Pengembangan Aplikasi *Mobile* Pengaduan Layanan dan *Maintenance* Berbasis *Android* Pada Fakultas Ilmu Komputer Universitas Brawijaya

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Hanoum Eva Hayati
NIM: 155150200111242

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

PENGESAHAN

PENGEMBANGAN APLIKASI *MOBILE* PENGADUAN LAYANAN DAN *MAINTENANCE* BERBASIS *ANDROID* PADA FAKULTAS ILMU KOMPUTER UNIVERSITAS BRAWIJAYA

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Hanoum Eva Hayati
NIM: 155150200111242

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

Dosen Pembimbing I	Dosen Pembimbing II
Denny Sagita Rusdianto, S.Kom, M.Kom	Faizatul Amalia, S.Pd.,M.Pd
NIP. 19851124 201504 1 001	NIK. 201309 860821 2 001

Mengetahui
Ketua Jurusan Teknik Informatika

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

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain dalam kegiatan 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 terbukti 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, 26 April 2019

METERAI TEMPEL
0360AAFF707217450
6000
ENAM RIBU RUPIAH

Hanoum Eva Hayati

NIM: 155150200111242

KATA PENGANTAR

Puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat, taufik, dan hidayah-Nya sehingga naskah skripsi yang berjudul "Pembangunan Aplikasi *Mobile* Pengaduan Layanan dan *Maintenance* Berbasis *Android* Pada Fakultas Ilmu Komputer Universitas Brawijaya" ini dapat terselesaikan. Melalui pengantar ini, penulis ingin menyampaikan rasa hormat dan terima kasih kepada:

1. Bapak Denny Sagita Rusdianto, S.Kom, M.Kom sebagai dosen pembimbing satu skripsi yang telah memberikan kesempatan kepada penulis untuk menjadi mahasiswa bimbingan skripsinya dan melakukan penelitian dengan studi kasus yang telah beliau berikan, serta menjadi pembimbing yang sabar, solutif dan senantiasa memberikan contoh sebagai pribadi yang profesional dalam membimbing penyelesaian skripsi penulis.
2. Ibu Faizatul Amalia, S.Pd.,M.Pd, sebagai dosen pembimbing dua yang telah memberikan kesempatan dan waktunya untuk membimbing penulis dalam menyelesaikan penelitian yang dilakukan, serta senantiasa menjadi pembimbing yang sabar dan teliti dalam memeriksa pengerjaan dokumen skripsi sampai terselesaikannya skripsi ini.
3. Bapak Agus Wahyu Widodo, S.T, M.Cs selaku Ketua Program Studi Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya.
4. Bapak Tri Astoto Kurniawan, S.T., M.T., Ph.d selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya.
5. Bapak Ferix Panji Andrianto, S.ST, sebagai pegawai Perlengkapan Fakultas Ilmu Komputer Universitas Brawijaya yang telah bersedia untuk menjadi narasumber, menguji aplikasi yang telah dikembangkan, dan memberi masukan-masukan terhadap pengembangan aplikasi ini.
6. Ayahanda dan Ibunda dan seluruh keluarga besar penulis atas segala nasehat, kasih sayang, perhatian dan kesabarannya di dalam membesarkan dan mendidik penulis, serta yang senantiasa tiada henti-hentinya memberikan doa dan semangat demi terselesaikannya skripsi ini.
7. Seluruh teman-teman penulis yang telah banyak memberi bantuan dan dukungan selama penyelesaian naskah skripsi ini.

Penulis menyadari bahwa dalam penyusunan naskah 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, 26 April 2019

Hanum.eva@student.ub.ac.id

ABSTRAK

Fakultas Ilmu Komputer Universitas Brawijaya (FILKOM) memiliki sub bagian umum dan perlengkapan yang salah satunya bertanggung jawab dalam menangani pengaduan-pengaduan keluhan dan juga melakukan *maintenance* layanan di FILKOM. Dalam melaksanakan tanggung jawab tersebut, sub bagian umum dan perlengkapan mengalami kesulitan akibat mekanisme yang masih belum berjalan dengan optimal. Hal tersebut mengakibatkan pengaduan berasal dari berbagai macam sumber sehingga tidak dapat terdata dengan baik dan pengadu sering mengeluh dengan penanganan pengaduan yang sudah dilaporkannya. Dalam melakukan *maintenance,* pendataan riwayat suatu inventaris juga masih belum optimal. Agar pelaksanaan fungsi tersebut berjalan dengan optimal maka dibutuhkan suatu wadah yang menjadi tempat pengaduan keluhan dan juga penjadwalan *maintenance.* Salah satu solusinya yaitu dengan pengembangan aplikasi pengaduan layanan dan *maintenance* (PLM) berbasis *android*. Penelitian ini diawali dengan proses rekayasa kebutuhan yang menghasilkan 18 kebutuhan fungsional dan 2 kebutuhan non fungsional serta 3 *role* aktor. Dalam pengembangannya, aplikasi PLM menerapkan metode *waterfall* karena kebutuhan sudah terdefinisi secara keseluruhan diawal. Kemudian perancangan dilakukan dengan pemodelan berorientasi objek dan implementasi program menggunakan bahasa java pada Android Studio serta implementasi data menggunakan Firebase. Berdasarkan proses pengujian, didapatkan hasil 100% valid untuk pengujian unit, validasi, dan *compatibility* serta didapatkan hasil pengujian *usability* dengan nilai 100% untuk metode skenario tugas dan 97,2% untuk metode kuesioner USE.

Kata kunci: *pengaduan layanan, maintenance, android*

ABSTRACT

The Faculty of Computer Science, Brawijaya University (FILKOM) has public and equipment affair sub section, which is responsible for handling complaints and maintenance services at FILKOM. In handling these responsibilities, public and equipment affair sub section have difficulties due to a mechanism that has not run optimally. This resulted in complaints come from various sources so that it could not be recorded properly and complainants often complained about the handling of complaints that had been reported. In handling maintenance, the data collection history of an inventory is still not maximal. In order for the implementation of these functions to run optimally, a container for complaint complaints and maintenance scheduling is needed. One solution is develop an Android-based service complaint and maintenance (PLM) application. This development begins with requirement engineering that produces 18 functional requirement and 2 non-functional requirement and 3 actor roles. In development process, the PLM application applies the waterfall method because the requirements have been defined as a whole at the beginning. The design process is done by object-oriented modeling and program implementation using java language on Android Studio and implementing data using Firebase. Based on the testing process, the results obtained 100% valid for unit testing, validation, and compatibility and obtained usability testing results with a value of 100% for the task screening method and 97.2% for the USE questionnaire method.

Keywords: *service complaint, maintenance, android*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Dalam suatu universitas, pengaduan layanan dan *maintenance* menjadi salah satu proses bisnis utama yang mendukung kelancaran semua kegiatan yang berjalan di universitas. Oleh karena itu, pada suatu instansi pendidikan harus ada mekanisme untuk keluhan-keluhan mengenai pelayanan dan penjadwalan *maintenance* suatu peralatan milik instansi tersebut. Menurut *website e-complaint* Universitas Brawijaya, mekanisme pengaduan layanan dimaksudkan agar sivitas pada instansi memperoleh kepuasan dan dapat berkembang secara berkelanjutan. Hal ini menunjukkan bahwa keluhan dari sivitas merupakan salah satu instrumen penting dari suatu instansi untuk melakukan evaluasi sehingga dapat dideteksi lebih dini dan dapat dilakukan perbaikan dan pengembangan secara cepat dan terus menerus (e-complaint.ub.ac.id, 2012).

Universitas Brawijaya (UB) adalah universitas yang berada di Kota Malang, Jawa Timur, yang sudah diakui oleh negara dan dunia atas prestasi-prestasinya (ub.ac.id, 2018). Sebagai Universitas yang ideal, UB telah memiliki mekanisme untuk mengadukan keluhan-keluhan. Mekanisme tersebut berupa *website E-Complaint* Universitas Brawijaya. Pada *website* tersebut keluhan-keluhan dapat berupa ketidakpuasan terhadap layanan yang diberikan oleh unit kerja UB baik urusan akademik maupun non akademik. Keluhan tersebut dapat diadukan oleh mahasiswa, dosen, pegawai, orang tua mahasiswa, serta pihak lain yang berhubungan dengan layanan yang diberikan oleh Universitas Brawijaya (e-complaint.ub.ac.id, 2012).

Dalam penelitian ini, penulis memilih studi kasus di Fakultas Ilmu Komputer yang berada di Universitas Brawijaya. Fakultas Ilmu Komputer atau yang disingkat FILKOM adalah sebuah fakultas yang berada di bawah naungan Universitas Brawijaya yang menyelenggarakan perkuliahan dalam ranah ilmu komputer. FILKOM pun sudah memiliki mekanisme pengaduan keluhan layanan melalui *Whatsapp* resmi yang dikelola oleh Bagian Umum dan Perlengkapan. Selain itu juga melalui pengisian *form* yang dilakukan di kantor Bagian Umum dan Perlengkapan. Bagian tersebut berfungsi untuk mendukung proses pemeliharaan dan perawatan di FILKOM sehingga perkuliahan dan kegiatan fakultas dapat terlaksana dengan lancar dan optimal.

Dalam melaksanakan fungsinya, pegawai Bagian Umum dan Perlengkapan sering kali mendapatkan pengaduan-pengaduan mengenai perawatan layanan di FILKOM. Berdasarkan wawancara yang dilakukan dengan Ferix Panji Andrianto, S.ST selaku koordinator kerumahtanggaan di Bagian Umum dan Perlengkapan FILKOM, peneliti mendapatkan informasi bahwa selama ini Bagian Umum dan Perlengkapan mendapatkan pengaduan dari berbagai macam sumber di berbagai area FILKOM. Sumber-sumber pengaduan yaitu dari *Whatsapp* resmi Bagian Umum dan Perlengkapan*, E-Complaint,* SMS, telepon, *Whatsapp* personal Pegawai Umum dan Perlengkapan, dan juga mendatangi langsung ke kantor

Umum dan Perlengkapan. Dengan adanya keluhan-keluhan tersebut, menurut Ferix, pihak *maintenance* di Bagian Umum dan Perlengkapan FILKOM, menyatakan bahwa Bagian Umum dan Perlengkapan selalu merespon melalui *Whatsapp.* Namun pihak pengadu sering tidak sabar sehingga memutuskan untuk mengadukan lagi menggunakan media lain yang dianggap lebih cepat responnya seperti menghubungi secara personal atau mendatangi langsung kantor Umum dan Perlengkapan. Hal ini mengakibatkan kurang efektifnya fungsi *Whatsapp* resmi tersebut dan juga mengakibatkan pengaduan-pengaduan yang berjumlah *double* sehingga sulit untuk menuliskan riwayat keluhan-keluhan tersebut pada *database* dan pencarian data menjadi sulit serta pengaduan tidak tertangani. Selain itu, kerap terjadi kesalahpahaman dari pihak pelapor akibat sulitnya mendapatkan info jika suatu keluhan sudah ditindaklanjuti.

Masalah lain yang muncul selain pengaduan layanan adalah *maintenance.* Keterbatasan catatan perawatan menyebabkan adanya kesulitan dalam *monitoring* status suatu barang sehingga pegawai Umum dan Perlengkapan harus sering memantau secara langsung untuk melakukan *monitoring.* Hal tersebut justru membuat Pegawai Umum dan Perlengkapan tidak sempat melakukan pengecekan keluhan-keluhan yang diadukan melalui *Whatsapp* resmi. Hal ini tentu merugikan Umum dan Perlengkapan karena pihak pelapor akan mengeluhkan kinerja yang kurang tanggap.

Berdasarkan masalah yang diangkat ini, perlu adanya tindak lanjut sebagai solusi permasalahan. Hal ini dimaksudkan agar permasalahan yang terjadi tidak semakin berlarut-larut. Oleh karena itu terdapat beberapa solusi yang dapat diberikan. Sebagai mahasiswa di FILKOM, salah satu solusi yang ingin diberikan oleh penulis adalah dengan pembuatan aplikasi pengaduan layanan berbasis *Android.* Aplikasi ini memiliki fitur-fitur untuk mengadukan layanan, penjadwalan *maintenance* secara berkala, mengubah status pengaduan, menerima notifikasi pengaduan, dan lain-lain. Berdasarkan hasil komunikasi melalui wawancara dengan perwakilan Bagian Umum dan Perlengkapan, Wakil Dekan II FILKOM, Kepala Tata Usaha FILKOM, beberapa dosen FILKOM, dan beberapa mahasiswa FILKOM, dapat disimpulkan bahwa solusi ini adalah solusi yang paling tepat karena memiliki fitur-fitur yang diinginkan penggunanya.

Penelitian dengan topik Pengaduan Layanan pernah dilakukan sebelumnya oleh Firdausy namun masih menggunakan proses bisnis yang lama dan sudah tidak digunakan serta belum dilengkapi fitur untuk *maintenance* (Firdausy, Wicaksono, & Pradana, 2018). Dengan ini maka dilakukan penelitian dengan judul "Pembangunan Aplikasi *Mobile* Pengaduan Layanan dan *Maintenance* Berbasis *Android* Pada Fakultas Ilmu Komputer Universitas Brawijaya", sehingga dapat memaksimalkan proses pengaduan layanan di FILKOM UB.

1.2 Rumusan Masalah

1. Bagaimana rekayasa kebutuhan untuk membangun aplikasi pengaduan layanan berbasis *Android* pada Fakultas Ilmu Komputer Universitas Brawijaya?

2. Bagaimana perancangan untuk membangun aplikasi pengaduan layanan berbasis *Android* pada Fakultas Ilmu Komputer Universitas Brawijaya?
3. Bagaimana implementasi dari pengembangan aplikasi pengaduan layanan berbasis *Android* pada Fakultas Ilmu Komputer Universitas Brawijaya?
4. Bagaimana pengujian dari aplikasi pengaduan layanan berbasis *Android* pada Fakultas Ilmu Komputer Universitas Brawijaya?

1.3 Tujuan

1. Mendapatkan hasil analisis kebutuhan untuk membangun aplikasi pengaduan layanan berbasis Android pada Fakultas Ilmu Komputer Universitas Brawijaya.
2. Mendapatkan hasil perancangan untuk membangun aplikasi pengaduan layanan berbasis Android pada Fakultas Ilmu Komputer Universitas Brawijaya.
3. Mendapatkan hasil implementasi dari pengembangan aplikasi pengaduan layanan berbasis Android pada Fakultas Ilmu Komputer Universitas Brawijaya.
4. Mendapatkan hasil pengujian dari aplikasi pengaduan layanan berbasis Android pada Fakultas Ilmu Komputer Universitas Brawijaya.

1.4 Manfaat

1. Manfaat bagi penulis
   Penulis mendapatkan pengetahuan baru mengenai mekanisme pengaduan layanan dan mekanisme *maintenance* yang ada di Fakultas Ilmu Komputer.
2. Manfaat bagi akademis
   Memberikan gambaran umum terkait sistem pengaduan di Fakultas Ilmu Komputer Universitas Brawijaya sehingga dapat menjadi acuan atau referensi penelitian dalam bidang yang serupa khususnya pembangunan sistem pengaduan pengaduan layanan dan *maintenance*.
3. Manfaat bagi sivitas Fakultas Ilmu Komputer
   Memberikan mekanisme lebih mudah dan lebih efisien dalam mengadukan layanan di FILKOM. Serta dapat mengetahui status dari pengaduan sehingga sivitas dapat melakukan *monitoring* yang sudah diajukan.
4. Manfaat bagi Bagian Umum dan Perlengkapan FILKOM
   Memaksimalkan fungsinya dalam hal merespon dan menindaklanjuti pengaduan layanan agar menjadi lebih efisien. Serta mempermudah dalam melakukan *maintenance* dengan adanya jadwal dan riwayat *maintenance* suatu barang. Sehingga dapat menciptakan pelayanan yang lebih baik di Fakultas Ilmu Komputer Universitas Brawijaya.

1.5 Batasan Masalah

Agar permasalahan yang dirumuskan lebih terfokus dan tidak terjadi pelebaran topik, maka penelitian ini dibatasi dalam hal berikut.

1. Penelitian hanya mencakup pengaduan layanan dan *maintenance* untuk perlengkapan di wilayah Fakultas Ilmu Komputer Universitas Brawijaya.
2. Ranah yang dilakukan bagian kerumahtanggaan oleh Bagian Umum dan Perlengkapan meliputi ruangan, barang-barang, taman, dan kebersihan yang difasilitasi oleh FILKOM.
3. Pada penelitian ini, data NIM dan NIP yang disimpan dalam *database* sistem merupakan data *dummy,* karena data yang sesungguhnya adalah data yang bersifat rahasia.

1.6 Sistematika Pembahasan

1. BAB 1 PENDAHULUAN

   Bab pendahuluan berisi tentang latar belakang, rumusan masalah, tujuan, manfaat, batasan masalah, dan sistematika pembahasan.

2. BAB 2 LANDASAN KEPUSTAKAAN

   Bab 2 berisi tentang landasan kepustakaan mengenai penelitian terdahulu, pengaduan layanan, Fakultas Ilmu Komputer Universitas Brawijaya, *platform mobile* berbasis *android*, *web services,* model SDLC *Waterfall*, pemodelan berorientasi objek, dan pengujian perangkat lunak.

3. BAB 3 METODOLOGI

   Bab 3 menjelaskan tentang langkah-langkah yang dilakukan dalam penelitian. Langkah-langkah dalam penelitian ini yaitu studi literatur, rekayasa kebutuhan, perancangan dan implementasi, pengujian, serta kesimpulan dan saran.

4. BAB 4 REKAYASA KEBUTUHAN

   Bab 4 membahas tentang hasil rekayasa kebutuhan yang dimulai dari proses elisitasi kebutuhan dan analisis kebutuhan, kemudian pemodelan kebutuhan dari aplikasi yang akan dibuat.

5. BAB 5 PERANCANGAN DAN IMPLEMENTASI

   Bab 5 menjelaskan tentang perancangan aplikasi yang akan dibuat berdasarkan hasil rekayasa kebutuhan yang telah dibuat. Selanjutnya juga membahas implementasi aplikasi yang dikembangkan berdasarkan perancangan yang telah dibuat.

6. BAB 6 PENGUJIAN

   Bab 6 membahas tentang pengujian yang dilakukan terhadap aplikasi yang telah dibuat.

7. BAB 7 PENUTUP

   Pada bab ini akan berisi tentang kesimpulan yang diperoleh dari pengembangan aplikasi dalam penelitian ini dan saran untuk pengembangan lebih lanjut.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

Pada tahun 2015, Nur Lutfiyana melakukan penelitian dengan judul "Rancang Bangun Sistem Informasi Layanan Pengaduan Nasabah Kartu Kredit Berbasis Web". Latar belakang penelitian ini adalah seringnya perusahaan perbankan penyedia layanan kartu kredit menerima pengaduan dari nasabah melalui telepon. Hal ini menyebabkan nasabah mengalami kesulitan menghubungi *customer service* yang dikarenakan semua *customer service* sedang melayani nasabah lain atau dikarenakan jumlah *customer service* yang terbatas. Kemudian, pengaduan dari nasabah oleh petugas *customer service* dimasukkan ke dalam data pengaduan menggunakan Microsoft Excel, hal ini dapat menyebabkan terjadinya kehilangan data, kesulitan pencarian data sampai dengan tidak tertanganinya pengaduan dari nasabah. Sehingga dibuatlah sistem informasi berbasis web yang memberikan kemudahan kepada nasabah kartu kredit dalam menyampaikan pengaduan dan mempermudah petugas *customer service* dalam mengontrol atau mencari pengaduan yang belum ditangani. Pengembangan sistem ini menggunakan *System Development Life Cycle* (SDLC) model *Waterfall* dan perancangannya menerapkan *Entity Data Relationship* (ERD) dan *Logical Record Structure (*LRS), implementasinya menggunakan bahasa pemrograman PHP dan *database management system* MySQL, dan pengujiannya menggunakan *black box testing.* Hasilnya adalah sistem ini dapat mempermudah proses pengaduan dan penanganannya (Lutfiyana, 2015).

Reetesh V. Golhar, Prasann A. Vyawahare, Pavan H. Borghare, dan Ashwini Manusmare pada tahun 2016 melakukan penelitian mengenai aplikasi *Android* yang berjudul "Design and implementation of android base mobile app for an institute". Latar belakang penelitian ini adalah selama bertahun-tahun proses penyampaian papan pengumuman dan pemberitahuan penting mengenai akademik masih dilakukan secara manual sehingga menghabiskan waktu dan tidak efisien. Sehingga dibuatlah aplikasi yang menghubungkan semua departemen dari suatu lembaga seperti administrasi, akun, bagian siswa dan modul lainnya. Dengan aplikasi ini pemberitahuan bisa didapatkan melalui *email* (Golhar, Vyawahare, Borghare, & Manusmare, 2016).

Penelitian mengenai aplikasi *mobile* dilakukan oleh Rizal Arif Zulfikar dan Ahmad Afif Supianto pada tahun 2018 di Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK) yang berjudul "Rancang Bangun Aplikasi Antrian Poliklinik Berbasis Mobile". Latar belakang penelitian ini adalah antrian konvensional sudah menjadi polemik yang umum di masyarakat. Lamanya proses dan waktu tunggu antrian sangat mengganggu aktivitas sehari-hari. Pada instansi kesehatan seperti rumah sakit dan poliklinik pasien juga diharuskan mengantri. Hal tersebut dapat berpengaruh pada kondisi pasien. Sehingga dibuatlah solusi berupa aplikasi yang memiliki fitur untuk memilih variasi jadwal poliklinik dan pemberian informasi antrian yang sedang berjalan. Pengembangannya berbasis pada *mobile phone*,

perancangan kode menggunakan metode MVC, dan pengembangan aplikasi menggunakan *hybrid mobile web framework*. Pengujian sistem menggunakan *White Box, Black Box, dan Usability Testing.* Hasilnya adalah aplikasi dapat digunakan dengan baik dan dapat mempermudah calon pasien poliklinik (Zulfikar & Supianto, 2018).

Sebelumnya juga telah dilakukan penelitian dengan topik yang sama yaitu Bagian Umum dan Perlengkapan di FILKOM mengenai pengaduan sarana dan prasarana. Penelitian ini dilakukan pada tahun 2018 oleh Danniar Reza Firdausy. Latar belakang penelitiannya adalah selama ini sivitas akademik FILKOM melaporkan mengenai sarana prasarana kepada pegawai Perlengkapan langsung secara lisan atau melalui *website E-Complaint* Universitas Brawijaya. Dari hal tersebut, permasalahan yang muncul adalah sulitnya identifikasi sarana prasarana yang dilaporkan sehingga membutuhkan waktu identifikasi yang cukup lama. Sehingga dibuatlah Sistem Informasi Pelaporan Sarana Prasarana yang dapat digunakan sivitas FILKOM untuk melaporkan keluhan sarana prasarana di lingkup wilayah FILKOM. Pengembangan dari sistem informasi ini dibagi menjadi dua jenis, yaitu aplikasi *Android* untuk pelapor dan *website* operator untuk pegawai Perlengkapan. Metode pengembangannya adalah menggunakan *waterfall model* dan pengujian implementasinya menggunakan *validation testing, usability testing,* dan perbandingan waktu. Hasil penelitian ini adalah perbandingan waktu yang diperlukan oleh pegawai perlengkapan untuk menerima keluhan sarana prasarana dan waktu yang diperlukan untuk melacak status dari sistem *as-is* menjadi sistem *to-be* dapat dipercepat (Firdausy, Wicaksono, & Pradana, 2018).

Berdasarkan penelitian-penelitian tersebut, maka penulis menjadikan referensi mengenai latar belakang dan metode pengembangan pada penelitian pertama (Lutfiyana, 2015). Latar belakang yang digunakan pada penelitian tersebut hampir sama dengan aplikasi yang akan dikembangkan oleh penulis yaitu *customer service* (CS) yang sering mendapatkan pengaduan-pengaduan namun jumlah CS masih terbatas dan masih mendata pengaduan tersebut melalui *Microsoft Excel.* Hal ini mengakibatkan sulitnya pencarian data pengaduan yang sudah dan belum ditangani, sehingga *customer* sering kecewa dan kinerja dinilai buruk. Selain itu, penelitian ini juga menggunakan metode SDLC *waterfall* yang menurut Lutfiyana paling sesuai untuk pengembangan aplikasi yang sudah jelas prosedur-prosedurnya dan terurut mulai dari analisis, desain, pengodean, dan pengujian. Pada penelitian kedua (Golhar, Vyawahare, Borghare, & Manusmare, 2016), penulis menjadikan referensi mengenai pengembangan aplikasi berbasis *Android.* Pengembangan berbasis *Android* dipilih karena *user-friendly* dan memberikan *user experiene* yang baik, serta difasilitasi dengan *Google Android SDK*. Pada penelitian ketiga (Zulfikar & Supianto, 2018), penulis menjadikan acuan mengenai perancangan yang menggunakan prinsip *heuristic usability*, pengimplementasian menggunakan metode MVC, dan pengujian menggunakan metode *White Box, Black Box, dan Usability Testing.* Penggunaan acuan ini didasarkan pada hasil pengujian yang menunjukkan bahwa prinsip *heuristic usability* dapat meningkatkan kenyamanan penggunaan aplikasi dan meminimalisir kemungkinan kesalahan. Pengimplementasian pada penelitian tersebut juga dapat dijadikan

acuan bagi penulis untuk memisahkan antara data dan tampilan serta cara pemrosesannya sehingga lebih terstruktur dan mudah. Selain itu pengujian pada penelitian ini juga dijadikan acuan untuk mendeteksi kesesuaian-kesesuaian *output* dan memastikan setiap fungsi tidak mengalami kesalahan saat digunakan. Kemudian pada penelitian keempat (Firdausy, Wicaksono, & Pradana, 2018), penulis menjadikan referensi mengenai latar belakang dan metode SDLC *waterfall.* Hal yang membedakan penelitian yang akan dilakukan dengan penelitian keempat adalah: 1) Acuan proses bisnis *as-is*, 2) Menghasilkan aplikasi yang berbasis *Android* bagi pengguna dan pegawai perlengkapan, 3) Proses bisnis *to-be*, 4) Pengujian implementasi yang digunakan, 5) Fitur *maintenance* belum ada.

2.2 Pengaduan Layanan

Pelayanan merupakan suatu proses keseluruhan dari pembentukan citra suatu instansi, baik melalui media berita, membentuk budaya instansi secara internal, maupun melakukan komunikasi tentang pandangan instansi kepada para pemimpin pemerintahan serta publik lainnya yang berkepentingan. Tujuan umum diberikannya kualitas pelayanan yang baik adalah agar konsumen merasakan kepuasan dan akan berdampak positif bagi instansi (Loina, 2010). Pelayanan juga dapat diartikan sebagai proses pemenuhan kebutuhan melalui aktivitas orang lain secara langsung. Pelayanan yang dimaksud ialah pelayanan yang diberikan dalam rangka untuk memenuhi kebutuhan dan kepuasan suatu warga (Moenir, 2014).

Dalam suatu instansi, pihak yang berwenang untuk melakukan pelayanan adalah bagian perlengkapan. Pihak tersebut bertanggung jawab dalam sarana dan fasilitas yang ada di suatu instansi agar dapat digunakan sesuai dengan fungsinya. Tujuan adanya perlengkapan pada suatu instansi biasanya adalah sebagai pengadaan perlengkapan sesuai kebutuhan, pendayagunaan perlengkapan yang ada secara optimal, perawatan perlengkapan yang ada secara baik, serta penghapusan perlengkapan yang rusak atau hilang. Perlengkapan yang dikelola meliputi barang yang tidak bergerak, misalnya taman dan bangunan, serta barang yang bergerak, baik yang habis pakai maupun tidak, misalnya perabot, alat kuliah, buku, alat peraga praktik, dan sebagainya.

