PENGEMBANGAN SISTEM INFORMASI PELAPORAN PENDATAAN KEPENDUDUKAN KABUPATEN MALANG BERBASIS WEBSITE DENGAN METODE *RATIONAL UNIFIED PROCESS* (RUP)

SKRIPSI

Untuk memenuhi sebagai persyaratan
Memperoleh gelar Sarjana Komputer

Disusun oleh:
Hana Nur Hanifah
145150401111041

PROGRAM STUDI SISTEM INFORMASI
JURUSAN SISTEM INFORMASI
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2019

PENGESAHAN

PENGEMBANGAN SISTEM INFORMASI PELAPORAN PENDATAAN KEPENDUDUKAN KABUPATEN MALANG BERBASIS WEBSITE DENGAN METODE *RATIONAL UNIFIED PROCESS* (RUP)

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Hana Nur Hanifah
145150401111041

Skripsi ini telah diuji dan dinyatakan lulus pada
2 Januari 2019
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Yusi Tyroni Mursityo, S.Kom., M.AB	Moch. Chandra Saputra, S.Kom, M.T, M.Eng.
NIP: 19800228 200604 1 001	NIK: 201609 860106 1 001

Mengetahui
Ketua Jurusan Sistem Informasi

Dr. Eng. Herman Tolle, S.T., M.T.
NIP: 19740823 200012 1 001

PERNYATAAN ORISINALITAS

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

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

Malang, 02 Januari 2019

Hana Nur Hanifah
NIM: 145150401111041

KATA PENGANTAR

Puji syukur penulis panjatkan ke hadirat Allah SWT yang telah melimpahkan rahmt dan karunia-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul "PENGEMBANGAN SISTEM INFORMASI PELAPORAN PENDATAAN KEPENDUDUKAN KEBUPATEN MALANG BERBASIS WEBSITE DENGAN METODE *RATIONAL UNIFIED PROCESS* (RUP)".

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

1. Bapak Yusi Tyroni Mursityo, S.Kom., M.AB selaku dosen pembimbing 1 dan M.Chandra Saputra, S.Kom., M.Eng selaku dosen pembimbing 2, yang telah membimbing, memberikan saran dan motivasi kepada penulis selama penyusunan laporan skripsi.
2. Orang tua dari penulis yang selama ini telah memberikan kasih sayang, doa, serta berbagai dukungan moral maupun materi kepada penulis. Serta saudara – saudara penulis yang penulis sayangi.
3. Seluruh dosen dan seluruh civitas akademika Fakultas Ilmu Komputer yang telah banyak memberi bantuan dan dukungan selama penulis menempuh studi di Universitas Brawijaya.
4. Seluruh staf Dinas Kependudukan dan Pencatatan Sipil Kabupaten Malang yang telah membantu dalam penyusunan laporan skripsi.
5. Seluruh teman-teman yang tidak dapat penulis sebutkan satu per satu yang telah memberikan semangat dan dukungan agar skripsi ini dapat terselesaikan.

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

Malang, 02 Januari 2019

Penulis
Hanahanifah33@gmail.com

ABSTRAK

Hana Nur Hanifah, Pengembangan Sistem Informasi Pelaporan Pendataan Kependudukan Kabupaten Malang Berbasis Website Dengan Metode *Rational Unified Process* (RUP)

Pembimbing: Yusi Tyroni Mursityo, S.Kom., M.AB dan Moch. Chandra Saputra, S.Kom, M.T, M.Eng.

Dispenduk Capil Kabupaten Malang merupakan suatu instansi pemerintah yang mengatur masalah administrasi kependudukan yang ada di kabupaten Malang. Dalam pelaksanaannya Dispenduk Capil Kabupaten Malang memiliki beberapa permasalahan terkait pengajuan administrasi kependudukan dan pelaporan data kependudukan yang dilakukan oleh desa/kelurahan. Pengajuan administrasi kependudukan memiliki beberapa permasalahan seperti proses pengajuan administrasi kependudukan membutuhkan waktu yang relatif cukup lama karena prosesnya yang cukup rumit dikarenakan masyarakat harus melalui serangkaian proses dimulai dari tingkat desa/kelurahan, kecamatan dan berakhir di dispenduk capil, banyaknya antrian yang terjadi setiap harinya, dan luasnya kabupaten Malang yang memicu jauhnya jarak tempuh dan mahalnya transportasi yang harus masyarakat tanggung. Proses pelaporan data kependudukan memiliki beberapa permasalahan yaitu tidak adanya keseragaman format laporan sehingga menyulitkan dalam mengelola laporan serta seringnya keterlambatan pengiriman laporan. Berdasarkan berbagai masalah tersebut maka solusi yang dapat diterapkan adalah dengan membuat sistem informasi pelaporan pendataan kependudukan. Sistem tersebut dikembangkan menggunakan metode *Relational Unified Process (RUP)* dengan menjalankan fase *inception, elaboration, construction,* dan *transition.* Proses pengembangan sistem dimulai dengan melakukan analisis persyaratan, perancangan sistem dan impelementasi sistem. Selanjutnya hasil dari implementasi sistem diuji untuk mengetahui apakah sistem berjalan sesuai dengan persyaratan yang telah teridentifikasi. Pengujian yang dilakukan meliputi pengujian validasi dengan menjalankan 6 kasus uji yang ada pada sistem dan semua kasus uji menunjukkan hasil yang valid, pengujian kompabilitas menghasilkan kesimpulan sistem dapat memenuhi persyaratan non fungsional pengguna yang telah diidentifikasi sebelumnya, meskipun terdapat beberapa *issues*. Pengujian usabilitas menghasilkan skor 80,35 yang berarti berada pada grade B. Sehingga sistem dapat dikatakan baik dan dapat digunakan dengan mudah oleh pengguna.

Kata kunci: sistem informasi, *administrasi kependudukan, laporan data kependudukan, RUP*

ABSTRACT

Hana Nur Hanifah, Development of Information System for Reporting Population Data Collection In Kabupaten Malang Based on Website With Rational Unified Process (RUP) Method

Mentor: Yusi Tyroni Mursityo, S.Kom., M.AB and Moch. Chandra Saputra, S.Kom, M.T, M.Eng.

Dinas Kependudukan dan Pencatatan Sipil of Malang is a government agency that functions for population problems in Malang. In the implementation Dispenduk Capil of Malang has several problems related to the submission of population administration and reporting of population data conducted by the village/kelurahan. Submission of population administration has several problems such as the process of submitting a population administration requiring a relatively long time because the process is quite complicated. This is because the community must go through a series of processes starting from the village/kelurahan, kecamatan level and ending up in the dispenduk capil, the number of queues that occur every day, and the extent of the malang district area can be make long distance to came Dispenduk Capil of malang, so the community must have pay the expensive transportation. The process of reporting population data has several problems, i.e. the absence of uniformity of report formats, which makes it difficult to manage reports and the frequent delays in sending reports. Based on these various problems, the solution that can be applied is to create an information system for reporting population data collection. The system was developed using the Rational Unified Process (RUP) method with the initial phase of inception, elaboration, construction, and transition. The system development process begins with performing requirements analysis, system design and system implementation.Furthermore, the results of the system implementation to find out whether the system runs in accordance with the requirements that have been identified. Tests carried out include validation by running 6 cases that are on the system and all cases that are available, which allow for use, which have several problems. Usability testing produces a score of 80,35 which means it is on grade B. plus the system can be used easily and can be used easily by the user.

Keywords: *Information system, population administration, report on population data, RUP*

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Semakin berkembangnya teknologi, informasi menjadi kebutuhan utama yang diperlukan oleh semua pihak. Kemudahan mendapatkan informasi menjadi hal yang penting dalam suatu perusahaan atau instansi pemerintah. Dengan adanya teknologi informasi dapat memudahkan perusahaan atau instansi pemerintah dalam menemukan suatu informasi yang relevan, akurat dan tepat waktu. Hal ini dapat dicapai dengan adanya suatu sistem informasi yang baik pada organisasi tersebut. Sistem informasi memberikan banyak keuntungan, dari tugas yang *simple* seperti proses transaksi pada level operasional sampai ke tugas yang sulit seperti membuat keputusan penting dan kompetitif pada tingkat strategis organisasi. Oleh karena itu sebuah perusahaan atau instansi pemerintah harus memiliki sistem informasi yang baik agar dapat meningkatkan kinerjanya sehingga menjadi lebih efektif dan efisien.

Instansi pemerintahan yang mengatur masalah administrasi kependudukan adalah Dinas Kependudukan dan Pencatatan Sipil. Administrasi kependudukan adalah rangkaian kegiatan penataan dan penerbitan dokumen dan data kependudukan melalui pendaftaran penduduk, pencatatan sipil, pengelolaan informasi administrasi kependudukan serta pendayagunaan hasilnya untuk pelayanan publik dan pembangunan sektor lain (Pasal 1 UU No 24 Tahun 2013). Dalam pengurusan administrasi kependudukan terdapat syarat tertentu yang harus dipenuhi oleh masyarakat atau orang yang ingin mengajukan maupun mengurus administrasi kependudukan. Selama ini dalam proses pengurusan administrasi kependudukan masyarakat relatif membutuhkan waktu yang cukup lama karena prosesnya yang cukup rumit. Misalnya saja untuk pengurusan surat pindah. Seseorang yang mengajukan surat pindah, terlebih dahulu harus mengurus surat pengantar di desa/kelurahan. Akan tetapi sebelum mengurus surat pengantar di desa/kelurahan harus meminta surat pengantar terlebih dahulu dari rt. Setelah mendapatkan surat pengantar dari desa/kelurahan pemohon harus ke kecamatan sebelum akhirnya ke Dispenduk Capil untuk menyerahkan berkas permohonan surat pindah dan dokumen pendukungnya Selanjutnya Dispenduk Capil memproses pengajuan tersebut sesuai dengan SOP yang berlaku. Selain faktor proses yang dirasa rumit ada faktor lain yang juga menjadi kendala bagi masyarakat. Faktor lain itu adalah proses antri yang sangat panjang di Dispenduk Capil. Setiap harinya di Dispenduk Capil ada sekitar 700 antrian untuk KTP dan 500 antrian untuk KK dan akta. Luasnya wilayah kabupaten malang juga menyumbang permasalahan bagi masyarakat. Berdasarkan data Dispenduk Capil tahun 2016, Kabupaten malang merupakan kabupaten terluas kedua di Jawa Timur dengan luas wilayah ± 3.534.86 km^2 dan dengan jumlah penduduk sebanyak 2.706.226 jiwa. Jarak tempuh yang jauh dan mahalnya biaya transportasi bagi masyarakat Kabupaten Malang untuk mendapatkan pelayanan merupakan salah satu permasalahan yang membutuhkan solusi. Faktor waktu pengurusan yang hanya bisa dilakukan di jam

dan hari kerja juga membuat masyarakat enggan untuk mengurus administrasi kependudukan. Semua kendala – kendala tersebut nantinya akan berdampak pada banyaknya masyarakat yang tidak tertib administrasi kependudukan dan banyaknya masyarakat yang lebih memilih untuk menggunakan jasa calo dan mengeluarkan uang yang cukup banyak untuk mengurus administrasi kependudukan yang sebenarnya tidak dipungut biaya atau gratis.

Selain masalah pengurusan administrasi kependudukan ada permasalahan lain yang dihadapi oleh Dispenduk Capil, yaitu terkait dengan laporan data kependudukan yang dilaporkan oleh desa/kelurahan melalui camat setiap bulannya. Seperti yang disebutkan pada Pasal 1 UU No 24 Tahun 2013 salah satu tugas Dispenduk Capil adalah pengelolaan informasi administrasi kependudukan serta pendayagunaan hasilnya untuk pelayanan publik dan pembangunan sektor lain. untuk menunjang tugas tersebut hal yang dilakukan Dispenduk Capil adalah dengan mengolah laporan yang diserahkan oleh desa/kelurahan sesuai dengan pasal 58 ayat 4 undang – undang nomor 24 tahun 2013 dimana disebutkan bahwa data kependudukan dimanfaatkan untuk pelayanan publik, perencanaan pembangunan, alokasi anggaran, pembangunan demokrasi, dan penegakan hukum dan pencegahan kriminal. Hasil pengolahan data tersebut dapat dipergunakan untuk berbagai kepentingan Dispenduk Capil maupun instansi terkait. Bagi instansi terkait misalnya, melalui rekapitulasi jumlah data kelahiran yang tinggi dapat digunakan oleh dinas kesehatan untuk melakukan perencanaan anggaran terkait penambahan jumlah fasilitas Rumah Sakit Ibu dan Anak.

Proses pendataan kependudukan yang berjalan selama ini membutuhkan waktu yang lama serta sulit melakukan perekapan dan analisa laporan dikarenakan data yang telah diberikan oleh desa/kelurahan tidak dapat langsung direkap dan di analisa melainkan petugas harus terlebih dahulu menginputkan satu persatu data kependudukan yang telah dilaporkan oleh camat ke dalam *Microsoft Excel*. Faktor lain yang menyebabkan proses menjadi lama adalah wilayah Kabupaten Malang yang luas.

Berdasarkan berbagai permasalahan yang sedang dialami oleh Dispenduk Capil Kabupaten Malang saat ini, maka perlu adanya pengembangan sistem informasi yang mampu mempermudah proses pengajuan administrasi kependudukan sehingga bisa menekan besarnya biaya, tenaga dan waktu yang harus dikeluarkan oleh masyarakat dalam mendapatkan pelayanan administrasi kependudukan serta perlu adanya sistem informasi yang mampu mempermudah mengelolah laporan, pengiriman serta penerimaan laporan dan mempermudah dalam melakukan perekapan laporan sehingga proses pelaporan menjadi lebih efektif dan efisien. Pengembangan sistem yang dilakukan dengan menggunakan pendekatan RUP, karena RUP dapat mendifinisikan kebutuhan dengan baik, dikembangkan dengan mengumpulkan berbagai *best practice*, dan menggunakan konsep *object oriented* yang berfokus pada pengembangan menggunakan UML (*Unified Modeling Language*). RUP memungkinkan adanya analisis resiko lebih

awal karena dikembangkan secara *iterative* dan *increment*. Pengembangan dilakukan sedikit demi sedikit atau bertahap sehingga dapat mengakomodasikan adanya perubahan kebutuhan Dispenduk Capil seiring proses pengembangan sistem informasi. Oleh karena itu , pengembangan sistem informasi pelaporan dianggap perlu yang akan dilakukan pada skripsi ini dengan judul "Pengembangan sistem informasi pelaporan pendataan kependudukan kabupaten malang berbasis website dengan metode *Rational Unified Process* (RUP)". Hasil dari penelitian ini diharapkan dapat membantu Dispenduk Capil Kabupaten Malang dalam menghadapi permasalahan yang sedang dihadapi sehingga dapat meningkatkan kinerja menjadi lebih efektif dan efisien.

1.2 Rumusan Masalah

Berikut adalah rumusan masalah yang ada pada penelitian ini :

1. Bagaimana proses bisnis pelaporan dan pendataan yang telah berjalan dan peningkatan terhadap proses bisnis tersebut pada Dinas Kependudukan dan Pencatatan Sipil Kabupaten Malang pada fase *inception* metode *Rational Unified Process*?
2. Bagaimana hasil analisa kebutuhan dan perancangan arsitektur sistem informasi pelaporan pendataan kependudukan pada fase *elaboration* metode *Rational Unified Process*?
3. Bagaimana implementasi rancangan arsitektur pada sistem informasi pelaporan pendataan kependudukan dan hasil dari pengujian sistem menggunakan *validation testing,* dan *compatibility testing* pada fase *contruction* metode *Rational Unified Process*?
4. Bagaimana hasil pengujian sistem informasi pelaporan pendataan kependudukan menggunakan *usability testing* pada fase *transition*?

1.3 Tujuan Penelitian

Tujuan yang ingin dicapai dalam penelitian ini adalah :

1. Menganalisa proses bisnis pelaporan dan pendataan yang telah berjalan di Dinas Kependudukan Dan Pencatatan Sipil Kabupaten Malang dan peningkatan terhadap proses bisnis tersebut.
2. Menganalisa kebutuhan sistem serta merancang sistem informasi pelaporan pendataan kependudukan kabupaten malang berbasis website dengan metode RUP (*Rational Unified Process*)
3. Membangun sistem informasi pelaporan pencatatan kependudukan kabupaten malang berbasis website dengan metode RUP yang sesuai dengan analisa kebutuhan dan perancangan
4. Menguji hasil sistem informasi pelaporan pencatatan kependudukan kabupaten malang berbasis website dengan metode RUP.

1.4 Manfaat Penelitian

Dengan tercapainya tujuan penelitian diatas, diharapkan dapat bermanfaat bagi semua pihak, manfaat dari penelitian ini adalah :

1. Bagi peneliti, hasil penelitian ini dapat memberikan pengetahuan serta pengalaman dalam merancang, mengimplementasi, dan menguji sistem informasi berbasis website.
2. Bagi Dinas Kependudukan Dan Pencatatan Sipil, hasil penelitian ini nantinya diharapkan dapat mempermudah dalam proses mengelolah laporan, pengiriman serta penerimaan laporan, mempermudah proses verifikasi laporan dan mempermudah dalam melakukan perekapan laporan secara otomatis.
3. Bagi pemustaka, hasil penelitian ini dapat dijadikan rujukan bagi mereka yang ingin melaksanakan penelitian dengan mengambil tema yang sama.

1.5 Batasan Masalah

1. Penelitian dilakukan pada Dinas Kependudukan Dan Penacatan Sipil Kabupaten Malang.
2. Metode yang digunakan dalam penelitian ini adalah *Rational Unified Process* (RUP).
3. Sistem informasi pelaporan pendataan kependudukan kabupaten malang adalah berbasis website.
4. Sistem informasi ini hanya memuat pencatatan data kependudukan untuk pengajuan kartu keluarga, akta kelahiran, akta kematian , surat pindah datang dan surat pindah keluar.
5. Pengujian sistem dilakukan dengan menggunakan *validation testing*, *compatibility testing* dan *usability testing.*

1.6 Sistematikan Pembahasan

Sistematika Pembahasan berguna untuk memberikan gambaran dan kerangka yang jelas mengenai pokok bahasan setiap bab dalam penelitian ini. Berikut adalah gambaran sistematika penulisan pada masing – masing bab yang ada pada penelitian ini :

BAB 1 : PENDAHULUAN

Bab ini berisikan latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penulisan skripsi.

BAB 2 : TINJAUAN PUSTAKA

Bab dua menjelaskan tentang tinjauan pustaka dan landasan teori yang berhubungan sistem informasi pelaporan pendataan penduduk kabupaten malang berbasis website dengan menggunakan metode *Rational Unified Process* (RUP).

BAB 3 : METEDOLOGI PENELITIAN

Pada bab ini akan dibahas tentang metode penelitian yang digunakan dalam menyelesaikan penelitian ini.

BAB 4 : ANALISA PERSYARATAN

Bab ini berisi tentang uraian proses bisnis yang telah berjalan pada Dispenduk Capil Kabupaten Malang serta bagaimana usulan proses bisnis baru. Pada bab ini juga berisi uraian terkait analisa kebutuhan sistem informasi pelaporan pendataan kependudukan kabupaten malang untuk mendukung kegiatan operasinal yang terdiri dari kebutuhan fungsional dan non fungsional sistem.

BAB 5 : PERANCANGAN

Bab ini menjelaskan tentang perancangan sistem informasi pelaporan pendataan kependudukan kabupaten malang berbasis website dengan menggunakan metode *Rational Unified Process* (RUP).

BAB 6 : IMPLEMENTASI

Bab ini berisi implementasi dari sistem informasi pelaporan pendataan kependudukan kabupaten malang berbasis website dengan menggunakan metode *Rational Unified Process* (RUP).

BAB 7 : PENGUJIAN

Bab ini menguraikan tentang pengujian yang dilakukan terhadap sistem informasi pelaporan pendataan kependudukan kabupaten malang berbasis website dengan menggunakan metode *Rational Unified Process* (RUP).

BAB 8 : PENUTUP

Pada bab ini akan dibahas tentang kesimpulan yang dihasilkan serta saran yang akan diberikan berdasarkan hasil yang telah dicapai sehingga dapat digunakan sebagai bahan pertimbangan bagi pihak - pihak yang berkepentingan serta kemungkinan perkembangan untuk penelitian selanjutnya.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

Pada penelitian yang berjudul "*Moving From waterfall to iterative Development An Empirical Evaluation of Advantages, Disadvantages and Risk Of RUP*" yang dilakukan oleh Jorge A Osorio, Michel R.V. Chaudron dan Werner Heijstek menjelaskan tentang manfaat menggunakan proses pengembangan *iterative* versus pengembangan *waterfall*. Penelitian dilakukan di sebuah divisi dari departemen TI perusahaan multinasional dengan menggunakan pendekatan studi kasus eksploratif. Pengumpulan data didasarkan pada wawancara semi-terstruktur dengan personel yang terlibat dengan implementasi RUP dalam suatu organisasi industri, meliputi orang – orang yang bekerja pada proyek – proyek TI yang menggunakan RUP dan orang – orang dari domain bisnis yang bertindak sebagai klien internal untuk proyek – proyek tersebut. Penelitian ini didasarkan pada pentingnya mengetahui kelebihan , kekurangan dan resiko dari RUP dikarenakan pengembangan *iterative* dianggap memiliki pengaruh positif terhadap proses dan kualitas produk. Namun, literatur yang melaporkan pengalaman menggunakan RUP di organisasi nyata masih langkah. Hasil dari penelitian ini menemukan beberapa kategori faktor dimana penggunaan pengembangan *iterative* lebih baik dari pada pengembangan *waterfall*. Namun ada beberapa biaya dan batasan yang jelas yang harus dipertimbangkan secara eksplisit oleh perusahaan yang berencana menggunakan proses pengembangan *iterative*.

Penelitian yang berjudul "*System modeling of vehicle management based on RUP and UML*" yang dilakukan oleh Lan Jin dan Xiujuan Liang membahas tentang pengembangan sistem informasi manajemen kendaraan dengan menggabungkan RUP dan UML. RUP menyediakan kriteria yang diperlukan, *template* dan paduan alat dalam aspek sebagai *use case driven*, pengembangan iterasi, pemodelan visual , arsitektur-sentris, dan sebagainya. UML adalah bahasa pemodelan yang sangat baik, dan dengan menggunakan UML dapat membantu pengembang untuk memiliki pemahaman yang jelas tentang keseluruhan sistem. Kombinasi RUP dan UML telah diakui sebagai kombinasi pengembangan perangkat lunak yang sangat efisien dalam industri komputer. Penelitian ini dilakukan dengan mengikuti fase siklus hidup dari RUP, terutama pada tiga alur proses inti yaitu pemodelan bisnis, analisis persyaratan dan desain. Pada waktu yang sama dilakukan pemodelan sistem dengan menggunakan UML diantaranya *use case diagram, class diagram* dan *sequence diagram*. Kesimpulan dari penelitian ini menunjukkan bahwa menggabungkan RUP dan UML untuk menggembangkan perangkat lunak dapat mengurangi resiko produk, menstandarisasi proses manajemen dan pengembangan proses, meningkatkan kualitas, efisien dan pemeliharaan pengembangan perangkat lunak.

Penelitian lainnya yang berjudul "Implementasi Metode *Rational Unified Process* Dalam Pengembangan Sistem Administrasi Kependudukan" yang

dilakukan oleh Aida Fitria dan Henny Widowati dari jurusan teknik informatika, fakultas teknologi industri, universitas Gunadarma mengembangkan sebuah sistem administrasi kependudukan dengan menggunakan metode RUP. Penelitian ini dilakukan karena adanya permasalahan terkait pelayanan administrasi kependudukan yang masih dilakukan secara konvensional khususnya di tingkat RT atau RW. Misalnya ketika membuat surat keterangan masih menggunakan mengunakan cara konvensional yaitu menulisakan pada selembar kertas yang ditulis dengan sebuah alat tulis, kemudian sebagai laporan ketua RT harus mencatat laporan tersebut dalam buku besar. Hal tersebut membutuhkan waktu proses yang relative lama. Kesalahan lain yang yang sering terjadi adalah kesalahan penulisan dalam pendataan, perekapan data, ataupun data yang hilang atau rusak. Tujuan dari pembuatan sisitem ini adalah untuk membantu dan mempermudah ketua RT, Ketua RW dan kelurahan dalam melakukan pendataan dan pengolahan data penduduk setempat. Sistem yang dikembangkan memiliki beberapa fitur antara lain fitur pencarian penduduk, rekap data penduduk dan fitur *backup restore*. Fitur pencarian penduduk memungkinkan pengguna baik itu sebagai Ketua RT, Ketua RW, maupun Kelurahan untuk mencari data penduduk berdasarkan NIK, nama, nomor kartu keluarga. Fitur rekap data memungkinkan pengguna mencetak daftar penduduk sesuai RT, RW, ataupun Kelurahan. Selain itu dengan adanya fitur *backup* dan *restore* pengguna dapat menyimpan data terbaru atau mengembalikan data yang sebelumnya telah disimpan.

Kajian penelitian selanjutnya adalah penelitian yang dilakukan oleh Rico Andrian, Dwi Sakethi dan Muhammad Chairuddin dengan judul "Pengembangan Sistem Informasi Penelitian dan Pengabdian Dosen Jurusan Ilmu Komputer Menggunakan Metode *Rational Unified Process* (RUP)". Penelitian ini dilakukan berdasarkan permasalahan terkait pengolahan data kegiatan dosen pada Jurusan Ilmu Komputer yang masih menggunakan *Spreadsheet*. Penggunaan *Spreadsheet* sebenarnya sudah mampu mengolah data penelitian dan pengabdian dosen dengan baik, akan tetapi pada penerapannya dosen belum mampu menggunakannya dengan maksimal karena ada beberapa kendala yang dihadapi oleh dosen yaitu dalam penulisan data yang sama misal nama dosen, sumber dana, jenis dana, jenis publikasi, tingkat publikasi, dan akreditasi harus dilakukan berulang – ulang. Tujuan dari pengembangan sistem informasi ini adalah untuk mengatasi permasalahan tersebut. Pengembangan sistem informasi pada penelitian ini menggunakan metode *Rational Unified Process* dengan melakukan enam tahapan yaitu *business modeling, requirements, analysis and design, implementation, test* dan *deployment*. Kesimpulan yang diperoleh dari penelitian ini adalah pengembangan sistem informasi dengan menggunakan metode RUP dapat memberikan keleluasaan bagi pengembang dalam mengembangkan sistem atau perangkat lunak.

Tabel 2.1 Tabel Tinjauan Pustaka

No	Nama Penulis, Nama Jurnal, Tahun	Tujuan Penelitian	Metode Penelitian/Pengembangan	Hasil dan Kesimpulan
1.	"*Moving From Waterfall To Iterative Development An Empirical Evaluation Of Advantages, Disadvantages And Risk Of RUP*" Jorge A Osorio, Michel R.V. Chaudron dan Werner Heijstek (2011), *EUROMICRO Conference on Software Engineering and Advanced Applications*	Untuk mengetahui kelebihan, kekurangan dan resiko dari RUP dikarenakan pengembangan *iterative* dianggap memiliki pengaruh positif terhadap proses dan kualitas produk.	Penelitian ini menggunakan pendekatan studi kasus eksploratif. Pengumpulan data didasarkan pada wawancara semi-terstruktur dengan personel yang terlibat dengan implementasi RUP dalam suatu organisasi industri, meliputi orang – orang yang bekerja pada proyek – proyek TI yang menggunakan RUP dan orang – orang dari domain bisnis yang bertindak sebagai klien internal untuk proyek – proyek tersebut.	Hasil dari penelitian ini menemukan beberapa kategori faktor dimana penggunaan pengembangan *iterative* lebih baik dari pada pengembangan *waterfall*. Namun ada beberapa biaya dan batasan yang jelas yang harus dipertimbangkan secara eksplisit oleh perusahaan yang berencana menggunakan proses pengembangan *iterative*.
2.	"*System modeling of vehicle management based on RUP and UML*" Lan Jin dan Xiujuang Liang (2012), *Fifth International Symposium on Computational Intelligence and Design*	Membuat sistem informasi manajemen kendaraan dengan menggabungkan metode RUP dan UML.	Penelitian ini dilakukan dengan mengikuti fase siklus hidup dari RUP, terutama pada tiga alur proses inti yaitu pemodelan bisnis, analisis persyaratan dan desain. Pada waktu yang sama dilakukan pemodelan sistem dengan menggunakan UML diantaranya *use case diagram, class diagram*	Hasil dari penelitian ini adalah sebuah sistem informasi manajemen kendaraan dan kesimpulan dari penelitian ini menunjukkan bahwa menggabungkan RUP dan UML untuk menggembangkan perangkat lunak dapat mengurangi resiko produk, menstandarisasi proses

No	Nama Penulis, Nama Jurnal, Tahun	Tujuan Penelitian	Metode Penelitian/Pengembangan	Hasil dan Kesimpulan
			dan *sequence diagram*.	manajemen dan pengembangan proses, meningkatkan kualitas, efisien dan pemeliharaan pengembangan perangkat lunak.
3.	"Implementasi Metode *Rational Unified Process* Dalam Pengembangan Sistem Administrasi Kependudukan" Aida fitria dan Henny Widowati (2017), Jurnal Teknologi Rekayasa Volume 22 No.1	Membuat sistem informasi administrasi kependudukan untuk membantu dan mempermudah ketua RT, ketua RW dan kelurahan dalam melakukan pendataan dan pengolahan data penduduk setempat	Pengembangan sistem informasi administrasi kependudukan ini dilakukan dengan menggunakan metode *Rational Unified Process* (RUP).	Hasil dari penelitian ini adalah sebuah sistem informasi administrasi kependudukan yang dapat membantu dan mempermudah ketua RT, ketua RW dam kelurahan dalam melakukan pendataan dan pengolahan data penduduk setempat.
4.	"Pengembangan Sistem Informasi Penelitian dan Pengabdian Dosen Jurusan Ilmu Komputer Menggubakan Metode *Rational Unified Process* (RUP)" Rico Adrian, Dwi Sakethi, dan Muhammad Choiruddin (2014), Jurnal Komputasi Vol.2 No.2	Membangun sistem informasi yang dapat membantu mengolah data penelitian dan pengabdian dosen jurusan ilmu komputer.	Pengembangan sistem informasi pada penelitian ini menggunakan metode *Rational Unified Process* (RUP) dengan melakukan enam tahapan yaitu *business modeling, requirements, analysis and design, implementation, test* dan *deployment*.	Hasil dari penelitian ini adalah sebuah sistem informasi yang dapat membantu mengolah data penelitian dan pengabdian dosen pada jurusan ilmu komputer dan kesimpulan dari penelitian ini adalah pengembangan sistem informasi dengan menggunakan metode RUP dapat memberikan keleluasaan bagi pengembang

No	Nama Penulis, Nama Jurnal, Tahun	Tujuan Penelitian	Metode Penelitian/Pengembangan	Hasil dan Kesimpulan
				dalam mengembangkan sistem atau perangkat lunak.

Dari studi kepustakaan yang telah dilakukan pada keempat penelitian diatas terdapat beberapa kesamaan dengan penelitian yang akan dilakukan. Persamaan tersebut meliputi permasalahan yang diangkat yaitu terkait permasalahan kependudukan serta metode yang digunakan dalam pengembangan sistem yaitu *Rational Unified Process* (RUP). Berdasarkan hal tersebut maka beberapa penelitian diatas dirasa cocok untuk mendukung penelitian yang akan dilakukan.

2.2 Dinas Kependudukan dan Pencatatan Sipil Kabupaten Malang

Dinas Kependudukan dan Pencatatan Sipil merupakan unsur pelaksana Pemerintah Daerah di bidang Kependudukan dan Pencatatan Sipil yang dipimpin oleh Kepala Dinas dan berkedudukan di bawah dan bertanggung jawab kepada Bupati melalui Sekretaris Daerah.

Dinas Kependudukan Dan Pencatatan sipil melaksanakan tugas pokok penyusunan dan pelaksanaan kebijakan daerah di bidang Administrasi Kependudukan. Untuk melaksanakan tugas pokok dimaksud, Dinas Kependudukan Dan Pencatatan sipil mempunyai fungsi :

1. Pengumpulan, pengelolaan dan pengendalian data yang berbentuk database serta analisis data untuk penyusunan program kegiatan.
2. Perencanaan strategis pada dinas kependudukan dan pencatatan sipil.
3. Perumusan kebijakan teknis bidang kependudukan dan pencatatan sipil.
4. Penyelenggaraan urusan pemerintahan dan urusan umum bidang kependudukan dan pencatatan sipil.
5. Pembinaan dan pelaksanaan tugas bidang kependudukan dan pencatatan sipil.
6. Pelaksanaan, pengawasan, pengendalian serta evaluasi dan pelaporan peyelenggaraan bidang kependudukan dan pencatatan sipil.
7. Pelaksanaan standart pelayanan minimal yang wajib dilaksanakan pada bidang kependudukan dan pencatatan sipil.
8. Penyelenggara kesekretariatan dinas kependudukan dan pencatatan sipil
9. Pelayanan pendaftaran penduduk, pencatatan sipil, pengelolaan informasi administrasi kependudukan dan penyerasian perkembangan kependudukan.
10. Pengkoordinasian integrasi dan sinkronisasi kegiatan bidang administrasi kependudukan dan penyerasian perkembangan kependudukan dilingkungan pemerintah daerah.
11. Pembinaan kepada masyarakat tentang kependudukan dan pencatatan sipil.
12. Pelaksanaan kerjasama dengan lembaga pemeritah dan lembaga lainnya.
13. Koordinasi dengan instansi terkait dalam hal kebijakan kependudukan, tertib administrasi kependudukan dan analisis dampak kependudukan.
14. Pelaksanaan sistem informasi administrasi kependudukan.
15. Pembangunan dan pengembangan jaringan komunikasi data kependudukan.

16. Perlindungan data pribadi penduduk dalam proses dan hasil pedaftaraan penduduk serta pencatatan sipil pada data base kependudukan.
17. Pembinaan dan pengembangan sumber daya manusia pengelola pendaftaran penduduk, pencatatan sipil, pengelolaan informasi administrasi kependudukan dan penyerasian perkembangan kependudukan .
18. Pengawasan dan pengendalian atas penyelenggaraan pendaftaran penduduk, pencatatan sipil pengelolaan informasi administrasi kependudukan dan penyerasian perkembangan kependudukan.

Dispenduk Capil dijalankan oleh 95 orang yang terdiri dari 27 PNS dan 68 non-PNS. Jumlah pelayanan yang dilakukan sekitar 3.500 pelayanan setiap harinya dengan berbagai jenis pelayanan diantaranya: pelayanan E-KTP, KK, Akta dan Surat Pindah serta komplain.

2.2.1 Visi dan Misi

Visi :

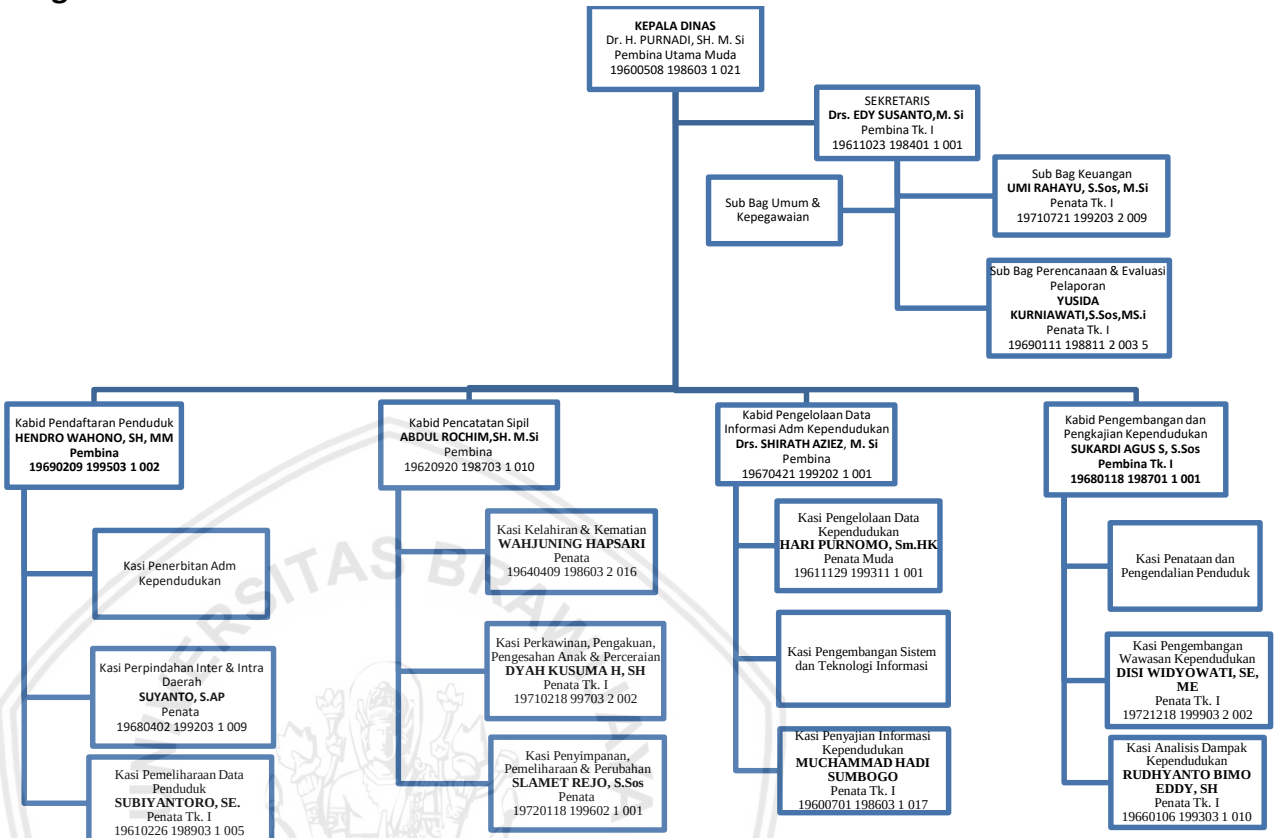
" Terwujudnya Pelayanan yang Profesional dalam Tertib Administrasi Kependudukan dan Pencatatan sipil tahun 2015 "

Misi :

1. Memberikan Pelayanan kepada masyarakat secara profesional dalam pengurusan administrasi kependudukan dan pencatatan sipil.
2. Melaksanakan sistem informasi administrasi kependudukan (SIAK) berbasis teknologi informasi.

Menyediakan data dan dokumen kependudukan secara akurat dan valid dalam penyelenggaraan kegiatan pemerintahan, pembangunan dan kemasyarakatan**.**

2.2.2 Struktur Organisasi



Gambar 2.1 Struktur Organisasi

Berdasarkan bagan struktur organisasi diatas diketahui bahwa Dinas Kependudukan dan Pencatatan Sipil Kabupaten Malang memiliki empat sub bidang, yaitu bidang pendaftaran penduduk, bidang pencatatan sipil, bidang pengolahan data informasi administrasi kependudukan dan bidang pengembangan dan pengkajian kependudukan. Pada penelitian ini berfokus pada sub bidang Pengelolaan Data Informasi Administrasi Kependudukan karena bidang Pengelolaan data Informasi Administrasi Kependudukan memiliki tugas untuk mengelola data informasi kependudukan sehingga dapat digunakan untuk kepentingan Dispenduk Capil sendiri dan untuk kepentingan dinas terkait.

Bidang Pengelolaan Data Informasi Administrasi Kependudukan mempunyai tugas :

1. Melaksanakan sebagian tugas Dinas Kependudukan dan Pencatatan Sipil dalam bidang pengelolaan data informasi administrasi kependudukan meliputi : fasilitasi, konsultasi, pengelolaan informasi administrasi penduduk, koordinasi pembangunan dan pengembangan jaringan komunikasi data penyediaan perangkat keras dan perlengkapannya serta jaringan komunikasi data sampai dengan tingkat Kecamatan atau Desa / Kelurahan sebagai tempat pelayanan dokumen penduduk pelaksanaan Sistim informasi Administrasi Kependudukan / SIAK , perekaman data hasil pendaftaran penduduk dan pencatatan sipil serta pemutakhiran data penduduk menggunakan SIAK, perlindungan data pribadi penduduk.
2. Melaksanakan tugas - tugas lain yang diberikan oleh Kepala Dinas sesuai dengan bidang tugasnya.

Bidang Pengelolaan Data Informasi Administrasi Kependudukan dipimpin oleh seorang Kepala Bidang yang berada dibawah dan bertanggung jawab kepada Kepala Dinas Kependudukan dan Pencatatan Sipil Kabupaten Malang. Bidang Pengelolaan Data Administrasi Kependudukan mempunyai fungsi :

1. Penyelenggaraan pelayanan pengelolaan data informasi administrasi Kependudukan.
2. Penyusunan program, penyelenggaraan dan evaluasi kegiatan pengelolaan data informasi administrasi kependudukan, pengelolaan dan pemeliharaan data penduduk.
3. Pelaksanaan penyiapan kegiatan pengelolaan data informasi administrasi kependudukan.
4. Koordinasi penyelenggaraan pengelolaan data informasi administrasi kependudukan.
5. Konsultasi pelaksanaan pengelolaan data informasi administrasi kependudukan.
6. Pengembangan dan pemeliharaan jaringan.
7. Pembangunan dan pemutakhiran database.
8. Penyajian dan desiminasi informasi.
9. Pemantauan, evaluasi, dan pelaporan penyeleggaraan pengelolaan data informasi administrasi kependudukan.

10. Pengawasan atas penyelenggaraan pengelolaan data informasi administrasi kependudukan.

Setiap harinya bidang Pengolahan Data Informasi Administrasi Kependudukan (PIAK) bertanggung jawab terhadap kondisi jaringan yang ada di Dispenduk Capil agar proses pelayanan tidak terhambat atau terganggu. Mereka juga melakukan *maintenance* terhadap jaringan dan database yang ada. Selain itu PIAK juga bertugas melakukan pemantauan dan pengolahan data informasi administrasi kependudukan.

Bidang Pengembangan dan Pengkajian Kependudukan mempunyai tugas :

1. Melaksanakan sebagian tugas Dinas Kependudukan dan Pencatatan Sipil dalam bidang pengembangan dan pengkajian kependudukan meliputi; penyerasian dan harmonis kebijakan kependudukan antar dan dengan lembaga pemerintah dan non pemerintah, pengendalian kuantitas / kualitas penduduk dan perlidungan penduduk serta pembangunan berwawasan kependudukan, penyelenggaraan kerjasama dengan organisasi kemasyarakatan dalam rangka tertib administrasi kependudukan penetapan indikator kependudukan, proyeksi penduduk dan dampak kependudukan serta kebijakan kependudukan terhadap khalayak sasaran, penilaian dan pelaporan kinerja pembangunan kependudukan secara periodik.
2. Melaksanakan tugas - tugas lain yang diberikan oleh Kepala Dinas sesuai dengan bidang dan tugasnya.

Bidang Pengembangan dan Pengkajian Kependudukan dipimpin oleh seorang Kepala Bidang yang berada dibawah dan bertanggung jawab kepada Kepala Dinas Kependudukan dan Pencatatan Sipil Kabupaten Malang. Bidang Pengembangan dan Pengkajian Kependudukan mempunyai fungsi :

1. Penyelenggaraan pengembangan dan pengkajian kependudukan.
2. Penyusunan program, penyelenggaraan dan evaluasi kegiatan pengembangan dan pengkajian kependudukan dan pengelolaannya.
3. Pelaksanaan penyiapan kegiatan pengembangan dan pengkajian kependudukan.
4. Koordinasi penyelenggaraan pengembangan dan pengkajian kependudukan.
5. Fisilitasi, sosialisasi, bimbingan teknis, dan konsultasi pelaksanaan pengembangan dan pengkajian kependudukan.
6. pemantauan, evaluasi dan pelaporan penyelenggaraan pengembangan dan pengkajian kependudukan.

2.3 Sistem Informasi

Menurut (Ladjamudin, 2005)sistem informasi dapat didefinisikan sebagai berikut :

a. Suatu sistem yang dibuat oleh manusia yang terdiri dari komponen – komponen dalam organisasi untuk mencapai satu tujuan yaitu menyajikan informasi.
b. Sekumpulan prosedur organisasi yang pada saat dilaksanakan akan memberikan informasi bagi pengambil keputusan dan / atau untuk mengendalikan organisasi
c. Suatu sistem didalam organisasi yang mempertemukan kebutuhan pengolahan transaksi – transaksi, mendukung operasi, bersifat manajerial, dan kegiatan strategis dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan – laporan yang diperlukan.

Sedangkan menurut (Murhada & Giap, 2011) sistem informasi adalah sistem yang mengumpulkan , menyimpan, mengolah, dan menyebarkan data dan informasi. Dari pengertian tersebut dapat disimpulkan bahwa sistem informasi adalah sistem yang menyediakan atau memberikan informasi bagi organisasi guna mendukung tercapainya tujuan suatu organisasi.

Sistem informasi memiliki tujuan untuk menghasilkan informasi yang berasal dari hasil pengolahan data menjadi bentuk yang berguna bagi pemakainya. Komponen - komponen yang terdapat dalam sistem informasi yaitu komponen input, komponen model, komponen output, komponen teknologi, komponen basis data dan komponen kontrol.

