail dari *class* pasien dapat dilihat pada tabel 5.4 berikut.

Tabel 5.4 *Class* Pasien

Nama Operasi	Visibility	Tipe
__construct()	public	void
index()	public	void
create()	public	void
create_func()	public	void
edit_func()	public	void
delete($id)	public	void

5. *Class* Rm

Detail dari *class* rm dapat dilihat pada Tabel 5.5 berikut.

Tabel 5.5 *Class* rm

Nama Operasi	Visibility	Tipe
__construct()	public	void
index()	public	void
rm_pasien()	public	void
create()	public	void
create_func()	public	void
edit_func()	public	void
delete($id)	public	void
all()	public	void
Info()	public	void
Upload	public	void

Tabel 5.5 *Class* rm (lanjutan)

Nama Operasi	Visibility	Tipe
Download	public	void
hapus_file	public	void

Pada tabel 5.5 di atas objek rm berbeda dengan yang menurunkan sifat dari *Rest_Controller* karena objek ini digunakan untuk melakukan permintaan dari klien untuk *server*.

6. *Class* Anamnesa

Detail dari *class* anamnesa dapat dilihat pada tabel 5.6 berikut.

Tabel 5.6 *Class* anamnesa

Nama Operasi	Visibility	Tipe
__construct()	public	void
index()	public	void
create()	public	Void
edit()	public	void
delete($id)	public	void

Pada tabel 5.6 di atas objek anamnesa juga berbeda dengan yang menurunkan sifat dari *Rest_Controller* karena objek ini digunakan untuk melakukan permintaan dari klien untuk server. Beberapa objek memiliki fungsi yang sama diantaranya adalah Cerna, Dpss, Fisik, Kardio, Kemih, Lab, Musku, Nafas, Pemum, Petugas, Repro dan Syaraf.

7. *Class* Diagnosis

Detail dari diagnosis dapat dilihat pada tabel 5.7 berikut.

Tabel 5.7 *Class* diagnosis

Nama Operasi	Visibility	Tipe
__construct()	public	void
index()	public	void
search()	public	void
hasil()	public	void
search_func()	public	void
create()	public	void
create_func()	public	void

Tabel 5.7 *Class* diagnosis (lanjutan)

Nama Operasi	Visibility	Tipe
edit()	public	void
delete($id)	public	void

Pada tabel 5.7 di atas sama dengan tabel sebelumnya yang merupakan objek pada klien yang mengolah data untuk dikirim ke sisi *server*.

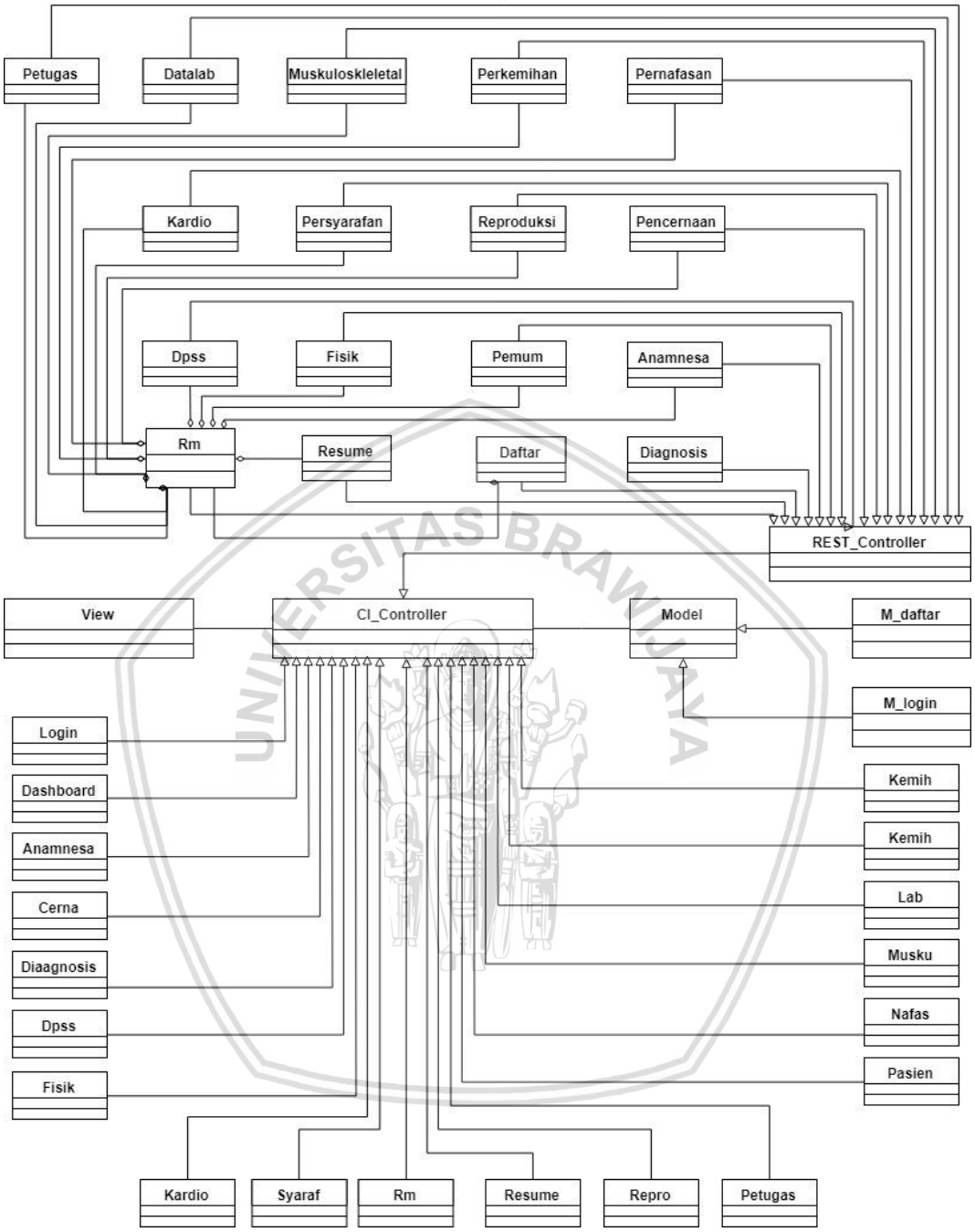
8. *Class* Resume

Detail dari resume dapat dilihat pada tabel 5.8 berikut.

Tabel 5.8 *Class* resume

Nama Operasi	Visibility	Tipe
__construct()	public	void
resume_pasien()	public	void
detail()	public	void
hasil()	public	void
create()	public	void
create_func()	public	void
edit()	public	void
delete($id)	public	void

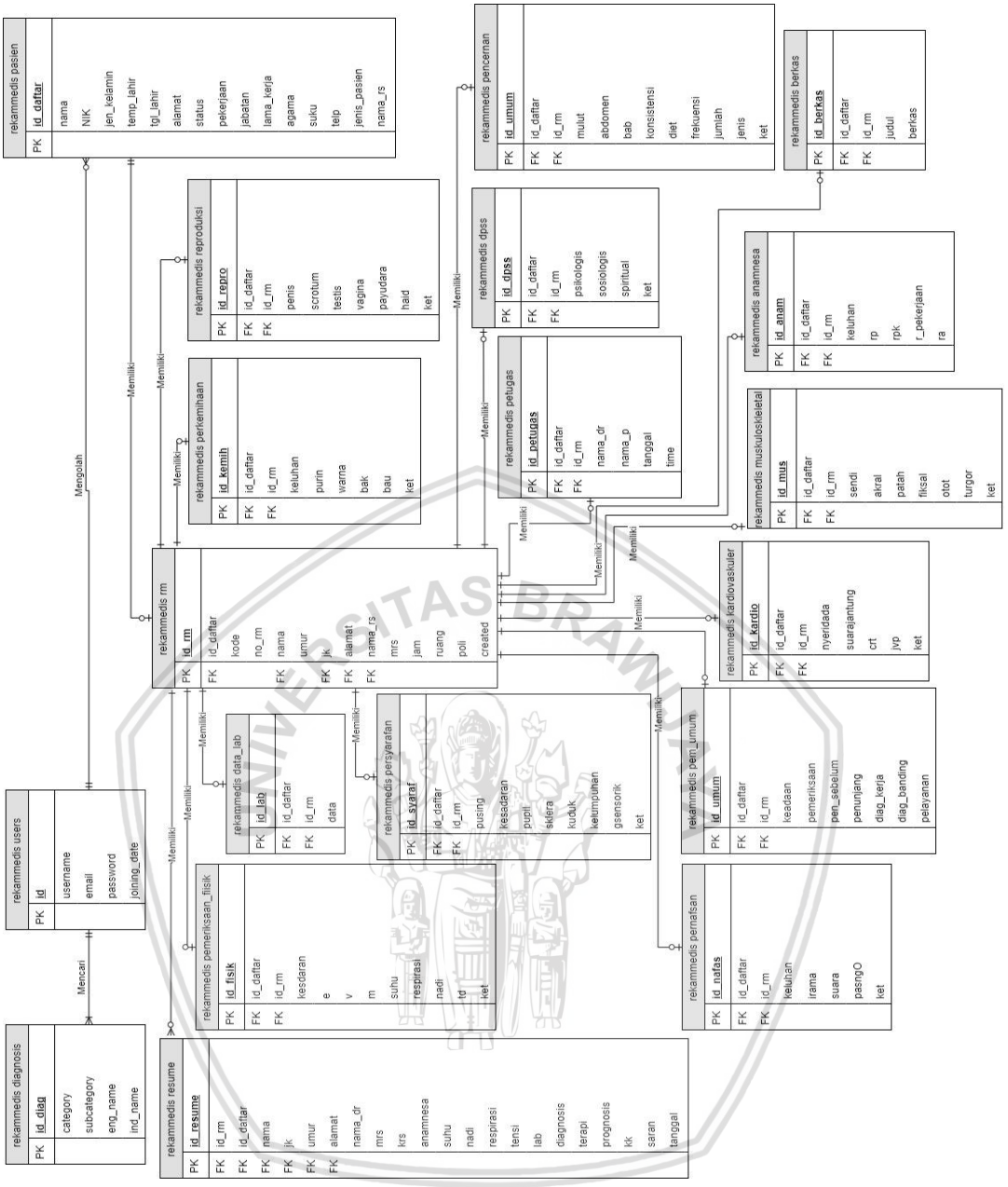
Perancangan *class diagram* dapat diperhatikan pada gambar 5.2 berikut.



Gambar 5.2 Perancangan *class diagram* sistem rekam medis terpusat

5.1.3 *Entity Relation Diagram*

Entity Relation Diagram (ERD) digunakan untuk menggambarkan hubungan antara entitas yang ada pada sistem dan akan digunakan sebagai dasar dalam implementasi basis data. *Database* yang digunakan adalah *database* rekammedis yang berada pada *server*. Perancangan *ERD* menggunakan tipe *logical data model* yang dapat dilihat pada gambar 5.3 berikut.



Gambar 5.3 *Entity Relation Diagram* (ERD)

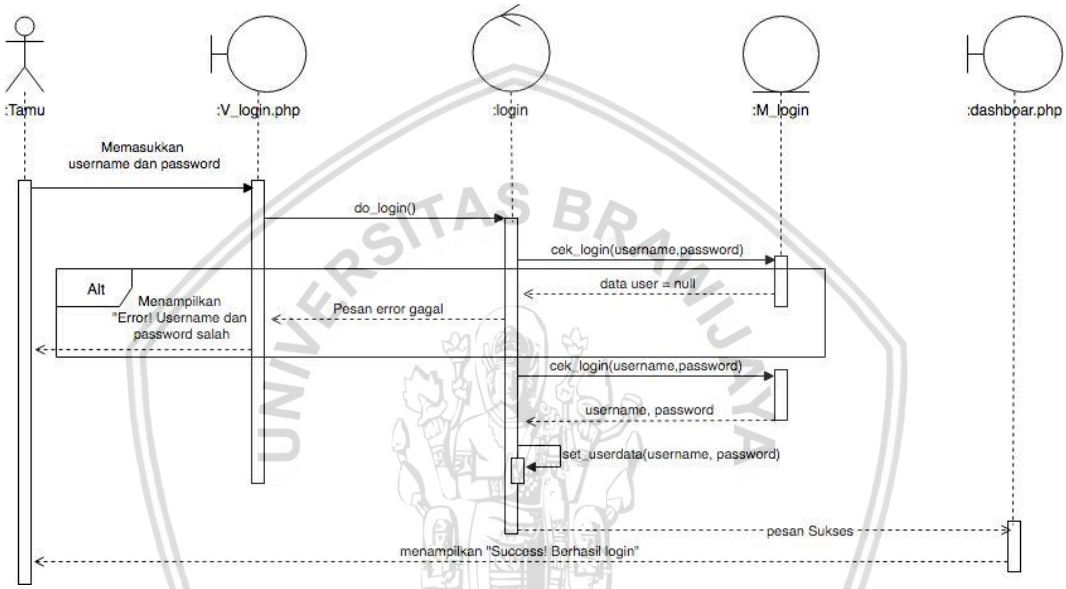
Pada gambar 5.3 di atas menggambarkan hubungan antara entitas pada sistem rekam medis terpusat. *Database* rekammedis memiliki beberapa entitas yaitu dblogin, diagnosis, resume, pemeriksaan_fisik, data_lab, persyarafan, rm, perkemihan, reproduksi, pasien, pencernaan, dpss, petugas, anamnesa, muskuloskleletal, kardiovaskuler, pem_umum, dan pernafasan dan berkas. Entitas *users* memiliki hubungan mencari dengan entitas diagnosis dan hubungan mengolah entitas pasien. Entitas pasien memiliki hubungan *one-to-one* dengan entitas rm. Entitas rm memliki hubungan *one-to-one* dengan beberapa entitas diantaranya adalah entitas anamnesa, perkemihan, reproduksi, pencernaan, dpss, petugas, muskuloskleletal, kardiovaskuler, pem_umum, berkas dan pernafasan.

Sedangkan entitas rm juga memiliki hubungan *one-to-many* dengan entitas resume.

5.1.4 Pemodelan *Sequence Diagram*

Sequence diagram merupakan diagram yang menggambarkan hubungan antar objek dan alur kerja setiap fungsi sistem. *Sequence diagram* dapat dibuat berdasarkan *use case scenario* dan hasil pemodelan *class diagram* yang telah dideskripsikan pada tahap sebelumnya, beberapa *sequence diagram* dapat dilihat pada bagian berikut.

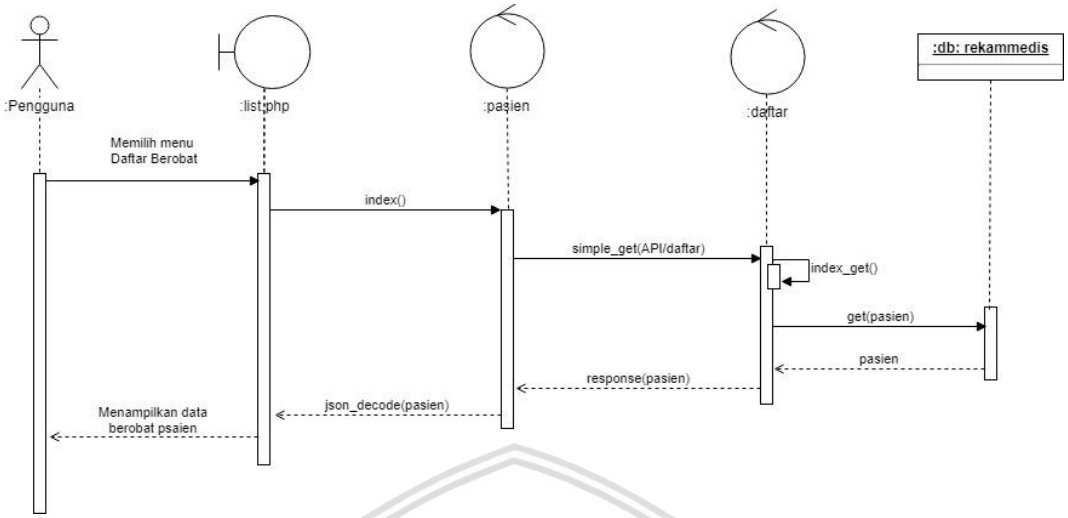
1. *Sequence Diagram Login*



Gambar 5.4 *Sequence diagram Login*

Pada gambar 5.4 menggambarkan interaksi antar objek yang saling berhubungan dalam fungsi *login*. Objek yang terlibat di dalamnya antara lain adalah aktor Tamu, *boundary V_login* dan *dashboard*, *control login* dan *Entity M_login*. Tamu sebagai aktor tanpa otoritas akan melakukan *login* sebelum dapat akses sistem rekam medis dengan memasukkan data *username* dan *password*, kemudian *control login* menjalankan proses *login* dengan *method do_login().* Kemudian data yang telah dimasukkan akan diperiksa dalam *entity M_login* dengan menggunakan *method cek_login()*. Ketika data tidak ditemukan atau bernilai nol maka akan mengembalikan pesan *error*, namun ketika data ditemukan maka data akan dimasukkan ke dalam *session* dengan *method set_userdata(username,password*). Setelah berhasil *login* maka akan diteruskan ke halaman *dashboard* sistem rekam medis.

