TUGAS AKHIR – KI141502

PENGEMBANGAN APLIKASI MODULAR SUARA WARGA PEMERINTAH KOTA KEDIRI BERBASIS ANDROID PADA PERANGKAT KOMUNIKASI BERGERAK

FAHMI ARMAND RACHMAN
NRP 5111 100 170

Dosen Pembimbing
Dr.tech. Ir. R. V. Hari Ginardi, M.Sc.
Dwi Sunaryono, S.Kom., M.Kom.

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA 2015

FINAL PROJECT – KI141502

DEVELOPMENT OF MODULAR APPLICATION FOR SUARA WARGA PEMERINTAH KOTA KEDIRI BASED ON ANDROID MOBILE DEVICE

FAHMI ARMAND RACHMAN
NRP 5111 100 170

Advisor
Dr.tech. Ir. R. V. Hari Ginardi, M.Sc.
Dwi Sunaryono, S.Kom., M.Kom.

INFORMATICS DEPARTEMENT
FACULTY OF INFORMATION TECHNOLOGY
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA 2015

LEMBAR PENGESAHAN

PENGEMBANGAN APLIKASI MODULAR SUARA WARGA PEMERINTAH KOTA KEDIRI BERBASIS ANDROID PADA PERANGKAT KOMUNIKASI BERGERAK

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Algoritma dan Pemrograman
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh:
FAHMI ARMAND RACHMAN
NRP. 5111 100 170

Disetujui oleh Pembimbing Tugas Akhir:

1. Dr.tech. Ir. R. V. Hari Ginardi, M.Sc. .....
NIP: 19650518 199203 1 003 (Pembimbing 1)

2. Dwi Sunaryono, S.Kom., M.Kom. .....
NIP: 19720528 199702 1 001 (Pembimbing 2)

SURABAYA
JUNI, 2015

LEMBAR PENGESAHAN

PENGEMBANGAN APLIKASI MODULAR SUARA WARGA PEMERINTAH KOTA KEDIRI BERBASIS ANDROID PADA PERANGKAT KOMUNIKASI BERGERAK

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Algoritma dan Pemrograman
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh:
FAHMI ARMAND RACHMAN
NRP. 5111 100 170

Disetujui oleh Pembimbing Tugas Akhir:

1. Dr.tech. Ir. R. V. Hari Ginardi, M.Sc. .....
   NIP: 19650518 199203 1 003 (Pembimbing 1)

2. Dwi Sunaryono, S.Kom., M.Kom. .....
   NIP: 19720528 199702 1 001 (Pembimbing 2)

SURABAYA
JUNI, 2015

PENGEMBANGAN APLIKASI MODULAR SUARA WARGA PEMERINTAH KOTA KEDIRI BERBASIS ANDROID PADA PERANGKAT KOMUNIKASI BERGERAK

Nama Mahasiswa : Fahmi Armand Rachman
NRP : 5111 100 170
Jurusan : Teknik Informatika, FTIF-ITS
Dosen Pembimbing 1 : Dr.tech. Ir. R. V. Hari Ginardi, M.Sc.
Dosen Pembimbing 2 : Dwi Sunaryono, S.Kom., M.Kom.

ABSTRAK

Perkembangan teknologi di bidang perangkat bergerak terus berkembang dari tahun ke tahun. Hal ini dapat dilihat sebagai sebuah peluang untuk meningkatkan layanan masyarakat oleh Pemkot Kediri. Sehingga Pemkot Kediri menginisialisasi untuk membuat sebuah layanan yang bisa menampung aduan, saran, dan informasi dari masyarakat Kota Kediri.

Tugas Akhir ini membangun suatu aplikasi yang bersifat modular untuk menanggapi setiap aduan, saran dan informasi yang diberikan oleh masyarakat. Dengan menerapkan konsep modularitas, berbagai modul dimungkinkan dapat ditambahkan ke dalam aplikasi ini dengan mengimplementasikan interface-nya. Aplikasi ini dibangun dengan menggunakan Eclipse Juno 4.2.1 dan SDK Android. Dengan dikembangkannya aplikasi ini diharapkan bisa membantu Pemerintah Kota Kediri dalam melayani aduan, saran, dan informasi masyarakat dimanapun dan kapanpun.

Aplikasi yang telah dikembangkan diuji berdasarkan modularitas dan fungsionalitasnya. Pengujian dilakukan melalui skenario dengan metode kotak hitam yang mencerminkan fitur aplikasi. Hasil pengujian menunjukkan bahwa aplikasi bisa berjalan dengan baik dari segi modularitas dan fungsionalitasnya.

Kata kunci**: Android, aduan, modularitas, perangkat bergerak

DEVELOPMENT OF MODULAR APPLICATION FOR SUARA WARGA PEMERINTAH KOTA KEDIRI BASED ON ANDROID MOBILE DEVICE

Student's Name	**: Fahmi Armand R.**
Student's ID	**: 5111 100 170**
Department	**: Informatics Departement, FTIF-ITS**
First Advisor	**: Dr.tech. Ir. R. V. Hari Ginardi, M.Sc.**
Second Advisor	**: Dwi Sunaryono, S.Kom., M.Kom.**

Abstract

Development of technology in mobile devices keep growing over the years. This can be seen as an opportunity for improve government service for people by the government of Kediri. So that the government of kediri initiate to make a service that can accomodate complaint, suggestion, and information from residents of Kediri.

This final project is to build a modular application for accommodate complaint, suggestion, and information from the people. By applying modularity, various of modules can be added to this application by implementing the interface class of this application. This application is built by Eclipse Juno 4.2.1 and Android SDK. With the development of this application, it's expected to help the government of Kediri to accommodate complaint, suggestion, and information anywhere and anytime.

The developed applications tested by modularity and functionality. This test is done by using scenario with a black box method that reflecting this application features. The result of this test shows that this application can perform well in terms of modularity and functionality.

Key word: Android, complaint, mobile device, modularity

KATA PENGANTAR

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

Bismillahirrohmanirohim.

Alhamdulillahirabil'alamin, segala puji hanya milik Allah SubhanahuWata'alla, atas segala rahmat dan karunia-Nya yang tak terhingga sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul "**PENGEMBANGAN APLIKASI MODULAR SUARA WARGA PEMERINTAH KOTA KEDIRI BERBASIS ANDROID PADA PERANGKAT KOMUNIKASI BERGERAK**" dengan baik dan tepat waktu.

Tugas Akhir ini dikerjakan demi memenuhi salah satu syarat guna memperoleh gelar Sarjana Komputer di Jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember Surabaya. Penulis menyadari bahwa Tugas Akhir ini bukanlah tujuan akhir dari belajar karena belajar adalah sesuatu yang tidak terbatas.

Penulis mendapatkan banyak sekali doa, bantuan dan dukungan dari berbagai pihak dalam menyelesaikan Tugas Akhir ini. Atas berbagai bantuan dan dukungan tersebut, pada kesempatan ini penulis menyampaikan ucapan terima kasih yang sebesar-besarnya kepada:

1. Allah SWT atas limpahan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan Tugas Akhir ini dengan baik dan tepat waktu.
2. Bapak Rachmat Harahap dan Ibu Nasmiarti Anas yang tidak henti-hentinya memberikan dukungan, semangat, kasih sayang, serta selalu memberikan doa yang dipanjatkan untuk penulis.
3. Atika Zahra Rahmayanti, saudara kandung penulis yang selalu memberikan doa dan dukungan.
4. Bapak Dr.tech. Ir. R. V. Hari Ginardi, M.Sc. selaku dosen pembimbing 1 dan Bapak Dwi Sunaryono, S.Kom., M.Kom. selaku dosen pembimbing 2, yang telah memberikan

kepercayaan, dukungan, bimbingan, nasehat, serta semangat dikala penulis mengalami kesulitan.

5. Segenap staf dan dosen pengajar di Jurusan Teknik Informatika ITS.
6. Didik dan Ajong yang telah banyak memberikan bantuan dalam penyusunan Tugas Akhir ini.
7. Bachmid, Ibet, Fandi, Kemal, Melfa, Kaspul, Ajong, Ata, Megi, Agung, Rifi, Adi, Tegar, Satrio, Baskara, dan teman-teman seperjuangan dan sepermainan selama berkegiatan sehari-hari didalam dan diluar Kampus Perjuangan.
8. Keluarga Besar Angkatan 2011 yang telah menjadi keluarga kedua penulis selama menempuh kuliah di Teknik Informatika ITS.
9. Serta pihak-pihak lain yang namanya tidak dapat penulis sebutkan satu persatu.

Semoga Allah SWT memberkati dan membalas semua kebaikan yang telah dilakukan. Penulis menyadari masih banyak yang dapat dikembangkan pada Tugas Akhir ini. Oleh karena itu, penulis menerima setiap masukan dan kritik yang diberikan. Semoga Tugas Akhir ini dapat memberikan manfaat.

Surabaya, Juni 2015

Fahmi Armand Rachman

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR KODE SUMBER

BIODATA PENULIS

Fahmi Armand Rachman, dilahirkan di kota Jakarta, DKI Jakarta pada tanggal 17 April 1993. Penulis adalah anak kedua dari dua bersaudara. Penulis telah menempuh pendidikan formal yaitu di SDN Gondangdia 01 Pagi (1999-2005), SMP Negeri 19 Jakarta (2005-2008) dan SMA Negeri 70 Jakarta (2008-2011). Setelah lulus dari SMA Negeri 70 Jakarta pada tahun 2011, Penulis diterima di Jurusan Teknik Informatika ITS pada tahun 2011 dan terdaftar dengan NRP 5111100170. Di Jurusan Teknik Informatika ITS ini, Penulis mengambil Bidang Algoritma dan Pemrograman (AP). Penulis memiliki ketertarikan pada pengembangan *software* perangkat bergerak khususnya *platform* Android serta pengembangan *web*. Penulis dapat dihubungi melalui alamat *e-mail* di fahmi.armand17@gmail.com.

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi informasi saat ini memiliki banyak sisi positif bagi kehidupan bermasyarakat. Ini bisa dilihat dengan semakin banyaknya masyarakat yang memakai perangkat bergerak atau telepon pintar dalam kehidupan sehari-hari. Misalnya, untuk berkomunikasi, menulis catatan, memesan tiket pesawat terbang, melakukan pengecekan rekening bank, dan hal-hal lainnya yang bisa dilakukan dengan mudah cukup lewat telepon pintar mereka.

Disamping memudahkan komunikasi antar sesama, perkembangan teknologi informasi juga menjadi peluang bagi pemerintah untuk meningkatkan layanan terhadap masyarakat. Peluang ini dilihat oleh Pemerintah Kota Kediri dengan meluncurkan layanan aduan masyarakat bernama "SURGA" atau akronim dari "Suara Warga" berbasis *web*.

Dengan jumlah warga yang tercatat di Badan Pusat Statistik yang mencapai ± 260.000 orang, diharapkan layanan ini dapat menampung semua jenis pengaduan, saran dan informasi demi mengakomodasi kebutuhan dari masyarakat kota Kediri. Untuk meningkatkan kualitas layanan, maka Pemerintah Kota Kediri berinisiatif untuk membuat layanan tersebut dalam bentuk perangkat bergerak yang bisa diakses telepon pintar petugas Pemerintah Kota Kediri.

Aplikasi perangkat bergerak suara warga ini dikembangkan pada kakas kerja Android dengan versi 4.1 (*Jelly Bean*) ke atas. Android dipilih menjadi kakas kerja karena mengingat sampai saat ini kakas kerja tersebutlah yang dipakai oleh kebanyakan masyarakat Indonesia. Aplikasi ini dikembangkan dengan desain arsitektur modular, dimana ini kita bisa menganti ataupun menambah suatu komponen (modul) tanpa mempengaruhi sistem yang lain dan tidak perlu untuk rekompilasi ulang. Aplikasi ini juga akan dibangun dengan basis data berupa MySQL dan *web service*

PHP untuk mendapatkan data dari basis data. Selain itu, suara warga ini juga dilengkapi dengan fitur Google Map untuk mengetahui lokasi aduan yang diadukan oleh masyarakat.

Diharapkan dengan adanya aplikasi berbasis perangkat bergerak ini maka Pemerintah Kota Kediri bisa lebih mudah dan cepat dalam menanggapi setiap aduan yang masuk. Sehingga diharapkan hal ini bisa meningkatkan pelayanan Pemerintah Kota Kediri terhadap aduan, saran, dan informasi yang diberikan oleh masyarakat Kota Kediri.

1.2 Rumusan Masalah

Rumusan masalah yang diangkat dalam Tugas Akhir ini dapat dipaparkan sebagai berikut:

1. Bagaimana merancang sistem untuk digunakan oleh petugas Pemerintah Kota Kediri dalam menanggapi setiap aduan, saran, dan informasi yang diberikan oleh masyarakat Kota Kediri dengan menggunakan aplikasi berbasis perangkat bergerak?
2. Bagaimana merancang dan mengimplementasi sistem yang bersifat modular?
3. Bagaimana mengimplementasi rancangan untuk menanggapi setiap aduan, saran, dan informasi yang diberikan oleh masyarakat Kota Kediri dengan menggunakan aplikasi berbasis perangkat bergerak?

1.3 Batasan Masalah

Permasalahan yang dibahas dalam Tugas Akhir ini memiliki beberapa batasan, diantaranya sebagai berikut:

1. Aplikasi ini adalah modul aplikasi perangkat bergerak yang berbasis pada sistem operasi Android dengan versi minimal 4.1 (*Jelly Bean*).
2. Aplikasi perangkat bergerak ini membutuhkan koneksi internet untuk beroperasi.

3. Pengguna aplikasi ini hanya untuk petugas Pemerintah Kota Kediri.
4. Progress realisasi tanggapan aduan tidak ada pada ruang lingkup sistem.

1.4 Tujuan dan Manfaat

Tujuan dari pengerjaan Tugas Akhir ini adalah sebagai berikut:

1. Untuk merancang sistem untuk digunakan oleh petugas Pemerintah Kota Kediri dalam menanggapi setiap aduan, saran, dan informasi yang diberikan oleh masyarakat Kota Kediri dengan menggunakan aplikasi perangkat bergerak.
2. Untuk merancang dan mengimplementasi sistem yang bersifat modular.
3. Untuk mengimplementasikan rancangan untuk menanggapi setiap aduan, saran, dan informasi yang diberikan oleh masyaratakat Kota Kediri dengan menggunakan aplikasi perangkat bergerak.

Tugas Akhir ini dikerjakan dengan harapan dapat memberikan manfaat pada bidang teknik informatika di pemerintahan dalam menerima dan menanggapi aduan, saran, dan informasi yang telah diperoleh dari warga menggunakan perangkat bergerak.

1.5 Metodologi

Ada beberapa tahap dalam proses pengerjaan Tugas Akhir ini. Berikut ini adalah tahap-tahap dalam pembuatan Tugas Akhir.

a. Studi Literatur

Tahap ini merupakan tahap pengumpulan informasi dan pembelajaran yang akan digunakan pada Tugas Akhir ini. Studi literatur meliputi diskusi dan pemahaman mengenai topik Tugas Akhir ini yaitu modularitas untuk aplikasi Android.Tahap ini merupakan perancangan sistem dengan menggunakan studi

literatur dan mempelajari konsep aplikasi yang akan dibuat. Dengan berbekal teori, metode, dan informasi yang sudah terkumpul pada tahap sebelumnya diharapkan dapat membantu dalam proses perancangan sistem.

b. Implementasi

Implementasi merupakan tahap membangun aplikasi, yaitu mengimplementasikan desain atau rancangan yang dibuat ke dalam baris kode program. Pengmbangan aplikasi ini dimulai dengan membuat fungsi pada *web service* untuk mengakses basis data pada server dengan menggunakan Bahasa pemrograman PHP. Kemudian dilanjutkan dengan membuat aplikasi klien Android dengan menggunakan bahasa pemrograman Java.

c. Uji Coba dan Evaluasi

Pada tahap ini dilakukan pengujian terhadap aplikasi yang dibuat menggunakan data ataupun kasus yang telah disiapkan. Tujuan pengujian ini adalah untuk menguji fungsionalitas dari aplikasi, mencari masalah yang mungkin muncul, dan melakukan perbaikan bila ada kekurangan.

d. Penyusunan Laporan Tugas Akhir

Tahap ini digunakan untuk membuat laporan Tugas Akhir yang berisi dokumentasi mengenai pembuatan serta hasil dari implementasi perancangan yang telah dibuat. Laporan Tugas Akhir ini bertujuan untuk mendokumentasikan pengerjaan Tugas Akhir dan menggambarkan keseluruhan proses pengerjaan Tugas Akhir dan dapat berguna bagi pembaca yang tertarik sebagai referensi untuk pengembangan lebih lanjut kedepannya.

1.6 Sistematika Penyusunan Laporan

Buku Tugas Akhir ini disusun dengan sistematika laporan sebagai berikut:

1. Bab I. Pendahuluan
   Bab ini meliputi latar belakang masalah, rumusan permasalahan, batasan masalah, tujuan dan manfaat pembuatan Tugas Akhir, metodologi yang digunakan, dan sistematika penyusunan laporan Tugas Akhir.

2. Bab II. Tinjauan Pustaka
   Bab ini meliputi dasar teori dan penunjang yang berkaitan dengan pokok pembahasan dan mendasari pembuatan Tugas Akhir ini.

3. Bab III. Perancangan Perangkat Lunak
   Bab ini membahas analisis dari sistem yang akan dibuat meliputi deskripsi umum perangkat lunak, spesifikasi kebutuhan, dan identifkasi pengguna. Kemudian membahas rancangan dari sistem yang dibuat meliputi rancangan skenario kasus penggunaaan, arsitektur sistem, data, antarmuka, dan proses aplikasi.

4. Bab IV. Implementasi
   Bab ini membahas implementasi dari desain sistem yang dilakukan pada tahap desain, meliputi *pseudocode* dan implementasi antarmuka dari perangkat lunak.

5. Bab V. Uji Coba dan Evaluasi
   Bab ini membahas uji coba dari perangkat lunak yang dibuat dengan melihat keluaran yang dihasilkan oleh perangkat lunak, analisis, dan evaluasi untuk mengetahui kemampuan perangkat lunak.

6. Bab VI. Kesimpulan dan Saran
   Bab ini berisi kesimpulan dari hasil uji coba yang dilakukan serta saran untuk pengembangan lebih lanjut perangkat lunak.

BAB II
TINJAUAN PUSTAKA

Bab ini berisi penjelasan teori-teori yang berkaitan dengan pengimplementasian perangkat lunak. Penjelasan ini bertujuan untuk memberikan gambaran secara umum terhadap sistem yang dibuat dan berguna sebagai penunjang dalam pengembangan.

2.1 Situs Sistem Layanan Pengaduan Masyarakat Kota Kediri

Kota Kediri yang berada di provinsi Jawa Timur dan memiliki jumlah yang tercatat di Badan Pusat Statistik yang mencapai ± 260.000 orang, sedang melakukan pembangunan besar-besaran di dalam hal teknologi informasi. Salah satu hasil dari pembangunan itu adalah sistem layanan pengaduan masyarakat kota kediri atau di kenal sebagai suara warga kota kediri.

Suara warga kota kediri ini merupakan wadah untuk menerima dan menanggapi aduan, saran, dan informasi yang telah diperoleh dari warga. Warga memiliki dua cara untuk dapat mengirim aduan, saran, dan informasi yaitu dengan cara pengirim pesan singkat ke nomor yang sudah ditentukan dengan format yang sudah ditentukan juga. Selain dengan mengirim pesan singkat, warga juga dapat menggunakan situs suara warga kota kediri untuk mengirim aduan, saran, dan informasi. Di dalam situs ini warga juga dapat mengirim informasi lokasi dan dokumen-dokumen yang dapat dijadikan bukti untuk mempercepat proses penanggapan aduan, saran, dan informasi tersebut.

Para Petugas juga menggunakan situs ini untuk menanggapi aduan-aduan yang sudah masuk. Petugas dapat mengklasifikasi aduan-aduan tersebut ke dalam dinas-dinas yang terkait perihal aduan tersebut. Setelah di klasifikasi, dinas-dinas terkait tersebut dapat menanggapi aduan dan

melakukan tindakan yang terkait aduan-aduan yang sudah masuk.

Gambar 2.1 merupakan halaman depan dari situs sistem layanan pengaduan masyarakat Kota Kediri.

Gambar 2.1 Halaman Depan Sistem Layanan Pengaduan Masyarakat Kota Kediri

2.2 Android SDK

Android SDK adalah *tools* API (*Aplication Programming Interface*) yang digunakan untuk mulai mengembangkan aplikasi pada kakas kerja Android menggunakan bahasa pemrograman Java. Android merupakan subset perangkat lunak untuk ponsel yang meliputi sistem operasi, *middleware* dan aplikasi kunci yang dirilis oleh Google. Saat ini disediakan Android SDK *(Software Development Kit)* sebagai alat bantu dan API untuk mulai mengembangkan aplikasi pada kakas kerja Android menggunakan bahasa pemrograman Java. Sebagai kakas kerja aplikasi netral, Android memberi Anda kesempatan untuk membuat aplikasi yang kita butuhkan yang bukan merupakan aplikasi bawaan telepon pintar. Beberapa fitur Android yang paling penting adalah:

- Kakas kerja aplikasi yang mendukung penggantian komponen dan *reusable*
- Mesin Virtual Dalvik dioptimalkan untuk perangkat bergerak
- *Integrated browser* berdasarkan *engine open source webkit*
- Grafis yang dioptimalkan dan didukung oleh *libraries* grafis 2D, grafis 3D berdasarkan spesifikasi *openGL ES* 1.0 (Opsional Akselerasi Perangkat Keras)
- SQLite untuk penyimpanan data
- Media Support yang mendukung audio, video dan gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF), GSM Telephony (tergantung perangakat bergerak)
- Bluetooth, EDGE, 3G, WiFi (tergantung perangakat bergerak)
- Kamera, GPS, kompas dan *accelerator* (tergantung perangakat bergerak)

Lingkungan pengembangan yang lengkap dan kaya, termasuk perangkat emulator, *tools* untuk *debuging*, profil dan kinerja memori, dan *plugin* untuk IDE Eclipse.

2.3 *Web Service*

Web service menurut *www.w3.org* mendefinisikan *web service* sebagai "sebuah perangkat lunak yang dapat teridentifikasi oleh URI dan memiliki *interface* yang didefinisikan, dideskripsikan, dan dimengerti oleh XML dan juga mendukung interaksi langsung dengan perangkat lunak yang lain dengan menggunakan pesan berbasis XML melalui protokol internet".

Web service adalah sebuah perangkat lunak yang tidak terpengaruh oleh kakas kerja, ia akan menyediakan *method-method* yang dapat diakses oleh jaringan. Ia juga akan

menggunakan XML untuk pertukaran data, khususnya pada dua entitas bisnis yang berbeda.

Definisi lain *web service* adalah perangkat lunak yang dirancang untuk mendukung interopabilitas mesin ke mesin yang dapat berinteraksi melalui jaringan. *Web service* memiliki antarmuka yang dijelaskan dalam format mesin-*processable* (khusus WSDL). Sistem lain berinteraksi dengan *web service* ditentukan oleh deskripsi dengan menggunakan pesan SOAP, biasanya disampaikan menggunakan HTTP dengan serialisasi XML dalam hubungannya dengan *web* lainnya yang terkait standar.

Dalam pengertian yang sederhana, XML *Web Service* dapat di definisikan sebagai aplikasi yang diakses oleh aplikasi yang lain. Mungkin orang berpendapat itu semacam situs, tetapi itu bukan demikian. Ada perbedaan-perbedaan yang membedakan *web service* dengan situs, diantaranya:

- *Web service* tidak memiliki *interface* atau antarmuka pengguna, sedangkan situs memilikinya.
- *Web service* dibuat untuk berinteraksi langsung dengan applikasi yang lain baik berbeda sistem operasi atau konsep sekalipun, sedangkan situs dibuat untuk berinteraksi langsung dengan pengguna.
- *Web service* Dibuat untuk bekerja pada semua tipe klien aplikasi atau perangkat *device*, sedangkan situs dibuat untuk bekerja pada *web browser*.

2.4 PHP

PHP: *Hypertext Preprocessor* adalah bahasa skrip yang dapat ditanamkan atau disisipkan ke dalam HTML. PHP banyak dipakai untuk memrogram situs yang dinamis. PHP dapat digunakan untuk membangun sebuah CMS.

Pada awalnya PHP merupakan kependekan dari *Personal Home Page* (situs personal). PHP pertama kali

dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada waktu itu PHP masih bernama *Form Interpreted* (FI), yang wujudnya berupa sekumpulan skrip yang digunakan untuk mengolah data formulir dari sebuah situs.

Selanjutnya Rasmus merilis kode sumber tersebut untuk umum dan menamakannya PHP/FI. Dengan perilisan kode sumber ini menjadi sumber terbuka, maka banyak pemrogram yang tertarik untuk ikut mengembangkan PHP. Pada Juni 2004, Zend merilis PHP 5.0. Dalam versi ini, inti dari interpreter PHP mengalami perubahan besar. Versi ini juga memasukkan model pemrograman berorientasi objek ke dalam PHP untuk menjawab perkembangan bahasa pemrograman ke arah paradigma berorientasi objek. PHP digunakan untuk mengembangkan aplikasi suara warga sebagai *web service* karena PHP memiliki dukungan lebih baik untuk sistem manajemen basis data MySQL.

2.5 JSON

JSON (dilafalkan "Jason"), singkatan dari *JavaScript Object Notation* (Bahasa Indonesia: notasi objek *JavaScript*), adalah suatu format ringkas pertukaran data komputer. Formatnya berbasis teks serta digunakan untuk merepresentasikan struktur data sederhana dan larik asosiatif (disebut objek). Format JSON sering digunakan untuk mentransmisikan data terstruktur melalui suatu koneksi jaringan pada suatu proses yang disebut serialisasi. Aplikasi utamanya adalah pada pemrograman aplikasi situs AJAX dengan berperan sebagai alternatif terhadap penggunaan tradisional format XML.

Walaupun JSON didasarkan pada subset bahasa pemrograman *JavaScript* dan umumnya digunakan dengan bahasa tersebut, JSON dianggap sebagai format data yang tak tergantung pada suatu bahasa. Kode untuk pengolahan dan pembuatan data JSON telah tersedia untuk banyak jenis bahasa pemrograman. Situs *www.json.org* menyediakan daftar komprehensif pengikatan JSON yang tersedia, disusun menurut bahasa.

2.6 Google Map API V2

Google Maps API V2 adalah kumpulan API yang memungkinkan Anda menghamparkan data pengguna di peta khusus Google. Anda dapat membuat aplikasi yang bisa diakses di situs internet dan seluler menarik dengan kakas kerja pemetaan canggih dari Google, termasuk basis data citra satelit, *street view*, profil ketinggian, petunjuk arah mengemudi, peta dengan sentuhan gaya, demografi, analisis, dan basis data yang besar. Dengan cakupan global yang paling akurat di dunia, dan komunitas pemetaan yang aktif memperbarui setiap harinya, pengguna mendapatkan manfaat dari layanan yang ditingkatkan secara terus-menerus.

Dalam aplikasi suara warga ini, penggunaan Google Maps API V2 digunakan untuk mendapatkan lokasi dari aduan yang diadukan.

2.7 Basis Data MySQL

MySQL adalah sebuah implementasi dari sistem manajemen basis data relasional (RDBMS) yang didistribusikan secara gratis dibawah lisensi GPL (*General Public License*). Setiap pengguna dapat secara bebas menggunakan MySQL, namun dengan batasan perangkat

lunak tersebut tidak boleh dijadikan produk turunan yang bersifat komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam basis data yang telah ada sebelumnya. SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian basis data, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

Kehandalan suatu sistem basis data (DBMS) dapat diketahui dengan cara melihat pengoptimasinya dalam melakukan proses perintah-perintah SQL yang dibuat oleh pengguna maupun program-program aplikasi yang memanfaatkannya. Sebagai peladen basis data, MySQL mendukung operasi basis data transaksional maupun operasi basis data non-transaksional. Pada modus operasi non-transaksional, MySQL dapat dikatakan unggul dalam hal unjuk kerja dibandingkan perangkat lunak peladen basis data kompetitor lainnya. Namun pada modus non-transaksional tidak ada jaminan atas reliabilitas terhadap data yang tersimpan, karenanya modus non-transaksional hanya cocok untuk jenis aplikasi yang tidak membutuhkan reliabilitas data seperti aplikasi blogging berbasis *web* (wordpress), CMS, dan sejenisnya. Untuk kebutuhan sistem yang ditujukan untuk bisnis sangat disarankan untuk menggunakan modus basisdata transaksional, hanya saja sebagai konsekuensinya unjuk kerja MySQL pada modus transaksional tidak secepat unjuk kerja pada modus non-transaksional.

2.8 Modularitas

Dalam ilmu teknik, sistem modular adalah sistem yang membagi suatu elemen dari produk ataupun proses dan menetapkannya ke dalam modul-modul sesuai dengan perencanaan yang telah dibuat.

Menurut sudut pandang ilmu teknik, modularitas secara umum memiliki tiga tujuan, yaitu:

- Untuk mengelola kompleksitas suatu pengembangan
- Untuk memungkinkan pengerjaan secara paralel
- Untuk mengakomodasi ketidakpastian yang akan terjadi di masa depan

Modularitas mengakomodasi ketidakpastian yang terjadi di masa depan karena elemen tertentu pada rencang bangun modular dapat diubah selama aturan- aturan perancangan ditaati. Oleh karena itu, dalam arsitektur modular, modul baru dapat mengganti modul lama dengan mudah dan dengan biaya yang rendah.

IEEE mendefinisikan modul sebagai sebuah unit program yang diskrit dan diidentifikasi sehubungan dengan komputasi, menggabungkan dengan unit lain dan memuat: sebagai contoh, input atau output, *assembler*, *compiler*, *linkage editor* dan *executive routine*. Dalam web, *programmer* Drupal menjelaskan bahwa modul adalah istilah spesifik untuk sebuah tipe proyek, sedangkan *programmer* Mambo menuliskan bahwa modul adalah menampilkan informasi tertentu. *Programmer* Ruby mendefinisikan modul adalah sebuah kumpulan *method* dan *variable.*

Konsep Modularitas berhubungan dengan dekomposisi. Pengembangan sistem menjadi lebih sederhana karena hanya terfokus pada satu modul terlebih dahulu, baru dilakukan integrasi antar modul. Bahasa-bahasa pemrograman modern menyediakan cara-cara untuk dapat membuat sistem perangkat lunak secara modular demikian, yaitu:

- Cara untuk mengelompokkan bersama data dan metoda operasi
- Cara untuk mendefinisikan *interface*
- Cara untuk menyembunyikan rincian internal dari data dan metoda-metoda operasi
- Cara untuk kompilasi secara terpisah

Bahasa Java memungkinkan konsep pemrograman modular, enkapsulasi, dan abstraksi data dengan disediakannya skema-skema *class*, *interface*, dan *packages*.

2.9 *Black-Box Testing*

Black-box testing adalah metode pengujian perangkat lunak yang tes fungsionalitas dari aplikasi yang bertentangan dengan struktur internal atau kerja (lihat pengujian *white-box*). pengetahuan khusus dari kode aplikasi / struktur internal dan pengetahuan pemrograman pada umumnya tidak diperlukan. Uji kasus dibangun di sekitar spesifikasi dan persyaratan, yakni, aplikasi apa yang seharusnya dilakukan. Menggunakan deskripsi eksternal perangkat lunak, termasuk spesifikasi, persyaratan, dan desain untuk menurunkan uji kasus. Tes ini dapat menjadi fungsional atau non-fungsional, meskipun biasanya fungsional. Perancang uji memilih input yang valid dan tidak valid dan menentukan output yang benar. Tidak ada pengetahuan tentang struktur internal benda uji itu.

Metode uji dapat diterapkan pada semua tingkat pengujian perangkat lunak: unit, integrasi, fungsional, sistem. Dan penerimaan. Ini biasanya terdiri dari kebanyakan jika tidak semua pengujian pada tingkat yang lebih tinggi, tetapi juga bisa mendominasi unit testing juga.

Metode ujicoba *black-box* memfokuskan pada keperluan fungsional dari software. Karena itu ujicoba *black-box* memungkinkan pengembang software untuk membuat himpunan kondisi input yang akan melatih seluruh syarat-syarat fungsional suatu program. Ujicoba *black-box* bukan

merupakan alternatif dari ujicoba *white-box*, tetapi merupakan pendekatan yang melengkapi untuk menemukan kesalahan lainnya, selain menggunakan metode *white-box*. Ujicoba *black-box* berusaha untuk menemukan kesalahan dalam beberapa kategori, diantaranya:

1. Fungsi-fungsi yang salah atau hilang
2. Kesalahan *interface*
3. Kesalahan dalam struktur data atau akses database eksternal
4. Kesalahan performa
5. Kesalahan inisialisasi dan terminasi

BAB III
PERANCANGAN PERANGKAT LUNAK

Perancangan merupakan bagian penting dari pembuatan suatu perangkat lunak yang berupa perencanaan-perencanaan secara teknis aplikasi yang dibuat. Sehingga bab ini secara khusus akan menjelaskan perancangan sistem yang dibuat dalam Tugas Akhir ini. Berawal dari deskripsi umum aplikasi hingga perancangan proses, alur, dan implementasinya.

3.1 Analisis Sistem

Aplikasi suara warga Pemerintah Kota Kediri ini dibangun pada perangkat komunikasi bergerak sebagai alternatif lain dari situs suara warga yang telah ada dan dipergunakan saat ini. Dibangun pada perangkat komunikasi bergerak untuk memberikan akses untuk menanggapi aduan dimana saja dan kapan saja.

Untuk itu pada tahap analisis ini didefinisikan kebutuhan yang akan dipenuhi dalam pembuatan aplikasi ini. Berikut penjabaran bagian-bagian tahap anbalisis yang mencakup deskripsi umum sistem, spesifikasi kebutuhan fungsional, spesifikasi kebutuhan non-fungsional, dan identifikasi pengguna.

3.1.1 Deskripsi Umum Sistem

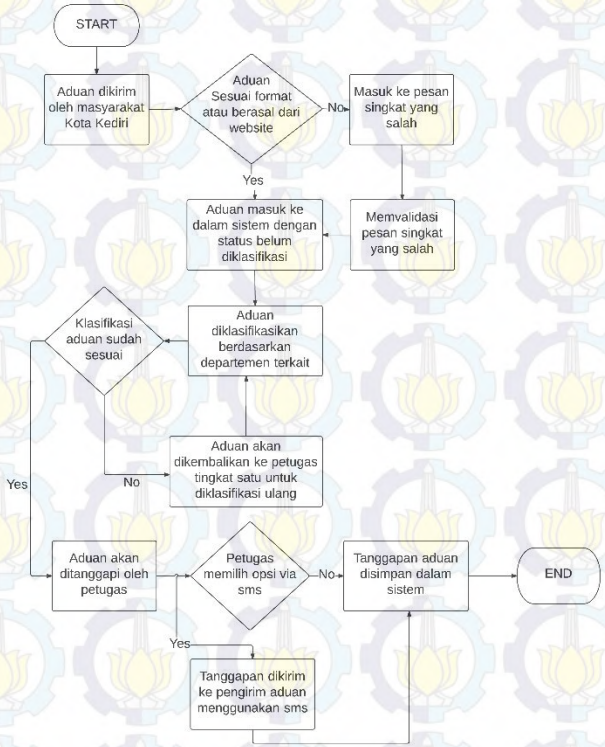
Pada Tugas Akhir ini akan dibangun suatu aplikasi Modular Suara Warga Kota Kediri yang berjalan pada perangkat bergerak berbasis Android. Tujuan dari aplikasi ini adalah untuk menjalankan modul-modul yang telah mengimplementasi *interface* dari aplikasi utama dan juga mempermudah akses petugas pemerintah Kota Kediri dalam menanggapi aduan, saran, dan informasi darimana saja dan kapan saja. Aplikasi ini diharapkan bisa meningkatkan

pelayanan pemerintah dan mempermudah menanggapi setiap aduan.

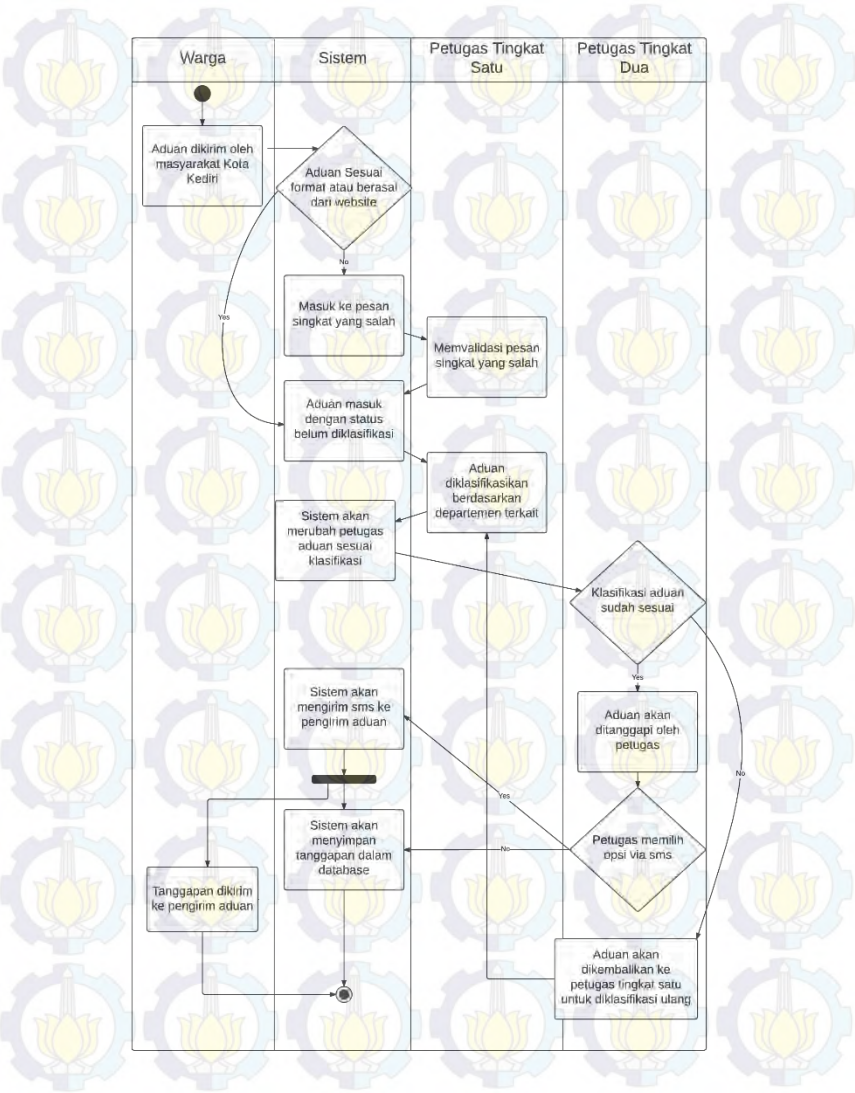
Pengguna dari aplikasi ini adalah pihak yang bertugas untuk menanggapi aduan, saran, dan informasi yakni petugas yang memiliki akses tingkat satu dan akses tingkat dua. Pengguna dengan akses satu dapat melihat aduan, mencari aduan, menindaklanjuti aduan, membalas aduan, melihat lampiran, melihat peta tempat lokasi aduan, mengirim pesan singkat kepada petugas, validasi pesan singkat yang salah, dan mengunduh laporan aduan. Sedangkan untuk petugas tingkat dua dapat melihat aduan, mencari aduan, menindaklanjuti aduan, membalas aduan, melihat lampiran, melihat peta tempat lokasi aduan, dan mengembalikan aduan kepada petugas satu. Untuk Petugas tingkat satu dapat menindaklanjuti aduan berupa memilih petugas tingkat dua untuk menindaklanjuti aduan, mengubah prioritas, mengubah status aduan, dan mengubah status aduan menjadi *spam*. Sedangkan untuk petugas tingkat dua dapat menindaklanjuti aduan berupa mengubah prioritas aduan, mengubah status aduan, mengubah status aduan menjadi *spam*, dan mengembalikan aduan ke petugas tingkat dua.

Proses bisnis sistem untuk menanggapi aduan berasal dari masyarakat yang mengirimkan aduan melalui pesan singkat sesuai format yang sudah ditentukan atau melalui *website* yang tersedia. Sistem akan mengecek apakah format dari pesan singkat itu benar atau tidak jika benar maka akan dimasukkan menjadi aduan sedangkan jika tidak maka dimasukkan menjadi pesan singkat tidak valid. Untuk pesan singkat yang tidak valid, dibutuhkan validasi oleh petugas untuk menjadi aduan. Aduan kemudian akan diklasifikasikan oleh petugas tingkat satu ke departemen-departemen(SKPD). Jika klasifikasinya tidak sesuai, maka petugas tingkat dua bisa mengembalikan aduan tersebut ke petugas tingkat satu untuk diklasifikasi ulang. Lalu aduan dapat di tanggapi oleh petugas.

Untuk diagram alir proses ini dapat dilihat pada Gambar 3.1 sedangkan untuk diagram aktifitas proses ini dapat dilihat pada Gambar 3.2.



Gambar 3.1 Diagram Alir Proses Bisnis



Gambar 3.2 Diagram Aktifitas Proses Bisnis Sistem

Untuk membangun modul aplikasi pada perangkat bergerak yang terhubung dengan sistem yang sudah ada, digunakan *web service* untuk dapat berkomunikasi ke basis data yang ada pada *server* seperti yang ada pada situs 'Suara Warga Kediri'.

3.1.2 Spesifikasi Kebutuhan Fungsional

Berdasarkan deskripsi umum sistem sistem diatas, maka dapat disimpulkan bahwa kebutuhan fungsionalitas dari aplikasi ini adalah sebagai berikut:

1. Menampilkan aduan
2. Mencari aduan
3. Menjawab aduan
4. Mengubah status aduan
5. Mengubah prioritas aduan
6. Memilih petugas aduan
7. Mengembalikan aduan
8. Mengubah status *spam* aduan
9. Menampilkan lampiran aduan
10. Mengunduh lampiran
11. Menampilkan peta lokasi aduan
12. Validasi Pesan Singkat yang salah
13. Mengirim Pesan Singkat ke petugas
14. Mengunduh laporan

3.1.3 Spesifikasi Kebutuhan Non-Fungsional

Berikut daftar kebutuhan non-fungsional yang harus dipenuhi agar aplikasi berjalan sesuai kebutuhan:

1. Koneksi internet
   Koneksi internet dibutuhkan untuk dapat mengakses basis data dari *server*.
2. Keamanan

Karena aplikasi pada situs menggunakan otentikasi, maka dibutuhkan otentikasi untuk mengakses sistem.

3.1.4 Identifikasi Pengguna

Berdasarkan deskripsi umum diatas, maka dapat diketahui bahwa pengguna yang akan menggunakan aplikasi ini adalah petugas atau pengguna yang memiliki akses tingkat satu dan tingkat dua. Petugas tingkat satu yaitu SKPD (Satuan Kerja Pemerintah Daerah) Hubungan masyarakat dan Wali Kota Kediri. Sedangkan petugas tingkat dua yaitu SKPD selain hubungan masyarakat. Penjelasan mengenai pengguna yang disebut aktor dalam sistem, dijelaskan pada Tabel 3.1.

Tabel 3.1 Identifikasi Pengguna

Nama Aktor	Definisi
Petugas Tingkat satu	Orang yang berinteraksi dengan sistem sebagai orang yang akan mengklasifikasikan aduan kepada SKPD-SKPD terkait dan juga dapat menindaklanjuti aduan.
Petugas Tingkat dua	Orang yang berinteraksi dengan sistem sebagai orang yang akan menindaklanjuti aduan yang di dapat dari petugas tingkat satu.

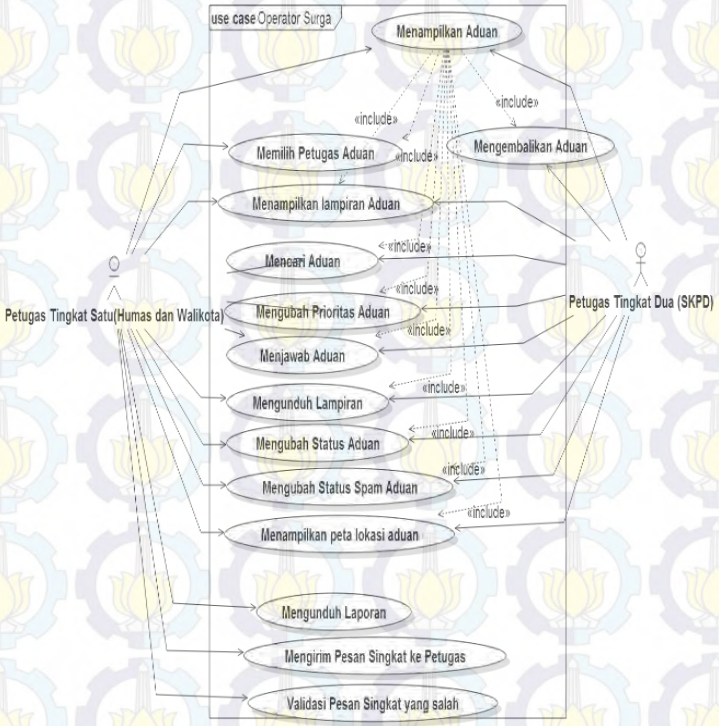
3.2 Perancangan Sistem

Tahap perancangan dalam subbab ini dibagi menjadi beberapa bagian yaitu perancangan diagram kebutuhan sistem, perancangan arsitektur sistem, perancangan data, perancangan modularitas, dan perancangan antarmuka sistem.

3.2.1 Perancangan Diagram Kebutuhan Sistem

Diagram kebutuhan sistem merupakan diagram kebutuhan yang menggambarkan fungsionalitas sistem dan aktor-aktornya.

Berdasarkan Gambar 3.3 ada dua aktor yang terlibat yaitu petugas tingkat satu dan tingkat dua. Pengguna berinteraksi dengan sistem untuk menjalankan segala fungsionalitas sistem. Diagram kebutuhan pada sistem secara umum dijelaskan pada Tabel 3.2.