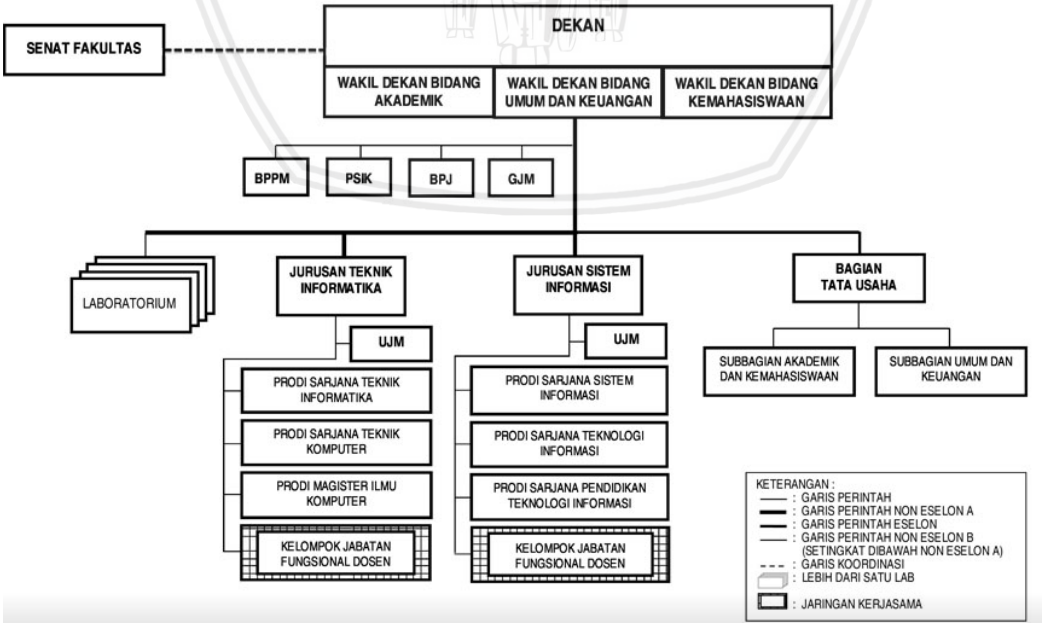
Pengaduan layanan sendiri menurut Undang-Undang Republik Indonesia Nomor 25 Tahun 2009 tentang Pelayanan Publik adalah mekanisme yang dapat digunakan oleh warga untuk menyampaikan keluhan dan protes kepada suatu pelayanan publik. Pada Undang-Undang tersebut warga mendapatkan jaminan untuk mengadukan penyelenggaraan layanan dan memperoleh respon atas pengaduan yang diajukannya (Dwiyanto, 2018). Hal-hal yang harus diperhatikan dalam menyelesaikan pengaduan masyarakat adalah: 1) Prioritas penyelesaian pengaduan, 2) Penentuan pejabat yang menyelesaikan pengaduan, 3) Prosedur penyelesaian pengaduan, 4) Rekomendasi penyelesaian pengaduan, 5) Pemantauan dan evaluasi pengaduan, 6) Pelaporan proses dan hasil penyelesaian pengaduan kepada pimpinan, 7) Penyampaian hasil penyelesaian pengaduan kepada yang mengadukan, dan 8) Dokumentasi penyelesaian pengaduan (Sulila, 2015).

2.2.1 Pengaduan Layanan dan *Maintenance* di FILKOM UB

Fakultas Ilmu Komputer yang biasa disebut FILKOM dibentuk berdasarkan Surat Keputusan Rektor Universitas Brawijaya Nomor: 516/SK/2011 sebagai Program Teknologi Informasi dan Ilmu Komputer (PTIIK) pada 27 Oktober 2011. Pada tahun 2015, status Program pada Program Teknologi Informasi dan Komunikasi secara resmi diganti menjadi Fakultas dengan nama Fakultas Ilmu Komputer.

Di Fakultas Ilmu Komputer, bagian yang mempunyai tugas melaksanakan urusan kerumahtanggaan adalah Bagian Tata Usaha sebagaimana yang telah disebutkan dalam Peraturan Rektor Nomor 20 Tahun 2016 Tentang Susunan Organisasi dan Tata Kerja pasal 517. Bagian Tata Usaha memiliki bagian yang salah satunya adalah Sub Bagian Umum dan Keuangan yang mempunyai tugas melakukan urusan perencanaan, keuangan, ketatalaksanaan, kepegawaian, ketatausahaan, kerumahtanggaan, dan pengelolaan barang milik negara sebagaimana yang telah disebutkan pada pasal 520. Dan pada pasal 26 disebutkan bahwa kerumahtanggaan mempunyai tugas melakukan urusan keamanan, kebersihan, pertamanan, pengaturan penggunaan, pemeliharaan, dan perawatan sarana kantor, serta urusan kerumahtanggaan lainnya.

Berdasarkan peraturan-peraturan tersebut, maka dapat disimpulkan bahwa yang bertanggung jawab dalam pelaksanaan pengaduan layanan serta penindaklanjutan mengenai layanan dan *maintenance* merupakan tanggung jawab Sub Bagian Umum dan Keuangan khususnya bagian Umum dan Perlengkapan yang merupakan satuan kerja dalam Sub Bagian Umum dan Keuangan. Sub Bagian tersebut berada di bawah Bagian Tata Usaha dan berada di bawah garis perintah Wakil Dekan Bidang Umum dan Keuangan. Hal ini dapat ditunjukkan pada Gambar 2.1.



Gambar 2.1 Struktur Organisasi Fakultas Ilmu Komputer (filkom.ub.ac.id, 2017)

2.3 *Platform* Berbasis *Android*

Android merupakan *software* yang digunakan pada perangkat *mobile* yang mencakup sistem operasi, *middleware*, dan aplikasi kunci yang dirilis oleh Google. Sehingga *Android* mencakup keseluruhan sebuah aplikasi, mulai dari sistem operasi sampai pada pengembangan aplikasi itu sendiri. Pengembangan aplikasi pada *platform Android* ini menggunakan dasar bahasa pemrograman Java.

Pada saat ini pengguna *Android* semakin banyak karena memiliki berbagai aplikasi yang relevan, dapat memenuhi kebutuhan, dan bersifat *open source* sehingga dapat dikembangkan di berbagai manufaktur *smartphone*. *Platform* bersifat *open-source* ini dapat dikembangkan untuk membangun aplikasi tanpa ada lisensi ke produsen atau vendor tertentu dan dapat dimodifikasi (EMS, 2015).

2.4 *Web Services*

Menurut *website* w3c.org, w*eb service* adalah sebuah sistem *software* yang didesain untuk mendukung interoperabilitas interaksi mesin ke mesin melalui sebuah jaringan. Secara umum, *web service* dapat diidentifikasikan dengan menggunakan URL seperti hanya web pada umumnya. Namun yang membedakan *web service* dengan web pada umumnya adalah interaksi yang diberikan oleh *web service*. Berbeda dengan URL web pada umumnya, URL *web service* hanya mengandung kumpulan informasi, perintah, konfigurasi atau sintaks yang berguna membangun sebuah fungsi-fungsi tertentu dari aplikasi (w3c.org, 2002).

2.4.1 *Firebase Database Management System*

Menurut *website* firebase.google.com, *Firebase* adalah aplikasi pengembangan web dan *mobile* yang berbasis *Backend as a Service* atau BAAS, yang artinya *Firebase* merupakan penyimpanan data bersifat NoSQL. *Firebase* memiliki beberapa layanan diantaranya *Realtime Database, Cloud Storage, Authentication,* dan *Hosting. Firebase Realtime* adalah *database* yang menggunakan hosting *cloud* sehingga data disimpan sebagai JSON dan disinkronkan secara *realtime* ke setiap klien yang terhubung (firebase.google.com, 2010).

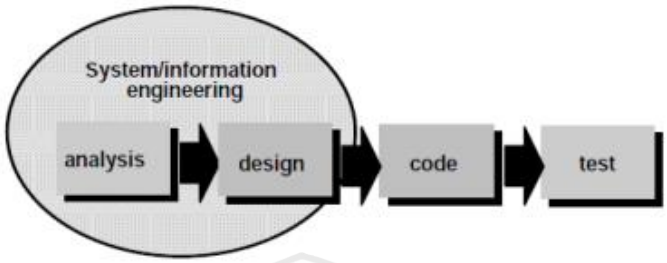
2.4.2 *One Signal*

Menurut *website* onesignal.com, One Signal adalah *service push notification* untuk *website* dan aplikasi *mobile*. OneSignal mendukung sebagian besar *native* dan *mobile platform* dengan menyediakan SDK untuk masing-masing *platform*, *RESTful server* API, dan *online dashboard* untuk melihat performa, statistik penggunaan, maupun operasi *push notif* (onesignal.com, 2014).

2.5 Model *Waterfall*

Waterfall Model merupakan salah satu model SDLC berbentuk linear sekuensial yang diperkenalkan oleh Royce pada tahun 1970 (Mishra & Dubey, 2013). Model SDLC ini terdiri dari tahapan *requirements analysis, design, implementation,* dan *testing* yang berurutan sehingga proses pengembangan tidak akan berlanjut ke tahap berikutnya apabila tahap sebelumnya belum sepenuhnya selesai.

Menurut Pressman (2010), *Waterfall Model* sebaiknya digunakan ketika semua kebutuhan dan persyaratan dari sistem telah dipahami dengan baik dan tidak mungkin berubah secara radikal selama pengembangan sistem karena model SDLC ini menekankan perencanaan pada tahap awal (Pressman, 2010). Tahap-tahap pengembangan aplikasi pada metode ini dapat dilihat pada Gambar 2.2.



Gambar 2.2 Diagram SDLC *Waterfall Model* (Pressman, 2002)

1. Tahap analisis, merupakan proses analisis kebutuhan sistem. Pengembang mengumpulkan data-data sebagai bahan pengembangan sistem. Pengumpulan data dapat dilakukan dengan teknik wawancara, teknik observasi, dan teknik kuesioner. Kemudian dilakukan analisis terhadap data-data yang sudah dikumpulkan. Berdasarkan analisis, maka data-data tersebut digambarkan ke dalam suatu pemodelan kebutuhan seperti *use case diagram.*
2. Tahap Perancangan, yaitu proses multi langkah yang berfokus pada empat atribut, yaitu struktur data, arsitektur perangkat lunak, representasi *interface*, dan detail prosedural. Proses desain menterjemahkan hasil analisis ke dalam representasi perangkat lunak.
3. Tahap *Code*, yaitu tahap penerjemahan desain ke dalam program perangkat lunak. Pada tahap pengimplementasian ke dalam kode program akan bergantung pada hasil desain perangkat lunak pada tahap sebelumnya.
4. Tahap Pengujian, yaitu untuk mengetahui kesesuaian hasil *output* dari sistem dengan kebutuhan yang telah dirancang pada tahap analisis (Pressman, 2010).

2.6 Pemodelan Berorientasi Objek

Pemodelan adalah bahasa buatan yang dapat merepresentasikan informasi atau suatu sistem ke dalam struktur yang didefinisikan dalam beberapa aturan yang konsisten. Aturan tersebut digunakan untuk interpretasi dari komponen-komponen pada struktur bahasa pemodelan. Penulisan untuk bahasa pemodelan dapat direpresentasikan dalam bentuk grafis atau tekstual. Sedangkan pemodelan Berorientasi Objek adalah pemodelan yang objek-objeknya memiliki karakteristik tersendiri dalam suatu perangkat lunak. Pemodelan berorientasi objek umumnya direpresentasikan dalam bentuk UML (*Unified Model Language).* UML adalah visualisasi dan dokumentasi hasil analisis dan desain yang berisi sintak dalam memodelkan sistem. UML juga merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menggambarkan sebuah sistem *software* yang terkait dengan objek (Haviluddin, 2011). UML mendefinisikan diagram-diagram sebagai *Use case diagram, Class diagram, Statechart diagram, Activity diagram, Sequence diagram, Component* dan *Deployment diagram* (Nugroho, 2010).

2.7 Pengujian Perangkat Lunak

Pengujian atau *Testing* adalah sebuah metode untuk melakukan verifikasi dengan tujuan mencari kesalahan pada sebuah aplikasi. Dari hasil verifikasi tersebut, hasilnya didokumentasikan dan ditinjau ulang lebih lanjut untuk menentukan kapabilitas dari kebutuhan perangkat lunak (Rizky, 2011).

2.7.1 Pengujian Kotak Hitam (*Black Box Testing)*

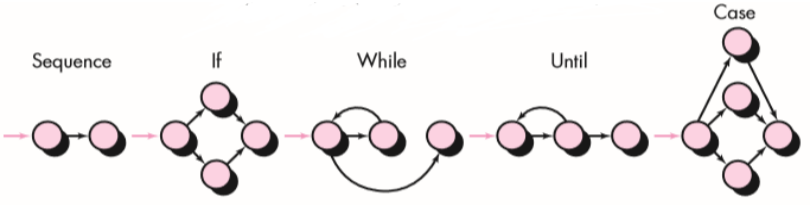
Black box Testing adalah tipe pengujian yang memperlakukan perangkat lunak yang tidak diketahui kinerja internalnya (Rizky, 2011). Para penguji memandang perangkat lunak seperti sebuah "kotak hitam" yang tidak dapat dilihat isinya namun cukup dilakukan proses pengujian di bagian luar. Jenis pengujian ini hanya memandang perangkat lunak dari sisi spesifikasi kebutuhan yang telah didefinisikan di awal perancangan. Beberapa teknik pengujian yang tergolong dalam *black box testing* yaitu *Cause Effect Graph, Equivalence Partitioning*, *Validation Testing, Boundary Value Analysis, Random Data Selection,* dan *Feature Test.* Salah satu teknik pengujian *black box* adalah *Validation Testing.* Pengujian ini menggunakan *software requirements spesifiaction* (SRS) sebagai panduan dalam memilih *validation test criteria*-nya (Pressman, 2010).

2.7.2 Pengujian Kotak Putih (*White Box Testing)*

White box Testing adalah jenis pengujian yang lebih fokus pada isi atau *source code* dari perangkat lunak. *White box Testing* membutuhkan waktu pengujian yang lebih lama karena membutuhkan ketelitian dari penguji serta kemampuan teknis pemrograman bagi para pengujinya (Rizky, 2011). Jenis pengujian ini hanya dapat dilakukan jika perangkat lunak telah dinyatakan selesai dan telah melalui tahap analisis awal. Prinsip keluarannya yaitu sebagai berikut.

1. Menjamin bahwa semua alur program independen (dalam bentuk modul, *form*, prosedur, *class*, dan lainnya) telah diuji minimal satu kali.
2. Telah melakukan pengujian pada semua kondisi percabangan dengan nilai *true* dan *false*.
3. Telah melakukan pengujian terhadap semua jenis perulangan dengan kondisi normal dan kondisi yang dianggap melebihi nilai batas perulangan.
4. Telah melakukan pengujian struktur data internal agar terjaga validitasnya.

Salah satu metode yang dapat dilakukan dalam pengujian *white box* adalah *Basic Path Testing* yang diusung oleh Tom McCabe pada tahun 1976. Metode ini memungkinkan penguji untuk mendapatkan tingkat kompleksitas dari perancangan algoritme sebagai petunjuk dalam mendefinisikan jalur eksekusi. Tahap awal dalam pengujian *white box* menggunakan *Basic Path Testing* adalah pendefinisian *flow graph*. *Flow graph* memberikan gambaran terkait dengan alur prosedur yang akan dilalui ketika prosedur dieksekusi. Dalam penggambaran strukturnya, notasi *flow graph* menggunakan simbol-simbol seperti yang terdapat pada Gambar 2.3.



Gambar 2.3 Simbol *Flow Graph* (Pressman, 2010)

Menurut Pressman (2010), komponen sebuah *flow graph* terdiri dari *node* berbentuk lingkaran yang merupakan representasi dari satu atau lebih perintah yang tidak bercabang, *edge* atau *link* berbentuk arah panah merepresentasikan alur eksekusi prosedur, dan *region* yang merepresentasikan sebuah sebuah ruang yang dibatasi oleh *node* dan *edge*, termasuk ruang di luar graf.

Dari hasil *flow graph* yang telah terbentuk, dapat dilakukan perhitungan *Cyclometic Complexity* untuk mengetahui seberapa kompleks logika dari prosedur yang diuji. Perhitungan *Cyclometic Complexity* V(G) dapat dilakukan dengan tiga cara, yakni sebagai berikut.

1. Jumlah *region* dari *flow graph*
2. G pada V(G) dapat diperoleh dengan V(G) = jumlah *edge* – jumlah *node* + 2
3. G pada V(G) juga dapat diperoleh dengan V(G) = jumlah *predicate node* + 1, *predicate node* merupakan *node* yang memiliki satu atau lebih *output*.

Dari hasil perhitungan *Cyclometic Complexity* juga dapat diketahui berapa banyak jalur independen yang harus digunakan untuk melakukan pengujian. Jalur independen adalah jalur yang dilewati oleh program dengan setidaknya terdapat satu set pernyataan (*node*) atau kondisi yang baru. Jalur inilah yang kemudian digunakan sebagai acuan untuk membuat kasus uji (Pressman, 2010).

2.7.3 Pengujian *Usability*

Usability testing adalah pengujian non fungsional yang dilakukan untuk mengukur tingkat pengalaman interaksi produk oleh pengguna dengan melakukan berbagai tugas yang telah ditentukan (Lazar, et al., 2017). Menurut Nielsen (2000), untuk mengidentifikasi permasalahan dalam desain sebuah sistem cukup menggunakan lima orang. Pengujian dengan lima orang memungkinkan untuk ditemukannya masalah usability lebih dari 80% dan jika menggunakan lebih banyak peserta tes maka peningkatannya tidak terlalu signifikan yaitu hanya 20%. Hasil yang didapat mendekati rasio maksimum pengujian pengguna ketika menggunakan 5 partisipan (Nielsen, 2000).

Pengujian *usability* dapat dilakukan dengan berbagai metode seperti skenario tugas dan USE *questionnaire.* Skenario tugas adalah tindakan yang peneliti minta kepada peserta untuk mencoba antarmuka yang ingin dilakukan pengujian. Sedangkan USE merupakan rangkaian kuesioner yang mencakup aspek *usefullnes, satisfaction,* dan *ease of use* sebagai alat untuk evaluasi produk (Nielsen, 2014). Untuk memudahkan pengujian *usability* maka digunakan suatu alat yaitu skala *likert.* Skala likert melakukan pengukuran sifat-sifat individu seperti sikap atau pengetahuan dengan menghitung skor total dari setiap pertanyaan. Dalam sifat

individu digunakan beberapa pertanyaan dengan merespon lima pilihan pada setiap pertanyaan yaitu sangat setuju, setuju, netral, tidak setuju dan sangat tidak setuju (Likert, 1932). Penjelasan skala *likert* dapat dilihat pada Tabel 2.1.

Tabel 2.1 Penjelasan Skala *Likert*

Nilai	Keterangan
1	**Sangat tidak setuju**, dari pertanyaan yang diberikan, responden menyatakan bahwa aplikasi jauh dari ekspektasi.
2	**Tidak setuju**, dari pertanyaan yang diberikan, responden menyatakan bahwa tidak sependapat.
3	**Netral,** dari pertanyaan yang diberikan, responden menyatakan bahwa tidak memiliki pandangan.
4	**Setuju**, dari pertanyaan yang diberikan, responden menyatakan bahwa sependapat.
5	**Sangat setuju**, dari pertanyaan yang diberikan, responden menyatakan bahwa aplikasi yang dibuat memenuhi dan melebihi ekspektasi.

2.7.4 Pengujian *Compatibility*

Compatibility testing adalah pengujian non fungsional yang digunakan untuk mengecek kapabilitas atau kelayakan aplikasi ketika dijalankan di berbagai perangkat, sistem operasi, atau jaringan yang berbeda. Pengguna aplikasi biasanya akan melihat kinerja keseluruhan suatu aplikasi berdasarkan kemudahannya ketika dipasang di suatu perangkat dan tidak mentolerir adanya kegagalan pemasangan. Uji coba dilakukan dengan pemasangan aplikasi di berbagai perangkat *Android.* Setelah dilakukan uji coba, maka hasil yang didapatkan akan dihitung dengan rumus persentase kapabilitas (Arzt, et al., 2014). Persentase kapabilitas adalah pengujian yang dilakukan dengan menggunakan teknik analisis deskriptif, yaitu menganalisis persentase hasil pengujian setiap *widget* aplikasi. Persentase kapabilitas dalam pengujian dapat dihitung dengan skala Guttman. Penghitungan persentase kapabilitas yaitu sebagai berikut.

$$Persentase\ kapabilitas = \frac{nilai\ total}{nilai\ maksimum} x100\% \quad \textbf{(2.1)}$$

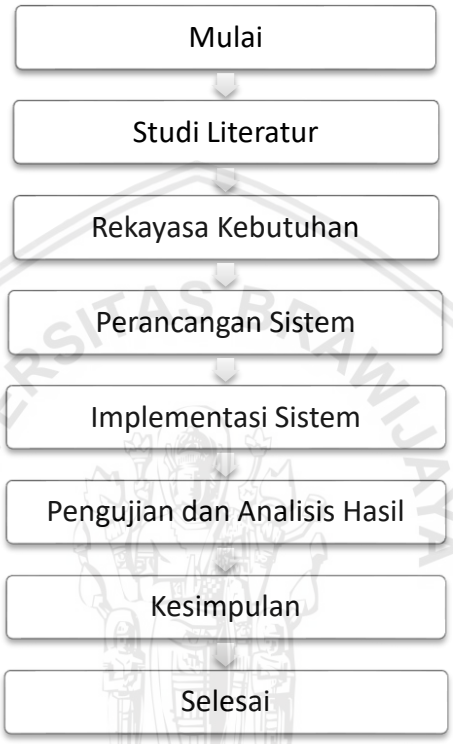
Hasil perhitungan kemudian dikonversi menjadi pernyataan predikat (Guritno, 2011). Pembagian kategori kelayakan dapat dilihat pada Tabel 2.2.

Tabel 2.2 Kategori Kelayakan Guttman

No.	Persentase	Interpretasi
1.	0% - 20%	Sangat Buruk
2.	21% – 40%	Buruk
3.	41% – 60%	Cukup
4.	61% – 80%	Baik
5.	81% – 100%	Sangat Baik

BAB 3 METODOLOGI

Bab metodologi penelitian membahas mengenai lokasi penelitian dan ruang lingkup penelitian, jadwal penelitian, serta langkah-langkah yang dilakukan dalam mengembangkan aplikasi *mobile* pengaduan layanan berbasis *Android*. Metodologi penelitian dilakukan melalui beberapa tahapan, yang dijelaskan pada Gambar 3.1.



Gambar 3.1 Metodologi Penelitian

3.1 Studi Literatur

Pada tahap ini, penggunaan studi literatur adalah sebagai bahan dasar untuk mengkaji teori-teori yang akan digunakan pada penelitian. Beberapa studi literatur yang digunakan pada penelitian ini berkaitan dengan:

1. Fakultas Ilmu Komputer Universitas Brawijaya
2. Pengaduan Layanan
3. *Maintenance*
4. *Android*
5. *Firebase*

3.2 Rekayasa Kebutuhan

Pada tahap ini, penggunaan rekayasa kebutuhan bertujuan untuk menghasilkan dasar yang akurat bagi pengembangan aplikasi, menyediakan referensi dilakukannya validasi aplikasi, serta mencapai kesepakatan antara penulis dan pengguna akhir. Pada tahap ini proses-proses yang harus dilalui yaitu elisitasi dan analisis kebutuhan serta pemodelan kebutuhan.

3.2.1 Elisitasi dan Analisis Kebutuhan

Penelitian dilakukan di Fakultas Ilmu Komputer Universitas Brawijaya. Hal ini dikarenakan, penulis menjumpai berbagai masalah yang terjadi di lokasi tersebut terutama pada bidang pengaduan layanan dan *maintenance.*

Pada proses ini, penulis menggali domain permasalahan dari mekanisme pengaduan layanan yang sudah ada di FILKOM UB dan menggali kebutuhan-kebutuhan pengguna sehingga dapat dianalisis kebutuhan-kebutuhan fungsional dan non fungsional untuk pengembangan aplikasi. Penulis melakukan penggalian dan pengumpulan data sebagai dasar untuk tahap-tahap selanjutnya pada penelitian ini. Ada dua metode penggalian dan pengumpulan data, antara lain dengan metode wawancara, yakni metode pengumpulan data yang dilakukan dengan cara bertatap muka langsung dengan narasumber untuk mendapatkan informasi tertentu. Narasumber pertama yaitu beberapa dosen dan mahasiswa FILKOM yang dapat memberikan informasi mengenai mekanisme pengaduan layanan dan mekanisme *maintenance* di FILKOM. Narasumber kedua adalah Ferix Panji Andrianto sebagai Koordinator Kerumahtanggaan Bagian Umum dan Perlengkapan yang dapat memberikan informasi mengenai mekanisme pengaduan layanan dan *maintenance* serta masalah-masalahnya kemudian memberikan klarifikasi mengenai masalah-masalah yang didapatkan berdasarkan wawancara dengan beberapa mahasiswa dan dosen. Narasumber selanjutnya adalah Suprapto, S.T, M.T sebagai Wakil Dekan bidang Umum dan Keuangan serta Sri Suyati Luhur Utami, S.E sebagai Kepala Tata Usaha yang dapat memberikan informasi dan klarifikasi mengenai masalah-masalah yang terjadi pada mekanisme pengaduan layanan dan mekanisme *maintenance* di FILKOM. Berdasarkan wawancara tersebut didapatkan informasi mengenai mekanisme pengaduan layanan dan *maintenance* di FILKOM yang sudah ada sehingga dapat diketahui berbagai permasalahan secara jelas dan berbagai kebutuhan yang melatarbelakangi pengembangan aplikasi pengaduan layanan dan *maintenance*.

Tahap selanjutnya adalah dilakukan observasi terhadap dokumen dengan cara mengumpulkan dokumen-dokumen yang berisi data pendukung antara lain struktur organisasi, *Standar of Procedure* Umum dan Perlengkapan, dan Peraturan Rektor Universitas Brawijaya. Kemudian proses selanjutnya adalah analisis proses bisnis *as-is.* Pada proses ini penulis melakukan analisis mengenai proses bisnis pengaduan layanan dan proses bisnis *maintenance* yang selama ini sudah ada di FILKOM UB berdasarkan hasil pada tahap sebelumnya. Kemudian berdasarkan proses bisnis *as-is*, maka penulis merancang pemodelan proses bisnis *to-be* pengaduan layanan dan proses bisnis *to-be maintenance* pada aplikasi yang akan diusulkan yang sesuai dengan kebutuhan pengguna. Berdasarkan proses bisnis yang diusulkan, penulis mengidentifikasi dan mengklasifikasikan aktor berdasarkan kepentingan dan perannya dalam pengaduan layanan di FILKOM. Setelah itu maka dibuat spesifikasi kebutuhan, yaitu tahap untuk menjelaskan kebutuhan-kebutuhan pengguna mengenai mekanisme pengaduan layanan dan *maintenance* yang telah didefinisikan pada proses sebelumnya secara lebih detail dan tepat sehingga dapat menjadi dasar bagi perancangan dan implementasi

pengembangan aplikasi pengaduan layanan dan *maintenance*. Pada tahap ini juga didefinisikan penomoran untuk memudahkan identifikasi kebutuhan. Kebutuhan yang didefinisikan yaitu kebutuhan fungsional yang dapat dilakukan oleh aplikasi dan dapat memenuhi kebutuhan pengguna. Serta kebutuhan non fungsional yang menjelaskan spesifikasi sistem yang dimiliki aplikasi.

3.2.2 Pemodelan Kebutuhan

Berdasarkan hasil elisitasi dan analisis, maka proses selanjutnya adalah pemodelan kebutuhan. Pada tahap ini akan dijelaskan mengenai metode pemodelan yang digunakan yaitu pemodelan berorientasi objek (*object oriented design).* Berdasarkan kebutuhan yang telah diidentifikasi, maka penulis merepresentasikan kedalam bentuk diagram *use case* yang didalamnya berisi aktor-aktor yang menggunakan aplikasi beserta fungsional-fungsional yang dapat mereka lakukan masing-masing. Kemudian dilanjutkan dengan pembuatan *use case scenario* yang menjelaskan langkah-langkah yang harus dilalui oleh pengguna untuk menjalankan masing-masing fungsional.