2. *Sequence Diagram* Daftar Berobat



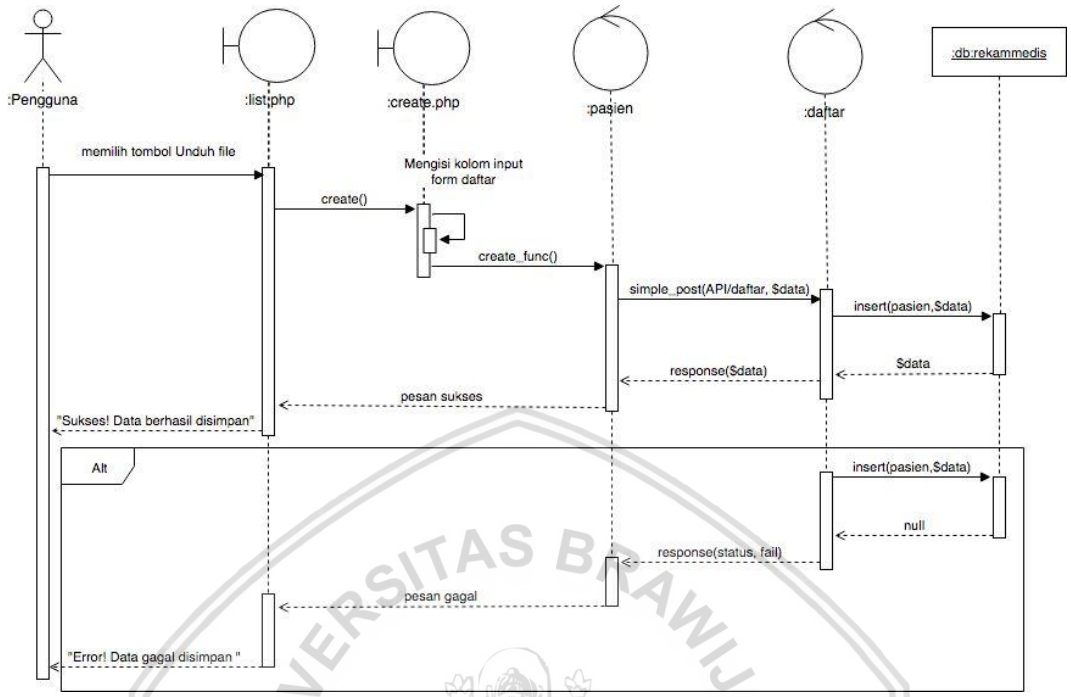
Gambar 5.5 *Sequence diagram* Daftar Berobat

Pada gambar 5.5 menggambarkan interaksi antar objek yang saling berhubungan dalam fungsi Daftar Berobat. Objek yang terlibat di dalamnya adalah aktor Pengguna, *boundary list*, *control* pada *client* yaitu pasien dan *control* pada *REST* yaitu daftar dan *database* rekammedis. *Control Client*/pasien merupakan *control* yang berada pada klien dengan nama *control* pasien, sedangkan *REST*/daftar merupakan *control server* (*web service*) dengan nama *control* daftar. Pengguna memilih menu Daftar Berobat untuk melihat daftar pasien berobat yang terdaftar pada rekam medis terpusat dengan menggunakan *method index*(). Kemudian *control Client*/pasien menggunakan *method simple_get*() dengan parameter *HTTP link service* untuk mengakses *control* REST/daftar. *Service* menjalankan *method index_get*() untuk melakukan proses *GET*, kemudian mengambil data pasien dengan *method get*() dari *database* rekammedis. Setelah mendapatkan data maka *service* akan mengirim *response* berupa *json* dengan data pasien. Data *json* kemudian dipecah kembali dan ditampilkan pada menu Daftar Berobat.

3. *Sequence Diagram* Tambah Daftar

Pada gambar 5.6 menggambarkan interaksi antar objek yang saling berhubungan dalam fungsi Tambah Daftar. Objek yang saling berhubungan diantaranya adalah aktor Pengguna, *boundary list* dan *create*, *control Client*/pasien dan REST/daftar, *entity* M_daftar dan *database* rekammedis. Pengguna menekan tombol Tambah ketika berada pada halaman Daftar Berobat, kemudian halaman akan dialihkan menuju halaman Tambah Daftar Pasien Berobat. Pengguna mengisi kolom *input* data dan data akan diproses pada *control Client*/pasien menggunakan *method create_func()*. Setelah data dimasukkan dalam bentuk variabel *array* $data, maka klien mengirim *request* pada *server* menggunakan *simple_post*() untuk menjalankan fungsi *POST service*. Data yang sampai pada *service* akan dimasukkan pada *database*. Setelah data berhasil

dimasukkan maka *server* akan mengirim status data kepada klien. Klien akan menerima respon data sebagai pesan.



Gambar 5.6 *Sequence diagram* Tambah Daftar

5.1.5 Perancangan Algoritme

Perancangan algoritme merupakan salah satu tahap perancangan yang digunakan untuk dasar logika kode program. Algoritme menjelaskan langkah-langkah logika untuk menjalankan sebuah fungsi dan mengeluarkan hasil sesuai tujuan pada sistem. Pada perancangan algoritme direpresentasikan dalam bentuk bentuk *pseudocode.* Pada tahap ini akan dijelaskan hanya lima fungsi yang akan dijelaskan karena hampir banyak fungsi yang sama diantaranya adalah *do_login*() pada kelas *LoginController*, *create_func*() pada kelas Pasien*Controller*, *index_post*() pada kelas Daftar*Controller*, *edit_func*() pada kelas Pasien*Controller* dan *index_put*() pada kelas Daftar*Controller*. Masing-masing fungsi algoritme dapat dijelaskan selengkapnya pada bagian berikut.

1. Algoritme fungsi *do_login*() kelas *LoginController*

Penjelasan algoritme fungsi *do_login*() dapat dilihat pada tabel 5.9 berikut.

Tabel 5.9 Algoritme fungsi do_*login*()

1	*Func*tion do_*login* start
2	if(set *POST*[*login*]) then
3	kode = input POST(*kode*)
4	*password* = input POST(*password*)
5	cek = m_login->cek_login(users,where)
6	if (cek->num_rows > 0) then
7	data = cek->row_array
8	if(data=[level]!= 1) then

Tabel 5.9 Algoritme fungsi *do_login*() (lanjutan)

```
9            session->set_userdata(status,login)
10           session->set_userdata(status,kode)
11         session->set_userdata(status,data[nama])
12         session->set_flashdata(sukses,berhasil login)
13         redirect(dashboard)
14       end if
15       else
16         print “anda memasuki halaman dinas kesehatan”
17       end if
18     end if
19  else
20       session->set_flashdata(gagal,salah username dan password)
21       load->view(main/headlogin)
22       load->view(V_login)
23       load->view(main/footer)
24   end if
25 end if
26 end
```

Pada tabel 5.9 merupakan algoritme ketika sistem menjalankan proses *login* atau melakukan pemeriksaan otoritas pengguna ketika akan masuk ke dalam sistem. Fungsi berjalan ketika pengguna menekan tombol *login*, fungsi akan melakukan cek data masukan kode dan *password* dengan melakukan inisialisasi variabel cek dan mengirim data ke *model* untuk melakukan pemeriksaan pada *database*. Jika data tidak bernilai nol maka data dimasukkan ke data *session* dan dapat melanjutkan ke dalam sistem dan menampilkan pesan sukses.

2. Algoritme fungsi *create_func()* pada Pasien *Controller*

Penjelasan algoritme *create_func()* dapat dilihat pada tabel 5.10 berikut.

Tabel 5.10 Algoritme *create_func*()

```
1  Function create_func() start
2        config = array(
3          array(field = NIK, label = NIK, rules = required|trim),
4          array(field = nama, label = Nama,rules = required|trim),
5          array(field = temp_lahir, label = Tempat Lahir,rules = required),
6          array(field = tgl_lahir, label = Tanggal Lahir,rules = required),
7        )
8        Form validation set_rules(config)
9        if(Form validation run() = FALSE) then
10         create()
11       else
12         data = array(
13           id_daftar          =  input POST(id_daftar),
14           nama               = input POST (nama),
15           NIK                = input POST (NIK),
```

Tabel 5.10 Algoritme *create_func*() (lanjutan)

```
16              jen_kelamin      = input POST (jen_kelamin),
17              temp_lahir       = input POST (temp_lahir),
18              tgl_lahir        = input POST (tgl_lahir),
19              alamat           = input POST (alamat),
20              status           = input POST (status),
21              pekerjaan        = input POST (pekerjaan),
22              jabatan          = input POST (jabatan),
23              lama_kerja       = input POST (lama_kerja),
24              agama            = input POST (agama),
25              suku             = input POST (suku),
26              telp             = input POST (telp),
27              jenis_pasien     = input POST (jenis_pasien),
28              nama_rs          = input POST (nama_rs))
29          insert =  curl simple POST(API./daftar, data)
30          if(insert) then
31              session set flashdata(Csukses,insert data sukses)
32          else
33              session set flashdata(Cgagal,insert data gagal)
34          end if
35          redirect(pasien)
36      end if
37  end
```

Pada tabel 5.10 merupakan algoritme ketika sistem menjalankan proses tambah daftar berobat pasien pada kelas pasien. Ketika pengguna telah selesai memasukkan data pada *form* tambah daftar maka proses ini akan dijalankan. Data yang telah dimasukkan disimpan pada variabel data, kemudian variabel baru dengan nama *insert* dibuat untuk mengirim data ke *service*. Jika variabel *insert* berhasil mengirim data dan menerima *response success* maka akan menampilkan pesan sukses, jika mendapat *response error* maka akan menampilkan pesan gagal.

3. Algoritme fungsi *index_post*() pada Daftar*Controller*

Penjelasan algoritme *index_post*() dapat dilihat pada tabel 5.11 berikut.

Tabel 5.11 Algoritme *index_post*()

```
1   Function index_POST() start
2       Load model(m_daftar)
3       data = array(
4           id_daftar        = POST(id_daftar),
5           nama             = POST(nama),
6           NIK              = POST(NIK),
7           jen_kelamin      = POST(jen_kelamin),
8           temp_lahir       = POST(temp_lahir),
9           tgl_lahir        = POST(tgl_lahir),
10          alamat           = POST(alamat),
11          status           = POST(status),
```

Tabel 5.11 Algoritme *index_post*() (lanjutan)

```
12              pekerjaan          = POST(pekerjaan),
13              lama_kerja         = POST(lama_kerja),
14              agama              = POST(agama),
15              suku               = POST(suku),
16              telp               = POST(telp),
17              jenis_pasien       = POST(jenis_pasien),
18              nama_rs            = POST(nama_rs))
19        NIK       = POST(NIK)
20        $nama  = POST(nama)
21        if (user exists(nama, NIK)=TRUE) then
22           response(array(status = Pasien sudah terdaftar!), 405)
23        else
24           insert = db insert(pasien, data)
25           if (insert) then
26              response(data, 200)
27           else
28              response(array(status = fail, 502))
29           end if
30        end if
31     end
```

Pada tabel 5.11 merupakan algoritme dari proses *POST* pada *Web Service* ketika melakukan tambah daftar pasien berobat. Pada saat klien mengirim data dengan memanggil *method POST* maka fungsi ini akan dijalankan. Pada inisialisasi variabel data *service*, data dari klien dimasukkan dalam bentuk *array*. Data NIK dan Nama dikirim untuk melakukan pengecekan data terdaftar yang sama menggunakan *method user_exits*(). Ketika data yang sama ditemukan maka akan mengirim *response* dengan pesan status pasien sudah terdaftar, namun jika data yang sama tidak ditemukan maka akan menjalankan *method insert*() pada *database*. Jika data berhasil dimasukkan maka akan mengirim *response* data, jika gagal akan mengirim *response* status *fail*.

4. Algoritme fungsi *edit_func*() pada Pasien*Controller*

Penjelasan algoritme fungsi *edit_func*() dapat dilihat pada tabel 5.12 berikut.

Tabel 5.12 Algoritme fungsi *edit_func*()

```
1   Function edit_func() start
2
3           data = array(
4              id_daftar          =  input POST(id_daftar),
5              nama               = input POST (nama),
6              NIK                = input POST (NIK),
7              jen_kelamin        = input POST (jen_kelamin),
8              temp_lahir         = input POST (temp_lahir),
9              tgl_lahir          = input POST (tgl_lahir),
```

Tabel 5.12 Algoritme fungsi *edit_func()* (lanjutan)

```
10            alamat          = input POST (alamat),
11            status          = input POST (status),
12            pekerjaan       = input POST (pekerjaan),
13            jabatan         = input POST (jabatan),
14            lama_kerja      = input POST (lama_kerja),
15            agama           = input POST (agama),
16            suku            = input POST (suku),
17            telp            = input POST (telp),
18            jenis_pasien    = input POST (jenis_pasien),
19            nama_rs         = input POST (nama_rs))
20        update =  curl simple PUT (API./daftar, data,
21                        array(CURLOPT_BUFFERSIZE =  99))
22        if(update) then
23          session set flashdata(Esukses,Update Data Berhasil)
24        else
25          session set flashdata(Egagal,Update Data Gagal)
26        end if
27        redirect(pasien)
28  end
```

Pada gambar 5.12 merupakan algoritme pada proses *edit* data pasien berobat. Fungsi ini dijalankan ketika pengguna telah selesai mengubah data pada halaman *edit* daftar pasien berobat. Ketika halaman *edit* dieksekusi maka data yang telah diubah dimasukkan ke dalam variabel data, kemudian variabel baru dibuat dengan nama *update*. Variabel *update* digunakan untuk mengirim data kepada *web service*, jika variabel *update* berhasil dieksekusi maka akan menampilkan pesan sukses dan jika menerima *response fail* maka akan menampilkan pesan gagal.

5. Algoritme fungsi *index_put()* pada Daftar*Controller*

Penjelasan algoritme fungsi *index_put*() dapat dilihat pada tabel 5.13 berikut.