Gambar 3.3 Diagram Kebutuhan

Tabel 3.2 Diskripsi Diagram Kebutuhan

No.	Kode	Nama	Keterangan
1.	UC-01	Menampilkan aduan	Pengguna dapat melihat Aduan yang di dapat dari masyarakat dalam bentuk *listview.*
2.	UC-02	Mencari aduan	Petugas dapat mencari aduan sesuai dengan id aduan, tanggal aduan, isi aduan, dan topik aduan.
3.	UC-03	Menjawab aduan	Pengguna dapat menjawab aduan. Pengguna dapat memilih menjawab aduan tersebut melalui pesan singkat atau situs jika aduan tersebut berasal dari pesan singkat.
4.	UC-04	Mengubah status aduan	Pengguna dapat mengubah prioritas dari aduan. Statusnya adalah diterima humas, diterima departemen terkait, diterima staff, dan selesai.
5.	UC-05	Mengubah prioritas aduan	Pengguna dapat mengubah prioritas dari aduan. Prioritasnya adalah rendah, sedang, dan tinggi.
6.	UC-06	Memilih petugas aduan	Petugas tingkat satu dapat memilih petugas tingkat dua untuk menindaklanjuti aduan.
7.	UC-07	Mengembalikan aduan	Petugas tingkat dua dapat mengembalikan aduan jika

No.	Kode	Nama	Keterangan
			aduan tersebut berada di luar SKPD dari petugas yang bersangkutan.
8.	UC-08	Mengubah status *spam* aduan	Pengguna dapat mengubah status aduan tersebut menjadi *spam.*
9.	UC-09	Menampilkan lampiran aduan	Pengguna dapat melihat dokumen-dokumen atau foto-foto yang dilampirkan di aduan.
10.	UC-10	Mengunduh lampiran	Pegguna dapat mengunduh file-file lampiran yang telah di unggah.
11.	UC-11	Menampilkan peta lokasi aduan	Pengguna dapat menampilkan peta lokasi aduan.
12.	UC-12	Mengirim Pesan Singkat ke petugas	Petugas tingkat satu dapat mengirim pesan singkat kepada petugas lain.
13.	UC-13	Validasi Pesan Singkat yang salah	Petugas tingkat satu dapat mem-validasi aduan yang salah *syntax* yang dikirim melalui pesan *singkat*.
14.	UC-14	Mengunduh laporan	Petugas tingkat satu dapat mengunduh laporan dari setiap aduan. Laporan tersebut dapat diunduh berdasarkan tahun.

3.2.2 Perancangan Arsitektur Sistem

Aplikasi ini dirancang memiliki dua sisi, yaitu sisi *server* dan sisi *client*. *Server* merupakan *web service* dan basis

data *server*, sedangkan *client* merupakan aplikasi yang beroperasi pada perangkat bergerak berbasis Android. *Server* bertugas untuk menerima, mengirim, dan menyimpan data dari *client* yang disimpan di dalam basis data. Kemudian mengolahnya dan mengirimkan hasilnya kepada *client*. *Client* bertugas untuk menerima input dari pengguna dan menampilkan data-data yang didapat dari basis data.

Berdasarkan perancangan arsitektur sistem pada Gambar 3.4, data-data disimpan dalam basis data dapat di peroleh dari *web service*. Untuk mendapatkan data-data, *web service* ini akan melakukan *query* basis data. Hasil *query* tersebut lalu dikirim ke perangkat Android dengan menggunakan *encoding* JSON. Lalu data tersebut ditampilkan di perangkat Android. Untuk mengirim data dari perangkat Android ke *web service,* Android menggunakan *method post* yang akan diterima oleh *web service*. *Web service* tersebut akan memasukan atau mengubah data tersebut ke dalam basis data.

Gambar 3.4 Perancangan Arsitektur Sistem

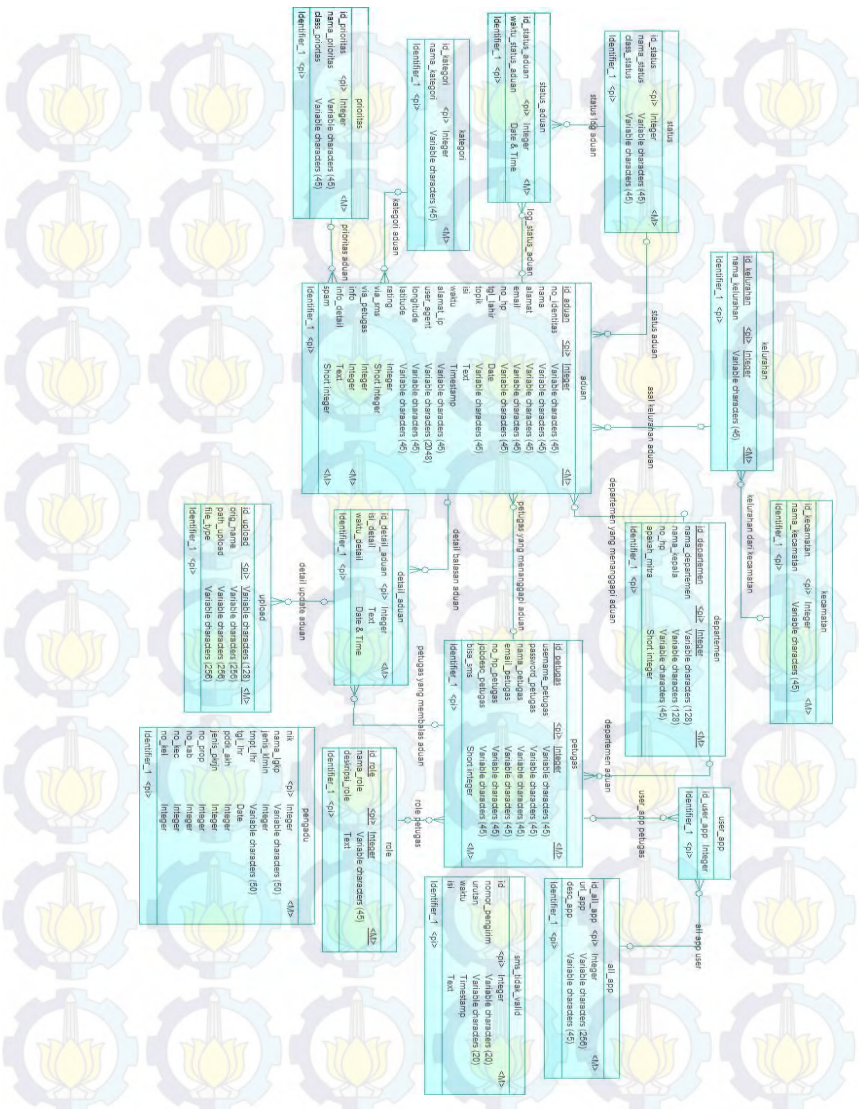
3.2.3 Perancangan Data

Perancangan data adalah hal yang penting dalam sistem, karena diperlukan data yang tepat agar sistem dapat beroperasi dengan benar. Sistem yang dibuat membutuhkan data masukan dan memberikan data keluaran.

Pada subbab ini dijelaskan tentang rancangan basis data yang digunakan pada modul aplikasi. Basis data yang ada pada sistem ini menggunakan RDBMS MySQL.

Basis data digunakan untuk menyimpan informasi yang dibutuhkan baik dalam aplikasi ini maupun situs Suara Warga Kota Kediri yang sudah ada. Basis data ini juga di pakai oleh modul-modul aplikasi ini.

Conceptual Data Model (CDM) adalah diagram yang menjelaskan desain basis data secara *independent* tanpa mengacu pada spesifikasi basis data tertentu. CDM berfungsi untuk menggambarkan tabel yang digunakan beserta relasinya Gambar 3.5 menunjukkan diagram *Conceptual Data Model* (CDM) untuk basis data pertama yang merupakan basis data Suara Warga Kota Kediri. Perancangan data untuk lebih detail dapat dilihat pada LAMPIRAN A PERANCANGAN DATA.



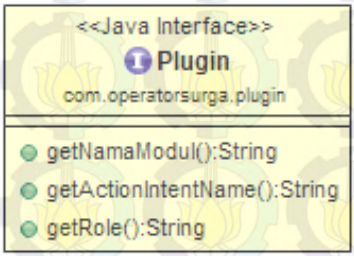
Gambar 3.5 Diagram CDM Basis Data

3.2.4 Perancangan Modularitas

Secara garis besar, modularitas yang dibangun dalam Tugas Akhir ini mirip dengan *launcher* yang ada di menu aplikasi Android. Modul-modul aplikasi tersebut hanya bisa dibuka melalui aplikasi ini.

Hal-hal yang harus disiapkan untuk membuat modul aplikasi ini yaitu aplikasi yang sudah mengimplementasikan interface dari aplikasi utama. Ditambah dengan sedikit mengubah Android *manifest* dari aplikasi ini supaya aplikasi ini tidak muncul di *launcher* Android dan bisa dijalankan di aplikasi utama.

Interface dari aplikasi utama ini memiliki tiga method yang masing-masing digunakan untuk nama modul, *activity* utama dari aplikasi modul, dan *role* yang dapat menggunakan modul ini. Kelas diagram untuk *interface* plugin ini ditunjukkan pada Gambar 3.6.



Gambar 3.6 Kelas Diagram dari *Interface* Plugin

3.2.5 Perancangan Antarmuka Sistem

Pada Tugas Akhir ini, antarmuka sistem hanya ada pada modul aplikasi *client*. Rancangan antarmuka sistem meliputi antarmuka login, menu modul, sub menu operator surga, manajemen aduan, detail aduan, balas aduan, tindak lanjut aduan, peta, list lampiran, gambar lampiran, *list* validasi sms, validasi sms, sms petugas, dan laporan.

3.2.5.1 Perancangan Antarmuka Login

Perancangan antarmuka login ditunjukkan pada Gambar 3.7. Antarmuka ini merupakan halaman pertama dalam aplikasi ini. Antarmuka ini berfungsi sebagai pembuktian keaslian pengguna untuk masuk ke dalam aplikasi.

Gambar 3.7 Perancangan Antarmuka Login

Komponen-komponen yang ada Gambar 3.7 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. Ikon dari aplikasi Suara Warga.
c. Dua *EditText* yang berfungsi untuk memasukan *username* dan *password* untuk pembuktian keaslian pengguna.
d. *Button login* yang berfungsi untuk masuk ke dalam aplikasi jika pembuktian keaslian berhasil.

e. *TextView* yang digunakan untuk menampilkan *copyright* dan alamat Pemerintah Kota Kediri.

3.2.5.2 Perancangan Antarmuka Menu Modul

Antarmuka menu modul berfungsi untuk menampilkan modul-modul yang ada di dalam aplikasi ini. Pada antarmuka ini terdapat *list* dari modul-modul yang sudah terdapat dalam aplikasi. Halaman ini ditunjukkan pada Gambar 3.8.

Gambar 3.8 Perancangan Antarmuka Menu

Komponen-komponen yang ada Gambar 3.8 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. Sebuah *ListView* yang berfungsi untuk menampilkan modul-modul yang sudah ada di dalam aplikasi.

3.2.5.3 Perancangan Antarmuka Sub Menu Operator Surga

Antarmuka sub menu operator surga merupakan antarmuka yang muncul ketika modul operator dipilih melalui antarmuka menu modul. Pada antarmuka ini terdapat empat sub menu. Sub menu tersebut adalah manajemen aduan, validasi sms, sms, dan laporan. Halaman tersebut ditunjukkan pada Gambar 3.9.

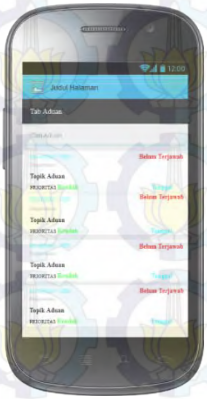
Gambar 3.9 Perancangan Antarmuka Halaman Sub menu Operator Surga

Komponen-komponen yang ada Gambar 3.9 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. *ListView* yang menampilkan sub menu-sub menu dari modul operator suara warga.
c. Satu buah ikon dan *text* untuk setiap sub menu yang berfungsi penamaan dari sub menu tersebut.

3.2.5.4 Perancangan Antarmuka Manajemen Aduan

Antarmuka manajemen aduan merupakan antarmuka yang muncul ketika sub menu manajemen aduan dipilih melalui antarmuka sub menu operator surga. Pada antarmuka ini terdapat empat *tab* jika *login* sebagai petugas tingkat satu dan terdapat tiga *tab* jika *login* sebagai petugas tingkat dua. Tab untuk petugas tingkat satu adalah aduan belum diklasifikasi, aduan sudah diklasifikasi, aduan dikembalikan, dan seluruh aduan. Sedangkan untuk tab untuk petugas tingkat dua adalah aduan belum diklasifikasi, aduan sudah diklasifikasi, dan aduan dikembalikan. Halaman tersebut ditunjukkan pada Gambar 3.10.



Gambar 3.10 Perancangan Antarmuka Halaman Manajemen Aduan

Komponen-komponen yang ada Gambar 3.8 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.

b. *Tab* yang digunakan untuk menampilkan judul dari *list* aduan.
c. *ListView* yang digunakan untuk menampilkan aduan.
d. Tiap aduan terdari dari enam teks yang digunakan untuk menampilkan id aduan, tanggal aduan, topik aduan, departemen aduan, status aduan, dan prioritas aduan.

3.2.5.5 Perancangan Antarmuka Detail Aduan

Antarmuka detail aduan merupakan antarmuka yang muncul ketika salah satu aduan dipilih melalui antarmuka manajemen aduan. Halaman tersebut ditunjukkan pada Gambar 3.11.

Gambar 3.11 Perancangan Antarmuka Halaman Detail Aduan

Komponen-komponen yang ada Gambar 3.11 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.

b. Empat buah *textView* yang digunakan untuk menampilkan id aduan, topik aduan, nama pengirim, dan dapartemen aduan.
c. Empat buah *button* yang digunakan untuk navigasi ke menu lain. Menu lainnya adalah balas aduan, lihat peta, tindak lanjuti aduan, dan lihat lampiran.
d. Ikon dan *textView* yang digunakan untuk menampilkan waktu, dikirim melalui, prioritas, dan status aduan.
e. *ListView* yang digunakan untuk menampilkan isi dari aduan dan juga balasan dari petugas.
f. Setiap isi dari sebuah *list* di dalam *listView* terdapat nama pengirim aduan atau petugas yang menjawab aduan, waktu, dan isi dari aduan.

3.2.5.6 Perancangan Antarmuka Jawab Aduan

Antarmuka jawab aduan merupakan antarmuka yang muncul ketika tombol jawab aduan dipilih di detail aduan. Pada antarmuka ini terdapat empat sub menu antarmuka ini digunakan untuk menjawab aduan. Halaman tersebut ditunjukkan pada Gambar 3.12.

Gambar 3.12 Perancangan Antarmuka Jawab Aduan

Komponen-komponen yang ada Gambar 3.12 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. *ListView* yang digunakan untuk menampilkan isi dari aduan dan juga balasan dari petugas.
c. Setiap isi dari sebuah list di dalam *listView* terdapat nama pengirim aduan atau petugas yang menjawab aduan, waktu, dan isi dari aduan.
d. Dua *radio button* yang akan dipilih untuk membalas aduan tersebut melalui situs atau pesan singkat. *Radio button* ini akan hilang jika aduan berasal dari situs.
e. *EditText* yang digunakan untuk memasukan input balasan aduan.
f. *Button* balas yang digunakan untuk memasukan balasan tersebut ke dalam basis data.

3.2.5.7 Perancangan Antarmuka Tindak Lanjut Aduan

Antarmuka tindak lanjut aduan merupakan antarmuka yang muncul ketika tombol tindak lanjuti dipilih di detail aduan. Untuk petugas tingkat satu, antarmuka ini digunakan untuk memilih petugas yang akan menindak lanjuti aduan, mengubah prioritas, dan mengubah status aduan. Halaman ini ditunjukkan pada Gambar 3.1311.

Gambar 3.13. Perancangan Antarmuka Tindak Lanjut Aduan

Komponen-komponen yang ada Gambar 3.11 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. Ikon dan *TextView* berfungsi untuk menampilkan nama petugas, prioritas, dan status dari aduan. Ikon jika dipilih akan menampilkan *list* dari nama petugas, prioritas, dan status yang dapat dipilih. Untuk petugas tingkat dua, ikon dan *textView* nama petugas akan dihilangkan.

c. Tiga *button* yang digunakan untuk *update* aduan, menjadikan aduan tersebut menjadi *spam,* dan mengembalikan aduan tersebut ke petugas tingkat satu. Untuk petugas tingkat satu, *button* kembalikan akan dihilangkan.

3.2.5.8 Perancangan Antarmuka Peta

Antarmuka peta merupakan antarmuka yang muncul ketika tombol lihat peta dipilih di detail aduan. Antarmuka ini digunakan untuk menampilkan lokasi peta dari aduan yang dilaporkan. Halaman tersebut ditunjukkan pada Gambar 3.14.

Gambar 3.14 Perancangan Antarmuka Peta

Komponen-komponen yang ada Gambar 3.12 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.

b. *Fragment* Google Maps yang digunakan untuk menampilkan peta.

3.2.5.9 Perancangan Antarmuka Lihat Lampiran

Antarmuka lihat lampiran merupakan antarmuka yang muncul ketika tombol lihat lampiran dipilih di detail aduan. Antarmuka ini digunakan melihat *list* lampiran yang dilampirkan. Halaman tersebut ditunjukkan pada Gambar 3.15.

Gambar 3.15 Perancangan Antarmuka Lihat Lampiran

Komponen-komponen yang ada Gambar 3.13 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. *ListView* yang digunakan untuk menampilkan *list* aduan.
c. Dalam satu lampiran terdiri atas tiga *TextView* yang menampilkan tipe *file*, tanggal, dan nama *file*.

3.2.5.10 Perancangan Antarmuka Lihat Gambar

Antarmuka lihat gambar merupakan antarmuka yang muncul ketika tombol ‘lihat’ dipilih pada antarmuka ‘lihat

lampiran'. Antarmuka ini digunakan melihat gambar. Halaman tersebut ditunjukkan pada Gambar 3.16.

Gambar 3.16. Perancangan Antarmuka Peta

Komponen-komponen yang ada Gambar 3.16 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. *ImageView* yang menampilkan lampiran yang bertipe gambar.

3.2.5.11 Perancangan Antarmuka List Validasi SMS

Antarmuka *list* validasi sms merupakan antarmuka yang muncul ketika sub menu validasi sms dipilih di sub menu Operator Surga. Antarmuka ini digunakan melihat kumpulan sms yang tidak valid. Halaman tersebut ditunjukkan pada Gambar 3.17.

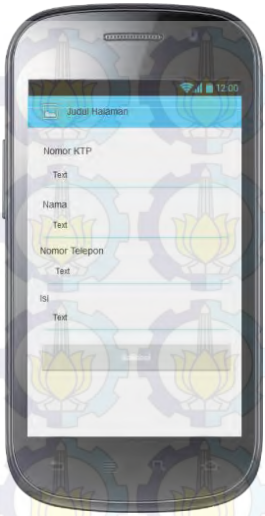
Gambar 3.17 Perancangan Antarmuka List Validasi SMS

Komponen-komponen yang ada Gambar 3.17 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. *ListView* yang berfungsi untuk menampilkan list dari sms yang tidak valid.
c. Satu sms tidak valid di dalam list tersebut terdapat empat *TextView* yaitu berfungsi untuk menampilkan id, nomor pengirim, isi sms, dan tanggal pengiriman.

3.2.5.12 Perancangan Antarmuka Validasi SMS

Antarmuka validasi sms merupakan antarmuka yang muncul ketika salah satu sms tidak valid tersebut dipilih divalidasi sms. Antarmuka ini digunakan melihat validasi sms yang tidak valid. Halaman tersebut ditunjukkan pada Gambar 3.18.



Gambar 3.18 Perancangan Antarmuka Validasi SMS

Komponen-komponen yang ada Gambar 3.18 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. Terdapat empat *editText* yang berfungsi untuk memasukkan input dari pengguna. Inputnya berupa nomor ktp, nama, nomor telepon, dan isi.
c. *Button* validasi yang digunakan untuk mengubah sms yang tidak valid tersebut menjadi aduan.

3.2.5.13 Perancangan Antarmuka SMS

Antarmuka sms aduan merupakan antarmuka yang muncul ketika sub menu validasi sms dipilih di sub menu operator surga. Antarmuka ini digunakan mengirim sms kepada petugas. Halaman tersebut ditunjukkan pada Gambar 3.19.

Gambar 3.19 Perancangan Antarmuka SMS

Komponen-komponen yang ada Gambar 3.16 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. Dua *EditText* yang satu berfungsi untuk memilih dinas mana yang akan dikirim sms dan yang kedua untuk menulis pesan yang akan dikirim.
c. Button kirim sms digunakan untuk mengirim sms tersebut ke petugas terkait.

3.2.5.14 Perancangan Antarmuka Laporan

Antarmuka laporan merupakan antarmuka yang muncul ketika sub menu laporan dipilih di sub menu operator surga. Antarmuka ini digunakan mengunduh laporan aduan. Halaman tersebut ditunjukkan pada Gambar 3.20.

Gambar 3.20 Perancangan Antarmuka Laporan

Komponen-komponen yang ada Gambar 3.20 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. *EditText* yang digunakan untuk memilih tahun laporan yang akan diunduh.
c. *Button download* yang digunakan untuk mengunduh laporan berdasarkan tahun yang dipilih.

BAB IV
IMPLEMENTASI PERANGKAT LUNAK

Proses implementasi sistem ini dilakukan setelah melewati proses analisis dan perancangan perangkat lunak. Dalam Bab ini akan dibahas mengenai algoritma, *pseudocode,* diagram alur serta implementasi dari perancangan antarmuka yang terdapat dalam perangkat lunak sebagaimana telah dibahas pada Bab III.

4.1 Lingkungan Implementasi

Dalam merancang dan mengimplementasikan perangkat lunak ini, digunakan beberapa perangkat pendukung sebagai berikut.

4.1.1 Lingkungan Implementasi Perangkat Keras

Spesifikasi perangkat keras yang digunakan pada lingkungan pengembangan perangkat lunak ini adalah sebagai berikut:

- Komputer Laptop Asus N-43SL
  - Windows 7 Home Premium 64-bit,
  - Prosesor Intel® Core(TM) i5-2430M CPU @ 2.40GHz, dan
  - RAM 8.00 GB.
- Perangkat Bergerak Asus Zefone 5
  - Sistem Operasi: Android v4.4 (Kitkat),
  - CPU: Dual core 2GHz Intel Atom Z2580
  - Memory internal: 16 GB,
  - RAM: 2.00 GB,
  - Speed: HSDPA, 14.4 Mbps; HSUPA, 5.76 Mbps, dan
  - WLAN: Wi-Fi 802.11 b/g/n, Wi-Fi hotspot

4.1.2 Lingkungan Implementasi Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan dalam pengembangan sistem adalah sebagai berikut:

- Microsoft Windows 7 *Home Premium* sebagai sistem operasi
- Eclipse Juno v4.2.1 sebagai IDE untuk mengimplementasikan modul aplikasi *mobile*
- MySQL sebagai basis data *server*
- StarUML dan www.draw.io untuk merancang diagram sistem

4.2 Implementasi Proses Modularitas

Pada subbab ini akan dijelaskan mengenai implementasi proses modularitas yang digunakan untuk membuka aplikasi yang mengimplementasi *interface* dari aplikasi utama. Proses ini dibagi menjadi dua yaitu proses pembuatan modul aplikasi dan proses memuat modul-modul di aplikasi utama.

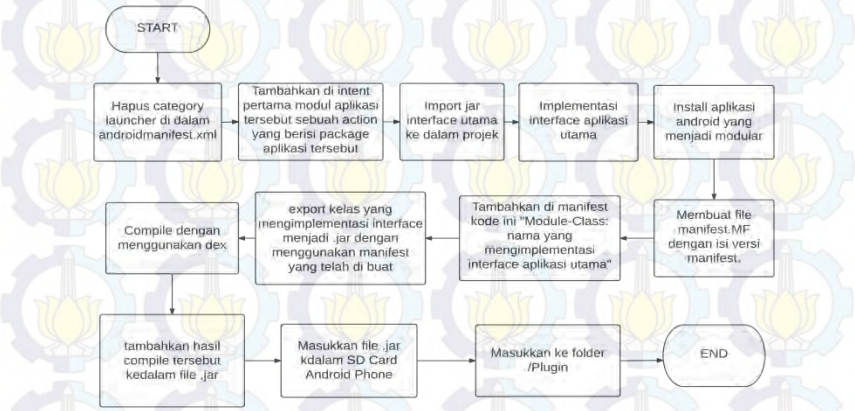
4.2.1 Implementasi Proses Pembuatan Modul Aplikasi

Implementasi proses pembuatan modul aplikasi diimplementasikan dengan cara menghapus *category launcher* di *androidmanifest.xml* pada modul aplikasi. Kemudian tambahkan sebuah *action* di *intent* pertama dari modul aplikasi yang berisi *package* utama dari aplikasi tersebut ditambah dengan ".MAIN". Impor *file jar interface* aplikasi utama ke dalam *project* modul aplikasi. Implementasikan *interface* tersebut dengan mengisi nama modul, nama *intent action* yang di dapat dari *androidmanifest,* dan *role* yang boleh membuka modul tersebut. Pasang modul aplikasi tersebut ke dalam perangkat Android. Buat *file manifest.mf* yang berisi versi *manifest* tersebut ditambah dengan Kode Sumber 4.1. Ekspor kelas yang mengimplementasi *interface* aplikasi utama dengan menggunakan *manifest* yang telah dibuat menjadi *file jar*.

Compile file jar tersebut menggunakan *dex*. Hasil dari *compile* adalah sebuah *file* yang berekstensi "*.dex"*. Masukkan *file* yang berekstensi "*.dex"* ke dalam *file jar.* Salin *file jar* tersebut ke perangkat Android ke dalam direktori yang sudah ditentukan. Proses ini dapat digambarkan pada Gambar 4.1.

```
Module-Class: "tempat kelas yang
mengimplementasikan interface aplikasi
utama"
```

Kode Sumber 4.1 Kode yang Tambah di Manifest



Gambar 4.1 Diagram Alir Pembuatan Modul Aplikasi

4.2.2 Implementasi Proses Memuat Modul di Aplikasi Utama

Implementasi proses memuat modul aplikasi digunakan untuk memuat modul-modul aplikasi untuk dapat dibuka oleh aplikasi utama. Modul aplikasi ini juga harus mengimplementasikan *interface* dari aplikasi utama dan modul aplikasi tersebut sudah terpasang di perangkat Android. Hasil ekspor implementasi *interface* tersebut harus di *copy* ke *directory* perangkat Android yang sudah ditentukan. Proses memuat modul di aplikasi ini membaca setiap *file* yang berekstensi ".jar" yang diambil dari direktori yang ditentukan. *File–file* ini lalu di-*cast* menjadi kelas *jarFile.*Setelah itu kelas tersebut diambil atribut *manifest* yang memiliki *value* "Module-Class" dan disimpan ke dalam sebuah *string*. Lalu kita *load file jar* ini dengan menggunakan kelas *dexClassLoader.* Setelah *load*, kelas tersebut di-*cast* ke *interface.* Setelah *cast,* maka kelas ini dimasukkan ke dalam *list* yang menyimpan semua modul aplikasi ini. Proses ini dapat ditunjukkan pada Kode Sumber 4.21.

```
1  ModuleLoader() {
2    file ← directory "/plugin/"
3    List<PluginModel> plugins ← empty list
4    file[] listfile  ← get file ends with
5  .jar
6      FOR i = 1 to listfile length{
7      jarFile manifest ← listfile[i]
8      String className ← get manifest
9  attribute with value "Module Class"
10     Instance classLoader using dexLoader
11     Class<?> claz ← load class using
12 classLoader with className
13     Plugin tempPlugin ← cast
14 claz.newInstance() to plugin
```

```
    Plugins.add(new
PluginModel(tempPlugin.getNamaModul(),
 tempPlugin.getActionIntentName(),
 tempPlugin.getRole())
 }
Return plugins
}
```

Kode Sumber 4.2 Implementasi Proses Memuat Modul di Aplikasi Utama

4.3 Implementasi Antarmuka

Pada subbab ini dijelaskan implementasi tampilan antarmuka yang telah dibahas pada subbab 3.2.5.

4.3.1 Implementasi Antarmuka Login

Antarmuka *login* merupakan antarmuka yang berfungsi sebagai pembuktian keaslian untuk masuk ke aplikasi. Pengguna harus memasukan *username* dan *password* yang sudah ada di dalam sistem. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.3 merupakan bentuk tampilan dari antarmuka *login*.

Gambar 4.2 Implementasi Antarmuka *Login*

4.3.2 Implementasi Antarmuka Menu

Antarmuka menu merupakan antarmuka pertama yang muncul jika pengguna berhasil dibuktikan keasliannya ke dalam sistem. Pada antarmuka ini pengguna dapat melihat menu-menu yang ada dalam aplikasi ini. Menu-menu yang ditampilkan merupakan modul-modul yang ada dalam aplikasi. Tidak semua modul akan terlihat di menu ini. Modul yang terlihat tergantung pada pengguna yang telah login. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.3 merupakan tampilan antarmuka menu.

Gambar 4.3. Implementasi Antarmuka Menu

4.3.3 Implementasi Antarmuka Sub Menu

Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan *list* operator surga di antarmuka menu. Pada antarmuka ini pengguna dapat melihat sub menu yang ada dalam aplikasi ini. Sub yang ditampilkan akan berbeda tergantung kepada petugas mana yang telah login. Jika petugas tingkat satu yang *login* mana akan terdapat empat sub menu sedangkan untuk petugas tingkat dua hanya terdapat satu sub menu. Sub menu untuk petugas tingkat satu adalah manajemen aduan, validasi sms, sms, dan laporan. Untuk petugas tingkat dua sub menunya hanya manejemen aduan. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.4 merupakan tampilan antarmuka sub menu.

Gambar 4.4. Implementasi Antarmuka Sub menu

4.3.4 Implementasi Antarmuka Manajemen Aduan

Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan *list* manajemen aduan di antarmuka sub menu. Pada antarmuka ini pengguna dapat melihat aduan-aduan yang ada dalam sistem. Untuk petugas tingkat satu, aduan-aduan tersebut dibagi menjadi 4 tab. Tab tersebut adalah aduan belum diklasifikasi, aduan diklasifikasi, aduan dikembalikan, dan seluruh aduan sedangkan untuk petugas tingkat dua memiliki 3 tab dan aduannya merupakan aduan yang sudah di arahkan ke SKPD (Satuan Kerja Pemerintah Daerah) terkait. Tab tersebut adalah aduan belum terjawab, aduan terjawab, dan seluruh aduan. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.5 merupakan tampilan antarmuka manajemen aduan.

Gambar 4.5 Antarmuka Manajemen Aduan

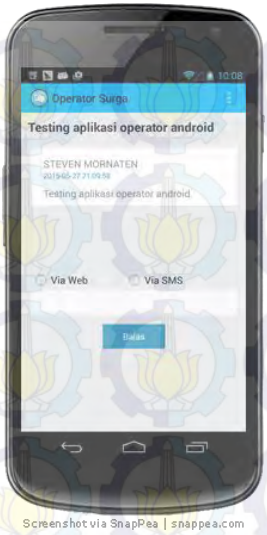
4.3.5 Implementasi Antarmuka Detail Aduan

Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan salah satu aduan di antarmuka manajemen aduan. Pada antarmuka ini pengguna dapat melihat satu aduan secara detail. Hal-hal yang dapat dilihat adalah id aduan, topik aduan, nama pengirim aduan, departemen yang terkait dengan aduan, waktu, dikirim melalui, status, prioritas, dan sebuah *list* yang berisi isi aduan dan balasan (jika sudah dibalas). Terdapat empat *button* di antarmuka ini yang berfungsi untuk navigasi ke antarmuka yang lain. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.6 merupakan tampilan antarmuka detail aduan.

Gambar 4.6. Implementasi Antarmuka Detail Aduan

4.3.6 Implementasi Antarmuka Jawab Aduan

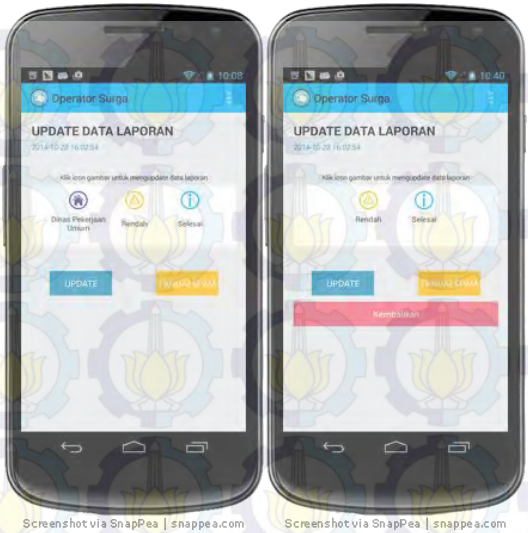
Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan *button* jawab aduan di antarmuka detail aduan. Antarmuka ini berfungsi untuk menjawab aduan. Aduan dapat dijawab menggunakan *via* sms jika pengirim aduan tersebut menyantumkan nomor telepon. Jika tidak, sistem akan secara otomatis menggunakan *via* situs untuk membalas aduan tersebut. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.7 merupakan tampilan antarmuka jawaba aduan.



Gambar 4.7. Implementasi Antarmuka Jawab Aduan

4.3.7 Implementasi Antarmuka Tindak Lanjut Aduan

Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan *button* tindak lanjuti aduan di antarmuka detail aduan. Antarmuka ini berfungsi untuk memilih SKPD untuk menanggapi aduan tersebut, mengubah status aduan, mengubah prioritas aduan, dan menandai aduan sebagai spam jika yang *login* merupakan petugas tingkat satu. Untuk petugas tingkat dua, antarmuka ini berfungsi untuk mengubah status, mengubah prioritas, menandai aduan sebagai *spam*, dan mengembalikan aduan. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.8 merupakan tampilan antarmuka tindak lanjut aduan.



Gambar 4.8. Implementasi Antarmuka Tindak Lanjut Aduan

4.3.8 Implementasi Antarmuka Peta

Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan *button* lihat peta di antarmuka detail aduan. Antarmuka ini berfungsi untuk menampilkan lokasi peta dari aduan yang dilaporkan. Antarmuka ini juga bisa memperbesar atau memperkecil peta. Antarmuka ini diimplementasikan menggunakan kode XML, Java, dan Google Map API v2. Gambar 4.9 merupakan tampilan antarmuka peta.

Gambar 4.9. Implementasi Antarmuka Peta

4.3.9 Implementasi Antarmuka Lihat Lampiran

Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan *button* lihat lampiran di antarmuka detail aduan. Antarmuka ini berfungsi untuk menampilkan lampiran-lampiran yang disertakan. Lampiran-lampiran ini bisa diunduh ke dalam perangkat Android atau dapat dilihat jika lampiran tersebut merupakan gambar. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Pada gambar 4.10 merupakan tampilan antarmuka lihat lampiran.

Gambar 4.10. Implementasi Antarmuka Lihat Lampiran

4.3.10 Implementasi Antarmuka Lihat Gambar

Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan *button* lihat di antarmuka lihat lampiran. Antarmuka ini berfungsi untuk menampilkan lampiran yang berupa gambar. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.11 merupakan tampilan antarmuka lihat gambar.

Gambar 4.11. Implementasi Antarmuka Lihat Gambar

4.3.11 Implementasi Antarmuka *List* Validasi SMS

Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan *list* validasi sms di antarmuka sub menu. Antarmuka ini berfungsi untuk menampilkan sms-sms yang tidak dikirim sesuai format. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.12 merupakan tampilan antarmuka *list* validasi sms.

Gambar 4.12. Implementasi Antarmuka *List* Validasi SMS

4.3.12 Implementasi Antarmuka Validasi SMS

Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan salah satu sms yang tidak valid di *list* validasi sms. Antarmuka ini berfungsi untuk memasukkan sms tidak valid tersebut ke dalam sistem. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.13 merupakan tampilan antarmuka validasi sms.

Gambar 4.13. Implementasi Antarmuka Validasi SMS

4.3.13 Implementasi Antarmuka SMS

Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan sub menu sms di antarmuka sub menu. Antarmuka ini berfungsi untuk mengirim sms kepada petugas yang sudah terdaftar dalam sistem. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.14 merupakan tampilan antarmuka sms.

Gambar 4.14. Implementasi Antarmuka SMS

4.3.14 Implementasi Antarmuka Laporan

Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan sub menu laporan di antarmuka sub menu. Antarmuka ini berfungsi untuk mengunduh laporan berdasarkan tahun. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.15 merupakan tampilan antarmuka laporan.

Gambar 4.15. Implementasi Antarmuka Laporan

4.4 Implementasi *Query* Basis Data

Pada subbab ini akan dijelaskan mengenai implementasi *query* yang digunakan untuk mengakses *server*, *query* ini disimpan dalam suatu *web service* berbahasa PHP pada *server*. *Query* yang diimplementasikan dibuat menggunakan bahasa SQL.

4.4.1 Implementasi *Query* Menampilkan Jumlah Aduan

Query menampilkan jumlah aduan dibagi menjadi dua. Pertama adalah *query* jumlah aduan untuk petugas tingkat satu yang dimana *query* tersebut menampilkan data jumlah total aduan yang belum diklasifikasi, jumlah aduan yang sudah diklasifikasi, jumlah aduan yang dikembalikan, dan jumlah seluruh aduan. Kedua adalah *query* jumlah aduan untuk

petugas tingkat dua yang dimana *query* tersebut menampilkan jumlah aduan yang belum terjawab, jumlah aduan yang sudah terjawab, dan jumlah seluruh aduan.

Query untuk aduan yang belum diklasifikasi diimplementasikan dengan cara menghitung jumlah aduan yang memiliki atribut info yang nilainya nol dan tidak termasuk *spam*. Imlementasi ini ditunjukkan pada Kode Sumber 4.3.

```
1 | select count(a.id_aduan) as jumlah_aduan
2 | from prioritas p, aduan a LEFT JOIN
3 | departemen d ON a.departemen
4 | =d.id_departemen
5 | where a.info='0' and a.spam='0' and
6 | a.prioritas=p.id_prioritas
```

Kode Sumber 4.3 Implementasi *Query* Menampilkan Jumlah Aduan Belum Terklasifikasi

Query aduan yang sudah diklasifikasi diimplementasikan dengan cara menghitung jumlah aduan yang memiliki atribut info yang nilainya satu dan tidak merupakan *spam*. Implementasi ini ditunjukkan pada Kode Sumber 4.4.

```
1 | select count(a.id_aduan) as jumlah_aduan
2 | from prioritas p, aduan a LEFT JOIN
3 | departemen d ON a.departemen
4 | =d.id_departemen
5 | where a.info='1' and a.spam='0' and
6 | a.prioritas=p.id_prioritas
```

Kode Sumber 4.4 Implementasi *Query* Menampilkan Jumlah Aduan Terklasifikasi

Query untuk aduan yang sudah dikembalikan diimplementasikan dengan cara menghitung jumlah aduan yang memiliki atribut info yang nilainya dua dan tidak merupakan *spam.* Implementasi *query* ini ditunjukkan pada Kode Sumber 4.5.

```
1  select count(a.id_aduan)as jumlah_aduan
2  from prioritas p, aduan a LEFT JOIN
3  departemen d ON a.departemen
4  =d.id_departemen
5  where a.spam='0' and a.info='2' and
6  a.prioritas=p.id_prioritas
```

Kode Sumber 4.5 Implementasi *Query* Menampilkan Jumlah Aduan DiKembalikan

Query untuk seluruh aduan diimplementasikan dengan cara menghitung jumlah aduan yang ada dan tidak merupakan *spam*. Implementasi *query* ini ditunjukkan pada Kode Sumber 4.6.

```
1  select count(a.id_aduan) as jumlah_aduan
2  from prioritas p, aduan a LEFT JOIN
3  departemen d ON a.departemen =
4  d.id_departemen where a.spam='0' and
5  a.prioritas=p.id_prioritas
```

Kode Sumber 4.6 Implementasi *Query* Menampilkan Jumlah Seluruh Aduan

Query untuk aduan yang belum terjawab diimplementasikan dengan cara menghitung aduan yang memiliki status tidak sama dengan empat, tidak merupakan *spam,* terkait pada SKPD yang telah *login*. Implementasi *query* ini ditunjukkan pada Kode Sumber 4.7

```
1 | select count(*) as jumlahBelumTerjawab
2 | from prioritas p, aduan a,departemen d
3 | where a.spam='0' and a.status<>'4' and
4 | a.departemen='".$departemen."' and
5 | a.departemen = d.id_departemen and
6 | a.prioritas=p.id_prioritas
```

Kode Sumber 4.7 Implementasi *Query* Menampilkan Jumlah Aduan Belum Terjawab

Query untuk aduan yang sudah terjawab diimplementasikan dengan cara menghitung aduan yang memiliki status sama dengan empat, tidak merupakan *spam,* dan terkait pada SKPD yang telah *login*. Implementasi *query* ini ditunjukkan pada Kode Sumber 4.8.

```
1 | select count(*) as jumlahTerjawab from
2 | prioritas p, aduan a,departemen d
3 | where a.spam='0' and a.status='4' and
4 | a.departemen='".$departemen."' and
5 | a.departemen = d.id_departemen and
6 | a.prioritas=p.id_prioritas
```

Kode Sumber 4.8 Implementasi *Query* Menampilkan Jumlah Aduan Terjawab

Query jumlah seluruh aduan diimplementasikan dengan cara menghitung aduan yang telah ada pada sistem, tidak merupakan *spam,* dan terkait pada SKPD yang telah *login.* Implementasi *query* ini ditunjukkan pada Kode Sumber 4.9.

```
1 | select count(*) as jumlahSemuaAduan from
2 | prioritas p, aduan a,departemen d where
```

```
3 | a.spam='0' and
4 | a.departemen='".$departemen."' and
5 | a.departemen = d.id_departemen and
6 | a.prioritas=p.id_prioritas
```

Kode Sumber 4.9 Implementasi *Query* Menampilkan Jumlah Seluruh Aduan

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android lalu ditampilkan di antarmuka manajemen aduan pada Gambar 4.5.

4.4.2 Implementasi *Query* Menampilkan Aduan

Query menampilkan aduan dibagi menjadi dua. Pertama adalah *query* aduan untuk petugas tingkat satu yang dimana *query* tersebut menampilkan data aduan yang belum diklasifikasi, aduan yang sudah diklasifikasi, aduan yang dikembalikan, dan seluruh aduan. Kedua adalah *query* jumlah aduan untuk petugas tingkat dua yang dimana *query* tersebut menampilkan aduan yang belum terjawab, aduan yang sudah terjawab, dan seluruh aduan. Untuk *query* petugas tingkat dua ini merupakan *query* berdasarkan SKPD petugas tersebut. Masing-masing *query* ini diurutkan berdasarkan waktu terakhir.

Query untuk aduan yang belum diklasifikasi diimplementasikan dengan cara memilih aduan yang memiliki atribut info yang nilainya nol dan tidak termasuk *spam*. Imlementasi ini ditunjukkan pada Kode Sumber 4.10

```
select
a.status,a.id_aduan,d.nama_departemen,
a.waktu,
a.topik,p.nama_prioritas,a.isi
from prioritas p, aduan a LEFT JOIN
departemen d ON a.departemen =
d.id_departemen
where a.info='0' and a.spam='0' and
a.prioritas=p.id_prioritas
order by a.waktu desc
```

Kode Sumber 4.10 Implementasi *Query* Menampilkan Aduan Belum Terklasifikasi

Query aduan yang sudah diklasifikasi diimplementasikan dengan cara memilih aduan yang memiliki atribut info yang nilainya satu dan tidak merupakan *spam*. Implementasi ini ditunjukkan pada Kode Sumber 4.11.

```
select
a.status,a.id_aduan,d.nama_departemen,a.waktu
,a.topik,p.nama_prioritas,a.isi
from prioritas p, aduan a LEFT JOIN
departemen d ON a.departemen =
d.id_departemen
where a.info='1' and a.spam='0' and
a.prioritas=p.id_prioritas
order by a.waktu desc
```

Kode Sumber 4.11 Implementasi *Query* Menampilkan Aduan Terklasifikasi

Query untuk aduan yang sudah dikembalikan diimplementasikan dengan cara memilih aduan yang memiliki atribut info yang nilainya dua dan tidak merupakan *spam.* Implementasi *query* ini ditunjukkan pada Kode Sumber 4.12.

```
select
a.status,a.id_aduan,d.nama_departemen,a.waktu
,a.topik,p.nama_prioritas,a.isi
from prioritas p, aduan a LEFT JOIN
departemen d ON a.departemen =
d.id_departemen
where a.spam='0' and a.info='2' and
a.prioritas=p.id_prioritas
order by a.waktu desc
```

Kode Sumber 4.12 Implementasi *Query* Menampilkan Aduan Dikembalikan

Query untuk seluruh aduan diimplementasikan dengan cara memilih aduan yang ada dan tidak merupakan *spam*. Implementasi *query* ini ditunjukkan pada Kode Sumber 4.13.

```
select a.status,a.id_aduan,
d.nama_departemen,a.waktu,a.topik,p.nama_prio
ritas,a.isi from prioritas p,
aduan a LEFT JOIN departemen d ON
 a.departemen = d.id_departemen where
a.spam='0' and a.prioritas=p.id_prioritas
order by a.waktu desc
```

Kode Sumber 4.13 Implementasi *Query* Menampilkan Seluruh Aduan

Query untuk aduan yang belum terjawab diimplementasikan dengan cara memilih aduan yang memiliki status tidak sama dengan empat, tidak merupakan *spam,* terkait pada SKPD yang telah *login*. Implementasi *query* ini ditunjukkan pada Kode Sumber 4.14.

```
select
a.id_aduan,d.nama_departemen,a.waktu,a.topik,
p.nama_prioritas,a.isi
from prioritas p, aduan a LEFT JOIN
departemen d ON a.departemen=d.id_departemen
where a.info<>'2' and a.spam='0' and
a.status<>'4' and a.petugas='".$id_petugas."'
and a.prioritas=p.id_prioritas
order by a.waktu desc;
```

Kode Sumber 4.14 Implementasi *Query* Menampilkan Aduan Belum Terjawab

Query untuk aduan yang sudah terjawab diimplementasikan dengan cara memilih aduan yang memiliki status sama dengan empat, tidak merupakan *spam,* dan terkait pada SKPD yang telah *login*. Implementasi *query* ini ditunjukkan pada Kode Sumber 4.15.

```
select
a.id_aduan,d.nama_departemen,a.waktu,a.topik,
p.nama_prioritas ,a.isi
from prioritas p, aduan a LEFT JOIN
departemen d ON a.departemen=d.id_departemen
where a.info<>'2' and a.spam='0' and
a.status='4' and a.petugas='".$id_petugas."'
and a.prioritas=p.id_prioritas order by
a.waktu desc;
```

Kode Sumber 4.15 Implementasi *Query* Menampilkan Aduan Terjawab

Query jumlah seluruh aduan diimplementasikan dengan cara memilih aduan yang telah ada pada sistem, tidak merupakan *spam,* dan terkait pada SKPD yang telah *login.* Implementasi *query* ini ditunjukkan pada Kode Sumber 4.16.

```
select
a.id_aduan,d.nama_departemen,a.waktu,a.topik,
p.nama_prioritas,a.isi
from prioritas p, aduan a LEFT JOIN
departemen d ON a.departemen=d.id_departemen
where a.info<>'2' and a.spam='0' and
a.petugas='".$id_petugas."' and
a.prioritas=p.id_prioritas order by a.waktu
desc;
```

Kode Sumber 4.16 Implementasi *Query* Menampilkan Seluruh Aduan

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android lalu ditampilkan di antarmuka manajemen aduan menggunakan *listview* pada Gambar 4.5.

