TUGAS AKHIR – KS141501

PENGEMBANGAN APLIKASI REGISTRASI PASIEN RAWAT JALAN ONLINE UNTUK RUMAH SAKIT UMUM DAERAH BANGIL BERBASIS ANDROID

OUTPATIENT REGISTRATION APPLICATION DEVELOPMENT FOR RUMAH SAKIT UMUM DAERAH BANGIL BASED ON ANDROID

ARIF PRADANA HERYANTARA
NRP 5211100115

Dosen Pembimbing:
Faizal Johan Atletiko, S.Kom., M.T.

Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

TUGAS AKHIR – KS141501

PENGEMBANGAN APLIKASI REGISTRASI PASIEN RAWAT JALAN ONLINE UNTUK RUMAH SAKIT UMUM DAERAH BANGIL BERBASIS ANDROID

ARIF PRADANA HERYANTARA
NRP 5211100115

Dosen Pembimbing
1. Faizal Johan Atletiko, S.Kom., M.T.

Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

TUGAS AKHIR – KS141501

OUTPATIENT REGISTRATION APPLICATION DEVELOPMENT FOR RUMAH SAKIT UMUM DAERAH BANGIL BASED ON ANDROID

ARIF PRADANA HERYANTARA
NRP 5211100115

Supervisor
1. Faizal Johan Atletiko, S.Kom., M.T.

Departement of Information Systems
Faculty of Information Technology and Communication
Institut Teknologi Sepuluh Nopember
Surabaya 2018

LEMBAR PENGESAHAN

PENGEMBANGAN APLIKASI REGISTRASI PASIEN RAWAT JALAN ONLINE UNTUK RUMAH SAKIT UMUM DAERAH BANGIL BERBASIS ANDROID

TUGAS AKHIR
Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh:

ARIF PRADANA HERYANTARA
NRP. 0521 11 4000 0115

Surabaya, 23 Juli 2018

KEPALA
DEPARTEMEN SISTEM INFORMASI

Dr. Ir. Aris Tjahyanto, M.Kom.
NIP 19650310 199102 1 001

LEMBAR PERSETUJUAN

PENGEMBANGAN APLIKASI REGISTRASI PASIEN RAWAT JALAN ONLINE UNTUK RUMAH SAKIT UMUM DAERAH BANGIL BERBASIS ANDROID

Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh:

ARIF PRADANA HERYANTARA
NRP. 0521 11 4000 0115

Disetujui Tim Penguji: Tanggal Ujian: 16 Juli 2018
Periode Wisuda: September 2018

Faizal Johan Atletiko, S.Kom., M.T	**(Pembimbing I)**
Nur Aini Rakhmawati, S.Kom, M.Sc.Eng, Ph.D	**(Penguji I)**
Radityo Prasetianto W., S.Kom., M.Kom.	**(Penguji II)**

PENGEMBANGAN APLIKASI REGISTRASI PASIEN RAWAT JALAN ONLINE UNTUK RUMAH SAKIT UMUM DAERAH BANGIL BERBASIS ANDROID

Nama Mahasiswa : Arif Pradana Heryantara

NRP : 05211140000115

Departemen : Sistem Informasi FTIK-ITS

Dosen Pembimbing :

1. Faizal Johan Atletiko, S.Kom., M.T.

ABSTRAK

Rumah Sakit Umum Daerah (RSUD) Bangil adalah rumah sakit umum terbesar di Kabupaten Pasuruan. Pengunjung RSUD Bangil berasal dari berbagai macam kecamatan yang ada di Kabupaten Pasuruan. Proses registrasi pasien rawat jalan di RSUD Bangil saat ini telah dilakukan secara komputerisasi. Ketika akan melakukan registrasi, para pasien mengantri di lobi registrasi sesuai dengan jenis asuransi. Setelah melakukan registrasi, pasien menuju poli. Dalam sehari, jumlah pasien rawat jalan bisa mencapai sekitar 400-500 orang per hari. Akibat jumlah pasien per hari yang begitu banyak, antrian bisa menjadi sangat panjang, terutama di pagi hari, dan membutuhkan waktu hingga pukul 11.00 (jam buka: 07.00) untuk menyelesaikan antrian di lobi registrasi rawat jalan. Dibutuhkan suatu aplikasi yang dapat mengurangi lama waktu antrian di loket-loket pendaftaran pasien rawat jalan. Metodologi pengembangan aplikasi yang digunakan adalah metodologi waterfall.

Pengembangan aplikasi untuk aplikasi pendaftaran pasien rawat jalan menggunakan bahasa pemrograman java. Aplikasi tersebut adalah aplikasi yang bisa dipasangkan pada device android. Untuk memverifikasi data pasien, dibutuhkan sebuah

aplikasi monitoring yang dapat diakses di browser. Aplikasi monitoring pendaftaran dikembangkan dengan Bahasa pemrograman PHP dan dengan pola MVC. Database yang digunakan adalah database MySQL. Untuk menghubungkan aplikasi pendaftaran pasien dengan database yang ada di MySQL, dikembangkan sebuah API dengan Bahasa pemrograman PHP. API aplikasi dikembangkan dengan arsitektur REST.

Kata kunci : Android, Java, PHP, Registrasi Pasien, Waterfall.

OUTPATIENT REGISTRATION APPLICATION DEVELOPMENT FOR RUMAH SAKIT UMUM DAERAH BANGIL BASED ON ANDROID

Student Name : **Arif Pradana Heryantara**

NRP : **05211140000115**

Departement : **Information System FTIK-ITS**

Supervisors :

1. Faizal Johan Atletiko, S.Kom., M.T.

ABSTRACT

Rumah Sakit Umum Daerah (RSUD) Bangil is the biggest hospital in Pasuruan Regency. RSUD Bangil's visitors comes from various districts in Pasuruan Regency. Nowadays, outpatient registration process in RSUD Bangil has been computerized. When some patients want to register, they will wait in registration lobby according to the insurance type. Having done registration, they will go to the poly. In a day, the number of outpatients can reach 400-500 people. Because of that, the queue in registration lobby will be too long, especially in the morning, and it is often finished until 11 A.M. (opening hour: 7 A.M.). Therefore, an application to decrease the time to register in registration lobby is needed. The application development's methodology is waterfall model.

The development of outpatient registration application uses java programming language. The application can be installed in an android device. To verify patient data, an application to monitor is needed. That application is installed in browser. That monitoring application is developed with PHP programming language and MVC pattern. The database is MySQL database. An API is needed to link registration application and the

database. The API is developed with PHP programming language and REST architecture.

Keywords : Android, Java, Patient Registration, PHP, Waterfall.

KATA PENGANTAR

Segala puji bagi Allah swt. karena berkat rahmat-Nya penulis dapat menyelesaikan tugas akhir dengan judul "Pengembangan Aplikasi Registrasi Pasien Rawat Jalan Online Untuk Rumah Sakit Umum Daerah Bangil Berbasis Android" sebagai salah satu syarat kelulusan pada Departemen Sistem Informasi, Fakultas Teknologi Informasi dan Komunikasi, Institut Teknologi Sepuluh Nopember. Semoga tugas akhir ini dapat bermanfaat bagi para pembaca dan dapat memberikan sumbangsih bagi ilmu pengetahuan.

Terima kasih penulis sampaikan kepada pihak-pihak yang telah mendukung demi tercapainya tujuan tugas akhir ini. Penulis ucapkan terima kasih atas segala dukungan dan bantuan dalam bentuk apapun kepada:

1. Kedua orang tua penulis yang memberikan dukungan dalam bentuk material dan non material.
2. Bapak Faizal Johan Atletiko, S.Kom, M.T selaku dosen pembimbing yang telah membimbing penulis selama pengerjaan tugas akhir ini.
3. Ibu Nur Aini Rakhmawati, S.Kom., M.Sc., Eng. Ph.D dan Bapak Radityo Prasetianto Wibowo, S.Kom, M.Kom selaku dosen penguji yang telah memberikan berbagai masukan untuk menyempurnakan Tugas Akhir ini.
4. Seluruh Dosen Departemen Sistem Informasi ITS yang telah memberikan ilmu pengetahuan yang bermanfaat.
5. Mas Roshi, Mas Octa, Mas Hendrik, dan Rendi dari bagian Pusat Data Elektronik (PDE) RSUD Bangil, serta Mbak Irma, Mbak Evira, dan pihak-pihak lain dari RSUD Bangil atas informasi-informasi dan masukan-masukan yang bermanfaat.

6. Hufadz Izzudin Robbani atas informasi tentang template yang digunakan penulis untuk membuat aplikasi “Monitoring Pendaftaran”.
7. Serta seluruh pihak yang namanya tidak bisa disebutkan satu per satu.

Tugas akhir ini masih jauh dari kata sempurna. Kritik dan saran dari pembaca akan sangat berarti untuk perbaikan ke depan. Apabila ada kekurangan dan salah kata, penulis mohon maaf.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR KODE PROGRAM

BAB I
PENDAHULUAN

Pada bab pendahuluan ini, akan diuraikan proses identifikasi masalah penelitian yang meliputi latar belakang masalah, perumusan masalah, batasan masalah, tujuan tugas akhir, manfaat kegiatan tugas akhir dan relevansi terhadap pengerjaan tugas akhir.

1.1 Latar Belakang

Rumah sakit adalah institusi pelayanan kesehatan bagi masyarakat dengan karateristik tersendiri yang dipengaruhi oleh perkembangan ilmu pengetahuan kesehatan, kemajuan teknologi, dan kehidupan sosial ekonomi masyarakat yang harus tetap mampu meningkatkan pelayanan yang lebih bermutu dan terjangkau oleh masyarakat agar terwujud derajat kesehatan yang setinggi-tingginya. Rumah Sakit adalah institusi pelayanan kesehatan yang menyelenggarakan pelayanan kesehatan perorangan secara paripurna yang menyediakan pelayanan rawat inap, rawat jalan, dan gawat darurat. [1] Tugas akhir ini hanya akan membahas pelayanan rawat jalan, terutama registrasi pasien rawat jalan.

Rumah Sakit Umum Daerah (RSUD) Bangil telah memiliki sebuah sistem informasi manajemen rumah sakit (SIMRS). SIMRS tersebut memiliki beberapa macam modul dan submodul, salah satunya adalah registrasi. Sebelum diberikan pelayanan rawat jalan, pasien terlebih dahulu melakukan registrasi di lobi registrasi. Pada lobi registrasi, terdapat beberapa loket sesuai dengan jenis asuransi.

Saat melakukan registrasi, pasien perlu untuk melakukan verifikasi berkas-berkas yang berupa KTP, nomor BPJS, nomor rujukan(bagi pasien rujukan dari puskesmas) dan nomor kontrol(bagi pasien lama yang hendak melakukan kontrol), sementara petugas registrasi perlu untuk melakukan input data

ke SIMRS sesuai dengan berkas-berkas pasien. Proses input data yang dilakukan petugas registrasi tentu memakan waktu. Antrian registrasi menjadi sangat panjang karena jumlah pasien rawat jalan mencapai sekitar 400 hingga 500 orang per hari. Pasien juga perlu untuk mengambil nomor antrian agar pasien yang datang lebih awal bisa dilayani terlebih dahulu. Dibutuhkan waktu hingga pukul 11.00 (jam buka: 07.00) untuk menyelesaikan seluruh antrian di lobi registrasi.

Android merupakan salah satu sistem operasi yang paling banyak digunakan di Indonesia. Pada tahun 2016, pangsa pasar android di Indonesia merupakan yang tertinggi di antara sistem operasi lainnya. [2]

Pengguna sistem operasi android di Indonesia juga cenderung berkembang dari tahun ke tahun. Tercatat pada tahun 2014, pengguna sistem operasi android masih kalah dari pengguna sistem operasi windows. Namun, pada tahun 2015 hingga 2016, jumlah pengguna sistem operasi android di Indonesia terus bertambah hingga akhirnya menjadi lebih banyak daripada pengguna sistem operasi windows. [3]

Sistem operasi android paling sering digunakan melalui telepon selular dan komputer tablet. Sepanjang tahun 2016, android menguasai pangsa pasar komputer tablet dan telepon selular. [4], [5]

Untuk menyelesaikan permasalahan lamanya waktu antrian di loket pendaftaran pasien rawat jalan yang dialami oleh RSUD Bangil, dibutuhkan suatu aplikasi registrasi pasien rawat jalan yang dapat diakses secara mandiri dan bersamaan (simultan) sehingga pasien rawat jalan bisa segera menuju ke poli. Pada tugas akhir ini, aplikasi tersebut berbasis android. Android dipilih karena merupakan sistem operasi mobile yang paling sering digunakan di Indonesia.

Aplikasi akan terdiri dari dua bagian. Aplikasi yang pertama adalah aplikasi registrasi berbasis android yang digunakan oleh pasien rawat jalan. Aplikasi kedua adalah aplikasi monitoring berbasis website yang digunakan oleh petugas registrasi untuk

melakukan verifikasi data dengan berkas-berkas pasien. Ketika pasien melakukan registrasi, data pasien akan diterima oleh API yang akan dipasang di sebuah server milik RSUD Bangil, lalu diproses sehingga data pasien dapat ditampilkan di aplikasi monitoring yang ada pada petugas registrasi. Ketika melakukan verifikasi, petugas registrasi mengecek kecocokan data antara data yang dimasukkan oleh pasien dengan berkas-berkas pasien. Setelah semua datanya cocok, pasien bisa mengantri ke poli yang hendak dituju.

1.2 Perumusan Masalah

Berdasarkan uraian latar belakang, maka rumusan permasalahan yang menjadi fokus dan akan diselesaikan dalam Tugas Akhir ini adalah cara membuat suatu aplikasi untuk memudahkan pendaftaran pasien rawat jalan.

1.3 Batasan Masalah

Pengerjaan tugas akhir ini memiliki beberapa batasan masalah / Ruang Lingkup sebagai berikut :

1. Studi kasus yakni RSUD Bangil.
2. Aplikasi utama akan dikembangkan berbasis android dengan alat pengembangan Android Studio versi 2.3.1.
3. API dan aplikasi monitoring akan dikembangkan dengan bahasa pemrogramam PHP 5.3.1 dan akan dipasang pada komputer dengan XAMPP versi 1.7.3.
4. Aplikasi utama dikembangkan hingga mendukung android API minimal versi 21.
5. Spesifikasi perangkat mobile harus bisa mengakses internet.

1.4 Tujuan Penelitian

Berdasarkan hasil perumusan masalah dan batasan masalah yang telah disebutkan sebelumnya, maka tujuan yang dicapai dari tugas akhir ini adalah :

1. Membuat sebuah aplikasi pendaftaran pasien rawat jalan untuk Rumah Sakit Umum Daerah Bangil berbasis android.
2. Memberikan sumbangsih bagi ilmu pengetahuan.

1.5 Manfaat Kegiatan Tugas Akhir

Manfaat dari implementasi tugas akhir ini antara lain :

1. Memudahkan pasien ketika melakukan registrasi.
2. Menjadi landasan utama bagi RSUD Bangil dalam pengembangan aplikasi registrasi berbasis android.

1.6 Relevansi

Tugas akhir ini berkaitan dengan mata kuliah Analisis dan Desain Perangkat Lunak, Konstruksi dan Pengujian Perangkat Lunak, Interaksi Manusia dan Komputer, dan Pemrograman Perangkat Bergerak.

BAB II
TINJAUAN PUSTAKA

Bab ini akan menjelaskan mengenai penelitian sebelumnya dan dasar teori yang dijadikan acuan atau landasan dalam pengerjaan tugas akhir ini. Landasan teori akan memberikan gambaran secara umum dari landasan penjabaran tugas akhir ini.

2.1 Penelitian Sebelumnya

Aplikasi registrasi pasien berbasis android telah diteliti sebelumnya oleh berbagai peneliti. Adapun daftar penelitian tersebut dicantumkan dalam tabel berikut.

Tabel 2.1 Penelitian Sebelumnya

Kategori	Uraian
Judul paper 1	Aplikasi mobile berbasis android untuk Administrasi Pemeriksaan Poliklinik Rawat Jalan di RSUD Kota Salatiga [7]
Penulis, tahun	Nanang Aryanto, Fajrian Nur Adnan,M.CS; 2015
Deskripsi penelitian	Pengembangan aplikasi registrasi pasien rawat jalan berbasis android untuk RSUD Kota Salatiga. Aplikasi ini dirancang untuk memberikan kemudahan dalam mengakses dan mendapatkan informasi pelayanan pemeriksaan rawat jalan di RSUD Kota Salatiga, seperti pendaftaran pasien dan jadwal dokter.
Keterkaitan	Pengembangan aplikasi registrasi pasien rawat jalan berbasis android.

Kategori	Uraian
Judul paper 2	Rancang Bangun Aplikasi Pendaftaran Online Jasa Pengobatan Berbasis Multimedia Pada Klinik Utama Siti Aksar Depok [8]
Penulis, tahun	Muhammad Wardianto, Qurotul Aini, M.T., Nur Aeni Hidayah, M. MSI; 2011
Deskripsi penelitian	Pengembangan aplikasi registrasi pasien online berbasis web khusus antrian dokter gigi yang memberikan nomor antrian dan perkiraan waktu giliran.
Keterkaitan	Pengembangan aplikasi registrasi pasien yang memberikan nomor antrian dan perkiraan waktu giliran.
Judul paper 3	Rancang Bangun Aplikasi Administrasi Rawat Jalan pada Klinik Geo Medika [9]
Penulis, tahun	Ongky Anjar Yamanta; 2013
Deskripsi penelitian	Pengembangan aplikasi administrasi pasien rawat jalan untuk dokter umum dan dokter spesialis pada klinik Geo Medika Sidoarjo. Aplikasi mencakup pendaftaran, pemeriksaan, dan pembayaran.
Keterkaitan	Mencakup registrasi pasien rawat jalan.

Perbedaan antara penelitian sebelumnya dengan tugas akhir ini yaitu pada tugas akhir ini aplikasi mencakup berbagai aspek, seperti registrasi, pemberian nomor antrian dan perkiraan waktu giliran, pengecekan jadwal poli dan dokter, serta perekaman riwayat kunjungan pasien ketika mendaftar melalui aplikasi.

Selain itu, pasien yang telah terdaftar pada SIMRS hanya perlu mengisikan nomor rekam medis, alamat *email*, dan kata sandi ketika akan mendaftarkan akun untuk melakukan *login* pada aplikasi.

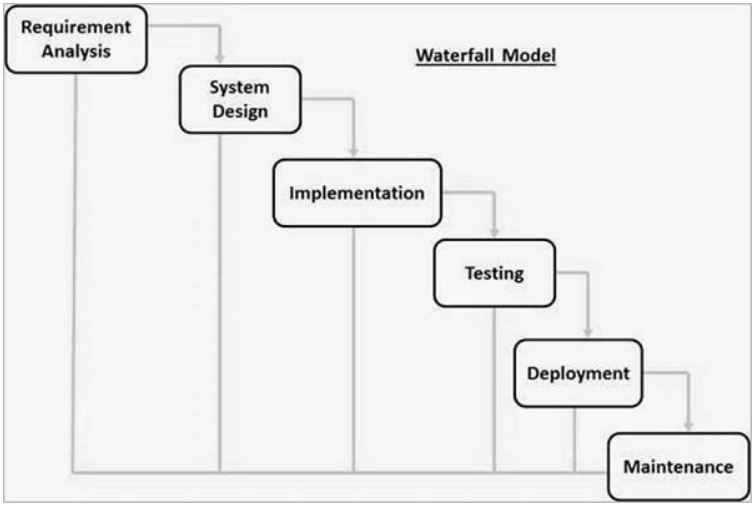
2.2 Dasar Teori

Bagian ini akan membahas teori dan bahan penelitian lain yang menjadi dasar informasi untuk mengerjakan tugas akhir ini.

2.2.1 Waterfall

Model system development life cycle (SDLC) Waterfall adalah model SDLC yang sangat sederhana untuk dipahami dan digunakan. Waterfall adalah pendekatan SDLC paling awal yang digunakan untuk pengembangan perangkat lunak. Waterfall mengilustrasikan proses pengembangan perangkat lunak dalam sebuah alur yang linear dan berurutan. Ini berarti setiap fase dalam proses pengembangan dimulai hanya jika fase sebelumnya telah diselesaikan.

Fase-fase dari model waterfall yang digunakan untuk mengembangkan aplikasi ini adalah sebagai berikut.



Gambar 2.1 Model Waterfall

Requirement Analysis: Semua kemungkinan kebutuhan aplikasi dikembangkan dan didokumentasikan di fase ini.

System Design: Kebutuhan aplikasi dipelajari di fase ini dan desain aplikasi dikembangkan.

Implementation: Dari desain yang telah dikembangkan, aplikasi dikembangkan.

Testing: Aplikasi yang telah dikembangkan kemudian dites fungsi-fungsinya satu per satu. Apabila terdapat kekurangan, aplikasi diperbaiki untuk menutup kekurangan tersebut. Setelah dites satu per satu, aplikasi kemudian dites secara keseluruhan.

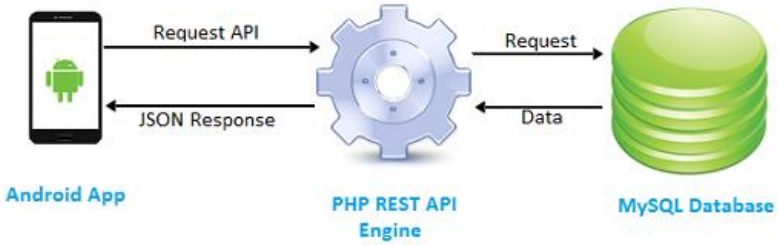
Deployment: Setelah dilakukan pengetesan, aplikasi dipasangkan ke sistem yang dimiliki rumah sakit.

Maintenance: Jika masih ada *bug*, *logical error*, ataupun hal-hal lain yang berkaitan dengan kecocokan aplikasi terhadap lingkungan instansi, maka fungsi-fungsi yang mengandung hal-hal tersebut diperbaiki. Setelah itu, fungsi-fungsi yang telah diperbaiki dipasang kembali ke sistem.[6]

2.2.2 REST

PHP REST API disokong dengan sebuah *database* MySQL adalah skema yang umum dari aplikasi mobile. Arsitektur REST sebaiknya digunakan ketika aplikasi membutuhkan data yang tersimpan dalam sebuah server yang tersentralisasi. Jika data dari aplikasi tidak disimpan dalam sebuah server yang tersentralisasi, maka bisa menggunakan *database* lokal di perangkat *mobile* sebagai tempat penyimpanan dan pengambilan informasi. [10] Pada aplikasi ini, data akan disimpan pada sebuah server yang tersentralisasi.