Tabel 5.13 Algoritme fungsi *index_put*()

```
1   Function index_PUT() then
2       id = PUT(id_daftar)
3       data = array(
4           id_daftar       =  PUT (id_daftar),
5           nama            = PUT (nama),
6           NIK             = PUT (NIK),
7           jen_kelamin     = PUT (jen_kelamin),
8           temp_lahir      = PUT (temp_lahir),
9           tgl_lahir       = PUT (tgl_lahir),
10          alamat          = PUT (alamat),
11          status          = PUT (status),
12          pekerjaan       = PUT (pekerjaan),
13          jabatan         = PUT (jabatan),
```

Tabel 5.13 Algoritme fungsi *index_put()* (lanjutan)

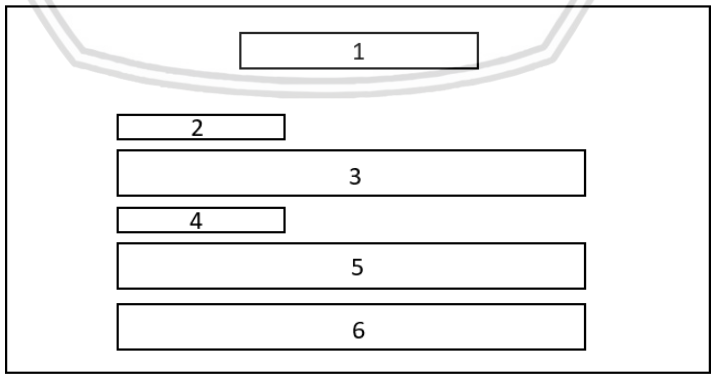
```
14          lama_kerja      = PUT (lama_kerja),
15          agama           = PUT (agama),
16          suku            = PUT (suku),
17          telp            = PUT (telp),
18          jenis_pasien    = PUT (jenis_pasien),
19          nama_rs         = PUT (nama_rs))
20      db where(id_daftar, id)
21      update = db update(pasien, data)
22      if (update) then
23        response(data, 200)
24      else
25        response(array(status = fail, 502))
26      end if
27    end
```

Pada gambar 5.13 merupakan algoritme dari proses *PUT* ketika melakukan *update* data pasien berobat. Fungsi memeriksa perintah *update* dengan mengecek id_daftar. Kemudian data dari klien yang telah dikirim dimasukkan ke dalam variabel data. Setelah mengecek *id* pada *database* maka menjalankan variabel *update*, jika berhasil akan mengirim *response* berupa data yang berhasil diperbarui namun jika gagal akan mengirim *response* pesan status *fail*.

5.1.6 Perancangan Antarmuka

Perancangan antarmuka merupakan tahap untuk menggambarkan rancangan tampilan halaman sistem yang akan dibuat. Perancangan antarmuka pada umumnya dinotasikan dengan gambar dua dimensi menyerupai persegi atau persegi panjang dengan memiliki nomor berupa angka di dalamnya untuk menandakan spesifik fungsi pada sistem.

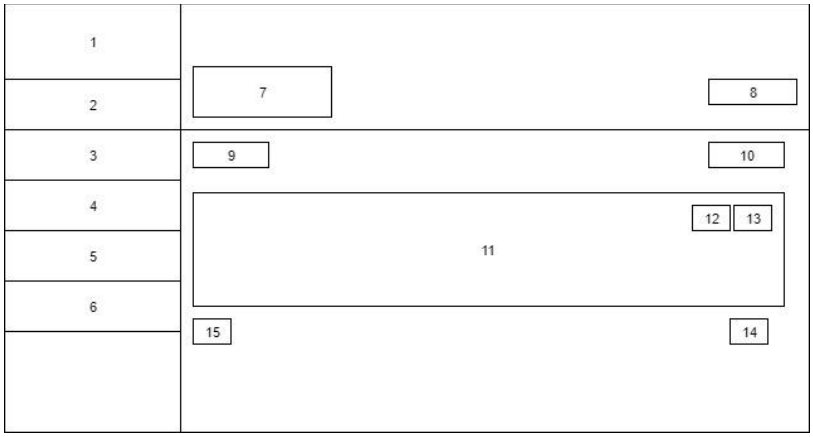
1. Antarmuka *Login*



Gambar 5.7 Perancangan antarmuka *login*

Pada gambar 5.24 merupakan perancangan antarmuka halaman *login*. Kotak bernomor 1 merupakan nama sistem, kotak nomor 2 dan 4 merupakan label untuk *username* dan *password*, kotak nomor 3 dan 5 merupakan kolom isi kode dan *password* dan kotak nomor 6 merupakan tombol *login*.

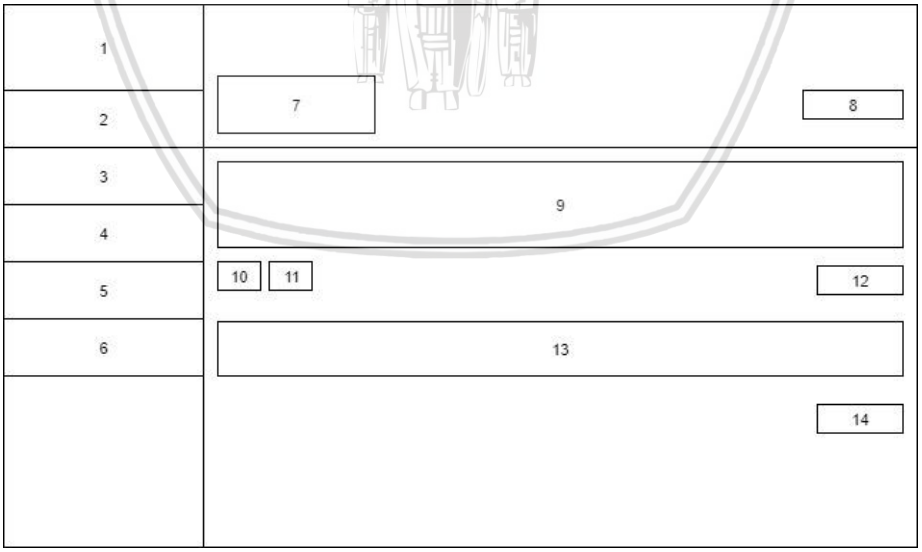
2. Antarmuka Daftar Berobat



Gambar 5.8 Perancangan antarmuka daftar berobat

Pada gambar 5.25 merupakan perancangan antarmuka halaman daftar berobat. Kotak nomor 1 merupakan nama sistem. Kotak nomor 2 sampai 6 merupakan fitur menu pada sistem. Kotak nomor 7 merupakan keterangan nama halaman, kotak nomor 8 merupakan tombol tambah daftar berobat pasien, kotak nomor 9 untuk mengatur banyak daftar yang ditampilkan. Kotak nomor 10 merupakan kolom pencarian pada daftar tabel. Kotak nomor 11 merupakan daftar pasien berobat. Kotak nomor 12 tombol untuk edit data pasien dan kotak nomor 13 untuk hapus data pasien berobat. Kotak nomor 14 untuk mengganti halaman tabel selanjutnya atau sebelumnya. Kotak nomor 15 merupakan tombol *toggle* untuk menyembunyikan navigasi menu.

3. Antarmuka Rekam Medis Pasien



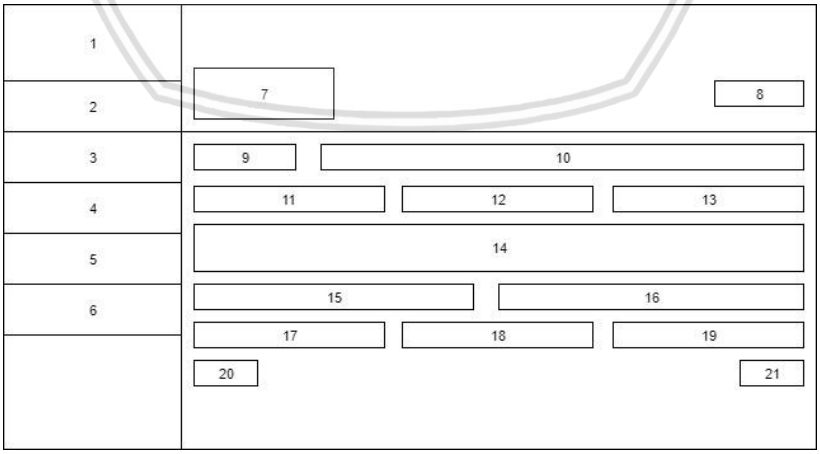
Gambar 5.9 Perancangan antarmuka halaman rekam medis pasien

Pada gambar 5.26 merupakan perancangan antarmuka rekam medis pasien. Kotak nomor 1 sampai 6 sama dengan keterangan pada antarmuka sebelumnya. Kotak nomor 7 merupakan keterangan halaman. Kotak nomor 8 merupakan tombol untuk kembali ke halaman sebelumnya. Kotak nomor 9

merupakan kolom tabel informasi rekam medis pasien. Kotak nomor 10 merupakan tombol untuk *edit* data rekam medis. Kotak nomor 11 merupakan tombol untuk melihat informasi resume pasien. Kotak nomor 12 merupakan tombol untuk mengunggah berkas rekam medis pasien dalam bentuk file. Kotak nomor 13 merupakan tombol *tab* untuk menampilkan data anamnesa, pemeriksaan umum, pemeriksaan fisik, pernafasan, kardiovaskuler, sistem persyarafan, perkemihan, sistem pencernaan, sistem muskuloskleletal sistem reproduksi, data psikologi sosiologi dan spiritual, data pemeriksaan penunjang dan petugas. Kotak nomor 14 adalah tombol *toggle* untuk menyembunyikan navigasi menu.

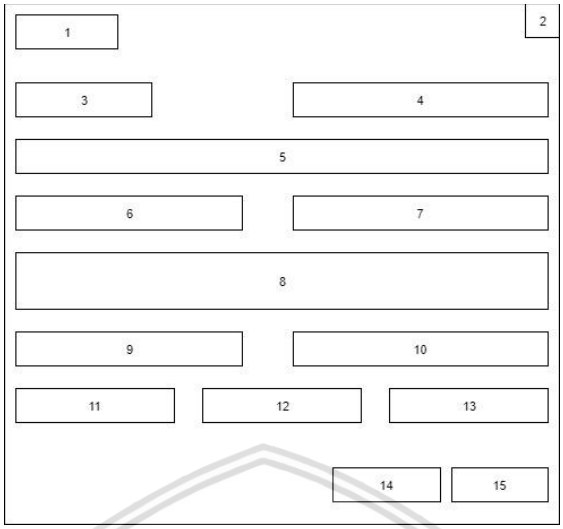
4. Antarmuka Tambah RM

Pada gambar 5.27 merupakan perancangan antarmuka halaman rekam medis pasien. Kotak nomor 1 sampai 6 sama dengan antarmuka sebelumnya. Kotak nomor 7 merupakan keterangan nama dari halaman. Kotak nomor 8 merupakan tombol untuk kembali ke halaman sebelumnya. Kotak nomor 9 merupakan kolom input untuk memasukkan kode rekam medis pasien berdasarkan lokasi pasien tinggal. Kotak nomor 10 merupakan kolom nomor rekam medis berdasarkan urutan pasien mendaftar. Kotak nomor 11 merupakan kolom untuk memasukkan data nama pasien. Kotak nomor 12 kolom untuk memasukkan data jenis kelamin pasien. Kotak nomor 13 untuk memasukkan data umur pasien. Kotak nomor 14 kolom untuk memasukkan data alamat pasien. Kotak nomor 15 kolom untuk memasukkan data tanggal masuk rumah sakit. Kotak nomor 16 kolom untuk memasukkan data jam masuk rumah sakit. Kotak nomor 17 kolom untuk memasukkan nama rumah sakit pasien dirawat setelah mendaftar. Kotak nomor 18 kolom untuk memasukkan data ruangan pasien dirawat. Kotak nomor 19 kolom untuk memasukkan data poli pasien berobat. Kotak nomor 20 tombol untuk menyimpan data rekam medis yang telah dimasukkan. Kotak nomor 21 tombol *toggle* untuk menutup navigasi menu.



Gambar 5.10 Perancangan antarmuka tambah rekam medis

5. Antarmuka *Edit* data rekam medis

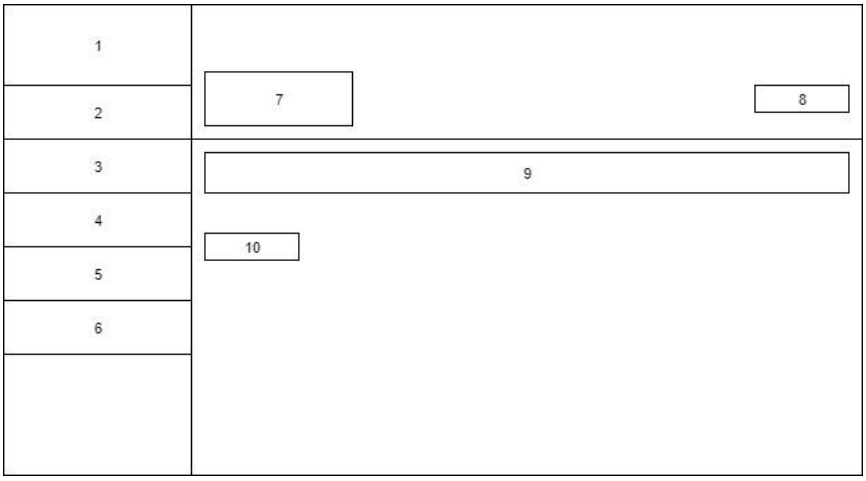


Gambar 5.11 Perancangan antarmuka resume medis pasien

Pada gambar 5.28 merupakan perancangan antarmuka *pop-up form Edit* RM. Kotak nomor 1 merupakan keterangan dari *pop-up form.* Kotak nomor 2 merupakan tombol untuk menutup *pop-up*. Kotak nomor 3 merupakan kolom untuk memasukkan kode rekam medis berdasarkan tempat tinggal pasien dan kotak nomor 4 untuk memasukkan nomor rekam medis berdasarkan urutan daftar pasien berobat. Kotak nomor 5 kolom untuk memasukkan nama. Kotak nomor 6 untuk data umur dan kotak nomor 7 kolom untuk memasukkan data jenis kelamin. Kotak nomor 8 kolom untuk memasukkan alamat. Kotak nomor 9 kolom untuk memasukkan data tanggal masuk rumah sakit dan kotak nomor 10 kolom untuk memasukkan jam masuk rumah sakit. Kotak nomor 11, 12 dan 13 secara berurutan kolom untuk memasukkan data ruang pasien dirawat, nama rumah sakit yang merawat dan poli penunjang. Kotak nomor 14 merupakan tombol untuk menyimpan perubahan data dan kotak nomor 15 tombol untuk menutup *pop-up*.

6. Antarmuka Cari Diagnosis

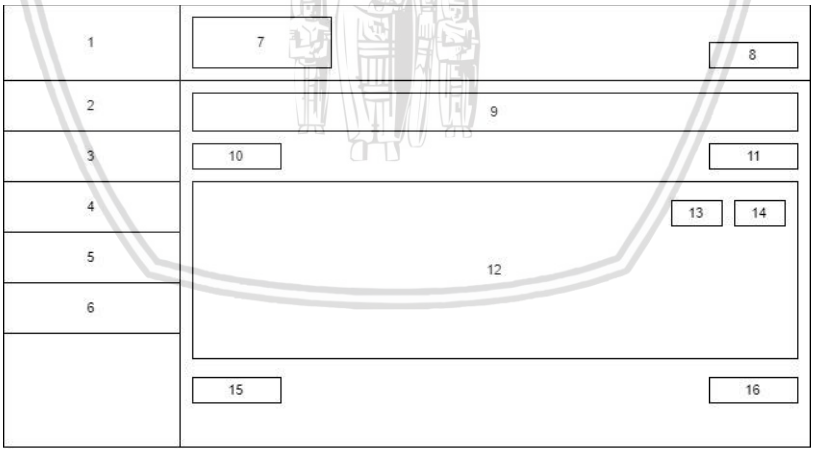
Pada gambar 5.29 merupakan perancangan antarmuka halaman cari diagnosis. Kotak nomor 1 sampai 6 merupakan menu pada navigasi. Kotak nomor 7 merupakan keterangan halaman. Kotak nomor 8 adalah tombol tambah untuk menambahkan data diagnosis baru. Kotak nomor 9 kolom untuk melakukan pencarian berdasarkan nomor *category* diagnosis. Kotak nomor 10 adalah tombol *toggle* untuk menyembunyikan navigasi menu.