3.3 Perancangan dan Implementasi

Pada tahap perancangan aplikasi yang akan dibangun, penulis menggunakan pemodelan berorientasi objek. Representasi pemodelan berorientasi objek ini yaitu menggunakan beberapa UML diagram yaitu *Sequence Diagram* dan juga *Class Diagram*. Terdapat pula perancangan komponen yang menjelaskan algoritme dan juga perancangan antarmuka (*User Interface*). Pada tahap ini, perancangan data dibuat menggunakan tabel dikarenakan DBMS yang digunakan adalah *Firebase* yang merupakan NoSQL dan tidak ada relasi antar tabel dalam *database.* Tahap selanjutnya yaitu implementasi yang dilakukan berdasarkan bentuk perancangan aplikasi yang telah dirancang sebelumnya. Pemrograman dilakukan dengan menggunakan *Android* Studio bahasa java untuk mengembangkan aplikasi berbasis *Android*. Selain itu pengembangan aplikasi ini juga menggunakan *Firebase* sebagai DBMS-nya.

3.4 Pengujian dan Analisis Hasil

Pengujian Aplikasi berguna untuk melakukan validasi serta verifikasi terhadap aplikasi yang telah dibangun. Pengujian tersebut ditujukan untuk membuktikan bahwa aplikasi yang dibangun sudah menyelesaikan masalah yang ada, sudah sesuai dengan kebutuhan yang telah dideskripsikan sebelumnya, serta seberapa optimal aplikasi saat dijalankan. Pada penelitian ini terdapat dua jenis metode pengujian fungsional dan dua jenis metode pengujian non fungsional.

Pengujian fungsional terdiri dari pengujian *black box* dan pengujian *white box.* Pengujian *black box* dilakukan untuk menguji fungionalitas sistem tampak dari luar. Pada penelitian ini, pengujian *black box* dilakukan menggunakan metode *validation testing* untuk mendeteksi kesesuaian fungsi dengan hasil yang diharapkan. Selanjutnya pengujian *white box* untuk menguji kelayakan kode yang telah diimplementasikan. Pengujian *white box* yang digunakan dalam penelitian ini adalah pengujian unit. Pengujian unit dilakukan menggunakan metode

McCabe's basic path testing yang menggunakan *test case.* Dengan pembuatan *test case*, maka penulis akan mendapatkan ukuran kompleks *logical* dari perancangan prosedural dan menggunakan ukuran ini sebagai petunjuk untuk mendefinisikan himpunan jalur yang akan diuji.

Pengujian non fungsional pada penelitian ini dilakukan dengan pengujian *usability* dan pengujian *compatibility.* Pengujian *usability* kepada pengguna dilakukan untuk mengukur tingkat pengalaman interaksi antara pengguna dengan aplikasi yang telah dibuat. Sehingga dapat diidentifikasi permasalahan pada interaksi tersebut yang harus diperbaiki dan tingkat kepuasan pengguna untuk meningkatkan kualitas aplikasi. Pada penelitian ini, perwakilan pengguna pada pengujian *usability* berjumlah lima orang yang mewakili *role* Kasubag, pegawai bagian umum dan perlengkapan, serta pengadu. Pengujian *usability* kepada lima responden tersebut dilakukan dengan metode skenario tugas dan kuesioner USE. Metode skenario tugas adalah metode pengujian *usability* yang dilakukan dengan memberikan *task-task* tertentu kepada pengguna agar pengguna melakukan *task* tersebut sesuai dengan pengetahuannya. Metode ini memungkinkan peneliti untuk mengidentifikasi keberhasilan interaksi pengguna dengan aplikasi dalam melakukan suatu fungsi tertentu. Sedangkan metode kuesioner USE adalah metode pengujian *usability* yang dilakukan dengan memberikan kuesioner kepada pengguna yang telah mencoba menggunakan aplikasi. Metode kuesioner USE memiliki tiga aspek penting yaitu *usefulness, satiscfaction, dan ease of use.* Metode ini memungkinkan peneliti untuk mengidentifikasi manfaat, kepuasan, dan kemudahan aplikasi bagi pengguna yang telah mencoba aplikasi pada metode skenario tugas. Nilai rata-rata masing-masing metode pengujian *usability* menunjukkan seberapa besar kasubag, pegawai bagian umum dan perlengkapan, serta pengadu setuju bahwa aplikasi dapat digunakan dan diterapkan pada layanan pengaduan dan *maintenance* FILKOM. Pengujian non fungsional selanjutnya adalah *compatibility* untuk mengecek kapabilitas aplikasi ketika dijalankan di berbagai perangkat, sistem operasi, atau jaringan yang berbeda. Pengujian *compatibility* pada penelitian ini dilakukan dengan melakukan instalasi aplikasi di berbagai *smartphone android* dengan dimensi, resolusi, sistem operasi, dan API yang berbeda-beda. Aspek yang diuji adalah keberhasilan instalasi aplikasi pada *smartphone* dan keberhasilan dalam menampilkan *widget.*

Berdasarkan hasil dari pengujian yang didapat maka dilakukan analisis untuk mengetahui apakah semua fungsionalitas sistem yang terdapat pada spesifikasi kebutuhan telah terpenuhi dan berjalan dengan baik.

3.5 Kesimpulan dan Saran

Pengambilan kesimpulan didasarkan pada hasil dari analisis hasil pada tahap sebelumnya. Pada kesimpulan yang diambil, dapat pula ditarik saran-saran yang berguna untuk pengembangan sistem pada waktu yang akan datang serta dapat menjadi acuan atau referensi penelitian dalam bidang yang serupa khususnya pembangunan sistem pengaduan layanan dan *maintenance*.

BAB 4 REKAYASA KEBUTUHAN

Bab rekayasa kebutuhan membahas mengenai langkah-langkah yang dilakukan dalam mengumpulkan dan mendefinisikan kebutuhan-kebutuhan dari aplikasi *mobile* Pengaduan Layanan berbasis *Android* yang akan dibangun.

4.1 Elisitasi dan Analisis Kebutuhan

Pada tahap elisitasi dan analisis kebutuhan, dilakukan pra-perencanaan terlebih dahulu agar penulis mendapatkan pengetahuan-pengetahuan dasar dan dokumen-dokumen yang akan dijadikan pedoman dalam pengembangan aplikasi pengaduan layanan dan *maintenance.* Kemudian berdasarkan dokumen-dokumen yang didapatkan maka penulis dapat menganalisis proses bisnis *as-is* yang sudah ada dan permasalahan yang ditimbulkan dari proses bisnis tersebut. Dari permasalahan-permasalahan tersebut maka dibuatlah proses bisnis *to-be* dari aplikasi yang akan dibangun yang merupakan solusi dari permasalahan-permasalahan yang ada. Dari proses bisnis *to-be* maka dapat diidentifikasi aktor yang berperan dalam pengembangan aplikasi pengaduan layanan dan *maintenance.* Aktivitas-aktivitas yang dapat dilakukan oleh aktor pada aplikasi ini akan dideskripsikan pada spesifikasi kebutuhan.

4.1.1 Elisitasi Kebutuhan

Pada tahap elisitasi kebutuhan ini, penulis melakukan penggalian kebutuhan dengan teknik wawancara dan observasi dokumen.

4.1.1.1 Teknik Wawancara

Pada penelitian ini, teknik wawancara digunakan karena penulis dapat berinteraksi secara langsung kepada narasumber yang merupakan pengguna aplikasi yang akan dibangun. Dengan interaksi secara langsung, penulis dapat memahami permasalahan yang terjadi sebenarnya di lapangan, memahami kebutuhan dan keinginan pengguna, serta memungkinkan pengguna untuk menyalurkan aspirasinya terhadap aplikasi yang akan dibangun. Selain itu, penulis juga dapat mengetahui berbagai asumsi yang dimiliki pengguna sehingga dapat dijadikan referensi untuk aplikasi yang akan dibangun sehingga dapat memenuhi kebutuhan pengguna.

Wawancara dilakukan kepada beberapa narasumber. Narasumber pertama adalah Bagian Umum dan Perlengkapan yang diwakili oleh Ferix Panji Andrianto yang menjadi koordinator kerumahtanggaan. Narasumber kedua adalah Suprapto, S.T, M.T sebagai Wakil Dekan bidang Umum dan Keuangan yang menjadi garis komando paling atas di bidang Umum dan Keuangan FILKOM. Narasumber ketiga adalah Sri Suyati Luhur Utami , S.E sebagai Kepala Tata Usaha yang bertanggung jawab terhadap Sub Bagian Umum dan Keuangan dalam melaksanakan urusan kerumahtanggaan sesuai Peraturan Rektor Nomor 20 Tahun 2016. Kemudian narasumber yang keempat adalah pihak yang sering mengadukan

keluhan mengenai layanan dan membutuhkan *maintenance* yang diwakilkan oleh beberapa dosen di FILKOM dan beberapa mahasiswa FILKOM.

Berdasarkan hasil wawancara yang dilakukan kepada Ferix Panji Andrianto, S.ST selaku koordinator kerumahtanggaan di Bagian Umum dan Perlengkapan FILKOM, peneliti mendapatkan informasi bahwa selama ini Bagian Umum dan Perlengkapan mendapatkan pengaduan dari berbagai macam sumber di berbagai area FILKOM. Sumber-sumber pengaduan yaitu dari *Whatsapp* resmi Bagian Umum dan Perlengkapan*, E-Complaint,* SMS, telepon, *Whatsapp* personal Pegawai Umum dan Perlengkapan, dan juga mendatangi langsung ke kantor Umum dan Perlengkapan. Area yang harus ditangani pegawai Umum dan Perlengkapan juga cukup luas meliputi ruang-ruang di gedung, peralatan, kendaraan, taman, dan sebagainya. Keluhan-keluhan yang masuk pun bermacam-macam diantaranya AC dan mobil yang mengalami kerusakan, toilet yang rusak, kerusakan barang di ruang kantor dan kelas, dan sebagainya.

Masalah lain yang muncul selain pengaduan layanan adalah *maintenance.* Bagian Umum dan Perlengkapan selama ini sudah melakukan *maintenance* dengan pemeliharaan dan perawatan barang-barang di FILKOM secara berkala. Namun keterbatasan catatan perawatan menyebabkan adanya kesulitan dalam *monitoring* status suatu barang sehingga barang akan mudah mengalami kerusakan karena waktu *maintenance* yang kurang dilakukan secara berkala. Selain itu, pegawai Umum dan Perlengkapan harus sering memantau secara langsung untuk melakukan *monitoring.* Hal tersebut justru membuat Pegawai Umum dan Perlengkapan tidak sempat melakukan pengecekan keluhan-keluhan yang diadukan melalui *Whatsapp* resmi. Hal ini tentu merugikan pihak Umum dan Perlengkapan karena pihak pengadu akan salah paham dan mengeluhkan kinerja yang kurang tanggap. Selain itu, kerap terjadi kesalahpahaman dari pihak pelapor akibat sulitnya mendapatkan info jika suatu keluhan sudah ditindaklanjuti.

Berdasarkan hasil analisis wawancara yang dilakukan kepada narasumber-narasumber maka didapatkan beberapa kebutuhan yang didefinisikan pada Tabel 4.1 sampai Tabel 4.2.

Tabel 4.1 Hasil Wawancara Kebutuhan Umum dan Perlengkapan

	Kebutuhan
1	Menginginkan semua sivitas mengetahui bahwa Bagian Umum dan Perlengkapan adalah yang bertanggung jawab terhadap pemeliharaan dan perawatan sehingga semua pelaporan langsung masuk kepada Bagian ini
2	Mendapatkan pemberitahuan untuk melakukan *maintenance* secara berkala berdasarkan riwayat masing-masing barang
3	Bisa melihat riwayat *maintenance* suatu barang
4	Dapat memberitahukan jika pengaduan sudah diterima, sedang diproses, atau sudah selesai

Tabel 4.1 Hasil Wawancara Kebutuhan Umum dan Perlengkapan (Lanjutan)

	Kebutuhan
5	Menerima pengaduan dari 1 wadah saja sehingga pendataannya mudah dan teratur
6	Semua sivitas dapat melihat keluhan apa saja yang sudah diadukan sehingga dapat menghindari pengaduan yang *double*
7	Pengaduan keluhan disertai foto sehingga memudahkan untuk diidentifikasi kerusakannya
8	Dapat menambahkan data lewat *database* saja

Tabel 4.2 Hasil Wawancara Kebutuhan Pengadu

	Kebutuhan
1	Dapat mengadukan keluhan tanpa harus mendatangi kantor Bagian Umum dan Perlengkapan
2	Dapat melihat keluhan apa saja yang sudah diadukan serta status tindak lanjutnya
3	Dapat melihat status tindak lanjut dari pengaduannya misal diterima, diproses, atau sudah selesai
4	Dapat mengadukan keluhan berdasarkan kategorinya misal kendaraan, taman, barang, dan ruang.

4.1.1.2 Observasi Dokumen

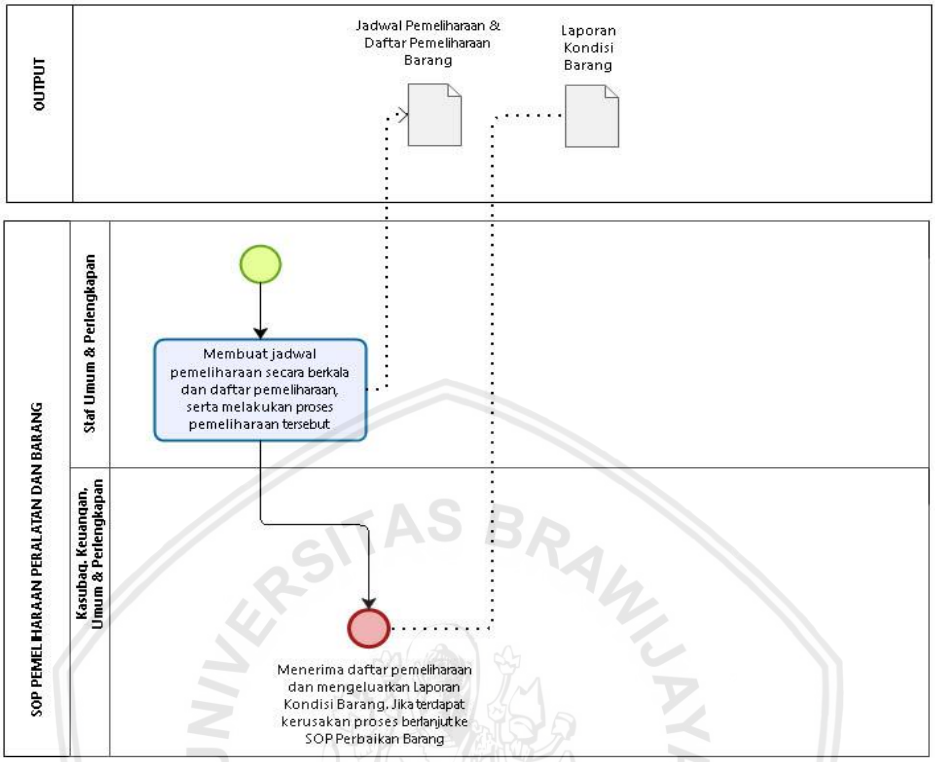
Pada tahap ini, penulis menggunakan beberapa dua dokumen pendukung yakni *Standar of Procedure* Bagian Umum dan Perlengkapan FILKOM serta Peraturan Rektor Nomor 20 Tahun 2016 Tentang Susunan Organisasi dan Tata Kerja. *Standar of Procedure* Bagian Umum dan Perlengkapan FILKOM digunakan untuk menganalisis proses bisnis As-Is yang sudah ada dan selama ini digunakan oleh Bagian Umum dan Perlengkapan sehingga dapat menjadi acuan untuk pemodelan proses bisnis To-Be agar dapat memenuhi kebutuhan pengguna. Peraturan Rektor Nomor 20 Tahun 2016 digunakan sebagai studi literatur mengenai pengaduan layanan dan *maintenance* di FILKOM.

4.1.2 Elisitasi Proses Bisnis *As-Is*

4.1.2.1 Proses Bisnis *As-Is* Pemeliharaan Peralatan dan Barang

Dalam pemeliharaan Peralatan dan Barang, aktor yang terlibat adalah staf Umum dan Perlengkapan yang membuat daftar dan jadwal pemeliharaan secara berkala dan melakukan pemeliharaan tersebut, serta aktor Kasubag Umum dan Perlengkapan yang menerima yang menerima daftar pemeliharaan dan mengeluarkan laporan kondisi barang, jika terdapat kerusakan maka berlanjut

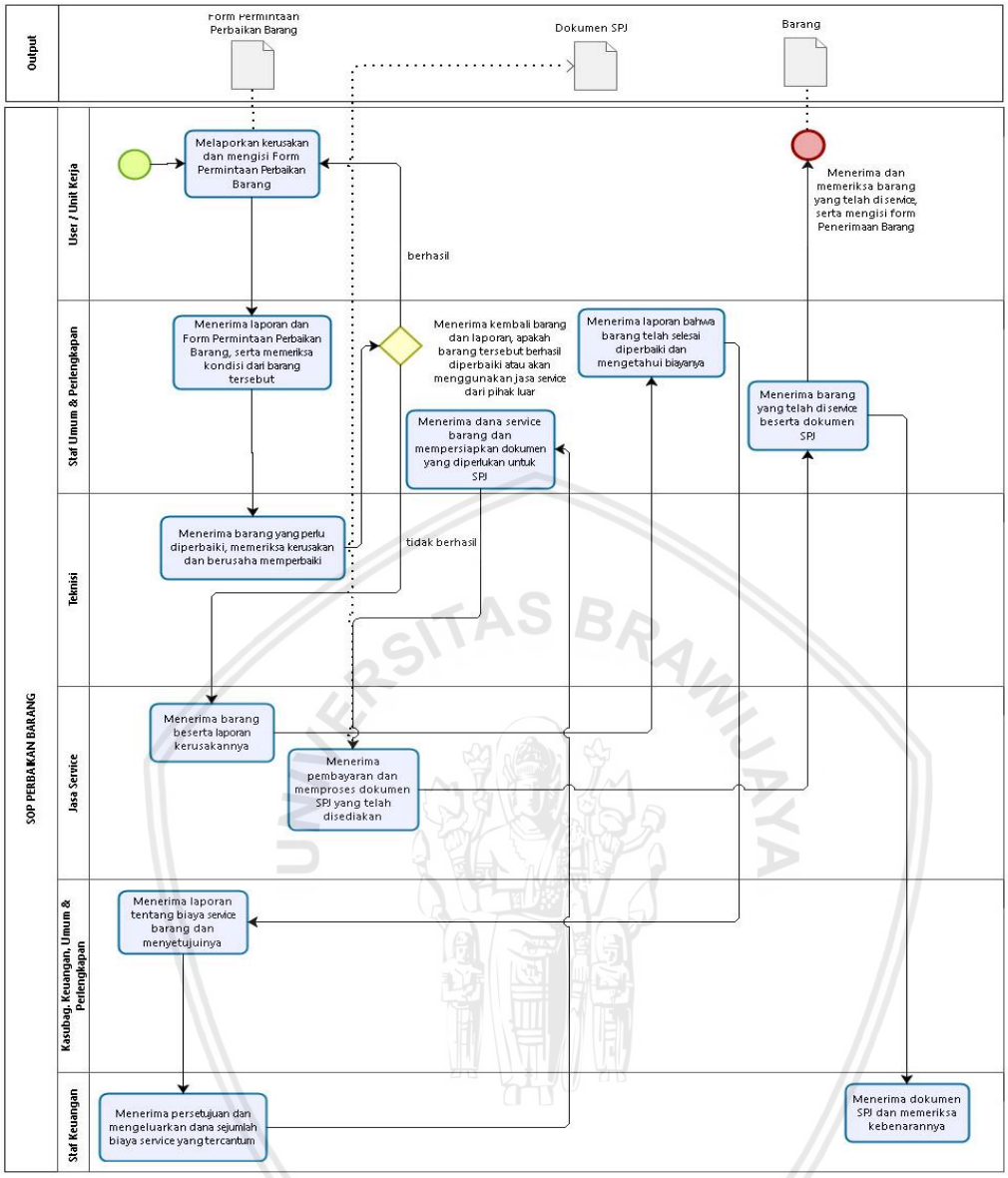
pada proses bisnis perbaikan barang. *Output* yang dihasilkan adalah daftar dan jadwal pemeliharaan barang serta laporan kondisi barang.



Gambar 4.1 Proses Bisnis *As-Is* Pemeliharaan Peralatan dan Barang

4.1.2.2 Proses Bisnis *As-Is* Perbaikan Barang

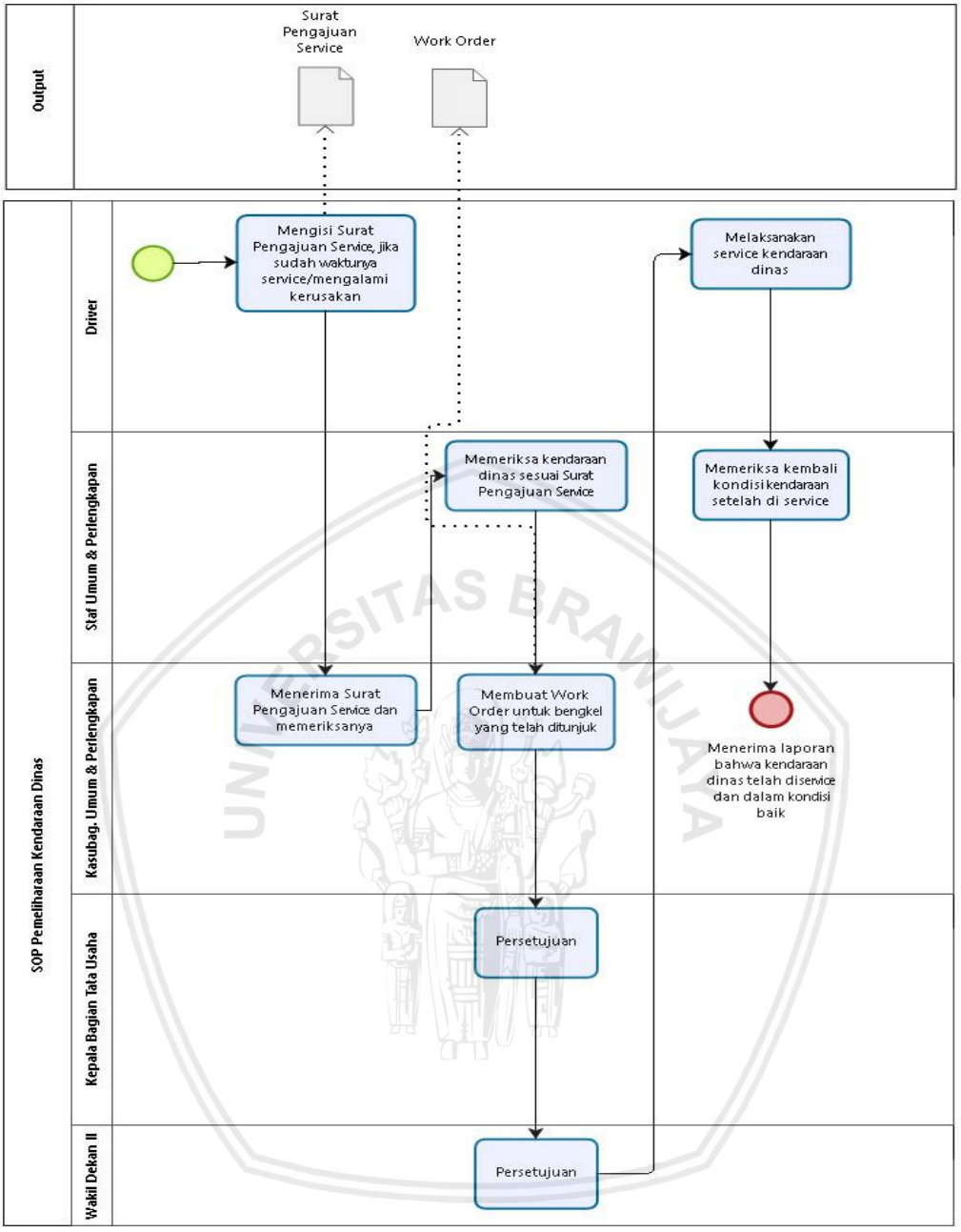
Dalam perbaikan barang, terdapat enam aktor. Aktor pertama adalah pengadu yang melaporkan kerusakan dan mengisi *form* perbaikan lalu menerima dan memeriksa barang setelah selesai diperbaiki serta mengisi *form* penerimaan barang. Aktor kedua adalah Staf Umum dan Perlengkapan yang menerima laporan dan *form* perbaikan barang lalu memeriksa kondisi barang, menerima dana dari keuangan untuk *service* barang dan menyiapkan SPJ atau surat pertanggungjawaban, menerima barang dari teknisi jika berhasil diperbaiki, dan menerima barang dari jasa *service* jika teknisi tidak berhasil memperbaikinya dan menerima SPJ dari jasa *service* tersebut. Aktor ketiga adalah teknisi yang menerima barang yang perlu diperbaiki serta memeriksaa kerusakan dan berusaha memperbaikinya. Aktor keempat adalah jasa *service* yang menerima barang serta laporan kerusakannya jika teknisi tidak berhasil dan menerima pembayaran serta proses dokumen SPJ yang disediakan. Kemudian aktor kelima adalah Kasubag Umum dan Perlengkapan yang menerima laporan biaya *service* dan menyetujuinya. Lalu dari Kasubag akan menerima persetujuan dan mengeluarkan dana *service.*



Gambar 4.2 Proses Bisnis *As-Is* Perbaikan Barang

4.1.2.3 Proses Bisnis *As-Is* Pemeliharaan Kendaraan Dinas

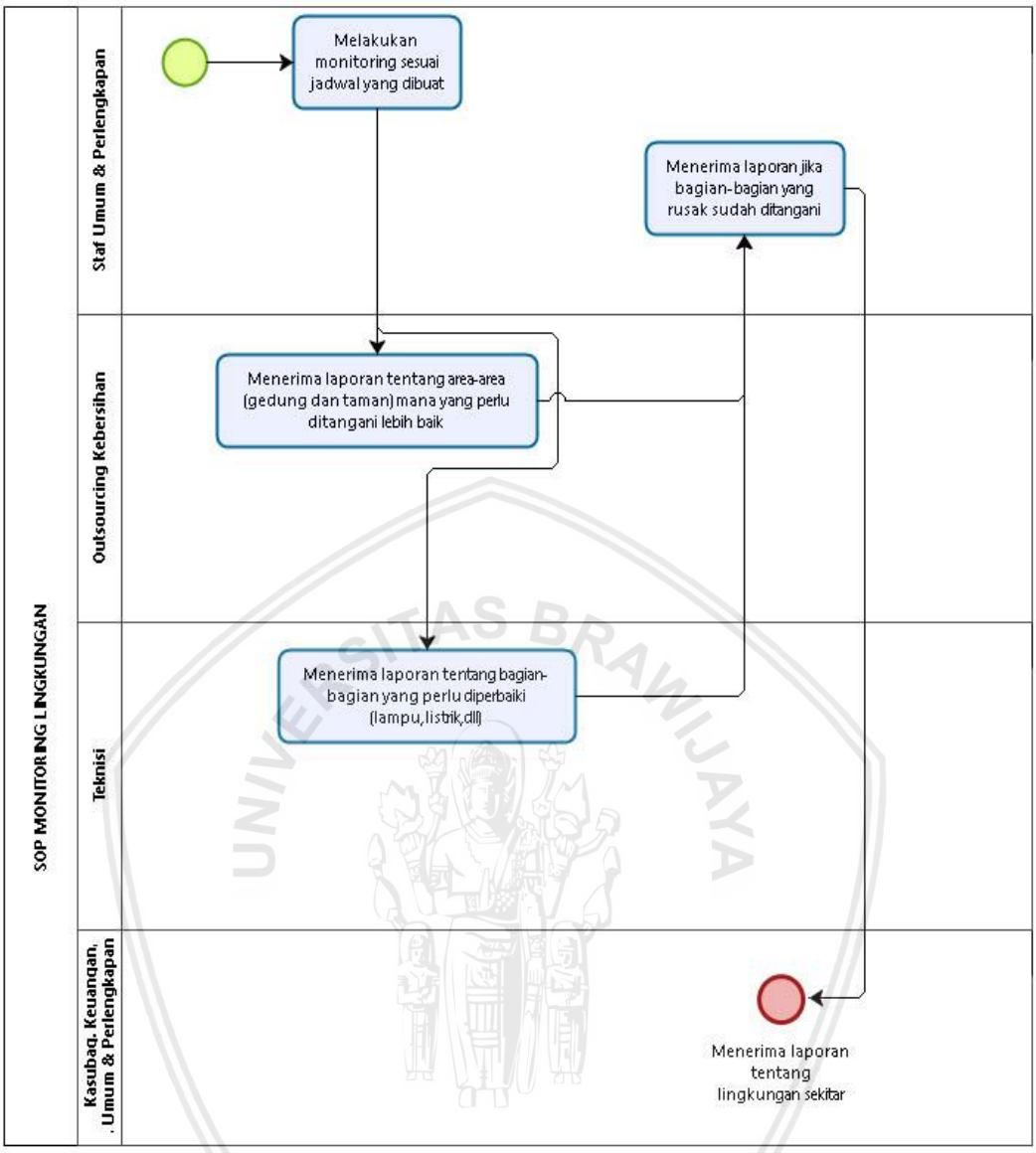
Pada pemeliharaan kendaraan dinas, ada 5 aktor yang terlibat. Aktor yang pertama adalah *driver* yang mengajukan surat *service* jika sudah waktunya *service* atau jika mengalami kerusakan. Kemudian Kasubag Umum dan Perlengkapan menerima surat pengajuan tersebut dan memeriksanya lalu menugaskan pegawai Umum dan perlengkapan untuk memeriksa kendaraan yang diajukan pada surat tersebut. Setelah itu maka dibuatlah *word order* untuk diberikan kepada bengkel yang telah ditunjuk lalu diajukan kepada Kasubag Tata Usaha dan Wakil Dekan II untuk mendapatkan persetujuan. Setelah mendapatkan persetujuan maka *driver* melaksanakan *service* ke bengkel. Setelah selesai, kemudian pegawai Umum dan Perlengkapan memeriksa kembali kondisi kendaraan. Jika sudah maka Kasubag akan mendapat laporan bahwa kendaraan selesai *service* dan dalam kondisi baik.