1. Komponen input merupakan data yang masuk ke dalam sistem informasi sebagai bahan dasar dalam pengolahan informasi.
2. Komponen model merupakan kombinasi dari prosedur, logika dan model matematika yang akan memanipulasi data input dan data yang tersimpan di basis data dengan cata yang sudah ditentukan untuk menghasilkan keluaran yang diinginkan.
3. Komponen output atau keluaran merupakan hasil dari sistem informasi yang merupakan informasi dan dokumentasi yang berguna bagi pemakai sistem .
4. Komponen teknologi merupakan alat dalam sistem informasi untuk menerima input, menjalankan model, menyompan, mengakses data, menghasilkan dan mengirimkan keluaran serta membantu pengendalian keseluruhan sistem.
5. Komponen basis data merupakan kumpulan data yang saling berkaitan dan berhubungan satu sama lain yang disimpan untuk keperluan penyediaan informasi lebih lanjut.
6. Komponen kontrol yang dperlukan untuk menjamin kualitas informasi yang dihasilkan oleh sistem informasi serta mencegah kerusakan dan sesalahan sistem informasi

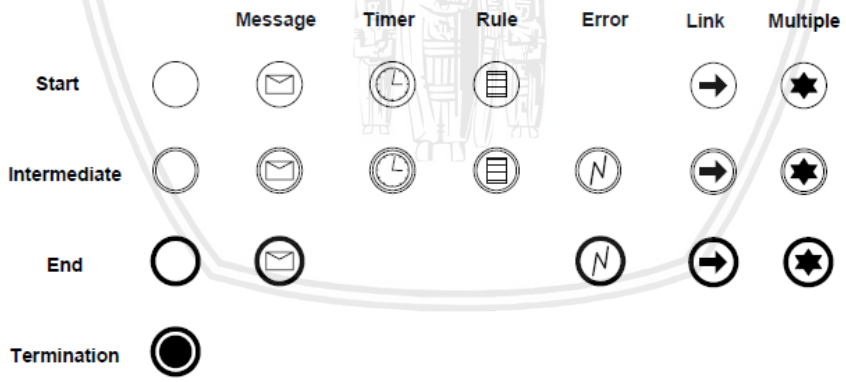
2.4 BPMN

BPMN merupakan standar notasi proses bisnis yang memiliki tujuan utama untuk memberikan notasi yang mudah dimengerti oleh semua pengguna bisnis, dari para analisis bisnis yang menciptakan konsep awal proses, kepada pengembang teknis yang bertanggung jawab untuk menerapkan teknologi yang akan melakukan proses tersebut, dan akhirnya kepada pebisnis yang akan mengelola dan memantau proses tersebut (Object Management Group, 2011).

2.4.1 *Flow Object*

1. Event

Event memainkan peran sentral dalam manajemen proses bisnis, karena *event* adalah perekat antara situasi dalam organisasi bisnis dan proses yang akan diberlakukan jika situasi ini terjadi. Dalam proses bisnis *event* terbagi menjadi tiga, yaitu *start event* atau *event* awal digunakan untuk memicu proses awal, *intermediate event* dapat menunda proses atau dapat terjadi selama proses, *End event* menandakan penghentian proses (weske, 2007). Notasi untuk masing – masing *event* dapat dilihat pada gambar 2.2. Untuk *start event* dinotasikan ke dalam simbol lingkaran dengan garis lingkaran tipis. *Intermediate event* dinotasikan ke dalam simbol lingkaran dengan dua garis lingkaran. Sedangkan untuk *end event* dinotasikan ke dalam simbol lingkaran dengan garis lingkaran yang tebal.



Gambar 2.2 Simbol - Simbol *Flow Object*
Sumber (Object Management Group, 2011)

Start events, dapat dipicu oleh :

1. *None* : tidak ada jenis pemicu spesifik yang diberikan. Ini digunakan saat subproses dimulai oleh proses induknya.
2. *User* : user atau pengguna secara manual memulai proses, membuat event awal dari proses bisnis.

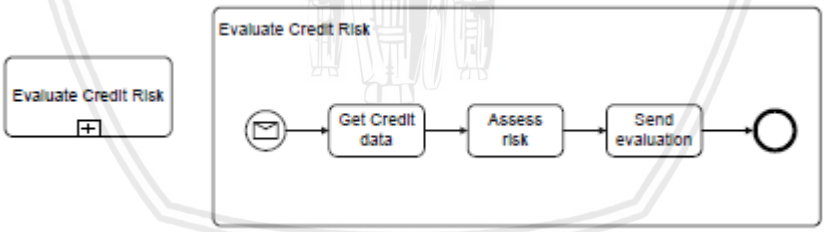
3. *Message* : pesan diterima oleh seorang *participant* dan tanda terima diwakili oleh *message event.*
4. *Timer* : tanggal tertentu atau siklus tertentu yang dapat diatur dana akan memicu dimuainya.
5. *Rule* : jenis *events* ini dipicu ketika aturan dievaluasi ke *true*.

Sedangkan *Intermediate events*, dapat dipicu oleh :

1. *None* : dapat digunakan untuk menandai perubahan keadaan dalam proses.
2. *Message* : langkah dalam proses tercapai ketika kemajuan tergantung pada pesan yang datang dari seorang *participant*. Ketika pesan tiba, proses dapat dilanjutkan.
3. *Timer* : *intermediate events* dipacu berdasarkan informasi penghitung waktu. Spesifikasi waktu bersifat *relative* atau absolut
4. *Error* : peristiwa kesalahan menengah menghasilkan pengecualian selama aliran normal proses.

2. *Activities*

Ativities merupakan unit kerja unsur utama proses bisnis ditunjukkan dengan persegi panjang dengan ujung-ujung bulat. Sebuah *activities* dapat berdiri sendiri atau gabungan. BPMN mendukung hirarki aktivitas yang bercabang sehingga setiap *activities* memuat baik kegiatan atomik (*task*) dan *sub prosess* yang dibedakan dengan tanda + pada bagian tengan bahwa dari bentuk tersebut (weske, 2007).



Gambar 2.3 Simbol *Activities*
Sumber (Object Management Group, 2011)

3. *Sequence flow* dan *Gateway*

Pada BPMN, aliran kontrol disebut juga dengan *sequence flow* yang ditunjukkan dengan panah solid antara *flow object*. Ada beberapa tipe *sequence flow* yaitu : *normal flow, exception flow, sequence flow* yang disebabkan oleh *link events* dan *adhoc flow*. *Normal flow* dari suatu proses bisnis mewakili perilaku yang diharapkan dan diinginkan dari proses tersebut. Dimulai dari awal hingga akhir. *Exception flow* dihasilkan dari *intermediate events* yang terhubung dengan notasi *activity*. Secara khusus, *intermediate events* hanya akan menghasilkan

exception flow jika *activity* yang dilampirkan masih aktif ketika *events* tersebut terjadi.

Gateway digunakan untuk mengontrol percabangan dan penggabungan *sequence flow.* Jadi, setiap *gateway* dapat bertindak sebagai *join node* atau *split node*. *Join node* memiliki setidaknya dua busur yang masuk dan tepat satu ujung yang keluar. *Split node* memiliki tepat satu busur masuk dan setidaknya dua sisi keluar. Berikut adalah notasi dari *gateway* yang ada pada BPMN. Data-based XOR digunakan untukmembuat alur alternatif dari proses bisnis. *Gateway* AND digunakan untuk membuat alur paralel dan menyinkronkan alur paralel yang telah dibuat. Event-based XOR digunakan untuk membuat alur alternatif pada proses bisnis. Alur yang berjalan dipicu oleh sebuah *events* tertentu yang terjadi. Gateway complex digunakan untuk memvisualisasikan alur kompleks dalam proses bisnis. Sedangkan *gateway* OR digunakan untuk membuat alur alternatif sekaligus dapat berjalan secara paralel pada proses bisnis.

Gambar 2.4 Simbol - Simbol *Gateway*
Sumber (Object Management Group, 2011)

2.4.2 *Connecting object*

Connecting object merupakan elemen yang menghubungkan *flow object*. Connecting object memiliki tiga jenis yaitu alur *sequence* (*sequence flow*), alur pesan (*message flow*), dan asosiasi (*association*). Notasi untuk masing – masing *connecting object* dapat dilihat pada gambar 2.5. Sedangkan untuk penjelasan dari masing – masing jenis *connecting object* adalah sebagai berikut (Object Management Group, 2011):

1. *Sequence flow*
   Objek penghubung yang menunjukkan urutan aktivitas yang dilakukan dalam proses dan diwakili dengan garis yang solid. Setiap aliran hanya memiliki satu sumber dan hanya satu target. *Sequence flow* dapat melintasi batas antara *lanes pool* tetapi tidak dapat melintasi batas *pool*.
2. *Message flow*
   Objek penghubung yang menunjukkan aliran pesan antara dua *participants*. *Message flow* diwakili oleh garis putus – putus.
3. *Association*
   Objek penguhung yang digunakan untuk menghubungkan informasi dan artifacts dengan *flow object*. Sebuah asosiasi direpresentasikan sebagai garis putus – putus dengan kepala panah untuk mewakili arah.

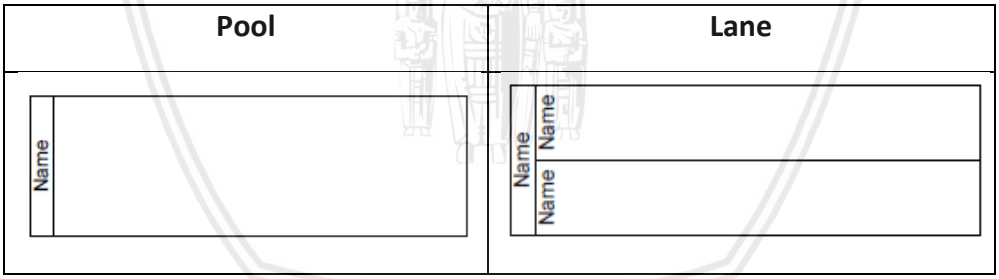
Sequence flow	*Message flow*	*Association*
		

Gambar 2.5 Simbol *Connecting Object*
Sumber (Object Management Group, 2011)

2.4.3 *Swinlanes*

Swimlane adalah wadah grafis untuk mempartisi satu set kegiatan dari kegiatan lain. BPMN memiliki dua tipe *swimlanes* yang berbeda yaitu *pool* dan *lane*. Notasi dari masing – masing swinlanes dapat dilihat pada gambar 2.6. Berikut adalah penjelasan dari *pool* dan *lane* (Object Management Group, 2011) :

1. *Pool*
   Sebuah *pool* mewakili partisipan dalam kolaborasi. Secara grafis, pool adalah wadah untuk mempartisi proses dari *pools* / *participants* lainnya.
2. *Lane*
   Lane merupakan sebuah partisi yang digunakan untuk mengatur dan mengkategorikan aktivitas dalam *pool*. Sebuah *lane* akan memperluas seluruh panjang *pool* baik secara vertikal maupun horizontal.



Gambar 2.6 Simbol Swimlanes
Sumber (Object Management Group, 2011)

2.4.4 Artifacts

Artifacts merupakan objek grafis yang memberikan informasi pendukung tentang proses atau elemen dalam proses. Namun tidak secara langsung mempengaruhi aliran proses. BPMN dirancang untuk memungkinkan pemodel dan alat pemodelan fleksibilitas untuk *artifacts* sebanyak yang diperlukan sesuai dengan konteks dari proses bisnis yang dimodelkan. Ada tiga tipe *artifacts*, yaitu data object, group, dan annotation. Notasi dari masing – masing *artifacts* dapat dilihat pada gambar 2.7. Berikut adalah penjelasan dari masing – masing *artifacts* yang ada pada BPMN.

1. *Data object*

Mekanisme untuk menunjukkan begaimana data dibutuhkan atau diproduksi oleh aktivitas. *Data object* dihubungkan degan aktivitas melalui *associations*.

2. *Group*
   Diwakili dengan persegi panjang dengan ujung bulat yang digambarkan dengan garis putus - putus. *Group* dapat digunakan untuk tujuan dokumentasi atau analisis, tetapi tidak mempengaruhi *sequence flow*.
3. *Anotation*
   Mekanisme untuk pemodel memberikan informasi teks tembahan untuk pembaca dari diagram BPMN.

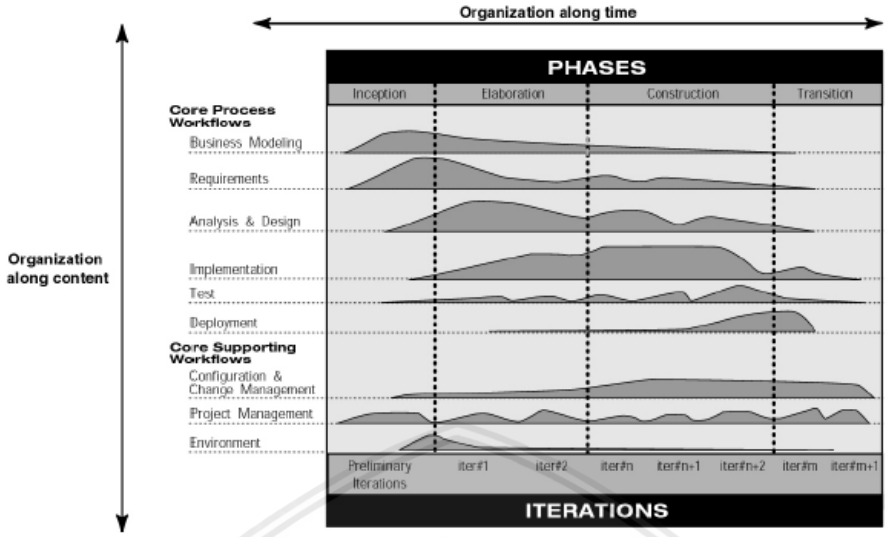
Data object	Group	Annotation
		

Gambar 2.7 Simbol *Artifacts*
Sumber (Object Management Group, 2011)

2.5 *Rational Unified Process* (RUP)

Menurut (IBM, Rational Unified Process Best Practices for Software Development Teams, 1998), *Rational Unified Process* (RUP) adalah proses rekayasa perangkat lunak yang memberikan pendekatan disiplin untuk menetapkan tugas dan tanggung jawab dalam sebuah organisasi. Tujuannya adalah untuk memastikan produksi perangkat lunak berkualitas tinggi yang memenuhi kebutuhan pengguna akhir, dalam jadwal dan anggaran yang dapat diprediksi . Proses dapat digambar dalam dua dimensi atau sepanjang dua sumbu yaitu :

1. Sumbu *horizontal* mewakili waktu dan menunjukkan aspek dinamik proses saat ditetapkan, dan dinyatakan dalam bentuk siklus, fase, iterasi , dan tonggak sejarah.
2. Sumbu *vertical* mewakili aspek statis dari proses, hal tersebut dijelaskan dalam hal aktivitas, artefak, pekerja dan alur kerja.



Gambar 2.8 *Rational Unified Process*

Sumber (IBM, Rational Unified Process Best Practices for Software Development Teams, 1998)

Melalui gambar 2.8 diatas dapat diketahui bahwa di dalam RUP terdapat 4 fase yaitu :

1. *Inception*

   Inception merupakan fase pertama dari fase RUP. Fase *inception* ini memiliki tujuan utama untuk mencapai persetujuan dari semua pemangku kepentingan. Menurut (Kruchten, 2003) tujuan utama dari fase ini meliputi:

   1. Menentukan ruang lingkup dari perangkat unak yang akan dikembangkan dan menetukan batasannya, termasuk konsep operasional, kriteria penerimaan, serta apa yang ada di dalam dan di luar sistem.
   2. Menentukan fungsionalitas dari sistem.
   3. Menentukan setidaknya satu kandidat solusi terhadap beberapa skenario utama.
   4. Memperkirakan keseluruhan biaya dan jadwal untuk keseluruhan proyek.
   5. Memperkirakan resiko.

   Menurut (Kruchten, 2003) kegiatan yang dilakukan pada fase ini meliputi merumuskan ruang lingkup, merencanakan kasus bisnis dan mengevaluasi *alternative* untuk manajemen resiko, kepegawaian, rencana proyek, biaya dan jadwal. Serta menentukan keputusan untuk *make / buy / reuse* sehingga biaya, jadwal, dan sumber daya dapat diperkirakan. Pada fase *inception* ini menurut (Kroll & Kruchten, 2003) kebanyakan proyek memiliki satu iterasi.

2. *Elaboration*

   Fase *elaboration* bertujuan untuk membuat dasar arsitektur sistem yang akan diimplementasikan pada fase *construction* (Kroll & Kruchten, 2003). Pada fase *elaboration* ini terdapat beberapa aktivitas yang dilakukan yaitu:

   1. Mendapatkankan pemahaman yang lebih mendalam mengenai kebutuhan yang berguna untuk membuat desain yang lebih detail.
   2. Melakukan perancangan, implementasi dan menguji struktur kerangka sistem.
   3. Membuat estimasi jadwal, biaya dan mengurangi resiko.
   4. Memperbaiki kasus pengembangan.

   Pada fase ini biasanya membutuhkan satu kali iterasi.

3. *Construction*

   Construction merupakan tahap ketiga dari fase RUP. Pada tahap ini dilakukan implementasi terhadap hasil rancangan pada fase *elaboration*. Pengujian sistem yang fokus pada implementasi kode program juga dilakukan pada fase ini. Fokus dari fase *construction* ini adalah melakukan pengembangan komponen dan fitur-fitur dari sistem. iterasi pada fase *construction* umunya lebih banyak dibandingkan dengan fase lainya.

4. *Transition*

   Pada tahap *transition* dilakukan *deployemnet* atau instalasi. Pelatihan user, pemeliharaan sistem dan pengujian sistem juga dilakukan pada fase ini. Menurut (Kroll & Kruchten, 2003) iterasi yang terjadi pada fase *transition* ini bervarisi tergantung pada kompleksitas sistem.

Didalam RUP terdapat terdapat enam *best practice* dalam pengembangan perangkat lunak modern yang sesuai dengan berbagai proyek dan organisasi (IBM, Rational Unified Process Best Practices for Software Development Teams, 1998). Enam *best practice* tersebut adalah :

1. Mengembangkan perangkat lunak secara *iterative*

   Merencanakan penambahan sistem berdasarkan prioritas pelanggan dan mengembangkan fitur sistem dengan prioritas tertinggi di awal pengembangan proses.

2. Mengelola persyaratan

   Mendokumentasikan persyaratan dan mencatat perubahan persyaratan tersebut serta menganalisa dampak perubahan dari sistem tersebut.

3. Menggunakan arsitektur berbasis komponen

   Arsitektur sisitem disusun menjadi komponen – komponen

4. Perangkat lunak model visual

   Menggunakan model UML untuk membuat tampilan perangkat lunak statis dan dinamis.

5. Verifikasi kualitas perangkat lunak

   Hal ini bertujuan untuk memastikan perangkat lunak memenuhi standar kualitas organisasi.

6. Mengontrol perubahan pada perangkat lunak

Perubahan pada perangkat lunak dikontrol menggunakan sisitem manajemen perubahan dan prosedur dan perangkat pengelolaan konfiguras.

2.6 Pemodelan *Use Case*

Pemodelan *use case* menggambarkan kebutuhan fungsional dari sebuah sistem yang berguna untuk menghubungkan antara kebutuhan stakeholder dan persyaratan perangkat lunak (IBM, writing Good Use Cases, 2007). Pemodelan *use case* terdiri dari aktor dan *use case*. *Use case* berfungsi dalam merepresentasikan unit interaksi diskrit antara pengguna dimana setiap *use case* memiliki deskripsi yang menggambarkan fungsionalitas dari sistem yang akan dikembangkan (SparxSystems, 2004). Menurut IBM pemodelan *use case* merupakan instrument perencanaan yang sangat kuat.

2.6.1 Identifikasi Tipe Pemangku Kepentingan

Pemangku kepentingan adalah seorang individu yang secara material dipengaruhi oleh hasil dari sistem atau proyek yang menghasilkan sistem (Bittner & Spence, 2002). Langkah pertama dalam memahami pemangku kepentingan adalah dengan mengidentifikasi jenis-jenis pemangku kepentingan yang dipengaruhi oleh sistem dan selanjutnya adalah mengidentifikasi peran dari pemangku kepentingan tersebut. Dalam mengidentifikasi peran pemangku kepentingan terdapat aktivitas memahami bagaimana peran berinteraksi dengan sistem atau proyek.

Menurut bittner dan spence ada lima ketegori pemangku kepentingan, yaitu :

1. Pengguna : orang – orang yang akan mengambil peran yang ditentukan oleh para aktor dalam *use case*.
2. Sponsor : para manajer bisnis, pemodal, pemegang saham, juara, kepala departemen, penjual, pemasar, anggota komite pengarah, dan orang lain yang berinvestasi dalam produksi sistem. Para pemangku kepentingan ini adalah pengguna tidak langsung dari sistem atau hanya dipengaruhi oleh hasil bisnis.
3. Pengembang : setiap pengembang yang terlibat dalam produksi dan dukungan sistem seperti manajer proyek, pengelola sistem, penguji, staf pendukung, perancang, pembuat kode.
4. Pihak berwenang : para ahli dalam aspek tertentu dari masalah atau solusi domain.
5. Pelanggan : orang – orang atau organisasi yang akan membeli sistem.

Akan tetapi pemangku kepentingan tidak terbatas pada lima kategori diatas. Daftar tipe pemangku kepentingan tersebut dapat berubah tergantung bagaimana identifikasi peran yang terkena dampak dari sistem.

2.6.2 Analisa Masalah

Masalah dapat didefinisikan sebagai perbedaan antara hal-hal yang dirasakan dan hal-hal yang diinginkan. Tujuan dari analisis masalah ini adalah untuk memastikan bahwa semua pihak yang terlibat setuju dengan masalah apa yang harus dipecahkan. Cara terbaik untuk menganalisa masalah adalah dengan membuat pernyataan masalah (Bittner & Spence, 2002). Pernyataan masalah ini merupakan ringkasan solusi netral dari wawancara yang dilakukan dengan beberapa pemangku kepentingan tentang masalah yang harus dipecahkan. Analisis masalah dapat dituliskan dengan menggunkanan tabel seperti pada tabel 2.2

Tabel 2.2 Kerangka Dokumentasi Pernyataan Masalah

The problem of	Deskripsi dari masalahnya
Affects	Para pemangku kepentingan yang terpengaruh oleh sistem
The impact of which is	Apa dampak dari masalah ini?
A successful solution would	Daftar beberapa manfaat utama atau solusi yang sukses

2.6.3 Identifikasi Kebutuhan Pemangku Kepentingan dan Pengguna

Para pemangku kepentingan biasanya memiliki perspektif yang berbeda tentang masalah dan kebutuhan yang harus ditangani. Kebutuhan pemangku kepentingan didefenisikan sebagai cerminan masalah bisnis, personal atau operasional yang harus diatasi untuk membenarkan pertimbangan, pembelian, atau penggunaan sistem baru (Bittner & Spence, 2002). Dengan mengidentifikasikan kebutuhan pemangku kepentingan memungkinkan untuk memahami bagaimana dan sejauh mana aspek-aspek berbeda dari masalah yang mempengaruhi berbagai jenis pemangku kepantingan. Menurut Bittner dan Spence Identifikasi kebutuhan pemangku kepentingan dan pengguna dapat dilakukan dengan menggunakan teknik seperti :

a. Wawancara
b. Brainstorming
c. *Prototyping* konseptual
d. Kuesioner
e. Analisis kompetitif.

2.6.4 Identifikasi Fitur

Fitur adalah kemampuan tingkat tinggi (layanan atau kualitas) dari sistem yang diperlukan untuk memberikan manfaat kepada pengguna dan membantu memenuhi kebutuhan pemangku kepentingan dan pengguna (Bittner & Spence, 2002). Fitur juga memberikan ringkasan manfaat yang ditawarkan oleh sistem yang dibangun. Menurut Bittner dan Spence penamaan fitur harus memiliki

tingkat detail yang umum agar dapat dipahami oleh semua pemangku kepentingan. Hal ini ditekankan karena fitur juga digunakan untuk merangkum kemampuan dan kualitas sistem yang akan dibangun. Fitur – fitur sistem dapat didokumentasikan dalam banyak cara. Salah satu cara yang mendokumentasikan fitur adalah dengan menggunakan tabel seperti berikut.

Tabel 2.3 Contoh Tabel Identifikasi Fitur

Kode Fitur	Deskripsi
Berisi kode identifikasi fitur	Berisi penjelasan tentang yang dapat dilakukan oleh sistem.

2.6.5 Identifikasi Persyaratan Fungsional dan Persyaratan Non Fungsional

Persyaratan perangkat lunak terbagi menjadi dua kategori yaitu persyaratan fungsional dan persyaratan non fungsional. Persyaratan fungsional menggambarkan apa yang harus dilakukan oleh sistem (Sommerville, 2011). Persyaratan fungsional menentukan perilaku input dan output dari suatu sistem. Sedangkan persyaratan non fungsional merupakan persyaratan yang tidak secara langsung terkait dengan layanan spesifik yang disampaikan oleh sistem kepada penggunanya (Sommerville, 2011). Persyaratan non fungsional menentukan kualitas lain yang harus dimiliki sistem, seperti yang terkait dengan kegunaan, keandalan, kinerja dan dukungan sistem. Menurut sommerville peryaratan non fungsional muncul melalui kebutuhan pengguna karena keterbatasan anggaran, kebijakan organisasi, kebutuhan interoperabilitas dengan perangkat lunak atau sistem perangkat keras lain, atau faktor eksternal seperti peraturan keselamatan atau undang-undang privasi. Persyaratan fungsional dan persyaratan non fungsional dapat didokumentasikan dalam berbagai cara, namun cara yang paling umum digunakan adalah dengan mendeskripsikan persyaratan-persyaratan tersebut dalam kalimat deskriptif (Bittner & Spence, 2002).

2.6.6 Pemodelan *Use Case*

Model *use case* digunakan untuk menggambarkan semua kemungkinan dalam hal penggunaan sistem (Bittner & Spence, 2002). Analisis persyaratan perangkat lunak dijadikan panduan dalam menentukan *use case*. Seringkali *use case* berisi persyaratan fungsional dari sistem. Dalam memodelkan *use case* terdapat dua komponen utama yaitu aktor dan *use case*

Aktor mendefiniskan peran yang dapat dimainkan oleh pengguna saat berinteraksi dengan sistem. Aktor dapat mewakili peran manusia, perangkat keras atau sistem lain yang berkaitan dengan sistem. Dalam pelaksanaanya satu objek dapat memainkan bagian dari banyak aktor. Misalnya, satu orang dapat menjadi *loan officer* dan pelanggan. Sedangkan seorang aktor mewakili satu aspek dari suatu objek.

Use case menggambarkan bagaimana aktor menggunakan sistem untuk mencapai tujuan dan apa yang dilakukan oleh sistem untuk aktor agar mencapai tujuan tersebut. Ini berarti bahwa use case menggambarkan bagaimana sistem dan para aktornya berkolaborasi untuk memberikan sesuatu yang berharga bagi setidaknya salah satu aktor. Dalam menentukan alur penggunaan dapat dilakukan dengan menjelaskan aliran peristiwa dalam teks agar dapat memudahkan untuk memahaminya.

2.7 Pendekatan Berorientasi Objek

Pendekatan berorientasi objek merupakan strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya (Shalahudin & Sukamto, 2014). Pada pengembangan berorientasi objek terdapat *Object Oriented Analysis* (OOA) dan *Object Oriented Design* (OOD). *Object Oriented Analysis* merupakan metode analisis yang memeriksa persyaratan dari perspektif kelas dan objek yang ditemukan dalam domain masalah (Booch, Object Oriented Analysis And Design, 1994). *Object Oriented Design* merupakan metode desain yang mencakup proses dekomposisi berorientasi objek dan notasi untuk menggambarkan baik *logical* dan *physical* serta model statis dan dinamis dari sistem yang sedang dirancang. Sedangkan *Object Oriented Programming* (OOP) merupakan metode implementasi di mana program disusun sebagai kumpulan objek koorporatif, yang masing – masing mewakili sebuah *instance* dari beberapa kelas, dan kelas – kelasnya adalah semua anggota dari hirarki kelas yang disatukan melalui relasi pewarisan. OOA, OOD dan OOP saling terkait dimana hasil dari OOA berfungsi sebagai model untuk memulai desain berorientasi objek. Selanjutnya hasil dari OOD digunakan sebagai cetak biru untuk menerapkan sistem dengan menggunakan metode OOP.

2.8 UML

Unified Modeling Language (UML) adalah bahasa standar untuk menulis *blueprints* perangkat lunak (Booch, Rumbaugh, & Jacobson, The Unified Modeling Language User Guide Second Edition, 2005). UML dapat digunakan untuk memvisualisasikan, menentukan, membuat, dan mendokumentasikan artefak dari sistem perangkat lunak yang intensif. UML sesuai untuk sistem pemodelan mulai dari sistem informasi perusahaan hingga aplikasi berbasis Web terdistribusi dan bahkan sampai sistem *embedded real time* yang sulit.

Sebagai suatu bahasa pemodelan sistem / perangkat lunak, UML memiliki 3 ciri khas. Ciri khas tersebut adalah :

1. Merupakan *Use-Case Driven Language.*
   Pada pendekatan UML, pengembang mulai dengan mencoba menangkap kebutuhan pengguna dengan membuat diagram *use case* yang merupakan pendekatan untuk memahami apa yang sesungguhnya diharapkan pengguna.

Kemudian pengembang menganalisis dan melakukan perancangan sistem / perangkat lunak dengan basis diagram *use case*.

2. Merupakan *Architecture-Centric Language.*
   Para pengembang dengan metode UML mengatakan bahwa pengendalian kegiatan pengembangan sistem / perangkat lunak dengan *use case diagram* belumlah cukup memadai untuk menghasilkan sistem / perangkat lunak yang berkualitas tinggi. Sesuatu yang lain dibutuhkan agar pengembang mendapatkan sistem / perangkat lunak yang benar – benar bekerja dengan baik. Sesuatu itu adalah arsitektur. Arsitektur adalah pandangan umum yang disetujui oleh pengembang dan calon pengguna mengenai sistem / perangkat lunak yang sedang dikembangkan. Yang dimaksud arsitektur perangkat lunak / sistem adalah :
   a. Organisasi dari sistem . Perangkat lunak
   b. Elemen – elemen serta antarmuka masing – masing yang tercakup dalam sistem / perangkat lunak. Termasuk didalamnya adalah bagaimana elemen – elemen berperilaku serta berkolaborasi sesuai yang diharapkan calon pengguna.
   c. Secara umum, arsitektur sistem memberi perhatian khusus juga pada fungsionalitas, kinerja, kelenturan, kemudahan dipahami, hingga ke batasan – batasan teknologi serta ekonomi, bahkan hingga ke faktor – faktor estetika.
3. Merupakan *Iterative And Incremental Language.*
   Pengembangan perangkat lunak / sistem tidaklah bersifat linier. Setiap langkah maju seharusnya dilakukan dengan proses menengok ke belakang dengan mempertanyakan apakah pengembang sudah melakukan hal yang benar, yang sesuai dengan *use case* serta arsitektur yang telah disepakati. Jika belum, pengembang bisa kembali ke langkah sebelumnya dengan melakukan perbaikan – perbaikan yang diperlukan sehingga pada langkah selanjutnya pengembang mendapatkan pemahaman tentang sistem secara lebih baik lagi.

UML menyediakan 4 macam diagram untuk memodelkan aplikasi perangkat lunak berorientasi objek (Sommerville, 2011), yaitu:

1. *Use Case Diagram*, yang menunjukkan interaksi anatara sistem dan lingkungannya.
2. *Class Diagram*, yang menunjukkan kelas objek dalam sistem dan asosiasi antara kelas – kelas.
3. *Activity Diagram*, yang menunjukkan aktivitas yang terlibat dalam suatu proses atau dalam pengolahan data.
4. *Sequence Diagram*, yang menunjukkan interaksi antara actor dan sistem dan antar komponen sistem.

2.8.1 Use Case Diagram

Use case diagram adalah sebuah diagram yang menunjukkan hubungan antara *actors* dan *use case* (IBM, writing Good Use Cases, 2007). *Use case*

diagram sangat penting untuk memodelkan perilaku suatu sistem, subsistem, atau kelas. Melalui *use case diagram* dapat diketahui fungsi apa saja yang ada di dalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi – fungsi tersebut. Yang ditekankan pada *use case diagram* ini adalah "apa" yang diperbuat sistem dan bukan "bagaimana". Berikut ini adalah bagian dari sebuah *use case* diagram :

a. *Use Cases*

Use cases menjelaskan tentang tindakan/aksi yang dilakukan oleh *actors*. Setiap use case mengekspresikan goal dari sistem yang harus dicapai. *Use case* digambarkan dalam bentuk *elips* yang *horizontal* dan diberi nama sesuai dengan *goal*-nya. Setiap *use case* biasanya memiliki *trigger* / pemicu yang menyebabkan *use case* memulai misalnya adalah pasien mendaftar dan membuat janji baru atau meminta untuk membatalkan atau mengubah janji yang sudah ada. Ada dua *trigger* yaitu *trigger* eksternal, seperti pelanggan memesan atau alarm kebakaran berbunyi dan *trigger* temporal, seperti tanggal pengembalian buku terlewati di perpustakaan atau keterlambatan bayar sewa.

b. Actors

Actors adalah seorang peran yang berinteraksi dengan sistem. *Actors* meliputi baik manusia maupun organisasi yang saling bertukar informasi. Cara mudah untuk menemukan *actors* adalah dengan bertanya hal – hal berikut :

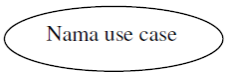
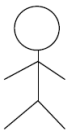
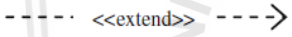
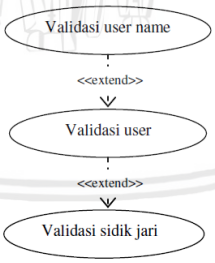
- Siapa yang akan menggunakan sistem?
- Apakah sistem tersebut akan memberikan nilai bagi *actors*?

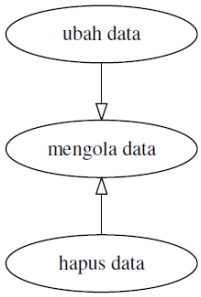
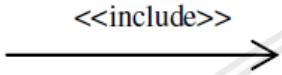
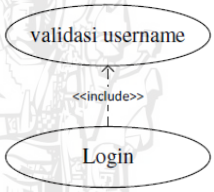
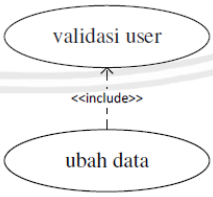
c. Relationship

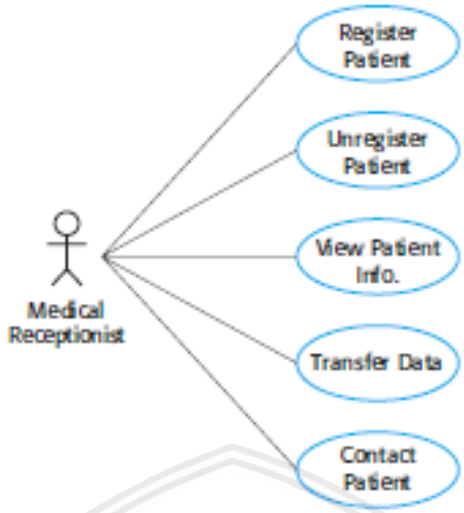
Relationship adalah hubungan antara *use cases* dengan *actors*. *Relationship* dalam *use case* diagram meliputi :

1. Asosiasi antara aktor dan *use case*.
   Hubungan antara aktor dan *use case* yang terjadi karena adanya interaksi antara kedua belah pihak. Asosiasi tipe ini menggunakan garis lurus dari aktor menuju *use case* baik dengan menggunakan mata panah terbuka ataupun tidak.
2. Asosiasi antara 2 *use case.*
   Hubungan antara *use case* yang satu dan *use case* lainnya yang terjadi karena adanya interaksi antara kedua belah pihak. Asosiasi tipe ini menggunakan garis putus-putus/garis lurus dengan mata panah terbuka di ujungnya.
3. Generalisasi antara 2 aktor.
   Hubungan *inheritance* (pewarisan) yang melibatkan actor yang satu (*the child*) dengan actor lainnya (*the parent*). Generalisasi tipe ini menggunakan garis lurus dengan mata panah tertutup di ujungnya.
4. Generalisasi antara 2 *use case*.
   Hubungan *inheritance* (pewarisan) yang melibatkan *use case* yang satu (*the child*) dengan *use case* lainnya(*the parent*). Generalisasi tipe ini menggunakan garis lurus dengan mata panah tertutup di ujungnya.

Tabel 2.4 Simbol- Simbol Diagram Use Case

No	Simbol	Deskripsi
1.	*Use case* 	*Use case* merupakan fungsionalitas sistem yang saling bertukar pesan antar unit atau aktor. *Use case* ini biasanya dinyatakan dengan menggunakan kata kerja diawal frase nama *use case.*
2.	Aktor/*actor* 	Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem yang akan dibuat, jadi walaupun simbol dari aktor adalah gambar orang belum tentu aktor tersebut berupa orang. Aktor biasanya dinyatakan dengan menggunakan kata benda di awal frase nama aktor.
3.	Assosiasi/*association* 	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
4.	Exstensi/*extend* 	Relasi *use case* tambahan kesebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu, mirip dengan prinsip *inheritance* pada pemrograman berorientasi objek, biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan, misal  Arah panah mengarah pada *use case* yang ditambahkan, biasanya *use case* yang menjadi *extend*-nya merupakan jenis yang sama dengan *use case* yang menjadi induknya.
5.	Generalisasi/*generalization* 	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya, misalnya :

No	Simbol	Deskripsi
		 Arah panah mengarah pada *use case* yang menjadi generalisasinya (umum).
6.	Menggunakan / *include* / *uses* 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini. Ada dua sudut pandang yang cukup besar mengenai *include* di *use case* : - *Include* berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan, missal pada kasus berikut :  - *Include* berarti *use case* yang tambahan akan selalu melakukan pengecekan apakah *use case* yang di tambahkan telah dijalankan sebelum *use case* tambahan di jalankan, misal pada kasus berikut :  Kedua interpretasi di atas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan.



Gambar 2.9 *Use Case Diagram*
Sumber (Sommerville, 2011)

Gambar 2.9 merupakan contoh dari *use case diagram.* Pada contoh diagram tersebut terdapat aktor *medical receptionist* yang berhubungan dengan lima use case yaitu *register patient, unregister patient, view patient info, transfer data,* dan *contact patient*. Hubungan yang digambarkan pada contoh *use case diagram* diatas adalah hubungan asosiasi yang dinotasikan dengan garis lurus.

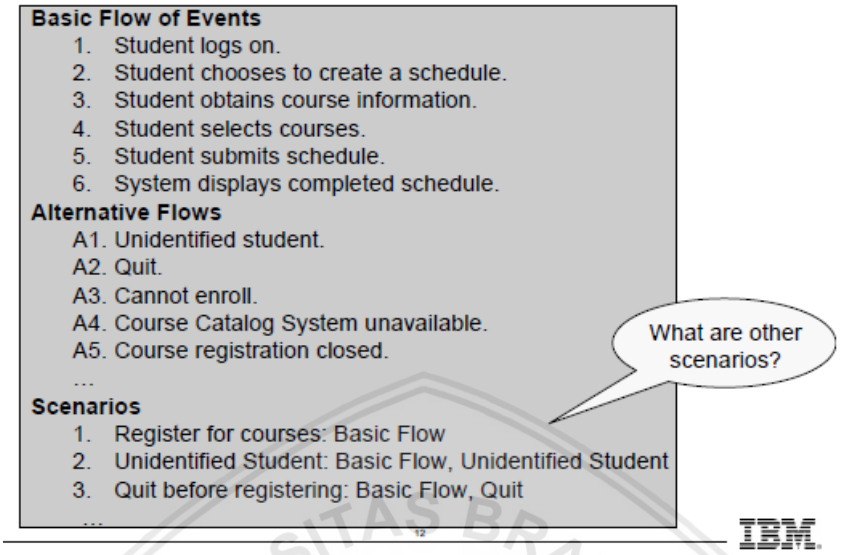
2.9.1 Use Case Scenario

Use case scenario menggambarkan contoh penggunaan sistem (contoh kasus penggunaan). *Scenario* mungkin melibatkan aliran dasar dan sejumlah arus alternatif dalam sejumlah kombinasi (IBM, writing Good Use Cases, 2007). Jumlah *scenario* yang dibuat bersifat absolut artinya *scenario* dapat dibuat sebanyak yang dibutuhkan untuk memahami sistem yang sedang dikembangkan. Menurut bittner dan spence dalam bukunya *use case modeling* mendokumentasikan *scenario* berguna untuk alasan – alasan berikut :

1. *Scenario* akan cocok dengan kasus uji.
2. *Scenario* adalah apa yang sebenarnya dilakukan, sehingga mereka berguna untuk mendiskusikan apa yang dilakukan sistem dalam praktiknya.
3. *Scenario* berguna untuk analisis dan desain, karena mereka membantu para pengembang untuk berpikir tentang bagaimana sistem akan digunakan.

Scenario dibuat di bagian terpisah dari spesifikasi *use case* dan *scenario* diberi nama deskriptif dan daftar aliran yang membentuk *scenario*. Beberapa orang menulis *scenario* terlebih dahulu dan kemudian mengekstraksi kasus penggunaan, tetapi ada juga yang menemukan kasus penggunaan terlebih

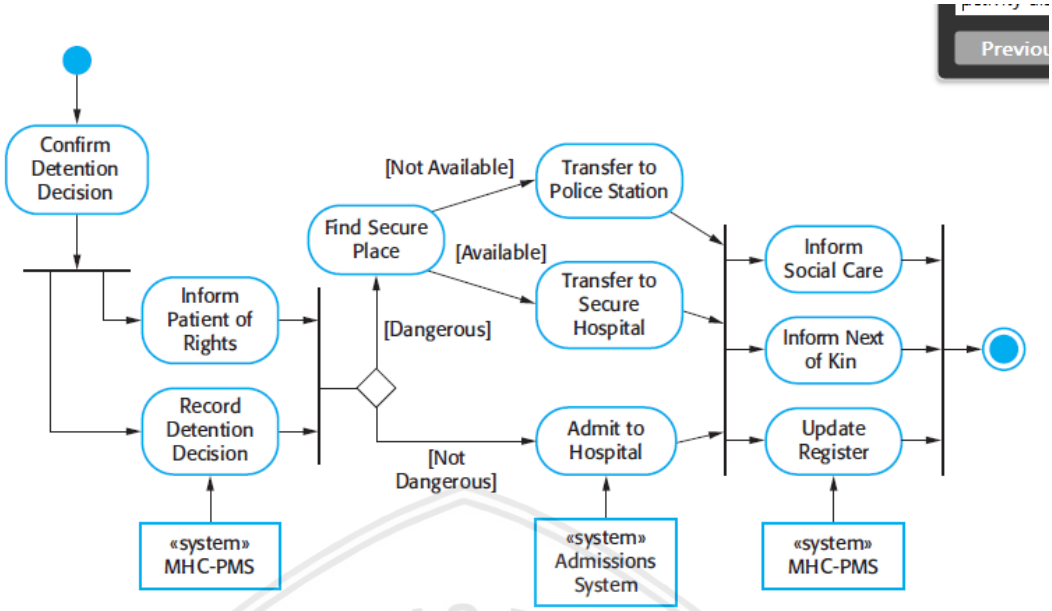
dahulu dan memvalidasi kasus penggunaan tersebut dengan menulis *scenario*. Berikut adalah contoh dari penulisan *use case scenario*.



Gambar 2.10 Contoh dari scenario
Sumber (IBM, writing Good Use Cases, 2007)

2.8.2 Activity Diagram

Activity diagram adalah salah satu dari lima diagram dalam UML untuk memodelkan aspek – aspek dinamis dari sistem. *Activity diagram* pada dasarnya adalah *flowchart*, menunjukkan aliran kontrol dari aktivitas ke aktivitas (Booch, Rumbaugh, & Jacobson, The Unified Modeling Language User Guide Second Edition, 2005). *Activity diagram* menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem. Contoh dari *activity diagram* adalah gambar 2.11.



Gambar 2.11 *Activity Diagram*
sumber (Sommerville, 2011)

Activity diagram memiliki beberapa notasi atau simbol. Notasi – notasi tersebut antara lain adalah statu awal, aktivitas, percabangan/decision, penggabungan/join, status akhir dan *Activity Partition*. Berikut adalah penjelasan dari simbol-simbol yang ada pada *activity diagram* :

Tabel 2.5 Simbol -Simbol *Activity Diagram*

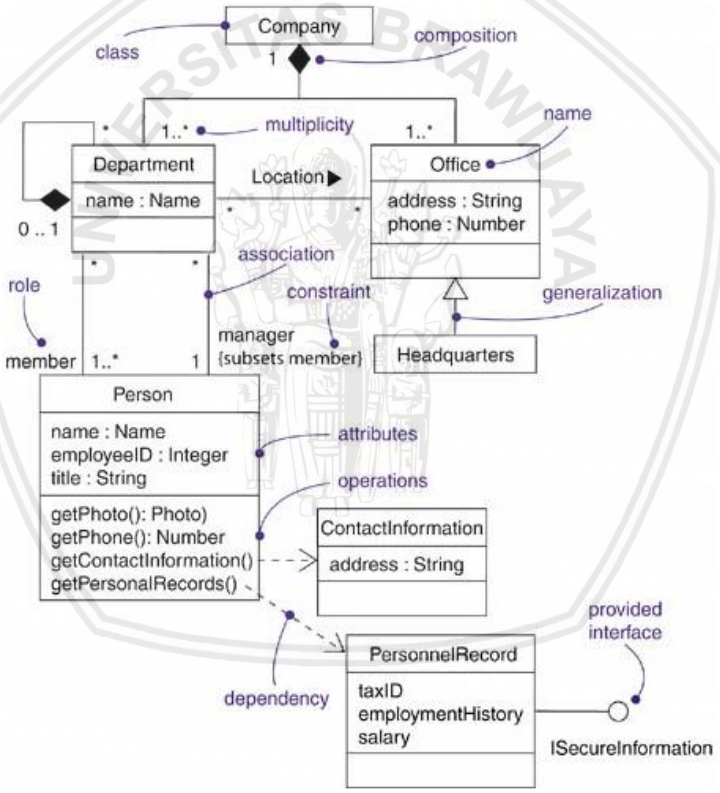
No.	Simbol	Deskripsi
1.	Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
2.	Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
3.	Percabangan/*decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
4.	Penggabungan/join 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
5.	Status akhir 	Status akhir yang dilakukan oleh sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
6.	*Activity Partition*	Digunakan untuk mempartisi atau mengelompokkan *action* berdasarkan domain spesifik (seperti unit sistem atau entitas) yang

No.	Simbol	Deskripsi
		menjalankan *action* tersebut.