Gambar 5.12 Perancangan antarmuka cari diagnosis

7. Antarmuka Daftar Diagnosis

Pada gambar 5.30 merupakan perancangan antarmuka halaman daftar diagnosis. Kotak nomor 1 sampai 6 merupakan navigasi menu. Kotak nomor 7 keterangan halaman. Kotak nomor 8 merupakan tombol kembali ke halaman sebelumnya. Kotak nomor 9 adalah kolom pencarian dengan *category*. Kotak nomor 10 pengurutan banyak data yang ditampilkan. Kotak nomor 11 pencarian data pada tabel yang ditampilkan. Kotak nomor 12 merupakan data diagnosis yang ditampilkan. Kotak nomor 13 tombol untuk melakukan *Edit* data. Kotak nomor 14 adalah tombol untuk menghapus data. Kotak 15 untuk *toggle* navigasi menu dan kotak nomor 16 untuk menampilkan data pada tabel selanjutnya.



Gambar 5.13 Perancangan antarmuka daftar diagnosis

5.2 Implementasi

Setelah menyelesaikan tahap perancangan sistem maka selanjutnya melakukan tahap implementasi. Perancangan yang telah dibuat maka akan digunakan sebagai dasar untuk melakukan tahap implementasi. Pada tahap implementasi akan menjelaskan spesifikasi sistem dalam perangkat lunak dan perangkat keras yang digunakan, implementasi arsitektur, implementasi *class*,

implementasi algoritme, implementasi basis data dan implementasi antarmuka. Penjelasan selengkapnya dapat dilihat pada bagian berikut.

5.2.1 Spesifikasi Sistem

Pada bagian ini akan menjelaskan spesifikasi perangkat lunak dan perangkat keras yang digunakan selama mendukung tahap implementasi sistem.

5.2.1.1 Spesifikasi perangkat keras

Spesifikasi perangkat keras yang digunakan selama "Implementasi *Web Service* Sistem Informasi Rekam Medis Terpusat" dapat dilihat pada tabel 5.14 berikut.

Tabel 5.14 Spesifikasi perangkat keras

Nama Komponen	Keterangan
Harddisk	560 GB
Processor	*Intel*(R) *Core*(TM) i3-3217U CPU @ 1.80 GHz
RAM	4 GB DDR3M
Graphic Card	*Intel Graphics* 4000 *HD* & *NVIDIA* GT740M

5.2.1.2 Spesifikasi perangkat lunak

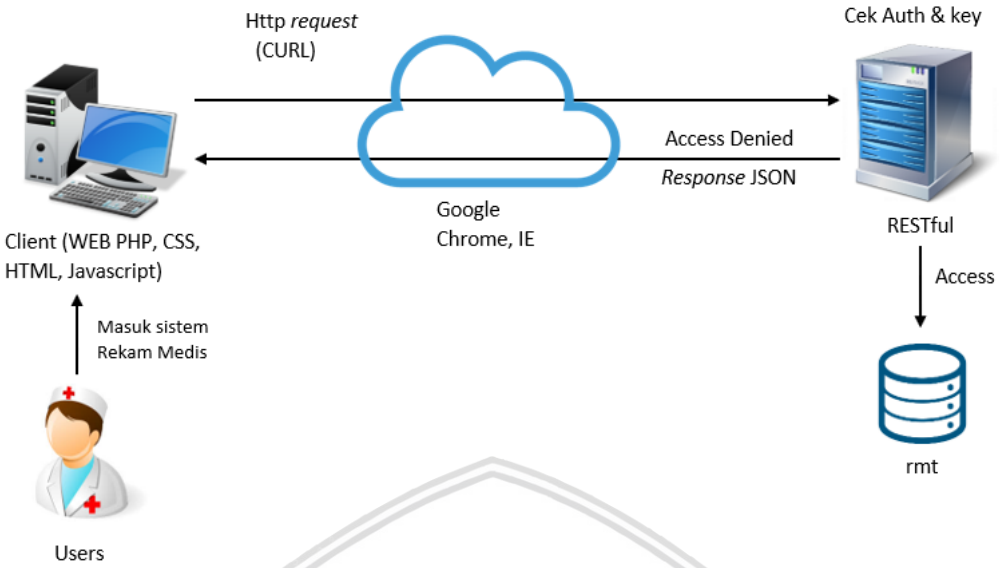
Spesifikasi perangkat lunak yang digunakan dalam implementasi sistem dapat dilihat pada tabel 5.15 berikut.

Tabel 5.15 Spesifikasi perangkat lunak

Hardware	Keterangan
Sistem Operasi	*Windows* 10 *Professional* 64-*bit*
Tools	*Sublime Text* 3
Bahasa Pemrograman	*PHP* 7, *HTML* 5, *CSS* 3, *Javascript*
DBMS	*MySQL phpMyadmin* 4.7
Server	*Localhost* (*XAMPP*) 3.2.2
Framework	*CodeIgniter* 3.1
Browser	*Google Chrome* 63.0.3239.132 & *Internet Explore* 11.192.16299.0
Service library	*RESTful Web Service*
Library client	*CURL*

5.2.2 Implementasi Arsitektur

Implementasi arsitektur menggambarkan rangkaian yang telah dirancang diimplementasikan dengan benar. Pada gambar 5.31 implementasi *client* menggunakan bahasa pemrograman *web*, kemudian ketika mengirim *request* pada *server* menggunakan *library CURL*. *CURL* digunakan untuk memindai data dari atau ke sebuah *server* tanpa interaksi dari masing-masing *user*. Setelah sampai pada *server*, kemudian akses diperiksa dengan autentikasi dan *key*, jika tidak sesuai maka akses akan ditolak. Ketika akses berhasil maka akan melakukan akses *database* dan mengirim kembali melalui *response json*.



Gambar 5.14 Implementasi arsitektur rekam medis terpusat

5.2.3 Implementasi *Class*

Implementasi *class* diterapkan dengan mengacu pada perancangan *class* di tahap sebelumnya. Setiap *class* yang didefinisikan pada *class diagram* akan diimplementasikan ke dalam bentuk *file* dengan ekstensi *(.php).* Daftar hasil implementasi *class* dapat dilihat pada tabel 5.16 berikut.

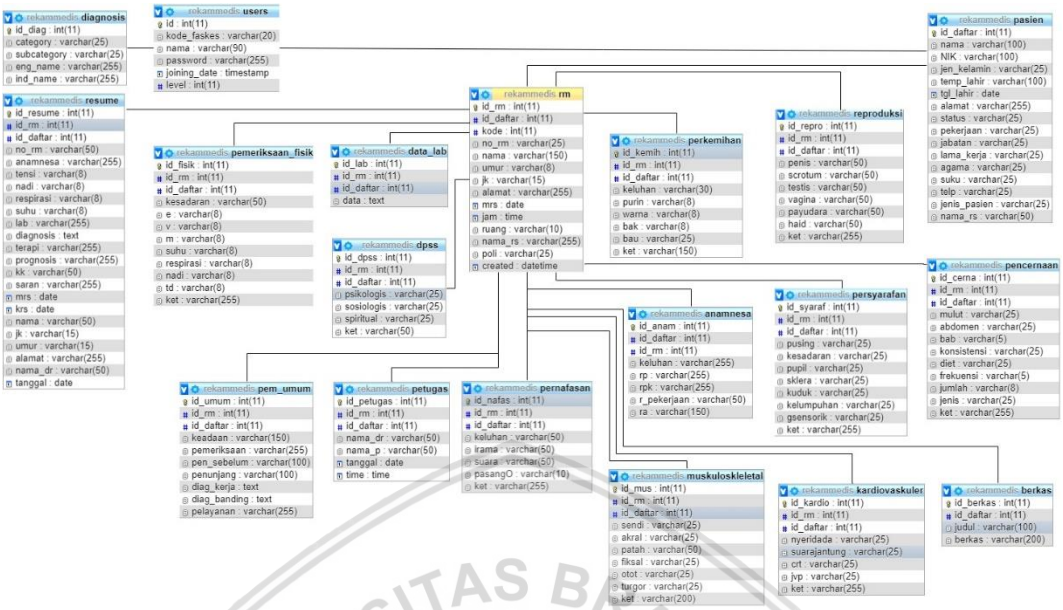
Tabel 5.16 Implementasi *class*

No	Nama *Class*	Nama File
1	Daftar	Server/application/controllers/Daftar.php
2	Diagnosis	Server/application/controllers/Diagnosis.php
3	Rm	Server/application /controllers/Rm.php
4	Resume	Server/application /controllers/Resume.php
5	Anamnesa	Server/application /controllers/Anamnesa.php
6	Pemum	Server/application /controllers/Pemum.php
7	Fisik	Server/application /controllers/Fisik.php
8	Dpss	Server/application /controllers/Dpss.php
9	Pencernaan	Server/application /controllers/Pencernaan.php
10	Reproduksi	Server/application /controllers/Reproduksi.php
11	Persyarafan	Server/application /controllers/Persyarafan.php
12	Kardio	Server/application /controllers/Kardio.php
13	Pernafasan	Server/application /controllers/Pernafasan.php
14	Perkemihan	Server/application /controllers/Perkemihan.php

Tabel 5.16 Implementasi *class* (lanjutan)

No	Nama *Class*	Nama File
15	Muskuloskleletal	Server/application /controllers/Muskuloskleletal.php
16	Datalab	Server/application /controllers/Datalab.php
17	Petugas	Server/application /controller/Petugas.php
18	M_daftar	Server/application/models/M_daftar.php
19	M_login	clientRM/application/models/M_login.php
30	Login	clientRM/application/controllers/Login.php
21	Dashboard	clientRM/application/controllers/Dashboard.php
22	Anamnesa	clientRM/application/controllers/Anamnesa.php
23	Cerna	clientRM/application/controllers/Cerna.php
24	Diagnosis	clientRM/application/controllers/Diagnosis.php
25	Dpss	clientRM/application/controllers/Dpss.php
26	Fisik	clientRM/application/controllers/Fisik.php
27	Kardio	clientRM/application/controllers/Kardio.php
28	Syaraf	clientRM/application/controllers/Syaraf.php
29	Rm	clientRM/application/controllers/Rm.php
30	Resume	clientRM/application/controllers/Resume.php
31	Repro	clientRM/application/controllers/Repro.php
32	Petugas	clientRM/application/controllers/Petugas.php
33	Kemih	clientRM/application/controllers/Kemih.php
34	Lab	clientRM/application/controllers/Lab.php
35	Musku	clientRM/application/controllers/Musku.php
36	Nafas	clientRM/application/controllers/Nafas.php
37	Pasien	clientRM/application/controllers/Pasien.php

5.2.4 Implementasi Basis Data



Gambar 5.15 Implementasi basis data

Pada gambar 5.32 menggambarkan implementasi basis data yang digunakan pada sistem informasi rekam medis terpusat. Dalam implementasinya sistem ini menggunakan *database* sebanyak total 19 tabel.

5.2.5 Implementasi Algoritme

Pada bagian ini akan menjelaskan mengenai implementasi dari perancangan algoritme pada tahap perancangan sebelumnya. Masing-masing algoritme yang telah dirancang diimplementasikan seperti pada bagian berikut.

1. Implementasi fungsi *do_login*

   Implementasi fungsi *do_login* dapat dilihat pada tabel 5.17.

Tabel 5.17 Implementasi fungsi *do_login*

```
1  function do_login(){
2  if(isset($_POST['login']))
3  {
4          $kode = $this->input->post('kode');
5          $password = $this->input->post('password');
6          $where = array(
7          'kode_faskes' => $kode,
8          'password' => md5($password));
9          $cek = $this->M_login->cek_login("users",$where);
10         if($cek->num_rows() > 0){
11                 $data =$cek->row_array();
12                 if ($data['level']!= "1") {
```

Tabel 5.17 Implementasi fungsi *do_login* (lanjutan)

```
13              $this->session->set_userdata('status',"login");
14              $this->session->set_userdata('kode',$kode);
15              $this->session->set_userdata('nama',$data['nama']);
16              $this->session->set_flashdata('sukses','Berhasil
login!');
17              redirect('dashboard');
18              }
19              else{
20              echo "anda memasuki halaman dinas kesehatan";
21              }
22      }
23      else {
24      $this->session->set_flashdata('gagal','Salah username dan
password!');
25      $this->load->view('main/headlogin');
26      $this->load->view('V_login');
27      $this->load->view('main/footer');
28      }
29  }
30  }
```

2. Implementasi fungsi *create_func* pada *controller* pasien

Implementasi fungsi *create_func* dapat dilihat pada tabel 5.18.

Tabel 5.18 Implementasi fungsi *create_func*

```
1   function create_func(){
2      $config = array(
3      array(
4          'field'=>'NIK','label'=>'NIK','rules'=>'required|trim'),
5      array(
6          'field'=>'nama','label'=>'Nama','rules'=>'required|trim'),
7      array(
8          'field'=>'temp_lahir','label'=>'Tempat Lahir',
9          'rules'=>'required'),
10     array(
11         'field'=>'tgl_lahir','label'=>'Tanggal Lahir',
12         'rules'=>'required'));
13     $this->form_validation->set_rules($config);
14     if($this->form_validation->run()==FALSE){
```

Tabel 5.18 Implementasi fungsi *create_func* (lanjutan)

```
15      $this->create();
16      }else{
17         $data = array(
18         'id_daftar'       => $this->input->POST('id_daftar'),
19         'nama'            => $this->input->POST('nama'),
20         'NIK'             => $this->input->POST('NIK'),
21         'jen_kelamin'     => $this->input->POST('jen_kelamin'),
22         'temp_lahir'      => $this->input->POST('temp_lahir'),
23         'tgl_lahir'       => $this->input->POST('tgl_lahir'),
24         'alamat'          => $this->input->POST('alamat'),
25         'status'          => $this->input->POST('status'),
26         'pekerjaan'       => $this->input->POST('pekerjaan'),
27         'jabatan'         => $this->input->POST('jabatan'),
28         'lama_kerja'      => $this->input->POST('lama_kerja'),
29         'agama'           => $this->input->POST('agama'),
30         'suku'            => $this->input->POST('suku'),
31         'telp'            => $this->input->POST('telp'),
32         'jenis_pasien'    => $this->input->POST('jenis_pasien'),
33         'nama_rs'         => $this->input->POST('nama_rs'));
34         $insert =  $this->curl->simple_POST($this->API.'/daftar',
35         $data);
36         if($insert){
37            $this->session->set_flashdata('Csukses','insert data
38            sukses');
39         }else{
40             $this->session->set_flashdata('Cgagal','insert data
41             gagal');
42         }
43         redirect('pasien');
44      }
45   }
```

3. Implementasi fungsi *index_post* pada *controller* Daftar

   Implementasi fungsi *index_post* dapat dilihat pada tabel 5.19.

Tabel 5.19 Implementasi fungsi *index_post*

```
1   function index_POST() {
2      $this->load->model('m_daftar');
3      $data = array(
```

Tabel 5.19 Implementasi fungsi *index_post* (lanjutan)

```
4         'id_daftar'              => $this->POST('id_daftar'),
5         'nama'                   => $this->POST('nama'),
6         'NIK'                    => $this->POST('NIK'),
7         'jen_kelamin'            => $this->POST('jen_kelamin'),
8         'temp_lahir'             => $this->POST('temp_lahir'),
9         'tgl_lahir'              => $this->POST('tgl_lahir'),
10        'alamat'                 => $this->POST('alamat'),
11        'status'                 => $this->POST('status'),
12        'pekerjaan'              => $this->POST('pekerjaan'),
13        'jabatan'                => $this->POST('jabatan'),
14        'lama_kerja'             => $this->POST('lama_kerja'),
15        'agama'                  => $this->POST('agama'),
16        'suku'                   => $this->POST('suku'),
17        'telp'                   => $this->POST('telp'),
18        'jenis_pasien'           => $this->POST('jenis_pasien'),
19        'nama_rs'                => $this->POST('nama_rs'));
20    $NIK    = $this->POST('NIK');
21    $nama  = $this->POST('nama');
22    if ($this->m_daftar->user_exists($nama, $NIK)==TRUE) {
23        $this->response(array('status' => 'Pasien sudah
24        terdaftar!'), 405);
25    } else {
26        $insert = $this->db->insert('pasien', $data);
27        if ($insert) {
28            this->response($data, 200);
29        } else {
30            $this->response(array('status' => 'fail', 502));
31        }
32    }
33 }
```

4. Implementasi fungsi *edit_func* pada *controller* pasien

   Implementasi fungsi *edit_func* dapat dilihat pada tabel 5.20.