API untuk aplikasi utama pada tugas akhir ini akan dikembangkan dengan gaya arsitektur REST. Alur pengiriman data dan penerimaan informasi pada arsitektur REST adalah sebagai berikut.



Gambar 2.2 REST API

Sumber gambar: www.phppot.com *[10]*

Aplikasi android mengirimkan *request* http ke PHP REST API. *Request* http dalam aplikasi yang akan dibuat bisa berupa isian dari form registrasi, form *login* (*user credential*), atau data pengguna yang tersimpan dalam *session*. PHP REST API kemudian mengirimkan *query* ke *MySQL Database* berdasarkan pada masukan dari *request* yang diterima. Setelah itu, *MySQL Database* mengirimkan data hasil *query* ke PHP REST API. PHP REST API kemudian mengirimkan *response* dalam bentuk JSON yang berisi data yang diterima dari *database*.

2.2.3 Volley Library

Volley adalah sebuah *library* HTTP yang mempermudah dan mempercepat fungsi jaringan (*networking*) pada android. Volley menawarkan keuntungan-keuntungan sebagai berikut.

- Penjadwalan otomatis dari *network request*.
- Banyak koneksi jaringan secara bersamaan.
- *Disk* yang transparan dan men-*cache* respon memori dengan standar *HTTP cache coherence*.
- Dukungan untuk prioritisasi *request*.
- Pembatalan *request* API.
- Kemudahan dalam kustomisasi.
- Mudah untuk menempatkan UI dengan data yang diambil secara asinkron dari jaringan.
- Alat untuk men-*debug* dan melacak. [11]

Pada tugas akhir ini, *library* Volley akan digunakan untuk mengirimkan *request* HTTP ke API sebelum menuju ke server.

Library ini juga digunakan untuk menerima *request* HTTP dari API setelah mendapatkan data dari server.

2.2.4 Gson

Gson adalah *library* Java yang dapat digunakan untuk mengkonversi obyek-obyek Java menjadi JSON. *Library* ini juga dapat digunakan untuk mengkonversi string-string JSON menjadi obyek-obyek Java.[12]

Pada aplikasi ini, *library* Gson digunakan untuk mengkonversi *input* dari form-form yang diisikan pasien menjadi data-data yang dapat ditampilkan di aplikasi "Monitoring Pendaftaran". *Library* ini juga digunakan untuk mengisi isian *combo box* form-form aplikasi dari database.

2.2.5 SQLite

SQLite adalah sebuah *library* yang mengimplementasikan sebuah mesin database SQL yang mandiri, tanpa server, tanpa konfigurasi, dan transaksional. *Source code* untuk SQLite berada di domain umum dan gratis digunakan untuk berbagai tujuan, baik komersial ataupun pribadi. SQLite adalah database yang paling banyak dikembangkan di dunia.

SQLite adalah sebuah mesin database SQL yang tertanam. Tidak seperti database SQL yang lain, SQLite tidak memiliki sebuah proses server yang terpisah. SQLite membaca dan menulis langsung ke file disk kebanyakan. Sebuah database SQL lengkap dengan berbagai tabel, index, trigger, dan view, termuat dalam dalam sebuah file disk.[13]

Pada tugas akhir ini, SQLite digunakan untuk menyimpan *session login user* sehingga data-data seperti nama dan nomor rekam medis (NRM) bisa digunakan untuk mengirim *request* ke API. Selain itu, juga untuk menyimpan *output* JSON *history* pasien sebelum ditampilkan dalam bentuk tabel di aplikasi.

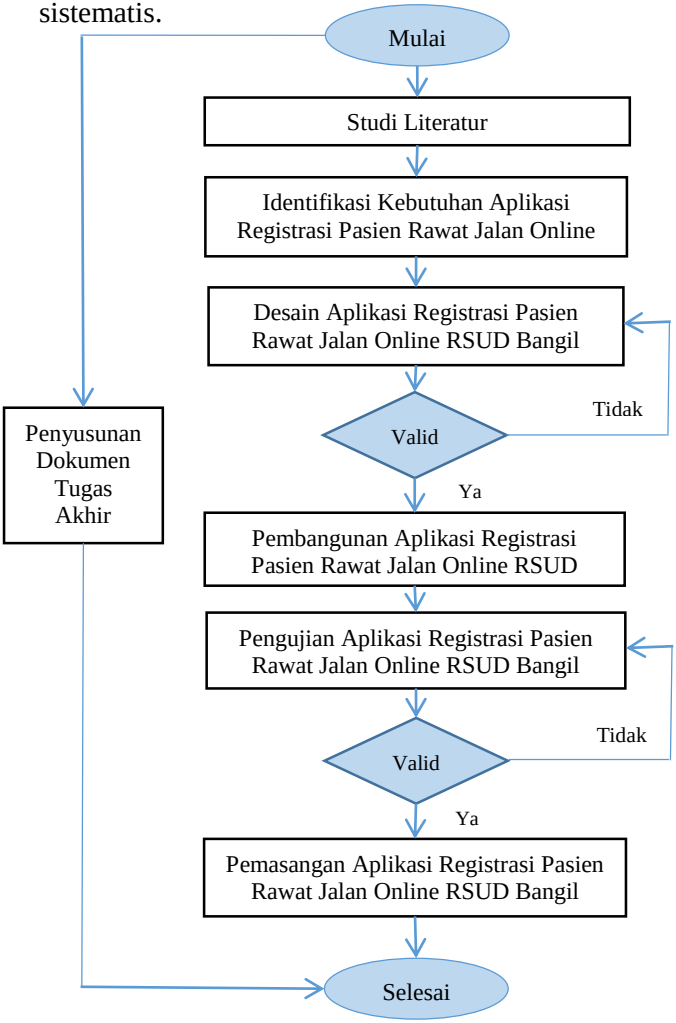
2.2.6 Model-View-Controller

Model-View-Controller, atau yang biasa disingkat MVC, adalah sebuah pola yang populer untuk mengorganisir *source code*. Ide dari MVC adalah setiap bagian kode memiliki sebuah tujuan, dan tujuan-tujuan tersebut berbeda-beda. Sebagian dari *source code* menyimpan data dari aplikasi, sebagian membuat aplikasi terlihat lebih indah, dan sebagian lagi mengendalikan jalannya aplikasi.[14]

Pada tugas akhir ini, aplikasi "Monitoring Pendaftaran" berbasis web dikembangkan dengan pola MVC.

BAB III
METODOLOGI

Bagian ini menjelaskan tentang proses pengerjaan tugas akhir. Metode yang digunakan bertujuan untuk menjadi pedoman, agar pengerjaan tugas akhir dapat berlangsung secara sistematis.



Gambar 3.1 Metodologi Penelitian

3.1 Studi literatur

Tahapan ini bertujuan untuk memahami metodologi dan teknologi yang akan digunakan untuk membuat tugas akhir. Literatur yang akan digunakan berasal dari situs, buku, dan sumber-sumber lainnya.

3.2 Identifikasi Kebutuhan Aplikasi Registrasi Pasien Rawat Jalan Online RSUD Bangil

Tahapan ini bertujuan untuk mencari tahu apa saja yang dibutuhkan untuk membuat aplikasi, observasi sistem yang telah ada, dan observasi infrastruktur yang tersedia di RSUD Bangil. Dari tahapan ini kemudian disusun kebutuhan-kebutuhan fungsional dan non fungsional aplikasi.

3.3 Desain Aplikasi Registrasi Pasien Rawat Jalan Online RSUD Bangil

Tahapan ini bertujuan untuk merancang aplikasi, lalu mendokumentasikannya. Rancangan aplikasi berupa siapa saja aktor yang terlibat, pembuatan use case dan use case description, pembuatan sequence diagram, class dan method apa saja yang ada pada aplikasi, serta pembuatan *graphical user interface* (GUI).

3.4 Pembangunan Aplikasi Registrasi Pasien Rawat Jalan Online RSUD Bangil

Tahapan ini bertujuan untuk mengembangkan aplikasi berdasarkan desain yang telah dibuat pada tahapan sebelumnya. Teknologi yang akan digunakan dalam membuat aplikasi adalah Android, MySQL, SQLite, dan web berbasis PHP dengan kerangka kerja MVC.

3.5 Pengujian Aplikasi Registrasi Pasien Rawat Jalan Online RSUD Bangil

Tahapan ini bertujuan untuk menguji aplikasi. Aplikasi perlu diuji apakah sudah berjalan dengan baik tanpa adanya kesalahan (error) dan sesuai dengan kebutuhan fungsional dan non fungsional sistem. Pengujian akan dilakukan pada perangkat android dengan API versi 22 (Android 5.1) karena akan ada beberapa method dan fitur aplikasi yang hanya bisa berjalan pada perangkat android dengan API versi 22 ke atas.

3.6 Pemasangan Aplikasi Registrasi Pasien Rawat Jalan Online RSUD Bangil

Tahapan ini bertujuan untuk memasang aplikasi "Monitoring Pendaftaran", API aplikasi, beserta pembuatan dan sinkronisasi database pada infrastruktur yang tersedia.

3.7 Penyusunan Dokumen Tugas Akhir

Tahapan ini bertujuan untuk dokumentasi dari proses-proses yang telah dilakukan pada tahapan-tahapan sebelumnya. Dokumentasi disusun dalam bentuk dokumen tugas akhir.

Halaman ini sengaja dikosongkan

3.8 Jadwal Penyusunan Tugas Akhir

Tabel 3.1 Jadwal Penyusunan Tugas Akhir

Kegiatan	Minggu ke-													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Studi Literatur														
Identifikasi Kebutuhan Aplikasi Registrasi Pasien Rawat Jalan Online RSUD Bangil														
Desain Aplikasi Registrasi Pasien Rawat Jalan Online RSUD Bangil														
Pembangunan Aplikasi Registrasi Pasien Rawat Jalan Online RSUD Bangil														
Pengujian Aplikasi Registrasi Pasien Rawat Jalan Online RSUD Bangil														
Pemasangan Aplikasi Registrasi Pasien Rawat Jalan Online RSUD Bangil														
Penyusunan Dokumen Tugas Akhir														

Halaman ini sengaja dikosongkan

BAB IV
PERANCANGAN

4.1 Identifikasi Kebutuhan Aplikasi

Pada bagian ini, proses pendaftaran pasien rawat jalan RSUD Bangil dijelaskan terlebih dahulu.

Proses pendaftaran pasien rawat jalan yang sudah ada adalah sebagai berikut. Pasien datang mengambil nomor antrian, kemudian pasien menunggu giliran di lobi pendaftaran. Setelah waktu gilirannya tiba, nomor antrian pasien akan muncul di sebuah layar monitor dan nomor tersebut dipanggil, kemudian pasien menuju ke salah satu loket pendaftaran. Petugas loket mengecek kelengkapan berkas yang dibawa oleh pasien, lalu memasukkan datanya ke SIMRS. Setelah itu, pasien menuju ke loket pembayaran. Setelah membayar, pasien menuju ke poli tujuan.

Dengan jumlah pasien yang bisa mencapai 750 orang dalam sehari, antrian di lobi pendaftaran bisa menjadi sangat panjang. Selain itu, petugas juga membutuhkan waktu untuk memasukkan data ke SIMRS.

Berdasarkan proses pendaftaran tersebut, dapat dilihat bahwa selama ini proses pendaftaran pasien rawat jalan bisa membuat antrian menjadi sangat lama. Maka dari itu, sebaiknya ada sebuah fasilitas yang memungkinkan pasien untuk memasukkan datanya sendiri sehingga petugas hanya perlu memverifikasi data yang diisikan. Fasilitas tersebut sebaiknya bisa diakses secara simultan. Selain itu, jika waktu giliran pasien bisa diperkirakan, maka pasien bisa meluangkan lebih banyak waktunya di rumah masing-masing, lalu datang ke RS sebelum waktu gilirannya tiba.

4.1.1 Kebutuhan Fungsional Sistem

Berdasarkan skenario alternatif yang dijabarkan sebelumnya, maka disusun kebutuhan fungsional aplikasi pendaftaran pasien rawat jalan *online* yang sudah disesuaikan dengan proses pendaftaran yang sudah dijabarkan sebelumnya. Kebutuhan fungsional sistem ini tertera pada tabel berikut.

Tabel 4.1 Kebutuhan Fungsional Aplikasi

ID	Kebutuhan Fungsional
FR-01	Pasien baru dapat melakukan registrasi akun
FR-02	Pasien lama dapat melakukan registrasi akun
FR-03	Pasien dapat mengelola akun
FR-04	Pasien dapat mendaftar berobat
FR-05	Pasien dapat mengecek jadwal poli dan dokter
FR-06	Pasien dapat mengecek riwayat kunjungan
FR-07	Petugas dapat mengelola akun
FR-08	Petugas dapat mengelola data pasien
FR-09	Pegawai PDE dapat memverifikasi data petugas

4.1.2 Kebutuhan Non-Fungsional Sistem

Dari hasil analisis kebutuhan yang dilakukan sebelumnya, didapatkan kebutuhan non-fungsional yang ada pada tabel berikut.

Tabel 4.2 Kebutuhan Non-Fungsional Aplikasi

ID	Kebutuhan Non-Fungsional	Jenis
NFR-01	Sistem dapat diakses selama 24 jam sehari, 7 hari seminggu	*Accessibility*
NFR-02	Aplikasi "Registrasi Pasien" dapat berjalan pada sistem operasi Android dengan versi API minimal 22.	*Accessibility*
NFR-03	Aplikasi "Monitoring Pendaftaran" dapat berjalan pada *browser*.	*Accessibility*
NFR-04	Sistem harus bisa menyimpan semua data yang dimasukkan pasien.	*Reliability*
NFR-05	Sistem harus bisa menyimpan semua data yang dimasukkan petugas.	*Reliability*

4.2 Desain Aplikasi

Pada tahap desain ini dijelaskan bagaimana aplikasi berjalan. Penjelasan dimulai dari aktor-aktor yang ada dalam sistem, pembuatan *use case* beserta deskripsinya, pembuatan daftar *class* beserta *method* yang akan digunakan pada sistem, serta desain *interface* aplikasi.

4.2.1 Aktor-aktor yang Ada dalam Sistem

Aktor-aktor yang terlibat dalam sistem serta deskripsinya dijelaskan pada tabel berikut.

Tabel 4.3 Aktor-aktor yang Ada dalam Sistem

Aktor	**Deskripsi**	**Aplikasi yang digunakan**
Pasien Baru	Pasien yang belum terdaftar di sistem yang sudah ada sebelumnya. Menggunakan aplikasi untuk mendaftarkan diri.	Registrasi Pasien
Pasien Lama	Pasien yang sudah terdaftar di sistem yang sudah ada sebelumnya. Menggunakan aplikasi untuk mendaftarkan diri dan mengecek apakah nomor rekam medisnya sudah terdaftar di aplikasi atau belum.	Registrasi Pasien
Pasien	Pasien baru dan pasien lama yang sudah terdaftar di aplikasi. Menggunakan aplikasi untuk mendaftar berobat, memperkirakan waktu giliran antrian, mengecek jadwal poli dan dokter, dan mengecek riwayat kunjungan.	Registasi Pasien
Petugas	Petugas loket, orang yang bertugas untuk memverifikasi data yang diisikan pasien. Menggunakan aplikasi untuk mengecek, memverifikasi, mengubah	Monitoring Pendaftaran

	dan menghapus data pasien apabila dibutuhkan.	
Pegawai PDE	Pegawai RSUD Bangil yang bertugas di bagian Pusat Data Elektronik. Memiliki hak akses ke database aplikasi.	MySQL

4.2.2 Use Case

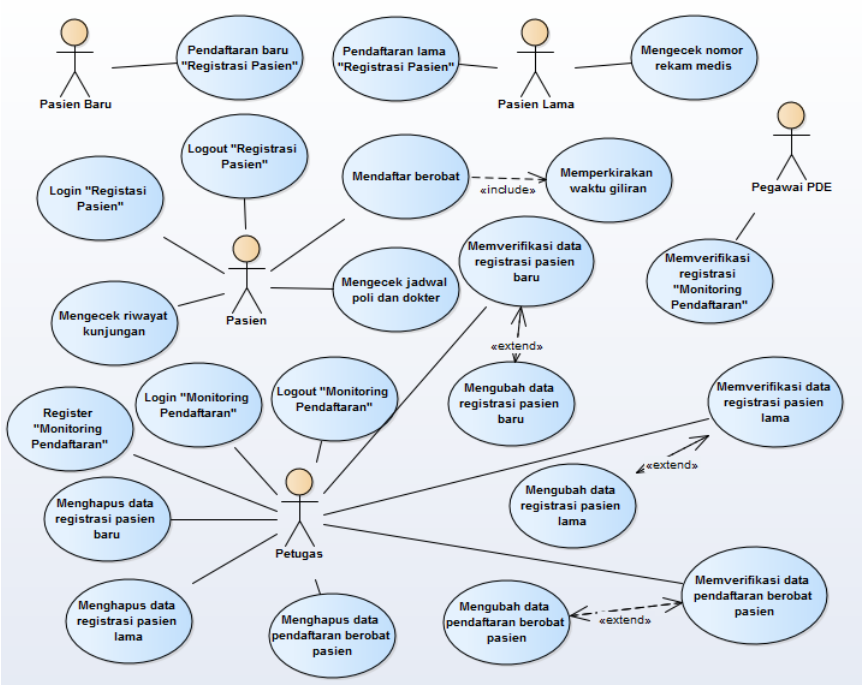
Berdasarkan analisis kebutuhan sistem yang telah dilakukan, fungsi-fungsi sistem dapat direpresentasikan dalam beberapa use case. Daftar use case beserta pemetaannya dijabarkan pada tabel berikut.

Tabel 4.4 Use Case

FR ID	UC ID	*Use Case*
FR-01	UC-01	Pendaftaran baru "Registrasi Pasien"
FR-02	UC-02	Mengecek nomor rekam medis
FR-02	UC-03	Pendaftaran lama "Registrasi Pasien"
FR-03	UC-04	Login "Registrasi Pasien"
FR-04	UC-05	Mendaftar berobat
FR-04	UC-06	Memperkirakan waktu giliran
FR-05	UC-07	Mengecek jadwal poli dan dokter
FR-06	UC-08	Mengecek riwayat kunjungan
FR-03	UC-09	Logout "Registrasi Pasien"
FR-07	UC-10	Register "Monitoring Pendaftaran"

FR-07	UC-11	Login “Monitoring Pendaftaran”
FR-08	UC-12	Memverifikasi data registrasi pasien baru
FR-08	UC-13	Mengubah data registrasi pasien baru
FR-08	UC-14	Menghapus data registrasi pasien baru
FR-08	UC-15	Memverifikasi data registrasi pasien lama
FR-08	UC-16	Mengubah data registrasi pasien lama
FR-08	UC-17	Menghapus data registrasi pasien lama
FR-08	UC-18	Memverifikasi data pendaftaran berobat pasien
FR-08	UC-19	Mengubah data pendaftaran berobat pasien
FR-08	UC-20	Menghapus data pendaftaran berobat pasien
FR-07	UC-21	Logout “Monitoring Pendaftaran”
FR-09	UC-22	Memverifikasi registrasi “Monitoring Pendaftaran”

Use case diagram dapat dilihat pada gambar berikut.



Gambar 4.1 Use Case Diagram

4.2.3 Use Case Description

Deskripsi *use case* pendaftaran baru “Registrasi Pasien” dapat dilihat di tabel berikut.

Tabel 4.5 Deskripsi Use Case Pendaftaran Baru “Registrasi Pasien”

UC-01: Pendaftaran baru “Registrasi Pasien”	
Ringkasan:	Pasien baru mendaftar akun di aplikasi “Registrasi Pasien”.
Aktor:	Pasien Baru

Kondisi:	Pasien baru belum pernah mendaftar baik pada sistem yang sudah ada maupun pada aplikasi "Registrasi Pasien".
Skenario Utama:	1. Pasien baru membuka aplikasi 2. Pasien baru menuju tautan "Belum memiliki nomor rekam medis? Daftar di tautan berikut!" 3. Pasien baru mengisikan form pendaftaran 4. Pasien baru menekan tombol "Daftar Baru", maka sistem akan menyimpan data pendaftaran
Skenario Alternatif:	1. Salah satu isian kosong dan tombol "Daftar Baru" ditekan, maka sistem akan menampilkan pesan error. 2. Pasien baru mengetikkan tanggal lahir, *email*, atau isian konfirmasi *password* yang salah, maka sistem akan menampilkan pesan error. 3. Pasien baru sudah memiliki akun di aplikasi "Registrasi Pasien", maka sistem akan menampilkan pesan error.

Deskripsi *use case* mengecek nomor rekam medis dapat dilihat di tabel berikut.

Tabel 4.6 Deskripsi Use Case Mengecek Nomor Rekam Medis

UC-02: Mengecek nomor rekam medis	
Ringkasan:	Pasien lama mengecek nomor rekam medis di aplikasi "Registrasi Pasien" untuk memastikan apakah sudah pernah mendaftar di sistem yang sudah ada atau belum.
Aktor:	Pasien Lama

Kondisi:	Pasien lama tidak mengetahui apakah sudah pernah mendaftar di sistem yang sudah ada atau belum
Skenario Utama:	1. Pasien lama membuka aplikasi "Registrasi Pasien" 2. Pasien lama menuju tautan "Sudah memiliki nomor rekam medis, tapi belum memiliki akun? Buat akun di tautan berikut!" 3. Pasien lama mengisikan isian nomor rekam medis 4. Pasien lama menekan tombol search.

Deskripsi *use case* pendaftaran lama dapat dilihat di tabel berikut.

Tabel 4.7 Deskripsi Use Case Pendaftaran Lama "Registrasi Pasien"

UC-03: Pendaftaran lama "Registrasi Pasien"	
Ringkasan:	Pasien lama mendaftar akun di aplikasi "Registrasi Pasien"
Aktor:	Pasien Lama
Kondisi:	Pasien lama
Skenario Utama:	1. Pasien lama membuka aplikasi "Registrasi Pasien" 2. Pasien lama menuju tautan "Sudah memiliki nomor rekam medis, tapi belum memiliki akun? Buat akun di tautan berikut!" 3. Pasien lama mengisikan form pendaftaran. 4. Pasien lama menekan tombol "Buat Akun", maka sistem akan menyimpan data pendaftaran.