Gambar 4.3 Proses Bisnis *As-Is* Pemeliharaan Kendaraan Dinas

4.1.2.4 Proses Bisnis *As-Is* Monitoring Lingkungan

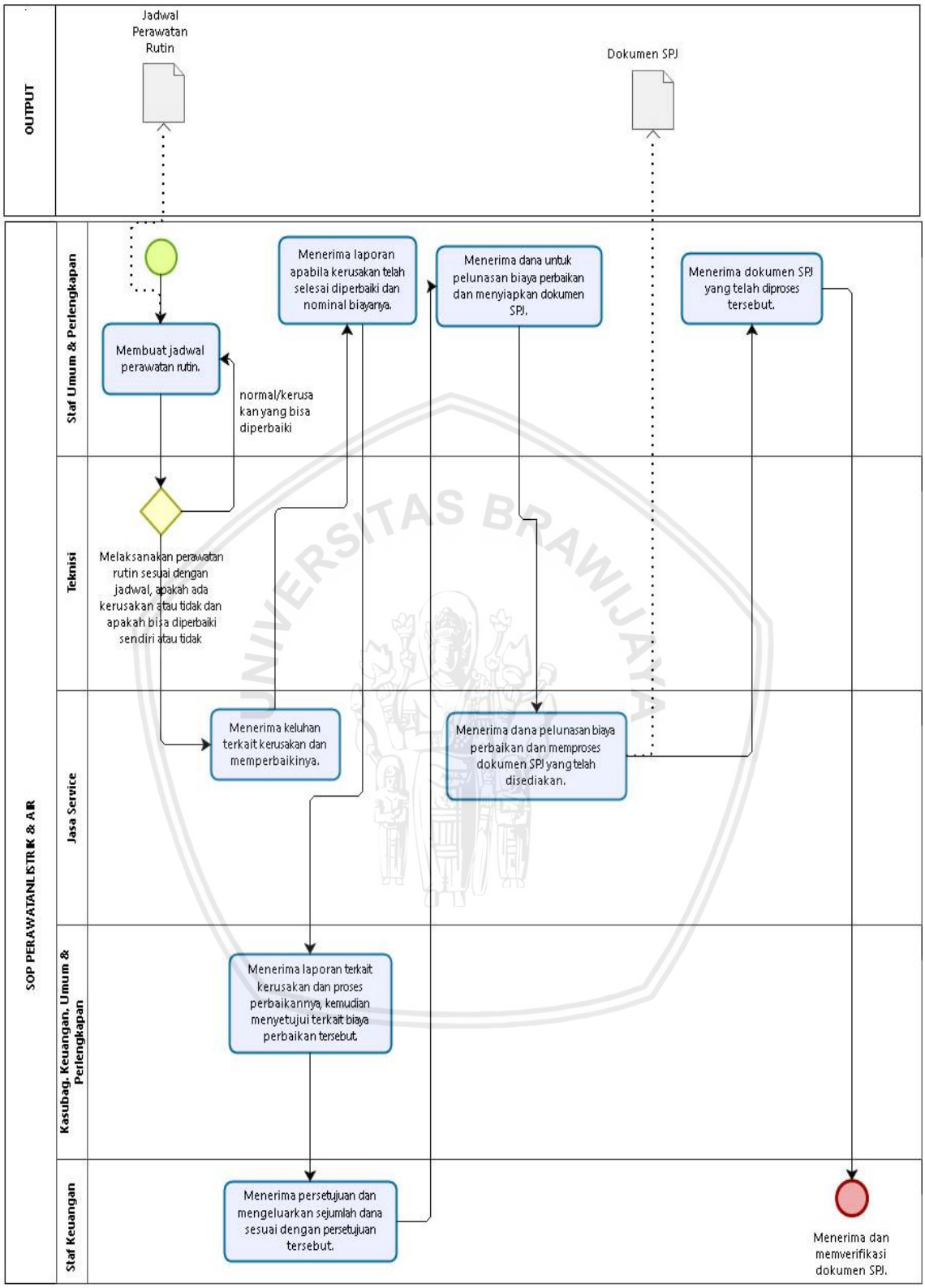
Pada proses ini, pegawai Umum dan Perlengkapan melakukan *monitoring* sesuai jadwal yang telah dibuat. Setelah *monitoring,* maka staf kebersihan dan teknisi menerima laporan-laporan mengenai area yang perlu ditangani dan diperbaiki. Setelah itu, staf kebersihan dan teknisi melaporkan kepada pegawai Umum dan Perlengkapan. Kemudian Kasubag Keuangan, Umum, dan Perlengkapan mendapatkan laporan mengenai kondisi lingkungan di FILKOM.



Gambar 4.4 Proses Bisnis *As-Is Monitoring* Lingkungan

4.1.2.5 Proses Bisnis *As-Is* Perawatan Listrik dan Air

Pada proses ini, pegawai Umum dan Perlengkapan membuat jadwal perawatan rutin yang menghasilkan *output* berupa jadwal perawatan rutin. Kemudian teknisi melaksanakan perawatan sesuai jadwal. Jika ada kerusakan maka teknisi akan memperbaikinya, namun jika tidak bisa memperbaiki sendiri maka akan dilaporkan kepada jasa *service.* Jika jasa *service* sudah memperbaikinya maka pegawai umum dan perlengkapan akan mendapatkan laporan. Dan laporan tersebut akan diteruskan kepada Kasubag Keuangan, Umum, dan Perlengkapan kemudian jika disetujui maka diteruskan kepada Kepala Tata Usaha. Kemudian pegawai Umum dan Perlengkapan akan menerima dana yang akan diserahkan pada jasa *service.* Kemudian pegawai Umum dan Perlengkapan menerima SPJ dari jasa *service* lalu KTU akan menerima dan memverifikasi SPJ tersebut.



Gambar 4.5 Proses Bisnis *As-Is* Perawatan Listrik dan Air

4.1.3 Pemodelan Proses Bisnis *To-Be*

Berdasarkan analisis proses bisnis *as-is* pada tahap sebelumnya maka timbul beberapa masalah yang dapat dilihat pada Tabel 4.3 berikut.

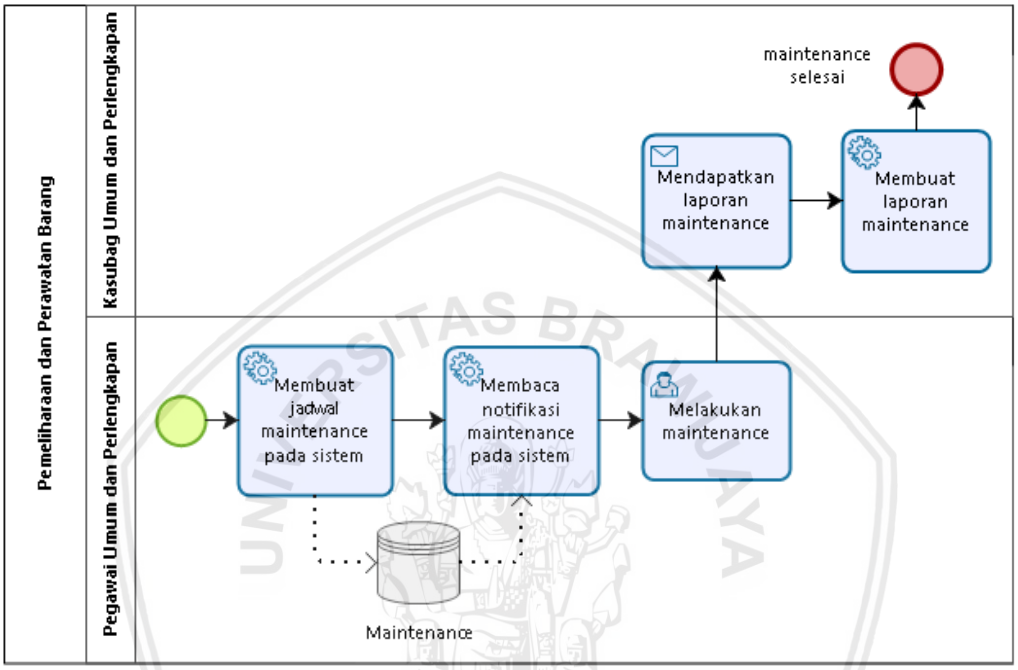
Tabel 4.3 Analisis Masalah Proses Bisnis *As-Is*

Masalah	Dampak
Tidak adanya pengingat untuk melakukan pemeliharaan secara berkala	Ada barang yang mengalami kerusakan akibat kurangnya pemeliharaan secara berkala
Masih belum ada *database* yang menyimpan daftar inventaris dan riwayat pemeliharaannya	Sulit untuk mendeteksi waktu pemeliharaan terakhir suatu inventaris sehingga tidak dapat diketahui kondisinya secara pasti
Pengaduan keluhan layanan masih melalui sumber yang bermacam-macam	Banyak pengaduan-pengaduan yang berjumlah *double* yang mengakibatkan kesulitan dalam menuliskan riwayat keluhan tersebut pada *database*
Tidak adanya transparansi mengenai status pengaduan sehingga pihak pelapor tidak mengetahui keluhan tersebut sudah dilaksanakan atau tidak	Pihak pengadu banyak yang mengadukan dan menanyakan status dari pengaduannya langsung ke kantor dan mengeluhkan kinerja pegawai Umum dan Perlengkapan jika tidak ada kejelasan status
Setelah mendapatkan suatu pengaduan, pegawai Umum dan Perlengkapan masih harus mendatangi tempat atau barang yang diadukan tersebut	Tidak efektif karena memakan waktu yang cukup banyak untuk mendatangi tempat atau barang yang diadukan sehingga pekerjaan lain jadi terbengkalai
Pengajuan *service* untuk kendaraan dinas masih belum memiliki *database* dan memakan waktu lama karena menunggu persetujuan Kasubag	Tidak efektif karena memakan waktu yang cukup banyak dan sulit untuk mendeteksi waktu *service* terakhir sehingga tidak dapat diketahui kondisi yang menyebabkan kerusakan kendaraan secara pasti

Berdasarkan masalah-masalah tersebut, penulis membuat pemodelan proses bisnis *to-be* dari aplikasi yang akan dibangun. Pemodelan proses bisnis *to-be* ini merupakan penggambaran berupa model proses bisnis yang merupakan solusi dari permasalahan pada Tabel 4.4 diatas. Berdasarkan hasil analisis terhadap permasalahan-permasalahan tersebut, maka solusi yang diajukan dapat dimodelkan dengan proses bisnis *to-be* pada Gambar 4.6 dan Gambar 4.7 yang sudah mencakup perawatan, pemeliharaan, dan *monitoring* listrik dan air, kendaraan dinas, lingkungan, dan sebagainya.

4.1.3.1 Proses Bisnis *To-Be* Pemeliharaan Peralatan dan Barang

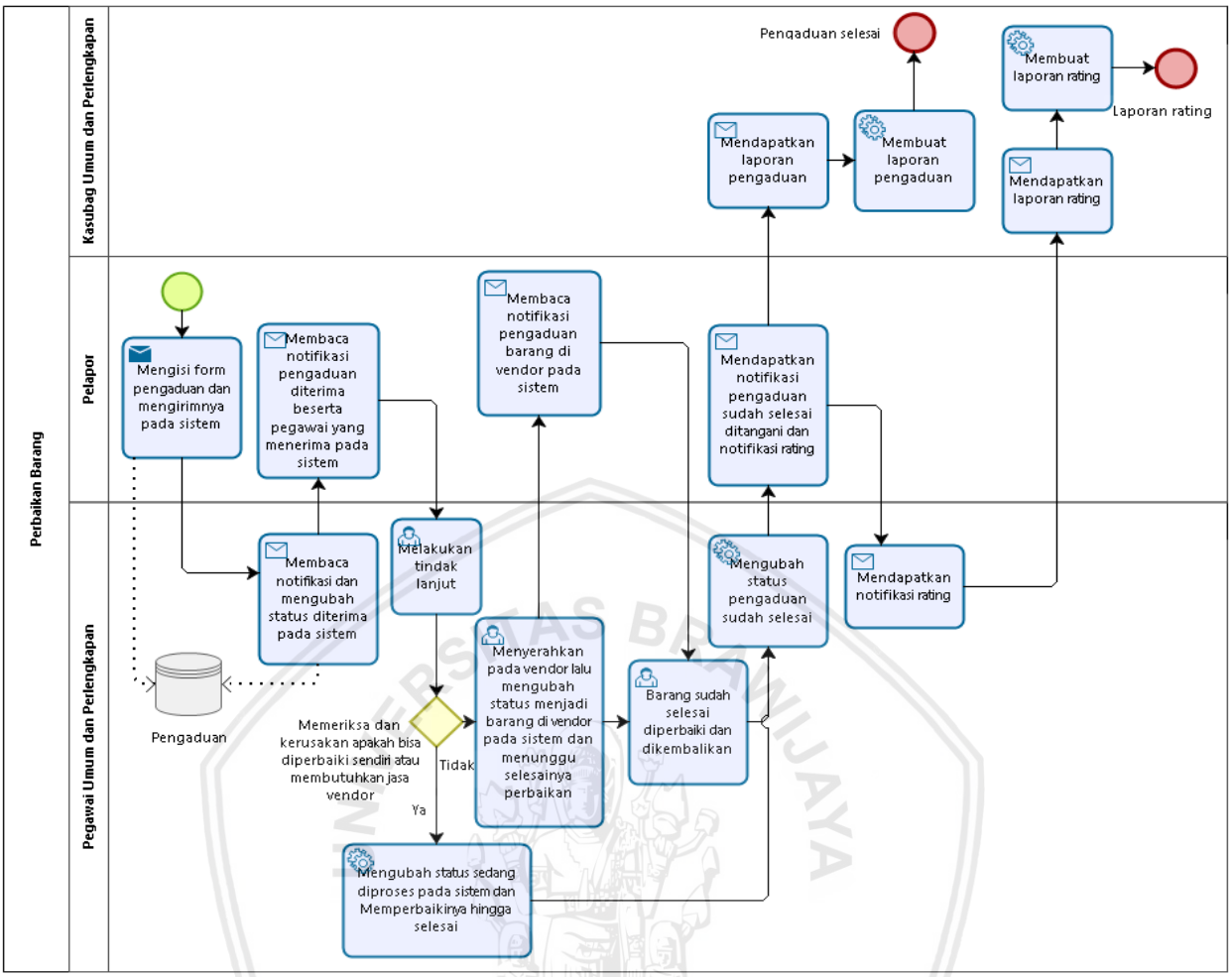
Pada proses bisnis *to-be* pemeliharaan barang, pegawai juga membuat jadwal pada *form* jadwal *maintenance.* Setelah jadwal berhasil ditambahkan, maka sistem akan memberi notifikasi otomatis untuk mengingatkan adanya jadwal *maintenance.* Kasubag umum dan perlengkapan akan menerima laporan *maintenance* yang sudah dijadwalkan dan membuat laporan *maintenance.* Proses bisnis ini dapat digambarkan seperti Gambar 4.6 berikut.



Gambar 4.6 Proses Bisnis To-Be Pemeliharaan Peralatan dan Barang

4.1.3.2 Proses Bisnis *To-Be* Perbaikan Barang

Pada proses bisnis *to-be* perbaikan barang, diawali ketika pelapor mengadukan keluhannya, kemudian pengaduan tersebut tersimpan dalam sistem dan dapat diterima oleh pegawai umum dan perlengkapan. Ketika pengaduan diterima maka sistem akan memberikan notifikasi jika pengaduan diterima beserta pegawai yang menangani pengaduan tersebut. Setelah pengaduan diterima maka pegawai umum dan perlengkapan melakukan tindak lanjut dan jika setelah diperiksa suatu kerusakan bisa diperbaiki sendiri maka akan diperbaiki sampai selesai dan pegawai umum dan perlengkapan akan mengubah status pengaduan sudah selesai sehingga sistem akan memberikan notifikasi pengaduan sudah selesai dan pelapor dapat memberikan *rating*. Kemudian jika kerusakan harus dibawa ke vendor maka barang akan diserahkan pada vendor lalu status akan diubah menjadi barang di vendor dan jika sudah selesai diperbaiki maka status juga akan selesai. kasubag umum dan perlengkapan dapat menerima laporan pengaduan dan membuat laporan pengaduan. Setelah pengadu memberikan *rating* maka pegawai umum dan perlengkapan dapat menerima notifikasi rating kemudian Kasubag akan mendapat laporan *rating* dan membuat laporan *rating.* Proses bisnis ini dapat digambarkan seperti Gambar 4.7 berikut.



Gambar 4.7 Proses Bisnis *To-Be* Perbaikan Barang

4.1.4 Identifikasi Aktor

Pada tahap ini, penulis melakukan identifikasi aktor yang dapat menggunakan aplikasi Pengaduan Layanan dan *Maintenance.* Bagian ini berisi *task definition* pengguna aplikasi yang akan dikembangkan yang dapat dilihat pada Tabel 4.4.

Tabel 4.4 Identifikasi Aktor

Aktor	Deskripsi
Kasubag Umum dan Perlengkapan	Aktor dari Bagian Umum dan Perlengkapan FILKOM yang dapat menerima serta membuat laporan *maintenance* dan laporan perbaikan barang
Pegawai Umum dan Perlengkapan	Aktor dari Bagian Umum dan Perlengkapan FILKOM yang dapat memasukkan penjadwalan *maintenance* suatu inventaris secara berkala, memberi status terhadap pengaduan, menerima notifikasi pengaduan, dan lainnya
Pengadu	Aktor dari civitas FILKOM yang dapat mengadukan keluhan-keluhan mengenai layanan di FILKOM meliputi barang-barang di FILKOM, kendaraan dinas, bangunan-bangunan, barang, dan tanaman.

4.1.5 Spesifikasi Kebutuhan

Spesifikasi kebutuhan digunakan untuk menjabarkan kebutuhan- kebutuhan user yang dikelompokkan menjadi 2 macam, kebutuhan fungsional dan kebutuhan non-fungsional. Sebagai bagian dari tahap manajemen kebutuhan yang bertujuan untuk memudahkan identifikasi kebutuhan maka penulis menetapkan aturan penomoran.

PLM_F/NF_000

Gambar 4.8 Aturan Penomoran Kebutuhan

Keterangan:

PLM : singkatan dari aplikasi yaitu Pengaduan Layanan dan *Maintenance*
F/NF : F merupakan kode untuk kebutuhan fungsional dan NF merupakan kode untuk kebutuhan non fungsional
000 : No. urut kebutuhan yang dimulai dari 100

4.1.5.1 Spesifikasi Kebutuhan Fungsional

Terdapat 18 kebutuhan fungsional pengguna yang dapat dilihat pada Tabel 4.5.

Tabel 4.5 Kebutuhan Fungsional Pengguna

Kode Fungsi	Kebutuhan Fungsional	Aktor	Spesifikasi Kebutuhan	Kode Spesifikasi
PLM_F_100	*Login*	Kasubag dan Pegawai Umum dan Perlengkapan, Pelapor	1) Sistem menyediakan tombol pilihan *login* yaitu sebagai pegawai, pengadu, dan kasubag	PLM_F_101
			2) Sistem harus menyediakan kolom NIP/NIM dan *password*	PLM_F_102
			3) Kolom NIP/NIM dan *password* tidak boleh kosong	PLM_F_103
PLM_F_200	*Logout*	Kasubag dan Pegawai Umum dan Perlengkapan, Pelapor	1) Sistem harus menyediakan *button logout* untuk keluar dari sistem	PLM_F_201
PLM_F_300	Melihat daftar pengaduan	Kasubag, Pegawai Umum Perlengkapan, Pelapor	1) Sistem menampilkan daftar pengaduan ketika aktor menekan menu daftar pengaduan di beranda	PLM_F_301

Tabel 4.5 Kebutuhan Fungsional Pengguna (Lanjutan)

Kode Fungsi	Kebutuhan Fungsional	Aktor	Spesifikasi Kebutuhan	Kode Spesifikasi
			2) Menu daftar pengaduan berisi seluruh daftar pengaduan yang telah diajukan meliputi judul keluhan, waktu, pelapor, serta status pengaduan	PLM_F_302
			3) Sistem menyediakan tombol *filter* untuk menampilkan pengaduan berdasarkan statusnya	PLM_F_303
PLM_F_400	Melihat detail pengaduan	Kasubag dan Pegawai Umum dan Perlengkapan, Pelapor	1) Sistem menampilkan halaman detail pengaduan ketika aktor menekan pengaduan yang dipilih	PLM_F_401
			2) Sistem menampilkan rincian data pengaduan yang telah diajukan meliputi nama, waktu, status, foto, deskripsi, dan pegawai yang menindaklanjuti pengaduan tersebut	PLM_F_402
PLM_F_500	Melihat statistik pengaduan	Kasubag dan Pegawai Umum dan Perlengkapan, Pelapor	1) Sistem menampilkan statistik pengaduan ketika aktor menekaan menu statistik pengaduan di beranda	PLM_F_501
			2) Sistem menampilkan statistik jumlah pengaduan terbanyak dengan bentuk diagram batang dan keterangannya	PLM_F_502
PLM_F_600	Membuat laporan pengaduan	Kasubag Umum dan Perlengkapan	1) Aktor dapat memilih jenis laporan pada halaman beranda yang terdiri dari laporan pengaduan, laporan *maintenance,* dan laporan *rating*	PLM_F_601

Tabel 4.5 Kebutuhan Fungsional Pengguna (Lanjutan)

Kode Fungsi	Kebutuhan Fungsional	Aktor	Spesifikasi Kebutuhan	Kode Spesifikasi
			2) Ketika aktor memilih laporan pengaduan maka sistem akan menampilkan tabel pengaduan yang berisi kolom nama pengaduan, status pengaduan, tanggal dibuat dan selesainya pengaduan serta diurutkan berdasar kolom status	PLM_F_602
			3) Ketika aktor memilih laporan *maintenance* maka sistem akan menampilkan tabel *maintenance* yang berisi kolom barang, jadwal *maintenance*, serta waktu pelaksanaan *maintenance*	PLM_F_603
			4) Ketika aktor memilih laporan *rating* maka sistem menampilkan tabel *rating* yang berisi kolom nomor, nama pegawai, dan *rating* pegawai tersebut	PLM_F_604
			5) Sistem menampilkan tombol unduh laporan	PLM_F_605
			6) Pada halaman laporan pegaduan dan laporan *maintenance* sistem menyediakan tombol *filter* untuk menampilkan laporan pada waktu tertentu	PLM_F_606
PLM_F_700	Membaca notifikasi pengaduan	Pegawai Umum dan Perlengkapan	1) Sistem menyediakan notifikasi yang diterima oleh aktor ketika pelapor berhasil mengirimkan keluhan	PLM_F_701

Tabel 4.5 Kebutuhan Fungsional Pengguna (Lanjutan)

Kode Fungsi	Kebutuhan Fungsional	Aktor	Spesifikasi Kebutuhan	Kode Spesifikasi
			2) Jika notifikasi ditekan akan menampilkan halaman detail pengaduan	PLM_F_702
			3) Sistem menyediakan tombol "terima"	PLM_F_703
PLM_F_800	Melihat pengaduan yang ditangani	Pegawai Umum dan Perlengkapan	1) Sistem menampilkan semua daftar pengaduan yang ditangani oleh aktor ketika aktor menekan tombol "tugas saya" di beranda	PLM_F_801
			2) Pada halaman tugas saya sistem menyediakan *filter* status untuk menampilkan pengaduan yaitu diterima, sedang diproses, sedang di vendor, dan sudah selesai	PLM_F_802
PLM_F_900	Mengubah Status Pengaduan	Pegawai Umum dan Perlengkapan	1) Sistem menyediakan tombol "terima" ketika notifikasi pengaduan diklik, kemudian status berubah menjadi "diterima"	PLM_F_901
			2) Sistem menyediakan tombol "perbaiki sendiri" dan "serahkan ke vendor" ketika tombol "terima" sudah diklik, kemudian status akan berubah sesuai pilihan yaitu "sedang di vendor" atau "sedang diproses"	PLM_F_902
			3) Sistem menyediakan tombol "selesai" ketika tombol "perbaiki sendiri" atau "serahkan ke vendor" sudah diklik, kemudian status berubah "selesai"	PLM_F_903

Tabel 4.5 Kebutuhan Fungsional Pengguna (Lanjutan)

Kode Fungsi	Kebutuhan Fungsional	Aktor	Spesifikasi Kebutuhan	Kode Spesifikasi
PLM_F_1000	Melihat jadwal *maintenance*	Pegawai Umum dan Perlengkapan	1) Sistem menampilkan semua *maintenance* yang harus dilaksanakan oleh aktor ketika menekan tombol "jadwal *maintenance"* di halaman beranda	PLM_F_1001
			2) Pada halaman jadwal *maintenance,* akan menampilkan *maintenance* yang telah dimasukkan sesuai tanggalnya	PLM_F_1002
			3) Sistem dapat menampilkan daftar *maintenance* pada tanggal yang ditekan aktor	PLM_F_1003
PLM_F_1100	Mengisi *form* jadwal *maintenance*	Pegawai Umum dan Perlengkapan	1) Sistem akan menampilkan *form* jadwal *maintenance* ketika aktor menekan tombol "buat jadwal" pada halaman beranda	PLM_F_1101
			2) Sistem harus menyediakan kolom kategori inventaris, nomor inventaris, dan skala waktu *maintenance* serta tombol "submit"	PLM_F_1102
			3) Kolom kategori inventaris, nomor inventaris, dan skala waktu *maintenance* tidak boleh kosong	PLM_F_1103
PLM_F_1200	Membaca notifikasi *maintenance*	Pegawai Umum dan Perlengkapan	1) Sistem menyediakan notifikasi yang diterima oleh aktor ketika hari-h jadwal *maintenance*	PLM_F_1201
			2) Sistem menyediakan tombol "kerjakan" maka status *maintenance* berubah menjadi selesai	PLM_F_1202

Tabel 4.5 Kebutuhan Fungsional Pengguna (Lanjutan)

Kode Fungsi	Kebutuhan Fungsional	Aktor	Spesifikasi Kebutuhan	Kode Spesifikasi
PLM_F_1300	Membaca notifikasi *rating*	Pegawai Umum dan Perlengkapan	1) Sistem menyediakan notifikasi jumlah *rating* yang diterima oleh aktor	PLM_F_1301
PLM_F_1400	Membuat pengaduan	Pelapor	1) Aktor dapat memilih kategori pengaduan pada halaman beranda yang terdiri dari kategori kendaraan dinas, kebersihan, peralatan, dan ruang/bangunan	PLM_F_1401
			2) Sistem menampilkan halaman *form* pengaduan ketika aktor memilih kategori pengaduan yang akan diajukan	PLM_F_1402
			3) Sistem harus menyediakan kolom nama, foto, dan deskripsi pengaduan yang akan diajukan serta tombol *submit*	PLM_F_1403
			4) Kolom nama, foto, dan deskripsi pengaduan tidak boleh kosong	PLM_F_1404
PLM_F_1500	Membaca notifikasi status pengaduan	Pelapor	1) Sistem menyediakan notifikasi yang diterima aktor ketika status pengaduan yang diajukannya berubah	PLM_F_1501
			2) Sistem menyediakan notifikasi pegawai yang menangani pengaduan kepada aktor	PLM_F_1502
PLM_F_1600	Memberi *rating*	Pelapor	1) Sistem menampilkan notifikasi untuk memberi *rating* kepada pegawai yang menangani pengaduan ketika keluhan sudah selesai ditangani	PLM_F_1601

Tabel 4.5 Kebutuhan Fungsional Pengguna (Lanjutan)

Kode Fungsi	Kebutuhan Fungsional	Aktor	Spesifikasi Kebutuhan	Kode Spesifikasi
			2) Sistem menampilkan tombol *rating* berupa 5 bintang dan tombol "kirim"	PLM_F_1602
PLM_F_1700	Melihat riwayat kerusakan	Kasubag dan Pegawai Umum dan Perlengkapan, Pelapor	1) Sistem menampilkan halaman riwayat kerusakan ketika aktor menekan nomor inventaris di halaman detail pengaduan	PLM_F_1701
			2) Sistem menampilkan daftar riwayat kerusakan berdasarkan nomor pengaduannya	PLM_F_1702
PLM_F_1800	Melihat riwayat *maintenance*	Kasubag dan Pegawai Umum dan Perlengkapan, Pelapor	1) Sistem menampilkan halaman riwayat *maintenance* ketika aktor menekan nomor inventaris di halaman tanggal jadwal *maintenance*	PLM_F_1801
			2) Sistem menampilkan daftar riwayat *maintenance* berdasarkan nomor *maintenance*	PLM_F_1802

4.1.5.2 Definisi Kebutuhan Non Fungsional

Kebutuhan non-fungsional merupakan kebutuhan yang mencakup properti perilaku yang dimiliki sistem untuk meningkatkan kualitas sistem. Kebutuhan non-fungsional dapat dilihat di Tabel 4.6 berikut.