4.4.3 Implementasi *Query* Menampilkan Detail Aduan

Query menampilkan detail aduan adalah *query* yang berfungsi untuk menampilkan data-data secara keseluruhan suatu aduan. Data-data yang ditampilkan yaitu id aduan, topik aduan, nama pengirim aduan, nama departemen aduan, waktu, dikirim melalui, status aduan, prioritas aduan, bujur, lintang, dan balasan dari petugas. Implementasi *query* tersebut ditunjukkan pada Kode Sumber 4.17.

```
select
a.latitude,a.longitude,a.id_aduan,a.topik,a.n
ama,d.nama_departemen,a.waktu,a.via_sms,s.nam
a_status,p.nama_prioritas
from aduan a LEFT JOIN departemen d ON
a.departemen = d.id_departemen,prioritas
p,status s where a.id_aduan='".$id."' and

```

```
9 a.status = s.id_status and a.prioritas =
  p.id_prioritas
```

Kode Sumber 4.17 Implementasi *Query* Menampilkan Detail Aduan

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android lalu ditampilkan di antarmuka detail aduan pada Gambar 4.6.

4.4.4 Implementasi *Query* Menampilkan Isi dan Balasan Aduan

Query menampilkan isi dan balasan aduan adalah *query* yang berfungsi untuk menampilkan data-data isi aduan, balasan aduan, nama pengirim aduan, nama petugas, waktu pengiriman aduan, dan waktu balas aduan. Implementasi *query* tersebut ditunjukkan pada Kode Sumber 4.18.

```
1 select
2 da.petugas_detail,p.nama_petugas,d.nama_depar
3 temen,a.nama,da.waktu_detail,da.isi_detail
4 from detail_aduan da LEFT JOIN petugas p ON
5 da.petugas_detail=p.id_petugas LEFT JOIN
6 departemen d ON p.departemen=d.id_departemen,
7 aduan a where da.aduan='".$id."' and da.aduan
8 = a.id_aduan
```

Kode Sumber 4.18 Implementasi *Query* Menampilkan Detail Aduan

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android lalu ditampilkan di antarmuka jawab aduan pada Gambar 4.7.

4.4.5 Implementasi *Query* Menampilkan Data petugas, prioritas, dan status Aduan

Query menampilkan data petugas, prioritas, dan status aduan berfungsi untuk memilih petugas yang akan menanggapi aduan, mengubah status aduan, dan prioritas aduan. *Query* menampilkan data petugas diimplementasikan dengan cara mengambil nama petugas dan *id* petugas yang dimana petugas memiliki *role* tiga. Implementasi ini ditunjukkan pada Kode Sumber 4.19.

```
1 | select id_petugas,nama_petugas from petugas
2 | where role='3' and nama_petugas IS NOT NULL
```

Kode Sumber 4.19 Implementasi *Query* Menampilkan prioritas

Query menampilkan data prioritas diimplementasikan dengan cara mengambil nama prioritas dan *id* prioritas yang dimana petugas memiliki *role* tiga. Implementasi ini ditunjukkan pada Kode Sumber 4.20

```
1 | select id_prioritas,nama_prioritas from
2 | prioritas
```

Kode Sumber 4.20 Implementasi *Query* Menampilkan prioritas

Query menampilkan data status diimplementasikan dengan cara mengambil nama status dan *id* status yang dimana petugas memiliki *role* tiga. Implementasi ini ditunjukkan pada Kode Sumber 4.21

```
1 | select id_status,nama_status from status
```

Kode Sumber 4.21 Implementasi *Query* Menampilkan prioritas

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android lalu ditampilkan di antarmuka tindak lanjut aduan pada Gambar 4.8.

4.4.6 Implementasi *Query* Mengubah Aduan

Query mengubah aduan berfungsi untuk mengubah aduan jika merupakan spam, aduan dikembalikan, mengubah status aduan, prioritas aduan, dan petugas yang menanggapi aduan. *Query* mengubah aduan menjadi *spam* diimplementasikan dengan cara mengeset atribut spam menjadi satu. Implementasi ini dapat ditunjukkan pada Kode Sumber 4.22.

```
1 | update aduan set spam='1' where
2 | id_aduan='".$id."'
```

Kode Sumber 4.22 Implementasi *Query* Mengubah Aduan Menjadi Spam

Query mengembalikan aduan diimplementasikan dengan cara mengeset atribut info menjadi dua. Implementasi ini dapat ditunjukkan pada Kode Sumber 4.23.

```
1 | update aduan set info='2' where
2 | id_aduan='".$id."'
```

Kode Sumber 4.23 Implementasi *Query* Mengembalikan Aduan

Query mengubah prioritas, status, dan petugas diimplementasikan dengan cara mengeset prioritas, status, dan

petugas dengan masukkan dari pengguna. Implementasi dini ditunjukkan pada Kode Sumber 4.24.

```
"update aduan set info='1',
prioritas='".$prioritas."',
petugas='".$petugas."',
status='".$status."' where
id_aduan='".$id."'
```

Kode Sumber 4.24 Implementasi *Query* Mengubah Prioritas, Status, dan Petugas Aduan

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang diambil oleh aplikasi Android lalu ditampilkan di antarmuka tindak lanjut aduan pada Gambar 4.8.

4.4.7 Implementasi *Query* Menampilkan Lampiran

Query menampilkan lampiran berfungsi untuk menampilkan lampiran yang disertakan aduan. Data yang diambil yaitu waktu, tipe *file*, dan nama *file*. Implementasi *query* tersebut ditunjukkan pada Kode Sumber 4.1

```
select
da.waktu_detail,u.file_type,u.orig_name as
nama_file,u.path_upload from aduan
a,detail_aduan da,upload u
where a.id_aduan='".$id."' and
a.id_aduan=da.aduan and
da.id_detail_aduan=u.detail_aduan
```

Kode Sumber 4.25 Implementasi *Query* Menampilkan Lampiran

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android

lalu ditampilkan di antarmuka lihat lampiran dalam bentuk *listview* aduan pada Gambar 4.10.

4.4.8 Implementasi *Query* Menampilkan Data SMS Tidak Valid

Query menampilkan data sms tidak valid berfungsi untuk menampilkan sms-sms tidak sesuai dengan format yang sudah ditentukan. Data-data yang diambil adalah id, nomor pengirim, waktu pengiriman, dan isi aduan. Implementasi *query* tersebut ditunjukkan pada Kode Sumber 4.26.

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android lalu ditampilkan di antarmuka *list* validasi sms dengan bentuk *listview* pada Gambar 4.12.

```
1 | select id,nomor_pengirim,waktu,isi from
2 | sms_tidak_valid
```

Kode Sumber 4.26 Implementasi *Query* Menampilkan Data SMS Tidak Valid

4.4.9 Implementasi *Query* Menampilkan Nomor Telepon Petugas

Query menampilkan nomor telepon petugas berfungsi untuk menampilkan nomor telepon para petugas untuk mengirim sms dari sistem. Data-data yang diambil adalah nama petugas dan nomor telepon petugas. Implementasi *query* tersebut ditunjukkan pada Kode Sumber 4.27.

```
1 | select nama_petugas,no_hp_petugas from
2 | petugas where no_hp_petugas<>'null';
```

Kode Sumber 4.27 Implementasi *Query* Menampilkan Nomor Telepon Petugas

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android lalu ditampilkan di antarmuka sms pada Gambar 4.14.

4.4.10 Implementasi *Query* Menampilkan Tahun

Query menampilkan tahun berfungsi untuk menampilkan tahun yang ada dalam sistem. Tahun ini digunakan untuk mengambil data-data laporan pada tahun tersebut. Data yang diambil adalah tahun saja. Implementasi *query* tersebut ditunjukkan pada Kode Sumber 4.28.

```
1 | select distinct YEAR(waktu) as tahun from
2 | aduan;
```

Kode Sumber 4.28 Implementasi *Query* Menampilkan Tahun

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android lalu ditampilkan di antarmuka laporan pada Gambar 4.15.

4.4.11 Implementasi *Query* Mengambil Data Laporan

Query mengambil data laporan berfungsi untuk mengambil data-data yang diperlukan yang akan ditampilkan di *file* yang berekstensi "*.csv*" berdasarkan tahun yang dipilih. Data-data yang diperlukan adalah jumlah aduan masuk, jumlah aduan yang belum ditindak lanjut, aduan yang sudah ditindak

lanjuti, aduan yang sudah selesai, aduan yang berstatus *spam*, dan setiap aduan yang berada pada tahun yang dipilih.

Query aduan masuk dimplementasikan dengan cara menghitung jumlah aduan yang waktunya ada pada tahun yang dipilih. Implementasi ini ditunjukkan pada Kode Sumber 4.29.

```
1 | select count(*) as aduan_masuk from aduan
2 | where aduan.waktu >= '".$awal."' and
3 | aduan.waktu < '".$akhir."'
```

Kode Sumber 4.29 Implementasi *Query* Aduan Masuk

Query aduan belum ditindak lanjut dimplementasikan dengan cara memilih aduan yang waktunya ada pada tahun yang dipilih dan memiliki info tidak sama dengan satu. Implementasi ini ditunjukkan pada Kode Sumber 4.30.

```
1 | select count(*) as belum_ditindak_lanjut
2 | from aduan where info <> 1 and aduan.waktu
3 | >= '".$awal."' and aduan.waktu <
4 | '".$akhir."'
```

Kode Sumber 4.30 Implementasi *Query* Aduan Belum Ditindak Lanjut

Query aduan sudah ditindak lanjut dimplementasikan dengan cara memilih aduan yang waktunya ada pada tahun yang dipilih dan memiliki info sama dengan satu. Implementasi ini ditunjukkan pada Kode Sumber 4.31.

```
1 | select count(*) as sudah_ditindak_lanjut
2 | from aduan where info = 1 and aduan.waktu
3 | >= '".$awal."' and aduan.waktu <
4 | '".$akhir."'
```

Kode Sumber 4.31 Implementasi *Query* Aduan Sudah Ditindak Lanjut

Query aduan sudah selesai dimplementasikan dengan cara memilih aduan yang waktunya ada pada tahun yang dipilih dan memiliki status sama dengan empat. Implementasi ini ditunjukkan pada Kode Sumber 4.32.

```
1 | select count(*) as sudah_selesai from aduan
2 | where aduan.status = 4 and aduan.waktu >=
3 | '".$awal."' and aduan.waktu < '".$akhir."'
```

Kode Sumber 4.32 Implementasi *Query* Aduan Sudah Selesai

Query aduan sudah selesai dimplementasikan dengan cara memilih aduan yang waktunya ada pada tahun yang dipilih dan memiliki *spam* sama dengan satu. Implementasi ini ditunjukkan pada Kode Sumber 4.33.

```
1 | select count(*) as `sampah` from aduan
2 | where spam = 1 and aduan.waktu >=
3 | '".$awal."' and aduan.waktu < '".$akhir."'
```

Kode Sumber 4.33 Implementasi *Query* Aduan Spam

Query aduan berdasarkan tahun dimplementasikan dengan cara memilih aduan yang waktunya ada pada tahun yang dipilih. Implementasi ini ditunjukkan pada Kode Sumber 4.34.

```
 1 | select a.id_aduan, a.no_identitas
 2 | `nik_pengadu`, a.nama `nama_pengadu`,
 3 | a.waktu,
 4 | p.nama_petugas, d.nama_departemen, spam,
 5 | a.topik `topik_aduan`, a.isi `isi_aduan`
 6 | from aduan a
 7 | LEFT JOIN petugas p ON
 8 | a.petugas=p.id_petugas LEFT JOIN departemen
 9 | d ON a.departemen=d.id_departemen
10 | where a.waktu>='".$awal."' and
11 | a.waktu<'".$akhir."';
```

Kode Sumber 4.34 Implementasi *Query* Aduan Berdasarkan Tahun

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android lalu ditampilkan di antarmuka sms pada Gambar 4.15

4.5 Implementasi Aplikasi Operator Suara Warga

Pada subbab ini akan dijelaskan mengenai implementasi aplikasi pada kasus penggunaan ke dalam sebuah baris kode. Implementasi ini dilakukan dengan menggunakan bahasa native pembuatan aplikasi Android yakni bahasa Java.

4.5.1 Implementasi Proses Pengambilan dan Menerima Data *Web Service*

Web service digunakan untuk melakukan *query* ke dalam basis data dimana hasil *query* tersebut akan digunakan untuk menampilkan data-data dalam sistem. Dalam pertukaran data dari *web service* ke perangkat Android digunakan *method post* dan *json*. *Method post* digunakan untuk mengirim data dari Android ke *web service* sedangkan *json* digunakan untuk menerima data dari *web service* ke perangkat Android. Semua proses ini menggunakan kelas *AsyncTask* di dalam *method*

doInBackground() yang dimana kelas ini berjalan sendiri diluar *thread* aplikasi utama.

Implementasi proses pengiriman data menggunakan kelas *NameValuePair* untuk menyimpan data yang akan dikirm dan *jParser* digunakan untuk mengirim dan menerima data dari *web service*. Implementasi pengiriman data dari perangkat Android ke *web service* ditunjukkan pada Kode Sumber 4.35 sedangkan untuk penerimaan data dari *web service* ke perangkat Android ditunjukkan pada Kode Sumber 4.36.

```
1  PROTECTED String doInBackground(String
2  ...args){
3   Inisialisasi list BasicNameValuePair
4   Masukkan data berupa key dan value ke
5  BasicNameValuePair
6
7   Inisialisasi jParser
8   jParser.makeHttpRequest(destinationURL,
9   ”post”, list BasicNameValuePair);
10  return null
11 }
```

Kode Sumber 4.35 Implementasi Mengirim Data ke Web Service

```
1  getDataJson(){
2  Inisialisasi JSONArray berdasarkan TAG
3  Inisialisasi list untuk menampung data
4  For i=0 to panjang JSONArray
5   JSONObject ← JSONArray[i]
6   ambil data dari JSONObject berdasarkan TAG
7  lalu dimasukkan kedalam list.
8  End for
9
10 return list
11 }
```

Kode Sumber 4.36 Implementasi Pengambilan Data dari Web Service

4.5.2 Implementasi *Session* Aplikasi

Agar aplikasi dapat berjalan terus sesuai dengan kebutuhan ketika pengguna masih login, dibutuhkan *session* dalam aplikasi untuk menyimpan data pengguna. Untuk itu dibuat objek session dengan atribut disesuaikan dengan kebutuhan. Digunakan *SharedPreferences* untuk menyimpan *session* tersebut. Implementasi Konstruktor dari *session* dapat dilihat dalam Kode Sumber 4.37.

```
1  String ROLE ← "role"
2  String DEPARTEMEN ← "departemen"
3  String ID_PETUGAS ← "id_petugas"
4  String SMS_DINAS ← "sms_dinas"
5  String PREFS_NAME ← "session_operator"
6
7  Inisialisasi SharedPreferences
8
9  Public SessionManager(Context context){
10  SharedPreferences ←
11 context.getSharedPreferences(PREFS_NAME,
12 Context.MODE_PRIVATE)
   }
```

Kode Sumber 4.37 Atribut dan Konstruktor Session Aplikasi

Saat aplikasi berjalan dan memanggil layanan *web service,* dibutuhkan beberapa data penting yang disimpan dalam *session.* Untuk dapat mengakses data dalam *session* tersebut dibuat suatu fungsi di dalam kelas *session* tersebut.

4.5.3 Implementasi Proses *Login*

Proses *login* ini berfungsi untuk pembuktian keaslian pengguna melalui *username* dan *password* yang dimasukkan sesuai dengan data yang ada dalam sistem suara warga untuk masuk ke dalam sistem. Implementasi dari proses ini ditunjukkan dalam Kode Sumber 4.38.

```
Inisialisasi list BasicNameValuePair
Masukkan key dan value username dan password
ke basicNameValuePair

Inisialisasi jParser
jParser.makeHttpRequest(destination,”post”,
list BasicNameValuePair);
   if json.getInt(TAG_SUCCESS) == 1
     validUser ← 1
     Inisialisasi JSONArray berdasarkan TAG
     JSONObject ← JSONArray[0]
     role ← JSONObject.getString(TAG_ROLE)
     departemen ←
JSONObject.getString(TAG_DEPARTEMEN)
     id_petugas ←
JSONObject.getString(TAG_ID_PETUGAS)
   end if
   else
     validUser ← 0
   end if
  if validUser==1
   if role == 1
   simpanSession(role,departemen,
id_petugas,true)
   end if
   else
    simpanSession(role,departemen,
id_petugas,false)
   end if
   Inisialisasi intent activity menu
```

```
31      startActivity(activityMenu)
32    end if
```

Kode Sumber 4.38 Implementasi Proses Login

Jika *username dan password* sesuai, maka pengguna dapat masuk ke dalam aplikasi dan *session* disimpan sampai pengguna melakukan *logout* aplikasi meskipun aplikasi ditutup. Jika *username* dan *password* salah, maka akan muncul informasi peringatan. Untuk membuat *session* aplikasi, ditunjukkan dalam Kode Sumber 4.39.

```
1  Public void simpanSession(String role,
2  String departemen, String id_petugas,
3  boolean sms dinas){
4   SharedPreferences.Editor editor ←
5   session.edit()
6   editor.putString(this.ROLE,role)
7
8  editor.putString(this.DEPARTEMEN,departemen
9  )
10
11 editor.putString(this.ID_PETUGAS,id_petugas
12 )
13
14 editor.putBoolean(this.SMS_DINAS,sms_dinas)
15  editor.commit()
16 }
```

Kode Sumber 4.39 Implementasi Penyimpanan Session Aplikasi

4.5.4 Implementasi Menampilkan Aduan

Menampilkan aduan diimplementasikan dengan cara mengambil seluruh aduan berdasarkan jenis-jenis aduan yang

ingin di tampilkan. Sebagai contoh jika petugas tingkat satu yang telah login ke dalam sistem, maka jenis-jenis aduan yang di ambil adalah aduan yang belum diklasifikasi, aduan yang terklasifikasi, aduan yang dikembalikan, dan seluruh aduan. Berikut Implementasi dari mengambil data dari *web service* yang ditunjukkan pada Kode Sumber 4.40 dan menampilkan data dari *web service* yang ditunjukkan pada Kode Sumber 4.41.

```
1  getDataJson(){
2  Inisialisasi JSONArray TAG_ADUAN
3  Inisialisasi list aduan
4  For i=0 to panjang JSONArray
5   JSONObject ← JSONArray[i]
6   ambil data dari JSONObject TAG ADUAN
7   masukkan ke dalam list
8  end for
9  return list
```

Kode Sumber 4.40 Implementasi Pengambilan Data Aduan

```
1  listSearch ← listAduan
2  inisialisasi
3  listAdapter(activity,listSearch)
4  listview.setAdapter(listAdapter)
   set onItemclickListener di listview
```

Kode Sumber 4.41 Implementasi Menampilkan Data Aduan

Secara garis besar, implementasi ini dilakukan dengan cara mengambil data aduan di *web service* lalu dimasukkan ke dalam sebuah *list* yang dimana *list* tersebut akan ditampilkan dengan menggunakan kelas *ListAduanAdapter*.

4.5.5 Implementasi Mencari Aduan

Implementasi mencari aduan diimplementasikan dengan cara mencari aduan dari *list* aduan menggunakan kata kunci yang dimasukkan. Kata kunci tersebut dijadikan menjadi huruf kecil begitu juga dengan kata-kata yang ada di aduan.Untuk pencariannya jika ada kata dari aduan yang mengandung unsur kata kunci, maka aduan tersebut dimasukkan ke dalam *list* pencarian. Untuk implementasi ini ditunjukkan Kode Sumber 4.53.

```
getSearchList(list aduanList, string
keyword){
  keylowercase ← keyword.toLowerCase()
  int statusTerjawab ← 0
  if keylowercase equal "terjawab"
     statusTerjawab ← 1
  end if
  else if keylowercase equal "tidak
terjawab"
     statusTerjawab ← 2
  end if
  inisialisasi list pencarian
  for i=0 sampai panjang list aduanList{
    if
aduanList[i].getDepartemen().toLowerCase().c
ontains (keylowercase)
      if(!pencarian.contains(aduanList[i])
        pencarian.add(aduanList[i])
      end if
    end if
    if
aduanList[i].getId().toLowerCase().contains
(keylowercase)
      if(!pencarian.contains(aduanList[i])
        pencarian.add(aduanList[i])
      end if
    end if

```

```
    if
aduanList[i].getIsi().toLowerCase().contains
(keylowercase)
      if(!pencarian.contains(aduanList[i])
        pencarian.add(aduanList[i])
      end if
    end if
    if
aduanList[i].getPrioritas().toLowerCase().co
ntains (keylowercase)
      if(!pencarian.contains(aduanList[i])
        pencarian.add(aduanList[i])
      end if
    end if
    if
aduanList[i].getTopik().toLowerCase().contai
ns (keylowercase)
      if(!pencarian.contains(aduanList[i])
        pencarian.add(aduanList[i])
      end if
    end if
    if
aduanList[i].getWaktu().toLowerCase().substr
ing(0,9).contains(keylowercase)
      if(!pencarian.contains(aduanList[i])
        pencarian.add(aduanList[i])
      end if
    end if
    if aduanList[i].getStatus() ==4 &&
statusTerjawab == 1
      if(!pencarian.contains(aduanList[i])
        pencarian.add(aduanList[i])
      end if
    end if
    if aduanList[i].getStatus() <=3 &&
statusTerjawab == 2
      if(!pencarian.contains(aduanList[i])
        pencarian.add(aduanList[i])
      end if
    end if
  }

```

```
68 | Return pencarian
   | }
```

Kode Sumber 4.42 Implementasi Pencarian Aduan

4.5.6 Implementasi Menjawab Aduan

Implementasi menjawab aduan diimplementasikan dengan cara mengirim data yang telah dimasukkan oleh pengguna untuk membalas aduan tersebut ke *web service. Web service* ini yang akan digunakan untuk memasukkan data tersebut ke dalam basis data. Implementasi ini ditunjukkan pada Kode Sumber 4.43.

```
1 | Inisialisasi list BasicNameValuePair
2 | Masukkan key dan value id,isi_detail
3 | petugas_detail,via,no_hp ke
4 | basicNameValuePair
5 |
6 | Inisialisasi jParser
7 | jParser.makeHttpRequest(destination,”post”,
8 | list BasicNameValuePair)
```

Kode Sumber 4.43 Implementasi Menjawab Aduan

4.5.7 Implementasi Mengubah Status Aduan

Implementasi mengubah status aduan diimplementasikan dengan cara mengambil data status dari *web service*. Data tersebut lalu dimasukkan ke dalam suatu *list* lalu jika ikon status di antarmuka tindak lanjut aduan ditekan maka *list* status tersebut akan muncul dan dapat dipilih. Jika sudah dipilih dan ditekan *button update* maka status untuk aduan tersebut terubah. Untuk implementasi pengambilan data status dapat ditunjukkan pada Kode Sumber 4.44. Untuk implementasi menampilkan data status dapat ditunjukkan pada

Kode Sumber 4.45. Implementasi mengubah data status untuk suatu aduan dapat ditunjukkan pada Kode Sumber 4.46.

```
getDataStatus(){
inisialisasi list status
try{
   if json.getInt(TAG_SUCCESS) == 1
     Inisialisasi JSONArray TAG STATUS
     For i=0 sampai panjang JSONArray{
      JSONObject ← JSONArray[i]
      status ←
JSONObject.getString(TAG_STATUS)
      nama_status ←
JSONObject.getString(TAG_NAMA_STATUS)
     status.add(new
status(status,nama_status)
     }
   return status
   end if
}
```

Kode Sumber 4.44 Implementasi Pengambilan Data Status

```
onClick(View arg0) {
 inisialisasi AlertDialog.Builder builder
 builder.setTitle("Pilih Status")
 builder.setItems(statusArray,
DialogInterface.OnClickListener(){
  public void onClick(DialogInterface arg0,
int arg1){
  textViewStatus.setText(statusArray[arg1])
  }
})builder.show()}
```

Kode Sumber 4.45 Implementasi Menampilkan Data Status

```
1  Inisialisasi list BasicNameValuePair
2  Masukkan key dan value id_status dan id
3  aduan ke basicNameValuePair
4
5  Inisialisasi jParser
6  jParser.makeHttpRequest(destination,"post",
7  list BasicNameValuePair)
```

Kode Sumber 4.46 Implementasi Mengubah Status Aduan

4.5.8 Implementasi Mengubah Prioritas Aduan

Implementasi mengubah prioritas aduan diimplementasikan dengan cara mengambil data prioritas dari *web service*. Data lalu dimasukkan ke suatu *list* lalu jika *icon* prioritas di antarmuka tindak lanjut aduan ditekan maka *list* prioritas tersebut akan muncul dan dapat dipilih. Jika sudah dipilih dan ditekan *button* update maka prioritas untuk aduan tersebut terubah. Untuk implementasi pengambilan data prioritas dapat ditunjukkan pada Kode Sumber 4.47. Untuk implementasi menampilkan data prioritas dapat ditunjukkan pada Kode Sumber 4.48. Implementasi mengubah data prioritas untuk suatu aduan dapat ditunjukkan pada Kode Sumber 4.49.

```
1  getDataPrioritas(){
2  inisialisasi list prioritas
3  try{
4     if json.getInt(TAG_SUCCESS) == 1
5       Inisialisasi JSONArray TAG PRIORITAS
6       For i=0 sampai panjang JSONArray{
7        JSONObject ← JSONArray[i]
8        prioritas ←
9  JSONObject.getString(TAG_ PRIORITAS)
10
```

```
      nama_ prioritas ←
JSONObject.getString(TAG_NAMA_PRIORITAS)
     prioritas.add(new prioritas
(prioritas,nama_ prioritas)
     }
   return prioritas
   end if
}
```

Kode Sumber 4.47 Implementasi Pengambilan Data Prioritas

```
onClick(View arg0) {
 inisialisasi AlertDialog.Builder builder
 builder.setTitle("Pilih Prioritas")
 builder.setItems(prioritasArray,
DialogInterface.OnClickListener(){
  public void onClick(DialogInterface arg0,
int arg1){
textViewPriortas.setText(prioritasArray[arg1
])
  }
})builder.show()
}
```

Kode Sumber 4.48 Implementasi Menampilkan Data Prioritas

```
1 | Inisialisasi list BasicNameValuePair
2 | Masukkan key dan value id_prioritas dan id
3 | aduan ke basicNameValuePair
4 | Inisialisasi jParser
5 | jParser.makeHttpRequest(destination,"post",
6 | list BasicNameValuePair)
```

Kode Sumber 4.49 Implementasi Mengubah Prioritas Aduan

4.5.9 Implementasi Memilih Petugas Aduan

Implementasi Memilih petugas aduan diimplementasikan dengan cara mengambil data petugas dari *web service*. Data tersebut lalu dimasukkan ke dalam suatu *list* lalu jika *icon* petugas di antarmuka tindak lanjut aduan ditekan maka *list* petugas tersebut akan muncul dan dapat dipilih. Jika sudah dipilih dan ditekan *button* update maka petugas untuk aduan tersebut terpilih. Untuk implementasi pengambilan data petugas dapat ditunjukkan pada Kode Sumber 4.50. Untuk implementasi menampilkan data petugas dapat ditunjukkan pada Kode Sumber 4.51. Implementasi mengubah data petugas untuk suatu aduan dapat ditunjukkan pada Kode Sumber 4.52.

```
 1 | getDataPetugas(){
 2 | inisialisasi list petugas
 3 | try{
 4 |    if json.getInt(TAG_SUCCESS) == 1
 5 |      Inisialisasi JSONArray TAG PETUGAS
 6 |      For i=0 sampai panjang JSONArray{
 7 |       JSONObject ← JSONArray[i]
 8 |       petugas ← JSONObject.getString(TAG_
 9 | PETUGAS)
10 |       nama_ petugas ←
11 | JSONObject.getString(TAG_NAMA_PETUGAS)
12 |      prioritas.add(new petugas (petugas,
13 | nama_petugas)
```

```
14        }
15      return petugas
16      end if
17  }
```

Kode Sumber 4.50 Implementasi Pengambilan Data Petugas

```
1   onClick(View arg0) {
2    inisialisasi AlertDialog.Builder builder
3    builder.setTitle("Pilih Petugas")
4    builder.setItems(petugasArray,
5   DialogInterface.OnClickListener(){
6     public void onClick(DialogInterface arg0,
7   int arg1){
8   textViewPetugas.setText(petugasArray[arg1])
9     }
10  })builder.show()
    }
```

Kode Sumber 4.51 Implementasi Menampilkan Data Petugas

```
1  Inisialisasi list BasicNameValuePair
2  Masukkan key dan value id_prioritas dan id
3  aduan ke basicNameValuePair
4
5  Inisialisasi jParser
6  jParser.makeHttpRequest(destination,"post",
7  list BasicNameValuePair);
```

Kode Sumber 4.52 Implementasi Mengubah Data Petugas yang Menanggapi Aduan

4.5.10 Implementasi Mengembalikan Aduan

Implementasi mengembalikan aduan diimplementasikan dengan cara mengirim status aduan ke *web service*. Untuk implementasi ini ditunjukkan Kode Sumber 4.53

```
1  Inisialisasi list BasicNameValuePair
2  spam ← 2
3  Masukkan key dan value spam ke
4  basicNameValuePair
5
6  Inisialisasi jParser
7  jParser.makeHttpRequest(destination,"post",
8  list BasicNameValuePair);
```

Kode Sumber 4.53 Implementasi Mengembalikan Aduan

4.5.11 Implementasi Mengubah Status Spam Aduan

Implementasi mengubah status *spam* aduan diimplementasikan dengan cara mengirim status aduan spam ke *web service*. Untuk implementasi ini ditunjukkan Kode Sumber 4.54.

```
1  Inisialisasi list BasicNameValuePair
2  spam ← 1
3  Masukkan key dan value spam ke
4  basicNameValuePair
5
6  Inisialisasi jParser
7  jParser.makeHttpRequest(destination,"post",
8  list BasicNameValuePair);
```

Kode Sumber 4.54 Implementasi Mengubah Status Spam Aduan

4.5.12 Implementasi Menampilkan Lampiran Aduan

Menampilkan lampiran aduan diimplementasikan dengan cara mengambil data seluruh lampiran aduan dari suatu aduan yang sudah dipilih dari *web service*. Data tersebut lalu ditampilkan menggunakan *listview* dengan menggunakan kelas

attachmentAdapter. Berikut Implementasi dari mengambil data dari *web service* yang ditunjukkan pada Kode Sumber 4.55 dan menampilkan data dari *web service* yang ditunjukkan pada Kode Sumber 4.56.

```
getDataLampiran(){
inisialisasi list lampiran
try{
   if json.getInt(TAG_SUCCESS) == 1
     Inisialisasi JSONArray TAG STATUS
     For i=0 sampai panjang JSONArray{
      JSONObject ← JSONArray[i]
      waktu_detail ←
JSONObject.getString(TAG_WAKTU_DETAIL)
      file_type ←
JSONObject.getString(TAG_FILE_TYPE)
      path_upload ←
JSONObject.getString(TAG_PATH_UPLOAD)
      nama_file ←
JSONObject.getString(TAG_NAMA_FILE)
     lampiran.add(new
lampiran(waktu_detail,file_type,path_upload
, nama_file)
     }
   return lampiran
   end if
}
```

Kode Sumber 4.55 Implementasi Pengambilan Lampiran Aduan

```
listLampiran
inisialisasi
listAdapter(activity,listLampiran)

listview.setAdapter(listAdapter)
set onItemclickListener di listview
```

Kode Sumber 4.56 Implementasi Menampilkan Lampiran Aduan

4.5.13 Implementasi Mengunduh Lampiran

Implementasi mengunduh lampiran diimplementasikan dengan cara mengambil data *url* dari lampiran yang dipilih kemudian dari *url* tersebut akan diunduh *file* lampirannya. Proses mengunduh ini diimplementasikan menggunakan kelas *URLConnection* dan *InputStream*. Selanjutnya, respon diolah oleh kelas *OutputStream* untuk ditulis ke dalam bentuk *file*. Implementasi proses ini ditunjukkan pada Kode Sumber 4.57.

```
unduhLampiran(string nama_file){
Inisialisasi URLConnection
Inisialisasi direktori di
"/OperatorSurga/Lampiran"
if direktori != tidak ada
 buat direktori
end if
Inisialisasi InputStream
Inisialisasi OutputStream ke direktori
"OperatorSurga/Lampiran/nama_file"
inisialisasi byte dengan panjang 1024
int total ← 0
while ( int count ← InputStream.read(byte)
!= -1)
  total ← total+count
  publishProgress()
  OutputStream.write(byte,0,count)
End while
```

```
18 OutputStream.flush()
19 OutputStream.close()
20 InputStream.close()
21 }
```

Kode Sumber 4.57 Implementasi Mengunduh Lampiran

4.5.14 Implementasi Menampilkan Peta Lokasi Aduan

Implementasi Menampilkan peta lokasi aduan diimplementasikan menggunakan Google Maps API v2. Di dalam Google Maps API v2 ini kita harus inisialiasi peta dan mengecek apakah layanan ini tersedia dari Google jika layanan ini tersedia, maka peta akan mengarah kepada lintang dan bujur sesuai dengan lokasi aduan. Implementasi pengecekkan layanan ditunjukkan pada Kode Sumber 4.58, Implementasi inisialisasi peta ditunjukkan pada Kode Sumber 4.59, Implementasi mengubah lokasi peta ditunjukkan pada Kode Sumber 4.60, dan Implementasi menampilkan peta ditunjukkan pada Kode Sumber 4.61.

```
1  public boolean LayananGoogleMapSiap(){
2  int isAvailable ←
3  GooglePlayServicesUtil.isGooglePlayServicesA
4  vailable(this)
5  If isAvailable==ConnectionResult.SUCCESS
6        return true
7  end if
8  else if
9  GooglePlayServicesUtil.isUserRecoverableErro
10 r(isAvailable)
11 Dialog d =
12 GooglePlayServicesUtil.getErrorDialog(isAvai
13 lable, this, GPS_ERRORDIALOG_REQUEST)
14 d.show();
15 end if
16
17 else
```

```
17 Toast.makeText(this, "Tidak bisa terhubung
18 dengan Google Service",
19 Toast.LENGTH_SHORT).show();
20 end if
21
22 return false;
   }
```

Kode Sumber 4.58 Implementasi Pengecekkan Layanan Google Maps

```
1 GoogleMap myMap
2 inisialiasiPeta(){
3 if myMap is null
4   inisialisasi fragmentPeta
5   myMap = fragmentPeta.getMap()
6 end if
  }
```

Kode Sumber 4.59 Implementasi Inisialisasi Peta

```
1 private void gotoLocation(double lintang,
2 double bujur,int zoom){
3 LatLng kordinatLokasi ←new
4 LatLng(lintang,bujur)
5
6 CameraUpdate update ←
7 CameraUpdateFactory.newLatLngZoom(kordinatLo
8 kasi, zoom);
9 myMap.moveCamera(update)
  }
```

Kode Sumber 4.60 Implementasi Mengubah Lokasi Peta

```
1 If LayananGoogleMapSiap()
2  setContentView(R.layout.gmaps);
3  if initMap()
4  gotoLocation(latitude, longitude, 15)
```

```
 myMap.addMarker(new
MarkerOptions().position(new
LatLng(latitude, longitude)).title("Lokasi
yang di tandai"))
 end if
end if

else
 Toast.makeText(this, "peta tidak ada",
Toast.LENGTH_SHORT).show()
End if
```

Kode Sumber 4.61 Implementasi Menampilkan Peta

4.5.15 Implementasi Validasi Pesan Singkat yang Salah

Implementasi validasi pesan singkat yang salah diimplementasi dengan mengirim data-data untuk validasi yang sudah dimasukkan oleh petugas. *Web service* ini yang akan memasukkan pesan singkat salah yang sudah divalidasi. Implementasi ini ditunjukkan pada Kode Sumber 4.62.

```
Inisialisasi list BasicNameValuePair
Masukkan key dan value nama, no_ktp, isi,
dan waktu ke basicNameValuePair

Inisialisasi jParser
jParser.makeHttpRequest(destination,”post”,
list BasicNameValuePair)
```

Kode Sumber 4.62 Implementasi Validasi Pesan Singkat yang Salah

4.5.16 Implementasi Mengirim Pesan Singkat ke Petugas

Implementasi mengirim pesan singkat ke petugas diimplementasikan dengan cara mengirim data-data untuk

mengirim pesan singkat ke *web service.* Dimana *web service* ini akan memasukkan data-data tersebut ke dalam basis data. Jika sudah dimasukkan ke dalam basis data dengan otomatis sistem akan mengirim pesan singkat tersebut ke petugas. Implementasi ini ditunjukkan pada Kode Sumber 4.63.

```
1  Inisialisasi list BasicNameValuePair
2  Masukkan key dan value nama, no_ktp, isi,
3  dan waktu ke basicNameValuePair
4
5  Inisialisasi jParser
6  jParser.makeHttpRequest(destination,"post",
7  list BasicNameValuePair)
```

Kode Sumber 4.63 Implementasi Mengirim Pesan Singkat ke Petugas

4.5.17 Implementasi Mengunduh Laporan

Implementasi mengunduh lampiran diimplementasikan dengan cara mengirim data tahun yang dipilih ke *web service. Web service* ini yang akan menjadikan hasil *query* dari basis data menjadi *file* yang berekstensi "*.csv*" yang dapat diunduh. Proses mengunduh ini diimplementasikan menggunakan kelas *DefaultHttpClient* dan *InputStream*. Selanjutnya, respon diolah oleh kelas *OutputStream* untuk ditulis ke dalam bentuk *file*. Implementasi proses ini ditunjukkan pada Kode Sumber 4.64.

```
1  Download Laporan(string tahun, url){
2  Inisialisasi list BasicNameValuePair
3  Masukkan key dan value tahun ke
4  basicNameValuePair
5
6  Inisialisasi DefaultHttpConnection client
7  Inisialisasi HttpPost post(url)
```

```
 8  Set entity httpPost dengan
 9  basicNameValuePair
10
11  httpResponse response ← client.execute(post)
12  httpEntity entity ← response.getEntity()
13
14  if entity is not null
15    InputStream ← entity.getContent()
16    inisialisasi direktori di
    "OperatorSurga/Laporan/"
17   if direktori is null
18    membuat direktori "OperatorSurga/Laporan/"
19   end if
20   Inisialisasi OutputStream dengan direktori
    ditambah nama file
21   Long length ← entity.getContentlength()
22   Inisialisasi buffer byte panjang 1024
23   Int bufferlength ← 0
24   While bufferLength =
    InputStream.read(buffer) > 0
25      publishProgress()
26
27  OutputStream.write(buffer,0,bufferLength)
28  End while
29  OutputStream.flush()
30  OutputStream.close()
31  End if
32  }
33
34
```

Kode Sumber 4.64 Implementasi Mengunduh Laporan

BAB V
UJI COBA DAN EVALUASI

Pada bab ini dijelaskan mengenai rangkaian uji coba dan evaluasi yang dilakukan terhadap sistem yang dibuat. Pembahasan yang dipaparkan meliputi lingkungan uji coba, data uji coba, skenario uji coba, hasil uji coba, dan evaluasi.

5.1 Lingkungan Uji Coba

Proses uji coba terhadap sistem ini dilakukan menggunakan *smartphone* Asus Zenfone 5 dengan spesifikasi seperti berikut:

- Sistem Operasi: Android v4.4 (Kitkat)
- CPU: Dual core 2GHz Intel Atom Z2580
- Memory internal: 16 GB
- RAM: 2 GB
- Speed: HSDPA, 14.4 Mbps; HSUPA, 5.76 Mbps
- WLAN: Wi-Fi 802.11 b/g/n, Wi-Fi hotspot

5.2 Pengujian Modularitas

Pengujian ini dilakukan untuk menguji apakah modularitas benar-benar diimplementasikan dan bekerja semestinya. Selain itu juga untuk mengetahui kesesuaian keluaran dari setiap tahapan atau langkah penggunaan terhadap skenario yang dipersiapkan. Pengujian dilakukan dengan metode kotak hitam.

5.2.1 Skenario Pengujian Modularitas

Pada bagian ini akan dilakukan sejumlah pengujian terhadap perangkat lunak untuk menguji kebenaran modularitas dari aplikasi ini. Pengujian ini dilakukan dengan secara mandiri dan didokumentasikan secara sistematis dengan menyiapkan skenario sebagai tolak ukur keberhasilan sistem.

Skenario yang disiapkan adalah skenario dalam melakukan membuat dan memuat modul dari aplikasi ini. Berikut ini adalah hasil pengujian membuat dan memuat modul yang telah diimplementasikan pada tahap pengembangan.

5.2.1.1 Pengujian Membuat dan Memuat Modul Aplikasi

Pengujian ini dilakukan terhadap memuat suatu modul aplikasi yang telah mengimplementasikan *interface* aplikasi utama. Modul aplikasi ini merupakan modul aplikasi *monitoring* surga. Pengujian ini dimulai ketika modul aplikasi *monitoring* surga telah disiapkan sudah mengimplementasikan *inteface* aplikasi utama dan mengubah *androidmanifest.xml* sesuai dengan subbab 3.2.4. Kemudian modul aplikasi tersebut dipasang ke dalam perangkat Android. Kelas yang mengimplementasi *interface* dari aplikasi utama di ekspor ke dalam *file jar* dengan menggunakan *manifest.mf* yang sudah dibuat. *File jar* tersebut di *compile* menggunakan *dex* yang ada di SDK Android dan menghasilkan *file classes.dex*. *File classes.dex* dimasukkan ke dalam *file jar*. Lalu *copy file jar* tersebut ke dalam perangkat Android di dalam direktori "Plugin". Hasil pengujian terhadap pengujian ini digambarkan dalam Gambar 5.1.

Gambar 5.1 Pengujian Membuat dan Memuat Modul Aplikasi Monitoring

5.3 Pengujian Fungsionalitas

Pengujian ini dilakukan untuk menguji apakah fungsionalitas yang diindentifikasi pada tahap kebutuhan benar-benar diimplementasikan dan bekerja semestinya. Selain itu juga untuk mengetahui kesesuaian keluaran dari setiap tahapan atau langkah penggunaan fitur terhadap skenario yang dipersiapkan. Pengujian dilakukan dengan metode *black-box.*

5.3.1 Skenario Pengujian Fungsionalitas

Pada bagian ini akan dilakukan sejumlah pengujian perangkat lunak untuk menguji kebenaran dari aplikasi ini. Pengujian fungsionalitas perangkat lunak ini dilakukan secara mandiri dan didokumentasikan secara sistematis dengan

menyiapkan sejumlah skenario sebagai tolak ukur keberhasilan sistem. Pengujian ini meliputi seluruh kasus penggunaan yang telah dijelaskan pada Bab 3. Pengujian fungsionalitas ini meliputi proses yg dijabarkan sebagai berikut.

1. Menampilkan Aduan
2. Mencari Aduan
3. Menjawab Aduan
4. Mengubah Status Aduan
5. Mengubah Prioritas Aduan
6. Memilih Petugas Aduan
7. Mengembalikan Aduan
8. Mengubah Status *Spam* Aduan
9. Menampilkan Lampiran Aduan
10. Mengunduh Lampiran
11. Menampilkan Peta Lokasi Aduan
12. Validasi Pesan Singkat yang Salah
13. Mengirim Pesan Singkat ke Petugas
14. Mengunduh Laporan

5.3.2 Hasil Pengujian Fungsionalitas

Hasil pengujian dari poin-poin dari skenario pada subbab sebelumnya dilampirkan pada bagian subbab ini. Berikut ini adalah hasil pengujian fungsionalitas fitur yang telah diimplementasikan pada tahap pengembangan.

5.3.2.1 Pengujian Menampilkan Aduan

Pengujian ini dilakukan terhadap fungsionalitas menampilkan daftar aduan berdasarkan pengguna yang telah login ke dalam sistem. Pengujian ini dimulai ketika pengguna sudah berhasil masuk ke dalam sistem melalui *login*, memilih menu operator surga, dan memilih sub menu manajemen aduan. dimana pada saat berhasil *login*, informasi ID pengguna akan disimpan dalam session aplikasi. Tabel 5.1 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.1.