2.8.3 Class Diagram

Class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem (Shalahudin & Sukamto, 2014). Kelas memiliki apa yang disebut atribut dan *method* atau operasi. Berikut penjelasan atribut dan *method* :

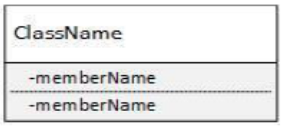
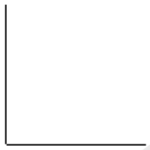
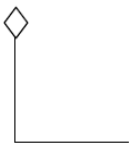
1. Atribut merupakan variable-variabel yang dimiliki oleh suatu kelas.
2. Operasi atau method adalah fungsi-fungsi yang dimiliki oleh suatu kelas.



Gambar 2.12 *Class Diagram*
Sumber (Booch, Rumbaugh, & Jacobson, The Unified Modeling Language User Guide Second Edition, 2005)

Gambar 2.12 merupakan contoh dari *class diagram*. *Class diagram* biasanya terdiri dari kelas, interface, dependensi, generalisasi, dan asosiasi. Berikut adalah penjelasan dari masing-masing notasi yang ada pada *class diagram* :

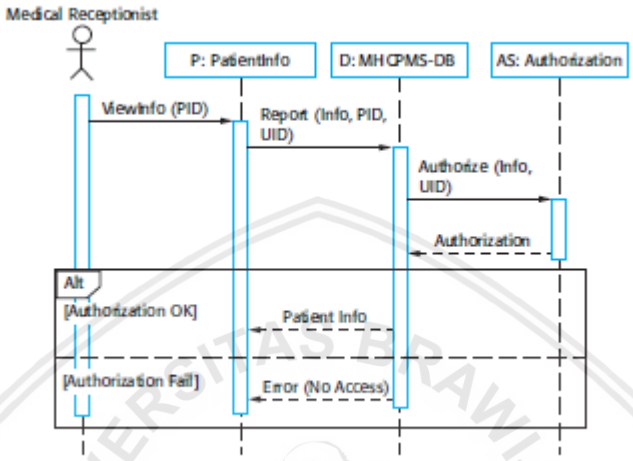
Tabel 2.6 Simbol- Simbol *Class Diagram*

No.	Simbol	Deskripsi
1.	Kelas 	Kelas pada struktur sistem.
2.	Antarmuka/*interface* 	Sama dengan konsep *interface* dalam pemrograman berorientasi objek.
3.	Asosiasi/*association* 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity.*
4.	Asosiasi berarah/ *directed Association* 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity.*
5.	Generalisasi 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum ke khusus).
6.	Kebergantungan/*dependensi* 	Relasi antar kelas dengan makna kebergantungan antar kelas.
7.	Agregasi/*aggregation* 	Relasi antar kelas dengan makna semua-bagian (*whole-part*).

2.8.4 Sequence Diagram

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dengan *massage* yang dikirimkan dan diterima antar objek (Shalahudin & Sukamto, 2014). Oleh karena itu untuk menggambarkan *sequence diagram* maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta *method* - *method* yang dimiliki kelas yang diinstansiasi menjadi objek itu. Membuat *sequence diagram* juga dibutuhkan untuk melihat skenario yang ada pada *use case*. Banyaknya *sequence diagram*

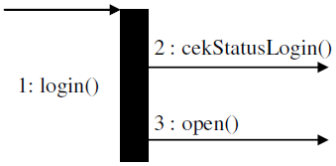
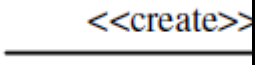
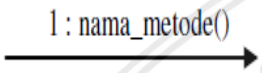
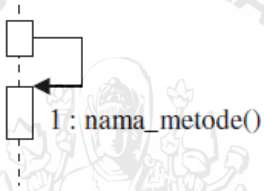
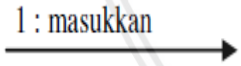
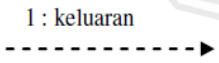
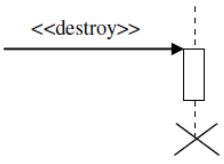
yang harus digambar adalah minimal sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup dalam *sequence diagram* sehingga semakin banyak *use case* yang didefinisikan maka *sequence diagram* yang harus dibuat juga semakin banyak. Berikut adalah contoh dari *sequence diagram:*



Gambar 2.13 *Sequence Diagram*
Sumber (Sommerville, 2011)

Sequence diagram memiliki beberapa notasi yang merepresentasikan *instances* dari sebuah kelas, antarmuka, komponen, dan *nodes,* beserta pertukaran pesan di antara *instances* tersebut. Penjelasan dari masing – masing notasi yang ada pada sequence diagram dapat dilihat pada tabel berikut :

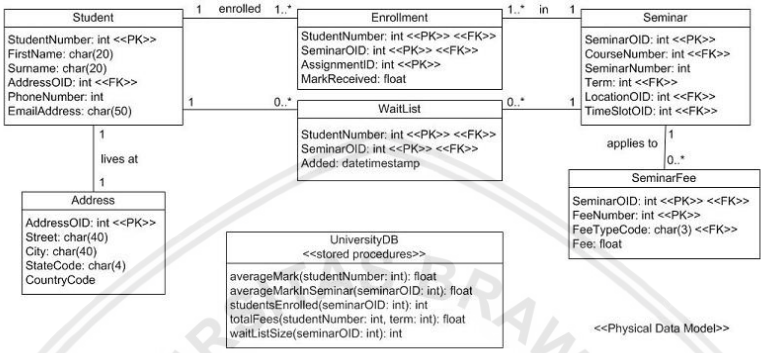
No.	Simbol	Deskripsi
1.	Aktor  Atau Nama aktor Tanpa waktu aktif	Pengguna dari sistem bisa berupa orang, mesin, sistem atau subsistem lain. Segala sesuatu yang berinteraksi dengan sistem disebut aktor.
2.	Garis hidup/*lifeline* 	Menyatakan kehidupan suatu objek.
3.	Objek Nama objek : nama kelas	Menyatakan objek yang berinteraksi pesan.
4.	Waktu aktif	Menyatakan objek dalam keadaan aktif dan

No.	Simbol	Deskripsi
		berinteraksi, semuanya yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya, misalnya  Maka cekStatusLogin() dan open() dilakukan didalam metode login().
5.`	Pesan tipe *create* 	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat.
6.	Pesan tipe *call* 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri.  Arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi/metode maka operasi/metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi.
7.	Pesan tipe *send* 	Menyatakan bahwa suatu objek mengirimkan data/masukkan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim.
8.	Pesan tipe *return* 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.
9.	Pesan tipe *destroy* 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaliknya jika ada *create* maka ada *destroy.*

2.9 Physical Data Model

Physical data model berguna untuk mambantu memvisualisasikan struktur *database* serta secara otomatis dapat memperoleh skema *database*

yang sesuai (sparxsystems). *Physical data model* menunjukkan struktur-struktur tabel yang terdiri dari nama kolom, batasan kolom, tipe data, *primary key*, *foreign key*, dan relasi antar tabel. Secara konseptual *physical data model* mirip dengan pemodelan kelas desain yang bertujuan untuk merancang skema internal dari *database*, menggambarkan tabel data, kolom data dari tabel tersebut, dan hubungan antar tabel (W.Ambler, 2004). Pemodelan *database* ke dalam *physical data model* dilakukan berdasarkan pada diagram kelas yang telah dibuat sebelumnya. Gambar 2.14 adalah contoh dari *Physical Data Model*.



Gambar 2.14 *Physical Data Model*
Sumber **(W.Ambler, 2004)**

2.10 *Model-View-Controller* (MVC)

Model-View-Controller (MVC) merupakan salah satu model infrastruktur WebApp dimana MVC memisahkan antramuka pengguna dari fungsionalitas WebApp dan konten informasi (Pressman, 2010). MVC terdiri dari 3 komponen yang saling berinteraksi satu sama lain yaitu *model, view* dan *controller*.

Menurut Pressman, *model* dalam MVC berisi semua konten khusus aplikasi dan logika pemrosesan, termasuk semua objek konten, akses ke sumber data / informasi eksternal, dan semua fungsi pemrosesan yang spesifik. *View* berisi semua fungsi antarmuka spesifik dan memungkinakn penyajian konten, termasuk semua objek konten, akses ke sumber data / informasi eksternal, dan semua fungsi pemrosesan yang dibutuhkan oleh pengguna akhir. Sedangkan *controller* memiliki peran dalam mengelola akses ke *model* dan *view* serta mengkoordinasikan aliran data di antara mereka.

Interaksi antara *model view* dan *controller* terjadi ketika ada sebuah *request* dari pengguna dimana *request* pengguna ditangani oleh *controller* untuk memilih objek tampilan yang berlaku berdasarkan *request* pengguna. Selanjutnya *request* akan dikirim ke *model* untuk mengimplementasikan fungsi atau mengambil konten yang diperlukan. Objek *model* dapat mengakses data yang disimpan dalam *database*, sebagai bagian dari penyimpanan data lokal atau sebagai kumpulan file independen. Data yang dikembangkan oleh *model*

selanjutnya diformat dan diatur oleh objek *view* untuk selanjutnya dikirimkan kembali ke *browser* klien untuk ditampilkan di komputer pengguna.

2.11 Pengujian

Pengujian dilakukan untuk menunjukkan bahwa suatu program yang dikembangkan dapat berjalan sesuai dengan tujuan yang telah ditentukan serta untuk menemukan cacat program sebelum perangkat lunak mulai digunakan (Sommerville, 2011). Menurut sommerville pengujian memiliki dua tujuan yang berbeda :

1. Untuk menunjukkan kepada pengembang dan pengguna bahwa perangkat lunak memenuhi persyaratan. Untuk perangkat lunak kustom, ini berarti setidaknya harus ada satu tes untuk setiap persyaratan dalam dokumen persyaratan. Untuk perangkat lunak generik, itu berarti harus ada tes untuk semua fitur sistem, ditambah kombinasi fitur ini, yang akan digabungkan dalam peluncuran produk.
2. Untuk menemukan situasi di mana perilaku perangkat lunak yang salah, tidak diinginkan, atau tidak sesuai dengan spesifikasi.

2.11.1 Validation Testing

Menurut (Pressman, 2010) *validation testing* berfokus pada tindakan yang terlihat pengguna dan keluaran yang dapat dikenali pengguna dari sistem. *validation testing* dilakukan melalui serangkaian tes yang menunjukkan kesesuaian dengan persyaratan. Sebuah rencana pengujian menguraikan kelas – kelas tes yang akan dilakukan, dan prosedur uji mendefinisikan kasus uji spesifik yang dirancang untuk memastikan bahwa semua persyaratan fungsional dipenuhi, semua karakteristik perilaku tercapai, semua konten akurat dan disajikan dengan tepat, semua persyaratan kinerja diperoleh, dokumentasi benar, dan kegunaan dan persyaratan lainnya terpenuhi. Setelah melakukan *validation testing* terdapat dua kondisi yang memungkinkan yaitu (Pressman, 2010) :

1. Fungsi atau karakteristik kinerja sesuai dengan spesifikasi dan diterima
2. Penyimpangan dari spesifikasi terungkap dan daftar kekurangan dibuat.

Dalam melakukan pengujian *validation testing* dibuat sebuah *test case* untuk memastikan tidak ada kesalahan dalam program dan jika ada kesalahan dalam program maka harus segera digambarkan.

2.11.2 Compatibility Testing

Compatibility testing adalah pengujian yang dilakukan dengan menjalankan webapp di berbagai host yang berbeda konfigurasi pada sisi *client* dan *server*. Tujuannya adalah untuk menemukan kesalahan yang spesifik untuk konfigurasi host yang berbeda (Pressman, 2010). Pengujian kompatibilitas yang

dilakukan pada penelitian ini adalah pengujian kompabilitas terhadap *browser*. Dari pengujian kompabilitas *browser* ini nantinya akan diperoleh sistem akan berjalan dengan baik pada *browser* apa saja. Karena setiap *browser* terkadang menghasilkan hasil atau tampilan yang berbeda dengan *browser* lainnya.

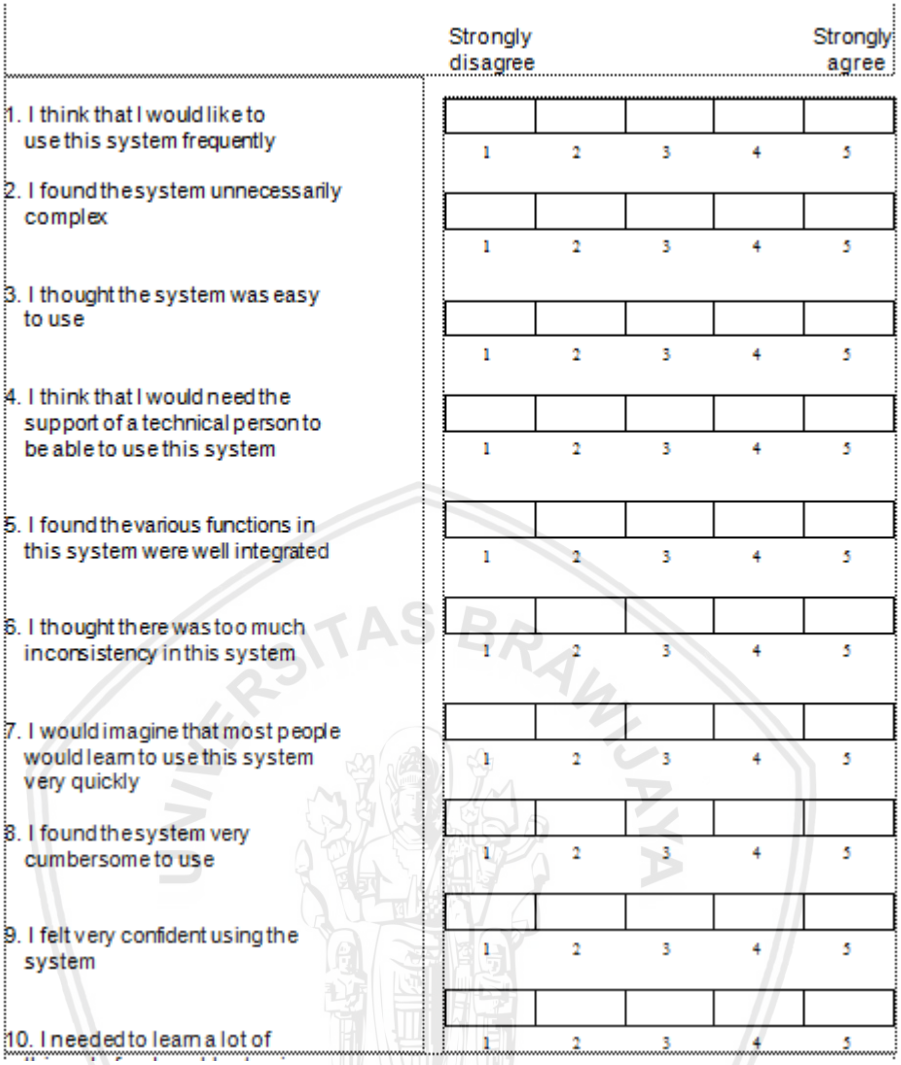
Pengujian kompabilitas dilakukan dengan menggunakan bantuan aplikasi *SortSite*. *SortSite* merupakan alat pengujian yang menjalankan ratusan *checkpoint* pemeriksaan kualitas pada setiap halaman web dan menghasilkan laporan yang mudah dibaca. *Browser* yang diuji pada aplikasi *SortSite* antara lain adalah *IE, Edge, Firefox , safari, opera, chrome, iOS* dan *Android*. Untuk mengetahui tingkat kompatibilitas sistem *SortSite* memiliki beberapa kategori masalah. Kategori tersebut ditunjukkan pada tabel 2.7.

Tabel 2. 7 Kategori Pengujian Kompabilitas

	Tidak terjadi permasalahan pada konten atau fungsionalitas
	Terdapat konten atau fungsionalitas yang hilang
	Terdapat masalah mayor pada layout atau performa
	Terdapat masalah minor pada layout atau performa

2.11.3 Usability Testing

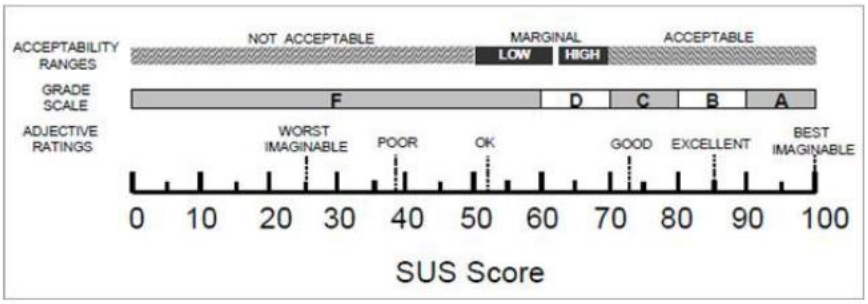
Pengujian *usability testing* dirancang untuk menentukan sejauh mana antarmuka WebApp dapat memudahkan pengguna (Pressman, 2010). Untuk melakukan pengujian *usability testing* dapat dilakukan dengan menggunakan *System Usability Scale (*SUS). *System Usability Scale* merupakan skala sederhana yang terdiri dari sepuluh item yang dapat memberikan pandangan global tentang penilaian subyektivitas kegunaan. SUS ini merupakan *skala likert* yang disajikan dalam bentuk pernyataan yang menyatakan bahwa responde sepakat dan tidak sepakat (Brooke, 1986). Pernyataan yang ada pada SUS mencakup berbagai aspek kegunaan sistem, seperti kebutuhan untuk dukungan, pelatihan, dan kompleksitas. Berikut adalah *System Usability Scale*.



Gambar 2.15 *System Usability Scale*

Sumber (Brooke, 1986)

Untuk mengitung skor dar SUS yang dihasilkan hal pertama yang dilakukan adalah dengan menjumlahkan kontribusi skor dari setiap item. Setiap kontribusi skor item akan berkisar dari 0 hingga 4. Untuk pertanyaan positif (item 1,3,5,7 dan 9) kontribusi skor adalah posisi skala dikurangi 1. Sedangkan untuk peranyaan negatif (item 2,4,6,8 dan 10) kontribusi skornya adalah 5 dikurangi posisi skala. Langkah selanjutnya adalah dengan mengkalikan jumlah skor dengan 2,5. SUS menghasilkan satu angka yang mewakili ukuran gabungan dari kegunaan keseluruhan sistem. Kisaran skor SUS yang dihasilkan adalah 0 hingga 100. Skor yang dihasilkan selanjutnya akan dibandingkan dengan *range* nilai *grade* skor. Pada skala *grade* skor SUS ini terdapat 5 grade yaitu F,D,E,C,B dan A. Intrepetasi dari *grade* skor SUS dapat dilihat pada gambar 2.16. Nilai 70 merupakan nilai rata rata yang diperoleh dari penelitian, sehingga *software, website* dapat dikatakan baik apabila mendapat skor lebih besar dari 70 atau sama dengan 70 yang berarti bahwa *software,website* berada pada grade C, B atau A. Jika skor yang diperoleh dibawah nilai rata – rata itu berarti terdapat masalah yang serius terkait kegunaan *software, website* yang harus ditangani.



Gambar 2.16 Intrepetasi Grade Skor SUS
Sumber (Bangor, Kortum, & Miller, 2009)

2.11.4 Test Case

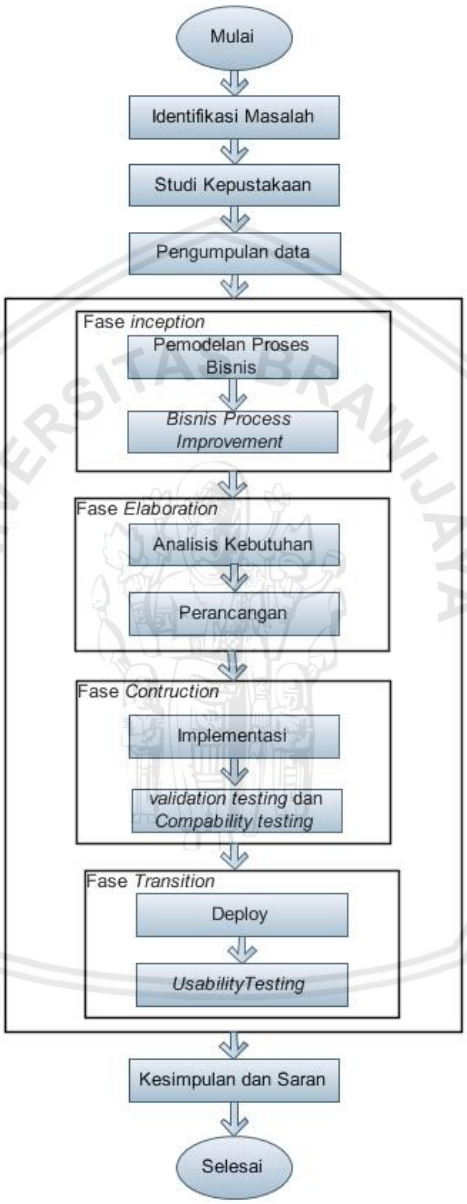
Test case merupakan sekumpulan instruksi yang dirancang untuk menemukan jenis kesalahan tertentu pada sistem perangkat lunak (B.B.Agarwal, Tayal, & Gupta, 2010). Setiap *test case* yang dibuat membutuhkan suatu dokumentasi yang tepat. Terdapat banyak format dokumen dalam membuat *test case*, salah satu format yang disarankan oleh (B.B.Agarwal, Tayal, & Gupta, 2010) adalah sebagai berikut.

Test case name	Test Case ID
Purpose of test	Testing object (unit, application, module, etc.)
Test attribute	
Tests focus (function, feature, process, interface, validation, verification, etc.)	
Test type (alpha, beta, unit, integration, system)	
Test process	A set of instructions for conducting the test-initial stating condition-inputs-specifications-output expected
Test results	Expected and actual and comparison, error description, post-process state
Action	Correction, authorization, and feedback through retest
Action to initialize the pre-test status	

Gambar 2.17 Format *Test Case*
Sumber (B.B.Agarwal, Tayal, & Gupta, 2010)

BAB 3 METODOLOGI

Pada Bab ini menjelaskan tentang bagaimana tahapan penelitian yang dilakukan untuk membangun sistem informasi pelaporan pendataan kependudukan kabupaten Malang sesuai dengan latar belakang yang telah dijelaskan sebelumnya.



Gambar 3.1 Diagram Alur Penelitian

3.1 Identifikasi Masalah

Tahap ini merupakan tahap untuk menemukan masalah – masalah yang terdapat pada Dispenduk Capil Kabupaten Malang. Proses identifikasi dilakukan melalui tahap wawancara dan obeservasi. Wawancara dilakukan dengan menggunakan metode wawancara tidak terstruktur di mana untuk mengumpulkan data yang diperlukan tidak menggunakan pedoman wawancara yang telah tersusun secara sistematis dan lengkap. Untuk mendapatkan informasi yang lengkap peneliti melakukan wawancara dengan pihak – pihak yang bersangkutan. Dalam penelitian ini wawancara dilakukan dengan pihak Dispenduk Capil khususnya bagian PIAK dan bidang Pengembangan dan Pengkajian Kependudukan yang berkepentingan terhadap proses pelaporan pendataan kependudukan serta pihak Desa/Kelurahan mengenai proses pengajuan administrasi kependudukan. Wawancara dengan Desa/Kelurahan dilakukan karena Desa/Kelurahan merupakan instansi pemerintahan yang paling dasar dalam menyelenggarakan pengajuan administrasi kependudukan. Melalui tahap wawancara dan observasi tersebut permasalahan yang ditemukanan pada Dispenduk Capil Kabupaten Malang adalah permasalahan terkait pengajuan administrasi kependudukan dan pelaporan pendataan kependudukan yang dilakukan setiap bulannya oleh desa/kelurahan yang selanjutnya dilakukan rekapitulasi oleh Dispenduk Capil Kabupaten Malang guna dilakukan analisa lebih lanjut sesuai dengan Undang – Undang. Permasalahan lain yang ditemukan yaitu mengenai proses pengajuan administrasi kependudukan yang berjalan selama ini memeliki beberapa permasalahan yang membutuhkan solusi. Permasalah tersebut meliputi antrian yang panjang setiap harinya pada Dispenduk Capil Kab Malang, jarak tempuh yang jauh serta mahalnya biaya transportasi bagi masyarakat kabupaten Malang untuk mendapatkan pelayanan. Hal ini dikarenakan wilayah kabupaten Malang yang luas yaitu ± 3.534.86 km^2 . Permasalahan lain yang ada pada proses pengajuan administrasi kependudukan yang terkesan rumit sehingga membutuhkan waktu yang tidak sedikit hanya untuk mengurus pengajuan administrasi kependudukan. Ditambah lagi dengan waktu pengurusan yang hanya bisa dilakukan pada jam dan hari kerja. Dari berbagai permasalahan ini menimbulkan permasalahan lainnya yaitu banyaknya masyarakat yang tidak tertib administrasi kependudukan dan banyaknya calo.

Proses pendataan kependudukan yang berjalan selama ini membutuhkan waktu yang lama serta sulit melakukan perekapan dan analisa laporan dikarenakan data yang telah diberikan oleh camat tersebut tidak dapat langsung direkap dan di analisa melainkan petugas harus terlebih dahulu menginputkan satu persatu data kependudukan yang telah dilaporkan oleh camat ke dalam *Microsoft Excel*. Faktor lain yang menyebabkan proses menjadi lama adalah wilayah Kabupaten Malang yang luas. Dari permasalahan yang ditemukan tersebut selanjutnya dibuat rumusan masalah guna membantu menentukan arah atau fokus dari penelitian yang dilakukan, menentukan jenis data yang diperlukan atau yang relevan dengan penelitian, serta mempermudah untuk menentukan populasi dan sampel penelitian.

3.2 Studi Kepustakaan

Studi kepustakaan adalah serangkaian kegiatan yang berkenaan dengan metode pengumpulan data pustaka, membaca dan mencatat serta mengolah bahan penelitian (Zed, 2004). Studi Kepustakaan ini dilakukan dengan mengumpulkan, membaca dan mencatat informasi yang relevan dengan topik atau permasalahan terkait penelitian yang dilakukan. Informasi – informasi tersebut diperoleh dari buku, jurnal, *ebook*, dan penelitian – penelitian sebelumnya yang sesuai dengan penelitian yang dilakukan. Melalui studi kepustakaan ini peneliti dapat memanfaatkan informasi dan pemikiran – pemikiran yang relevan dengan penelitian yang dilakukan.

3.3 Pengumpulan Data

Pengumpulan data dilakukan untuk memperoleh informasi yang dibutuhkan dalam rangka mencapai tujuan pengembangan sistem. Teknik pengumpulan data yang digunakan peneliti pada penelitian ini adalah teknik wawancara. Wawancara dilakukan dengan menggunakan metode wawancara tidak terstruktur di mana peneliti tidak menggunakan pedoman wawancara yang telah tersusun secara sistematis dan lengkap. Pedoman yang digunakan dalam metode ini adalah dengan menanyakan berupa garis-garis besar permasalahan. Dalam pengumpulan data ini wawancara dilakukan dengan *face to face.* Wawancara dilakukan dengan pihak Dispenduk Capil dan pihak Desa/Kelurahan. Teknik wawancara dipilih karena melalui wawancara masalah akan langsung mengenai sasaran serta penegasan pertanyaan dapat langsung diutarakan. Selain itu data dan informasi akan diperoleh secara langsung dan cepat. Pengumpulan data ini akan menghasilkan sebuah informasi terkait kebutuhan pengguna, keinginan pengguna serta kondisi lingkungan bisnis sebelum adanya sistem.

3.4 Metode *Rational Unified Process* (RUP)

Pada tahap ini dilakukan pengembangan sistem informasi dengan metode yang telah dipilih yaitu metode *Rational Unified Process* (RUP). Metode RUP memiliki 4 fase yang terdiri dari beberapa aspek dinamis. Fase tersebut terdiri dari fase *inception*, fase *elaboration*, fase *contruction*, dann fase *transition*.

3.4.1 Inception

Fase *inception* merupakan tahap yang bertujuan untuk menentukan ruang lingkup dan *business case*. Proses yang dilakukan pada tahap ini adalah analisa proses bisnis yang telah berjalan di dispenduk capil kabupaten Malang. Proses bisnis tersebut digambarkan dengan menggunakan metode *business process model and notation* (BPMN). Pemilihan pemodelan berdasarkan BPMN ini didasari akan kelebihan dari BPMN dalam mengakomodir penyajian kebutuhan bisnis menjadi model proses bisnis yang dapat dengan mudah dijelaskan dan ditransformasikan. Pemodelan proses bisnis meliputi proses bisnis

as-is dan proses bisnis *to-be*. Pemodelan proses bisnis *as-is* bertujuan untuk menggambarkan proses bisnis yang telah ada dan yang akan dikembangkan. Sedangkan proses bisnis *to-be* bertujuan untuk menggambarkan peningkatan dan perubahan terhadap proses bisnis saat ini atau *as-is*. Pemodelan proses bisnis *to-be* ini dilakukan agar proses bisnis yang berjalan nantinya sesuai dengan tujuan sistem yang dibuat. Proses bisnis *to-be* dimodelkan berdasarkan pada permasalahan yang ada pada proses bisnis *as-is*.

3.4.2 Elaboration

Fase *elaborasi* merupakan fase kedua pada metode RUP. Pada fase *elaboration* ini dilakukan analisa kebutuhan dan perancangan arsitektur sesuai dengan identifikasi kebutuhan yang telah dilakukan pada tahap *inception.* Jadi pada fase *elaboration* ini terdapat dua tahap yaitu analisa kebutuhan dan perancangan. Berikut adalah uraian dari fase *elaboration* :

a. Analisa kebutuhan

   Analisa kebutuhan merupakan proses pengumpulan informasi tentang kebutuhan – kebutuhan pengguna terhadap sistem yang akan dikembangkan. Tahap pertama yang dilakukan pada analisa kebutuhan ini adalah melakukan identifikasi pengguna (aktor) yang akan menggunakan sistem. Indentifikasi aktor ini bertujuan untuk mengetahui karakter dari masing – masing aktor. Setelah itu dilakukan penggalian kebutuhan lebih lanjut yang dilakukan dengan wawancara kepada pihak dispenduk capil. Informasi dari wawancara yang dilakukan ini akan menghasilkan kebutuhan fungsional dan kebutuhan non fungsional dari sistem yang akan dikembangkan. Kebutuhan fungsional merupakan kebutuhan yang harus ada pada sistem. Sedangkan kebutuhan non fungsional merupakan batasan layanan atau fungsi yang ditawarkan sistem seperti kemudahan penggunaan sistem dan keamanan sistem. Kebutuhan – kebutuhan ini nantinya akan digunakan sebagai acuan untuk mengetahui fitur apa saja yang akan ada pada sistem.

b. Perancangan

   Setelah analisi kebutuhan dilakukan selanjutnya dilakukan perancangan arsitektur dari sistem yang akan dikembangkan. Pada tahap perancangan ini akan dilakukan pembuatan *use case diagram, class diagram, activity diagram* dan *sequence diagram*. *Use case diagram* menggambarkan interaksi antara pengguna (aktor) dengan sistem. *Use case diagram* ini akan didukung dengan *use case scenario* untuk memperjelas interaksi yang dilakukan oleh pengguna (aktor) dengan sistem. *Class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas –kelas yang akan dibuat untuk membangun sistem. *Activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sistem. Sedangkan *sequence diagram* menggambarkan kolaborasi dinamis antar sejumlah *object*. Selanjutnya adalah perancangan database

yang digambarkan dengan menggunakan *physical data model*. Pada tahap perancangan ini juga dilakukan perancangan terhadap *user interface* atau tampilan sistem yang nantinya akan digunakan oleh pengguna serta menentukan tata letak dari semua fitur sistem informasi pelaporan pendataan kependudukan kabupaten malang. Setelah proses perancangan selesai dilakukan iterasi untuk memvalidasi kebutuhan sistem sebelum ke tahap implementasi. Validasi ini dilakukan untuk mengantisipasi apabila ada perubahan kebutuhan baik penambahan maupun pengurangan dari pihak *stakeholder*. Kebutuhan yang dimaksud meliputi kebutuhan fungsional dan kebutuhan non fungsional.

3.4.3 Construction

Pada fase *construction* ini dilakukan implementasi dari hasil perancangan sistem yang telah dilakukan pada fase *elaboration* serta pengujian sistem. Proses *coding* sebagian besar dilakukan pada fase ini. Selain itu pada fase *contruction* ini juga dilakukan pemeriksaan ulang terhadap hasil perancangan yang telah dilakukan sebelumnya agar desain yang telah dibuat sesuai dengan analisis sistem yang akan dikembangkan.

a. Implementasi

   Untuk dapat dimengerti oleh mesin, dalam hal ini adalah komputer, maka perancangan yang telah dibuat pada fase *elaboration* harus diubah bentuknya menjadi bentuk yang dapat dimengerti oleh mesin, yaitu ke dalam bahasa pemrograman melalui proses *coding*. Implementasi ini dilakukan dengan menggunakan metode *object oriented programming* (oop) dan menggunakan bahasa pemrograman php serta menggunakan *framework codeigniter* (CI) untuk memudahkan dalam menerapkan konsep MVC (*model, view dan controller*). Untuk sistem manajemen basis data atau *database* menggunakan MySql.

b. Pengujian

   Pengujian yang dilakukan pada fase *contruction* ini adalah pengujian *validation testing,* dan *campability*. Pengujian *validation testing* menunjukkan bahwa sistem telah memenuhi semua persyaratan fungsional. Pengujian ini hanya dilakukan pada tampilan luarnya dan fungsionalitasnya tanpa mengetahui apa yang sesungguhnya terjadi dalam proses detailnya (hanya mengetahui input dan output). *Validation testing* ini dilakukan dengan mengamati hasil eksekusi melalui data uji dan memeriksa fungsional dari sistem. Uji kasus dibangun pada spesifikasi dan persyaratan sistem, yakni apa yang seharusnya dilakukan oleh sistem. Pada pengujian *validation testing* ini alur *use case* digunakan sebagai panduan pengujian. Untuk mendeteksi kemungkinan kesalahan pada sistem menggunakan *use case scenario*.

   Campability testing adalah pengujian yang digunakan untuk memeriksa apakah sistem dapat berjalan pada *hardware*, sistem operasi,

aplikasi ataupun lingkungan jaringan yang berbeda . Pengujian kampabilitas ini dipilih karena ada berbagai macam perangkat yang digunakan untuk menjalankan sistem informasi berbasis *website* ini. *Compatibility testing* dilakukan dengan menggunakan suatu aplikasi bernama SortSite. Melalui *SortSite* ini nantinya akan diketahui tingkat kompabilitas sistem terhadap berbagai jenis *browser* dan berbagai jenis sitem operasi. Semakin aplikasi dapat berjalan di banyak perangkat yang berbeda, maka semakin baik aspek kompabilitasnya. Pilihan pencetakan juga termasuk pada *compability testing* ini. Pengujian ini dilakukan untuk memastikan bahwa *font*, perataan halaman, grafik ukuran dan lain – lain dicetak dengan benar.

3.4.4 Transition

Transition merupaka fase terakhir dari metode RUP. Pada fase ini ada dua aktivitas yang dilakukan yaitu deploy sistem dan pengujian dengan menggunakan *usability testing*. *Usability testing* dilakukan untuk mengetahui seberapa mudah sistem dapat digunakan oleh pengguna. Pengujian *usability testing* dilakukan dengan menggunakan *System Usability Scala* (SUS). SUS merupakan skala *likert* dengan sepuluh pernyataan yang dapat mendeskripsikan pandangan global tentang penilaian subyektivitas kegunaan. Skor yang dihasilkan pada perhitungan SUS ini mewakili ukuran gabungan dari kegunaan keseluruhan dari sistem.

3.5 Kesimpulan dan Saran

Pengambilan kesimpulan dilakukan setelah semua tahap pengembangan sistem informasi pelaporan pendataan kependudukan kabupaten malang selesai dilakukan. Kesimpulan yang diambil disesuaikan dengan rumusan masalah yang telah dijelaskan sebelumnya. Saran dituliskan untuk dapat melakukan perbaikan atau pengembangan terhadap sistem yang telah dikembangkan apabila terdapat kesalahan atau kekurangan. Penulisan kesimpulan dan saran ini diharapkan dapat menjadi bahan evaluasi untuk penelitian pengembangan sistem selanjutnya.

BAB 4 ANALISIS PERSYARATAN

Pada bab ini membahas tentang analisis persyaratan yang dilakukan untuk sistem berdasarkan pada hasil wawancara yang dilakukan dengan pihak Dispenduk Capil Kabupaten Malang. Berdasarkan fase yang ada dalam RUP, bab ini merupakan fase *inception*. Keluaran yang dihasilkan meliputi model proses bisnis *as-is,* model proses bisnis *to-be*, identifikasi tipe pemangku kepentingan, analisis masalah, identifikasi kebutuhan pemangku kepentingan dan pengguna, identifikasi fitur, persyaratan fungsional dan non fungsional,*use case* dan *activity diagram*.

4.1 Pemodelan Proses Bisnis

Pemodelan proses bisnis bertujuan untuk memetakan proses bisnis yang ada di Dispenduk Capil dan menemukan cara untuk memperbaikinya. Pemodelan proses bisnis yang akan dilakukan akan menghasilkan proses bisnis *as-is* dan proses bisnis *to-be*. Dalam memodelakan proses bisnis ini data diperoleh dari hasil wawancara dengan pegawai Dispenduk Capil Kabupaten Malang.

4.1.1 Pemodelan Proses Bisnis *As-is*

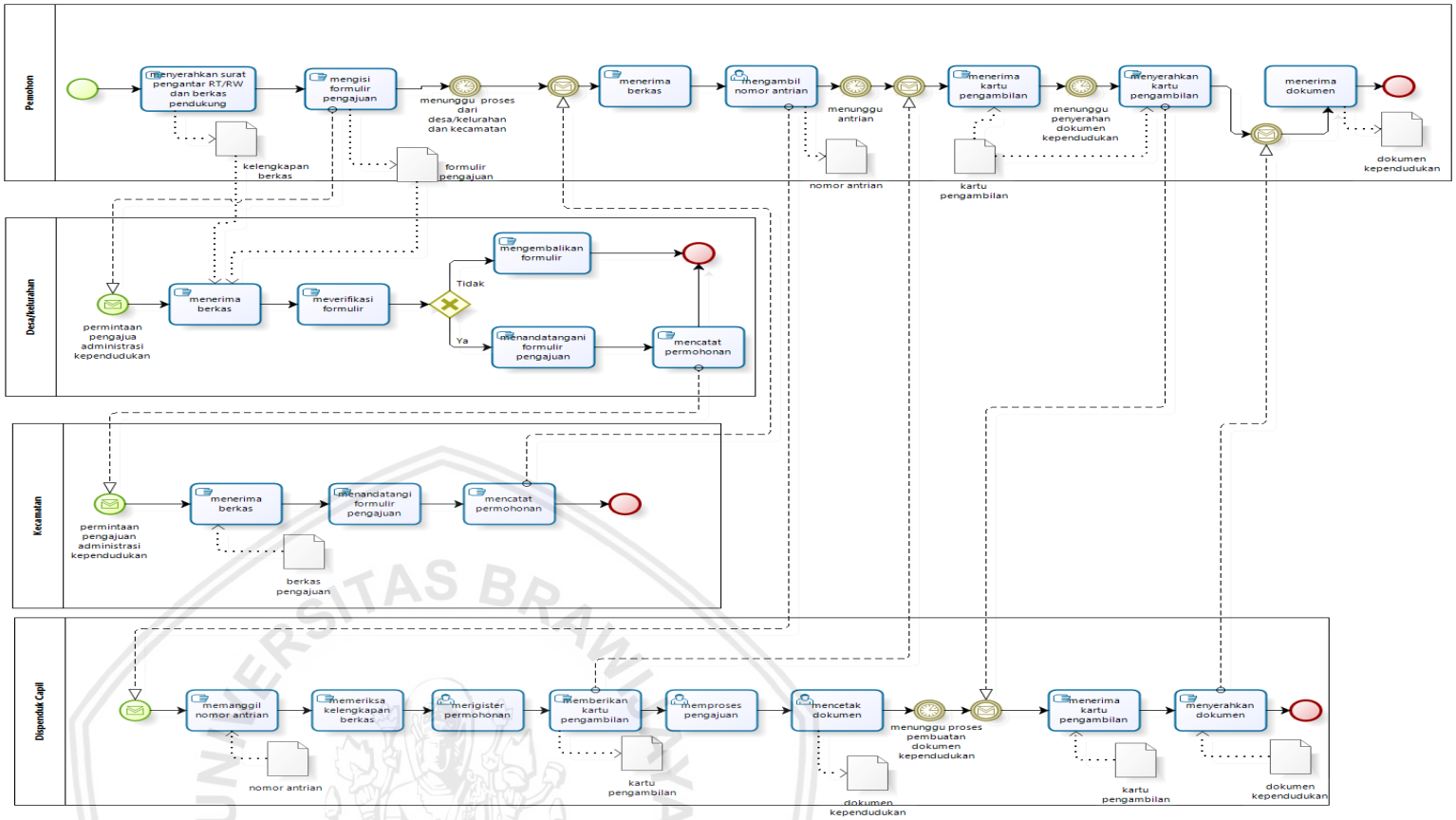
Pada penelitian ini proses bisnis *as-is* diidentifikasi berdasarkan keadaan proses bisnis yang berjalan saat ini pada dispenduk capil. Analisis proses bisnis *as-is* ini diperoleh dari hasil wawancara dengan pegawai dispenduk capil dan berdasarkan hasil observasi di dispenduk capil. Analisis dan pemodelan proses bisnis *as-is* dilakukan agar dapat membantu mengidentifikasi kemungkinan perbaikan yang dapat dilakukan. Analisa proses bisnis *as-is* dimodelkan dalam bentuk notasi BPMN agar dapat membantu pemangku kepentingan dalam memahami hasil analisa proses bisnis *as-is*.

4.1.1.1 Pemodelan Proses Bisnis *As-is* Pengajuan Dokumen Kependudukan

Gambar 4.1 merupakan pemodelan proses bisnis pengajuan dokumen kependudukan yang berjalan saat ini. Proses bisnis ini dimulai dari pemohon yang mengajukan permohonan dengan menyerahkan berkas persyaratan kepada desa/kelurahan tempat pemohon tinggal. Berkas persyaratan ini meliputi formulir permohonan dan dokumen-dokumen pendukung lainnya. Di desa/kelurahan ini berkas yang telah dipenuhi oleh pemohon akan diverifikasi terlebih dahulu sebelum ditadatangani oleh kepala desa/lurah. Setelah formulir permohonan ditandatangani selanjutnya petugas desa/kelurahan akan mencatat dalam buku harian peristiwa kependudukan dan peristiwa penting. Kemudian pemohon harus membawa formulir permohon yang telah berisi tanda tangan kepala desa/lurah tersebut kepada kecamatan untuk mendapatkan tanda tangan camat. Setelah semua berkas terpenuhi pemohon harus ke dispenduk capil kabupaten malang untuk memproses pengajauan lebih lanjut. Pada dispenduk capil ini pemohon akan dicek kelangkapan berkasnya sebelum mendapatkan

kartu pengambilan yang harus dibawa ketika hendak mengambil dokumen yang diajukan. Pada kartu pengambilan ini terdapat tanggal masuknya berkas pengajuan dan tanggal pengambilan berkas.

