PENGEMBANGAN SISTEM MANAJEMEN DATA PASIEN KLINIK GIGI BERBASIS WEB
Studi Kasus Klinik Drg. Damayanti Bogor

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Deanti Siti Ajrina
NIM: 125150207111039

PROGRAM STUDI INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2016

PENGESAHAN

PENGEMBANGAN SISTEM MANAJEMEN DATA PASIEN KLINIK GIGI BERBASIS WEB
Studi Kasus Klinik Drg. Damayanti Bogor

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Deanti Siti Ajrina
NIM: 125150207111039

Skripsi ini telah diuji dan dinyatakan lulus pada
30 Juni 2016
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 2003121 001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 2003121 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, 29 Juli 2016

Deanti Siti Ajrina

NIM: 125150207111039

KATA PENGANTAR

Puji syukur kehadirat Allah SWT yang telah memberikan rahmat dan karunia-Nya sehinnga penulis dapat menyelesaikan skripsi dengan judul "Pengembangan Sistem Manajemen Data Pasien Klinik Gigi Berbasis *Web* - Studi Kasus Klinik Drg. Damayanti Bogor". Skripsi ini disusun untuk memenuhi persyaratan untuk mendapatkan gelar Sarjana Komputer pada Program Studi Teknik Informatika Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya Malang.

Melalui kesempatan ini penulis ingin menyampaikan rasa hormat dan terima kasih penulis sebesar-besarnya kepada semua pihak yang telah memberikan bantuan dan dukungan selama penulisan skripsi ini. Penulis ingin menyampaikan rasa hormat dan terima kasih kepada:

1. Kedua orang tua saya Tatang Setiawan dan Lilis Cumaryati yang selalu mendukung penulis selama ini dan selalu mendoakan penulis agar skripsi penulis dapat berjalan dengan lancar dan dimudahkan oleh Allah.
2. Bapak Tri Astoto Kurniawan, S.T, M.T, Ph.D selaku dosen pembimbing I yang telah dengan sabar membimbing dan mengarahkan penulis sehingga skripsi ini dapat selesai.
3. Segenap Bapak dan Ibu Dosen Program Studi Teknik Informatika / Ilmu Komputer yang telah membagikan ilmunya kepada penulis dan staff pegawai yang telah membantu penulis selama masa studi.
4. Kakak penulis Tiana Siti Amalia yang telah banyak membantu penulis dalam pengerjaan skripsi ini.
5. Kedua sahabat saya tercinta dari semester satu Ardana Prakasita Devi dan Silvia Ajrini yang telah meberikan semangat agar penulis dapat semangat mengerjakan skripsi ini.
6. Sahabat tercinta Witri Kharisma Wardani, Indira Rizky Utami, Destiazmi Andaliva, dan Bella Anggi Lestari yang selalu memberikan saya semangat.
7. Teman penulis Dio Saputra yang telah mengajari untuk membuat *web* dan telah banyak membantu penulis dalam pembuatan skripsi ini.
8. Teman seperjuangan Dini Khairuzadi yang telah saling memberikan semangat dan saling membantu jika ada kesusahan.
9. Teman penulis Yuni Widyaningtyas yang telah bersedia membagikan ilmunya mengenai teori RPL.
10. Teman-teman penulis Zata Ismah, Siti Azza Amira, I Wayan Vendy, Rangga Dinata, Ria Febriyani, Sandra Dwi Septika, Cika Herdanis, dan Frans Agum Gumelar yang telah menemani hari-hari penulis di Malang.

Menyadari bahwa penulisan skripsi ini masih banyak kekurangan, oleh karena itu kritik dan saran yang bersifat membangun sangat diharapkan untuk

menyempurnakan skripsi ini. Penulis berharap skripsi ini dapat bermanfaat bagi semua pihak.

Malang, 29 Juli 2015

Penulis

ABSTRAK

Deanti Siti Ajrina. 2016. Pengembangan Sistem Manajemen Data Pasien Klinik Gigi Berbasis Web - Studi Kasus Klinik Drg. Damayanti Bogor. Program Studi Informatika, Fakultas Ilmu Komputer, Universitas Brawijaya, Malang. Dosen Pembimbing : Tri Astoto Kurniawan, S.T, M.T, Ph.D

Pendaftaran konsultasi pada klinik Drg. Damayanti Bogor masih dilakukan secara manual yaitu pasien harus datang ke klinik dan melakukan pendaftaran. Kerugian yang didapat dari pendaftaran manual adalah pasien harus pergi ke klinik untuk melakukan pendaftaran. Tidak hanya itu, pencatatan rekam medis pun masih dilakukan secara manual dengan menulis pada lembar rekam medis. Kerugian yang dapat dialami adalah proses pencarian data dan rekam medis pasien membutuhkan waktu yang lama, lembar rekam medis dapat berisiko terkena air atau terbakar yang dapat menyebabkan data rekam medis pasien hilang. Untuk mengatasi permasalahan tersebut diperlukan sistem yang dapat membantu kegiatan manajemen data pasien, sehingga seluruh data pasien dapat tersimpan dengan aman, memudahkan pencatatan rekam medis, dan memudahkan pasien di dalam mengatur pendaftaran konsultasi. Sistem manajemen data pasien klinik gigi adalah sebuah aplikasi yang dapat memudahkan dokter dan perawat untuk melakukan kegiatan manajemen data pasien. Sistem ini dibuat dalam bentuk *web*, sehingga pengguna dapat menggunakan sistem ini dimana saja dan kapan saja. Sistem ini diuji menggunakan metode *white box testing* untuk pengujian unit dan integrasi dan menggunakan metode *black box testing* untuk pengujian validasi. Sistem ini telah dilakukan pengujian dengan baik pada pengujian unit, integrasi, dan validasi.

Kata Kunci: sistem manajemen klinik, pengembangan berorientasi objek, *waterfall model,* rekam medis, klinik gigi

ABSTRACT

Consultation registration in clinic Drg. Damayanti is still done manually, patients do the registration in the clinic. Disadvantages of manual registration is patients have to go to the clinic to perform registration and wait the queue in the clinic. Medical records are written on a sheet of the medical record. Disadvantages of write the medical record manually is nurse have to takes a long time to seek the medical record and the medical record can be exposed water or burned, which could make the medical record data lost. These problems can be solved by a system that can help the management activities of patient data, so all patient data is stored securely, ease the doctor to write medical records and ease the patients to manage the consultation registration. System of patient data management in dental clinic is an application that would facilitate patients to do the consultation registration, store the medical record securely, and ease the doctor and the nurse to manage the patient data. The system was created in the form of web, so users can use this system anywhere and anytime. The system was tested using white box testing for unit testing and integration testing and use black box testing for validation testing. The system has been tested well on unit testing, integration testing, and validation testing.

Keywords: *clinic management system, object oriented, waterfall model, medical record, dental clinic*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Klinik Drg. Damayanti merupakan klinik gigi yang berada di jalan Jenderal Ahmad Yani, Bogor. Klinik gigi tersebut terletak berdampingan dengan apotek Satria. Klinik ini buka dari jam tujuh malam hingga jam sepuluh malam. Terdapat pasien tetap yang rutin datang ke klinik tersebut untuk melakukan pengobatan atau hanya sekedar untuk melakukan perawatan gigi. Hal tersebut mengharuskan klinik Drg. Damayanti untuk mengelola seluruh data pasien tetap tersebut. Manajemen data pasien pada Klinik Drg. Damayanti masih dilakukan secara manual yaitu dengan pencatatan secara manual tanpa bantuan komputer. Manajemen data yang dilakukan pada Klinik Drg. Damayanti adalah pelayanan pendaftaran konsultasi, pencatatan rekam medis, dan penyimpanan identitas dan rekam medis pasien tetap.

Pendaftaran konsultasi pada klinik Drg. Damayanti dilakukan secara manual yaitu pasien harus datang ke klinik dan melakukan pendaftaran, lalu perawat akan memanggil nomor urut yang sudah dapat melakukan konsultasi. Terdapat banyak kerugian yang dapat dialami oleh pasien dari pendaftaran konsultasi yang masih dilakukan secara manual. Kerugian yang dapat dialami oleh pasien adalah pasien harus pergi ke klinik untuk melakukan pendaftaran dan menunggu antrian di klinik. Beruntung bagi pasien yang mendapat nomor antrian pertama karena tidak perlu menunggu lama untuk melakukan pemeriksaan. Namun, bagi pasien yang mendapat urutan terakhir pastilah mereka merasa bosan untuk menunggu di klinik. Ditambah lagi pasien tidak mengetahui perkiraan waktu kapan mereka akan diperiksa oleh dokter.

Tidak hanya pendaftaran konsultasi yang masih dilakukan secara manual, pencatatan rekam medis pun masih dilakukan secara manual. Pencatatan rekam medis dilakukan oleh dokter gigi secara manual pada lembar rekam medis. Pencatatan tersebut dapat merugikan dokter bahkan pasien. Salah satu kerugian yang dapat dialami adalah proses pencarian data dan rekam medis pasien yang membutuhkan waktu lama, karena harus mencari rekam medis dari banyaknya lembar rekam medis yang tersimpan pada klinik, sehingga dapat menghambat proses konsultasi atau pendaftaran. Kerugian lain yang dapat dialami adalah lembar rekam medis dapat berisiko terkena air dan terbakar. Hal tersebut dapat menyebabkan data dan rekam medis pasien hilang. Tidak hanya itu dokter juga dapat salah dalam mengambil tindakan pada gigi pasien jika rekam medis hilang, karena rekam medis berfungsi sebagai acuan untuk menentukan tindakan lebih lanjut dalam upaya pelayanan maupun tindakan medis yang diberikan kepada pasien (Depkes RI Direktorat Jenderal Pelayanan Medik, 2006). Hilangnya data rekam medis dapat sangat merugikan bagi dokter atau perawat gigi yang tidak mempunyai cadangan data.

Untuk mengatasi permasalahan manajemen data pasien tersebut diperlukan sistem manajemen data pasien yang diharapkan dapat membantu seluruh

kegiatan pencatatan rekam medis, sehingga seluruh data identitas dan rekam medis pasien dapat tersimpan dengan aman pada penyimpanan *database* dan membuat pasien tidak perlu datang ke klinik gigi untuk melakukan pendaftaran. Sistem ini akan dibuat menggunakan *platform web,* karena *web* merupakan media yang dapat diakses dimana saja dan kapan pun. Sistem ini juga akan menggunakan teknologi *SMS gateway* untuk notifikasi antrian. *SMS gateway* itu sendiri merupakan suatu *platform* yang menyediakan mekanisme untuk mengirim dan menerima pesan dari peralatan *mobile* (Nurlela, 2013). Sistem ini akan dibangun menggunakan bahasa pemrograman *PHP* dan *CodeIgniter*. Dimana *CodeIgniter* sendiri merupakan salah satu *framework* tercepat dibandingkan dengan *framework* lainnya (Daqiqil, 2011). Sistem manajemen data pasien tersebut akan dibuat dalam penelitian skripsi dengan judul "Pengembangan Sistem Manajemen Data Pasien Klinik Gigi Berbasis *Web* - Studi Kasus Klinik Drg. Damayanti Bogor".

1.2 Rumusan Masalah

Berdasarkan uraian latar belakang di atas, maka dapat dirumuskan permasalahan sebagai berikut:

1. Kebutuhan apa saja yang terdapat pada sistem manajemen data pasien klinik gigi berbasis *web*?
2. Bagaimana perancangan dan implementasi sistem manajemen data pasien klinik gigi berbasis *web* berdasarkan analisis kebutuhan yang telah dilakukan sebelumnya?
3. Bagaimana pengujian sistem manajemen data pasien klinik gigi berbasis *web* untuk memastikan seluruh kebutuhan terpenuhi dengan baik?

1.3 Tujuan

Tujuan dari penelitian ini adalah untuk membangun sistem yang dapat mengelola data pasien di klinik Drg. Damayanti.

1.4 Manfaat

Sistem manajemen data pasien klinik gigi berbasis *web* ini diharapkan dapat bermanfaat bagi pasien, dan dokter. Manfaat sistem ini akan dijelaskan sebagai berikut.

A. Pasien
    1. Memudahkan proses pendaftaran untuk menjadi pasien tetap dan pendaftaran konsultasi tanpa harus pergi ke klinik gigi.
    2. Pasien mengetahui urutan antrian melalui *SMS.*
    3. Pasien tidak perlu menunggu antrian dalam waktu yang lama di klinik.
    4. Pasien mengingat jadwal konsultasi lanjut, karena diingatkan melalui *SMS.*

B. Dokter

1. Memudahkan dokter dalam melakukan pencatatan rekam medis pasien.
2. Dokter dapat mencari data rekam medis pasien dengan mudah dan dalam waktu yang cepat.
3. Pemeriksaan gigi lancar karena pasien datang tepat waktu.
4. Memudahkan dokter dalam memeberikan resep obat kepada apoteker
5. Memudahkan dokter untuk mengingatkan pasien tentang jadwal konsultasi lanjut.

1.5 Batasan Masalah

Ruang lingkup dalam penelitian ini adalah penelitian ini difokuskan pada proses Rekayasa Perangkat Lunak (meliputi analisis kebutuhan, perancangan, implementasi, dan pengujian) terhadap sistem manajemen data pasien klinik gigi. Sedangkan batasan masalah dalam penelitian ini adalah sebagai berikut:

1. Sistem manajemen data pasien klinik gigi dibuat menggunakan *platform WEB*.
2. Sistem dibangun dengan menggunakan bahasa pemrograman *PHP, JavaScript,* dan *framework CodeIgniter*.
3. Sistem menggunakan *database MySQL.*
4. Pengembangan sistem dilakukan dari alanlisis kebutuhan, perancangan, implementasi, hingga pengujian.

1.6 Sistematika Pembahasan

Penelitian ini diuraikan dengan sistematika penulisan sebagai berikut:

BAB I Pendahuluan

Dalam bab ini dijelaskan latar belakang serta tujuan dilakukannya penelitian tentang "Pengembangan Sistem Manajemen Data Pasien Klinik Gigi Berbasis Web" termasuk juga perumusan masalah, manfaat penelitian, batasan penelitian, dan sistematika pembahasan.

BAB II Landasan Kepustakaan

Bab ini berisi tentang teori-teori pendukung dan bahan penelitian mengenai teknologi yang diimplementasikan pada penelitian ini. Teori yang diambil mengenai pengembangan perangkat lunak, *framework* yang digunakan yaitu *CodeIgniter*, dan Bahasa pemrograman yang digunakan yaitu Bahasa pemrograman PHP.

BAB III Metodologi

Dalam bab ini dijelaskan metode-metode penelitian yang digunakan seperti studi literature, analisis kebutuhan, metode perancangan, metode pengujian, dan metodi lain yang berkaitan dengan penelitian ini.

BAB IV Analisis Kebutuhan

Bab ini akan menjelaskan analisis kebutuhan yang direpresentasikan dengan menggunakan *use case* dan *use case scenario*.

BAB V Perancangan dan Implementasi

Bab ini membahas tentang perancangan perangkat lunak berdasarkan analisis kebutuhan dan akan dirancang menggunakan perancangan basis data, diagram *sequence*, dan *class* diagram. Bab ini juga membahas implementasi perangkat lunak berupa bahasa pemrogrman yang digunakan, metode yang digunakan, serta prosedur-prosedur yang dilakukan pada pembuatan perangkat lunak.

BAB VI Pengujian

Bab ini berisi penjelasan tentang strategi pengujian dengan teknik pengujian (black box dan white box) yang dilakukan.

BAB VII Penutup

Bab ini membahas tentang kesimpulan dan saran dari pelaksanaan penelitian "Pengembangan Sistem Manajemen Data Pasien Klinik Gigi Berbasis *Web* - Studi Kasus Klinik Drg. Damayanti Bogor".

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Manajemen Data Pasien

Manajemen data adalah proses pengelolaan data. Manajemen data merupakan kegiatan yang memastikan bahwa sumber data yang didapat akurat, mutakhir, aman, dan juga berguna bagi pengguna. Manajemen data dapat dilakukan pada berbagai level seperti perusahaan, organisasi, dan usaha pribadi. Kegiatan manajemen data terdiri dari: pengumpulan data, integritas dan pengujian, penyimpanan, pemeliharaan, keamanan, organisasi, dan pengambilan (Gaol, 2008). Berdasarkan pengertian manajemen data di atas dapat disimpulkan bahwa manajemen data pasien adalah kegiatan dalam mengelola data pasien.

Data pasien yang dikelola pada klinik Drg. Damayanti adalah identitas pasien, rekam medis pasien, dan resep obat. Data pasien tersebut didapat dari proses pendaftaran dan proses pencatatan rekam medis yang akan dikembangkan pada sistem manajemen data pasien klinik gigi ini. Terdapat alur atau prosedur dalam proses pendaftaran, rekam medis, dan pemberian resep obat yang akan dijelaskan pada sub bab dibawah ini.

2.1.1 Prosedur Pendaftaran

Prosedur pendaftaran konsultasi pada klinik Drg. Damayanti terbagi menjadi dua yaitu untuk pasien tetap dan untuk pasien baru. Alur pendaftaran tersebut adalah sebagai berikut:

A. Pendaftaran Pasien Tetap
   1. Pasien datang ke klinik gigi dan pergi ke meja pendaftaran.
   2. Pasien menulis namanya pada kertas yang ada di meja pendaftaran.
   3. Perawat mencari lembar rekam medis pasien.
   4. Pasien menunggu hingga nomor urutnya dipanggil oleh perawat.

B. Pendaftaran Pasien Baru
   1. Pasien datang ke klinik gigi dan pergi ke meja pendaftaran.
   2. Pasien menulis namanya pada kertas yang ada di meja pendaftaran.
   3. Perawat akan meminta identitas pasien baru.
   4. Perawat mencatat identitas pasien yang dibutuhkan pada lembar rekam medis.
   5. Pasien menunggu hingga nomor urutnya dipanggil oleh perawat.

2.1.2 Prosedur Rekam Medis

Rekam medis adalah suatu keterangan yang tertulis maupun terekam tentang identitas, *anamnesa,* pemeriksaan fisik, laboratorium, diagnosa,

tindakan medis, serta pengobatan yang diberikan kepada pasien. Rekam medis tersebut diberikan kepada pasien yang menjalani pengobatan rawat inap, rawat jalan maupun yang mendapatkan pelayanan gawat darurat. *Anamnesa* merupakan hasil dari wawancara antara pasien atau keluarga pasien dengan dokter atau tenaga kesehatan untuk memperoleh keterangan tentang keluhan dan penyakit yang diderita oleh pasien. Rekam medis juga dapat digunakan sebagai acuan untuk menentukan tindakan lebih lanjut dalam upaya pelayanan maupun tindakan medis yang diberikan kepada pasien. Sesuai dengan penjelasan pasal 46 ayat (1) UU No. 29 Tahun 2004 tentang Praktik Kedokteran disebutkan bahwa, rekam medis merupakan sebuah berkas yang berisikan catatan dan dokumen tentang identitas pasien, pemeriksaan, pengobatan, tindakan dan pelayanan lain yang telah diberikan kepada pasien (Depkes RI Direktorat Jenderal Pelayanan Medik, 2006). Proses kegiatan pencatatan data rekam medis dimulai sejak saat pasien diterima di rumah sakit, lalu dilanjutkan dengan kegiatan pencatatan data medis pasien oleh dokter atau dokter gigi atau tenaga kesehatan lain yang memberikan pelayanan kesehatan langsung kepada pasien.

Prosedur rekam medis pada klinik Drg. Damayanti adalah sebagai berikut:

1. Pasien melakukan pendaftaran konsultasi.
2. Perawat memanggil pasien untuk masuk ke ruangan dokter.
3. Pasien memberitahu keluhan kepada dokter.
4. Dokter memeriksa gigi pasien dan memberikan terapi
5. Dokter menulis rekam medis pada lembar rekam medis pasien.
6. Lembar rekam medis disimpan kembali oleh perawat.

Lembar rekam medis pada klinik Drg. Damayanti berisi identitas pasien dan data rekam medis. Identitas pasien dan data rekam medis yang harus dicatat pada lembar rekam medis adalah sebagai berikut:

1. Identitas pasien : nomor pasien, nama pasien, umur, alamat, nama orangtua, nomor telepon
2. Tanggal konsultasi
3. Elemen : ditulis dengan kode angka sesuai dengan gigi yang diperiksa. Kode angka pertama ditulis sesuai dengan bagian rahang yang diperiksa. Selanjutnya kode angka kedua ditulis sesuai dengan gigi yang diperiksa.
4. Diagnosa : prediksi penyakit yang diderita pasien berdasarkan keluhan-keluhan yang dialami pasien
5. Terapi : perawatan atau pengobatan yang diberikan oleh dokter kepada pasien

2.1.3 Prosedur Pemberian Resep Obat

Pada saat setelah dokter memberikan terapi kepada pasien, terkadang dokter memberikan resep obat jika pasien membutuhkan obat untuk masa

penyembuhannya. Prosedur pemberian resep obat pada klinik Drg. Damayanti tersebut adalah sebagai berikut:

1. Dokter menulis resep obat yang dibutuhkan oleh pasien berdasarkan hasil pemeriksaan pada kertas resep.
2. Dokter memberikan kertas resep kepada pasien.
3. Pasien memberikan kertas resep kepada apoteker.
4. Apoteker membuat obat berdasarkan resep yang diberikan.
5. Jika obat sudah jadi apoteker akan memanggil pasien untuk mengambil obat dan melakukan pembayaran.

2.2 Pengembangan Perangkat Lunak

Pengembangan perangkat lunak merupakan tahapan yang memusatkan perhatiannya pada bagaimana data dikonstruksikan, bagaimana fungsi-fungsi diimplementasikan, bagaimana prosedur diimplementasikan, bagaimana antarmuka dibuat, bagaimana perancangan akan diterjemahkan ke dalam bahasa pemrograman, dan bagaimana pengujian dilakukan (Pressman, 2001). Tahapan yang harus ada pada pengembangan perangkat lunak, yaitu analisis kebutuhan, perancangan, implementasi, dan pengujian. Analisis kebutuhan merupakan tahapan untuk mengidentifikasi kebutuhan yang harus terdapat pada sistem berdasarkan keinginan pelanggan. Perancangan merupakan tahapan dimana kebutuhan yang sudah diidentifikasi dibuat ke dalam bentuk pemodelan. Implementasi merupakan tahapan untuk menerjemahkan perancangan ke dalam bentuk kode yang dapat diproses oleh mesin. Pengujian merupakan tahapan untuk memastikan perangkat lunak berjalan dengan baik tanpa adanya kesalahan.

Proses pengembangan perangkat lunak dapat dikerjakan dengan menggunakan metode konvensional atau berorientasi objek (Pressman, 2001). Metode konvensional merupakan metode pengembangan perangkat lunak yang menjelaskan aliran data pada perancangannya. Sedangkan metode berorientasi objek merupakan metode perangkat lunak yang mendefinisikan objek-objek pada domain masalah. Metode pengembangan perangkat lunak yang akan digunakan pada sistem manajemen data pasien klinik gigi ini adalah metode berorientasi objek. Karena dengan menggunakan metode berorientasi objek pembangunan perangkat lunak dapat berlangsung dengan cepat dan juga memudahkan dalam melakukan perubahan program. Proses pembangunan perangkat lunak menggunakan metode berorientasi objek dapat berlangsung lebih cepat karena metode berorientasi objek menggunakan konsep *reusability,* yaitu penggunaan kembali komponen program.

Rekayasa perangkat lunak merupakan suatu ilmu yang berkaitan dengan semua aspek pada proses pembangunan perangkat lunak seperti alat pengembangan, metode, dan teori yang mendukung proses pembangunan perangkat lunak. (Sommerville, 2011).

2.2.1 Model Pengembangan Perangkat Lunak

Proses pengembangan perangkat lunak dapat menggunakan model-model pengembangan perangkat lunak yang sering dikenal dengan *software development life cycle (SDLC)*. Model pengembangan perangkat lunak dipilih berdasarkan sifat dari proyek atau aplikasi, metode, dan alat-alat yang akan digunakan. Terdapat beberapa model pada pengembangan perangkat lunak, diantaranya *waterfall, prototyping,* dan *RAD.* Model yang digunakan dalam pengembangan sistem manajemen data pasien klinik gigi ini adalah w*aterfall.* Karena kebutuhan pada sistem tersebut sudah didapat pada tahap awal. Dimana syarat *waterfall model* adalah semua kebutuhan harus sudah terdefinisi dengan jelas pada tahapan paling awal yaitu tahap analisis kebutuhan (Pressman, 2001).

2.2.2 *Waterfall Model*

Waterfall model merupakan salah satu model pengembangan perangkat lunak yang sekuensial. Tahapan pada model ini dimulai dari analisis kebutuhan, perancangan, implementasi, pengujian, dan pemeliharaan (Pressman, 2001). Tahapan pengembangan pada *waterfall model* dapat berjalan jika tahapan sebelumnya sudah diselesaikan. Dokumentasi pada *waterfall model* dilakukan pada setiap tahapan pengembangan.

1. Analisis Kebutuhan
   Analisis kebutuhan merupakan tahapan untuk mendefinisikan kebutuhan sistem berdasarkan kesepakatan antara pelanggan dengan *developer.* Tahapan analisis harus didokumentasikan agar pelanggan dapat melihat proses analisis dan mengevaluasi jika terdapat kebutuhan yang tidak terpenuhi (Pressman, 2001). Terdapat dua kebutuhan yang harus dimiliki oleh sistem, yaitu kebutuhan fungsional dan non-fungsional. Kebutuhan fungsional merupakan suatu layanan yang harus disediakan oleh sistem, bagaimana sistem harus memberi reaksi terhadap data yang masuk, dan bagaimana sistem harus berjalan dalam beberapa situasi. Kebutuhan non-fungsional merupakan batasan yang diberikan oleh sistem seperti *usability, security, performance,* dan *reliability* (Sommerville, 2011).
2. Perancangan
   Perancangan merupakan tahapan yang menerjemahkan kebutuhan ke dalam bentuk pemodelan yang terfokus pada struktur data, arsitektur sistem, antarmuka, dan komponen. Perancangan struktur data mengubah model domain informasi dari hasil analisis kebutuhan menjadi struktur data. Perancangan arsitektur mendifinisikan elemen-elemen utama dan relasi antar elemen-elemen utama tersebut. Perancangan antarmuka menjelaskan bagaimana perangkat lunak berkomunikasi dengan manusia sebagai pemakainya. Perancangan komponen mengubah elemen menjadi komponen yang detail. Tahapan perancangan juga didokumentasikan sama seperti tahapan analisis kebutuhan.
3. Implementasi

Implementasi merupakan tahapan menerjemahkan perancangan ke dalam bentuk kode yang dapat dibaca oleh mesin. Implementasi dilakukan dengan menggunakan bahasa pemrograman sepeti *Java, C++, PHP,* dan lain-lain. Tahapan implementasi juga didokumentasikan seperti tahapan-tahapan sebelumnya.

4. Pengujian

   Pengujian merupakan tahapan yang dilakukan setelah implementasi. Pengujian perangkat lunak dilakukan untuk mengetahui apakah terdapat kesalahan pada sistem atau tidak dan memastikan bahwa sebuah masukan akan menghasilkan keluaran yang sesuai dengan kebutuhan yang sudah terdefinisi pada tahapan analisis. Strategi pengujian yang dapat dilakukan diantaranya adalah:

   a. Pengujian Unit

      Pengujian unit merupakan pengujian yang fokus pada klas-klas atau modul-modul perangkat lunak.

   b. Pengujian Integrasi

      Pengujian integrasi merupakan pengujian yang fokus pada hubungan antar klas dan bertujuan untuk memastikan bahwa tidak terdapat kesalahan pada saat menggabungkan klas yang berhubungan.

   c. Pengujian Validasi

      Pengujian validasi merupakan pengujian yang dilakukan untuk memastikan bahwa semua kebutuhan terpenuhi pada sistem dan untuk memastikan bahwa tidak terdapat kesalahan pada saat menjalankan kebutuhan tersebut.

   Terdapat dua teknik pengujian yang dapat digunakan, yaitu *white box testing* yang digunakan untuk pengujian unit dan *black box testing* yang digunakan untuk pengujian validasi.

   a. *White Box Testing*

      White box testing merupakan jenis pengujian perangkat lunak yang lebih berkonsentrasi terhadap struktur program dan memastikan bahwa semua jalur independen telah dieksekusi setidaknya sekali. Terdapat beberapa jenis *white box testing,* diantaranya (Pressman, 2001):

      - *Basis Path Testing*

        Pengujian untuk mengetahui jumlah jalur eksekusi yang harus dilewati berdasarkan pengukuran *cyclomatic complexity*. Sebelum menghitung c*yclomatic complexity,* digambarkan terlebih dahulu alur algoritmanya. Suatu kondisi pada algoritma digambarkan oleh node dan panah atau *edge* akan digambarkan oleh garis lurus. C*yclomatic complexity* didapatkan dari *edge* dikurangi *node* lalu ditambah dengan dua. Atau juga bisa didapat dari jumlah *node* bercabang ditambah satu. *Test case* akan dibuat pada setiap jalur eksekusi.

      - *Condition Testing*

        Pengujian yang mendesain *test case* berdasarkan kondisi logis yang ada pada suatu program. Kondisi sederhana adalah variabel *boolean* atau suatu persamaan relasional. Operator relasional adalah adalah

salah satu operator dari “<”,”>”,”=”,”≠”,”<=”,>=”. Kondisi tanpa persamaan relasional merupakan persamaan *boolean*. *Condition testing* merupakan pengujian yang berfokus pada pengujian kondisi pada program.

- *Loop Testing*
  Pengujian yang berfokus pada pernyataan perulangan pada konstruksi program. Terdapat empat jenis perulangan yaitu *simple loops, nested loops, concatenated loops, unstructured loops.* Pengujian *simple loops* dapat dilakukan dengan meloncati seluruh *loop,* hanya satu yang melalui *loop,* dua yang melalui *loop,* m melewati *loop* dimana m kurang dari n, dan n-1, n, n+1 melewati *loop*. Pengujian *nested loops* dapat dimulai dari lingkaran terdalam dan melakukan *simple loop testing* untuk *loop* terdalam. Pengujian *concatenated loops* dapat dilakukan menggunakan *simple loops testing* untuk *loop* independen dan dapat menggunakan *nested loops testing* untuk *loop* yang tidak independen. Jika terdapat *unstructured loops,* sebaiknya *loop* dirancang ulang.

b. *Black Box Testing*
   Black box testing merupakan pengujian perangkat lunak yang tidak memperhatikan struktur program. Pengujian ini hanya memperhatikan spesifikasi kebutuhan dari perangkat lunak yang telah didefinisikan pada tahapan analisis kebutuhan, memeriksa apakah *input* dan *output* sesuai dengan yang diinginkan pelanggan atau tidak dan memeriksa kesalahan pada antarmuka. Terdapat beberapa jenis *black box testing,* diantaranya (Pressman, 2001):
   - *Graph-Based Testing*
     Pengujian yang memeriksa objek-objek dan hubungan antar objek-objek tersebut. Objek akan digambarkan dalam bentuk node dan hubungannya akan digambarkan oleh garis lurus atau *link weight.* Setiap objek harus berhubungan dengan objek lainnya. Pengujian dilakukan untuk memastikan bahwa tidak ada *error* pada hubungan antar objek.
   - *Scenario-Based Testing*
     Pengujian yang memusatkan pada apa yang dilakukan oleh pengguna. Pengujian ini dilakukan untuk mengungkapkan kesalahan pada saat pengguna berinteraksi dengan sistem. Pengujian ini akan menerapkan *use case scenario* untuk menentukan apa yang harus pengguna lakukan sebagai acuan untuk pengujian.
   - *Equivalence Partitioning*
     Pengujian yang membagi domain masukan ke dalam kelas-kelas data, lalu memeriksa hasil yang dikeluarkan sistem berdasarkan masukan tersebut apakah sesuai atau tidak sesuai dengan yang diharapkan.
   - *Boundary Value Analysis*

Pengujian yang berguna untuk melakukan pengujian terhadap *boundary values* atau nilai tiap sisi dari nilai batasan. *Boundary values* diusahakan mempunyai selisih sekecil mungkin dari nilai batasan.

- *Comparison Testing*
  Pengujian seluruh versi perangkat lunak yang dilakukan secara paralel untuk membandingkan hasil keluaran apakah konsisten atau tidak.

Dari berbagai teknik pengujian pada *white box testing* dan *black box testing*, tidak semua teknik pengujian dipakai. Hanya beberapa teknik yang dipakai yaitu, *basis path testing, scenario based testing*, dan *equivalence partitioning*.

5. Pemeliharaan
   Tahapan pemeliharaan merupakan tahapan yang dilakukan jika dibutuhkan perubahan pada sistem. Perubahan dapat terjadi karena terdapat kesalahan pada saat pemekaian sistem, karena diperlukan operasi baru pada sistem, atau karena pelanggan membutuhkan kebutuhan fungsional atau peningkatan kinerja pada sistem (Pressman, 2001).

2.2.3 Pendekatan *Object Oriented*

Pendekatan berorientasi objek atau yang sering dikenal dengan *object oriented (OO)* diusulkan untuk pengembangan perangkat lunak pertama kali pada tahun 1960-an. *OO* dapat digunakan pada pembangunan perangkat lunak dengan menggunakan bahasa pemrograman yang berorientasi objek seperti *Java, C++, Eiffel,* dan *Smalltalk*. Kelebihan *OO* yaitu, mudah dalam melakukan perubahan program, konsisten antara analisis dengan perancangan, dan waktu pengembangan perangkat lunak lebih cepat. Proses pembangunan perangkat lunak menggunakan *OO* dimulai dari analisis kebutuhan untuk mengidentifikasi kebutuhan-kebutuhan perangkat lunak, objek-objek, dan klas-klas yang didapat dari domain masalah. Proses selanjutnya adalah perancangan berdasarkan hasil dari analisis kebutuhan. Perancangan yang dilakukan pada *OO* adalah perancangan arsitektur, antarmuka, dan komponen. Proses selanjutynya adalah mengubah perancangan menjadi kode yang dapat dibaca oleh mesin menggunakan bahasa pemrograman yang berorientasi objek. Proses terakhir adalah pengujian yang menguji perancangan arsitektur, antarmuka, dan komponen (Pressman, 2001).

2.2.3.1 Konsep *Object Oriented*

Pendekatan berorientasi objek atau *OO* membagi domain masalah ke dalam bentuk objek nyata dimana masing-masing objek tersebut memiliki atribut yang mendeskripsikan objek tersebut dan perilaku yang spesifik. Pada pendekatan *OO* objek yang sudah terdifinisi nantinya akan menjadi klas-klas atau sub klas. Setiap klas akan memiliki sekumpulan atribut dan sekumpulan operasi yang mendeskripsikan perilaku dari klas tersebut. Terdapat tiga konsep pada pendekatan *OO,* yaitu enkapsulasi, *inheritance*, dan *polymorphism*. Enkapsulasi membungkus atribut dan operasi pada suatu klas agar tidak dapat diakses secara sembarangan. *Inheritance* merupakan suatu hubungan dimana anak klas dapat mewarisi atribut dan operasi dari induk klas. *Polymorphism* memungkinkan

setiap anak klas memiliki nama operasi yang sama dengan nama operasi yang ada pada induk klas (Pressman, 2001).

Proses pertama yang dilakukan pada pengembangan perangkat lunak menggunakan pendekatan berientasi objek adalah *object oriented analysis (OOA)*. *OOA* mengidentifikasi kebutuhan, objek-objek, atribut, dan perilaku yang mendeskripsikan objek tersebut berdasarkan domain masalah yang dideskripsikan oleh pelanggan. *OOA* menggunakan *UML* pada pemodelannya. Kebutuhan pada *OOA* dimodelkan menggunakan diagram *use case.* Lalu dari diagram *use case* tersebut dibuatlah *use case scenario* untuk menjelaskan bagaimana sistem digunakan. Proses selanjutnya pada pendekatan berorientasi objek adalah *object oriented design (OOD)*. *OOD* mentransformasikan model analisis yang telah dibuat pada *OOA* ke dalam bentuk model perancangan. *OOD* juga menggunakan *UML* untuk memodelkan perancangannya. Perancangan yang dilakukan pada OOD adalah perancangan sistem, perancangan objek, dan perancangan antarmuka. Perancangan sistem menggambarkan struktur dari klas dan relasi antar klas tersebut dengan menggunakan *class diagram.* Perancangan objek mendeskripsikan objek-objek dan interaksinya satu sama lain dengan menggunakan *state diagram, sequence diagram,* dan *activity diagram*. Proses *OO* selanjutnya adalah *object oriented programming (OOP). OOP* mengubah model perancangan ke dalam bentuk kode yang dapat dieksekusi oleh mesin menggunakan bahasa pemrograman yang mengimplementasikan *OOP*. Proses *OO* yang terakhir adalah *object oriented testing (OOT).* Pengujian yang dapat dilakukan oleh *OOT* adalah pengujian unit, pengujian integrasi, dan pengjuian validasi. (Pressman, 2001).

2.2.3.2 Pemodelan *Object Oriented*

Pemodelan *OOA* dan OOD dapat dimodelkan menggunakan diagram yang terdapat pada *UML* (Pressman, 2001). *UML* yang disebut juga bahasa pemodelan merupakan sebuah notasi yang digunakan untuk menggambarkan suatu perancangan perangkat lunak pada pengembangan perangkat lunak yang menggunakan *OO*. Terdapat tiga diagram *UML* yang digunakan dalam pengembangan sistem manajemen data pasien klinik gigi, yaitu (Fowler, 2003):

1. *Use Case Diagram*

   Use case diagram adalah diagram yang digunakan untuk menggambarkan sekumpulan *use case,* aktor yang terlibat, beserta hubungan antara aktor dan *use case* tersebut. *Use case* adalah sekumpulan skenario untuk mendeskripsikan bagaimana sistem bekerja pada sebuah situasi untuk mencapai tujuan. Skenario adalah urutan langkah-langkah yang mendeskripsikan hubungan antara pengguna dengan sistem. Berikut adalah simbol-simbol yang ada pada *use case diagram*:

Tabel 2.1 Simbol Pada Use Case Diagram

Simbol	Deskripsi
	***Use case*** adalah sekumpulan skenario untuk

	mendeskripsikan bagaimana sistem bekerja pada sebuah situasi untuk mencapai tujuan.
	Asosiasi adalah menggambarkan interaksi antara aktor dengan *use case*
<<extend>>	***Extend*** adalah relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu. Arah panah mengarah pada *use case* yang ditambahkan.
<<include>>	***Include*** adalah relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya.
	Generalisasi digunakan ketika terdapat satu *use case* yang mirip dengan *use case* lain atau aktor yang mirip dengan aktor lain. Arah panah mengarah pada *use case* atau aktor yang menjadi generalisasinya.
	Aktor adalah orang, atau sistem lain yang berinteraksi dengan sistem yang akan dibuat.