Skenario Alternatif:	1. Pasien lama salah mengisikan *email* atau isian konfirmasi *password*, maka sistem akan menampilkan pesan error. 2. Pasien lama sudah memiliki akun di aplikasi "Registrasi Pasien", maka sistem akan menampilkan pesan error 3. Pasien lama belum memiliki nomor rekam medis, maka sistem akan menampilkan pesan error.

Deskripsi *use case login* "Registrasi Pasien" dapat dilihat di tabel berikut.

Tabel 4.8 Deskripsi Use Case Login "Registrasi Pasien"

UC-04: Login "Registrasi Pasien"	
Ringkasan:	Pasien masuk ke dalam aplikasi "Registrasi Pasien" sesuai akun yang dimiliki.
Aktor:	Pasien
Kondisi:	Pasien telah memiliki akun di aplikasi "Registrasi Pasien" yang telah terverfikasi.
Skenario Utama:	1. Pasien membukan aplikasi "Registrasi Pasien" 2. Pasien mengetikkan nomor rekam medis dan *password* 3. Pasien menekan tombol "Masuk"
Skenario Alternatif:	1. Pasien salah mengisikan *password*, maka sistem akan menampilkan pesan error. 2. Pasien belum memiliki akun, maka sistem akan menampilkan pesan error. 3. Pasien sudah memiliki akun, tapi belum melakukan verifikasi, maka sistem akan menampilkan pesan error.

	4. Pasien belum memiliki nomor rekam medis, maka sistem akan memunculkan pesan error.

Deskripsi *use case* mendaftar berobat dapat dilihat di tabel berikut.

Tabel 4.9 Deskripsi Use Case Mendaftar Berobat

UC-05: Mendaftar berobat	
Ringkasan:	Pasien mendaftar berobat ke RSUD melalui aplikasi "Registrasi Pasien"
Aktor:	Pasien
Kondisi:	Pasien telah login ke dalam aplikasi
Skenario Utama:	1. Pasien memilih menu "Pendaftaran Pasien" 2. Pasien mengisikan form isian pendaftaran 3. Pasien menekan tombol "Mendaftar", maka sistem akan menyimpan data pendaftaran, lalu menampilkan halaman perkiraan waktu giliran beserta nomor antrian
Skenario Alternatif:	1. Pasien salah mengisikan tanggal kunjungan, maka sistem akan menampilkan pesan error 2. Salah satu kotak isian kosong, maka sistem akan menampilkan pesan error

Deskripsi *use case* mengecek poli dan dokter dapat dilihat di tabel berikut.

Tabel 4.10 Deskripsi Use Case Mengecek Jadwal Poli dan Dokter

UC-07: Mengecek jadwal poli dan dokter	
Ringkasan:	Pasien mengecek jadwal poli dan dokter

Aktor:	Pasien
Kondisi:	Pasien telah login ke dalam aplikasi "Registrasi Pasien"
Skenario Utama:	1. Pasien memilih menu "Poli dan Dokter" 2. Pasien memilih poli yang ingin dicek jadwalnya, maka sistem menampilkan jadwal dokter sesuai poli yang dipilih

Deskripsi *use case* mengecek riwayat kunjungan dapat dilihat di tabel berikut.

Tabel 4.11 Deskripsi Use Case Mengecek Riwayat Kunjungan

UC-08: Mengecek riwayat kunjungan	
Ringkasan:	Pasien mengecek riwayat kunjungan
Aktor:	Pasien
Kondisi:	Pasien telah *login* ke dalam aplikasi "Registrasi Pasien"
Skenario Utama:	1. Pasien memilih menu "Riwayat", maka sistem akan menampilkan riwayat kunjungan pasien.

Deskripsi *use case* logout "Registrasi Pasien" dapat dilihat di tabel berikut.

Tabel 4.12 Deskripsi Use Case Logout "Registrasi Pasien"

UC-09: Logout "Registrasi Pasien"	
Ringkasan:	Pasien melakukan logout dari aplikasi "Registrasi Pasien"
Aktor:	Pasien
Kondisi:	Pasien telah *login* ke dalam aplikasi "Registrasi Pasien"

Skenario Utama:	1. Pasien menekan menu yang ada di pojok kanan atas, lalu memilih menu logout. 2. Pasien keluar dari aplikasi.

Deskripsi *use case* register "Monitoring Pendaftaran" dapat dilihat di tabel berikut.

Tabel 4.13 Deskripsi Use Case Register "Monitoring Pendaftaran"

UC-10: Register "Monitoring Pendaftaran"	
Ringkasan:	Petugas melakukan registrasi akun di aplikasi "Monitoring Pendaftaran"
Aktor:	Petugas
Kondisi:	Petugas belum memiliki akun di aplikasi
Skenario Utama:	1. Petugas membuka aplikasi "Monitoring Pendaftaran" 2. Petugas menuju tautan pendaftaran akun 3. Petugas mengisikan form pendaftaran 4. Petugas menekan tombol sign-up, maka sistem akan menyimpan data pendaftaran
Skenario Alternatif:	1. Petugas mengisikan isian *password* kurang dari 6 karakter, maka sistem akan memunculkan error. 2. Petugas salah mengisikan isian konfirmasi *password*, maka sistem akan memunculkan error.

Deskripsi *use case login* "Monitoring Pendaftaran" dapat dilihat di tabel berikut.

Tabel 4.14 Deskripsi Use Case Login "Monitoring Pendaftaran"

UC-11: Login "Monitoring Pendaftaran"	
Ringkasan:	Petugas melakukan login di aplikasi "Monitoring Pendaftaran"

Aktor:	Petugas
Kondisi:	Petugas telah memiliki akun di aplikasi "Monitoring Pendaftaran" yang terverifikasi.
Skenario Utama:	1. Petugas membuka aplikasi "Monitoring Pendaftaran", maka sistem akan menampilkan halaman login. 2. Petugas mengisikan isian username dan *password* yang sesuai. 3. Petugas menekan tombol sign in, maka petugas berhasil masuk ke beranda aplikasi.
Skenario Alternatif:	1. Petugas memiliki akun yang belum terverifikasi, maka sistem akan menampilkan pesan error. 2. Petugas salah mengisikan *password*, maka sistem akan memunculkan pesan error. 3. Petugas belum memiliki akun, maka sistem akan memunculkan pesan error.

Deskripsi *use case* memverifikasi data registrasi pasien baru dapat dilihat di tabel berikut.

Tabel 4.15 Deskripsi Use Case Memverifikasi Data Registrasi Pasien Baru

UC-12: Memverifikasi data registrasi pasien baru	
Ringkasan:	Petugas melakukan verifikasi data registrasi pasien baru di aplikasi "Monitoring Pendaftaran"
Aktor:	Petugas
Kondisi:	Petugas telah *login* ke dalam aplikasi
Skenario Utama:	1. Petugas memilih menu "Pasien Baru", maka sistem akan menampilkan semua data pasien baru.

	2. Petugas memilih pasien yang akan diverifikasi datanya, lalu mengklik tautan "Edit", maka sistem akan menampilkan form edit data pasien baru dan men-generate nomor rekam medis. 3. Petugas mengecek kesesuaian data pasien dengan berkas-berkas yang dibawa oleh pasien. 4. Petugas menekan tombol update, maka sistem akan menyimpan data pendaftaran.
Skenario alternatif:	1. Data yang diisikan pasien tidak sesuai dengan berkas-berkas yang dibawa oleh pasien, maka petugas menyesuaikan data pasien dengan berkas-berkas pasien, lalu menekan tombol update. 2. Petugas menekan tombol cancel, maka sistem tidak menyimpan data pendaftaran.

Deskripsi *use case* menghapus data registrasi pasien baru dapat dilihat di tabel berikut.

Tabel 4.16 Deskripsi Use Case Menghapus Data Registrasi Pasien Baru

UC-14: Menghapus data registrasi pasien baru	
Ringkasan:	Petugas dapat menghapus data registrasi pasien baru.
Aktor:	Petugas
Kondisi:	Petugas telah *login* ke dalam aplikasi
Skenario Utama:	1. Petugas memilih menu "Pasien Baru", maka sistem akan menampilkan semua data pasien baru. 2. Petugas memilih pasien yang akan dihapus datanya, lalu mengklik tautan

	"Delete", maka sistem akan menampilkan form konfirmasi delete data pasien baru. 3. Petugas menekan tombol "Ya", maka data pasien baru terhapus.
Skenario Alternatif:	1. Petugas menekan tombol "Tidak", maka data pasien baru tidak terhapus.

Deskripsi *use case* memverifikasi data registrasi pasien lama dapat dilihat di tabel berikut.

Tabel 4.17 Deskripsi Use Case Memverifikasi Data Registrasi Pasien Lama

UC-15: Memverifikasi data registrasi pasien lama	
Ringkasan:	Petugas melakukan verifikasi data registrasi pasien lama di aplikasi "Monitoring Pendaftaran"
Aktor:	Petugas
Kondisi:	Petugas telah *login* ke dalam aplikasi
Skenario Utama:	1. Petugas memilih menu "Pasien Lama Membuat Akun", maka sistem akan menampilkan semua data pasien lama. 2. Petugas memilih pasien yang akan diverifikasi datanya, lalu mengklik tautan "Edit", maka sistem akan menampilkan form edit data pasien lama. 3. Petugas mengecek kesesuaian data pasien dengan berkas-berkas yang dibawa oleh pasien. 4. Petugas menekan tombol update, maka sistem akan menyimpan data pendaftaran.
Skenario alternatif:	1. Data yang diisikan pasien tidak sesuai dengan berkas-berkas yang dibawa oleh

	pasien, maka petugas menyesuaikan data pasien dengan berkas-berkas pasien, lalu menekan tombol update. 2. Petugas menekan tombol cancel, maka sistem tidak menyimpan data pendaftaran.

Deskripsi *use case* menghapus data registrasi pasien lama dapat dilihat di tabel berikut.

Tabel 4.18 Deskripsi Use Case Menghapus Data Registrasi Pasien Lama

UC-17: Menghapus data registrasi pasien lama	
Ringkasan:	Petugas dapat menghapus data registrasi pasien lama.
Aktor:	Petugas
Kondisi:	Petugas telah *login* ke dalam aplikasi
Skenario Utama:	1. Petugas memilih menu "Pasien Lama Membuat Akun", maka sistem akan menampilkan semua data pasien lama. 2. Petugas memilih pasien yang akan dihapus datanya, lalu mengklik tautan "Delete", maka sistem akan menampilkan form konfirmasi delete data pasien lama. 3. Petugas menekan tombol "Ya", maka data pasien lama terhapus.
Skenario Alternatif:	1. Petugas menekan tombol "Tidak", maka data pasien lama tidak terhapus.

Deskripsi *use case* memverifikasi data pendaftaran berobat pasien dapat dilihat di tabel berikut.

Tabel 4.19 Deskripsi Use Case Memverifikasi Data Pendaftaran Berobat Pasien

UC-18: Memverifikasi data pendaftaran berobat pasien	
Ringkasan:	Petugas melakukan verifikasi data pendaftaran berobat pasien di aplikasi "Monitoring Pendaftaran"
Aktor:	Petugas
Kondisi:	Petugas telah *login* ke dalam aplikasi
Skenario Utama:	1. Petugas memilih menu "Antrian", maka sistem akan menampilkan semua data pendaftaran berobat pasien pada hari itu. 2. Petugas memilih pasien yang akan diverifikasi datanya, lalu mengklik tautan "Edit", maka sistem akan menampilkan form edit data pendaftaran berobat. 3. Petugas mengecek kesesuaian data pasien dengan berkas-berkas yang dibawa oleh pasien. 4. Petugas menekan tombol update, maka sistem akan menyimpan data pendaftaran.
Skenario alternatif:	1. Data yang diisikan pasien tidak sesuai dengan berkas-berkas yang dibawa oleh pasien, maka petugas menyesuaikan data pasien dengan berkas-berkas pasien, lalu menekan tombol update. 2. Petugas menekan tombol cancel, maka sistem tidak menyimpan data pendaftaran.

Deskripsi *use case* menghapus data pendaftaran berobat pasien dapat dilihat di tabel berikut.

Tabel 4.20 Deskripsi Use Case Menghapus Data Pendaftaran Berobat Pasien

UC-20: Menghapus data pendaftaran berobat pasien	
Ringkasan:	Petugas dapat menghapus data pendaftaran berobat pasien.
Aktor:	Petugas
Kondisi:	Petugas telah *login* ke dalam aplikasi
Skenario Utama:	1. Petugas memilih menu "Antrian", maka sistem akan menampilkan semua data pendaftaran berobat pasien. 2. Petugas memilih pasien yang akan dihapus datanya, lalu mengklik tautan "Delete", maka sistem akan menampilkan form konfirmasi delete data pendaftaran berobat. 3. Petugas menekan tombol "Ya", maka data pendaftaran berobat pasien terhapus.
Skenario Alternatif:	1. Petugas menekan tombol "Tidak", maka data pendaftaran berobat pasien tidak terhapus.

Deskripsi *use case* logout "Monitoring Pendaftaran" dapat dilihat di tabel berikut.

Tabel 4.21 Deskripsi Use Case Logout "Monitoring Pendaftaran"

UC-21: Logout "Monitoring Pendaftaran"	
Ringkasan:	Petugas melakukan logout dari aplikasi "Monitoring Pendaftaran"
Aktor:	Petugas
Kondisi:	Petugas telah *login* ke dalam aplikasi "Monitoring Pendaftaran"

Skenario Utama:	1. Petugas memilih menu logout. 2. Petugas keluar dari aplikasi.

Deskripsi *use case* memverifikasi registrasi "Monitoring Pendaftaran" dapat dilihat di tabel berikut.

Tabel 4.22 Deskripsi Use Case Memverifikasi Registrasi "Monitoring Pendaftaran"

UC-22: Memverifikasi registrasi "Monitoring Pendaftaran"	
Ringkasan:	Pegawai RSUD Bangil bagian Pusat Data Elektronik (PDE) memverifikasi registrasi akun petugas pada aplikasi "Monitoring Pendaftaran"
Aktor:	Pegawai PDE
Kondisi:	Pegawai PDE telah *login* ke dalam database.
Skenario Utama:	1. Pegawai PDE memilih akun petugas yang akan diverfikasi. 2. Pegawai PDE mengubah value role dari 0 menjadi 1, lalu menyimpan data akun.

4.2.4 Daftar class dan method

Nama *class* dan *method* yang akan digunakan pada pembuatan aplikasi akan dijabarkan. Penjabarannya dibagi menjadi tiga bagian, yaitu *class* dan *method* untuk aplikasi "Registrasi Pasien", API, dan aplikasi "Monitoring Pendaftaran".

1. *Class* dan *method* untuk aplikasi "Registrasi Pasien"

Tabel 4.23 Daftar Class dan Method Aplikasi "Registrasi Pasien"

Class	Method

LoginActivity	onCreate();
	verifyLogin();
	showProgress();
	hideProgress();

MainActivity	onCreate();
	onBackPressed();
	setUpMenuView();
	loadHomeFragment();
	getHomeFragment();
	setToolbarTitle();
	selectMenu();
	loadMenuHeader();
	logoutUser();

RegisterPatient	onCreate();
	isValidEmail();
	isValidName();
	isValidBPlace();
	isValidAddress();
	isValidNoId();
	checkBDate();
	requestJsonObject_religion();
	requestJsonObject_education();
	requestJsonObject_occupation();
	registerPatient();
	registerPatientUser();

	showProgress(); hideProgress();
RegisterPatientSuccess	onCreate();

RegisterUser	onCreate();
	isValidNRM();
	isValidEmail();
	searchPatient();
	registerUser();
	showProgress();
	hideProgress();

RegisUserSuccess	onCreate();
AppConfig	

AppController	getInstance();
	getRequestQueue();
	addToRequestQueue();

fragment_term	onCreateView();
fragment_aboutus	onCreateView();
fragment_home	onCreateView();
fragment_polidokter	onCreateView();

fragment_register	onCreateView();
	registerAntrian();
	isValidKeluhan();
	isValidNoAsuransi();
	showProgress();
	hideProgress();

poli_anak	onCreateView();
poli_andrologi	onCreateView();
poli_asma	onCreateView();
poli_bayi	onCreateView();
poli_bedahsyaraf	onCreateView();
poli_bedahumum	onCreateView();
poli_cst	onCreateView();
poli_dalam	onCreateView();
poli_diabetes	onCreateView();
poli_gigi	onCreateView();
poli_gizi	onCreateView();
poli_hemodialisa	onCreateView();
poli_igd	onCreateView();
poli_jantung	onCreateView();
poli_komplementer	onCreateView();
poli_kulit	onCreateView();
poli_mata	onCreateView();
poli_obsgyn	onCreateView();
poli_onkologi	onCreateView();
poli_orthopedi	onCreateView();
poli_paru	onCreateView();
poli_psikologi	onCreateView();
poli_rehabmedik	onCreateView();
poli_syaraf	onCreateView();

poli_tht	onCreateView();
poli_umum	onCreateView();
poli_vct	onCreateView();
DataObject_education	getName(); setName();

SpinnerAdapter_education	getCount();
	getItem();
	getItemId();
	getView();
	getDropDownView();

DataObject_occupation	getName(); setName();

SpinnerAdapter_occupation	getCount();
	getItem();
	getItemId();
	getView();
	getDropDownView();

DataObject_religion	getName(); settName();

SpinnerAdapter_religion	getCount();
	getItem();
	getItemId();
	getView();
	getDropDownView();

SessionManager	setLogin();

	isLoggedIn();

SQLiteHandler	onCreate();
	onUpgrade();
	addUser();
	getUserDetails();
	deleteUser();

SQLite_history	onCreate();
	onUpgrade();
	addHistory();

Utilities	hideKeyboard();
	hideKeyboardOnScrollViewStart();
	bottomScroll();

2. *Class* dan *method* untuk API

Tabel 4.24 Daftar Class dan Method API Aplikasi

Class	Method
Config	
DB_Connect	connect();
patient_register	
education	
get_education	getEducation();
religion	
get_religion	getReligion();
occupation	

get_occupation	getOccupation();
patient_user_register	
user_register	
patient_search	
login	
antrian	
cek_antrian	
history	

DB_Functions	getUserDetail();
	checkHash();
	checkUserValidation();
	isNRMexisted();
	isPatientNotRegistered();
	isEmailExisted();
	isIdExistedOld();
	isIdExistedNew();
	isIdExistedUser();
	registerPatient();
	registerPatientUser();
	registerUser();
	searchPatient();
	getID();
	isRegistered();
	registerAntrian();
	cekAntrian();

	cekHistory(); countHistoryData();

3. *Class* dan *method* untuk aplikasi "Monitoring Pendaftaran".

Tabel 4.25 Daftar Class dan Method Aplikasi "Monitoring Pendaftaran"

Class	Method
c_antrian	
c_delete	
c_deleteAntrian	
c_deleteLama	
c_edit	
c_editAntrian	
c_editLama	
c_pasien_baru	
c_pasien_lama	
c_register	
logout	
validate	
admin	
antrian	
delete	
deleteAntrian	

deleteLama	
edit	
editAntrian	
editLama	
home	
pasien_baru	
pasien_lama	
register	
sign-in.html	
sign-up.html	

account	validate();
	register();
	checkRole();
	checkHash();

m_antrian	showQueue();
	showAntrian();
	getQueue();
	edit();
	deleteAntrian();
	delete();

m_pasien_baru	showNewPatient();
	showBaru();
	getNewPatient();
	edit();

	deleteEmpty();
	generate();
	deleteBaru();
	delete();

m_pasien_lama	showOldPatient();
	showLama();
	getOldPatient();
	edit();
	deleteLama();
	delete();

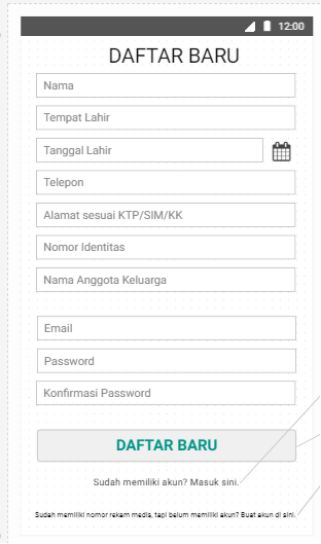
4.2.5 Desain interface aplikasi

1. Desain interface aplikasi “Registrasi Pasien”

Berikut ini adalah desain interface dari aplikasi “Registrasi Pasien”.

i. Pendaftaran Baru

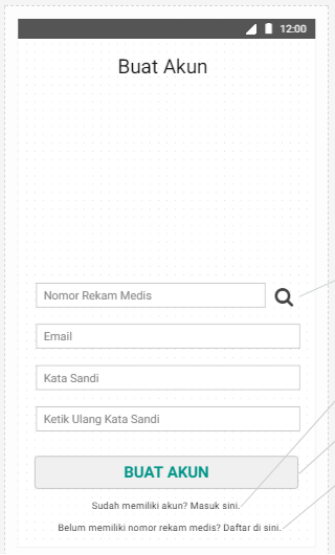
Gambar berikut ini adalah desain interface pendaftaran pasien baru. Halaman pendaftaran baru setidaknya mengandung kotak isian nama, tempat lahir, tanggal lahir, telepon, alamat, nomor identitas, nama anggota keluarga yang bisa dihubungi, *email*, *password*, dan konfirmasi *password*.



Gambar 4.2 Desain Interface Pendaftaran Pasien Baru

ii. Pendaftaran Lama

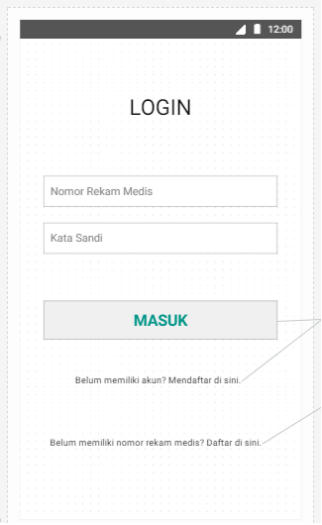
Gambar berikut ini adalah desain interface pendaftaran pasien lama. Untuk pendaftaran lama, setidaknya mengandung kotak isian nomor rekam medis, *email*, password, dan konfirmasi *password*. Tombol pencarian nomor rekam medis juga dibutuhkan untuk mengecek apakah pasien sudah pernah mendaftar di sistem yang sudah ada.