Tabel 5.20 Implementasi fungsi *edit_func*

```
1  function edit_func(){
2        $data = array(
3        'id_daftar'       => $this->input->POST('id_daftar'),
4        'nama'            => $this->input->POST('nama'),
```

Tabel 5.20 Implementasi fungsi *edit_func* (lanjutan)

```
5         'NIK'              => $this->input->POST('NIK'),
6         'jen_kelamin'      => $this->input->POST('jen_kelamin'),
7         'temp_lahir'       => $this->input->POST('temp_lahir'),
8         'tgl_lahir'        => $this->input->POST('tgl_lahir'),
9         'alamat'           => $this->input->POST('alamat'),
10        'status'           => $this->input->POST('status'),
11        'pekerjaan'        => $this->input->POST('pekerjaan'),
12        'jabatan'          => $this->input->POST('jabatan'),
13        'lama_kerja'       => $this->input->POST('lama_kerja'),
14        'agama'            => $this->input->POST('agama'),
15        'suku'             => $this->input->POST('suku'),
16        'telp'             => $this->input->POST('telp'),
17        'jenis_pasien'     => $this->input->POST('jenis_pasien'),
18        'nama_rs'          => $this->input->POST('nama_rs'));
19        $update =  $this->curl->simple_PUT($this->API.'/daftar',
20        $data, array(CURLOPT_BUFFERSIZE => 99));
21        if($update){
22           $this->session->set_flashdata('Esukses','Update Data
23           Berhasil');
24        }else{
25           $this->session->set_flashdata('Egagal','Update Data
26           Gagal'. $this->curl->error_string);
27        }
28        redirect('pasien');
29  }
```

5. Implementasi fungsi *index_put* pada *controller* Daftar

   Implementasi fungsi *index_put* dapat dilihat pada tabel 5.21.

Tabel 5.21 Implementasi fungsi *index_put*

```
1   function index_PUT() {
2      $id = $this->PUT('id_daftar');
3      $data = array(
4         'id_daftar'             => $this->PUT('id_daftar'),
5         'nama'                  => $this->PUT('nama'),
6         'NIK'                   => $this->PUT('NIK'),
7         'jen_kelamin'           => $this->PUT('jen_kelamin'),
8         'temp_lahir'            => $this->PUT('temp_lahir'),
9         'tgl_lahir'             => $this->PUT('tgl_lahir'),
```

Tabel 5.21 Implementasi fungsi *index_put* (lanjutan)

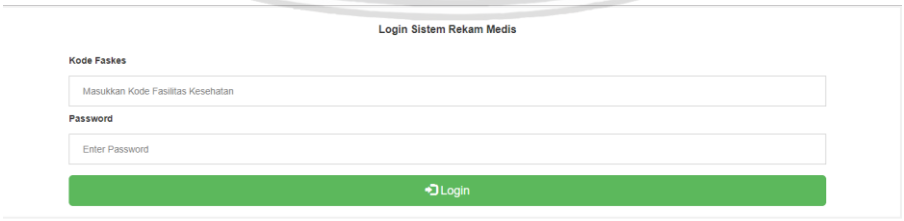
```
10          'alamat'                => $this->PUT('alamat'),
11          'status'                => $this->PUT('status'),
12          'pekerjaan'             => $this->PUT('pekerjaan'),
13          'jabatan'               => $this->PUT('jabatan'),
14          'lama_kerja'            => $this->PUT('lama_kerja'),
15          'agama'                 => $this->PUT('agama'),
16          'suku'                  => $this->PUT('suku'),
17          'telp'                  => $this->PUT('telp'),
18          'jenis_pasien'          => $this->PUT('jenis_pasien'),
19          'nama_rs'               => $this->PUT('nama_rs'));
20      $this->db->where('id_daftar', $id);
21      $update = $this->db->update('pasien', $data);
22      if ($update) {
23          $this->response($data, 200);
24      } else {
25          $this->response(array('status' => 'fail', 502));
26      }
27  }
```

5.2.6 Implementasi Antarmuka

Pada proses implementasi antarmuka mengacu pada hasil perancangan antarmuka pada tahap sebelumnya. Masing-masing implementasi dari hasil perancangan antarmuka yang telah dirancang diantaranya dapat dilihat pada bagian berikut.

1. Tampilan antarmuka *Login*

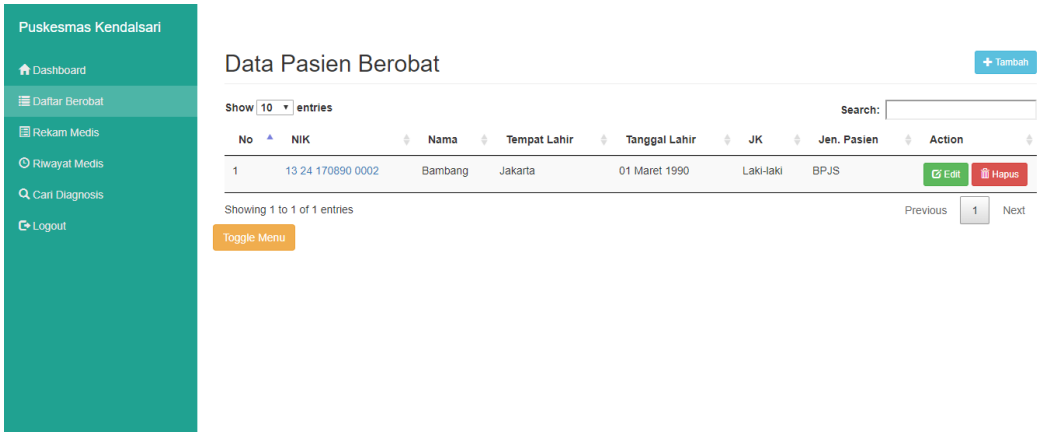
   Hasil implementasi antarmuka *login* dapat dilihat pada gambar 5.33 berikut.



Gambar 5.16 Tampilan antarmuka *Login*

2. Tampilan antarmuka Daftar Berobat

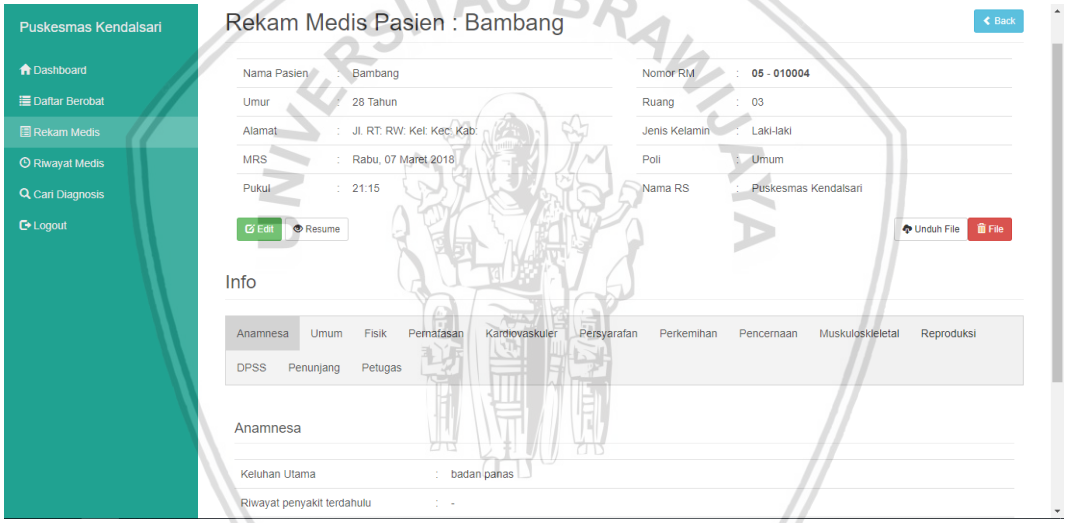
   Hasil implementasi antarmuka Daftar Berobat dapat dilihat pada gambar 5.34 berikut.



Gambar 5.17 Tampilan antarmuka Daftar Berobat

3. Tampilan antarmuka rekam medis pasien

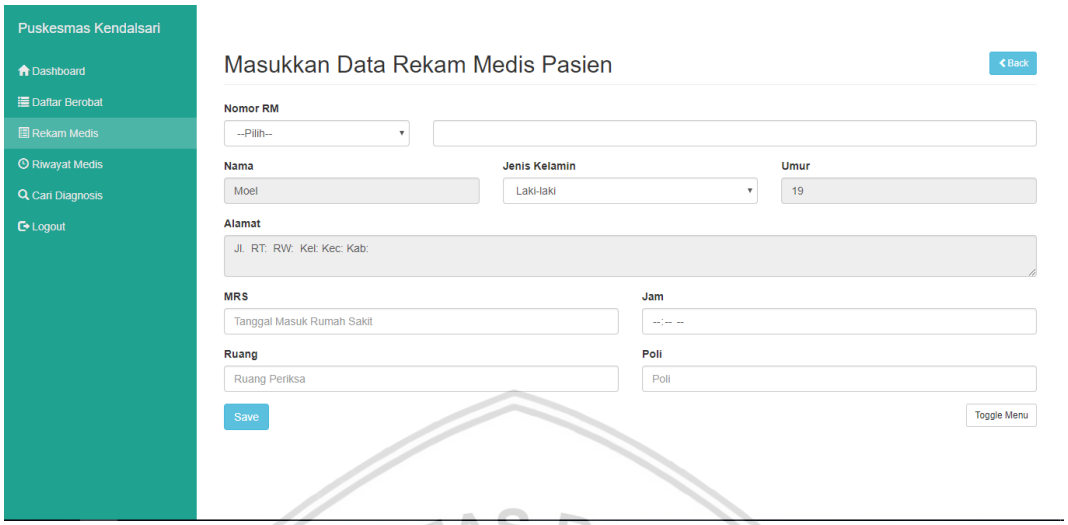
Hasil implementasi antarmuka rekam medis pasien dapat dilihat pada gambar 5.35.



Gambar 5.18 Tampilan antarmuka rekam medis pasien

4. Tampilan antarmuka tambah RM

Hasil implementasi antarmuka tambah RM dapat dilihat pada gambar 5.36.



Gambar 5.19 Tampilan antarmuka tambah RM

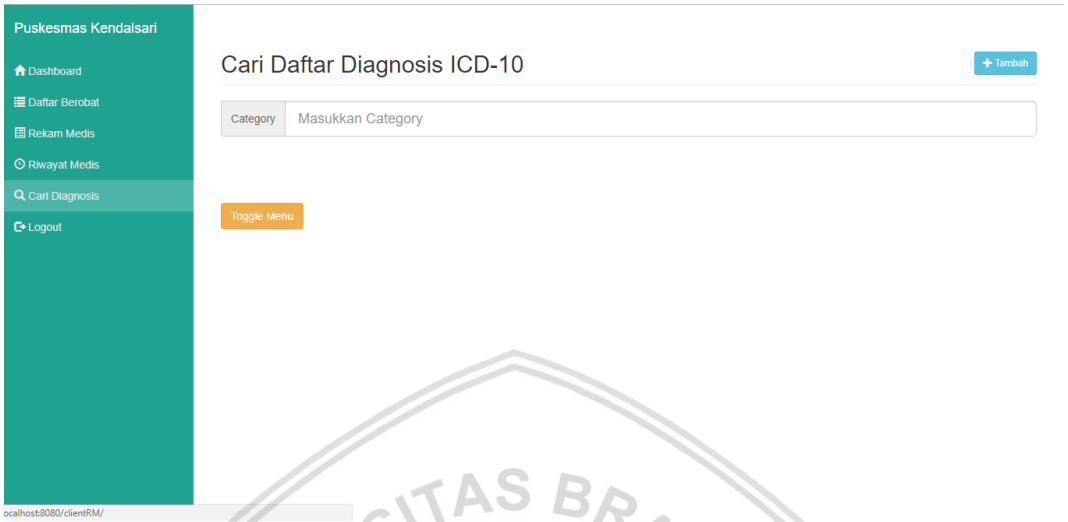
5. Tampilan antarmuka *edit* data rekam medis

Hasil implementasi antarmuka *edit* data rekam medis dapat dilihat pada gambar 5.37 berikut.

Gambar 5.20 Implementasi antarmuka *edit* data rekam medis

6. Tampilan antarmuka Cari Diagnosis

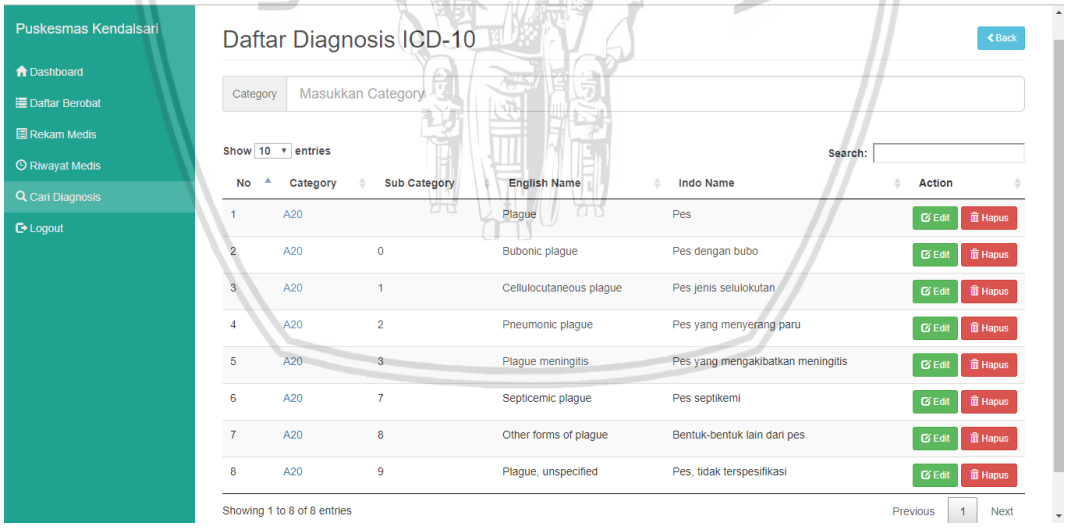
Hasil implementasi antarmuka Cari Diagnosis dapat dilihat pada gambar 5.38 berikut.



Gambar 5.21 Implementasi antarmuka Cari diagnosis

7. Tampilan antarmuka Daftar Diagnosis

Hasil implementasi antarmuka Daftar Diagnosis dapat dilihat pada gambar 5.39 berikut.



Gambar 5.22 Implementasi antamuka Daftar Diagnosis

BAB 6 PENGUJIAN DAN ANALISIS

6.1 Pengujian *White Box*

Pengujian *white box* yang dilakukan pada penelitian ini berfokus pada pengujian unit dengan tipe pengujian yang digunakan adalah *path coverage* atau biasa disebut *basis path testing*. Pengujian ini dilakukan terhadap tiga fungsi pada sistem rekam medis yaitu *create_func()* pada *class* pasien, *upload()* pada *class* rm dan *rm_pasien()* pada *class* rm. Fungsi *create_func()* merupakan *method* untuk mengirim permintaan tambah data daftar berobat dari *class* pasien pada *client* menuju *class* daftar pada *server*. Fungsi *upload()* merupakan fungsi yang digunakan untuk mengunggah berkas rekam medis berbentuk *soft file*. Fungsi *rm_pasien()* merupakan fungsi untuk menampilkan data rekam medis pasien tertentu.