Sumber: Fowler (2003)

2. *Sequence Diagram*

Sequence diagram merupakan diagram yang menggambarkan alur kontrol diantara objek-objek yang terdapat pada *use case.* Alur kontrol tersebut mendeskripsikan operasi apa saja yang berjalan diantara objek tersebut. Berikut adalah simbol-simbol yang ada pada *sequence diagram*:

Tabel 2.2 Simbol Pada Sequence Diagram

Simbol	Deskripsi
	Aktor adalah orang atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri.
	Pesan menyatakan suatu objek memanggil operasi yang ada pada objek lain atau dirinya sendiri.
	Objek menyatakan objek yang berinteraksi.

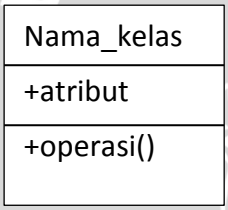
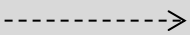
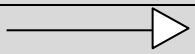
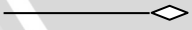
	**Return** menyatakan nilai kembalian dari sebuah operasi yang dijalankan pada suatu objek dan juga menyatakan pembentukan objek baru
	Lifeline menyatakan kehidupan suatu objek selama interaksi.

Sumber: Fowler (2003)

3. *Class Diagram*

Class diagram merupakan diagram yang digunakan untuk mendeskripsikan objek-objek dan relasi diantara objek-objek tersebut. *Class diagram* juga menunjukkan atribut-atribut dan operasi-operasi yang ada pada suatu objek. Berikut adalah simbol-simbol yang ada pada *class diagram*:

Tabel 2.3 Simbol pada Class Diagram

Simbol	Deskripsi
	**Klas** adalah objek pada struktur sistem yang memiliki atribut dan operasi. **Atribut** adalah ciri yang mendeskripsikan klas. **Operasi** adalah perilaku yang mendeskripsikan klas.
	Asosiasi adalah relasi antar klas dimana suatu klas memanggil objek dari klas yang lain
	***Dependency*** adalah relasi antar klas dimana method suatu klas membutuhkan objek klas lain untuk dijadikan parameter
	**Generalisasi** adalah garis yang menunjukkan hubungan *inheritance* antar induk klas dengan anak klas.
	**Agregasi** adalah relasi antar klas yang merupakan bagian dari klas lain, namun klas-klas tersebut tidak saling bergantung.
	Komposisi adalah relasi antar klas yang saling bergantung dimana jika klas yang dikomposisi hilang maka klas yang mengkomposisi klas tersebut juga akan hilang.

Sumber: Fowler (2003)

2.3 Teknologi Pengembangan Sistem

Terdapat beberapa teknologi yang digunakan untuk membangun sistem manajemen data pasien klinik gigi berbasis *web,* yaitu *SMS Gateway, PHP, JavaScript, CodeIgniter,* dan *MySQL* . Teknologi tersebut akan dijelaskan pada sub bab di bawah ini.

2.3.1 *SMS* dan *SMS Gateway*

Short Message Service yang merupakan kepanjangan dari *SMS* adalah fitur yang digunakan untuk berkirim pesan antar telepon genggap (*handphone)* dalam format teks. *SMS* dapat dinikmati oleh seluruh pengguna telepon genggam. Dengan adanya *SMS*, dapat dipastikan bahwa tiap pesan yang masuk pasti terbaca oleh pemilik telepon genggam tersebut. Saat ini *SMS* sudah dapat dibuat menjadi otomatis dengan menggunakan komputer. Karena komputer dapat mengirimkan pesan secara otomatis kepada nomor yang dituju. Biasanya *SMS* otomatis tersebut digunakan untuk mengirim pesan dalam jumlah yang banyak. Pengiriman pesan menggunakan komputer biasanya menggunakan teknologi *SMS Gateway.* Layanan SMS memiliki beberapa keunggulan, yaitu (Nurlela, 2013):

1. Biaya terjangkau, pengiriman terjamin sampai ke nomor tujuan dengan catatan nomor dalam keadaan aktif, selain itu waktu pengiriman juga cepat.
2. Dengan layanan ini juga pengguna dapat mengirimkan pesan kapanpun dan dimana saja.
3. Layanan SMS ini mudah digunakan untuk semua orang dari berbagai kalangan.

SMS Gateway adalah suatu *platform* yang menyediakan fasilitas untuk mengirim dan menerima pesan dari telepon, telepon seluler (*handphone*), dan lain-lain. Setiap telepon seluler (*handphone*) yang menggunakan *GSM* dapat mengirim dan menerima pesan *SMS*. Antarmuka handphone yang efektif dapat mengizinkan pengguna untuk membaca dan menulis pesan *SMS* dengan mudah dan cepat. Aplikasi *SMS* sangat terintegrasi baik dengan *device* (Nurlela, 2013). Banyak cara untuk membangun *SMS Gateway* pada sebuah *web*. Sistem manajemen data pasien klinik gigi ini akan menggunakan aplikasi *Gammu* untuk membangun *SMS Gateway*.

Gammu adalah salah satu modul *SMS Gateway* yang tidak berbayar atau gratis. *Gammu* bukan merupakan sebuah aplikasi, namun *Gammu* adalah sebuah modul yang bisa digabungkan dengan bahasa pemrograman apa saja. Kelebihan *Gammu* dibandingkan dengan *tools sms gateway* lainnya adalah (Muhadkly, 2007):

1. *Gammu* bisa dijalankan di *Windows* maupun *Linux*
2. Banyak *device* yang kompatibel oleh *gammu*
3. *Gammu* menggunakan *database MySQL*

4. Baik kabel data USB maupun SERIAL, semuanya kompatibel di *Gammu*

Berikut yang harus disiapkan untuk membuat *SMS Gateway* menggunakan *Gammu*.

1. *Gammu* untuk *Windows*
2. *Handphone* atau modem *GSM*
3. *Driver handphone* atau modem
4. *Apache* dan *MySQL* (Bisa menggunakan *Xampp*)

Gammu harus dipasang terlebih dahulu pada *windows* jika ingin menggunakannya. Proses pemasangan *Gammu* dimulai dari mengekstrak folder *Gammu*. Pada folder *Gammu* terdapat beberapa berkas antara lain: gammurc dan smsdrc. P*ort* yang ada pada berkas gammurc dan smsdrc harus diubah sesuai dengan *port* yang digunakan oleh modem atau hp. Baris *user, password, pc,* dan *database* pada berkas smsdrc harus diubah sesuai dengan *database* yang digunakan. Untuk mengubah isi dari berkas gammurc dan smsdrc dapat menggunakan aplikasi *notepad++.* Untuk memeriksa apakah modem atau hp sudah dapat digunakan oleh *Gammu* atau tidak tuliskan perintah "gammu –identify" pada *command prompt*. Untuk mengaktifkan *sms service* tuliskan perintah "gammu --smsd MYSQL smsdrc" pada *command prompt*. Jika *sms service* sudah berjalan, maka *SMS* sudah dapat dikirim ke nomor hp tujuan (Muhadkly, 2007).

2.3.2 *PHP*

Bahasa pemrograman yang digunakan untuk membangun sistem manajemen data pasien klinik gigi ini adalah bahasa pemrograman *PHP. PHP* merupakan salah satu bahasa pemrograman *website* yang sangat terkenal dan banyak sekali digunakan oleh para pembuat *website.* Dengan menggunakan PHP, *website* dapat menjadi dinamis karena *PHP* dapat berbasis *database.* Salah satu keunggulan *PHP* yaitu dapat diperoleh secara gratis. *PHP* juga terkenal aman dibandingkan dengan Bahasa pemrograman *website* yang lain (Wardana, 2010). *PHP* juga merupakan salah satu Bahasa pemrograman yang sudah mendukung *OOP (Object Oriented Programming)* sehingga dapat lebih mudah digunakan dibandingkan dengan *procedural. PHP* adalah Bahasa pemrograman *script* yang menyatu dengan *HTML* (kode dasar *website)* dan dijalankan pada sisi *server.* Sintaks *PHP* yang dibuat akan sepenuhnya dijalankan pada *server* dan hasilnya akan dikirimkan ke *browser.*

2.3.3 *JavaScript*

Selain menggunakan Bahasa pemrograman *PHP,* bahasa pemrograman yang digunakan dalam membangun sistem manajemen data pasien klinik gigi adalah *JavaScript*. *JavaScript* merupakan bahasa pemrograman yang berfungsi untuk membuat tampilan pada *web* lebih interaktif atau menarik. Sebagai contoh, *JavaScript* dapat memeriksa masukan pengguna apakah sah atau tidak sebelum masukan tersebut dikirim ke *server*, dapat menampilkan waktu lokal pengguna,

dapat melakukan perhitungan, dan lain-lain. Untuk membuat sebuah program menggunakan *JavaScript* hanya diperlukan dua alat, yaitu editor teks dan *web browser*. *JavaScript* dituliskan pada *web* dengan menggunakan tag *<SCRIPT>* sehingga tidak perlu menulisnya pada *file* terpisah. Penulisan huruf kapital dan huruf kecil pada *JavaScript* harus diperhatikan karena JavaScript membedakan huruf kapital dengan huruf kecil. Kode *JavaScript* harus diletakkan di dalam tag *<HEAD>* atau di atas tag *<FORM>* agar program *JavaScript* dimuat terlebih dahulu ke memori (Pranata, 1997).

2.3.4 *CodeIgniter*

Framework yang digunakan dalam membangun sistem manajemen data pasien klinik gigi ini adalah *framework CodeIgniter. CodeIgniter* merupakan sebuah *framework* PHP yang dapat mempercepat pengembang dalam membuat sebuah aplikasi web. Terdapat banyak *library* dan *helper* yang berguna didalamnya dan dapat mempermudah proses pengembangan. *CodeIgniter* bersifat *open source* yang dapat digunakan untuk membangun aplikasi php dinamis. *Framework* merupakan sebuah struktur konseptual dasar yang digunakan untuk memecahkan sebuah permasalahan, bahkan isu-isu kompleks yang ada. *CodeIgniter* dapat membantu *developer* untuk mengerjakan aplikasi lebih cepat daripada menulis semua code dari awal. *CodeIgniter* sendiri dibangun menggunakan konsep pengembangan pola *Model-View-Controller*. Kelebihan *CodeIgniter* yaitu sangat ringan, terstruktur, mudah dipelajari, dokumentasi lengkap dan dunkungan yang luar biasa dari forum *CodeIgniter*. Selain itu *CodeIgniter* juga memiliki fitur-fitur lainnya yang sangat bermanfaat, antara lain (Daqiqil, 2011):

1. Menggunakan pola MVC

   Dengan menggunakan pola MVC ini, struktur kode yang dihasilkan menjadi lebih terstuktur dan memiliki standar yang jelas.

2. *URL Friendly*

   URL yang dihasilkan sangat *URL friendly*. *CodeIgniter* meminimalisasi penggunaan $_GET dan digantikan dengan URL.

3. Kemudahan

   Kemudahan dalam mempelajari, membuat *library* dan *helper*, memodifikasi dan mengintegrasikan *Library* dan *helper.*

CodeIgniter juga memiliki beberpa keunggulan diantarnya adalah seperti dibawah ini.

1. *CodeIgniter* merupakan salah satu *framework PHP* tersecapat saat ini.
2. Sangat mudah untuk memodifikasi *behavior framework* ini dan juga sangat mudah untuk mengadopsi *library* lainnya.
3. *CodeIgniter* sangat mudah untuk dipelajari.

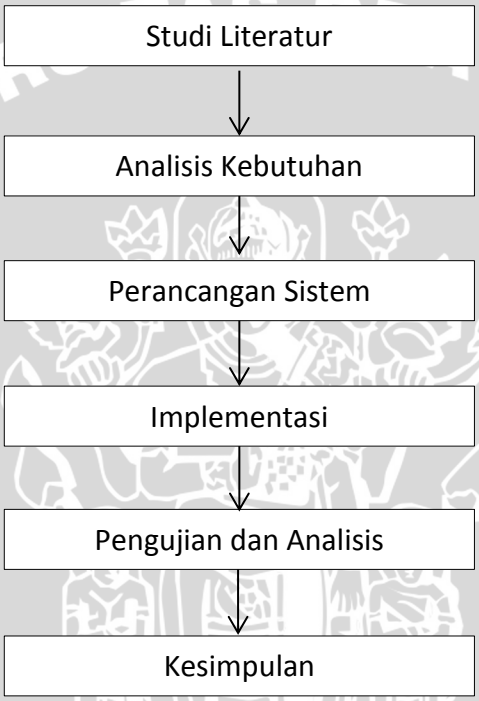
2.3.5 *MySQL*

Penyimpanan data pada sistem manajemen data pasien klinik gigi ini menggunakan *database MySQL. MySQL* merupakan sebuah perangkat lunak sistem manajemen basis data SQL atau *DBMS* yang *multithread*, *multi-user,* dengan sekitar 6 juta instalasi di seluruh dunia. *DBMS* itu sendiri merupakan suatu sistem perangkat lunak yang memungkinkan pengguna untuk membuat, memelihara, mengontrol, dan mengakses *database* secara praktis dan efisien. *MySQL* dimiliki dan disponsori oleh sbuah perusahaan komersial Swedia yaitu *MySQL AB*. *MySQL AB* memegang penuh hak cipta hampir atas semua kode sumbernya. Beberapa kelebihan *MySQL* antara lain (Solichin, 2008):

1. *Free* (bebas diunduh)
2. Stabil dan tangguh
3. Fleksibel dengan berbagai pemrograman
4. *Security* yang baik
5. Dukungan dari banyak komunitas
6. Kemudahan *management database*
7. Mendukung transaksi
8. Perkembangan *software* yang cukup pesat

BAB 3 METODOLOGI

Bab ini menjelaskan langkah-langkah yang dilakukan untuk membuat sistem manajemen data pasien klinik gigi berbasis *web*. Hal pertama yang dilakukan adalah mencari dan memilih studi literatur yang tepat, dilanjutkan dengan menganalisis kebutuhan yang diperlukan. Setelah itu melakukan perancangan sistem sesuai dengan kebutuhan yang telah ditetapkan yang nantinya akan menjadi acuan untuk proses implementasi. Setelah proses implementasi selesai, pengujian dan analisis sistem akan dilakukan untuk mendapatkan kesimpulan dari sistem yang telah dibuat. Tahapan-tahapan dalam penelitian tersebut diilustrasikan pada Gambar 3.1 di bawah ini.



Gambar 3.1 Diagram Alur Metodologi Penelitian

3.1 Studi Literatur

Studi literatur adalah metode untuk mendalami konsep yang digunakan dalam membangun sebuah sistem. Studi literatur berisi landasan teori yang terkait dengan konsep tersebut. Hal ini sangat perlu dilakukan agar pengetahuan dasar untuk membangun sebuah sistem terpenuhi dengan baik. Studi literatur yang digunakan dalam membangun sistem pendaftaran online klinik gigi ini didapat dari buku, skripsi, artikel ilmiah, dan informasi yang tersedia di internet baik berupa proyek maupun artikel. Studi literatur yang digunakan meliputi di bawah ini.

1. Manajemen Data Pasien
    a. Prosedur Pendaftaran

b. Prosedur Rekam Medis
c. Prosedur Pemberian Resep Obat

2. Pengembangan Perangkat Lunak
   a. Model Pengembangan Perangkat Lunak
   b. *Waterfall Model*
   c. Pendekatan *Object Oriented*
      - Konsep *Object Oriented*
      - Pemodelan *Object Oriented*
3. Teknologi Pengembangan Sistem
   a. *SMS* dan *SMS Gateway*
   b. *PHP*
   c. *JavaScript*
   d. *CodeIgniter*
   e. MySQL

3.2 Analisis Kebutuhan

Analisis kebutuhan merupakan metode untuk mengidentifikasi kebutuhan sistem manajemen data pasien klinik gigi ini. Proses analisis kebutuhan pada sistem ini akan dilakukan menggunakan pendekatan *OOA (Object Oriented Analysis).* Kebutuhan didapat dari hasil wawancara dengan Drg. Damayanti. Berdasarkan kebutuhan fungsional yang telah diidentifikasi, dapat dibuat pemodelan kebutuhan dengan menggunakan salah satu diagram *UML* yaitu *use case diagram.* Pemodelan kebutuhan dilakukan untuk menjelaskan kebutuhan sistem yang akan dibangun dan siapa saja yang berinteraksi dengan sistem tersebut. Setelah itu dilakukan pembuatan *use case scenario* untuk menjelaskan cara kerja sistem berdasarkan *use case diagram* yang telah dibuat. Kebutuhan non-fungsional juga diidentifikasi pada tahap ini.

3.3 Perancangan Sistem

Perancangan sistem dilakukan setelah semua kebutuhan sudah terdefinisi pada analisis kebutuhan sistem. Perancangan sistem pada penelitian ini akan dilakukan menggunakan metode *OOD* (*Object Oriented Design*) dan menggunakan *UML* (*Unified Modeling Language*). Perancangan dimulai dari perancangan *class diagram, sequence diagram, database,* dan *interface.* Hanya beberapa diagram UML yang digunakan pada perancangan sistem manajemen data pasien klinik gigi ini, yaitu:

1. *Class Diagram* merupakan diagram yang menunjukkan klas apa saja yang ada pada sistem dan bagaimana hubungannya dalam sistem.

2. *Sequence Diagram* merupakan diagram yang menjelaskan urutan proses yang terjadi untuk mencapai tujuan *use case,* interaksi antar klas, operasi apa saja yang terlibat, urutan antar operasi dan informasi yang diperlukan oleh masing-masing operasi.

3.4 Implementasi

Implementasi sistem merupakan tahapan untuk membangun sistem yang dilakukan setelah menyelesaikan tahapan perancangan sistem. Tahap ini merubah hasil perancangan sistem menjadi Bahasa pemrograman agar bisa menjadi sebuah bentuk sistem aplikasi yang dapat digunakan pengguna. Implementasi akan dilakukan dengan metode *OOP (Object Oriented Programming).* Sistem akan dibuat dalam bentuk website dengan desain antar muka yang menarik agar dapat digunakan oleh pengguna dengan mudah. Bahasa pemrograman yang akan digunakan pada tahapan implementasi ini adalah Bahasa pemrograman *PHP* dengan menggunakan *framework CodeIgniter. Software* yang digunakan untuk membantu tahapan implementasi ini adalah *Notepad++, XAMPP*, dan *MySQL* untuk pembuatan *database.*

3.5 Pengujian dan Analisis

Tahap pengujian ini merupakan tahap untuk menguji sistem apakah sistem layak dipakai oleh pengguna atau tidak dan agar sistem dapat digunakan tanpa menemui kendala-kendala apapun. Pengujian tersebut meliputi :

1. Pengujian Unit digunakan untuk menguji setiap klas yang dihasilkan dari proses perancangan. Pengujian unit dilakukan dengan menggunakan *basis path testing* yang merupakan salah satu jenis pengujian dari *white box testing.* Pengujian *basis path testing* akan menguji kode program berdasarkan algoritma yang ada pada masing-masing metode pada setiap klas. Pengujian unit akan dilakukan pada tiga sampel uji atau algoritma.

2. Pengujian Integrasi digunakan untuk menguji klas-klas yang saling berhubungan. Pengujian integrasi dilakukan dengan menggunakan *basis path testing* yang merupakan salah satu jenis pengujian dari *white box testing.* Pengujian *basis path testing* akan menguji kode program berdasarkan algoritma yang ada pada masing-masing metode pada setiap klas. Pengujian integrasi akan dilakukan pada tiga sampel uji atau algoritma.

3. Pengujian Validasi digunakan untuk menguji seluruh kebutuhan apakah sistem dapat memenuhi spesifikasi fungsional sistem yang telah ditetapkan atau tidak. Pengujian validasi dilakukan dengan cara memeriksa apakah sistem sudah berjalan dengan baik dan tidak ada *error* yang terjadi. Pengujian validasi akan dilakukan pada semua kebutuhan sistem dengan menggunakan equivalence partitioning testing dan *scenario-based testing* yang merupakan jenis pengujian dari *black box testing*.

Dengan menganalisis dari hasil pengujian tersebut juga didapatkan apa saja kekurangan dari sistem yang telah dibuat dan juga mendapatkan apa saja yang

harus diperbaiki pada sistem. Analisis sistem dilakukan untuk mengetahui apakah sistem sesuai dengan spesifikasi kebutuhan yang telah ditetapkan sebelumnya sehingga hasil pengujian akan menjadi pertimbangan sebagai kesimpulan atau sebagai evaluasi untuk membuat perbaikan pada sistem.

3.6 Kesimpulan

Tahap pengambilan kesimpulan dilakukan setelah melakukan pengujian dan analisis terhadap penelitian. Kesimpulan yang diambil berdasarkan pada hasil pengujian dan analisis. Dengan adanya kesimmpulan maka akan didapatkan inti dari hasil keseluruhan proses penelitian. Selain itu kesimpulan dapat memberi saran untuk pembangunan sistem selanjutnya.

BAB 4 ANALISIS KEBUTUHAN

Analisis kebutuhan merupakan tahapan pertama yang dilakukan pada pengembangan sebuah sistem. Analisis kebutuhan merupakan tahapan untuk menentukan kebutuhan apa saja yang harus terdapat pada sistem berdasarkan hasil wawancara dengan Drg. Damayanti. Permasalahan yang didapat adalah pendaftaran konsultasi dan pencatatan rekam medis yang masih manual. Sehingga dibutuhkan kebutuhan yang dapat menangani masalah tersebut. Analisis kebutuhan juga menjelaskan siapa saja yang dapat mengakses sistem. Kebutuhan-kebutuhan dan siapa saja yang dapat mengakses sistem akan dijelaskan pada sub bab di bawah ini.

4.1 Gambaran Umum Sistem

Berdasarkan permasalahan yang ada pada latar belakang, sistem ini harus dapat digunakan oleh pasien, dokter, perawat, apoteker, dan admin. Pasien, dokter, perawat, apoteker, dan admin dapat menggunakan sistem dengan kebutuhan yang berbeda-beda. Pasien dapat melakukan pendaftaran konsultasi maupun membatalkan konsultasi melalui sistem ini. Selain itu pasien juga dapat mengetahui urutan antrian dan jadwal konsultasi lanjut melalui *SMS*. Dokter dan Perawat dapat melihat data rekam medis pasien melalui sistem ini. Melalui sistem ini, dokter dapat mencatat rekam medis, memberikan resep obat langsung kepada apoteker, mengingatkan jadwal konsultasi lanjut kepada pasien. Perawat juga dapat menggunakan sistem ini yaitu melakukan pendaftaran konsultasi, menghapus nomor urut antiran, memajukan urutan antrian, dan mengubah urutan antrian. Apoteker dapat dengan mudah melihat resep obat yang diminta dokter, dan dapat memberitahu pasien jika obatnya sudah selesai dibuat. Admin dapat mengelola pengguna yang dapat menggunakan sistem. Pusat data seluruhnya berada pada *database*.

4.2 Identifiaksi Aktor

Berdasarkan dari permasalahan yang ada pada latar belakang, aktor pada sistem ini yaitu pasien, dokter, perawat, apoteker, dan admin. Identifikasi aktor ini bertujuan untuk mengidentifikasi aktor-aktor yang nantinya akan berinteraksi dengan sistem. Aktor-aktor tersebut didapat berdasarkan hasil wawancara dengan Drg. Damayanti. Tabel ini terdiri dari kolom aktor yang menyebutkan aktor-aktor yang berperan dalam sistem dan kolom deskripsi untuk mendeskripsikan peran dari masing-masing aktor. Proses identifikasi terlihat pada tabel 4.1 seperti dibawah ini.

Tabel 4.1 Tabel Identifikasi Aktor

Aktor	Deskripsi
Tamu	Tamu merupakan aktor yang dapat menggunakan sistem. Tamu adalah aktor yang belum melakukan *login.* Tamu hanya dapat melihat halaman awal sistem saja dan

	memilih aksi *login* atau mendaftar menjadi pasien tetap.
Pasien	Pasien memiliki hubungan *inheritance* dengan Tamu karena pada saat Pasien belum *login* Pasien hanya dapat melihat halaman awal sistem. Pasien adalah aktor yang dapat melakukan aktivitas pada sistem seperti mendaftar konsultasi atau embatalkan konsultasi.
Dokter	Dokter memiliki hubungan inheritance dengan Tamu. Dokter adalah aktor yang dapat melakukan pencatatan rekam medis, menulis resep, mengingatkan jadwal konsultasi lanjut, dan melihat rekam medis.
Perawat	Perawat memiliki hubungan inheritance dengan Tamu. Perawat memiliki aktivitas mengelola antrian
Apoteker	Apoteker memiliki hubungan *inheritance* dengan Tamu. Apoteker adalah seseorang yang membuat obat sesuai dengan resep yang diberi oleh dokter. Pada sistem ini Apoteker hanya dapat melihat resep obat yang diberikan oleh dokter dan memberitahu pasien jika obat sudah selesai dibuat.
Admin	Admin memiliki hubungan *inheritance* dengan Tamu. Admin adalah seseorang yang mengatur pengguna pada sistem seperti tambah pengguna, hapus pengguna, ubah pengguna.

4.3 Daftar Kebutuhan Fungsional

Kebutuhan fungsional merupakan suatu layanan yang harus disediakan oleh sistem, bagaimana sistem harus memberi reaksi terhadap data yang masuk, dan sistem harus berjalan dalam beberapa situasi. Spesifikasi kebutuhan fungsional ditunjukkan pada Tabel 4.2. Setiap kebutuhan akan diberikan kode MCF_XXYY. XX menunjukkan nomor kebutuhan utama dan YY menunjukkan nomor spesifikasi dari kebutuhan utama.

Tabel 4.2 Spesifikasi Kebutuhan Sistem

No.	Spesifikasi	Kode
1.	Sistem harus menyediakan fungsi *login* untuk mengizinkan pengguna mengakses sistem sesuai dengan otoritasnya dengan cara memasukkan *username* dan *password*	MCF_0100
2.	Sistem harus menyediakan menu *logout* agar pengguna yang sudah masuk ke dalam sistem dapat keluar dari sistem dengan benar	MCF_0200
3.	Sistem harus menyediakan pendaftaran untuk menjadi pasien tetap	MCF_0300

3.1.	Data pasien yang harus ditambah adalah *username* yang belum digunakan*, password,* nama lengkap, nomor *handphone,* tempat dan tanggal lahir, alamat, jenis kelamin, nama ayah, dan nama ibu	MCF_0301
3.2.	Sistem hanya menerima penulisan nama lengkap, nama ayah, dan nama ibu yang tidak disertai dengan gelar dan tidak disingkat	MCF_0302
3.3.	Sistem harus dapat memastikan bahwa pengguna tidak dapat mendaftar menjadi pasien tetap lebih dari satu kali	MCF_0303
3.4.	Pengguna yang mendaftar menjadi pasien tetap akan diberi kode verifikasi oleh sistem melalui pesan *SMS* yang harus dimasukkan ke dalam sistem	MCF_0304
3.5.	Sistem akan menghapus data pasien yang tidak memasukkan kode verifikasi dalam waktu satu jam setelah waktu pendaftaran	MCF_0305
4.	Sistem harus menyediakan fungsi untuk melakukan pendaftaran konsultasi yang akan memberi tahu nomor antriannya melalui *SMS*	MCF_0400
4.1.	Pendaftaran konsultasi dilakukan untuk mendaftar konsultasi pada hari ini saja dan hanya dapat dilakukan satu jam sebelum klinik dibuka sampai satu jam sebelum klinik ditutup	MCF_0401
4.2.	Pendaftaran konsultasi dibatasi oleh kuota pendaftaran konsultasi *online* yang ditentukan oleh perawat	MCF_0402
5.	Sistem harus menyediakan fungsi untuk membatalkan konsultasi	MCF_0500
5.1.	Pembatalan konsultasi hanya dapat dilakukan jika pasien telah melakukan pendaftaran konsultasi baik melalui *online* maupun langsung datang ke klinik	MCF_0501
5.2.	Pembatalan konsultasi hanya dapat dilakukan pada hari dimana pasien terdaftar untuk melakukan konsultasi	MCF_0502
5.3.	Sistem harus dapat mengirimkan pesan perubahan nomor antrian kepada pasien yang memiliki nomor antrian lebih besar dari pada nomor antrian yang membatalkan konsultasi	MCF_0503
6.	Sistem harus dapat menampilkan data pribadi dan rekam medis pasien yang sedang *login*	MCF_0600
6.1.	Data pribadi yang ditampilkan adalah nomor pasien, nama pasien, tempat tanggal lahir, alamat, nomor *handphone,* jenis kelamin, nama ayah, dan nama ibu. Data rekam medis yang ditampilkan adalah tanggal rekam medis dibuat, elemen, diagnosa, dan terapi	MCF_0601
7.	Sistem harus menyediakan fungsi untuk mengubah data pribadi pasien yang sedang *login*	MCF_0700

7.1.	Data pribadi yang dapat diubah adalah alamat dan nomor *handphone*	MCF_0701
8.	Sistem harus menyediakan fungsi untuk memasukkan data rekam medis pasien yang sedang konsultasi dengan cara memasukkan elemen, diagnosa, dan terapi	MCF_0800
9.	Sistem harus meyediakan fungsi untuk menampilkan data pribadi dan rekam medis seluruh paisen tetap	MCF_0900
9.1.	Data pribadi yang ditampilkan adalah nomor pasien, nama pasien, tempat dan tanggal lahir, alamat, nomor *handphone,* jenis kelamin, nama ayah, dan nama ibu. Sedangkan data rekam medis yang ditampilkan adalah tanggal rekam medis dibuat, elemen, diagnosa, dan terapi	MCF_0901
10.	Sistem harus menyediakan fungsi untuk memasukkan jadwal konsultasi lanjut pasien yang sedang konsultasi beserta keterangannya	MCF_1000
11.	Sistem harus menyediakan fungsi untuk memasukkan resep obat pasien yang sedang konsultasi	MCF_1100
12.	Sistem harus menyediakan fungsi untuk dapat menambahkan pasien ke dalam antrian konsultasi yang akan memberitahukan nomor antriannya kepada pasien tersebut melalui *SMS*	MCF_1200
12.1.	Dilakukan untuk memasukkan pasien ke dalam antrian konsultasi pada hari ini saja dan tanpa dibatasi oleh waktu dan kuota	MCF_1201
13.	Sistem harus menyediakan fungsi untuk menghapus pasien dari antrian konsultasi yang terdaftar pada hari ini	MCF_1300
13.1.	Sistem harus dapat memberitahu perubahan nomor antrian kepada pasien yang memiliki nomor antrian lebih besar dari pada nomor antrian pasien yang dihapus melalui *SMS*	MCF_1301
14.	Sistem harus menyediakan fungsi untuk mengubah urutan antrian konsultasi yang terdaftar pada hari ini	MCF_1400
14.1.	Sistem harus dapat memberitahu perubahan nomor antrian kepada pasien yang mengalami perubahan nomor urut antrian melalui *SMS*	MCF_1401
15.	Sistem harus menyediakan fungsi untuk dapat memajukan pasien yang akan konsultasi yang terdaftar pada hari ini	MCF_1500
15.1.	Sistem harus dapat memberitahu nomor antrian yang sedang konsultasi melalui *SMS* kepada pasien yang memiliki nomor antrian lebih besar dari nomor antrian pasien yang sedang konsultasi	MCF_1501
16.	Sistem harus menyediakan fungsi untuk menentukan kuota pendaftaran konsultasi *online*	MCF_1600

17.	Sistem harus menyediakan fungsi untuk melihat data resep obat yang diminta seperti nomor pasien, nama pasien yang meminta obat, dan resep obat yang diminta	MCF_1700
18.	Sistem harus menyediakan fungsi untuk memberitahu kepada pasien bahwa resep sudah selesai dibuat melalui *SMS*	MCF_1800
19.	Sistem harus menyediakan fungsi untuk menambah pengguna yang dapat mengakses sistem	MCF_1900
19.1.	Data yang harus ditambahkan adalah nama, *username* yang belum digunakan, *password,* status, alamat, dan nomor *handphone*	MCF_1901
20.	Sistem harus menyediakan fungsi untuk menghapus pengguna yang dapat mengakses sistem	MCF_2000
21.	Sistem harus menyediakan fungsi untuk mengubah data pengguna yang mengakses sistem	MCF_2100
21.1.	Data yang dapat diubah adalah nama, *username*, *password,* status, alamat, dan nomor *handphone*	MCF_2101
22.	Sistem harus menyediakan fungsi untuk mengirimkan *username* dan *password* pasien bagi pasien yang lupa *username* dan *password* miliknya	MCF_2200
22.1.	Data yang dimasukkan untuk mendapatkan *username* dan *password* adalah nama lengkap, nama ayah, nama ibu, dan nomor *handphone*	MCF_2201
22.2.	Sistem harus dapat mengirimkan *username* dan *password* kepada pengguna melalui *SMS* ke nomor *handphone* yang sudah dimasukan pengguna	MCF_2202
23.	Sistem harus menyediakan fungsi untuk membuat cadangan seluruh data pribadi dan rekam medis pasien dalam format .sql	MCF_2300

Tabel 4.3 di bawah ini menjelaskan pemetaan kode kebutuhan menjadi nama *use case.*

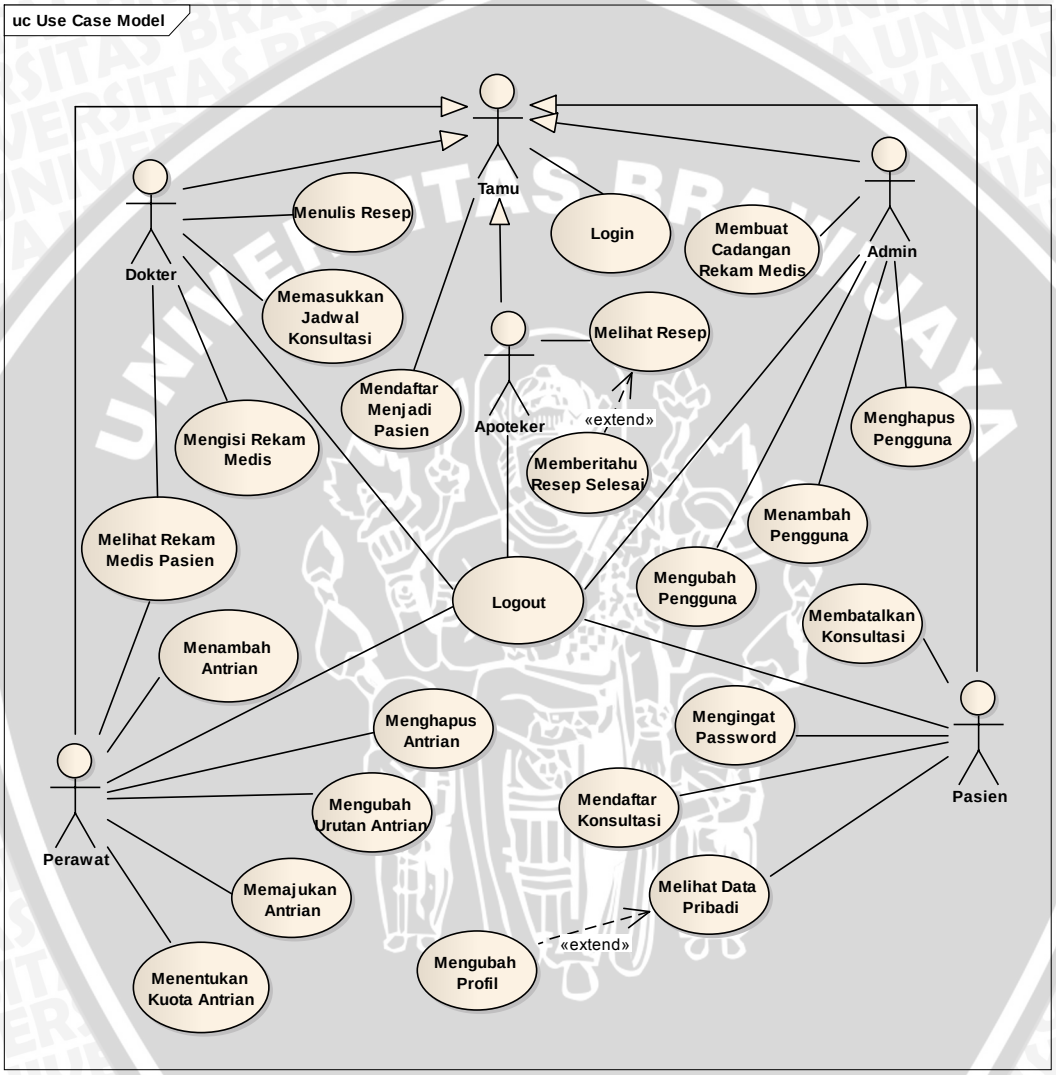
Tabel 4.3 Pemetaan Nama Use Case

Kode	Use Case
MCF_0100	Login
MCF_0200	Logout
MCF_0300	Mendaftar Menjadi Pasien
MCF_0301	
MCF_0302	
MCF_0303	
MCF_0304	
MCF_0305	

MCF_0400	Mendaftar Konsultasi
MCF_0401	
MCF_0402	
MCF_0500	Membatalkan Konsultasi
MCF_0501	
MCF_0502	
MCF_0503	
MCF_0600	Melihat Data Pribadi
MCF_0601	
MCF_0700	Mengubah Profil
MCF_0701	
MCF_0800	Mengisi Rekam Medis
MCF_0900	Melihat Rekam Medis Pasien
MCF_0901	
MCF_0902	
MCF_1000	Memasukkan Jadwal Konsultasi
MCF_1100	Menulis Resep
MCF_1200	Menambah Antrian
MCF_1201	
MCF_1300	Menghapus Antrian
MCF_1301	
MCF_1400	Mengubah Urutan Antrian
MCF_1401	
MCF_1500	Memajukan Urutan Antrian
MCF_1501	
MCF_1600	Menentukan Kuota Antrian
MCF_1700	Melihat Resep
MCF_1800	Memberitahu Resep Selesai
MCF_1900	Menambah Pengguna
MCF_1901	
MCF_2000	Menghapus Pengguna
MCF_2100	Mengubah Pengguna
MCF_2101	
MCF_2200	Mengingat Password
MCF_2201	
MCF_2202	
MCF_2300	Membuat Cadangan Rekam Medis

4.4 *Use Case Diagram*

Diagram *Use Case* adalah diagram yang digunakan untuk memodelkan perilaku sistem. Diagram *Use Case* menggambarkan seluruh kebutuhan sistem yang akan dibuat sehingga siapa pun yang melihat diagram *use case* tersebut akan memahami sistem yang akan dibangun. Diagram *Use Case* dari sistem ini dibuat berdasarkan daftar kebutuhan fungsional yang telah dibuat dan akan digambarkan pada Gambar 4.1 dibawah ini.