Tabel 4.6 Definisi Kebutuhan Non Fungsional

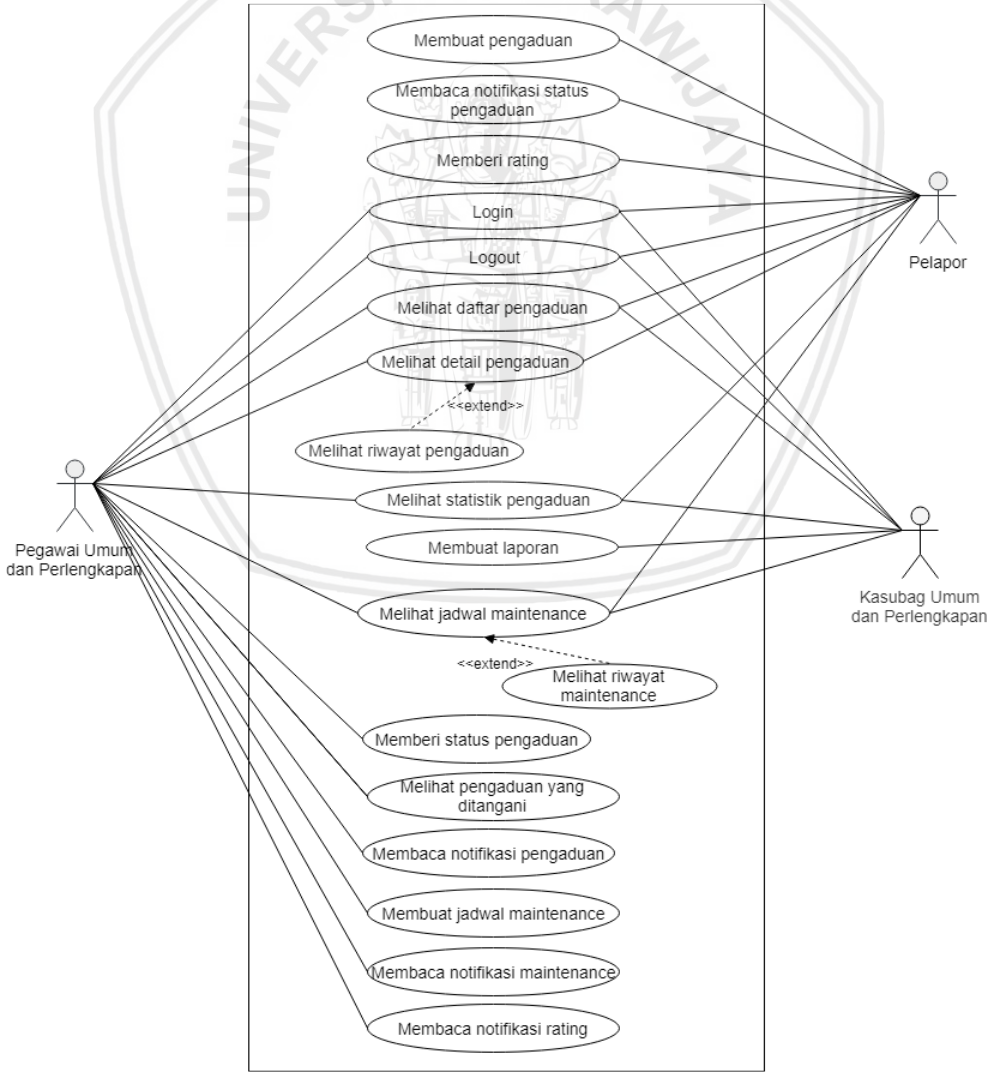
Kode	Parameter	Deskripsi Kebutuhan
PLM_NF_01	*Usability*	Sistem dapat digunakan oleh pengguna dengan mudah
PLM_NF_02	*Compatibility*	Sistem dapat berjalan pada berbagai *device* dengan resolusi layar dan sistem operasi yang berbeda.

4.2 Pemodelan Kebutuhan

Pemodelan kebutuhan merupakan tahap untuk memodelkan analisis kebutuhan ke dalam bentuk diagram yang bertujuan untuk memberikan gambaran umum fitur-fitur yang terdapat dalam suatu sistem. Pada penelitian ini, pemodelan yang digunakan berorientasi objek (*Object Oriented Design).* pemodelan ini digunakan karena pada aplikasi yang akan dikembangkan memiliki objek-objek dengan karakteristik tersendiri. Pemodelan ini direpresentasikan dalam bentuk UML (*Unified Model Language)* yang pada penelitian ini terdiri dari *Use case diagram, Class diagram,* dan *Sequence diagram.*

4.2.1 Diagram *Use Case*

Diagram *use case* menyatakan visualisasi interaksi yang terjadi antara aktor dengan sistem. Diagram ini bisa menjadi gambaran yang bagus untuk menjelaskan konteks dari sebuah sistem sehingga terlihat jelas batasan dari sistem (Larman, 2005). Diagram *use case* dari aplikasi pengaduan layanan dan *maintenance* dapat dilihat pada Gambar 4.9 berikut.



Gambar 4.9 Diagram *Use Case* Aplikasi Pengaduan Layanan dan *Maintenance*

4.2.2 Skenario *Use Case*

Skenario *Use Case* mendeskripsikan langkah yang dilakukan aktor ketika berinteraksi dengan sistem, baik yang berhasil maupun gagal (Kurniawan, 2018). Berdasarkan *use case diagram* pada Gambar 4.9 maka dibuatlah *use case scenario* pada Tabel 4.7 sampai dengan Tabel 4.24.

Tabel 4.7 Skenario *Use Case* Login

***Login* (PLM_F_100)**	
Actor	Kasubag dan Pegawai Umum dan Perlengkapan, Pengadu
Objective	Masuk ke dalam aplikasi
Pre-condition	Aktor belum melakukan *login*
Main Flow	1. Aktor memilih *login* sebagai pengadu, pegawai, atau kasubag 2. Sistem menampilkan halaman *login* 3. Aktor mengisi NIM/NIP dan *password* 4. Aktor menekan tombol "login"
Alternative Flows	1. Jika NIM/NIP atau *password* salah maka akan muncul pesan "data yang dimasukkan salah" 2. Jika NIM/NIP atau *password* masih kosong maka akan muncul pesan "NIM/NIP dan *password* harus diisi"
Post-Condition	Aktor berhasil masuk ke dalam aplikasi

Tabel 4.8 Skenario *Use Case Logout*

***Logout* (PLM_F_200)**	
Actor	Kasubag dan Pegawai Umum dan Perlengkapan, Pengadu
Objective	Keluar dari aplikasi
Pre-condition	Aktor telah membuka aplikasi
Main Flow	1. Aktor menekan tombol "logout" 2. Sistem menampilkan halaman *login*
Post-Condition	Aktor berhasil keluar dari aplikasi

Tabel 4.9 Skenario *Use Case* Melihat Daftar Pengaduan

Melihat daftar pengaduan (PLM_F_300)	
Actor	Kasubag dan Pegawai Umum dan Perlengkapan, Pengadu
Objective	Melihat semua pengaduan yang telah diajukan
Pre-condition	Aktor telah berada di halaman beranda
Main Flow	1. Aktor menekan menu daftar pengaduan 2. Sistem menampilkan halaman daftar pengaduan
Post-Condition	Aktor berhasil melihat daftar pengaduan

Tabel 4.10 Skenario *Use Case* Melihat Detail Pengaduan

Melihat detail pengaduan (PLM_F_400)	
Actor	Kasubag dan Pegawai Umum dan Perlengkapan, Pengadu
Objective	Melihat rincian data pengaduan yang telah diajukan
Pre-condition	Aktor telah berada di halaman daftar pengaduan
Main Flow	1. Aktor menekan pengaduan yang dipilih 2. Sistem menampilkan halaman detail pengaduan
Post-Condition	Aktor berhasil melihat detail pengaduan

Tabel 4.11 Skenario *Use Case* Melihat Statistik Pengaduan

Melihat statistik pengaduan (PLM_F_500)	
Actor	Kasubag dan Pegawai Umum dan Perlengkapan, Pengadu
Objective	Melihat statistik jumlah pengaduan yang diterima
Pre-condition	Aktor telah berada di halaman beranda
Main Flow	1. Aktor menekan menu statistik pengaduan 2. Sistem menampilkan halaman statistik pengaduan
Post-Condition	Aktor berhasil melihat statistik pengaduan

Tabel 4.12 Skenario *Use Case* Membuat Laporan

Membuat laporan (PLM_F_600)	
Actor	Kasubag Umum dan Perlengkapan
Objective	Melihat dan mengunduh laporan
Pre-condition	Aktor telah berada di halaman beranda
Main Flow	1. Aktor menekan salah satu menu laporan 2. Sistem menampilkan tabel laporan yang dipilih 3. Aktor menekan tombol “unduh laporan” 4. Sistem mengunduh laporan
Post-Condition	Aktor berhasil mengunduh laporan

Tabel 4.13 Skenario *Use Case* Membaca Notifikasi Pengaduan

Membaca notifikasi pengaduan (PLM_F_700)	
Actor	Pegawai Umum dan Perlengkapan
Objective	Membaca notifikasi pengaduan yang dikirim oleh pengadu
Pre-condition	Sistem mengirimkan notifikasi pengaduan
Main Flow	1. Aktor menekan notifikasi pengaduan 2. Sistem menampilkan halaman detail pengaduan

Tabel 4.13 Skenario *Use Case* Membaca Notifikasi Pengaduan (Lanjutan)

Membaca notifikasi pengaduan (PLM_F_700)	
Alternative Flows	1. Jika aktor melakukan *swipe* pada notifikasi yang diterima maka aktor akan tetap berada *current screen*
Post-Condition	Aktor berhasil membaca pengaduan

Tabel 4.14 Skenario *Use Case* Melihat Pengaduan yang Ditangani

Melihat pengaduan yang ditangani (PLM_F_800)	
Actor	Pegawai Umum dan Perlengkapan
Objective	Melihat semua pengaduan yang ditangani
Pre-condition	Aktor telah berada di halaman beranda
Main Flow	1. Aktor menekan tombol "tugas saya" 2. Sistem menampilkan halaman tugas saya
Post-Condition	Aktor berhasil melihat semua pengaduan yang ditanganinya

Tabel 4.15 Skenario *Use Case* Mengubah Status Pengaduan

Mengubah status pengaduan (PLM_F_900)	
Actor	Pegawai Umum dan Perlengkapan
Objective	Memberi status terhadap pengaduan yang telah diajukan
Pre-condition	Aktor berada pada halaman detail pengaduan
Main Flow	1. Aktor menekan tombol "terima" 2. Sistem mengubah status pengaduan "diterima" 3. Aktor menekan tombol "perbaiki sendiri" 4. Sistem mengubah status pengaduan "sedang diproses" 5. Aktor menekan tombol "selesai" 6. Sistem mengubah status pengaduan "selesai" 7. Sistem akan mengubah status pengaduan sesuai dengan *dropdown* status yang ditekan oleh aktor
Alternative Flows	1. Jika aktor menekan tombol "sedang di vendor" setelah menerima detail pengaduan maka sistem akan mengubah status pengaduan "sedang di vendor"
Post-Condition	Aktor berhasil mengubah status pengaduan

Tabel 4.16 Skenario *Use Case* Melihat Jadwal *Maintenance*

Melihat jadwal *maintenance* (PLM_F_1000)	
Actor	Kasubag dan Pegawai Umum dan Perlengkapan, Pengadu
Objective	Melihat jadwal *maintenance* suatu inventaris yang sudah dibuat

Tabel 4.16 Skenario *Use Case* Melihat Jadwal *Maintenance* (Lanjutan)

Melihat jadwal *maintenance* (PLM_F_1000)	
Pre-condition	Aktor telah berada di halaman beranda
Main Flow	1. Aktor menekan tombol "jadwal *maintenance*" 2. Sistem menampilkan jadwal *maintenance* 3. Aktor menekan tanggal tertentu 4. Sistem menampilkan *maintenance* pada tanggal tersebut
Post-Condition	Aktor berhasil melihat jadwal *maintenance*

Tabel 4.17 Skenario *Use Case* Mengisi *Form* Jadwal *Maintenance*

Mengisi *form* jadwal *maintenance* (PLM_F_1100)	
Actor	Pegawai Umum dan Perlengkapan
Objective	Mengisi *form* untuk membuat jadwal *maintenance*
Pre-condition	Aktor telah berada pada halaman beranda
Main Flow	1. Aktor menekan tombol "buat jadwal" 2. Sistem menampilkan halaman jadwal *maintenance* 3. Aktor menekan tanggal *maintenance* 4. Sistem menampilkan *form* jadwal *maintenance* 5. Aktor mengisi semua kolom di *form* jadwal *maintenance* 6. Aktor menekan tombol "submit" 7. Sistem menampilkan halaman jadwal *maintenance*
Alternative Flows	1. Jika ada kolom yang belum diisi maka muncul notifikasi "semua kolom harus diisi"
Post-Condition	Aktor berhasil membuat jadwal *maintenance*

Tabel 4.18 Skenario *Use Case* Membaca Notifikasi *Maintenance*

Membaca notifikasi *maintenance* (PLM_F_1200)	
Actor	Pegawai Umum dan Perlengkapan
Objective	Membaca notifikasi *maintenance* dan melaksanakannya
Pre-condition	Sistem mengirimkan notifikasi *maintenance*
Main Flow	1. Aktor menekan notifikasi *maintenance* 2. Sistem menampilkan halaman jadwal *maintenance* 3. Aktor menekan tombol "kerjakan" 4. Sistem mengubah status menjadi selesai
Alternative Flows	1. Jika aktor melakukan *swipe* pada notifikasi yang diterima maka aktor akan tetap berada *current screen* 2. Jika aktor tidak menekan tombol kerjakan maka status "belum dikerjakan"
Post-Condition	Aktor berhasil mengerjakan *maintenance*

Tabel 4.19 Skenario *Use Case* Membaca Notifikasi *Rating*

Membaca notifikasi *rating* (PLM_F_1300)	
Actor	Pegawai Umum dan Perlengkapan
Objective	Membaca notifikasi *rating* dari pengadu
Pre-condition	Sistem mengirimkan notifikasi jumlah *rating* pegawai
Main Flow	1. Aktor menekan notifikasi *rating* 2. Sistem menampilkan halaman beranda yang terdapat *rating* pegawai
Post-Condition	Aktor berhasil melihat *rating* yang didapatkan

Tabel 4.20 Skenario *Use Case* Membuat Pengaduan

Membuat pengaduan (PLM_F_1400)	
Actor	Pengadu
Objective	Memilih kategori dan mengisi *form* pengaduan
Pre-condition	Aktor telah berada pada halaman beranda
Main Flow	1. Aktor memilih salah satu kategori pengaduan 2. Sistem menampilkan *form* pengaduan 3. Aktor mengisi semua kolom di *form* pengaduan 4. Aktor menekan tombol *submit* 5. Sistem menampilkan halaman daftar pengaduan
Alternative Flows	1. Jika ada kolom yang belum diisi maka muncul notifikasi "semua kolom harus diisi"
Post-Condition	Aktor berhasil mengajukan pengaduan

Tabel 4.21 Skenario *Use Case* Membaca Notifikasi Pengaduan

Membaca notifikasi status pengaduan (PLM_F_1500)	
Actor	Pengadu
Objective	Menerima notifikasi mengenai status pengaduan yang telah diajukan
Pre-condition	Sistem mengirimkan notifikasi status pengaduan
Main Flow	1. Aktor menekan notifikasi status pengaduan 2. Sistem menampilkan halaman detail pengaduan
Alternative Flows	1. Jika aktor melakukan *swipe* pada notifikasi yang diterima maka aktor akan tetap berada *current screen*
Post-Condition	Aktor dapat melihat status pengaduan yang telah diajukannya beserta status terbaru pengaduan dan pegawai yang menangani

Tabel 4.22 Skenario *Use Case* Memberi *Rating*

Memberi *rating* (PLM_F_1600)	
Actor	Pengadu
Objective	Memberi *rating* kepada pegawai yang ditugaskan menangani pengaduan
Pre-condition	Sistem menampilkan notifikasi *rating* pengaduan
Main Flow	1. Aktor memberi *rating* 2. Aktor menekan tombol *submit* 3. Sistem menyimpan *rating* pegawai dalam *database*
Alternative Flows	1. Jika aktor melakukan *swipe* pada notifikasi yang diterima maka aktor akan tetap berada *current screen*
Post-Condition	Aktor dapat memberi *rating* kepada pegawai

Tabel 4.23 Skenario *Use Case* Melihat Riwayat Kerusakan

Melihat riwayat kerusakan (PLM_F_1700)	
Actor	Kasubag dan Pegawai Umum dan Perlengkapan, Pengadu
Objective	Melihat daftar riwayat kerusakan yang pernah dialami suatu inventaris
Pre-condition	Aktor berada di halaman detail pengaduan
Main Flow	1. Aktor menekan nomor inventaris 2. Sistem menampilkan halaman riwayat kerusakan
Post-Condition	Aktor dapat melihat riwayat kerusakan inventaris

Tabel 4.24 Skenario *Use Case* Melihat Riwayat *Maintenance*

Melihat riwayat *maintenance* (PLM_F_1800)	
Actor	Pegawai Umum dan Perlengkapan
Objective	Melihat daftar riwayat *maintenance* yang pernah dilakukan pada suatu inventaris
Pre-condition	Aktor berada di halaman jadwal *maintenance*
Main Flow	1. Aktor menekan salah satu tanggal pada jadwal 2. Sistem menampilkan daftar *maintenance* pada tanggal tersebut 3. Aktor menekan nomor inventaris 4. Sistem menampilkan halaman riwayat *maintenance*
Post-Condition	Aktor dapat melihat riwayat *maintenance* inventaris

BAB 5 PERANCANGAN DAN IMPLEMENTASI

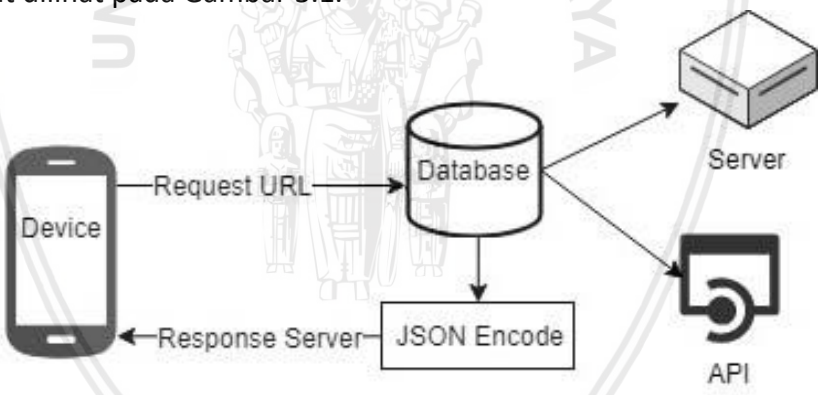
Bab perancangan dan implementasi membahas mengenai perancangan-perancangan aplikasi pengaduan layanan dan *maintenance* beserta implementasi dari perancangan-perancangan yang telah dibuat meliputi langkah-langkah yang dilakukan dalam mengumpulkan dan mendefinisikan kebutuhan-kebutuhan dari aplikasi *mobile* Pengaduan Layanan berbasis *Android* yang akan dibangun.

5.1 Perancangan Aplikasi

Setelah tahap rekayasa kebutuhan yang dilakukan dengan *Object Oriented Analysis* (OOA), maka tahap selanjutnya adalah perancangan dengan melakukan *Object Oriented Design* (OOD). OOD merupakan salah satu rangkaian pendekatan berorientasi objek untuk melakukan perancangan suatu aplikasi. Pada penelitian ini proses perancangan terbagi menjadi perancangan arsitektur, *sequence diagram, class diagram,* perancangan komponen, perancangan data, dan perancangan antarmuka.

5.1.1 Perancangan Arsitektur

Perancangan arsitektur aplikasi pengaduan layanan dan *maintenance* berbasi *android* dapat dilihat pada Gambar 5.1.



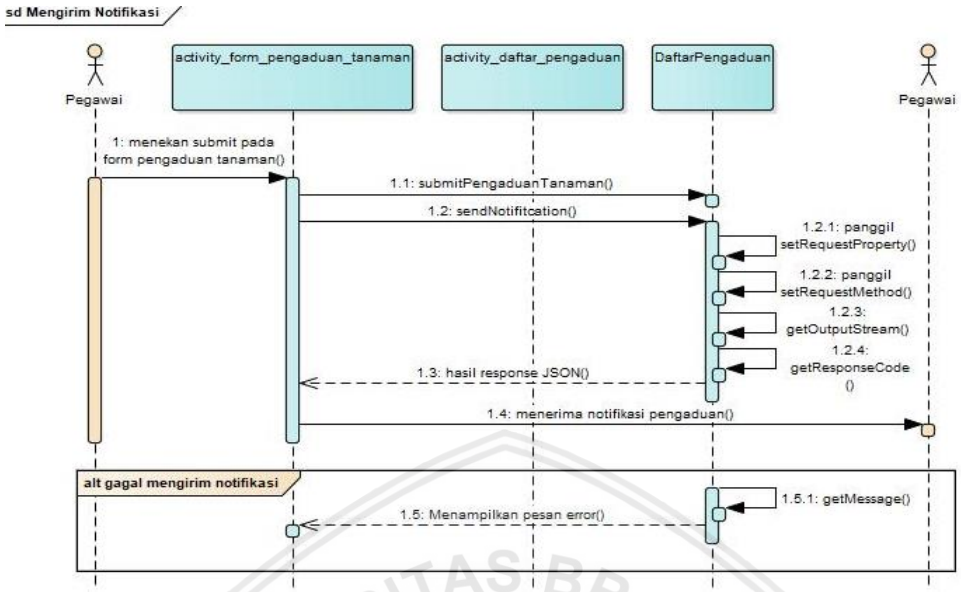
Gambar 5.1 Arsitektur Aplikasi

Aplikasi dibangun menggunakan bahasa java dengan tampilan menggunakan xml. Arsitektur *Firebase* yang digunakan adalah *two-tier.* Pada arsitektur ini, d*atabase* menjadi penghubung *device* dengan *server* dan juga mengambil data dari API. Alur komunikasi dimulai ketika *client* mengirimkan *request* URL ke *server* menggunakan HTTP. Kemudian *database* yang terhubung dengan *server* menerima *request* dan mengambil data tersebut. *Server* memberi *response* dan *database* melakukan *encode* data menjadi JSON. *Response* kepada *client* adalah berupa JSON yang kemudian diterima oleh *client* dengan melakukan *decode.* Kemudian hasil *decode* tersebut ditampilkan pada layar antarmuka.

5.1.2 *Sequence Diagram*

Sequence diagram menjelaskan interaksi antar objek dalam satu fungsi. *Sequence diagram* dalam dapat dilihat pada Gambar 5.2 sampai Gambar 5.4.

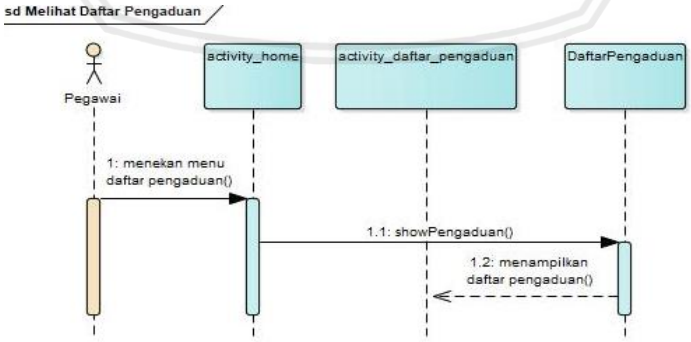
5.1.2.1 *Sequence Diagram* Mengirim Notifikasi



Gambar 5.2 *Sequence Diagram* Mengirim Notifikasi

Pada Gambar 5.2 tersebut, interaksi antar objek dimulai ketika aktor menekan *submit* pada *form* pengaduan tanaman. Kemudian memanggil *method* submitPengaduanTanaman() dan *method* sendNotification() pada *class* DaftarPengaduan.java yang kemudian memanggil *method* setRequestProperty() dan setRequestMethod() pada kelas tersebut. Selanjutnya pada kelas yang sama, memanggil *method* getOutputStream() dan getResponseCode() untuk mengirim *request* pada API pada OneSignal. Kemudian OneSignal mengirimkan *response* berupa *query* JSON untuk menampilkan halaman daftar pengaduan jika pengaduan berhasil dikirim. Jika pengaduan gagal dikirim dan pegawai tidak mendapatkan notifikasi pengaduan maka pada *class* DaftarPengaduan.java memanggil getMessage() dan menampilkan pesan *error*.