6.1.1 Pengujian Fungsi *create_func*

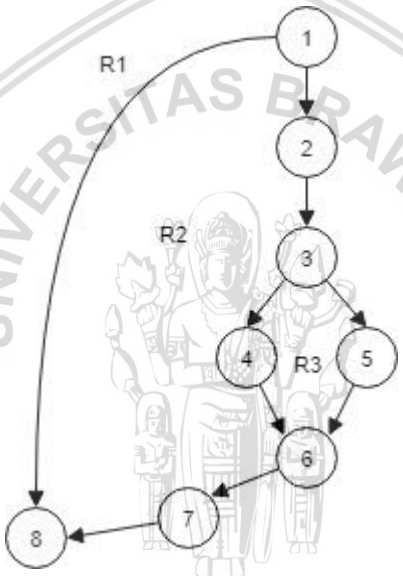
Tabel 6.1 *Pseudocode* fungsi *create_func*

```
1   config = array( 1
2           array(field = NIK, label = NIK, rules = required|trim), 1
3           array(field = nama, label = Nama,rules = required|trim), 1
4           array(field = temp_lahir, label = Tempat Lahir,rules = required), 1
5           array(field = tgl_lahir, label = Tanggal Lahir,rules = required), 1
6    )
7    Form validation set_rules(config) 1
8    if(Form validation run() = FALSE) then 1
9       create() 2
10   else 3
11      data = array( 3
12         id_daftar          =  input POST(id_daftar), 3
13         nama               = input POST (nama), 3
14         NIK                = input POST (NIK), 3
15         jen_kelamin        = input POST (jen_kelamin), 3
16         temp_lahir         = input POST (temp_lahir), 3
17         tgl_lahir          = input POST (tgl_lahir), 3
18         alamat             = input POST (alamat), 3
19         status             = input POST (status), 3
20         pekerjaan          = input POST (pekerjaan), 3
21         jabatan            = input POST (jabatan), 3
22         lama_kerja         = input POST (lama_kerja), 3
23         agama              = input POST (agama), 3
24         suku               = input POST (suku), 3
25         telp               = input POST (telp), 3
26         jenis_pasien       = input POST (jenis_pasien), 3
27         nama_rs            = input POST (nama_rs)) 3
28      insert =  curl simple POST(API./daftar, data) 3
29      if(insert) then 3
30         session set flashdata(Csukses, insert data sukses) 4
```

Tabel 6.1 *Pseudocode* fungsi *create_func* (lanjutan)

31	else 5
32	*session* set flashdata(Cgagal, *insert* data gagal) 5
33	end if 6
34	redirect(pasien) 7
35	end if 8

Pada tabel 6.1 merupakan *pseudocode* dari fungsi *create_func* untuk menjelaskan alur kode baris program sebelum dilakukan pengujian menggunakan *flowgraph*. *Flowgraph* digunakan untuk melihat alur kode program perbaris apakah sudah berjalan dengan baik atau belum. *Flowgraph* fungsi *create_func* dapat dilihat pada gambar 6.1 berikut.



Gambar 6.1 *Flowgraph create_func*

Pada gambar 6.1 merupakan *flowgraph* dari fungsi *create_func*. Pada *flowgraph* tersebut terdapat tiga region, delapan *node* dan sembilan *edge* yang digunakan untuk menghitung *cyclomatic complexity*. Untuk menghitung *cyclomatic complexity* (mencari nilai V(G)) dapat dilihat pada bagian berikut.

Perhitungan V(G):

- V(G) = Region = 3
- V(G) = *Edge* – *Node* + 2

  = 9 – 8 + 2 = 3

Jalur Independen :

R1: 1-8

R2: 1-2-3-4-6-7-8

R3: 1-2-3-5-6-7-8

Tabel 6.2 Pengujian unit *create_func*

No	Jalur	Data	Hasil yang diharapkan	Hasil yang diperoleh	Status
1	1-8	*Form validation run = False*	Sistem memanggil *method create*() dan menampilkan halaman input daftar berobat	*Method create*() berjalan dan menampilkan halaman *input* daftar berobat	*Valid*
2	1-2-3-4-6-7-8	*Form validation run != False,* *Insert*	Sistem menampilkan pesan sukses tambahkan data	Sistem menampilkan pesan sukses berhasil menambahkan data	*Valid*
3	1-2-3-5-6-7-8	*Form validation run != False,* *!insert*	Sistem menampilkan pesan gagal	Sistem menampilkan pesan gagal menambahkan data	*Valid*

Pada tabel 6.2 merupakan pengujian unit fungsi *create_func*. Pengujian dilakukan sesuai dengan jumlah jalur independen yaitu tiga kali. Dari pengujian yang telah dilakukan semua hasil yang diharapkan sesuai dengan hasil yang diperoleh.

6.1.2 Pengujian Fungsi *upload*

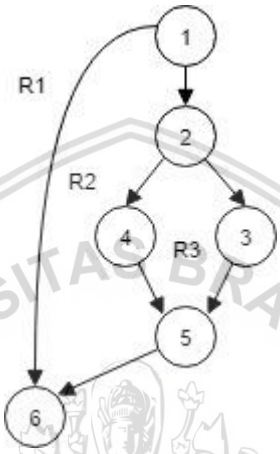
Tabel 6.3 *Pseudocode* fungsi *upload*

```
1   config[upload_path]    = ./assets/berkas;  1
2   config[allowed_types] = doc|pdf|csv|txt;  1
3   config[max_size]  = 1024 * 8;  1
4   load->library(upload,config);  1
5   if(upload->do_upload(file)) then  1
6       data = upload_data => upload->data()  2
7       id      = input(id)  2
8       judul = input(judul)  2
9       file   = data[upload_data][file_name]  2
10
11      result = m_upload->simpan_upload(id,judul,file)  2
12
13      if (result) then  2
14          print json(result)  3
```

Tabel 6.3 *Pseudocode* fungsi *upload* (lanjutan)

15	else 4
16	print upload->display_errors() 4
17	end if 5
18	end if 6

Pada tabel 6.3 merupakan *pseudocode* dari fungsi *upload* untuk menjelaskan alur kode baris program sebelum dilakukan pengujian menggunakan *flowgraph*. *Flowgraph* fungsi *upload* dapat dilihat pada gambar 6.2 berikut.



Gambar 6.2 *Flowgraph upload*

Pada gambar 6.2 merupakan *flowgraph* dari fungsi *upload*. Pada *flowgraph* tersebut terdapat tiga region, enam *node* dan tujuh *edge*. Untuk menghitung *cyclomatic complexity* (mencari nilai V(G)) dapat dilihat pada bagian berikut.

Perhitungan V(G):

- V(G) = Region = 3
- V(G) = *Edge* – *Node* + 2

  = 7 – 6 + 2 = 3

Jalur independen:

R1: 1-6

R2: 1-2-3-5-6

R3: 1-2-4-5-6

Tabel 6.4 Pengujian unit fungsi *upload*

No	Jalur	Data	Hasil yang diharapkan	Hasil yang diperoleh	Status
1	1-6	*do_upload(file)*	Parameter dapat dikirm ke model simpan_upload dan proses *upload* berhasil disimpan pada *path*	Parameter terdefinisi dan berhasil terunggah pada *path* dan berhasil dikirim ke model	*Valid*
2	1-2-3-5-6	*do_upload(file), result*	Mengembalikan data *result* dengan format *json*	Data *result* dikembalikan dengan format *json*	*Valid*
3	1-2-4-5-6	*do_upload(file), !result*	Sistem menampilkan pesan *error*	Sistem tidak melakukan aksi apapun	*Invalid*

Pada tabel 6.4 merupakan pengujian unit pada fungsi *upload*. Pengujian dilakukan sesuai dengan jumlah jalur independen yaitu tiga kali. Dari pengujian yang telah dilakukan hanya dua yang berhasil sesuai harapan dengan hasil yang diperoleh dan satu mengalami kendala.

6.1.3 Pengujian Fungsi rm_pasien

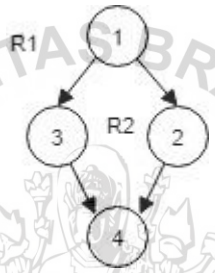
Tabel 6.5 *Pseudocode* fungsi rm_pasien

```
1  data[link_rm] = class=active  1
2  obj = array(id_daftar=> uri->segment(3)  1
3  params = array(id=> uri->segment(3)  1
4  data[pasien] = json(curl->simple_get(API./daftar,params)  1
5  data[rm]    = json(curl->simple_get(API./rm,obj) 1
6  data[anam]= json(curl->simple_get(API./anamnesa,obj)  1
7  data[pemum] = json (curl->simple_get(API./pemum,obj)  1
8  data[fisik]  = json(curl->simple_get(API./fisik,obj)  1
9  data[nafas] = json(curl->simple_get(API./pernafasan,obj)  1
10 data[kardio] = json(curl->simple_get(API./kardio,obj)  1
11 data[syaraf] = json(curl->simple_get(API./persyarafan,obj)  1
12 data['kemih'] = json(curl->simple_get(API./perkemihan,obj)  1
13 data[cerna] = json(curl->simple_get(API./pencernaan,obj)  1
14 data[musku] = json(curl->simple_get(API./muskuloskleletal,obj)  1
15 data[repro] = json(curl->simple_get(API./reproduksi,obj)  1
16 data[dpss]  = json(curl->simple_get(API./dpss,obj)  1
17 data[lab] = json(curl->simple_get(API./datalab,obj)  1
18 data[ptgs] = json(curl->simple_get(API./petugas,obj)  1
```

Tabel 6.5 *Pseudocode* fungsi rm_pasien (lanjutan)

19	data[file] = json(curl->simple_get(API./berkas,obj) 1
20	if (data[rm]=null) then 1
21	load->view(main/header,data) 2
22	load->view(rm/kosong,data) 2
23	load->view(main/footer) 2
24	else 3
25	load->view(main/header,data) 3
26	load->view(rm/detailrm,data) 3
27	load->view(main/footer) 3
28	end if 4

Pada tabel 6.5 merupakan *pseudocode* dari fungsi rm_pasien untuk menjelaskan alur kode baris program sebelum dilakukan pengujian menggunakan *flowgraph*. *Flowgraph* fungsi rm_pasien dapat dilihat pada gambar 6.3 berikut.



Gambar 6.3 *Flowgraph* rm_pasien

Pada gambar 6.3 merupakan alur *flowgraph* pada fungsi rm_pasien. Dari hasil *flowgraph* yang digambarkan diperoleh dua region, empat *node* dan empat *edge*. Untuk menghitung *cyclomatic complexity* (mencari nilai V(G)) dapat dilihat pada bagian berikut.

Perhitungan V(G):

- V(G) = Region = 2
- V(G) = *Edge* – *Node* + 2

  = 4 – 4 + 2 = 2

Jalur independen:

R1: 1-3-4

R2: 1-2-4

Tabel 6.6 Pengujian unit fungsi rm_pasien

No	Jalur	Data	Hasil yang diharapkan	Hasil yang diperoleh	Status
1	1-3-4	*data[rm]!=null*	Sistem menampilkan halaman rekam medis pasien	Sistem menampilkan halaman rekam medis pasien	*Valid*
2	1-2-4	*data[rm]=null*	Sistem menampilkan halaman rekam medis kosong	Sistem menampilkan halaman rekam medis kosong	*Valid*

Pada tabel 6.6 merupakan hasil pengujian unit untuk fungsi rm_pasien. Terdapat dua jalur independen dan hasil dari kedua jalur tersebut mendapatkan hasil sesuai dengan hasil yang diharapkan.

6.2 Pengujian *Black Box*

Pengujian *black box* digunakan untuk mengetahui program apakah berjalan sebagaimana mestinya dilihat dari sisi *input-output*. Tipe pengujian yang digunakan yaitu menggunakan pengujian validasi atau dengan kasus uji. Pada pengujian ini dilakukan untuk menguji kebutuhan fungsional sistem yang telah diimplementasikan. Pengujian validasi dapat dilihat pada tabel 6.7 berikut.

Tabel 6.7 Pengujian validasi kebutuhan fungsional

No	Nama Kebutuhan	Kasus Uji	Hasil yang diharapkan	Hasil yang diberikan	Status
1	*Login*	*User* memasukkan: Kode faskes = "13230501" dan *password* = "1"	Berhasil *login* sesuai dengan otoritas dan menampilkan *dashboard*	Berhasil *login* dengan otoritas dan menampilkan *dashboard*	*Valid*
2	*Login*	*User* memasukkan: Kode faskes = "qwe" dan *password* = "qqq"	Sistem memberikan pesan *error* kepada *user*	Sistem menampilkan pesan *error* "*Error*! *Username* dan *password* salah" dan menampilkan halaman *login*	*Valid*

Tabel 6.7 Pengujian validasi kebutuhan fungsional (lanjutan)

No	Nama Kebutuhan	Kasus Uji	Hasil yang diharapkan	Hasil yang diberikan	Status
3	Daftar Berobat	*User* memilih *menu* daftar berobat	Sistem menampilkan daftar berobat pasien	Sistem menampilkan data daftar berobat pasien berdasarkan fasilitas *user*	*Valid*
4	*Edit* Daftar	User memasukkan data sesuai dengan *form* atau data kosong	Data tetap disimpan ke dalam *database*	Data disimpan dalam *database* dan ditampilkan pada halaman daftar berobat	*Valid*
5	Tambah Daftar	*User* memasukkan data pada *form* sesuai kolom input	Berhasil menyimpan data daftar dan menampilkan pesan sukses	Data berhasil disimpan pada *database* dan menampilkan pesan sukses	*Valid*
6	Lihat RM	*User* memilih tombol lihat RM pada *pop-up* detail pasien	Sistem menampilkan data rekam medis pasien jika ada, jika tidak maka menampilkan halaman rekam medis kosong	Sistem menampilkan data rekam medis sesuai data berobat pasien	*Valid*
7	Tambah RM	*User* memasukkan data pada *form*	Berhasil menyimpan data RM dan menampilkan pesan sukses	Data berhasil disimpan pada *database* dan menampilkan pesan sukses	*Valid*
8	*Edit* RM	*User* memasukkan perubahan data sesuai dengan *form*	Sistem mengirim perubahan data ke *service* dan *service* akan merubah data kemudian mengirim respon pesan sukses	Data berhasil diubah dan menampilkan pesan sukses mengubah data	*Valid*

Tabel 6.7 Pengujian validasi kebutuhan fungsional (lanjutan)

No	Nama Kebutuhan	Kasus Uji	Hasil yang diharapkan	Hasil yang diberikan	Status
9	Tambah Anamnesa	*User* memasukkan data sesuai dengan *form* anamnesa	Berhasil menyimpan data anamnesa dan menampilkan pesan sukses	Data berhasil disimpan pada *database* dan menampilkan pesan sukses	*Valid*
10	*Edit* Anamnesa	*User* memasukkan perubahan data sesuai dengan *form* anamnesa	Sistem mengirim perubahan data ke *service* dan *service* akan merubah data kemudian mengirim respon pesan sukses	Data berhasil diubah dan menampilkan pesan sukses mengubah data	*Valid*
11	Tambah Pemum	*User* memasukkan data sesuai dengan *form* pemum	Berhasil menyimpan data pemum dan menampilkan pesan sukses	Data berhasil disimpan pada *database* dan menampilkan pesan sukses	*Valid*
12	*Edit* Pemum	*User* memasukkan perubahan data sesuai dengan *form* pemum	Sistem mengirim perubahan data ke *service* dan *service* akan merubah data kemudian mengirim respon pesan sukses	Data berhasil diubah dan menampilkan pesan sukses mengubah data	*Valid*
13	Tambah Fisik	*User* memasukkan data sesuai dengan *form* fisik	Berhasil menyimpan data fisik dan menampilkan pesan sukses	Data berhasil disimpan pada *database* dan menampilkan pesan sukses	*Valid*
14	*Edit* Fisik	*User* memasukkan perubahan data sesuai dengan *form* fisik	Sistem mengirim perubahan data ke *service* dan *service* akan merubah data kemudian mengirim respon pesan sukses	Data berhasil diubah dan menampilkan pesan sukses mengubah data	*Valid*