Gambar 4.1 Use Case Sistem

4.5 *Use Case Scenario*

U*se case scenario* adalah urutan alur dari *use case* yang telah didefiniskan pada diagram *use case.* Skenario *use case* akan dibuat menggunakan tabel yang terdiri dari *Actor*, *Objective, Pre-Condition, Main Flow, Alternative Flow, Post-Condition.* Skenario *use case* dari sistem manajemen data pasien klinik gigi ini akan dijelaskan dibawah ini.

4.5.1 Login

Tabel 4.4 Skenario Use Case Login

Login	
Actor	Tamu
Objective	Mengizinkan tamu mengakses sistem sebagai pasien, dokter, perawat, atau apoteker.
Pre-Condition	Sudah membuka formulir login pada halaman utama.
Main Flow	1. Tamu memasukkan *username* dan *password, lalu* memilih tombol *Login* 2. Sistem menampilkan halaman awal sesuai dengan status akun tamu yaitu sebagai pasien, dokter, perawat, atau apoteker.
Alternative Flow	1. Jika pada saat tamu menekan tombol *Login* namun *username* atau *password* kosong, maka sistem akan menampilkan tulisan "Kotak ini tidak boleh kosong" pada kotak yang kosong. 2. Jika *username* dan *password* yang dimasukkan salah maka sistem akan menampilkan pesan kesalahan "Username dan password salah!". 3. Jika *username* yang dimasukkan salah dan *password* yang dimasukkan benar maka sistem akan menampilkan pesan kesalahan *"*Username salah!" 4. Jika *password* yang dimasukkan salah dan *username* yang dimasukkan benar maka sistem akan menampilkan pesan kesalahan "Pasword salah!"
Post-Condition	Tamu dapat mengakses sistem sebagai pasien, dokter, perawat, atau apoteker.

4.6.2 Logout

Tabel 4.5 Skenario Use Case Logout

Logout	
Actor	Pasien, Dokter, Perawat, dan Apoteker
Objective	Mengizinkan pasien, dokter, perawat, dan apoteker untuk keluar dari sistem
Pre-Condition	Pasien, dokter, perawat, atau apoteker masih dalam keadaan *login.*
Main Flow	1. Pasien, dokter, perawat, dan apoteker memilih menu Logout 2. Sistem akan menghapus *session login* yang sedang berjalan, lalu menampilkan halaman utama
Alternative Flow	-
Post-Condition	Pasien, dokter, perawat, dan apoteker berhasil keluar

	dari sistem

4.6.3 Mendaftar Menjadi Pasien

Tabel 4.6 Skenario Use Case Mendaftar Menjadi Pasien

Mendaftar Menjadi Pasien	
Actor	Tamu
Objective	Mengizinkan tamu untuk menjadi pasien tetap di klinik.
Pre-Condition	Sudah membuka halaman Pendaftaran Pasien Tetap.
Main Flow	1. Tamu memasukkan data yang diminta seperti *username* yang belum digunakan oleh pasien lain, *password,* nama, nomor *handphone,* tempat dan tanggal lahir, alamat, jenis kelamin, nama ayah, dan nama ibu. Lalu menekan tombol Daftar 2. Sistem akan menyimpan data pasien pada *database* dan sistem akan mengirim kode verifikasi kepada tamu melalui *SMS* 3. Sistem akan menampilkan pesan sukses “Anda telah berhasi melakukan pendaftaran. Silahkan masukkan kode verifikasi Anda. “ pada halaman untuk mengisi kode verifikasi 4. Tamu memasukkan kode verifikasi yang telah dikirim oleh sistem, lalu memilih tombol Verifikasi 5. Sistem akan mengubah status pasien menjadi “terverifikasi” pada *database,* lalu sistem akan menampilkan halaman awal pasien
Alternative Flow	1. Jika tamu memasukkan *username* yang telah digunakan oleh pasien lain, maka sistem akan langsung menampilkan tulisan “Username telah digunakan”. 2. Jika pada saat tamu menekan tombol Daftar terdapat data yang kosong maka sistem akan menampilkan tulisan “Kotak ini tidak boleh kosong” pada kotak yang kosong 3. Jika tamu mendaftar lebih dari satu kali maka sistem akan menampilkan pesan “Data pribadi yang Anda masukkan sudah terdaftar sebagai pasien tetap” 4. Jika pada saat tamu menekan tombol Verifikasi dan kotak kode verifikasi kosong, maka sistem akan menampilkan tulisan “Kotak ini tidak boleh kosong” pada kotak kode verifikasi yang kosong 5. Jika pasien memasukkan kode verifikasi yang benar namun melebihi batas waktu yang ditentukan, maka data yang telah dimasukkan oleh tamu tersebut akan dihapus dari *database* dan sistem akan menampilkan

	pesan kesalahan “Batas waktu verifikasi sudah habis. Silahkan daftar kembali.” 6. Jika kode verifikasi yang dimasukkan oleh tamu tidak sesuai, maka sistem akan menampilkan pesan kesalahan “Akun Anda gagal di verifikasi, Kode Verifikasi yang Anda masukkan salah.”
Post-Condition	Tamu telah terdaftar menjadi pasien tetap di klinik.

4.6.4 Mendaftar Konsultasi

Tabel 4.7 Skenario Use Case Mendaftar Konsultasi

Mendaftar Konsultasi	
Actor	Pasien
Objective	Mengizinkan pasien untuk mendaftar konsultasi secara *online.*
Pre-Condition	Telah *login* sebagai pasien
Main Flow	1. Pasien memilih menu Daftar Konsultasi 2. Sistem menampilkan halaman Daftar Konsultasi yang berisi status pendaftaran *online*, jumlah antrian konsultasi, nomor antrian yang sedang diperiksa oleh dokter, pemberitahuan jam buka dan jam tutup pendaftaran *online,* pemberitahuan nomor antrian yang didapat jika pasien mendaftar konsultasi, dan tombol Daftar untuk mendaftar konsultasi 3. Pasien menekan tombol Daftar 4. Sistem akan memasukkan pasien kedalam *database* dan sistem akan memberitahu nomor urut antrian pasien tersebut melalui *SMS.* Lalu sistem akan menampilkan halaman Daftar Konsultasi yang berisi status pendaftaran *online*, jumlah antrian konsultasi, nomor antrian yang sedang diperiksa oleh dokter, pemberitahuan jam buka dan jam tutup pendaftaran *online,* pemberitahuan nomor urut antrian pasien yang sedang *login*, dan tombol Batal untuk membatalkan konsultasi
Alternative Flow	1. Jika Pasien melakukan pendaftaran konsultasi ketika kuota pendaftaran *online* sudah habis, maka sistem akan menampilkan pesan “Pendaftaran konsultasi online gagal dilakukan. Kuota pendaftaran online sudah habis.” 2. Jika Pasien melakukan pendaftaran konsultasi lebih dari batas waktu pendaftaran konsultasi *online*, maka sistem akan menampilkan pesan “Pendaftaran konsultasi online gagal dilakukan. Waktu

	pendaftaran konsultasi online sudah habis”
Post-Condition	Pasien berhasil mendaftar konsultasi secara *online.*

4.6.5 Membatalkan Konsultasi

Tabel 4.8 Skenario Use Case Membatalkan Konsultasi

Membatalkan Konsultasi	
Actor	Pasien
Objective	Mengizinkan pasien untuk membatalkan konsultasi
Pre-Condition	Telah *login* sebagai pasien dan telah melakukan pendaftaran konsultasi atau sudah terdaftar pada antrian konsultasi
Main Flow	1. Pasien memilih menu Daftar Konsultasi 2. Sistem akan menampilkan halaman Daftar Konsultasi yang berisi status pendaftaran *online*, jumlah antrian konsultasi, nomor antrian yang sedang diperiksa oleh dokter, pemberitahuan jam buka dan jam tutup pendaftaran *online,* pemberitahuan nomor urut antrian pasien yang sedang *login*, dan tombol Batal untuk membatalkan konsultasi 3. Pasien menekan tombol Batal 4. Sistem akan menghapus pasien dari antrian, mengirimkan pesan *SMS* bahwa nomor urut antrian berubah kepada pasien yang memiliki nomor urut antrian lebih besar dari pasien tersebut, dan akan menampilkan halaman Daftar Konsultasi
Alternative Flow	-
Post-Condition	Pasien berhasil dihapus dari antrian

4.6.6 Melihat Data Pribadi

Tabel 4.9 Skenario Use Case Melihat Data Pribadi

Melihat Rekam Medis	
Actor	Pasien
Objective	Mengizinkan pasien untuk melihat data pribadi dan rekam medis miliknya sendiri
Pre-Condition	Telah *login* sebagai pasien
Main Flow	1. Pasien memilih menu Profil 2. Sistem akan menampilkan halaman Profil yang berisi data pribadi pasien tersebut seperti nomor pasien, nama pasien, tempat dan tanggal lahir, alamat, nomor telepon, jenis kelamin, nama ayah, dan nama ibu, dan menampilkan rekam medis pasien tersebut seperti tanggal, elemen, diagnosa, dan terapi

Alternative Flow	-
Post-Condition	Pasien dapat melihat data pribadi dan melihat rekam medis miliknya sendiri

4.6.7 Mengubah Profil

Tabel 4.10 Skenario Use Case Mengubah Profil

Mengubah Profil	
Actor	Pasien
Objective	Mengizinkan pasien untuk mengubah alamat dan nomor *handphone* miliknya
Pre-Condition	Pasien telah berhasil *login* dan telah membuka halaman Profil
Main Flow	1. Pasien memilih tombol Edit Profil 2. Sistem akan menampilkan halaman untuk mengubah data pribadi pasien yang menampilkan kotak untuk memasukkan alamat yang berisi alamat pasien tersebut dan kotak untuk memasukkan nomor *handphone* yang berisi nomor *handphone* pasien tersebut 3. Pasien mengubah alamat atau nomor *handphone*, lalu menekan tombol Simpan 4. Sistem akan mengubah alamat dan nomor *handphone* pasien tersebut sesuai dengan alamat dan nomor *handphone* yang telah diubah oleh pasien tersebut
Alternative Flow	1. Jika alamat atau nomor *handphone* kosong maka sistem akan menampilkan tulisan “Kotak ini tidak boleh kosong” pada kotak yang kosong
Post-Condition	Pasien berhasil mengubah alamat dan nomor *handphone* miliknya

4.6.8 Mengisi Rekam Medis

Tabel 4.11 Skenario Use Case Mengisi Rekam Medis

Mengisi Rekam Medis	
Actor	Dokter
Objective	Mengizinkan dokter memasukkan data rekam medis pasien ke dalam *database*
Pre-Condition	Dokter telah berhasil *login*
Main Flow	1. Dokter memilih menu Konsultasi 2. Sistem akan menampilkan data pribadi pasien yang sedang konsultasi seperti nomor pasien, nama pasien, tanggal lahir, jenis kelamin, nama ayah, nama ibu, nomor *handphone,* dan alamat. Pada halaman

	konsultasi juga ditampilkan rekam medis pasien yang sedang konsultasi seperti tanggal, elemen, diagnosa, dan terapi. Terdapat juga tombol Resep Obat untuk menampilkan formulir untuk memasukkan resep obat, tombol Konsultasi Lanjut untuk menmapilkan formulir untuk memasukkan jadwal konsutlasi lanjut, dan formulir untuk memasukkan rekam medis ke dalam *database* yang terdiri dari kotak untuk memasukkan elemen, diagnosa, dan terapi 3. Dokter mengisi data rekam medis seperti elemen, diagnosa, dan terapi, lalu memilih tombol Simpan 4. Sistem akan memasukkan data rekam medis tersebut ke dalam *database,* kemudian sistem akan menampilkan halaman Konsultasi
Alternative Flow	1. Jika dokter menekan tombol Simpan namun terdapat data rekam medis yang kosong, maka sistem akan menampilkan tulisan "Kotak ini tidak boleh kosong" pada kotak yang kosong 2. Jika dokter mengisi data rekam medis dan menekan tombol Simpan, namun tidak ada pasien yang sedang konsultasi maka sistem akan menampilkan pesan kesalahan "Tidak dapat memasukkan data. Karena tidak ada pasien yang konsultasi."
Post-Condition	Data rekam medis yang dimasukkan oleh dokter berhasil tersimpan ke dalam *database*

4.6.9 Melihat Rekam Medis Pasien

Tabel 4.12 Skenario Use Case Melihat Rekam Medis Pasien

Melihat Rekam Medis Pasien	
Actor	Dokter dan Perawat
Objective	Mengizinkan dokter dan perawat melihat rekam medis pasien yang ingin dilihat rekam medisnya
Pre-Condition	Perawat dan dokter telah berhasil *login*
Main Flow	1. Dokter dan perawat memilih menu Pasien 2. Sistem akan menampilkan seluruh pasien tetap beserta data pribadinya seperti nomor pasien, nama pasien, tempat tanggal lahir, alamat, dan jenis kelamin. Pada setiap baris pasien tersebut terdapat tombol Rekam Medis 3. Dokter dan perawat memilih tombol Rekam Medis pada baris pasien yang ingin dilihat rekam medisnya 4. Sistem akan menampilkan halaman yang berisi data pribadi pasien tersebut seperti nomor pasien, nama pasien, tanggal lahir, jenis kelamin, alamat, nomor

	handphone, nama ayah, dan nama ibu. Pada halaman tersebut juga akan menampilkan data rekam medis pasien yang dipilih seperti tanggal, elemen, diagnosa, dan terapi
Alternative Flow	-
Post-Condition	Dokter dan perawat dapat melihat rekam medis pasien yang ingin dilihat rekam medisnya

4.6.10 Memasukkan Jadwal Konsultasi

Tabel 4.13 Skenario Use Case Memasukkan Jadwal Konsultasi

Memasukkan Jadwal Konsultasi	
Actor	Dokter
Objective	Mengizinkan dokter untuk memasukkan jadwal konsultasi lanjut pasien ke dalam *database*
Pre-Condition	Dokter telah berhasil *login* dan telah membuka halaman Konsultasi
Main Flow	1. Dokter menekan tombol Konsultasi Lanjut 2. Sistem akan langsung menampilkan kotak untuk memasukkan jadwal konsultasi lanjut dan keterangan 3. Dokter mengisi jadwal konsultasi lanjut dan keterangannya lalu memilih tombol Simpan 4. Sistem akan memasukkan jadwal konsultasi lanjut dan keterangannya ke dalam *database* dan sistem akan menampilkan halaman Konsultasi
Alternative Flow	1. Jika dokter menekan tombol Simpan namun jadwal konsultasi atau keterangan kosong, maka sistem akan menampilkan tulisan “Kotak ini tidak boleh kosong” pada kotak yang kosong 2. Jika dokter memasukkan data jadwal konsultasi lanjut dan menekan tombol Simpan, namun tidak ada pasien yang sedang konsultasi maka sistem akan menampilkan pesan kesalahan “Tidak dapat memasukkan data. Karena tidak ada pasien yang konsultasi.”
Post-Condition	Jadwal konsultasi lanjut yang dimasukkan oleh dokter dan perawat berhasil tersimpan di dalam *database*

4.6.11 Menulis Resep

Tabel 4.14 Skenario Use Case Menulis Resep

Menulis Resep	
Actor	Dokter

Objective	Mengizinkan dokter memasukkan resep obat pasien ke dalam *database* untuk dilihat oleh apoteker
Pre-Condition	Dokter telah berhasil *login* dan telah membuka halaman konsultasi
Main Flow	1. Dokter memilih menu Resep Obat 2. Sistem akan langsung menampilkan kotak untuk memasukkan resep obat 3. Dokter memasukkan resep obat pada kotak resep obat lalu memilih tombol Simpan 4. Sistem akan memasukkan resep obat ke dalam *database,* lalu menampilkan halaman Konsultasi
Alternative Flow	1. Jika dokter menekan tombol Simpan namun resep obat kosong, maka sistem akan menampilkan tulisan "Kotak ini tidak boleh kosong" pada kotak yang kosong 2. Jika dokter menulis resep dan menekan tombol Simpan, namun tidak ada pasien yang sedang konsultasi maka sistem akan menampilkan pesan kesalahan "Tidak dapat memasukkan data. Karena tidak ada pasien yang konsultasi."
Post-Condition	Resep obat yang dimasukkan oleh dokter tersimpan di dalam *database* dan dapat dilihat oleh apoteker

4.6.12 Menambah Antrian

Tabel 4.15 Skenario Use Case Menambah Antrian

Menambah Antrian	
Actor	Perawat
Objective	Mengizinkan perawat memasukkan pasien kedalam antrian konsultasi
Pre-Condition	Perawat telah berhasil *login*
Main Flow	1. Perawat memilih menu Antrian 2. Sistem akan menampilkan halaman Antrian yang berisi status pendaftaran *online,* jumlah kuota pendaftaran *online,* tombol untuk menentukan kuota pendaftaran *online,* daftar antrian yang berisi nomor pasien, nama pasien, status konsultasi, dan terdapat tombol Hapus dan tombol panah pada setiap baris antrian, tombol Selanjutnya, kotak pencarian, dan daftar seluruh pasien yang berisi nomor pasien, nama pasien, nomor *handphone,* alamat, tempat tanggal lahir, jenis kelamin, dan tombol Tambah pada setiap baris pasien 3. Perawat memilih tombol Tambah 4. Sistem akan menambahkan pasien yang dipilih ke

	dalam antrian konsultasi, sistem akan mengirimkan pesan *SMS* kepada pasien yang tadi ditambahkan mengenai nomor urut antrian yang didapat, dan menampilkan pada halaman antrian
Alternative Flow	-
Post-Condition	Perawat berhasil memasukkan pasien kedalam antrian

4.6.13 Menghapus Antrian

Tabel 4.16 Skenario Use Case Menghapus Antrian

Menghapus Antrian	
Actor	Perawat
Objective	Mengizinkan perawat menghapus pasien dari antrian konsultasi
Pre-Condition	Perawat telah berhasil *login* dan telah membuka halaman Antrian
Main Flow	1. Perawat memilih tombol Hapus pada baris nama pasien yang ingin dihapus dari antrian 2. Sistem akan menghapus pasien dari antrian, mengirimkan pesan *SMS* bahwa nomor urut antrian berubah kepada pasien yang memiliki nomor urut antrian lebih besar dari pasien yang tadi dihapus, dan menampilkan halaman Antrian
Alternative Flow	-
Post-Condition	Perawat berhasil menghapus pasien dari antrian konsultasi

4.6.14 Mengubah Urutan Antrian

Tabel 4.17 Skenario Use Case Mengubah Urutan Antrian

Mengubah Urutan Antrian	
Actor	Perawat
Objective	Mengizinkan perawat mengubah nomor urut pasien yang ingin diubah nomor urutnya
Pre-Condition	Perawat telah berhasil *login* dan telah membuka halaman Antrian
Main Flow	1. Perawat memilih tombol ikon panah atas atau panah bawah pada baris pasien yang ingin diubah nomor urutnya 2. Sistem akan mengubah nomor urut pasien tersebut sesuai dengan panah yang dipilih oleh perawat dan sistem akan mengirimkan pesan *SMS* mengenai nomor urut yang baru kepada pasien yang nomor urutnya terubah. Kemudian sistem menampilkan

	halaman Antrian
Alternative Flow	-
Post-Condition	Perawat berhasil mengubah nomor urut pasien yang ingin diubah nomor urutnya

4.6.15 Memajukan Urutan Antrian

Tabel 4.18 Skenario Use Case Memajukan Urutan Antrian

Memajukan Urutan Antrian	
Actor	Perawat
Objective	Mengizinkan perawat untuk memajukan urutan antrian konsultasi pasien
Pre-Condition	Perawat telah berhasil *login* dan telah membuka halaman Antrian
Main Flow	1. Perawat memilih tombol Selanjutnya 2. Sistem akan mengubah status antrian pasien menjadi "konsultasi" pada pasien yang akan konsultasi selanjutnya dan mengubah status antrian pasien yang sedang konsultasi sebelumnya menjadi "selesai". Sistem akan mengirimkan nomor urut pasien yang sedang diperiksa oleh dokter kepada seluruh pasien yang masih menunggu antrian melalui *SMS*
Alternative Flow	1. Jika pada saat perawat menekan tombol Selanjutnya namun seluruh pasien sudah diperiksa oleh dokter, maka sistem akan menampilkan pesan "Antrian sudah habis."
Post-Condition	Perawat berhasil memajukan urutan antrian konsultasi pasien

4.6.16 Menentukan Kuota Antrian

Tabel 4.19 Skenario Use Case Menentukan Kuota Antrian

Memajukan Kuota Antrian	
Actor	Perawat
Objective	Mengizinkan perawat untuk menentukan kuota pendaftaran konsultasi *online*
Pre-Condition	Perawat telah berhasil *login* dan telah membuka halaman Antrian
Main Flow	1. Perawat memilih tombol Pendaftaran Online 2. Sistem akan menampilkan modal yang berisi kotak untuk mengisi kuota pendaftaran 3. Perawat memasukkan jumlah kuota (nol atau lebih dari nol). Nol untuk menutup pendaftaran *online.*

	Sedangkan lebih dari nol untuk membuka pendaftaran *online.* Setelah memasukkan jumlah kuota perawat menekan tombol Simpan 4. Sistem akan mengubah kuota pendaftaran pada *database* sesuai dengan kuota pendaftaran yang dimasukkan oleh perawat. Setiap kali perawat membuka pendaftaran *online,* sistem akan mengirimkan pesan *SMS* untuk mengingatkan pasien yang memiliki janji konsultasi lanjut dengan dokter pada tanggal yang sama dengan tanggal pada setiap kali perawat membuka pendaftaran *online*. Kemudian sistem akan menampilkan halaman Antrian
Alternative Flow	1. Jika pada saat menekan tombol Simpan kuota kosong, maka sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Post-Condition	Kuota pendaftaran *online* berhasil diubah pada *database*

4.6.17 Melihat Resep

Tabel 4.20 Skenario Use Case Melihat Resep

Melihat Resep	
Actor	Apoteker
Objective	Mengizinkan apoteker untuk melihat seluruh resep obat yang ingin dilihat
Pre-Condition	Apoteker telah berhasil *login*
Main Flow	1. Apoteker memilih menu Resep Obat 2. Sistem akan menampilkan daftar resep obat yang diminta beserta nama dan nomor pasien yang memintanya, dan terdapat tombol Lihat pada setiap baris resep
Alternative Flow	1. Apoteker memilih tombol Lihat pada baris resep obat yang ingin dilihat. Sistem akan menampilkan halaman Detail Resep yang berisi resep obat yang dipilih oleh apoteker, tombol Selesai untuk memberitahu resep selesai, nama pasien, dan nomor pasien yang meminta resep terebut
Post-Condition	Apoteker berhasil melihat resep obat yang ingin dilihat

4.6.18 Memberitahu Resep Selesai

Tabel 4.21 Skenario Use Case Memberitahu Resep Selesai

Memberitahu Resep Selesai

Actor	Apoteker
Objective	Mengizinkan apoteker untuk mengirim pesan *SMS* kepada pasien bahwa obatnya sudah jadi
Pre-Condition	Apoteker telah berhasil *login* dan telah menekan tombol Lihat pada baris resep obat yang sudah selesai
Main Flow	1. Apoteker memilih tombol Selesai 2. Sistem akan mengirim pesan *SMS* kepada pasien bahwa obatnya sudah selesai dibuat dan menghapus resep obat tersebut dari *databse.* Kemudian sistem menampilkan halaman Resep Obat
Alternative Flow	-
Post-Condition	Apoteker berhasil mengirim pesan *SMS* kepada pasien bahwa obatnya sudah jadi

4.6.19 Menambah Pengguna

Tabel 4.22 Scenario Use Case Menambah Pengguna

Menambah Pengguna	
Actor	Admin
Objective	Mengizinkan admin untuk menambah pengguna yang dapat mengakses sistem
Pre-Condition	Admin telah berhasil *login*
Main Flow	1. Admin memilih tombol Tambah 2. Sistem akan menampilkan formulir untuk menambah pengguna 3. Admin memasukkan nama, *username*, *password,* status, alamat, dan nomor *handphone* lalu menekan tombol Simpan 4. Sistem akan menambahkan data pengguna ke dalam *database* lalu menampilkan halaman awal admin yang menampilkan data pengguna seperti nama, *username, password,* status, alamat, dan nomor *handphone*
Alternative Flow	1. Jika *username* yang dimasukkan telah digunakan oleh akun lain, maka sistem akan langsung menampilkan tulisan “x”. 2. Jika pada saat menekan tombol Simpan masih terdapat data yang kosong, maka sistem akan menampilkan pesan kesalahan “Kotak ini tidak boleh kosong” pada kotak yang kosong
Post-Condition	Admin berhasil menambah pengguna yang dapat mengakses sistem

4.6.20 Menghapus Pengguna

Tabel 4.23 Scenario Use Case Menghapus Pengguna

Menghapus Pengguna	
Actor	Admin
Objective	Mengizinkan admin untuk menghapus pengguna
Pre-Condition	Admin telah berhasil *login*
Main Flow	1. Admin memilih tombol Hapus pada baris pengguna yang akan dihapus 2. Sistem akan menghapus data pengguna dari *database* lalu menampilkan halaman awal admin yang menampilkan data pengguna seperti nama, *username, password,* status, alamat, dan nomor *handphone*
Alternative Flow	-
Post-Condition	Admin berhasil menghapus pengguna

4.6.21 Mengubah Pengguna

Tabel 4.24 Scenario Use Case Mengubah Pengguna

Mengubah Pengguna	
Actor	Admin
Objective	Mengizinkan admin untuk mengubah data pengguna
Pre-Condition	Admin telah berhasil *login*
Main Flow	1. Admin memilih tombol Ubah pada baris pengguna yang akan diubah 2. Sistem akan menampilkan formulir untuk mengubah data pengguna 3. Admin mengubah nama, *username*, *password,* status, alamat, dan nomor *handphone* lalu menekan tombol Simpan 4. Sistem akan mengubah data pengguna dengan data pengguna yang baru sesuai dengan yang dimasukkan oleh admin. Lalu sistem akan menampilkan halaman awal admin yang menampilkan data pengguna seperti nama, *username, password,* status, alamat, dan nomor *handphone*
Alternative Flow	1. Jika *username* yang dimasukkan telah digunakan oleh akun lain, maka sistem akan langsung menampilkan tulisan “x”. 2. Jika pada saat menekan tombol Simpan masih terdapat data yang kosong, maka sistem akan menampilkan pesan kesalahan “Kotak ini tidak boleh kosong” pada kotak yang kosong

Post-Condition	Admin berhasil mengubah data pengguna

4.6.22 Mengingat Password

Tabel 4.25 Scenario Use Case Mengingat Password

Mengingat Password	
Actor	Pasien
Objective	Mengizinkan pasien untuk mendapatkan *username* dan *password* miliknya
Pre-Condition	Pasien telah membuka halaman utama
Main Flow	1. Pasien menekan tombol Lupa Password 2. Sistem akan menampilkan formulir untuk memasukkan data yang terkait dengan data pribadi pasien tersebut 3. Pasien memasukkan nama lengkap, nama ayah, nama ibu, dan nomor *handphone* yang akan dikirimi *username* dan *password* lalu menekan tombol Kirim 4. Sistem akan mengirimkan *username* dan *password* kepada pasien melalui pesan *SMS*
Alternative Flow	1. Jika pada saat menekan tombol Kirim masih terdapat data yang kosong, maka sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong 2. Jika nama pasien yang dimasukkan tidak sesuai dengan nama ayah atau nama ibu yang dimasukkan maka sistem akan menampilkan pesan kesalahan "Data yang Anda masukkan tidak ditemukan. Silahkan memasukkan data diri Anda dengan Benar."
Post-Condition	Admin berhasil mengubah data pengguna

4.5.2 Membuat Cadangan Rekam Medis

Tabel 4.26 Scenario Use Case Membuat Cadangan Rekam Medis

Membuat Cadangan Rekam Medis	
Actor	Admin
Objective	Mengizinkan admin membuat cadangan rekam medis pasien dalam format .sql
Pre-Condition	Admin telah berhasil *login*
Main Flow	1. Admin menekan tombol Backup 2. Sistem akan mengunduh rekam medis pasien dalam format .sql
Alternative Flow	-
Post-Condition	Admin dapat membuat cadangan rekam medis pasien

	dalam format .sql

4.6 Pembahasan Hasil Analisis Kebutuhan

Dari analisis kebutuhan yang telah dilakukan, sistem ini menghasilkan 22 kebutuhan yang dapat membantu mengelola data pasien di klinik Drg. Damayanti. Dari 23 kebutuhan yang didapat kebutuhan yang dapat membantu dokter dalam mencatat rekam medis adalah kebutuhan mengisi rekam medis, melihat rekam medis pasien, menulis resep, memasukkan jadwal konsultasi. Untuk melihat resep obat yang diminta dokter, apoteker dapat menggunakan kebutuhan melihat resep dan kebutuhan memberitahu resep selesai. Sedangkan kebutuhan yang dapat membantu pasien dalam melakukan pendaftaran konsultasi adalah kebutuhan mendaftar konsultasi atau membatalkan konsultasi. Pendaftaran konsultasi menghasilkan antrian konsutlasi yang dapat dikelola oleh perawat melalui kebutuhan manambah antrian, menentukan kuota antrian, menghapus antrian, mengubah urutan antrian, dan memajukan antrian. Pendaftaran konsultasi dapat dilakukan jika seseorang sudah melakukan pendaftaran sebagai pasien tetap di klinik. Dengan kebutuhan mendaftar sebagai pasien setiap orang dapat mendaftarkan dirinya sebagai pasien tetap di klinik. Kebutuhan membuat cadangan rekam medis digunakan untuk memastikan keamanan penyimpanan data pasien dan rekam medis, karena data pasien dan rekam medis dapat disimpan dalam format lain untuk mengantisipasi jika *database* mengalami masalah.

BAB 5 PERANCANGAN DAN IMPLEMENTASI

5.1 Perancangan

Setelah proses analisis kebutuhan selesai dilakukan, tahap selanjutnya dalam pengembangan perangkat lunak adalah perancangan. Perancangan dilakukan berdasarkan hasil dari analisis kebutuhan yang telah dilakukan. Pada tahap ini objek-objek akan diidentifikasikan berdasarkan spesifikasi kebutuhan dan *use case scenario* yang ada pada tahap analisis kebutuhan. Objek-objek tersebut akan dijelaskan detailnya dan bagaimana hubungan antar objek tersebut. Proses perancangan pada sistem manajemen data pasien klinik gigi berbasis *web* ini dibagi menjadi empat tahap yaitu perancangan *sequence diagram, class diagram, database,* dan antarmuka. Perancangan *class diagram* akan dibagi ke dalam sub bab yaitu perancangan umum dan perancangan detail.

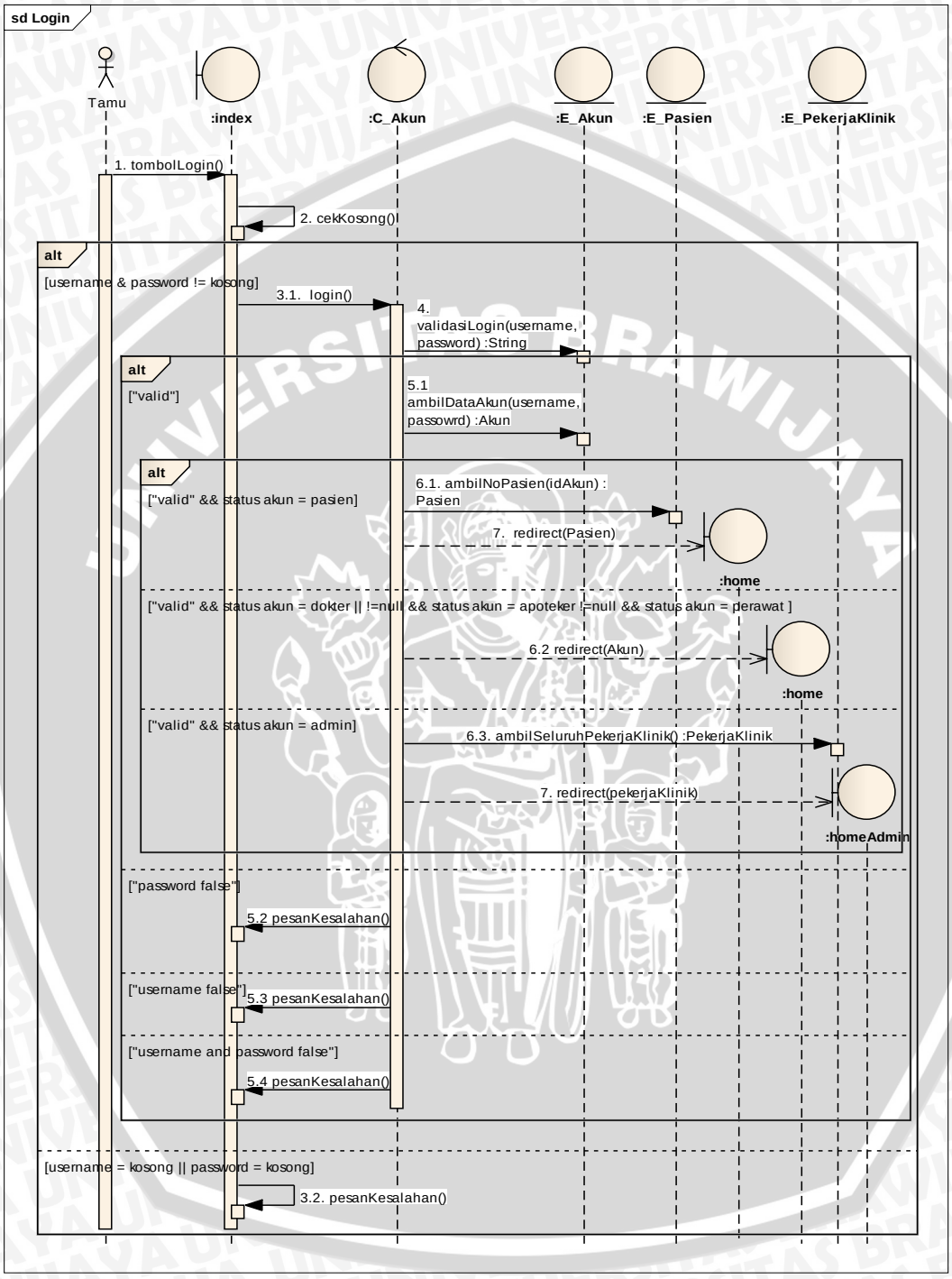
5.1.1 *Sequence Diagram*

Sequence diagram merupakan diagram yang menjelaskan urutan proses yang terjadi untuk mencapai tujuan *use case,* interaksi antar objek, operasi apa saja yang terlibat diantara objek-objek. Objek-objek yang ada pada *sequence diagram* merupakan hasil identifikasi berdasarkan spesifikasi kebutuhan dan *use case scenario* yang ada pada tahap analisis kebutuhan. *Sequence diagram* dibuat untuk setiap *use case* yang didapat dari hasil analisis kebutuhan. Sehingga terdapat 22 *sequence diagram* yang dibuat. Namun hanya lima buah *sequence diagram* yang akan ditampilkan sebagai sampel yaitu *sequence diagram* untuk kebutuhan *login, logout,* menjadi pasien tetap, mendaftar konsutlasi, dan mengisi rekam medis.

5.1.1.1 Login

Sequence diagram login pada Gambar 5.1 menjelaskan alur kontrol yang terjadi pada saat menjalankan fungsi *login*. Pertama tamu memilih menu Login pada halaman utama *website.* Kemudian sistem akan menampilkan formulir login. Tamu memasukkan *username* dan *password* lalu menekan tombol Login. Jika *username* dan *password* kosong maka sistem akan menampilkan pesan kesalahan pada kotak yang kosong. Namun jika tidak kosong maka sistem akan menjalankan fungsi login(). Method login() akan memanggil method validasiLogin(username, password):String pada model untuk memeriksa apakah *username* dan *password* sesuai dengan yang ada pada *database* atau tidak. Jika method validasiLogin(username, password):String menghasilkan “valid” maka sistem akan menjalankan method ambilDatakun(username, password):Akun. Dan jika yang *login* berstatus pasien maka sistem akan menjalankan method ambilNoPasien(idAkun):Akun untuk mengambil data pasien yang akan *login*. Lalu sistem akan meneruskan ke halaman *home.* Jika yang *login* adalah dokter, perawat, atau apoteker maka sistem akan langsung menampilkan halaman home. Jika yang *login* adalah admin maka sistem akan menjalankan method ambilSeluruhPekerjaKlinik():PekerjaKlinik untuk mengambil seluruh data pekerja

klinik dan ditampilkan pada halaman home. Namun jika *username* salah, *password* salah, atau *username* dan *password* salah maka sistem akan menampilkan pesan kesalahan.