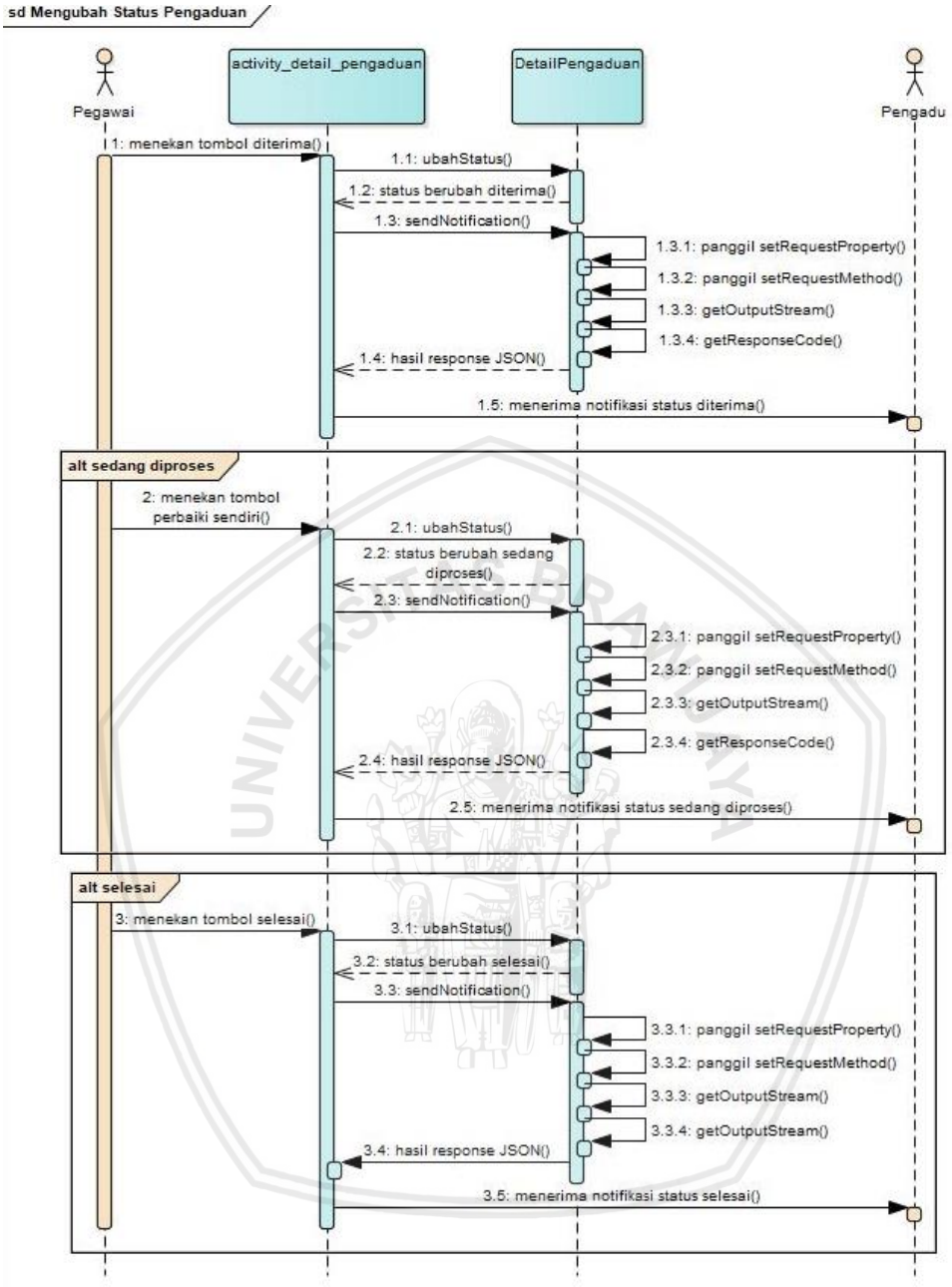
5.1.2.2 *Sequence Diagram* Melihat Daftar Pengaduan



Gambar 5.3 *Sequence Diagram* Melihat Daftar Pengaduan

Pada Gambar 5.3 tersebut, interaksi antar objek dimulai ketika aktor menekan menu daftar pengaduan pada halaman beranda. Kemudian memanggil *method* showPengaduan() pada *class* DaftarPengaduan.java yang kemudian memanggil id pengaduan pada entitas pengaduan di *database.* Hasil *query* pemanggilan tersebut akan menampilkan daftar pengaduan.

5.1.2.3 *Sequence Diagram* Mengubah Status Pengaduan

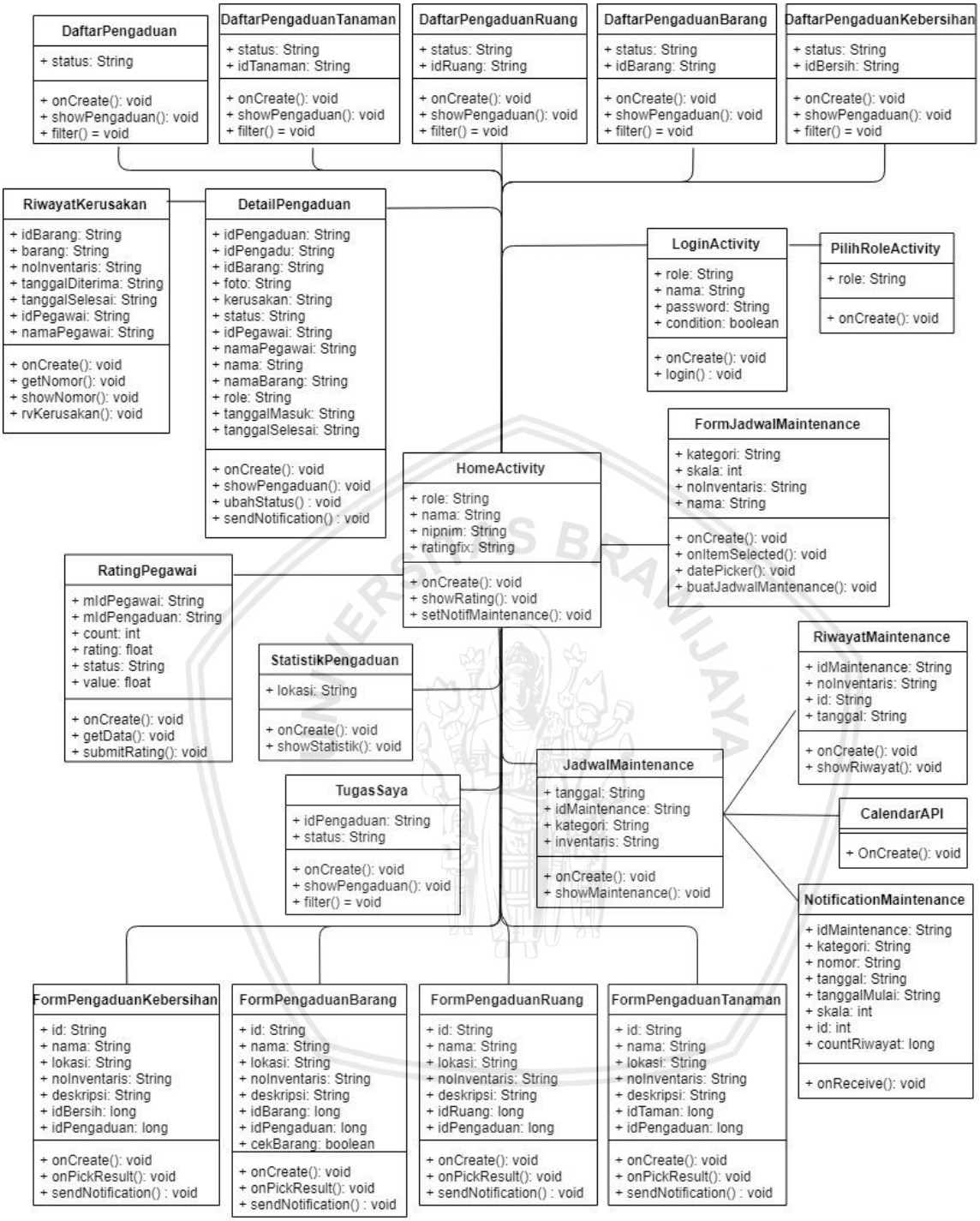


Gambar 5.4 *Sequence Diagram* Mengubah Status Pengaduan

Pada Gambar 5.4 tersebut, interaksi dimulai ketika aktor memilih terima di halaman detail pengaduan. Kemudian akan memanggil *method* ubahStatus() yang akan memanggil dan mengubah variabel status pada entitas Pengaduan di *database.* Setelah itu maka status pengaduan pun akan berubah. Setelah status berubah maka akan memanggil *method* sendNotification untuk mengirim notifikasi pada pengadu.

5.1.3 *Class Diagram*

Class diagram menjelaskan relasi antar kelas dan struktur kelas, termasuk atribut dan *method. Class diagram* penelitian ini dapat dilihat pada Gambar 5.5.



Gambar 5.5 *Class Diagram* Aplikasi Pengaduan Layanan dan *Maintenance*

Berdasarkan gambar tersebut, aplikasi pengaduan layanan dan *maintenance* memiliki 18 kelas dalam implementasinya.

5.1.4 Perancangan Komponen

Perancangan komponen menjelaskan algoritme dalam bentuk *pseudocode* dari beberapa *method* yang akan menjadi dasar dalam tahap implementasi yang dapat dilihat pada Tabel 5.1 sampai dengan Tabel 5.3.

Tabel 5.1 *Pseudocode Method* Mengirimkan Notifikasi

```
1   Mulai method sendNotification
2     TRY untuk mengecek koneksi HTTP response
3     Otorisasi input dengan tipe data JSON
4         IF koneksi terpenuhi
5         AND koneksi kurang dari internal error
6             Mengirim json respon
7         ELSE jika koneksi gagal
8         END IF
9     Mengirim json response
10    CATCH untuk kondisi error
11    END TRY
12  Selesai
```

Tabel 5.1 merupakan hasil perancangan komponen dari fungsi mengirimkan notifikasi. *Pseudocode* dimulai dengan operasi *try* untuk mengecek koneksi dari HTTP *response* kemudian otorisasi input pada projek API OneSignal dengan tipe data JSON. Jika syarat koneksi terpenuhi maka akan mengirim respon dalam bentuk JSON dan jika gagal maka tidak dapat mengirim respon JSON.

Tabel 5.2 *Pseudocode* Komponen *Method* Melihat Daftar Pengaduan

```
1   Mulai method onDataChange
2     FOR untuk mengambil data pada child
3     Inisialisasi data yang diambil
4     END FOR
5     IF jika data pengaduan kosong
6     Menampilkan data kosong
7     ELSE jika data ada
8     END IF
9   Selesai
```

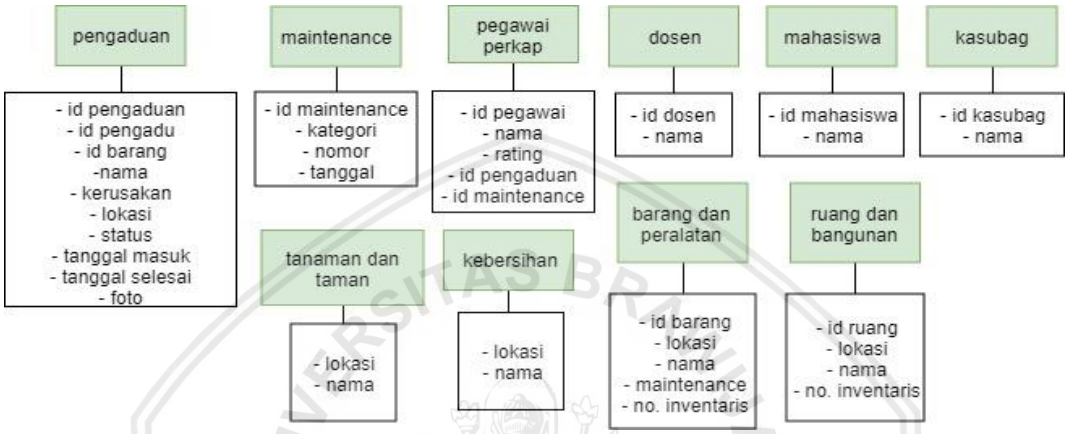
Tabel 5.2 merupakan hasil perancangan komponen dari fungsi melihat daftar pengaduan. *Pseudocode* dimulai dengan operasi *for* untuk mengambil data pada *child* pada *database.* Kemudian menginisialisasi data-data yang akan diambil untuk kemudian dimasukkan kedalam *list* pengaduan. Jika data telah dimasukkan maka dapat menampilkan data pengaduan.

Tabel 5.3 *Pseudocode Method* Mengubah Status Pengaduan

```
1   Mulai method ubahStatus
2     IF role pegawai
3         IF status belum diterima
4         Mengubah status diterima pada database
5         ELSE IF status diterima
6             IF role pegawai
7             Menampilkan tombol perbaiki sendiri dan sedang di
              vendor
8             END IF
9             Mengubah status sedang diproses dan sedang di
              vendor pada database
10        ELSE IF status sedang diproses
11        OR status sedang di vendor
12            IF role pegawai
13            END IF
14            Menampilkan tombol selesai
15            END IF
16     END IF
17            Selesai
```

Tabel 5.3 merupakan hasil perancangan komponen dari fungsi Mengubah Status Pengaduan. *Pseudocode* dimulai dengan operasi seleksi apakah *user* adalah pegawai dan seleksi apakah status belum diterima, maka status akan berubah menjadi diterima pada *database.* Kemudian jika status sudah diterima maka akan menampilkan tombol perbaiki sendiri dan tombol sedang di vendor. Seleksi selanjutnya adalah apakah status sedang diproses atau sedang di vendor dan apakah *role user* adalah pegawai maka akan menampilkan tombol selesai.

5.1.5 Perancangan Data

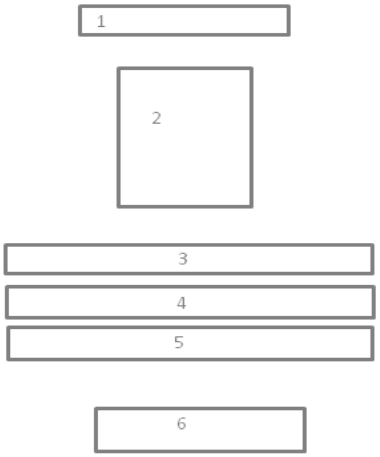


Gambar 5.6 Perancangan Data

Gambar 5.6 merupakan perancangan data yang menggambarkan struktur *database* dalam penelitian ini. Terdapat 10 entitas yang telah diidentifikasi yaitu entitas dosen, mahasiswa, pegawai perkap, pengaduan, *maintenance,* kebersihan, ruang dan bangunan, barang dan peralatan, serta tanaman dan taman. Implementasi perancangan *database* ini akan dibuat menggunakan *Firebase Realtime Database* yang merupakan *database* NoSQL sehingga tidak ada relasi antar entitas. *Firebase Realtime Database* merupakan data yang disimpan sebagai JSON dan disinkronkan secara *realtime* ke setiap klien yang terhubung.

5.1.6 Perancangan Antarmuka

5.1.6.1 Perancangan Antarmuka *Form* Pengaduan Tanaman



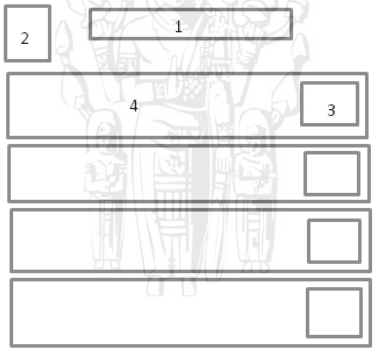
Gambar 5.7 Perancangan Antarmuka *Form* Pengaduan Tanaman

Gambar 5.7 merupakan gambaran perancangan antarmuka halaman *form* pengaduan yang masing-masing komponennya akan dijelaskan pada Tabel 5.4.

Tabel 5.4 Perancangan Antarmuka *Form* Pengaduan Tanaman

No	Nama Objek	Tipe	Keterangan
1	Judul halaman	*TextView*	Menampilkan judul halaman yaitu *form* pengaduan tanaman
2	*Upload* gambar	*Image*	Digunakan untuk mengunggah gambar yang dilaporkan
3	Judul pengaduan	*Textbox*	Digunakan untuk memasukkan judul pengaduan yang akan dilaporkan
4	Lokasi	*Textbox*	Digunakan untuk memasukkan lokasi tanaman atau taman yang akan dilaporkan
5	Deskripsi	*Textbox*	Digunakan untuk memasukkan deskripsi tanaman atau taman yang akan dilaporkan
6	*Submit*	*Button*	Digunakan untuk mengirimkan data pengaduan

5.1.6.2 Perancangan Antarmuka Halaman Daftar Pengadaun



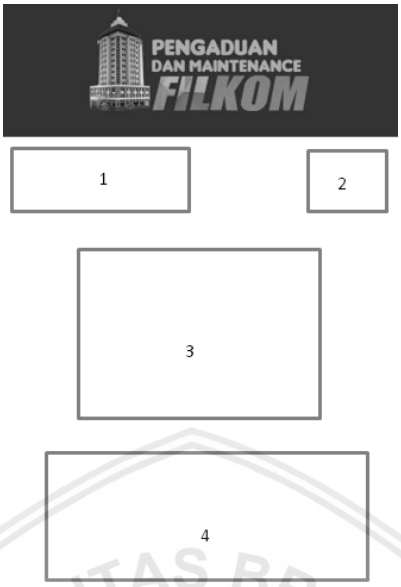
Gambar 5.8 Perancangan Antarmuka Halaman Daftar Pengaduan

Gambar 5.8 merupakan gambaran perancangan antarmuka halaman daftar pengaduan yang masing-masing komponennya akan dijelaskan pada Tabel 5.5.

Tabel 5.5 Perancangan Antarmuka Halaman Daftar Pengaduan

No	Nama Objek	Tipe	Keterangan
1	Judul halaman	*TextView*	Menampilkan judul halaman yaitu *form* pengaduan tanaman
2	*Filter*	*Button Filter*	Digunakan untuk menampilkan pengaduan berdasarkan statusnya
3	Status pengaduan	*TextView*	Menampilkan status pengaduan
4	Penjelasan singkat pengaduan	*TextView*	Digunakan untuk melihat tanggal, judul, dan deskripsi singkat pengaduan

5.1.6.3 Perancangan Antarmuka Halaman Detail Pengaduan



Gambar 5.9 Perancangan Antarmuka Halaman Detail Pengaduan

Gambar 5.9 merupakan gambaran perancangan antarmuka halaman detail pengaduan yang masing-masing komponennya akan dijelaskan pada Tabel 5.6.

Tabel 5.6 Perancangan Antarmuka Halaman Detail Pengaduan

No	Nama objek	Tipe	Keterangan
1	Identitas pelapor	*TextView*	Menampilkan identitas pelapor yaitu nama dan NIM/NIP nya
2	Status pengaduan	*TextView*	Menampilkan status pengaduan
3	Foto keluhan	*ImageView*	Menampilkan gambar keluhan
4	Deskripsi pengaduan	*TextView*	Menampilkan deskripsi pengaduan

5.2 Implementasi aplikasi

5.2.1 Spesifikasi Sistem

Dalam melakukan proses implementasi, penulis membutuhkan perangkat keras yaitu laptop dan juga perangkat lunak untuk membuat kode program yang akan dijalankan pada *smartphone.* Spesifikasi perangkat keras dan perangkat lunak yang digunakan terdapat pada Tabel 5.7 dan 5.8.

Tabel 5.7 Spesifikasi Perangkat Keras

Dell Inspiron 3442	
Processor	Intel® Core™ i3-4005U
Memory	4 GB RAM
Graphic Card	NVIDIA GeForce

Tabel 5.8 Spesifikasi Perangkat Lunak

Dell Inspiron 3442	
Operating System	Microsoft Windows 10 Home 64bit
Programming Language	Java
Programming Tool	Android Studio 3.3
Android SDK Tools	26.1.0
Gradle Version	4.12
Database Management System (DBMS)	Firebase 11.0.4
Notification Service	OneSignal

5.2.2 Implementasi Kode Program

5.2.2.1 Kode Program *Method* Mengirim Notifikasi

Nama *class*: FormPengaduanTanaman.java

Nama Operasi: `sendNotification()`

Deskripsi: kode ini Tabel 5.9 berfungsi mengirimkan notifikasi melalui json. Baris ke-3 sampai 12 merupakan *code* instansiasi masukan dan otorisasi tipe data pada json. Pada baris ke-13, jika tipe data sesuai maka menampilkan "pengaduan telah diterima". Pada baris ke-20 sampai dengan 28 merupakan kode untuk mengecek koneksi HttpResponse.

Source Code:

Tabel 5.9 Kode Program *Method* Mengirim Notifikasi

```
1     private void sendNotification(){
2         try {
3             String jsonResponse;
4             URL url = new
    URL("https://onesignal.com/api/v1/notifications");
5             HttpURLConnection con =
    (HttpURLConnection)url.openConnection();
6             con.setUseCaches(false);
7             con.setDoOutput(true);
8             con.setDoInput(true);
9             con.setRequestProperty("Content-Type",
    "application/json; charset=UTF-8");
10            con.setRequestProperty("Authorization", "Basic
    NzExODUxODAtMTVhNy00OTM3LWI4OWUtY2JjYzJiODIzMDk0");
11    con.setRequestMethod("POST");
12                String strJsonBody = ("{"
                      + "\"app_id\": \"10d60748-fe76-4739-b4d3-
    8e4b91743c3a\","
                      + "\"filters\": [{\"field\": \"tag\",
    \"key\": \"User\", \"relation\": \"=\", \"value\":
    \"Pegawai\"}],"
                      + "\"data\": {\"foo\": \"bar\"},"
                      + "\"contents\": {\"en\": \"Pengaduan Anda
    telah diterima\"},"+ "\"headings\": {\"en\": \"Pengaduan dan
    Maintenance FILKOM\"}"+ "}");
```

Tabel 5.9 Kode Program *Method* Mengirim Notifikasi (Lanjutan)

```
13  System.out.println("strJsonBody:\n" + strJsonBody);
14              byte[] sendBytes = strJsonBody.getBytes("UTF-8");
15              con.setFixedLengthStreamingMode(sendBytes.length);
16              OutputStream outputStream = con.getOutputStream();
17              outputStream.write(sendBytes);
18              int httpResponse = con.getResponseCode();
19              System.out.println("httpResponse: " +
    httpResponse);
20              if (  httpResponse >= HttpURLConnection.HTTP_OK
                        && httpResponse <
    HttpURLConnection.HTTP_BAD_REQUEST) {
21                  Scanner scanner = new
    Scanner(con.getInputStream(), "UTF-8");
22                  jsonResponse =
    scanner.useDelimiter("\\A").hasNext() ? scanner.next() : "";
23                  scanner.close();}
24              else {
25  Scanner scanner = new Scanner(con.getErrorStream(), "UTF-8");
26  jsonResponse= scanner.useDelimiter("\\A").hasNext() ?
27  scanner.next() : "";
28                  scanner.close();            }
29              System.out.println("jsonResponse:\n" +
30  jsonResponse);
            } catch(Throwable t) {
31  //              t.printStackTrace();
32              Toast.makeText(this, "Gagal mengirimkan
33  notifikasi", Toast.LENGTH_SHORT).show();}}
```

5.2.2.2 Kode Program *Method* Melihat Daftar Pengaduan

Nama *class*: DaftarPengaduan.java

Nama Operasi: `showPengaduan()`

Deskripsi: kode pada Tabel 5.10 berfungsi menampilkan halaman daftar pengaduan. Baris ke-3 sampai 13 berfungsi mengambil data pengaduan. Baris ke-14 sampai 21 merupakan kode_untuk mengecek perubahan data dengan adapter dan menampilkan list dalam bentuk *recycler view*.

Source Code:

Tabel 5.10 Kode Program *Method* Melihat Daftar Pengaduan

```
1   public void onDataChange(DataSnapshot dataSnapshot) {
2               pengaduanList.clear();
3               for (DataSnapshot data: dataSnapshot.getChildren()){
4                   Pengaduan pengaduan = new Pengaduan();
5   pengaduan.setFoto(data.child("foto").getValue(String.class));
6   pengaduan.setIdBarang(data.child("idBarang").getValue(String.cla
    ss));
7   pengaduan.setIdPengadu(data.child("idPengadu").getValue(String.c
    lass));
8   pengaduan.setIdPengaduan(data.child("idPengaduan").getValue(Stri
    ng.class));
9   pengaduan.setKerusakan(data.child("kerusakan").getValue(String.c
    lass));
10  pengaduan.setLokasi(data.child("lokasi").getValue(String.class))
11  pengaduan.setStatus(data.child("status").getValue(String.class);
12  pengaduan.setTanggalMasuk(data.child("tanggalMasuk").getValue(St
    ring.class));
```

Tabel 5.10 Kode Program *Method* Melihat Daftar Pengaduan (Lanjutan)

```
13  pengaduan.setTanggalSelesai(data.child("tanggalSelesai").get
    Value(String.class));
14                  pengaduanList.add(pengaduan);
15                  adapter.notifyDataSetChanged();}
16              if                      (pengaduanList.isEmpty()){
17                  recyclerView.setVisibility(View.GONE);
18                  tvKosong.setVisibility(View.VISIBLE);
19              }else{
20                  recyclerView.setVisibility(View.VISIBLE);
21                  tvKosong.setVisibility(View.GONE); }}
```

5.2.2.3 Kode Program *Method* Mengubah Status Pengaduan

Nama *class*: DetailPengaduan.java

Nama Operasi: `ubahStatus()`

Deskripsi: kode pada Tabel 5.11 berfungsi untuk mengubah status pengaduan. Pada baris ke-7 sampai dengan 18 adalah kode untuk status belum diterima. Pada baris ke-19 sampai dengan 36 adalah kode untuk status diterima. Pada baris-37 sampai dengan 50 adalah kode untuk status sedang diproses atau sedang di vendor.

Source Code:

Tabel 5.11 Kode Program *Method* Mengubah Status Pengaduan

```
1   public void ubahStatus(String idPengaduan,String idPengadu,
2   String idPegawai, String status){
3       String role = preferences.getString("role",null);
4       Log.d("Role",role);
5       String idPegawaiCurrent = preferences.getString("id",null);
6       if (role.equals("pegawai")){
7           if (status.equals("belum diterima")){
8               btnTerima.setVisibility(View.VISIBLE);
9               btnTerima.setOnClickListener(v->{
10                  sendNotifitcation(idPengadu,status);
11  mRef.child("pengaduan").child(idPengaduan).child("status").setVa
    lue("diterima");
12  mRef.child("pengaduan").child(idPengaduan).child("idPegawai").se
    tValue(idPegawaiCurrent);
13  mRef.child("pegawaiPerkap").child(idPegawaiCurrent).child("idPen
    gaduan").child(idPengaduan).setValue(idPengaduan);
14                  btnPerbaiki.setVisibility(View.VISIBLE);
15                  btnVendor.setVisibility(View.VISIBLE);
16                  btnTerima.setVisibility(View.GONE);
17                  tvStatus.setText("diterima");
18  tvStatus.setBackgroundResource(R.drawable.bg_diterima);});
19          }else if (status.equals("diterima")){
20              if (idPegawaiCurrent.equals(idPegawai)){
21                  btnPerbaiki.setVisibility(View.VISIBLE);
22                  btnVendor.setVisibility(View.VISIBLE);}
23              btnPerbaiki.setOnClickListener(v->{
24  mRef.child("pengaduan").child(idPengaduan).child("status").setVa
    lue("sedang diproses");
25                  btnPerbaiki.setVisibility(View.GONE);
26                  btnVendor.setVisibility(View.GONE);
27                  btnSelesai.setVisibility(View.VISIBLE);
28                  tvStatus.setText("sedang diproses");
```

Tabel 5.11 Kode Program *Method* Mengubah Status Pengaduan (Lanjutan)

```
29  tvStatus.setBackgroundResource(R.drawable.bg_diproses);});
30              btnVendor.setOnClickListener(v->{
31  mRef.child("pengaduan").child(idPengaduan).child("status").setVa
    lue("sedang divendor");
32                  btnPerbaiki.setVisibility(View.GONE);
33                  btnVendor.setVisibility(View.GONE);
34                  btnSelesai.setVisibility(View.VISIBLE);
35                  tvStatus.setText("sedang divendor");
36  tvStatus.setBackgroundResource(R.drawable.bg_divendor); });
37          }else if (status.equals("sedang diproses") ||
38  status.equals("sedang divendor")){
39              if (idPegawaiCurrent.equals(idPegawai)){
40                  btnSelesai.setVisibility(View.VISIBLE);}
41  btnSelesai.setOnClickListener(v->{
42                  sendNotifitcation(idPengadu,status);
43                  Calendar calendar = Calendar.getInstance();
44                  SimpleDateFormat dateFormat = new
    SimpleDateFormat("dd MMMM yyyy",new Locale("ID"));
45                String tanggalSelesai =
    dateFormat.format(calendar.getTime());
46  mRef.child("pengaduan").child(idPengaduan).child("status").setValue(
    "selesai");
47  mRef.child("pengaduan").child(idPengaduan).child("tanggalSelesai"
    ).setValue(tanggalSelesai);
48                  btnSelesai.setVisibility(View.GONE);
49                  tvStatus.setText("selesai");
50  tvStatus.setBackgroundResource(R.drawable.bg_selesai); });}}}
```

5.2.3 Impelementasi Data

Dalam melakukan implementasi data, penulis menggunakan DBMS *Firebase* 11.0.4 *Realtime Database* yang merupakan *database* NoSQL. Dalam *database* ini, pemilik dapat memasukkan data dengan *login* terlebih dahulu pada *database* yang ada dalam *firebase.* Pemilik juga dapat mengundang pengguna lain dengan cara *invite* sehingga pengguna tersebut dapat mengedit *database.* Tampilan *database* dapat dilihat pada Gambar 5.10.