Gambar 4.3 Desain Interface Pendaftaran Pasien Lama

iii. Login

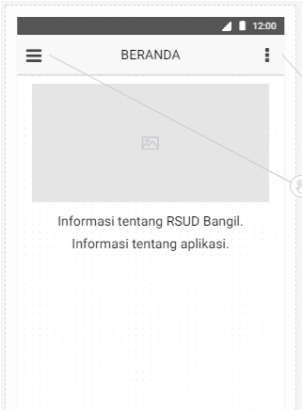
Gambar berikut ini adalah desain interface *login*. Untuk desain interface halaman *login*, setidaknya mengandung kotak isian nomor rekam medis dan *password* untuk melakukan akses ke aplikasi.



Gambar 4.4 Desain Interface Login Aplikasi "Registrasi Pasien"

iv. Beranda

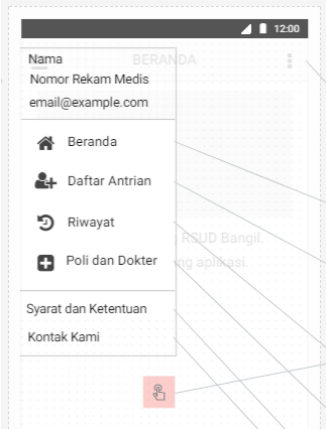
Berikut ini adalah desain interface halaman beranda.



Gambar 4.5 Desain Interface Halaman Beranda Aplikasi "Registrasi Pasien"

v. Menu Drawer

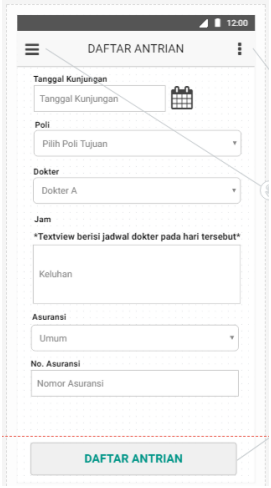
Berikut ini adalah menu navigation drawer yang akan ditampilkan ketika layar digeser dari kiri ke kanan atau tombol di pojok kiri atas ditekan. Menu-menu yang ada adalah menu untuk menuju ke halaman beranda, pendaftaran berobat, pengecekan riwayat kunjungan, pengecekan jadwal poli dan dokter, halaman yang berisi kontak yang RSUD, dan halaman yang berisikan beberapa syarat dan ketentuan bagi pasien yang ingin berobat.



Gambar 4.6 Desain Interface Menu Drawer Aplikasi "Registrasi Pasien"

vi. Pendaftaran Berobat

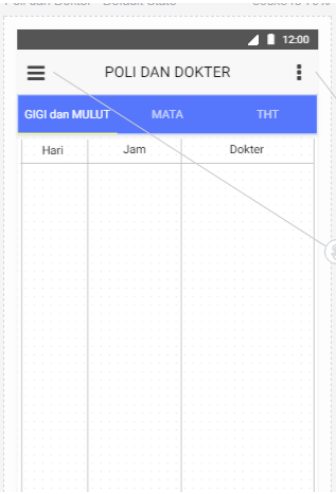
Berikut ini adalah desain interface pendaftaran berobat. Halaman ini setidaknya berisi kotak isian tanggal kunjungan, poli tujuan, dokter, keluhan, jenis asuransi, nomor asuransi, dan informasi tentang jam kehadiran dokter.



Gambar 4.7 Desain Interface Pendaftaran Berobat

vii. Poli dan Dokter

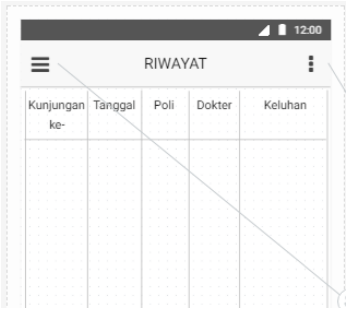
Berikut ini adalah desain interface halaman poli dan dokter. Halaman ini diperlukan untuk mengecek jadwal poli dan dokter. Pada halaman ini, informasi tentang hari, jam kehadiran, dan nama dokter harus ada. Selain itu, nama-nama poli juga harus ada.



Gambar 4.8 Desain Interface Halaman Poli dan Dokter

viii. Riwayat Kunjungan

Berikut ini adalah desain interface halaman pengecekan riwayat kunjungan. Pada halaman ini, setidaknya harus ada kunjungan pasien ke-berapa, tanggal, poli tujuan, dokter yang menangani, dan keluhan pasien.



Gambar 4.9 Desain Interface Halaman Riwayat Kunjungan

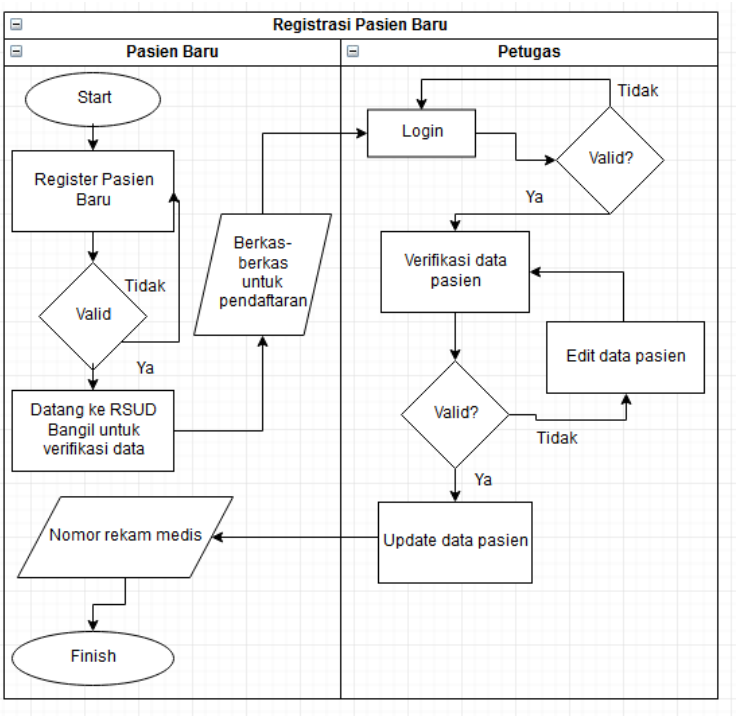
Untuk API dari aplikasi, keluarannya menggunakan JSON. Jadi, tidak diperlukan adanya interface.

4.3 Alur Penggunaan Aplikasi

Alur dari penggunaan aplikasi ini diawali dari pendaftaran pasien di aplikasi “Registrasi Pasien” hingga verifikasi data pasien di aplikasi “Monitoring Pendaftaran”. Untuk lebih jelasnya, alur aplikasi adalah sebagai berikut.

4.3.1 Register Pasien Baru

Pasien baru adalah pasien yang belum terdaftar baik di aplikasi ini maupun di sistem yang sudah ada sebelumnya. Pasien baru belum memiliki nomor rekam medis. Alur pendaftaran pasien baru adalah sebagai berikut.

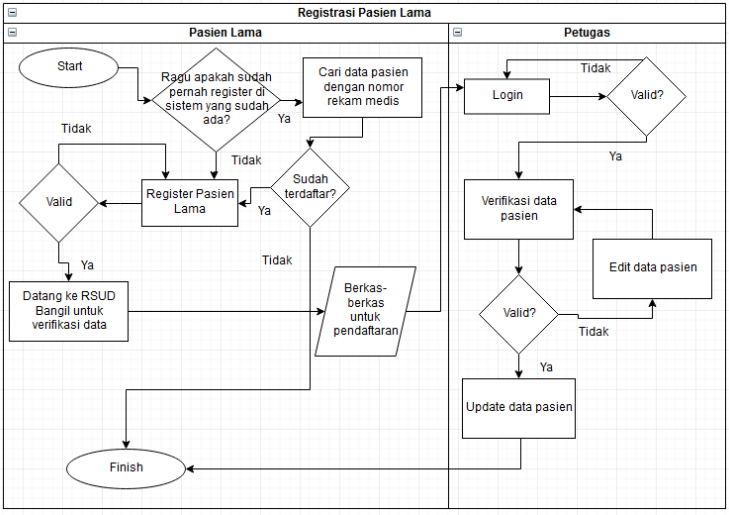


Gambar 4.10 Alur Registrasi Pasien Baru

Pasien baru melakukan registrasi di aplikasi. Setelah itu, pasien baru datang ke RSUD Bangil, dengan membawa berkas-berkas pendaftaran, untuk melakukan verifikasi data. Petugas memverifikasi data pasien. Apabila ada yang tidak sesuai, maka data pasien diubah agar sesuai dengan berkas yang dibawa. Setelah itu, petugas memperbarui data pasien. Pasien baru mendapatkan nomor rekam medis dan sudah bisa mengakses aplikasi “Registrasi Pasien”.

4.3.2 Register Pasien Lama

Pasien lama adalah pasien yang sudah terdaftar di sistem yang sudah ada sebelumnya, namun belum terdaftar di aplikasi. Pasien lama memiliki nomor rekam medis, tapi belum memiliki hak akses ke aplikasi "Registrasi Pasien". Alur pendaftaran pasien baru adalah sebagai berikut.



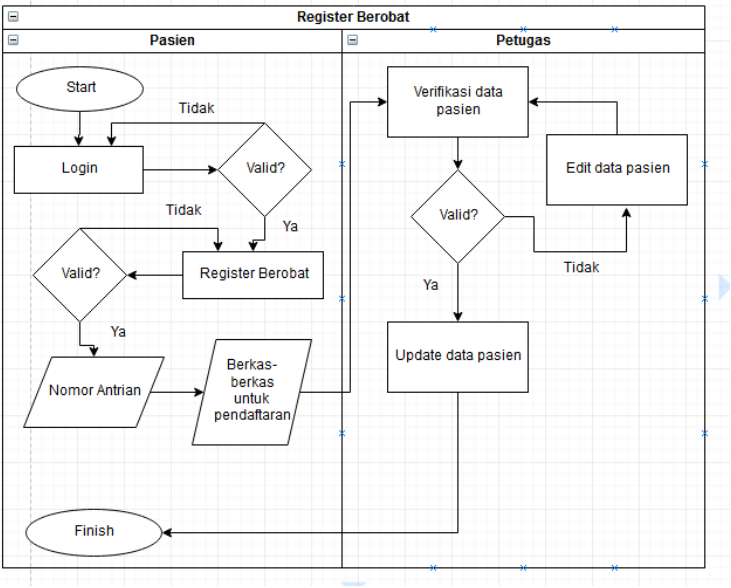
Gambar 4.11 Alur Registrasi Pasien Lama

Pasien lama melakukan registrasi di aplikasi. Apabila pasien lama ragu apakah sudah terdaftar di sistem yang sudah ada atau belum, atau ragu apakah nomor rekam medis yang dimilikinya itu sesuai dengan data dirinya, pasien lama bisa melakukan pencarian data pasien dengan memasukkan nomor rekam medis. Setelah register di aplikasi, pasien lama datang ke RSUD Bangil, dengan membawa berkas-berkas pendaftaran, untuk melakukan verifikasi data. Petugas memverifikasi data pasien. Apabila ada yang tidak sesuai, maka data pasien lama diubah agar sesuai dengan berkas yang dibawa. Setelah itu, petugas

memperbarui data pasien lama. Pasien lama sudah bisa mengakses aplikasi "Registrasi Pasien".

4.3.3 Mendaftar Berobat

Pada tugas akhir ini, baik pasien lama maupun pasien baru yang sudah memiliki hak akses ke aplikasi sama-sama disebut dengan pasien. Pasien bisa melakukan pendaftaran berobat melalui aplikasi, setelah melakukan *login*.



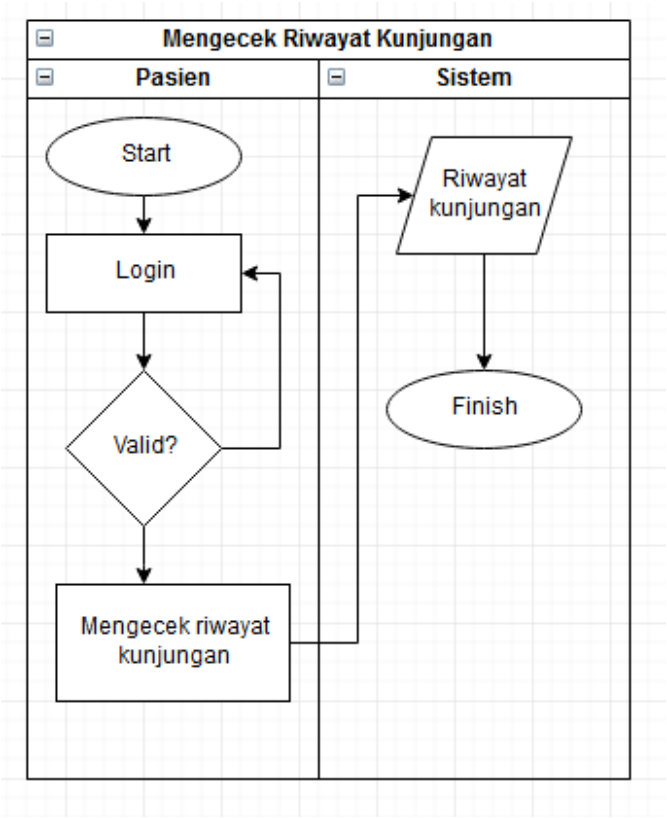
Gambar 4.12 Alur Register Berobat

Setelah melakukan *login*, pasien membuka menu untuk menuju ke halaman register berobat. Pasien mengisikan form registrasi dan melakukan registrasi. Setelah itu, pasien mendapatkan nomor antrian. Pasien kemudian datang ke RSUD Bangil dengan membawa berkas-berkas pendaftaran. Petugas melakukan verifikasi data pasien di database dengan berkas-

berkas yang dibawa. Apabila ada ketidaksesuaian, maka data yang tidak sesuai diubah isinya agar menjadi sesuai dengan berkas yang dibawa. Setelah itu, petugas meng-*update* data pasien. Pasien sudah terdaftar di poli dan bisa segera menuju ke loket pembayaran.

4.3.4 Mengecek Riwayat Kunjungan

Pasien dapat mengecek riwayat kunjungan berobat melalui aplikasi "Registrasi Pasien", setelah melakukan login. Berikut ini adalah alurnya.

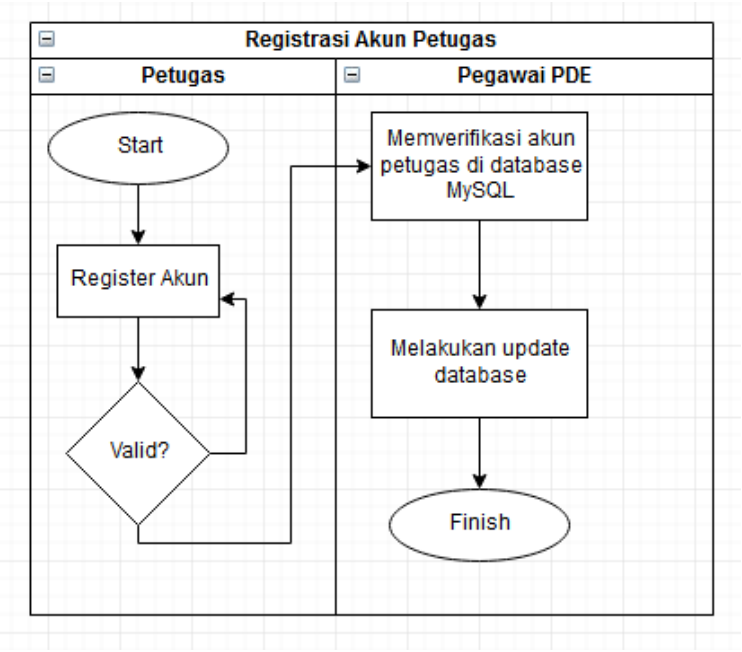


Gambar 4.13 Alur Pengecekan Riwayat Kunjungan

Setelah melakukan login, pasien membuka menu untuk menuju ke halaman pengecekan riwayat kunjungan. Aplikasi "Registrasi Pasien" akan menampilkan riwayat kunjungan pasien.

4.3.5 Mendaftarkan Akun Petugas

Untuk melakukan verifikasi data pasien, petugas perlu untuk memiliki akun di aplikasi "Monitoring Pendaftaran". Petugas mendaftarkan akunnya terlebih dahulu, lalu pegawai bagian Pusat Data Elektronik (PDE) akan memverifikasi datanya.



Gambar 4.14 Alur Registrasi Akun Petugas

Petugas membuka aplikasi "Monitoring Pendaftaran", lalu mengklik tautan registrasi untuk membuat akun. Petugas

mengisi form pendaftaran akun. Setelah itu, petugas menghubungi bagian Pusat Data Elektronik(PDE) kalau sudah mendaftar dengan suatu *username*. Pegawai bagian PDE mengecek database apakah *username* tersebut memang ada. Jika ada, maka pegawai bagian PDE mengubah *value role* petugas agar bisa mengakses aplikasi "Monitoring Pendaftaran". Petugas sudah bisa melakukan *login* pada aplikasi.

BAB V
IMPLEMENTASI

Bab ini berisi implementasi dari perancangan yang sudah dijelaskan pada bab perancangan.

5.1 Teknologi yang Digunakan

Pengembangan aplikasi ini menggunakan komputer dengan spesifikasi sebagai berikut.

Tabel 5.1 Spesifikasi Komputer untuk Pengembangan Aplikasi

Hardware/Software	Spesifikasi
Prosesor	Intel® Core™ i5 CPU @2.40 GHz
Memori	4096 MB RAM
Sistem Operasi	Windows 7 Professional 64-bit (6.1 Build 7601)

Aplikasi dikembangkan dengan beberapa teknologi sebagai berikut.

Tabel 5.2 Teknologi yang Digunakan untuk Pengembangan Aplikasi

Fungsi	Software
Webserver	Apache 1.7.4
Bahasa Pemrograman	PHP 5.3.5, Java
Database	MySQL 5.5.8
SDK	Android SDK 5.1.1
Editor	Android Studio 2.3, Notepad++
Browser	Firefox 61.0
Library	Bootstrap v3, Volley, Gson, upgradephp-19

5.2 Lingkungan Implementasi

Aplikasi “Registrasi Pasien” akan dipasangkan pada perangkat *mobile.* Kebutuhan sistem minimum untuk pemasangan aplikasi “Registrasi Pasien” adalah:

Tabel 5.3 Kebutuhan Sistem Minimum Aplikasi "Registrasi Pasien"

Fungsi	Teknologi
Sistem Operasi	Android Lollipop 5.0.1
Network Connection	Yes

Aplikasi “Monitoring Pendaftaran” dan API akan dipasang pada server dengan spesifikasi sebagai berikut.

Tabel 5.4 Spesifikasi Server

Fungsi	Software
Sistem Operasi	Windows Server 2012
Webserver	Apache 1.7.4
Database	MySQL 5.5.8
Prosesor	Intel® Core™ i7 CPU @3.30 GHz
Memori	16 GB RAM

5.3 Pembuatan Daftar Poli

Pembuatan daftar poli bertujuan untuk melengkapi fitur pengecekan jadwal poli dan dokter, serta untuk melengkapi isian spinner poli tujuan pada pendaftaran berobat pasien. Daftar nama poli diambil dari rekap laporan pasien kunjungan rawat jalan pada bulan Maret tahun 2017.

5.4 Pengisian Spinner Halaman Pendaftaran Baru

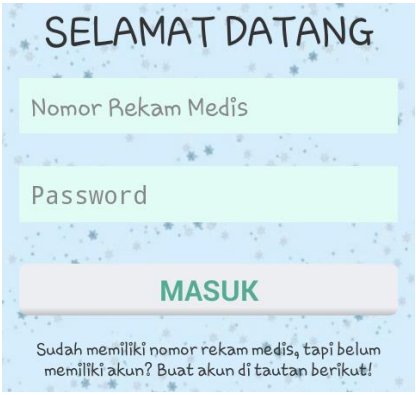
Isian spinner pada halaman pendaftaran baru menyesuaikan dengan isian-isian yang sudah pernah diisikan oleh pasien pada sistem yang sudah ada. Spinner yang isinya disesuaikan adalah spinner agama, pendidikan terakhir, dan pekerjaan. Isian ketiga spinner tersebut disesuaikan dengan data dari database sehingga ketika ada pasien yang mengisikan data baru pada sistem yang sudah ada, isian ketiga spinner tersebut bisa ditambahkan.

5.5 Pembangunan Aplikasi

Pada bab ini akan dijelaskan mengenai pembangunan aplikasi yang telah dilakukan sesuai dengan metode pengembangan aplikasi.

5.5.1 Fungsi Login

Pasien akan menggunakan kombinasi nomor rekam medis dan *password* untuk melakukan *login* ke aplikasi. *Login* ke dalam aplikasi "Registrasi Pasien" menggunakan *library* volley untuk menjalankan file login.php yang ada pada server. Halaman *login* seperti gambar berikut akan ditampilkan ketika pertama kali membuka aplikasi.