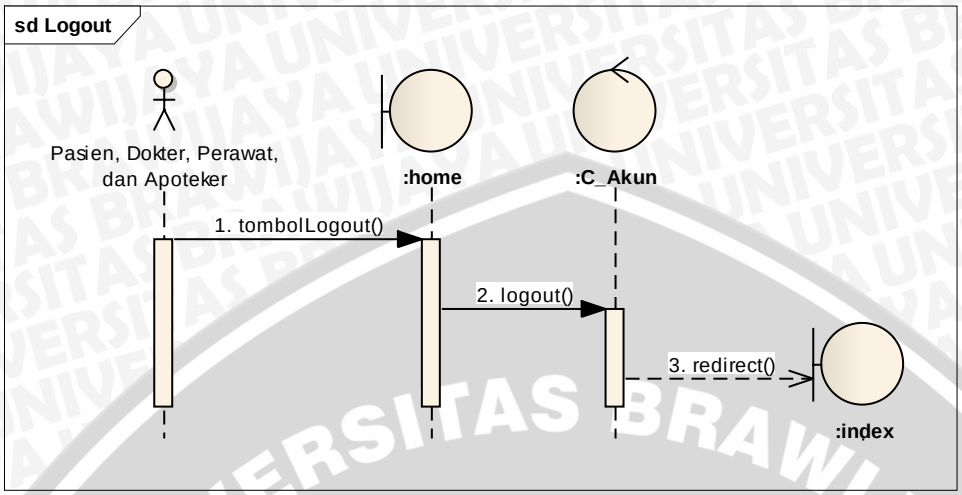


Gambar 5.1 Sequence Diagram Login

5.1.1.2 Logout

Sequence diagram *logout* pada Gambar 5.2 menjelaskan urutan proses jika pasien *logout*. Pertama pasien, dokter, perawat, dan apoteker memilih tombol

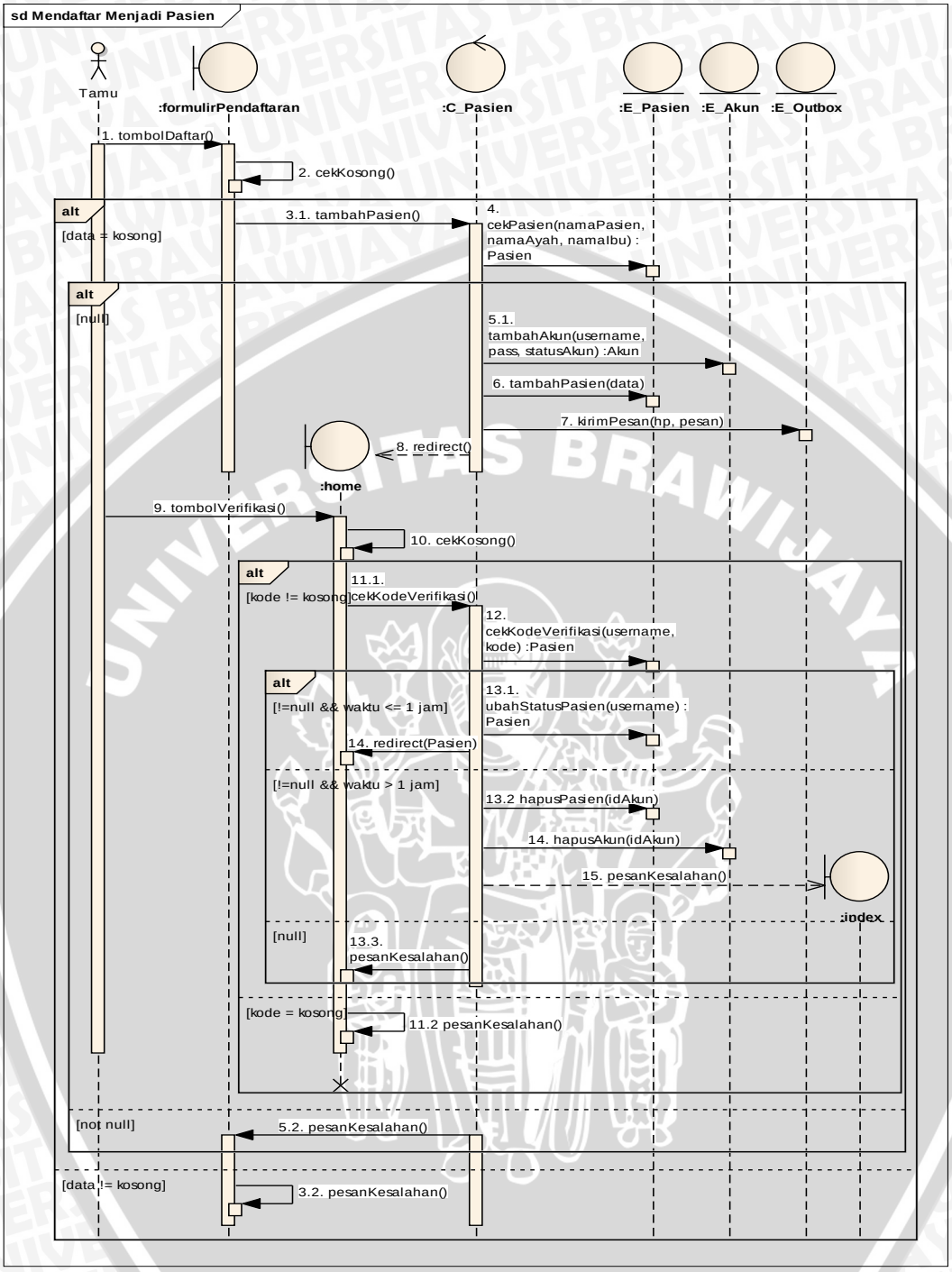
Logout. Sistem akan menjalankan method logout(). Kemudian sistem akan menampilkan halaman utama.



Gambar 5.2 Sequence Diagram Logout

5.1.1.3 Mendaftar Menjadi Pasien

Sequence diagram mendaftar menjadi pasien pada Gambar 5.3 menjelaskan urutan proses pada saat tamu ingin mendaftar menjadi pasien tetap. Pertama tamu memasukkan data pendaftaran lalu menekan tombol Daftar. Jika terdapat data yang kosong maka sistem akan menampilkan kembali halaman pendaftaran beserta pesan kesalahan pada kotak yang belum diisi. Namun jika data terisi semua maka sistem akan menjalankan method tambahPasien(). Kemudian sistem akan menjalankan method cekPasien(namaPasien, namaAyah, namaIbu) yang akan memeriksa apakah pasien sudah terdaftar atau belum. Jika pasien sudah terdaftar maka sistem akan menampilkan pesan kesalahan. Namun jika pasien belum terdaftar maka sistem akan menjalankan method tambahAkun(username, pass, statusAkun):Akun dan tambahPasien(data) untuk memasukkan data pasien kedalam *database*. SIstem juga akan menjalankan method kirimPesan(hp, pesan) untuk mengirimkan kode verifikasi kepada pasien. Setelah menerima kode verifikasi tamu memasukkan kode verifikasi lalu menekan tombol Verifikasi. Jika kode verifiaksi kosong maka sistem akan memunculkan pesan kesalahan pada kotak yang kosong. Namun jika kode verifikasi terisi maka sistem akan menjalankan method cekKodeVerifikasi() yang akan memanggil cekKodeVerifikasi(username, kode):Pasien pada model untuk melihat apakah kode verifikasi sesuai dengan yang dikirimkan melalui sms atau tidak. Jika kode verifikasi benar dan waktu masih kurang dari satu jam setelah pendaftaran maka sistem akan menjalankan method ubahStatusPasien(username):Pasien lalu menampilkan halaman pasien. Jika kode verifikasi benar namun waktu sudah lebih dari satu jam maka sistem akan menjalankan method hapusPasien(idAkun) dan method hapusAkun(idAkun) untuk menghapus pasien dari *database* lalu menampilkan pesan kesalahan pada halaman utama. Jika kode verifikasi tidak sesuai maka sistem akan menampilkan pesan kesalahan.

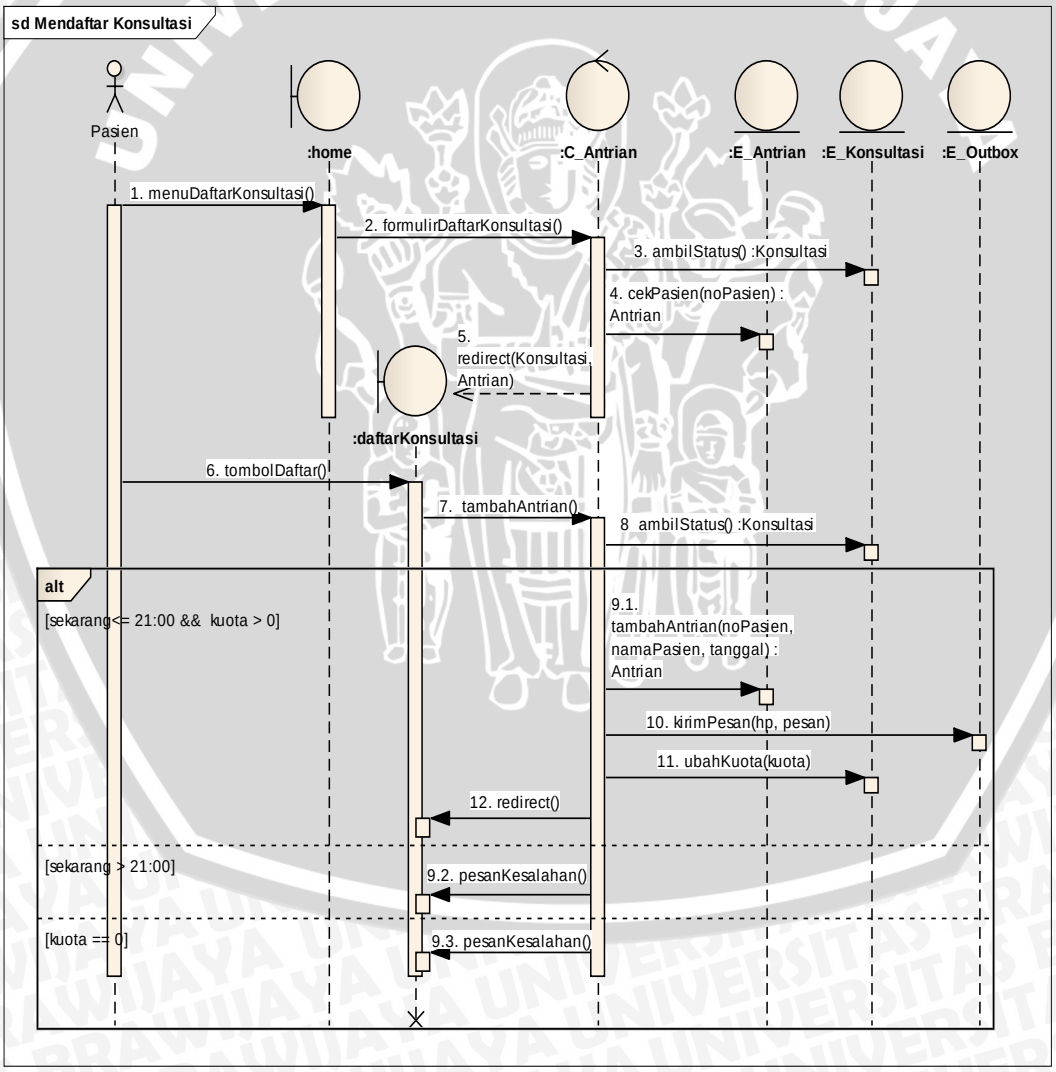


Gambar 5.3 Sequence Diagram Mendaftar Menjadi Pasien

5.1.1.4 Mendaftar Konsultasi

Sequence diagram mendaftar konsultasi pada Gambar 5.4 menjelaskan urutan proses pendaftaran konsultasi. Pertama pasien memilih menu Daftar Konsultasi. Kemudian sistem akan menjalankan method formulirDaftarKonsultasi() yang akan memanggil method ambilStatus():Konsultasi untuk mengambil status pendaftaran *online* dari *database* dan akan dikembalikan dalam bentuk *array,* dan memanggil method

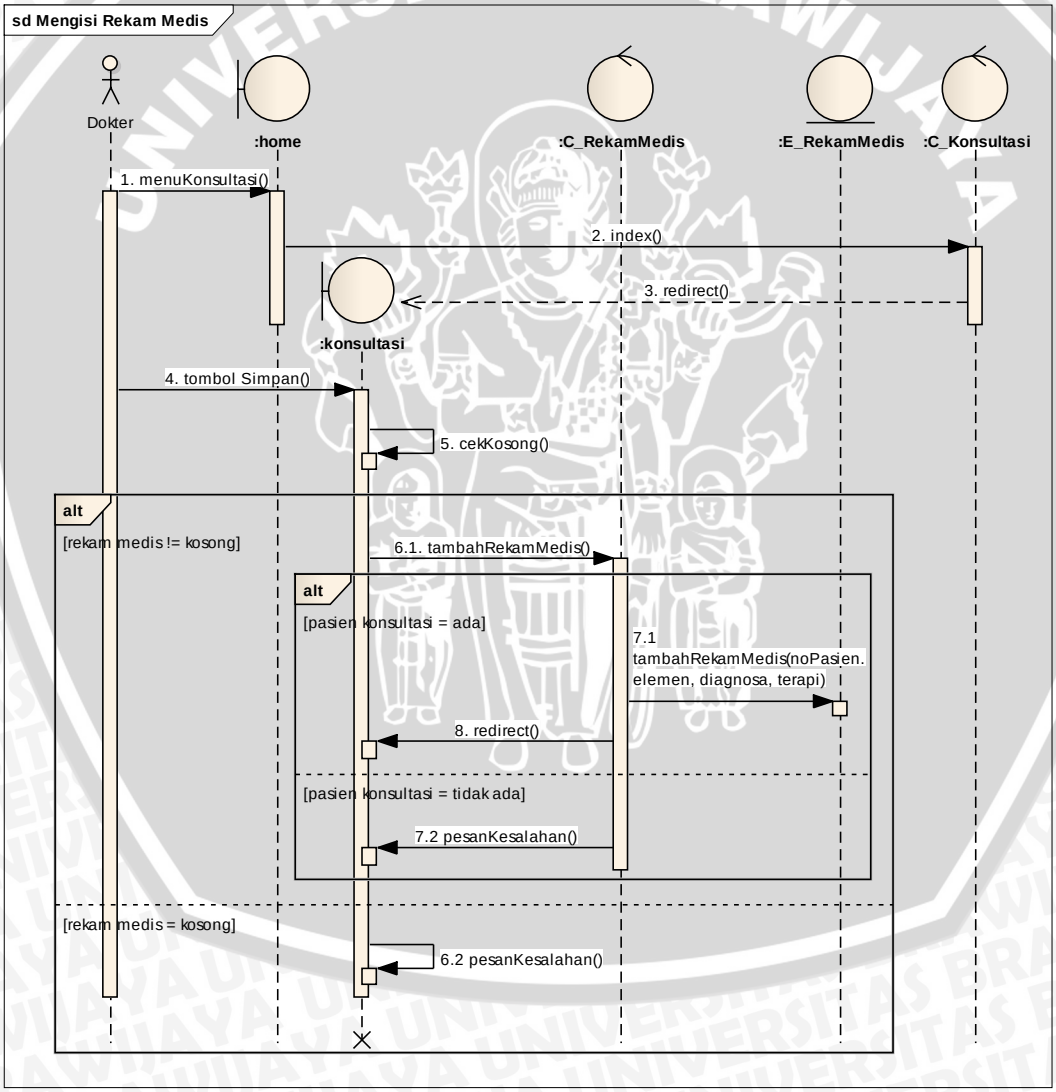
cekPasien(noPasien):Antrian untuk memeriksa apakah pasien sudah terdapat pada antrian atau belum dan hasilnya akan dikembalikan dalam bentuk *array.* Kemudian pasien menekan tombol Daftar. Kemudian sistem akan menjalankan method tambahAntrian() pada *controller.* Method tersebut akan menjalankan method method ambilSatus():Konsultasi untuk mengetahui jumlah kuota pendaftaran. Jika pendaftaran dilakukan sebelum jam Sembilan malam dan kuota lebih dari nol maka sistem akan menjalankan method tambahAntrian(noPasien, namaPasien, tanggal):Antrian untuk menyimpan data pasien ke dalam tabel Antrian. Method kirimPesan(hp, pesan) akan mengirimkan pesan nomor urut antrian pasien melalui *SMS.* Method ubahKuota(kuota) untuk mengurangi jumlah kuota pendaftaran. Kemudian sistem akan menampilkan halaman daftar konsultasi. Jika pendaftaran dilakukan melebihi jam sembilan malam maka sistem akan menampilkan pesan kesalahan. Jika pendaftaran dilakukan ketika kuota sama dengan nol maka sistem juga akan menampilkan pesan kesalahan



Gambar 5.4 Sequence Diagram Mendaftar Konsultasi

5.1.1.5 Mengisi Rekam Medis

Sequence diagram mengisi rekam medis pada Gambar 5.8 menjelaskan urutan proses pada aktivitas mengisi rekam medis. Pertama dokter memilih menu Konsultasi. Kemudian sistem akan menjalankan method index() untuk menampilkan halaman Konsultasi. Lalu dokter memasukkan data rekam medis seperti elemen, diagnosa, dan terapi lalu menekan tombol Simpan. Jika data rekam medis tidak ada yang kosong maka sistem akan menjalankan method tambahRekamMedis(). Jika terdapat pasien yang sedang konsultasi maka sistem akan memanggil method tambahRekamMedis(noPasien, elemen, diagnosa, tindakan) untuk memasukkan data rekam medis ke dalam *database.* Namun jika tidak terdapat pasien yang konsultasi maka sistem akan menampilkan pesan kesalahan. Jika terdapat data rekam medis yang kosong, maka sistem akan menampiilkan pesan kesalahan pada kotak yang kosong.



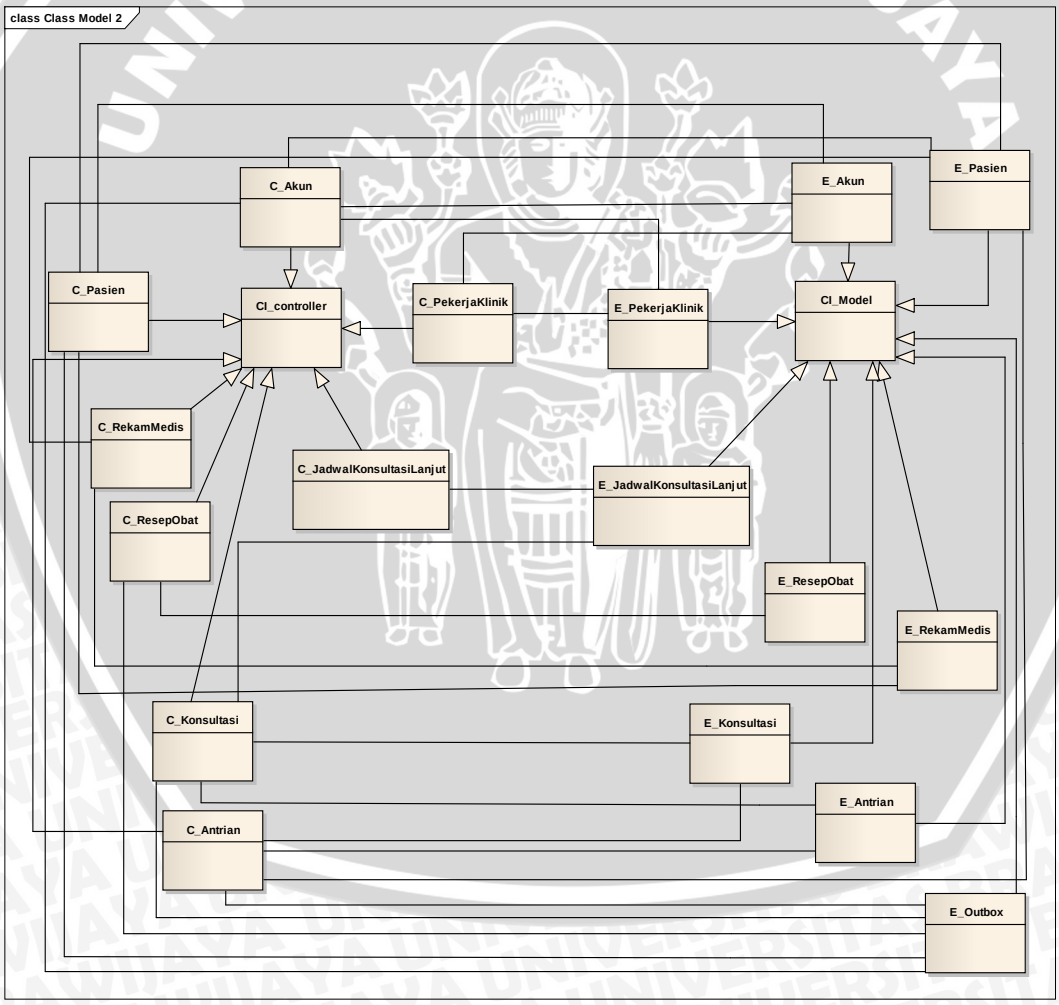
Gambar 5.5 Sequence Diagram Mengisi Rekam Medis

5.1.2 *Class Diagram*

Perancangan c*lass diagram* dilakukan untuk menggambarkan objek-objek yang terbentuk dan relasi diantara objek-objek tersebut. Relasi diantar objek-objek akan dijelaskan pada sub bab perancangan umum. *Class diagram* juga menunjukkan atribut-atribut dan operasi-operasi yang ada pada suatu objek yang akan dijelaskan pada sub bab perancangan detail. Sub bab perancangan detail juga akan menjelaskan bagaimana algoritma pada setiap operasi yang ada pada suatu klas.

5.1.2.1 Perancangan Umum

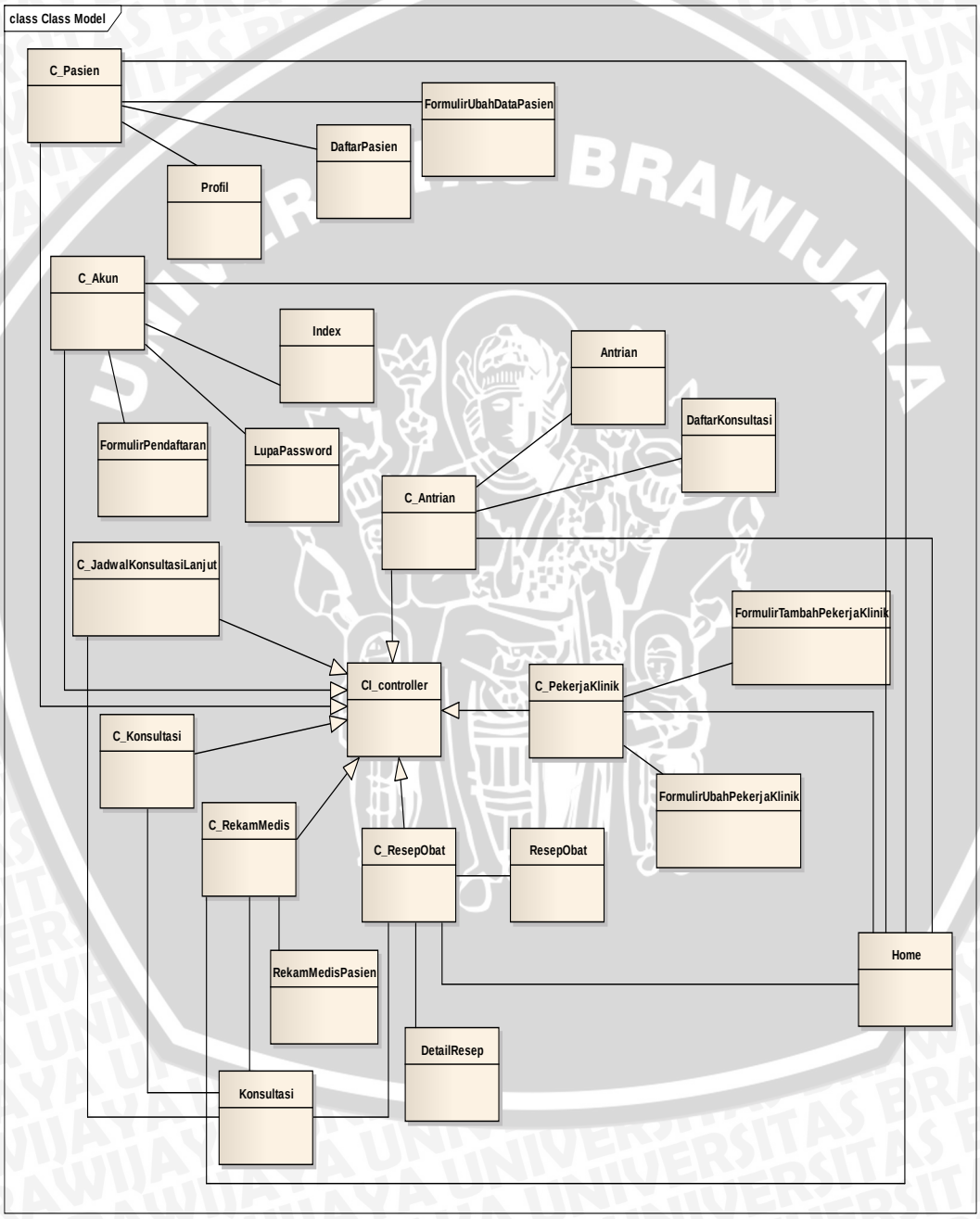
Perancangan umum dilakukan untuk mendeskripsikan objek-objek yang terbentuk dan relasi diantara objek-objek tersebut. Gambar *class diagram* akan dibagi menjadi dua yaitu relasi antar *controller* dengan *model* dan relasi *controller* dengan *boundary. Class diagram* akan di jelaskan melalui gambar 5.6 dan 5.7 di bawah ini.



Gambar 5.6 Class Diagram Relasi antar Controller dengan Model

Gambar 5.6 menjelaskan perancangan *class diagram* pada sistem manajemen data pasien klinik gigi berbasis *web.* Karena sistem ini

mengimplementasikan *CodeIgniter*, maka terdapat kelas CI_controller dan CI_Model sebagai klas induk. CI_controller merupakan induk kelas dari C_Pasien, C_RekamMedis, C_Akun, C_JadwalKonsultasiLanjut, C_Konsultasi, C_Antrian, C_ResepObat, C_Dokter, C_Perawat, dan C_Apoteker. Sedangkan CI_Model merupakan induk klas dari E_Pasien, E_RekamMedis, E_Akun, E_JadwalKonsultasiLanjut, E_Konsultasi, E_Antrian, E_ResepObat, dan E_Outbox. Klas-klas yang merupakan anak klas dari CI_controller juga saling berinteraksi dengan klas-klas yang merupakan anak klas dari CI_Model.



Gambar 5.7 Class Diagram Relasi antar Boundary dengan Controller

Gambar 5.7 menjelaskan interaksi antar klas yang terjadi diantara klas-klas *boundary* dengan klas-klas *controller.*

5.1.2.2 Perancangan Detail

Perancangan detail merupakan perancangan yang menjelaskan atribut dan operasi apa saja yang terdapat pada klas-klas yang sudah digambarkan pada Gambar 5.6 dan Gambar 5.7. Perancangan Detail juga akan menjelaskan algoritma-algoritma pada setiap operasi. Pada perancangan detail klas setiap klas harus dijelaskan atribut dan oeprasinya. Namun hanya empat klas yang akan dijelaskan sebagai sampel yaitu klas C_Akun, C_Pasien, C_Antrian, dan C_RekamMedis.

1. Klas C_Akun

Detail dari klas C_Akun yang berisi atribut dan operasi apa saja yang terdapat pada klas tersebut akan dijelaskan pada Tabel 5.1.

Tabel 5.1 Detail Klas C_Akun

Nama Atribut	Visibility	Tipe
username	private	String
password	private	String
statusAkun	private	String
idAkun	private	int
Nama Operasi	**Visibility**	**Tipe**
setUsername(username)	public	void
setPassword(password)	public	void
setStatusAkun(statusAkun)	public	void
setIdAkun(idAkun)	public	void
getUsername()	public	String
getPassword()	public	String
getStatusAkun()	public	String
getIdAkun()	public	int
index()	public	void
home()	public	void
lupaPassword()	public	void
login()	public	void
logout()	public	void

2. Klas C_Antrian

Detail dari klas C_Antrian yang berisi atribut dan operasi apa saja yang terdapat pada klas tersebut akan dijelaskan pada Tabel 5.2.

Tabel 5.2 Tabel Detail Klas C_Antrian

Nama Atribut	Visibility	Tipe
noAntrian	private	int
noPasien	private	int
tanggal	private	String
statusAntrian	private	String

Tabel 5.2 (lanjutan)

Nama Atribut	Visibility	Tipe
kuota	private	int
Nama Operasi	**Visibility**	**Tipe**
setNoAntrian(noAntrian)	public	void
setNoPasien(noPasien)	public	void
setTanggal(tanggal)	public	void
setStatusAntrian(statusAntrian)	public	void
setKuota(kuota)	public	void
getNoAntrian()	public	int
getNoPasien()	public	int
getTanggal()	public	String
getStatusAntrian()	public	String
getKuota()	public	int

3. Klas C_Pasien

Detail dari klas C_Pasien yang berisi atribut dan operasi apa saja yang terdapat pada klas tersebut akan dijelaskan pada Tabel 5.3.

Tabel 5.3 Tabel Detail Klas C_Pasien

Nama Atribut	Visibility	Tipe
noPasien	private	int
namaPasien	private	String
username	private	String
password	private	String
pass	private	String
ttl	private	String
alamat	private	String
jenisKelamin	private	String
namaAyah	private	String
namaIbu	private	String
hp	private	String
statusPendaftaran	private	String
kode	private	String
Nama Operasi	**Visibility**	**Tipe**
getPass()	public	String
getTtl()	public	String
getAlamat()	public	String
getJenisKelamin()	public	String
getNamaAyah()	public	String
getNamaIbu()	public	String
getHp()	public	String
getStatusPendaftaran()	public	String
getKode()	public	String

Tabel 5.3 (Lanjutan)

Nama Operasi	Visibility	Tipe
index()	public	void
formulirPendaftaran()	public	void
ambilSeluruhPasien()	public	void
ambilDataPasien()	public	void
tambahPasien()	public	void
ubahDataPasien()	public	void
formulirUbahDataPasien()	public	void
cekKodeVerifikasi()	public	void
kirimPassword()	public	void

4. Klas C_RekamMedis

Detail dari klas C_RekamMedis yang berisi atribut dan operasi apa saja yang terdapat pada klas tersebut akan dijelaskan pada Tabel 5.4.

Tabel 5.4 Tabel Detail Klas C_RekamMedis

Nama Atribut	Visibility	Tipe
noPasien	private	int
elemen	private	String
diagnosa	private	String
terapi	private	String
Nama Operasi	**Visibility**	**Tipe**
setNoPasien(noPasien)	public	void
setElemen(elemen)	public	void
setDiagnosa(diagnosa)	public	void
setTerapi(terapi)	public	void
getNoPasien()	public	int
getElemen()	public	String
getDiagnosa()	public	String
getTerapi()	public	String
tambahRekamMedis()	public	void
ambilRekamMedis()	public	void

Setelah dijelaskan atribut dan operasi apa saja yang terdapat pada klas-klas, lalu akan dijelaskan algoritma pada operasi tersebut. Algortima pada operasi akan menunjukkan bagaimana alur yang dijalankan sistem untuk dapat menghasilkan suatu keluaran. Perancangan algoritma operasi dibuat hanya beberapa sampel operasi saja yaitu yaitu operasi login() pada klas C_Akun, operasi cekKodeVerifiaksi() pada klas C_Akun, operasi tambahAntrin() pada klas C_Antrian, dan operasi tambahRekamMedis() pada klas C_RekamMedis.

Nama Klas : C_Akun

Nama Operasi : login

Algoritma :

```
inisialisasi username, password
password = enkripsi md5 password
result = memeriksa apakah username dan password valid
if(result = "valid") then
    data = array[] of String
    result = ambil data akun dari database berdasarkan username dan
     password
    data[idAkun] = result[idAkun]
    data[username] = result[username]
    data[statusAkun] = result[statusAkun]
    if(data[statusAkun] = "pasien") then
        result  = mengambil data pasien dari database berdasarkan
        username   dan password
        data[noPasien] = result[noPasien]
        data[namaPasien] = result[namaPasien]
        data[hp] = result[hp]
        data[statusPendaftaran] = result[StatusPendaftaran]
        set session data
    else  if(data[statusAkun]  ="dokter"  atau  "perawat"  atau
        "apoteker") then
        set session data
    else if(data[statusAkun] ="admin") then
        set session data
        mengambil seluruh data pengguna yang dapat mengakses
        sistem dari database
    end if
    menampilkan halaman awal pengguna
else if(result = "password false") then
```

menampilkan pesan kesalahan pada halaman login

else if(result = "username false") then

menampilkan pesan kesalahan pada halaman login

else if(result = "username and password false")then

menampilkan pesan kesalahan pada halaman login

end if

Nama Klas : C_Pasien

Nama Operasi : cekKodeVerifikasi

Algoritma :

Inisialisasi kodeVerifikasi, username

idAkun = idAkun yang sedang login

result = ambil data pasien dari database berdasarkan kodeVerifikasi dan username

if(result = not null)then

ambil tanggal pendaftaran dari database berdasakan username dan password

a = tanggal dan waktu sekarang

b = tanggal pendaftaran + 1 jam

if(a<b) then

ubah status pasien pada database menjadi "terverifikasi" berdasarkan username

mengambil data pasien dari database berdasarkan idAkun

unset session data

set session data pasien

menampilkan data pasien pada halaman home

else

hapus pasien dari database berdasarkan idAkun

hapus akun dari database berdasarkan idAkun

menampilkan pesan kesalahan pada halaman awal

end if

else

```
    menampilkan pesan kesalahan pada halaman home
end if
```

Nama Klas : C_Antrian

Nama Operasi : tambahAntrian

Algoritma :

```
session_data = data session
statusAkun = session_data[statusAkun]
if(statusAkun = "pasien") then
    noPasien = session_data[noPasien]
    namaPasien = session_data[namaPasien]
    hp = session_data[hp]
    jam = waktu sekarang
    kuota = ambil kuota dari tabel konsultasi
    if(jam <= "21:00" & kuota > 0) then
        memasukkan pasien yang sedang login ke dalam tabel antrian
        noAntrian = nomor antrian yang didapat dari hasil memasukkan pasien
        yang sedang login ke dalam antrian
        pesan = "Anda sudah melakukan pendaftaran konsultasi pada Klinik Drg.
        Damayanti. Anda mendapat nomor antrian " + noAntrian;
        mengirim pesan ke hp
        kuota = kuota – 1
        ubah kuota pada database sesuai dengan hasil kuota
        menampilkan halaman daftar konsultasi
    else if (jam > "21:00") then
        noAntrian = 0
        menampilkan pesan kesalahan pada halaman daftar konsultasi
    else if (kuota == 0) then
        noAntrian = 0
        menampilkan pesan kesalahan pada halaman daftar konsultasi
    end if
else
    inisialisasi noPasien, namaPasien, hp
```

memasukkan noPasien dan namaPasien ke dalam tabel antrian

noAntrian = nomor antrian yang didapat dari hasil memasukkan pasien dalam antrian

pesan = "Anda sudah melakukan pendaftaran konsultasi pada Klinik Drg. Damayanti. Anda mendapat nomor antrian " + noAntrian;

mengirim pesan ke hp

mengirim pesan ke nomor hp

menampilkan halaman antrian

end if

Nama Klas : C_RekamMedis

Nama Operasi : tambahRekamMedis

Algoritma :

Inisialisasi noPasien, elemen, diagnosa, terapi

If (noPasien == 0) then

session_data = data session

session_data['kesalahan'] = 'true'

tampil halaman konsultasi

else

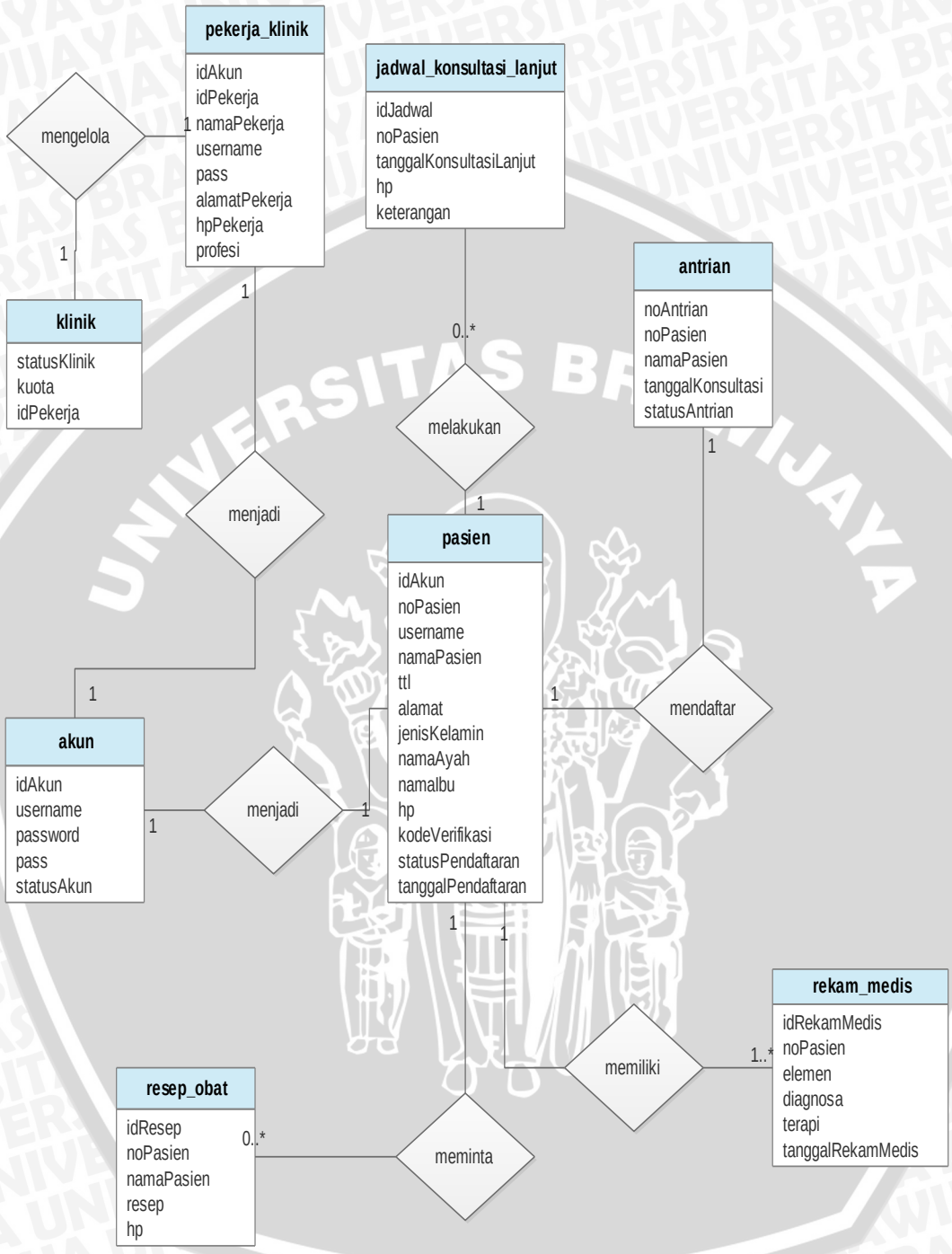
memasukkan noPasien, elemen, diagnosa, terapi ke dalam tabel rekam_medis pada database

menampilkan halaman konsultasi

5.1.3 Perancangan *Database*

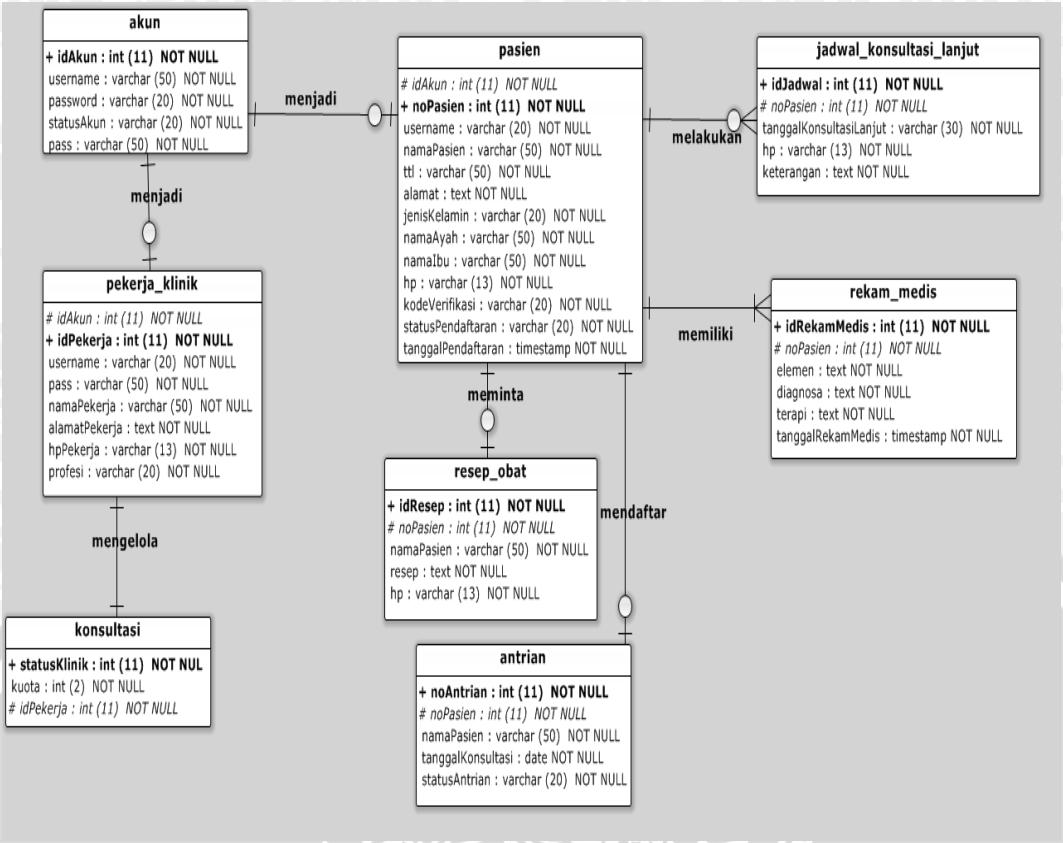
Perancangan *database* akan dilakukan secara konseptual dan secara fisik. Perancangan secara konseptual akan dilakukan menggunakan *Entity Relationship Diagram (ERD) yang* akan digambarkan pada Gambar 5.8. Terdapat delapan entitas diantaranya adalah entitas resep_obat, rekam_medis, jadwal_konsultasi_lanjut, pasien, akun, pekerja_klinik, konsultasi, dan antrian. Pada gambar tersebut terlihat terdapat entitas yang berelasi. Entitas yang berelasi diantaranya adalah entitas pasien yang memiliki *primary key* yaitu noPasien berelasi dengan entitas resep_obat, rekam_medis, jadwal_konsultasi_lanjut, antrian. Sehingga entitas-entitas tersebut memiliki *foreign key* yaitu noPasien. Selain itu entitas akun yang memiliki *primary key* yaitu idAkun berelasi dengan entitas pasien dan pekerjaklinik sehingga entitas-entitas tersebut memiliki *foregin key* yaitu idAkun. Selain itu entitas

pekerja_klinik berelasi dengan entitas konsultasi sehingga entitas konsultasi memiliki *foreign key* yaitu idPekerja.