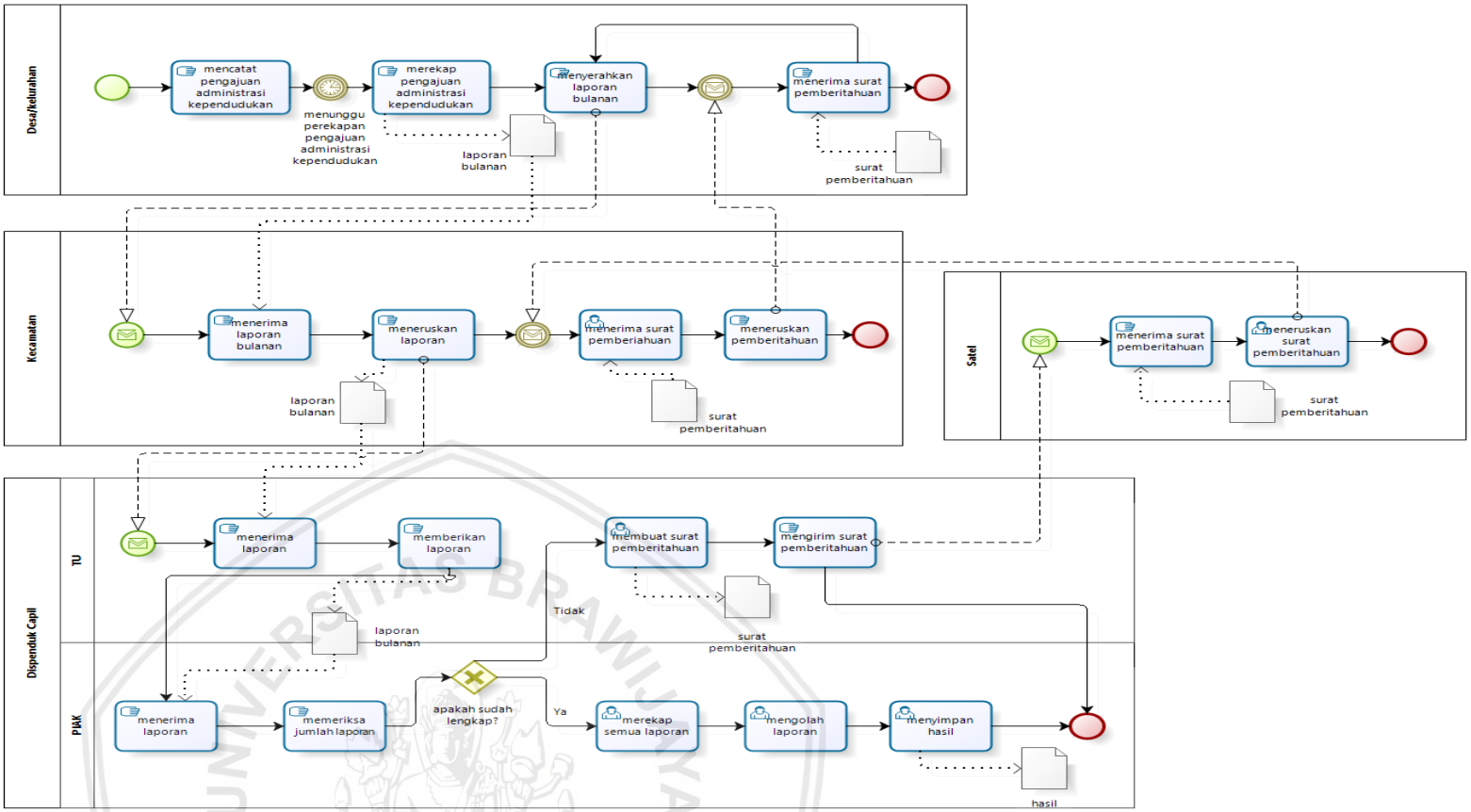
Sebelum mengeluarkan kartu pengambilan ada beberapa aktivitas yang dilakukan oleh petugas Dispenduk Capil. Proses pertama yang dilakukan adalah memanggil pemohon sesuai dengan nomor antrian yang sebelumnya telah diambil oleh pemohon melalui mesin antrian. Selanjutya petugas Dispenduk Capil memeriksa kelengkapan berkas dan meregister data permohonan kedalam database. Setelah data berhasil diregister, petugas dipenduk capil akan memberikan kartu permohonan dan memproses pengajuan. Petugas Dispenduk capil yang bertugas memproses pengajuan dibagi dua yaitu petugas kependudukan dan catatan sipil. Petugas kependudukan ini malayani permohon kk, ktp, dan surat pindah. Sedangkan petugas catatan sipil melayani permohonan akta kelahiran, akta kematian, akta perkawinan, akta perceraian, akta pengakuan anak, akta pengesahan anak, akta pengangkatan anak, pembatalan akta dan akta perubahan nama.



Gambar 4.1 Proses Bisnis *as-is* Pengajuan Dokumen Kependudukan

4.1.1.2 Pemodelan Proses Bisnis *As-is* Pelaporan Pendataan Kependudukan

Pemodelan proses bisnis *as-is* pelaporan pendataan kependudukan ditujukkan dengan gambar 4.2. dari gambar tersebut dapat diketahui bahwa proses bisnis pelaporan dimulai dari proses pelaporan yang dilakukan oleh desa/kelurahan yang ada di kabupaten malang yang selanjutnya laporan tersebut diberikan kepada kecamatan setempat untuk selanjutkan diteruskan kepada Dispenduk Capil Kabupaten Malang. Laporan yang telah diterima oleh Dispenduk Capil selanjutnya diolah. Sebelum laporan tersebut diolah Dispenduk Capil terlebih dahulu mengecek laporan yang telah masuk apabila ada kecamatan yang belum melaporkan dan telah melewati batas waktu yakni per-tanggal 10 maka pihak Dispenduk Capil akan membuat surat pemberitahuan. Surat pemberitahuan ini tidak diserahkan secara langsung kepada kecamatan melainkan diserahkan atau ditujukan kepada pihak satel yang ada di pemerintah kabupatan malang. Dari satel ini surat pemberitahuan akan diteruskan kepada kecamatan yang bersangkutan melalui *phonogram*. Hal ini dilakukan dikarenakan butuh waktu yang lama untuk memberikan surat pemberitahuan itu secara langsung kepada kecamatan, mengingat wilayah kecamatan yang saling berjauhan. Sedangkan apabila surat pemberitahuan disampaikan melalui pihak satel surat tersebut akan lebih cepat karena di satel ada orang yang berjaga selama 24 jam dan surat pemberitahuan tersebut akan langsung dikirim melalui phonogram.



Gambar 4.2 Proses Bisni *as-is* Pelaporan Pendataan Kependudukan

4.1.2 Pemodelan Proses Bisnis *To –be*

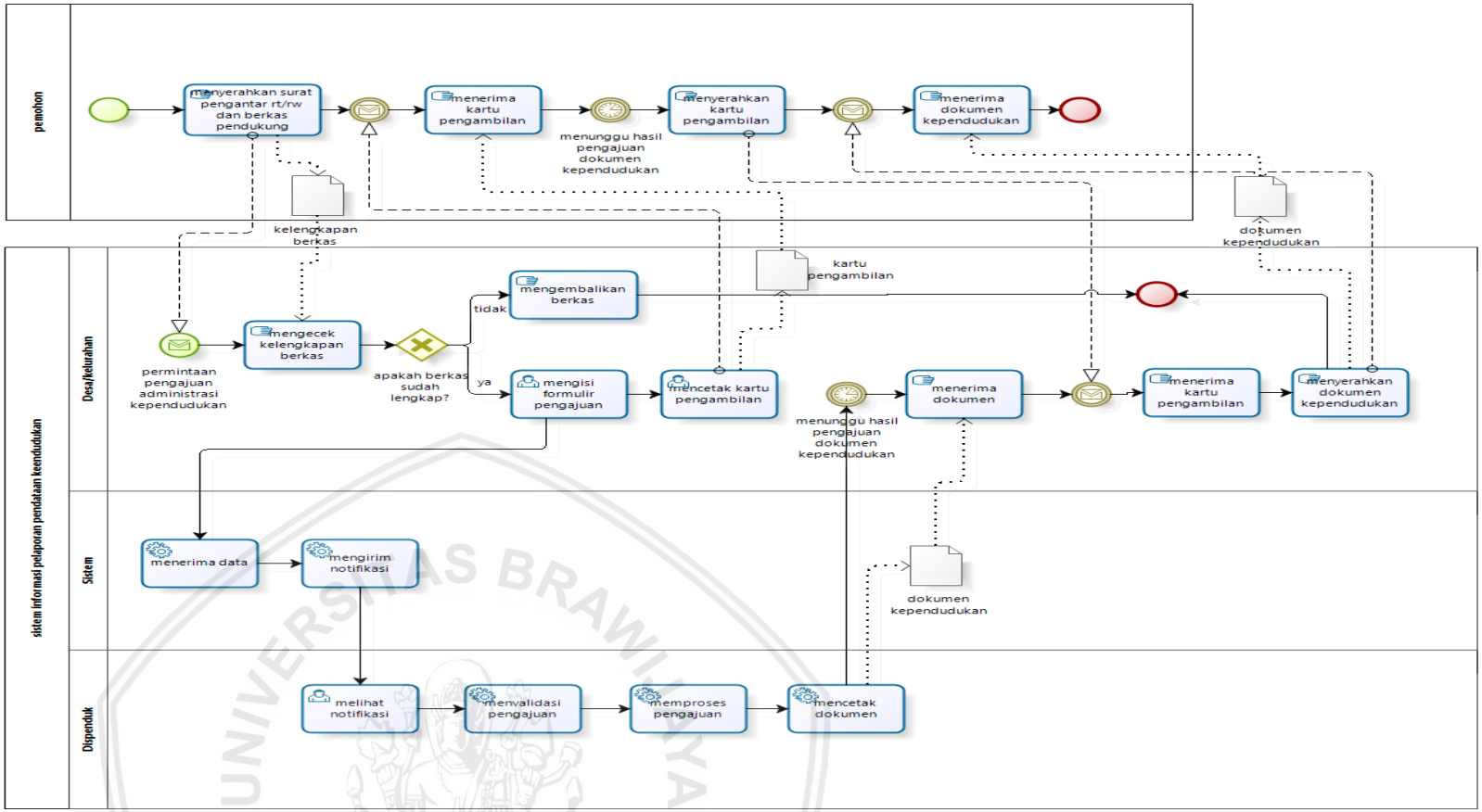
Pemodelan proses bisnis *to – be* merupakan proses bisnis usulan yang ditawarkan kepada pemangku kepentingan. Pemodelan proses bisnis *to – be* ini diidentifikasi berdasarkan hasil analisis terhadap proses bisnis as-is. Pemodelan proses bisnis *to-be* dilakukan untuk menggambarkan perubahan atau tambahan yang diusulkan. Hasil dari pemodelan proses bisnis *to-be* ini digunakan sebagai masukan dalam kegiatan analisis persyaratan.

4.1.2.1 Pemodelan Proses Bisni *to-be* Pengajuan Administrasi Kependudukan

Pemodelan proses bisnis *to-be* pengajuan administrasi kependudukan ditujukkan pada gambar 4.4. Melalui gambar tersebut dapat diketahui bahwa pada proses bisnis usulan pengajuan tidak lagi dilakukan secara langsung antara pemohon dan dispenduk capil, akan tetapi pengajuan dilakukan secara online melalui desa/kelurahan. Jadi, pemohon tidak perlu lagi datang ke dispenduk capil dan mengantri disana. Pemohon cukup menyerahkan semua berkas sesuai dengan persyaratan kepada pihak desa/kelurahan. Selanjutnya pihak desa/kelurahan akan memasukkan pengajuan tersebut ke dalam sistem dimana petugas dispenduk capil dapat langsung mengetahui adanya pengajuan administrasi kependudukan. Pada proses bisnis usulan ini proses penyimpanan data juga dilakukan secara terpusat. Hal ini dilakukan supaya pihak dispenduk capil dapat langsung mengetahui apabila ada pengajuan administrasi kependudukan. Berikut adalah tabel perubahan aktivitas yang ada pada proses bisnis pengajuan administrasi kependudukan.

Tabel 4.1 Perubahan Aktivitas Pengajuan Administrasi Kependudukan

Unit bisnis	Proses bisnis as-is	Proses bisnis to-be	Keterangan
Pemohon	Mengambil nomor antrian	-	Dieleminasi
Pemohon	Menerima kartu pengambilan	Menerima kartu pengambilan	Diubah
Pemohon	Mengisi formulir pengajuan	Mengisi formulir pengajuan	Diubah
Dispenduk capil	Memanggil nomor antrian	-	Dieleminasi
Desa/kelurahan	Mencatat permohonan	Mencatat permohonan	Diubah



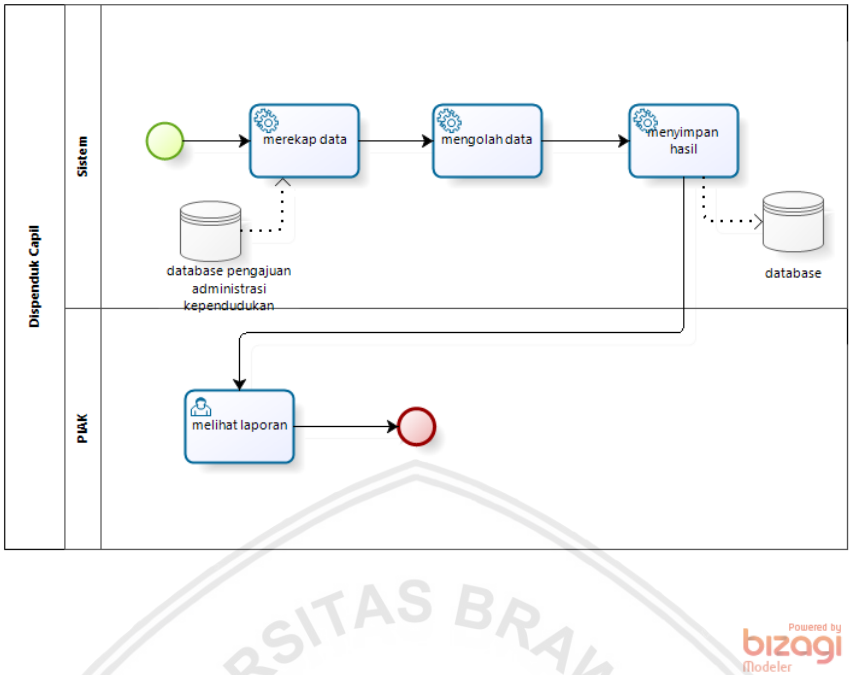
Gambar 4.3 Proses Bisnis *to-be* Pengajuan Administrasi Kependudukan

4.1.2.2 Pemodelan Proses Bisnis *to-be* Pelaporan Pendataan Kependudukan

Pemodelan proses bisnis *to-be* pelporan pendataan kependudukan ditunjukkan pada Gambar 4.4. Dari gambar tersebut dapat diketahui bahwa pada proses bisnis usulan untuk penyimpanan laporan data kependudukan yang dilakukan oleh desa/kelurahan disimpan secara terpusat. Penggunaan penyimpanan terpusat ini berguna untuk proses pencatatan dan pengiriman laporan yang dilakukan oleh desa/kelurahan serta untuk proses pengolahan laporan oleh Dispenduk Capil. Melalui penyimpanan terpusat ini juga pihak dispenduk capil tidak perlu lagi membuat surat pemberitahuan untuk kecamatan yang tidak mengumpulkan laporan sampai waktu yang ditentukan. Hal ini dikarenakan data akan langsung tersimpan ke dalam database ketika pihak desa/kelurahan memasukkan data dan pihak dispenduk capil dapat langsung melihat data tersebut tanpa menunggu waktu pelaporan. Data yang akan diolah pada sistem berasal dari data pengajuan administrasi kependudukan yang telah dimasukkan oleh petugas Desa/Kelurahan dan telah tersimpan pada *database* sistem. Berikut adalah tabel perubahan aktivitas yang ada pada proses bisnis pelaporan pendataan data kependudukan.

Tabel 4.2 Perubahan Aktivitas Proses Bisnis Pelporan Pendataan Data Kependudukan

Unit bisnis	Proses bisnis as-is	Proses bisnis to-be	Keterangan
Desa/kelurahan	Memberikan laporan data kependudukan	-	Diubah
Kecamatan	menerima laporan data kependudukan	-	Diubah
Kecamatan	Memberikan laporan data kependudukan	-	Diubah
Dispenduk capil	Mengolah data kependudukan	-	Diubah
Dispenduk capil	Membuat surat pemberitahuan	-	Dieleminasi
Kecamatan	Menerima surat pemberitahuan	-	Dieleminasi
Desa/kelurahan	Menerima surat pemberitahuan	-	Dieleminasi



Gambar 4.4 Proses Bisnis *to-be* Pelaporan Data Kependudukan

4.2 Analisa Kebutuhan

Analisa kebutuhan berfungsi bertujuan untuk menentukan apa yang dapat dilakukan oleh sistem dan harus memenuhi tujuan dari sistem tersebut. Kebutuhan sistem dibagi menjadi dua yaitu kebutuhan fungsional dan kebutunan non fungsional. Kebutuhan fungsional adalah aktivitas dan *service* yang harus disediakan oleh sistem yang akan dikembangkan. Sedangkan kebutuhan non fungsional adalah batasan dari sistem atau fitur yang ditawarkan oleh sistem.

4.2.1 Identifikasi Tipe Pemangku Kepentingan

Identifikasi pemangku kepentingan digunakan untuk mengetahui siapa saja pemangku kepentingan dari sebuah sistem. Pemangku kepentingan yang telah diidentifikasi selanjutnya dikelompokkan berdasarkan karakteristik dan hubungan pemangku kepentingan dengan sistem yang sedang dikerjakan. Identifikasi pemangku kepentingan diperoleh dari hasil analisis wawancara yang dilakukan dengan pihak Dispenduk capil. Hasil dari identifikasi tipe pemangku kepentingan ditujukkan oleh tabel 4.3. Hasil ini nantinya akan berguna sebagai informasi dalam analisis masalah.

Tabel 4. 3 Identifikasi Tipe Pemangku Kepentingan

Tipe Pemangku Kepentingan	Deskripsi	Pemangku Kepentingan
Pengguna	Seseorang yang	Desa/kelurahan,

Tipe Pemangku Kepentingan	Deskripsi	Pemangku Kepentingan
	berinteraksi secara langsung dengan sistem dan berperan sebagai actor pada use case.	Kecamatan, Dispenduk Capil Kabupaten Malang
Pihak yang berwenang	Pihak yang berwenang merupakan organisasi atau individu yang memberikan informasi mengenai regulasi sehingga sistem informasi yang akan dikembangkan tidak menyimpang dari aturan yang berlaku.	Dispenduk Capil Kabupaten Malang
Pengembang	Organisasi atau individu yang mengembangkan sistem informasi.	Peneliti
Pelanggan	Pelanggan adalah mereka yang akan mendapatkan manfaat dari pengembangan sistem informasi ini baik organisasi maupun individu.	Desa/kelurahan, Kecamatan, Dispenduk Capil Kabupaten Malang dan masyarakat yang mengajukan administrasi kependudukan.

4.2.2 Analisis Masalah

Analisis masalah dilakukan untuk mengetahui permasalahan yang akan diselesaikan dan untuk mengusulkan solusi yang tepat sesuai kebutuhan pengguna. Dalam penelitian ini, analisis masalah diperoleh dari hasil wawancara yang telah dilakukan dengan pemangku kepentingan. Hasil dari analisis masalah didokumentasikan dalam tabel *problem statement.* Berikut adalah penjelasansingkat mengenai *problem statement.*

Tabel 4.4 *Problem Statement Template*

The problem of	Mendeskripsiskan secara singkat mengenai permasalahan yang ada
Affects	Pemangku kepentingan yang dipengaruhi oleh masalah
The impact of which is	Apa dampak dari masalah yang ada?
A successful solution would	Daftar beberapa manfaat utama dari solusi yang sukses.

Berikut adalah *problem statement* yang diperoleh dari hasil wawancara dengan pihak dispenduk capil. Dari hasil wawancara yang dilakukan ditemukan

beberapa permasalah terkait proses pengajuan administrasi kependudukan. Hasil analisis masalah ini nantinya akan digunakan sebagai informasi dalam mengidentifikasi daftar kebutuhan pemangku kepentingan.

Tabel 4.5 *Problem Statement* 1

The problem of	1. Proses pengajuan administrasi kependudukan membutuhkan waktu yang lama. 2. Proses antrian yang panjang di Dispenduk Capil. 3. Luasnya wilayah kabupaten malang membuat lokasi dispenduk capil memiliki jarak yang jauh dari lokasi tertentu.
Affects	Masyarakat, Dispenduk Capil
The impact of which is	1. Banyaknya masayarakat yang tidak tertib administrasi kependudukan. 2. Banyaknya waktu dan biaya yang harus dihabiskan oleh masyarakat.
A successful solution would	Sistem yang menyediakan layanan pengajuan administrasi kependudukan secara online sehingga dapat mempermudahkan masyarakat dalam melakukan pengajuan administrasi kependudukan.

Problem statement yang berikutnya adalah terkait pelaporan administrasi kependudukan. Analisis masalah ini diperoleh melalui hasil dari wawancara dengan pihak dispenduk capil, desa/kelurahan dan observasi. Dari hasil analisis yang ditemukan ini nantinya akan digunakan sebagai informasi dalam mengidentifikasi daftar kebutuhan pemangku kepentingan.

Tabel 4.6 *Problem Statement* 2

The problem of	1. Desa/kelurahan memiliki format laporan data yang berbeda-beda. 2. Kecamatan harus mengumpulkan laporan sekecamatan sebelum menyerahkan laporan kepada Dispenduk Capil. 3. Dispenduk Capil harus memasukkan data laporan dari desa/kelurahan satu persatu ke *Microsoft excel* secara manual. 4. Seringnya terjadi salah ketik ketika memasukkan data. 5. Proses memasukkan data satu persatu yang dilakukan oleh Dispenduk Capil membuat proses pengolahan laporan menjadi lama.
Affects	Desa/kelurahan, Kecamatan, Dispeduk Capil
The impact of which	1. Proses pengolahan laporan yang dilakukan oleh

is	Dispenduk menjadi sulit dan lama. 2. Data yang dilaporkan oleh desa/kelurahan menjadi tidak valid dan tidak baku
A successful solution would	Sistem yang menyediakan pengolahan laporan data kependudukan untuk mempermudah proses pelaporan dan pengolahan data.

4.2.3 Perbandingan Proses Bisnis

Pada proses bisnis *as-is* dan *to-be* yang telah dimodelkan dilakukan simulasi pada level *time* analisis. Hal ini dilakukan untuk mengatahui perbedaan waktu yang dihasilkan. Dari perbedaan waktu yang dihasilkan ini selanjutnya akan dilakukan perbandingan proses bisnis untuk mengetahui apakah proses bisnis *to-be* yang telah dimodelkan memberikan dampak yang signifikan terhadap proses bisnis *as-is.*

Pada proses bisnis pengajuan administrasi kependudukan *as-is* estimasi waktu simulasi proses bisnis masing-masing aktivitas yaitu 10 menit untuk menyerahkan surat pengantar RT/RW dan berkas pendukung ke desa/kelurahan, 5 menit untuk mengisi formulir pengajuan, 5 menit untuk memverifikasi formulir, 2 menit untuk menandatangi formulir, 2 menit untuk mencatat permohonan, 40 menit untuk menunggu proses dari desa/kelurahan dan kecamatan, 1 jam untuk mengambill nomor antrian dengan estimasi perjalanan ke dispenduk capil, 1 menit untuk memanggil nomor antrian, 5 menit untuk memeriksa kelengkapan berkas, 2 menit untuk meregister permohonan, 2 menit untuk memberikan kartu pengambilan, 1 hari untuk memproses pengajuan, 15 menit untuk mencetak dokumen, 1 hari untuk menandatangi dokumen, 2 jam untuk menunggu antrian untuk menyerahkan dokumen kependudukan ke petugas dispenduk capil, 5 hari untuk menunggu penyerahan dokumen kependudukan.

Perubahan aktivitas pada pengajuan administrasi dokumen kependudukan setelah dilakukan simulasi time analisis terjadi perubahan waktu. Hal ini terjadi karena adanya eliminasi aktivitas. Aktivitas-aktivitas tersebut meliputi waktu untuk menunggu proses dari desa/kelurahan, waktu untuk mengambil nomor antrian ke dispenduk capil, waktu untuk memanggil nomor antrian, waktu menunggu antrian di dispenduk capil.

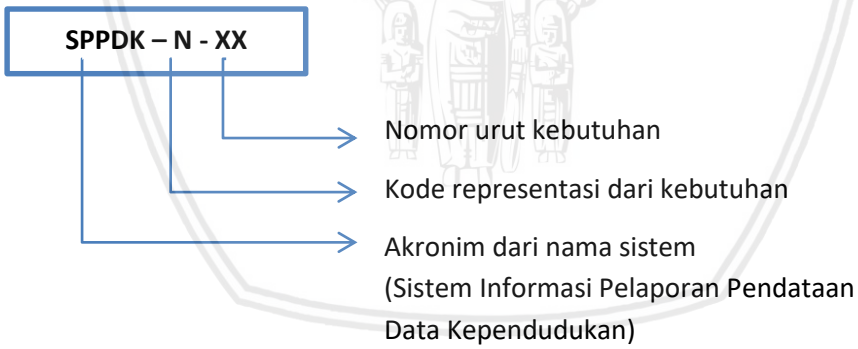
Pada proses bisnis pelaporan pendataan kependudukan as-is waktu simulasi proses bisnis masing-masing aktivitas yaitu 30 hari menunggu proses perekapan pengajuan administrasi kependudukan oleh desa/kelurahanan, 10 menit menyerahkan laporan bulanan, 1 jam untuk meneruskan laporan ke dispenduk capil, 5 menit untuk memeriksa jumlah laporan, 3 hari untuk mengolah laporan.

Sama halnya dengan pengajuan administrasi kependudukan, proses bisnis pelaporan pendataan data kependuudukan mengalami terjadi perubahan waktu karena adanya eliminasi beberapa aktivitas. Aktivitas tersebut yaitu aktivitas merekap pengajuan administrasi kependudukan, menyerahkan laporan ke

kecamatan, memeriksa jumlah laporan, membuat dan mengirim surat pemberitahuan. Untuk merekap pengajuan dilakukan secara otomatis sehingga waktu yang dibutuhkan menjadi lebih singkat. Dimana sebelumnya harus menunggu 30 hari dan membutuhkan waktu 2 jam pada proses bisnis to-be berubah menjadi 5 menit tanpa ada proses menunggu selama 30 hari. Proses mengelolah laporan juga dilakukan secara otomatis. Waktu yang dibutuhkan juga lebih singkat dimana pada proses bisnis as-is membutuhkan waktu 3 hari sedangkan pada proses bisnis to-be membutuhkan waktu 5 menit.

4.2.4 Identifikasi Kebutuhan Pemangku Kepentingan dan Pengguna

Identifikasi kebutuhan pemangku kepentingan dan pengguna berguna untuk memperoleh informasi mengenai kebutuhan pengguna yang berisi pernyataan-pernyataan yang berhubungan dengan masalah yang telah diidentifikasi pada tahap analisis masalah. Pada penelitian ini, identifikasi kebutuhan pemangku kepentingan dan pengguna dilakukan melalui wawancara dengan pegawai Dispenduk Capil yang berwenang serta melalui analisis terhadap *problem statement*. Hasil dari identifikasi kebutuhan pengguna ini didokumentasikan pada tabel 4.3. Pada tabel 4.3 berisi informasi kode kebutuhan, pernyataan kebutuhan pengguna, pemangku kepentingan, situasi saat ini yang sedang berjalan dan solusi yang ditwarkan untuk memenuhi kebutuhan yang diidentifikasi. Penulisan kode kebutuhan pengguna pada tahap ini mengikuti kode penomoran sesuai gambar 4.3.



Gambar 4.5 Aturan Penomoran Kebutuhan Pengguna

Tabel 4. 7 Identifikasi Kebutuhan Pemangku Kepentingan Dan Pengguna

Kode kebutuhan	Kebutuhan pengguna	Pemangku kepentingan	Situasi saat ini	Solusi yang ditawarkan
SPPDK-N-01	Sistem menyediakan layanan untuk	pengguna	Pengajuan dilakukan dengan mendatangi instansi terkait	Sistem informasi yang menyediakan layanan

	memperm udah pengajuan administra si kependudu kan		satu persatu mulai dari desa/kelurahan, kecamatan dan dispenduk capil.	pengajuan administrasi secara *online.*
SPPDK-N-02	Sistem dapat memperm udah pembuatan laporan bulanan	Desa/kelurah an	Pembuatan laporan dilakukan secara manual dengan merekap pengajuan administrasi kependudukan yang ada pada buku register dengan bantuan *Microsoft excel*.	Sistem informasi dapat menyajikan laporan secara berkala.
SPPDK-N-03	Sistem menyediak an layanan untuk mencatat pengajuan administra si kependudu kan	Desa/kelurah an	Pencatatan pengajuan administrasi kependudukan masih dilakukan secara manual dengan menuliskannya pada buku.	Sistem dapat melakukan pencatatan secara otomatis berdasarkan data pengajuan administrasi kependudukan yang telah diinputkan pada sistem.
SPPDK-N-04	Sistem dapat memperm udah untuk mengolah hasil laporan bulanan dari desa/kelur ahan	Dispenduk Capil	Pengolahan laporan bulanan dari desa/kelurahan dilakukan dengan memasukkan satu persatu data yang ada pada laporan bulanan ke dalam *Microsoft Excel*.	Sistem informasi dapat melakukan perekapan laporan bulanan dan menampilkan hasilnya.
SPPDK-N-05	Informasi di dalam sistem	Pengguna	Tidak ada	Sistem informasi menyediakan

	harus dapat diakses sesuai dengan identitas pengguna			layanan untuk membatasi hak akses pengguna terhadap informasi yang dapat diakses.

4.2.5 Identifkasi Pengguna

Identifikasi pengguna adalah tahap menganalisa pengguna / aktor yang berinteraksi dengan sistem. Pada tahap ini peneliti mengacu pada tahap identifikasi kebutuhan pengguna yang telah dilakukan sebelumnya. Yang mana kebutuhan pengguna sebelumnya menghasilkan kebutuhan sistem secara umum. Informasi mengenai identifikasi pengguna ini berguna untuk identifikasi aktor yang akan dimodelkan ke dalam diagram *use case*. Berikut adalah penjabaran dari aktor yang dapat berinteraksi dengan sistem.

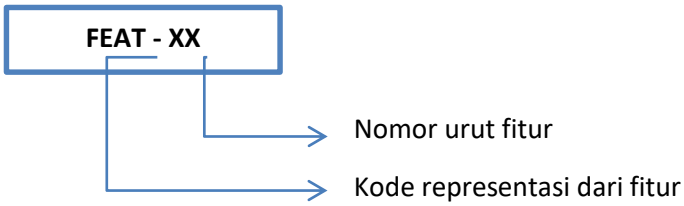
Tabel 4.8 Identifikasi Pengguna

Tipe Pemangku Kepentingan	Aktor	Deskripsi
Pengguna	Dispenduk capil	Dispenduk capil adalah pengguna/ aktor yang memiliki hak akses paling tinggi dibandingkan dengan pengguna/aktor yang lain. dispenduk capil bertanggung jawab terhadap segala sesuatu yang berhubungan dengan administrasi kependudukan.
	Kecamatan	Kecamatan memiliki tugas untuk melaporkan laporan bulanan yang telah dibuat oleh desa/kelurahan pada dispenduk capil.
	Desa/kelurahan	Desa/kelurahan merupakan instansi dasar yang bertanggung jawab dalam pengajuan administrasi kependudukan. Desa/kelurahan juga memiliki tugas bulanan untuk membuat laporan data kependudukan.

4.2.6 Identifikasi Fitur

Identifikasi fitur bertujuan untuk mempresentasikan solusi yang ditawarkan untuk memenuhi kebutuhan pengguna sehingga masalah yang

dihadapi oleh pemangku kpentingan dapat diselesaikan. Hasil dari identifikasi fitur ini nantinya akan digunakan sebagai informasi untuk menentukan kebutuhan fungsional dan non fungsional sistem. Tabel 4.5 menjelaskan tentang identifikasi fitur yang ada pada sisitem informasi pelaporan pendataan data kependudukan kabupaten malang. Di dalam tabel tersebut setiap fitur memiliki kode penomoran dengan aturan penomoran sebagai berikut :



Gambar 4.6 Aturan Penomoran Fitur

Tabel 4. 9 Identifikasi Fitur

Kode Fitur	Fitur	Deskripsi
FEAT- 01	Autentifikasi pengguna	Sistem dapat mengenali pengguna sebelum menggunakan sistem serta dapat membatasi pengguna terhadap informasi dan layanan yang diberikan sesuai dengan hak akses yang dimiliki.
FEAT -02	Pengajuan administrasi kependudukan	Sistem dapat digunakan untuk melakukan pengajuan administrasi kependudukan secara *online*.
FEAT -03	Laporan kependudukan	Sistem dapat digunakan untuk menampilkan laporan pengajuan administrasi kependudukan secara berkala dan menyediakan fitur untuk mengunduh atau mencetak laporan tersebut.
FEAT - 04	Rekap laporan	Sistem dapat digunakan untuk menampilkan rekap dari laporan kependudukan berdasarkan bulan dan menyediakan fitur untuk mengunduh atau mencetak rekap laporan tersebut.
FEAT -05	Cetak formulir pengajuan administrasi kependudukan	Sistem menyediakan fitur untuk mengunduh atau mencetak formulir pengajuan administrasi kependudukan.
FEAT – 06	Cetak kartu pengambilan	Sistem menyediakan fitur untuk mengunduh atau mencetak kartu pengambilan.

4.2.6.1 Perbaikan Identifikasi Fitur

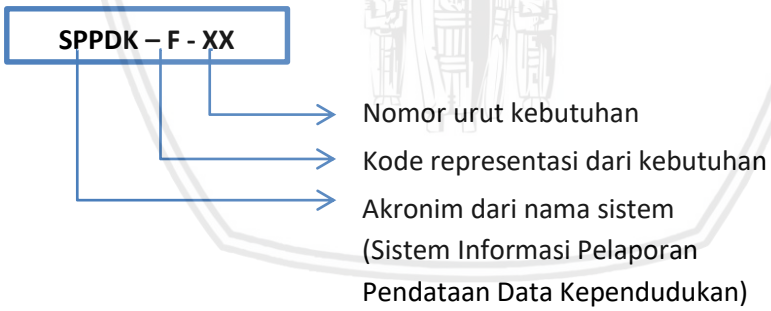
Perbaikan identifikasi fitur ini terjadi karena adanya pengidentifikasian ulang yang dilakukan oleh *stakeholder* pada fase *elaboration* metode *Rational Unified Process*. Hasil dari perbaikan identifikasi fitur pada sistem informasi yang akan dikembangkan ditunjukkan oleh tabel 4.10.

Tabel 4.10 Perbaikan fitur

Kode Fitur	Fitur	Deskripsi
FEAT - 07	Manajemen desa	Sistem menyediakan fitur untuk memasukkan atau mengubah data desa/kelurahan yang meliputi nama kecamatan, nama desa/kelurahan dan nama kepala desa atau nama lurah.

Pada tabel 4.10 menunjukkan bahwa terdapat penambahan 1 fitur yaitu fitur dengan kode FEAT – 07. Penambahan 1 fitur ini dihasilkan dari proses iterasi identifikasi ulang yang dilakukan oleh *stakeholder*. Hasil dari identifikasi ulang tersebut didapat bahwa *stakeholder* ingin sistem yang akan dikembangkan dapat digunakan untuk memasukkan dan mengubah data desa/kelurahan yang meliputi nama kecamatan, nama desa/kelurahan dan nama kepala desa atau nama lurah. Hal ini bertujuan untuk memudahkan pihak desa/kelurahan apabila terjadi perubahan data terkait ketiga hal tersebut.

4.2.7 Kebutuhan Fungsional



Gambar 4.7 Aturan Penomoran Kebutuhan Fungsional

Kebutuhan fungsional adalah aktivitas dan *service* yang harus disediakan oleh sistem yang akan dikembangkan. Daftar kebutuhan fungsional ini dituliskan dalam bentuk tabel dan dikelompokkan berdasarkan aktor yang telah ditetapkan sebelumnya. Kebutuhan fungsional memiliki kode penomoran sesuai gambar 4.7. Berikut adalah daftar kebutuhan fungsional dari sistem yang akan dikembangkan :

Tabel 4.11 Kebutuhan Fungsional Sistem

Kode Fitur	Kode Persyaratan fungsional	Nama fungsi	Deskripsi
FEAT- 01	SPPDK-F-01	Autentifikasi pengguna	Sistem menyediakan fasilitas untuk mengakses sistem sesuai dengan hak akses yang dimiliki.
FEAT -02	SPPDK-F-02	Mengajukan kartu keluarga	Sistem menyediakan fasilitas untuk melakukan pengajuan kartu keluarga.
	SPPDK-F-03	Mengajukan akta kelahiran	Sistem menyediakan fasilitas untuk melakukan pengajuan akta kelahiran.
	SPPDK-F-04	Mengajukan akta kematian	Sistem menyediakan fasilitas untuk melakukan pengajuan akta kematian.
	SPPDK-F-05	Mengajukan pidah datang	Sistem menyediakan fasilitas untuk melakukan pengajuan surat pindah datang ke kabupaten malang.
	SPPDK-F-06	Mengajukan pindah keluar	Sistem menyediakan fasilitas untuk melakukan pengajuan surat pindah keluar dari kabupaten malang.
	SPPDK – F- 07	Validasi pengajuan	Sistem menyediakan fasilitas untuk memvalidasi pengajuan administrasi kependudukan.
FEAT- 03	SPPDK – F -08	Melihat laporan	Sistem menyediakan fasilitas bagi pengguna untuk melihat laporan pengajuan administrasi kependudukan berdasarkan bulan dalam bentuk tabel.
	SPPDK – F - 09	Cetak laporan	Sistem dapat menyediakan fasilitas untuk mengunduh atau mencetak laporan
FEAT-04	SPPDK – F- 10	Melihat rekap laporan	Sistem menyediakan fasilitas untuk melihat rekap laporan secara berkala dalam bentuk tabel dan grafik.
	SPPDK – F -11	Cetak rekap laporan	Sistem menyediakan fasilitas untuk mengunduh atau mencetak rekap laporan pengajuan administrasi

Kode Fitur	Kode Persyaratan fungsional	Nama fungsi	Deskripsi
			kependudukan.
FEAT -05	SPPDK – F - 12	Cetak pengajuan kartu keluarga	Sistem menyediakan fasilitas untuk mengunduh atau mencetak formulir pengajuan kk yang telah diinputkan sebelumnya ke dalam sistem.
	SPPDK – F- 13	Cetak pengajuan akta kelahiran	Sistem menyediakan fasilitas untuk mengunduh atau mencetak formulir pengajuan akta kelahiran yang telah diinputkan sebelumnya ke dalam sistem.
	SPPDK – F - 14	Cetak pengajuan akta kematian	Sistem menyediakan fasilitas untuk mengunduh atau mencetak formulir pengajuan akta kematian yang telah diinputkan sebelumnya ke dalam sistem.
	SPPDK – F- 15	Cetak pengajuan surat pindah datang	Sistem menyediakan fasilitas untuk mengunduh atau mencetak formulir pengajuan pindah datang ke kabupaten malang yang telah diinputkan sebelumnya ke dalam sistem.
	SPPDK – F - 16	Cetak pengajuan surat pindah keluar	Sistem menyediakan fasilitas untuk mengunduh atau mencetak formulir pengajuan pindah keluar dari kabupaten malang yang telah diinputkan sebelumnya ke dalam sistem.
FEAT - 06	SPPDK – F -17	Cetak kartu pengambilan	Sistem menyediakan fasilitas untuk mengunduh atau mencetak kartu pengambilan.

4.2.7.1 Perbaikan Kebutuhan Fungsional

Pengeditifikasian kebutuhan fungsional yang dilakukan oleh *stakeholder* pada fase *elaboration* metode *Rational Unified Process* menghasilkan perbaikan kebutuhan fungsional. Perbaikan kebutuhan fungsional tersebut dapat dilihat pada tabel 4.12

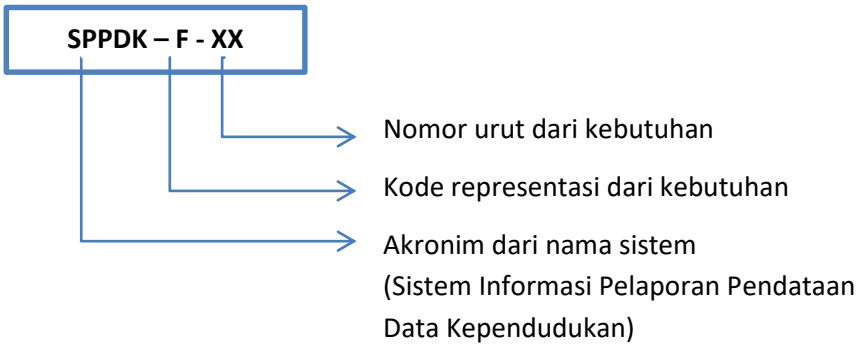
Tabel 4.12 Perbaikan Kebutuhan Fungsional

Kode fitur	Kode persyaratan fungsional	Nama fungsi	Deskripsi
FEAT - 03	SPPDK – F- 08	Melihat laporan	Sistem menyediakan fasilitas untuk melihat laporan secara berkala yaitu berdasarkan bulan dan berdasarkan range tanggal.
FEAT - 07	SPPDK – F - 18	Manajemen desa	Sistem menyediakan fasilitas untuk memasukkan atau mengubah data yang dimiliki oleh desa/kelurahan meliputi nama kecamatan, nama desa/kelurahan dan nama kepala desa atau lurah.

Pada tabel 4.12 menunjukkan terdapat 2 perbaikan kebutuhan fungsional yang terjadi karena adanya iterasi pengidentifikasian ulang kebutuhan fungsional oleh *stakeholder*. Perbaikan kebutuhan fungsional yang pertama yaitu kebutuhan fungsional dengan kode SPPDK – F -08. Perbaikan kebutuhan fungsional dengan kode SPPDK – F- 08 yaitu pada kebutuhan sebelumnya *stakeholder* hanya ingin menampilkan laporan berdasarkan bulan yang diinginkan akan tetapi setelah dilakukan pengidentifikasian ulang terhadap kebutuhan fungsional *stakeholder* ingin menampilkan laporan tidak hanya berdasarkan bulan tetapi berdasarkan range tanggal agar laporan yang dihasilkan lebih mudah dipantau.

Perbaikan kebutuhan fungsional selanjutnya adalah kebutuhan fungsional dengan kode SPPDK – F- 18. Kebutuhan fungsional dengan kode SPPDK – F- 18 merupakan hasil penambahan kebutuhan yang mana *stakeholder* ingin sistem dapat digunakan untuk memasukkan atau mengubah data dasar yang dimiliki oleh desa/kelurahan meliputi nama kecamatan, nama desa/kelurahan dan nama kepala desa atau nama lurah.

4.2.8 Kebutuhan Non Fungsional



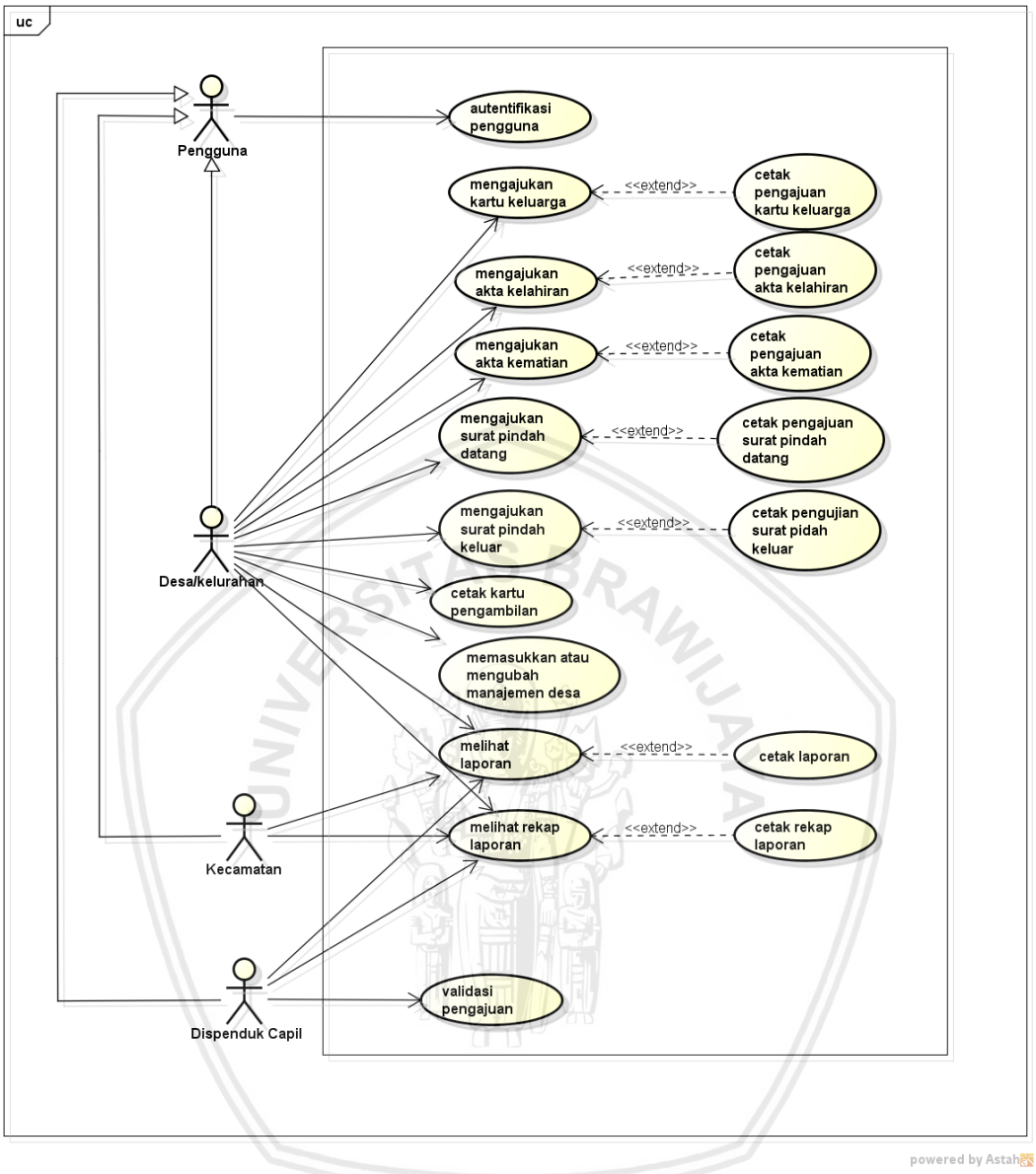
Gambar 4.8 Aturan Penomoran Kebutuhan Non Fungsional

Kebutuhan non fungsional merupakan batasan dari sistem. Daftar kebutuhan non fungsional dari sistem yang akan dikembangkan dituliskan dalam bentuk tabel. Kebutuhan fungsional memiliki kode penomoran sesuai gambar 4.8. Berikut adalah daftar kebutuhan non fungsional :

Tabel 4. 13 Daftar Kebutuhan Non Fungsional

No.	Kode kebutuhan	Nama fungsi	Deskripsi
1.	SPPDK – NF - 001	Compability	Sistem dapat berjalan dengan baik setidaknya pada 3 *browser* yang berbeda.