Gambar 5.10 Implementasi *Database*

5.2.4 Implementasi Antarmuka

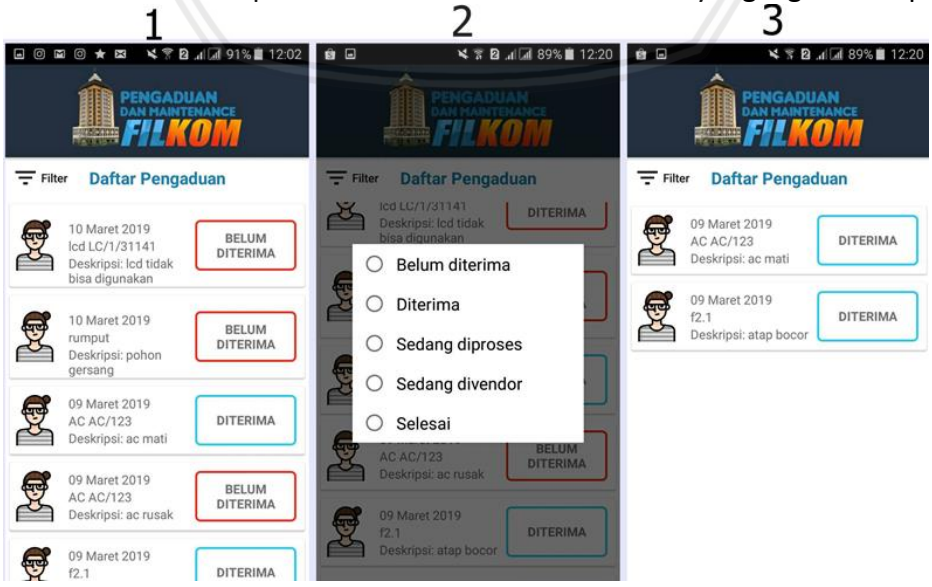
5.2.4.1 Antarmuka Membuat Pengaduan

Halaman ini digunakan untuk mengadukan keluhan sesuai kategori yang dipilih pengadu. Pada halaman ini, aktor diharuskan menginput nama pengaduan, lokasi, dan deskripsi dari keluhan tersebut. Implementasi antarmuka *form* pengaduan dibagi menjadi empat kategori yaitu tanaman, kebersihan, barang, dan bangunan. Antarmuka *form* pengaduan barang dapat dilihat pada Gambar 5.11.

Gambar 5.11 *Form* Pengaduan

5.2.4.2 Antarmuka Melihat Daftar Pengaduan

Halaman ini digunakan untuk menampilkan daftar pengaduan yang telah dikirim pengadu. Implementasi antarmuka halaman ini dapat dilihat pada Gambar 5.12. Pada gambar tersebut, nomor 1 merupakan tampilan halaman ketika aktor memilih menu daftar pengaduan pada halaman beranda. Nomor 2 adalah tampilan ketika aktor melakukan *filter* berdasarkan status pengaduan tersebut dan nomor 3 adalah tampilan setelah aktor memilih status yang ingin ditampilkan.



Gambar 5.12 Halaman Daftar Pengaduan

5.2.4.3 Antarmuka Melihat Detail Pengaduan

Halaman ini digunakan untuk menampilkan detail pengaduan yang telah dikirim oleh pengadu. Pada halaman ini pegawai dapat menekan tombol terima yang kemudian pegawai tersebut bertugas menanganinya. Pegawai yang menangani tersebut dapat mengubah status pengaduan. Implementasi antarmuka halaman ini dapat dilihat pada Gambar 5.13. Gambar tersebut adalah ketika pengaduan sudah diterima oleh pegawai sehingga statusnya sudah diterima dan tombol berubah menjadi “perbaiki sendiri” dan tombol “serahkan ke vendor”.

Gambar 5.13 Halaman Detail Pengaduan

BAB 6 PENGUJIAN

Pengujian perangkat lunak komponen aplikasi pengaduan layanan dan *maintenance* (aplikasi PLM) dilakukan dengan tujuan untuk menemukan kekurangan maupun kesalahan pada sistem, serta mengetahui kesesuaian antara fungsi-fungsi aplikasi yang diimplementasikan berdasarkan hasil rekayasa kebutuhan dan perancangan yang sudah ditentukan pada tahap sebelumnya. Selain itu dengan dilakukannya pengujian, dapat diketahui adanya kesalahan-kesalahan dalam proses *coding* maupun dalam menghasilkan *output* program. Sehingga kesalahan-kesalahan tersebut dapat diperbaiki.

Dalam pengerjaan aplikasi pengaduan layanan dan *maintenance*, pengujian fungsional dilakukan dengan pengujian *black box* dan *white box.* Pengujian *black box* adalah pengujian yang difokuskan pada persyaratan fungsional atau kevalidan *input* dan *output* yang dihasilkan dari perangkat lunak yang dibangun. Pengujian *black box* ini menggunakan metode validasi dengan cara memberi *input* dari *user* kepada sistem yang sudah berjalan dan mengamati hasil *output* dari sistem. Sedangkan pengujian *white box* adalah pengujian yang difokuskan pada pengecekan detail perancangan aplikasi yang dibangun. Pengujian *white box* ini dilakukan dengan pengujian unit menggunakan metode *basic path testing*, yaitu dengan membuat *independent path* berdasarkan rancangan algoritme berupa *pseudocode* atau *flowchart* yang diubah menjadi *flowgraph* terlebih dahulu.

Pengujian selanjutnya yang dilakukan adalah pengujian untuk kebutuhan non fungsional perangkat lunak. Pengujian non fungsional yang dilakukan dalam pengembangan aplikasi pengaduan layanan dan *maintenance* adalah pengujian *usability* dan pengujian *compatibility.* Pengujian *usability* dilakukan untuk menguji kemudahan aplikasi dalam berinteraksi dengan pengguna. Sedangkan pengujian *compatibility* untuk menguji kemampuan aplikasi yang sudah dibuat ketika dijalankan di berbagai perangkat, sistem operasi, atau jaringan yang berbeda.

6.1 Pengujian *Black Box*

6.1.1 Pengujian Validasi *Login* (PLM_F_001)

Tabel 6.1 Skenario Uji *Login*

Skenario Uji *Login*	
Skenario	1. *User* memilih *login* sebagai pengadu 2. *User* mengisi kolom NIM/NIP = 155111222333 dan *password* = 155111 3. Klik tombol “login”
Expected Result	*User* berhasil *login* dan dialihkan ke halaman beranda aplikasi
Result	*User* berhasil *login* dan dialihkan ke halaman beranda aplikasi
Status	Valid

Tabel 6.2 Skenario Uji *Login* Alternatif 1

Skenario Uji *Login* Alternatif 1 (Sebagai Pegawai)	
Skenario	1. *User* mengakses halaman *login* sebagai pegawai 2. *User* mengisi kolom NIM/NIP = 111 dan *password* = 111 3. Klik tombol "login"
Expected Result	*User* berhasil *login* dan dialihkan ke halaman beranda aplikasi
Result	*User* berhasil *login* dan dialihkan ke halaman beranda aplikasi
Status	Valid

Tabel 6.3 Skenario Uji *Login* Alternatif 2

Skenario Uji *Login* Alternatif 2 (Sebagai Kasubag)	
Skenario	4. *User* mengakses halaman *login* sebagai Kasubag 5. *User* mengisi kolom NIM/NIP = 18281 dan *password* = 18281 6. Klik tombol "login"
Expected Result	*User* berhasil *login* dan dialihkan ke halaman beranda aplikasi
Result	*User* berhasil *login* dan dialihkan ke halaman beranda aplikasi
Status	Valid

Tabel 6.4 Skenario Uji *Login* Alternatif 3

Skenario Uji *Login* Alternatif 3 (Kolom NIM/NIP dan atau password salah)	
Skenario	1. *User* memilih *login* sebagai pengadu 2. *User* mengisi kolom NIM/NIP = 1 dan *password* = 90 3. Klik tombol "login"
Expected Result	*User* tidak dapat masuk ke halaman beranda aplikasi PLM dan muncul pesan "data yang dimasukkan salah"
Result	*User* tidak dapat masuk ke halaman beranda aplikasi PLM dan muncul pesan "data yang dimasukkan salah"
Status	Valid

Tabel 6.5 Skenario Uji *Login* Alternatif 4

Skenario Uji *Login* Alternatif 4 (Kolom NIM/NIP dan atau password kosong)	
Skenario	1. *User* memilih *login* sebagai pengadu 2. Klik tombol "login"
Expected Result	*User* tidak dapat masuk ke halaman utama aplikasi PLM dan muncul pesan "NIM/NIP dan *password* harus diisi"
Result	*User* tidak dapat masuk ke halaman utama aplikasi PLM dan muncul pesan "NIM/NIP dan *password* harus diisi"aplikasi
Status	Valid

6.1.2 Pengujian Validasi Logout (PLM_F_200)

Tabel 6.6 Skenario Uji *Logout*

Skenario Uji *Logout*	
Skenario	1. *Login* ke aplikasi 2. *User* memilih menu *logout*
Expected Result	*User* berhasil keluar dari aplikasi dan dialihkan ke halaman *login*
Result	*User* berhasil keluar dari aplikasi dan dialihkan ke halaman *login*
Status	Valid

6.1.3 Pengujian Validasi Melihat Daftar Pengaduan (PLM_F_300)

Tabel 6.7 Skenario Uji Melihat Daftar Pengaduan

Skenario Uji Melihat Daftar Pengaduan	
Skenario	1. *Login* ke aplikasi 2. *User* memilih menu daftar pengaduan
Expected Result	*User* berhasil melihat daftar pengaduan
Result	*User* berhasil melihat daftar pengaduan
Status	Valid

Tabel 6.8 Skenario Uji Melihat Daftar Pengaduan Alternatif 1

Skenario Uji Melihat Daftar Pengaduan Alternatif 1 (Berdasar status yang dipilih)	
Skenario	1. *Login* ke aplikasi 2. *User* memilih menu daftar pengaduan 3. *User* memilih status pada ikon filter
Expected Result	*User* berhasil melihat pengaduan berdasarkan status yang dipilih
Result	*User* berhasil melihat pengaduan berdasarkan status yang dipilih
Status	Valid

6.1.4 Pengujian Validasi Melihat Detail Pengaduan (PLM_F_400)

Tabel 6.9 Skenario Uji Melihat Detail Pengaduan

Skenario Uji Melihat Detail Pengaduan	
Skenario	1. *Login* ke aplikasi 2. *User* memilih menu daftar pengaduan 3. *User* memilih salah satu pengaduan
Expected Result	*User* berhasil melihat detail pengaduan
Result	*User* berhasil melihat detail pengaduan
Status	Valid

6.1.5 Pengujian Validasi Melihat Statistik Pengaduan (PLM_F_500)

Tabel 6.10 Skenario Uji Melihat Statistik Pengaduan

Skenario Uji Melihat Statistik Pengaduan	
Skenario	1. *Login* ke aplikasi 2. *User* memilih menu statistik pengaduan
Expected Result	*User* berhasil melihat statistik pengaduan
Result	*User* berhasil melihat statistik pengaduan
Status	Valid

6.1.6 Pengujian Validasi Membuat Laporan (PLM_F_600)

Tabel 6.11 Skenario Uji Membuat Laporan

Skenario Uji Membuat Laporan	
Skenario	1. *Login* ke aplikasi sebagai Kasubag 2. *User* memilih menu laporan pengaduan 3. *User* menekan ikon unduh
Expected Result	*User* berhasil mengunduh berkas laporan pengaduan
Result	*User* berhasil mengunduh berkas laporan pengaduan
Status	Valid

Tabel 6.12 Skenario Uji Membuat Laporan Alternatif 1

Skenario Uji Membuat Laporan Alternatif 1 (Laporan *Maintenance*)	
Skenario	1. *Login* ke aplikasi sebagai Kasubag 2. *User* memilih menu laporan *maintenance* 3. *User* menekan ikon unduh
Expected Result	*User* berhasil mengunduh berkas laporan *maintenance*
Result	*User* berhasil mengunduh berkas laporan *maintenance*
Status	Valid

Tabel 6.13 Skenario Uji Membuat Laporan Alternatif 2

Skenario Uji Membuat Laporan Alternatif 2 (Laporan *Rating*)	
Skenario	1. *Login* ke aplikasi sebagai Kasubag 2. *User* memilih menu laporan *rating* 3. *User* menekan ikon unduh
Expected Result	*User* berhasil mengunduh berkas laporan *rating*
Result	*User* berhasil mengunduh berkas laporan *rating*
Status	Valid

Tabel 6.14 Skenario Uji Membuat Laporan Alternatif 3

Skenario Uji Membuat Laporan Alternatif 3 (*Filter* Bulan)	
Skenario	1. *Login* ke aplikasi sebagai Kasubag

Tabel 6.14 Skenario Uji Membuat Laporan Alternatif 3 (Lanjutan)

Skenario Uji Membuat Laporan Alternatif 3 (*Filter* Bulan)	
Skenario	2. *User* memilih menu laporan pengaduan 3. *User* memilih bulan pada ikon *filter*
Expected Result	*User* berhasil melihat laporan berdasarkan bulan yang dipilih
Result	*User* berhasil melihat laporan berdasarkan bulan yang dipilih
Status	Valid

6.1.7 Pengujian Validasi Membaca Notifikasi Pengaduan (PLM_F_700)

Tabel 6.15 Skenario Uji Membaca Notifikasi Pengaduan

Skenario Uji Membaca Notifikasi Pengaduan	
Skenario	1. *Login* ke aplikasi sebagai pegawai 2. *User* menekan notifikasi pengaduan yang diterima 3. *User* menekan menu daftar pengaduan
Expected Result	*User* berhasil membaca pengaduan baru yang masuk
Result	*User* berhasil membaca pengaduan baru yang masuk
Status	Valid

6.1.8 Pengujian Validasi Melihat Pengaduan yang Ditangani (PLM_F_800)

Tabel 6.16 Skenario Uji Melihat Pengaduan Yang Ditangani

Skenario Uji Melihat Pengaduan Yang Ditangani	
Skenario	1. *Login* ke aplikasi sebagai pegawai 2. *User* memilih menu tugas saya
Expected Result	*User* berhasil melihat daftar pengaduan yang ditanganinya
Result	*User* berhasil melihat daftar pengaduan yang ditanganinya
Status	Valid

Tabel 6.17 Skenario Uji Melihat Pengaduan yang Ditangani Alternatif 1

Skenario Uji Melihat Pengaduan yang Ditangani Alternatif 1 (Berdasar Status yang Dipilih)	
Skenario	1. *Login* ke aplikasi sebagai pegawai 2. *User* memilih menu tugas saya 3. *User* memilih status pada ikon filter
Expected Result	*User* berhasil melihat pengaduan yang ditangani berdasarkan status yang dipilih
Result	*User* berhasil melihat pengaduan yang ditangani berdasarkan status yang dipilih
Status	Valid

6.1.9 Pengujian Validasi Mengubah Status Pengaduan (PLM_F_900)

Tabel 6.18 Skenario Uji Mengubah Status Pengaduan

Skenario Uji Mengubah Status Pengaduan	
Skenario	1. *Login* ke aplikasi sebagai pegawai 2. *User* memilih menu tugas saya 3. *User* memilih salah satu pengaduan dengan status “belum diterima” 4. *User* menekan tombol “terima”
Expected Result	*User* berhasil mengubah status pengaduan menjadi “diterima”
Result	*User* berhasil mengubah status pengaduan menjadi “diterima”
Status	Valid

Tabel 6.19 Skenario Uji Mengubah Status Pengaduan Alternatif 1

Skenario Uji Mengubah Status Pengaduan Alternatif 1 (Status Sedang Diproses)	
Skenario	1. *Login* ke aplikasi sebagai pegawai 2. *User* memilih menu tugas saya 3. *User* memilih salah satu pengaduan dengan status “diterima” 4. *User* menekan tombol “perbaiki sendiri”
Expected Result	*User* berhasil mengubah status pengaduan menjadi “sedang diproses”
Result	*User* berhasil mengubah status pengaduan menjadi “sedang diproses”
Status	Valid

Tabel 6.20 Skenario Uji Mengubah Status Pengaduan Alternatif 2

Skenario Uji Mengubah Status Pengaduan Alternatif 2 (Status Sedang di Vendor)	
Skenario	1. *Login* ke aplikasi sebagai pegawai 2. *User* memilih menu tugas saya 3. *User* memilih salah satu pengaduan dengan status “diterima” 4. *User* menekan tombol “serahkan ke vendor”
Expected Result	*User* berhasil mengubah status pengaduan menjadi “sedang di vendor”
Result	*User* berhasil mengubah status pengaduan menjadi “sedang di vendor”
Status	Valid

Tabel 6.21 Skenario Uji Mengubah Status Pengaduan Alternatif 3

Skenario Uji Mengubah Status Pengaduan Alternatif 3 (Status Selesai)	
Skenario	1. *Login* ke aplikasi sebagai pegawai 2. *User* memilih menu tugas saya 3. *User* memilih pengaduan dengan status “sedang diproses” 4. *User* menekan tombol “selesai”
Expected Result	*User* berhasil mengubah status pengaduan menjadi “selesai”
Result	*User* berhasil mengubah status pengaduan menjadi “selesai”
Status	Valid

6.1.10 Pengujian Validasi Melihat Jadwal *Maintenance* (PLM_F_1000)

Tabel 6.22 Skenario Uji Melihat Jadwal *Maintenance*

Skenario Uji Melihat Jadwal *Maintenance*	
Skenario	1. *Login* ke aplikasi 2. *User* memilih menu jadwal *maintenance*
Expected Result	*User* berhasil melihat jadwal *maintenance*
Result	*User* berhasil melihat jadwal *maintenance*
Status	Valid

Tabel 6.23 Skenario Uji Melihat Jadwal *Maintenance* Alternatif 1

Skenario Uji Melihat Jadwal *Maintenance* Alternatif 1 (Melihat Tanggal Tertentu)	
Skenario	1. *Login* ke aplikasi 2. *User* memilih menu jadwal *maintenance* 3. *User* memilih salah satu tanggal
Expected Result	*User* berhasil melihat jadwal maintenance pada tanggal yang dipilih
Result	*User* berhasil melihat jadwal maintenance pada tanggal yang dipilih
Status	Valid

6.1.11 Pengujian Validasi Membuat Jadwal *Maintenance* (PLM_F_1100)

Tabel 6.24 Skenario Uji Membuat Jadwal *Maintenance*

Skenario Uji Membuat Jadwal Maintenance	
Skenario	1. *Login* ke aplikasi sebagai pegawai 2. *User* memilih menu buat jadwal 3. *User* mengisi semua kolom 4. Klik tombol “submit”
Expected Result	*User* berhasil memasukkan jadwal *maintenance*
Result	*User* berhasil memasukkan jadwal *maintenance*
Status	Valid

Tabel 6.25 Skenario Uji Membuat Jadwal *Maintenance* Alternatif 1

Skenario Uji Membuat Jadwal *Maintenance* Alternatif 1 (Kolom Ada yang Kosong)	
Skenario	1. *Login* ke aplikasi sebagai pegawai 2. *User* memilih menu buat jadwal 3. *User* mengosongkan salah satu kolom atau lebih 4. Klik tombol “submit”
Expected Result	*User* tidak bisa memasukkan jadwal *maintenance* dan muncul notifikasi “semua kolom harus diisi”
Result	*User* tidak bisa memasukkan jadwal *maintenance* dan muncul notifikasi “semua kolom harus diisi”
Status	Valid

6.1.12 Pengujian Validasi Notifikasi *Maintenance* (PLM_F_1200)

Tabel 6.26 Skenario Uji Notifikasi *Maintenance*

Skenario Uji Membuat Jadwal Maintenance	
Skenario	1. *Login* ke aplikasi sebagai pegawai 2. *User* menekan notifikasi *maintenance* yang diterima
Expected Result	*User* berhasil melihat jadwal *maintenance* pada tanggal itu
Result	*User* berhasil melihat jadwal *maintenance* pada tanggal itu
Status	Valid

6.1.13 Pengujian Validasi Membaca Notifikasi *Rating* (PLM_F_1300)

Tabel 6.27 Skenario Uji Membaca Notifikasi *Rating*

Skenario Uji Membaca Notifikasi *Rating*	
Skenario	1. *Login* ke aplikasi sebagai pegawai 2. *User* menekan notifikasi *rating* yang diterima
Expected Result	*User* berhasil melihat akumulasi *rating* pada halaman beranda
Result	*User* berhasil melihat akumulasi *rating* pada halaman beranda
Status	Valid

6.1.14 Pengujian Validasi Membuat Pengaduan (PLM_F_1400)

Tabel 6.28 Skenario Uji Membuat Pengaduan

Skenario Uji Membuat Pengaduan	
Skenario	1. *Login* ke aplikasi sebagai pengadu 2. *User* memilih salah satu kategori pengaduan 3. *User* mengisi semua kolom 4. Klik tombol “submit”
Expected Result	*User* berhasil mengirimkan pengaduan
Result	*User* berhasil mengirimkan pengaduan
Status	Valid

Tabel 6.29 Skenario Uji Membuat Pengaduan Alternatif 1

Skenario Uji Membuat Pengaduan Alternatif 1 (Kolom Ada yang Kosong)	
Skenario	1. *Login* ke aplikasi sebagai pengadu 2. *User* memilih salah satu kategori pengaduan 3. *User* mengisi mengosongkan salah satu kolom atau lebih 4. Klik tombol “submit”
Expected Result	*User* tidak bisa mengirimkan pengaduan dan muncul notifikasi “semua kolom harus diisi”
Result	*User* tidak bisa mengirimkan pengaduan dan muncul notifikasi “semua kolom harus diisi”
Status	Valid

6.1.15 Pengujian Validasi Membaca Notifikasi Status (PLM_F_1500)

Tabel 6.30 Skenario Uji Membaca Notifikasi Status

Skenario Uji Membaca Notifikasi Status	
Skenario	1. *Login* ke aplikasi sebagai pengadu 2. *User* menekan notifikasi status yang diterima
Expected Result	*User* berhasil melihat status pengaduan pada halaman detail pengaduan
Result	*User* berhasil melihat status pengaduan pada halaman detail pengaduan
Status	Valid

6.1.16 Pengujian Validasi Memberi *Rating* (PLM_F_1600)

Tabel 6.31 Skenario Uji Memberi *Rating*

Skenario Uji Memberi *Rating*	
Skenario	1. *Login* ke aplikasi sebagai pengadu 2. *User* menekan jumlah *rating* dan menekan tombol kirim pada notifikasi rating yang sudah selesai
Expected Result	*User* berhasil memberi *rating* pegawai yang menangani keluhannya
Result	*User* berhasil memberi *rating* pegawai yang menangani keluhannya
Status	Valid

6.1.17 Pengujian Validasi Melihat Riwayat Kerusakan (PLM_F_1700)

Tabel 6.32 Skenario Uji Melihat Riwayat Kerusakan

Skenario Uji Melihat Riwayat Kerusakan	
Skenario	1. *Login* ke aplikasi 2. *User* menekan menu daftar pengaduan 3. *User* menekan salah satu pengaduan barang 4. *User* menekan nomor inventaris
Expected Result	*User* berhasil melihat daftar riwayat kerusakan
Result	*User* berhasil melihat daftar riwayat kerusakan
Status	Valid

6.1.18 Pengujian Validasi Melihat Riwayat *Maintenance* (PLM_F_1800)

Tabel 6.33 Skenario Uji Melihat Riwayat *Maintenance*

Skenario Uji Membaca Notifikasi *Rating*	
Skenario	1. *Login* ke aplikasi 2. *User* menekan menu jadwal *maintenance* 3. *User* menekan salah satu tanggal 4. *User* menekan nomor inventaris
Expected Result	*User* berhasil melihat daftar riwayat *maintenance*
Result	*User* berhasil melihat daftar riwayat *maintenance*
Status	Valid

6.2 Pengujian *White Box*

Dalam melakukan pengujian *white box*, pengujian unit yang digunakan adalah metode *basis-path testing.* Metode ini meliputi pemodelan algoritme dalam suatu *flow graph*, menghitung *cyclomatic complexity,* menentukan *independent path,* dan memberi kasus uji setiap basis set yang ditentukan berdasarkan *independent path.*

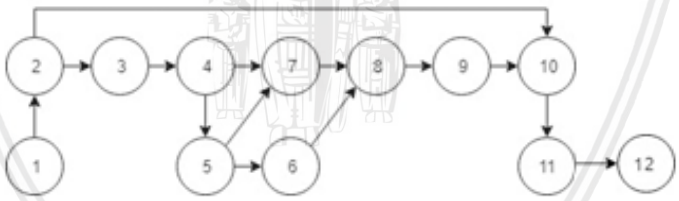
6.2.1 Pengujian *Method* Mengirim Notifikasi

Pengujian unit *method* mengirim notifikasi digunakan untuk mengecek keberhasilan notifikasi yang diterima oleh pegawai ketika data pengaduan telah dikirimkan oleh pengadu.

Tabel 6.34 *Pseudocode Method* Mengirim Notifikasi

```
1   Mulai method sendNotification
2     TRY untuk mengecek koneksi HTTP response
3     Otorisasi tipe data pada json
4         IF koneksi terpenuhi
5         AND koneksi kurang dari internal error
6             Mengirim json respon
7         ELSE jika koneksi gagal
8         END IF
9     Mengirim json response
10    CATCH untuk kondisi error
11    END TRY
12  Selesai
```

Berdasarkan node-node yang telah didefinisikan pada Tabel 6.2 diatas maka *flow graph* yang dihasilkan dapat pada Gambar 6.1 berikut.



Gambar 6.1 *Flow Graph Method* Mengirim Notifikasi

Berdasarkan *flow graph* diatas, maka dapat diketahui bahwa jumlah node yaitu 12 dan jumlah *edge* yaitu 14. Untuk menghitung *cyclomatic complexity* maka digunakan persamaan V(G) = E – N + 2. Sehingga didapatkan hasil berikut.

$$V(G) = 14 - 12 + 2$$
$$= 2 + 2$$
$$= 4$$

Berdasarkan hasil perhitungan *cyclomatic complexity,* maka didapatkan empat jalur basis *path*, yaitu sebagai berikut.

Jalur 1 : 1-2-10-11 -12

Jalur 2 : 1-2-3-4-7-8-9-10-11 -12

Jalur 3 : 1-2-3-4-5-7-8-9-10-11-12

Jalur 4 : 1-2-3-4-5-6-8-9-10-11

Berdasarkan jalur yang didapatkan maka tahap selanjutnya adalah pengujian tiap jalur yang dapat dilihat pada Tabel 6.3.

Tabel 6.35 Kasus Uji *Method* Mengirim Notifikasi

No	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Ketika melakukan pengecekan HttpURLConnection, dan terjadi eror	Sistem menampilkan pesan "Gagal mengirimkan notifikasi"	Sistem menampilkan pesan "Gagal mengirimkan notifikasi"
2	Ketika melakukan pengecekan HttpURLConnection, dan koneksi tersambung	Sistem berhasil mengirimkan pengaduan	Sistem berhasil mengirimkan pengaduan
3	Ketika melakukan pengecekan HttpURLConnection, dan koneksi eror	Sistem gagal mengirimkan pengaduan	Sistem gagal mengirimkan pengaduan
4	Ketika melakukan pengecekan HttpURLConnection, dan koneksi eror	Sistem gagal mengirimkan pengaduan dan menampilkan pesan "Gagal mengirimkan notifikasi"	Sistem gagal mengirimkan pengaduan dan menampilkan pesan "Gagal mengirimkan notifikasi"

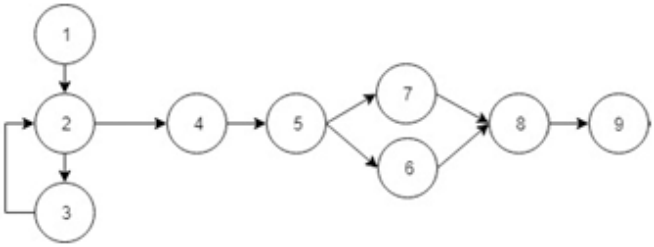
6.2.2 Pengujian *Method* Melihat Daftar Pengaduan

Pengujian unit *method* melihat daftar pengaduan digunakan untuk melihat daftar pengaduan yang telah dikirimkan pengadu dan masuk kedalam *database*.