Tabel 5.1 Prosedur Uji Coba Menampilkan Aduan

ID	**UJ-01**
Referensi Use Case	**UC-01**
Nama	**Uji Coba Menampilkan Aduan**
Tujuan Uji Coba	Menguji fitur untuk menampilkan aduan setelah menekan sub menu manajemen aduan
Kondisi Awal	Pengguna telah berada pada antarmuka sub menu
Data Masukan	-*Username* pengguna 'humas' untuk petugas tingkat satu -*Username* pengguna 'pu1' untuk petugas tingkat dua
Prosedur Pengujian	1. Menekan sub menu manajemen aduan untuk masuk ke antarmuka manajemen aduan dan dapat melihat aduan
Data Keluaran	Data-data aduan berdasarkan pengguna yang telah *login*.
Hasil Uji Coba	**Berhasil**

Gambar 5.2 Pengujian Menampilkan Aduan

5.3.2.2 Pengujian Mencari Aduan

Pengujian ini dilakukan terhadap fungsionalitas mencari aduan. Pengujian ini dimulai ketika pengguna mengisi *textView* pada halaman manajemen aduan di *tab* manapun. Tabel 5.2 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.3.

Tabel 5.2 Prosedur Uji Coba Mencari Aduan

ID	**UJ-02**
Referensi Use Case	**UC-02**
Nama	**Uji Coba Mencari Aduan**
Tujuan Uji Coba	Menguji fitur untuk mencari aduan.
Kondisi Awal	Pengguna telah berada pada antarmuka manajemen aduan
Data Masukan	-*Username* pengguna 'humas'
Prosedur Pengujian	1. Memilih tab aduan diklasifikasi 2. Memasukkan kata 'dinas sosial'
Data Keluaran	Aduan-aduan yang memiliki kata 'dinas sosial' pada aduannya.
Hasil Uji Coba	**Berhasil**

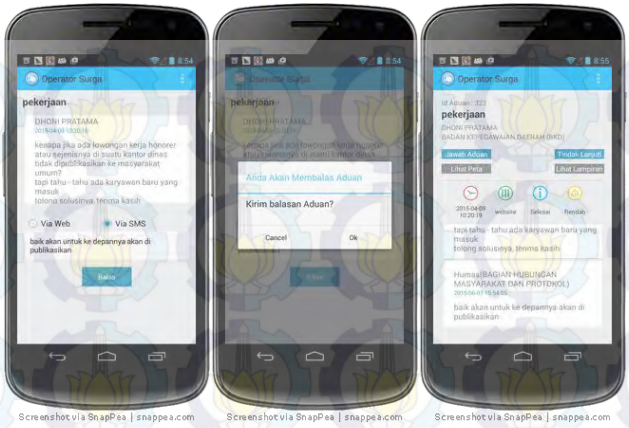
Gambar 5.3 Pengujian Mencari Aduan

5.3.2.3 Pengujian Menjawab Aduan

Pengujian ini dilakukan terhadap fungsionalitas menjawab aduan untuk menjawab aduan yang dipilih. Pengujian ini dimulai ketika pengguna menekan *button* jawab di antarmuka detail aduan dan masuk ke antarmuka jawab aduan. Tabel 5.3 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.4.

Tabel 5.3 Prosedur Uji Coba Menjawab Aduan

ID	**UJ-03**
Referensi Use Case	**UC-03**
Nama	**Uji Coba Menjawab Aduan**
Tujuan Uji Coba	Menguji fitur untuk menjawab aduan untuk menjawab aduan yang dipilih
Kondisi Awal	Pengguna telah berada pada antarmuka jawab aduan
Data Masukan	-*Username* pengguna 'humas' -Memilih aduan yang mempunyai id '323'
Prosedur Pengujian	1. Mengisi balasan aduan 2. Memilih *radio button* untuk memilih membalas melalui situs atau pesan singkat 3. Tekan *button* balas 4. Menekan tombol ok di *pop up dialog*
Data Keluaran	Data balasan aduan akan muncul
Hasil Uji Coba	**Berhasil**



Gambar 5.4 Pengujian Menjawab Aduan

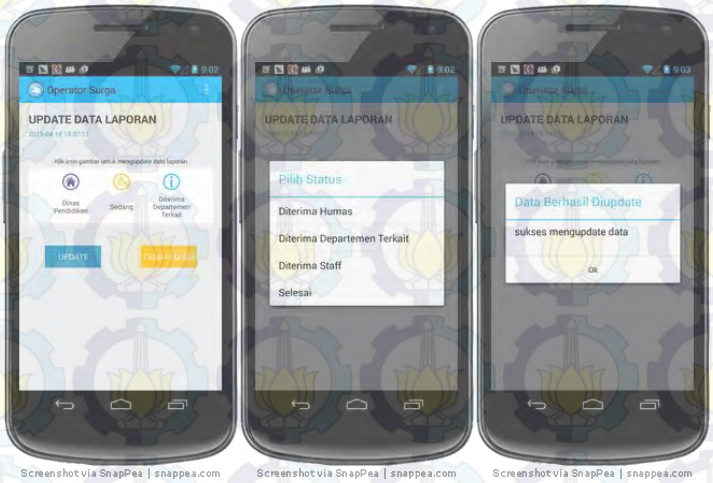
5.3.2.4 Pengujian Mengubah Status Aduan

Pengujian ini dilakukan terhadap fungsionalitas mengubah status aduan aduan untuk menanggapi aduan yang dipilih. Pengujian ini dimulai ketika pengguna menekan *button* tindak lanjuti di antarmuka detail aduan dan masuk ke antarmuka tindak lanjut aduan. Pengujian ini akan menampilkan semua pilihan status dengan menekan *icon* status dan dapat memilih status tersebut. Tabel 5.4 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.5.

Tabel 5.4 Prosedur Uji Coba Mengubah Status Aduan

ID	**UJ-04**
Referensi Use Case	**UC-04**
Nama	**Uji Coba Memilih Status Aduan**
Tujuan Uji Coba	Menguji fitur untuk memilih status aduan untuk aduan yang dipilih.

Kondisi Awal	Pengguna telah berada pada antarmuka tindak lanjut aduan
Data Masukan	-*Username* pengguna ‘humas’ -Memilih aduan yang mempunyai id ‘328’
Prosedur Pengujian	1. Menekan *icon* status 2. Memilih status diterima staff 3. Menekan tombol update
Data Keluaran	Data status untuk aduan 328 berubah menjadi diterima staff
Hasil Uji Coba	**Berhasil**



Gambar 5.5 Pengujian Mengubah Status Aduan

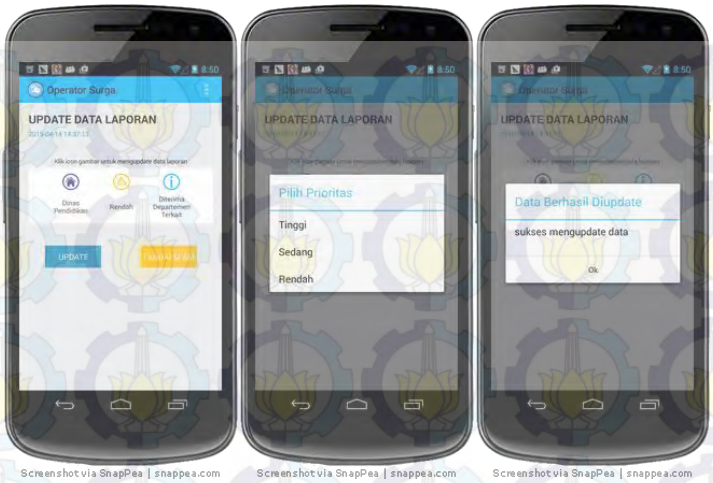
5.3.2.5 Pengujian Mengubah Prioritas Aduan

Pengujian ini dilakukan terhadap fungsionalitas mengubah prioritas aduan aduan untuk menanggapi aduan yang dipilih. Pengujian ini dimulai ketika pengguna menekan *button* tindak lanjuti di antarmuka detail aduan dan masuk ke antarmuka tindak lanjut aduan. Pengujian ini akan

menampilkan semua pilihan prioritas dengan menekan *icon* prioritas dan dapat memilih prioritas tersebut. Tabel 5.5 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.6.

Tabel 5.5 Prosedur Uji Coba Mengubah Prioritas Aduan

ID	**UJ-05**
Referensi Use Case	**UC-05**
Nama	**Uji Coba Mengubah Prioritas Aduan**
Tujuan Uji Coba	Menguji fitur untuk mengubah prioritas aduan untuk menanggapi aduan yang dipilih.
Kondisi Awal	Pengguna telah berada pada antarmuka tindak lanjut aduan
Data Masukan	-*Username* pengguna 'humas' -Memilih aduan yang mempunyai id '328'
Prosedur Pengujian	1. Menekan *icon* prioritas 2. Memilih prioritas sedang 3. Menekan tombol *update* 4. Menekan tombol *update pop up*
Data Keluaran	Data prioritas untuk aduan 328 berubah menjadi sedang
Hasil Uji Coba	**Berhasil**



Gambar 5.6 Pengujian Mengubah Prioritas Aduan

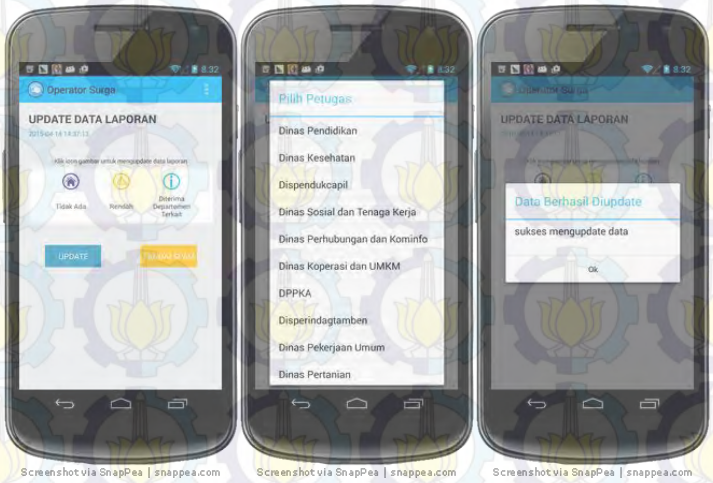
5.3.2.6 Pengujian Memilih Petugas Aduan

Pengujian ini dilakukan terhadap fungsionalitas memilih petugas aduan untuk menanggapi aduan yang dipilih. Pengujian ini dimulai ketika pengguna menekan *button* tindak lanjuti di antarmuka detail aduan dan masuk ke antarmuka tindak lanjut aduan. Pengujian ini akan menampilkan semua petugas tingkat dua jika menekan *icon* petugas dan dapat memilih petugas tersebut. Tabel 5.6 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.7.

Tabel 5.6 Prosedur Uji Coba Memilih Petugas Aduan

ID	**UJ-06**
Referensi Use Case	**UC-06**
Nama	**Uji Coba Memilih Petugas Aduan**
Tujuan Uji Coba	Menguji fitur untuk memilih petugas aduan untuk menanggapi aduan yang dipilih.

Kondisi Awal	Pengguna telah berada pada antarmuka tindak lanjut aduan
Data Masukan	-*Username* pengguna ‘humas’ untuk petugas tingkat satu -Memilih aduan yang mempunyai id ‘328’
Prosedur Pengujian	1. Menekan *icon* petugas 2. Memilih petugas dinas pendidikan 3. Menekan tombol *update* 4. Menekan tombol *update pop up*
Data Keluaran	Data petugas yang menanggapi aduan 328 berubah menjadi dinas pendidikan
Hasil Uji Coba	**Berhasil**



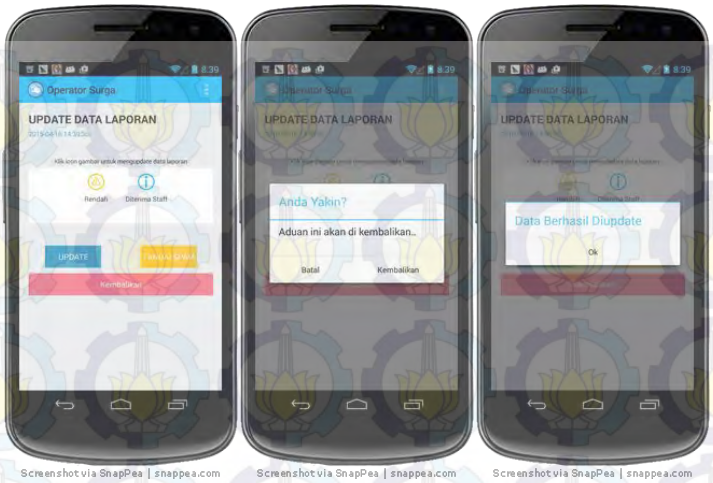
Gambar 5.7 Pengujian Memilih Petugas

5.3.2.7 Pengujian Mengembalikan Aduan

Pengujian ini dilakukan terhadap fungsionalitas mengambalikan aduan kepada petugas tingkat satu karena tidak berhubungan dengan SKPD petugas yang *login*. Pengujian ini dimulai ketika pengguna menekan *button* tindak lanjuti di antarmuka detail aduan dan masuk ke antarmuka tindak lanjut aduan. Tabel 5.7 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.8.

Tabel 5.7 Prosedur Uji Coba Mengembalikan Aduan

ID	**UJ-07**
Referensi Use Case	**UC-07**
Nama	**Uji Coba Mengembalikan Aduan**
Tujuan Uji Coba	Menguji fitur untuk mengembalikan aduan kepada petugas tingkat satu
Kondisi Awal	Pengguna telah berada pada antarmuka tindak lanjut aduan
Data Masukan	-*Username* pengguna ‘pu1’ -Memilih aduan yang mempunyai id ‘330’
Prosedur Pengujian	1. Menekan *button* kembalikan
Data Keluaran	Data aduan yang ber id ‘330’ dikembalikan di petugas tingkat satu
Hasil Uji Coba	**Berhasil**



Gambar 5.8 Pengujian Mengembalikan Aduan

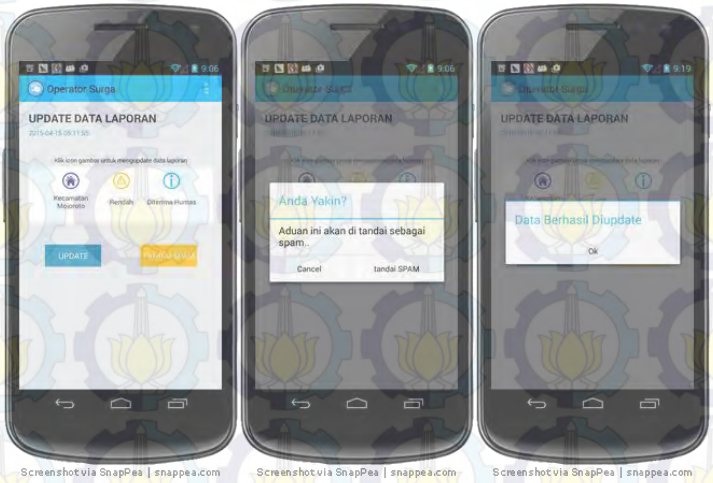
5.3.2.8 Pengujian Status *Spam* Aduan

Pengujian ini dilakukan terhadap fungsionalitas status spam aduan untuk menandai bahwa aduan tersebut adalah spam. Pengujian ini dimulai ketika pengguna menekan *button* tindak lanjuti di antarmuka detail aduan dan masuk ke antarmuka tindak lanjut aduan. Tabel 5.8 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.9

Tabel 5.8 Prosedur Uji Coba Status Spam Aduan

ID	**UJ-08**
Referensi Use Case	**UC-08**
Nama	**Uji Coba Status *Spam* Aduan**
Tujuan Uji Coba	Menguji fitur untuk menandai aduan sebagai *spam*
Kondisi Awal	Pengguna telah berada pada antarmuka tindak lanjut aduan

Data Masukan	-*Username* pengguna ‘humas’ -Memilih aduan yang mempunyai id ‘329’
Prosedur Pengujian	1. Menekan *button* tandai *spam*
Data Keluaran	Aduan yang mempunyai id ‘329’ ditandai menjadi *spam*
Hasil Uji Coba	**Berhasil**



Gambar 5.9 Pengujian Status *Spam* Aduan

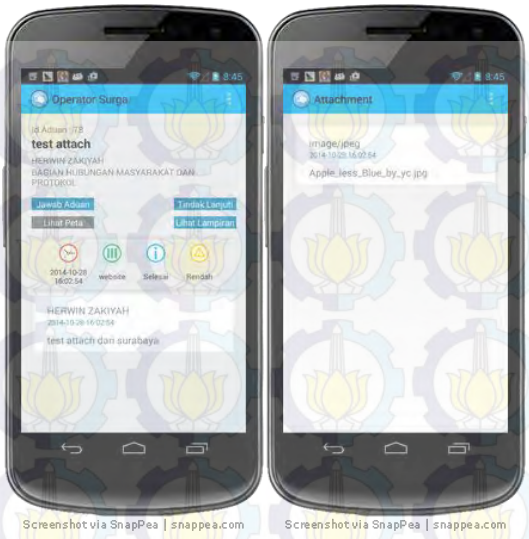
5.3.2.9 Pengujian Menampilkan Lampiran Aduan

Pengujian ini dilakukan terhadap fungsionalitas menampilkan lampiran-lampiran aduan yang disertakan. Pengujian ini dimulai ketika pengguna menekan *button* lihat lampiran di antarmuka detail aduan dan masuk ke antarmuka lihat lampiran. Pengujian ini akan menampilkan semua lampiran yang disertakan pada aduan yang dipilih. Tabel 5.9 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil

pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.10.

Tabel 5.9 Prosedur Uji Coba Menampilkan Lampiran Aduan

ID	**UJ-09**
Referensi Use Case	**UC-09**
Nama	**Uji Coba Menampilkan Lampiran Aduan**
Tujuan Uji Coba	Menguji fitur untuk melihat semua lampiran yang disertakan pada aduan yang dipilih
Kondisi Awal	Pengguna telah berada pada antarmuka detail aduan
Data Masukan	-*Username* pengguna 'humas' -Memilih aduan yang mempunyai id '30'
Prosedur Pengujian	1. Menekan *button* lihat lampiran
Data Keluaran	Data semua lampiran yang disertakan pada aduan yang dipilih
Hasil Uji Coba	**Berhasil**



Gambar 5.10 Pengujian Menampilkan Lampiran Aduan

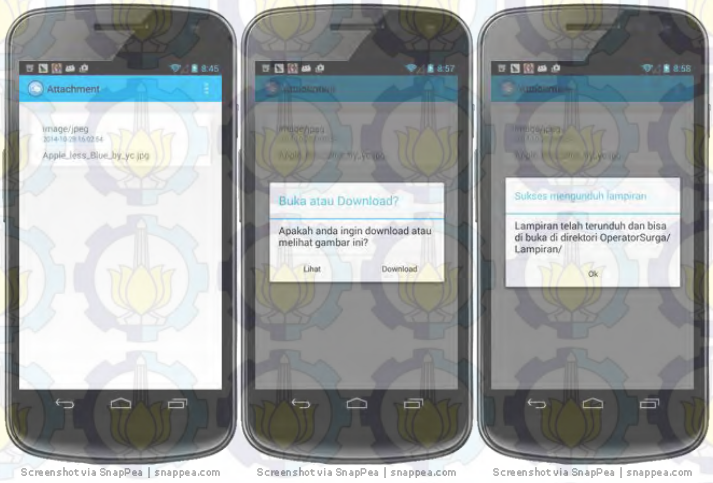
5.3.2.10 Pengujian Mengunduh Lampiran

Pengujian ini dilakukan terhadap fungsionalitas mengunduh lampiran untuk lampiran yang dipilih. Pengujian ini dimulai ketika pengguna menekan *listview* lampiran di antarmuka lihat lampiran. Tabel 5.10 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.11.

Tabel 5.10 Prosedur Uji Coba Mengunduh Lampiran

ID	**UJ-10**
Referensi Use Case	**UC-10**
Nama	**Uji Coba Mengunduh Lampiran**
Tujuan Uji Coba	Menguji fitur untuk Mengunduh lampiran yang dipilih

Kondisi Awal	Pengguna telah berada pada antarmuka lihat lampiran
Data Masukan	-*Username* pengguna ‘humas’ -Memilih aduan yang mempunyai id ‘78’
Prosedur Pengujian	1. Memilih lampiran ‘’ 2. Menekan tombol unduh
Data Keluaran	*File* yang dipilih akan terunduh.
Hasil Uji Coba	**Berhasil**



Gambar 5.11 Pengujian Mengunduh Lampiran

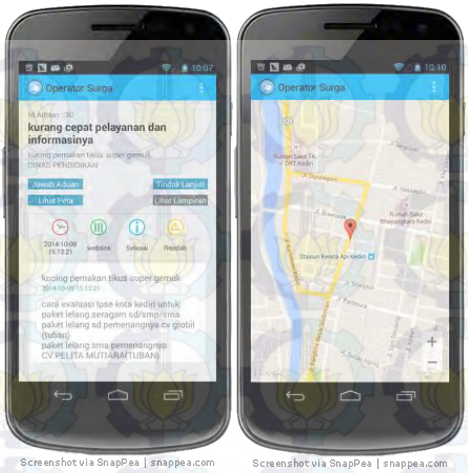
5.3.2.11 Pengujian Menampilkan Peta Lokasi Aduan

Pengujian ini dilakukan terhadap fungsionalitas menampilkan lokasi peta dari aduan yang disertakan. Pengujian ini dimulai ketika pengguna menekan *button* lihat peta di antarmuka detail aduan. Pengujian ini akan menampilkan lokasi peta disertakan pada aduan yang dipilih. Tabel 5.11 menjelaskan skenario dari pengujian fungsionalitas

ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.12.

Tabel 5.11 Prosedur Uji Coba Menampilkan Peta Lokasi Aduan

ID	**UJ-11**
Referensi Use Case	**UC-11**
Nama	**Uji Coba Menampilkan Peta Lokasi Aduan**
Tujuan Uji Coba	Menguji fitur untuk menampilkan peta lokasi aduan.
Kondisi Awal	Pengguna telah berada pada antarmuka detail aduan
Data Masukan	-*Username* pengguna 'humas' -Memilih aduan yang mempunyai id '78'
Prosedur Pengujian	1. Menekan *button* lihat peta
Data Keluaran	Lokasi peta akan muncul
Hasil Uji Coba	**Berhasil**



Gambar 5.12 Pengujian Menampilkan Peta Lokasi Aduan

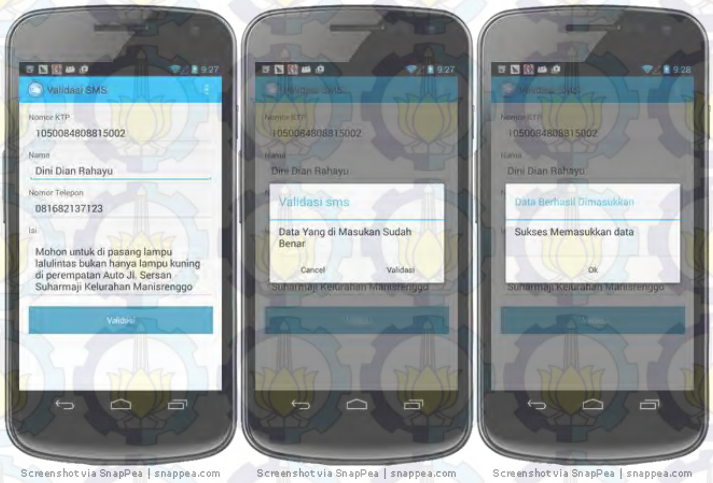
5.3.2.12 Pengujian Validasi Pesan Singkat yang Salah

Pengujian ini dilakukan terhadap fungsionalitas validasi pesan singkat yang salah. Pengujian ini dimulai ketika pengguna menekan *listview* sms yang tidak valid di antarmuka *list* validasi sms. Pengujian ini akan menampilkan nomor telepon pengirim dan isi laporan aduan. Tabel 5.12 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.13.

Tabel 5.12 Prosedur Uji Coba Validasi SMS yang Salah

ID	**UJ-12**
Referensi Use Case	**UC-12**
Nama	**Uji Coba Validasi Pesan Singkat yang Salah**
Tujuan Uji Coba	Menguji fitur untuk memvalidasi pesan singkat yang salah

Kondisi Awal	Pengguna telah berada pada antarmuka validasi sms
Data Masukan	-*Username* pengguna ‘humas’ -*id sms tidak valid* ‘2’
Prosedur Pengujian	1. Memasukkan nomor ktp 2. Memasukkan nama 3. Menekan tombol *validasi*
Data Keluaran	Sms yang divalidasi tersebut menjadi aduan.
Hasil Uji Coba	**Berhasil**



Gambar 5.13 Pengujian Validasi Pesan Singkat yang Salah

5.3.2.13 Pengujian Mengirim Pesan Singkat ke Petugas

Pengujian ini dilakukan terhadap fungsionalitas mengirim pesan singkat ke petugas. Pengujian ini dimulai ketika pengguna menekan *listview* sms di antarmuka sub menu. Pengujian ini akan menampilkan petugas yang dapat dikirimi sms. Tabel 5.13 menjelaskan skenario dari pengujian

fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.14.

Tabel 5.13 Prosedur Uji Coba Mengirim Pesan Singkat ke Petugas

ID	**UJ-13**
Referensi Use Case	**UC-13**
Nama	**Uji Coba Mengirim Pesan Singkat ke Petugas**
Tujuan Uji Coba	Menguji fitur untuk mengirim pesan singkat ke petugas
Kondisi Awal	Pengguna telah berada pada antarmuka sms
Data Masukan	-*Username* pengguna 'humas'
Prosedur Pengujian	1. Menekan *textview* petugas 2. Memilih petugas 'dinas pendidikan' 3. Memasukkan pesan pesan singkat 4. Menekan *button* kirim
Data Keluaran	Pesan singkat akan terkirim ke petugas
Hasil Uji Coba	**Berhasil**

Gambar 5.14 Pengujian Mengirim Pesan Singkat ke Petugas

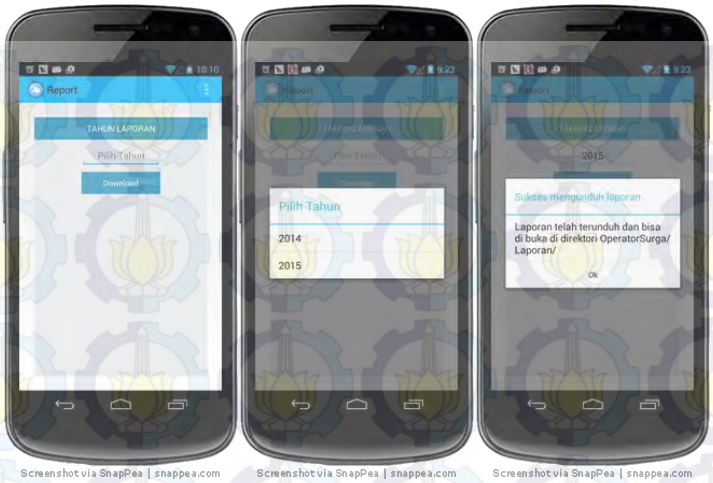
5.3.2.14 Pengujian Mengunduh Laporan

Pengujian ini dilakukan terhadap fungsionalitas mengunduh laporan. Pengujian ini dimulai ketika pengguna menekan *listview* laporan di antarmuka sub menu. Pengujian ini akan menampilkan tahun laporan yang ingin diunduh. Tabel 5.14 menjelaskan skenario dari pengujian fungsionalitas ini.

Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.15.

Tabel 5.14 Prosedur Uji Coba Mengunduh Laporan

ID	**UJ-14**
Referensi Use Case	**UC-14**
Nama	**Uji Coba Mengunduh Laporan**
Tujuan Uji Coba	Menguji fitur untuk mengunduh laporan berdasarkan tahun
Kondisi Awal	Pengguna telah berada pada antarmuka laporan
Data Masukan	-*Username* pengguna 'humas'
Prosedur Pengujian	1. Menekan *textview* tahun 2. Memilih tahun '2015' 3. Menekan *button* unduh
Data Keluaran	Laporan tahun 2015 terunduh.
Hasil Uji Coba	**Berhasil**



Gambar 5.15 Pengujian Mengunduh Laporan

5.4 Evaluasi

Dari uji coba yang telah dilakukan, diketahui bahwa sistem telah dapat bekerja dengan baik dan benar dalam menjalankan fungsionalitasnya.

Pada uji coba modularitas, Modul berhasil di tampilkan pada antarmuka menu. Modul juga dapat dibuka dan berjalan sesuai dengan harapan.

Kemudian, berdasarkan hasil pengujian fungsionalitas, seluruh skenario berhasil dilakukan. Evaluasi terhadap pengujian fungsionalitas yang telah dilaksanakan dijelaskan sebagai berikut:

1. Fungsionalitas menampilkan aduan berjalan sesuai dengan yang diharapkan
2. Fungsionalitas mencari aduan berjalan sesuai dengan yang diharapkan.
3. Fungsionalitas menjawab aduan berjalan sesuai dengan yang diharapkan.

4. Fungsionalitas mengubah status aduan berjalan sesuai dengan yang diharapkan.
5. Fungsionalitas mengubah prioritas aduan berjalan sesuai dengan yang diharapkan.
6. Fungsionalitas memilih petugas berjalan sesuai dengan yang diharapkan.
7. Fungsionalitas mengembalikan aduan berjalan sesuai dengan yang diharapkan.
8. Fungsionalitas mengubah status *spam* aduan berjalan sesuai dengan yang diharapkan.
9. Fungsionalitas menampilkan lampiran sesuai dengan yang diharapkan.
10. Fungsionalitas mengunduh lampiran sesuai dengan yang diharapkan.
11. Fungsionalitas menampilkan peta klokasi aduan sesuai dengan yang diharapkan.
12. Fungsionalitas validasi SMS yang salah sesuai dengan yang diharapkan.
13. Fungsionalitas mengirim SMS ke petugas sesuai dengan yang diharapkan.
14. Fungsionalitas mengunduh laporan sesuai dengan yang diharapkan.

Dari keseluruhan hasil pengujian fungsionalitas pada modul aplikasi Tugas Akhir ini, seluruh skenario telah berhasil dilakukan. Modul aplikasi ini telah berjalan sesuai dengan yang diharapkan.

BAB VI
KESIMPULAN DAN SARAN

Dalam bab terakhir ini dijelaskan kesimpulan yang didapat dari pengerjaan Tugas Akhir ini beserta saran-saran yang dapat dipertimbangkan untuk pengembangan atau penelitian lebih lanjut.

6.1 Kesimpulan

Berdasarkan hasil uji coba yang telah dilakukan, terdapat beberapa kesimpulan yang bisa diambil, yaitu:

1. Aplikasi Modular Suara Warga berhasil dibangun pada perangkat Android 4.1 keatas.
2. Aplikasi dapat digunakan oleh petugas Pemerintah Kota Kediri dalam menanggapi setiap aduan, saran, dan informasi yang diberikan oleh masyarakat Kota Kediri.
3. Aplikasi ini dapat mengimplementasikan modularitas. oleh karena itu, aplikasi ini dapat menambah modul-modul aplikasi yand dimana modul aplikasi tersebut mengimplementasikan *interface* dari aplikasi ini.
4. Aplikasi ini diimplementasikan menurut rancangan untuk menanggapi setiap aduan, saran, dan informasi yang diberikan oleh masyarakat Kota Kediri dengan menggunakan aplikasi berbasis perangkat bergerak.

6.2 Saran

Adapun saran yang dapat dipertimbangkan untuk pengembangan atau penelitian lebih lanjut adalah sebagai berikut:

1. Meskipun pengguna Android sudah sangat marak, akan lebih baik lagi jika Tugas Akhir ini dapat

diimplementasikan juga ke dalam *platform* lain seperti iOS dan Windows Phone.

2. Antarmuka dapat dioptimalkan mengikuti perkembangan sistem operasi Android.
3. Untuk mengoptimalkan proses modularitas dibutuhkan riset lebih lanjut.

DAFTAR PUSTAKA

[1] BPS, "Sensus Kota Kediri", 2013, http://kedirikota.bps.go.id/?hal=publikasi , diakses pada 17 Desember 2014

[2] Giurca, Adrian, "JSON Rules", 2007, Brandenburg University of Technology, Germany

[3] Google, Inc, "Introduction to Android", http://developer.android.com/guide/index.html diakses pada 17 Desember 2014

[4] MySQL™, "MySQL Documentation: About MySQL Documentation" http://dev.mysql.com/doc/index-about.html diakses pada 17 Desember 2014

[5] Shunfu Hu, "Online Map Application Development Using Google Maps API, SQL Database, and ASP.NET" 2013, International Journal of Information and Communication Technology Research

[6] Tsenov, Martin "Web Services Example with PHP/SOAP", 2007 , International Conference on Computer Systems and Technologies, German,

BAB VI
KESIMPULAN DAN SARAN

Dalam bab terakhir ini dijelaskan kesimpulan yang didapat dari pengerjaan Tugas Akhir ini beserta saran-saran yang dapat dipertimbangkan untuk pengembangan atau penelitian lebih lanjut.

6.1 Kesimpulan

Berdasarkan hasil uji coba yang telah dilakukan, terdapat beberapa kesimpulan yang bisa diambil, yaitu:

1. Aplikasi Modular Suara Warga berhasil dibangun pada perangkat Android 4.1 keatas.
2. Aplikasi dapat digunakan oleh petugas Pemerintah Kota Kediri dalam menanggapi setiap aduan, saran, dan informasi yang diberikan oleh masyarakat Kota Kediri.
3. Aplikasi ini dapat mengimplementasikan modularitas. oleh karena itu, aplikasi ini dapat menambah modul-modul aplikasi yand dimana modul aplikasi tersebut mengimplementasikan *interface* dari aplikasi ini.
4. Aplikasi ini diimplementasikan menurut rancangan untuk menanggapi setiap aduan, saran, dan informasi yang diberikan oleh masyarakat Kota Kediri dengan menggunakan aplikasi berbasis perangkat bergerak.

6.2 Saran

Adapun saran yang dapat dipertimbangkan untuk pengembangan atau penelitian lebih lanjut adalah sebagai berikut:

1. Meskipun pengguna Android sudah sangat marak, akan lebih baik lagi jika Tugas Akhir ini dapat

LAMPIRAN A
PERANCANGAN DATA

Pada subbab ini dijelaskan tentang rancangan basis data yang akan digunakan pada Tugas Akhir ini. Basis data yang digunakan merupakan basis data pada sistem Suara Warga Kota Kediri yang dibangun menggunakan RDBMS MySQL.

1. Tabel aduan

Tabel aduan digunakan untuk menyimpan data seluruh aduan yang masuk ke sistem Suara Warga Kota Kediri. Dimana setiap baris pada tabel mewakili satu buah aduan dari warga. Atribut pada tabel ini dijelaskan pada Tabel A.1.

Tabel A.1 Tabel Aduan

Atribut	**Tipe**	***Null***	***Default***	**Keterangan**
id_aduan	int	Tidak	*autoincrement*	PK
no_identitas	varchar (45)	Ya	*null*	ktp pengadu
nama	varchar (45)	Ya	*null*	nama pengadu
alamat	varchar (45)	Ya	*null*	alamat pengadu
email	varchar (45)	Ya	*null*	email pengadu
no_hp	varchar (45)	Ya	*null*	no. telepon pengadu
tgl_lahir	date	Ya	*null*	tanggal lahir pengadu
topik	varchar (45)	Ya	*null*	topik aduan
isi	text	Ya	*null*	isi aduan

Atribut	Tipe	*Null*	*Default*	Keterangan
waktu	time stamp	Ya	*null*	waktu aduan ditulis
alamat_ip	varchar (45)	Ya	*null*	alamat ip pengadu
user_agent	varchar (2048)	Ya	*null*	*browser* pengadu jika aduan dikirim via *website*
longitude	varchar (45)	Ya	*null*	bujur informasi peta
latitude	varchar (45)	Ya	*null*	lintang informasi peta
rating	int	Ya	0	rating aduan
via_sms	tinyint	Ya	*null*	dikirim melalui sms
via_petugas	int	Ya	*null*	dikirim melalui petugas
info	int	Tidak	0	informasi aduan
info_detail	text	Ya	*null*	informasi detail aduan
spam	tinyint	Tidak	0	penanda jika aduan termasuk spam

2. Tabel detail_aduan

Tabel detail_aduan digunakan untuk menyimpan data isi aduan dan tanggapan dari petugas yang ada pada sistem Suara Warga Kota Kediri. Dimana setiap baris pada tabel mewakili satu buah aduan atau tanggapan. Atribut pada tabel ini dijelaskan pada Tabel A.2.

Tabel A.2 Tabel detail_aduan

Atribut	Tipe	*Null*	*Default*	Keterangan
id_detail_aduan	int	Tidak	*autoincrement*	PK
isi_detail	text	Ya	*null*	isi aduan atau tanggapan
waktu_detail	datetime	Ya	*null*	waktu aduan atau tanggapan ditulis

3. Tabel petugas

Tabel petugas digunakan untuk menyimpan data petugas yang memiliki akun pada sistem Suara Warga Kota Kediri. Dimana setiap baris pada tabel mewakili satu akun milik petugas. Atribut pada tabel ini dijelaskan pada Tabel A.3.

Tabel A.3 Tabel petugas

Atribut	Tipe	*Null*	*Default*	Keterangan
id_petugas	int	Tidak	*autoincrement*	PK
username_petugas	varchar (45)	Ya	*null*	username petugas
password_petugas	varchar (45)	Ya	*null*	password petugas

Atribut	Tipe	*Null*	*Default*	Keterangan
nama_ petugas	varchar (45)	Ya	*null*	nama petugas
email_ petugas	varchar (45)	Ya	*null*	email petugas
no_hp_ petugas	varchar (45)	Ya	*null*	no. telepon petugas
jobdesc_ petugas	varchar (45)	Ya	*null*	deskripsi petugas
bisa_sms	tinyint	Tidak	0	penanda jika petugas dapat mengirim sms

4. Tabel departemen

Tabel departemen digunakan untuk menyimpan data seluruh departemen yang ada pada struktur pemerintahan Kota Kediri. Dimana setiap baris pada tabel mewakili satu departemen. Atribut pada tabel ini dijelaskan pada Tabel A.4.

Tabel A.4 Tabel departemen

Atribut	Tipe	*Null*	*Default*	Keterangan
id_ departemen	int	Tidak	*autoincrement*	PK
nama_ departemen	varchar (128)	Ya	*null*	nama departemen
nama_ kepala	varchar (128)	Ya	*null*	nama kepala departemen
no_hp	varchar (45)	Ya	*null*	no. telepon kepala departemen

apakah_ mitra	tinyint	Ya	*null*	penanda jika suatu departemen termasuk mitra

5. Tabel *role*

Tabel role digunakan untuk menyimpan data *role* atau peranan dari setiap petugas. *Role* ini akan berpengaruh pada hak akses yang dimiliki oleh suatu petugas dalam mengakses sistem Suara Warga Kota Kediri. Dimana setiap baris pada tabel mewakili satu jenis *role*. Atribut pada tabel ini dijelaskan pada Tabel A.5.

Tabel A.5 Tabel *role*

Atribut	Tipe	*Null*	*Default*	Keterangan
id_role	int	Tidak	-	PK
nama_role	varchar (45)	Ya	*null*	nama role
deskripsi_ role	text	Ya	*null*	deskripsi role

6. Tabel kategori

Tabel kategori digunakan untuk menyimpan data kategori-kategori aduan. Dimana setiap baris pada tabel mewakili satu kategori aduan. Atribut pada tabel ini dijelaskan pada Tabel A.6.

Tabel A.6 Tabel kategori

Atribut	Tipe	*Null*	*Default*	Keterangan
id_kategori	int	Tidak	*autoincrement*	PK
nama_ kategori	varchar (45)	Ya	*null*	nama kategori

7. Tabel pengadu

Tabel pengadu digunakan untuk menyimpan data seluruh warga kota Kediri. Tabel ini berfungsi untuk verifikasi apakah warga yang menulis aduan adalah warga Kediri atau bukan. Dimana setiap baris pada tabel mewakili satu pengadu. Atribut pada tabel ini dijelaskan pada Tabel A.7.

Tabel A.7 Tabel pengadu

Atribut	Tipe	*Null*	*Default*	Keterangan
nik	varchar (50)	Tidak	-	PK
nama_lgkp	varchar (50)	Ya	*null*	nama pengadu
jenis_klmn	tinyint (4)	Ya	*null*	jenis kelamin pengadu
tmpt_lhr	varchar (50)	Ya	*null*	tempat lahir pengadu
tgl_lhr	date	Ya	*null*	tanggal lahir pengadu
pddk_akh	int	Ya	*null*	pendidikan terakhir
jenis_pkrjn	int	Ya	*null*	pekerjaan pengadu
no_prop	int	Ya	*null*	propinsi pengadu

Atribut	Tipe	*Null*	*Default*	Keterangan
no_kab	int	Ya	*null*	kabupaten pengadu
no_kec	int	Ya	*null*	kecamatan pengadu
no_kel	int	Ya	*null*	kelurahan pengadu

8. Tabel status

Tabel status digunakan untuk menyimpan data status aduan yang ditulis warga (apakah aduan sudah diterima departemen atau sudah selesai ditanggapi). Dimana setiap baris pada tabel mewakili satu jenis status aduan. Atribut pada tabel ini dijelaskan pada Tabel A.8.

Tabel A.8 Tabel status

Atribut	Tipe	*Null*	*Default*	Keterangan
id_status	int	Tidak	*autoincrement*	PK
nama_status	varchar (45)	Ya	*null*	nama status
class_status	varchar (45)	Ya	*null*	kelas status

9. Tabel status_aduan

Tabel status_aduan digunakan untuk menyimpan data *log* jika terjadi perubahan status aduan di tabel aduan. Dimana setiap baris pada tabel mewakili satu *log* dari suatu aduan. Atribut pada tabel ini dijelaskan pada Tabel A.9

Tabel A.9 Tabel aduan

Atribut	Tipe	*Null*	*Default*	Keterangan
id_status_ aduan	int	Tidak	*autoincrement*	PK
waktu_ status_ aduan	varchar (45)	Ya	*null*	Waktu *log* aduan

10. Tabel prioritas

Tabel prioritas digunakan untuk menyimpan data tingkat prioritas aduan yang ditulis warga. Dimana setiap baris pada tabel mewakili satu tingkat prioritas aduan. Atribut pada tabel ini dijelaskan pada Tabel A.10.

Tabel A.10 Tabel prioritas

Atribut	Tipe	*Null*	*Default*	Keterangan
id_prioritas	int	Tidak	*autoincrement*	PK
nama_ prioritas	varchar (45)	Ya	*null*	nama tingkatan prioritas
class_ prioritas	varchar (45)	Ya	*null*	kelas prioritas

11. Tabel kelurahan

Tabel kelurahan digunakan untuk menyimpan data kelurahan yang ada di Kota Kediri. Dimana setiap baris pada tabel mewakili satu kelurahan. Atribut pada tabel ini dijelaskan pada Tabel A.11.

Tabel A.11 Tabel kelurahan

Atribut	Tipe	*Null*	*Default*	Keterangan
id_ kelurahan	int	Tidak	*autoincrement*	PK
nama_ kelurahan	varchar (45)	Ya	*null*	nama kelurahan

12. Tabel kecamatan

Tabel kecamatan digunakan untuk menyimpan data kecamatan yang ada di Kota Kediri. Dimana setiap baris pada tabel mewakili satu kecamatan. Atribut pada tabel ini dijelaskan pada tabel Tabel A.12

Tabel A.12 Tabel kecamatan

Atribut	Tipe	*Null*	*Default*	Keterangan
id_ kecamatan	int	Tidak	*autoincrement*	PK
nama_ kecamatan	varchar (45)	Ya	*null*	nama kecamatan

13. Tabel *upload*

Tabel *upload* digunakan untuk menyimpan data berupa *path* dari *file* yang disimpan di *server*. *File* yang dimaksud adalah lampiran yang diunggah warga ketika mengisi aduan melalui *website*. Dimana setiap baris pada tabel mewakili satu *path* dari *file* lampiran. Atribut pada tabel ini dijelaskan pada Tabel A.13.

Tabel A.13 Tabel *upload*

Atribut	Tipe	*Null*	*Default*	Keterangan
id_upload	varchar (128)	Tidak	-	PK
orig_name	varchar (256)	Ya	*null*	nama asli dari *file*
path_upload	varchar (256)	Ya	*null*	*path* dari *file*
file_type	varchar (256)	Ya	*null*	tipe *file*

14. Tabel sms_tidak_valid

Tabel sms_tidak_valid digunakan untuk menyimpan data pesan singkat yang masuk namun tidak valid. Kriteria tidak valid yang dimaksud adalah ketika pesan singkat tidak sesuai dengan format penulisan yang sudah ditentukan pemerintah Kota Kediri. Dimana setiap baris pada tabel mewakili satu pesan singkat yang tidak valid. Atribut pada tabel ini dijelaskan pada tabel Tabel A.14

Tabel A.14 Tabel sms_tidak_valid

Atribut	Tipe	*Null*	*Default*	Keterangan
id	int	Tidak	-	PK
nomor_ pengirim	varchar (20)	Ya	-	nomor pengirim pesan singkat
urutan	varchar (20)	Ya	-	urutan pesan singkat

Atribut	Tipe	*Null*	*Default*	Keterangan
waktu	timestamp	Ya	*current timestamp*	waktu pesan singkat diterima
isi	text		-	isi pesan singkat

15. Tabel all_app

Tabel all_app digunakan untuk menyimpan *url* dari aplikasi *web* suara warga. Dimana setiap baris pada tabel mewakili satu buah *url*. Atribut pada tabel ini dijelaskan pada Tabel A.15.

Tabel A.15 Tabel all_app

Atribut	Tipe	*Null*	*Default*	Keterangan
id_all_app	int	Tidak	*autoincrement*	PK
url_app	varchar (256)	Ya	*null*	nama kategori
desc_app	varchar (45)	Ya	*null*	Deskripsi aplikasi

16. Tabel user_app

Tabel user_app digunakan untuk menyimpan *log* dari petugas yang mengakses aplikasi *web* suara warga. Dimana setiap baris pada tabel mewakili satu *user*. Atribut pada tabel ini dijelaskan pada tabel Tabel A.16.