Gambar 5.8 Entity Relationship Diagram

Setelah *Entity Relationship Diagram (ERD)* dibuat langkah selanjutnya adalah membuat perancangan *database* secara fisik. Jumlah tabel yang akan dibuat sama dengan jumlah entitas yang telah didefinisikan. Karena tidak terdapat atribut baru yang terbentuk dari relasi antar entitas. Sehingga tidak terdapat tabel baru yang dibutuhkan untuk menyimpan atribut tersebut. Perancangan database secara fisik akan dijelaskan pada Gambar 5.9



Gambar 5.9 Physical Data Model

Dari perancangan *physical data model* pada Gambar 5.9 maka dapat dijelaskan struktur dari tabel-tabel diatas. Berikut ini merupakan penjelasan struktur dari tabel-tabel yang terbentuk pada perancangan *database* secara fisik.

Tabel 5.5 Struktur Tabel akun

No	Nama Field	Tipe Data	Keterangan
1	idAkun (pk)	int(11)	ID akun (1,2,3…) yang diberikan secara otomatis menggunakan fungsi *Auto Increment*
2	username	varchar(20)	*Username* yang digunakan untuk *login*
3	password	varchar(50)	*Password* menggunakan enkripsi *md5*
4	pass	varchar(50)	*Password* yang tidak dienkripsi *md5*
5	statusAkun	varchar(20)	Status akun (pasien, perawat, dokter, apoteker)

Tabel 5.5 merupakan struktur dari tabel akun. Tabel akun memiliki kolom idAkun yang merupakan *primary key, username, password,* dan statusAkun. Tabel akun menyimpan data yang diperlukan dalam proses *login.*

Tabel 5.6 Struktur Tabel pasien

No	Nama Field	Tipe Data	Keterangan
1	idAkun (fk)	int(11)	idAkun merupakan *foreign key* dari tabel akun
2	noPasien (pk)	int(5)	Nomor pasien tetap (1,2,3…) dan Nomor pasien sementara diawali angka 999 dan digabung dengan idAkun
3	username	varchar(20)	Sama dengan *username* yang ada pada tabel akun
4	namaPasien	varchar(50)	Nama lengkap pasien
5	ttl	varchar(50)	Tempat dan tanggal lahir pasien
6	alamat	text	Alamat pasien
7	jenisKelamin	varchar(20)	Jenis kelamin pasien
8	namaAyah	varchar(50)	Nama ayah pasien
9	namaIbu	varchar(50)	Nama ibu pasien
10	hp	varchar(13)	Nomor *handphone* pasien
11	kodeVerifikasi	varchar(20)	Kode verifikasi yang didapat pasien saat melakukan pendaftaran untuk menjadi pasien tetap. Kode diambil dari tanggal, bulan, jam, menit, dan detik pasien melakukan pendaftaran
12	statusPendaftaran	varchar(20)	Status pendaftaran “sementara” untuk pasien yang belum memasukkan kode verifikasi dan “terverivikasi” untuk pasien yang telah memasukkan kode verifikasi
13	tanggalPendaftaran	timestamp	Tanggal pasien mendaftar menjadi pasien tetap

Tabel 5.6 merupakan struktur dari tabel pasien. Tabel pasien memiliki kolom idAkun yang merupakan *foreign key*, noPasien yang merupakan *primary key, username,* namaPasien, ttl, alamat, jenisKelamin, namaAyah, namaIbu, hp, kodeVerifikasi, statusPendaftaran, tanggalPendaftaran. Tabel pasien menyimpan data pribadi pasien ketika pasien melakukan pendaftaran untuk menjadi pasien tetap.

Tabel 5.7 Struktur Tabel antrian

No	Nama Field	Tipe Data	Keterangan
1	noAntrian (pk)	int(11)	Nomor antrian (1,2,3…) yang diberikan secara otomatis menggunakan fungsi *Auto Increment*
2	noPasien (fk)	int(5)	Nomor pasien yang mendaftar antrian
3	namaPasien	varchar(50)	Nama pasien yang mendaftar antrian
4	tanggalKonsultasi	date	Tanggal konsultasi
5	statusAntrian	varchar(20)	Status antrian “menunggu” untuk pasien yang masih menunggu, status antrian “konsultasi” untuk pasien yang sedang diperksa oleh dokter, status pasien “selesai” untuk pasien yang sudah melakukan konsultasi

Tabel 5.7 merupakan struktur dari tabel antrian. Tabel antrian memiliki kolom noAntrian yang merupakan *primary key,* noPasien yang merupakan *foreign key,* namaPasien, tanggalKonsultasi, dan statusAntrian. Tabel antrian menyimpan data pasien ketika pasien melakukan pendaftaran konsutlasi,

Tabel 5.8 Struktur Tabel jadwal_konsultasi_lanjut

No	Nama Field	Tipe Data	Keterangan
1	idJadwal (pk)	int(11)	ID jadwal (1,2,3…) yang diberikan secara otomatis menggunakan fungsi *Auto Increment*
2	noPasien (fk)	int(5)	Nomor pasien yang akan melakukan konsultasi lanjut
3	tanggalKKonsultasiLanjut	varchar(30)	Tanggal konsultasi lanjut
4	hp	varchar(13)	Nomor *handphone* pasien yang akan melakukan konsultasi lanjut
5	keterangan	text	Keterangan jadwal konsultasi lanjut

Tabel 5.8 merupakan struktur dari tabel jadwal_konsultasi_lanjut yang memiliki kolom idJadwal yang merupakan *primary key,* noPasien yang merupakan *foreign key,* tanggalKonsultasiLanjut, hp, dan keterangan. Tabel

jadwal_konsultasi_lanjut menyimpan data pasien dan jadwal konsultasi yang akan dilakukan.

Tabel 5.9 Struktur Tabel rekam_medis

No	Nama Field	Tipe Data	Keterangan
1	idRekamMedis (pk)	int(11)	ID rekam medis (1,2,3…) yang diberikan secara otomatis menggunakan fungsi *Auto Increment*
2	noPasien (fk)	int(5)	Nomor pasien yang diberikan rekam medis
3	elemen	text	Elemen gigi pasien yang diperiksa
4	diagnosa	text	Diagnosa penyakit pasien
5	terapi	text	Terapi yang diberikan dokter untuk pasien
6	tanggalRekamMedis	timestamp	Tanggal rekam medis dibuat

Tabel 5.9 merupakan struktur tabel rekam_medis yang terdiri dari kolom idRekamMedis yang merupakan *primary key,* noPasien yang merupakan *foreign key,* elemen, diagnose, terapi, dan tanggalRekamMedis. Tabel rekam_medis menyimpan data pasien dan rekam medis ketika dokter memberikan rekam medis kepada pasien.

Tabel 5.10 Struktur Tabel resep_obat

No	Nama Field	Tipe Data	Keterangan
1	idResep (pk)	int(11)	ID pesan (1,2,3…) yang diberikan secara otomatis menggunakan fungsi *Auto Increment*
2	noPasien (fk)	int(5)	Nomor pasien yang meminta resep obat
3	namaPasien	varchar(50)	Nama pasien yang meminta resep obat
4	resep	text	Resep obat yang diberikan dokter untuk pasien
5	hp	varchar(50)	Nomor *handphone* pasien yang meminta resep obat

Tabel 5.10 merupakan struktur tabel resep_obat yang terdiri dari kolom idResep yang merupakan *primary key,* noPasien yang merupakan *foreign key,* namaPasien, resep, dan hp. Tabel resep_obat menyimpan data resep ketika dokter menuliskan resep obat untuk pasien.

Tabel 5.11 Struktur Tabel pekerja_klinik

No	Nama Field	Tipe Data	Keterangan
1	idPekerja (pk)	int(11)	ID informasi (1,2,3…) yang diberikan secara otomatis menggunakan fungsi *Auto Increment*
2	idAkun (fk)	int(11)	ID akun pekerja klinik
3	username	varchar(20)	*Username* pekerja klinik
4	pass	varchar(50)	*Password* pekerja klinik yang tidak dienkripsi *md5*
5	namaPekerja	varchar(50)	Nama pekerja klinik
6	alamatPekerja	text	Alamat pekerja klinik
7	hpPekerja	varchar(13)	Nomor *handphone* pekerja klinik
8	profesi	varchar(20)	Profesi pekerja klinik

Tabel 5.11 merupakan struktur tabel pekerjaklinik yang terdiri dari kolom idPekerja yang merupakan *primary key,* idAkun yang merupakan *foreign key, username*, *pasword*, namaPekerja, alamatPekerja, hpPekerja, profesi. Tabel pekerjaklinik menyimpan data pekerja klinik yang bekerja pada klinik Drg. Damayanti

Tabel 5.12 Struktur Tabel konsultasi

No	Nama Field	Tipe Data	Keterangan
1	statusKlinik (pk)	varchar(5)	Status klinik yang berisi “Buka” atau “Tutup”
2	kuota	int(2)	Jumlah kuota konsultasi
3	idPekerja (fk)	int(11)	ID pekerja klinik yang mengubah status klinik

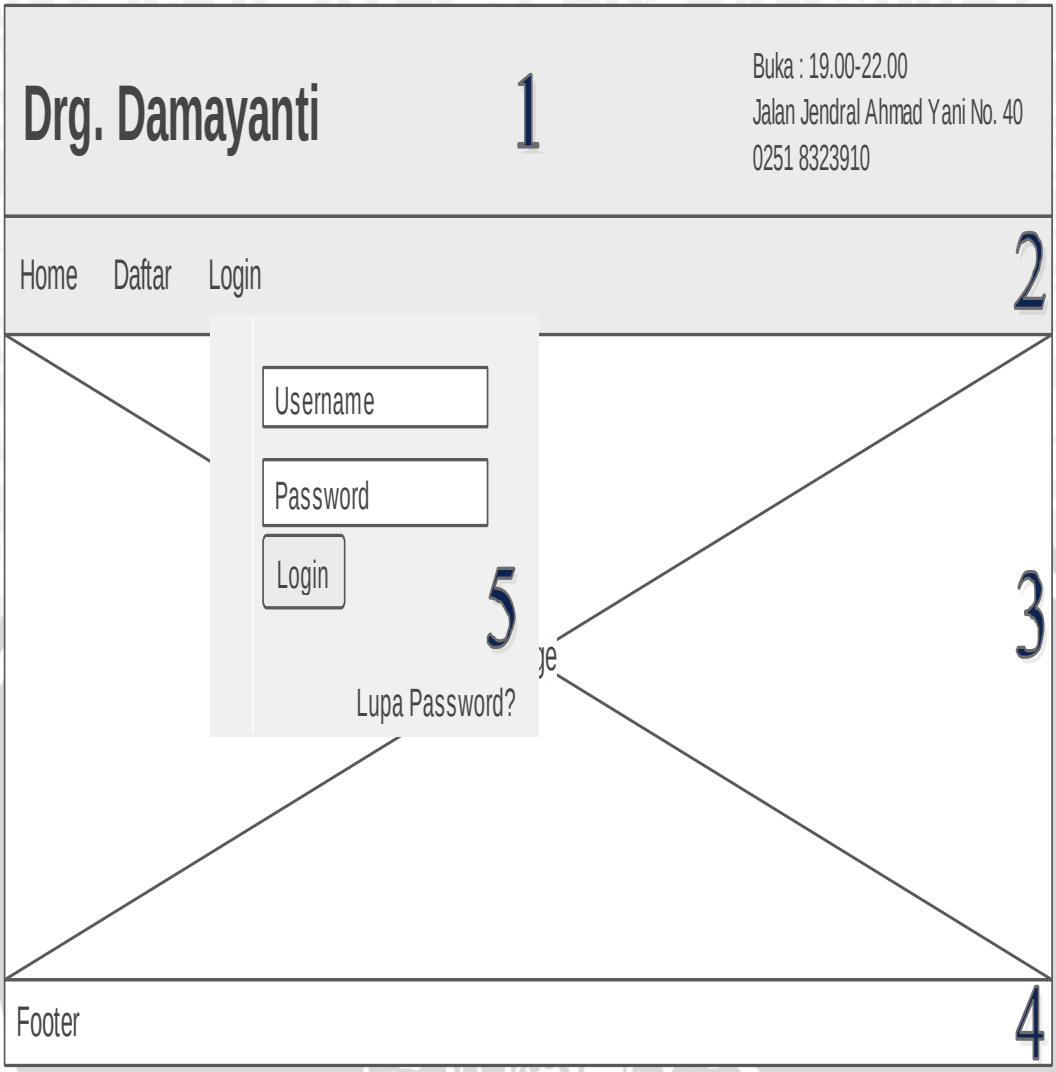
Tabel 5.12 merupakan struktur tabel konsultasi yang terdiri dari kolom statusKlinik yang merupakan *primary key,* idPekerja yang merupakan *foreign key,* dan kuota. Tabel konsultasi menyimpan informasi yang berkaitan dengan klinik.

5.1.4 Perancangan Antarmuka

Perancangan antarmuka merupakan perancangan tampilan halaman *web* pada sistem yang akan dibuat. Berikut adalah tampilan perancangan antarmuka sistem manajemen data pasien klinik gigi.

5.1.4.1 Halaman Awal

Halaman awal akan digambarkan pada Gambar 5.10 dan penjelasan detail dari halaman tersebut akan dijelaskan pada tabel 5.13.



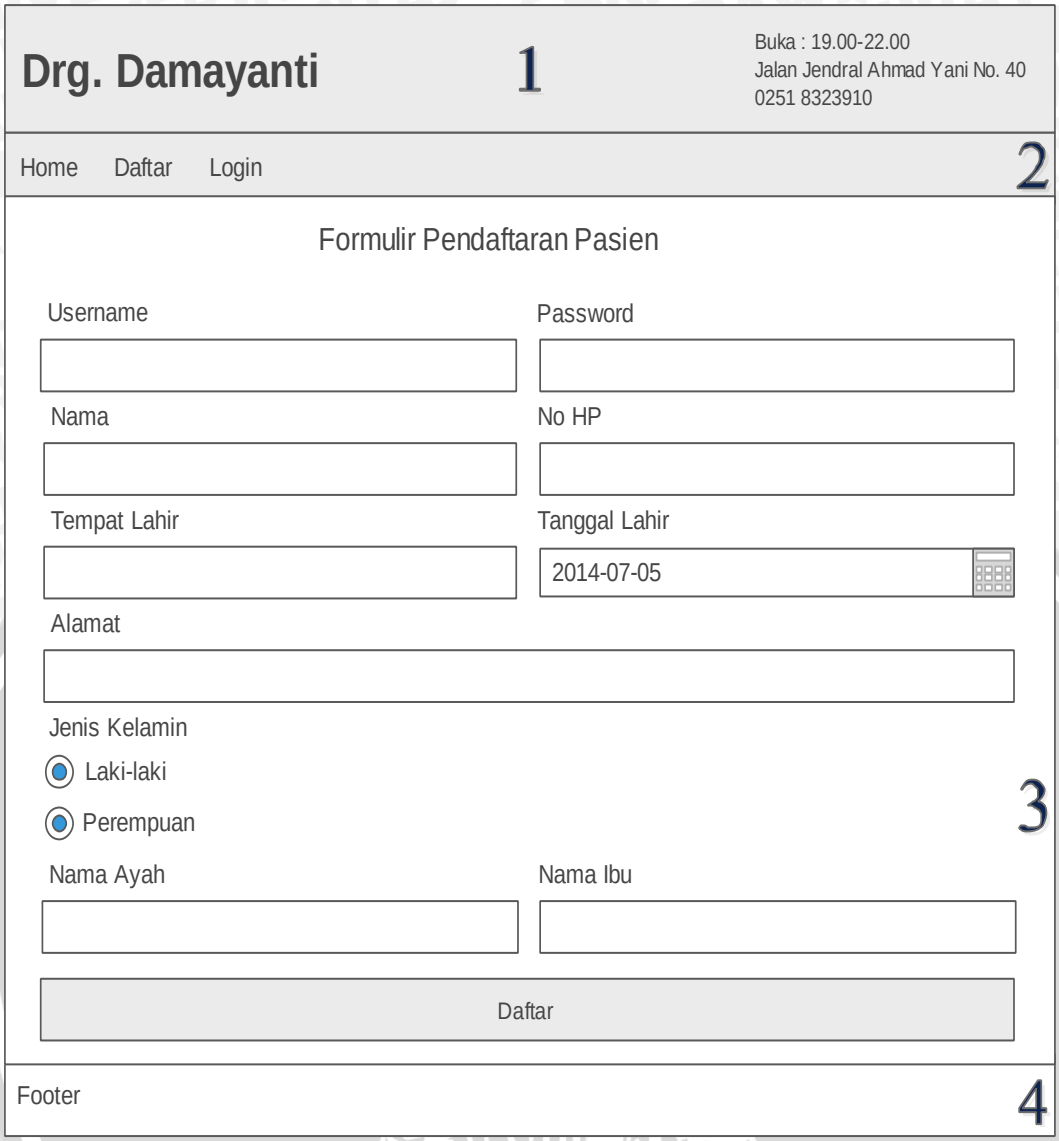
Gambar 5.10 Tampilan Antarmuka Halaman Awal

Tabel 5.13 Penjelasan Antarmuka Halaman Awal

No	Nama Objek	Keterangan
1	Header	Menampilkan nama klinik gigi, alamat, nomor telepon, dan jam buka klinik
2	Navbar	Menampilkan menu Home, Daftar, Login
3	Image	Menampilkan Foto yang berkaitan dengan klinik gigi
4	Footer	Menampilkan *footer website*
5	Formulir Login	Merupakan *dropdown* dari menu Login

5.1.4.2 Halaman Pendaftaran Pasien Tetap

Halaman pendaftaran pasien tetap akan digambarkan pada Gambar 5.11 dan penjelasan detail dari halaman tersebut akan dijelaskan pada tabel 5.14.



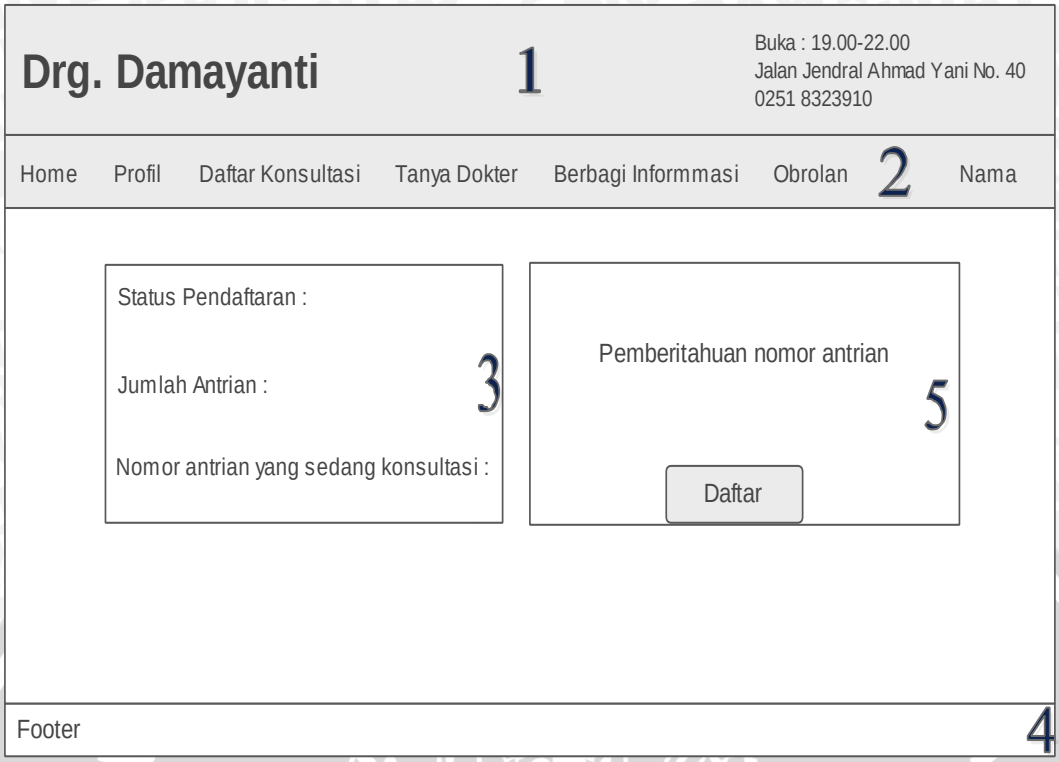
Gambar 5.11 Tampilan Antarmuka Halaman Pendaftaran Pasien Tetap

Tabel 5.14 Penjelasan Antarmuka Halaman Pendaftaran Pasien Tetap

No	Nama Objek	Keterangan
1	Header	Menampilkan nama klinik gigi, alamat, nomor telepon, dan jam buka klinik
2	Navbar	Menampilkan menu Home, Daftar, Login
3	Formulir	Menampilkan formulir untuk memasukkan data pribadi
4	Footer	Menampilkan *footer website*

5.1.4.3 Halaman Daftar Konsultasi

Halaman awal akan digambarkan pada Gambar 5.12 dan penjelasan detail dari halaman tersebut akan dijelaskan pada tabel 5.15.



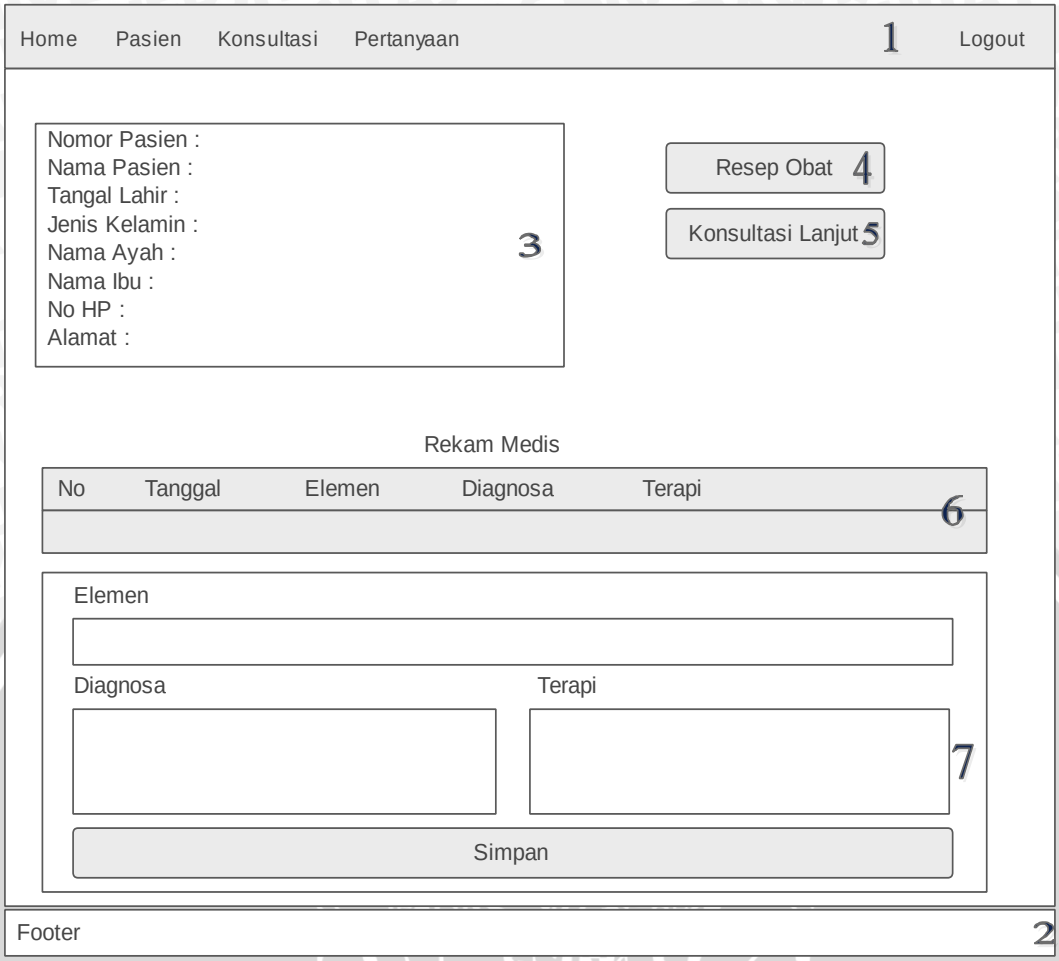
Gambar 5.12 Tampilan Antarmuka Halaman Daftar Konsultasi

Tabel 5.15 Penjelasan Antarmuka Halaman Daftar Konsultasi

No	Nama Objek	Keterangan
1	Header	Menampilkan nama klinik gigi, alamat, nomor telepon, dan jam buka klinik
2	Navbar	Menampilkan menu Home, Profil, Daftar Konsultasi, Tanya Dokter yang memiliki sub menu Tanya Dokter dan Jawaban Dokter, Berbagi Informasi yang memiliki sub menu Berbagi Informasi dan Informasi, Obrolan, Nama yang memiliki sub menu Logout
3	Status Pendaftaran	Menampilkan status pendaftaran konsultasi *online,* jumlah antrian yang mendaftar konsultasi, dan nomor antrian yang sedang diperiksa
4	Footer	Menampilkan *footer website*
5	Formulir Pendaftaran	Menampilkan nomor antrian yang didapat jika pasien mendaftar konsultasi

5.1.4.4 Halaman Kosultasi

Halaman awal akan digambarkan pada Gambar 5.13 dan penjelasan detail dari halaman tersebut akan dijelaskan pada tabel 5.16.



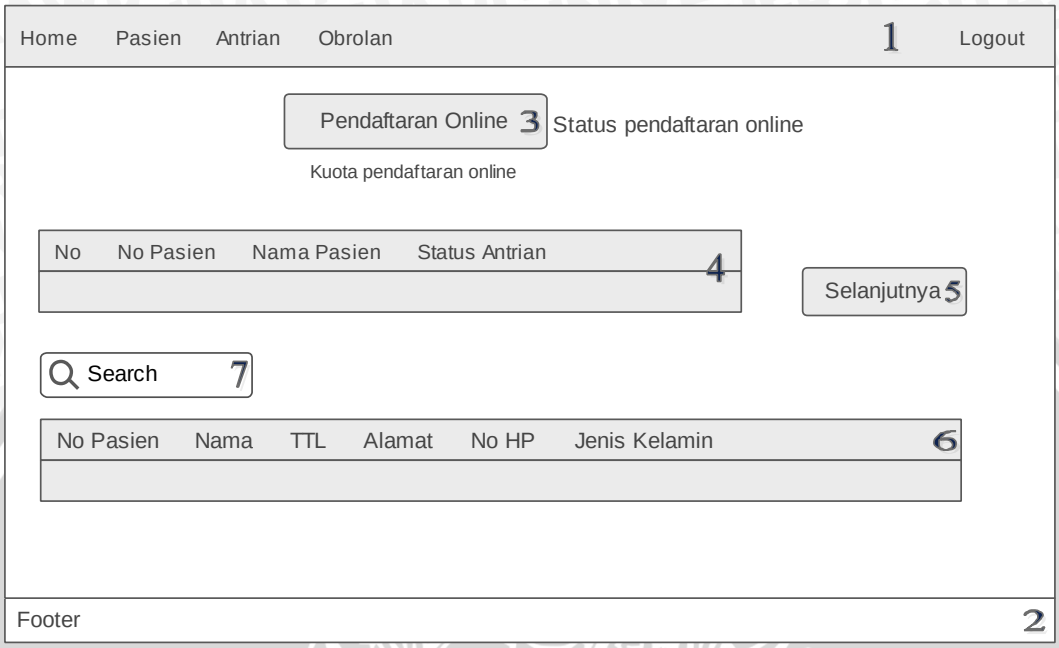
Gambar 5.13 Tampilan Antarmuka Halaman Konsutlasi

Tabel 5.16 Penjelasan Antarmuka Halaman Konsultasi

No	Nama Objek	Keterangan
1	Navbar	Menampilkan menu Home, Pasien, Konsultasi, Pertanyaan, dan Logout
2	Footer	Menampilkan *footer website*
3	Data Pasien	Menampilkan seluruh data pribadi pasien
4	Tombol Resep Obat	Merupakan tombol untuk menampilkan formulir untuk memasukkan resep obat ke dalam *database*
5	Tombol Konsultasi Lanjut	Merupakan tombol untuk menampilkan formulir untuk
6	Rekam Medis Pasien	Menampilkan rekam medis pasien yang sedang konsultasi dalam bentuk tabel
7	Formulir Rekam Medis	Merupakan formulir untuk memasukkan rekam medis ke dalam *database*

5.1.4.5 Halaman Antrian

Halaman awal akan digambarkan pada Gambar 5.14 dan penjelasan detail dari halaman tersebut akan dijelaskan pada tabel 5.17.



Gambar 5.14 Tampilan Antarmuka Halaman Antrian

Tabel 5.17 Penjelasan Antarmuka Halaman Antrian

No	Nama Objek	Keterangan
1	Navbar	Menampilkan menu Home, Pasien, Antrian, Obrolan, dan Logout
2	Footer	Menampilkan *footer website*
3	Tombol Pendaftaran Online	Merupakan tombol untuk menampilkan formulir untuk memasukkan kuota pendaftaran *online* ke dalam *database*
4	Daftar Antrian	Menampilkan daftar antrian dalam bentuk tabel
5	Tombol Selanjutnya	Merupakan tombol untuk memajukan antrian
6	Daftar Pasien	Menampilkan daftar pasien tetap dalam bentuk tabel
7	Kotak Pencarian	Merupakan kotak yang dapat mempercepat pencarian pasien

5.1.5 Pembahasan Hasil Perancangan

Dari tahap perancangan yang telah dilakukan, perancangan menghasilkan klas-klas, detail klas, algoritma, data model, dan antarmuka yang digunakan sebagai acuan untuk membuat sistem yang dapat mengelola data pasien. Klas-klas yang terbentuk adalah klas-klas *controller, entity,* dan *boundary.* Secara teori setiap *use case scenario* berkolerasi dengan satu *sequence* dengan satu *controller class.* Namun pada perancangan yang telah dilakukan, *controller class* yang terbentuk ada delapan *controller class* sedangkan terdapat 23 *use case.* Hal tersebut terjadi karena *controller class* dibuat berdasarkan objek yang teridentifikasi. Klas-klas dan algoritma nantinya harus diimplementasikan ke dalam struktur program sehingga dapat terbentuk sistem yang dapat membantu mengelola data pasien. Perancangan *database* menghasilkan perancangan model data secara konseptual dan fisik yang dijadikan sebagai acuan untuk membuat *database*. Perancangan antarmuka akan dijadikan sebagai acuan untuk membuat tampilan *web.*

5.2 Implementasi

Setelah perancangan selesai dilakukan tahapan selanjutnya adalah melakukan implementasi. Implementasi dilakukan berdasarkan hasil yang didapatkan dari analisis kebutuhan dan perancangan. Setiap kebutuhan akan diimplementasikan pada sistem. Tahap perancangan menghasilkan klas-klas dan algoritma. Klas-klas yang dihasilkan pada tahap perancangan harus dimasukkan ke dalam struktur program. Implementasi juga harus membuat *source code* berdasarkan algoritma yang sudah dibuat pada perancangan. Tahap Implementasi akan menjelaskan spesifikasi sistem, batasan implementasi, implementasi *Class*, implementasi kode program, dan implementasi antarmuka.

5.2.1 Spesifikasi Sistem

Proses implementasi dibantu oleh perangkat keras dan perangkat lunak yang dapat mewujudkan sistem manajemen data pasien klinik gigi ini. Sub bab di bawah ini akan menjelaskan spesifikasi dari perangkat lunak dan perangkat keras yang digunakan dalam membuat sistem manajemen data pasien klinik gigi ini.

5.2.1.1 Spesifikasi Perangkat Keras

Pembuatan sistem manajemen data pasien klinik gigi ini menggunakan komputer dengan spesifikasi yang dijelaskan pada tabel 5.18 di bawah ini

Tabel 5.18 Spesifikasi Perangkat Keras

Nama Komponen	Spesifikasi
System Model	Acer Aspire 4750
Processor	Intel® Core™ i3-2330M (2.2GHz, 3MB L3 cache)
Memory	2 GB DDR3

Display	Intel® HD Graphics
Hardisk	500 GB

5.2.1.2 Spesifikasi Perangkat Lunak

Pembuatan sistem manajemen data pasien klinik gigi ini menggunakan perangkat lunak dengan spesifikasi yang dijelaskan pada tabel 5.19 di bawah ini.

Tabel 5.19 Spesifikasi Perangkat Lunak

Nama Komponen	Spesifikasi
Operating System	Windows 7 Ultimate 64-bit
Programming Language	PHP, JavaScript
Programming Environment	Notpad++
Database Management System	MySQL 5.6.3

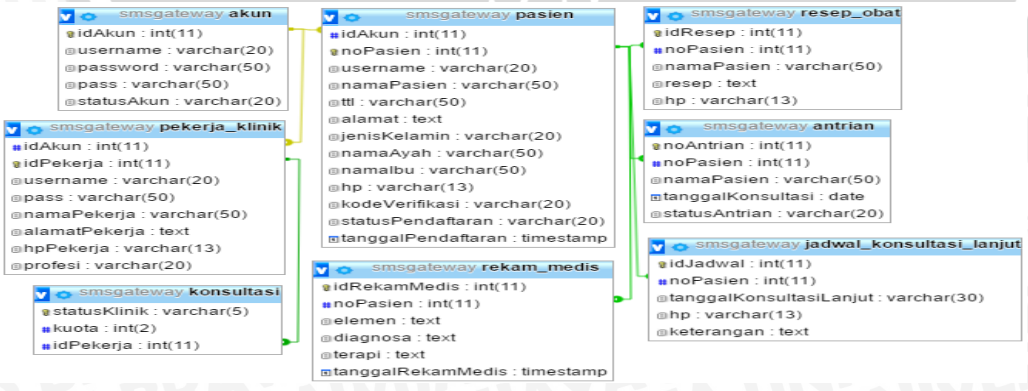
5.2.2 Batasan Implementasi

Terdapat batasan-batasan pada proses implementasi sistem manajemen data pasien klinik gigi ini yaitu sebagai berikut:

1. Aplikasi dibangun berbasis *web* menggunakan pahasa pemrograman *PHP* dan *JavaScript*
2. *Framework* yang digunakan adalah *CodeIgniter*
3. *Database* dibangun menggunakan *MySQL*
4. Pembuatan *layout* pada antarmuka aplikasi menggunakan *HTML, CSS,* dan *JavaScript.*

5.2.3 Implementasi Basis Data

Implementasi basis data dibuat berdasarkan perancangan *database* secara fisik atau *physical data model* yang sudah dibuat pada bab perancangan. Implementasi basis data digambarkan pada Gambar 5.15 seperti di bawah ini.