4.3 Pemodelan Use Case



Gambar 4.9 *Use Case Diagram*

Tahap pemodelan *use case* dilakukan setelah tahap analisis persyaratan. Pada pemodelan *use case* ini dilakukan identifikasi aktor, identifikasi *use case*, dan *use case scenario* untuk masing – masing use case yang telah diidentifikasi. Identifikasi aktor dilakukan dengan mengelompokkan pengguna sistem berdasarkan karakteristik yang dimiliki. Sedangkan identifikasi *use case* dilakukan dengan mengindentifikasi tujuan aktor ketika menggunakan sistem. Identifikasi use case ini dilakukan berdasarkan hasil identifikasi kebutuhan aktor yang telah dilakukan sebelumnya pada tahap analisis persyaratan.

Selanjutnya dari masing-masing *use case* yang telah terindentifikasi dibuat *use case scenario*. *Use case scenario* ini berisi penjelasan singkat dari masing – masing *use case*, kondisi yang harus dipenuhi sebelum *use case*

dilakukan dan kondisi setelah *use case* dilakukan, urutan tahap yang harus dilakukan untuk menyelesaikan use case. Urutan tahap ini terbagi menjadi dua yaitu skenario utama dan skenario alternatif. Urutan tahap ini nantinya menjadi informasi untuk pemodelan aktivitas dalam *diagram activity*.

4.3.1 Use Case Diagram

Use case diagram menggambarkan hubungan antara aktor dengan *use case.* Aktor pada diagram *use case* diidentifikasi dengan menggelompokkan pengguna sistem ke dalam beberapa peran. Sedangkan *use case* diidentifikasi dengan menetukan tujuan pemangku kepentingan menggunakan sistem. Diagram *use case* pada penelitian ini ditujukkan oleh gambar 4.9.

Use case yang telah diidentifikasi akan dihubungkan dengan solusi yang ditawarkan dalam aktivitas proses bisnis *to-be*, dengan pemangku kepentingan serta dengan fitur – fitur yang telah diidentifikasi. Tujuan penghubungan *use case* ini adalah untuk menunjukkan hubungan yang lebih baik dalam penggunaan pemodelan proses bisnis dan *use case* dalan pengembangan sistem. Hubungan aktivitas proses bisnis *to-be* dengan *use case* ditujukkan pada tabel 4.12.

Tabel 4. 12 Hubungan Aktivitas Proses Bisnis *To-Be* Dengan *Use Case*

Proses bisnis	Aktivitas	Use case
Pengajuan administrasi kependudukan	Mengisi formulir pengajuan	Mengajukan kartu keluarga, akta kelahiran, akta kematian, surat pindah datang, surat pindah keluar
	mencetak kartu pengambilan	Cetak kartu pengambilan
	Menvalidasi pengajuan	Validasi pengajuan
Pelaporan data kependudukan	Melihat laporan	Melihat laporan
	Melihat rekap laporan	Melihat rekap laporan

Selanjutnya adalah menghubungkan *use case* dengan pemangku kepentingan. Hal ini dilakukan untuk membantu pemangku kepentingan dalam memahami apa saja yang dapat dilakukan oleh pengguna terhadap sistem informasi yang dikembangkan. Berikut adalah tabel hubungan *use case* dengan pemangku kepentingan.

Tabel 4.13 Hubungan Use Case Dengan Pemangku Kepentingan

Use case	Pengguna	Tipe pemangku kepentingan
Autentifikasi pengguna	Desa/kelurahan, kecamatan, dispenduk capil	Pengguna
Mengajukan kartu keluarga	Desa/kelurahan	
Mengajukan akta kelahiran	Desa/kelurahan	
Mengajukan akta kematian	Desa/kelurahan	
Mengajukan surat pidah datang	Desa/kelurahan	
Mengajukan surat pindah keluar	Desa/kelurahan	
Cetak pengajuan kartu keluarga	Desa/kelurahan	
Cetak pengajuan akta kelahiran	Desa/kelurahan	
Cetak pengajuan akta kematian	Desa/keluarahan	
Cetak pengajuan surat pindah datang	Desa/kelurahan	
Cetak pengajuan surat pindah keluar	Desa/keluarahan	
Cetak kartu pengambilan	Desa/kelurahan	
Melihat laporan	Desa/kelurahan, kecamatan, dispenduk capil	
Cetak laporan	Desa/kelurahan, kecamatan , dispenduk capil	
Melihat rekap laporan	Dispenduk capil	
Cetak rekap laporan	Dispenduk capil	
Validasi pengajuan	Dispenduk capil	
Memasukkan atau	Desa/kelurahan	

Use case	Pengguna	Tipe pemangku kepentingan
mengubah manajemen desa		

4.3.2 Deskripsi Aktor

Deskripsi aktor merupakan penjelasan secara singkat mengenai batasan pengguna sistem informasi yang telah teridentifikasi. Di dalam deskripsi aktor juga terdapat penjelasan mengenai tujuan aktor menggunakan sistem yang akan dikembangkan. Berikut adalah tabel deskripsi aktor pada *use case* sistem informasi pelaporan dan pendataan data kependudukan kabupaten malang.

Tabel 4.15 Deskripsi Aktor

Nama Aktor	Deskripsi
Pengguna	Aktor pengguna diperankan oleh pihak desa/kelurahan, kecamatan dan Dispenduk Capil yang akan menggunakan sistem. Aktor-aktor ini dapat menggunakan sistem informasi serta dapat mengakses informasi yang ada pada sistem sesuai dengan hak akses yang dimiliki.
Desa/kelurahan	Aktor desa/kelurahan diperankan oleh para petugas desa/kelurahan yang menggunakan sistem untuk mengolah laporan data kependudukan, pengajuan administrasi kependudukan serta melihat hasil rekapitulasi laporan data kependudukan dari desa/kelurahan tersebut.
Kecamatan	Aktor kecamatan diperankan oleh para petugas kecamatan yang menggunakan sistem untuk melihat hasil dari rekapitulasi laporan harian data kependudukan yang telah diinputkan oleh desa/kelurahan se-kecamatan.
Dispenduk Capil	Aktor Dispenduk Capil diperankan oleh para petugas Dispenduk capil yang menggunakan sistem untuk memvalidasi pengajuan administrasi kependudukan dari desa/kelurahan dan untuk melihat hasil laporan harian data kependudukan dari desa/kelurahan yang ada di wilayah kabupaten malang.

4.3.3 Use Case Scenario

Use case scenario dibuat berdasarkan diagram *use case* yang telah dimodelkan. *Use case scenario* dari setiap kebutuhan fungsional ini akan digambarkan dalam bentuk tabel . Pada tabel *use case scenario* ini ada beberapa keterangan seperti deskripsi singkat mengenai *use case*, aktor yang dapat mengakses *use case*, kondisi sebelum dan sesudah *use case* dijalankan, dan

langkah – langkah dalam menjalankan *use case*. Langkah – langkah ini akan digunakan sebagai acuan dalam memodelkan aktivitas pada diagram aktivitas.

4.3.3.1 *Use Case Scenario* Autentifikasi Pengguna

Use case scenario autentifikasi pengguna ini menjelaskan tujuan dan bagaimana aktor pengguna menggunakan sistem ini. Di dalam *use case scenario* autentifikasi pengguna terdapat kondisi yang dibutuhkan agar aktor dapat menggunakan sistem dan kondisi apabila use case berhasil dilakukan. *Use case scenario* autentifikasi pengguna ini juga terdapat urutan tahapan yang harus dilakukan oleh aktor. Berikut adalah *use case scenario* autentifikasi pengguna.

Tabel 4.14 *Use Case Scenario* Autentifikasi Pengguna

Brief Description	*Use case* autentifikasi pengguna menjelaskan bagaimana sistem dapat melakukan verifikasi terhadap aktor yang menggunakan sistem. Sehingga aktor dapat menggunakan sistem sesuai dengan hak akses yang dimilikinya.
Actor	Pengguna
Pre-condition	• Halam login sudah terbuka atau tersedia
Post-condition	• Identitas aktor Pengguna teridentifikasi oleh sistem. • Aktor pengguna berhasil masuk ke sistem sesuai dengan hak akses yang dimiliki.
Basic Flow	1. Sistem menampilkan halaman login 2. Aktor memasukkan *username* dan *password* **{memasukkan *username* dan *password*}** 3. Aktor menekan tombol 'login' 4. Sistem melakukan pengecekan terhadap *username* dan *password* yang dimasukkan **{mengidentifikasi pengguna}** Jika benar maka aktor dapat masuk ke dalam sistem sesuai dengan hak akses masing – masing.
Alternative Flow	**A1 : menangani kegagalan memasukkan username dan password** Pada saat {memasukkan *username* dan *password*} pada *basic flow*, jika *username* dan *password* tidak diisi oleh aktor , maka sistem akan menampilkan pesan peringatan bahwa *username* dan *password* harus diisi. **A2 : menangani kegagalan mengidentifikasi pengguna** Pada saat {mengidentifikasi pegguna} pada *basic flow*, jika *username* dan *password* gagal diidentifikasi, maka sistem akan menampilkan pesan bahwa *username* dan *password* gagal diidentifikasi.

4.3.3.2 *Use Case Scenario* Mengajukan Kartu Keluarga

Pada *use case scenario* mengajukan kartu keluarga terdapat penjelasan singkat mengenai tujuan dan bagaimana aktor menggunakan sistem. Di dalam *use case scenario* mengajukan kartu keluarga ini terdapat kondisi yang dibutuhkan agar aktor dapat menggunan sistem dan kondisi setelah *use case* berhasil dilakukan. Pada *use case scenario* mengajukan kartu keluarga ini juga terdapat urutan kegiatan yang harus dilakukan oleh aktor ketika menggunakan sistem. Berikut adalah tabel *use case scenario* mengajukan kartu keluarga.

Tabel 4.15 *Use Case Scenario* Mengajukan Kartu Keluarga

Brief Description	*Use case* ini menjelaskan proses bagaimana aktor menggunakan sistem untuk memasukkan data pengajuan kartu keluarga
Actor	Desa/kelurahan
Pre-condition	• Aktor telah berhasil login ke dalam sistem • Halaman pengajuan yang berisi formulir pengajuan telah terbuka atau tersedia
Post-condition	Data pengajuan kartu keluarga akan disimpan ke dalam basis data dan aktor lain yang memiliki hak akses dapat melihat data tersebut.
Basic Flow	1. Use case akan dimulai ketika aktor memilih fungsi untuk memasukkan data pengajuan kartu keluarga 2. Sistem menampilkan formulir pengajuan kartu keluarga 3. Aktor mengisikan data pengajuan kartu keluarga **{mengisi data pengajuan kartu keluarga}** 4. Akto rmemilih fungsi menyimpan data tersebut 5. Sistem menyimpan data tersebut ke dalam *database* **{menyimpan data pengajuan}** 6. Use case berakhir
Alternative Flow	**A1 : menangani kegagalan menyimpan data pengajuan** Pada saat **{menyimpan data pengajuan}** pada *basic flow*, jika ada data penting yang tidak diisikan oleh aktor , maka: 1. Sistem akan menampilkan pemberitahuan untuk mengisi data penting yang tidak diisi. 2. Use case kembali pada *basic flow* tahap **{mengisi data pengajuan kartu keluarga}** **A2 : menangani kegagalan menyimpan data pengajuan** pada saat **{menyimpan data pengajuan} pada basic flow,** jika ada data NIK yang diisikan tidak sama dengan 16 karakter, maka : 1. Sistem akan menampilkan pemberitahuan bahwa NIK

	yang tidak sesuai tersebut harus diisi dengan 16 karakter. 2. Use case kembali pada *basic flow* tahap **{mengisi data pengajuan kartu keluarga}**

4.3.3.3 *Use Case Scenario* Mengajukan Akta Kelahiran

Pada *use case scenario* mengajukan akta kelahiran terdapat penjelasan singkat mengenai tujuan dan bagaimana aktor menggunakan sistem. Di dalam *use case scenario* mengajukan akta kelahiran ini terdapat kondisi yang dibutuhkan agar aktor dapat menggunan sistem dan kondisi setelah *use case* berhasil dilakukan. Pada *use case scenario* mengajukan akta kelahiran ini juga terdapat urutan kegiatan yang harus dilakukan oleh aktor ketika menggunakan sistem. Berikut adalah tabel *use case scenario* mengajukan akta kelahiran.

Tabel 4.16 *Use Case Scenario* Mengajukan Akta Kelahiran

Brief Description	*Use case* ini menjelaskan proses bagaimana aktor menggunakan sistem untuk memasukkan data pengajuan akta kelahiran
Actor	Desa/kelurahan
Pre-condition	• Aktor telah berhasil login ke dalam sistem • Halaman pengajuan yang berisi formulir pengajuan akta kelahiran telah terbuka atau tersedia
Post-condition	Data pengajuan akta kelahiran akan disimpan ke dalam basis data dan aktor lain yang memiliki hak akses dapat melihat data tersebut.
Basic Flow	1. Use case akan dimulai ketika aktor memilih fungsi untuk memasukkan data pengajuan akta kelahiran 2. Sistem menampilkan formulir pengajuan akta kelahiran 3. Aktor mengisikan data pengajuan akta kelahiran **{mengisi data pengajuan akta kelahiran }** 4. Akto rmemilih fungsi menyimpan data tersebut 5. Sistem menyimpan data tersebut ke dalam *database* **{menyimpan data pengajuan}** 6. Use case berakhir
Alternative Flow	**A1 : menangani kegagalan menyimpan data pengajuan** Pada saat **{menyimpan data pengajuan}** pada *basic flow*, jika ada data penting yang tidak diisikan oleh aktor , maka: 1. Sistem akan menampilkan pemberitahuan untuk mengisi data penting yang tidak diisi. 2. Use case kembali pada *basic flow* tahap **{mengisi data**

	pengajuan akta kelahiran } **A2 : menangani kegagalan menyimpan data pengajuan** pada saat **{menyimpan data pengajuan} pada basic flow,** jika ada data NIK yang diisikan tidak sama dengan 16 karakter, maka : 1. Sistem akan menampilkan pemberitahuan bahwa NIK yang tidak sesuai tersebut harus diisi dengan 16 karakter. 2. Use case kembali pada *basic flow* tahap **{mengisi data pengajuan akta kelahiran}**

4.3.3.3 *Use Case Scenario* Mengajuakan Akta Kematian

Pada *use case scenario* mengajukan akta kelahiran terdapat penjelasan singkat mengenai tujuan dan bagaimana aktor menggunakan sistem. Di dalam *use case scenario* mengajukan akta kelahiran ini terdapat kondisi yang dibutuhkan agar aktor dapat menggunan sistem dan kondisi setelah *use case* berhasil dilakukan. Pada *use case scenario* mengajukan akta kelahiran ini juga terdapat urutan kegiatan yang harus dilakukan oleh aktor ketika menggunakan sistem. Berikut adalah tabel *use case scenario* mengajukan akta kelahiran.

Tabel 4.17 *Use Case Scenario* Mengajukan Akta Kematian

Brief Description	*Use case* ini menjelaskan proses bagaimana aktor menggunakan sistem untuk memasukkan data pengajuan akta kematian
Actor	Desa/kelurahan
Pre-condition	• Aktor telah berhasil login ke dalam sistem • Halaman pengajuan yang berisi formulir pengajuan akta kematian telah terbuka atau tersedia
Post-condition	Data pengajuan akta kematian akan disimpan ke dalam basis data dan aktor lain yang memiliki hak akses dapat melihat data tersebut.
Basic Flow	1. Use case akan dimulai ketika aktor memilih fungsi untuk memasukkan data pengajuan akta kematian 2. Sistem menampilkan formulir pengajuan akta kematian 3. Aktor mengisikan data pengajuan akta kematian **{mengisi data pengajuan akta kematian }** 4. Akto rmemilih fungsi menyimpan data tersebut 5. Sistem menyimpan data tersebut ke dalam *database* **{menyimpan data pengajuan}** 6. Use case berakhir

Alternative Flow	**A1 : menangani kegagalan menyimpan data pengajuan** Pada saat **{menyimpan data pengajuan}** pada *basic flow*, jika ada data yang tidak diisikan oleh aktor, maka: 1. Sistem akan menampilkan pemberitahuan untuk mengisi data penting yang tidak diisi. 2. Use case kembali pada *basic flow* tahap **{mengisi data pengajuan akta kematian }** **A2 : menangani kegagalan menyimpan data pengajuan** pada saat **{menyimpan data pengajuan} pada basic flow,** jika ada data NIK yang diisikan tidak sama dengan 16 karakter, maka : 3. Sistem akan menampilkan pemberitahuan bahwa NIK yang tidak sesuai tersebut harus diisi dengan 16 karakter. **4.** Use case kembali pada *basic flow* tahap **{mengisi data pengajuan akta kematian}**

4.3.3.4 *Use Case Scenario* Mengajukan Surat Pindah Datang

Pada *use case scenario* mengajukan surat pindah datang terdapat penjelasan singkat mengenai tujuan dan bagaimana aktor menggunakan sistem. Di dalam *use case scenario* mengajukan surat pindah datang ini terdapat kondisi yang dibutuhkan agar aktor dapat menggunan sistem dan kondisi setelah *use case* berhasil dilakukan. Pada use case scenario mengajukan surat pindah datang ini juga terdapat urutan kegiatan yang harus dilakukan oleh aktor ketika menggunakan sistem. Berikut adalah tabel *use case scenario* mengajukan surat pindah datang.

Tabel 4.18 *Use Case Scenario* Mengajukan Surat Pindah Datang

Brief Description	*Use case* ini menjelaskan proses bagaimana aktor menggunakan sistem untuk memasukkan data pengajuan surat pindah datang
Actor	Desa/kelurahan
Pre-condition	• Aktor telah berhasil login ke dalam sistem • Halaman pengajuan yang berisi formulir pengajuan surat pindah datang telah terbuka atau tersedia
Post-condition	Data pengajuan akta kematian akan disimpan ke dalam basis data dan aktor lain yang memiliki hak akses dapat melihat data tersebut.
Basic Flow	1. Use case akan dimulai ketika aktor memilih fungsi untuk memasukkan data pengajuan surat pindah datang 2. Sistem menampilkan formulir pengajuan surat pindah datang 3. Aktor mengisikan data pengajuan surat pindah datang

	{mengisi data pengajuan surat pindah datang } 4. Akto rmemilih fungsi menyimpan data tersebut 5. Sistem menyimpan data tersebut ke dalam *database* **{menyimpan data pengajuan}** 6. Use case berakhir
Alternative Flow	**A1 : menangani kegagalan menyimpan data pengajuan** Pada saat **{menyimpan data pengajuan}** pada *basic flow*, jika ada data penting yang tidak diisikan oleh aktor , maka: 1. Sistem akan menampilkan pemberitahuan untuk mengisi data penting yang tidak diisi. 2. Use case kembali pada *basic flow* tahap **{mengisi data pengajuan surat pindah datang }** **A2 : menangani kegagalan menyimpan data pengajuan** pada saat **{menyimpan data pengajuan} pada basic flow,** jika ada data NIK yang diisikan tidak sama dengan 16 karakter, maka : 1. Sistem akan menampilkan pemberitahuan bahwa NIK yang tidak sesuai tersebut harus diisi dengan 16 karakter. 2. Use case kembali pada *basic flow* tahap **{mengisi data pengajuan surat pindah datang}**

4.3.3.5 *Use Case Scenario* Mengajukan Surat Pindah Keluar

Pada *use case scenario* mengajukan surat pindah datang terdapat penjelasan singkat mengenai tujuan dan bagaimana aktor menggunakan sistem. Di dalam *use case scenario* mengajukan surat pindah keluar ini terdapat kondisi yang dibutuhkan agar aktor dapat menggunan sistem dan kondisi setelah *use case* berhasil dilakukan. Pada *use case scenario* mengajukan surat pindah keluar ini juga terdapat urutan kegiatan yang harus dilakukan oleh aktor ketika menggunakan sistem. Berikut adalah tabel *use case scenario* mengajukan surat pindah keluar.

Tabel 4.19 *Use Case Scenario* Mengajuakan Surat Pindah Keluar

Brief Description	*Use case* ini menjelaskan proses bagaimana aktor menggunakan sistem untuk memasukkan data pengajuan surat pindah keluar
Actor	Desa/kelurahan
Pre-condition	• Aktor telah berhasil login ke dalam sistem • Halaman pengajuan yang berisi formulir pengajuan surat pindah keluar telah terbuka atau tersedia
Post-condition	Data pengajuan akta kematian akan disimpan ke dalam basis data dan aktor lain yang memiliki hak akses dapat melihat data

	tersebut.
Basic Flow	1. Use case akan dimulai ketika aktor memilih fungsi untuk memasukkan data pengajuan surat pindah keluar 2. Sistem menampilkan formulir pengajuan surat pindah keluar 3. Aktor mengisikan data pengajuan surat pindah keluar **{mengisi data pengajuan surat pindah keluar }** 4. Akto rmemilih fungsi menyimpan data tersebut 5. Sistem menyimpan data tersebut ke dalam *database* **{menyimpan data pengajuan}** 6. Use case berakhir
Alternative Flow	**A1 : menangani kegagalan menyimpan data pengajuan** Pada saat **{menyimpan data pengajuan}** pada *basic flow*, jika ada data penting yang tidak diisikan oleh aktor , maka: 1. Sistem akan menampilkan pemberitahuan untuk mengisi data penting yang tidak diisi. 2. Use case kembali pada *basic flow* tahap **{mengisi data pengajuan surat pindah keluar }** **A2 : menangani kegagalan menyimpan data pengajuan** pada saat **{menyimpan data pengajuan} pada basic flow,** jika ada data NIK yang diisikan tidak sama dengan 16 karakter, maka : 1. Sistem akan menampilkan pemberitahuan bahwa NIK yang tidak sesuai tersebut harus diisi dengan 16 karakter. 2. Use case kembali pada *basic flow* tahap **{mengisi data pengajuan surat pindah keluar}**

4.3.3.6 *Use Case* Menambah Kartu Pengambilan

Pada *use case scenario* menambah kartu pengambilan terdapat penjelasan singkat mengenai tujuan dan bagaimana aktor menggunakan sistem. Di dalam *use case scenario* menambah kartu pengambilan ini terdapat kondisi yang dibutuhkan agar aktor dapat menggunan sistem dan kondisi setelah *use case* berhasil dilakukan. Pada *use case scenario* menambah kartu pengambilan ini juga terdapat urutan kegiatan yang harus dilakukan oleh aktor ketika menggunakan sistem. Berikut adalah tabel *use case scenario* cetak kartu pengambilan.

Tabel 4.20 *Use Case Scenario* Menambah Kartu Pengambilan

Brief Description	*Use case* ini menjelaskan proses bagaimana aktor menggunakan sistem untuk menambah kartu pengambilan yang nantinya akan

	diberikan kepada pemohon sebagai bukti untuk mengambil dokumen kependudukan.
Actor	Desa/kelurahan
Pre-condition	• Halaman menambah kartu pengambilan telah terbuka atau tersedia
Post-condition	Aktor dapat mengetahui dan menambah kartu pengambilan yang nantinya akan befungsi sebagi bukti untuk pengambilan dokumen kependudukan
Basic Flow	1. Use case dimulai ketika aktor memilih fungsi kartu pengambilan dan memilih fungsi tambah 2. Sistem menampilkan form untuk memasukkan data kartu pengambilan 3. Aktor mengisikan data kartu pengambilan 4. Aktor memilih fungsi menyimpan data tersebut 5. Sistem menyimpan data tersebut ke dalam *database* 6. Use case berakhir
Alternative Flow	-

4.3.3.7 *Use Case Scenario* Cetak Kartu Pengambilan

Pada *use case scenario* cetak kartu pengambilan terdapat penjelasan singkat mengenai tujuan dan bagaimana aktor menggunakan sistem. Di dalam *use case scenario* cetak kartu pengambilan ini terdapat kondisi yang dibutuhkan agar aktor dapat menggunan sistem dan kondisi setelah *use case* berhasil dilakukan. Pada *use case scenario* cetak kartu pengambilan ini juga terdapat urutan kegiatan yang harus dilakukan oleh aktor ketika menggunakan sistem. Berikut adalah tabel *use case scenario* cetak kartu pengambilan.

Tabel 4.21 *Use Case Scenario* Cetak Kartu Pengambilan

Brief Description	*Use case* ini menjelaskan proses bagaimana aktor menggunakan sistem untuk mencetak kartu pengambilan yang nantinya akan diberikan kepada pemohon sebagai bukti untuk mengambil dokumen kependudukan.
Actor	Pengguna
Pre-condition	• Halaman cetak kartu pengambilan telah terbuka atau tersedia
Post-condition	Pengguna dapat mengetahui dan mencetak kartu pengambilan yang nantinya akan befungsi sebagai bukti untuk pengambilan dokumen kependudukan
Basic Flow	1. Use case dimulai ketika aktor memilih fungsi cetak kartu

	pengambilan 2. Sistem menampilkan kartu pengambilan 3. Pengguna memilih fungsi untuk mencetak kartu pengambilan 4. Use case berakhir
Alternative Flow	-

4.3.3.8 Use Case Scenario Melihat Laporan

Use case scenario melihat laporan berisi penjelasan singkat mengenai tujuan dan bagaimana aktor menggunakan sistem. Di dalam *use case scenario* ini juga terdapat kondisi yang dibutuhkan agar aktor dapat menggunakan sistem serta kondisi setelah *use case* berhasil dilakukan. Pada *use case scenario* ini terdapat urutan kegiatan dalam menggunakan sistem. Berikut adalah tabel *use case scenario* melihat laporan.

Tabel 4.22 *Use Case Scenario* Melihat Laporan

Brief Description	*Use case* ini menjelaskan proses bagaimana aktor menggunakan sistem untuk melihat hasil laporan yang diperoleh dari data pengajuan administrasi kependudukan yang telah dimasukkan ke dalam sistem oleh desa/kelurahan.
Actor	Pengguna
Pre-condition	• Halaman laporan data kependudukan telah tersedia atau terbuka
Post-condition	Pengguna dapat mengetahui hasil laporan data pengajuan administrasi kependudukan berdasarkan skala tertentu
Basic Flow	1. Use case dimulai ketika aktor memilih fungsi laporan 2. Aktor memilih manampilkan laporan berdasarkan skala tertentu (bulan atau range tanggal) 3. Aktor memasukkan bulan atau range tanggal tertentu 4. Sistem memuat laporan data kependudukan berdasarkan masukan dari aktor dari database **{memuat data dari database}** 5. Sistem menampilkan data kependudukan ke dalam bentuk tabel 6. Use case berakhir
Alternative Flow	**A1 : menangani kegagalan memuat data dari database** Pada saat {memuat data dari database} pada *basic flow*, jika data

	yang dimuat tidak ada maka sistem akan menampilkan pesan bahwa data yang diinginkan tidak ada.

4.3.3.9 Use Case Scenario Melihat Rekap Laporan

Use case scenario melihat rekap laporan berisi penjelasan singkat mengenai tujuan dan bagaimana aktor menggunakan sistem. Di dalam *use case scenario* ini juga terdapat kondisi yang dibutuhkan agar aktor dapat menggunakan sistem serta kondisi setelah *use case* berhasil dilakukan. Pada *use case scenario* ini terdapat urutan kegiatan dalam menggunakan sistem. Berikut adalah tabel *use case scenario* melihat rekap laporan.

Tabel 4.23 *Use Case Scenario* Melihat Rekap Laporan

Brief Description	*Use case* ini menjelaskan proses bagaimana aktor menggunakan sistem untuk melihat rekap hasil laporan yang diperoleh dari data pengajuan administrasi kependudukan yang telah dimasukkan ke dalam sistem oleh desa/kelurahan.
Actor	Pengguna
Pre-condition	• Halaman rekap laporan data kependudukan telah tersedia atau terbuka
Post-condition	Pengguna dapat mengetahui hasil rekap laporan data pengajuan administrasi kependudukan berdasarkan skala tertentu
Basic Flow	1. Use case dimulai ketika aktor memilih fungsi laporan 2. Aktor memilih manampilkan laporan berdasarkan skala tertentu (bulan atau range tanggal) 3. Aktor memasukkan bulan atau range tanggal tertentu 4. Sistem memuat rekap laporan data kependudukan berdasarkan masukan dari aktor dari database **{memuat data dari database}** 5. Sistem menampilkan data kependudukan ke dalam bentuk tabel dan grafik 6. Use case berakhir
Alternative Flow	**A1 : menangani kegagalan memuat data dari database** Pada saat {memuat data dari database} pada *basic flow*, jika data yang dimuat tidak ada maka sistem akan menampilkan pesan bahwa data yang diinginkan tidak ada.

4.3.3.10 *Use Case Scenario* Validasi Pengajuan

Tabel 4.24 merupakan tabel *use case scenario* validasi pengajuan administrasi kependudukan. Pada tabel ini terdapat penjelasan mengenai tujuan dan bagaimana aktor menggunakan sistem. Di dalam *use case scenario* ini juga terdapat kondisi yang dibutuhkan dalam menggunakan sistem dan kondisi setelah use case berhasil dilakukan. *Use case scenario* ini juga berisi langkah – langkah yang harus dilakukan oleh aktor ketika menggunakan sistem.

Tabel 4.24 *Use Case Scenario* Validasi Pengajuan

Brief Description	*Use case* ini menjelaskan proses bagaimana aktor menggunakan sistem untuk melihat daftar pengajuan administrasi kependudukan dan untuk melakukan validasi terhadap pengajuan administrasi kependudukan yang telah dilakukan oleh desa/kelurahan
Actor	Dispenduk capil
Pre-condition	• Halaman validasi pengajuan administrasi kependudukan telah terbuka atau tersedia
Post-condition	• Status validasi akan disimpan ke dalam basis data dan aktor lain yang memiliki hak akses dapat melihat data tersebut.
Basic Flow	1. Use case dimulai ketika sistem menampilkan daftar pengajuan administrasi kependudukan 2. Aktor memilih untuk melihat detail salah satu data pengajuan administrasi kependudukan 3. Sistem menampikan detail data pengajuan administrasi data kependudukan 4. Aktor memilih fungsi validasi data 5. Sistem menyimpan status validasi ke dalam *database* 6. Use case berakhir
Alternative Flow	-

4.3.3.11 *Use Case Scenario* Memasukkan Atau Mengubah Manajemen Desa

Use case scenario dari memasukkan atau mengubah manajemen desa ditunjukkan oleh tabel 4.25. Pada tabel 4.25 terdapat penjelasan mengenai tujuan dan bagaimana aktor menggunakan sistem. Di dalam use case scenario ini juga terdapat kondisi yang dibutuhkan dalam menggunakan sistem dan kondisi setelah use case berhasil dilakukan. *Use case scenario* ini juga berisi langkah – langkah yang harus dilakukan oleh aktor ketika menggunakan sistem.

Tabel 4.25 *Use Case Scenario* Memasukkan Atau Mengubah Manajemen Desa

Brief Description	*Use case* ini menjelaskan proses bagaimana aktor menggunakan sistem untuk memasukkan atau mengubah data dasar yang dimiliki oleh desa/kelurahan. Data tersebut meliputi nama kecamatan, nama desa/kelurahan dan nama kepala desa atau nama lurah.
Actor	Desa/kelurahan
Pre-condition	• Halaman manajemen desa telah terbuka atau tersedia
Post-condition	• Data manajemen desa akan disimpan ke dalam basis data.
Basic Flow	1. Use case dimulai ketika sistem menampilkan data manajemen desa yang dimiliki oleh aktor. 2. Aktor memilih untuk mengupdate data manajemen desa yang dimiliki. 3. Sistem menampikan form untuk mengisi data manajemen desa yang terdiri dari nama kecamatan, nama desa/kelurahan dan nama kepala desa atau lurah. 4. Aktor memilih fungsi menyimpan data. 5. Sistem menyimpan data manajemen desa ke dalam *database.* 6. Use case berakhir.
Alternative Flow	-

4.4 Pemodelan Aktivitas

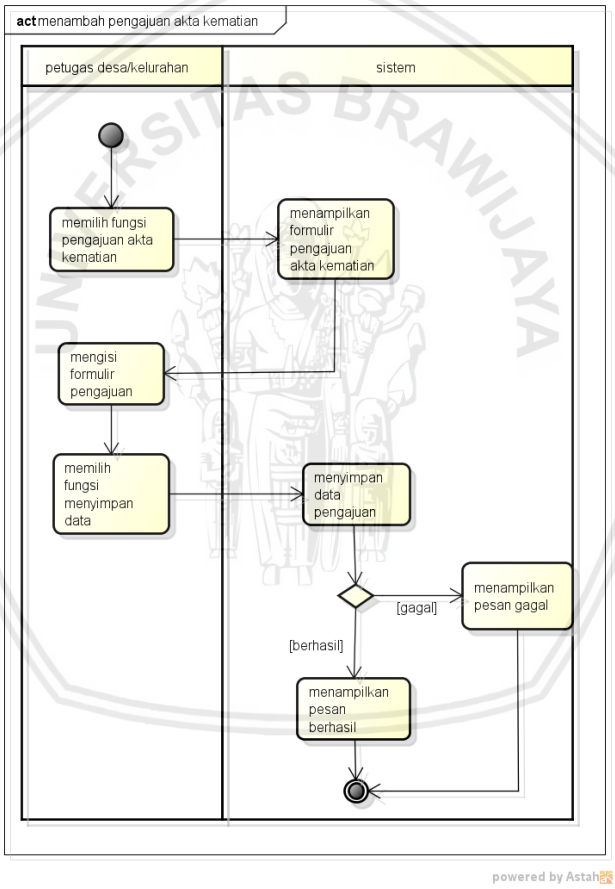
Pemodelan aktivitas dilakukan berdasarakan alur pada *use case scenario* yang telah dibuat pada tahap sebelumnya. Tujuan pemodelan aktivitas ini adalah untuk memudahkan pemangku kepentingan dalam memahami alur kerja pada suatu use case. Di dalam pemodelan aktivitas ini terdapat beberapa aktivitas yang dimodelkan. Aktvitas tersebut meliputi aktivitas mengelola laporan, aktivitas mengelola pengajuan administrasi kependudukan, aktivitas melihat hasil rekapitulasi laporan data kependudukan dan aktivitas menvalidasi pengajuan administrasi kependudukan.

4.4.1 Diagram Aktivitas Mengajukan Akta Kematian

Diagram aktivitas mengajukan akta kematian menunjukkan aktivitas yang dilakukan oleh aktor desa/kelurahan dan sistem pada saat sistem digunakan untuk melakukan pengajuan akta kematian secara online. Alur diagram aktivitas

ini dibuat berdasarkan alur kerja pada *use case scenario* yang ada pada tabel 4.17.

Alur aktivitas mengajukan akta kematian ini dimulai dari aktor memilih fungsi untuk memasukkan data pengajuan akta kematian. Setelah itu sistem akan menampilkan formulir pengajuan yang harus diisi oleh aktor. Apabila aktor selesai mengisikan data pada formulir pengajuan selanjutnya aktor memilih fungsi untuk menyimpan data tersebut maka sistem akan secara otomatis menyimpannya ke dalam *database*. Jika proses penyimpanan tidak berhasil karena ada data yang belum terisi maka sistem akan menampilkan pemberitahuan untuk mengisi data tersebut. Dan apabila data berhasil disimpan maka sistem akan menampilkan pemberitahuan bahwa data telah berhasil disimpan.

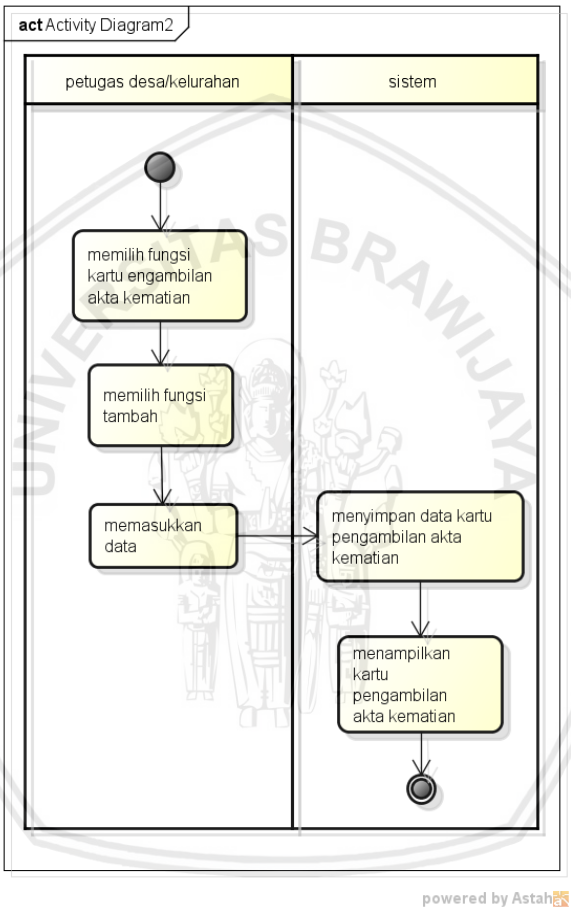


Gambar 4.10 Diagram Aktivitas Mengajukan Akta Kematian

4.4.2 Diagram Aktivitas Menambah Kartu Pengambilan Akta Kematian

Diagram aktivitas menambah kartu pengambilan akta kematian menunjukkan aktivitas yang dilakukan oleh aktor desa/kelurahan pada saat aktor ingin membuat kartu pengambilan yang nantinya akan diberikan kepada pemohon sebagai syarat untuk mengambil akta kematian yang telah diajukan. Alur aktivitas ini dimulai dengan aktor memilih menu untuk menambah kartu

pengambilan akta kematian. Selanjutnya sistem akan menampilkan formulir yang harus diisi oleh aktor. Apabila aktor selesai mengisikan data pada formulir yang disediakan maka aktor harus memilih fungsi untuk menyimpan data tersebut agar data yang telah diisikan tersimpan ke dalam database. Jika sistem berhasil menyimpan data tersebut ke dalam database maka sistem akan menampilkan pesan bahwa data telah berhasil disimpana. Begitu pula sebaliknya jika sistem tidak berhasil menyimpan data tersebut ke dalam database maka sistem akan menampilkan pesan gagal. Berikut adalah diagram aktivitas menambah kartu pengambilan akta kematian.

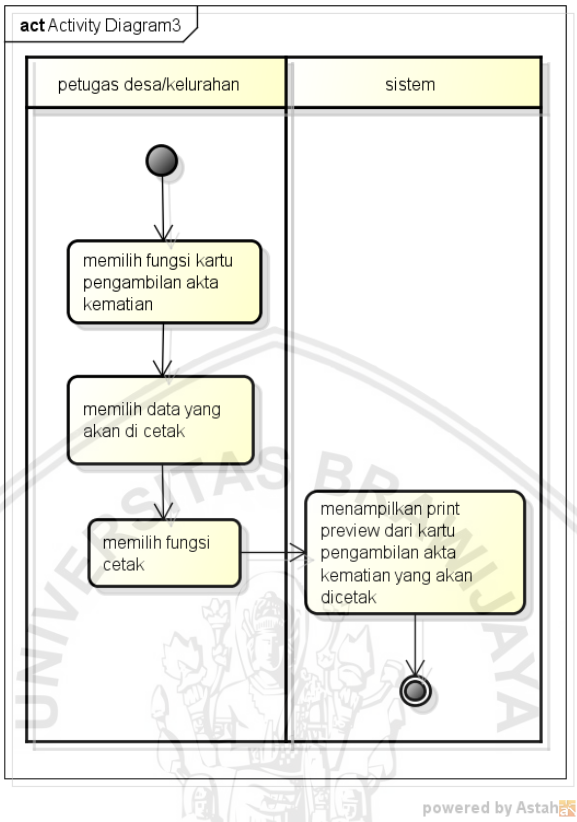


Gambar 4.11 Diagram Aktivitas Menambah Kartu Pengambilan Akta Kematian

4.4.3 Diagram Aktivitas Mencetak Kartu Pengambilan Akta Kematian

Diagram aktivitas mencetak kartu pengambilan akta kematian menunjukkan aktivitas yang dilakukan oleh aktor desa/kelurahan pada saat aktor akan mencetak kartu pengambilan akta kematian yang telah dibuat sebelumnya. Aktivitas ini merupakan kelanjutan dari aktivitas menambah kartu pengambilan akta kematian. Alur aktivitas ini dimulai ketika aktor memilih fungsi kartu pengambilan akta kematian. Selanjutnya sistem akan menampilkan semua data kartu pengambilan akta kematian yang telah dibuat sebelumnya dalam bentuk tabel. Pada tabel yang ditampilkan ini terdapat kolom opsi yang berisi fungsi

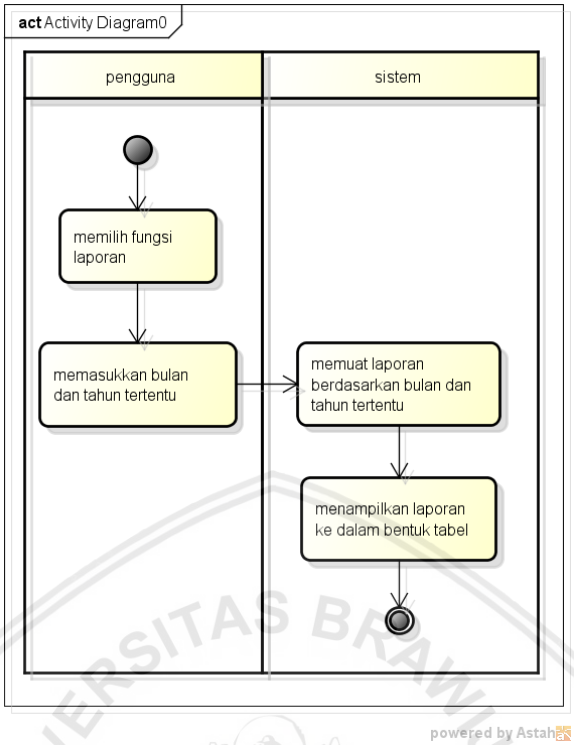
cetak. Aktor desa/kelurahan harus memilih fungsi cetak ini untuk melakukan proses cetak. Berikut adalah digram aktivitas untuk mencetak kartu pengambilan akta kematian.



Gambar 4.12 Diagram Aktivitas Mencetak Kartu Pengambilan Akta Kematian

4.4.4 Diagram Aktivitas Melihat Laporan Bulanan

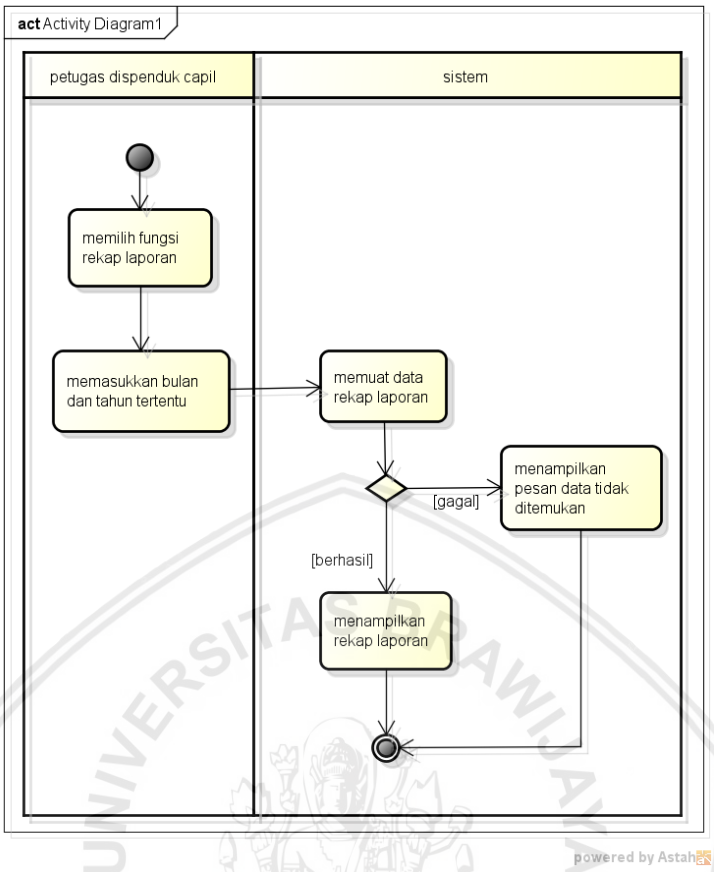
Diagram aktivitas melihat laporan menunjukkan aktivitas yang dilakukan oleh aktor pengguna untuk melihat laporan data kependudukan yang diperoleh dari data pengajuan administrasi kependudukan yang ada pada desa/kelurahan yang telah tersimpan pada database. Alur aktivitas ini dimulai dengan aktor pengguna memilih fungsi laporan. Selanjutnya aktor harus memasukkan kondisi tertentu sesuai dengan level pengguna untuk *menfilter* data yang akan ditampilkan oleh sistem. Laporan bulanan data kependudukan ini akan ditampilkan dalam bentuk tabel. Apabila data yang diinginkan oleh aktor pengguna tidak ada pada database maka sistem akan menampilkan tabel kosongan. Berikut adalah diagram aktivitas dari melihat laporan bulanan.