Tabel 6.36 *Pseudocode* Komponen *Method* Melihat Daftar Pengaduan

```
1  Mulai method onDataChange
2     FOR untuk mengambil data pada child
3     Inisialisasi data yang diambil
4     END FOR
5     IF jika data pengaduan kosong
6     Menampilkan data kosong
7     ELSE jika data ada
8     END IF
9  Selesai
```

Berdasarkan node-node yang telah didefinisikan pada Tabel 6.4 maka *flow graph* yang dihasilkan dapat dilihat pada Gambar 6.2 berikut.



Gambar 6.2 *Flow Graph Method* Melihat Daftar Pengaduan

Berdasarkan *flow graph* diatas, maka dapat diketahui bahwa jumlah node yaitu 9 dan jumlah *edge* yaitu 10. Untuk menghitung *cyclomatic complexity* maka digunakan persamaan V(G) = E – N + 2. Sehingga didapatkan hasil berikut.

$$V(G) = 10 - 9 + 2$$
$$= 1 + 2$$
$$= 3$$

Berdasarkan hasil perhitungan *cyclomatic complexity,* maka didapatkan empat jalur basis *path*, yaitu sebagai berikut.

Jalur 1 : 1-2-4-5-7-8 -9

Jalur 2 : 1-2-4-5-6-8-9

Jalur 3 : 1-2-3-2-4-5-6-8-9

Berdasarkan jalur yang didapatkan maka tahap selanjutnya adalah pengujian tiap jalur yang dapat dilihat pada Tabel 6.3.

Tabel 6.37 Kasus Uji *Method* Melihat Daftar Pengaduan

No	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Ketika semua data pada variabel data sudah diulang dan dan pengaduanList tidak *empty*	Sistem menampilkan list data pengaduan	Sistem menampilkan list data pengaduan
2	Ketika semua data pada variabel data sudah diulang dan dan pengaduanList bernilai *empty*	Sistem tidak akan menampilkan list data pengaduan	Sistem tidak akan menampilkan list data pengaduan
3	Ketika terdapat data pada variabel data yang memenuhi syarat dan dapat diulang dan pengaduanList bernilai *empty*	Sistem tidak akan menampilkan list data pengaduan	Sistem tidak akan menampilkan list data pengaduan

6.2.3 Pengujian *Method* Mengubah Status *Pengaduan*

Pengujian unit *method* mengubah status pengaduan digunakan megubah status pengaduan oleh pegawai yang menangani pengaduan tersebut.

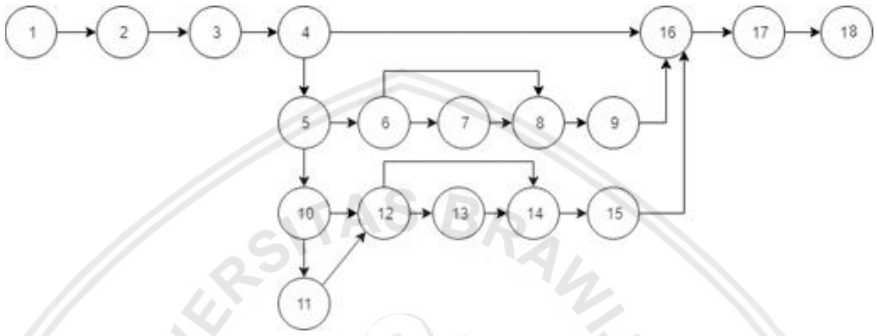
Tabel 6.38 *Pseudocode Method* Mengubah Status Pengaduan

```
1   Mulai method ubahStatus
2     IF role pegawai
3         IF status belum diterima
4         Mengubah status diterima pada database
5         ELSE IF status diterima
6             IF role pegawai
7             Menampilkan tombol perbaiki sendiri dan sedang di
              vendor
8             END IF
9             Mengubah status sedang diproses dan sedang di
              vendor pada database
```

Tabel 6.6 *Pseudocode Method* Mengubah Status Pengaduan (Lanjutan)

```
10        ELSE IF status sedang diproses
11        OR status sedang di vendor
12            IF role pegawai
13            END IF
14            Menampilkan tombol selesai
15            END IF
16     END IF
17  Selesai
```

Berdasarkan node-node yang telah didefinisikan pada Tabel 6.6 diatas maka *flow graph* yang dihasilkan yaitu pada Gambar 6.3 berikut.



Gambar 6.3 *Flow Graph Method* Mengubah Status Pengaduan

Berdasarkan *flow graph* diatas, maka dapat diketahui bahwa jumlah node yaitu 18 dan jumlah *edge* yaitu 22. Untuk menghitung *cyclomatic complexity* maka digunakan persamaan V(G) = E – N + 2. Sehingga didapatkan hasil berikut.

$$V(G) = 22 - 18 + 2$$
$$= 4 + 2$$
$$= 6$$

Berdasarkan hasil perhitungan *cyclomatic complexity,* maka didapatkan lima jalur basis *path*, yaitu sebagai berikut.

Jalur 1 : 1-2-3-4-15-16-17-18

Jalur 2 : 1-2-3-4-5-6-8-9-15-16-17-18

Jalur 3 : 1-2-3-4-5-6-7-8-9-15-16-17-18

Jalur 4 : 1-2-3-4-5-10-12-13-14-15-16-17-18

Jalur 5 : 1-2-3-4-5-10-11-12-14-15-16-17-18

Jalur 6 : 1-2-3-4-5-10-11-12-13-14-15-16-17-18

Berdasarkan jalur yang didapatkan maka tahap selanjutnya adalah pengujian tiap jalur yang dapat dilihat pada Tabel 6.7.

Tabel 6.39 Kasus Uji *Method* Mengubah Status Pengaduan

No	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
1	Ketika variabel *role* bernilai “pegawai” dan status bernilai “belum diterima”	Sistem menampilkan tombol terima	Sistem menampilkan tombol terima

Tabel 6.12 Kasus Uji *Method* Mengubah Status Pengaduan (Lanjutan)

No	Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan
2	Ketika variabel *role* bernilai "pegawai", status bernilai "diterima", dan id pegawai tidak sama dengan *current*	Sistem tidak akan menampilkan tombol perbaiki sendiri dan tombol serahkan ke vendor	Sistem tidak akan menampilkan tombol perbaiki sendiri dan tombol serahkan ke vendor
3	Ketika variabel *role* bernilai "pegawai", status bernilai "diterima", dan id pegawai bernilai sama dengan *current*	Sistem menampilkan tombol perbaiki sendiri dan serahkan ke vendor	Sistem menampilkan tombol perbaiki sendiri dan serahkan ke vendor
4	Ketika variabel *role* bernilai "pegawai", status bernilai "sedang diproses", dan id pegawai bernilai sama dengan *current*	Sistem menampilkan tombol selesai	Sistem menampilkan tombol selesai
5	Ketika variabel *role* bernilai "pegawai", status bernilai "sedang di vendor", dan id pegawai tidak sama dengan *current*	Sistem tidak akan menampilkan tombol selesai	Sistem tidak akan menampilkan tombol selesai
6	Ketika variabel *role* bernilai "pegawai", status bernilai "sedang di vendor", dan id pegawai bernilai sama dengan *current*	Sistem menampilkan tombol selesai	Sistem menampilkan tombol selesai

6.3 Pengujian *Usability*

Pengujian *usability* dalam penelitian ini dilakukan untuk mengukur tingkat pengalaman interaksi antara pengguna dengan aplikasi yang telah dibuat. Sehingga dapat diidentifikasi permasalahan pada interaksi tersebut yang harus diperbaiki dan mengukur tingkat kepuasan pengguna untuk meningkatkan kualitas aplikasi. Sesuai dengan pernyataan Nielsen bahwa responden yang paling baik tidak lebih dari lima orang, maka pada penelitian ini perwakilan pengguna berjumlah lima orang yang mewakili *role* Kasubag, pegawai bagian umum dan perlengkapan, serta pengadu yaitu civitas FILKOM. Berikut lima perwakilan pengguna yang terlibat.

P1 : Ferix Panji Andrianto, S.ST (Pegawai Sub Bagian Umum dan Perlengkapan)

P2 : Daud Siswo Sarjono (Pegawai Sub Bagian Umum dan Perlengkapan)

P3 : Ofi Eka Noviyanti (Civitas FILKOM)

P4 : Septian Dwi Cahyo (Civitas FILKOM)

P5 : Abd. Jahiduddin (Civitas FILKOM)

Pengujian *usability* kepada lima responden tersebut dilakukan dengan metode skenario tugas dan kuesioner USE.

6.3.1 Pengujian *Usability* Skenario Tugas

Pengujian skenario tugas dilakukan dengan memberikan *task-task* tertentu kepada pengguna agar pengguna melakukan *task* tersebut sesuai dengan pengetahuannya. Metode ini memungkinkan peneliti untuk mengidentifikasi keberhasilan interaksi pengguna dengan aplikasi. Skenario tugas yang diberikan berbeda-beda sesuai dengan *role* tiap pengguna yang dapat dilihat pada Tabel 6.8.

Tabel 6.40 Skenario Tugas

No.	Tugas	Kasubag	Pegawai	Pengadu
1	*Login*	T1	T1	T1
2	Melihat daftar pengaduan	T2	T2	T2
3	Melihat detail pengaduan	T3	T3	T3
4	Melihat statistik pengaduan	T4	T4	T4
5	Membuat laporan pengaduan	T5		
6	Membaca notifikasi pengaduan		T5	
7	Melihat pengaduan yang ditangani		T6	
8	Mengubah status pengaduan		T7	
9	Melihat jadwal *maintenance*		T8	
10	Mengisi *form* jadwal *maintenance*		T9	
11	Membaca notifikasi *maintenance*		T10	
12	Membaca notifikasi *rating*		T11	
13	Membuat pengaduan			T5
14	Membaca notifikasi status pengaduan			T6
15	Memberi *rating*			T7
16	Melihat riwayat kerusakan	T6	T12	T8
17	Melihat riwayat *maintenance*		T13	
18	*Logout*	T7	T14	T9

Setelah dilakukan pengujian skenario tugas terhadap lima perwakilan pengguna, maka didapatkan hasil yang dapat dilihat pada Tabel 6.9 sampai dengan 6.11.

Tabel 6.41 Hasil Skenario Tugas Kasubag

Nama	Hasil Pengujian							Ket
	T1	T2	T3	T4	T5	T6	T7	
P1	√	√	√	√	√	√	√	-
Hasil	100%	100%	100%	100%	100%	100%	100%	

Tabel 6.42 Hasil Skenario Tugas Pegawai

Nama	Hasil Pengujian													Ket
	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12	T13	T14	
P2	√	√	√	√	√	√	√	√	√	√	√	√	√	√	-
Hasil	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	100 %	

Tabel 6.43 Hasil Skenario Tugas Pengadu

Nama	Hasil Pengujian									Ket
	T1	T2	T3	T4	T5	T6	T7	T8	T9	
P3	√	√	√	√	√	√	√	√	√	-
P4	√	√	√	√	√	√	√	√	√	-
P5	√	√	√	√	√	√	√	√	√	-
Hasil	100%	100%	100%	100%	100%	100%	100%	100%	100%	

Tabel 6.11 sampai dengan 6.13 menunjukkan sukses atau gagalnya suatu tugas dan alasan mengapa tugas tersebut tidak dapat dijalankan. Pada tabel pengujian skenario tugas pengguna aplikasi pengaduan layanan dan *maintenance* terdapat simbol (√) yang berarti dapat dijalankan dan simbol (x) berarti tidak dapat dijalankan sama sekali. Berdasarkan hasil pengujian skenario tugas ini maka didapatkan rata-rata 100%. Sehingga dapat disimpulkan bahwa pengguna setuju bahwa aplikasi pengaduan layanan dan *maintenance* dapat digunakan dengan baik dan mudah.

6.3.2 Pengujian *Usability* Kuesioner USE

Pengujian ini dilakukan setelah lima perwakilan pengguna yang telah ditetapkan sebelumnya mencoba untuk mengoperasikan aplikasi pengaduan layanan dan *maintenance.* Kuesioner yang diberikan mencakup aspek *usefulnes, satisfaction,* dan *ease of use.* Hasil pengujian dapat dilihat pada Tabel 6.12.

Tabel 6.44 Hasil Kuesioner USE

USEFULNESS		***STS***	***TS***	***N***	***S***	***SS***
1	Aplikasi PLM dapat membantu melaporkan kerusakan dan keluhan di area FILKOM					5
2	Aplikasi PLM dapat membantu mengingatkan jadwal *maintenance*					5
3	Aplikasi PLM dapat membantu membuat laporan pengaduan, *maintenance,* dan *rating*					5
4	Aplikasi PLM dapat membantu melihat statistik pengaduan yang telah dilaporkan di area FILKOM				3	2
5	Aplikasi PLM dapat membantu penggunanya untuk melihat riwayat kerusakan suatu barang					5
SATISFACTION						
6	Aplikasi PLM bekerja sesuai keinginan saya				2	3
7	Saya harus memiliki aplikasi PLM					5
8	Saya puas dengan aplikasi PLM					5
9	Saya senang selama mengoperasikan aplikasi PLM					5
EASE OF USE						
10	Aplikasi PLM dapat dengan mudah dimengerti					5

Tabel 6.12 Hasil Kuesioner USE (lanjutan)

EASE OF USE		***STS***	***TS***	***N***	***S***	***SS***
11	Aplikasi PLM memiliki antarmuka yang mudah dipahami sesuai kegunaannya					5
12	Aplikasi PLM dapat dengan mudah digunakan tanpa menggunakan petunjuk penggunaan				3	2
13	Aplikasi PLM dapat diakses setiap saat				1	4

Berdasarkan hasil kuesioner USE maka dilakukan penghitungan persentase tiap aspek yang hasilnya dapat dilihat pada Tabel 6.13.

Tabel 6.45 Penghitungan Kuesioner USE

USEFULNESS	**Skor**	**Persentase**
1	25	100%
2	25	100%
3	25	100%
4	22	88%
5	25	100%
Rata-rata		97,6%
SATISFACTION		
6	23	92%
7	25	100%
8	25	100%
9	25	100%
Rata-rata		98%
EASE OF USE		
10	25	100%
11	25	100%
12	22	88%
13	24	96%
Rata-rata		96%
Total rata-rata		97,2%

Berdasarkan hasil penghitungan persentase kuesioner USE pada Tabel 6.16 maka didapatkan total rata-rata 97,2%. Sehingga dapat disimpulkan bahwa aplikasi memiliki tingkat *usefulness, satisfaction,* dan *ease of use* yang sangat baik.

6.4 Pengujian *Compatibility*

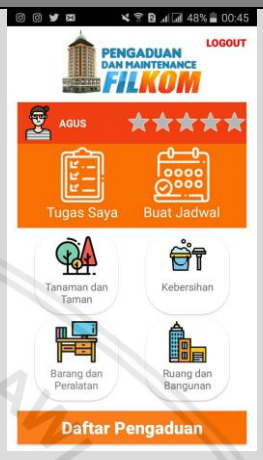
Pengujian *compatibility* pada penelitian ini dilakukan dengan melakukan instalasi aplikasi di berbagai *smartphone android* dengan dimensi, resolusi, sistem operasi, dan API yang berbeda-beda. Aspek yang diuji adalah keberhasilan instalasi dan *widget.* Langkah-langkah pengujian *compatibility* dalam penelitian ini yaitu:

1. Melakukan instalasi pada tiga tipe *smartphone android* dengan dimensi, resolusi, sistem operasi, dan API yang berbeda-beda yaitu Samsung SM-J320G, ASUS-Z017DB, dan Xiaomi Mi A2 Lite.

2. Melakukan penghitungan persentase *compatibility* dengan parameter keberhasilan instalasi aplikasi pada *smartphone* dan keberhasilan dalam menampilkan *widget* sesuai dengan Rumus 2.1.

Hasil dari pengujian *compatibility* dapat dilihat pada Tabel 6.14.

Tabel 6.46 Pengujian *Compatibility*

No	Perangkat	Screenshot	Status
1	Samsung SM-J320G a) Dimensi: 142.3 x 71 x 7.9 mm (5.60 x 2.80 x 0.31 in) b) Resolusi layar: 720 x 1280 *pixels*, 16:9 *ratio* (~294 ppi *density*) c) Sistem operasi: *Android* 5.1.1 (lollipop) d) API level: 22		Valid
2	ASUS-Z017DB a) Dimensi: 146.9 x 74 x 7.7 mm (5.78 x 2.91 x 0.30 in) b) Resolusi layar: 1080 x 1920 *pixels*, 16:9 *ratio* (~424 ppi *density*) c) Sistem operasi: *Android* 8.0.0 (oreo) d) API level: 26		Valid
3	Xiaomi Mi A2 Lite a) Dimensi: 149.33 x 71.68 x 8.75 mm (5.88 x 2.82 x 0.34 in) b) Resolusi layar: 1080 x 2280 *pixels*, 19:9 *ratio* (~432 ppi *density)* c) Sistem operasi: *Android* 9 (pie) d) API level: 28		Valid

Berdasarkan Tabel 6.14, terdapat 3 macam perangkat yang digunakan untuk pengujian. Hasil dari pengujian tersebut kemudian dihitung dengan persentase *compatibility*. Penghitungan persentase dapat dilihat pada Tabel 6.12.

Tabel 6.47 Hasil Pengujian *Compatibility*

Pengujian	Nilai Maksimum	Berjalan	Gagal
Menjalankan aplikasi pada perangkat *Android*	3	3	0
Memasang *widget*	3	3	0
Nilai total		3	0

$$Persentase\ Kelayakan = \frac{3}{3} x100\% = 100\%$$

Berdasarkan hasil pengujian pada Tabel 6.11 dengan hasil total 3 maka merujuk pada rumus 2.1, persentase aspek *compatibility* dihasilkan 100% atau dalam kategori sangat baik dalam persentase kelayakan Guttman. Sehingga dapat disimpulkan bahwa aplikasi pengaduan layanan dan *maintenance* dapat berjalan di berbagai tipe *android.*

BAB 7 PENUTUP

Bagian penutup berisi kesimpulan dan saran pada penelitian ini. Kesimpulan berisi hasil dari rumusan masalah yang telah didefinisikan sebelumnya. Sedangkan saran digunakan sebagai acuan untuk pengembangan sistem yang lebih lanjut.

7.1 Kesimpulan

Berdasarkan rekayasa kebutuhan, perancangan, implementasi, dan pengujian di penelitian ini, maka dapat disimpulkan antara lain:

1. Berdasarkan hasil rekayasa kebutuhan pada pengembangan aplikasi pengaduan layanan dan *maintenance,* didapatkan 2 proses bisnis *to-be* yang sudah mencakup 5 proses bisnis *as-is.* Proses bisnis *to-be* terdiri dari proses pemeliharaan dan proses perbaikan. Berdasarkan hasil identifikasi proses bisnis *to-be* maka didapatkan 3 aktor yaitu Kasubag Umum dan Perlengkapan, Pegawai Umum dan Perlengkapan, dan pengadu. Kemudian didapatkan 18 kebutuhan fungsional dan 2 kebutuhan non fungsional. Hasil tersebut kemudian dimodelkan menjadi diagram *use case* dan skenario *use case.*
2. Berdasarkan hasil perancangan pada pengembangan aplikasi pengaduan layanan dan maintenance, maka didapatkan perancangan berupa arsitektur aplikasi, *sequence diagram, class diagram,* perancangan komponen, perancangan data, dan perancangan antarmuka. Perancangan *sequence diagram* terdiri dari *sequence diagram* melihat daftar pengaduan, membuat laporan, dan mengubah status pengaduan. Perancangan komponen terdiri dari perancangan komponen *method* Mengirimkan Notifikasi, *method* Melihat Daftar Pengaduan, dan *method* Mengubah Status Pengaduan. Kemudian perancangan antarmuka terdiri dari *form* pengaduan tanaman, halaman daftar pengaduan, dan halaman detail pengaduan.
3. Berdasarkan hasil implementasi pengembangan aplikasi pengaduan layanan dan *maintenance,* dapat diidentifikasi spesifikasi sistem yang terdiri dari spesifikasi perangkat keras dan spesifikasi perangkat lunak. Pada hasil implementasi didapatkan implementasi kode program, implementasi data, dan implementasi antarmuka aplikasi. Implementasi program dan implementasi antarmuka dilakukan menggunakan Android Studio 3.3 dengan bahasa pemrograman java. Implementasi data dilakukan dengan menggunakan Firebase 11.0.4 yang menyediakan fitur *realtime database* untuk melakukan sinkronisasi data antar klien secara *realtime.*
4. Berdasarkan pengujian pengembangan aplikasi pengaduan layanan dan *maintenance*, maka didapatkan hasil pengujian *black box,* pengujian *white box*, pengujian *usability* dan pengujian *compatibility.* Pengujian *black box* dengan metode *validation testing* menghasilkan nilai valid terhadap 18 kebutuhan fungsional. Pengujian unit *white box* dengan metode *McCabe's basic path testing* menghasilkan angka *cyclomatic complexity* dibawah 10 yang berarti rancangan algoritme dari fungsi-fungsi yang diuji sudah baik dan seluruh fungsi

bernilai valid. Kemudian pengujian non fungsional *usability* dengan metode skenario tugas menghasilkan persentase 100% sehingga pengguna dapat berinteraksi dengan baik terhadap aplikasi. Untuk pengujian *usability* dengan metode kuesioner USE mengasilkan persentase 97,2% sehingga pengguna setuju bahwa aplikasi bermanfaat, memberi kepuasan, dan mudah digunakan. Secara keseluruhan hasil pengujian *usability* menunjukkan pengguna setuju bahwa aplikasi dapat digunakan dan diterapkan pada layanan pengaduan dan *maintenance* FILKOM. Kemudian pengujian non fungsional *compatibility* menghasilkan nilai 100% sehingga dapat disimpulkan bahwa aplikasi pengaduan layanan dan *maintenance* dapat berjalan di berbagai tipe *android.*

7.2 Saran

Saran untuk pengembangan aplikasi pengaduan layanan dan *maintenance* adalah sebagai berikut.

1. Aplikasi dapat mengirimkan notifikasi kepada pegawai ketika pengadu berhasil mengirimkan pengaduan dan ketika pegawai mengubah status pengaduan maka pengadu akan mendapatkan notifikasi. Namun aplikasi pengaduan layanan dan *maintenance* belum bisa mengirimkan notifikasi kepada pegawai ketika status pengaduan belum diubah dalam jangka waktu tertentu. Sehingga perlu dilakukan penambahan fitur notifikasi kepada pegawai untuk segera mengubah status pengaduan ketika status belum diubah dalam jangka waktu tertentu sehingga pegawai dapat melaksanakan tugasnya dalam waktu yang lebih cepat.
2. Aplikasi dapat menampilkan pengaduan berdasarkan statusnya dan bagi pegawai serta kasubag juga dapat menampilkan pengaduan berdasarkan kategorinya. Namun aplikasi pengaduan layanan dan *maintenance* masih belum dapat melakukan pencarian dengan memasukkan kata kunci untuk mencari pengaduan dan *maintenance* yang diinginkan sehingga perlu ditambahkan fitur tersebut agar pencarian lebih cepat dan mudah.

DAFTAR REFERENSI

Arzt, S., Huber, S., Rasthofer, S. & Bodden, E., 2014. *Denial-of-App Attack : Inhibiting the Installation of Android Apps on Stock Phones.* SPSM.

Dwiyanto, A., 2018. *Manajemen Pelayanan Publik: Peduli Inklusif Dan Kolaborasi.* Yogyakarta: UGM PRESS.

e-complaint.ub.ac.id, 2012. [Online] Available at: https://e-complaint.ub.ac.id/ [Diakses 19 September 2018].

EMS, T., 2015. *Pemrograman Android dalam Sehari.* Jakarta: Elex Media Komputindo.

filkom.ub.ac.id, 2017. *Profil Fakultas Ilmu Komputer Universitas Brawijaya.* [Online] Available at: https://filkom.ub.ac.id [Diakses 20 September 2018].

Firdausy, D. R., Wicaksono, S. A. & Pradana, F., 2018. Pengembangan Sistem Informasi Manajemen Pelaporan Sarana dan Prasarana Studi pada Fakultas Ilmu Komputer Universitas Brawijaya. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer ,* pp. 5365-5374.

firebase.google.com, 2010. *Firebase.* [Online] Available at: https://firebase.google.com/docs/ [Diakses 10 Januari 2019].

Golhar, R. V., Vyawahare, P. A., Borghare, P. H. & Manusmare, A., 2016. Design And Implementation Of Android Base Mobile App For An Institute. *International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT).*

Guritno, S. S. R. U., 2011. *Theory and apllication of IT.* Yogyakarta: Penerbit Andi.

Haviluddin, 2011. Memahami Penggunaan UML (Unified Modelling Language). *Jurnal Informatika Mulawarman, Vol.6 No. 1.*

Kurniawan, T. A., 2018. Pemodelan Use Case (UML): Evaluasi Terhadap Beberapa Kesalahan dalam Praktik. *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK),* pp. 77-86.

Larman, C., 2005. *Applying UML and Patterns 3rd Edition.* s.l.:NJ: Prentice Hall.

Lazar, J., Feng, J. H. & Hochheiser, H., 2017. *Research Methods in Human-Computer Interaction.* 2nd penyunt. Cambridge: Morgan Kaufmann.

Likert, R., 1932. Technique for the measurement of attitudes. Dalam: *Archives of Psychology.* s.l.:s.n., pp. 1-55.

Loina, P. A., 2010. *Hubungan Masyarakat : Membina Hubungan Baik Dengan Publik.* Bandung: CV. Lalolo.

Lutfiyana, N., 2015. Rancang Bangun Sistem Informasi Layanan Pengaduan Nasabah Kartu Kredit Berbasis Web. *Seminar Nasional Ilmu Pengetahuan dan Teknologi (SNIPTEK) 2015 ISBN: 978-602-72850-6-4.*

Mishra, A. & Dubey, 2013. *A Comparative Study of Different Software Development Life Cycle Models in Different Scenarios.* s.l.:International Journal of Advance Research in Computer Science and Management Studies.

Moenir, H., 2014. *Manajemen Pelayanan Umum di Indonesia.* Jakarta: Bumi Aksara.

Nielsen, J., 2000. *Why You Only Need to Test with 5 Users.* [Online] Available at: https://www.nngroup.com/articles/whyyou-only-needto-test-with-5-users/ [Diakses 5 Januari 2019].

Nielsen, J., 2014. *Turn User Goals into Task.* [Online] Available at: https://www.nngroup.com/articles/taskscenarios-usability-testing/ [Diakses 5 Januari 2019].

Nugroho, A., 2010. *Rekayasa Perangkat Lunak Berorientasi Objek dengan Metode USDP.* Yogyakarta: ANDI.

onesignal.com, 2014. *Product Overview.* [Online] Available at: https://documentation.onesignal.com/docs [Diakses 10 Januari 2019].

Pressman, R., 2002. *Rekayasa Perangkat Lunak: Pendekatan Praktisi(Buku Dua).* Yogyakarta: Penerbit Andi.

Pressman, R., 2010. *Software Engineering :a practitioner's approach.* New York: s.n.

Rizky, S., 2011. *Konsep Dasar Rekayasa Perangkat Lunak.* Jakarta: Prestasi Pustakaraya.

Sulila, I., 2015. *Implementasi Dimensi Layanan Publik dalam Konteks Otonomi Daerah.* Yogyakarta: Deepublish.

ub.ac.id, 2018. *Sejarah Universitas Brawijaya.* [Online] Available at: https://ub.ac.id/id/about/history/ [Diakses 2018].

w3c.org, 2002. *Web Secrvices Architecture.* [Online] Available at: http://www.w3c.org [Diakses 20 September 2018].

Zulfikar, A. R. & Supianto, A. A., 2018. Rancang Bangun Aplikasi Antrian Poliklinik Berbasis Mobile. *Jurnal Teknologi Informasi dan Ilmu Komputer Vol. 5, No.3, Agustus 2018, hlm. 361-370 p-ISSN: 2355-77699 Akreditasi KEMENRISTEKDIKTI, No. 51/E/KPT/2017.*