Gambar 5.1 Halaman Login

File login.php dijalankan kemudian memanggil file DB_Functions.php yang berisi kumpulan method yang berguna untuk API. *Method* untuk memverifikasi kesesuaian isi nomor rekam medis dan *password* adalah *method* getUserDetail().

```
public function getUserDetail($nomor_medrec, $password) {

    $stmt = $this->conn->prepare("SELECT u.nomor_medrec, u.email, u.encry

    $stmt->bind_param("ss", $nomor_medrec, $nomor_medrec);

    if($stmt->execute()){
        $user = $stmt->get_result()->fetch_assoc();

        $stmt->close();

        $encrypted = $user['encrypted_password'];
        $salt = $user['salt'];

        $hash = $this->checkHash($salt, $password);

        if($encrypted == $hash){
            return $user;
        }
        else {
            return false;
        }
    }
    else {
        return NULL;
    }
```

Kode Program 5.1 Method getUserDetail()

Sistem mengirimkan *query* untuk mengambil *value-value* data pasien dari database, lalu mengecek kesesuaian isian *password* yang diisikan dengan enkripsi *password*. Untuk menghindari sql injection, maka query pada file DB_Functions.php menggunakan sql prepared statement.

```php
$response = array("error" => FALSE);

if (isset($_POST['nomor_medrec']) && isset($_POST['password'])) {

    // receiving the post params
    $nomor_medrec = $_POST['nomor_medrec'];
    $password = $_POST['password'];

    // get the user by nomor_medrec and password
    $user = $db->getUserDetail($nomor_medrec, $password);

    if ($user) {
        // user is found
        // Mengecek apakah user telah melakukan validasi di loket pendaftaran atau belum
        $valid = $db->checkUserValidation($nomor_medrec);

        if ($valid != false){
            $response["error"] = FALSE;
            $response["user"]["nomor_medrec"] = $user["nomor_medrec"];
            $response["user"]["email"] = $user["email"];
            $response["user"]["nama"] = $user["nama"];
            $response["user"]["tgl_lahir"] = $user["tgl_lahir"];
            $response["user"]["tempat_lahir"] = $user["tempat_lahir"];
            $response["user"]["umur"] = $user["umur"];
            $response["user"]["jenis_kelamin"] = $user["jenis_kelamin"];
            $response["user"]["alamat"] = $user["alamat"];
            echo json_encode($response);
        }
        else {
            $response["error"] = TRUE;
            $response["error_msg"] = "Anda belum melakukan validasi akun di loket pendaftaran!";
            echo json_encode($response);
        }
```

Kode Program 5.2 Class Login.php

Jika hasilnya *true*, sistem akan membuat array JSON yang bernama “user”. Jika tidak, maka keluarannya berupa *boolean* dengan nilai *false*. Array ini kemudian akan disimpan dalam database SQLite yang ada di aplikasi, dengan nama “sqliteDB”.

```
if (!error){
    //berhasil login
    session.setLogin(true);

    //menyimpan user di database SQLite

    JSONObject user = object.getJSONObject("user");
    String nrm = user.getString("nomor_medrec");
    String email = user.getString("email");
    String name = user.getString("nama");
    String bdate = user.getString("tgl_lahir");
    String bplace = user.getString("tempat_lahir");
    String age = user.getString("umur");
    String gender = user.getString("jenis_kelamin");
    String address = user.getString("alamat");

    //menambahkan baris di tabel user
    sqliteDB.addUser(nrm, email, name, bdate, bplace, age, gender, address);

    Intent intent = new Intent(LoginActivity.this, MainActivity.class);
    startActivity(intent);
    finish();
```

Kode Program 5.3 Aplikasi "Registrasi Pasien" mengambil array JSON dan menyimpannya ke database SQLite

Agar user tidak perlu login lagi ketika keluar dari aplikasi (tanpa logout), maka dibuatlah session.

```
public class SessionManager {
    private static String TAG = SessionManager.class.getSimpleName();
    private SharedPreferences pref;
    private SharedPreferences.Editor editor;
    private Context _context;
    private int PRIVATE_MODE = 0;
    private static final String PREF_NAME = "Login Aplikasi Pendaftaran Pasien RSUD Bangil";
    private static final String KEY_IS_LOGGEDIN = "isLoggedIn";

    public SessionManager(Context context){
        this._context = context;
        pref = _context.getSharedPreferences(PREF_NAME, PRIVATE_MODE);
        editor = pref.edit();
    }

    public void setLogin(boolean isLoggedIn){
        editor.putBoolean(KEY_IS_LOGGEDIN, isLoggedIn);
        editor.commit();
        Log.d(TAG, "User login session modified!");
    }

    public boolean isLoggedIn(){
        return pref.getBoolean(KEY_IS_LOGGEDIN, false);
    }
}
```

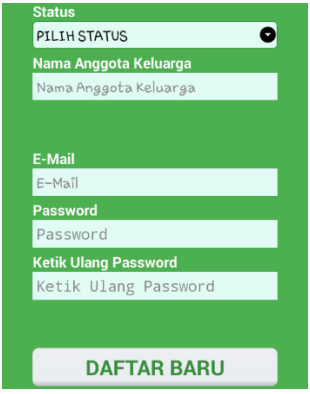
Kode Program 5.4 Session

5.5.2 Fungsi Register Pasien Baru

Untuk registrasi pasien baru, pasien mengisikan data diri seperti pada gambar berikut.

Gambar 5.2 Form Registrasi Pasien Baru

Pasien juga mengisikan *email* dan *password*.



Gambar 5.3 Lanjutan Form Registrasi Pasien Baru

Isian spinner agama di-generate dari database. Sistem menggunakan method requestJsonObject_religion() untuk melakukan request ke *library* volley untuk memanggil file religion.php yang ada pada *package* spinner.

```
private void requestJsonObject_religion() {
    StringRequest stringRequest1 = new StringRequest(Request.Method.GET, AppConfig.URL_SPINNER_RELIGION, (response) -> {
            GsonBuilder builder1 = new GsonBuilder();
            Gson mGson1 = builder1.create();

            spinnerDataReligion = Arrays.asList(mGson1.fromJson(response, DataObject_religion[].class));

            assert patientSpinnerReligion != null;
            patientSpinnerReligion.setVisibility(View.VISIBLE);
            SpinnerAdapter_religion spinnerAdapterReligion = new SpinnerAdapter_religion(getApplicationContext(), spinner
            patientSpinnerReligion.setAdapter(spinnerAdapterReligion);
    }, (error) -> {
            Log.e(TAG, "Error ketika login: " + error.getMessage());
            Toast.makeText(getApplicationContext(), error.getMessage(), Toast.LENGTH_SHORT).show();
    });
    AppController.getInstance().addToRequestQueue(stringRequest1);
}
```

Kode Program 5.5 Method requestJSONObject_religion()

Begitu juga pada spinner pendidikan terakhir. Sistem menggunakan method requestJsonObject_education() untuk melakukan request ke *library* volley untuk memanggil file education.php yang ada pada *package* spinner.

```
private void requestJsonObject_education() {
    StringRequest stringRequest2 = new StringRequest(Request.Method.GET, AppConfig.URL_SPINNER_EDUCATION, (response) -> {
            GsonBuilder builder2 = new GsonBuilder();
            Gson mGson2 = builder2.create();

            spinnerDataEducation = Arrays.asList(mGson2.fromJson(response, DataObject_education[].class));

            assert patientSpinnerEducation != null;
            patientSpinnerEducation.setVisibility(View.VISIBLE);
            SpinnerAdapter_education spinnerAdapterEducation = new SpinnerAdapter_education(getApplicationContext(), spir
            patientSpinnerEducation.setAdapter(spinnerAdapterEducation);
    }, (err) -> {
            Log.e(TAG, "Error ketika login: " + err.getMessage());
            Toast.makeText(getApplicationContext(), err.getMessage(), Toast.LENGTH_SHORT).show();
    });
    AppController.getInstance().addToRequestQueue(stringRequest2);
}
```

Kode Program 5.6 Method requestJSONObject_education()

Dan juga spinner pekerjaan. Sistem menggunakan method requestJsonObject_occupation() untuk melakukan request ke *library* volley untuk memanggil file occupation.php yang ada pada *package* spinner.

```
private void requestJsonObject_occupation() {
    StringRequest stringRequest3 = new StringRequest(Request.Method.GET, AppConfig.URL_SPINNER_OCCUPATION, (response)
            GsonBuilder builder3 = new GsonBuilder();
            Gson mGson3 = builder3.create();

            spinnerDataOccupation = Arrays.asList(mGson3.fromJson(response, DataObject_occupation[].class));

            assert patientSpinnerOccupation != null;
            patientSpinnerOccupation.setVisibility(View.VISIBLE);
            SpinnerAdapter_occupation spinnerAdapterOccupation = new SpinnerAdapter_occupation(getApplicationContext(),
            patientSpinnerOccupation.setAdapter(spinnerAdapterOccupation);
    }, (error) -> {
            Log.e(TAG, "Error ketika login: " + error.getMessage());
            Toast.makeText(getApplicationContext(), error.getMessage(), Toast.LENGTH_SHORT).show();
    });
    AppController.getInstance().addToRequestQueue(stringRequest3);
}
```

Kode Program 5.7 Method requestJSONObject_occupation()

Pada spinner agama, file yang akan dipanggil adalah religion.php.

```php
<?php
require_once dirname (__FILE__) . './get_religion.php';

$queryObj = new get_religion();
$queryObj->getReligion();

?>
```

Kode Program 5.8 religion.php

File religion.php memanggil file lain yang bernama get_religion.php untuk melakukan query pada database, lalu hasilnya dibentuk dalam JSON.

```php
<?php
include_once dirname(dirname (__FILE__)) . './includes/DB_Connect.php';
include_once dirname(dirname (__FILE__)) . './libs/upgradephp-19/upgrade.php';

class get_religion{
    private $tables;

    public function __construct(){
        $this->tables = "religion";
    }

    public function getReligion(){
        $religions = array();
        $com = new DB_Connect();
        $sql = "SELECT * FROM religion";
        $result = mysqli_query($com->connect(), $sql);
        $rows = mysqli_num_rows($result);

        if ($rows > 0){
            while ($row = $result->fetch_assoc()){
                $religions[] = $row;
            }
            echo json_encode($religions, JSON_PRETTY_PRINT);
        }
    }
}
?>
```

Kode Program 5.9 get_religion.php

Hasil tersebut dijadikan sebuah *data object* pada aplikasi "Registrasi Pasien", dengan nama class DataObject_religion. *Data object* adalah sebuah obyek buatan penulis yang mendefinisikan sesuatu, dalam hal ini isian spinner.

```php
public class DataObject_religion {
    @SerializedName("religion_nm")
    private String religion_nm;

    public DataObject_religion(){

    }

    public DataObject_religion(String name) { this.religion_nm = name; }

    public String getName() { return religion_nm; }

    public void setName(String name) { this.religion_nm = name; }
}
```

Kode Program 5.10 DataObject_religion

Setelah data object dibentuk, dibuatlah *adapter spinner* tersebut, dengan nama class SpinnerAdapter_education, yang nantinya dijadikan daftar isian dari spinner. *Adapter spinner* adalah item-item yang menjadi isi dari spinner.

```
public class SpinnerAdapter_religion extends BaseAdapter {
    private LayoutInflater layoutInflater;
    private List<DataObject_religion> listData;
    private Context context;

    public SpinnerAdapter_religion(Context context, List<DataObject_religion> listData){
        this.context = context;
        layoutInflater = (LayoutInflater) context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
        this.listData = listData;
    }

    @Override
    public int getCount() { return listData.size(); }

    @Override
    public Object getItem(int position) { return (DataObject_religion) listData.get(position); }

    @Override
    public long getItemId(int position) { return 0; }

    //Untuk spinner sebelum diklik
    @Override
    public View getView(int position, View view, ViewGroup viewGroup) {
        ViewHolder spinnerHolder;
        if (view == null){
            spinnerHolder = new ViewHolder();
            view = layoutInflater.inflate(R.layout.spinner_list, viewGroup, false);
            spinnerHolder.spinnerItemList = (TextView) view.findViewById(R.id.spinner_list_item);
            view.setTag(spinnerHolder);
```

Kode Program 5.11 SpinnerAdapter_religion

Pada spinner pendidikan terakhir, file yang akan dipanggil adalah education.php.

```
<?php
require_once dirname (__FILE__) . './get_education.php';

$queryObj = new get_education();
$queryObj->getEducation();

?>
```

Kode Program 5.12 education.php

File education.php memanggil file lain yang bernama get_education.php untuk melakukan query pada database, lalu hasilnya dibentuk dalam JSON.

```php
<?php
include_once dirname(dirname (__FILE__)) . './includes/DB_Connect.php';

/** Load library upgradephp (source: http://include-once.org/p/upgradephp) agar PHP versi di
include_once dirname(dirname (__FILE__)) . './libs/upgradephp-19/upgrade.php';

class get_education{
    private $tables;

    public function __construct(){
        $this->tables = "education";
    }

    public function getEducation(){
        $educations = array();
        $com = new DB_Connect();
        $sql = "SELECT * FROM education";
        $result = mysqli_query($com->connect(), $sql);
        $rows = mysqli_num_rows($result);

        if ($rows > 0){
            while ($row = $result->fetch_assoc()){
                $educations[] = $row;
            }
            echo json_encode($educations, JSON_PRETTY_PRINT);
        }
    }
}
?>
```

Kode Program 5.13 get_education.php

Hasil tersebut dijadikan sebuah data object pada aplikasi “Registrasi Pasien”, dengan nama class DataObject_education.

```
public class DataObject_education {
    @SerializedName("education_nm")
    private String education_nm;

    public DataObject_education(){

    }

    public DataObject_education(String name){
        this.education_nm = name;
    }

    public String getName(){
        return education_nm;
    }

    public void setName(String name){
        this.education_nm = name;
    }
}
```

Kode Program 5.14 DataObject_education

Setelah berbentuk data object, dibuatlah adapter spinner tersebut, dengan nama class SpinnerAdapter_education, yang nantinya dijadikan daftar isian dari spinner.

```
public class SpinnerAdapter_education extends BaseAdapter {
    private LayoutInflater layoutInflater;
    private List<DataObject_education> listData;
    private Context context;

    public SpinnerAdapter_education(Context context, List<DataObject_education> listData){
        this.context = context;
        layoutInflater = (LayoutInflater) context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
        this.listData = listData;
    }

    @Override
    public int getCount() { return listData.size(); }

    @Override
    public Object getItem(int position) { return (DataObject_education) listData.get(position); }

    @Override
    public long getItemId(int position) { return 0; }

    @Override
    public View getView(int position, View view, ViewGroup viewGroup) {
        ViewHolder spinnerHolder;
        if (view == null){
            spinnerHolder = new ViewHolder();
            view = layoutInflater.inflate(R.layout.spinner_list, viewGroup, false);
            spinnerHolder.spinnerItemList = (TextView) view.findViewById(R.id.spinner_list_item);
            view.setTag(spinnerHolder);
        }
        else {
            spinnerHolder = (ViewHolder) view.getTag();
        }
```

Kode Program 5.15 SpinnerAdapter_education

Pada spinner pekerjaan, file yang akan dipanggil adalah occupation.php.

```
<?php
require_once dirname (__FILE__) . './get_occupation.php';

$queryObj = new get_occupation();
$queryObj->getOccupation();

?>
```

Kode Program 5.16 occupation.php

File occupation.php memanggil file lain yang bernama get_occupation.php untuk melakukan query pada database, lalu hasilnya dibentuk dalam JSON.

```php
<?php
include_once dirname(dirname (__FILE__)) . './includes/DB_Connect.php';
include_once dirname(dirname (__FILE__)) . './libs/upgradephp-19/upgrade.php';

class get_occupation{
    private $tables;

    public function __construct(){
        $this->tables = "occupation";
    }

    public function getOccupation(){
        $occupations = array();
        $com = new DB_Connect();
        $sql = "SELECT * FROM occupation";
        $result = mysqli_query($com->connect(), $sql);
        $rows = mysqli_num_rows($result);

        if ($rows > 0){
            while ($row = $result->fetch_assoc()){
                $occupations[] = $row;
            }
            echo json_encode($occupations, JSON_PRETTY_PRINT);
        }
    }
}
?>
```

Kode Program 5.17 get_occupation.php

Hasil tersebut dijadikan sebuah data object pada aplikasi "Registrasi Pasien", dengan nama class DataObject_occupation.

```php
public class DataObject_occupation {
    @SerializedName("occupation_nm")
    private String occupation_nm;

    public DataObject_occupation(){

    }

    public DataObject_occupation(String name) { this.occupation_nm = name; }

    public String getName() { return occupation_nm; }

    public void setName(String name) { this.occupation_nm = name; }
}
```

Kode Program 5.18 DataObject_occupation

Setelah berbentuk data object, dibuatlah adapter spinner tersebut, dengan nama class SpinnerAdapter_occupation, yang nantinya dijadikan daftar isian dari spinner.

```
public class SpinnerAdapter_education extends BaseAdapter {
    private LayoutInflater layoutInflater;
    private List<DataObject_education> listData;
    private Context context;

    public SpinnerAdapter_education(Context context, List<DataObject_education> listData){
        this.context = context;
        layoutInflater = (LayoutInflater) context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
        this.listData = listData;
    }

    @Override
    public int getCount() { return listData.size(); }

    @Override
    public Object getItem(int position) { return (DataObject_education) listData.get(position); }

    @Override
    public long getItemId(int position) { return 0; }

    @Override
    public View getView(int position, View view, ViewGroup viewGroup) {
        ViewHolder spinnerHolder;
        if (view == null){
            spinnerHolder = new ViewHolder();
            view = layoutInflater.inflate(R.layout.spinner_list, viewGroup, false);
            spinnerHolder.spinnerItemList = (TextView) view.findViewById(R.id.spinner_list_item);
            view.setTag(spinnerHolder);
        }
        else {
            spinnerHolder = (ViewHolder) view.getTag();
        }
```

Kode Program 5.19 SpinnerAdapter_education

Setelah pasien mengisi semua isian, pasien menekan tombol daftar baru, lalu sistem akan melakukan request ke *library* volley sebanyak 2 kali. Yang pertama adalah untuk memanggil file patient_register.php yang akan digunakan untuk menyimpan data pasien baru.

```
StringRequest requestString = new StringRequest(Request.Method.POST, AppConfig.URL_REGISTER_PATIENT, new Response.Listener<
```

Kode Program 5.20 Aplikasi "Registrasi Pasien" memanggil file patient_register.php

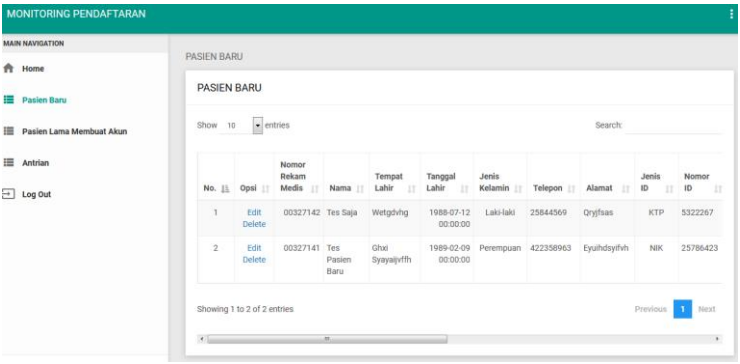
Yang kedua adalah untuk memanggil file register_patient_user.php untuk menyimpan data pasien baru

sebagai data user yang berisi *email* dan *password*. Sehingga data yang ini nantinya bisa digabung dengan data pasien lama.

```
StringRequest requestString2 = new StringRequest(Request.Method.POST, AppConfig.URL_REGISTER_PATIENT_USER, new Response.Listene
```

Kode Program 5.21 Aplikasi "Registrasi Pasien" memanggil file register_patient_user.php

Petugas kemudian membuka aplikasi "Monitoring Pendaftaran" dan menuju ke halaman Pasien Baru, seperti pada gambar berikut.