Gambar 4.13 Diagram Aktivitas Melihat Laporan

4.4.5 Diagram Aktivitas Melihat Rekap Laporan Bulanan

Diagram aktivitas rekap laporan bulanan menunjukkan aktivitas yang dilakukan oleh aktor dispenduk capil untuk mengetahui rekapan laporan berdasarkan bulan dan tahun tertentu. Alur aktivitas ini dimulai dengan aktor dispenduk capil memilih fungsi rekap bulanan dimana pada fungsi ini aktir dispenduk capil harus mengisikan bulan serta tahun yang diinginkan. Setelah aktor mengisikan kondisi bulan dan tahun tersebut selanjutnya sistem akan *menfilter* data yang ada pada database berdasarkan pada bulan dan tahun tertentu. Selanjutnya data akan ditampilkan dalam bentuk tabel. Apabila data yang dicari tidak ada pada database maka sistem akan menampilkan pesan bahwa data tidak ditemukan. Berikut adalah diagram aktivitas melihat rekapan laporan bulanan.



Gambar 4.14 Diagram Aktivitas Rekap Laporan

BAB 5 PERANCANGAN SISTEM

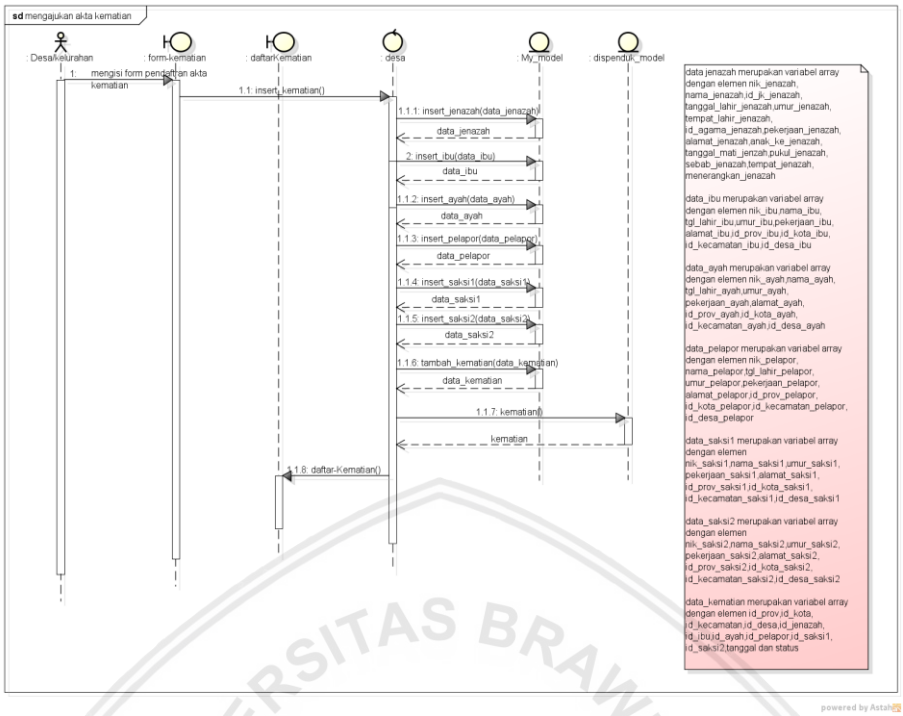
Bab ini berisi dokumentasi perancangan dari sistem informasi pelaporan dan pendataan kependudukan kabupaten malang yang akan dibangun. Dokumentasi perancangan dimodelkan dengan menggunakan UML. Pemodelan yang dijelaskan dalam bab ini antara lain adalah *sequence diagram* dan *class diagram*. Untuk perancangan basis data menggunakan *Physical Data Model* (PDM). Pada bab ini juga terdapat perancangan antarmuka pengguna.

4.1 Sequence Diagram

Pemodelan *sequence diagram* dilakukan untuk memvisualisasikan pertukaran pesan antar entitas aktor, entitas *boundary*, objek *control*, dan objek *model* yang saling berinteraksi untuk memenuhi kebutuhan pengguna sistem. *Sequence diagram* ini menekankan urutan aktivitas berbasis waktu yang terjadi antara seperangkat objek. Pada subbab ini *sequence diagram* dibuat berdasarkan aliran pada spesifikasi *use case* yang telah didokumentasikan pada bab sebelumnya.

4.1.1 *Sequence Diagram* Mengajukan Akta Kematian

Sequence diagram pada gambar 5.1 merupakan visualisai interaksi antar objek pada aktivitas mengajukan akta kematian yang dilakukan oleh aktor desa . Beberapa objek yang terlibat dalam interaksi mengajukan akta kematian ini antara lain adalah aktor desa, *form-kematian.php* dan *daftarKematian.php* sebagai objek *boundary*, kelas desa sebagai objek *control*, dan kelas *My_model* sebagai objek *model*. Berikut gambar dari *sequence diagram* mengajukan akta kematian.



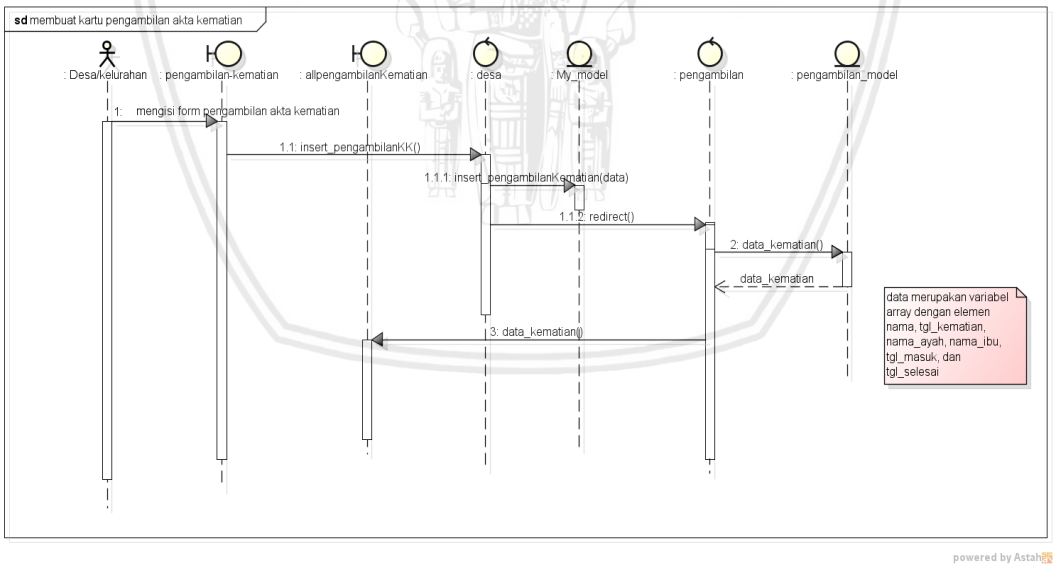
Gambar 5.1 *Sequence Diagram* Mengajukan Akta Kematian

Pertukaran pesan pada aktivitas mengajukan akta kematian dimulai ketika aktor desa mengakses event mengisi form pengajuan akta kematian melalui objek *form-kematian.php* sebagai objek *boundary*. Selanjutnya pesan diteruskan ke kelas desa sebagai objek *control*. Objek *control* kemudian berinterkasi dengan kelas *My_model* sebagai objek *model*. Dalam interaksi ini kelas desa akan memanggil *function* insert_jenazah, insert_ibu, insert_ayah, insert_pelapor, insert_saksi1, insert_saksi2 dan insert_kematian yang ada pada kelas *My_model*. Setiap *function* memliki *argument* yang berbeda-beda dan masing-masing merupakan variabel array. *Function* insert_jenazah memiliki *argument* data_jenazah dengan elemen nik_jenazah, nama_jenazah, id_jk_jenazah, tanggal_lahir_jenazah, umur_jenazah, tempat_lahir_jenazah, id_agama_jenazah, pekerjaan_jenazah, alamat_jenazah, anak_ke_jenazah, tanggal_mati_jenzah, pukul_jenazah,sebab_jenazah, tempat_jenazah, menerangkan_jenazah. *Function* insert_ibu memiliki *argument* data_ibu dengan elemen nik_ibu, nama_ibu, tgl_lahir_ibu, umur_ibu, pekerjaan_ibu, alamat_ibu, id_prov_ibu, id_kota_ibu, id_kecamatan_ibu, id_desa_ibu. *Function* insert_ayah memiliki *argument* data_ayah dengan elemen nik_ayah, nama_ayah, tgl_lahir_ayah, umur_ayah, pekerjaan_ayah, alamat_ayah, id_prov_ayah, id_kota_ayah, id_kecamatan_ayah, id_desa_ayah. *Function* insert_pelapor memiliki *argument* data_pelapor dengan elemen nik_pelapor, nama_pelapor, tgl_lahir_pelapor, umur_pelapor, pekerjaan_pelapor, alamat_pelapor, id_prov_pelapor, id_kota_pelapor, id_kecamatan_pelapor, id_desa_pelapor. *Function* insert_saksi1 memiliki *argument* data_saksi1 dengan elemen nik_saksi1, nama_saksi1, umur_saksi1, pekerjaan_saksi1, alamat_saksi1, id_prov_saksi1, id_kota_saksi1, id_kecamatan_saksi1, id_desa_saksi1. Function

insert_saksi2 memiliki argument data_saksi2 dengan elemen nik_saksi2, nama_saksi2, umur_saksi2, pekerjaan_saksi2, alamat_saksi2,id_prov_saksi2, id_kota_saksi2, id_kecamatan_saksi2, id_desa_saksi2. Sedangkan f*unction* insert_kematian memiliki *argument* data_kematian dengan elemen id_prov, id_kota, id_kecamatan, id_desa, id_jenazah, id_ibu, id_ayah, id_pelapor, id_saksi1, id_saksi2, tanggal dan status. Selanjutnya objek *control* berinteraksi lagi dengan kelas *dispenduk_model* untuk mendapatkan data pengajuan akta kematian melalui *method* kematian. Setelah mendapatkan data pengajuan akta kematian selanjutnya kelas desa akan menampilkan data tersebut melalui *daftar-kematian.php*.

4.1.2 *Sequence Diagram* Menambahkan Kartu Pengambilan Akta Kematian

Sequence diagram pada gambar 5.2 merupakan visualisai interaksi antar objek pada aktivitas menambahkan kartu pengambilan akta kematian yang dilakukan oleh aktor desa. Beberapa objek yang terlibat dalam interaksi ini antara lain adalah aktor pengguna, *pengambilanKematian.php* dan *allpengambilanKK.php* sebagai objek *boundary*, kelas desa dan pengambilan sebagai objek *control*, dan kelas *My_model* dan kelas pengambilan_model sebagai objek *model*. Berikut adalah gambar dari *sequence diagram* menambahkan kartu pengambilan akta kematian.



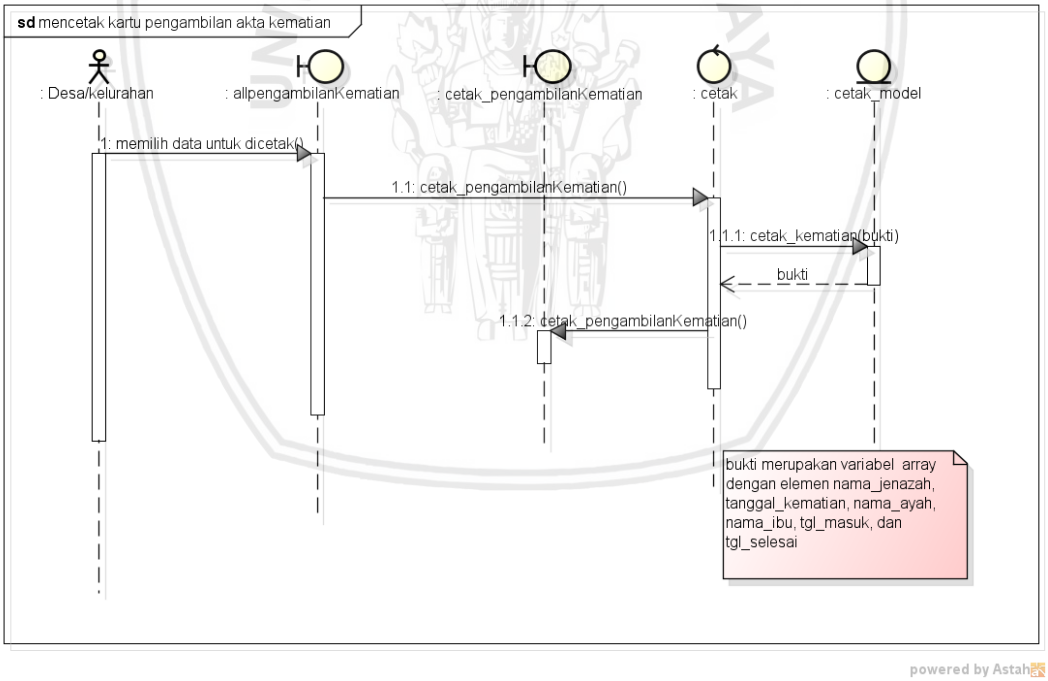
Gambar 5.2 *Sequence Diagram* Menambahkan Kartu Pengambilan Kematian

Pertukaran pesan pada aktivitas menambahkan kartu pengambilan akta kematian ini dimulai ketika aktor pengguna mengakses *event* mengisi form pengambilan akta kematian melalui objek *pengambilan-kematian.php* sebagai objek *boundary*. Kemudian pesan diteruskan ke kelas desa sebagai objek *control*. Objek *control* selanjutnya berinteraksi dengan kelas *My_model* sebagai objek *model* melalui *function* insertpengambilanKematian. *Function* ini memiliki fungsi

untuk memasukkan data yang telah diinputkan pengguna ke dalam *database*. Adapun data yang diinputkan akan disimpan dalam variable array dengan nama data. Elemen pada variable array ini meliputi nama, tgl_kematian, nama_ayah, nama_ibu, tgl_masuk dan tgl_selesai. Selanjutnya pesan akan di *redirect* ke kelas pengambilan sebagai objek *control* dan diteruskan ke kelas pengambilan_model sebagai objek *model* untuk mendapatkan data kartu pengambilan akta kematian melalui *function* data_kematian. Setelah mendapatkan data kartu pengambilan akta kematian tersebut data akan ditampilkan melalui *allpengambilanKematian.php.*

4.1.3 *Sequence Diagram* Mencetak Kartu Pengambilan Akta Kematian

Sequence diagram pada gambar 5.3 merupakan visualisai interaksi antar objek pada aktivitas mencetak kartu pengambilan akta kematian yang dilakukan oleh aktor desa. Beberapa objek yang terlibat dalam interaksi untuk proses mencetak kartu pengambilan ini antara lain adalah aktor pengguna, *allpengambilanKematian.php* sebagai objek *boundary*, kelas cetak sebagai objek *control*, dan kelas cetak_model sebagai objek *model*. Berikut adalah *sequence diagram* untuk mencetak kartu pengambilan akat kematian.



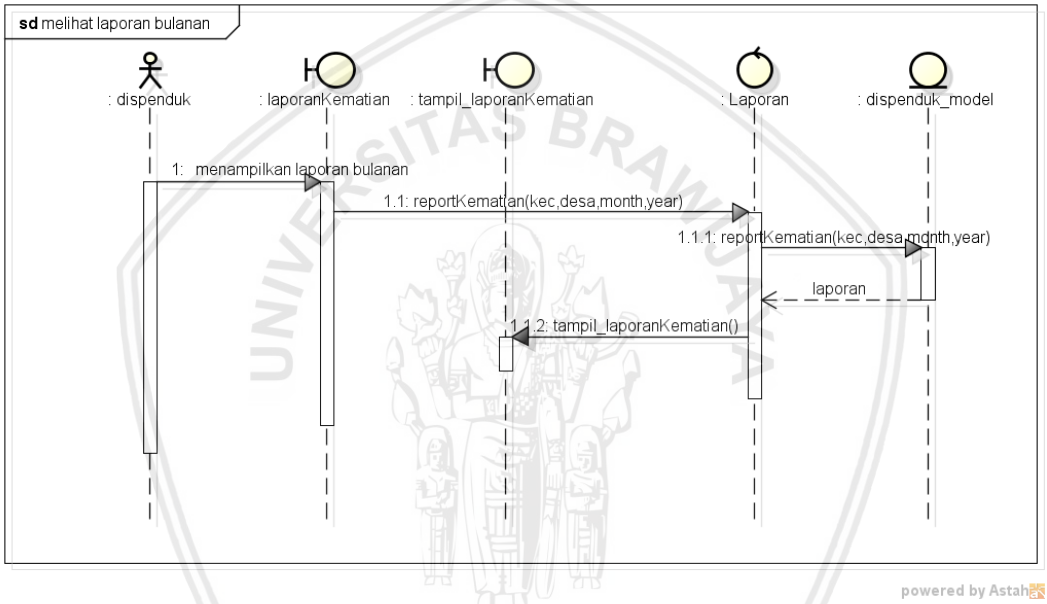
Gambar 5.3 *Sequence Diagram* Cetak Kartu Pengambilan Akta Kematian

Pertukaran pesan pada aktivitas mencetak kartu pengambilan ini dimulai ketika aktor pengguna mengakses halaman *allpengambilanKematian.php se*bagai objek *boundary*. Kemudian pesan diteruskan ke kelas *cetak* sebagai objek *control* dengan *function* cetak_pengambilanKematian. Objek *control* selanjutnya berinteraksi dengan kelas cetak_model sebagai objek *model* melalui *function* cetak_kematian dimana *function* ini memiliki *argument* bukti yang merupakan

variabel array dengan elemen nama_jenazah, tanggal_kematian, nama_ayah, nama_ibu, tgl_masuk, dan tgl_selesai .

4.1.4 *Sequence Diagram* Melihat Laporan Bulanan

Sequence diagram pada gambar 5.4 merupakan visualisai interaksi antar objek pada aktivitas melihat laporan bulanan yang dilakukan oleh aktor dispenduk. Beberapa objek yang terlibat dalam interaksi untuk proses mencetak kartu pengambilan ini antara lain adalah aktor dispenduk, *laporanKematian.php* sebagai objek *boundary*, kelas laporan sebagai objek *control*, dan kelas dispenduk_model sebagai objek *model*. Berikut adalah *sequence diagram* melihat laporan bulanan.



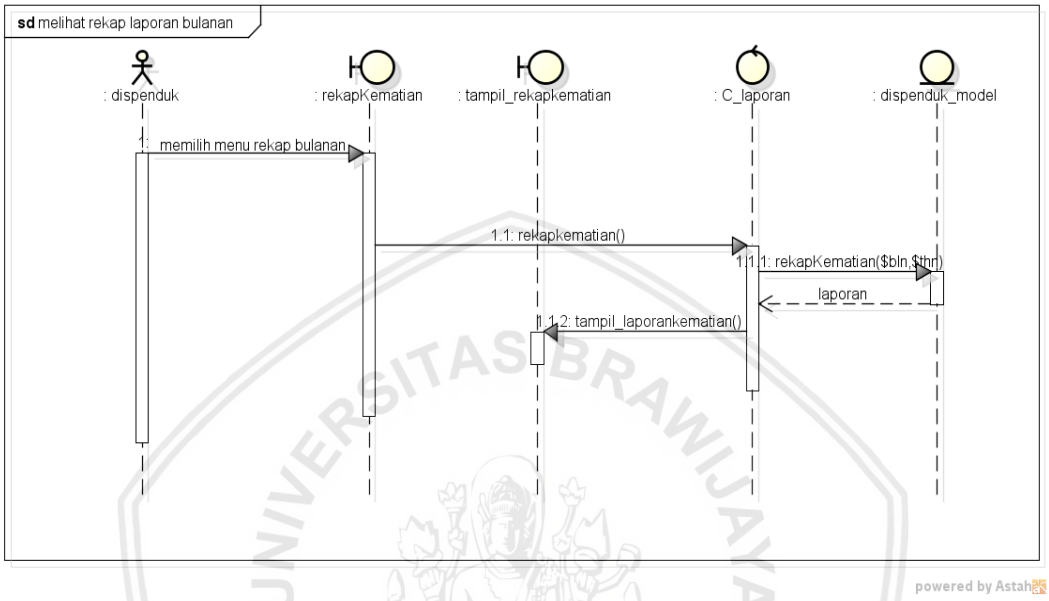
Gambar 5.4 *Sequence Diagram* Melihat Laporan Bulanan

Pertukaran pesan pada aktivitas melihat laporan bulanan ini dimulai ketika aktor pengguna mengakses halaman *laporanKematian.php se*bagai objek *boundary*. Kemudian pesan diteruskan ke kelas laporan sebagai objek *control*. Objek *control* selanjutnya berinteraksi dengan kelas dispenduk_model sebagai objek *model* melalui *function* reportKematian untuk mendapatkan laporan sesuai dengan bulan, tahun, kecamatan dan desa yang diinginkan. Data laporan yang telah diperoleh selanjutnya akan ditampilkan melalui halaman *tampil_laporanKematian.php*.

4.1.5 *Sequence Diagram* Melihat Rekap Laporan Bulanan

Sequence diagram pada gambar 5.5 merupakan visualisai interaksi antar objek pada aktivitas melihat rekap laporan bulanan yang dilakukan oleh aktor dispenduk. Rekap laporan bulanan ini merupakan ringkasan dari laporan bulanan, dimana pada rekap laporan bulanan ini hanya menunjukkan jumlah

pengajuan dokumen kependudukan untuk setiap kecamatan. Beberapa objek yang terlibat dalam interaksi untuk proses mencetak kartu pengambilan ini antara lain adalah aktor dispenduk, *rekapKematian.php* dan *tampil_rekapKematian.php* sebagai objek *boundary*, kelas c_laporan sebagai objek *control*, dan kelas dispenduk_model sebagai objek *model*. Berikut adalah *sequence diagram* melihat rekap laporan bulanan.



Gambar 5.5 Sequence Diagram Melihat Rekap Laporan Bulanan

Pertukaran pesan pada aktivitas melihat rekap laporan bulanan ini dimulai ketika aktor pengguna mengakses halaman *rekapKematian.php se*bagai objek *boundary*. Kemudian pesan diteruskan ke kelas *c_laporan* sebagai objek *control*. Objek *control* selanjutnya berinteraksi dengan kelas dispenduk_model sebagai objek *model* melalui *function* reportKematian untuk mendapatkan laporan sesuai dengan bulan dan tahun. Data laporan yang telah diperoleh selanjutnya akan ditampilkan melalui halaman *tampil_laporanKematian.php*.

4.2 Pemodelan *Class Diagram*

Pemodelan *class diagram* digunakan untuk menunjukkan sekumpulan *class*, *interface*, kolaborasi (*collaboration*) dan hubungan (*relationship*). Pemodelan *class diagram* ini dilakukan berdasarkan *sequence diagram* yang telah dibuat sebelumnya. Pada penelitian ini akan dibuat *class diagram* analisis dan *class diagram* perancangan. *Class diagram* analisis akan digunakan sebagai acuan dalam membuat *class diagram* perancangan. Sedangkan *class diagram* perancangan akan menjadi acuan dalam melakukan pengembangan sistem.

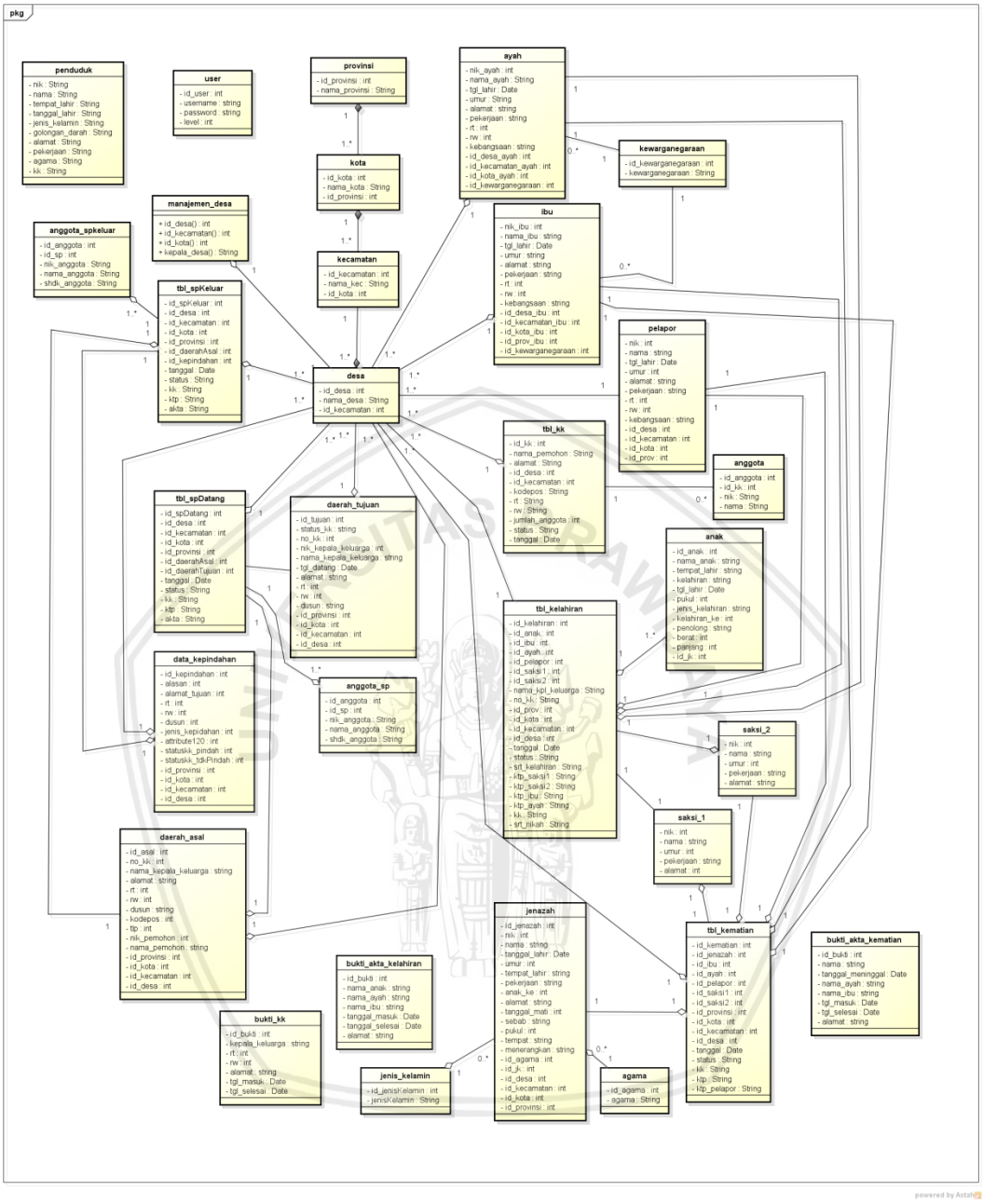
4.2.1 *Class Diagram* Analisis

Gambar 5.6 merupakan gambar dari *class diagram* analisis. *Class Diagram* analisis ini merupakan visualisasi hubungan antar kelas sebagai representasi

objek yang menjadi komponen penyusun sistem yang akan dikembangkan. Berdasarkan gambar 5.6 dapat diketahui bahwa kelas tbl_kk merupakan kelas yang menyimpan data pengajuan kartu keluarga. Kelas tbl_kelahiran merupakan kelas yang menyimpan data pengajuan akta kelahiran. Kelas tbl_kelahiran ini disusun oleh beberapa kelas. Kelas – kelas penyusun tersebut meliputi kelas ibu, kelas ayah, kelas anak, kelas pelapor, kelas saksi1 dan kelas saksi2. Kelas- kelas tersebut memiliki atribut yang mempresentasikan detail data dari pengajuan akta kelahiran.

Sedangkan kelas tbl_kematian merupakan kelas yang menyimpan data terkait pengajuan akta kematian. Kelas tbl_kematian ini juga disusun oleh beberapa kelas. Kelas – kelas penyusun tersebut meliputi kelas ayah, kelas ibu, kelas jenazah, kelas pelapor, kelas saksi1 dan kelas saksi2.

Kelas tbl_spDatang dan kelas tbl_spKeluar merupakan kelas yang mempresentasikan objek yang menyimpan data terkait surat pindah masuk dan surat pindah keluar. Kedua kelas ini tersusun oleh beberapa kelas. Untuk kelas tbl_spDatang disusun oleh kelas daerah_asal dan daerah_tujuan. Sedangkan untuk tbl_spKeluar disusun oleh daerah_asal dan data_kepindahan.

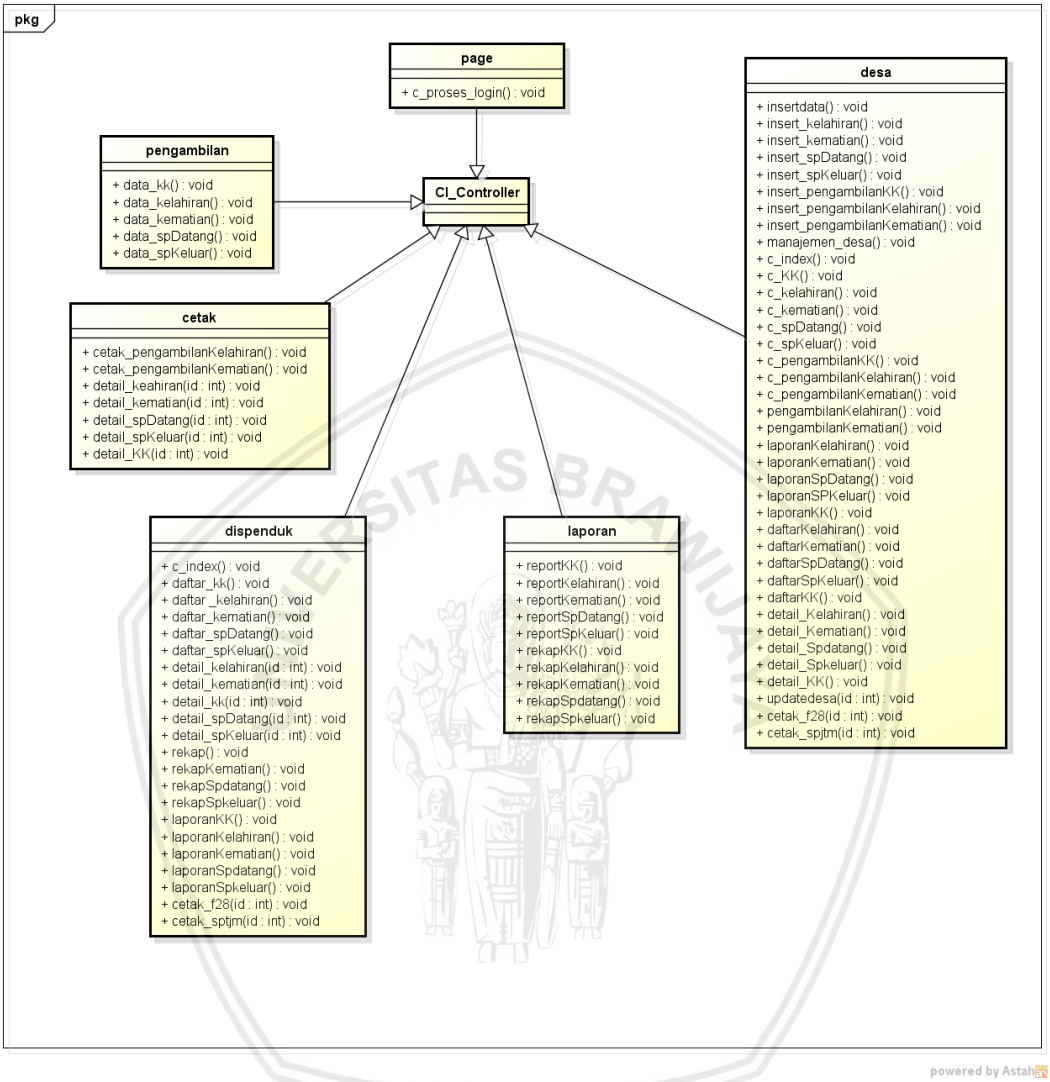


Gambar 5.6 *Class Diagram* Analisis

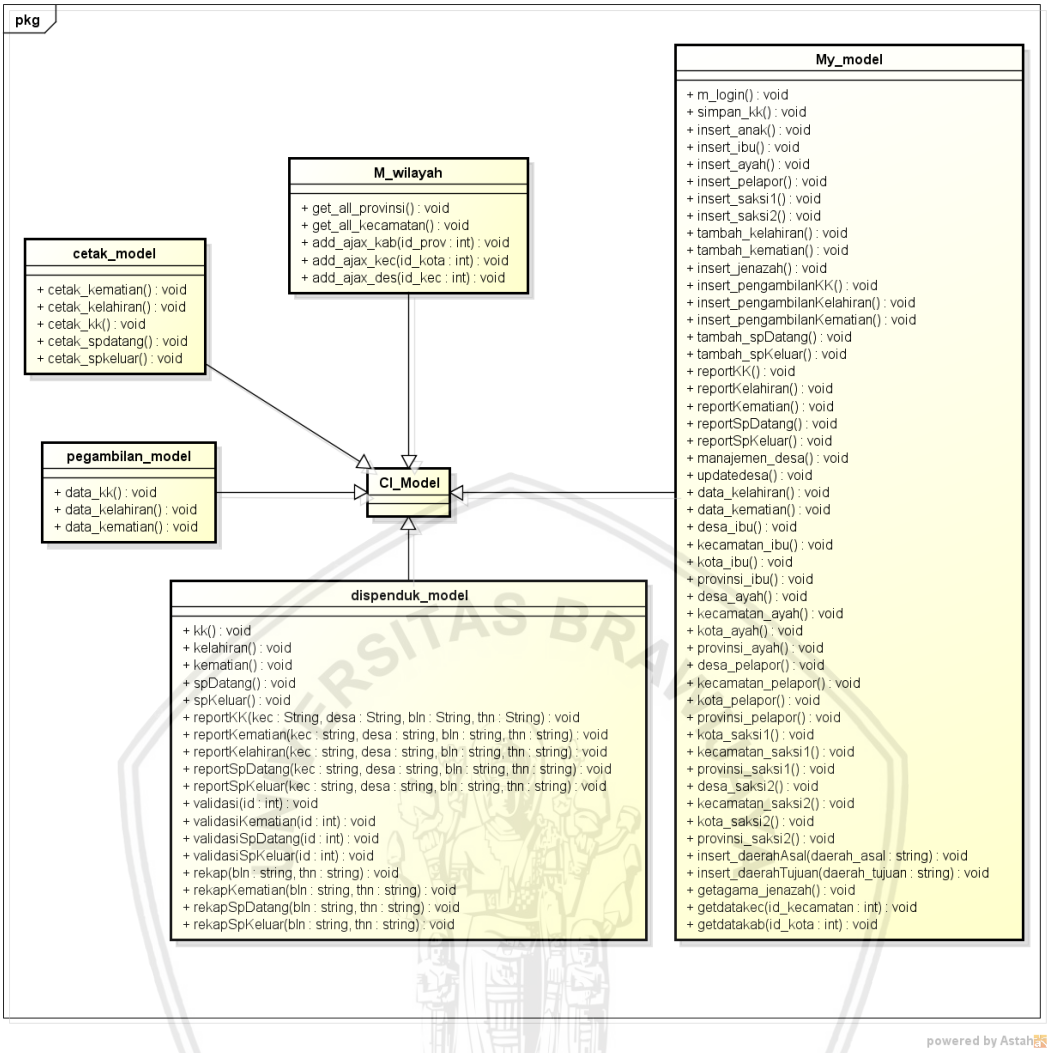
5.2.2. *Class Diagram* Perancangan

Class diagram perancangan merupakan hasil pengembangan dari *class diagram* analisis yang telah dibuat sebelumnya. Pada pemodelan *class diagram* perancangan ini terdapat 2 subkelas yaitu subkelas model dan subkelas controller. *Class diagram* untuk subkelas controller ditunjukkan pada gambar 5.7. Dari gambar 5.7 tersebut dapat diketahui bahwa terdapat 6 kelas yaitu page,

desa, dispenduk, cetak, laporan dan pengambilan. *Class diagram* untuk subkelas model ditujukkan pada gambar 5.8. Dari gambar 5.8 dapat diketahui bahwa terdapat 3 kelas yaitu My_model, dispenduk_model dan pengambilan_model.



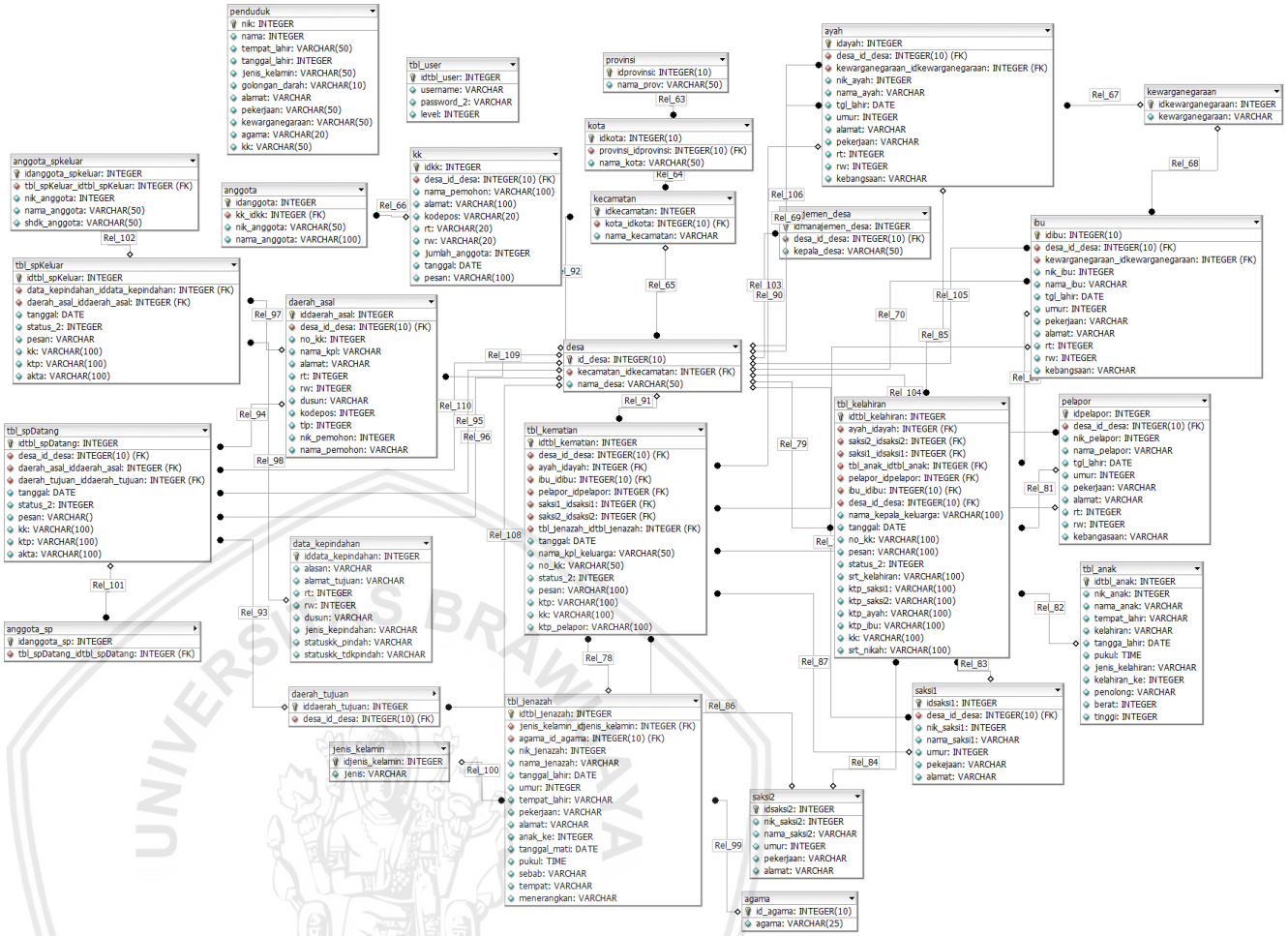
Gambar 5.7 *Class Diagram Controller*



Gambar 5.8 *Class Diagram Model*

4.3 Perancangan Basis Data

Rancangan basis data sistem informasi yang akan dikembangkan pada penelitian ini akan dimodelkan dalam bentuk *Physical Data Model* (PDM). Perancangan basis data ini berdasarkan pada hasil perancangan *class diagram* analisis. Tabel- tabel yang ada pada pada PDM mempresentasikan kelas – kelas pada *class diagram* analisis. Sedangkan untuk atribut pada *class diagram* analisis direpresentasikan ke dalam nama kolom. Rancangan basis data ini nantinya akan diimplementasikan ke dalam *database* sistem yang akan dibangun.



Gambar 5.9 *Physical Data Model*

Pada gambar 5.9 tersebut dapat diketahui bahwa basis data terdiri dari 32 tabel. Data pengguna yang dapat mengakses sistem disimpan pada tabel tbl_user. Data wilayah administrative tersimpan pada tabel tbl_provinsi, tbl_kota, tbl_kecamatan dan tbl_desa. Data pengajuan kartu keluarga dicatat dan disimpan pada tbl_kk. Data pengajuan akta kelahiran dan akta kematian dicatat dan disimpan pada tbl_kelahiran, tbl_kematian, tbl_anak, tbl_jenazah, tbl_saksi1, tbl_saksi2, tbl_pelapor, tbl_ayah, tbl_ibu. Data pengajuan surat pindah datang dan surat pindah keluar dicatat dan disimpan pada tbl_spDatang, tbl_spKeluar, tbl_daerahAsal, tbl_daerahTujuan dan data_kepindahan. Tabel agama berisi data agama yaitu islam, Kristen, katolik, hindu, budha, lainnya. Tabel jenis kelamin berisi data jenis kelamin yaitu laki – laki dan perempuan. Tabel kewarganegaraan menyimpan data kewarganegaraan yaitu WNI dan WNA.

4.4 Perancangan Algoritma

Perancangan algoritma dilakukan untuk menentukan dan menyusun sekumpulan langkah operasi logika untuk memecahkan masalah melakukan perhitungan otomatis, pemrosesan data dan tugas tugas lainnya. Perancangan algoritma dalam penelitian ini didokumentasikan ke dalam *pseudocode*. *Pseudocode* yang telah dibuat akan dijadikan panduan dalam pengembangan sistem informasi pada tahap implementasi.

4.4.1 Mengajukan Akta kematian

Subbab ini berisi rancangan algoritma dari sebuah fungsi yang disediakan oleh sistem untuk pengguna ketika mengajukan akta kematian. Rancangan algoritma ini ditunjukkan oleh tabel 5.1. melalui tabel tersebut dapat diketahui bahwa algoritma dimulai dengan inisialisasi variable id_prov, id_kota, id_kecamatan, id_desa, nama, nik, no_kk, alamat, rt, rw, alasan, jumlah_anggota dan tanggal. Selanjutnya dilakukan pengecekan validasi nilai yang dikirimkan dari formulir pengajuan akta kematian. Jika nilai dinyatakan valid maka memanggil fungsi untuk menyimpan nilai pengajuan akta kematian ke dalam *database* dan menampilkan daftar pengajuan akta kematian serta pesan bahwa proses penyimpanan berhasil. Jika nilai dinyatakan tidak valid maka akan menampilkan pesan bahwa proses gagal.

Tabel 5.1 *Pseudocode* Mengajukan Akta Kematian

No	Pseudocode
1.	START
2.	INIT id_prov as sended id_prov
3.	INIT id_kota as sended id_kota
4.	INIT id_kecamatan as sended id_kecamatan
5.	INIT id_desa as sended id_desa
6.	INIT nama as sended nama
7.	INIT nik as sended nik
8.	INIT no_kk as sended no_kk
9.	INIT alamat as sended alamat

```
10.       INIT rt as sended rt
11.       INIT rw as sended rw
12.       INIT alasan as sended alasan
13.       INIT jumlah _anggota as sended jumlah
14.       INIT tanggal as sended tanggal
15.    IF form validation = TRUE THEN
16.       CALL a function for storing akta kematian
17.       DISPLAY the web page contain  lists the filing of
          akta kematian and success message
18.    ELSE
19.       DISPLAY the web page contain error message
20.  END
```

4.4.2 Menambah Kartu Pengambilan Akta Kematian

Pada subbab ini perancangan algoritma menambah kartu pengambilan akta kematian ini ditunjukkan oleh gambar 5.2. Pada gambar tersebut dapat diketahui bahwa algoritma dimulai dari inisialisasi variabel nama, tanggal_kematian, nama_ayah, nama_ibu, date, dan tanggal masuk. Selanjutnya dilakukan pengecekan hasil validasi nilai yang dikirmkan dari formulir kartu pengambilan akta kematian. Jika nilai dinyatakan valid maka akan memanggil fungsi untuk menyimpan nilai ke dalam *database* kartu pengambilan akta kematian dan menampilkan pesan proses penyimpanan berhasil. Jika nilai dinyatakan tidak valid maka akan menampilkan pesan proses gagal.