Tabel A.16 Tabel user_app

Atribut	Tipe	*Null*	*Default*	Keterangan
id_all_app	int	Tidak	*autoincrement*	PK
id_app	int	Tidak	-	*url web*
petugas	int	Tidak	-	Petugas yang mengakses *url*

PENGEMBANGAN APLIKASI MODULAR SUARA WARGA PEMERINTAH KOTA KEDIRI BERBASIS ANDROID PADA PERANGKAT KOMUNIKASI BERGERAK

Nama Mahasiswa : Fahmi Armand Rachman
NRP : 5111 100 170
Jurusan : Teknik Informatika, FTIF-ITS
Dosen Pembimbing 1 : Dr.tech. Ir. R. V. Hari Ginardi, M.Sc.
Dosen Pembimbing 2 : Dwi Sunaryono, S.Kom., M.Kom.

ABSTRAK

Perkembangan teknologi di bidang perangkat bergerak terus berkembang dari tahun ke tahun. Hal ini dapat dilihat sebagai sebuah peluang untuk meningkatkan layanan masyarakat oleh Pemkot Kediri. Sehingga Pemkot Kediri menginisialisasi untuk membuat sebuah layanan yang bisa menampung aduan, saran, dan informasi dari masyarakat Kota Kediri.

Tugas Akhir ini membangun suatu aplikasi yang bersifat modular untuk menanggapi setiap aduan, saran dan informasi yang diberikan oleh masyarakat. Dengan menerapkan konsep modularitas, berbagai modul dimungkinkan dapat ditambahkan ke dalam aplikasi ini dengan mengimplementasikan interface-nya. Aplikasi ini dibangun dengan menggunakan Eclipse Juno 4.2.1 dan SDK Android. Dengan dikembangkannya aplikasi ini diharapkan bisa membantu Pemerintah Kota Kediri dalam melayani aduan, saran, dan informasi masyarakat dimanapun dan kapanpun.

Aplikasi yang telah dikembangkan diuji berdasarkan modularitas dan fungsionalitasnya. Pengujian dilakukan melalui skenario dengan metode kotak hitam yang mencerminkan fitur aplikasi. Hasil pengujian menunjukkan bahwa aplikasi bisa berjalan dengan baik dari segi modularitas dan fungsionalitasnya.

Kata kunci**: **Android, aduan, modularitas, perangkat bergerak

DEVELOPMENT OF MODULAR APPLICATION FOR SUARA WARGA PEMERINTAH KOTA KEDIRI BASED ON ANDROID MOBILE DEVICE

Student's Name	**: Fahmi Armand R.**
Student's ID	**: 5111 100 170**
Department	**: Informatics Departement, FTIF-ITS**
First Advisor	**: Dr.tech. Ir. R. V. Hari Ginardi, M.Sc.**
Second Advisor	**: Dwi Sunaryono, S.Kom., M.Kom.**

Abstract

Development of technology in mobile devices keep growing over the years. This can be seen as an opportunity for improve government service for people by the government of Kediri. So that the government of kediri initiate to make a service that can accomodate complaint, suggestion, and information from residents of Kediri.

This final project is to build a modular application for accommodate complaint, suggestion, and information from the people. By applying modularity, various of modules can be added to this application by implementing the interface class of this application. This application is built by Eclipse Juno 4.2.1 and Android SDK. With the development of this application, it's expected to help the government of Kediri to accommodate complaint, suggestion, and information anywhere and anytime.

The developed applications tested by modularity and functionality. This test is done by using scenario with a black box method that reflecting this application features. The result of this test shows that this application can perform well in terms of modularity and functionality.

Key word: Android, complaint, mobile device, modularity

KATA PENGANTAR

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

Bismillahirrohmanirohim.

Alhamdulillahirabil'alamin, segala puji hanya milik Allah SubhanahuWata'alla, atas segala rahmat dan karunia-Nya yang tak terhingga sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul "**PENGEMBANGAN APLIKASI MODULAR SUARA WARGA PEMERINTAH KOTA KEDIRI BERBASIS ANDROID PADA PERANGKAT KOMUNIKASI BERGERAK**" dengan baik dan tepat waktu.

Tugas Akhir ini dikerjakan demi memenuhi salah satu syarat guna memperoleh gelar Sarjana Komputer di Jurusan Teknik Informatika, Fakultas Teknologi Informasi, Institut Teknologi Sepuluh Nopember Surabaya. Penulis menyadari bahwa Tugas Akhir ini bukanlah tujuan akhir dari belajar karena belajar adalah sesuatu yang tidak terbatas.

Penulis mendapatkan banyak sekali doa, bantuan dan dukungan dari berbagai pihak dalam menyelesaikan Tugas Akhir ini. Atas berbagai bantuan dan dukungan tersebut, pada kesempatan ini penulis menyampaikan ucapan terima kasih yang sebesar-besarnya kepada:

1. Allah SWT atas limpahan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan Tugas Akhir ini dengan baik dan tepat waktu.
2. Bapak Rachmat Harahap dan Ibu Nasmiarti Anas yang tidak henti-hentinya memberikan dukungan, semangat, kasih sayang, serta selalu memberikan doa yang dipanjatkan untuk penulis.
3. Atika Zahra Rahmayanti, saudara kandung penulis yang selalu memberikan doa dan dukungan.
4. Bapak Dr.tech. Ir. R. V. Hari Ginardi, M.Sc. selaku dosen pembimbing 1 dan Bapak Dwi Sunaryono, S.Kom., M.Kom. selaku dosen pembimbing 2, yang telah memberikan

kepercayaan, dukungan, bimbingan, nasehat, serta semangat dikala penulis mengalami kesulitan.

5. Segenap staf dan dosen pengajar di Jurusan Teknik Informatika ITS.
6. Didik dan Ajong yang telah banyak memberikan bantuan dalam penyusunan Tugas Akhir ini.
7. Bachmid, Ibet, Fandi, Kemal, Melfa, Kaspul, Ajong, Ata, Megi, Agung, Rifi, Adi, Tegar, Satrio, Baskara, dan teman-teman seperjuangan dan sepermainan selama berkegiatan sehari-hari didalam dan diluar Kampus Perjuangan.
8. Keluarga Besar Angkatan 2011 yang telah menjadi keluarga kedua penulis selama menempuh kuliah di Teknik Informatika ITS.
9. Serta pihak-pihak lain yang namanya tidak dapat penulis sebutkan satu persatu.

Semoga Allah SWT memberkati dan membalas semua kebaikan yang telah dilakukan. Penulis menyadari masih banyak yang dapat dikembangkan pada Tugas Akhir ini. Oleh karena itu, penulis menerima setiap masukan dan kritik yang diberikan. Semoga Tugas Akhir ini dapat memberikan manfaat.

Surabaya, Juni 2015

Fahmi Armand Rachman

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR KODE SUMBER

BIODATA PENULIS

Fahmi Armand Rachman, dilahirkan di kota Jakarta, DKI Jakarta pada tanggal 17 April 1993. Penulis adalah anak kedua dari dua bersaudara. Penulis telah menempuh pendidikan formal yaitu di SDN Gondangdia 01 Pagi (1999-2005), SMP Negeri 19 Jakarta (2005-2008) dan SMA Negeri 70 Jakarta (2008-2011). Setelah lulus dari SMA Negeri 70 Jakarta pada tahun 2011, Penulis diterima di Jurusan Teknik Informatika ITS pada tahun 2011 dan terdaftar dengan NRP 5111100170. Di Jurusan Teknik Informatika ITS ini, Penulis mengambil Bidang Algoritma dan Pemrograman (AP). Penulis memiliki ketertarikan pada pengembangan *software* perangkat bergerak khususnya *platform* Android serta pengembangan *web*. Penulis dapat dihubungi melalui alamat *e-mail* di fahmi.armand17@gmail.com.

BAB I
PENDAHULUAN

1.1 Latar Belakang

Perkembangan teknologi informasi saat ini memiliki banyak sisi positif bagi kehidupan bermasyarakat. Ini bisa dilihat dengan semakin banyaknya masyarakat yang memakai perangkat bergerak atau telepon pintar dalam kehidupan sehari-hari. Misalnya, untuk berkomunikasi, menulis catatan, memesan tiket pesawat terbang, melakukan pengecekan rekening bank, dan hal-hal lainnya yang bisa dilakukan dengan mudah cukup lewat telepon pintar mereka.

Disamping memudahkan komunikasi antar sesama, perkembangan teknologi informasi juga menjadi peluang bagi pemerintah untuk meningkatkan layanan terhadap masyarakat. Peluang ini dilihat oleh Pemerintah Kota Kediri dengan meluncurkan layanan aduan masyarakat bernama "SURGA" atau akronim dari "Suara Warga" berbasis *web*.

Dengan jumlah warga yang tercatat di Badan Pusat Statistik yang mencapai ± 260.000 orang, diharapkan layanan ini dapat menampung semua jenis pengaduan, saran dan informasi demi mengakomodasi kebutuhan dari masyarakat kota Kediri. Untuk meningkatkan kualitas layanan, maka Pemerintah Kota Kediri berinisiatif untuk membuat layanan tersebut dalam bentuk perangkat bergerak yang bisa diakses telepon pintar petugas Pemerintah Kota Kediri.

Aplikasi perangkat bergerak suara warga ini dikembangkan pada kakas kerja Android dengan versi 4.1 (*Jelly Bean*) ke atas. Android dipilih menjadi kakas kerja karena mengingat sampai saat ini kakas kerja tersebutlah yang dipakai oleh kebanyakan masyarakat Indonesia. Aplikasi ini dikembangkan dengan desain arsitektur modular, dimana ini kita bisa menganti ataupun menambah suatu komponen (modul) tanpa mempengaruhi sistem yang lain dan tidak perlu untuk rekompilasi ulang. Aplikasi ini juga akan dibangun dengan basis data berupa MySQL dan *web service*

PHP untuk mendapatkan data dari basis data. Selain itu, suara warga ini juga dilengkapi dengan fitur Google Map untuk mengetahui lokasi aduan yang diadukan oleh masyarakat.

Diharapkan dengan adanya aplikasi berbasis perangkat bergerak ini maka Pemerintah Kota Kediri bisa lebih mudah dan cepat dalam menanggapi setiap aduan yang masuk. Sehingga diharapkan hal ini bisa meningkatkan pelayanan Pemerintah Kota Kediri terhadap aduan, saran, dan informasi yang diberikan oleh masyarakat Kota Kediri.

1.2 Rumusan Masalah

Rumusan masalah yang diangkat dalam Tugas Akhir ini dapat dipaparkan sebagai berikut:

1. Bagaimana merancang sistem untuk digunakan oleh petugas Pemerintah Kota Kediri dalam menanggapi setiap aduan, saran, dan informasi yang diberikan oleh masyarakat Kota Kediri dengan menggunakan aplikasi berbasis perangkat bergerak?
2. Bagaimana merancang dan mengimplementasi sistem yang bersifat modular?
3. Bagaimana mengimplementasi rancangan untuk menanggapi setiap aduan, saran, dan informasi yang diberikan oleh masyarakat Kota Kediri dengan menggunakan aplikasi berbasis perangkat bergerak?

1.3 Batasan Masalah

Permasalahan yang dibahas dalam Tugas Akhir ini memiliki beberapa batasan, diantaranya sebagai berikut:

1. Aplikasi ini adalah modul aplikasi perangkat bergerak yang berbasis pada sistem operasi Android dengan versi minimal 4.1 (*Jelly Bean*).
2. Aplikasi perangkat bergerak ini membutuhkan koneksi internet untuk beroperasi.

3. Pengguna aplikasi ini hanya untuk petugas Pemerintah Kota Kediri.
4. Progress realisasi tanggapan aduan tidak ada pada ruang lingkup sistem.

1.4 Tujuan dan Manfaat

Tujuan dari pengerjaan Tugas Akhir ini adalah sebagai berikut:

1. Untuk merancang sistem untuk digunakan oleh petugas Pemerintah Kota Kediri dalam menanggapi setiap aduan, saran, dan informasi yang diberikan oleh masyarakat Kota Kediri dengan menggunakan aplikasi perangkat bergerak.
2. Untuk merancang dan mengimplementasi sistem yang bersifat modular.
3. Untuk mengimplementasikan rancangan untuk menanggapi setiap aduan, saran, dan informasi yang diberikan oleh masyaratakat Kota Kediri dengan menggunakan aplikasi perangkat bergerak.

Tugas Akhir ini dikerjakan dengan harapan dapat memberikan manfaat pada bidang teknik informatika di pemerintahan dalam menerima dan menanggapi aduan, saran, dan informasi yang telah diperoleh dari warga menggunakan perangkat bergerak.

1.5 Metodologi

Ada beberapa tahap dalam proses pengerjaan Tugas Akhir ini. Berikut ini adalah tahap-tahap dalam pembuatan Tugas Akhir.

a. Studi Literatur

Tahap ini merupakan tahap pengumpulan informasi dan pembelajaran yang akan digunakan pada Tugas Akhir ini. Studi literatur meliputi diskusi dan pemahaman mengenai topik Tugas Akhir ini yaitu modularitas untuk aplikasi Android.Tahap ini merupakan perancangan sistem dengan menggunakan studi

literatur dan mempelajari konsep aplikasi yang akan dibuat. Dengan berbekal teori, metode, dan informasi yang sudah terkumpul pada tahap sebelumnya diharapkan dapat membantu dalam proses perancangan sistem.

b. Implementasi

Implementasi merupakan tahap membangun aplikasi, yaitu mengimplementasikan desain atau rancangan yang dibuat ke dalam baris kode program. Pengmbangan aplikasi ini dimulai dengan membuat fungsi pada *web service* untuk mengakses basis data pada server dengan menggunakan Bahasa pemrograman PHP. Kemudian dilanjutkan dengan membuat aplikasi klien Android dengan menggunakan bahasa pemrograman Java.

c. Uji Coba dan Evaluasi

Pada tahap ini dilakukan pengujian terhadap aplikasi yang dibuat menggunakan data ataupun kasus yang telah disiapkan. Tujuan pengujian ini adalah untuk menguji fungsionalitas dari aplikasi, mencari masalah yang mungkin muncul, dan melakukan perbaikan bila ada kekurangan.

d. Penyusunan Laporan Tugas Akhir

Tahap ini digunakan untuk membuat laporan Tugas Akhir yang berisi dokumentasi mengenai pembuatan serta hasil dari implementasi perancangan yang telah dibuat. Laporan Tugas Akhir ini bertujuan untuk mendokumentasikan pengerjaan Tugas Akhir dan menggambarkan keseluruhan proses pengerjaan Tugas Akhir dan dapat berguna bagi pembaca yang tertarik sebagai referensi untuk pengembangan lebih lanjut kedepannya.

1.6 Sistematika Penyusunan Laporan

Buku Tugas Akhir ini disusun dengan sistematika laporan sebagai berikut:

1. Bab I. Pendahuluan
   Bab ini meliputi latar belakang masalah, rumusan permasalahan, batasan masalah, tujuan dan manfaat pembuatan Tugas Akhir, metodologi yang digunakan, dan sistematika penyusunan laporan Tugas Akhir.

2. Bab II. Tinjauan Pustaka
   Bab ini meliputi dasar teori dan penunjang yang berkaitan dengan pokok pembahasan dan mendasari pembuatan Tugas Akhir ini.

3. Bab III. Perancangan Perangkat Lunak
   Bab ini membahas analisis dari sistem yang akan dibuat meliputi deskripsi umum perangkat lunak, spesifikasi kebutuhan, dan identifkasi pengguna. Kemudian membahas rancangan dari sistem yang dibuat meliputi rancangan skenario kasus penggunaaan, arsitektur sistem, data, antarmuka, dan proses aplikasi.

4. Bab IV. Implementasi
   Bab ini membahas implementasi dari desain sistem yang dilakukan pada tahap desain, meliputi *pseudocode* dan implementasi antarmuka dari perangkat lunak.

5. Bab V. Uji Coba dan Evaluasi
   Bab ini membahas uji coba dari perangkat lunak yang dibuat dengan melihat keluaran yang dihasilkan oleh perangkat lunak, analisis, dan evaluasi untuk mengetahui kemampuan perangkat lunak.

6. Bab VI. Kesimpulan dan Saran

   Bab ini berisi kesimpulan dari hasil uji coba yang dilakukan serta saran untuk pengembangan lebih lanjut perangkat lunak.

BAB II
TINJAUAN PUSTAKA

Bab ini berisi penjelasan teori-teori yang berkaitan dengan pengimplementasian perangkat lunak. Penjelasan ini bertujuan untuk memberikan gambaran secara umum terhadap sistem yang dibuat dan berguna sebagai penunjang dalam pengembangan.

2.1 Situs Sistem Layanan Pengaduan Masyarakat Kota Kediri

Kota Kediri yang berada di provinsi Jawa Timur dan memiliki jumlah yang tercatat di Badan Pusat Statistik yang mencapai ± 260.000 orang, sedang melakukan pembangunan besar-besaran di dalam hal teknologi informasi. Salah satu hasil dari pembangunan itu adalah sistem layanan pengaduan masyarakat kota kediri atau di kenal sebagai suara warga kota kediri.

Suara warga kota kediri ini merupakan wadah untuk menerima dan menanggapi aduan, saran, dan informasi yang telah diperoleh dari warga. Warga memiliki dua cara untuk dapat mengirim aduan, saran, dan informasi yaitu dengan cara pengirim pesan singkat ke nomor yang sudah ditentukan dengan format yang sudah ditentukan juga. Selain dengan mengirim pesan singkat, warga juga dapat menggunakan situs suara warga kota kediri untuk mengirim aduan, saran, dan informasi. Di dalam situs ini warga juga dapat mengirim informasi lokasi dan dokumen-dokumen yang dapat dijadikan bukti untuk mempercepat proses penanggapan aduan, saran, dan informasi tersebut.

Para Petugas juga menggunakan situs ini untuk menanggapi aduan-aduan yang sudah masuk. Petugas dapat mengklasifikasi aduan-aduan tersebut ke dalam dinas-dinas yang terkait perihal aduan tersebut. Setelah di klasifikasi, dinas-dinas terkait tersebut dapat menanggapi aduan dan

melakukan tindakan yang terkait aduan-aduan yang sudah masuk.

Gambar 2.1 merupakan halaman depan dari situs sistem layanan pengaduan masyarakat Kota Kediri.

Gambar 2.1 Halaman Depan Sistem Layanan Pengaduan Masyarakat Kota Kediri

2.2 Android SDK

Android SDK adalah *tools* API (*Aplication Programming Interface*) yang digunakan untuk mulai mengembangkan aplikasi pada kakas kerja Android menggunakan bahasa pemrograman Java. Android merupakan subset perangkat lunak untuk ponsel yang meliputi sistem operasi, *middleware* dan aplikasi kunci yang dirilis oleh Google. Saat ini disediakan Android SDK *(Software Development Kit)* sebagai alat bantu dan API untuk mulai mengembangkan aplikasi pada kakas kerja Android menggunakan bahasa pemrograman Java. Sebagai kakas kerja aplikasi netral, Android memberi Anda kesempatan untuk membuat aplikasi yang kita butuhkan yang bukan merupakan aplikasi bawaan telepon pintar. Beberapa fitur Android yang paling penting adalah:

- Kakas kerja aplikasi yang mendukung penggantian komponen dan *reusable*
- Mesin Virtual Dalvik dioptimalkan untuk perangkat bergerak
- *Integrated browser* berdasarkan *engine open source webkit*
- Grafis yang dioptimalkan dan didukung oleh *libraries* grafis 2D, grafis 3D berdasarkan spesifikasi *openGL ES* 1.0 (Opsional Akselerasi Perangkat Keras)
- SQLite untuk penyimpanan data
- Media Support yang mendukung audio, video dan gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF), GSM Telephony (tergantung perangakat bergerak)
- Bluetooth, EDGE, 3G, WiFi (tergantung perangakat bergerak)
- Kamera, GPS, kompas dan *accelerator* (tergantung perangakat bergerak)

Lingkungan pengembangan yang lengkap dan kaya, termasuk perangkat emulator, *tools* untuk *debuging*, profil dan kinerja memori, dan *plugin* untuk IDE Eclipse.

2.3 *Web Service*

Web service menurut *www.w3.org* mendefinisikan *web service* sebagai "sebuah perangkat lunak yang dapat teridentifikasi oleh URI dan memiliki *interface* yang didefinisikan, dideskripsikan, dan dimengerti oleh XML dan juga mendukung interaksi langsung dengan perangkat lunak yang lain dengan menggunakan pesan berbasis XML melalui protokol internet".

Web service adalah sebuah perangkat lunak yang tidak terpengaruh oleh kakas kerja, ia akan menyediakan *method-method* yang dapat diakses oleh jaringan. Ia juga akan

menggunakan XML untuk pertukaran data, khususnya pada dua entitas bisnis yang berbeda.

Definisi lain *web service* adalah perangkat lunak yang dirancang untuk mendukung interopabilitas mesin ke mesin yang dapat berinteraksi melalui jaringan. *Web service* memiliki antarmuka yang dijelaskan dalam format mesin-*processable* (khusus WSDL). Sistem lain berinteraksi dengan *web service* ditentukan oleh deskripsi dengan menggunakan pesan SOAP, biasanya disampaikan menggunakan HTTP dengan serialisasi XML dalam hubungannya dengan *web* lainnya yang terkait standar.

Dalam pengertian yang sederhana, XML *Web Service* dapat di definisikan sebagai aplikasi yang diakses oleh aplikasi yang lain. Mungkin orang berpendapat itu semacam situs, tetapi itu bukan demikian. Ada perbedaan-perbedaan yang membedakan *web service* dengan situs, diantaranya:

- *Web service* tidak memiliki *interface* atau antarmuka pengguna, sedangkan situs memilikinya.
- *Web service* dibuat untuk berinteraksi langsung dengan applikasi yang lain baik berbeda sistem operasi atau konsep sekalipun, sedangkan situs dibuat untuk berinteraksi langsung dengan pengguna.
- *Web service* Dibuat untuk bekerja pada semua tipe klien aplikasi atau perangkat *device*, sedangkan situs dibuat untuk bekerja pada *web browser*.

2.4 PHP

PHP: *Hypertext Preprocessor* adalah bahasa skrip yang dapat ditanamkan atau disisipkan ke dalam HTML. PHP banyak dipakai untuk memrogram situs yang dinamis. PHP dapat digunakan untuk membangun sebuah CMS.

Pada awalnya PHP merupakan kependekan dari *Personal Home Page* (situs personal). PHP pertama kali

dibuat oleh Rasmus Lerdorf pada tahun 1995. Pada waktu itu PHP masih bernama *Form Interpreted* (FI), yang wujudnya berupa sekumpulan skrip yang digunakan untuk mengolah data formulir dari sebuah situs.

Selanjutnya Rasmus merilis kode sumber tersebut untuk umum dan menamakannya PHP/FI. Dengan perilisan kode sumber ini menjadi sumber terbuka, maka banyak pemrogram yang tertarik untuk ikut mengembangkan PHP. Pada Juni 2004, Zend merilis PHP 5.0. Dalam versi ini, inti dari interpreter PHP mengalami perubahan besar. Versi ini juga memasukkan model pemrograman berorientasi objek ke dalam PHP untuk menjawab perkembangan bahasa pemrograman ke arah paradigma berorientasi objek. PHP digunakan untuk mengembangkan aplikasi suara warga sebagai *web service* karena PHP memiliki dukungan lebih baik untuk sistem manajemen basis data MySQL.

2.5 JSON

JSON (dilafalkan "Jason"), singkatan dari *JavaScript Object Notation* (Bahasa Indonesia: notasi objek *JavaScript*), adalah suatu format ringkas pertukaran data komputer. Formatnya berbasis teks serta digunakan untuk merepresentasikan struktur data sederhana dan larik asosiatif (disebut objek). Format JSON sering digunakan untuk mentransmisikan data terstruktur melalui suatu koneksi jaringan pada suatu proses yang disebut serialisasi. Aplikasi utamanya adalah pada pemrograman aplikasi situs AJAX dengan berperan sebagai alternatif terhadap penggunaan tradisional format XML.

Walaupun JSON didasarkan pada subset bahasa pemrograman *JavaScript* dan umumnya digunakan dengan bahasa tersebut, JSON dianggap sebagai format data yang tak tergantung pada suatu bahasa. Kode untuk pengolahan dan pembuatan data JSON telah tersedia untuk banyak jenis bahasa pemrograman. Situs *www.json.org* menyediakan daftar komprehensif pengikatan JSON yang tersedia, disusun menurut bahasa.

2.6 Google Map API V2

Google Maps API V2 adalah kumpulan API yang memungkinkan Anda menghamparkan data pengguna di peta khusus Google. Anda dapat membuat aplikasi yang bisa diakses di situs internet dan seluler menarik dengan kakas kerja pemetaan canggih dari Google, termasuk basis data citra satelit, *street view*, profil ketinggian, petunjuk arah mengemudi, peta dengan sentuhan gaya, demografi, analisis, dan basis data yang besar. Dengan cakupan global yang paling akurat di dunia, dan komunitas pemetaan yang aktif memperbarui setiap harinya, pengguna mendapatkan manfaat dari layanan yang ditingkatkan secara terus-menerus.

Dalam aplikasi suara warga ini, penggunaan Google Maps API V2 digunakan untuk mendapatkan lokasi dari aduan yang diadukan.

2.7 Basis Data MySQL

MySQL adalah sebuah implementasi dari sistem manajemen basis data relasional (RDBMS) yang didistribusikan secara gratis dibawah lisensi GPL (*General Public License*). Setiap pengguna dapat secara bebas menggunakan MySQL, namun dengan batasan perangkat

lunak tersebut tidak boleh dijadikan produk turunan yang bersifat komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam basis data yang telah ada sebelumnya. SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian basis data, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

Kehandalan suatu sistem basis data (DBMS) dapat diketahui dengan cara melihat pengoptimasinya dalam melakukan proses perintah-perintah SQL yang dibuat oleh pengguna maupun program-program aplikasi yang memanfaatkannya. Sebagai peladen basis data, MySQL mendukung operasi basis data transaksional maupun operasi basis data non-transaksional. Pada modus operasi non-transaksional, MySQL dapat dikatakan unggul dalam hal unjuk kerja dibandingkan perangkat lunak peladen basis data kompetitor lainnya. Namun pada modus non-transaksional tidak ada jaminan atas reliabilitas terhadap data yang tersimpan, karenanya modus non-transaksional hanya cocok untuk jenis aplikasi yang tidak membutuhkan reliabilitas data seperti aplikasi blogging berbasis *web* (wordpress), CMS, dan sejenisnya. Untuk kebutuhan sistem yang ditujukan untuk bisnis sangat disarankan untuk menggunakan modus basisdata transaksional, hanya saja sebagai konsekuensinya unjuk kerja MySQL pada modus transaksional tidak secepat unjuk kerja pada modus non-transaksional.

2.8 Modularitas

Dalam ilmu teknik, sistem modular adalah sistem yang membagi suatu elemen dari produk ataupun proses dan menetapkannya ke dalam modul-modul sesuai dengan perencanaan yang telah dibuat.

Menurut sudut pandang ilmu teknik, modularitas secara umum memiliki tiga tujuan, yaitu:

- Untuk mengelola kompleksitas suatu pengembangan
- Untuk memungkinkan pengerjaan secara paralel
- Untuk mengakomodasi ketidakpastian yang akan terjadi di masa depan

Modularitas mengakomodasi ketidakpastian yang terjadi di masa depan karena elemen tertentu pada rencang bangun modular dapat diubah selama aturan- aturan perancangan ditaati. Oleh karena itu, dalam arsitektur modular, modul baru dapat mengganti modul lama dengan mudah dan dengan biaya yang rendah.

IEEE mendefinisikan modul sebagai sebuah unit program yang diskrit dan diidentifikasi sehubungan dengan komputasi, menggabungkan dengan unit lain dan memuat: sebagai contoh, input atau output, *assembler*, *compiler*, *linkage editor* dan *executive routine*. Dalam web, *programmer* Drupal menjelaskan bahwa modul adalah istilah spesifik untuk sebuah tipe proyek, sedangkan *programmer* Mambo menuliskan bahwa modul adalah menampilkan informasi tertentu. *Programmer* Ruby mendefinisikan modul adalah sebuah kumpulan *method* dan *variable.*

Konsep Modularitas berhubungan dengan dekomposisi. Pengembangan sistem menjadi lebih sederhana karena hanya terfokus pada satu modul terlebih dahulu, baru dilakukan integrasi antar modul. Bahasa-bahasa pemrograman modern menyediakan cara-cara untuk dapat membuat sistem perangkat lunak secara modular demikian, yaitu:

- Cara untuk mengelompokkan bersama data dan metoda operasi
- Cara untuk mendefinisikan *interface*
- Cara untuk menyembunyikan rincian internal dari data dan metoda-metoda operasi
- Cara untuk kompilasi secara terpisah

Bahasa Java memungkinkan konsep pemrograman modular, enkapsulasi, dan abstraksi data dengan disediakannya skema-skema *class*, *interface*, dan *packages*.

2.9 *Black-Box Testing*

Black-box testing adalah metode pengujian perangkat lunak yang tes fungsionalitas dari aplikasi yang bertentangan dengan struktur internal atau kerja (lihat pengujian *white-box*). pengetahuan khusus dari kode aplikasi / struktur internal dan pengetahuan pemrograman pada umumnya tidak diperlukan. Uji kasus dibangun di sekitar spesifikasi dan persyaratan, yakni, aplikasi apa yang seharusnya dilakukan. Menggunakan deskripsi eksternal perangkat lunak, termasuk spesifikasi, persyaratan, dan desain untuk menurunkan uji kasus. Tes ini dapat menjadi fungsional atau non-fungsional, meskipun biasanya fungsional. Perancang uji memilih input yang valid dan tidak valid dan menentukan output yang benar. Tidak ada pengetahuan tentang struktur internal benda uji itu.

Metode uji dapat diterapkan pada semua tingkat pengujian perangkat lunak: unit, integrasi, fungsional, sistem. Dan penerimaan. Ini biasanya terdiri dari kebanyakan jika tidak semua pengujian pada tingkat yang lebih tinggi, tetapi juga bisa mendominasi unit testing juga.

Metode ujicoba *black-box* memfokuskan pada keperluan fungsional dari software. Karena itu ujicoba *black-box* memungkinkan pengembang software untuk membuat himpunan kondisi input yang akan melatih seluruh syarat-syarat fungsional suatu program. Ujicoba *black-box* bukan

merupakan alternatif dari ujicoba *white-box*, tetapi merupakan pendekatan yang melengkapi untuk menemukan kesalahan lainnya, selain menggunakan metode *white-box*. Ujicoba *black-box* berusaha untuk menemukan kesalahan dalam beberapa kategori, diantaranya:

1. Fungsi-fungsi yang salah atau hilang
2. Kesalahan *interface*
3. Kesalahan dalam struktur data atau akses database eksternal
4. Kesalahan performa
5. Kesalahan inisialisasi dan terminasi

BAB III
PERANCANGAN PERANGKAT LUNAK

Perancangan merupakan bagian penting dari pembuatan suatu perangkat lunak yang berupa perencanaan-perencanaan secara teknis aplikasi yang dibuat. Sehingga bab ini secara khusus akan menjelaskan perancangan sistem yang dibuat dalam Tugas Akhir ini. Berawal dari deskripsi umum aplikasi hingga perancangan proses, alur, dan implementasinya.

3.1 Analisis Sistem

Aplikasi suara warga Pemerintah Kota Kediri ini dibangun pada perangkat komunikasi bergerak sebagai alternatif lain dari situs suara warga yang telah ada dan dipergunakan saat ini. Dibangun pada perangkat komunikasi bergerak untuk memberikan akses untuk menanggapi aduan dimana saja dan kapan saja.

Untuk itu pada tahap analisis ini didefinisikan kebutuhan yang akan dipenuhi dalam pembuatan aplikasi ini. Berikut penjabaran bagian-bagian tahap anbalisis yang mencakup deskripsi umum sistem, spesifikasi kebutuhan fungsional, spesifikasi kebutuhan non-fungsional, dan identifikasi pengguna.

3.1.1 Deskripsi Umum Sistem

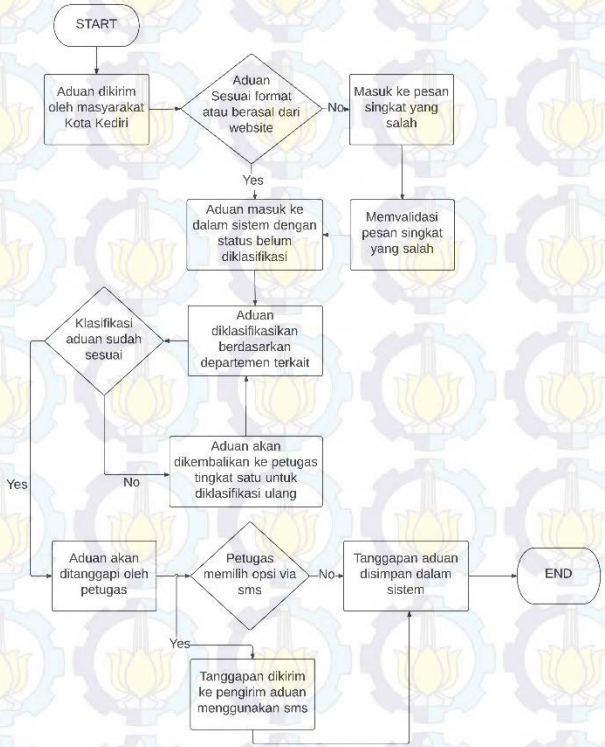
Pada Tugas Akhir ini akan dibangun suatu aplikasi Modular Suara Warga Kota Kediri yang berjalan pada perangkat bergerak berbasis Android. Tujuan dari aplikasi ini adalah untuk menjalankan modul-modul yang telah mengimplementasi *interface* dari aplikasi utama dan juga mempermudah akses petugas pemerintah Kota Kediri dalam menanggapi aduan, saran, dan informasi darimana saja dan kapan saja. Aplikasi ini diharapkan bisa meningkatkan

pelayanan pemerintah dan mempermudah menanggapi setiap aduan.

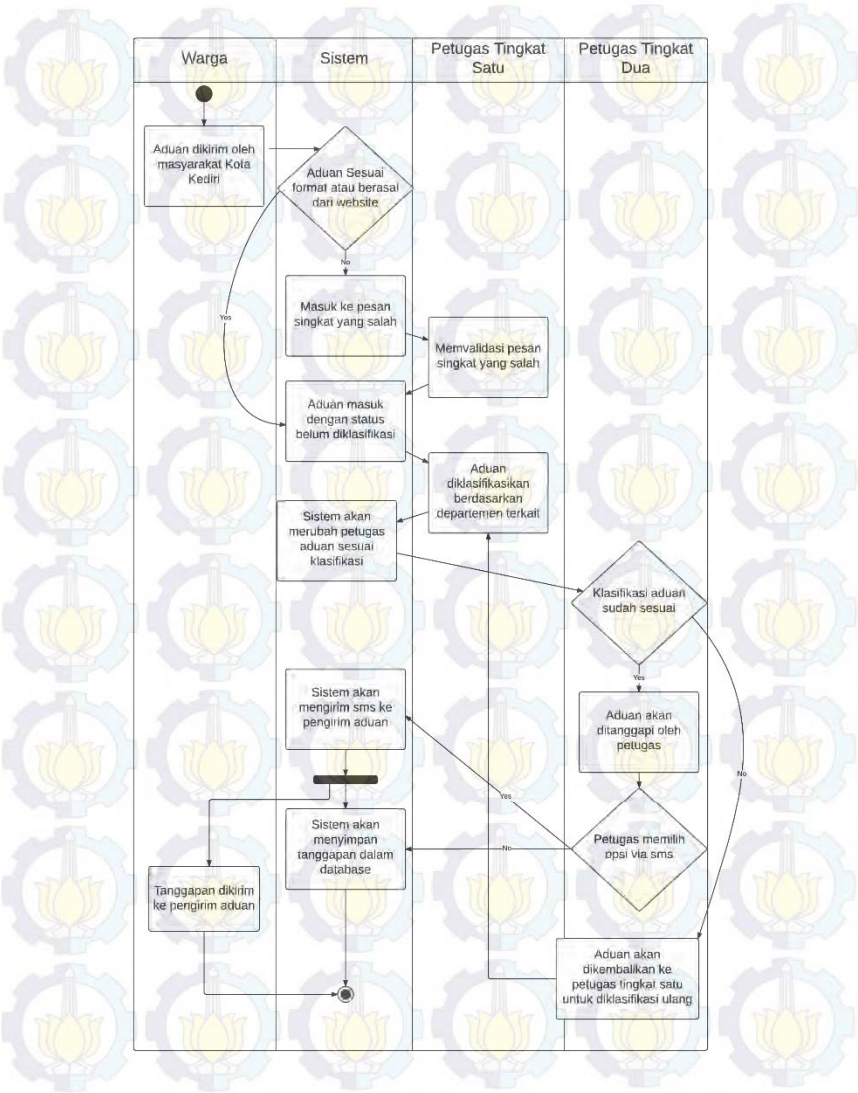
Pengguna dari aplikasi ini adalah pihak yang bertugas untuk menanggapi aduan, saran, dan informasi yakni petugas yang memiliki akses tingkat satu dan akses tingkat dua. Pengguna dengan akses satu dapat melihat aduan, mencari aduan, menindaklanjuti aduan, membalas aduan, melihat lampiran, melihat peta tempat lokasi aduan, mengirim pesan singkat kepada petugas, validasi pesan singkat yang salah, dan mengunduh laporan aduan. Sedangkan untuk petugas tingkat dua dapat melihat aduan, mencari aduan, menindaklanjuti aduan, membalas aduan, melihat lampiran, melihat peta tempat lokasi aduan, dan mengembalikan aduan kepada petugas satu. Untuk Petugas tingkat satu dapat menindaklanjuti aduan berupa memilih petugas tingkat dua untuk menindaklanjuti aduan, mengubah prioritas, mengubah status aduan, dan mengubah status aduan menjadi *spam*. Sedangkan untuk petugas tingkat dua dapat menindaklanjuti aduan berupa mengubah prioritas aduan, mengubah status aduan, mengubah status aduan menjadi *spam*, dan mengembalikan aduan ke petugas tingkat dua.

Proses bisnis sistem untuk menanggapi aduan berasal dari masyarakat yang mengirimkan aduan melalui pesan singkat sesuai format yang sudah ditentukan atau melalui *website* yang tersedia. Sistem akan mengecek apakah format dari pesan singkat itu benar atau tidak jika benar maka akan dimasukkan menjadi aduan sedangkan jika tidak maka dimasukkan menjadi pesan singkat tidak valid. Untuk pesan singkat yang tidak valid, dibutuhkan validasi oleh petugas untuk menjadi aduan. Aduan kemudian akan diklasifikasikan oleh petugas tingkat satu ke departemen-departemen(SKPD). Jika klasifikasinya tidak sesuai, maka petugas tingkat dua bisa mengembalikan aduan tersebut ke petugas tingkat satu untuk diklasifikasi ulang. Lalu aduan dapat di tanggapi oleh petugas.

Untuk diagram alir proses ini dapat dilihat pada Gambar 3.1 sedangkan untuk diagram aktifitas proses ini dapat dilihat pada Gambar 3.2.



Gambar 3.1 Diagram Alir Proses Bisnis



Gambar 3.2 Diagram Aktifitas Proses Bisnis Sistem

Untuk membangun modul aplikasi pada perangkat bergerak yang terhubung dengan sistem yang sudah ada, digunakan *web service* untuk dapat berkomunikasi ke basis data yang ada pada *server* seperti yang ada pada situs 'Suara Warga Kediri'.

3.1.2 Spesifikasi Kebutuhan Fungsional

Berdasarkan deskripsi umum sistem sistem diatas, maka dapat disimpulkan bahwa kebutuhan fungsionalitas dari aplikasi ini adalah sebagai berikut:

1. Menampilkan aduan
2. Mencari aduan
3. Menjawab aduan
4. Mengubah status aduan
5. Mengubah prioritas aduan
6. Memilih petugas aduan
7. Mengembalikan aduan
8. Mengubah status *spam* aduan
9. Menampilkan lampiran aduan
10. Mengunduh lampiran
11. Menampilkan peta lokasi aduan
12. Validasi Pesan Singkat yang salah
13. Mengirim Pesan Singkat ke petugas
14. Mengunduh laporan

3.1.3 Spesifikasi Kebutuhan Non-Fungsional

Berikut daftar kebutuhan non-fungsional yang harus dipenuhi agar aplikasi berjalan sesuai kebutuhan:

1. Koneksi internet
   Koneksi internet dibutuhkan untuk dapat mengakses basis data dari *server*.
2. Keamanan

Karena aplikasi pada situs menggunakan otentikasi, maka dibutuhkan otentikasi untuk mengakses sistem.

3.1.4 Identifikasi Pengguna

Berdasarkan deskripsi umum diatas, maka dapat diketahui bahwa pengguna yang akan menggunakan aplikasi ini adalah petugas atau pengguna yang memiliki akses tingkat satu dan tingkat dua. Petugas tingkat satu yaitu SKPD (Satuan Kerja Pemerintah Daerah) Hubungan masyarakat dan Wali Kota Kediri. Sedangkan petugas tingkat dua yaitu SKPD selain hubungan masyarakat. Penjelasan mengenai pengguna yang disebut aktor dalam sistem, dijelaskan pada Tabel 3.1.

Tabel 3.1 Identifikasi Pengguna

Nama Aktor	Definisi
Petugas Tingkat satu	Orang yang berinteraksi dengan sistem sebagai orang yang akan mengklasifikasikan aduan kepada SKPD-SKPD terkait dan juga dapat menindaklanjuti aduan.
Petugas Tingkat dua	Orang yang berinteraksi dengan sistem sebagai orang yang akan menindaklanjuti aduan yang di dapat dari petugas tingkat satu.

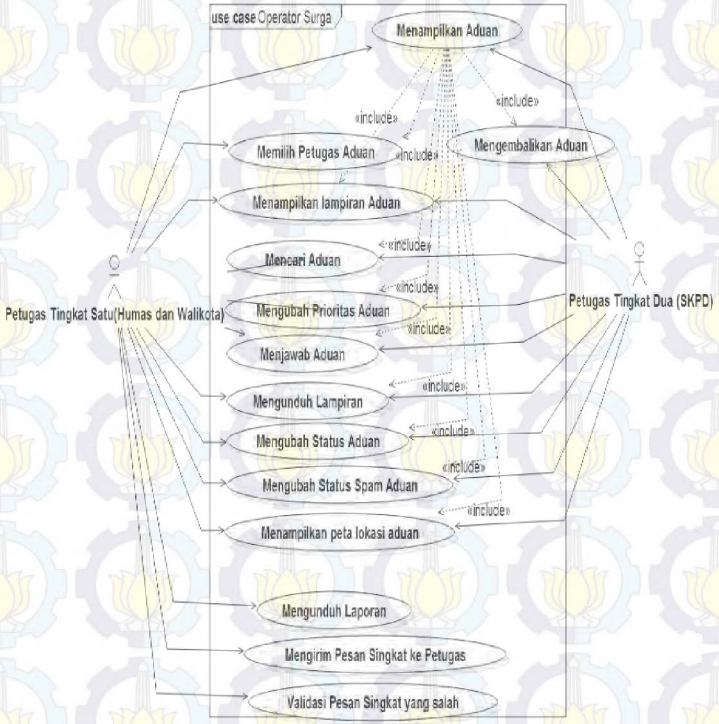
3.2 Perancangan Sistem

Tahap perancangan dalam subbab ini dibagi menjadi beberapa bagian yaitu perancangan diagram kebutuhan sistem, perancangan arsitektur sistem, perancangan data, perancangan modularitas, dan perancangan antarmuka sistem.

3.2.1 Perancangan Diagram Kebutuhan Sistem

Diagram kebutuhan sistem merupakan diagram kebutuhan yang menggambarkan fungsionalitas sistem dan aktor-aktornya.

Berdasarkan Gambar 3.3 ada dua aktor yang terlibat yaitu petugas tingkat satu dan tingkat dua. Pengguna berinteraksi dengan sistem untuk menjalankan segala fungsionalitas sistem. Diagram kebutuhan pada sistem secara umum dijelaskan pada Tabel 3.2.



Gambar 3.3 Diagram Kebutuhan

Tabel 3.2 Diskripsi Diagram Kebutuhan

No.	Kode	Nama	Keterangan
1.	UC-01	Menampilkan aduan	Pengguna dapat melihat Aduan yang di dapat dari masyarakat dalam bentuk *listview*.
2.	UC-02	Mencari aduan	Petugas dapat mencari aduan sesuai dengan id aduan, tanggal aduan, isi aduan, dan topik aduan.
3.	UC-03	Menjawab aduan	Pengguna dapat menjawab aduan. Pengguna dapat memilih menjawab aduan tersebut melalui pesan singkat atau situs jika aduan tersebut berasal dari pesan singkat.
4.	UC-04	Mengubah status aduan	Pengguna dapat mengubah prioritas dari aduan. Statusnya adalah diterima humas, diterima departemen terkait, diterima staff, dan selesai.
5.	UC-05	Mengubah prioritas aduan	Pengguna dapat mengubah prioritas dari aduan. Prioritasnya adalah rendah, sedang, dan tinggi.
6.	UC-06	Memilih petugas aduan	Petugas tingkat satu dapat memilih petugas tingkat dua untuk menindaklanjuti aduan.
7.	UC-07	Mengembalikan aduan	Petugas tingkat dua dapat mengembalikan aduan jika

No.	Kode	Nama	Keterangan
			aduan tersebut berada di luar SKPD dari petugas yang bersangkutan.
8.	UC-08	Mengubah status *spam* aduan	Pengguna dapat mengubah status aduan tersebut menjadi *spam*.
9.	UC-09	Menampilkan lampiran aduan	Pengguna dapat melihat dokumen-dokumen atau foto-foto yang dilampirkan di aduan.
10.	UC-10	Mengunduh lampiran	Pegguna dapat mengunduh file-file lampiran yang telah di unggah.
11.	UC-11	Menampilkan peta lokasi aduan	Pengguna dapat menampilkan peta lokasi aduan.
12.	UC-12	Mengirim Pesan Singkat ke petugas	Petugas tingkat satu dapat mengirim pesan singkat kepada petugas lain.
13.	UC-13	Validasi Pesan Singkat yang salah	Petugas tingkat satu dapat mem-validasi aduan yang salah *syntax* yang dikirim melalui pesan *singkat*.
14.	UC-14	Mengunduh laporan	Petugas tingkat satu dapat mengunduh laporan dari setiap aduan. Laporan tersebut dapat diunduh berdasarkan tahun.