Gambar 5.4 Halaman Pasien Baru pada Aplikasi "Monitoring Pendaftaran"

Isian dari tabel di atas di-generate dari fungsi pasien_baru.php yang ada pada aplikasi "Monitoring Pendaftaran", *package* view.

```
$mBaru = new m_pasien_baru;
$data = $mBaru->showNewPatient();

for($b = 0; $b < $mBaru->getNewPatient(); $b++){
//$db = new DB_Connect();
echo "<tr>
    <td align='center'>".($b+1)."</td>
    <td align='center'>
        <a href='../view/edit.php?b=".str_replace(' ', '_', $data[$b][8])."'>Edit</a>
        <a href='../view/delete.php?b=".str_replace(' ', '_', $data[$b][8])."'>Delete</a>
    </td>
    <td align='right' style='padding-right:5px;'>".$data[$b][0]."</td>
    <td align='left' style='padding-left:5px;'>".$data[$b][1]."</td>
    <td align='left' style='padding-left:5px;'>".$data[$b][2]."</td>
    <td align='right' style='padding-right:5px;'>".$data[$b][3]."</td>
    <td align='center'>".$data[$b][4]."</td>
    <td align='left' style='padding-left:5px;'>".$data[$b][5]."</td>
    <td align='left' style='padding-left:5px;'>".$data[$b][6]."</td>
    <td align='center'>".$data[$b][7]."</td>
    <td align='left' style='padding-left:5px;'>".$data[$b][8]."</td>
    <td align='left' style='padding-left:5px;'>".$data[$b][9]."</td>
    <td align='left' style='padding-left:5px;'>".$data[$b][10]."</td>
    <td align='left' style='padding-left:5px;'>".$data[$b][11]."</td>
    <td align='center'>".$data[$b][12]."</td>
    <td align='left' style='padding-left:5px;'>".$data[$b][13]."</td>
    <td align='left' style='padding-left:5px;'>".$data[$b][14]."</td>
    <td align='right' style='padding-right:5px;'>".$data[$b][15]."</td>
    <td align='center' style='padding-left:5px;'>".$data[$b][16]."</td>
    </tr>";
}
```

Kode Program 5.22 pasien_baru.php

Sebelum menuju ke situ, sistem memanggil fungsi c_pasien_baru.php terlebih dahulu yang ada pada *package* controller. Dari situ, pasien_baru.php yang ada pada *package* view akan dihubungkan dengan m_pasien_baru.php yang ada pada *package* model. Pada gambar di atas, isian didapat dari fungsi showNewPatient() pada m_pasien_baru.php. Fungsi tersebut adalah sebagai berikut.

```
function showNewPatient(){

    $tglIni = date('Y-m-d');

    $query = $this->conn->prepare("SELECT p.*, u.email, u.tgl_daftar, u.vali
    $query->execute();
    $res = $query->get_result();

    $data = array(array(0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16));
    $b = 0;
    while($row = $res->fetch_assoc()){
        $data[$b][0] = "{$row['nomor_medrec']}";
        $data[$b][1] = "{$row['nama']}";
        $data[$b][2] = "{$row['tempat_lahir']}";
        $data[$b][3] = "{$row['tgl_lahir']}";
        $data[$b][4] = "{$row['jenis_kelamin']}";
        $data[$b][5] = "{$row['telepon']}";
        $data[$b][6] = "{$row['alamat']}";
        $data[$b][7] = "{$row['jenis_id']}";
        $data[$b][8] = "{$row['no_identitas']}";
        $data[$b][9] = "{$row['agama']}";
        $data[$b][10] = "{$row['pendidikan']}";
        $data[$b][11] = "{$row['pekerjaan']}";
        $data[$b][12] = "{$row['status']}";
        $data[$b][13] = "{$row['anggota_keluarga']}";
        $data[$b][14] = "{$row['email']}";
        $data[$b][15] = "{$row['tgl_daftar']}";
        $data[$b][16] = "{$row['valid']}";

        if($data[$b][4] == 'M'){
            $data[$b][4] = 'Laki-laki';
        }
        else if($data[$b][4] == 'F'){
            $data[$b][4] = 'Perempuan';
```

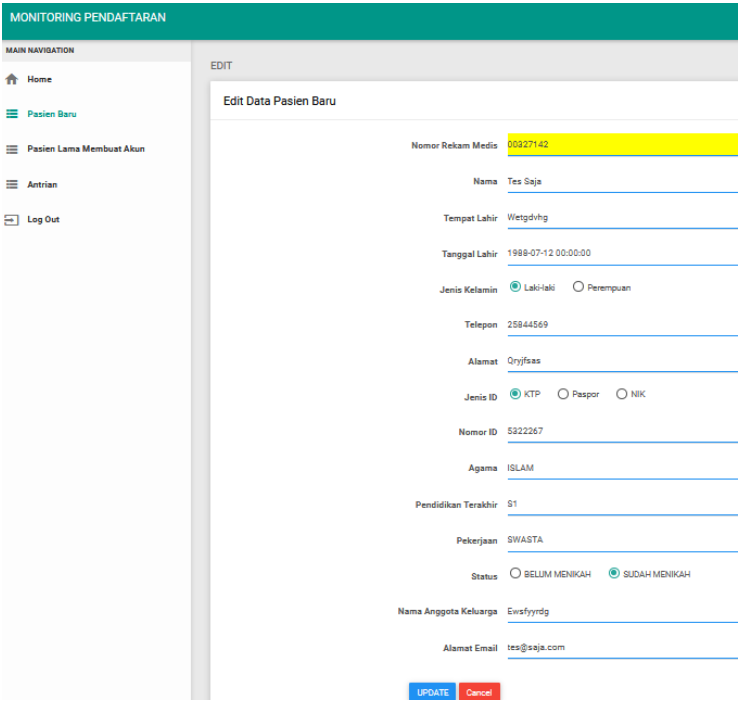
Kode Program 5.23 Method showNewPatient()

Sementara itu, jumlah baris bisa lebih dari satu sehingga dibutuhkan perulangan for untuk semua barisnya. Batasan akhir pada perulangan for didapat dari method getNewPatient() pada file m_pasien_baru.php.

```php
function getNewPatient(){
    $query = $this->conn->prepare("SELECT * from patient_new");
    $query->execute();
    $res = $query->get_result();

    $b = 0;
    while($row = $res->fetch_assoc()){
        $b++;
    }
    $query->close();

    return $b;
}
```

Kode Program 5.24 Method getNewPatient()

Untuk memilih pasien yang akan diverifikasi datanya, petugas mengklik tautan edit, lalu akan muncul gambar seperti berikut.



Gambar 5.5 Halaman Edit Data Pasien Baru

Nomor rekam medis yang ada pada halaman edit data pasien baru otomatis di-generate, dan tidak bisa diubah oleh petugas. Nomor rekam medis diawali dengan angka “00”. Kode program untuk men-generate nomor rekam medis ada pada edit.php di *package* view, yaitu sebagai berikut.

```php
include dirname(dirname(__FILE__)) . '../config/koneksi.php';

$sql = sprintf("SELECT MAX(nomor_medrec) AS terbesar FROM patient_old");
$res = mysql_query($sql) or die(mysql_error());
$row = mysql_fetch_array($res);
$highest = $row['terbesar'];
$nrm = $highest + 1;
// echo $row['terbesar'];

$sql2 = sprintf("INSERT INTO patient_old (nomor_medrec) VALUES ('%s')", mysql_real_escape_string($nrm));

$res2 = mysql_query($sql2) or die(mysql_error());

$nrm2 = "00" . $nrm;
```

Kode Program 5.25 Generate Nomor Rekam Medis

Agar petugas tidak bisa mengubah nomor rekam medis pada halaman edit data pasien baru, ditambahkan sebuah kata “disabled” pada kotak isian nomor rekam medis.

```
lass="col-lg-9 col-md-10 col-sm-8 col-xs-7">
v class="form-group">
<div class="form-line">
    <input type="text" id="nrm" name="nrm" class="form-control" placeholder="Masukkan Nomor Rekam Medis" value="<?php echo $nrm2 ?>" style="background-color:yellow" disabled>
</div>
iv>
```

Kode Program 5.26 Disabled pada Kotak Isian Nomor Rekam Medis

Isian pada halaman tersebut langsung terisi karena adanya method showBaru() yang ada pada m_pasien_baru.php

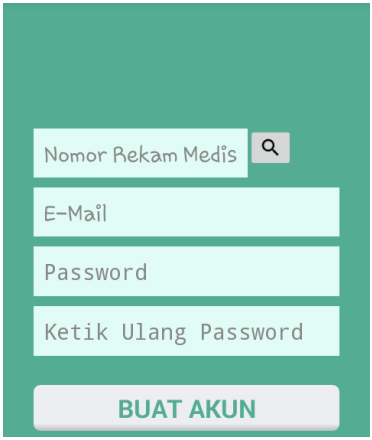
```php
function showBaru($id){
    $query = $this->conn->prepare("SELECT p.*, u.email, u.tgl_daftar, u
    $query->bind_param("s", $id);
    $query->execute();
    $res = $query->get_result();

    $data = array(array(0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16));
    $b = 0;
    while($row = $res->fetch_assoc()){
        $data[$b][0] = "{$row['nomor_medrec']}";
        $data[$b][1] = "{$row['nama']}";
        $data[$b][2] = "{$row['tempat_lahir']}";
        $data[$b][3] = "{$row['tgl_lahir']}";
        $data[$b][4] = "{$row['jenis_kelamin']}";
        $data[$b][5] = "{$row['telepon']}";
        $data[$b][6] = "{$row['alamat']}";
        $data[$b][7] = "{$row['jenis_id']}";
        $data[$b][8] = "{$row['no_identitas']}";
        $data[$b][9] = "{$row['agama']}";
        $data[$b][10] = "{$row['pendidikan']}";
        $data[$b][11] = "{$row['pekerjaan']}";
        $data[$b][12] = "{$row['status']}";
        $data[$b][13] = "{$row['anggota_keluarga']}";
        $data[$b][14] = "{$row['email']}";
        $data[$b][15] = "{$row['tgl_daftar']}";
        $data[$b][16] = "{$row['valid']}";

        if($data[$b][4] == 'M'){
            $data[$b][4] = 'Laki-laki';
        }
        else if($data[$b][4] == 'F'){
            $data[$b][4] = 'Perempuan';
        }
```

Kode Program 5.27 Method showBaru()

Sistem mengirimkan sebuah *query* ke database MySQL, lalu hasil dari *query* dijadikan sebuah array dua dimensi, dan tiap *value* hasil dari *query* dimasukkan ke dalam array. Array inilah yang akan dimasukkan ke dalam kotak-kotak isian pada halaman edit data pasien baru.

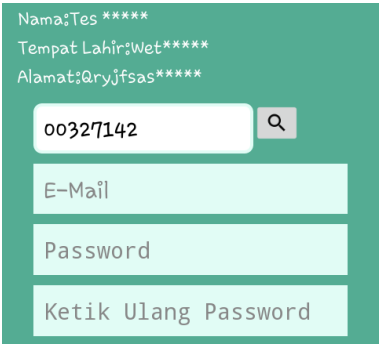
5.5.3 Fungsi Register Pasien Lama

Pasien lama mengisikan nomor rekam medis, *email* dan *password* ketika akan membuat akun di aplikasi.



Gambar 5.6 Form Pendaftaran Pasien Lama

Pasien juga bisa mengecek apakah dirinya sudah memiliki nomor rekam medis atau belum dengan cara mengisikan nomor rekam medis, lalu menekan tombol *search*.



Gambar 5.7 Pencarian Data Pasien dengan Nomor Rekam Medis pada Form Pendaftaran Pasien Lama

Data nama, tempat lahir, dan alamat yang ditampilkan hanya beberapa karakter di depan. Karakter-karakter setelah itu disensor dengan symbol bintang (*) karena alasan privasi.

Ketika menekan tombol search, sistem melakukan request ke *library* volley untuk memanggil file patient_search.php

```php
if (isset($_POST['nomor_medrec'])){
    // menerima parameter post dari aplikasi
    $nomor_medrec = $_POST['nomor_medrec'];

    // mencari pasien berdasarkan nomor rekam medis
    $searchObj = $db->searchPatient($nomor_medrec);

    if ($searchObj){
        // pasien ditemukan
        $response["error"] = FALSE;
        $response["searchObj"]["nama"] = substr($searchObj["nama"], 0, 4) . "******";
        $response["searchObj"]["tempat_lahir"] = substr($searchObj["tempat_lahir"], 0, 3) . "******";
        $response["searchObj"]["alamat"] = substr($searchObj["alamat"], 0, 8) . "******";
        echo json_encode($response);
    }
    else {
        // pasien tidak ditemukan
        $response["error"] = TRUE;
        $response["error_msg"] = "Pasien dengan nomor rekam medis " . $nomor_medrec ." tidak ditemukan.";
        echo json_encode($response);
    }
}
```

Kode Program 5.28 patient_search.php

File tersebut akan memanggil method searchPatient() yang ada pada DB_Functions.php

```php
// Method untuk mencari pasien
public function searchPatient($nomor_medrec) {
    $stmt = $this->conn->prepare("SELECT nama, tempat_lahir, alamat FROM patient_old WHERE nomor_medrec = ?");

    $stmt->bind_param("s", $nomor_medrec);

    if ($stmt->execute()) {
        $searchObj = $stmt->get_result()->fetch_assoc();
        $stmt->close();

        return $searchObj;
    }
    else {
        return FALSE;
    }
```

Kode Program 5.29 Method searchPatient()

Method tersebut memungkinkan sistem untuk melakukan query ke database, setelah itu hasilnya dicetak dalam bentuk JSON pada file patient_search.php.

Setelah pasien mengisi seluruh kotak isian dan menekan tombol buat akun, sistem akan melakukan request ke *library* volley untuk memanggil file user_register.php di API.

```php
// check if user is already existed with the same nrm
if($db->isNRMexisted($nomor_medrec)){
    // user already existed
    $response["error"] = TRUE;
    $response["error_msg"] = "Pasien dengan nomor rekam medis " . $nomor_medrec . " telah mendaftar.";
    echo json_encode($response);
}
else if($db->isPatientNotRegistered($nomor_medrec)){
    // pasien belum pernah melakukan pendaftaran
    $response["error"] = TRUE;
    $response["error_msg"] = "Anda belum pernah mendaftar sebagai pasien di RSUD Bangil. Silahkan melakukan pendaftaran pasien baru!";
    echo json_encode($response);
}
else if ($db->isEmailExisted($email)) {
    // email already existed
    $response["error"] = TRUE;
    $response["error_msg"] = "Sudah ada orang yang mendaftar dengan email " . $email;
    echo json_encode($response);
}
```

Kode Program 5.30 user_register.php

Pada file itu, akan dilakukan tiga kali pengecekan. Pengecekan yang pertama adalah dengan memanggil method isNRMexisted() dari DB_Functions.php untuk mengecek apakah pasien dengan nomor rekam medis tertentu sudah pernah mendaftar sebelumnya di aplikasi ini.

```php
public function isNRMexisted($nomor_medrec) {
    $stmt = $this->conn->prepare("SELECT nomor_medrec from user WHERE nomor_medrec = ?");

    $stmt->bind_param("s", $nomor_medrec);

    $stmt->execute();

    $stmt->store_result();

    if ($stmt->num_rows > 0) {
        /** nomor_medrec telah digunakan **/
        $stmt->close();
        return true;
    } else {
        /** nomor_medrec belum digunakan **/
        $stmt->close();
        return false;
    }
```

Kode Program 5.31 Method isNRMexisted()

Pengecekan yang kedua adalah dengan memanggil method isPatientNotRegistered() dari DB_Functions.php untuk mengecek apakah pasien sudah pernah mendaftar sebelumnya di sistem yang sudah ada.

```php
// Mengecek apakah pasien dengan suatu nomor_medrec telah terdaftar di database RSUD atau belum
public function isPatientNotRegistered($nomor_medrec) {
    $stmt = $this->conn->prepare("SELECT nomor_medrec from patient_old WHERE nomor_medrec = ?");

    $stmt->bind_param("s", $nomor_medrec);

    $stmt->execute();

    $stmt->store_result();

    if ($stmt->num_rows > 0) {
        /** Pasien telah mendaftar **/
        $stmt->close();
        return false;
    } else {
        /** Pasien belum mendaftar **/
        $stmt->close();
        return true;
    }
```

Kode Program 5.32 Method isPatientNotRegistered()

Pengecekan yang ketiga adalah dengan memanggil method isEmailExisted() untuk mengecek apakah sebuah *email* tertentu pernah digunakan untuk mendaftar di aplikasi ini.

```php
// Mengecek apakah email telah digunakan
public function isEmailExisted($email) {
    $stmt = $this->conn->prepare("SELECT email from user WHERE email = ?");

    $stmt->bind_param("s", $email);

    $stmt->execute();

    $stmt->store_result();

    if ($stmt->num_rows > 0) {
        /** email telah digunakan **/
        $stmt->close();
        return true;
    } else {
        /** email belum digunakan **/
        $stmt->close();
        return false;
    }
```

Kode Program 5.33 Method isEmailExisted()

Untuk menyamakan isian user antara pasien lama dan pasien baru, maka data yang diperlukan harus sama kolomnya. Maka dari itu, data untuk pasien lama juga perlu ditambahkan nomor identitas.

```php
public function getID($nomor_medrec) {
    $stmt = $this->conn->prepare("SELECT jenis_id, no_identitas FROM patient_old WHERE nomor_medrec = ?");

    $stmt->bind_param("s", $nomor_medrec);

    if ($stmt->execute()) {
        $getId = $stmt->get_result()->fetch_assoc();
        $stmt->close();

        return $getId;
    }
    else {
        return NULL;
    }
```

Kode Program 5.34 Method getID()

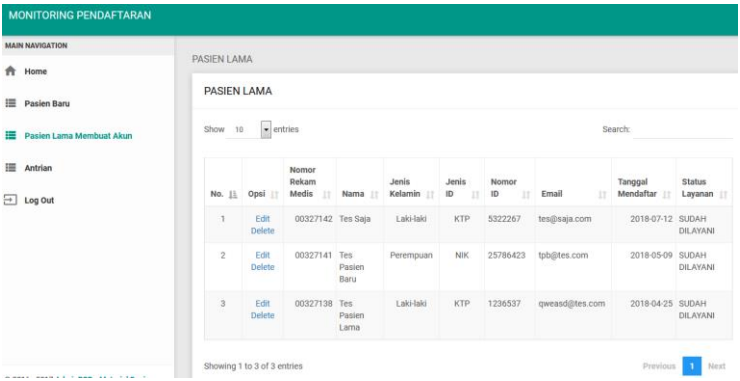
Setelah itu, sistem memanggil method registerUser() dari DB_Functions.php untuk menyimpan data pendaftaran.

```php
public function registerUser($nomor_medrec, $email, $password, $jenis_id, $no_identitas) {
    $salt = md5(rand());
    $salt = substr($salt, 0, 9);
    $encrypted_password = base64_encode(hash('sha512', $password . $salt));

    $stmt = $this->conn->prepare("INSERT INTO user(nomor_medrec, email, encrypted_password, salt, jenis_id, no_i
    $stmt->bind_param("ssssss", $nomor_medrec, $email, $encrypted_password, $salt, $jenis_id, $no_identitas);
    $result = $stmt->execute();
    $stmt->close();

    // check for successful store
    if ($result) {
        $stmt = $this->conn->prepare("SELECT * FROM user WHERE nomor_medrec = ?");
        $stmt->bind_param("s", $nomor_medrec);
        $stmt->execute();
        $user = $stmt->get_result()->fetch_assoc();
        $stmt->close();

        return $user;
    } else {
        return false;
    }
}
```

Kode Program 5.35 Method registerUser()

Password user dienkripsi dengan enkripsi sha generasi 2 ditambah salt untuk keunikan, lalu dimasukkan encoding base64_encode agar karakter yang asing dapat tersimpan.

Petugas kemudian membuka aplikasi "Monitoring Pendaftaran" dan mengklik menu "Pasien Lama Membuat Akun", seperti pada gambar berikut.



Gambar 5.8 Halaman Pasien Lama Membuat Akun

Isian dari tabel di atas di-generate dari fungsi pasien_lama.php yang ada pada aplikasi "Monitoring Pendaftaran", *package* view.

```php
<?php
    $mLama = new m_pasien_lama;
    $data = $mLama->showOldPatient();

    for($l = 0; $l < $mLama->getOldPatient(); $l++){

    echo "<tr>
        <td align='center'>".($l+1)."</td>
        <td align='center'>
            <a href='../view/editLama.php?l=" . $data[$l][0] . "'>Edit</a>
            <a href='../view/deleteLama.php?l=" . $data[$l][0] . "'>Delete</a>
        </td>
        <td align='right' style='padding-right:5px;'>".$data[$l][0]."</td>
        <td align='left' style='padding-left:5px;'>".$data[$l][1]."</td>
        <td align='center'>".$data[$l][2]."</td>
        <td align='center'>".$data[$l][3]."</td>
        <td align='left' style='padding-left:5px;'>".$data[$l][4]."</td>
        <td align='left' style='padding-left:5px;'>".$data[$l][5]."</td>
        <td align='right' style='padding-right:5px;'>".$data[$l][6]."</td>
        <td align='left' style='padding-left:5px;'>".$data[$l][7]."</td>
        </tr>";
    }
```

Kode Program 5.36 pasien_lama.php

Sebelum menuju ke situ, sistem memanggil fungsi c_pasien_lama.php terlebih dahulu yang ada pada *package* controller. Dari situ, pasien_lama.php yang ada pada *package* view akan dihubungkan dengan m_pasien_lama.php yang ada pada *package* model. Pada gambar di atas, isian didapat dari fungsi showOldPatient() pada m_pasien_lama.php. Fungsi tersebut adalah sebagai berikut.

```
// Ini method buat menampilkan tabel pasien lama yang membuat akun
function showOldPatient(){
    $query = $this->conn->prepare("SELECT u.*, p.nomor_medrec, p.nama, p.jenis_kelam
    $query->execute();
    $res = $query->get_result();

    $data = array(array(0,1,2,3,4,5,6,7));
    $l = 0;
    // while($row = mysql_fetch_array($res)){
    while($row = $res->fetch_assoc()){
    //echo "tes";
        $data[$l][0] = "{$row['nomor_medrec']}";
        $data[$l][1] = "{$row['nama']}";
        $data[$l][2] = "{$row['jenis_kelamin']}";
        $data[$l][3] = "{$row['jenis_id']}";
        $data[$l][4] = "{$row['no_identitas']}";
        $data[$l][5] = "{$row['email']}";
        $data[$l][6] = "{$row['tgl_daftar']}";
        $data[$l][7] = "{$row['valid']}";

        if($data[$l][2] == 'M'){
            $data[$l][2] = 'Laki-laki';
        }
        else if($data[$l][2] == 'F'){
            $data[$l][2] = 'Perempuan';
        }

        if($data[$l][7] == 1){
            $data[$l][7] = 'SUDAH DILAYANI';
        }
        else {
            $data[$l][7] = 'BELUM DILAYANI';
        }

        $data[$l][4] = str_replace(' ', ' ', $data[$l][4]);
```

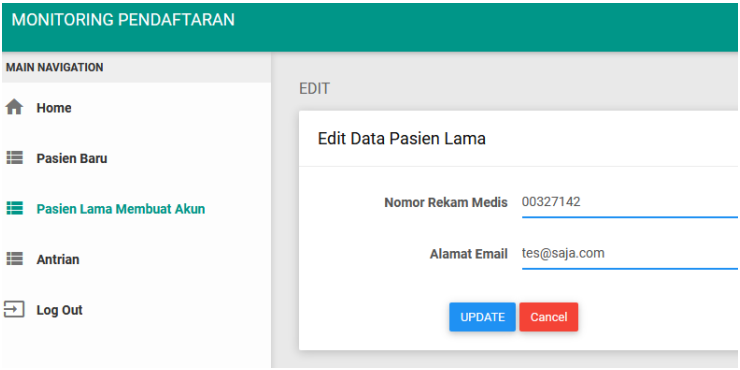
Kode Program 5.37 Method showOldPatient()

Pada tabel, bisa saja terdapat lebih dari 1 baris. Oleh karena itu dibutuhkan sebuah perulangan for untuk mengisikan baris satu per satu. Batasan akhir dari perulangan for didapat dari fungsi getOldPatient() pada m_pasien_lama.php, yaitu sebagai berikut.

```php
// Ini method untuk mengambil data pasien lama dari database
function getOldPatient(){
    $query = $this->conn->prepare("SELECT * from user WHERE nomor_medrec IS NOT NULL");
    $query->execute();
    $res = $query->get_result();

    $l = 0;
    while($row = $res->fetch_assoc()){
        $l++;
    }
    $query->close();

    return $l;
}
```

Kode Program 5.38 Method getOldPatient()

Untuk memilih pasien yang akan diverifikasi datanya, petugas mengklik tautan edit.