Tabel 5.2 *Pseudocode* Menambah Kartu Pengambilan Akta Kematian

No	Pseudocode
1.	START
2.	INIT nama as sended nama
3.	INIT tanggal_kematian as sended taggal_kematian
4.	INIT nama_ayah as sended nama_ayah
5.	INIT nama_ibu as sended nama_ibu
6.	INIT date as sended tanggal_selesai
7.	INIT tanggal_masuk as sended tanggal_masuk
8.	IF form validation = TRUE THEN
9.	CALL function for storing kartu pengambilan akta kematian
10.	DISPLAY a web page containing success message
11.	ELSE
12.	DISPLAY a web page containing error message
13.	END

4.4.3 Mencetak Kartu Pengambilan Akta Kematian

Subbab ini berisi rancangan algoritma dari sebuah fungsi yang disediakan oleh sistem untuk mencetak kartu pengambilan akta kematian. Perancangan algoritma ini ditunjukkan oleh tabel 5.3. Dari tabel tersebut diketahui bahwa algoritma mencetak kartu pengambilan kata kematian dimulai dari pemuatan data kartu pengambilan kematian. Selanjutnya akan menampilkan *print preview* dari kartu pengambilan akta kematian yang akan dicetak.

Tabel 5.3 *Pseudocode* Mencetak Kartu Pengambilan Akta Kematian

No	Pseudocode
1.	START
2.	READ daftar kartu pengambilan akta kematian from database
3.	CALL function to print kartu pengambilan akta kematian
4.	DISPLAY print preview for kartu pengambilan akta kematian
5.	END

4.4.4 Melihat Laporan Bulanan

Subbab ini berisi rancangan algoritma dari sebuah fungsi yang disediakan oleh sistem untuk melihat laporan bulanan. Perancangan algoritma ini ditunjukkan oleh tabel 5.4. Dari tabel 5.4 tersebut dapat diketahui bahwa algoritma dimulai dari inisialisasi *month* dan *year*. Selanjutya dilakukan pemuatan data laporan dari database. Hasil pemuatan data selanjutanya akan ditampilkan berdasarkan *month* dan *year*.

Tabel 5.4 *Pseudocode* Melihat Laporan Bulanan

No	Pseudocode
1.	START
2.	INIT kecamatan as sended kecamatan
3.	INIT desa as sended desa
4.	INIT month as sended bulan
5.	INIT year as sended tahun
6.	READ laporan from database
7.	DISPLAY laporan
8.	END

4.4.5 Melihat Rekap Laporan Bulanan

Subbab ini berisi rancangan algoritma dari sebuah fungsi yang disediakan oleh sistem untuk melihat rekap laporan bulanan. Perancangan algoritma ini ditunjukkan oleh tabel 5.5. Berdasarkan tabel 5.5 tersebut dapat diketahui bahwa algoritma dimulai dari inisialisasi *month* dan *year*. Selanjutya dilakukan pemuatan data laporan dari database. Hasil pemuatan data selanjutanya akan ditampilkan berdasarkan *month* dan *year*.

Tabel 5.5 *Pseudocode* Melihat Rekap Laporan Bulanan

No	Pseudocode
1.	START
2.	INIT month as sended bulan
3.	INIT year as sended tahun
4.	READ laporan from database
5.	DISPLAY laporan
6.	END

4.5 Perancangan Antarmuka

Pada subbab ini berisi beberapa rancangan tampilan antarmuka pengguna sistem informasi yang akan dikembangkan. Beberapa rancangan antarmuka pengguna yang dibuat pada subbab ini meliputi antarmuka login, antarmuka formulir pengajuan dokumen kependudukan dan antar muka daftar pengajuan dokumen kependudukan.

4.5.1 Antarmuka Login

Gambar 5.9 merupakan rancangan antarmuka login yang dapat dilihat oleh pengguna pada saat akan melakukan autentifikasi sistem. Pada antarmuka login ini terdapat beberapa komponen yaitu logo, *username*, *password* dan *sign in*. Komponen username merupakan kolom untuk memasukkan nama atau identitas pengguna. Komponen password merupakan kolom untuk memasukkan *password* pengguna.

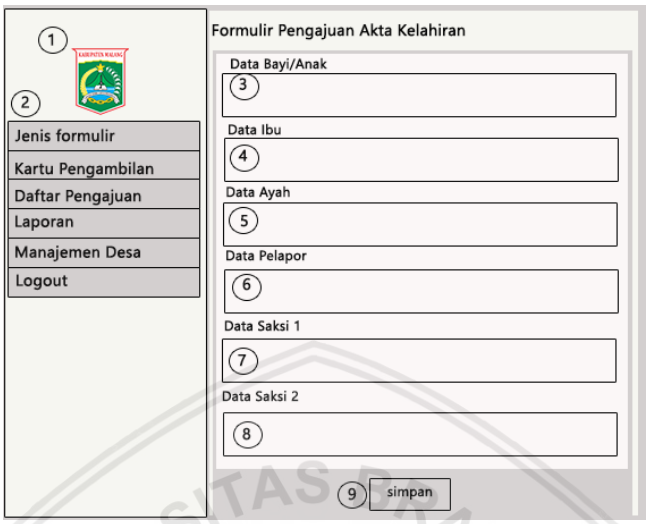
Gambar 5.10 Antarmuka Login

Pada gambar diatas terdapat beberapa nomor yang masing – masing menunjukkan komponen yang ada pada halaman login. Nomor 1 menunjukkan logo atau gambar. Sedangkan no 2, dan 3 menunjukkan kolom yang digunakan untuk memasukkan identitas pengguna ketika ingin masuk ke dalam sistem atau login. Untuk no 4 merupakan tombol submit untuk memulai proses login.

4.5.2 Antarmuka Formulir Pengajuan

Rancangan antarmuka formulir pengajuan ditujukkan oleh gambar 5.10. Antarmuka formulir login ini dapat dilihat ketika aktor ingin memasukkan pengajuan akta kaelahiran. Pada gambar 5.10 tersebut dapat diketahui bahwa antarmuka formulir pengajuan memiliki beberapa komponen diantaranya adalah *sidebar* dan kolom untuk memasukkan data terkait pengajuan dokumen kependudukan. Bagian *sidebar* terdiri dari logo, jenis formulir, kartu

pengambilan, daftar pengajuan, laporan, manajemen desa serta logout. Sedangkan untuk form yang harus diisikan oleh pengguna ada data bayi/anak, data ibu, data ayah, data pelapor, data saksi 1 dan saksi 2.

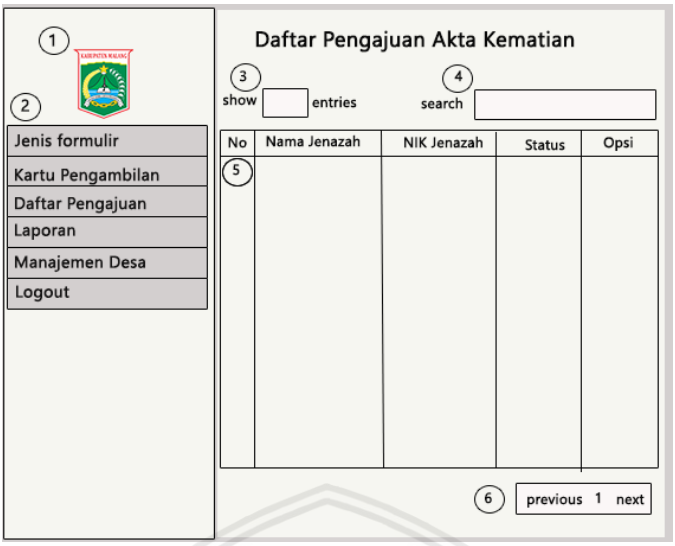


Gambar 5.11 Antarmuka Formulir Pengajuan

Gambar diatas merupana antarmuka dari formulir pengajuan akta kelahiran. Pada gambar tersebut terdapat beberapa penomoran yang masing – masing menujukkan komponen dari halaman formulir pengajuan. No 1 menunjukkan logo dari kabupaten malang. Untuk no 2 menunjukkan *sidebar* dimana *sidebar* ini memiliki beberapa komponen yaitu jenis formulir, kartu pengambilan, daftar pengajuan, laporan, manjemen data, dan logout yang digunakan untuk keluar dari sistem. Sedangkan no 3, 4, 5, 6 , 7, dan 8 merupakan kolom formulir yang harus diisikan oleh pengguna desa/kelurahan ketika ingin melakukan pengajuan pendaftran dokumen kependudukan. No 3 menunjukkan kolom untuk mengisikan data terkait identitas bayi/anak. No 4 merupakan kolom untuk mengisikan data terkait identitas ibu. No 5 merupakan kolom untuk mengisikan data terkait identitas ayah. No 6 merupakan kolom untuk mengisikan data terkait identitas pelapor. Dan untuk no 7 serta 8 merupakan kolom untuk mengisikan data terkait saksi 1 serta saksi 2. Tombol submit untuk menyimpan data yang telah diisikan ditujukkan oleh no 9.

4.5.3 Antarmuka Daftar Pengajuan

Gambar 5.11 menujukkan rancangan antarmuka daftar pengajuan akta kelahiran yang dapat dilihat oleh aktor dispenduk pada saat ingin melakukan validasi pengajuan. Di dalam antarmuka daftar pengajuan ini terdapat beberapa komponen diantaranya *sidebar*, dan tabel yang berisi daftar pengajuan dokumen kependudukan. Pada sidebar terdapat logo, jenis formulir, kartu pengambilan, daftar pengajuan, laporan, manajemen data serta logout. Pada tabel daftar pengajuan dokumen terdapat *button* opsi yang digunakan untuk melihat detail dari pengajuan dokumen kependudukan serta terdapat kolom status untuk mengetahui status pengajuan akta kelahiran.

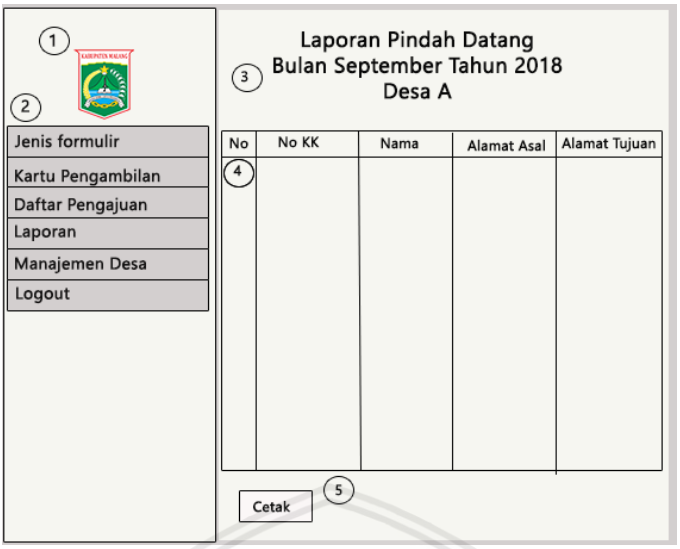


Gambar 5.12 Antarmuka Daftar Pengajuan

Gambar diatas merupakan antarmuka dari daftar pengajuan akta kematian. Pada gambar tersebut terdapat beberapa nomor yang menunjukkan komponen – komponen dari halaman daftar pengajuan akta kematian. No 1 merupakan logo dari kabupaten malang. Untuk no 2 merupakan *sidebar* yang terdiri dari jenis formulir, kartu pengambilan, daftar pengajuan, laporan, manajemen desa serta logout. No 3 merupakan kolom yang menunjukkan jumlah data yang ditampilkan pada tabel 5. Pada tabel 5 ini terdapat 5 jenis data yang akan ditampilkan. Data tersebut yaitu no yang menunjukkan urutan data, nama jenazah, nik jenazah, status serta opsi. No 4 merupakan kolom *search* yang berfungsi untuk mempercepat dalam pencarian data. Sedangkan no 6 merupakan *paging* yang menunjukkan jumlah halaman yang dimiliki oleh daftar pengajuan tersebut.

4.5.4 Antarmuka Laporan Bulanan

Rancangan antarmuka formulir pengajuan ditujukkan oleh gambar 5.12. Antarmuka laporan bulanan ini dapat dilihat ketika aktor telah memasukkan bulan dan tahun yang diinginkan. Pada gambar 5.12 tersebut dapat diketahui bahwa antarmuka formulir pengajuan memiliki beberapa komponen diantaranya adalah *sidebar* dan kolom yang berisi data laporan berdsarkan bulan dan tahun tertentu. Bagian *sidebar* terdiri dari logo, jenis formulir, kartu pengambilan, daftar pengajuan, laporan, manajemen desa serta logout.



Gambar 5.13 Antarmuka Laporan Bulanan

Pada antarmuka laporan bulanan ini terdapat 5 komponen. No 1 merupakan logo dari kabupaten malang. No 2 merupakan *sidebar* yang memiliki komponen jenis formulir, kartu pengambilan, daftar pengajuan, laporan, manajemen desa dan logout untuk keluar dari sistem. No 3 merupakan *kop* dari laporan dimana pada kop ini terdapan nama bulan, tahun serta nama desa sesuai dengan identitas pengguna yang sedang login. Untuk tabel yang ditunjukkan no 4 merupakan tabel yang berisi data sesuai dengan bulan, tahun dan desa tertentu. Untuk no 5 menunjukkan tombol cetak yang digunakan untuk mencetak data yang telah ditampilkan.

BAB 6 IMPLEMENTASI

Bab ini menjelaskan mengenai kebutuhan implementasi dan hasil implementasi dari sistem informasi yang dikembangkan. Adapun beberapa hal yang dijelaskan pada bab ini meliputi spesifiksi lingkungan, bebrapa kode program, dan beberapa gambar antarmuka pengguna. Implementasi yang dilakukan berdasarkan pada hasil perancangan yang telah dijelaskan pada bab sebelumnya.

6.1 Spesifikasi Lingkungan Implementasi

Subbab ini menjelaskan mengenai spesifikasi perangkat keras dan perangkat lunak yang digunakan selama proses pengambangan sistem informasi. Perangkat keras yang digunakan dalam proses pengembangan sistem informasi ditunjukkan pada tabel 6.1. Sedangkan spesifikasi perangkat lunak yang digunakan dalam pengembangan sistem informasi ditunjukkan pada tabel 6.2.

Tabel 6.1 Spesifikasi Perangkat Keras

Unit Komputasi	HP
CPU	Core(TM) i3-3217U 1.80 GHz
Kapasitas RAM	6.00 GB
Kapasitas Penyimpanan	465.76 GB
Kartu Grafis	Intel(R) HD Graphics 4000
Resolusi Layar	1366 x 768 pixels

Tabel 6.2 Spesifikasi Perangkat Lunak

Sistem Operasi	Windows 8 Pro 64-bit
Web Server	Apache 2.4.23
DBMS	MariaDB 10.1.16
Bahasa Pemrograman	PHP 7.0.9
Editor Kode Program	Sublime Text 3
Perangkat lunak pendukung	Xampp Google chrome Astah PowerDesigner Bizagi Photoshop
Library pendukung	Bootstrap, Jquery, database, session, upload, datetimepicker, chart

Tabel 6.3 Spesifikasi Minimal Lingkungan *Deployment*

Kapasitas RAM	64 MB
Kapasitas memori	750 MB

6.2 Implementasi Algoritme

Subbab ini berisi beberapa kode program yang telah dibuat berdasarkan algoritme dalam *pseudocode* yang dibuat pada tahap perancangan. Ada lima alrgoritma yang akan dijelaskan pada bab ini, yaitu mengajukan akta kematian, menambah kartu pengambilan akta kematian, mencetak kartu pengambilan akta kematian, melihat laporan bulanan dan melihat rekap laporan bulanan. Masing – masing dari kode program akan dijelaskan pada tabel 6.4, 6.5, 6.6, 6.7 dan 6.8.

6.2.1 Mengajukan akta kematian

Kode program yang terdapat pada tabel 6.4 merupakan implementasi algoritme dari fungsi mengajukan akta kematian yang disediakan oleh sistem. Fungsi ini digunakan untuk memenuhi kebutuhan pengguna terkait pengisian formulir pengajuan akta kematian. Implementasi algoritme ini dilakukan berdasarkan pada rancangan algoritme mengajukan akta kematian yang telah dibuat sebelumnya.

Tabel 6.4 Implementasi Algoritme Mengajukan Akta Kematian

No	Kode Program
1	`public function insert_kematian(){`
2	`$cek = $this->db->query("SELECT * FROM penduduk where`
3	`nik='".$this->input->post('nik_jenazah')."'")->num_rows();`
4	`if ($cek==1) { //proses mengingatkan data sudah ada`
5	`$data_jenazah= array(`
6	`"nik_jenazah"=>$this->input->post('nik_jenazah'),`
7	`"nama_jenazah"=>$this->input->post('nama_jenazah'),`
8	`"id_jk_jenazah"=>$this->input->post('jk_jenazah'),`
9	`"tanggal_lahir_jenazah"=>$this->input->post('tanggal_jenazah'),`
10	`"umur_jenazah"=>$this->input->post('umur_jenazah'),`
11	`"tempat_lahir_jenazah"=>$this->input->post('tempat_lahir'),`
12	`"id_agama_jenazah"=>$this->input->post('agama_jenazah'),`
13	`"pekerjaan_jenazah"=>$this->input->post('pekerjaan_jenazah'),`
14	`"alamat_jenazah"=>$this->input->post('alamat_jenazah'),`
15	`"anak_ke_jenazah"=>$this->input-`

```
    >post('anak_ke'),
16                  "tanggal_mati_jenazah"=>$this->input-
    >post('tanggal_mati'),
17                  "pukul_jenazah"=>$this->input-
    >post('pukul'),
18                  "sebab_jenazah"=>$this->input-
    >post('sebab'),
19                  "tempat_jenazah"=>$this->input-
    >post('tempat_mati'),
20                  "menerangkan_jenazah"=>$this->input-
    >post('menerangkan'),
21                  "id_prov_jenazah"=>$this->input-
    >post('propinsi-jenazah'),
22                  "id_kota_jenazah"=>$this->input-
    >post('kab/kota-jenazah'),
23                  "id_kecamatan_jenazah"=>$this->input-
    >post('kecamatan-jenazah'),
24                  "id_desa_jenazah"=>$this->input-
    >post('desa/kelurahan-jenazah')

25  );


26                  }else{
27                      echo "<script>alert('NIK
    salah');history.go(-1) </script>";
28                  }

29              $cekibu = $this->db->query("SELECT * FROM
    penduduk where nik='".$this->input->post('nik_ibu')."'")-
    >num_rows();
30              if ($cekibu==1) { //proses mengingatkan data
    sudah ada
31                      $data_ibu = array(
32                          "nik_ibu"=>$this->input-
    >post('nik_ibu'),
33                          "nama_ibu"=>$this->input-
    >post('nama_ibu'),
34                          "tgl_lahir_ibu"=>$this-
    >input->post('tanggal_ibu'),
35                          "umur_ibu"=>$this->input-
    >post('umur_ibu'),
36                          "pekerjaan_ibu"=>$this-
    >input->post('pekerjaan_ibu'),
37                          "alamat_ibu"=>$this->input-
    >post('alamat_ibu'),

38          "id_kewarganegaraan_ibu"=>$this->input-
    >post('kewarganegaraan_ibu'),
39                          "id_prov_ibu"=>$this->input-
    >post('propinsi-ibu'),
40                          "id_kota_ibu"=>$this->input-
    >post('kab/kota-ibu'),
41                          "id_kecamatan_ibu"=>$this-
    >input->post('kecamatan-ibu'),
42                          "id_desa_ibu"=>$this->input-
    >post('desa/kelurahan-ibu')

43                      );
```

```
44              }else{
45              echo "<script>alert('NIK ibu
    salah');history.go(-1) </script>";
46              }
47              $cekayah = $this->db->query("SELECT * FROM
    penduduk where nik='".$this->input->post('nik_ayah')."'")-
    >num_rows();
48               if ($cekayah==1) { //proses mengingatkan data
    sudah ada
49              $data_ayah=array(
50                      "nik_ayah"=>$this->input-
51  >post('nik_ayah'),
                        "nama_ayah"=>$this->input-
52  >post('nama_ayah'),
                        "tgl_lahir_ayah"=>$this->input-
53  >post('tanggal_ayah'),
                        "umur_ayah"=>$this->input-
54  >post('umur_ayah'),
                        "pekerjaan_ayah"=>$this->input-
55  >post('pekerjaan_ayah'),
                        "alamat_ayah"=>$this->input-
56  >post('alamat_ayah'),
                        "id_kewarganegaraan_ayah"=>$this->input-
57  >post('kewarganegaraan_ayah'),
                        "id_prov_ayah"=>$this->input-
58  >post('propinsi-ayah'),
                        "id_kota_ayah"=>$this->input-
59  >post('kab/kota-ayah'),
                        "id_kecamatan_ayah"=>$this->input-
60  >post('kecamatan-ayah'),
                        "id_desa_ayah"=>$this->input-
61  >post('desa/kelurahan-ayah')
                        );
62
63              }else{
                        echo "<script>alert('NIK ayah
64  salah');history.go(-1) </script>";
                }
65
                $cekpelapor = $this->db->query("SELECT * FROM
    penduduk where nik='".$this->input-
66  >post('nik_pelapor')."'")->num_rows();
67               if ($cekpelapor==1) {
68              $data_pelapor=array(
                        "nik_pelapor"=>$this->input-
69  >post('nik_pelapor'),
                        "nama_pelapor"=>$this->input-
70  >post('nama_pelapor'),
                        "tgl_lahir_pelapor"=>$this->input-
71  >post('tanggal_pelapor'),
                        "umur_pelapor"=>$this->input-
72  >post('umur_pelapor'),
                        "pekerjaan_pelapor"=>$this->input-
73  >post('pekerjaan_pelapor'),
                        "alamat_pelapor"=>$this->input-
74  >post('alamat_pelapor'),
                        "id_prov_pelapor"=>$this->input-
75  >post('propinsi-pelapor'),
```

```
                    "id_kota_pelapor"=>$this->input-
76  >post('kab/kota-pelapor'),
                    "id_kecamatan_pelapor"=>$this->input-
77  >post('kecamatan-pelapor'),
                    "id_desa_pelapor"=>$this->input-
78  >post('desa/kelurahan-pelapor')
79                  //"id_kewarganegaraan"=>$this->input-
    >post('kewarganegaraan_pelapor'),
80
                    );
81
82        }else{
                echo "<script>alert('NIK pelapor
83  salah');history.go(-1) </script>";
84        }
    $ceksaksi1 = $this->db->query("SELECT * FROM penduduk
85  where nik='".$this->input->post('nik_saksi1')."'")-
86  >num_rows();
87              if ($ceksaksi1==1) {
                $data_saksi1=array(
88                  "nik_saksi1"=>$this->input-
    >post('nik_saksi1'),
89                  "nama_saksi1"=>$this->input-
    >post('nama_saksi1'),
90                  "umur_saksi1"=>$this->input-
    >post('umur_saksi1'),
91                  "pekerjaan_saksi1"=>$this->input-
    >post('pekerjaan_saksi1'),
92                  "alamat_saksi1"=>$this->input-
    >post('alamat_saksi1'),
93                  "id_prov_saksi1"=>$this->input-
    >post('propinsi-saksi1'),
94                      "id_kota_saksi1"=>$this->input-
    >post('kab/kota-saksi1'),
95                      "id_kecamatan_saksi1"=>$this-
    >input->post('kecamatan-saksi1'),
96                      "id_desa_saksi1"=>$this->input-
    >post('desa/kelurahan-saksi1')
97                  );
98
        }else{
99              echo "<script>alert('NIK saksi1
100 salah');history.go(-1) </script>";
        }
101 $ceksaksi2 = $this->db->query("SELECT * FROM penduduk
102 where nik='".$this->input->post('nik_saksi2')."'")-
103 >num_rows();
              if ($ceksaksi2==1) {
104             $data_saksi2=array(
                    "nik_saksi2"=>$this->input-
105 >post('nik_saksi2'),
                    "nama_saksi2"=>$this->input-
106 >post('nama_saksi2'),
                    "umur_saksi2"=>$this->input-
107 >post('umur_saksi2'),
                    "pekerjaan_saksi2"=>$this->input-
108 >post('pekerjaan_saksi2'),
                    "alamat_saksi2"=>$this->input-
109 >post('alamat_saksi2'),
```

```
                    "id_prov_saksi2"=>$this->input->post('propinsi-saksi2'),
                    "id_kota_saksi2"=>$this->input->post('kab/kota-saksi2'),
                    "id_kecamatan_saksi2"=>$this->input->post('kecamatan-saksi2'),
                    "id_desa_saksi2"=>$this->input->post('desa/kelurahan-saksi2')
                );

      }else{
            echo "<script>alert('NIK saksi2 salah');history.go(-1) </script>";
      }
            $id_jenazah=$this->My_model->insert_jenazah($data_jenazah);
            $id_ibu=$this->My_model->insert_ibu($data_ibu);
            $id_ayah=$this->My_model->insert_ayah($data_ayah);
            $id_pelapor=$this->My_model->insert_pelapor($data_pelapor);
            $id_saksi1=$this->My_model->insert_saksi1($data_saksi1);
            $id_saksi2=$this->My_model->insert_saksi2($data_saksi2);
            $data_kematian=array(
                    "id_prov"=>$this->input->post('propinsi'),
                    "id_kota"=>$this->input->post('kab/kota'),
                    "id_kecamatan"=>$this->input->post('kecamatan'),
                    "id_desa"=>$this->input->post('desa/kelurahan'),
                    "id_jenazah"=>$id_jenazah,
                    "id_ibu"=>$id_ibu,
                    "id_ayah"=>$id_ayah,
                    "id_pelapor"=>$id_pelapor,
                    "id_saksi1"=>$id_saksi1,
                    "id_saksi2"=>$id_saksi2,
                    'tanggal'=>date('Y-m-d'),
                    "status"=>0

                    );

            $this->My_model->tambah_kematian($data_kematian);


            echo "<script> alert('Data berhasil Dientry');</script>";
            $this->load->model('dispenduk_model');
            $daftar_kematian=$data_kematian;
            $data['daftar_kematian']=$this->dispenduk_model->kematian();
            $this->load->view('Desa/daftarKematian',$data);
```

	}

6.2.2 Menambah Kartu Pengambilan Akta Kematian

Kode program dari fungsi menambah daftar pengambilan akta kematian ditujunkkan pada tabel 6.5. Fungsi menambah kartu pengambilan akta kematian ini bertujuan untuk menampilkan daftar kartu pengambilan yang telah dibuat. Kode program ini dibuat berdasarkan pada rancangan algoritme menambah kartu pengambilan kematian yang telah dibuat pada tahap perancangan. Berikut adalah kode program menambah kartu pengambilan akta kematian.

Tabel 6.5 Implementasi Algoritme Menambah Kartu Pengambilan Akta Kematian

No	Kode Program
1	`public function  insert_pengambilanKematian(){`
2	`      $data = array('nama' =>$this->input->post('nama') ,`
3	`                                    'tanggal_kematian' =>date('Y-m-d',strtotime($this->input->post('tanggal_kematian'))),`
4	`                                    'nama_ayah'=>$this->input->post('nama_ayah'),`
5	`                                    'nama_ibu'=>$this->input->post('nama_ibu'),`
6	`                                    'tgl_selesai'=>date('Y-m-d',strtotime($this->input->post('date'))),`
7	`                                    'tgl_masuk'=>date('Y-m-d'),`
8	`                                    "status"=>0`
9	`       );`
10	`            $this->My_model->insert_pengambilanKematian($data);`
11	`      redirect('Desa/pengambilan/data_kematian','refresh');`
12	`}`

6.2.3 Mencetak Kartu Pengambilan Akta Kematian

Kode program yang terdapat pada tabel 6.6 merupakan implementasi algoritme dari mencetak kartu pengambilan akta kematian yang disediakan oleh sistem. Fungsi ini digunakan untuk memenuhi kebutuhan pengguna untuk mencetak kartu pengambilan akta kelahiran dimana kartu pengambilan ini digunakan untuk mengambil akta kematian yang telah diajukkan. Implementasi algoritme ini dilakukan berdasarkan pada rancangan algoritme mencetak kartu pengambilan akta kematian yang telah dibuat sebelumnya.

Tabel 6.6 Implementasi Algortime Mencetak Kartu Pengambilan Akta Kematian

No	Kode Program
1	`public function cetak_pengambilanKematian($bukti){`
2	`    $data['data']=$this->db->get_where('bukti_akta_kematian',['id_bukti'=>$bukti])->row();`

```
3       $this->load-
>view('Desa/cetak_pengambilanKematian',$data);
4     }
```

No	Kode Program
1	`public function data_kematian(){`
2	`$query=$this->db->query("SELECT * FROM bukti_akta_kematian ORDER BY id_bukti DESC");`
3	`return $query->result();`
4	`}`

6.2.4 Melihat Laporan Bulanan

Kode program yang terdapat pada tabel 6.7 merupakan implementasi algoritme dari fungsi melihat laporan bulanan yang disediakan oleh sistem. Fungsi ini digunakan untuk memenuhi kebutuhan pengguna terkait laporan bulanan. Implementasi algoritme ini dilakukan berdasarkan pada rancangan algoritme yang telah dibuat sebelumnya.

Tabel 6.7 Implementasi Algoritme Melihat Laporan Bulanan

No	Kode Program
1	`public function reportKelahiran(){`
2	`$bln=$this->input->post('bulan');`
3	`$thn=$this->input->post('tahun');`
4	`$data['x']=$this->My_model->reportKelahiran($bln,$thn)->result();`
5	`$this->load->view('Desa/tampil_laporanKelahiran',$data);`
6	`}`
7	`public function reportKematian(){`
8	`$bln=$this->input->post('bulan');`
9	`$thn=$this->input->post('tahun');`
10	`$data['x']=$this->My_model->reportKematian($bln,$thn)->result();`
11	`$this->load->view('Desa/tampil_laporanKematian',$data);`
12	`}`
13	`public function reportSpDatang(){`
14	
15	`$bln=$this->input->post('bulan');`
16	`$thn=$this->input->post('tahun');`
17	`$data['x']=$this->My_model->reportSpDatang($bln,$thn)->result();`
18	`$this->load->view('Desa/tampil_laporanSpDatang',$data);`

```
19  }

20  public function reportSpKeluar(){
21
22        $bln=$this->input->post('bulan');
          $thn=$this->input->post('tahun');
23        $data['x']=$this->My_model-
    >reportSpKeluar($bln,$thn)->result();
24        $this->load-
    >view('Desa/tampil_laporanspKeluar',$data);
25
    }
26
27  public function reportKK(){
28
          $bln=$this->input->post('bulan');
29        $thn=$this->input->post('tahun');
          $data['x']=$this->My_model-
30  >reportSpKeluar($bln,$thn)->result();
          $this->load->view('Desa/tampil_laporanKK',$data);

    }
```

6.2.5 Melihat Rekap Laporan Bulanan

Kode program yang terdapat pada tabel 6.8 merupakan implementasi algoritme dari fungsi melihat rekap laporan bulanan yang disediakan oleh sistem. Fungsi ini digunakan untuk memenuhi kebutuhan pengguna terkait rekap laporan bulanan. Implementasi algoritme ini dilakukan berdasarkan pada rancangan algoritme yang telah dibuat sebelumnya. Beriku adalah potongan kode program dari fungsi melihat laporan bulanan.

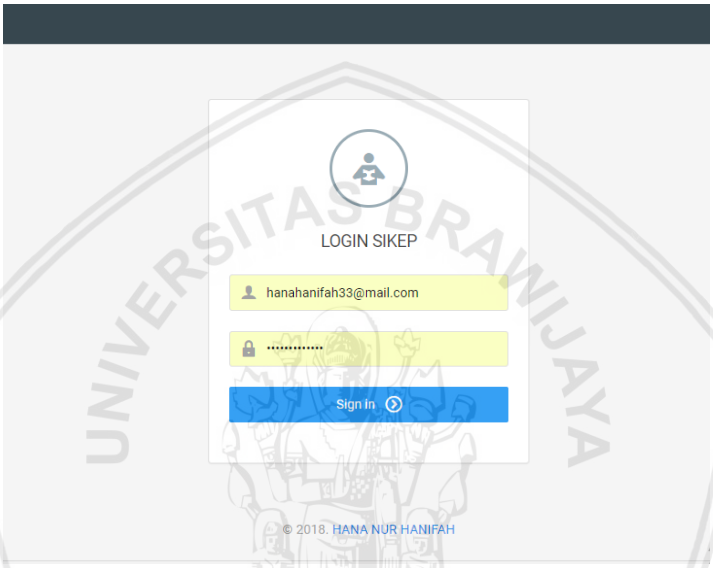
Tabel 6.8 Implementasi Rekap Laporan Bulanan

No	Kode program
1	`public function rekapkematian(){`
2	`$bln=$this->input->post('bulan');`
3	`$thn=$this->input->post('tahun');`
4	`$this->load->model('dispenduk_model');`
5	`$x['data'] = $this->dispenduk_model->rekapkematian($bln,$thn);`
6	`if ($x['data']) {`
7	`$this->load->view('Dispenduk/tampil_rekapkematian',$x);`
8	`}else{`
9	`$this->session->set_flashdata('error','Tidak ada data yang ditemukan');`
10	`redirect('Dispenduk/dispenduk/rekapkematian');`
11	`}`
12	`}`

6.3 Implementasi Antarmuka Pengguna

Subbab ini berisi tentang beberapa hasil implementasi antarmuka yang dijelaskan dalam bentuk potongan gambar. Ada beberapa antarmuka yang dijelaskan dalam bab ini yaitu antarmuka login, antarmuka formulir pengajuan, antarmuka daftar pengajuan, dan antarmuka laporan bulanan. Implementasi antarmuka ini dilakukan berdasarkan pada hasil perancangan antarmuka yang telah dilakukan sebelumnya.

6.3.1 Antarmuka Login



Gambar 6.1 Antarmuka Login

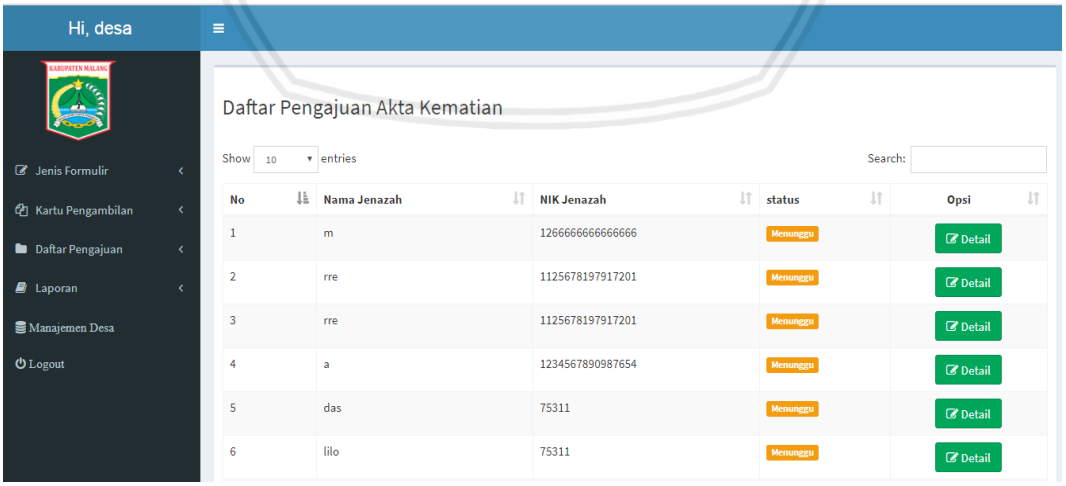
Antarmuka login ini akan ditampilkan ketika pengguna akan menggunakan sistem dan harus melakukan autentifikasi terlebih dahulu. Berdasarkan gambar 6.1 antarmuka login memeliki beberapa komponen yaitu judul atau nama sistem, kolom untuk memasukkan username, kolom untuk memasukkan password serta tombol submit untuk memproses masukkan dari pengguna. Ada tiga aktor yang dapat terindetikasi pada form login ini yaitu, aktor desa/kelurahan, aktor dispenduk dan aktor kecamatan.

6.3.2 Antarmuka Formulir Pengajuan

Gambar 6.2 Antarmuka Formulir Pengajuan

Antarmuka formulir pengajuan kelahiran ini akan ditampilkan ketika pengguna memilih fungsi jenis formulir kemudian memilih akta kelahiran. Pada antarmuka formulir pengajuan akta kelahiran ini terdapat beberapa kolom yang harus diisikan oleh pengguna. Kolom – kolom ini dikelompokkan menjadi beberapa bagian yaitu kolom mengenai identitas bayi/anak, kolom mengenai identitas ayah, kolom mengenai identitas ibu, kolom mengenai identitas pelapor, kolom mengenai identitas saksi 1 dan kolom mengenai identitas saksi2. Pada antarmuka formulir pengajuan akta kelahiran ini juga terdapat tombol *submit* untuk menyimpan data yang telah dimasukkan oleh pengguna ke dalam *database*. Tombol *submit* ini terlitak pada bagian bawah formulir.

6.3.3 Antarmuka Daftar Pengajuan

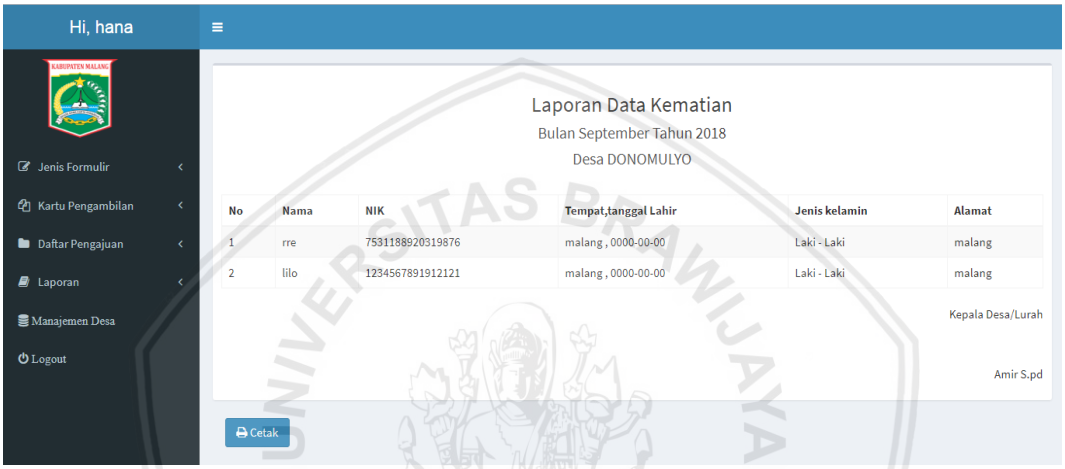


Gambar 6.3 Antarmuka Daftar Pengajuan

Antarmuka daftar pengajuan ini muncul ketika pengguna memilih menu daftar pengajuan. Ada 5 submenu yang dapat dipilih oleh pengguna yaitu, akta

kelahiran, akta kematian, kartu keluarga, surat pindah datang dan surat pindah keluar. Gambar 6.3 merupakan potongan gambar untuk antarmuka daftar pengajuan akta kematian. Pada antarmuka ini terdapat tabel yang berisikan data pengajuan akta kematian yang telah dimasukkan sebelumnya. Kolom status berguna untuk memberikan informasi terkait status pengajuan yang dilakukan oleh pengguna. Ada 3 status yang dapat muncul pada kolom ini yaitu, menunggu, disetujui dan tidak disetujui. Sedangkan kolom opsi memungkinakan pengguna untuk melihat detail dari setiap pengajuan yang ada.

6.3.4 Antarmuka Laporan Bulanan



Gambar 6.4 Antarmuka Laporan Bulanan

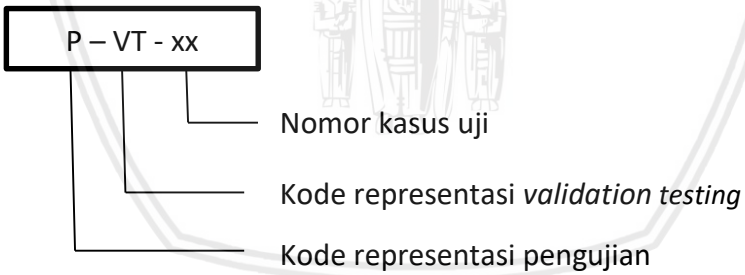
Antarmuka laporan bulanan ini muncul ketika pengguna memilih menu laporan dan memasukkan bulan serta tahun yang diinginkan. Gambar 6.4 merupakan potongan gambar untuk antarmukan laporan bulanan akta kematian. Berdasarkan gambar tersebut dapat diketahui bahwa terdapat beberapa data yang akan ditampilkan oleh sistem yaitu kolom nama jenazah, NIK jenazah, tempat, tanggal lahir, jenis kelamin serta alamat. Data – data ini akan ditampilkan sesuai dengan bulan, tahun dan nama desa tertentu. Pada antarmuka laporan bulanan ini terdapat tombol cetak untuk mencetak dokumen sesuai dengan apa yang ditampilkan oleh sistem.

BAB 7 PENGUJIAN

Bab ini menjelaskan mengenai pengujian yang dilakukan pada sistem informasi pendataan dan pelaporan data kependudukan. Pengujian yang dilakukan adalah dengan menggunakan metode *validation testing*, *compatibility testing* dan *usability testing* . *Validation testing* dilakukan untuk menguji persyaratan fungsional sistem, *compatibility testing* dilakukan sebagai dari persyaratan non fungsional sistem, dan *usability testing* dilakukan untuk mengetahui seberapa mudah antarmuka pengguna digunakan.

7.1 Pengujian *Validation Testing*

Validation testing merupakan salah satu teknik pengujian yang ada pada metode *black box testing*. Pengujian validasi ini dilakukan untuk menguji apakah sistem dapat berjalan sesuai dengan harapan pengguna. Menurut (Pressman, 2010) pengujian validasi berbasis skenario pada *use case* dapat mengetahui kesalahan interaksi pada sistem. Berdasarkan hal tersebut maka pengujian validasi akan dilakukan berdasarkan beberapa alur *use case* yang telah dibuat sebelumnya. Terdapat lima alur *use case* yang akan diuji pada pengujian validasi ini yaitu, mengajukan akta kematian, menambah kartu pengambilan akta kematian, mencetak kartu pengambilan akta kematian, melihat laporan bulanan serta melihat rekap laporan bulanan. Dalam pengujian validasi ini terdapat pemberian kode sebagai penanda *test case* yang dirancang. Berikut adalah ketentuan kode pengujian.



Gambar 7.1 Pengkodean Pengujian *Validation Testing*

Pada pengujian *validation testing* dibuat sebuah *test case* dengan format penulisan seperti yang ada pada gambar 2.16 akan tetapi ada dua penambahan kolom yaitu, kode kebutuhan dan status. Penambahan kode kebutuhan dilakukan karena pengujian *validation testing* bertujuan untuk memastikan bahwa fungsi atau karakteristik sesuai dengan kebutuhan. Jadi, penambahan kode kebutuhan dilakukan untuk memudahkan pemetaan pengujian dengan kebutuhan sistem. Sedangkan penambahan kolom status bertujuan untuk memudahkan dalam memahami hasil pengujian. Berikut adalah beberapa hasil dari pengujian *validation testing*

Tabel 7.1 Pengujian *Validation Testing* Menambah Pengajuan Akta Kematian

Test case name	P – VT - 01
Purpose of test	Pengujian dilakukan untuk memastikan bahwa fungsi mengajukkan akta kematian dapat menambah data pengajuan akta kematian
Test attribute	
Test focus	Validation
Kode kebutuhan	SPPDK-F-04
Test process	1. Penguji mengakses menu jenis formulir. 2. Penguji memilih menu akta kematian. 3. Penguji mengisi formulir pengajuan akta kematian sesuai dengan format yang ditentukan. 4. Penguji menekan tombol simpan. 5. Sistem menampilkan pesan data berhasil di-*entry*
Test result	Sistem menampilkan pesan berhasil di-*entry*
Status	Valid

Berdasarkan tabel hasil pengujian diatas dapat diketahui bahwa pengujian *validation testing* terhadap fungsi menambah pengajuan akta kematian dapat berjalan sesuai dengan *use case scenario* yang dibuat serta sesuai dengan kebutuhan yang ditetapkan sebelumnya. Hal ini dapat diketahui dengan mudah melalui kolom status, dimana pada kolom tersebut bernilai valid. Mengenai bagaimana alur dari pengujian yang dilakukan dapat diketahui melalui kolom *test process*. *Test process* pengujian *validation testing* untuk fungsi menambah pengajuan akta kematian dimulai dari penguji mengakses menu jenis formulir dan berakhir dengan sistem menampilkan pesan bahwa data berhasil di-*entry*. Untuk kolom *test result* menunjukkan hasil akhir yang didapat dari pengujian menambah akta kematian.

Tabel 7.2 Pengujian *Validation Testing* Menambah Pengajuan Akta Kematian : Alternatif 1 (A1)

Test case name	P – VT – 02
Purpose of test	Pengujian dilakukan untuk memastikan bahwa aktor tidak dapat menambah pengajuan akta kematian jika terdapat kolom input yang tidak di isi.
Test attribute	
Test focus	Validation
Kode kebutuhan	SPPDK-F-04
Test process	1. Penguji mengakses menu jenis formulir. 2. Penguji memilih menu akta kematian. 3. Penguji mengosongkan beberapa kolom input formulir pengajuan akta kematian. 4. Penguji menekan tombol simpan. 5. Sistem menampilkan input data bagian formulir yang

	tidak boleh kosong
Test result	Sistem menampilkan input data bagian formulir yang tidak boleh kosong
Status	Valid

Berdasarkan tabel hasil pengujian diatas dapat diketahui bahwa pengujian *validation testing* terhadap fungsi menambah pengajuan akta kematian alternatif 1 dapat berjalan sesuai dengan *use case scenario* yang dibuat serta sesuai dengan kebutuhan yang ditetapkan sebelumnya. Hal ini dapat diketahui melalui kolom status, dimana pada kolom tersebut bernilai valid. Mengenai bagaimana alur dari pengujian yang dilakukan dapat diketahui melalui kolom *test process*. *Test process* pengujian *validation testing* untuk fungsi menambah pengajuan akta kematian alternatif 1 dimulai dari penguji mengakses menu jenis formulir dan berakhir dengan sistem menampilkan input data bagian formulir yang tidak boleh kosong. Untuk kolom *test result* menunjukkan hasil akhir yang didapat dari pengujian menambah akta kematian.