Gambar 5.15 Implementasi Database

Gambar 5.15 menggambarkan tabel-tabel yang telah dibuat. Terdapat sebelas tabel yang dibuat sesuai dengan yang ada pada perancangan *database.* Diantaranya adalah resep_obat, rekam_medis, jadwal_konsultasi_lanjut, pasien, akun, pesan, konsultasi, dan antrian. Pada gambar tersebut terlihat terdapat tabel yang berelasi dan juga tabel yang tidak berelasi. Tabel yang berelasi antara lain: tabel pasien yang memiliki *primary key* yaitu noPasien berelasi dengan tabel resep_obat, rekam_medis, jadwal_konsultasi_lanjut, dan antrian, sehingga tabel-tabel tersebut memiliki *foreign key* yaitu noPasien. Selain itu tabel akun yang memiliki *primary key* yaitu idAkun berelasi dengan tabel pasien, dan pekerja_klinik sehingga tabel-tabel tersebut memiliki *foregin key* yaitu idAkun.

5.2.4 Implementasi Klas

Proses implementasi klas dilakukan berdasarkan perancangan *class diagram*. Klas-klas yang ada pada *class diagram* akan direalisasikan ke dalam bentuk *file* dengan format (.php). Hasil implementasi akan ditunjukkan pada Tabel 5.20 berikut.

Tabel 5.20 Implementasi *Class Diagram*

No	Package	Nama *Class*	Nama File
1	Controller	C_Akun	C_Akun.php
2	Controller	C_Antrian	C_Antrian.php
3	Controller	C_JadwalKonsultasiLanjut	C_JadwalKonsultasiLanjut.php
4	Controller	C_Konsultasi	C_Konsultasi.php
5	Controller	C_Pasien	C_Pasien.php
6	Controller	C_RekamMedis	C_RekamMedis.php
7	Controller	C_ResepObat	C_ResepObat.php
8	Controller	C_PekerjaKlinik	C_PekerjaKlinik.php
9	Model	E_Akun	E_Akun.php
10	Model	E_Antrian	E_Antrian.php
11	Model	E_JadwalKonsultasiLanjut	E_JadwalKonsultasiLanjut.php
12	Model	E_Konsultasi	E_Konsultasi.php
13	Model	E_Outbox	E_Outbox.php
14	Model	E_Pasien	E_Pasien.php
15	Model	E_RekamMedis	E_RekamMedis.php
16	Model	E_ResepObat	E_ResepObat.php
17	Model	E_PekerjaKlinik	E_PekerjaKlinik.php
18	Views	Index	index.php

Tabel 5.20 (lanjutan)

No	Package	Nama *Class*	Nama File
19	Views	LupaPassword	lupaPassword.php
20	Views	FormulirPendaftaran	formulirPendaftaran.php
21	Views	FormulirUbahDataPasien	formulirUbahDataPasien.php
22	Views	DaftarPasien	daftarPasien.php
23	Views	Profil	profil.php
24	Views	Antrian	antrian.php
25	Views	DaftarKonsultasi	daftarKonsultasi.php
26	Views	FormulirTambahPekerjaKlinik	formulirTambahPekerjaKlinik.php
27	Views	FormulirUbahPekerjaKlinik	formulirUbahPekerjaKlinik.php
28	Views	RekamMedisPasien	rekamMedisPasien.php
29	Views	Konsultasi	konsultasi.php
30	Views	DetailResep	detailResep.php
31	Views	ResepObat	resepObat.php
32	Views	Home	home.php

5.2.5 Implementasi Kode Program

Proses implementasi kode program dilakukan berdasarkan perancangan algoritma. Tahap implementasi ini akan menunjukan perubahan algoritma menjadi kode program atau *source code* yang dapat dimengerti oleh komputer. Algoritma-algoritma yang telah dibuat akan diubah menjadi kode program menggunakan Bahasa pemrograman *PHP.* Implementasi kode program akan ditunjukkan pada beberapa sampel operasi yaitu operasi login() pada klas C_Akun, operasi cekKodeVerifiaksi() pada klas C_Akun, operasi tambahAntrin() pada klas C_Antrian, dan operasi tambahRekamMedis pada klas C_RekamMedis.

Nama Klas : C_Akun

Nama Operasi : login

Kode Program :

```
1  $username = $this->input->post('username');
2  $password = $this->input->post('password');
3  $password = md5($password);
4  $result   =   $this->E_Akun->validasiLogin($username,
5  $password);
6  if($result == "valid"){
```

```
  $data = array();
  $result  =  $this->E_Akun->ambilDataAkun($username,
  $password);
  foreach($result as $row) {
    $data['idAkun'] = $row->idAkun;
    $data['username'] = $row->username;
    $data['statusAkun'] = $row->statusAkun;
  }
  if($data['statusAkun'] == 'pasien'){
    $idAkun = $data['idAkun'];
    $result            =            $this->E_Pasien-
     >ambilNoPasien($idAkun);
    foreach($result as $row) {
      $data['noPasien'] = $row->noPasien;
      $data['namaPasien'] = $row->namaPasien;
      $data['hp'] = $row->hp;
      $data['statusPendaftaran']        =        $row-
       >statusPendaftaran;
    }
    $this->session->set_userdata('logged_in',
     $data);
  }
  else   if($data['statusAkun']   ==   'perawat'   ||
   $data['statusAkun']     ==       'dokter'       ||
   $data['statusAkun'] == 'apoteker'){
    $this->session->set_userdata('logged_in',
     $data);
  }
  else if($data['statusAkun'] == 'admin'){
    $this->session->set_userdata('logged_in',
     $data);
    $data['result']    =     $this->E_PekerjaKlinik-
     >ambilSeluruhPekerjaKlinik();
  }
  $this->load->view('home', $data);
}
else if($result == "password false"){
  $data['login'] = "password";
  $this->load->view('index',$data);
}
else if($result == "username false"){
  $data['login'] = "username";
  $this->load->view('index',$data);
}
else if($result == "username and password false"){
  $data['login'] = "false";
  $this->load->view('index',$data);
}
```

Nama Klas : C_Pasien

Nama Operasi : cekKodeVerifikasi

Kode Program :

```
1  date_default_timezone_set("Asia/Jakarta");
2  $session_data = $this->session-
3  >userdata('logged_in');
4  $idAkun = $session_data['idAkun'];
5  $kode = $this->input->post('kodeVerifikasi');
6  $username = $this->input->post('username');
7  $result = $this->E_Pasien-
8  >cekKodeVerifikasi($username,$kode);
9  if($result){
10   $tanggalPendaftaran = "";
11   foreach($result as $row){
12     $tanggalPendaftaran = $row->tanggalPendaftaran;
13   }
14   $a = date("Y-m-d h:i:s");
15   $b = date("Y-m-d h:i:s",
16    strtotime('.$tanggalPendaftaran.' +1 hours'));
17   if($a<$b){
18     $this->E_Pasien->ubahStatusPasien($username);
19     $result = $this->E_Pasien-
20      >ambilNoPasien($idAkun);
21     foreach($result as $row){
22       $session_data = array(
23       'idAkun' => $row->idAkun,
24       'username' => $row->username,
25       'statusAkun'=> 'pasien',
26       'noPasien' => $row->noPasien,
27       'namaPasien' => $row->namaPasien,
28       'hp'=>$row->hp,
29       'statusPendaftaran' => $row-
30        >statusPendaftaran);
31     }
32     $this->session->unset_userdata('logged_in');
33     $this->session->set_userdata('logged_in',
34      $session_data);
35     redirect('C_Pasien', 'refresh');
36   }
37   else{
38     $this->E_Pasien->hapusPasien($idAkun);
39     $this->E_Akun->hapusAkun($idAkun);
40     $data['verifikasi']= "timeout";
41     $this->load->view('index', $data);
42   }
43 }
44 else{
45   $session_data['verifikasi'] = 'false';
46   $this->load->view('pasien/home',$session_data);
47 }
```

Nama Klas : C_Antrian

Nama Operasi : tambahAntrian

Kode Program :

```
1  date_default_timezone_set("Asia/Jakarta");
```

```
2  $session_data = $this->session-
3  >userdata('logged_in');
4  $statusAkun = $session_data['statusAkun'];
5  $tanggal = date("Y-m-d");
6  if($statusAkun == 'pasien'){
7    $noPasien = $session_data['noPasien'];
8    $namaPasien = $session_data['namaPasien'];
9    $hp = $session_data['hp'];
10   $jam = date("H:i");
11   $result = $this->E_Konsultasi->ambilStatus();
12   foreach($result as $row){
13     $kuota = $row->kuota;
14   }
15   if($jam <= "21:00" && $kuota > 0){
16     $result = $this->E_Antrian-
17     >tambahAntrian($noPasien, $namaPasien,
18     $tanggal);
19     foreach($result as $row){
20       $noAntrian = $row->noAntrian;
21     }
22     $pesan = "Anda sudah melakukan pendaftaran
23      konsultasi pada Klinik Drg. Damayanti. Anda
24      mendapat nomor antrian ".$noAntrian.".";
25     $this->E_Outbox->kirimPesan($hp, $pesan);
26     $kuota = $kuota-1;
27     $this->E_Konsultasi->ubahKuota($kuota);
28     $session_data['terdaftar'] = "true";
29     $session_data['noAntrian'] = $noAntrian;
30     $session_data['statusKlinik'] = 'Buka';
31     $this->load->view('pasien/daftarKonsultasi',
32      $session_data);
33   }
34   else if ($jam>"21:00"){
35     $noAntrian = 0;
36     $session_data['terdaftar'] = "false";
37     $session_data['noAntrian'] = $noAntrian;
38     $session_data['statusKlinik'] = 'Tutup';
39     $session_data['error'] = 'jam';
40     $this->load->view('pasien/daftarKonsultasi',
41      $session_data);
42   }
43   else if ($kuota == 0){
44     $noAntrian = 0;
45     $session_data['terdaftar'] = "false";
46     $session_data['noAntrian'] = $noAntrian;
47     $session_data['statusKlinik'] = 'Tutup';
48     $session_data['error'] = 'kuota';
49     $this->load->view('pasien/daftarKonsultasi',
50     $session_data);
51   }
52 }
53 else{
54   $noPasien = $this->input->post('noPasien');
```

```
55    $namaPasien = $this->input->post('namaPasien');
56    $hp = $this->input->post('hp');
57    $result = $this->E_Antrian-
58     >tambahAntrian($noPasien, $namaPasien, $tanggal);
59    foreach($result as $row){
60      $noAntrian = $row->noAntrian;
61    }
62    $pesan = "Anda sudah melakukan pendaftaran
63     konsultasi pada Klinik Drg. Damayanti. Anda
64     mendapat nomor antrian ".$noAntrian.".";
65    $this->E_Outbox->kirimPesan($hp, $pesan);
66    $error = 'kosong';
67    redirect('C_Antrian/index/'.$error.'');
68  }
```

Nama Klas : C_RekamMedis

Nama Operasi : tambahRekamMedis

Kode Program :

```
1   $noPasien = $this->input->post('noPasien');
2   $elemen = $this->input->post('elemen');
3   $diagnosa = $this->input->post('diagnosa');
4   $terapi = $this->input->post('terapi');
5   if($noPasien == 0){$session_data = $this->session-
6   >userdata('logged_in');
7   $session_data['kesalahan'] = 'true';
8   $this->load->view('dokter/konsultasi',
9   $session_data);
10  }
11  else{$this->E_RekamMedis-
12  >tambahRekamMedis($noPasien, $elemen, $diagnosa,
13  $terapi);
14  redirect('C_Konsultasi');
15  }
```

5.2.6 Implementasi Antarmuka

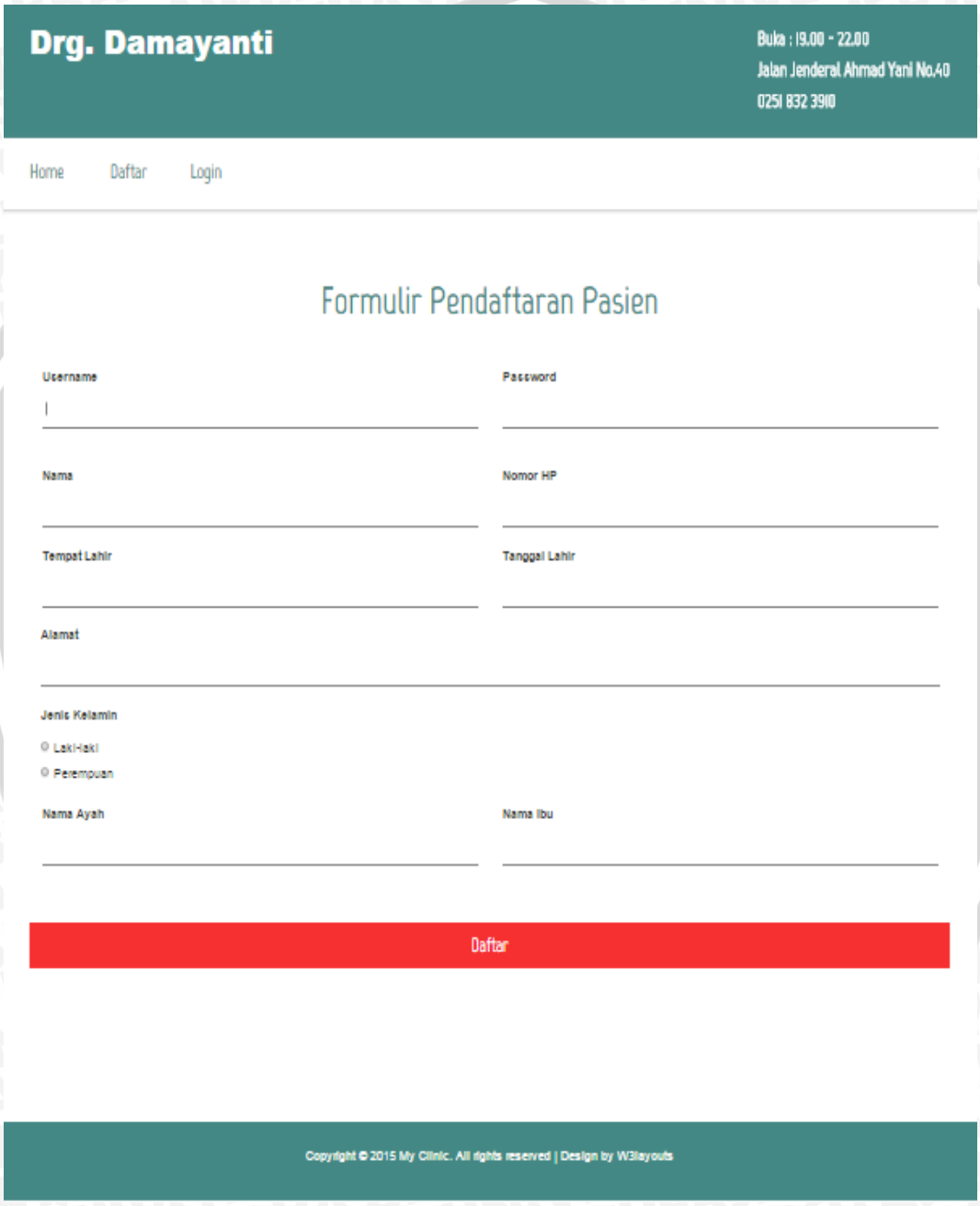
Implementasi antarmuka dilakukan berdasarkan perancangan antarmuka sebelumnya. Berikut adalah halaman antarmuka yang terdapat pada sistem manajemen data pasien klinik gigi ini.

5.2.6.1 Implementasi Halaman Awal

Gambar 5.16 Implementasi Antarmuka Halaman Awal

Gambar 5.16 merupakan tampilan halaman awal sistem. Tampilan antarmuka halaman awal dibuat berdasarkan hasil perancangan antarmuka halaman awal pada Gambar 5.31. Halaman awal merupakan halaman yang pertama kali ditampilkan saat tamu mengakses sistem manajemen data pasien klinik gigi.

5.2.6.2 Implementasi Halaman Pendaftaran Pasien Tetap



Gambar 5.17 Implementasi Antarmuka Halaman Pendaftaran Pasien Tetap

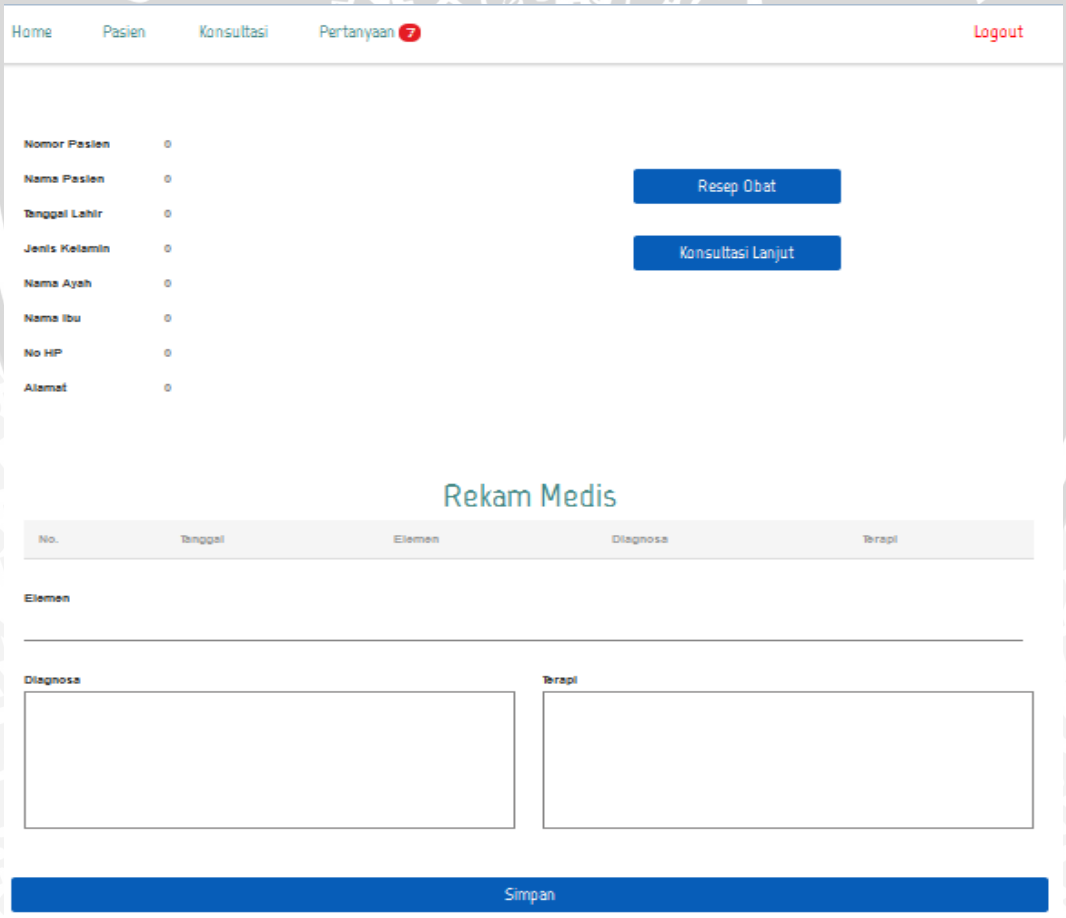
Gambar 5.17 merupakan tampilan halaman untuk mendaftar menjadi pasien tetap. Tampilan antarmuka halaman pendaftaran pasien tetap dibuat berdasarkan hasil perancangan antarmuka pada Gambar 5.32.

5.2.6.3 Implementasi Halaman Daftar Konsultasi

Gambar 5.18 Implementasi Halaman Daftar Konsultasi

Gambar 5.18 merupakan implementasi dari perancangan antarmuka halaman daftar konsultasi pada Gambar 5.36. Halaman daftar konsultasi merupakan halaman yang digunakan pasien untuk mendaftar konsultasi secara *online.*

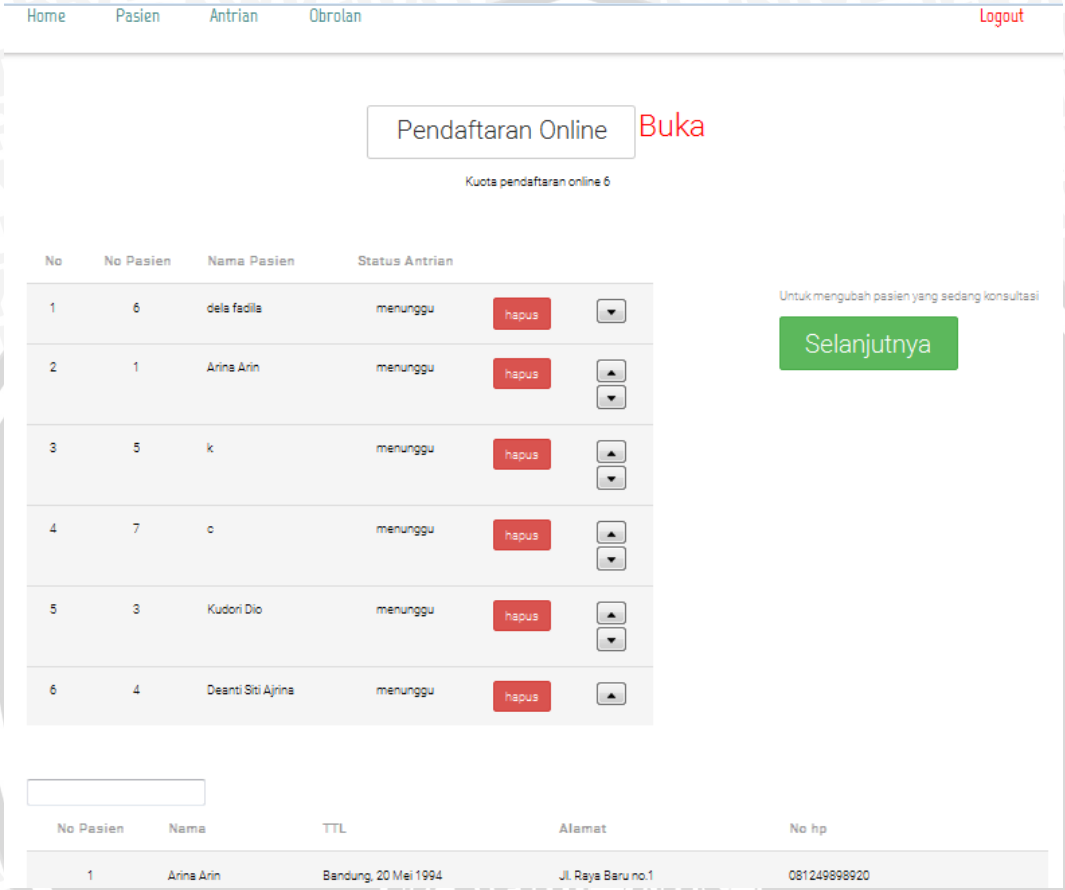
5.2.6.4 Implementasi Halaman Konsultasi



Gambar 5.19 Implementasi Halaman Konsultasi

Gambar 5.19 merupakan implementasi antarmuka berdasarkan perancangan antarmuka halaman konsultasi pada Gambar 5.47. Halaman konsultasi merupakan halaman yang menampilkan data pribadi dan rekam medis pasien yang sedang konsultasi beserta menampilkan formulir untuk memasukkan rekam medis, resep obat, dan jadwal konsultasi lanjut.

5.2.6.5 Implementasi Halaman Antrian



Gambar 5.20 Implementasi Halaman Antrian

Gambar 5.20 merupakan implementasi antarmuka berdasarkan perancangan antarmuka halaman antrian yang ada pada Gambar 5.53. Halaman antrian ini merupakan halaman yang digunakan oleh perawat untuk mengelola antrian.

5.2.7 Pembahasan Hasil Implementasi

Dari tahap implementasi yang telah dilakukan, implementasi menghasilkan sistem yang dapat membantu mengelola data pasien. Implementasi menghasilkan fitur yang dapat membantu mengelola data pasien yaitu fitur mendaftar konsultasi, mengelola rekam medis, mengelola resep obat, dan mengelola antrian.

BAB 6 PENGUJIAN

Tahap pengujian dilakukan setelah implementasi selesai dilakukan. Pengujian dilakukan untuk memeriksa apakah hasil implementasi sesuai dengan analisis kebutuhan dan perancangan atau tidak. Pengujian yang akan dilakukan adalah pengujian unit, pengujian integrasi, dan pengujian validasi.

6.1 Pengujian Unit

Pengujian Unit digunakan untuk menguji setiap klas yang dihasilkan dari proses perancangan. Pengujian unit dilakukan dengan menggunakan *basis path testing* yang merupakan salah satu jenis pengujian dari *white box testing.* Pengujian unit akan dilakukan pada tiga sampel uji yaitu klas C_Akun, klas E_Pasien, dan klas E_Akun.

6.1.1 Pengujian Unit Klas C_Akun

a. Operasi index()

Basis Path Testing

- Flow Graph

- Cyclomatic Complexity
  - V(G) = Jumlah Region = 1
  - V(G) = E – N + 2 = 0 – 1 + 2 = 1
  - V(G) = P + 1 = 0 + 1 = 1
- Independent path
  - Jalur 1 = 1

Test case dan hasil akan dijelaskan pada Tabel 6.1 dibawah ini.

Tabel 6.1 Hasil Pengujian Unit Klas C_Akun Operasi index()

No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil operasi index()	Menampilkan halaman *index*	Menampilkan halaman *index*	Valid

b. Operasi home()

Basis Path Testing

- Flow Graph

- Cyclomatic Complexity
  - V(G) = Jumlah Region = 1
  - V(G) = E – N + 2 = 0 – 1 + 2 = 1
  - V(G) = P + 1 = 0 + 1 = 1
- Independent path
  - Jalur 1 = 1

Test case dan hasil akan dijelaskan pada Tabel 6.2 dibawah ini.

Tabel 6.2 Hasil Pengujian Unit Klas C_Akun Operasi home()

No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil operasi home()	Menampilkan halaman home	Menampilkan halaman home	Valid

c. Operasi lupaPassword()

Basis Path Testing

- Flow Graph

- Cyclomatic Complexity
  - V(G) = Jumlah Region = 1
  - V(G) = E – N + 2 = 0 – 1 + 2 = 1
  - V(G) = P + 1 = 0 + 1 = 1
- Independent path
  - Jalur 1 = 1

Test case dan hasil akan dijelaskan pada Tabel 6.3 dibawah ini.

Tabel 6.3 Hasil Pengujian Unit Klas C_Akun Operasi lupaPassword()

No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil operasi lupaPassword()	Menampilkan halaman lupa password	Menampilkan halaman lupa password	Valid

d. Operasi login()

```
password = enkripsi md5 password
result = memeriksa apakah username dan password valid                    (1)
if(result = "valid") then                                                (2)
    data = array[] of String
    result = data akun berdasarkan username dan password
    data[idAkun] = result[idAkun]                                        (3)
    data[username] = result[username]
    data[statusAkun] = result[statusAkun]
    if(data[statusAkun] = "pasien") then                                 (4)
        result = mengambil data pasien dari database berdasarkan
        username dan password
        data[noPasien] = result[noPasien]
        data[namaPasien] = result[namaPasien]                            (5)
        data[hp] = result[hp]
        data[statusPendaftaran] = result[StatusPendaftaran]
        set session data
    else if(data[statusAkun] ="dokter" atau "perawat" atau               (6)
        "apoteker") then
        set session data                                                 (7)
    else if(data[statusAkun] ="admin") then                              (8)
        set session data
        mengambil seluruh data pengguna yang dapat mengakses             (9)
        sistem
    end if
    menampilkan halaman awal pengguna                                    (10)
else if(result = "password false") then                                  (11)
    menampilkan pesan kesalahan pada halaman login                       (12)
else if(result = "username false") then                                  (13)
    menampilkan pesan kesalahan pada halaman login                       (14)
```

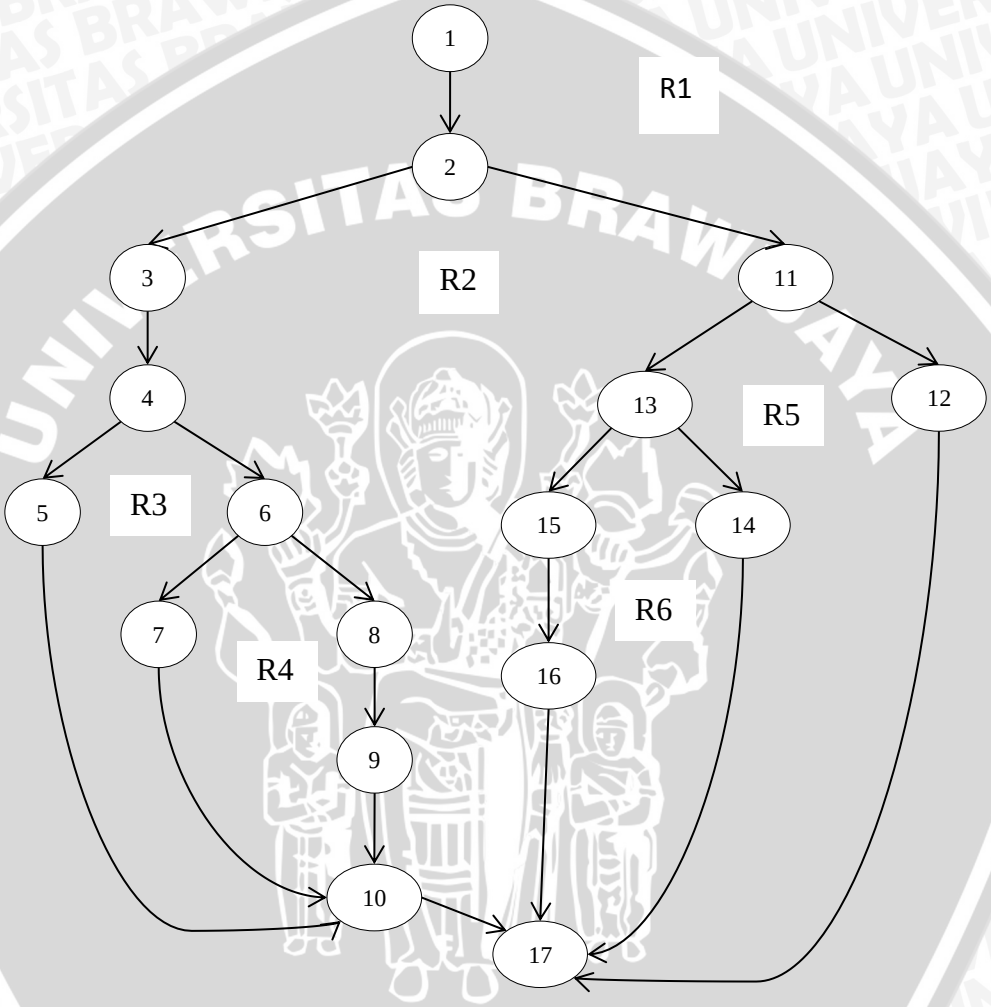
else if(result = "username and password false")then — 15

menampilkan pesan kesalahan pada halaman login — 16

end if — 17

Basis Path Testing

- Flow Graph



- Cyclomatic Complexity
  - V(G) = Jumlah Region = 6
  - V(G) = E – N + 2 = 21 – 17 + 2 = 6
  - V(G) = P + 1 = 5 + 1 = 6
- Independent path
  - Jalur 1 = 1 – 2 – 3 – 4 – 5 – 10 – 17
  - Jalur 2 = 1 – 2 – 3 – 4 – 6 – 7 – 10 – 17
  - Jalur 3 = 1 – 2 – 3 – 4 – 6 – 8 – 9 – 10 – 17

- Jalur 4 = 1 – 2 – 11 – 12 – 17
- Jalur 5 = 1 – 2 – 11 – 13 – 14 - 17
- Jalur 6 = 1 – 2 – 11 – 13 – 15 – 16 - 17

Test case dan hasil akan dijelaskan pada Tabel 6.4 dibawah ini.

Tabel 6.4 Hasil Pengujian Unit Klas C_Akun Operasi login()

No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil operasi login() dengan variabel *username* = "dea" dan *password* = "dea"	Berhasil *login* sebagai pasien. Menampilkan halaman home	Berhasil *login* sebagai pasien. Menampilkan halaman home	Valid
2.	2	Memanggil operasi login() dengan variabel u*sername* = "tiana" dan p*assword* = "tiana"	Berhasil *login* sebagai dokter. Menampilkan halaman home	Berhasil *login* sebagai dokter. Menampilkan halaman home	Valid
3.	2	Memanggil operasi login() dengan variabel u*sername* = "deanti" dan *Password* = "deanti"	Berhasil *login* sebagai perawat. Menampilkan halaman home	Berhasil *login* sebagai perawat. Menampilkan halaman home	Valid
4.	2	Memanggil operasi login() dengan variabel u*sername* = "apoteker" dan *password* = "apoteker"	Berhasil *login* sebagai apoteker. Menampilkan halaman home	Berhasil *login* sebagai apoteker. Menampilkan halaman home	Valid
5.	3	Memanggil operasi login() dengan variabel *username* = "admin" dan *password* = "admin"	Berhasil *login* sebagai admin. Menampilkan halaman home	Berhasil *login* sebagai admin. Menampilkan halaman home	Valid
6.	4	Memanggil	Gagal *login*.	Gagal *login*.	Valid

		operasi login() dengan variabel *username* = "dea" dan *password* = "abcd"	Menampilkan pesan kesalahan "Password salah!" pada halaman index	Menampilkan pesan kesalahan "Password salah!" pada halaman index	
7.	5	Memanggil operasi login() dengan variabel *username* = "abcd" dan *password* = "dea"	Gagal *login*. Menampilkan pesan kesalahan "Username salah!" pada halaman index	Gagal *login*. Menampilkan pesan kesalahan "Username salah!" pada halaman index	Valid
8.	6	Memanggil operasi login() dengan variabel *username* = "abcd" dan *password* = "abcd"	Gagal *login*. Menampilkan pesan kesalahan "Username dan password salah!" pada halaman index	Gagal *login*. Menampilkan pesan kesalahan "Username dan password salah!" pada halaman index	Valid

e. Operasi logout()

Menghapus session

Menampilkan halaman index

1

Basis Path Testing

- Flow Graph
- Cyclomatic Complexity
  - V(G) = Jumlah Region = 1
  - V(G) = E – N + 2 = 0 – 1 + 2 = 1
  - V(G) = P + 1 = 0 + 1 = 1
- Independent path
  - Jalur 1 = 1

Test case dan hasil akan dijelaskan pada Tabel 6.5 dibawah ini.

Tabel 6.5 Hasil Pengujian Unit Klas C_Akun Operasi logout()

No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil	Session	Session	Valid

		operasi logout()	terhapus, lalu menampilkan halaman index	terhapus, lalu menampilkan halaman index	

6.1.2 Pengujian Unit Klas C_PekerjaKlinik

a. Operasi index()

data = data login

menampilkan seluruh data pekerja klinik pada halaman home — 1

Basis Path Testing

- Flow Graph

- Cyclomatic Complexity
  - V(G) = Jumlah Region = 1
  - V(G) = E – N + 2 = 0 – 1 + 2 = 1
  - V(G) = P + 1 = 0 + 1 = 1
- Independent path
  - Jalur 1 = 1

Test case dan hasil akan dijelaskan pada Tabel 6.6 dibawah ini.

Tabel 6.6 Hasil Pengujian Unit Klas C_PekerjaKlinik Operasi index()

No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil operasi index()	Menampilkan seluruh data pekerja klinik pada halaman home	Menampilkan seluruh data pekerja klinik pada halaman home	Valid

b. Operasi formulirTambahPekerjaKlinik()

menampilkan formulir tambah pekerja klinik — 1

Basis Path Testing

- Flow Graph

- Cyclomatic Complexity
  - V(G) = Jumlah Region = 1

- V(G) = E – N + 2 = 0 – 1 + 2 = 1
- V(G) = P + 1 = 0 + 1 = 1

- Independent path
  - Jalur 1 = 1

Test case dan hasil akan dijelaskan pada Tabel 6.7 dibawah ini.

Tabel 6.7 Hasil Pengujian Unit Klas C_PekerjaKlinik Operasi formulirTambahPekerjaKlinik()

No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil operasi formulirTambah PekerjaKlinik()	Menampilkan formulir tambah pekerja klinik	Menampilkan formulir tambah pekerja klinik	Valid

c. Operasi formulirUbahPekerjaKlinik()

```
data = data session
data[result] = data pekerja klinik yang akan diubah
kirim data ke formulir ubah pekerja klinik
```

1

Basis Path Testing

- Flow Graph

- Cyclomatic Complexity
  - V(G) = Jumlah Region = 1
  - V(G) = E – N + 2 = 0 – 1 + 2 = 1
  - V(G) = P + 1 = 0 + 1 = 1
- Independent path
  - Jalur 1 = 1

Test case dan hasil akan dijelaskan pada Tabel 6.8 dibawah ini.

Tabel 6.8 Hasil Pengujian Unit Klas C_PekerjaKlinik Operasi formulirUbahPekerjaKlinik()

No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil operasi formulirUbahPek erjaKlinik()	Menmapilkan formulir ubah pekerja klinik	Menmapilkan formulir ubah pekerja klinik	Valid

d. Operasi tambahPekerjaKlinik()

password = enkripsi md5 passoword
memasukkan usename, password, pass, statusAkun ke dalam tabel akun
idAkun = idAkun dari username, password, pass, statusAkun yang dimasukkan ke dalam tabel akun
memasukkan idAkun, username, pass, nama, alamat, hp, statusAkun ke dalam tabel pekerja_klinik
menampilkan halaman home

1

Basis Path Testing

- Flow Graph

- Cyclomatic Complexity
  - V(G) = Jumlah Region = 1
  - V(G) = E – N + 2 = 0 – 1 + 2 = 1
  - V(G) = P + 1 = 0 + 1 = 1
- Independent path
  - Jalur 1 = 1

Test case dan hasil akan dijelaskan pada Tabel 6.9 dibawah ini.

Tabel 6.9 Hasil Pengujian Unit Klas C_PekerjaKlinik Operasi tambahPekerjaKlinik()

No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil operasi tambahPekerjaKlinik() dengan variabel nama = “Tiana Siti”, username = “tian”, pass = “tian”, password = “tian”, profesi = “dokter”, alamat = “Permata	Berhasil memasukkan data pekerja klinik ke dalam *database,* lalu menampilkan halaman home	Berhasil memasukkan data pekerja klinik ke dalam *database,* lalu menampilkan halaman home	Valid

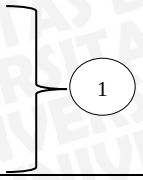
		Cimanggu", hp = "0812498989"			

e. Operasi hapusPekerjaKlinik()

menghapus data idPekerja dari tabel pekerja_klinik

menghapus data idAkun dari tabel akun

menampilkan halaman home



Basis Path Testing

- Flow Graph

- Cyclomatic Complexity
  - V(G) = Jumlah Region = 1
  - V(G) = E – N + 2 = 0 – 1 + 2 = 1
  - V(G) = P + 1 = 0 + 1 = 1
- Independent path
  - Jalur 1 = 1

Test case dan hasil akan dijelaskan pada Tabel 6.10 dibawah ini.