Gambar 5.9 Halaman Edit Data Pasien Lama

Kotak isian pada halaman tersebut otomatis terisi karena method showLama(), yaitu sebagai berikut.

```php
// Ini method buat menampilkan tabel edit pasien lama yang membuat akun
function showLama($id){
    $query = $this->conn->prepare("SELECT p.*, u.email, u.tgl_daftar, u.valid FROM us
    $query->bind_param("s", $id);
    $query->execute();
    $res = $query->get_result();

    $data = array(array(0,1,2,3,4,5,6,7));
    $l = 0;
    while($row = $res->fetch_assoc()){
        $data[$l][0] = "{$row['nomor_medrec']}";
        $data[$l][1] = "{$row['nama']}";
        $data[$l][2] = "{$row['jenis_kelamin']}";
        $data[$l][3] = "{$row['jenis_id']}";
        $data[$l][4] = "{$row['no_identitas']}";
        $data[$l][5] = "{$row['email']}";
        $data[$l][6] = "{$row['tgl_daftar']}";
        $data[$l][7] = "{$row['valid']}";

        if($data[$l][2] == 'M'){
            $data[$l][2] = 'Laki-laki';
        }
        else if($data[$l][2] == 'F'){
            $data[$l][2] = 'Perempuan';
        }

        $data[$l][4] = str_replace('_', ' ', $data[$l][4]);

        $l++;
    }
    $query->close();

    return $data;
}
```

Kode Program 5.39 Method showLama()

Sistem mengirimkan sebuah *query* ke database MySQL, lalu hasil dari *query* dijadikan sebuah array dua dimensi, dan tiap *value* hasil dari *query* dimasukkan ke dalam array. Array inilah yang akan dimasukkan ke dalam kotak-kotak isian pada halaman edit data pasien lama.

5.5.4 Fungsi Pendaftaran Berobat

Pasien mengisikan data diri pada form seperti pada gambar berikut.

Kode Program 5.40 Form Pendaftaran Berobat

Selanjutnya sistem akan meneruskan masukan dari user ke *library* volley untuk memanggil fungsi antrian.php yang ada pada API. Data nomor rekam medis, tanggal kunjungan, dan poli tujuan dikirimkan ke halaman selanjutnya melalui intent, seperti pada gambar berikut.

```
private void registerAntrian(final String rDate, final String poli, final String dokter, final String jamKunjunga
    String tag = "request antrian";

    progress.setMessage("Memuat");
    showProgress();

    SQLiteHandler sqliteDB = new SQLiteHandler(getActivity().getApplicationContext());
    HashMap<String, String> user = sqliteDB.getUserDetails();
    final String name = user.get("nama");
    final String nrm = user.get("nrm");

    StringRequest requestString = new StringRequest(Request.Method.POST, AppConfig.URL_REGISTER_ANTRIAN, new Resp
        @Override
        public void onResponse(String response) {
            Log.d(TAG, "Register Antrian Response: " + response);
            hideProgress();

            try {
                JSONObject object = new JSONObject(response);
                boolean error = object.getBoolean("error");

                //mengecek error di JSON
                if (!error) {
                    // berhasil mendaftar antrian

                    Intent intent = new Intent(getActivity().getApplicationContext(), RegisAntrianSuccess.class);
                    intent.putExtra("nomor_medrec", nrm);
                    intent.putExtra("tgl_kunjungan", rDate);
                    intent.putExtra("poli", poli);
                    startActivity(intent);
```

Kode Program 5.41 Method registerAntrian() pada Aplikasi "Registrasi Pasien"

```
    protected Map<String, String> getParams(){
        //menaruh parameter ke url pendaftaran antrian, untuk regist
        Map<String, String> params = new HashMap<~>();
        params.put("nama", name);
        params.put("nomor_medrec", nrm);
        params.put("tgl_regis", rDate);
        params.put("poli", poli);
        params.put("dokter", dokter);
        params.put("jam_kunjungan", jamKunjungan);
        params.put("keluhan", keluhan);
        params.put("asuransi", asuransi);
        params.put("no_asuransi", noAsuransi);

        return params;
    }
};

AppController.getInstance().addToRequestQueue(requestString, tag);
```

Kode Program 5.42 Lanjutan Method registerAntrian()

Pada fungsi antrian.php, sistem akan kembali merujuk pada DB_Functions.php. Sebelum menambahkan data pendaftaran ke database, sistem akan mengecek terlebih dahulu apakah user sudah pernah mendaftar pada hari itu dengan poli tujuan yang sama atau belum.

```php
// Mengecek apakah pasien telah mendaftar
public function isRegistered($nomor_medrec, $tgl_regis, $poli) {
    $stmt = $this->conn->prepare("SELECT * from antrian WHERE (nomor_medrec = ? AND tgl_regis = ? AND poli = ?)");

    $stmt->bind_param("sss", $nomor_medrec, $tgl_regis, $poli);

    $stmt->execute();

    $stmt->store_result();

    if ($stmt->num_rows > 0) {
        /** pasien sudah registrasi **/
        $stmt->close();
        return true;
    } else {
        /** pasien belum registrasi **/
        $stmt->close();
        return false;
    }
}
```

Kode Program 5.43 Method isRegistered()

Setelah itu, sistem akan menyimpan data pendaftaran ke database. Untuk perkiraan waktu giliran, setiap pasien diasumsikan membutuhkan waktu 2 menit ketika berada di loket.

```php
// Menyimpan registrasi pasien
public function registerAntrian($nama, $nomor_medrec, $tgl_regis, $poli, $dokter, $jam_kunjungan, $keluhan, $asuransi, $no_asuransi) {
    $stmt1 = $this->conn->prepare("SELECT MAX(no_antrian) AS terakhir FROM antrian WHERE tgl_regis = ?");
    $stmt1->bind_param("s", $tgl_regis);
    $stmt1->execute();
    $row1 = $stmt1->get_result()->fetch_assoc();
    $last = $row1['terakhir'];
    $no_antrian = $last + 1;
    $stmt1->close();

    $giliran = new DateTime($tgl_regis . " 07:00:00");
    if($no_antrian > 1){
        for($x = 2; $x <= $no_antrian; $x++){
            $giliran->add(new DateInterval('PT' . 2 . 'M'));
        }
    }
    $perkiraan_waktu = $giliran->format('Y-m-d H:i:s');

    $stmt = $this->conn->prepare("INSERT INTO antrian(no_antrian, nama, nomor_medrec, tgl_regis, poli, dokter, jam_kunjungan, keluhan,
    $stmt->bind_param("ssssssssss", $no_antrian, $nama, $nomor_medrec, $tgl_regis, $poli, $dokter, $jam_kunjungan, $keluhan, $asuransi

    $result = $stmt->execute();
    $stmt->close();

    /** check for successful store **/
    if ($result) {
        $stmt1 = $this->conn->prepare("SELECT * FROM antrian WHERE nomor_medrec = ? AND tgl_regis = ? AND poli = ?");
        $stmt1->bind_param("sss", $nomor_medrec, $tgl_regis, $poli);
        $stmt1->execute();
        $patient = $stmt1->get_result()->fetch_assoc();
        $stmt1->close();

        return $patient;
    } else {
```

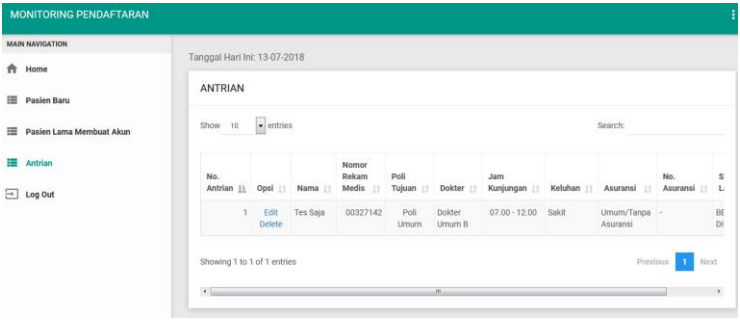
Kode Program 5.44 Method registerAntrian() pada API

Setelah itu, sistem akan membuat halaman dalam bentuk JSON. JSON tersebut perlu dibuat agar ketika ada error, penulis bisa mengetahui apakah letak kesalahannya ada pada source code aplikasinya atau API nya. Untuk mengetes API nya saja, penulis menggunakan sebuah tools berupa addon yang bernama RESTClient.

```
else{
    // mendaftar
    $patient = $db->registerAntrian($nama, $nomor_medrec, $tgl_regis, $poli, $dokter, $jam_kunjungan, $keluhan, $asuransi, $no_asuransi);
    if ($patient) {
        // pasien berhasil terdaftar
        $response["error"] = FALSE;
        $response["patient"]["no_antrian"] = $patient["no_antrian"];
        $response["patient"]["nama"] = $patient["nama"];
        $response["patient"]["nomor_medrec"] = $patient["nomor_medrec"];
        $response["patient"]["tgl_regis"] = $patient["tgl_regis"];
        $response["patient"]["poli"] = $patient["poli"];
        $response["patient"]["dokter"] = $patient["dokter"];
        $response["patient"]["jam_kunjungan"] = $patient["jam_kunjungan"];
        $response["patient"]["keluhan"] = $patient["keluhan"];
        $response["patient"]["asuransi"] = $patient["asuransi"];
        $response["patient"]["no_asuransi"] = $patient["no_asuransi"];
        echo json_encode($response);
    } else {
        // pasien gagal terdaftar
        $response["error"] = TRUE;
        $response["error_msg"] = "Error: Gagal registrasi. Silahkan coba lagi!";
        echo json_encode($response);
    }
}
```

Kode Program 5.45 antrian.php pada API

Pada saat proses verifikasi, petugas loket membuka aplikasi “Monitoring Pendaftaran”, dan mengklik menu antrian.



Gambar 5.10 Halaman Antrian pada Aplikasi "Monitoring Pendaftaran"

Isian dari tabel di atas di-generate dari fungsi antrian.php yang ada pada aplikasi “Monitoring Pendaftaran”, *package* view.

```php
<?php
    $mAntrian = new m_antrian;
    $data = $mAntrian->showQueue();

    for($a = 0; $a < $mAntrian->getQueue(); $a++){

    echo "<tr>
        <td align='right' style='padding-right:5px;'>".$data[$a][1]."</td>
        <td align='center'>
            <a href='../view/editAntrian.php?a=" . $data[$a][0] . "'>Edit</a>
            <a href='../view/deleteAntrian.php?a=" . $data[$a][0] . "'>Delete</a>
        </td>
        <td align='left' style='padding-left:5px;'>".$data[$a][2]."</td>
        <td align='right' style='padding-right:5px;'>".$data[$a][3]."</td>
        <td align='center'>".$data[$a][4]."</td>
        <td align='left' style='padding-left:5px;'>".$data[$a][5]."</td>
        <td align='center'>".$data[$a][6]."</td>
        <td align='left' style='padding-left:5px;'>".$data[$a][7]."</td>
        <td align='left' style='padding-left:5px;'>".$data[$a][8]."</td>
        <td align='left' style='padding-left:5px;'>".$data[$a][9]."</td>
        <td align='left' style='padding-left:5px;'>".$data[$a][10]."</td>
        </tr>";
    }
    echo "</table>";
?>
```

Kode Program 5.46 antrian.php pada Aplikasi "Monitoring Pendaftaran"

Sebelum menuju ke situ, sistem memanggil fungsi c_antrian.php terlebih dahulu yang ada pada *package* controller. Dari situ, antrian.php yang ada pada *package* view akan dihubungkan dengan m_antrian.php yang ada pada *package* model. Pada gambar di atas, isian didapat dari fungsi showQueue() pada m_antrian.php. Fungsi tersebut adalah sebagai berikut.

```php
function showQueue(){
    // include dirname(dirname(__FILE__)) . '../config/koneksi.php';

    $date = date("Y-m-d");
    $query = $this->conn->prepare("SELECT * FROM antrian WHERE tgl_regis = ? ORDER BY no_antrian ASC");
    $query->bind_param("s", $date);
    $query->execute();
    $res = $query->get_result();

    $data = array(array(0,1,2,3,4,5,6,7,8,9));
    $a = 0;

    while($row = $res->fetch_assoc()){

        $data[$a][0] = "{$row['no']}";
        $data[$a][1] = "{$row['no_antrian']}";
        $data[$a][2] = "{$row['nama']}";
        $data[$a][3] = "{$row['nomor_medrec']}";
        $data[$a][4] = "{$row['poli']}";
        $data[$a][5] = "{$row['dokter']}";
        $data[$a][6] = "{$row['jam_kunjungan']}";
        $data[$a][7] = "{$row['keluhan']}";
        $data[$a][8] = "{$row['asuransi']}";
        $data[$a][9] = "{$row['no_asuransi']}";
        $data[$a][10] = "{$row['status_layanan']}";

        if($data[$a][10] == 1){
            $data[$a][10] = 'SUDAH DILAYANI';
        }
        else {
            $data[$a][10] = 'BELUM DILAYANI';
        }

        $a++;
```

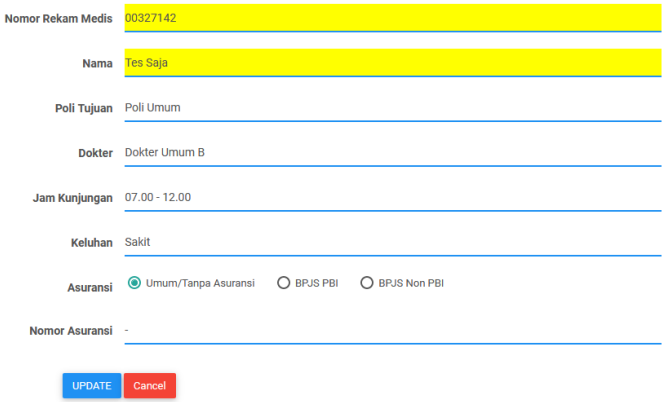
Kode Program 5.47 Method showQueue()

Pada tabel, bisa saja terdapat lebih dari 1 baris. Oleh karena itu dibutuhkan sebuah perulangan for untuk mengisikan baris satu per satu. Batasan akhir dari perulangan for didapat dari fungsi getQueue() pada m_antrian.php, yaitu sebagai berikut.

```php
// Ini method untuk mengambil data antrian dari database
function getQueue(){
    // include dirname(dirname(__FILE__)) . '../config/koneksi.php';

    $date = date("Y-m-d");
    $query = $this->conn->prepare("SELECT * from antrian WHERE tgl_regis = ? ORDER BY no_antrian ASC");
    $query->bind_param("s", $date);
    $query->execute();
    $res = $query->get_result();

    $a = 0;
    while($row = $res->fetch_assoc()){
        $a++;
    }
    // mysql_close();
    $query->close();

    return $a;
}
```

Kode Program 5.48 Method getQueue()

Untuk memilih pasien yang akan diverifikasi datanya, petugas mengklik tautan edit.



Gambar 5.11 Halaman Edit Data Pendaftaran Berobat

Untuk men-generate data pada database langsung diisikan ke tiap kotak isian, dibutuhkan fungsi showAntrian() yang ada pada m_antrian.php, yaitu seperti berikut.

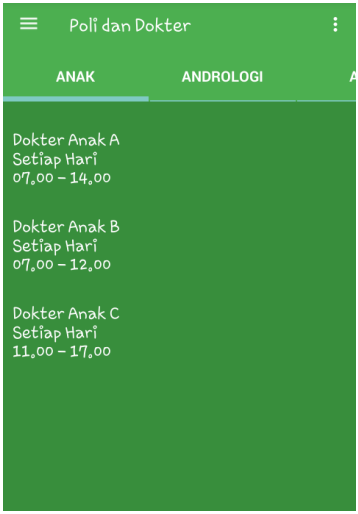
```php
// Ini method buat menampilkan tabel edit antrian
function showAntrian($id){
    // include dirname(dirname(__FILE__)) . '../config/koneksi.php';

    $query = $this->conn->prepare("SELECT * FROM antrian WHERE no = ?");
    $query->bind_param("s", $id);
    $query->execute();
    $res = $query->get_result();

    $data = array(array(0,1,2,3,4,5,6,7,8,9,10));
    $a = 0;
    while($row = $res->fetch_assoc()){
        $data[$a][0] = "{$row['no']}";
        $data[$a][1] = "{$row['nomor_medrec']}";
        $data[$a][2] = "{$row['nama']}";
        $data[$a][3] = "{$row['poli']}";
        $data[$a][4] = "{$row['dokter']}";
        $data[$a][5] = "{$row['jam_kunjungan']}";
        $data[$a][6] = "{$row['keluhan']}";
        $data[$a][7] = "{$row['asuransi']}";
        $data[$a][8] = "{$row['no_asuransi']}";
        $data[$a][9] = "{$row['status_layanan']}";
        $data[$a][10] = "{$row['no_antrian']}";

        $a++;
    }
    // mysql_close();
    $query->close();

    return $data;
}
```

Kode Program 5.49 Method showAntrian()

Sistem mengirimkan sebuah *query* ke database MySQL, lalu hasil dari *query* dijadikan sebuah array dua dimensi, dan tiap *value* hasil dari *query* dimasukkan ke dalam array. Array inilah yang akan dimasukkan ke dalam kotak-kotak isian pada halaman edit data antrian.

5.5.5 Fungsi Pengecekan Jadwal Poli dan Dokter

Pasien mengecek jadwal poli dan dokter seperti pada gambar berikut.



Gambar 5.12 Halaman Pengecekan Poli dan Dokter pada Aplikasi "Registrasi Pasien"

5.5.6 Fungsi Pengecekan Riwayat

Ketika pasien mengecek riwayat, sistem langsung menggunakan *library* volley untuk melakukan request ke history.php pada API.

```
StringRequest requestString = new StringRequest(Request.Method.POST, AppConfig.URL_HISTORY, new Response.Listener<String>() {
```

Kode Program 5.50 Aplikasi "Registrasi Pasien" memanggil file history.php pada API

```php
if(isset($_POST['nomor_medrec']) && isset($_POST['nama'])){
    $nomor_medrec = $_POST['nomor_medrec'];
    $nama = $_POST['nama'];
    $user = $db->cekHistory($nomor_medrec, $nama);

    if($user){
        $response["error"] = FALSE;

        for($a = 0; $a<$db->countHistoryData($nomor_medrec, $nama); $a++){
            $response["user"][$a+1]["nomor_medrec"] = $user[$a][0];
            $response["user"][$a+1]["nama"] = $user[$a][1];
            $response["user"][$a+1]["tgl_kunjungan"] = $user[$a][2];
            $response["user"][$a+1]["poli_tujuan"] = $user[$a][3];
            $response["user"][$a+1]["dokter"] = $user[$a][4];
            $response["user"][$a+1]["keluhan"] = $user[$a][5];
```

Kode Program 5.51 history.php

Sistem mengecek riwayat kunjungan pasien dengan method cekHistory() dari DB_Functions.php, untuk mengisi baris pada tabel.

```php
// Mengecek history
public function cekHistory($nomor_medrec, $nama){
    $stmt = $this->conn->prepare("SELECT * FROM history WHERE (nomor_medrec = ? AND nama = ?)");
    $stmt->bind_param("ss", $nomor_medrec, $nama);

    if($stmt->execute()){
        $user = array(array(0,1,2,3,4,5));
        $a = 0;

        $result = $stmt->get_result();
        while($row = $result->fetch_assoc()){
            $user[$a][0] = "{$row['nomor_medrec']}";
            $user[$a][1] = "{$row['nama']}";
            $user[$a][2] = "{$row['tgl_kunjungan']}";
            $user[$a][3] = "{$row['poli_tujuan']}";
            $user[$a][4] = "{$row['dokter']}";
            $user[$a][5] = "{$row['keluhan']}";

            $a++;
        }
        $stmt->close();

        return $user;
    }
    else {
        $stmt->close();
        return false;
    }
}
```

Kode Program 5.52 Method cekHistory()

Untuk mengecek batasan array jumlah barisnya, sistem menggunakan method countHistoryData pada DB_Functions.php.

```php
// menghitung batasan array untuk perulangan (for) pada history.php
public function countHistoryData($nomor_medrec, $nama){
    $stmt = $this->conn->prepare("SELECT * from history WHERE nomor_medrec = ? AND nama = ?");
    $stmt->bind_param("ss", $nomor_medrec, $nama);
    $stmt->execute();
    $result = $stmt->get_result();

    $a = 0;
    while($row = $result->fetch_assoc()){
        $a++;
    }
    // mysql_close();
    $stmt->close();

    return $a;
}
```

Kode Program 5.53 Method countHistoryData()

Keluaran JSON dari API aplikasi diolah kembali seperti yang ada pada kode program berikut.

```
JSONObject jArray1 = object.optJSONObject("user");
if (jArray1 == null) return;

for(int i = 1; i <= jArray1.length(); i++){
    tRow = new TableRow(getActivity().getApplicationContext());
    tRow.setBackgroundColor(getResources().getColor(R.color.bg_main));
    tRow.setLayoutParams(new TableRow.LayoutParams(TableRow.LayoutParams.F

    JSONObject baris = jArray1.optJSONObject(String.valueOf(i));
    String tgl_kunjungan = baris.optString("tgl_kunjungan");
    String poli_tujuan = baris.optString("poli_tujuan");
    String dokter = baris.optString("dokter");
    String keluhan = baris.optString("keluhan");

    historyDB.addHistory(i, tgl_kunjungan, poli_tujuan, dokter, keluhan);
```

Kode Program 5.54 Mengolah Keluaran JSON di Halaman Pengecekan Riwayat Kunjungan

Keluaran JSON yang berupa array dengan nama “user” dijadikan obyek pada aplikasi “Registrasi Pasien” dengan *method* optJSONObject(). Penulis tidak menggunakan *method* getJSONObject() untuk halaman riwayat kunjungan karena

isian riwayat kunjungan bisa saja kosong atau bernilai *null*. Apabila isian riwayat kunjungan masih kosong, dan penulis menggunakan *method* getJSONObject(), maka akan muncul sebuah *exception* yang bernama *NullPointerException*. Tiap *value* dari array "user" kemudian dijadikan *String*, lalu ditambahkan ke tabel SQLite yang bernama historyDB pada aplikasi "Registrasi Pasien".

Hasil tampilan riwayat kunjungan adalah sebagai berikut.