Tabel 7.3 Pengujian *Validation Testing* Menambah Pengajuan Akta Kematian : Alternatif 2 (A2)

Test case name	P – VT - 03
Purpose of test	Pengujian dilakukan untuk memastikan bahwa aktor tidak dapat menambah pengajuan akta kematian jika terdapat kolom NIK yang diisi kurang dari 16 karakter.
Test attribute	
Test focus	Validation
Kode kebutuhan	SPPDK-F-04
Test process	1. Penguji mengakses menu jenis formulir. 2. Penguji memilih menu akta kematian. 3. Penguji mengisikan salah satu kolom NIK dengan panjang karakter 5. 4. Penguji menekan tombol simpan. 5. Sistem menampilkan input data bagian NIK yang harus diisi sebanyak 16 karakter.
Test result	Sistem menampilkan bagian NIK yang tidak sesuai dengan ketentuan (kurang dari 16 karakter).
Status	Valid

Berdasarkan tabel hasil pengujian diatas dapat diketahui bahwa pengujian *validation testing* terhadap fungsi menambah pengajuan akta kematian alternatif 2 dapat berjalan sesuai dengan *use case scenario* yang dibuat serta sesuai dengan kebutuhan yang ditetapkan sebelumnya. Hal ini dapat diketahui melalui kolom status, dimana pada kolom tersebut bernilai valid. Mengenai bagaimana alur dari pengujian yang dilakukan dapat diketahui melalui kolom *test process*. *Test process* pengujian *validation testing* untuk fungsi

menambah pengajuan akta kematian alternatif 2 dimulai dari penguji mengakses menu jenis formulir dan berakhir dengan sistem menampilkan input data bagian formulir yang tidak boleh kosong. Untuk kolom *test result* menunjukkan hasil akhir yang didapat dari pengujian menambah akta kematian.

Tabel 7.4 Pengujian *Validation Testing* Menambah Kartu Pengambilan

Test case name	P – VT - 04
Purpose of test	Pengujian dilakukan untuk memastikan bahwa fungsi kartu pengambilan akta kematian dapat menambah data kartu pengambilan akta kematian
Test attribute	
Test focus	Validation
Kode kebutuhan	SPDK – F - 17
Test process	1. Penguji memilih fungsi kartu pengambilan dan memilih fungsi tambah 2. Penguji memasukkan data terkait kartu pengambilan akta kematian sesuai dengan format yang ditentukan 3. Penguji menekan tombol simpan untuk menyimpan data kartu pengambilan akta kematian 4. Sistem menampilkan daftar kartu pengambilan akta kematian yang telah berhasil dimasukkan
Test result	Sistem menampilkan daftar kartu pengambilan akta kematian
Status	Valid

Berdasarkan tabel hasil pengujian diatas dapat diketahui bahwa pengujian *validation testing* terhadap fungsi menambah kartu pengambilan akta kematian dapat berjalan sesuai dengan *use case scenario* yang dibuat serta sesuai dengan kebutuhan yang ditetapkan sebelumnya. Hal ini dapat diketahui melalui kolom status, dimana pada kolom tersebut bernilai valid. Mengenai bagaimana alur dari pengujian yang dilakukan dapat diketahui melalui kolom *test process*. *Test process* pengujian *validation testing* untuk fungsi menambah kartu pengambilan akta kematian dimulai dari penguji memilih fungsi kartu pengambilan dan memilih fungsi tambah dan berakhir dengan sistem menampilkan daftar kartu pengambilan akta kematian. Untuk kolom *test result* menunjukkan hasil akhir yang didapat dari pengujian menambah kartu pengambilan akta kematian.

Tabel 7.5 Pengujian *Validation Testing* Mencetak Kartu Pengambilan

Test case name	P – VT - 05
Purpose of test	Pengujian dilakukan untuk memastikan bahwa fungsi mengajukkan akta kematian dapat menambah data kartu pengambilan akta kematian

Test attribute	
Test focus	*Validation*
Kode kebutuhan	SPPDK-F-15
Test process	1. Penguji memilih fungsi kartu pengambilan 2. Penguji memilih data kartu pengambilan akta kematian yang ingin dicetak 3. Penguji menekan tombol cetak 4. Sistem manampilkan *print preview* dari kartu pengambilan akta kematian yang akan dicetak
Test result	Sistem menampilkan *print preview* dari kartu pengambilan akta kematian yang akan dicetak
Status	Valid

Berdasarkan tabel hasil pengujian diatas dapat diketahui bahwa pengujian *validation testing* terhadap fungsi mencetak kartu pengambilan akta kematian dapat berjalan sesuai dengan *use case scenario* yang dibuat serta sesuai dengan kebutuhan yang ditetapkan sebelumnya. Hal ini dapat diketahui melalui kolom status, dimana pada kolom tersebut bernilai valid. Mengenai bagaimana alur dari pengujian yang dilakukan dapat diketahui melalui kolom *test process*. *Test process* pengujian *validation testing* untuk fungsi mencetak kartu pengambilan akta kematian dimulai dari penguji memilih fungsi kartu pengambilan dan berakhir dengan sistem manampilkan *print preview* dari kartu pengambilan akta kematian yang akan dicetak. Untuk kolom *test result* menunjukkan hasil akhir yang didapat dari pengujian mencetak kartu pengambilan akta kematian.

Tabel 7.6 Pengujian *Validation Testing* Melihat Laporan Bulanan

Test case name	P – VT - 06
Purpose of test	Pengujian dilakukan untuk memastikan bahwa fungsi laporan dapat menampilkan laporan data pengajuan akta kematian sesuai dengan bulan dan tanggal yang diinginkan
Test attribute	
Test focus	*Validation*
Kode kebutuhan	SPPDK-F-08
Test process	1. Penguji memilih fungsi laporan akta kematian 2. Penguji memasukkan bulan dan tahun yang diinginkan 3. Penguji menekan tombol tampil 4. Sistem manampilkan laporan data pengajuan akta kematian sesuai dengan bulan dan tahun tertentu
Test result	Sistem menampilkan laporan data pengajuan akta kematian sesuai dengan bulan dan tahun tertentu
Status	Valid

Berdasarkan tabel hasil pengujian diatas dapat diketahui bahwa pengujian *validation testing* terhadap fungsi melihat laporan bulanan dapat berjalan sesuai dengan *use case scenario* yang dibuat serta sesuai dengan kebutuhan yang ditetapkan sebelumnya. Hal ini dapat diketahui melalui kolom status, dimana pada kolom tersebut bernilai valid. Mengenai bagaimana alur dari pengujian yang dilakukan dapat diketahui melalui kolom *test process*. *Test process* pengujian *validation testing* untuk fungsi melihat laporan bulanan dimulai dari penguji memilih fungsi laporan akta kematian dan berakhir dengan sistem manampilkan laporan data pengajuan akta kematian sesuai dengan bulan dan tahun tertentu. Untuk kolom *test result* menunjukkan hasil akhir yang didapat dari pengujian melihat laporan bulanan.

Tabel 7.7 Pengujian *Validation Testing* Melihat Rekap Laporan

Test case name	P – VT - 07
Purpose of test	Pengujian dilakukan untuk memastikan bahwa fungsi laporan dapat menampilkan rekap jumlah data pengajuan akta kematian berdasarkan bulan dan tahun tertentu
Test attribute	
Test focus	*Validation*
Kode kebutuhan	SPPDK-F-08
Test process	1. Penguji memilih fungsi rekap laporan 2. Penguji memasukkan bulan dan tahun sesuai dengan yang diinginkan 3. Penguji menekan tombol tampil 4. Sistem manampilkan rekap jumlah data pengajuan akta kematian berdasarkan bulan dan tahun tertentu
Test result	Sistem menampilkan rekap data pengajuan akta kematian berdasarkan buln dan tahun tertentu
Status	Valid

Berdasarkan tabel hasil pengujian diatas dapat diketahui bahwa pengujian *validation testing* terhadap fungsi melihat rekap laporan dapat berjalan sesuai dengan *use case scenario* yang dibuat serta sesuai dengan kebutuhan yang ditetapkan sebelumnya. Hal ini dapat diketahui melalui kolom status, dimana pada kolom tersebut bernilai valid. Mengenai bagaimana alur dari pengujian yang dilakukan dapat diketahui melalui kolom *test process*. *Test process* pengujian *validation testing* untuk fungsi melihat rekap laporan dimulai dari penguji memilih fungsi rekap laporan dan berakhir dengan sistem menampilkan rekap jumlah data pengajuan akta kematian berdasarkan bulan dan tahun tertentu. Untuk kolom *test result* menunjukkan hasil akhir yang didapat dari pengujian mencetak kartu pengambilan akta kematian.

Berdasarkan hasil pengujian validation testing yang telah dilakukan dapat disimpulkan bahwa fungsi – fungsi menambah pengajuan akta kematian, menambah kartu pengambilan akta kematian, mencetak laporan kartu

pengambilan akta kematian, melihat laporan bulanan dan melihat rekap laporan dapat berjalan dengan baik dan sesuai dengan spesifikasi kebutuhan.

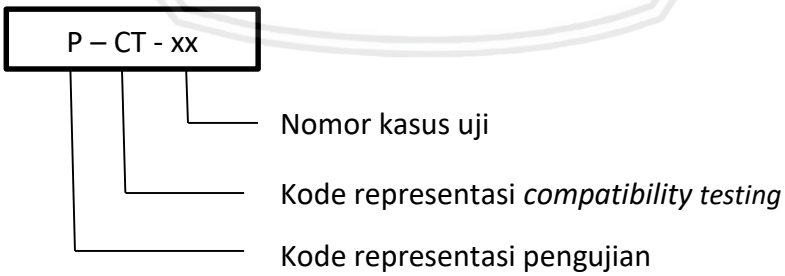
7.2 Pengujian *Compatibility Testing*

Pengujian *compatibility testing* ini dilakukan untuk mengetahui kesesuaian sistem yang akan dikembangkan dengan beberapa aplikasi peramban. Pengujian *compatibility testing* ini dilakukan dengan menggunakan *software SortSite* versi 5.30.822.0. Berdasarkan pada *software* yang digunakan terdapat beberapa kategori masalah untuk mengetahui tingkat kompatibilitas sistem. Kategori tersebut ditunjukkan pada tabel 7.1

Tabel 7.8 Kategori Masalah Kompatibilitas

	Tidak terjadi permasalahan pada konten atau fungsionalitas
	Terdapat konten atau fungsionalitas yang hilang
	Terdapat masalah mayor pada layout atau performa
	Terdapat masalah minor pada layout atau performa

Sistem informasi yang dikembangkan akan diuji pada beberapa peramban untuk mengetahui kompabilitas sistem. Beberapa peramban itu antara lain adalah *IE, Edge, Firefox , safari, opera, chrome, iOS* dan *Android*. Pada pengujian compatibility testing ini akan dibuat beberapa *test case* sesuai dengan format yang ada. Format *test case* yang digunakan pada pengujian *compatibility testing* ini sama dengan format pengujian yang digunakan pada pengujian *validation*, dimana ada dua penambahan kolom kode kebutuhan. Dalam pengujian *compatibility testing* ini juga terdapat pemberian kode sebagai penanda *test case* yang dirancang. Berikut adalah ketentuan kode pengujian.



Gambar 7.2 Pengkodeaan Pengujian *Compatibiliy Testing*

Tabel 7.9 *Test Case* Pengujian *Compability Testing* Pada *Internet Explorer*

Test case name	P – CT – 01
Purpose of test	Pengujian dilakukan untuk memastikan bahwa sistem dapat berjalan dengan baik pada peramban *internet*

	explorer versi 11
Test attribute	
Test focus	*Compatibility*
Kode kebutuhan	SPPDK – NF – 001
Test process	1. Penguji mengakses dan menjalankan sistem pada aplikasi *SortSite* dengan cara memasukkan url sistem pada menu bar *address.* 2. Penguji menekan tombol *check* untuk memulai proses *checking*. 3. *SortSite* memindai setiap halaman yang dimiliki oleh sistem. 4. Setiap halaman diperiksa untuk masalah kualitas dengan menggunakan *checkpoints,* kemudian *SortSite* memeriksa halaman yang terhubung dengan halaman tersebut. Selanjutnya *SortSite* akan memindai halaman baru yang ditemukan dan mengulang proses ini sampai akhir halaman. 5. *SortSite* menampilkan serangkaian tes validasi kompatibilitas yang diperoleh untuk mengungkapkan kesalahan yang dapat ditelusuri.
Test result	*SortSite* menampilkan serangkaian hasil tes kompabilitas sistem yang menunjukkan bahwa terdapat *major issues* karena adanya CSS yang tidak mendukung sehingga tampilan sistem tidak dapat berjalan dengan baik.

Tabel diatas merupakan *test case* yang digunakan pada pengujian *compatibility testing.* Pengujian ini bertujuan untuk memastikan bahwa sistem yang dibangun dapat berjalan dengan baik pada peramban *internet explorer* versi 11. Terdapat beberapa proses untuk melakukan pengujian ini. Alur proses yang digunakan ini disesuaikan dengan alur proses yang terjadi pada aplikasi *SortSite*. Proses pengujian dimulai dari penguji mengakses aplikasi *SortSite* dan menjalankan sistem pada aplikasi tersebut dan proses pengujian berakhir ketika *SortSite* menampilkan serangkaian hasil tes kompabilitas sistem terhadap peramban *internet explorer*.

Tabel 7.10 *Test Case* Pengujian *Compability Testing* Pada *Microsoft Edge*

Test case name	P – CT - 02
Purpose of test	Pengujian dilakukan untuk memastikan bahwa sistem dapat berjalan dengan baik pada peramban *Microsoft Edge* versi 17
Test attribute	
Test focus	*Compatibility*
Kode kebutuhan	SPPDK – NF - 001
Test process	1. Penguji mengakses dan menjalankan sistem pada aplikasi *SortSite* dengan cara memasukkan url sistem pada menu bar *address.*

	2. Penguji menekan tombol *check* untuk memulai proses *checking*. 3. *SortSite* memindai setiap halaman yang dimiliki oleh sistem. 4. Setiap halaman diperiksa untuk masalah kualitas dengan menggunakan *checkpoints,* kemudian *SortSite* memeriksa halaman yang terhubung dengan halaman tersebut. Selanjutnya *SortSite* akan memindai halaman baru yang ditemukan dan mengulang proses ini sampai akhir halaman. 5. *SortSite* menampilkan serangkaian tes validasi kompatibilitas yang diperoleh untuk mengungkapkan kesalahan yang dapat ditelusuri.
Test result	*SortSite* menampilkan serangkaian hasil tes kompabilitas sistem yang menunjukkan bahwa sistem dapat berjalan dengan baik

Tabel diatas merupakan *test case* yang digunakan pada pengujian *compatibility testing.* Pengujian ini bertujuan untuk memastikan bahwa sistem yang dibangun dapat berjalan dengan baik pada peramban *Microsoft edge*. Terdapat beberapa proses untuk melakukan pengujian ini. Alur proses yang digunakan ini disesuaikan dengan alur proses yang terjadi pada aplikasi *SortSite*. Proses pengujian dimulai dari penguji mengakses aplikasi *SortSite* dan menjalankan sistem pada aplikasi tersebut dan proses pengujian berakhir ketika *SortSite* menampilkan serangkaian hasil tes kompabilitas sistem terhadap peramban *Microsoft edge*.

Tabel 7.11 *Test Case* Pengujian *Compability Testing* Pada *Firefox*

Test case name	P – CT - 03
Purpose of test	Pengujian dilakukan untuk memastikan bahwa sistem dapat berjalan dengan baik pada peramban *Firefox* versi 61
Test attribute	
Test focus	*Compatibility*
Kode kebutuhan	SPPDK – NF - 001
Test process	1. Penguji mengakses dan menjalankan sistem pada aplikasi *SortSite* dengan cara memasukkan url sistem pada menu bar *address.* 2. Penguji menekan tombol *check* untuk memulai proses *checking*. 3. *SortSite* memindai setiap halaman yang dimiliki oleh sistem. 4. Setiap halaman diperiksa untuk masalah kualitas dengan menggunakan *checkpoints,* kemudian *SortSite* memeriksa halaman yang terhubung dengan halaman tersebut. Selanjutnya *SortSite* akan memindai halaman baru yang ditemukan dan

	mengulang proses ini sampai akhir halaman. 5. *SortSite* menampilkan serangkaian tes validasi kompatibilitas yang diperoleh untuk mengungkapkan kesalahan yang dapat ditelusuri.
Test result	*SortSite* menampilkan serangkaian hasil tes kompabilitas sistem yang menunjukkan bahwa terdapat *critical issues* dan *minor issues* sehingga tampilan sistem tidak dapat berjalan dengan baik.

Tabel diatas merupakan *test case* yang digunakan pada pengujian *compatibility testing.* Pengujian ini bertujuan untuk memastikan bahwa sistem yang dibangun dapat berjalan dengan baik pada peramban *firefox*. Terdapat beberapa proses untuk melakukan pengujian ini. Alur proses yang digunakan ini disesuaikan dengan alur proses yang terjadi pada aplikasi *SortSite*. Proses pengujian dimulai dari penguji mengakses aplikasi *SortSite* dan menjalankan sistem pada aplikasi tersebut dan proses pengujian berakhir ketika *SortSite* menampilkan serangkaian hasil tes kompabilitas sistem terhadap peramban *firefox*. Hasil dari pengujian *compability testing* menunjukkan bahwa terdapat *critical issues* dan *minor issues*. *Critical issues* ini terjadi karena adanya input yang berupa *password* dan untuk versi terakhir dari *firefox* sangat disarankan menggunakan HTTPS. Sedangkan *minor issues* terjadi karena adanya CSS yang tidak mendukung sehingga tampilan sistem kurang dapat berjalan dengan baik pada *browser firefox*.

Tabel 7.12 *Test Case* Pengujian *Compability Testing* Pada Internet *Safari*

Test case name	P – CT - 04
Purpose of test	Pengujian dilakukan untuk memastikan bahwa sistem dapat berjalan dengan baik pada peramban *Safari* versi 10 11
Test attribute	
Test focus	*Compatibility*
Kode kebutuhan	SPPDK – NF - 001
Test process	1. Penguji mengakses dan menjalankan sistem pada aplikasi *SortSite* dengan cara memasukkan url sistem pada menu bar *address.* 2. Penguji menekan tombol *check* untuk memulai proses *checking*. 3. *SortSite* memindai setiap halaman yang dimiliki oleh sistem. 4. Setiap halaman diperiksa untuk masalah kualitas dengan menggunakan *checkpoints,* kemudian *SortSite* memeriksa halaman yang terhubung dengan halaman tersebut. Selanjutnya *SortSite* akan memindai halaman baru yang ditemukan dan mengulang proses ini sampai akhir halaman. 5. *SortSite* menampilkan serangkaian tes validasi

	kompatibilitas yang diperoleh untuk mengungkapkan kesalahan yang dapat ditelusuri.
Test result	*SortSite* menampilkan serangkaian hasil tes kompabilitas sistem yang menunjukkan bahwa terdapat *major issues* dan *minor issues* karena adanya CSS yang tidak mendukung sehingga tampilan sistem tidak dapat berjalan dengan baik.

Tabel diatas merupakan *test case* yang digunakan pada pengujian *compatibility testing.* Pengujian ini bertujuan untuk memastikan bahwa sistem yang dibangun dapat berjalan dengan baik pada peramban *safari*. Terdapat beberapa proses untuk melakukan pengujian ini. Alur proses yang digunakan ini disesuaikan dengan alur proses yang terjadi pada aplikasi *SortSite*. Proses pengujian dimulai dari penguji mengakses aplikasi *SortSite* dan menjalankan sistem pada aplikasi tersebut dan proses pengujian berakhir ketika *SortSite* menampilkan serangkaian hasil tes kompabilitas sistem terhadap peramban *safari*. Hasil dari pengujian menunjukkan bahwa terdapat *major issues* dan *minor issues* yang disebabkan karena adanya CSS yang tidak mendukung sehingga tampilan sistem kurang dapat berjalan dengan baik pada *browser safari*.

Tabel 7.13 Test Case Pengujian Compability Testing Pada Opera

Test case name	P – CT - 06
Purpose of test	Pengujian dilakukan untuk memastikan bahwa sistem dapat berjalan dengan baik pada peramban *Opera* versi 54
Test attribute	
Test focus	*Compatibility*
Kode kebutuhan	SPPDK – NF - 001
Test process	1. Penguji mengakses dan menjalankan sistem pada aplikasi *SortSite* dengan cara memasukkan url sistem pada menu bar *address.* 2. Penguji menekan tombol *check* untuk memulai proses *checking*. 3. *SortSite* memindai setiap halaman yang dimiliki oleh sistem. 4. Setiap halaman diperiksa untuk masalah kualitas dengan menggunakan *checkpoints,* kemudian *SortSite* memeriksa halaman yang terhubung dengan halaman tersebut. Selanjutnya *SortSite* akan memindai halaman baru yang ditemukan dan mengulang proses ini sampai akhir halaman. 5. *SortSite* menampilkan serangkaian tes validasi kompatibilitas yang diperoleh untuk mengungkapkan kesalahan yang dapat ditelusuri.
Test result	*SortSite* menampilkan serangkaian hasil tes kompabilitas sistem yang menunjukkan bahwa sistem dapat berjalan dengan baik

Tabel diatas merupakan *test case* yang digunakan pada pengujian *compatibility testing.* Pengujian ini bertujuan untuk memastikan bahwa sistem yang dibangun dapat berjalan dengan baik pada peramban *opera*. Terdapat beberapa proses untuk melakukan pengujian ini. Alur proses yang digunakan ini disesuaikan dengan alur proses yang terjadi pada aplikasi *SortSite*. Proses pengujian dimulai dari penguji mengakses aplikasi *SortSite* dan menjalankan sistem pada aplikasi tersebut dan proses pengujian berakhir ketika *SortSite* menampilkan serangkaian hasil tes kompabilitas sistem terhadap peramban *opera*.

Tabel 7.14 *Test Case* Pengujian *Compability Testing* Pada *Chrome*

Test case name	P – CT - 05
Purpose of test	Pengujian dilakukan untuk memastikan bahwa sistem dapat berjalan dengan baik pada peramban *Chrome*
Test attribute	
Test focus	*Compatibility*
Kode kebutuhan	SPPDK – NF - 001
Test process	1. Penguji mengakses dan menjalankan sistem pada aplikasi *SortSite* dengan cara memasukkan url sistem pada menu bar *address.* 2. Penguji menekan tombol *check* untuk memulai proses *checking*. 3. *SortSite* memindai setiap halaman yang dimiliki oleh sistem. 4. Setiap halaman diperiksa untuk masalah kualitas dengan menggunakan *checkpoints,* kemudian *SortSite* memeriksa halaman yang terhubung dengan halaman tersebut. Selanjutnya *SortSite* akan memindai halaman baru yang ditemukan dan mengulang proses ini sampai akhir halaman. 5. *SortSite* menampilkan serangkaian tes validasi kompatibilitas yang diperoleh untuk mengungkapkan kesalahan yang dapat ditelusuri.
Test result	*SortSite* menampilkan serangkaian hasil tes kompabilitas sistem yang menunjukkan bahwa terdapat *critical issues* sehingga sistem tidak dapat berjalan dengan baik.

Tabel diatas merupakan *test case* yang digunakan pada pengujian *compatibility testing.* Pengujian ini bertujuan untuk memastikan bahwa sistem yang dibangun dapat berjalan dengan baik pada peramban *chrome*. Terdapat beberapa proses untuk melakukan pengujian ini. Alur proses yang digunakan ini disesuaikan dengan alur proses yang terjadi pada aplikasi *SortSite*. Proses pengujian dimulai dari penguji mengakses aplikasi *SortSite* dan menjalankan sistem pada aplikasi tersebut dan proses pengujian berakhir ketika *SortSite* menampilkan serangkaian hasil tes kompabilitas sistem terhadap peramban *chrome*. Hasil dari pengujian *compability testing* pada peramban *chrome*

menunjukkan bahwa terdapat *critical issues. Critical issues* ini disebabkan karena adanya input yang berupa *password* dan untuk versi terakhir dari *chrome* sangat disarankan menggunakan HTTPS.

Tabel 7.15 *Test Case* Pengujian *Compability Testing* Pada *iOS*

Test case name	P – CT - 06
Purpose of test	Pengujian dilakukan untuk memastikan bahwa sistem dapat berjalan dengan baik pada peramban *iOS.*
Test attribute	
Test focus	*Compatibility*
Kode kebutuhan	SPPDK – NF - 001
Test process	1. Penguji mengakses dan menjalankan sistem pada aplikasi *SortSite* dengan cara memasukkan url sistem pada menu bar *address.* 2. Penguji menekan tombol *check* untuk memulai proses *checking*. 3. *SortSite* memindai setiap halaman yang dimiliki oleh sistem. 4. Setiap halaman diperiksa untuk masalah kualitas dengan menggunakan *checkpoints,* kemudian *SortSite* memeriksa halaman yang terhubung dengan halaman tersebut. Selanjutnya *SortSite* akan memindai halaman baru yang ditemukan dan mengulang proses ini sampai akhir halaman. 5. *SortSite* menampilkan serangkaian tes validasi kompatibilitas yang diperoleh untuk mengungkapkan kesalahan yang dapat ditelusuri.
	SortSite menampilkan serangkaian hasil tes kompabilitas sistem yang menunjukkan bahwa sistem dapat berjalan dengan baik

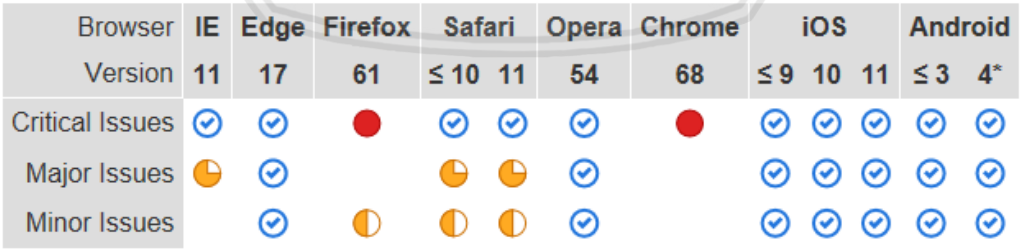
Tabel diatas merupakan *test case* yang digunakan pada pengujian *compatibility testing.* Pengujian ini bertujuan untuk memastikan bahwa sistem yang dibangun dapat berjalan dengan baik pada peramban *ios*. Terdapat beberapa proses untuk melakukan pengujian ini. Alur proses yang digunakan ini disesuaikan dengan alur proses yang terjadi pada aplikasi *SortSite*. Proses pengujian dimulai dari penguji mengakses aplikasi *SortSite* dan menjalankan sistem pada aplikasi tersebut dan proses pengujian berakhir ketika *SortSite* menampilkan serangkaian hasil tes kompabilitas sistem terhadap peramban *ios*.

Tabel 7.16 *Test Case* Pengujian *Compability Testing* Pada *Android*

Test case name	P – CT - 07
Purpose of test	Pengujian dilakukan untuk memastikan bahwa sistem dapat berjalan dengan baik pada peramban *Android.*
Test attribute	

Test focus	*Compatibility*
Kode kebutuhan	SPPDK – NF - 001
Test process	1. Penguji mengakses dan menjalankan sistem pada aplikasi *SortSite* dengan cara memasukkan url sistem pada menu bar *address.* 2. Penguji menekan tombol *check* untuk memulai proses *checking*. 3. *SortSite* memindai setiap halaman yang dimiliki oleh sistem. 4. Setiap halaman diperiksa untuk masalah kualitas dengan menggunakan *checkpoints,* kemudian *SortSite* memeriksa halaman yang terhubung dengan halaman tersebut. Selanjutnya *SortSite* akan memindai halaman baru yang ditemukan dan mengulang proses ini sampai akhir halaman. 5. *SortSite* menampilkan serangkaian tes validasi kompatibilitas yang diperoleh untuk mengungkapkan kesalahan yang dapat ditelusuri.
Test result	*SortSite* menampilkan serangkaian hasil tes kompabilitas sistem yang menunjukkan bahwa sistem dapat berjalan dengan baik.

Tabel diatas merupakan *test case* yang digunakan pada pengujian *compatibility testing.* Pengujian ini bertujuan untuk memastikan bahwa sistem yang dibangun dapat berjalan dengan baik pada peramban *android*. Terdapat beberapa proses untuk melakukan pengujian ini. Alur proses yang digunakan ini disesuaikan dengan alur proses yang terjadi pada aplikasi *SortSite*. Proses pengujian dimulai dari penguji mengakses aplikasi *SortSite* dan menjalankan sistem pada aplikasi tersebut dan proses pengujian berakhir ketika *SortSite* menampilkan serangkaian hasil tes kompabilitas sistem terhadap peramban *android*.



Gambar 7.3 Hasil Dari Pengujian Compatibility Testing

Hasil dari pengujian *compability testing* yang dilakukan berdasarkan *test case* yang telah dibuat sebelumnya ditunjukkan oleh gambar 7.3. Hasil pengujian *compatibility testing* tersebut menunjukkan bahwa terdapat 2 *critical issues* yaitu pada peramban *firefox* dan *chrome*. Hal ini disebabkan karena tidak diaktifkannya fungsi SSL (*Secure Socket Layer*). Hasil pengujian tersebut juga

menunjukkan bahwa terdapat 3 *major issues* dan 3 *minor issues*. *Major* dan *minor issues* ini terjadi karena adanya *CSS* yang tidak mendukung untuk beberapa peramban. Berdasarkan kebutuhan non fungsional yang didefinisikan dapat disimpulkan bahwa sistem dapat memenuhi persyaratan non fungsional pengguna karena *error* yang terjadi pada peramban tersebut tidak termasuk pada kebutuhan persyaratan non fungsional yang sudah didefinisikan sebelumnya.

7.3 Pengujian *Usability Testing*

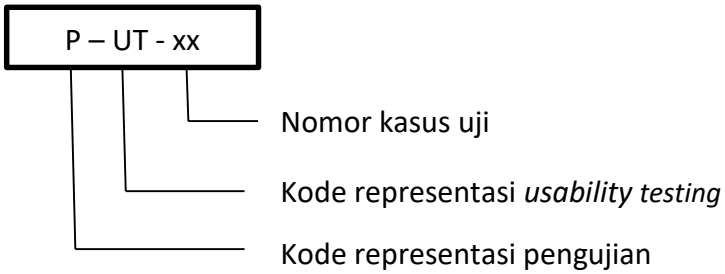
Pengujian *usability testing* ditunjukkan untuk mengetahui seberapa mudah sistem dapat digunakan oleh pengguna. Pengukuran *usability testing* ini dapat dilakukan dengan menggunakan *System Usability Scale (SUS)*. *System Usability Scale* merupakan skala *likert* dengan sepuluh pertanyaan yang dapat memberikan pandangan global tentang penilaian subyektivitas kegunaan. Pertanyaan yang dipilih pada *System Usability Testing* ini mencakup berbagai aspek kegunaan sistem, seperti kebutuhan untuk dukungan, pelatihan, dan kompleksitas. Berikut merupakan sepuluh pertanyaan yang akan digunakan pada pengujian *usability testing*.

Tabel 7.17 *System Usability Scale*

No	Pertanyaan	Sangat tidak setuju — Sangat setuju
1.	Saya akan sering menggunakan sistem informasi pelaporan pendataan kependudukan kabupaten malang	1 2 3 4 5
2.	Sistem informasi pelaporan pendataan kependudukan kabupaten malang terlalu kompleks atau rumit	1 2 3 4 5
3.	Sistem informasi pelaporan pendataan kependudukan kabupaten malang mudah digunakan	1 2 3 4 5
4.	Saya akan membutuhkan dukungan dari orang teknis untuk dapat menggunakan sistem informasi pelaporan pendataan kependudukan kabupaten malang	1 2 3 4 5
5.	Berbagai fungsi dalam sistem informasi pelaporan pendataan kependudukan kabupaten	1 2 3 4 5

	malang sudah terintegrasi dengan baik	
6.	Terlalu banyak ketidakkonsistensi dalam sistem pelaporan pendataan kependudukan kabupaten malang	1 2 3 4 5
7.	Saya rasa banyak orang yang akan belajar menggunakan sistem informasi pelaporan pendataan kependudukan kabupaten malang ini dengan cepat	1 2 3 4 5
8.	Saya rasa sistem informasi pelaporan pendataan kependudukan kabupaten malang ini sangat sulit untuk digunakan	1 2 3 4 5
9.	Saya merasa percaya diri menggunakan sistem informasi pelaporan pendataan kependudukan kabupaten malang	1 2 3 4 5
10.	Saya perlu belajar banyak hal sebelum saya bisa menggunakan sistem informasi pelaporan pendataan kependudukan kabupaten malang ini	1 2 3 4 5

Pada pengujian *usability testing* ini akan dibuat beberapa *test case* untuk mempermudah dalam proses pengujian. Terdapat pemberian kode sebagai penanda *test case* yang dirancang. Kode yang dibuat pada pengujian *usability testing* ini ditunjukkan oleh gambar 7.4.



Gambar 7.4 Pengkodean Pengujian *Usability Testing*

Test case yang dirancang pada pengujian *usability testing* ini sesuai dengan format *test case* yang ada pada gambar 2.16. Berikut adalah rancangan *test case* dari pengujian *usability testing*.

Tabel 7.18 *Test Case Usability Testing*

Test case name	P – UT - 01
Purpose of test	Pengujian dilakukan untuk mengetahui seberapa mudah sistem dapat digunakan oleh pengguna.
Test attribute	
Test focus	*Usability*
Test process	1. Penguji mengisi kuesioner pengujian usability testing 2. Dilakukan perhitungan skor sesuai dengan *system usability scale* terhadap kuesioner yang telah diisi oleh penguji 3. Diperoleh hasil pengujian sistem dengan menggunakan metode *usability testing*
Test result	Diperoleh hasil pengujian sistem dengan menggunakan metode *usability testing.*

Tabel 7.19 merupakan hasil pengujian dari *usability testing*. Skor pada masing –masing responden dihitung sesuai dengan metode SUS dimana untuk pernyataan ganjil merupakan pernyataan positif dan skornya didapat dari nila mk bi yang diberikan oleh responden dikurangi 1. Sedangkan untuk pernyataan genap merupakan pernyataan negatif dan skornya didapat dari 5 dikurangi dengan nilai yang diberikan oleh responden. Selanjutnya setiap skor yang diperoleh oleh masing – masing responden akan di jumlah untuk selanjutnya dikali dengan 2.5.

Tabel 7.19 Hasil Pengujian *Usability Testing*

Nama	Jabatan	Skor	Jumlah skor	Nilai SU (skor x 2,5)
Disi Sawitri Widowati, SE.ME	Kasi inovasi pelayanan	4+3+3+3+3+4+4+3+3+3	33	82,5
Slamet Rejo	Kasi pemanfaatan data dan dokumen	4+4+4+1+4+2+3+3+3+3	31	77,5
Vedo	Staf	4+3+4+2+4+4+4+2+3+2	31	77,5
Rudhy	PNS	3+4+4+2+3+3+3+3+3+4	32	80
Dyah	PNS	3+4+4+1+3+3+3+4+3+4	32	80
Yanto	Pegawai desa	4+3+3+3+3+3+3+4+3+3	32	80
Afif Jauhari	Pegawai desa	4+3+4+3+3+4+3+4+3+3	34	85

Berdasarkan hasil perhitungan skor diatas, grade dari nilai *System Usability*

(82.5 + 77.5 + 77.5+80 +80 +80+85)/7=562,5/7=80,35

Berdasarkan interpretasi *grade* skor SUS (Bangor, Kortum, & Miller, 2009) nilai system usability 80,35 mendapatkan *grade* B. menurut Bangor, Kortum, & Miller sistem dikatakan baik apabila mendapat skor diatas 70 dengan *grade* C. sistem informasi pelaporan pendataan kependudukan kabupaten malang ini mendapat skor B sehingga dapat disimpulkan bahwa sistem dikatakan baik dan dapat digunakan dengan mudah oleh pengguna.

BAB 8 PENUTUP

8.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan, dapat diambil kesimpulan bahwa sistem informasi pendataan dan pelaporan data kependudukan kabupaten malang dapat dikembangkan melalui beberapa kegiatan pengembangan sistem dengan hasil sebagai berikut :

1. Pada fase *inception* dilakukan analisis proses bisnis dengan melakukan analisis terhadap proses bisnis saat ini (as-is) yang menghasilkan proses bisnis perbaikan (to-be). Analisis proses bisnis to-be menghasilkan perbaikan proses bisnis pada proses bisnis pengajuan administrasi kependudukan dan proses bisnis pelaporan data kependudukan. Hasil dari pemodelan proses bisnis ini dijadikan sebagai bahan masukan dalam melakukan analisa kebutuhan yang dilakukan pada fase *elaboration*. Pemodelan proses bisnis yang dilakukan telah disetujui oleh pemangku kepentingan maka *lifecycle objective milestone* fase *inception* pada RUP telah selesai.
2. Pada fase *elaboration* dilakukan analisa kebutuhan sesuai dengan hasil analisis proses bisnis yang telah dilakukan dan dilakukan perancangan sistem sesuai dengan analisis kebutuhan yang telah dilakukan pada fase *inception*. Analisis kebutuhan yang dilakukan meliputi analisis kebutuhan pemangku kepentingan dan pengguna, analisis pengguna, analisis fitur, analisis persyaratan fungsional dan non fungsional sistem. Pada kegiatan ini diperoleh 7 fitur, 18 kebutuhan fungsional dan 1 kebutuhan non fungsional. Permasalah terkait pengajuan administrasi kependudukan dapat diselesaikan dengan kebutuhan fungsional (SPPDK-F-02,SPPDK-F-03,SPPDK-F-04,SPPDK-F-05, SPPDK-F-06,SPPDK-F-07,SPPDK-F-10,SPPDK-F-11,SPPDK-F-12,SPPDK-F-13, SPPDK-F-14 DAN SPPDK-F-15). Sedangkan untuk permasalahan terkait pelaporan data kependudukan dapat diselesaikan dengan kebutuhan fungsional (SPPDK-F-08,SPPDK-F-09). Perancangan pada fase *elaboration* ini menghasilkan beberapa jenis model interaksi. Model interaksi yang dihasilkan pada fase *elaboration* ini meliputi, *sequence diagram* untuk memvisualisasikan pertukaran pesan antar entitas aktor, entitas *boundary*, objek *control*, dan objek *model* yang saling berinteraksi. Model interaksi selanjutnya adalah *class diagram* untuk menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat, *physical data model* untuk mendefinisikan struktur fisik dan hubungan data dalam domain atau aplikasi subjek, perancangan algoritma untuk menentukan dan menyusun sekumpulan langkah operasi logika untuk memecahkan masalah melakukan perhitungan otomatis, pemrosesan data dan tugas -tugas lainnya, dan perancangan antarmuka sistem. Syarat yang harus dipenuhi pada fase *elaboration* ini adalah deskripsi arsitektur sistem telah dilakukan sehingga *lifecycle architecture milestone* fase *elaboration* telah terpenuhi.

3. Pada fase *contruction* dilakukan implementasi sistem berdasarkan pada perancangan yang telah dibuat pada fase *elaboration*. Implementasi sistem ini dilakukan dengan menggunakan kerangka kerja *code igniter*. Pada fase *contruction* ini juga dilakukan pengujian sistem dengan menggunakan *validation testing dan compatibility testing*. Hasil pengujian dengan *validation testing* terhadap fungsi – fungsi menambah pengajuan akta kematian, menambah kartu pengambilan akta kematian, mencetak laporan kartu pengambilan akta kematian, melihat laporan bulanan dan melihat rekap laporan menghasilkan status valid yang berarti bahwa fungsionalitas sistem dapat berjalan dengan baik dan sesuai dengan spesifikasi kebutuhan. Sedangkan hasil dari pengujian *compatibility testing* didapat 2 *critical issues dan* 3 *major issues* dan 3 *minor issues.* 2 *critical issues* karena tidak diaktifkannya fungsi SSL (*Secure Socket Layer*). Sedangkan 3 *major issues* dan 3 *minor issues* ini terjadi karena adanya *CSS* yang tidak mendukung untuk beberapa peramban. Berdasarkan hasil pengujian yang telah dilakukan menunjukkan bahwa sistem yang telah diuji telah sesuai dengan kebutuhan fungsional dan non fungsional yang telah diidentifikasi sebelumnya. *Initial operational capability milestone* fase *contruction* telah terpenuhi.
4. Pada fase *transition* dilakukan pengujian dengan menggunakan *usability testing* dengan menggunakan *System Usability Scale (SUS)*. Pada pengujian *usability testing* ini diperoleh skor 80,35 yang berarti berada pada grade B. Berdasarkan hasil perhitungan nilai usabilitas sistem dapat dikatakan bahwa sistem baik dan dapat dengan mudah digunakan oleh pengguna.

8.2 Saran

Saran yang dapat diberikan sebagai bahan pertimbangan untuk melaksanakan pengembangan lebih lanjut pada Sistem Informasi Pelaporan Pendataan Kependudukan Kabupaten Malang di antaranya adalah:

1. Perlu dilakukan evaluasi pada antarmuka pengguna Sistem Informasi Pelaporan Pendataan Kependudukan Kabupaten Malang untuk mengetahui apabila perlu dilakukan perbaikan atau pengembangan lanjut pada bagian antarmuka pengguna.
2. Perlu adanya integrasi database sistem yang dikembangkan dengan database pusat.

DAFTAR PUSTAKA

Anhar, 2010. *Panduan Menguasai PHP & MySQL Secara Otodidak.* Jakarta: Media Kita.

B.B.Agarwal, Tayal, S. P., & Gupta, M., 2010. *Software Engineering & Testing.* USA: Jones and Bartlett Publishers.

Bangor, A., Kortum, P., & Miller, J., 2009. *Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale.* Journal of Usability Studies.

Bittner, K., & Spence, I., 2002. *Use Case Modelling.* U.S: Addison Wesley.

Booch, G., 1994. *Object Oriented Analysis And Design.* California: Addison-Wesley.

Booch, G., Rumbaugh, J., & Jacobson, I., 2005. *The Unified Modeling Language User Guide Second Edition.* U.S: Addison Wesley Professional.

Brooke, J., 1986. *SUS-A Quick AND Dirty Usability Scale.* United Kingdom: Redhatch Consulting Ltd.

Fatta, H. A., 2007. *Analisis dan Perancangan Sistem Informasi untuk Keunggulan Bersaing Perusahaan dan Organisasi Modern.* Yogyakarta: Andi.

Hambling, B., & Goethem, P. V., 2013. *User Acceptance Testing A step by ste guide.* UK: BSC Learning and Development Ltd.

IBM., 1998. *Rational Unified Process Best Practices for Software Development Teams.* United States: IBM Rational Software.

IBM., 2007. *writing Good Use Cases.* USA: IBM Corporation.

Indonesia, R., 2013. *Undang - Undang No 24 Tahun 2013 tentang Adminitrasi Kependudukan.*

Kroll, P., & Kruchten, P., 2003. *The Rational Unified Process Made Easy: A Practitioner's Guide to the RUP.* Boston: Addison Wesley.

Kruchten, P., 2003. *The Relational Unified Process An Introduction Third Edition.* Addison Wesley.

Ladjamudin, A.-B. b., 2005 . *Analisis dan Desain Sistem Informasi.* Yogyakarta: Graha Ilmu.

Mulyani, S., 2016. *Metode Analisis dan Perancangan Sistem.* Bandung: Abdi Sistematika.

Murhada, & Giap, Y. C., 2011. *Pengantar Teknologi Informasi.* Jakarta: Mitra Wacana Media.

Negeri, M. D., 2016. *Peraturan Menteri Dalam Negeri Pepublik Indonesia No 4 Tahun 2016 tentang Adminitrasi Kependudukan.*

Object Management Group, 2011 *Business Process Model and Notation (BPMN)*. [online] Tersedia di: <http://www.omg.org/spec/BPMN/2.0> [Diakses 20 Januari 2018]

Pressman, R. S., 2010. *Software Engineering: A practicioner's Approach, Seventh Edition.* New York: McGraw-Hill.

Rerung, R. R., 2018. *Pemrograman Web Dasar.* Yogyakarta: Budi Utama.

Shalahudin, M., & Sukamto, R. A., 2014. *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek.* Bandung: Informatika.

Shinde, V., 2013. *Software Testing Career Package A Software Tester's Journey From Getting a Job to Becoming a Test Leader.* Software Testing Help.

Sidik, B., 2004. *Pemrograman web dengan PHP.* Bandung: Informatika.

Sommerville, I., 2011. *Software Engineering Ninth edition.* United States: Addison Wesley.

SparxSystems, 2004. *The Use Case Model.* [online] Tersedia di: <www.sparxsystems.com.au.> [Diakses 02 Mei 2018]

sparxsystems, 2014. *Physical Data Models*. [online] Tersedia di: <https://sparxsystems.com/enterprise_architect_user_guide/12.1/database_engineering/physical_data_model.html> [Diakses 10 Mei 2018]

Sutabri, T., 2004. *Analisa Sistem Informasi.* Yogyakarta: Andi.

W.Ambler, S., 2004. *The Object Primer 3rd Edition Agile Model Driven Development with UML 2.* New York: Cambridge University Press.

weske, M., 2007. *Biusiness Process Management Concept, Languages, Architectures.* New York: Springer.

Zed, M., 2004. *Metode Penelitian Kepustakaan.* Jakarta: Yayasan Obor Indonesia.