Tabel 6.10 Hasil Pengujian Unit Klas C_PekerjaKlinik Operasi hapusPekerjaKlinik()

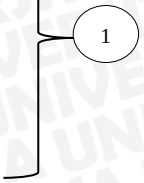
No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil operasi hapusPekrejaKlinik() dengan variabel idPekerja = 1 dan idAkun = 8	Pekerja klinik dengan idPekerja = 5 dan idAkun = 20 terhapus dari *database.* Menampilkan halaman home	Pekerja klinik dengan idPekerja = 5 dan idAkun = 20 terhapus dari *database.* Menampilkan halaman home	Valid

f. Operasi ubahPekerjaKlinik()

mengubah data idPekerja pada tabel pekerja_klinik menjadi username, pass, nama, profesi, alamat, hp

mengubah data idAkun pada tabel akun menjadi username, password, statusAkun

menampilkan halaman home



Basis Path Testing

- Flow Graph

- Cyclomatic Complexity
  - V(G) = Jumlah Region = 1
  - V(G) = E – N + 2 = 0 – 1 + 2 = 1
  - V(G) = P + 1 = 0 + 1 = 1
- Independent path
  - Jalur 1 = 1

Test case dan hasil akan dijelaskan pada Tabel 6.11 dibawah ini.

Tabel 6.11 Hasil Pengujian Unit Klas C_PekerjaKlinik Operasi ubahPekerjaKlinik()

No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil operasi ubahPekerjaKlinik() dengan variabel nama = “Tiana Siti Amalia”, username = “tiana”, pass = “tiana”, password = “tiana”, profesi = “dokter”, alamat = “Permata Cimanggu Blok F-3”, hp = “081249898920”, idPekerja = 6, idAkun = 1	Data pekerja klinik dengan idPekerja = 6 dan idAkun = 1 berhasil diubah pada tabel pekerja_klinik dan tabel akun. Menampilkan halaman home	Data pekerja klinik dengan idPekerja = 6 dan idAkun = 1 berhasil diubah pada tabel pekerja_klinik dan tabel akun. Menampilkan halaman home	Valid

6.1.3 Pengujian Unit Klas E_Akun

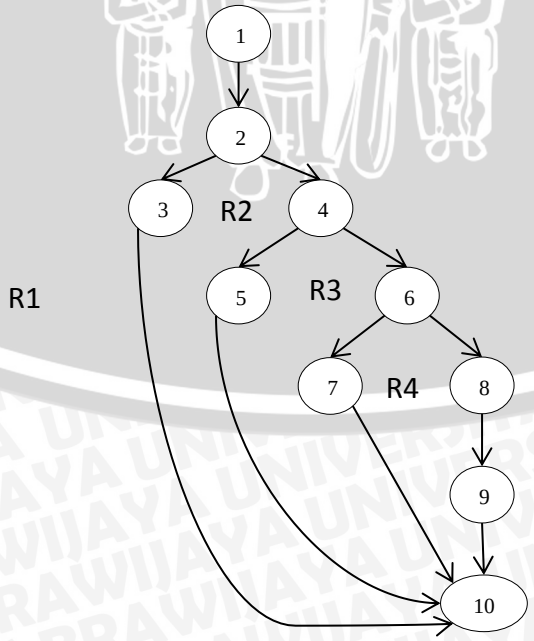
a. Operasi validasiLogin(username, password)

```
user = mengambil data akun berdasarkan username                    1
pass = mengambil data akun berdasarkan password
query = mengambil data akun dari database berdasarkan
username dan password
if(baris hasil query = 1) then                                     2
    return "valid"                                                 3
else if(baris hasil user = 1 & baris hasil pass != 1) then         4
    return "password false"                                        5
else if(baris hasil pass = 1 & baris hasil user != 1) then         6
    return "username false"                                        7

else if (baris hasil query != 1 & baris hasil user != 1 & baris hasil   8
pass != 1) then
    return "username and password false"                           9
end if                                                             10
```

Basis Path Testing

- Flow Graph



- Cyclomatic Complexity
  - V(G) = Jumlah Region = 4
  - V(G) = E – N + 2 = 12 – 10 + 2 = 4
  - V(G) = P + 1 = 3 + 1 = 4
- Independent path
  - Jalur 1 = 1 – 2 – 3 – 10
  - Jalur 2 = 1 – 2 – 4 – 5 – 10
  - Jalur 3 = 1 – 2 – 4 – 6 – 7 – 10
  - Jalur 4 = 1 – 2 – 4 – 6 – 8 – 9 – 10

Test case dan hasil akan dijelaskan pada Tabel 6.12 dibawah ini.

Tabel 6.12 Hasil Pengujian Unit Klas E_Akun Operasi validasiLogin(username, password)

No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil operasi validasiLogin(username, password) dengan username = “dea” dan password = “dea”	Menghasilkan nilai String “valid”	Menghasilkan nilai String “valid”	Valid
2.	2	Memanggil operasi validasiLogin(username, password) dengan username = “dea” dan password = “asd”	Menghasilkan nilai String “password false”	Menghasilkan nilai String “password false”	Valid
3.	3	Memanggil operasi validasiLogin(username, password) dengan username = “asd” dan	Menghasilkan nilai String “username false”	Menghasilkan nilai String “username false”	Valid

		"password" = "dea"			
4.	4	Memanggil operasi validasiLogin(username, password) dengan username = "asd" dan password = "asd"	Menghasilkan nilai String "username and password fasle"	Menghasilkan nilai String "username and password fasle"	Valid

b. Operasi tambahAkun(username, pass, statusAkun)

password = enkripsi md5 pass

Memasukkan username, password, pass, dan statusAkun ke dalam tabel akun pada database

query = mengambil data akun dari database berdasarkan username

return query

1

Basis Path Testing

- Flow Graph

- Cyclomatic Complexity
  - V(G) = Jumlah Region = 1
  - V(G) = E – N + 2 = 0 – 1 + 2 = 1
  - V(G) = P + 1 = 0 + 1 = 1
- Independent path
  - Jalur 1 = 1

Test case dan hasil akan dijelaskan pada Tabel 6.13 dibawah ini.

Tabel 6.13 Hasil Pengujian Unit Klas E_Akun Operasi tambahAkun(username, password, statusAkun)

No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil operasi	Username, password, dan	Username, password, dan	Valid

		tambahAkun(username, password, statusAkun) dengan username = "silvi", pass= "silvi", dan statusAkun = "pasien"	statusAkun berhasil dimasukkan ke dalam database dan mengembalikan data akun berdasarkan username tersebut	statusAkun berhasil dimasukkan ke dalam database dan mengembalikan data akun berdasarkan username tersebut	

c. Operasi hapusAkun(idAkun)

Menghapus data akun dari tabel akun pada database berdasarkan idAKun } 1

Basis Path Testing

- Flow Graph

- Cyclomatic Complexity
  - V(G) = Jumlah Region = 1
  - V(G) = E – N + 2 = 0 – 1 + 2 = 1
  - V(G) = P + 1 = 0 + 1 = 1
- Independent path
  - Jalur 1 = 1

Test ccase dan hasil akan dijelaskan pada Tabel 6.14 dibawah ini.

Tabel 6.14 Hasil Pengujian Unit Klas E_Akun Operasi hapusAkun(idAkun)

No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil operasi hapusAkun(idAkun) dengan idAkun = 4	Data akun yang memiliki idAkun = 4 berhasil dihapus dari *database*	Data akun yang memiliki idAkun = 4 berhasil dihapus dari *database*	Valid

d. Operasi ambilPassword(idAkun)

query = mengambil data akun dari database berdasarkan idAkun

return query } 1

Basis Path Testing

- Flow Graph

- Cyclomatic Complexity
  - V(G) = Jumlah Region = 1
  - V(G) = E – N + 2 = 0 – 1 + 2 = 1
  - V(G) = P + 1 = 0 + 1 = 1
- Independent path
  - Jalur 1 = 1

Test case dan hasil akan dijelaskan pada Tabel 6.15 dibawah ini.

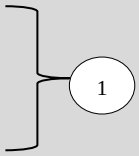
Tabel 6.15 Hasil Pengujian Unit Klas E_Akun Operasi ambilPassword(idAkun)

No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil operasi ambilPassword(idAkun) dengan idAkun = 6	Menghasilkan data akun yang memiliki idAkun = 6	Menghasilkan data akun yang memiliki idAkun = 6	Valid

e. Operasi ubahAkun()

password = enkripsi md5 password

mengubah data idAkun pada database menjadi username, password, pass, dan statusAkun



Basis Path Testing

- Flow Graph

- Cyclomatic Complexity
  - V(G) = Jumlah Region = 1
  - V(G) = E – N + 2 = 0 – 1 + 2 = 1
  - V(G) = P + 1 = 0 + 1 = 1
- Independent path
  - Jalur 1 = 1

Test case dan hasil akan dijelaskan pada Tabel 6.16 dibawah ini.

Tabel 6.16 Hasil Pengujian Unit Klas E_Akun Operasi ubahAkun()

No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil operasi ubahAkun() dengan variabel *username* = "apoteker", *password* = "apoteker", pass = "apoteker", statusAkun = "apoteker", idAkun = 43	*Username*, *password*, pass dan statusAkun dengan idAkun 43 berhasil diubah pada *database*	*Username*, *password*, pass dan statusAkun dengan idAkun 43 berhasil diubah pada *database*	Valid

f. Operasi ambilDataAkun(username, password)

```
query = mengambil data akun dari database berdasarkan username dan password
return query
```

1

Basis Path Testing

- Flow Graph

- Cyclomatic Complexity
  - V(G) = Jumlah Region = 1
  - V(G) = E – N + 2 = 0 – 1 + 2 = 1
  - V(G) = P + 1 = 0 + 1 = 1
- Independent path
  - Jalur 1 = 1

Test case dan hasil akan dijelaskan pada Tabel 6.17 dibawah ini.

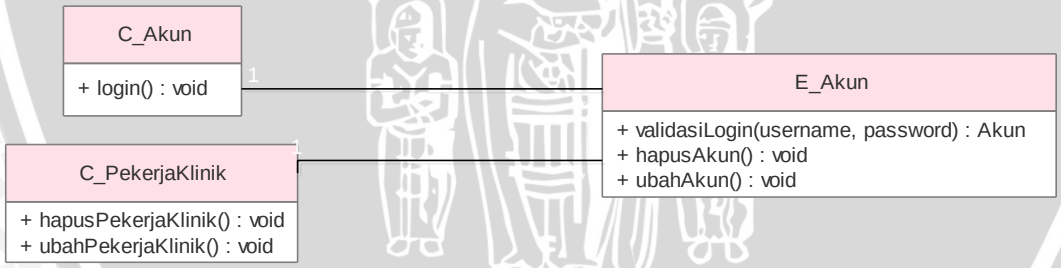
Tabel 6.17 Hasil Pengujian Unit Klas E_Akun Operasi ambilDataAkun(username, password)

No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil	Menghasilkan	Menghasilkan	Valid

		operasi ambilDataAkun(username, password dengan username = “dea” dan password = “dea”	data akun yang memiliki username = dea dan password = dea	data akun yang memiliki username = dea dan password = dea	

6.2 Pengujian Integrasi

Pengujian Integrasi digunakan untuk menguji klas-klas yang saling berhubungan. Pengujian integrasi dilakukan dengan menggunakan *basis path testing* yang merupakan salah satu jenis pengujian dari *white box testing*. Pengujian integrasi akan dilakukan untuk menguji interaksi antar klas C_Akun dengan klas E_Akun dan interaksi antar klas C_PekerjaKlinik dengan klas E_Akun sebagai sampel. Interaksi antar klas C_Akun dengan klas E_Akun dan interaksi antar klas C_PekerjaKlinik dengan klas E_Akun akan digambarkan pada Gambar 6.1. Pengujian yang pertama akan dilakukan pada operasi login() pada klas C_Akun sebagai operasi yang memanggil operasi validasiLogin(username, password) pada klas E_Akun. Pengujian yang kedua akan dilakukan pada operasi hapusPekerjaKlinik() pada klas C_PekerjaKlinik sebagai operasi yang memanggil operasi hapusAkun(idAkun) pada klas E_Akun. Pengujian yang ketiga akan dilakukan pada operasi ubahPekerjaKlinik() pada klas C_PekerjaKlinik yang memanggil operasi ubahAkun() pada klas E_Akun.



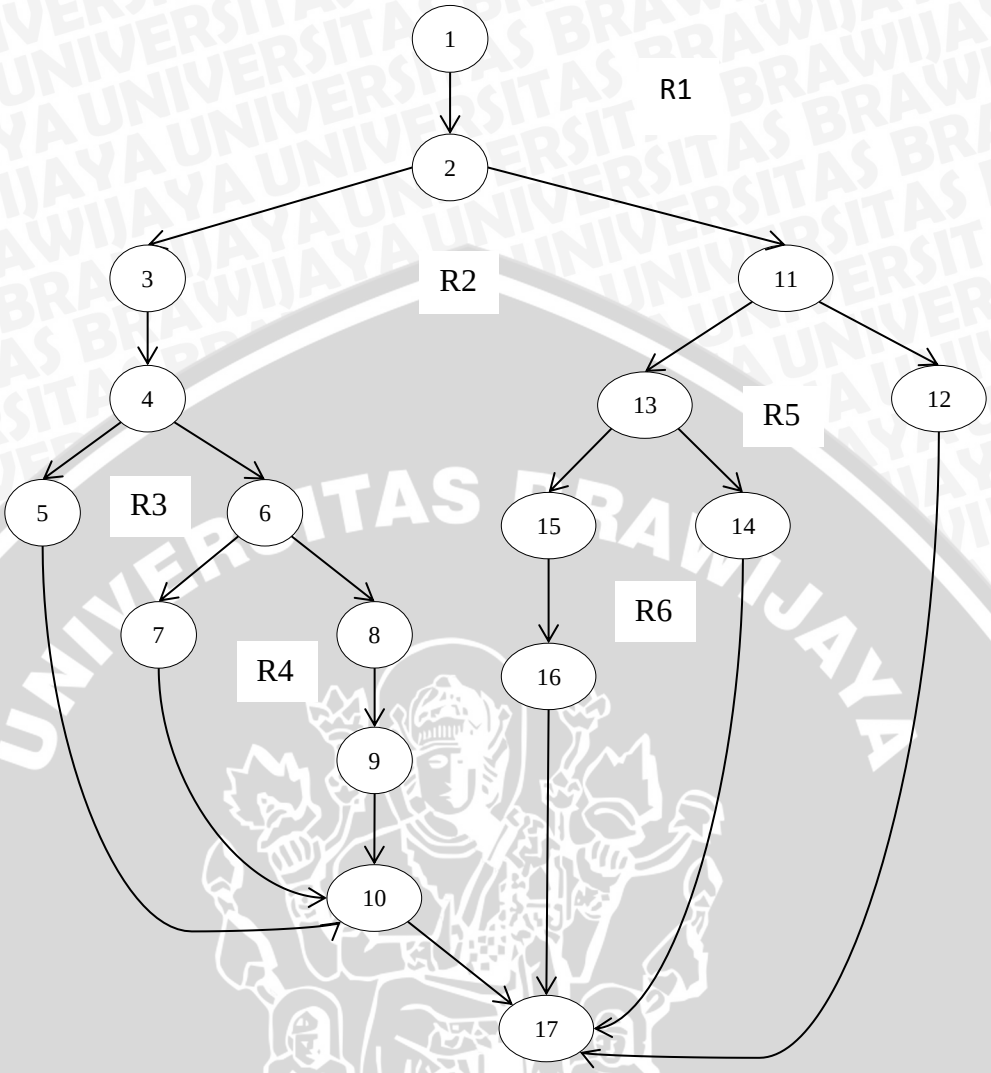
Gambar 6.1 Relasi Klas C_Akun dengan klas E_Akun dan Klas C_PekerjaKlinik dengan Klas E_Akun

6.2.1 Operasi login()

```
password = enkripsi md5 password
result = validasiLogin(username, password)        1
if(result = “valid”) then                         2
    data = array[] of String
    result = data akun berdasarkan username dan password   3
    data[idAkun] = result[idAkun]
```

Basis Path Testing

- Flow Graph



- Cyclomatic Complexity
  - V(G) = Jumlah Region = 6
  - V(G) = E – N + 2 = 21 – 17 + 2 = 6
  - V(G) = P + 1 = 5 + 1 = 6
- Independent path
  - Jalur 1 = 1 – 2 – 3 – 4 – 5 – 10 – 17
  - Jalur 2 = 1 – 2 – 3 – 4 – 6 – 7 – 10 – 17
  - Jalur 3 = 1 – 2 – 3 – 4 – 6 – 8 – 9 – 10 – 17
  - Jalur 4 = 1 – 2 – 11 – 12 – 17
  - Jalur 5 = 1 – 2 – 11 – 13 – 14 - 17
  - Jalur 6 = 1 – 2 – 11 – 13 – 15 – 16 - 17

Test case dan hasil akan dijelaskan pada Tabel 6.18 dibawah ini.

Tabel 6.18 Hasil Pengujian Integrasi Klas C_Akun dengan Klas E_Akun Pada Operasi login()

No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil operasi login() dengan variabel *username* = "dea" dan *password* = "dea"	Berhasil *login* sebagai pasien. Menampilkan halaman home	Berhasil *login* sebagai pasien. Menampilkan halaman home	Valid
2.	2	Memanggil operasi login() dengan variabel u*sername* = "tiana" dan p*assword* = "tiana"	Berhasil *login* sebagai dokter. Menampilkan halaman home	Berhasil *login* sebagai dokter. Menampilkan halaman home	Valid
3.	2	Memanggil operasi login() dengan variabel u*sername* = "deanti" dan *Password* = "deanti"	Berhasil *login* sebagai perawat. Menampilkan halaman home	Berhasil *login* sebagai perawat. Menampilkan halaman home	Valid
4.	2	Memanggil operasi login() dengan variabel u*sername* = "apoteker" dan *password* = "apoteker"	Berhasil *login* sebagai apoteker. Menampilkan halaman home	Berhasil *login* sebagai apoteker. Menampilkan halaman home	Valid
5.	3	Memanggil operasi login() dengan variabel *username* = "admin" dan *password* = "admin"	Berhasil *login* sebagai admin. Menampilkan halaman home	Berhasil *login* sebagai admin. Menampilkan halaman home	Valid
6.	4	Memanggil operasi login() dengan variabel *username* =	Gagal *login*. Menampilkan pesan kesalahan "Password	Gagal *login*. Menampilkan pesan kesalahan "Password	Valid

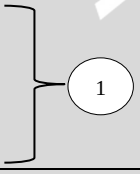
		"dea" dan *password* = "abcd"	salah!" pada halaman index	salah!" pada halaman index	
7.	5	Memanggil operasi login() dengan variabel *username* = "abcd" dan *password* = "dea"	Gagal *login*. Menampilkan pesan kesalahan "Username salah!" pada halaman index	Gagal *login*. Menampilkan pesan kesalahan "Username salah!" pada halaman index	Valid
8.	6	Memanggil operasi login() dengan variabel *username* = "abcd" dan *password* = "abcd"	Gagal *login*. Menampilkan pesan kesalahan "Username dan password salah!" pada halaman index	Gagal *login*. Menampilkan pesan kesalahan "Username dan password salah!" pada halaman index	Valid

6.2.2 Operasi hapusPekerjaKlinik()

menghapus data idPekerja dari tabel pekerja_klinik

hapusAkun(idAkun)

menampilkan halaman admin



Basis Path Testing

- Flow Graph

- Cyclomatic Complexity
  - V(G) = Jumlah Region = 1
  - V(G) = E – N + 2 = 0 – 1 + 2 = 1
  - V(G) = P + 1 = 0 + 1 = 1
- Independent path
  - Jalur 1 = 1

Test case dan hasil akan dijelaskan pada Tabel 6.19 dibawah ini.

Tabel 6.19 Hasil Pengujian Integrasi Klas C_PekerjaKlinik dengan Klas E_Akun Pada Operasi hapusPekerjaKlinik()

No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil operasi	Pekerja klinik dengan	Pekerja klinik dengan	Valid

		hapusPekrejaKlinik() dengan variabel idPekerja = 1 dan idAkun = 8	idPekerja = 5 dan idAkun = 20 terhapus dari *database.* Menampilkan halaman home	idPekerja = 5 dan idAkun = 20 terhapus dari *database.* Menampilkan halaman home	

6.2.3 Operasi ubahPekerjaKlinik()

mengubah data idPekerja pada tabel pekerja_klinik menjadi username, pass, nama, profesi, alamat, hp

ubahAkun()

1

Basis Path Testing

- Flow Graph

- Cyclomatic Complexity
  - V(G) = Jumlah Region = 1
  - V(G) = E – N + 2 = 0 – 1 + 2 = 1
  - V(G) = P + 1 = 0 + 1 = 1
- Independent path
  - Jalur 1 = 1

Test case dan hasil akan dijelaskan pada Tabel 6.20 dibawah ini.

Tabel 6.20 Hasil Pengujian Integrasi Klas C_PekerjaKlinik dengan Klas E_Akun Pada Operasi ubahPekerjaKlinik()

No	No Jalur	Prosedur Uji	Expected Result	Result	Status
1.	1	Memanggil operasi ubahPekerjaKlinik() dengan variabel nama = “Fadia Izna”, username = “apoteker”, pass = “apoteker”, password = “apoteker”,	Data pekerja klinik dengan idPekerja = 3 dan idAkun = 43 berhasil diubah pada tabel pekerja_klinik dan tabel akun. Menampilkan halaman home	Data pekerja klinik dengan idPekerja = 3 dan idAkun = 43 berhasil diubah pada tabel pekerja_klinik dan tabel akun. Menampilkan halaman home	Valid

		profesi = “apoteker”, alamat = “Jl. Kedung Badak Blok G-9” , idPekerja = 3, idAkun = 43			

6.3 Pengujian Validasi

Pengujian Validasi digunakan untuk mengetahui apakah sistem yang dibangun sudah sesuai dengan seluruh kebutuhan yang telah ditetapkan atau tidak. Pengujian validasi dilakukan dengan cara memeriksa apakah sistem sudah berjalan dengan baik dan tidak ada *error* yang terjadi. Proses pengujian validasi mengacu pada daftar kebutuhan fungsional maupun non-fungsional yang telah ditetapkan berdasarkan hasil proses analisis kebutuhan. Pengujian validasi akan dilakukan pada semua kebutuhan sistem dengan menggunakan metode *black box testing*.

6.3.1 Pengujian Login

Pengujian validasi *login* dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_0100. Berikut ini adalah prosedur kasus uji yang dilakukan.

a. Kasus Uji Berhasil Login Sebagai Pasien

Kasus uji berhasil login sebagai pasien akan dijelaskan pada Tabel 6.21.

Tabel 6.21 Kasus Uji Berhasil Login Sebagai Pasien

Nama Kasus Uji	Kasus uji berhasil login Sebagai Pasien
Prosedur	1. Membuka formulir login pada halaman utama 2. Memasukkan *username = dea* dan *password* = dea 3. Menekan tombol Login
Hasil yang diharapkan	Sistem akan menampilkan halaman awal pasien
Hasil	Sistem akan menampilkan halaman awal pasien
Status	Valid

b. Kasus Uji Berhasil Login Sebagai Dokter

Kasus uji berhasil login sebagai dokter akan dijelaskan pada Tabel 6.22.

Tabel 6.22 Kasus Uji Berhasil Login Sebagai Dokter

Nama Kasus Uji	Kasus uji berhasil login sebagai dokter
Prosedur	1. Membuka formulir login pada halaman utama 2. Memasukkan *username = tiana* dan *password* = tiana

	3. Menekan tombol Login
Hasil yang diharapkan	Sistem akan menampilkan halaman awal dokter
Hasil	Sistem akan menampilkan halaman awal dokter
Status	Valid

c. Kasus Uji Berhasil Login Sebagai Perawat

Kasus uji berhasil login sebagai perawat akan dijelaskan pada Tabel 6.23.

Tabel 6.23 Kasus Uji Berhasil Login Sebagai Perawat

Nama Kasus Uji	Kasus uji berhasil login sebagai perawat
Prosedur	1. Membuka formulir login pada halaman utama 2. Memasukkan *username = deanti* dan *password* = deanti 3. Menekan tombol Login
Hasil yang diharapkan	Sistem akan menampilkan halaman awal perawat
Hasil	Sistem akan menampilkan halaman awal perawat
Status	Valid

d. Kasus Uji Berhasil Login Sebagai Apoteker

Kasus uji berhasil login sebagai apoteker akan dijelaskan pada Tabel 6.24.

Tabel 6.24 Kasus Uji Berhasil Login Sebagai Apoteker

Nama Kasus Uji	Kasus uji berhasil login sebagai apoteker
Prosedur	1. Membuka formulir login pada halaman utama 2. Memasukkan *username = apoteker* dan *password* = apoteker 3. Menekan tombol Login
Hasil yang diharapkan	Sistem akan menampilkan halaman awal apoteker
Hasil	Sistem akan menampilkan halaman awal apoteker
Status	Valid

e. Kasus Uji Berhasil Login Sebagai Admin

Kasus uji berhasil login sebagai admin akan dijelaskan pada Tabel 6.25.

Tabel 6.25 Kasus Uji Berhasil Login Sebagai Admin

Nama Kasus Uji	Kasus uji berhasil login Sebagai Admin
Prosedur	1. Membuka formulir login pada halaman utama 2. Memasukkan *username = admin* dan *password* = admin 3. Menekan tombol Login

Hasil yang diharapkan	Sistem akan menampilkan halaman awal admin
Hasil	Sistem akan menampilkan halaman awal admin
Status	Valid

f. Kasus Uji Data Login Username Kosong

Kasus uji data login username kosong akan dijelaskan pada Tabel 6.26.

Tabel 6.26 Kasus Uji Data Login Username Kosong

Nama Kasus Uji	Kasus uji data login username kosong
Prosedur	1. Membuka formulir login pada halaman utama 2. Memasukkan *password* 3. Menekan tombol Login
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Hasil	Sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Status	Valid

g. Kasus Uji Data Login Password Kosong

Kasus uji data login password kosongakan dijelaskan pada Tabel 6.27.

Tabel 6.27 Kasus Uji Data Login Password Kosong

Nama Kasus Uji	Kasus uji data login password kosong
Prosedur	1. Membuka formulir login pada halaman utama 2. Memasukkan *username* 3. Menekan tombol Login
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Hasil	Sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Status	Valid

h. Kasus Uji Data Login Kosong

Kasus uji data login kosong akan dijelaskan pada Tabel 6.28.

Tabel 6.28 Kasus Uji Data Login Kosong

Nama Kasus Uji	Kasus uji data login kosong
Prosedur	1. Membuka formulir login pada halaman utama 2. Menekan tombol Login
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Kotak

	ini tidak boleh kosong" pada kotak yang kosong
Hasil	Sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Status	Valid

i. Kasus Uji Password Salah

Kasus uji password salah akan dijelaskan pada Tabel 6.29.

Tabel 6.29 Kasus Uji Password Salah

Nama Kasus Uji	Kasus uji *password* salah
Prosedur	1. Membuka formulir login pada halaman utama 2. Memasukkan 'asdf' pada kotak *passowrd* dan 'dea' pada kotak *username* 3. Menekan tombol Login
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Passowrd salah!"
Hasil	Sistem menampilkan pesan kesalahan "Passowrd salah!"
Status	Valid

j. Kasus Uji Username dan Password Salah

Kasus uji username dan password salah akan dijelaskan pada Tabel 6.30.

Tabel 6.30 Kasus Uji Username dan Password Salah

Nama Kasus Uji	Kasus uji username dan password salah
Prosedur	1. Membuka formulir login pada halaman utama 2. Memasukkan 'asdf' pada kotak *username* dan 'asdf' pada kotak *password* 3. Menekan tombol Login
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Username dan password salah!"
Hasil	Sistem menampilkan pesan kesalahan "Username dan password salah!"
Status	Valid

k. Kasus Uji Username Salah

Kasus uji username salah akan dijelaskan pada Tabel 6.31.

Tabel 6.31 Kasus Uji Username Salah

Nama Kasus Uji	Kasus uji *username* salah
Prosedur	1. Membuka formulir login pada halaman utama

	2. Memasukkan ‘asdf’ pada kotak *username* dan ‘dea’ pada kotak *password* 3. Menekan tombol Login
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan *“*Username salah!”
Hasil	Sistem menampilkan pesan kesalahan *“*Username salah!”
Status	Valid

6.3.2 Pengujian Logout

Pengujian validasi *logout* dilakukan untuk Dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_0200. Kasus uji logout akan dijelaskan pada Tabel 6.32.

Tabel 6.32 Kasus Uji Logout

Nama Kasus Uji	Kasus uji logout
Prosedur	1. Menekan tombol Logout
Hasil yang diharapkan	Sistem akan menghapus *session* pengguna yang ingin *logout.* Kemudian menampilkan halaman awal sistem
Hasil	Sistem menghapus *session* pengguna yang ingin *logout.* Kemudian menampilkan halaman awal sistem
Status	Valid

6.3.3 Pengujian Mendaftar Menjadi Pasien

Pengujian validasi mendaftar menjadi pasien tetap dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_0300, MCF_0301, MCF_0302, MCF_0303, MCF_0304, MCF_0305. Terdapat enam kasus uji yaitu berhasil mendaftar menjadi pasien, mendaftar dengan *username* yang sudah digunakan, data pendaftaran menjadi pasien tidak lengkap, mendaftar menjadi pasien lebih dari satu kali, kode verifikasi pendaftaran menjadi pasien kosong, kode verifikasi pendaftaran menjadi pasien salah, verifikasi kode pendaftaran menjadi pasien melebihi batas waktu. Berikut ini adalah prosedur kasus uji yang dilakukan.

a. Kasus Uji Berhasil Mendaftar Menjadi Pasien

Kasus uji berhasil mendaftar menjadi pasien akan dijelaskan pada Tabel 6.33.

Tabel 6.33 Kasus Uji Berhasil Mendaftar Menjadi Pasien

Nama Kasus Uji	Kasus uji berhasil mendaftar menjadi pasien
Prosedur	1. Membuka halaman pendaftaran pasien tetap

	2. Memasukkan *username* yang belum digunakan 3. Memasukkan data pribadi yang benar dan lengkap 4. Menekan tombol Daftar 5. Memasukkan kode verifikasi yang benar 6. Menekan tombol Verifikasi
Hasil yang diharapkan	Pasien akan langsung *login* ke dalam sistem dan sistem menampilkan halaman awal pasien
Hasil	Pasien langsung *login* ke dalam sistem dan sistem menampilkan halaman awal pasien
Status	Valid

b. Kasus Uji Mendaftar Dengan *Username* yang Sudah Digunakan

Kasus uji mendaftar dengan username yang sudah digunakan akan dijelaskan pada Tabel 6.34.

Tabel 6.34 Kasus Uji Mendaftar Dengan Username yang Sudah Digunakan

Nama Kasus Uji	Kasus uji mendaftar dengan *username* yang sudah digunakan
Prosedur	1. Membuka halaman pendaftaran pasien tetap 2. Memasukkan *username* yang sudah digunakan
Hasil yang diharapkan	Sistem akan langsung menampilkan tulisan "Username telah digunakan"
Hasil	Sistem langsung menampilkan tulisan "Username telah digunakan"
Status	Valid

c. Kasus Uji Data Pendaftaran Menjadi Pasien Tidak Lengkap

Kasus uji data pendaftaran menjadi pasien tidak lengkap akan dijelaskan pada Tabel 6.35.

Tabel 6.35 Kasus Uji Data Pendaftaran Menjadi Pasien Tidak Lengkap

Nama Kasus Uji	Kasus uji data pendaftaran menjadi pasien tidak lengkap
Prosedur	1. Membuka halaman pendaftaran pasien tetap 2. Memasukkan *username* yang belum digunakan 3. Memasukkan data pribadi, namun tidak lengkap atau terdapat data pribadi yang dikosongkan 4. Menekan tombol Daftar
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong

Hasil	Sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Status	Valid

d. Kasus Uji Mendatar Menjadi Pasien Lebih Dari Satu Kali

Kasus uji mendaftar menjadi pasien lebih dari satu kali akan dijelaskan pada Tabel 6.36.

Tabel 6.36 Kasus Uji Mendaftar Menjadi Pasien Lebih Dari Satu Kali

Nama Kasus Uji	Kasus uji mendaftar menjadi pasien lebih dari satu kali
Prosedur	1. Membuka halaman pendaftaran pasien tetap 2. Memasukkan *username* yang belum digunakan 3. Memasukkan data pribadi yang telah terdaftar sebelumnya 4. Menekan tombol Daftar
Hasil yang diharapkan	Sistem akan menampilkan pesan "Data pribadi yang Anda masukkan sudah terdaftar sebagai pasien tetap"
Hasil	Sistem akan menampilkan pesan "Data pribadi yang Anda masukkan sudah terdaftar sebagai pasien tetap"
Status	Valid

e. Kasus Uji Kode Verifikasi Pendaftaran Menjadi Pasien Kosong

Kasus uji kode verifikasi pendaftaran menjadi pasien kosong akan dijelaskan pada Tabel 6.37.

Tabel 6.37 Kasus Uji Kode Verifikasi Pendaftaran Menjadi Pasien Kosong

Nama Kasus Uji	Kasus uji kode verifikasi pendaftaran menjadi pasien kosong
Prosedur	1. Membuka halaman pendaftaran pasien tetap 2. Memasukkan *username* yang belum digunakan 3. Memasukkan data pribadi dengan benar dan lengkap 4. Menekan tombol Daftar 5. Menekan tombol Verifikasi
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Hasil	Sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong

Status	Valid

f. Kasus Uji Kode Verifiaksi Pendaftaran Menjadi Pasien Salah

Kasus uji kode verifikasi pendaftaran menjadi pasien salah akan dijelaskan pada Tabel 6.38.

Tabel 6.38 Kasus Uji Kode Verifikasi Pendaftaran Menjadi Pasien Salah

Nama Kasus Uji	Kasus uji kode verifikasi pendaftaran menjadi pasien salah
Prosedur	1. Membuka halaman pendaftaran pasien tetap 2. Memasukkan *username* yang belum digunakan 3. Memasukkan data pribadi dengan benar dan lengkap 4. Menekan tombol Daftar 5. Memasukkan kode verifikasi yang salah 6. Menekan tombol Verifikasi
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Akun Anda gagal di verifikasi, Kode Verifikasi yang Anda masukkan salah."
Hasil	Sistem akan menampilkan pesan kesalahan "Akun Anda gagal di verifikasi, Kode Verifikasi yang Anda masukkan salah."
Status	Valid

g. Kasus Uji Verifikasi Kode Pendaftaran Menjadi Pasien Melebihi Batas Waktu

Kasus uji verifikasi kode pendaftaran menjadi pasien melebihi batas waktu akan dijelaskan pada Tabel 6.39.

Tabel 6.39 Kasus Uji Verifikasi Kode Pendaftaran Menjadi Pasien Melebihi Batas Waktu

Nama Kasus Uji	Kasus uji verifikasi kode pendaftaran menjadi pasien melebihi batas waktu
Prosedur	1. Membuka halaman pendaftaran pasien tetap 2. Memasukkan *username* yang belum digunakan 3. Memasukkan data pribadi dengan benar dan lengkap 4. Menekan tombol Daftar 5. Memasukkan kode verifikasi yang benar pada saat waktu sudah melebihi batas waktu yang ditentukan 6. Menekan tombol Verifikasi
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Batas waktu verifikasi sudah habis. Silahkan daftar

	kembali.”
Hasil	Sistem akan menampilkan pesan kesalahan “Batas waktu verifikasi sudah habis. Silahkan daftar kembali.”
Status	Valid

6.3.4 Pengujian Mendaftar Konsultasi

Pengujian validasi mendaftar konsultasi dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_0400, MCF_0401, MCF_0402. Terdapat tiga kasus uji yaitu berhasil mendaftar konsultasi, mendaftar konsultasi ketika kuota habis, dan mendaftar konsultasi ketika pendaftaran sudah tutup. Berikut ini adalah prosedur kasus uji yang dilakukan.

a. Kasus Uji Berhasil Mendaftar Konsultasi

Kasus uji berhasil mendaftar konsultasi akan dijelaskan pada Tabel 6.40.

Tabel 6.40 Kasus Uji Berhasil Mendaftar Konsultasi

Nama Kasus Uji	Kasus uji berhasil mendaftar konsultasi
Prosedur	1. Membuka halaman daftar konsultasi 2. Menekan tombol Daftar pada saat kuota belum habis dan waktu pendaftaran *online* belum ditutup
Hasil yang diharapkan	Sistem akan mengirim nomor antrian kepada pasien melalui *SMS.* Kemudian sistem menampilkan kembali halaman daftar konsultasi dan tombol yang ditampilkan berubah menjadi tombol Batal
Hasil	Sistem akan mengirim nomor antrian kepada pasien melalui *SMS.* Kemudian sistem menampilkan kembali halaman daftar konsultasi dan tombol yang ditampilkan berubah menjadi tombol Batal
Status	Valid

b. Kasus Uji Mendaftar Konsultasi Ketika Kuota Habis

Kasus uji mendaftar konsultasi ketika kuota habis akan dijelaskan pada Tabel 6.41.

Tabel 6.41 Kasus Uji Mendaftar Konsultasi Ketika Kuota Habis

Nama Kasus Uji	Kasus uji mendaftar konsultasi ketika kuota habis
Prosedur	1. Membuka halaman daftar konsultasi 2. Menekan tombol Daftar ketika kuota sudah habis
Hasil yang diharapkan	Sistem akan menampilkan pesan “Pendaftaran konsultasi online gagal dilakukan. Kuota pendaftaran

	online sudah habis."
Hasil	Sistem akan menampilkan pesan "Pendaftaran konsultasi online gagal dilakukan. Kuota pendaftaran online sudah habis."
Status	Valid

c. Kasus Uji Mendaftar Konsultasi Ketika Pendaftaran Sudah Ditutup

Kasus uji mendaftar konsultasi ketika pendaftaran sudah tutup akan dijelaskan pada Tabel 6.42.

Tabel 6.42 Kasus Uji Mendaftar Konsultasi Ketika Pendaftaran Sudah Ditutup

Nama Kasus Uji	Kasus uji mendaftar konsultasi ketika pendaftaran sudah ditutup
Prosedur	1. Membuka halaman daftar konsultasi 2. Menekan tombol Daftar lebih dari jam Sembilan malam
Hasil yang diharapkan	Sistem akan menampilkan pesan "Pendaftaran konsultasi online gagal dilakukan. Waktu pendaftaran konsultasi online sudah habis"
Hasil	Sistem menampilkan pesan "Pendaftaran konsultasi online gagal dilakukan. Waktu pendaftaran konsultasi online sudah habis"
Status	Valid

6.3.5 Pengujian Membatalkan Konsultasi

Pengujian validasi membatalkan konsultasi dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_0500, MCF_0501, MCF_0502, dan MCF_0503. Kasus uji membatalkan konsultasi akan dijelaskan pada Tabel 6.43.