Gambar 5.13 Halaman Riwayat Kunjungan

5.5.7 Fungsi Logout

Ketika ingin melakukan logout, pasien mengklik menu yang ada pada pojok kanan atas. Kode programnya adalah sebagai berikut.

```
@Override
public boolean onOptionsItemSelected(MenuItem item){
    int id = item.getItemId();

    if(id==R.id.action_logout){
        Toast.makeText(getApplicationContext(), "Keluar dari aplikasi...", Toast.LENGTH_SHORT).show();
        logoutUser();
        return true;
    }

    if (drawerToggle.onOptionsItemSelected(item)){
        return true;
    }

    return super.onOptionsItemSelected(item);
}
```

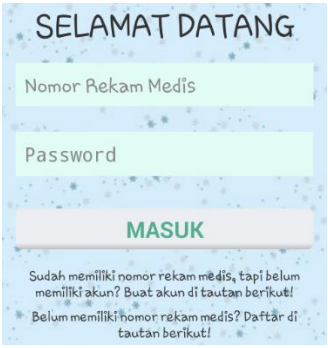
Kode Program 5.55 Method onOptionsItemSelected()

Sistem memanggil method logoutUser(), yaitu sebagai berikut.

```
private void logoutUser() {
    session.setLogin(false);
    sqliteDB.deleteUser();
//      sqlite_history.deleteHistory();

    Intent intent = new Intent(MainActivity.this, LoginActivity.class);
    startActivity(intent);
    finish();
}
```

Kode Program 5.56 Method logoutUser()

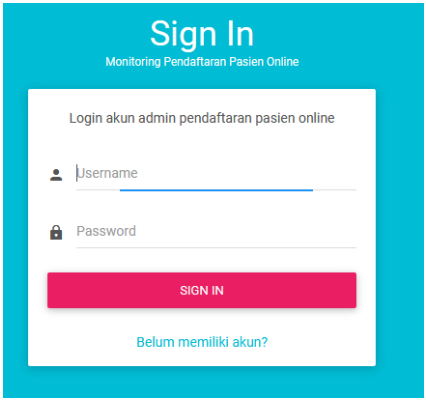
Setelah mengklik menu logout pada pojok kanan atas, session setLogin jadi false dan data user pada database sqlite dihapus, lalu aplikasi akan kembali ke halaman *login*.



Gambar 5.14 Kembali ke Halaman Login

5.5.8 Fungsi Login dan Register Aplikasi "Monitoring Pendaftaran"

Ketika baru membuka aplikasi "Monitoring Pendaftaran", petugas dihadapkan pada halaman *login*.



Gambar 5.15 Halaman Login Aplikasi "Monitoring Pendaftaran"

Hal itu disebabkan karena di folder aplikasi "Monitoring Pendaftaran" terdapat sebuah file dengan nama index.php. Sistem akan memanggil file tersebut terlebih dahulu. Kode program dari file index.php adalah sebagai berikut.

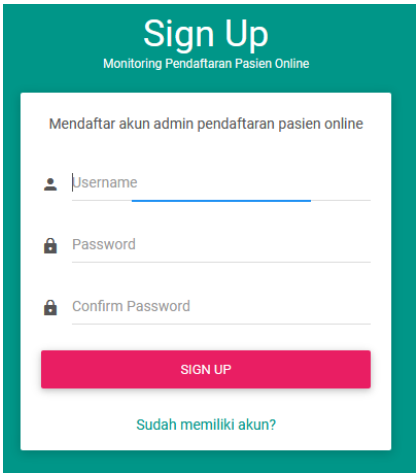
```php
<?php
    session_start();
    if(!isset($_SESSION['login'])){
        header("location:view/sign-in.html");
    }
    else {
        header("location:view/home.php");
    }
?>
```

Kode Program 5.57 index.php

Sistem mulai mengecek session, apabila petugas telah dalam keadaan *login*, maka sistem akan memanggil file home.php yang ada di *package* view, yang merupakan file untuk halaman "home". Apabila petugas belum dalam keadaan *login*, seperti

contoh kali ini, maka sistem akan memanggil file sign-in.html, yang akan menampilkan halaman *login*.

Petugas mengisikan username dan *password* yang dimiliki. Apabila belum memiliki akun di aplikasi tersebut, petugas mendaftar terlebih dahulu dengan mengklik tautan “Belum memiliki akun?”



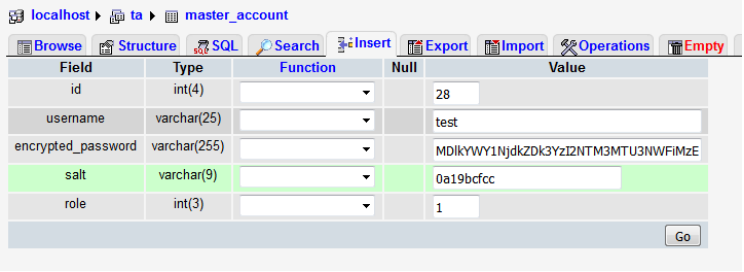
Gambar 5.16 Halaman Registrasi Aplikasi "Monitoring Pendaftaran"

Petugas mengisikan username yang diinginkan. Petugas juga mengisikan *password* dan kotak isian konfirmasi *password*. *Password* minimal harus mengandung enam karakter, sedangkan isian konfirmasi *password* harus sama dengan isian *password*. Setelah mengisi form dan mengklik tombol *sign up*, sistem akan memanggil class c_register.php yang ada di *package* controller. Class c_register.php berfungsi untuk memanggil *method* register() yang ada pada class account.php di *package* model.

```
public function register($username, $password){
    $result = false;
    $salt = md5(rand());
    $salt = substr($salt, 0, 9);
    $encrypted = base64_encode(hash('sha512', $password . $salt));

    $query = sprintf("INSERT INTO master_account(username, encrypted_password, salt) VALUES ('%s', '%s', '%s')", mysql_real_escape_string($username
    $res = mysql_query($query);
```

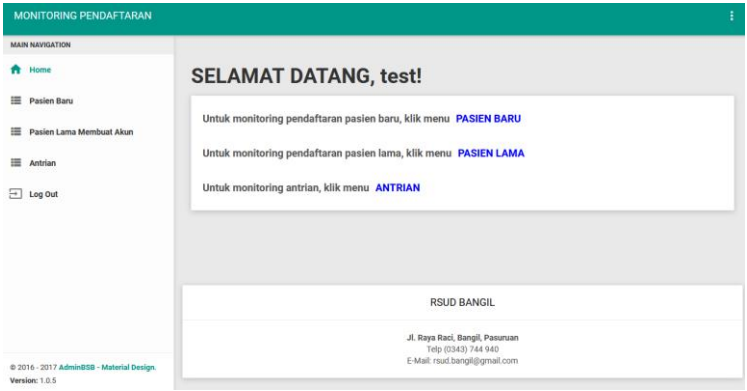
Kode Program 5.58 Method register()

Password menggunakan enkripsi sha generasi 2 ditambah dengan salt, lalu menggunakan encoding base-64 untuk menangani karakter-karakter yang asing. Setelah itu, petugas menginformasikan pegawai bagian PDE tentang *username* yang baru didaftarkannya. Pegawai PDE mengecek *username* tersebut di database MySQL, lalu mengganti *value* role dari 0 menjadi 1 dan mengklik tombol "Go".



Gambar 5.17 Memverifikasi Akun Petugas

Setelah diverifikasi data akunnya, petugas menuju halaman *login* dan melakukan *login*. Sistem akan menampilkan halaman "home" dari aplikasi "Monitoring Pendaftaran".



Gambar 5.18 Halaman Home Aplikasi "Monitoring Pendaftaran"

5.5.9 Menambah Tingkat Keamanan API Aplikasi

Untuk menghindari listing direktori pada API, penulis menggunakan sebuah file .htaccess. File tersebut juga berfungsi untuk mencegah bot dan beberapa tool hacking. Berikut ini adalah kode program dari file .htaccess.

```
RewriteEngine On
RewriteCond %{REQUEST_FILENAME} !-f
RewriteRule ^(.*)$ %{ENV:BASE}index.php [QSA,L]

## Mencegah listing direktori
IndexIgnore *
```

Kode Program 5.59 Mencegah Listing Direktori

Kode program di atas dibuat agar ketika siapapun yang mengakses API dari browser, hanya dengan memasukkan alamatnya di *address bar*, sistem akan melakukan *redirect* ke file index.php, di mana file tersebut tidak ada di folder API aplikasi. Ketika memasukkan nama folder-folder yang ada di API, sistem juga akan melakukan *redirect* ke file index.php.

Sementara itu, untuk mencegah bot-bot yang populer, kode programnya adalah sebagai berikut.

```
## Mencegah beberapa bot
RewriteEngine On
RewriteCond %{HTTP_USER_AGENT} ^BlackWidow [OR]
RewriteCond %{HTTP_USER_AGENT} ^Bot\ mailto:craftbot@yahoo.com [OR]
RewriteCond %{HTTP_USER_AGENT} ^ChinaClaw [OR]
RewriteCond %{HTTP_USER_AGENT} ^Custo [OR]
RewriteCond %{HTTP_USER_AGENT} ^DISCo [OR]
RewriteCond %{HTTP_USER_AGENT} ^Download\ Demon [OR]
RewriteCond %{HTTP_USER_AGENT} ^eCatch [OR]
RewriteCond %{HTTP_USER_AGENT} ^EirGrabber [OR]
RewriteCond %{HTTP_USER_AGENT} ^EmailSiphon [OR]
RewriteCond %{HTTP_USER_AGENT} ^EmailWolf [OR]
RewriteCond %{HTTP_USER_AGENT} ^Express\ WebPictures [OR]
RewriteCond %{HTTP_USER_AGENT} ^ExtractorPro [OR]
RewriteCond %{HTTP_USER_AGENT} ^EyeNetIE [OR]
RewriteCond %{HTTP_USER_AGENT} ^FlashGet [OR]
RewriteCond %{HTTP_USER_AGENT} ^GetRight [OR]
RewriteCond %{HTTP_USER_AGENT} ^GetWeb! [OR]
RewriteCond %{HTTP_USER_AGENT} ^Go!Zilla [OR]
RewriteCond %{HTTP_USER_AGENT} ^Go-Ahead-Got-It [OR]
RewriteCond %{HTTP_USER_AGENT} ^GrabNet [OR]
RewriteCond %{HTTP_USER_AGENT} ^Grafula [OR]
RewriteCond %{HTTP_USER_AGENT} ^HMView [OR]
RewriteCond %{HTTP_USER_AGENT} HTTrack [NC,OR]
RewriteCond %{HTTP_USER_AGENT} ^Image\ Stripper [OR]
```

Kode Program 5.60 Mencegah Bot

Setelah menuliskan nama-nama botnya, diakhiri dengan kode di bawah.

```
RewriteCond %{HTTP_USER_AGENT}
RewriteRule ^.* - [F,L]
```

Kode Program 5.61 Lanjutan Mencegah Bot

Ketika ada bot-bot seperti BlackWidow, Go!Zilla, GetWeb!, dan lain-lain, server akan memberikan respon 403 forbidden.

Sementara itu, untuk mencegah beberapa tool hacking, kode programnya adalah sebagai berikut.

```
## Mencegah beberapa tool hacking
RewriteEngine On
<IfModule mod_rewrite.c>
RewriteCond %{HTTP_USER_AGENT} ^w3af.sourceforge.net [NC,OR]
RewriteCond %{HTTP_USER_AGENT} dirbuster [NC,OR]
RewriteCond %{HTTP_USER_AGENT} nikto [NC,OR]
RewriteCond %{HTTP_USER_AGENT} SF [OR]
RewriteCond %{HTTP_USER_AGENT} sqlmap [NC,OR]
RewriteCond %{HTTP_USER_AGENT} fimap [NC,OR]
RewriteCond %{HTTP_USER_AGENT} nessus [NC,OR]
RewriteCond %{HTTP_USER_AGENT} whatweb [NC,OR]
RewriteCond %{HTTP_USER_AGENT} Openvas [NC,OR]
RewriteCond %{HTTP_USER_AGENT} jbrofuzz [NC,OR]
RewriteCond %{HTTP_USER_AGENT} libwhisker [NC,OR]
RewriteCond %{HTTP_USER_AGENT} webshag [NC,OR]
RewriteCond %{HTTP_USER_AGENT} zenmap [NC,OR]
RewriteCond %{HTTP:Acunetix-Product} ^WVS
RewriteRule ^.* http://127.0.0.1/ [R=301,L]
</IfModule>
```

Kode Program 5.62 Mencegah Tool Hacking

Baris kode yang mengandung *command* "RewriteRule" berfungsi untuk melakukan *redirect* ke 127.0.0.1, sedangkan baris-baris di atasnya adalah untuk mendeteksi tool hacking. Ketika melakukan *redirect*, tool-tool hacking tersebut tidak akan ikut melakukannya sehingga webserver tidak mengeksekusi request-request dari dari tool-tool hacking yang telah dituliskan namanya.

BAB VI
HASIL DAN PEMBAHASAN

6.1 Pengujian Aplikasi

Aplikasi diuji coba menggunakan *mobile phone* dengan spesifikasi:

Tabel 6.1 Spesifikasi Mobile Phone untuk Pengujian Aplikasi "Registrasi Pasien"

Hardware/Fungsi	Spesifikasi
Sistem Operasi	Android Lollipop 5.0.1
CPU	Dual-core 1.3 GHz Cortex-A7
RAM	512 MB
Memori Internal	4 GB
Network Connection	Yes

Aplikasi "Monitoring Pendaftaran" dan API diuji menggunakan teknologi:

Tabel 6.2 Teknologi unutk Pengujian Aplikasi "Monitoring Pendaftaran" dan API Aplikasi

Fungsi	Software
Webserver	Apache 1.7.4
Database	MySQL 5.5.8
Browser	Firefox 61.0

6.1.1 Pengujian Fungsional Aplikasi "Registrasi Pasien"

Tabel 6.3 Pengujian Fungsional Aplikasi "Registrasi Pasien"

Kode Test Case	Nama Test Case	Status
TC-01	Login	Terpenuhi
TC-02	Register pasien baru	Terpenuhi
TC-03	Register pasien lama	Terpenuhi
TC-04	Pendaftaran berobat	Terpenuhi
TC-05	Pengecekan Jadwal Poli dan Dokter	Terpenuhi
TC-06	Pengecekan riwayat kunjungan	Terpenuhi
TC-07	Logout	Terpenuhi

6.1.2 Pengujian Fungsional Aplikasi "Monitoring Pendaftaran"

Tabel 6.4 Pengujian Fungsional Aplikasi "Monitoring Pendaftaran"

Kode Test Case	Nama Test Case	Status
TC-08	Login	Terpenuhi
TC-09	Register	Terpenuhi
TC-10	Verifikasi data pendaftaran pasien baru	Terpenuhi
TC-11	Verifikasi data pendaftaran pasien lama	Terpenuhi

TC-12	Verifikasi data pendaftaran berobat	Terpenuhi

6.1.3 Pengujian Non-Fungsional Aplikasi

Tabel 6.5 Pengujian Non-Fungsional Aplikasi

ID	Kebutuhan Non-Fungsional	Status
NFR-01	Sistem dapat diakses selama 24 jam sehari, 7 hari seminggu	Terpenuhi
NFR-02	Aplikasi "Registrasi Pasien" dapat berjalan pada sistem operasi Android dengan versi API minimal 22.	Terpenuhi
NFR-03	Aplikasi "Monitoring Pendaftaran" dapat berjalan pada *browser*.	Terpenuhi
NFR-04	Sistem harus bisa menyimpan semua data yang dimasukkan pasien.	Terpenuhi
NFR-05	Sistem harus bisa menyimpan semua data yang dimasukkan petugas.	Terpenuhi

Halaman ini sengaja dikosongkan

BAB VII
KESIMPULAN DAN SARAN

Bab ini menjelaskan tentang kesimpulan yang didapat dari hasil penelitian. Kesimpulan ini diharapkan dapat menjawab rumusan masalah yang telah dijabarkan sebelumnya. Saran dapat digunakan untuk penelitian selanjutnya.

7.1 Kesimpulan

Dari pelaksanaan penelitian tugas akhir ini, didapatkan kesimpulan sebagai berikut.

1. Pengembangan aplikasi "Registrasi Pasien" menggunakan teknologi android dan bahasa pemrograman Java, sedangkan pengembangan API aplikasi dan aplikasi "Monitoring Pendaftaran" menggunakan teknologi PHP. Pengembangan aplikasi "Monitoring Pendaftaran" menggunakan pola MVC.
2. API aplikasi berfungsi untuk menghubungkan antara aplikasi "Registrasi Pasien" dengan database MySQL. API aplikasi dikembangkan dengan arsitektur REST. Keluaran dari API aplikasi berupa data dalam format JSON.
3. Isian spinner agama, pendidikan terakhir, dan pekerjaan pada registrasi pasien baru disesuaikan dengan isian-isian dari sistem yang sudah ada sebelumnya. Cara pengisiannya dengan memanggil fungsi-fungsi yang ada di API aplikasi untuk mengirimkan *query* ke database. Hasil dari *query* dicetak dalam bentuk JSON, lalu dibuatlah sebuah obyek baru pada aplikasi "Registrasi Pasien" dan disimpan ke *adapter* spinner.
4. Database SQLite pada aplikasi "Registrasi Pasien" digunakan untuk menyimpan data diri pasien dan data riwayat kunjungan. Data diri pasien tersimpan setelah

pasien melakukan *login*, sedangkan data riwayat kunjungan tersimpan setelah pasien membuka halaman pengecekan riwayat kunjungan.

5. Semua fungsi aplikasi sudah berjalan dengan baik.
6. Aplikasi "Registrasi Pasien" dapat berjalan dengan baik pada *device* dengan minimum API 21.
7. Aplikasi "Monitoring Pendaftaran" dan API aplikasi dapat berjalan dengan baik pada *webserver* Apache 1.7.4 dan dengan *database* MySQL 5.5.8.
8. Dengan adanya aplikasi, pasien bisa menjadi lebih mudah mendaftar rawat jalan di RSUD karena aplikasi memungkinkan pasien untuk melakukan pendaftaran dari rumah.

7.2 Saran

Tugas akhir ini memiliki potensi yang dapat dikembangkan ke depannya agar lebih baik. Untuk itu beberapa saran yang dapat dipertimbangkan adalah sebagai berikut.

1. Perlu adanya sebuah metode untuk mengganti *password* agar tingkat *security* bisa bertambah.
2. Perlu adanya sebuah metode untuk melakukan *reset password* sehingga pasien yang lupa *password* bisa melakukan *login.*
3. Perlu dibuat sebuah *user guide* untuk aplikasi "Registrasi Pasien" dan juga aplikasi "Monitoring Pendaftaran".

DAFTAR PUSTAKA

[1] Republik Indonesia. 2009. Undang-Undang No. 44 tahun 2009 tentang Rumah Sakit. Kementerian Kesehatan Republik Indonesia. Jakarta.

[2] "Desktop, Mobile & Tablet Operating System Market Share in Indonesia Jan - Dec 2016". [Daring]. Tersedia pada: http://gs.statcounter.com/os-market-share/desktop-mobile-tablet/indonesia/#monthly-201601-201612-bar [Diakses: 18-Apr-2017].

[3] "Desktop, Mobile & Tablet Operating System Market Share in Indonesia Jan 2014 - Dec 2016". [Daring]. Tersedia pada: http://gs.statcounter.com/os-market-share/desktop-mobile-tablet/indonesia/#monthly-201401-201612 [Diakses: 17-Apr-2017].

[4] "Tablet Operating System Market Share in Indonesia Jan - Dec 2016". [Daring]. Tersedia pada: http://gs.statcounter.com/os-market-share/tablet/indonesia/#monthly-201601-201612-bar [Diakses: 17-Apr-2017].

[5] "Mobile Operating System Market Share in Indonesia Jan - Dec 2016". [Daring]. Tersedia pada: http://gs.statcounter.com/os-market-share/mobile/indonesia/#monthly-201601-201612-bar [Diakses: 17-Apr-2017].

[6] "SDLC - Waterfall Model". [Daring]. Tersedia pada: https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm [Diakses: 29-Agu-2017]

[7] Nanang Aryanto, Fajrian Nur Adnan,M.CS. 2015. "Aplikasi mobile berbasis android untuk Administrasi Pemeriksaan Poliklinik Rawat Jalan di RSUD Kota Salatiga".

[8] Muhammad Wardianto, Qurotul Aini, M.T., Nur Aeni Hidayah, M. MSI. 2011. "Rancang Bangun Aplikasi Pendaftaran Online Jasa Pengobatan Berbasis Multimedia Pada Klinik Utama Siti Aksar Depok".

[9] Ongky Anjar Yamanta. 2013. “Rancang Bangun Aplikasi Administrasi Rawat Jalan pada Klinik Geo Medika”.
[10] “PHP MySQL REST API for Android”. [Daring]. Tersedia pada: http://phppot.com/php/php-mysql-rest-api-for-android/ [Diakses: 13-Okt-2017]
[11] “Transmitting Network Data Using Volley”. [Daring]. Tersedia pada: https://developer.android.com/training/volley/index.html [Diakses: 29-Sep-2017]
[12] “google-gson”. [Daring] Tersedia pada https://github.com/google/gson [Diakses: 8-Mar-2018]
[13] “About SQLite”. [Daring] Tersedia pada https://www.sqlite.org/about.html [Diakses: 6-Jul-2018]
[14] “MVC: Model, View, Controller”. [Daring] Tersedia pada https://www.codecademy.com/articles/mvc [Diakses: 6-Jul-2018]

BIODATA PENULIS

Penulis bernama lengkap Arif Pradana Heryantara, lahir di Pasuruan, Jawa Timur pada tanggal 14 Oktober 1993. Merupakan anak pertama dari tiga bersaudara. Penulis menempuh pendidikan formal di SD Negeri 1 Kebonsari Pasuruan, SMP Negeri 2 Pasuruan, SMA Negeri 1 Pasuruan. Selama masa sekolah menengah atas, penulis aktif sebagai atlet catur Kota Pasuruan pada tingkat junior. Penulis juga pernah mewakili Kota Pasuruan dalam ajang olimpiade sains bidang komputer di tingkat Jawa Timur.

Pada tahun 2011, penulis melanjutkan studi ke jenjang yang lebih tinggi di Institut Teknologi Sepuluh Nopember sebagai mahasiswa Departemen Sistem Informasi, Fakultas Teknologi Informasi dan Komunikasi. Selama masa perkuliahan, penulis aktif di Kajian Islam Sistem Informasi di tingkat jurusan dan UKM Catur ITS. Penulis juga aktif di kepanitiaan sebagai staf perizinan dan bendahara. Penulis juga aktif sebagai atlet catur Kota Pasuruan pada tingkat junior dan atlet catur ITS.
Apabila terdapat pertanyaan mengenai Tugas Akhir ini, penulis dapat dihubungi melalui e-mail arif.pradana.heryantara@gmail.com.