Tabel 6.7 Pengujian validasi kebutuhan fungsional (lanjutan)

No	Nama Kebutuhan	Kasus Uji	Hasil yang diharapkan	Hasil yang diberikan	Status
15	Tambah Kardiovaskuler	*User* memasukkan data sesuai dengan *form* kardiovaskuler	Berhasil menyimpan data kardiovaskuler dan menampilkan pesan sukses	Data berhasil disimpan pada *database* dan menampilkan pesan sukses	*Valid*
16	*Edit* Kardiovaskuler	*User* memasukkan perubahan data sesuai dengan *form* kardiovaskuler	Sistem mengirim perubahan data ke *service* dan *service* akan merubah data kemudian mengirim respon pesan sukses	Data berhasil diubah dan menampilkan pesan sukses mengubah data	*Valid*
17	Tambah Syaraf	*User* memasukkan data sesuai dengan *form* syaraf	Berhasil menyimpan data syaraf dan menampilkan pesan sukses	Data berhasil disimpan pada *database* dan menampilkan pesan sukses	*Valid*
18	*Edit* Syaraf	*User* memasukkan perubahan data sesuai dengan *form* syaraf	Sistem mengirim perubahan data ke *service* dan *service* akan merubah data kemudian mengirim respon pesan sukses	Data berhasil diubah dan menampilkan pesan sukses mengubah data	*Valid*
19	Tambah Perkemihan	*User* memasukkan data sesuai dengan *form* perkemihan	Berhasil menyimpan data perkemihan dan menampilkan pesan sukses	Data berhasil disimpan pada *database* dan menampilkan pesan sukses	*Valid*
20	*Edit* Perkemihan	*User* memasukkan perubahan data sesuai dengan *form* perkemihan	Sistem mengirim perubahan data ke *service* dan *service* akan merubah data kemudian mengirim respon pesan sukses	Data berhasil diubah dan menampilkan pesan sukses mengubah data	*Valid*

Tabel 6.7 Pengujian validasi kebutuhan fungsional (lanjutan)

No	Nama Kebutuhan	Kasus Uji	Hasil yang diharapkan	Hasil yang diberikan	Status
21	Tambah Pencernaan	*User* memasukkan data sesuai dengan *form* pencernaan	Berhasil menyimpan data pencernaan dan menampilkan pesan sukses	Data berhasil disimpan pada *database* dan menampilkan pesan sukses	*Valid*
22	*Edit* Pencernaan	*User* memasukkan perubahan data sesuai dengan *form* pencernaan	Sistem mengirim perubahan data ke *service* dan *service* akan merubah data kemudian mengirim respon pesan sukses	Data berhasil diubah dan menampilkan pesan sukses mengubah data	*Valid*
23	Tambah Muskulosklel etal	*User* memasukkan data sesuai dengan *form* muskuloskleletal	Berhasil menyimpan data muskuloskleletal dan menampilkan pesan sukses	Data berhasil disimpan pada *database* dan menampilkan pesan sukses	*Valid*
24	*Edit* Muskulosklel etal	*User* memasukkan perubahan data sesuai dengan *form* muskuloskleletal	Sistem mengirim perubahan data ke *service* dan *service* akan merubah data kemudian mengirim respon pesan sukses	Data berhasil diubah dan menampilkan pesan sukses mengubah data	*Valid*
25	Tambah Reproduksi	*User* memasukkan data sesuai dengan *form* reproduksi	Berhasil menyimpan data reproduksi dan menampilkan pesan sukses	Data berhasil disimpan pada *database* dan menampilkan pesan sukses	*Valid*
26	*Edit* Reproduksi	*User* memasukkan perubahan data sesuai dengan *form* reproduksi	Sistem mengirim perubahan data ke *service* dan *service* akan merubah data kemudian mengirim respon pesan sukses	Data berhasil diubah dan menampilkan pesan sukses mengubah data	*Valid*

Tabel 6.7 Pengujian validasi kebutuhan fungsional (lanjutan)

No	Nama Kebutuhan	Kasus Uji	Hasil yang diharapkan	Hasil yang diberikan	Status
27	Tambah Dpss	*User* memasukkan data sesuai dengan *form* dpss	Berhasil menyimpan data dpss dan menampilkan pesan sukses	Data berhasil disimpan pada *database* dan menampilkan pesan sukses	*Valid*
28	*Edit* Dpss	*User* memasukkan perubahan data sesuai dengan *form* dpss	Sistem mengirim perubahan data ke *service* dan *service* akan merubah data kemudian mengirim respon pesan sukses	Data berhasil diubah dan menampilkan pesan sukses mengubah data	*Valid*
29	Tambah Datalab	*User* memasukkan data sesuai dengan *form* datalab	Berhasil menyimpan data datalab dan menampilkan pesan sukses	Data berhasil disimpan pada *database* dan menampilkan pesan sukses	*Valid*
30	*Edit* Datalab	*User* memasukkan perubahan data sesuai dengan *form* datalab	Sistem mengirim perubahan data ke *service* dan *service* akan merubah data kemudian mengirim respon pesan sukses	Data berhasil diubah dan menampilkan pesan sukses mengubah data	*Valid*
31	Tambah Petugas	*User* memasukkan data sesuai dengan *form* petugas	Berhasil menyimpan data petugas dan menampilkan pesan sukses	Data berhasil disimpan pada *database* dan menampilkan pesan sukses	*Valid*
32	*Edit* Petugas	*User* memasukkan perubahan data sesuai dengan *form* petugas	Sistem mengirim perubahan data ke *service* dan *service* akan merubah data kemudian mengirim respon pesan sukses	Data berhasil diubah dan menampilkan pesan sukses mengubah data	*Valid*

Tabel 6.7 Pengujian validasi kebutuhan fungsional (lanjutan)

No	Nama Kebutuhan	Kasus Uji	Hasil yang diharapkan	Hasil yang diberikan	Status
33	Resume	*User* menekan tombol resume pada halaman rekam medis	Sistem akan menampilkan halaman resume sesuai data pemilik rekam medis pasien	Sistem menampilkan data resume susai pemilik rekam medis pasien	*Valid*
34	Tambah Resume	*User* memasukkan data sesuai dengan *form* resume	Berhasil menyimpan data resume dan menampilkan pesan sukses	Data berhasil disimpan pada *database* dan menampilkan pesan sukses	*Valid*
35	*Edit* Resume	*User* memasukkan perubahan data sesuai dengan *form* resume	Sistem mengirim perubahan data ke *service* dan *service* akan merubah data kemudian mengirim respon pesan sukses	Data berhasil diubah dan menampilkan pesan sukses mengubah data	*Valid*
36	Cari diagnosis	*User* memasukkan kode *category* "A20" dan menekan *enter*	Sistem menampilkan daftar diagnosis berdasarkan kode ICD yang dimasukkan	Sistem menampilkan daftar diagnosis kode A20	*Valid*
37	Tambah Diagnosis	*User* memasukkan data sesuai dengan *form* diagnosis	Berhasil menyimpan data diagnosis dan menampilkan pesan sukses	Data berhasil disimpan pada *database* dan menampilkan pesan sukses	*Valid*
38	*Edit* Diagnosis	*User* memasukkan perubahan data sesuai dengan *form* diagnosis	Sistem mengirim perubahan data ke *service* dan *service* akan merubah data kemudian mengirim respon pesan sukses	Data berhasil diubah dan menampilkan pesan sukses mengubah data	*Valid*

Tabel 6.7 Pengujian validasi kebutuhan fungsional (lanjutan)

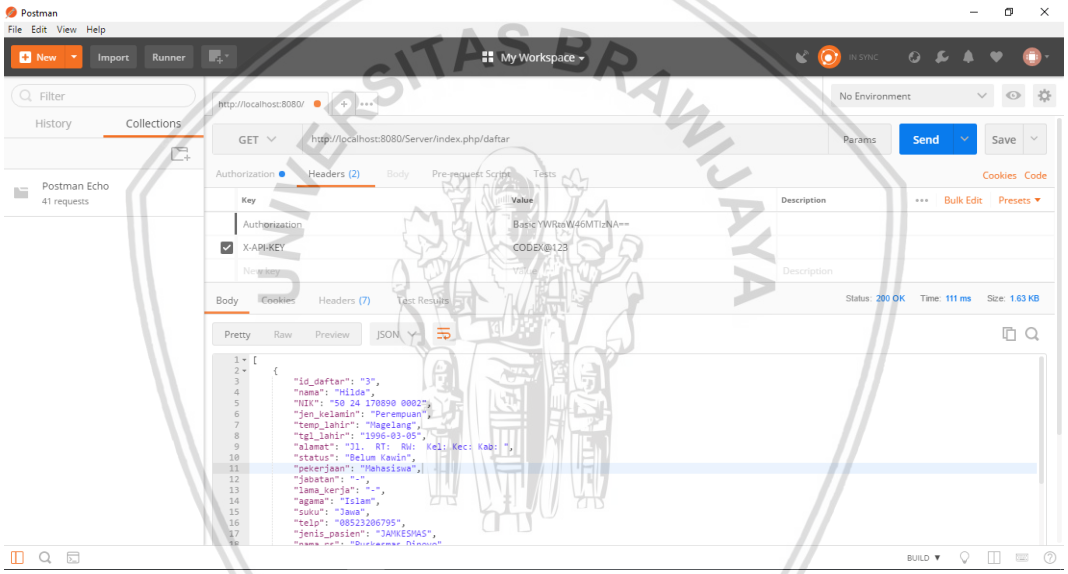
No	Nama Kebutuhan	Kasus Uji	Hasil yang diharapkan	Hasil yang diberikan	Status
39	Hapus Diagnosis	*User* menekan tombol hapus di Halaman Daftar Diagnosis	Sistem akan menampilkan peringatan dan menghapus data jika permintaan dilanjutkan	Sistem menghapus data jika permintaan dilanjutkan dan menampilkan pesan berhasil hapus data	*Valid*
40	Hapus Data	*User* menekan tombol hapus di halaman daftar berobat pasien	Sistem akan menghapus seluruh *record* data pasien dari pendaftaran, rekam medis dan resume	Sistem menghapus seluruh *record* data milik pasien yang dipilih *user*	*Valid*
41	*Upload* berkas	*User* menambahkan berkas tidak sesuai ukuran dan format yang diminta sistem	Sistem tidak melakukan aksi *upload* berkas	Sistem tidak melakukan aksi apapun	*Valid*
42	*Download* berkas	*User* memilih tombol *unduh file*	Sistem menampilkan peringatan jika *user* memilih OK maka berkas akan mulai diunduh	Sistem melakukan unduh berkas jika *user* yakin untuk mengunduh	*Valid*
43	Riwayat medis	*User* memilih menu riwayat medis	Sistem akan menampilkan daftar rekam medis pasien dari seluruh fasilitas kesehatan	Sistem menampilkan daftar rekam medis pasien dari berbagai fasilitas kesehatan yang terdapat pada *database* rekam medis terpusat	*Valid*
44	*Logout*	*User* memilih menu *logout*	Sistem menampilkan peringatan, jika *user* memilih OK maka *user* keluar dari sistem	Sistem mematikan *session* jika *user* yakin ingin keluar dari sistem	*Valid*

Pada tabel 6.7 di atas merupakan pengujian validasi menggunakan kasus uji untuk menguji sistem berjalan dengan baik menurut proses *input* dan *output*. Total pengujian yang dicantumkan pada tabel di atas didapatkan sebanyak 44 kali dan mendapatkan status *valid*.

6.3 Pengujian *REST Web Service*

Pada tahap pengujian ini dilakukan dengan menggunakan *tool* yang berperan sebagai sisi *client* untuk menguji *service API* yang telah dibuat. Seperti yang sudah dijelaskan pada bab 2 sebelumya, *tool* yang akan digunakan adalah Aplikasi Postman. Pengujian ini dilakukan untuk mengetahui apakah fungsi atau *method-method* pada *service* berjalan sesuai dengan normal. Berikut beberapa pengujian *method-method service* dan daftar status *method* yang telah dilakukan pengujian.

1. Pengujian *method GET* pada objek Daftar

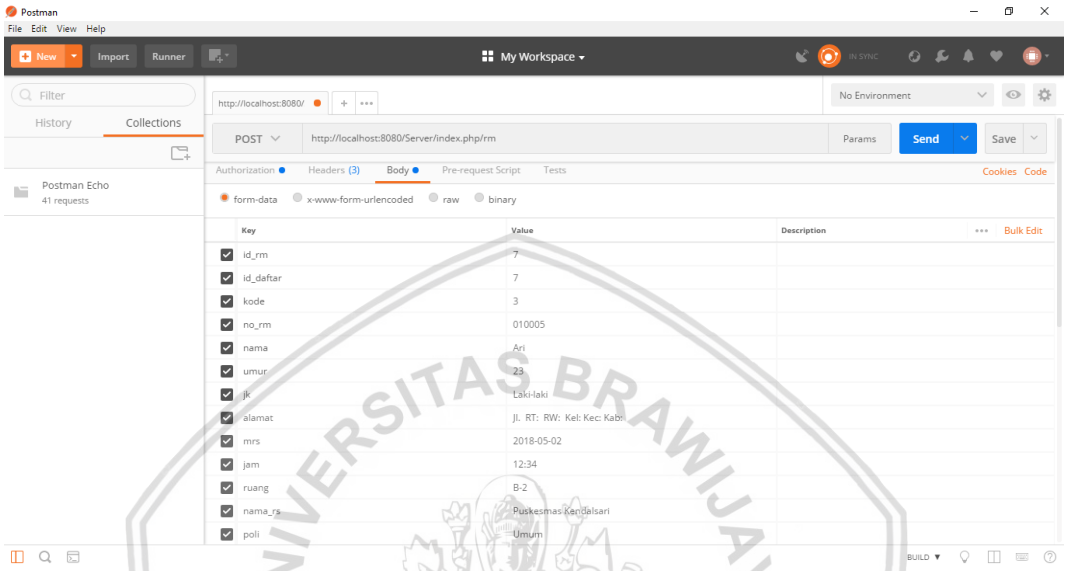


Gambar 6.4 Pengujian *method GET* objek daftar

Pada gambar 6.4 menggambarkan pengujian method *GET* pada objek Daftar. Dapat dilihat bahwa alamat *service* yang dilakukan pengujian adalah <http://localhost:8080/Server/index.php/daftar>. *Method GET* digunakan untuk memanggil dan menampilkan data dari *database* sistem. Pengujian *method* dapat dilakukan dengan mengubah tombol *GET* pada Postman menjadi *method* lain seperti *POST*, *PUT* dan *DELETE*, namun menyertakan parameter sebagai *KEY* untuk melakukan eksekusi. Pada kolom *header* dapat dilihat beberapa *key* yaitu *authorization* dan X-API-KEY. *Authorization* merupakan salah satu keamanan yang disiapkan oleh *REST service* dengan tiga tipe keamanan autentikasi diantaranya adalah *basic*, *digest* dan *session*. Sedangkan *X-API-KEY* merupakan *key name* untuk mengakses sebuah *service* . *Key* ini bersifat unik dapat didistribusikan berbeda-beda untuk tiap *client* sesuai dengan data *key* yang disediakan pada *database service.* Pada kolom *body* menampilkan data yang telah diminta pada *URL request*.