3.2.2 Perancangan Arsitektur Sistem

Aplikasi ini dirancang memiliki dua sisi, yaitu sisi *server* dan sisi *client*. *Server* merupakan *web service* dan basis

data *server*, sedangkan *client* merupakan aplikasi yang beroperasi pada perangkat bergerak berbasis Android. *Server* bertugas untuk menerima, mengirim, dan menyimpan data dari *client* yang disimpan di dalam basis data. Kemudian mengolahnya dan mengirimkan hasilnya kepada *client*. *Client* bertugas untuk menerima input dari pengguna dan menampilkan data-data yang didapat dari basis data.

Berdasarkan perancangan arsitektur sistem pada Gambar 3.4, data-data disimpan dalam basis data dapat di peroleh dari *web service.* Untuk mendapatkan data-data, *web service* ini akan melakukan *query* basis data. Hasil *query* tersebut lalu dikirim ke perangkat Android dengan menggunakan *encoding* JSON. Lalu data tersebut ditampilkan di perangkat Android. Untuk mengirim data dari perangkat Android ke *web service,* Android menggunakan *method post* yang akan diterima oleh *web service*. *Web service* tersebut akan memasukan atau mengubah data tersebut ke dalam basis data.

Gambar 3.4 Perancangan Arsitektur Sistem

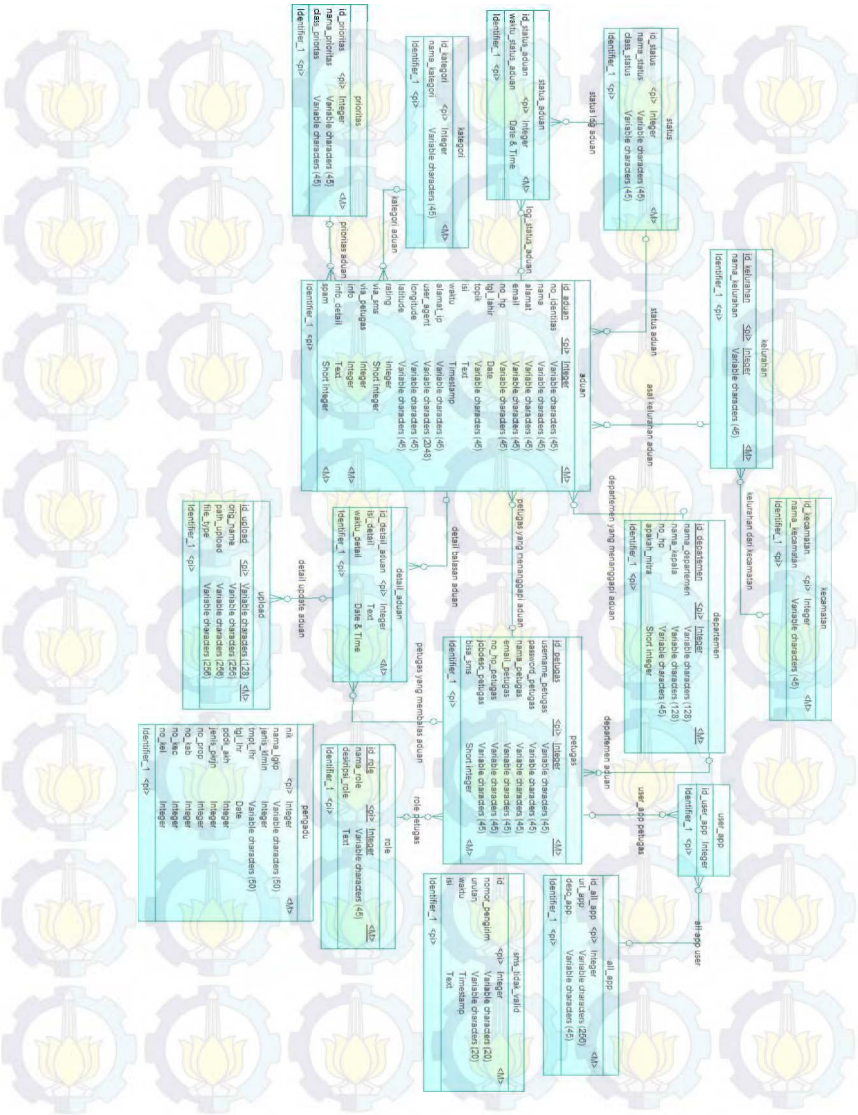
3.2.3 Perancangan Data

Perancangan data adalah hal yang penting dalam sistem, karena diperlukan data yang tepat agar sistem dapat beroperasi dengan benar. Sistem yang dibuat membutuhkan data masukan dan memberikan data keluaran.

Pada subbab ini dijelaskan tentang rancangan basis data yang digunakan pada modul aplikasi. Basis data yang ada pada sistem ini menggunakan RDBMS MySQL.

Basis data digunakan untuk menyimpan informasi yang dibutuhkan baik dalam aplikasi ini maupun situs Suara Warga Kota Kediri yang sudah ada. Basis data ini juga di pakai oleh modul-modul aplikasi ini.

Conceptual Data Model (CDM) adalah diagram yang menjelaskan desain basis data secara *independent* tanpa mengacu pada spesifikasi basis data tertentu. CDM berfungsi untuk menggambarkan tabel yang digunakan beserta relasinya Gambar 3.5 menunjukkan diagram *Conceptual Data Model* (CDM) untuk basis data pertama yang merupakan basis data Suara Warga Kota Kediri. Perancangan data untuk lebih detail dapat dilihat pada LAMPIRAN A PERANCANGAN DATA.



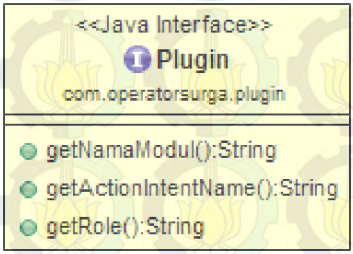
Gambar 3.5 Diagram CDM Basis Data

3.2.4 Perancangan Modularitas

Secara garis besar, modularitas yang dibangun dalam Tugas Akhir ini mirip dengan *launcher* yang ada di menu aplikasi Android. Modul-modul aplikasi tersebut hanya bisa dibuka melalui aplikasi ini.

Hal-hal yang harus disiapkan untuk membuat modul aplikasi ini yaitu aplikasi yang sudah mengimplementasikan interface dari aplikasi utama. Ditambah dengan sedikit mengubah Android *manifest* dari aplikasi ini supaya aplikasi ini tidak muncul di *launcher* Android dan bisa dijalankan di aplikasi utama.

Interface dari aplikasi utama ini memiliki tiga method yang masing-masing digunakan untuk nama modul, *activity* utama dari aplikasi modul, dan *role* yang dapat menggunakan modul ini. Kelas diagram untuk *interface* plugin ini ditunjukkan pada Gambar 3.6.



Gambar 3.6 Kelas Diagram dari *Interface* Plugin

3.2.5 Perancangan Antarmuka Sistem

Pada Tugas Akhir ini, antarmuka sistem hanya ada pada modul aplikasi *client*. Rancangan antarmuka sistem meliputi antarmuka login, menu modul, sub menu operator surga, manajemen aduan, detail aduan, balas aduan, tindak lanjut aduan, peta, list lampiran, gambar lampiran, *list* validasi sms, validasi sms, sms petugas, dan laporan.

3.2.5.1 Perancangan Antarmuka Login

Perancangan antarmuka login ditunjukkan pada Gambar 3.7. Antarmuka ini merupakan halaman pertama dalam aplikasi ini. Antarmuka ini berfungsi sebagai pembuktian keaslian pengguna untuk masuk ke dalam aplikasi.

Gambar 3.7 Perancangan Antarmuka Login

Komponen-komponen yang ada Gambar 3.7 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. Ikon dari aplikasi Suara Warga.
c. Dua *EditText* yang berfungsi untuk memasukan *username* dan *password* untuk pembuktian keaslian pengguna.
d. *Button login* yang berfungsi untuk masuk ke dalam aplikasi jika pembuktian keaslian berhasil.

e. *TextView* yang digunakan untuk menampilkan *copyright* dan alamat Pemerintah Kota Kediri.

3.2.5.2 Perancangan Antarmuka Menu Modul

Antarmuka menu modul berfungsi untuk menampilkan modul-modul yang ada di dalam aplikasi ini. Pada antarmuka ini terdapat *list* dari modul-modul yang sudah terdapat dalam aplikasi. Halaman ini ditunjukkan pada Gambar 3.8.

Gambar 3.8 Perancangan Antarmuka Menu

Komponen-komponen yang ada Gambar 3.8 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. Sebuah *ListView* yang berfungsi untuk menampilkan modul-modul yang sudah ada di dalam aplikasi.

3.2.5.3 Perancangan Antarmuka Sub Menu Operator Surga

Antarmuka sub menu operator surga merupakan antarmuka yang muncul ketika modul operator dipilih melalui antarmuka menu modul. Pada antarmuka ini terdapat empat sub menu. Sub menu tersebut adalah manajemen aduan, validasi sms, sms, dan laporan. Halaman tersebut ditunjukkan pada Gambar 3.9.

Gambar 3.9 Perancangan Antarmuka Halaman Sub menu Operator Surga

Komponen-komponen yang ada Gambar 3.9 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. *ListView* yang menampilkan sub menu-sub menu dari modul operator suara warga.
c. Satu buah ikon dan *text* untuk setiap sub menu yang berfungsi penamaan dari sub menu tersebut.

3.2.5.4 Perancangan Antarmuka Manajemen Aduan

Antarmuka manajemen aduan merupakan antarmuka yang muncul ketika sub menu manajemen aduan dipilih melalui antarmuka sub menu operator surga. Pada antarmuka ini terdapat empat *tab* jika *login* sebagai petugas tingkat satu dan terdapat tiga *tab* jika *login* sebagai petugas tingkat dua. Tab untuk petugas tingkat satu adalah aduan belum diklasifikasi, aduan sudah diklasifikasi, aduan dikembalikan, dan seluruh aduan. Sedangkan untuk tab untuk petugas tingkat dua adalah aduan belum diklasifikasi, aduan sudah diklasifikasi, dan aduan dikembalikan. Halaman tersebut ditunjukkan pada Gambar 3.10.

Gambar 3.10 Perancangan Antarmuka Halaman Manajemen Aduan

Komponen-komponen yang ada Gambar 3.8 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.

b. *Tab* yang digunakan untuk menampilkan judul dari *list* aduan.
c. *ListView* yang digunakan untuk menampilkan aduan.
d. Tiap aduan terdari dari enam teks yang digunakan untuk menampilkan id aduan, tanggal aduan, topik aduan, departemen aduan, status aduan, dan prioritas aduan.

3.2.5.5 Perancangan Antarmuka Detail Aduan

Antarmuka detail aduan merupakan antarmuka yang muncul ketika salah satu aduan dipilih melalui antarmuka manajemen aduan. Halaman tersebut ditunjukkan pada Gambar 3.11.

Gambar 3.11 Perancangan Antarmuka Halaman Detail Aduan

Komponen-komponen yang ada Gambar 3.11 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.

b. Empat buah *textView* yang digunakan untuk menampilkan id aduan, topik aduan, nama pengirim, dan dapartemen aduan.
c. Empat buah *button* yang digunakan untuk navigasi ke menu lain. Menu lainnya adalah balas aduan, lihat peta, tindak lanjuti aduan, dan lihat lampiran.
d. Ikon dan *textView* yang digunakan untuk menampilkan waktu, dikirim melalui, prioritas, dan status aduan.
e. *ListView* yang digunakan untuk menampilkan isi dari aduan dan juga balasan dari petugas.
f. Setiap isi dari sebuah *list* di dalam *listView* terdapat nama pengirim aduan atau petugas yang menjawab aduan, waktu, dan isi dari aduan.

3.2.5.6 Perancangan Antarmuka Jawab Aduan

Antarmuka jawab aduan merupakan antarmuka yang muncul ketika tombol jawab aduan dipilih di detail aduan. Pada antarmuka ini terdapat empat sub menu antarmuka ini digunakan untuk menjawab aduan. Halaman tersebut ditunjukkan pada Gambar 3.12.

Gambar 3.12 Perancangan Antarmuka Jawab Aduan

Komponen-komponen yang ada Gambar 3.12 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. *ListView* yang digunakan untuk menampilkan isi dari aduan dan juga balasan dari petugas.
c. Setiap isi dari sebuah list di dalam *listView* terdapat nama pengirim aduan atau petugas yang menjawab aduan, waktu, dan isi dari aduan.
d. Dua *radio button* yang akan dipilih untuk membalas aduan tersebut melalui situs atau pesan singkat. *Radio button* ini akan hilang jika aduan berasal dari situs.
e. *EditText* yang digunakan untuk memasukan input balasan aduan.
f. *Button* balas yang digunakan untuk memasukan balasan tersebut ke dalam basis data.

3.2.5.7 Perancangan Antarmuka Tindak Lanjut Aduan

Antarmuka tindak lanjut aduan merupakan antarmuka yang muncul ketika tombol tindak lanjuti dipilih di detail aduan. Untuk petugas tingkat satu, antarmuka ini digunakan untuk memilih petugas yang akan menindak lanjuti aduan, mengubah prioritas, dan mengubah status aduan. Halaman ini ditunjukkan pada Gambar 3.1311.

Gambar 3.13. Perancangan Antarmuka Tindak Lanjut Aduan

Komponen-komponen yang ada Gambar 3.11 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. Ikon dan *TextView* berfungsi untuk menampilkan nama petugas, prioritas, dan status dari aduan. Ikon jika dipilih akan menampilkan *list* dari nama petugas, prioritas, dan status yang dapat dipilih. Untuk petugas tingkat dua, ikon dan *textView* nama petugas akan dihilangkan.

c. Tiga *button* yang digunakan untuk *update* aduan, menjadikan aduan tersebut menjadi *spam,* dan mengembalikan aduan tersebut ke petugas tingkat satu. Untuk petugas tingkat satu, *button* kembalikan akan dihilangkan.

3.2.5.8 Perancangan Antarmuka Peta

Antarmuka peta merupakan antarmuka yang muncul ketika tombol lihat peta dipilih di detail aduan. Antarmuka ini digunakan untuk menampilkan lokasi peta dari aduan yang dilaporkan. Halaman tersebut ditunjukkan pada Gambar 3.14.

Gambar 3.14 Perancangan Antarmuka Peta

Komponen-komponen yang ada Gambar 3.12 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. *Fragment* Google Maps yang digunakan untuk menampilkan peta.

3.2.5.9 Perancangan Antarmuka Lihat Lampiran

Antarmuka lihat lampiran merupakan antarmuka yang muncul ketika tombol lihat lampiran dipilih di detail aduan. Antarmuka ini digunakan melihat *list* lampiran yang dilampirkan. Halaman tersebut ditunjukkan pada Gambar 3.15.

Gambar 3.15 Perancangan Antarmuka Lihat Lampiran

Komponen-komponen yang ada Gambar 3.13 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. *ListView* yang digunakan untuk menampilkan *list* aduan.
c. Dalam satu lampiran terdiri atas tiga *TextView* yang menampilkan tipe *file*, tanggal, dan nama *file*.

3.2.5.10 Perancangan Antarmuka Lihat Gambar

Antarmuka lihat gambar merupakan antarmuka yang muncul ketika tombol ‘lihat’ dipilih pada antarmuka ‘lihat

lampiran'. Antarmuka ini digunakan melihat gambar. Halaman tersebut ditunjukkan pada Gambar 3.16.

Gambar 3.16. Perancangan Antarmuka Peta

Komponen-komponen yang ada Gambar 3.16 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. *ImageView* yang menampilkan lampiran yang bertipe gambar.

3.2.5.11 Perancangan Antarmuka List Validasi SMS

Antarmuka *list* validasi sms merupakan antarmuka yang muncul ketika sub menu validasi sms dipilih di sub menu Operator Surga. Antarmuka ini digunakan melihat kumpulan sms yang tidak valid. Halaman tersebut ditunjukkan pada Gambar 3.17.

Gambar 3.17 Perancangan Antarmuka List Validasi SMS

Komponen-komponen yang ada Gambar 3.17 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. *ListView* yang berfungsi untuk menampilkan list dari sms yang tidak valid.
c. Satu sms tidak valid di dalam list tersebut terdapat empat *TextView* yaitu berfungsi untuk menampilkan id, nomor pengirim, isi sms, dan tanggal pengiriman.

3.2.5.12 Perancangan Antarmuka Validasi SMS

Antarmuka validasi sms merupakan antarmuka yang muncul ketika salah satu sms tidak valid tersebut dipilih divalidasi sms. Antarmuka ini digunakan melihat validasi sms yang tidak valid. Halaman tersebut ditunjukkan pada Gambar 3.18.

Gambar 3.18 Perancangan Antarmuka Validasi SMS

Komponen-komponen yang ada Gambar 3.18 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. Terdapat empat *editText* yang berfungsi untuk memasukkan input dari pengguna. Inputnya berupa nomor ktp, nama, nomor telepon, dan isi.
c. *Button* validasi yang digunakan untuk mengubah sms yang tidak valid tersebut menjadi aduan.

3.2.5.13 Perancangan Antarmuka SMS

Antarmuka sms aduan merupakan antarmuka yang muncul ketika sub menu validasi sms dipilih di sub menu operator surga. Antarmuka ini digunakan mengirim sms kepada petugas. Halaman tersebut ditunjukkan pada Gambar 3.19.

Gambar 3.19 Perancangan Antarmuka SMS

Komponen-komponen yang ada Gambar 3.16 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. Dua *EditText* yang satu berfungsi untuk memilih dinas mana yang akan dikirim sms dan yang kedua untuk menulis pesan yang akan dikirim.
c. Button kirim sms digunakan untuk mengirim sms tersebut ke petugas terkait.

3.2.5.14 Perancangan Antarmuka Laporan

Antarmuka laporan merupakan antarmuka yang muncul ketika sub menu laporan dipilih di sub menu operator surga. Antarmuka ini digunakan mengunduh laporan aduan. Halaman tersebut ditunjukkan pada Gambar 3.20.

Gambar 3.20 Perancangan Antarmuka Laporan

Komponen-komponen yang ada Gambar 3.20 adalah sebagai berikut :

a. *TextView* judul halaman digunakan untuk menampilkan judul menu.
b. *EditText* yang digunakan untuk memilih tahun laporan yang akan diunduh.
c. *Button download* yang digunakan untuk mengunduh laporan berdasarkan tahun yang dipilih.

BAB IV
IMPLEMENTASI PERANGKAT LUNAK

Proses implementasi sistem ini dilakukan setelah melewati proses analisis dan perancangan perangkat lunak. Dalam Bab ini akan dibahas mengenai algoritma, *pseudocode,* diagram alur serta implementasi dari perancangan antarmuka yang terdapat dalam perangkat lunak sebagaimana telah dibahas pada Bab III.

4.1 Lingkungan Implementasi

Dalam merancang dan mengimplementasikan perangkat lunak ini, digunakan beberapa perangkat pendukung sebagai berikut.

4.1.1 Lingkungan Implementasi Perangkat Keras

Spesifikasi perangkat keras yang digunakan pada lingkungan pengembangan perangkat lunak ini adalah sebagai berikut:

- Komputer Laptop Asus N-43SL
  - Windows 7 Home Premium 64-bit,
  - Prosesor Intel® Core(TM) i5-2430M CPU @ 2.40GHz, dan
  - RAM 8.00 GB.
- Perangkat Bergerak Asus Zefone 5
  - Sistem Operasi: Android v4.4 (Kitkat),
  - CPU: Dual core 2GHz Intel Atom Z2580
  - Memory internal: 16 GB,
  - RAM: 2.00 GB,
  - Speed: HSDPA, 14.4 Mbps; HSUPA, 5.76 Mbps, dan
  - WLAN: Wi-Fi 802.11 b/g/n, Wi-Fi hotspot

4.1.2 Lingkungan Implementasi Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan dalam pengembangan sistem adalah sebagai berikut:

- Microsoft Windows 7 *Home Premium* sebagai sistem operasi
- Eclipse Juno v4.2.1 sebagai IDE untuk mengimplementasikan modul aplikasi *mobile*
- MySQL sebagai basis data *server*
- StarUML dan www.draw.io untuk merancang diagram sistem

4.2 Implementasi Proses Modularitas

Pada subbab ini akan dijelaskan mengenai implementasi proses modularitas yang digunakan untuk membuka aplikasi yang mengimplementasi *interface* dari aplikasi utama. Proses ini dibagi menjadi dua yaitu proses pembuatan modul aplikasi dan proses memuat modul-modul di aplikasi utama.

4.2.1 Implementasi Proses Pembuatan Modul Aplikasi

Implementasi proses pembuatan modul aplikasi diimplementasikan dengan cara menghapus *category launcher* di *androidmanifest.xml* pada modul aplikasi. Kemudian tambahkan sebuah *action* di *intent* pertama dari modul aplikasi yang berisi *package* utama dari aplikasi tersebut ditambah dengan ".MAIN". Impor *file jar interface* aplikasi utama ke dalam *project* modul aplikasi. Implementasikan *interface* tersebut dengan mengisi nama modul, nama *intent action* yang di dapat dari *androidmanifest,* dan *role* yang boleh membuka modul tersebut. Pasang modul aplikasi tersebut ke dalam perangkat Android. Buat *file manifest.mf* yang berisi versi *manifest* tersebut ditambah dengan Kode Sumber 4.1. Ekspor kelas yang mengimplementasi *interface* aplikasi utama dengan menggunakan *manifest* yang telah dibuat menjadi *file jar*.

Compile file jar tersebut menggunakan *dex*. Hasil dari *compile* adalah sebuah *file* yang berekstensi "*.dex"*. Masukkan *file* yang berekstensi "*.dex"* ke dalam *file jar.* Salin *file jar* tersebut ke perangkat Android ke dalam direktori yang sudah ditentukan. Proses ini dapat digambarkan pada Gambar 4.1.

```
Module-Class: "tempat kelas yang
mengimplementasikan interface aplikasi
utama"
```

Kode Sumber 4.1 Kode yang Tambah di Manifest

Gambar 4.1 Diagram Alir Pembuatan Modul Aplikasi

4.2.2 Implementasi Proses Memuat Modul di Aplikasi Utama

Implementasi proses memuat modul aplikasi digunakan untuk memuat modul-modul aplikasi untuk dapat dibuka oleh aplikasi utama. Modul aplikasi ini juga harus mengimplementasikan *interface* dari aplikasi utama dan modul aplikasi tersebut sudah terpasang di perangkat Android. Hasil ekspor implementasi *interface* tersebut harus di *copy* ke *directory* perangkat Android yang sudah ditentukan. Proses memuat modul di aplikasi ini membaca setiap *file* yang berekstensi ".jar" yang diambil dari direktori yang ditentukan. *File–file* ini lalu di-*cast* menjadi kelas *jarFile.*Setelah itu kelas tersebut diambil atribut *manifest* yang memiliki *value* "Module-Class" dan disimpan ke dalam sebuah *string*. Lalu kita *load file jar* ini dengan menggunakan kelas *dexClassLoader.* Setelah *load*, kelas tersebut di-*cast* ke *interface.* Setelah *cast,* maka kelas ini dimasukkan ke dalam *list* yang menyimpan semua modul aplikasi ini. Proses ini dapat ditunjukkan pada Kode Sumber 4.21.

```
1  ModuleLoader() {
2    file ← directory "/plugin/"
3    List<PluginModel> plugins ← empty list
4    file[] listfile  ← get file ends with
5  .jar
6      FOR i = 1 to listfile length{
7      jarFile manifest ← listfile[i]
8      String className ← get manifest
9  attribute with value "Module Class"
10     Instance classLoader using dexLoader
11     Class<?> claz ← load class using
12 classLoader with className
13     Plugin tempPlugin ← cast
14 claz.newInstance() to plugin
```

```
    Plugins.add(new
PluginModel(tempPlugin.getNamaModul(),
 tempPlugin.getActionIntentName(),
 tempPlugin.getRole())
 }
Return plugins
}
```

Kode Sumber 4.2 Implementasi Proses Memuat Modul di Aplikasi Utama

4.3 Implementasi Antarmuka

Pada subbab ini dijelaskan implementasi tampilan antarmuka yang telah dibahas pada subbab 3.2.5.

4.3.1 Implementasi Antarmuka Login

Antarmuka *login* merupakan antarmuka yang berfungsi sebagai pembuktian keaslian untuk masuk ke aplikasi. Pengguna harus memasukan *username* dan *password* yang sudah ada di dalam sistem. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.3 merupakan bentuk tampilan dari antarmuka *login*.

Gambar 4.2 Implementasi Antarmuka *Login*

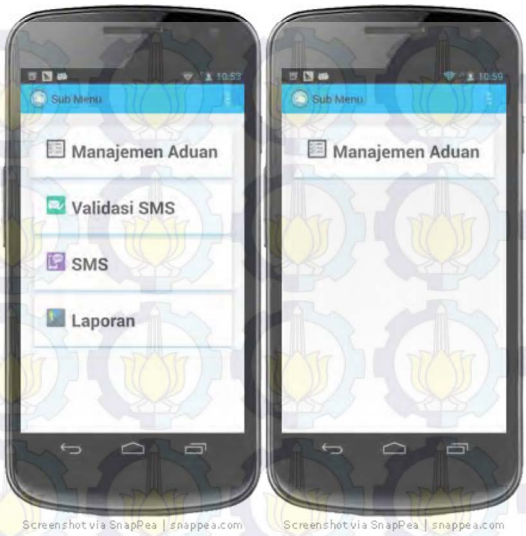
4.3.2 Implementasi Antarmuka Menu

Antarmuka menu merupakan antarmuka pertama yang muncul jika pengguna berhasil dibuktikan keasliannya ke dalam sistem. Pada antarmuka ini pengguna dapat melihat menu-menu yang ada dalam aplikasi ini. Menu-menu yang ditampilkan merupakan modul-modul yang ada dalam aplikasi. Tidak semua modul akan terlihat di menu ini. Modul yang terlihat tergantung pada pengguna yang telah login. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.3 merupakan tampilan antarmuka menu.

Gambar 4.3. Implementasi Antarmuka Menu

4.3.3 Implementasi Antarmuka Sub Menu

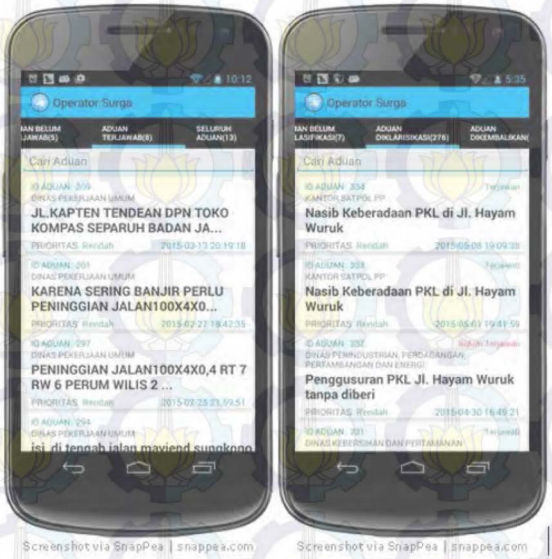
Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan *list* operator surga di antarmuka menu. Pada antarmuka ini pengguna dapat melihat sub menu yang ada dalam aplikasi ini. Sub yang ditampilkan akan berbeda tergantung kepada petugas mana yang telah login. Jika petugas tingkat satu yang *login* mana akan terdapat empat sub menu sedangkan untuk petugas tingkat dua hanya terdapat satu sub menu. Sub menu untuk petugas tingkat satu adalah manajemen aduan, validasi sms, sms, dan laporan. Untuk petugas tingkat dua sub menunya hanya manejemen aduan. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.4 merupakan tampilan antarmuka sub menu.



Gambar 4.4. Implementasi Antarmuka Sub menu

4.3.4 Implementasi Antarmuka Manajemen Aduan

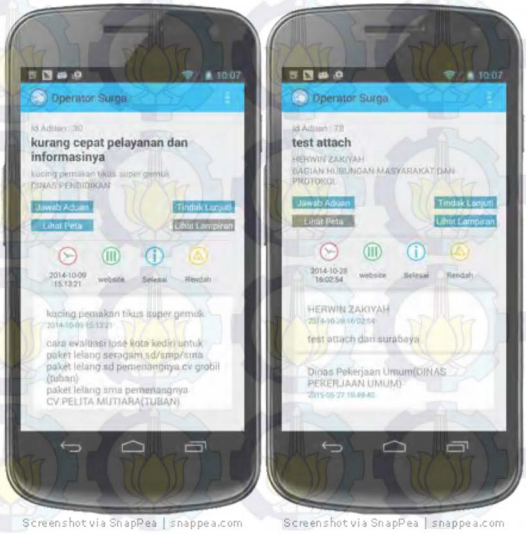
Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan *list* manajemen aduan di antarmuka sub menu. Pada antarmuka ini pengguna dapat melihat aduan-aduan yang ada dalam sistem. Untuk petugas tingkat satu, aduan-aduan tersebut dibagi menjadi 4 tab. Tab tersebut adalah aduan belum diklasifikasi, aduan diklasifikasi, aduan dikembalikan, dan seluruh aduan sedangkan untuk petugas tingkat dua memiliki 3 tab dan aduannya merupakan aduan yang sudah di arahkan ke SKPD (Satuan Kerja Pemerintah Daerah) terkait. Tab tersebut adalah aduan belum terjawab, aduan terjawab, dan seluruh aduan. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.5 merupakan tampilan antarmuka manajemen aduan.



Gambar 4.5 Antarmuka Manajemen Aduan

4.3.5 Implementasi Antarmuka Detail Aduan

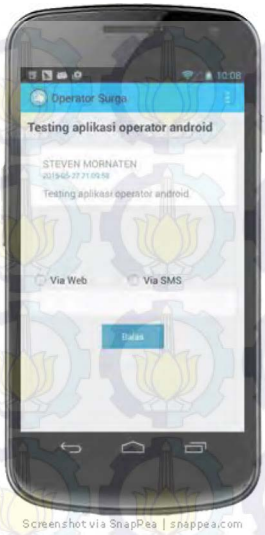
Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan salah satu aduan di antarmuka manajemen aduan. Pada antarmuka ini pengguna dapat melihat satu aduan secara detail. Hal-hal yang dapat dilihat adalah id aduan, topik aduan, nama pengirim aduan, departemen yang terkait dengan aduan, waktu, dikirim melalui, status, prioritas, dan sebuah *list* yang berisi isi aduan dan balasan (jika sudah dibalas). Terdapat empat *button* di antarmuka ini yang berfungsi untuk navigasi ke antarmuka yang lain. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.6 merupakan tampilan antarmuka detail aduan.



Gambar 4.6. Implementasi Antarmuka Detail Aduan

4.3.6 Implementasi Antarmuka Jawab Aduan

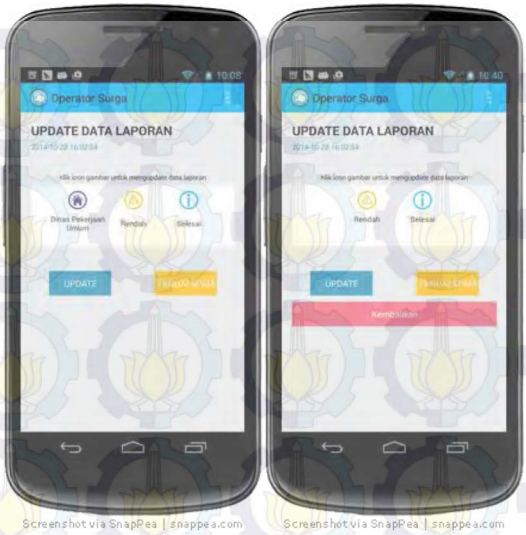
Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan *button* jawab aduan di antarmuka detail aduan. Antarmuka ini berfungsi untuk menjawab aduan. Aduan dapat dijawab menggunakan *via* sms jika pengirim aduan tersebut menyantumkan nomor telepon. Jika tidak, sistem akan secara otomatis menggunakan *via* situs untuk membalas aduan tersebut. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.7 merupakan tampilan antarmuka jawaba aduan.



Gambar 4.7. Implementasi Antarmuka Jawab Aduan

4.3.7 Implementasi Antarmuka Tindak Lanjut Aduan

Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan *button* tindak lanjuti aduan di antarmuka detail aduan. Antarmuka ini berfungsi untuk memilih SKPD untuk menanggapi aduan tersebut, mengubah status aduan, mengubah prioritas aduan, dan menandai aduan sebagai spam jika yang *login* merupakan petugas tingkat satu. Untuk petugas tingkat dua, antarmuka ini berfungsi untuk mengubah status, mengubah prioritas, menandai aduan sebagai *spam*, dan mengembalikan aduan. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.8 merupakan tampilan antarmuka tindak lanjut aduan.



Gambar 4.8. Implementasi Antarmuka Tindak Lanjut Aduan

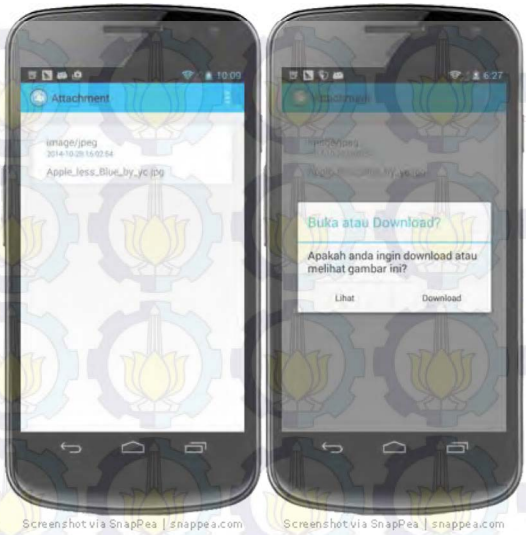
4.3.8 Implementasi Antarmuka Peta

Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan *button* lihat peta di antarmuka detail aduan. Antarmuka ini berfungsi untuk menampilkan lokasi peta dari aduan yang dilaporkan. Antarmuka ini juga bisa memperbesar atau memperkecil peta. Antarmuka ini diimplementasikan menggunakan kode XML, Java, dan Google Map API v2. Gambar 4.9 merupakan tampilan antarmuka peta.

Gambar 4.9. Implementasi Antarmuka Peta

4.3.9 Implementasi Antarmuka Lihat Lampiran

Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan *button* lihat lampiran di antarmuka detail aduan. Antarmuka ini berfungsi untuk menampilkan lampiran-lampiran yang disertakan. Lampiran-lampiran ini bisa diunduh ke dalam perangkat Android atau dapat dilihat jika lampiran tersebut merupakan gambar. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Pada gambar 4.10 merupakan tampilan antarmuka lihat lampiran.



Gambar 4.10. Implementasi Antarmuka Lihat Lampiran

4.3.10 Implementasi Antarmuka Lihat Gambar

Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan *button* lihat di antarmuka lihat lampiran. Antarmuka ini berfungsi untuk menampilkan lampiran yang berupa gambar. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.11 merupakan tampilan antarmuka lihat gambar.

Gambar 4.11. Implementasi Antarmuka Lihat Gambar

4.3.11 Implementasi Antarmuka *List* Validasi SMS

Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan *list* validasi sms di antarmuka sub menu. Antarmuka ini berfungsi untuk menampilkan sms-sms yang tidak dikirim sesuai format. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.12 merupakan tampilan antarmuka *list* validasi sms.

Gambar 4.12. Implementasi Antarmuka *List* Validasi SMS

4.3.12 Implementasi Antarmuka Validasi SMS

Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan salah satu sms yang tidak valid di *list* validasi sms. Antarmuka ini berfungsi untuk memasukkan sms tidak valid tersebut ke dalam sistem. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.13 merupakan tampilan antarmuka validasi sms.

Gambar 4.13. Implementasi Antarmuka Validasi SMS

4.3.13 Implementasi Antarmuka SMS

Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan sub menu sms di antarmuka sub menu. Antarmuka ini berfungsi untuk mengirim sms kepada petugas yang sudah terdaftar dalam sistem. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.14 merupakan tampilan antarmuka sms.

Gambar 4.14. Implementasi Antarmuka SMS

4.3.14 Implementasi Antarmuka Laporan

Antarmuka menu merupakan antarmuka yang akan muncul jika pengguna menekan sub menu laporan di antarmuka sub menu. Antarmuka ini berfungsi untuk mengunduh laporan berdasarkan tahun. Antarmuka ini diimplementasikan menggunakan kode XML dan Java. Gambar 4.15 merupakan tampilan antarmuka laporan.

Gambar 4.15. Implementasi Antarmuka Laporan

4.4 Implementasi *Query* Basis Data

Pada subbab ini akan dijelaskan mengenai implementasi *query* yang digunakan untuk mengakses *server*, *query* ini disimpan dalam suatu *web service* berbahasa PHP pada *server*. *Query* yang diimplementasikan dibuat menggunakan bahasa SQL.

4.4.1 Implementasi *Query* Menampilkan Jumlah Aduan

Query menampilkan jumlah aduan dibagi menjadi dua. Pertama adalah *query* jumlah aduan untuk petugas tingkat satu yang dimana *query* tersebut menampilkan data jumlah total aduan yang belum diklasifikasi, jumlah aduan yang sudah diklasifikasi, jumlah aduan yang dikembalikan, dan jumlah seluruh aduan. Kedua adalah *query* jumlah aduan untuk

petugas tingkat dua yang dimana *query* tersebut menampilkan jumlah aduan yang belum terjawab, jumlah aduan yang sudah terjawab, dan jumlah seluruh aduan.

Query untuk aduan yang belum diklasifikasi diimplementasikan dengan cara menghitung jumlah aduan yang memiliki atribut info yang nilainya nol dan tidak termasuk *spam*. Imlementasi ini ditunjukkan pada Kode Sumber 4.3.

```
1 | select count(a.id_aduan) as jumlah_aduan
2 | from prioritas p, aduan a LEFT JOIN
3 | departemen d ON a.departemen
4 | =d.id_departemen
5 | where a.info='0' and a.spam='0' and
6 | a.prioritas=p.id_prioritas
```

Kode Sumber 4.3 Implementasi *Query* Menampilkan Jumlah Aduan Belum Terklasifikasi

Query aduan yang sudah diklasifikasi diimplementasikan dengan cara menghitung jumlah aduan yang memiliki atribut info yang nilainya satu dan tidak merupakan *spam*. Implementasi ini ditunjukkan pada Kode Sumber 4.4.

```
1 | select count(a.id_aduan) as jumlah_aduan
2 | from prioritas p, aduan a LEFT JOIN
3 | departemen d ON a.departemen
4 | =d.id_departemen
5 | where a.info='1' and a.spam='0' and
6 | a.prioritas=p.id_prioritas
```

Kode Sumber 4.4 Implementasi *Query* Menampilkan Jumlah Aduan Terklasifikasi

Query untuk aduan yang sudah dikembalikan diimplementasikan dengan cara menghitung jumlah aduan yang memiliki atribut info yang nilainya dua dan tidak merupakan *spam.* Implementasi *query* ini ditunjukkan pada Kode Sumber 4.5.

```
1  select count(a.id_aduan)as jumlah_aduan
2  from prioritas p, aduan a LEFT JOIN
3  departemen d ON a.departemen
4  =d.id_departemen
5  where a.spam='0' and a.info='2' and
6  a.prioritas=p.id_prioritas
```

Kode Sumber 4.5 Implementasi *Query* Menampilkan Jumlah Aduan DiKembalikan

Query untuk seluruh aduan diimplementasikan dengan cara menghitung jumlah aduan yang ada dan tidak merupakan *spam*. Implementasi *query* ini ditunjukkan pada Kode Sumber 4.6.

```
1  select count(a.id_aduan) as jumlah_aduan
2  from prioritas p, aduan a LEFT JOIN
3  departemen d ON a.departemen =
4  d.id_departemen where a.spam='0' and
5  a.prioritas=p.id_prioritas
```

Kode Sumber 4.6 Implementasi *Query* Menampilkan Jumlah Seluruh Aduan

Query untuk aduan yang belum terjawab diimplementasikan dengan cara menghitung aduan yang memiliki status tidak sama dengan empat, tidak merupakan *spam,* terkait pada SKPD yang telah *login*. Implementasi *query* ini ditunjukkan pada Kode Sumber 4.7

```
1 | select count(*) as jumlahBelumTerjawab
2 | from prioritas p, aduan a,departemen d
3 | where a.spam='0' and a.status<>'4' and
4 | a.departemen='".$departemen."' and
5 | a.departemen = d.id_departemen and
6 | a.prioritas=p.id_prioritas
```

Kode Sumber 4.7 Implementasi *Query* Menampilkan Jumlah Aduan Belum Terjawab

Query untuk aduan yang sudah terjawab diimplementasikan dengan cara menghitung aduan yang memiliki status sama dengan empat, tidak merupakan *spam,* dan terkait pada SKPD yang telah *login*. Implementasi *query* ini ditunjukkan pada Kode Sumber 4.8.

```
1 | select count(*) as jumlahTerjawab from
2 | prioritas p, aduan a,departemen d
3 | where a.spam='0' and a.status='4' and
4 | a.departemen='".$departemen."' and
5 | a.departemen = d.id_departemen and
6 | a.prioritas=p.id_prioritas
```

Kode Sumber 4.8 Implementasi *Query* Menampilkan Jumlah Aduan Terjawab

Query jumlah seluruh aduan diimplementasikan dengan cara menghitung aduan yang telah ada pada sistem, tidak merupakan *spam,* dan terkait pada SKPD yang telah *login.* Implementasi *query* ini ditunjukkan pada Kode Sumber 4.9.

```
1 | select count(*) as jumlahSemuaAduan from
2 | prioritas p, aduan a,departemen d where
```

```
3 | a.spam='0' and
4 | a.departemen='".$departemen."' and
5 | a.departemen = d.id_departemen and
6 | a.prioritas=p.id_prioritas
```

Kode Sumber 4.9 Implementasi *Query* Menampilkan Jumlah Seluruh Aduan

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android lalu ditampilkan di antarmuka manajemen aduan pada Gambar 4.5.

4.4.2 Implementasi *Query* Menampilkan Aduan

Query menampilkan aduan dibagi menjadi dua. Pertama adalah *query* aduan untuk petugas tingkat satu yang dimana *query* tersebut menampilkan data aduan yang belum diklasifikasi, aduan yang sudah diklasifikasi, aduan yang dikembalikan, dan seluruh aduan. Kedua adalah *query* jumlah aduan untuk petugas tingkat dua yang dimana *query* tersebut menampilkan aduan yang belum terjawab, aduan yang sudah terjawab, dan seluruh aduan. Untuk *query* petugas tingkat dua ini merupakan *query* berdasarkan SKPD petugas tersebut. Masing-masing *query* ini diurutkan berdasarkan waktu terakhir.

Query untuk aduan yang belum diklasifikasi diimplementasikan dengan cara memilih aduan yang memiliki atribut info yang nilainya nol dan tidak termasuk *spam*. Imlementasi ini ditunjukkan pada Kode Sumber 4.10

```
select
a.status,a.id_aduan,d.nama_departemen,
a.waktu,
a.topik,p.nama_prioritas,a.isi
from prioritas p, aduan a LEFT JOIN
departemen d ON a.departemen =
d.id_departemen
where a.info='0' and a.spam='0' and
a.prioritas=p.id_prioritas
order by a.waktu desc
```

Kode Sumber 4.10 Implementasi *Query* Menampilkan Aduan Belum Terklasifikasi

Query aduan yang sudah diklasifikasi diimplementasikan dengan cara memilih aduan yang memiliki atribut info yang nilainya satu dan tidak merupakan *spam*. Implementasi ini ditunjukkan pada Kode Sumber 4.11.

```
select
a.status,a.id_aduan,d.nama_departemen,a.waktu
,a.topik,p.nama_prioritas,a.isi
from prioritas p, aduan a LEFT JOIN
departemen d ON a.departemen =
d.id_departemen
where a.info='1' and a.spam='0' and
a.prioritas=p.id_prioritas
order by a.waktu desc
```

Kode Sumber 4.11 Implementasi *Query* Menampilkan Aduan Terklasifikasi

Query untuk aduan yang sudah dikembalikan diimplementasikan dengan cara memilih aduan yang memiliki atribut info yang nilainya dua dan tidak merupakan *spam.* Implementasi *query* ini ditunjukkan pada Kode Sumber 4.12.

```
select
a.status,a.id_aduan,d.nama_departemen,a.waktu
,a.topik,p.nama_prioritas,a.isi
from prioritas p, aduan a LEFT JOIN
departemen d ON a.departemen =
d.id_departemen
where a.spam='0' and a.info='2' and
a.prioritas=p.id_prioritas
order by a.waktu desc
```

Kode Sumber 4.12 Implementasi *Query* Menampilkan Aduan Dikembalikan

Query untuk seluruh aduan diimplementasikan dengan cara memilih aduan yang ada dan tidak merupakan *spam*. Implementasi *query* ini ditunjukkan pada Kode Sumber 4.13.

```
select a.status,a.id_aduan,
d.nama_departemen,a.waktu,a.topik,p.nama_prio
ritas,a.isi from prioritas p,
aduan a LEFT JOIN departemen d ON
 a.departemen = d.id_departemen where
a.spam='0' and a.prioritas=p.id_prioritas
order by a.waktu desc
```

Kode Sumber 4.13 Implementasi *Query* Menampilkan Seluruh Aduan

Query untuk aduan yang belum terjawab diimplementasikan dengan cara memilih aduan yang memiliki status tidak sama dengan empat, tidak merupakan *spam,* terkait pada SKPD yang telah *login*. Implementasi *query* ini ditunjukkan pada Kode Sumber 4.14.

```
select
a.id_aduan,d.nama_departemen,a.waktu,a.topik,
p.nama_prioritas,a.isi
from prioritas p, aduan a LEFT JOIN
departemen d ON a.departemen=d.id_departemen
where a.info<>'2' and a.spam='0' and
a.status<>'4' and a.petugas='".$id_petugas."'
and a.prioritas=p.id_prioritas
order by a.waktu desc;
```

Kode Sumber 4.14 Implementasi *Query* Menampilkan Aduan Belum Terjawab

Query untuk aduan yang sudah terjawab diimplementasikan dengan cara memilih aduan yang memiliki status sama dengan empat, tidak merupakan *spam,* dan terkait pada SKPD yang telah *login*. Implementasi *query* ini ditunjukkan pada Kode Sumber 4.15.

```
select
a.id_aduan,d.nama_departemen,a.waktu,a.topik,
p.nama_prioritas ,a.isi
from prioritas p, aduan a LEFT JOIN
departemen d ON a.departemen=d.id_departemen
where a.info<>'2' and a.spam='0' and
a.status='4' and a.petugas='".$id_petugas."'
and a.prioritas=p.id_prioritas order by
a.waktu desc;
```

Kode Sumber 4.15 Implementasi *Query* Menampilkan Aduan Terjawab

Query jumlah seluruh aduan diimplementasikan dengan cara memilih aduan yang telah ada pada sistem, tidak merupakan *spam,* dan terkait pada SKPD yang telah *login.* Implementasi *query* ini ditunjukkan pada Kode Sumber 4.16.

```
select
a.id_aduan,d.nama_departemen,a.waktu,a.topik,
p.nama_prioritas,a.isi
from prioritas p, aduan a LEFT JOIN
departemen d ON a.departemen=d.id_departemen
where a.info<>'2' and a.spam='0' and
a.petugas='".$id_petugas."' and
a.prioritas=p.id_prioritas order by a.waktu
desc;
```

Kode Sumber 4.16 Implementasi *Query* Menampilkan Seluruh Aduan

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android lalu ditampilkan di antarmuka manajemen aduan menggunakan *listview* pada Gambar 4.5.