Tabel 6.43 Kasus Uji Membatalkan Konsultasi

Nama Kasus Uji	Kasus uji membatalkan konsultasi
Prosedur	1. Membuka halaman daftar konsultasi 2. Menekan tombol Batal
Hasil yang diharapkan	Sistem akan menampilkan kembali halaman daftar konsultasi dan tombol yang ditampilkan berubah menjadi tombol Daftar
Hasil	Sistem akan menampilkan kembali halaman daftar konsultasi dan tombol yang ditampilkan berubah menjadi tombol Daftar
Status	Valid

6.3.6 Pengujian Melihat Data Pribadi

Pengujian validasi melihat data pribadi dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_0600 dan MCF_0601. Tabel 6.44 adalah kasus uji untuk pengujian validasi melihat data pribadi.

Tabel 6.44 Kasus Uji Melihat Data Pribadi

Nama Kasus Uji	Kasus uji melihat data pribadi
Prosedur	1. Memilih menu Profil pada halaman awal pasien
Hasil yang diharapkan	Sistem akan menampilkan halaman Profil yang berisi data pribadi pasien dan rekam medis pasien tersebut
Hasil	Sistem menampilkan halaman Profil yang berisi data pribadi pasien dan rekam medis pasien tersebut
Status	Valid

6.3.7 Pengujian Mengubah Profil

Pengujian validasi mengubah profil dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_0700 dan MCF_0701. Terdapat dua kasus uji yaitu berhasil mengubah profil dan gagal mengubah profil. Berikut ini adalah prosedur kasus uji yang dilakukan.

a. Kasus Uji Berhasil Mengubah Profil

Kasus uji berhasul mengubah profil akan dijelaskan pada Tabel 6.45.

Tabel 6.45 Kasus Uji Berhasil Mengubah Profil

Nama Kasus Uji	Kasus uji berhasil mengubah profil
Prosedur	1. Membuka halaman profil 2. Menekan tombol Edit Profil 3. Memasukkan alamat dan nomor *handphone* yang baru 4. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan menampilkan halaman profil dengan alamat dan nomor *handphone* yang tadi diubah
Hasil	Sistem menampilkan halaman profil dengan alamat dan nomor *handphone* yang tadi diubah
Status	Valid

b. Kasus Uji Mengubah Alamat Profil Dengan Data Kosong

Kaasus uji mengubah alamat profil dengan data kosong akan dijelaskan pada Tabel 6.46.

Tabel 6.46 Kasus Uji Mengubah Alamat Profil Dengan Data Kosong

Nama Kasus Uji	Kasus uji mengubah alamat profil dengan data

	kosong
Prosedur	1. Membuka halaman profil 2. Menekan tombol Edit Profil 3. Mengosongkan alamat 4. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Hasil	Sistem menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Status	Valid

c. Kasus Uji Mengubah Nomor Handphone Profil Dengan Data Kosong

Kasus uji mengubah nomor handphone profil dengan data kosong akan dijelaskan pada Tabel 6.47.

Tabel 6.47 Kasus Uji Gagal Mengubah Nomor Handphone Dengan Data Kosong

Nama Kasus Uji	Kasus uji mengubah nomor handphone dengan data kosong
Prosedur	1. Membuka halaman profil 2. Menekan tombol Edit Profil 3. Mengosongkan nomor *handphone* 4. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Hasil	Sistem menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Status	Valid

d. Kasus Uji Gagal Mengubah Profil

Kasus uji gagal mengubah profil akan dijelaskan pada Tabel 6.48.

Tabel 6.48 Kasus Uji Gagal Mengubah Profil

Nama Kasus Uji	Kasus uji gagal mengubah profil
Prosedur	1. Membuka halaman profil 2. Menekan tombol Edit Profil 3. Mengosongkan alamat dan nomor *handphone* 4. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Hasil	Sistem menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong

Status	Valid

6.3.8 Pengujian Mengisi Rekam Medis

Pengujian validasi mengisi rekam medis dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_0800. Terdapat tiga kasus uji yaitu berhasil mengisi rekam medis, mengisi rekam medis tidak lengkap, dan gagal mengisi rekam medis. Berikut ini adalah prosedur kasus uji yang dilakukan.

a. Kasus Uji Berhasil Mengisi Rekam Medis

Kasus uji berhasil mengisi rekam medis akan dijelaskan pada Tabel 6.49.

Tabel 6.49 Kasus Uji Berhasil Mengisi Rekam Medis

Nama Kasus Uji	Kasus uji berhasil mengisi rekam medis
Prosedur	1. Memilih menu Konsultasi 2. Memasukkan elemen, diagnosa, dan terapi pada formulir 3. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan menambahkan rekam medis ke dalam *database.* Lalu menampilkan kembali halaman konsultasi
Hasil	Sistem menambahkan rekam medis ke dalam *database.* Lalu menampilkan kembali halaman konsultasi
Status	Valid

b. Kasus Uji Mengisi Rekam Medis Dengan Elemen Kosong

Kasus uji mengisi rekam medis dengan elemen kosong akan dijelaskan pada Tabel 6.50.

Tabel 6.50 Kasus Uji Mengisi Rekam Medis Dengan elemen kosong

Nama Kasus Uji	Kasus uji mengisi rekam medis dengan elemen kosong
Prosedur	1. Memilih menu Konsultasi 2. Memasukkan diagnosa dan terapi 3. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan “Kotak ini tidak boleh kosong” pada kotak yang kosong
Hasil	Sistem menampilkan pesan kesalahan “Kotak ini tidak boleh kosong” pada kotak yang kosong
Status	Valid

c. Kasus Uji Mengisi Rekam Medis Dengan Diagnosa Kosong

Kasus uji mengisi rekam medis dengan diagnose kosong akan dijelaskan pada Tabel 6.51.

Tabel 6.51 Kasus Uji Mengisi Rekam Medis Dengan Diagnosa Kosong

Nama Kasus Uji	Kasus uji mengisi rekam medis dengan diagnosa kosong
Prosedur	1. Memilih menu Konsultasi 2. Memasukkan elemen dan terapi 3. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Hasil	Sistem menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Status	Valid

d. Kasus Uji Mengisi Rekam Medis Dengan Terapi Kosong

Kasus uji mengisi rekam medis dengan terapi kosong akan dijelaskan pada Tabel 6.52.

Tabel 6.52 Kasus Uji Mengisi Rekam Medis Dengan Terapi Kosong

Nama Kasus Uji	Kasus uji mengisi rekam medis dengan terapi kosong
Prosedur	1. Memilih menu Konsultasi 2. Memasukkan elemen dan diagnosa 3. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Hasil	Sistem menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Status	Valid

e. Kasus Uji Mengisi Rekam Medis Tidak Lengkap

Kasus uji mengisi rekam medis tidak lengkap akan dijelaskan pada Tabel 6.53.

Tabel 6.53 Kasus Uji Mengisi Rekam Medis Tidak Lengkap

Nama Kasus Uji	Kasus uji mengisi rekam medis tidak lengkap
Prosedur	4. Memilih menu Konsultasi 5. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Hasil	Sistem menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong

Status	Valid

f. Kasus Uji Gagal Mengisi Rekam Medis

Kasus uji gagal mengisi rekam medis akan dijelaskan pada Tabel 6.54.

Tabel 6.54 Kasus Uji Gagal Mengisi Rekam Medis

Nama Kasus Uji	Kasus uji gagal mengisi rekam medis
Prosedur	1. Memilih menu Konsultasi 2. Memasukkan elemen, diagnosa, dan terapi pada formulir 3. Menekan tombol Simpan pada saat data pribadi pasien pada halaman konsultasi bernilai nol
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Tidak dapat memasukkan data. Karena tidak ada pasien yang konsultasi."
Hasil	Sistem menampilkan pesan kesalahan "Tidak dapat memasukkan data. Karena tidak ada pasien yang konsultasi."
Status	Valid

6.3.9 Pengujian Melihat Rekam Medis Pasien

Pengujian validasi melihat rekam medis pasien dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_0900 dan MCF_0901. Kasus uji melihat rekam medis pasien akan dijelaskan pada Tabel 6.55.

Tabel 6.55 Kasus Uji Melihat Rekam Medis Pasien

Nama Kasus Uji	Kasus uji melihat rekam medis pasien
Prosedur	1. Memilih menu Pasien 2. Menekan tombol Rekam Medis pada baris pasien yang ingin dilihat rekam medisnya
Hasil yang diharapkan	Sistem akan menampilkan data pribadi dan rekam medis pasien yang dipilih
Hasil	Sistem menampilkan data pribadi dan rekam medis pasien yang dipilih

6.3.10 Pengujian Memasukkan Jadwal Konsultasi

Pengujian validasi memasukkan jadwal konsultasi dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_1000. Terdapat tiga kasus uji yaitu berhasil memasukkan jadwal konsultasi, memasukkan jadwal konsultasi tidak lengkap, dan gagal memasukkan jadwal konsultasi. Berikut ini adalah prosedur kasus uji yang dilakukan.

a. Kasus Uji Berhasil Memasukkan Jadwal Konsultasi

Kasus uji berhasil memasukkan jadwal konsultasi akan dijelaskan pada Tabel 6.56.

Tabel 6.56 Kasus Uji Berhasil Memasukkan Jadwal Konsultasi

Nama Kasus Uji	Kasus uji berhasil memasukkan jadwal konsultasi
Prosedur	1. Memilih menu Konsultasi 2. Menekan tombol Konsultasi Lanjut 3. Memasukkan tanggal konsultasi lanjut dan keterangan 4. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan menambahkan jadwal konsultasi lanjut ke dalam *database.* Lalu menampilkan kembali halaman konsultasi
Hasil	Sistem menambahkan jadwal konsultasi lanjut ke dalam *database.* Lalu menampilkan kembali halaman konsultasi
Status	Valid

b. Kasus Uji Memasukkan Jadwal Konsultasi Dengan Mengosongkan Keterangan

Kasus uji memasukkan jadwal konsultasi dengan mengosongkan keterangan akan dijelaskan pada Tabel 6.57.

Tabel 6.57 Kasus Uji Memasukkan Jadwal Konsutlasi Dengan engosongkan Keterangan

Nama Kasus Uji	Kasus uji memasukkan jadwal konsutlasi dengan mengosongkan keterangan
Prosedur	1. Memilih menu Konsultasi 2. Menekan tombol Konsultasi Lanjut 3. Memasukkan tanggal konsultasi lanjut 4. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan “Kotak ini tidak boleh kosong” pada kotak yang kosong
Hasil	Sistem menampilkan pesan kesalahan “Kotak ini tidak boleh kosong” pada kotak yang kosong
Status	Valid

c. Kasus Uji Memasukkan Jadwal Konsultasi Dengan Mengosongkan Tanggal

Kasus uji memasukkan jadwal konsultasi dengan mengosongkan tanggal akan dijelaskan pada Tabel 6.58.

Tabel 6.58 Kasus Uji Memasukkan Jadwal Konsutlasi Tidak Lengkap

Nama Kasus Uji	Kasus uji memasukkan jadwal konsutlasi tidak lengkap
Prosedur	1. Memilih menu Konsultasi 2. Menekan tombol Konsultasi Lanjut 3. Memasukkan keterangan 4. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Hasil	Sistem menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Status	Valid

d. Kasus Uji Memasukkan Jadwal Konsultasi Tidak Lengkap

Kasus uji memasukkan jadwal konsultasi tidak lengkap akan dijelaskan pada Tabel 6.59.

Tabel 6.59 Kasus Uji Memasukkan Jadwal Konsutlasi Tidak Lengkap

Nama Kasus Uji	Kasus uji memasukkan jadwal konsutlasi tidak lengkap
Prosedur	1. Memilih menu Konsultasi 2. Menekan tombol Konsultasi Lanjut 3. Memasukkan data konsultasi lanjut namun terdapat data yang dikosongkan 4. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Hasil	Sistem menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Status	Valid

e. Kasus Uji Gagal Memasukkan Jadwal Konsutlasi

Kasus uji gagal memasukkan jadwal konsultasi akan dijelaskan pada Tabel 6.60.

Tabel 6.60 Kasus Uji Gagal Memasukkan Jadwal Konsutlasi

Nama Kasus Uji	Kasus uji gagal memasukkan jadwal konsutlasi
Prosedur	1. Memilih Konsultasi 2. Menekan tombol Konsultasi Lanjut 3. Memasukkan jadwal konsultasi lanjut dan keterangan 4. Menekan tombol Simpan pada saat data pribadi

	pasien pada halaman konsultasi bernilai nol
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Tidak dapat memasukkan data. Karena tidak ada pasien yang konsultasi."
Hasil	Sistem menampilkan pesan kesalahan "Tidak dapat memasukkan data. Karena tidak ada pasien yang konsultasi."
Status	Valid

6.3.11 Pengujian Menulis Resep

Pengujian validasi menulis resep dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_1100. Terdapat tiga kasus uji yaitu berhasil menulis resep, menulis resep kosong, dan gagal menulis resep. Berikut ini adalah prosedur kasus uji yang dilakukan.

a. Kasus Uji Berhasil Menulis Resep

Kasus uji berhasil menulis resep akan dijelaskan pada Tabel 6.61.

Tabel 6.61 Kasus Uji Berhasil Menulis Resep

Nama Kasus Uji	Kasus uji berhasil menulis resep
Prosedur	1. Memilih menu Konsultasi 2. Menekan tombol Resep Obat 3. Memasukkan resep obat 4. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan menambahkan resep obat ke dalam *database.* Lalu menampilkan kembali halaman konsultasi
Hasil	Sistem menambahkan resep obat ke dalam *database.* Lalu menampilkan kembali halaman konsultasi
Status	Valid

b. Kasus Uji Menulis Resep Tidak Lengkap

Kasus uji menulis resep tidak lengkap akan dijelaskan pada Tabel 6.62.

Tabel 6.62 Kasus Uji Menulis Resep Tidak Lengkap

Nama Kasus Uji	Kasus uji menulis resep tidak lengkap
Prosedur	1. Memilih menu Konsultasi 2. Menekan tombol Resep Obat 3. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan “Kotak

	ini tidak boleh kosong" pada kotak yang kosong
Hasil	Sistem menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Status	Valid

c. Kasus Uji Gagal Menulis Resep

Kasus uji gagal menulis resep obat akan dijelaskan pada Tabel 6.63.

Tabel 6.63 Kasus Uji Gagal Menulis Resep

Nama Kasus Uji	Kasus uji gagal menulis resep
Prosedur	1. Memilih menu Konsultasi 2. Menekan tombol Resep Obat 3. Memasukkan resep obat 4. Menekan tombol Simpan pada saat data pribadi pasien pada halaman konsultasi bernilai nol
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Tidak dapat memasukkan data. Karena tidak ada pasien yang konsultasi."
Hasil	Sistem menampilkan pesan kesalahan "Tidak dapat memasukkan data. Karena tidak ada pasien yang konsultasi."
Status	Valid

6.3.12 Pengujian Menambah Antrian

Pengujian validasi menambah antrian dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_1200 dan MCF_1201. Prosedur menambah antrian akan dijelaskan pada Tabel 6.64.

Tabel 6.64 Kasus Uji Menambah Antrian

Nama Kasus Uji	Kasus uji menambah antrian
Prosedur	1. Memilih menu Antrian 2. Menekan tombol Tambah pada baris pasien yang ingin ditambahkan ke antrian
Hasil yang diharapkan	Sistem akan menampilkan pasien pada daftar antrian pada halaman Antrian
Hasil	Sistem menampilkan pasien pada daftar antrian pada halaman Antrian
Status	Valid

6.3.13 Pengujian Menghapus Antrian

Pengujian validasi menghapus antrian dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_1300 dan MCF_1301. Prosedur kasus uji menghapus antrian akan dijelaskan pada Tabel 6.65.

Tabel 6.65 Kasus Uji Menghapus Antrian

Nama Kasus Uji	Kasus uji menghapus antrian
Prosedur	1. Memilih menu Antrian 2. Menekan tombol Hapus pada baris pasien yang ingin dihapus dari antrian
Hasil yang diharapkan	Sistem akan mengirimkan pesan *SMS* bahwa nomor antrian berubah kepada pasien yang nomor antriannya lebih besar dari nomor antrian yang dihapus. Lalu menampilkan halama antrian
Hasil	Sistem akan mengirimkan pesan *SMS* bahwa nomor antrian berubah kepada pasien yang nomor antriannya lebih besar dari nomor antrian yang dihapus. Lalu menampilkan halama antrian
Status	Valid

6.3.14 Pengujian Mengubah Urutan Antrian

Pengujian validasi mengubah urutan antrian dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_1400 dan MCF_1401. Prosedur kasus uji mengubah urutan antrian akan dijelaskan pada Tabel 6.66.

Tabel 6.66 Kasus Uji Mengubah Urutan Antrian

Nama Kasus Uji	Kasus uji mengubah urutan antrian
Prosedur	1. Memilih menu Antrian 2. Menekan tombol ikon panah pada baris pasien yang ingin diubah urutannya
Hasil yang diharapkan	Sistem akan mengubah urutan antrian sesuai dengan panah yang ditekan oleh perawat
Hasil	Sistem mengubah urutan antrian sesuai dengan panah yang ditekan oleh perawat
Status	Valid

6.3.15 Pengujian Memajukan Urutan Antrian

Pengujian validasi memajukan urutan antrian dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_1500 dan MCF_1501. Terdapat dua kasus uji yaitu berhasil memajukan urutan antrian dan gagal memajukan urutan antrian. Berikut ini adalah prosedur kasus uji yang dilakukan.

a. Kasus Uji Berhasil Memajukan Urutan Antrian

Kasus ujii berhasil memajukan urutan antrian akan dijelaskan pada Tabel 6.67.

Tabel 6.67 Kasus Uji Berhasil Memajukan Urutan Antrian

Nama Kasus Uji	Kasus uji berhasil memajukan urutan antrian
Prosedur	1. Memilih menu Antrian 2. Menekan tombol Selanjutnya
Hasil yang diharapkan	Sistem akan mengubah status antrian pasien menjadi "konsultasi" pada pasien yang akan konsultasi selanjutnya dan mengubah status antrian pasien yang sedang konsultasi sebelumnya menjadi "selesai". Sistem akan mengirimkan nomor urut pasien yang sedang diperiksa oleh dokter kepada seluruh pasien yang masih menunggu antrian melalui *SMS*
Hasil	Sistem mengubah status antrian pasien menjadi "konsultasi" pada pasien yang akan konsultasi selanjutnya dan mengubah status antrian pasien yang sedang konsultasi sebelumnya menjadi "selesai". Sistem akan mengirimkan nomor urut pasien yang sedang diperiksa oleh dokter kepada seluruh pasien yang masih menunggu antrian melalui *SMS*
Status	Valid

b. Kasus Uji Gagal Memajukan Urutan Antrian

Kasus uji gagal memajukan urutan antrian akan dijelaskan pada Tabel 6.68.

Tabel 6.68 Kasus Uji Gagal Memajukan Urutan Antrian

Nama Kasus Uji	Kasus uji gagal memajukan urutan antrian
Prosedur	1. Memilih menu Antrian 2. Menekan tombol Selanjutnya ketika tidak ada pasien dalam antrian yang berstatus menunggu
Hasil yang diharapkan	Sistem akan menampilkan pesan "Antrian sudah habis."
Hasil	Sistem menampilkan pesan "Antrian sudah habis."
Status	Valid

6.3.16 Pengujian Menentukan Kuota Antrian

Pengujian validasi menetukan kuota antrian dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_1600. Terdapat dua kasus uji yaitu

berhasil menentukan kuota antrian dan gagal menentukan kuota antrian. Berikut ini adalah prosedur kasus uji yang dilakukan.

a. Kasus Uji Berhasil Menentukan Kuota Antrian

Kasus uji berhasiil menentukan kuota antrian akan dijelaskan pada Tabel 6.69.

Tabel 6.69 Kasus Uji Berhasil Menentukan Kuota Antrian

Nama Kasus Uji	Kasus uji berhasil menentukan kuota antrian
Prosedur	1. Memilih menu Antrian 2. Menekan tombol Pendaftaran Online 3. Memasukkan kuota 4. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan mengubah kuota pendaftaran pada *database* sesuai dengan kuota pendaftaran yang dimasukkan oleh perawat. Kemudian sistem akan menampilkan halaman Antrian
Hasil	Sistem mengubah kuota pendaftaran pada *database* sesuai dengan kuota pendaftaran yang dimasukkan oleh perawat. Kemudian sistem akan menampilkan halaman Antrian
Status	Valid

b. Kasus Uji Gagal Menentukan Kuota Antrian

Kasus uji gagal menentukan kuota antrian akan dijelaskan pada Tabel 6.70.

Tabel 6.70 Kasus Uji Gagal Menentukan Kuota Antrian

Nama Kasus Uji	Kasus uji gagal menentukan kuota antrian
Prosedur	1. Memilih menu Antrian 2. Menekan tombol Pendaftaran Online 3. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan “Kotak ini tidak boleh kosong” pada kotak kuota
Hasil	Sistem menampilkan pesan kesalahan “Kotak ini tidak boleh kosong” pada kotak kuota
Status	Valid

6.3.17 Pengujian Melihat Resep

Pengujian validasi melihat resep dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_1700. Terdapat dua kasus uji yaitu melihat seluruh permintaan resep dan melihat detail permintaan resep. Berikut ini adalah prosedur kasus uji yang dilakukan.

a. Kasus Uji Melihat Seluruh Permintaan Resep

Kasus uji melihat seluruh permintaan resep akan dijelaskan pada Tabel 6.71.

Tabel 6.71 Kasus Uji Melihat Seluruh Permintaan Resep

Nama Kasus Uji	Kasus uji melihat seluruh permintaan resep
Prosedur	1. Memilih menu Resep Obat
Hasil yang diharapkan	Sistem akan menampilkan seluruh daftar resep obat yang diminta
Hasil	Sistem menampilkan seluruh daftar resep obat yang diminta
Status	Valid

b. Kasus Uji Melihat Detail Permintaan Resep

Kasus uji melihat detail permintaan resep akan dijelaskan pada Tabel 6.72.

Tabel 6.72 Kasus Uji Melihat Detail Permintaan Resep

Nama Kasus Uji	Kasus uji melihat detail permintaan resep
Prosedur	1. Memilih menu Resep Obat 2. Menekan tombol Lihat pada baris resep yang ingin dilihat
Hasil yang diharapkan	Sistem akan menampilkan halaman Detail Resep yang berisi resep obat yang dipilih oleh apoteker
Hasil	Sistem menampilkan halaman Detail Resep yang berisi resep obat yang dipilih oleh apoteker
Status	Valid

6.3.18 Pengujian Memberitahu Resep Selesai

Pengujian validasi memberitahu resep selesai dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_1800. Berikut ini adalah prosedur kasus uji yang dilakukan dalam pengujian validasi memberitahu resep selesai pada Tabel 6.73.

Tabel 6.73 Kasus Uji Memberitahu Resep Selesai

Nama Kasus Uji	Kasus uji memberitahu resep selesai
Prosedur	1. Memilih menu Resep Obat 2. Menekan tombol Lihat pada baris resep yang sudah selesai 3. Menekan tombol Selesai
Hasil yang diharapkan	Sistem akan mengirim pesan *SMS* kepada pasien bahwa obatnya sudah selesai dibuat dan menghapus resep obat tersebut dari *databse.* Kemudian sistem

	menampilkan halaman Resep Obat
Hasil	Sistem mengirim pesan *SMS* kepada pasien bahwa obatnya sudah selesai dibuat dan menghapus resep obat tersebut dari *databse.* Kemudian sistem menampilkan halaman Resep Obat
Status	Valid

6.3.19 Pengujian Menambah Pengguna

Pengujian validasi menambah pengguna dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_1900 dan MCF_1901. Terdapat tiga kasus uji yaitu berhasil menambah pengguna, menambah pengguna dengan username yang sudah digunakan, dan data menambah pengguna tidak lengkap. Berikut adalah prosedur kasus uji yang akan dilakukan pada pengujian validasi.

a. Kasus Uji Berhasil Menambah Pengguna

Kasus uji berhasil menambah pengguna akan dijelaskan pada Tabel 6.74.

Tabel 6.74 Kasus Uji Berhasil Menambah Pengguna

Nama Kasus Uji	Kasus uji berhasil menambah pengguna
Prosedur	1. Memilih tombol Tambah 2. Memasukkan nama, *username*, *password,* status, alamat, dan nomor *handphone* 3. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan menambahkan data pengguna ke dalam *database* lalu menampilkan halaman awal admin yang menampilkan data pengguna seperti nama, *username, password,* status, alamat, dan nomor *handphone*
Hasil	Sistem menambahkan data pengguna ke dalam *database* lalu menampilkan halaman awal admin yang menampilkan data pengguna seperti nama, *username, password,* status, alamat, dan nomor *handphone*
Status	Valid

b. Kasus Uji Menambah Pengguna Dengan Username yang Sudah Digunakan

Kasus uji menambah pengguna dengan username yang sudah digunakan akan dijelaskan pada Tabel 6.75.

Tabel 6.75 Kasus Uji Menambah Pengguna Dengan Username yang Sudah Dipakai

Nama Kasus Uji	Kasus uji menambah pengguna dengan username

	yang sudah digunakan
Prosedur	1. Memilih tombol Tambah 2. Memasukkan nama, *username* yang suda digunakan oleh akun lain, *password,* status, alamat, dan nomor *handphone* 3. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan langsung menampilkan tulisan "x".
Hasil	Sistem langsung menampilkan tulisan "x".
Status	Valid

c. Kasus Uji Data Menambah Pengguna Tidak Lengkap

Kasus uji data menambah pengguna tidak lengkap akan dijelaskan pada Tabel 6.76.

Tabel 6.76 Kasus Uji Data Menambah Pengguna Tidak Lengkap

Nama Kasus Uji	Kasus uji data menambah pengguna tidak lengkap
Prosedur	1. Memilih tombol Tambah 2. Memasukkan hanya beberapa data yang diminta saja 3. Menekan tombol Simpan
Hasil yang diharapkan	sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Hasil	sistem menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Status	Valid

6.3.20 Pengujian Menghapus Pengguna

Pengujian validasi menghapus pengguna dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_2000. Tabel 6.77 berikut adalah prosedur yang dilakukan dalam melakukan pengujian validasi.

Tabel 6.77 Kasus Uji Menghapus Pengguna

Nama Kasus Uji	Kasus uji menghapus pengguna
Prosedur	1. Memilih tombol Hapus pada baris pengguna yang ingin dihapus
Hasil yang diharapkan	Sistem akan menghapus data pengguna dari *database* lalu menampilkan halaman awal admin yang menampilkan data pengguna seperti nama, *username, password,* status, alamat, dan nomor *handphone*

Hasil	Sistem menghapus data pengguna dari *database* lalu menampilkan halaman awal admin yang menampilkan data pengguna seperti nama, *username, password,* status, alamat, dan nomor *handphone*
Status	Valid

6.3.21 Pengujian Mengubah Pengguna

Pengujian validasi mengubah pengguna dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_2100 dan MCF_2101. Terdapat tiga kasus uji yaitu berhasil mengubah pengguna, mengubah pengguna dengan username yang sudah digunakan, dan data mengubah pengguna tidak lengkap. Berikut adalah prosedur kasus uji yang akan dilakukan pada pengujian validasi.

a. Kasus Uji Berhasil Mengubah Pengguna

Kasus uji berhasil mengubah pengguna akan dijelaskan pada Tabel 6.78.

Tabel 6.78 Kasus Uji Berhasil Mengubah Pengguna

Nama Kasus Uji	Kasus uji berhasil mengubah pengguna
Prosedur	1. Memilih tombol Ubah pada baris pasien yang ingin diubah 2. Memasukkan nama, *username*, *password,* status, alamat, dan nomor *handphone* yang baru 3. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan menyimpan data pengguna terbaru ke dalam *database* lalu menampilkan halaman awal admin yang menampilkan data pengguna seperti nama, *username, password,* status, alamat, dan nomor *handphone*
Hasil	Sistem akan menyimpan data pengguna terbaru ke dalam *database* lalu menampilkan halaman awal admin yang menampilkan data pengguna seperti nama, *username, password,* status, alamat, dan nomor *handphone*
Status	Valid

b. Kasus Uji Mengubah Pengguna Dengan Username yang Sudah Digunakan

Kasus uji mengubah pengguna dengan username yang sudah digunakan akan dijelaskan pada Tabel 6.79 dibawah ini.

Tabel 6.79 Kasus Uji Mengubah Pengguna Dengan Username yang Sudah Dipakai

Nama Kasus Uji	Kasus uji mengubah pengguna dengan username

	yang sudah digunakan
Prosedur	1. Memilih tombol Ubah 2. Memasukkan nama, *username* yang telah digunakan oleh akun lain yang suda digunakan oleh akun lain, *password,* status, alamat, dan nomor *handphone* 3. Menekan tombol Simpan
Hasil yang diharapkan	Sistem akan langsung menampilkan tulisan "x".
Hasil	Sistem langsung menampilkan tulisan "x".
Status	Valid

c. Kasus Uji Data Mengubah Pengguna Tidak Lengkap

Kasus uji data mengubah pengguna tidak lengkap akan dijelaskan pada Tabel 6.80.

Tabel 6.80 Kasus Uji Data Mengubah Pengguna Tidak Lengkap

Nama Kasus Uji	Kasus uji data mengubah pengguna tidak lengkap
Prosedur	1. Memilih tombol Ubah 2. Memasukkan hanya beberapa data yang diminta saja 3. Menekan tombol Simpan
Hasil yang diharapkan	sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Hasil	sistem menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Status	Valid

6.3.22 Pengujian Mengingat Password

Pengujian validasi mengingat *password* dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_2200, MCF_22001, dan MCF_2202. Terdapat tiga kasus uji yaitu berhasil mengingat *password*, gagal mengingat *password*, dan data mengingat *password* tidak lengkap. Berikut adalah prosedur kasus uji yang akan dilakukan pada pengujian validasi.

a. Kasus Uji Berhasil Mengingat Password

Kasus uji berhasil mengingat password akan dijelaskan pada Tabel 6.81.

Tabel 6.81 Kasus Uji Berhasil Mengingat Password

Nama Kasus Uji	Kasus uji berhasil mengubah pengguna
Prosedur	1. Memilih tombol Lupa *Password* 2. Memasukkan nama, nama ayah, nama ibu, dan

	nomor *handphone* 3. Menekan tombol Kirim
Hasil yang diharapkan	Sistem akan mengirimkan *username* dan *password* kepada pasien melalui pesan *SMS*
Hasil	Sistem mengirimkan *username* dan *password* kepada pasien melalui pesan *SMS*
Status	Valid

b. Kasus Uji Gagal Mengingat Password

Kasus uji gagal mengingat password akan dijelaskan pada Tabel 6.82.

Tabel 6.82 Kasus Uji Gagal Mengingat Password

Nama Kasus Uji	Kasus uji gagal mengingat password
Prosedur	1. Memilih tombol Lupa *Password* 2. Memasukkan nama, nama ayah, nama ibu, dan nomor *handphone* yang tidak sesuai 3. Menekan tombol Kirim
Hasil yang diharapkan	sistem akan menampilkan pesan kesalahan "Data yang Anda masukkan tidak ditemukan. Silahkan memasukkan data diri Anda dengan Benar."
Hasil	sistem menampilkan pesan kesalahan "Data yang Anda masukkan tidak ditemukan. Silahkan memasukkan data diri Anda dengan Benar."
Status	Valid

c. Kasus Uji Data Mengubah Pengguna Tidak Lengkap

Kasus uji data mengubah pengguna tidak lengkap akan dijelaskan pada Tabel 6.83.

Tabel 6.83 Kasus Uji Data Mengubah Pengguna Tidak Lengkap

Nama Kasus Uji	Kasus uji data mengubah pengguna tidak lengkap
Prosedur	1. Memilih tombol Lupa *Password* 2. Memasukkan beberapa data yang diminta 3. Menekan tombol Kirim
Hasil yang diharapkan	Sistem akan menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Hasil	Sistem menampilkan pesan kesalahan "Kotak ini tidak boleh kosong" pada kotak yang kosong
Status	Valid

6.3.3 Pengujian Membuat Cadangan Rekam Medis

Pengujian validasi melihat rekam medis pasien dilakukan untuk menguji kebutuhan fungsional dengan kode MCF_2300. Kasus uji membuat cadangan rekam medis akan dijelaskan pada Tabel 6.84.

Tabel 6.84 Kasus Uji Membuat Cadangan Rekam Medis

Nama Kasus Uji	Kasus uji membuat cadangan rekam medis pasien
Prosedur	1. Memilih tombol Backup
Hasil yang diharapkan	Sistem akan mengunduh rekam medis pasien dalam format .sql
Hasil	Sistem mengunduh rekam medis pasien dalam format .sql
Status	Valid

6.4 Pembahasan Hasil Pengujian

Hasil dari pengujian unit, integrasi, dan validasi adalah seluruhnya valid. Dari hasil pengujian tersebut, sistem ini layak digunakan untuk mengelola data pasien. Pengujian validasi mendaftar konsultasi menghasilkan hasil yang valid, dan itu membuktikan bahwa sistem ini dapat membantu pasien mendaftar konsultasi secara *online*. Pengujian validasi mengisi rekam medis menghasilkan hasil yang valid, sehingga membuktikan bahwa sistem ini dapat membantu dokter dalam melakukan pencatatan rekam medis. Kebutuhan membuat cadangan rekam medis yang telah diuji pada pengujian validasi juga menghasilkan hasil yang valid, sehingga terbukti bahwa sistem ini dapat memberikan keamanan dalam penyimpanan data pasien.

BAB 7 PENUTUP

7.1 Kesimpulan

Berdasarkan hasil analisis kebutuhan, perancangan, implementasi, dan pengujian yang telah dilakukan, maka dapat diambil kesimpulan sebagai berikut:

1. Berdasarkan hasil analisis kebutuhan, sistem manajemen data pasien klinik gigi ini memiliki 23 kebutuhan fungsional yang dapat membantu proses pengelolaan data pasien di klinik Drg. Damayanti. Kebutuhan fungsional yang dapat menyelesaikan permasalahan keamanan penyimpanan rekam medis pasien, pencatatan rekam medis, dan pendaftaran konsultasi adalah kebutuhan membuat cadangan rekam medis, kebutuhan mengisi rekam medis, dan kebutuhan mendaftar konsultasi.
2. Berdasarkan pada perancangan OO yang telah dilakukan, perancangan menghasilkan *controller class, entity class, boundary class,* detail klas, algoritma, data model, dan antarmuka yang digunakan sebagai acuan untuk membuat sistem yang dapat mengelola data pasien. Dalam pembuatan *database,* tabel-tabel yang terbentuk didasarkan pada *entity class.* Sehingga *entity class* dengan tabel memiliki jumlah yang sama. Berdasarkan hasil implementasi yang dilakukan sistem ini telah menghasilkan fitur yang dapat membantu mengelola data pasien yaitu fitur mendaftar konsultasi, mengelola rekam medis, mengelola resep obat, dan mengelola antrian.
3. Berdasarkan hasil pengujian yang telah dilakukan dengan menggunakan teknik pengujian *white box testing* yang dilakukan pada pengujian unit dan integrasi, tidak terdapat kesalahan yang terjadi selama pengujian dilakukan. Begitu pun juga dengan pengujian yang dilakukan dengan menggunakan teknik pengujian *black box testing* untuk pengujian validasi yang dilakukan pada seluruh kebutuhan. Hal itu membuktikan bahwa sistem ini dapat digunakan untuk melakukan manajemen data pasien di klinik Drg. Damayanti Bogor.

7.2 Saran

Saran yang dapat diberikan untuk pengembangan sistem manajemen data pasien klinik gigi berbasis *web* selanjutnya antara lain:

1. Dapat ditambah fitur-fitur yang lebih berguna lagi seperti fitur upload foto *rontgen* dan hasil tes laboratorium, fitur penentuan tindakan medis berdasarkan diagnosa yang ditentukan oleh dokter, dan fitur penentu penyakit gigi berdasarkan foto yang dimasukkan oleh dokter.
2. Aplikasi ini dapat dibangun untuk android
3. Dapat ditambah fitur-fitur yang lebih berguna lagi

DAFTAR PUSTAKA

Daqiqil, I., 2011. *Framework CodeIgniter Sebuah Panduan dan Best Practice.* Pekanbaru: CodeIgniter Lovers Community.

Depkes RI Direktorat Jenderal Pelayanan Medik, 2006. *Pedoman Penyelenggaraan dan Prosedur Rekam Medis Rumah Sakit di Indonesia Revisi II.* Jakarta: Direktorat Jenderal Bina Pelayanan Medik.

Fowler, M., 2003. *UML Distilled Third Edition A Brief Guide to the Standard Object Modeling Language.* Boston: Addison Wesley.

Gaol, J. L., 2008. *Sistem Informasi Manajemen Pemahaman dan Aplikasi.* [e-book] Jakarta: Grasindo. Tersedia melalui: Google Books <http://booksgoogle.co.id> [Diakses 6 Desember 2015].

Muhadkly, 2007. *SMS Gateway Menggunakan Gammu.* [Online] Tersedia melalui: http://ilmukomputer.org/2007/09/27/sms-gateway-menggunakan-gammu/ [Diakses 9 November 2015].

Nurlela, F., 2013. Aplikasi SMS Gateway Sebagai Sarana Penunjang Informasi Perpustakaan Pada Sekolah Menengah Pertama Negeri 1 Arjosari. *Indonesian Journal on Netwoking and Security,* 2(4), pp. 20-25.

Pranata, A., 1997. *Panduan Pemrograman JavaScript.* [e-book] Yogyakarta: ANDI. Tersedia melalui: Google Books <http://booksgoogle.co.id> [Diakses 6 Desember 2015].

Pressman, R. S., 2001. *Software Engineering A Practitioner's Approach.* 5th ed. New York: McGraw-Hill Higher Education.

Solichin, A., 2008. *Pemrograman Web dengan PHP dan MySQL.* Jakarta: Universitas Budi Luhur.

Sommerville, I., 2011. *Software Engineering.* 9th ed. London: Addison-wesley.

Wardana, 2010. *Menjadi Master PHP dengan Framework CodeIgniter.* [e-book] Jakarta: Elex Media Komputindo. Tersedia melalui: Google Books <http://booksgoogle.co.id> [Diakses 7 November 2015].