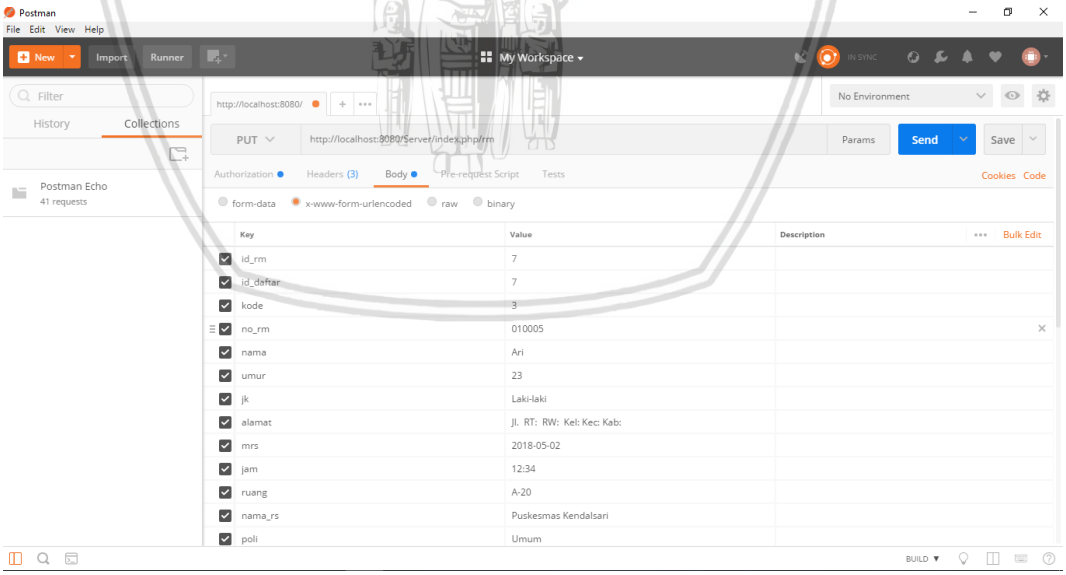
2. Pengujian *method POST* pada objek rm

Pada gambar 6.5 menjelaskan pengujian *method POST* pada objek rm. Berbeda dengan *method GET* pada pengujian sebelumnya, pada pengujian *method POST* perlu ditambahkan parameter pada *body*. *Body* disini merupakan *content* yang perlu diproses dengan menggunakan *method POST* untuk menambahkan data baru pada *database*.



Gambar 6.5 Pengujian *method POST* objek rm

3. Pengujian method *PUT* pada objek rm

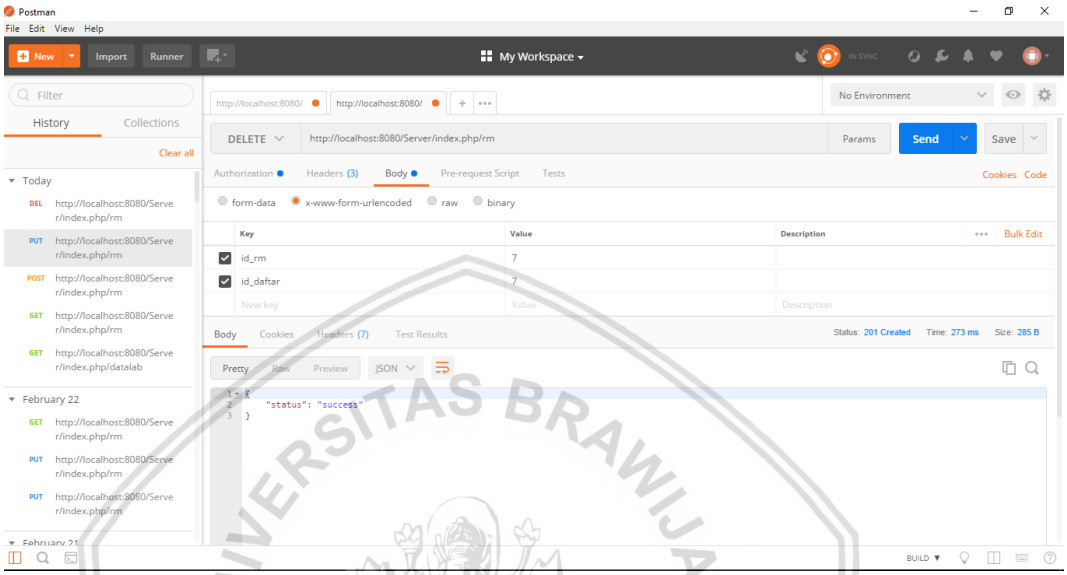


Gambar 6.6 Pengujian *method PUT* objek rm

Pada gambar 6.6 menjelaskan pengujian *service* pada *method PUT*. Sama dengan pengujian yang dilakukan pada *method POST* yaitu mendefinisikan parameter dengan *value* yang berbeda sesuai dengan data yang akan diubah pada *database*.

4. Pengujian *method DELETE* pada objek rm

Pada gambar 6.7 merupakan pengujian *service method DELETE*. Pada kolom di bawah menampilkan *response* status, *success* jika berhasil dan *fail* jika gagal.



Gambar 6.7 Pengujian *method DELETE* objek rm

Daftar lengkap pengujian fungsi-fungsi pada *service* dapat dilihat pada tabel 6.8 berikut.

Tabel 6.8 Pengujian fungsi *web service*

No	Fungsi	Status
1	*CRUD* daftar	Berhasil
2	*CRUD* rm	Berhasil
3	*CRUD* anamnesa	Berhasil
4	*CRUD* diagnosis	Berhasil
5	*CRUD* resume	Berhasil
6	*CRUD* datalab	Berhasil
7	*CRUD* dpss	Berhasil
8	*CRUD* fisik	Berhasil
9	*CRUD* kardio	Berhasil
10	*CRUD* muskuloskleletal	Berhasil
11	*CRUD* pemum	Berhasil
12	*CRUD* pencernaan	Berhasil
13	*CRUD* pencernaan	Berhasil

Tabel 6.8 Pengujian fungsi *web service* (lanjutan)

No	Fungsi	Status
14	*CRUD* perkemihan	Berhasil
15	*CRUD* pernafasan	Berhasil
16	*CRUD* persyarafan	Berhasil
17	*CRUD* petugas	Berhasil

6.4 Analisis Pengujian

Berdasarkan hasil pengujian *white box* pada pengujian unit atau *cyclomatic complexity* didapatkan delapan kali pengujian pada tiga fungsi yaitu *create_func*, *upload* dan *rm_pasien* sehingga menghasilkan tujuh keberhasilan dan satu kendala. Dari hasil pengujian pada tabel 6.2, tabel 6.4 dan tabel 6.6 diperoleh keberhasilan sebanyak tujuh dari total delapan pengujian, sehingga dapat dihitung dalam rumus persamaan 2.1.

$$Persentase\ Keberhasilan = \frac{jumlah\ pengujian\ berhasil}{jumlah\ total\ pengujian} \times 100\%$$

Maka dengan menggunakan persamaan rumus di atas untuk menghitung total presentase keberhasilan pengujian *white box* didapatkan nilai:

$$\frac{7}{8} \times 100\% = 87{,}5\%$$

Jadi, tingkat keberhasilan pengujian *white box* yang didapatkan adalah sebesar 87,5%.

Sedangkan pengujian *black box* menggunakan kasus uji pada fungsional sistem yang telah dapat dilihat pada tabel 6.7. Dari total pengujian yang dilakukan pada pengujian validasi diapatkan sebanyak 44 kasus yang diuji dan tidak terdapat kesalahan dan dinyatakan mendapatkan data *valid*. Sehingga dapat dihitung dalam persamaan berikut.

$$\frac{44}{44} \times 100\% = 100\%$$

Jadi, tingkat keberhasilan pengujian *black box* yang didapatkan adalah sebesar 100%.

Pengujian *web service* menggunakan *tool* yaitu Aplikasi Postman dengan menguji *method-method* pada tiap objek *service* dan dapat dilihat hasil dari pengujian pada tabel 6.8. Pada pengujian *service* terdapat 17 pengujian dan didapatkan 17 keberhasilan, sehingga dapat dihitung pada persamaan berikut.

$$\frac{17}{17} \times 100\% = 100\%$$

Jadi, tingkat keberhasilan pengujian *web service* menggunakan Aplikasi Postman mendapatkan hasil sebesar 100%.

Berdasarkan rata-rata persentase keberhasilan pengujian fungsional yang didapatkan adalah lebih dari 80%, jika dilakukan konversi ke dalam tabel predikat seperti pada tabel 2.11 maka hasil pengujian fungsional sistem dinyatakan ke dalam predikat sangat layak. Dikarenakan hasil implementasi *web service* pada sistem rekam medis terpusat berupa *prototype*, pengujian yang digunakan hanya untuk menguji fungsional sistem apakah fungsi pada implementasi *web service* dapat berjalan sebagaimana mestinya.

BAB 7 PENUTUP

7.1 Kesimpulan

Kesimpulan yang diperoleh dari hasil penelitian dengan judul Implementasi *Web Service* Pada Sistem Rekam Medis Terpusat adalah:

1. Dalam pembangunan sistem rekam medis menggunakan *web service*, informasi pasien berobat seperti data rekam medis pasien terolah secara terpusat, sehingga informasi bisa didapatkan melalui pusat secara langsung tanpa menunggu proses validasi dan verifikasi ketika dalam keadaan darurat.
2. Pada pembangunan sistem rekam medis terpusat ini, pusat menyediakan layanan berbentuk *website* untuk mengolah data medis pasien. Tiap fasilitas kesehatan memiliki otoritas masing-masing dalam mengolah data tiap pasien yang berobat ke fasilitas tersebut. Data yang telah diolah maka akan disimpan pada basis data pusat yang dimiliki oleh dinas kesehatan sebagai pusat yang menyediakan layanan *web service*.
3. Proses pengujian *web service* menggunakan *tool* yaitu Aplikasi Postman. Aplikasi Postman merupakan aplikasi yang digunakan sebagai *client* untuk menguji API *service* yang telah dibangun. Dari hasil pengujian yang telah dilakukan tidak ditemukan kendala pada tiap fungsi *web service* rekam medis terpusat.

7.2 Saran

Pada penelitian ini masih terdapat banyak kekurangan ini dan dapat dilakukan pada penelitian selanjutnya. Terdapat beberapa saran yang dapat digunakan dalam penelitian selanjutnya diantaranya adalah:

1. Pembangunan aplikasi *client* pada penelitian ini masih mengandalkan satu *platform* yaitu aplikasi berbasis *website*. *Web service* yang telah dibangun dapat diimplementasikan secara *multi-platform*.
2. Sistem ini masih menggunakan prosedur standar fasilitas kesehatan Puskesmas Dinoyo dan Puskesmas Kendalsari Kota Malang. Pada penelitian selanjutnya dapat ditambahakan data yang dispesifikkan pada tiap poli atau fasilitas kesehatan rumah sakit.
3. Sistem ini belum memberikan batas akses pada setiap petugas di fasilitas kesehatan, diharapkan penelitian selanjutnya dapat diberikan akses yang berbeda untuk mengidentifikasi setiap petugas pada setiap fasilitas kesehatan.

DAFTAR PUSTAKA

Adenowo, Adetokunbo A. A. dan Adenowo A. Basirat, 2013. *Software Engineering Methodologies: A Review of theWaterfall Model and Object-Oriented Approach.* International Journal of Scientific & Engineering Research, Vol. 4, Issue 7.

Adiputra, I Made Swasta., 2012. Perancangan Sistem Informasi Rekam Medis Pasien Eelektronik Terpusat (Studi Kasus : Kota Madya Denpasar). Surabaya: STMIK STIKOM.

Aljufri, A., 2013. *Aplikasi Rekam Medis Studi Kasus Klinik Universitas Widyatama,* Bandung: International Association Of Universities.

Bender, 2003. *Systems Development Life Cycle: Objectives and Requirements.* New York: Bender RBT Inc.

Budiono, 2017. *Alur Rekam Medis Pasien Puskesmas Dinoyo Malang.* Diwawancarai oleh Bahtyar. Malang, 04 Oktober 2017.

Devi, T.Rajani dan V. S. Reddy, 2012. *Work Breakdown Structure of the Project.* Warangal : International Journal of Engineering Research and Applications (IJERA), Volume 2, pp. 683-686.

Guritno, S., Sudaryono., dan Rahardja, U., 2011. *Theory and Application of IT Research*. Yogyakarta : Andi Publisher.

Kreger, H., 2001. *Web Service Conceptual Architecture (WSCA 1.0).* United Stated: IBM Software Group.

Kurniawan, E., 2014. *Implementasi Rest Web Service Untuk Sales Order Dan Sales Tracking Berbasis Mobile.* Jombang: Jurnal EKSIS, Volume 7, pp. 1-12.

Nurul, 2017. *Rekam Medis Puskesmas Kendalsari.* Diwawancarai oleh Bahtyar. Malang, 11 September 2017.

Pemkes, 2017. *Peraturan Menteri Kesehatan Republik Indonesia Nomor: 749a|MENKES/PER/XII/1989.* [Online] Tersedia di: <http://jdih.pom.go.id/showpdf.php?u=Iq1lSEIp%2F7lN36n4mRl%2BHnbuD8g%2BUv8BfJ7Oqvn7Ufs%3D> [Diakses 13 Juli 2017].

Permenkes, 2008. *Peraturan Menteri Kesehatan Republik Indonesia Nomor 269 Tahun 2008 Tentang Rekam Medis.* [Online] Tersedia di: <http://ngada.org/menkes269-2008.htm> [Diakses 08 Agustus 2017].

Postman, 2018. *Postman Makes API Development Simple*. [Online] Tersedia di : <https://www.getpostman.com/> [Diakses 3 Mei 2018].

Pressman, R. S. dan Maxim, B. R., 2015. *Software Engineering A Practitioner's Approach Eighth Edition*. New York: McGraw-Hill Education.

Putra, Rizqi A. S., 2016. *Aplikasi Sistem Pakar Untuk Membantu Guru Dalam Memilih Jenis Media Pembelajaran*. Yogyakarta: Universitas Negeri Yogyakarta.

RI, P. P., 2016. *Peraturan Pemerintah Republik Indonesia Tentang Fasilitas Pelayanan Kesehatan.* [Online] Tersedia di: < http://www.kemendagri.go.id/media/documents/2017/02/03/p/p/pp_no.47_th_2016.pdf> [Diakses 25 Oktober 2017].

Riyadi, Damar, 2013. *Rancang Bangun Rest Web Service Untuk Perbandingan Harga Pengiriman Dengan Metode Web Scrapping Dan Pemanfaatan API,* Yogyakarta: Sekolah Tinggi Manajemen Informatika Dan Komputer Amikom.

Setiana, Dian, 2016. *Pengembangan Modul Pelaporan Harga Komoditas Pertanian Pada Sisi Pengguna Admin dan Pemerintah Menggunakan Rest API*, Bogor: Institut Pertanian Bogor.

Setiawan, Edi, 2016. *Pengembangan Sistem Rekam Medis Pasien Rawat Jalan Pada Puskesmas,* Kediri: Universitas Nusantara PGRI.

Sulaen, Muhlis & Finandhita A., 2017. *Penilaian Kualitas Perangkat Lunak Pada Aplikasi Akta Notaris Fidusia di CV. Freedavelop*. Bandung: Universitas Komputer Indonesia.

Susanto, F. N. R. d. A., 2017. *Implementasi RESTful Web Service untuk Sistem Penghitungan Suara Secara Cepat pada Pilkada*. EKSPLORA INFORMATIKA, Volume 6, pp. 159-168.

Suyanto, A. H., 2007. *Web Service.* [Online] tersedia di : <http://jurnalkomputer.com/attachments/article/238/*Web*%20Service.pdf> [Diakses 06 Februari 2018].

Weningsih, A. F. D. d. I. R., 2016. *Tinjauan Proses Bisnis Unit Kerja Rekam Medis Dalam Menunjang Sistem Informasi Rumah Sakit di Rumah Sakit Gigi dan Mulut Maranatha Bandung*. Jurnal Kesehatan "Caring and Enthusiasm", Volume 5, pp. 62-80.