4.4.3 Implementasi *Query* Menampilkan Detail Aduan

Query menampilkan detail aduan adalah *query* yang berfungsi untuk menampilkan data-data secara keseluruhan suatu aduan. Data-data yang ditampilkan yaitu id aduan, topik aduan, nama pengirim aduan, nama departemen aduan, waktu, dikirim melalui, status aduan, prioritas aduan, bujur, lintang, dan balasan dari petugas. Implementasi *query* tersebut ditunjukkan pada Kode Sumber 4.17.

```
select
a.latitude,a.longitude,a.id_aduan,a.topik,a.n
ama,d.nama_departemen,a.waktu,a.via_sms,s.nam
a_status,p.nama_prioritas
from aduan a LEFT JOIN departemen d ON
a.departemen = d.id_departemen,prioritas
p,status s where a.id_aduan='".$id."' and

```

```
9 a.status = s.id_status and a.prioritas =
  p.id_prioritas
```

Kode Sumber 4.17 Implementasi *Query* Menampilkan Detail Aduan

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android lalu ditampilkan di antarmuka detail aduan pada Gambar 4.6.

4.4.4 Implementasi *Query* Menampilkan Isi dan Balasan Aduan

Query menampilkan isi dan balasan aduan adalah *query* yang berfungsi untuk menampilkan data-data isi aduan, balasan aduan, nama pengirim aduan, nama petugas, waktu pengiriman aduan, dan waktu balas aduan. Implementasi *query* tersebut ditunjukkan pada Kode Sumber 4.18.

```
1 select
2 da.petugas_detail,p.nama_petugas,d.nama_depar
3 temen,a.nama,da.waktu_detail,da.isi_detail
4 from detail_aduan da LEFT JOIN petugas p ON
5 da.petugas_detail=p.id_petugas LEFT JOIN
6 departemen d ON p.departemen=d.id_departemen,
7 aduan a where da.aduan='".$id."' and da.aduan
8 = a.id_aduan
```

Kode Sumber 4.18 Implementasi *Query* Menampilkan Detail Aduan

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android lalu ditampilkan di antarmuka jawab aduan pada Gambar 4.7.

4.4.5 Implementasi *Query* Menampilkan Data petugas, prioritas, dan status Aduan

Query menampilkan data petugas, prioritas, dan status aduan berfungsi untuk memilih petugas yang akan menanggapi aduan, mengubah status aduan, dan prioritas aduan. *Query* menampilkan data petugas diimplementasikan dengan cara mengambil nama petugas dan *id* petugas yang dimana petugas memiliki *role* tiga. Implementasi ini ditunjukkan pada Kode Sumber 4.19.

```
1 | select id_petugas,nama_petugas from petugas
2 | where role='3' and nama_petugas IS NOT NULL
```

Kode Sumber 4.19 Implementasi *Query* Menampilkan prioritas

Query menampilkan data prioritas diimplementasikan dengan cara mengambil nama prioritas dan *id* prioritas yang dimana petugas memiliki *role* tiga. Implementasi ini ditunjukkan pada Kode Sumber 4.20

```
1 | select id_prioritas,nama_prioritas from
2 | prioritas
```

Kode Sumber 4.20 Implementasi *Query* Menampilkan prioritas

Query menampilkan data status diimplementasikan dengan cara mengambil nama status dan *id* status yang dimana petugas memiliki *role* tiga. Implementasi ini ditunjukkan pada Kode Sumber 4.21

```
1 | select id_status,nama_status from status
```

Kode Sumber 4.21 Implementasi *Query* Menampilkan prioritas

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android lalu ditampilkan di antarmuka tindak lanjut aduan pada Gambar 4.8.

4.4.6 Implementasi *Query* Mengubah Aduan

Query mengubah aduan berfungsi untuk mengubah aduan jika merupakan spam, aduan dikembalikan, mengubah status aduan, prioritas aduan, dan petugas yang menanggapi aduan. *Query* mengubah aduan menjadi *spam* diimplementasikan dengan cara mengeset atribut spam menjadi satu. Implementasi ini dapat ditunjukkan pada Kode Sumber 4.22.

```
1 | update aduan set spam='1' where
2 | id_aduan='".$id."'
```

Kode Sumber 4.22 Implementasi *Query* Mengubah Aduan Menjadi Spam

Query mengembalikan aduan diimplementasikan dengan cara mengeset atribut info menjadi dua. Implementasi ini dapat ditunjukkan pada Kode Sumber 4.23.

```
1 | update aduan set info='2' where
2 | id_aduan='".$id."'
```

Kode Sumber 4.23 Implementasi *Query* Mengembalikan Aduan

Query mengubah prioritas, status, dan petugas diimplementasikan dengan cara mengeset prioritas, status, dan

petugas dengan masukkan dari pengguna. Implementasi dini ditunjukkan pada Kode Sumber 4.24.

```
"update aduan set info='1',
prioritas='".$prioritas."',
petugas='".$petugas."',
status='".$status."' where
id_aduan='".$id."'
```

Kode Sumber 4.24 Implementasi *Query* Mengubah Prioritas, Status, dan Petugas Aduan

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang diambil oleh aplikasi Android lalu ditampilkan di antarmuka tindak lanjut aduan pada Gambar 4.8.

4.4.7 Implementasi *Query* Menampilkan Lampiran

Query menampilkan lampiran berfungsi untuk menampilkan lampiran yang disertakan aduan. Data yang diambil yaitu waktu, tipe *file*, dan nama *file*. Implementasi *query* tersebut ditunjukkan pada Kode Sumber 4.1

```
select
da.waktu_detail,u.file_type,u.orig_name as
nama_file,u.path_upload from aduan
a,detail_aduan da,upload u
where a.id_aduan='".$id."' and
a.id_aduan=da.aduan and
da.id_detail_aduan=u.detail_aduan
```

Kode Sumber 4.25 Implementasi *Query* Menampilkan Lampiran

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android

lalu ditampilkan di antarmuka lihat lampiran dalam bentuk *listview* aduan pada Gambar 4.10.

4.4.8 Implementasi *Query* Menampilkan Data SMS Tidak Valid

Query menampilkan data sms tidak valid berfungsi untuk menampilkan sms-sms tidak sesuai dengan format yang sudah ditentukan. Data-data yang diambil adalah id, nomor pengirim, waktu pengiriman, dan isi aduan. Implementasi *query* tersebut ditunjukkan pada Kode Sumber 4.26.

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android lalu ditampilkan di antarmuka *list* validasi sms dengan bentuk *listview* pada Gambar 4.12.

```
1 | select id,nomor_pengirim,waktu,isi from
2 | sms_tidak_valid
```

Kode Sumber 4.26 Implementasi *Query* Menampilkan Data SMS Tidak Valid

4.4.9 Implementasi *Query* Menampilkan Nomor Telepon Petugas

Query menampilkan nomor telepon petugas berfungsi untuk menampilkan nomor telepon para petugas untuk mengirim sms dari sistem. Data-data yang diambil adalah nama petugas dan nomor telepon petugas. Implementasi *query* tersebut ditunjukkan pada Kode Sumber 4.27.

```
1 | select nama_petugas,no_hp_petugas from
2 | petugas where no_hp_petugas<>'null';
```

Kode Sumber 4.27 Implementasi *Query* Menampilkan Nomor Telepon Petugas

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android lalu ditampilkan di antarmuka sms pada Gambar 4.14.

4.4.10 Implementasi *Query* Menampilkan Tahun

Query menampilkan tahun berfungsi untuk menampilkan tahun yang ada dalam sistem. Tahun ini digunakan untuk mengambil data-data laporan pada tahun tersebut. Data yang diambil adalah tahun saja. Implementasi *query* tersebut ditunjukkan pada Kode Sumber 4.28.

```
1 | select distinct YEAR(waktu) as tahun from
2 | aduan;
```

Kode Sumber 4.28 Implementasi *Query* Menampilkan Tahun

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android lalu ditampilkan di antarmuka laporan pada Gambar 4.15.

4.4.11 Implementasi *Query* Mengambil Data Laporan

Query mengambil data laporan berfungsi untuk mengambil data-data yang diperlukan yang akan ditampilkan di *file* yang berekstensi "*.csv*" berdasarkan tahun yang dipilih. Data-data yang diperlukan adalah jumlah aduan masuk, jumlah aduan yang belum ditindak lanjut, aduan yang sudah ditindak

lanjuti, aduan yang sudah selesai, aduan yang berstatus *spam*, dan setiap aduan yang berada pada tahun yang dipilih.

Query aduan masuk dimplementasikan dengan cara menghitung jumlah aduan yang waktunya ada pada tahun yang dipilih. Implementasi ini ditunjukkan pada Kode Sumber 4.29.

```
1 | select count(*) as aduan_masuk from aduan
2 | where aduan.waktu >= '".$awal."' and
3 | aduan.waktu < '".$akhir."'
```

Kode Sumber 4.29 Implementasi *Query* Aduan Masuk

Query aduan belum ditindak lanjut dimplementasikan dengan cara memilih aduan yang waktunya ada pada tahun yang dipilih dan memiliki info tidak sama dengan satu. Implementasi ini ditunjukkan pada Kode Sumber 4.30.

```
1 | select count(*) as belum_ditindak_lanjut
2 | from aduan where info <> 1 and aduan.waktu
3 | >= '".$awal."' and aduan.waktu <
4 | '".$akhir."'
```

Kode Sumber 4.30 Implementasi *Query* Aduan Belum Ditindak Lanjut

Query aduan sudah ditindak lanjut dimplementasikan dengan cara memilih aduan yang waktunya ada pada tahun yang dipilih dan memiliki info sama dengan satu. Implementasi ini ditunjukkan pada Kode Sumber 4.31.

```
1 | select count(*) as sudah_ditindak_lanjut
2 | from aduan where info = 1 and aduan.waktu
3 | >= '".$awal."' and aduan.waktu <
4 | '".$akhir."'
```

Kode Sumber 4.31 Implementasi *Query* Aduan Sudah Ditindak Lanjut

Query aduan sudah selesai dimplementasikan dengan cara memilih aduan yang waktunya ada pada tahun yang dipilih dan memiliki status sama dengan empat. Implementasi ini ditunjukkan pada Kode Sumber 4.32.

```
1 | select count(*) as sudah_selesai from aduan
2 | where aduan.status = 4 and aduan.waktu >=
3 | '".$awal."' and aduan.waktu < '".$akhir."'
```

Kode Sumber 4.32 Implementasi *Query* Aduan Sudah Selesai

Query aduan sudah selesai dimplementasikan dengan cara memilih aduan yang waktunya ada pada tahun yang dipilih dan memiliki *spam* sama dengan satu. Implementasi ini ditunjukkan pada Kode Sumber 4.33.

```
1 | select count(*) as `sampah` from aduan
2 | where spam = 1 and aduan.waktu >=
3 | '".$awal."' and aduan.waktu < '".$akhir."'
```

Kode Sumber 4.33 Implementasi *Query* Aduan Spam

Query aduan berdasarkan tahun dimplementasikan dengan cara memilih aduan yang waktunya ada pada tahun yang dipilih. Implementasi ini ditunjukkan pada Kode Sumber 4.34.

```
 1 | select a.id_aduan, a.no_identitas
 2 | `nik_pengadu`, a.nama `nama_pengadu`,
 3 | a.waktu,
 4 | p.nama_petugas, d.nama_departemen, spam,
 5 | a.topik `topik_aduan`, a.isi `isi_aduan`
 6 | from aduan a
 7 | LEFT JOIN petugas p ON
 8 | a.petugas=p.id_petugas LEFT JOIN departemen
 9 | d ON a.departemen=d.id_departemen
10 | where a.waktu>='".$awal."' and
11 | a.waktu<'".$akhir."';
```

Kode Sumber 4.34 Implementasi *Query* Aduan Berdasarkan Tahun

Query ini dieksekusi oleh *web service* lalu ditampilkan menggunakan *json* yang akan diambil oleh aplikasi Android lalu ditampilkan di antarmuka sms pada Gambar 4.15

4.5 Implementasi Aplikasi Operator Suara Warga

Pada subbab ini akan dijelaskan mengenai implementasi aplikasi pada kasus penggunaan ke dalam sebuah baris kode. Implementasi ini dilakukan dengan menggunakan bahasa native pembuatan aplikasi Android yakni bahasa Java.

4.5.1 Implementasi Proses Pengambilan dan Menerima Data *Web Service*

Web service digunakan untuk melakukan *query* ke dalam basis data dimana hasil *query* tersebut akan digunakan untuk menampilkan data-data dalam sistem. Dalam pertukaran data dari *web service* ke perangkat Android digunakan *method post* dan *json*. *Method post* digunakan untuk mengirim data dari Android ke *web service* sedangkan *json* digunakan untuk menerima data dari *web service* ke perangkat Android. Semua proses ini menggunakan kelas *AsyncTask* di dalam *method*

doInBackground() yang dimana kelas ini berjalan sendiri diluar *thread* aplikasi utama.

Implementasi proses pengiriman data menggunakan kelas *NameValuePair* untuk menyimpan data yang akan dikirm dan *jParser* digunakan untuk mengirim dan menerima data dari *web service*. Implementasi pengiriman data dari perangkat Android ke *web service* ditunjukkan pada Kode Sumber 4.35 sedangkan untuk penerimaan data dari *web service* ke perangkat Android ditunjukkan pada Kode Sumber 4.36.

```
1  PROTECTED String doInBackground(String
2  ...args){
3   Inisialisasi list BasicNameValuePair
4   Masukkan data berupa key dan value ke
5  BasicNameValuePair
6
7   Inisialisasi jParser
8   jParser.makeHttpRequest(destinationURL,
9   ”post”, list BasicNameValuePair);
10  return null
11 }
```

Kode Sumber 4.35 Implementasi Mengirim Data ke Web Service

```
1  getDataJson(){
2  Inisialisasi JSONArray berdasarkan TAG
3  Inisialisasi list untuk menampung data
4  For i=0 to panjang JSONArray
5   JSONObject ← JSONArray[i]
6   ambil data dari JSONObject berdasarkan TAG
7  lalu dimasukkan kedalam list.
8  End for
9
10 return list
11 }
```

Kode Sumber 4.36 Implementasi Pengambilan Data dari Web Service

4.5.2 Implementasi *Session* Aplikasi

Agar aplikasi dapat berjalan terus sesuai dengan kebutuhan ketika pengguna masih login, dibutuhkan *session* dalam aplikasi untuk menyimpan data pengguna. Untuk itu dibuat objek session dengan atribut disesuaikan dengan kebutuhan. Digunakan *SharedPreferences* untuk menyimpan *session* tersebut. Implementasi Konstruktor dari *session* dapat dilihat dalam Kode Sumber 4.37.

```
1  String ROLE ← "role"
2  String DEPARTEMEN ← "departemen"
3  String ID_PETUGAS ← "id_petugas"
4  String SMS_DINAS ← "sms_dinas"
5  String PREFS_NAME ← "session_operator"
6
7  Inisialisasi SharedPreferences
8
9  Public SessionManager(Context context){
10  SharedPreferences ←
11 context.getSharedPreferences(PREFS_NAME,
12 Context.MODE_PRIVATE)
   }
```

Kode Sumber 4.37 Atribut dan Konstruktor Session Aplikasi

Saat aplikasi berjalan dan memanggil layanan *web service,* dibutuhkan beberapa data penting yang disimpan dalam *session.* Untuk dapat mengakses data dalam *session* tersebut dibuat suatu fungsi di dalam kelas *session* tersebut.

4.5.3 Implementasi Proses *Login*

Proses *login* ini berfungsi untuk pembuktian keaslian pengguna melalui *username* dan *password* yang dimasukkan sesuai dengan data yang ada dalam sistem suara warga untuk masuk ke dalam sistem. Implementasi dari proses ini ditunjukkan dalam Kode Sumber 4.38.

```
1  Inisialisasi list BasicNameValuePair
2  Masukkan key dan value username dan password
3  ke basicNameValuePair
4
5  Inisialisasi jParser
6  jParser.makeHttpRequest(destination,"post",
7  list BasicNameValuePair);
8     if json.getInt(TAG_SUCCESS) == 1
9       validUser ← 1
10      Inisialisasi JSONArray berdasarkan TAG
11      JSONObject ← JSONArray[0]
12      role ← JSONObject.getString(TAG_ROLE)
13      departemen ←
14 JSONObject.getString(TAG_DEPARTEMEN)
15      id_petugas ←
16 JSONObject.getString(TAG_ID_PETUGAS)
17    end if
18    else
19      validUser ← 0
20    end if
21   if validUser==1
22    if role == 1
23    simpanSession(role,departemen,
24 id_petugas,true)
25    end if
26    else
27     simpanSession(role,departemen,
28 id_petugas,false)
29    end if
30    Inisialisasi intent activity menu
```

```
31      startActivity(activityMenu)
32    end if
```

Kode Sumber 4.38 Implementasi Proses Login

Jika *username dan password* sesuai, maka pengguna dapat masuk ke dalam aplikasi dan *session* disimpan sampai pengguna melakukan *logout* aplikasi meskipun aplikasi ditutup. Jika *username* dan *password* salah, maka akan muncul informasi peringatan. Untuk membuat *session* aplikasi, ditunjukkan dalam Kode Sumber 4.39.

```
1  Public void simpanSession(String role,
2  String departemen, String id_petugas,
3  boolean sms dinas){
4   SharedPreferences.Editor editor ←
5   session.edit()
6   editor.putString(this.ROLE,role)
7
8  editor.putString(this.DEPARTEMEN,departemen
9  )
10
11 editor.putString(this.ID_PETUGAS,id_petugas
12 )
13
14 editor.putBoolean(this.SMS_DINAS,sms_dinas)
15  editor.commit()
16 }
```

Kode Sumber 4.39 Implementasi Penyimpanan Session Aplikasi

4.5.4 Implementasi Menampilkan Aduan

Menampilkan aduan diimplementasikan dengan cara mengambil seluruh aduan berdasarkan jenis-jenis aduan yang

ingin di tampilkan. Sebagai contoh jika petugas tingkat satu yang telah login ke dalam sistem, maka jenis-jenis aduan yang di ambil adalah aduan yang belum diklasifikasi, aduan yang terklasifikasi, aduan yang dikembalikan, dan seluruh aduan. Berikut Implementasi dari mengambil data dari *web service* yang ditunjukkan pada Kode Sumber 4.40 dan menampilkan data dari *web service* yang ditunjukkan pada Kode Sumber 4.41.

```
1  getDataJson(){
2  Inisialisasi JSONArray TAG_ADUAN
3  Inisialisasi list aduan
4  For i=0 to panjang JSONArray
5   JSONObject ← JSONArray[i]
6   ambil data dari JSONObject TAG ADUAN
7   masukkan ke dalam list
8  end for
9  return list
```

Kode Sumber 4.40 Implementasi Pengambilan Data Aduan

```
1  listSearch ← listAduan
2  inisialisasi
3  listAdapter(activity,listSearch)
4  listview.setAdapter(listAdapter)
   set onItemclickListener di listview
```

Kode Sumber 4.41 Implementasi Menampilkan Data Aduan

Secara garis besar, implementasi ini dilakukan dengan cara mengambil data aduan di *web service* lalu dimasukkan ke dalam sebuah *list* yang dimana *list* tersebut akan ditampilkan dengan menggunakan kelas *ListAduanAdapter*.

4.5.5 Implementasi Mencari Aduan

Implementasi mencari aduan diimplementasikan dengan cara mencari aduan dari *list* aduan menggunakan kata kunci yang dimasukkan. Kata kunci tersebut dijadikan menjadi huruf kecil begitu juga dengan kata-kata yang ada di aduan.Untuk pencariannya jika ada kata dari aduan yang mengandung unsur kata kunci, maka aduan tersebut dimasukkan ke dalam *list* pencarian. Untuk implementasi ini ditunjukkan Kode Sumber 4.53.

```
getSearchList(list aduanList, string
keyword){
  keylowercase ← keyword.toLowerCase()
  int statusTerjawab ← 0
  if keylowercase equal "terjawab"
     statusTerjawab ← 1
  end if
  else if keylowercase equal "tidak
terjawab"
     statusTerjawab ← 2
  end if
  inisialisasi list pencarian
  for i=0 sampai panjang list aduanList{
    if
aduanList[i].getDepartemen().toLowerCase().c
ontains (keylowercase)
      if(!pencarian.contains(aduanList[i])
        pencarian.add(aduanList[i])
      end if
    end if
    if
aduanList[i].getId().toLowerCase().contains
(keylowercase)
      if(!pencarian.contains(aduanList[i])
        pencarian.add(aduanList[i])
      end if
    end if

```

```
    if
aduanList[i].getIsi().toLowerCase().contains
(keylowercase)
      if(!pencarian.contains(aduanList[i])
        pencarian.add(aduanList[i])
      end if
    end if
    if
aduanList[i].getPrioritas().toLowerCase().co
ntains (keylowercase)
      if(!pencarian.contains(aduanList[i])
        pencarian.add(aduanList[i])
      end if
    end if
    if
aduanList[i].getTopik().toLowerCase().contai
ns (keylowercase)
      if(!pencarian.contains(aduanList[i])
        pencarian.add(aduanList[i])
      end if
    end if
    if
aduanList[i].getWaktu().toLowerCase().substr
ing(0,9).contains(keylowercase)
      if(!pencarian.contains(aduanList[i])
        pencarian.add(aduanList[i])
      end if
    end if
    if aduanList[i].getStatus() ==4 &&
statusTerjawab == 1
      if(!pencarian.contains(aduanList[i])
        pencarian.add(aduanList[i])
      end if
    end if
    if aduanList[i].getStatus() <=3 &&
statusTerjawab == 2
      if(!pencarian.contains(aduanList[i])
        pencarian.add(aduanList[i])
      end if
    end if
  }

```

```
68 | Return pencarian
   | }
```

Kode Sumber 4.42 Implementasi Pencarian Aduan

4.5.6 Implementasi Menjawab Aduan

Implementasi menjawab aduan diimplementasikan dengan cara mengirim data yang telah dimasukkan oleh pengguna untuk membalas aduan tersebut ke *web service. Web service* ini yang akan digunakan untuk memasukkan data tersebut ke dalam basis data. Implementasi ini ditunjukkan pada Kode Sumber 4.43.

```
1 | Inisialisasi list BasicNameValuePair
2 | Masukkan key dan value id,isi_detail
3 | petugas_detail,via,no_hp ke
4 | basicNameValuePair
5 |
6 | Inisialisasi jParser
7 | jParser.makeHttpRequest(destination,”post”,
8 | list BasicNameValuePair)
```

Kode Sumber 4.43 Implementasi Menjawab Aduan

4.5.7 Implementasi Mengubah Status Aduan

Implementasi mengubah status aduan diimplementasikan dengan cara mengambil data status dari *web service*. Data tersebut lalu dimasukkan ke dalam suatu *list* lalu jika ikon status di antarmuka tindak lanjut aduan ditekan maka *list* status tersebut akan muncul dan dapat dipilih. Jika sudah dipilih dan ditekan *button update* maka status untuk aduan tersebut terubah. Untuk implementasi pengambilan data status dapat ditunjukkan pada Kode Sumber 4.44. Untuk implementasi menampilkan data status dapat ditunjukkan pada

Kode Sumber 4.45. Implementasi mengubah data status untuk suatu aduan dapat ditunjukkan pada Kode Sumber 4.46.

```
getDataStatus(){
inisialisasi list status
try{
   if json.getInt(TAG_SUCCESS) == 1
     Inisialisasi JSONArray TAG STATUS
     For i=0 sampai panjang JSONArray{
      JSONObject ← JSONArray[i]
      status ←
JSONObject.getString(TAG_STATUS)
      nama_status ←
JSONObject.getString(TAG_NAMA_STATUS)
     status.add(new
status(status,nama_status)
     }
   return status
   end if
}
```

Kode Sumber 4.44 Implementasi Pengambilan Data Status

```
onClick(View arg0) {
 inisialisasi AlertDialog.Builder builder
 builder.setTitle("Pilih Status")
 builder.setItems(statusArray,
DialogInterface.OnClickListener(){
  public void onClick(DialogInterface arg0,
int arg1){
  textViewStatus.setText(statusArray[arg1])
  }
})builder.show()}
```

Kode Sumber 4.45 Implementasi Menampilkan Data Status

```
1  Inisialisasi list BasicNameValuePair
2  Masukkan key dan value id_status dan id
3  aduan ke basicNameValuePair
4
5  Inisialisasi jParser
6  jParser.makeHttpRequest(destination,"post",
7  list BasicNameValuePair)
```

Kode Sumber 4.46 Implementasi Mengubah Status Aduan

4.5.8 Implementasi Mengubah Prioritas Aduan

Implementasi mengubah prioritas aduan diimplementasikan dengan cara mengambil data prioritas dari *web service*. Data lalu dimasukkan ke suatu *list* lalu jika *icon* prioritas di antarmuka tindak lanjut aduan ditekan maka *list* prioritas tersebut akan muncul dan dapat dipilih. Jika sudah dipilih dan ditekan *button* update maka prioritas untuk aduan tersebut terubah. Untuk implementasi pengambilan data prioritas dapat ditunjukkan pada Kode Sumber 4.47. Untuk implementasi menampilkan data prioritas dapat ditunjukkan pada Kode Sumber 4.48. Implementasi mengubah data prioritas untuk suatu aduan dapat ditunjukkan pada Kode Sumber 4.49.

```
 1  getDataPrioritas(){
 2  inisialisasi list prioritas
 3  try{
 4     if json.getInt(TAG_SUCCESS) == 1
 5       Inisialisasi JSONArray TAG PRIORITAS
 6       For i=0 sampai panjang JSONArray{
 7        JSONObject ← JSONArray[i]
 8        prioritas ←
 9  JSONObject.getString(TAG_ PRIORITAS)
10
```

```
      nama_ prioritas ←
JSONObject.getString(TAG_NAMA_PRIORITAS)
     prioritas.add(new prioritas
(prioritas,nama_ prioritas)
     }
   return prioritas
   end if
}
```

Kode Sumber 4.47 Implementasi Pengambilan Data Prioritas

```
onClick(View arg0) {
 inisialisasi AlertDialog.Builder builder
 builder.setTitle("Pilih Prioritas")
 builder.setItems(prioritasArray,
DialogInterface.OnClickListener(){
  public void onClick(DialogInterface arg0,
int arg1){
textViewPriortas.setText(prioritasArray[arg1
])
  }
})builder.show()
}
```

Kode Sumber 4.48 Implementasi Menampilkan Data Prioritas

```
1 | Inisialisasi list BasicNameValuePair
2 | Masukkan key dan value id_prioritas dan id
3 | aduan ke basicNameValuePair
4 | Inisialisasi jParser
5 | jParser.makeHttpRequest(destination,"post",
6 | list BasicNameValuePair)
```

Kode Sumber 4.49 Implementasi Mengubah Prioritas Aduan

4.5.9 Implementasi Memilih Petugas Aduan

Implementasi Memilih petugas aduan diimplementasikan dengan cara mengambil data petugas dari *web service*. Data tersebut lalu dimasukkan ke dalam suatu *list* lalu jika *icon* petugas di antarmuka tindak lanjut aduan ditekan maka *list* petugas tersebut akan muncul dan dapat dipilih. Jika sudah dipilih dan ditekan *button* update maka petugas untuk aduan tersebut terpilih. Untuk implementasi pengambilan data petugas dapat ditunjukkan pada Kode Sumber 4.50. Untuk implementasi menampilkan data petugas dapat ditunjukkan pada Kode Sumber 4.51. Implementasi mengubah data petugas untuk suatu aduan dapat ditunjukkan pada Kode Sumber 4.52.

```
 1 | getDataPetugas(){
 2 | inisialisasi list petugas
 3 | try{
 4 |    if json.getInt(TAG_SUCCESS) == 1
 5 |      Inisialisasi JSONArray TAG PETUGAS
 6 |      For i=0 sampai panjang JSONArray{
 7 |       JSONObject ← JSONArray[i]
 8 |       petugas ← JSONObject.getString(TAG_
 9 | PETUGAS)
10 |       nama_ petugas ←
11 | JSONObject.getString(TAG_NAMA_PETUGAS)
12 |      prioritas.add(new petugas (petugas,
13 | nama_petugas)
```

```
14        }
15      return petugas
16      end if
17  }
```

Kode Sumber 4.50 Implementasi Pengambilan Data Petugas

```
1   onClick(View arg0) {
2    inisialisasi AlertDialog.Builder builder
3    builder.setTitle("Pilih Petugas")
4    builder.setItems(petugasArray,
5   DialogInterface.OnClickListener(){
6     public void onClick(DialogInterface arg0,
7   int arg1){
8   textViewPetugas.setText(petugasArray[arg1])
9     }
10  })builder.show()
    }
```

Kode Sumber 4.51 Implementasi Menampilkan Data Petugas

```
1  Inisialisasi list BasicNameValuePair
2  Masukkan key dan value id_prioritas dan id
3  aduan ke basicNameValuePair
4
5  Inisialisasi jParser
6  jParser.makeHttpRequest(destination,"post",
7  list BasicNameValuePair);
```

Kode Sumber 4.52 Implementasi Mengubah Data Petugas yang Menanggapi Aduan

4.5.10 Implementasi Mengembalikan Aduan

Implementasi mengembalikan aduan diimplementasikan dengan cara mengirim status aduan ke *web service*. Untuk implementasi ini ditunjukkan Kode Sumber 4.53

```
1  Inisialisasi list BasicNameValuePair
2  spam ← 2
3  Masukkan key dan value spam ke
4  basicNameValuePair
5
6  Inisialisasi jParser
7  jParser.makeHttpRequest(destination,"post",
8  list BasicNameValuePair);
```

Kode Sumber 4.53 Implementasi Mengembalikan Aduan

4.5.11 Implementasi Mengubah Status Spam Aduan

Implementasi mengubah status *spam* aduan diimplementasikan dengan cara mengirim status aduan spam ke *web service*. Untuk implementasi ini ditunjukkan Kode Sumber 4.54.

```
1  Inisialisasi list BasicNameValuePair
2  spam ← 1
3  Masukkan key dan value spam ke
4  basicNameValuePair
5
6  Inisialisasi jParser
7  jParser.makeHttpRequest(destination,"post",
8  list BasicNameValuePair);
```

Kode Sumber 4.54 Implementasi Mengubah Status Spam Aduan

4.5.12 Implementasi Menampilkan Lampiran Aduan

Menampilkan lampiran aduan diimplementasikan dengan cara mengambil data seluruh lampiran aduan dari suatu aduan yang sudah dipilih dari *web service*. Data tersebut lalu ditampilkan menggunakan *listview* dengan menggunakan kelas

attachmentAdapter. Berikut Implementasi dari mengambil data dari *web service* yang ditunjukkan pada Kode Sumber 4.55 dan menampilkan data dari *web service* yang ditunjukkan pada Kode Sumber 4.56.

```
getDataLampiran(){
inisialisasi list lampiran
try{
   if json.getInt(TAG_SUCCESS) == 1
     Inisialisasi JSONArray TAG STATUS
     For i=0 sampai panjang JSONArray{
      JSONObject ← JSONArray[i]
      waktu_detail ←
JSONObject.getString(TAG_WAKTU_DETAIL)
      file_type ←
JSONObject.getString(TAG_FILE_TYPE)
      path_upload ←
JSONObject.getString(TAG_PATH_UPLOAD)
      nama_file ←
JSONObject.getString(TAG_NAMA_FILE)
     lampiran.add(new
lampiran(waktu_detail,file_type,path_upload
, nama_file)
     }
   return lampiran
   end if
}
```

Kode Sumber 4.55 Implementasi Pengambilan Lampiran Aduan

```
listLampiran
inisialisasi
listAdapter(activity,listLampiran)

listview.setAdapter(listAdapter)
set onItemclickListener di listview
```

Kode Sumber 4.56 Implementasi Menampilkan Lampiran Aduan

4.5.13 Implementasi Mengunduh Lampiran

Implementasi mengunduh lampiran diimplementasikan dengan cara mengambil data *url* dari lampiran yang dipilih kemudian dari *url* tersebut akan diunduh *file* lampirannya. Proses mengunduh ini diimplementasikan menggunakan kelas *URLConnection* dan *InputStream*. Selanjutnya, respon diolah oleh kelas *OutputStream* untuk ditulis ke dalam bentuk *file*. Implementasi proses ini ditunjukkan pada Kode Sumber 4.57.

```
unduhLampiran(string nama_file){
Inisialisasi URLConnection
Inisialisasi direktori di
"/OperatorSurga/Lampiran"
if direktori != tidak ada
 buat direktori
end if
Inisialisasi InputStream
Inisialisasi OutputStream ke direktori
"OperatorSurga/Lampiran/nama_file"
inisialisasi byte dengan panjang 1024
int total ← 0
while ( int count ← InputStream.read(byte)
!= -1)
  total ← total+count
  publishProgress()
  OutputStream.write(byte,0,count)
End while
```

```
OutputStream.flush()
OutputStream.close()
InputStream.close()
}
```

Kode Sumber 4.57 Implementasi Mengunduh Lampiran

4.5.14 Implementasi Menampilkan Peta Lokasi Aduan

Implementasi Menampilkan peta lokasi aduan diimplementasikan menggunakan Google Maps API v2. Di dalam Google Maps API v2 ini kita harus inisialiasi peta dan mengecek apakah layanan ini tersedia dari Google jika layanan ini tersedia, maka peta akan mengarah kepada lintang dan bujur sesuai dengan lokasi aduan. Implementasi pengecekkan layanan ditunjukkan pada Kode Sumber 4.58, Implementasi inisialisasi peta ditunjukkan pada Kode Sumber 4.59, Implementasi mengubah lokasi peta ditunjukkan pada Kode Sumber 4.60, dan Implementasi menampilkan peta ditunjukkan pada Kode Sumber 4.61.

```
public boolean LayananGoogleMapSiap(){
int isAvailable ←
GooglePlayServicesUtil.isGooglePlayServicesA
vailable(this)
If isAvailable==ConnectionResult.SUCCESS
      return true
end if
else if
GooglePlayServicesUtil.isUserRecoverableErro
r(isAvailable)
Dialog d =
GooglePlayServicesUtil.getErrorDialog(isAvai
lable, this, GPS_ERRORDIALOG_REQUEST)
d.show();
end if

else
```

```
17  Toast.makeText(this, "Tidak bisa terhubung
18  dengan Google Service",
19  Toast.LENGTH_SHORT).show();
20  end if
21
22  return false;
    }
```

Kode Sumber 4.58 Implementasi Pengecekkan Layanan Google Maps

```
1  GoogleMap myMap
2  inisialiasiPeta(){
3  if myMap is null
4    inisialisasi fragmentPeta
5    myMap = fragmentPeta.getMap()
6  end if
   }
```

Kode Sumber 4.59 Implementasi Inisialisasi Peta

```
1  private void gotoLocation(double lintang,
2  double bujur,int zoom){
3  LatLng kordinatLokasi ←new
4  LatLng(lintang,bujur)
5
6  CameraUpdate update ←
7  CameraUpdateFactory.newLatLngZoom(kordinatLo
8  kasi, zoom);
9  myMap.moveCamera(update)
   }
```

Kode Sumber 4.60 Implementasi Mengubah Lokasi Peta

```
1  If LayananGoogleMapSiap()
2   setContentView(R.layout.gmaps);
3   if initMap()
4   gotoLocation(latitude, longitude, 15)
```

```
 5   myMap.addMarker(new
 6  MarkerOptions().position(new
 7  LatLng(latitude, longitude)).title("Lokasi
 8  yang di tandai"))
 9   end if
10  end if
11
12  else
13   Toast.makeText(this, "peta tidak ada",
14  Toast.LENGTH_SHORT).show()
15  End if
```

Kode Sumber 4.61 Implementasi Menampilkan Peta

4.5.15 Implementasi Validasi Pesan Singkat yang Salah

Implementasi validasi pesan singkat yang salah diimplementasi dengan mengirim data-data untuk validasi yang sudah dimasukkan oleh petugas. *Web service* ini yang akan memasukkan pesan singkat salah yang sudah divalidasi. Implementasi ini ditunjukkan pada Kode Sumber 4.62.

```
1  Inisialisasi list BasicNameValuePair
2  Masukkan key dan value nama, no_ktp, isi,
3  dan waktu ke basicNameValuePair
4
5  Inisialisasi jParser
6  jParser.makeHttpRequest(destination,"post",
7  list BasicNameValuePair)
```

Kode Sumber 4.62 Implementasi Validasi Pesan Singkat yang Salah

4.5.16 Implementasi Mengirim Pesan Singkat ke Petugas

Implementasi mengirim pesan singkat ke petugas diimplementasikan dengan cara mengirim data-data untuk

mengirim pesan singkat ke *web service.* Dimana *web service* ini akan memasukkan data-data tersebut ke dalam basis data. Jika sudah dimasukkan ke dalam basis data dengan otomatis sistem akan mengirim pesan singkat tersebut ke petugas. Implementasi ini ditunjukkan pada Kode Sumber 4.63.

```
1  Inisialisasi list BasicNameValuePair
2  Masukkan key dan value nama, no_ktp, isi,
3  dan waktu ke basicNameValuePair
4
5  Inisialisasi jParser
6  jParser.makeHttpRequest(destination,"post",
7  list BasicNameValuePair)
```

Kode Sumber 4.63 Implementasi Mengirim Pesan Singkat ke Petugas

4.5.17 Implementasi Mengunduh Laporan

Implementasi mengunduh lampiran diimplementasikan dengan cara mengirim data tahun yang dipilih ke *web service. Web service* ini yang akan menjadikan hasil *query* dari basis data menjadi *file* yang berekstensi "*.csv*" yang dapat diunduh. Proses mengunduh ini diimplementasikan menggunakan kelas *DefaultHttpClient* dan *InputStream*. Selanjutnya, respon diolah oleh kelas *OutputStream* untuk ditulis ke dalam bentuk *file*. Implementasi proses ini ditunjukkan pada Kode Sumber 4.64.

```
1  Download Laporan(string tahun, url){
2  Inisialisasi list BasicNameValuePair
3  Masukkan key dan value tahun ke
4  basicNameValuePair
5
6  Inisialisasi DefaultHttpConnection client
7  Inisialisasi HttpPost post(url)
```

```
 8  Set entity httpPost dengan
 9  basicNameValuePair
10
11  httpResponse response ← client.execute(post)
12  httpEntity entity ← response.getEntity()
13
14  if entity is not null
15    InputStream ← entity.getContent()
16    inisialisasi direktori di
17  "OperatorSurga/Laporan/"
18   if direktori is null
19    membuat direktori "OperatorSurga/Laporan/"
20   end if
21   Inisialisasi OutputStream dengan direktori
22  ditambah nama file
23   Long length ← entity.getContentlength()
24   Inisialisasi buffer byte panjang 1024
25   Int bufferlength ← 0
26   While bufferLength =
27  InputStream.read(buffer) > 0
28      publishProgress()
29
30  OutputStream.write(buffer,0,bufferLength)
31  End while
32  OutputStream.flush()
33  OutputStream.close()
34  End if
    }
```

Kode Sumber 4.64 Implementasi Mengunduh Laporan

BAB V
UJI COBA DAN EVALUASI

Pada bab ini dijelaskan mengenai rangkaian uji coba dan evaluasi yang dilakukan terhadap sistem yang dibuat. Pembahasan yang dipaparkan meliputi lingkungan uji coba, data uji coba, skenario uji coba, hasil uji coba, dan evaluasi.

5.1 Lingkungan Uji Coba

Proses uji coba terhadap sistem ini dilakukan menggunakan *smartphone* Asus Zenfone 5 dengan spesifikasi seperti berikut:

- Sistem Operasi: Android v4.4 (Kitkat)
- CPU: Dual core 2GHz Intel Atom Z2580
- Memory internal: 16 GB
- RAM: 2 GB
- Speed: HSDPA, 14.4 Mbps; HSUPA, 5.76 Mbps
- WLAN: Wi-Fi 802.11 b/g/n, Wi-Fi hotspot

5.2 Pengujian Modularitas

Pengujian ini dilakukan untuk menguji apakah modularitas benar-benar diimplementasikan dan bekerja semestinya. Selain itu juga untuk mengetahui kesesuaian keluaran dari setiap tahapan atau langkah penggunaan terhadap skenario yang dipersiapkan. Pengujian dilakukan dengan metode kotak hitam.

5.2.1 Skenario Pengujian Modularitas

Pada bagian ini akan dilakukan sejumlah pengujian terhadap perangkat lunak untuk menguji kebenaran modularitas dari aplikasi ini. Pengujian ini dilakukan dengan secara mandiri dan didokumentasikan secara sistematis dengan menyiapkan skenario sebagai tolak ukur keberhasilan sistem.

Skenario yang disiapkan adalah skenario dalam melakukan membuat dan memuat modul dari aplikasi ini. Berikut ini adalah hasil pengujian membuat dan memuat modul yang telah diimplementasikan pada tahap pengembangan.

5.2.1.1 Pengujian Membuat dan Memuat Modul Aplikasi

Pengujian ini dilakukan terhadap memuat suatu modul aplikasi yang telah mengimplementasikan *interface* aplikasi utama. Modul aplikasi ini merupakan modul aplikasi *monitoring* surga. Pengujian ini dimulai ketika modul aplikasi *monitoring* surga telah disiapkan sudah mengimplementasikan *inteface* aplikasi utama dan mengubah *androidmanifest.xml* sesuai dengan subbab 3.2.4. Kemudian modul aplikasi tersebut dipasang ke dalam perangkat Android. Kelas yang mengimplementasi *interface* dari aplikasi utama di ekspor ke dalam *file jar* dengan menggunakan *manifest.mf* yang sudah dibuat. *File jar* tersebut di *compile* menggunakan *dex* yang ada di SDK Android dan menghasilkan *file classes.dex*. *File classes.dex* dimasukkan ke dalam *file jar*. Lalu *copy file jar* tersebut ke dalam perangkat Android di dalam direktori "Plugin". Hasil pengujian terhadap pengujian ini digambarkan dalam Gambar 5.1.

Gambar 5.1 Pengujian Membuat dan Memuat Modul Aplikasi Monitoring

5.3 Pengujian Fungsionalitas

Pengujian ini dilakukan untuk menguji apakah fungsionalitas yang diindentifikasi pada tahap kebutuhan benar-benar diimplementasikan dan bekerja semestinya. Selain itu juga untuk mengetahui kesesuaian keluaran dari setiap tahapan atau langkah penggunaan fitur terhadap skenario yang dipersiapkan. Pengujian dilakukan dengan metode *black-box.*

5.3.1 Skenario Pengujian Fungsionalitas

Pada bagian ini akan dilakukan sejumlah pengujian perangkat lunak untuk menguji kebenaran dari aplikasi ini. Pengujian fungsionalitas perangkat lunak ini dilakukan secara mandiri dan didokumentasikan secara sistematis dengan

menyiapkan sejumlah skenario sebagai tolak ukur keberhasilan sistem. Pengujian ini meliputi seluruh kasus penggunaan yang telah dijelaskan pada Bab 3. Pengujian fungsionalitas ini meliputi proses yg dijabarkan sebagai berikut.

1. Menampilkan Aduan
2. Mencari Aduan
3. Menjawab Aduan
4. Mengubah Status Aduan
5. Mengubah Prioritas Aduan
6. Memilih Petugas Aduan
7. Mengembalikan Aduan
8. Mengubah Status *Spam* Aduan
9. Menampilkan Lampiran Aduan
10. Mengunduh Lampiran
11. Menampilkan Peta Lokasi Aduan
12. Validasi Pesan Singkat yang Salah
13. Mengirim Pesan Singkat ke Petugas
14. Mengunduh Laporan

5.3.2 Hasil Pengujian Fungsionalitas

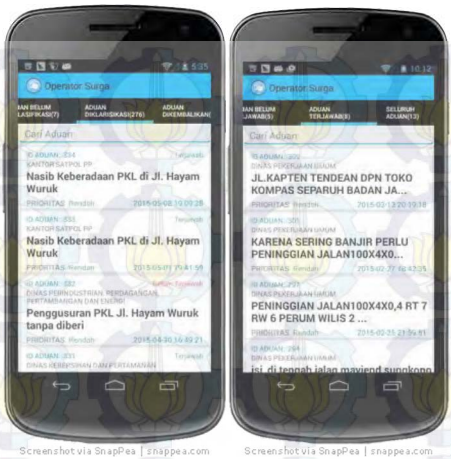
Hasil pengujian dari poin-poin dari skenario pada subbab sebelumnya dilampirkan pada bagian subbab ini. Berikut ini adalah hasil pengujian fungsionalitas fitur yang telah diimplementasikan pada tahap pengembangan.

5.3.2.1 Pengujian Menampilkan Aduan

Pengujian ini dilakukan terhadap fungsionalitas menampilkan daftar aduan berdasarkan pengguna yang telah login ke dalam sistem. Pengujian ini dimulai ketika pengguna sudah berhasil masuk ke dalam sistem melalui *login*, memilih menu operator surga, dan memilih sub menu manajemen aduan. dimana pada saat berhasil *login*, informasi ID pengguna akan disimpan dalam session aplikasi. Tabel 5.1 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.1.

Tabel 5.1 Prosedur Uji Coba Menampilkan Aduan

ID	**UJ-01**
Referensi Use Case	**UC-01**
Nama	**Uji Coba Menampilkan Aduan**
Tujuan Uji Coba	Menguji fitur untuk menampilkan aduan setelah menekan sub menu manajemen aduan
Kondisi Awal	Pengguna telah berada pada antarmuka sub menu
Data Masukan	-*Username* pengguna 'humas' untuk petugas tingkat satu -*Username* pengguna 'pu1' untuk petugas tingkat dua
Prosedur Pengujian	1. Menekan sub menu manajemen aduan untuk masuk ke antarmuka manajemen aduan dan dapat melihat aduan
Data Keluaran	Data-data aduan berdasarkan pengguna yang telah *login*.
Hasil Uji Coba	**Berhasil**



Gambar 5.2 Pengujian Menampilkan Aduan

5.3.2.2 Pengujian Mencari Aduan

Pengujian ini dilakukan terhadap fungsionalitas mencari aduan. Pengujian ini dimulai ketika pengguna mengisi *textView* pada halaman manajemen aduan di *tab* manapun. Tabel 5.2 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.3.

Tabel 5.2 Prosedur Uji Coba Mencari Aduan

ID	**UJ-02**
Referensi Use Case	**UC-02**
Nama	**Uji Coba Mencari Aduan**
Tujuan Uji Coba	Menguji fitur untuk mencari aduan.
Kondisi Awal	Pengguna telah berada pada antarmuka manajemen aduan
Data Masukan	-*Username* pengguna 'humas'
Prosedur Pengujian	1. Memilih tab aduan diklasifikasi 2. Memasukkan kata 'dinas sosial'
Data Keluaran	Aduan-aduan yang memiliki kata 'dinas sosial' pada aduannya.
Hasil Uji Coba	**Berhasil**

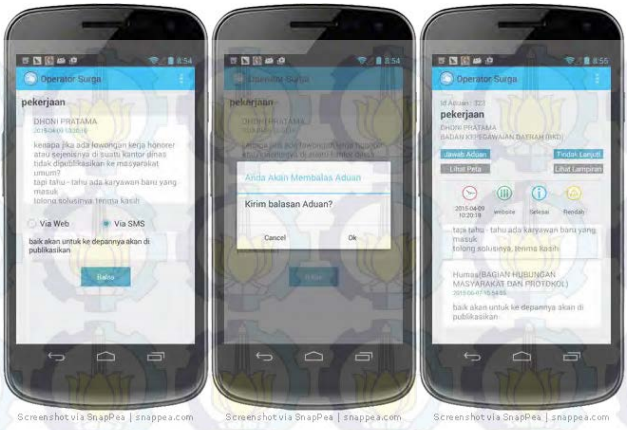
Gambar 5.3 Pengujian Mencari Aduan

5.3.2.3 Pengujian Menjawab Aduan

Pengujian ini dilakukan terhadap fungsionalitas menjawab aduan untuk menjawab aduan yang dipilih. Pengujian ini dimulai ketika pengguna menekan *button* jawab di antarmuka detail aduan dan masuk ke antarmuka jawab aduan. Tabel 5.3 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.4.

Tabel 5.3 Prosedur Uji Coba Menjawab Aduan

ID	**UJ-03**
Referensi Use Case	**UC-03**
Nama	**Uji Coba Menjawab Aduan**
Tujuan Uji Coba	Menguji fitur untuk menjawab aduan untuk menjawab aduan yang dipilih
Kondisi Awal	Pengguna telah berada pada antarmuka jawab aduan
Data Masukan	-*Username* pengguna 'humas' -Memilih aduan yang mempunyai id '323'
Prosedur Pengujian	1. Mengisi balasan aduan 2. Memilih *radio button* untuk memilih membalas melalui situs atau pesan singkat 3. Tekan *button* balas 4. Menekan tombol ok di *pop up dialog*
Data Keluaran	Data balasan aduan akan muncul
Hasil Uji Coba	**Berhasil**



Gambar 5.4 Pengujian Menjawab Aduan

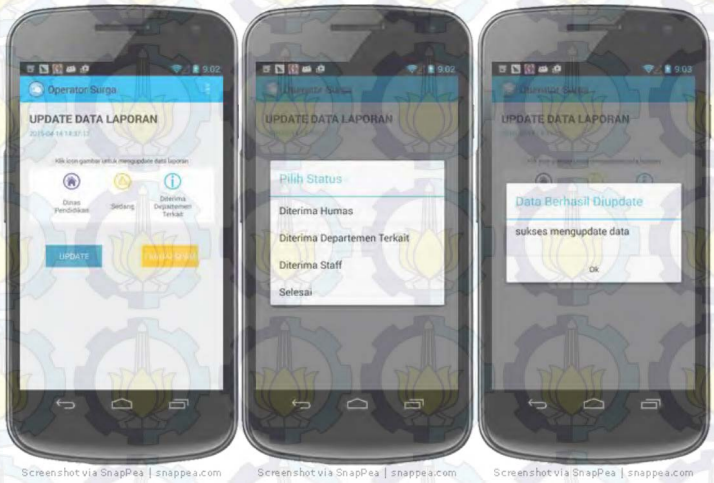
5.3.2.4 Pengujian Mengubah Status Aduan

Pengujian ini dilakukan terhadap fungsionalitas mengubah status aduan aduan untuk menanggapi aduan yang dipilih. Pengujian ini dimulai ketika pengguna menekan *button* tindak lanjuti di antarmuka detail aduan dan masuk ke antarmuka tindak lanjut aduan. Pengujian ini akan menampilkan semua pilihan status dengan menekan *icon* status dan dapat memilih status tersebut. Tabel 5.4 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.5.

Tabel 5.4 Prosedur Uji Coba Mengubah Status Aduan

ID	**UJ-04**
Referensi Use Case	**UC-04**
Nama	**Uji Coba Memilih Status Aduan**
Tujuan Uji Coba	Menguji fitur untuk memilih status aduan untuk aduan yang dipilih.

Kondisi Awal	Pengguna telah berada pada antarmuka tindak lanjut aduan
Data Masukan	-*Username* pengguna 'humas' -Memilih aduan yang mempunyai id '328'
Prosedur Pengujian	1. Menekan *icon* status 2. Memilih status diterima staff 3. Menekan tombol update
Data Keluaran	Data status untuk aduan 328 berubah menjadi diterima staff
Hasil Uji Coba	**Berhasil**



Gambar 5.5 Pengujian Mengubah Status Aduan

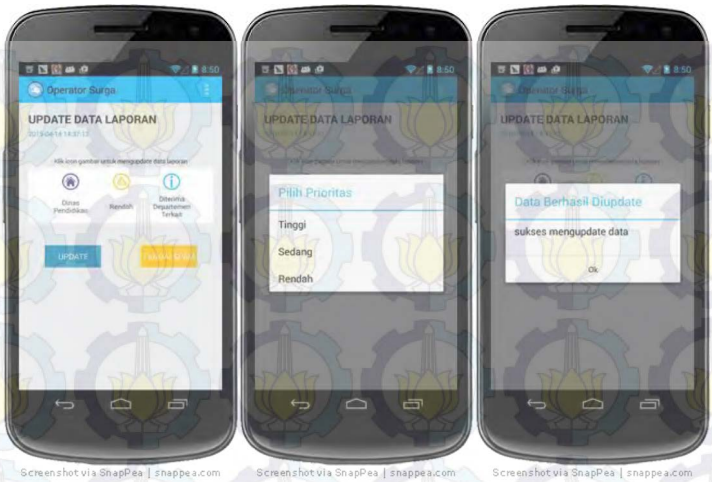
5.3.2.5 Pengujian Mengubah Prioritas Aduan

Pengujian ini dilakukan terhadap fungsionalitas mengubah prioritas aduan aduan untuk menanggapi aduan yang dipilih. Pengujian ini dimulai ketika pengguna menekan *button* tindak lanjuti di antarmuka detail aduan dan masuk ke antarmuka tindak lanjut aduan. Pengujian ini akan

menampilkan semua pilihan prioritas dengan menekan *icon* prioritas dan dapat memilih prioritas tersebut. Tabel 5.5 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.6.

Tabel 5.5 Prosedur Uji Coba Mengubah Prioritas Aduan

ID	**UJ-05**
Referensi Use Case	**UC-05**
Nama	**Uji Coba Mengubah Prioritas Aduan**
Tujuan Uji Coba	Menguji fitur untuk mengubah prioritas aduan untuk menanggapi aduan yang dipilih.
Kondisi Awal	Pengguna telah berada pada antarmuka tindak lanjut aduan
Data Masukan	-*Username* pengguna 'humas' -Memilih aduan yang mempunyai id '328'
Prosedur Pengujian	1. Menekan *icon* prioritas 2. Memilih prioritas sedang 3. Menekan tombol *update* 4. Menekan tombol *update pop up*
Data Keluaran	Data prioritas untuk aduan 328 berubah menjadi sedang
Hasil Uji Coba	**Berhasil**



Gambar 5.6 Pengujian Mengubah Prioritas Aduan

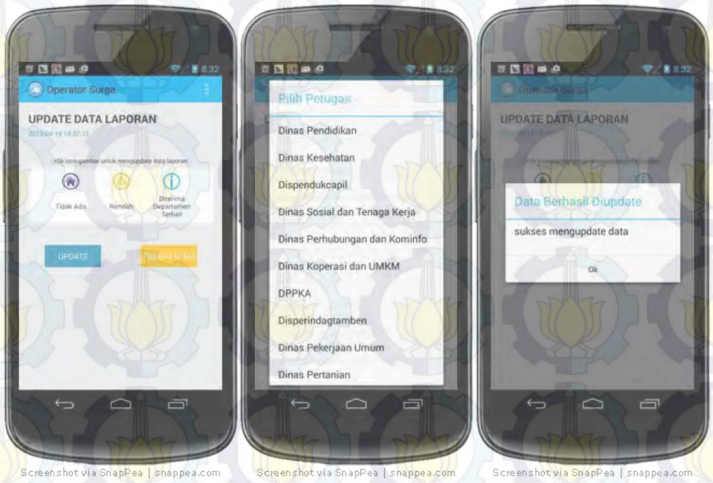
5.3.2.6 Pengujian Memilih Petugas Aduan

Pengujian ini dilakukan terhadap fungsionalitas memilih petugas aduan untuk menanggapi aduan yang dipilih. Pengujian ini dimulai ketika pengguna menekan *button* tindak lanjuti di antarmuka detail aduan dan masuk ke antarmuka tindak lanjut aduan. Pengujian ini akan menampilkan semua petugas tingkat dua jika menekan *icon* petugas dan dapat memilih petugas tersebut. Tabel 5.6 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.7.

Tabel 5.6 Prosedur Uji Coba Memilih Petugas Aduan

ID	**UJ-06**
Referensi Use Case	**UC-06**
Nama	**Uji Coba Memilih Petugas Aduan**
Tujuan Uji Coba	Menguji fitur untuk memilih petugas aduan untuk menanggapi aduan yang dipilih.

Kondisi Awal	Pengguna telah berada pada antarmuka tindak lanjut aduan
Data Masukan	-*Username* pengguna 'humas' untuk petugas tingkat satu -Memilih aduan yang mempunyai id '328'
Prosedur Pengujian	1. Menekan *icon* petugas 2. Memilih petugas dinas pendidikan 3. Menekan tombol *update* 4. Menekan tombol *update pop up*
Data Keluaran	Data petugas yang menanggapi aduan 328 berubah menjadi dinas pendidikan
Hasil Uji Coba	**Berhasil**



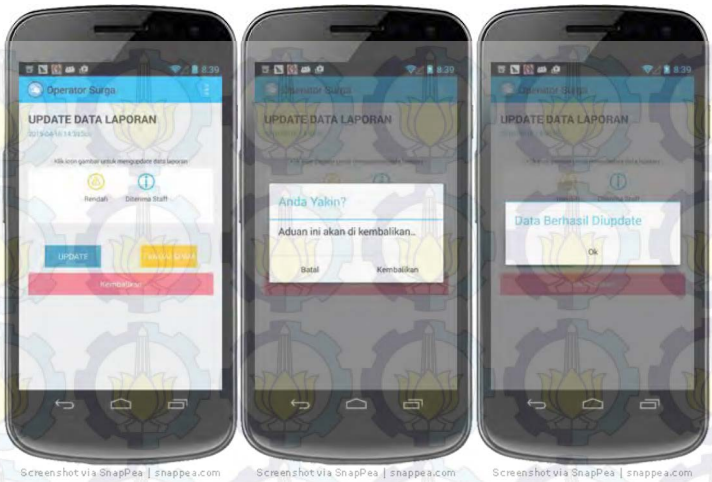
Gambar 5.7 Pengujian Memilih Petugas

5.3.2.7 Pengujian Mengembalikan Aduan

Pengujian ini dilakukan terhadap fungsionalitas mengambalikan aduan kepada petugas tingkat satu karena tidak berhubungan dengan SKPD petugas yang *login*. Pengujian ini dimulai ketika pengguna menekan *button* tindak lanjuti di antarmuka detail aduan dan masuk ke antarmuka tindak lanjut aduan. Tabel 5.7 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.8.

Tabel 5.7 Prosedur Uji Coba Mengembalikan Aduan

ID	**UJ-07**
Referensi Use Case	**UC-07**
Nama	**Uji Coba Mengembalikan Aduan**
Tujuan Uji Coba	Menguji fitur untuk mengembalikan aduan kepada petugas tingkat satu
Kondisi Awal	Pengguna telah berada pada antarmuka tindak lanjut aduan
Data Masukan	-*Username* pengguna ‘pu1’ -Memilih aduan yang mempunyai id ‘330’
Prosedur Pengujian	1. Menekan *button* kembalikan
Data Keluaran	Data aduan yang ber id ‘330’ dikembalikan di petugas tingkat satu
Hasil Uji Coba	**Berhasil**



Gambar 5.8 Pengujian Mengembalikan Aduan

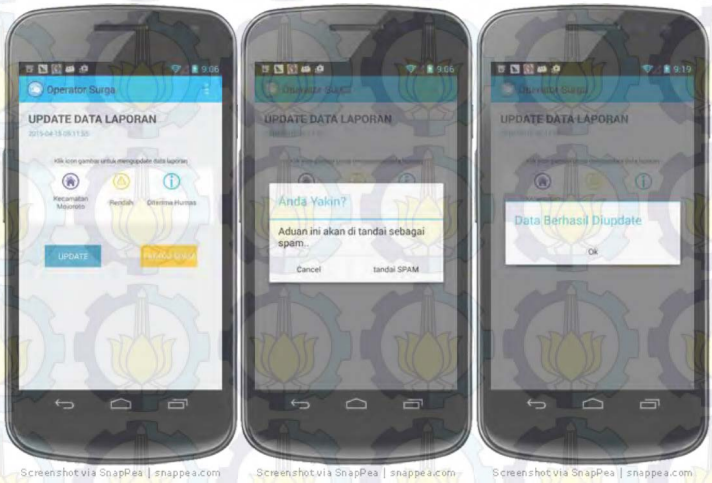
5.3.2.8 Pengujian Status *Spam* Aduan

Pengujian ini dilakukan terhadap fungsionalitas status spam aduan untuk menandai bahwa aduan tersebut adalah spam. Pengujian ini dimulai ketika pengguna menekan *button* tindak lanjuti di antarmuka detail aduan dan masuk ke antarmuka tindak lanjut aduan. Tabel 5.8 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.9

Tabel 5.8 Prosedur Uji Coba Status Spam Aduan

ID	**UJ-08**
Referensi Use Case	**UC-08**
Nama	**Uji Coba Status *Spam* Aduan**
Tujuan Uji Coba	Menguji fitur untuk menandai aduan sebagai *spam*
Kondisi Awal	Pengguna telah berada pada antarmuka tindak lanjut aduan

Data Masukan	-*Username* pengguna ‘humas’ -Memilih aduan yang mempunyai id ‘329’
Prosedur Pengujian	1. Menekan *button* tandai *spam*
Data Keluaran	Aduan yang mempunyai id ‘329’ ditandai menjadi *spam*
Hasil Uji Coba	**Berhasil**



Gambar 5.9 Pengujian Status *Spam* Aduan

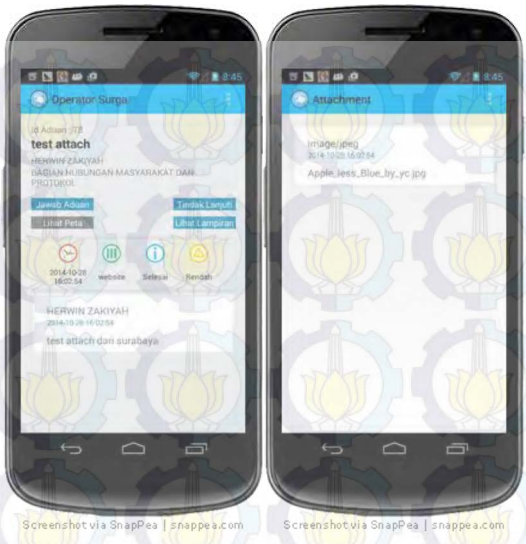
5.3.2.9 Pengujian Menampilkan Lampiran Aduan

Pengujian ini dilakukan terhadap fungsionalitas menampilkan lampiran-lampiran aduan yang disertakan. Pengujian ini dimulai ketika pengguna menekan *button* lihat lampiran di antarmuka detail aduan dan masuk ke antarmuka lihat lampiran. Pengujian ini akan menampilkan semua lampiran yang disertakan pada aduan yang dipilih. Tabel 5.9 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil

pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.10.

Tabel 5.9 Prosedur Uji Coba Menampilkan Lampiran Aduan

ID	**UJ-09**
Referensi Use Case	**UC-09**
Nama	**Uji Coba Menampilkan Lampiran Aduan**
Tujuan Uji Coba	Menguji fitur untuk melihat semua lampiran yang disertakan pada aduan yang dipilih
Kondisi Awal	Pengguna telah berada pada antarmuka detail aduan
Data Masukan	-*Username* pengguna 'humas' -Memilih aduan yang mempunyai id '30'
Prosedur Pengujian	1. Menekan *button* lihat lampiran
Data Keluaran	Data semua lampiran yang disertakan pada aduan yang dipilih
Hasil Uji Coba	**Berhasil**



Gambar 5.10 Pengujian Menampilkan Lampiran Aduan

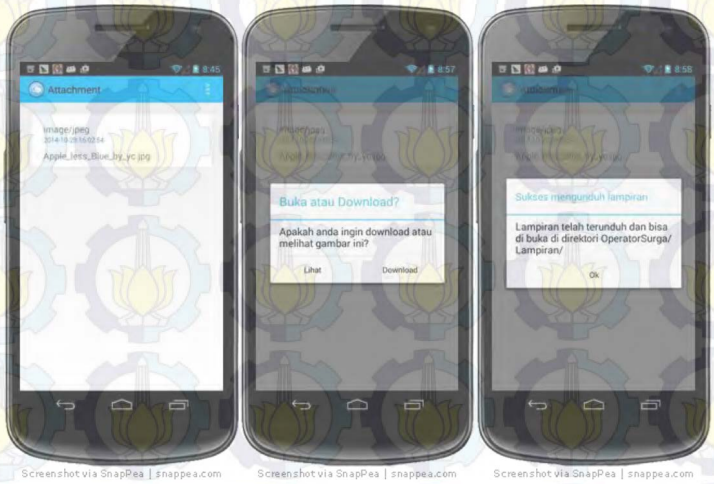
5.3.2.10 Pengujian Mengunduh Lampiran

Pengujian ini dilakukan terhadap fungsionalitas mengunduh lampiran untuk lampiran yang dipilih. Pengujian ini dimulai ketika pengguna menekan *listview* lampiran di antarmuka lihat lampiran. Tabel 5.10 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.11.

Tabel 5.10 Prosedur Uji Coba Mengunduh Lampiran

ID	**UJ-10**
Referensi Use Case	**UC-10**
Nama	**Uji Coba Mengunduh Lampiran**
Tujuan Uji Coba	Menguji fitur untuk Mengunduh lampiran yang dipilih

Kondisi Awal	Pengguna telah berada pada antarmuka lihat lampiran
Data Masukan	-*Username* pengguna 'humas' -Memilih aduan yang mempunyai id '78'
Prosedur Pengujian	1. Memilih lampiran '' 2. Menekan tombol unduh
Data Keluaran	*File* yang dipilih akan terunduh.
Hasil Uji Coba	**Berhasil**



Gambar 5.11 Pengujian Mengunduh Lampiran

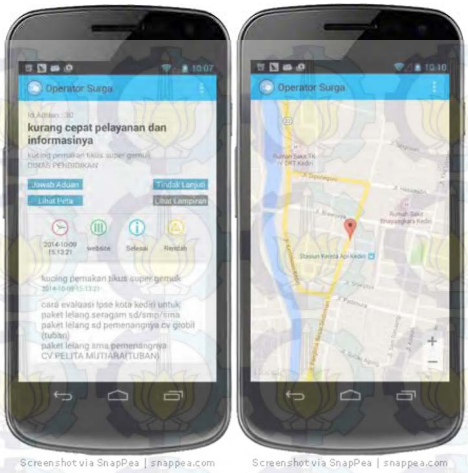
5.3.2.11 Pengujian Menampilkan Peta Lokasi Aduan

Pengujian ini dilakukan terhadap fungsionalitas menampilkan lokasi peta dari aduan yang disertakan. Pengujian ini dimulai ketika pengguna menekan *button* lihat peta di antarmuka detail aduan. Pengujian ini akan menampilkan lokasi peta disertakan pada aduan yang dipilih. Tabel 5.11 menjelaskan skenario dari pengujian fungsionalitas

ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.12.

Tabel 5.11 Prosedur Uji Coba Menampilkan Peta Lokasi Aduan

ID	**UJ-11**
Referensi Use Case	**UC-11**
Nama	**Uji Coba Menampilkan Peta Lokasi Aduan**
Tujuan Uji Coba	Menguji fitur untuk menampilkan peta lokasi aduan.
Kondisi Awal	Pengguna telah berada pada antarmuka detail aduan
Data Masukan	-*Username* pengguna 'humas' -Memilih aduan yang mempunyai id '78'
Prosedur Pengujian	1. Menekan *button* lihat peta
Data Keluaran	Lokasi peta akan muncul
Hasil Uji Coba	**Berhasil**



Gambar 5.12 Pengujian Menampilkan Peta Lokasi Aduan

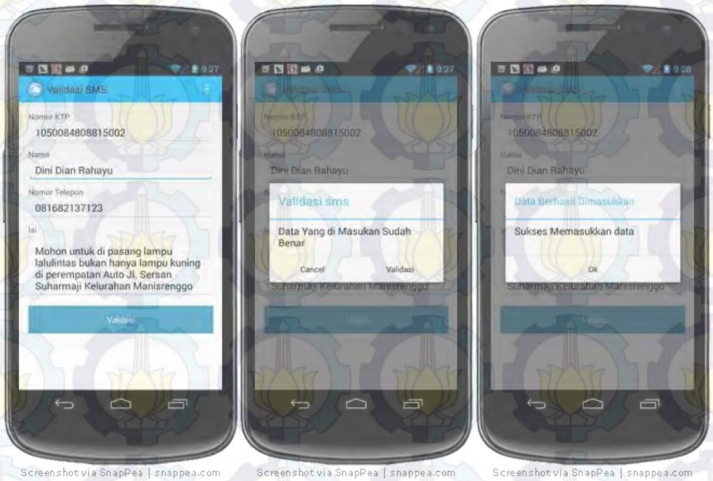
5.3.2.12 Pengujian Validasi Pesan Singkat yang Salah

Pengujian ini dilakukan terhadap fungsionalitas validasi pesan singkat yang salah. Pengujian ini dimulai ketika pengguna menekan *listview* sms yang tidak valid di antarmuka *list* validasi sms. Pengujian ini akan menampilkan nomor telepon pengirim dan isi laporan aduan. Tabel 5.12 menjelaskan skenario dari pengujian fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.13.

Tabel 5.12 Prosedur Uji Coba Validasi SMS yang Salah

ID	**UJ-12**
Referensi Use Case	**UC-12**
Nama	**Uji Coba Validasi Pesan Singkat yang Salah**
Tujuan Uji Coba	Menguji fitur untuk memvalidasi pesan singkat yang salah

Kondisi Awal	Pengguna telah berada pada antarmuka validasi sms
Data Masukan	-*Username* pengguna 'humas' -*id sms tidak valid* '2'
Prosedur Pengujian	1. Memasukkan nomor ktp 2. Memasukkan nama 3. Menekan tombol *validasi*
Data Keluaran	Sms yang divalidasi tersebut menjadi aduan.
Hasil Uji Coba	**Berhasil**



Gambar 5.13 Pengujian Validasi Pesan Singkat yang Salah

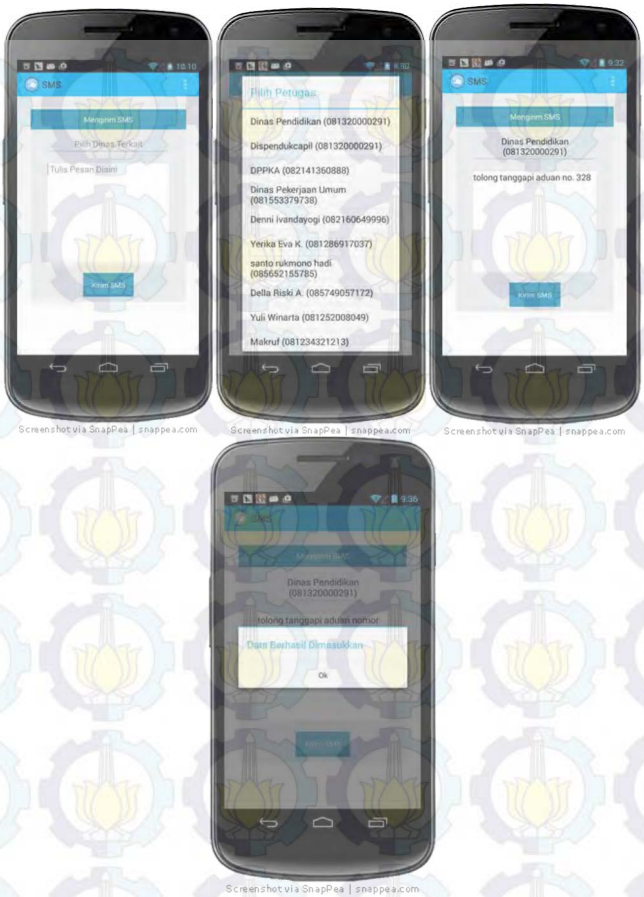
5.3.2.13 Pengujian Mengirim Pesan Singkat ke Petugas

Pengujian ini dilakukan terhadap fungsionalitas mengirim pesan singkat ke petugas. Pengujian ini dimulai ketika pengguna menekan *listview* sms di antarmuka sub menu. Pengujian ini akan menampilkan petugas yang dapat dikirimi sms. Tabel 5.13 menjelaskan skenario dari pengujian

fungsionalitas ini. Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.14.

Tabel 5.13 Prosedur Uji Coba Mengirim Pesan Singkat ke Petugas

ID	**UJ-13**
Referensi Use Case	**UC-13**
Nama	**Uji Coba Mengirim Pesan Singkat ke Petugas**
Tujuan Uji Coba	Menguji fitur untuk mengirim pesan singkat ke petugas
Kondisi Awal	Pengguna telah berada pada antarmuka sms
Data Masukan	-*Username* pengguna 'humas'
Prosedur Pengujian	1. Menekan *textview* petugas 2. Memilih petugas 'dinas pendidikan' 3. Memasukkan pesan pesan singkat 4. Menekan *button* kirim
Data Keluaran	Pesan singkat akan terkirim ke petugas
Hasil Uji Coba	**Berhasil**



Gambar 5.14 Pengujian Mengirim Pesan Singkat ke Petugas

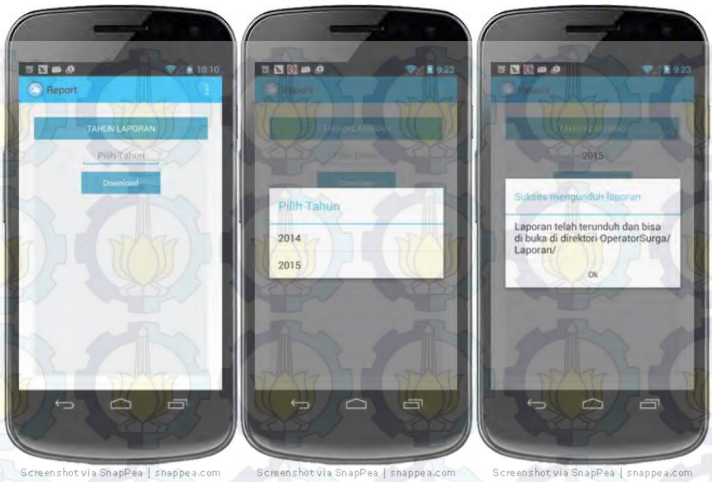
5.3.2.14 Pengujian Mengunduh Laporan

Pengujian ini dilakukan terhadap fungsionalitas mengunduh laporan. Pengujian ini dimulai ketika pengguna menekan *listview* laporan di antarmuka sub menu. Pengujian ini akan menampilkan tahun laporan yang ingin diunduh. Tabel 5.14 menjelaskan skenario dari pengujian fungsionalitas ini.

Hasil pengujian terhadap fungsionalitas digambarkan dalam Gambar 5.15.

Tabel 5.14 Prosedur Uji Coba Mengunduh Laporan

ID	**UJ-14**
Referensi Use Case	**UC-14**
Nama	**Uji Coba Mengunduh Laporan**
Tujuan Uji Coba	Menguji fitur untuk mengunduh laporan berdasarkan tahun
Kondisi Awal	Pengguna telah berada pada antarmuka laporan
Data Masukan	-*Username* pengguna 'humas'
Prosedur Pengujian	1. Menekan *textview* tahun 2. Memilih tahun '2015' 3. Menekan *button* unduh
Data Keluaran	Laporan tahun 2015 terunduh.
Hasil Uji Coba	**Berhasil**



Gambar 5.15 Pengujian Mengunduh Laporan

5.4 Evaluasi

Dari uji coba yang telah dilakukan, diketahui bahwa sistem telah dapat bekerja dengan baik dan benar dalam menjalankan fungsionalitasnya.

Pada uji coba modularitas, Modul berhasil di tampilkan pada antarmuka menu. Modul juga dapat dibuka dan berjalan sesuai dengan harapan.

Kemudian, berdasarkan hasil pengujian fungsionalitas, seluruh skenario berhasil dilakukan. Evaluasi terhadap pengujian fungsionalitas yang telah dilaksanakan dijelaskan sebagai berikut:

1. Fungsionalitas menampilkan aduan berjalan sesuai dengan yang diharapkan
2. Fungsionalitas mencari aduan berjalan sesuai dengan yang diharapkan.
3. Fungsionalitas menjawab aduan berjalan sesuai dengan yang diharapkan.

4. Fungsionalitas mengubah status aduan berjalan sesuai dengan yang diharapkan.
5. Fungsionalitas mengubah prioritas aduan berjalan sesuai dengan yang diharapkan.
6. Fungsionalitas memilih petugas berjalan sesuai dengan yang diharapkan.
7. Fungsionalitas mengembalikan aduan berjalan sesuai dengan yang diharapkan.
8. Fungsionalitas mengubah status *spam* aduan berjalan sesuai dengan yang diharapkan.
9. Fungsionalitas menampilkan lampiran sesuai dengan yang diharapkan.
10. Fungsionalitas mengunduh lampiran sesuai dengan yang diharapkan.
11. Fungsionalitas menampilkan peta klokasi aduan sesuai dengan yang diharapkan.
12. Fungsionalitas validasi SMS yang salah sesuai dengan yang diharapkan.
13. Fungsionalitas mengirim SMS ke petugas sesuai dengan yang diharapkan.
14. Fungsionalitas mengunduh laporan sesuai dengan yang diharapkan.

Dari keseluruhan hasil pengujian fungsionalitas pada modul aplikasi Tugas Akhir ini, seluruh skenario telah berhasil dilakukan. Modul aplikasi ini telah berjalan sesuai dengan yang diharapkan.

BAB VI
KESIMPULAN DAN SARAN

Dalam bab terakhir ini dijelaskan kesimpulan yang didapat dari pengerjaan Tugas Akhir ini beserta saran-saran yang dapat dipertimbangkan untuk pengembangan atau penelitian lebih lanjut.

6.1 Kesimpulan

Berdasarkan hasil uji coba yang telah dilakukan, terdapat beberapa kesimpulan yang bisa diambil, yaitu:

1. Aplikasi Modular Suara Warga berhasil dibangun pada perangkat Android 4.1 keatas.
2. Aplikasi dapat digunakan oleh petugas Pemerintah Kota Kediri dalam menanggapi setiap aduan, saran, dan informasi yang diberikan oleh masyarakat Kota Kediri.
3. Aplikasi ini dapat mengimplementasikan modularitas. oleh karena itu, aplikasi ini dapat menambah modul-modul aplikasi yand dimana modul aplikasi tersebut mengimplementasikan *interface* dari aplikasi ini.
4. Aplikasi ini diimplementasikan menurut rancangan untuk menanggapi setiap aduan, saran, dan informasi yang diberikan oleh masyarakat Kota Kediri dengan menggunakan aplikasi berbasis perangkat bergerak.

6.2 Saran

Adapun saran yang dapat dipertimbangkan untuk pengembangan atau penelitian lebih lanjut adalah sebagai berikut:

1. Meskipun pengguna Android sudah sangat marak, akan lebih baik lagi jika Tugas Akhir ini dapat

diimplementasikan juga ke dalam *platform* lain seperti iOS dan Windows Phone.

2. Antarmuka dapat dioptimalkan mengikuti perkembangan sistem operasi Android.
3. Untuk mengoptimalkan proses modularitas dibutuhkan riset lebih lanjut.

DAFTAR PUSTAKA

[1] BPS, "Sensus Kota Kediri", 2013, http://kedirikota.bps.go.id/?hal=publikasi , diakses pada 17 Desember 2014

[2] Giurca, Adrian, "JSON Rules", 2007, Brandenburg University of Technology, Germany

[3] Google, Inc, "Introduction to Android", http://developer.android.com/guide/index.html diakses pada 17 Desember 2014

[4] MySQL™, "MySQL Documentation: About MySQL Documentation" http://dev.mysql.com/doc/index-about.html diakses pada 17 Desember 2014

[5] Shunfu Hu, "Online Map Application Development Using Google Maps API, SQL Database, and ASP.NET" 2013, International Journal of Information and Communication Technology Research

[6] Tsenov, Martin "Web Services Example with PHP/SOAP", 2007 , International Conference on Computer Systems and Technologies, German,

BAB VI
KESIMPULAN DAN SARAN

Dalam bab terakhir ini dijelaskan kesimpulan yang didapat dari pengerjaan Tugas Akhir ini beserta saran-saran yang dapat dipertimbangkan untuk pengembangan atau penelitian lebih lanjut.

6.1 Kesimpulan

Berdasarkan hasil uji coba yang telah dilakukan, terdapat beberapa kesimpulan yang bisa diambil, yaitu:

1. Aplikasi Modular Suara Warga berhasil dibangun pada perangkat Android 4.1 keatas.
2. Aplikasi dapat digunakan oleh petugas Pemerintah Kota Kediri dalam menanggapi setiap aduan, saran, dan informasi yang diberikan oleh masyarakat Kota Kediri.
3. Aplikasi ini dapat mengimplementasikan modularitas. oleh karena itu, aplikasi ini dapat menambah modul-modul aplikasi yand dimana modul aplikasi tersebut mengimplementasikan *interface* dari aplikasi ini.
4. Aplikasi ini diimplementasikan menurut rancangan untuk menanggapi setiap aduan, saran, dan informasi yang diberikan oleh masyarakat Kota Kediri dengan menggunakan aplikasi berbasis perangkat bergerak.

6.2 Saran

Adapun saran yang dapat dipertimbangkan untuk pengembangan atau penelitian lebih lanjut adalah sebagai berikut:

1. Meskipun pengguna Android sudah sangat marak, akan lebih baik lagi jika Tugas Akhir ini dapat

LAMPIRAN A
PERANCANGAN DATA

Pada subbab ini dijelaskan tentang rancangan basis data yang akan digunakan pada Tugas Akhir ini. Basis data yang digunakan merupakan basis data pada sistem Suara Warga Kota Kediri yang dibangun menggunakan RDBMS MySQL.

1. Tabel aduan

Tabel aduan digunakan untuk menyimpan data seluruh aduan yang masuk ke sistem Suara Warga Kota Kediri. Dimana setiap baris pada tabel mewakili satu buah aduan dari warga. Atribut pada tabel ini dijelaskan pada Tabel A.1.

Tabel A.1 Tabel Aduan

Atribut	**Tipe**	***Null***	***Default***	**Keterangan**
id_aduan	int	Tidak	*autoincrement*	PK
no_identitas	varchar (45)	Ya	*null*	ktp pengadu
nama	varchar (45)	Ya	*null*	nama pengadu
alamat	varchar (45)	Ya	*null*	alamat pengadu
email	varchar (45)	Ya	*null*	email pengadu
no_hp	varchar (45)	Ya	*null*	no. telepon pengadu
tgl_lahir	date	Ya	*null*	tanggal lahir pengadu
topik	varchar (45)	Ya	*null*	topik aduan
isi	text	Ya	*null*	isi aduan

Atribut	Tipe	*Null*	*Default*	Keterangan
waktu	time stamp	Ya	*null*	waktu aduan ditulis
alamat_ip	varchar (45)	Ya	*null*	alamat ip pengadu
user_agent	varchar (2048)	Ya	*null*	*browser* pengadu jika aduan dikirim via *website*
longitude	varchar (45)	Ya	*null*	bujur informasi peta
latitude	varchar (45)	Ya	*null*	lintang informasi peta
rating	int	Ya	0	rating aduan
via_sms	tinyint	Ya	*null*	dikirim melalui sms
via_petugas	int	Ya	*null*	dikirim melalui petugas
info	int	Tidak	0	informasi aduan
info_detail	text	Ya	*null*	informasi detail aduan
spam	tinyint	Tidak	0	penanda jika aduan termasuk spam

2. Tabel detail_aduan

Tabel detail_aduan digunakan untuk menyimpan data isi aduan dan tanggapan dari petugas yang ada pada sistem Suara Warga Kota Kediri. Dimana setiap baris pada tabel mewakili satu buah aduan atau tanggapan. Atribut pada tabel ini dijelaskan pada Tabel A.2.

Tabel A.2 Tabel detail_aduan

Atribut	Tipe	*Null*	*Default*	Keterangan
id_detail_aduan	int	Tidak	*autoincrement*	PK
isi_detail	text	Ya	*null*	isi aduan atau tanggapan
waktu_detail	datetime	Ya	*null*	waktu aduan atau tanggapan ditulis

3. Tabel petugas

Tabel petugas digunakan untuk menyimpan data petugas yang memiliki akun pada sistem Suara Warga Kota Kediri. Dimana setiap baris pada tabel mewakili satu akun milik petugas. Atribut pada tabel ini dijelaskan pada Tabel A.3.

Tabel A.3 Tabel petugas

Atribut	Tipe	*Null*	*Default*	Keterangan
id_petugas	int	Tidak	*autoincrement*	PK
username_petugas	varchar (45)	Ya	*null*	username petugas
password_petugas	varchar (45)	Ya	*null*	password petugas

Atribut	Tipe	*Null*	*Default*	Keterangan
nama_ petugas	varchar (45)	Ya	*null*	nama petugas
email_ petugas	varchar (45)	Ya	*null*	email petugas
no_hp_ petugas	varchar (45)	Ya	*null*	no. telepon petugas
jobdesc_ petugas	varchar (45)	Ya	*null*	deskripsi petugas
bisa_sms	tinyint	Tidak	0	penanda jika petugas dapat mengirim sms

4. Tabel departemen

Tabel departemen digunakan untuk menyimpan data seluruh departemen yang ada pada struktur pemerintahan Kota Kediri. Dimana setiap baris pada tabel mewakili satu departemen. Atribut pada tabel ini dijelaskan pada Tabel A.4.

Tabel A.4 Tabel departemen

Atribut	Tipe	*Null*	*Default*	Keterangan
id_ departemen	int	Tidak	*autoincrement*	PK
nama_ departemen	varchar (128)	Ya	*null*	nama departemen
nama_ kepala	varchar (128)	Ya	*null*	nama kepala departemen
no_hp	varchar (45)	Ya	*null*	no. telepon kepala departemen

apakah_ mitra	tinyint	Ya	*null*	penanda jika suatu departemen termasuk mitra

5. Tabel *role*

Tabel role digunakan untuk menyimpan data *role* atau peranan dari setiap petugas. *Role* ini akan berpengaruh pada hak akses yang dimiliki oleh suatu petugas dalam mengakses sistem Suara Warga Kota Kediri. Dimana setiap baris pada tabel mewakili satu jenis *role*. Atribut pada tabel ini dijelaskan pada Tabel A.5.

Tabel A.5 Tabel *role*

Atribut	Tipe	*Null*	*Default*	Keterangan
id_role	int	Tidak	-	PK
nama_role	varchar (45)	Ya	*null*	nama role
deskripsi_ role	text	Ya	*null*	deskripsi role

6. Tabel kategori

Tabel kategori digunakan untuk menyimpan data kategori-kategori aduan. Dimana setiap baris pada tabel mewakili satu kategori aduan. Atribut pada tabel ini dijelaskan pada Tabel A.6.

Tabel A.6 Tabel kategori

Atribut	Tipe	*Null*	*Default*	Keterangan
id_kategori	int	Tidak	*autoincrement*	PK
nama_ kategori	varchar (45)	Ya	*null*	nama kategori

7. Tabel pengadu

Tabel pengadu digunakan untuk menyimpan data seluruh warga kota Kediri. Tabel ini berfungsi untuk verifikasi apakah warga yang menulis aduan adalah warga Kediri atau bukan. Dimana setiap baris pada tabel mewakili satu pengadu. Atribut pada tabel ini dijelaskan pada Tabel A.7.

Tabel A.7 Tabel pengadu

Atribut	Tipe	*Null*	*Default*	Keterangan
nik	varchar (50)	Tidak	-	PK
nama_lgkp	varchar (50)	Ya	*null*	nama pengadu
jenis_klmn	tinyint (4)	Ya	*null*	jenis kelamin pengadu
tmpt_lhr	varchar (50)	Ya	*null*	tempat lahir pengadu
tgl_lhr	date	Ya	*null*	tanggal lahir pengadu
pddk_akh	int	Ya	*null*	pendidikan terakhir
jenis_pkrjn	int	Ya	*null*	pekerjaan pengadu
no_prop	int	Ya	*null*	propinsi pengadu

Atribut	Tipe	*Null*	*Default*	Keterangan
no_kab	int	Ya	*null*	kabupaten pengadu
no_kec	int	Ya	*null*	kecamatan pengadu
no_kel	int	Ya	*null*	kelurahan pengadu

8. Tabel status

Tabel status digunakan untuk menyimpan data status aduan yang ditulis warga (apakah aduan sudah diterima departemen atau sudah selesai ditanggapi). Dimana setiap baris pada tabel mewakili satu jenis status aduan. Atribut pada tabel ini dijelaskan pada Tabel A.8.

Tabel A.8 Tabel status

Atribut	Tipe	*Null*	*Default*	Keterangan
id_status	int	Tidak	*autoincrement*	PK
nama_status	varchar (45)	Ya	*null*	nama status
class_status	varchar (45)	Ya	*null*	kelas status

9. Tabel status_aduan

Tabel status_aduan digunakan untuk menyimpan data *log* jika terjadi perubahan status aduan di tabel aduan. Dimana setiap baris pada tabel mewakili satu *log* dari suatu aduan. Atribut pada tabel ini dijelaskan pada Tabel A.9

Tabel A.9 Tabel aduan

Atribut	Tipe	*Null*	*Default*	Keterangan
id_status_ aduan	int	Tidak	*autoincrement*	PK
waktu_ status_ aduan	varchar (45)	Ya	*null*	Waktu *log* aduan

10. Tabel prioritas

Tabel prioritas digunakan untuk menyimpan data tingkat prioritas aduan yang ditulis warga. Dimana setiap baris pada tabel mewakili satu tingkat prioritas aduan. Atribut pada tabel ini dijelaskan pada Tabel A.10.

Tabel A.10 Tabel prioritas

Atribut	Tipe	*Null*	*Default*	Keterangan
id_prioritas	int	Tidak	*autoincrement*	PK
nama_ prioritas	varchar (45)	Ya	*null*	nama tingkatan prioritas
class_ prioritas	varchar (45)	Ya	*null*	kelas prioritas

11. Tabel kelurahan

Tabel kelurahan digunakan untuk menyimpan data kelurahan yang ada di Kota Kediri. Dimana setiap baris pada tabel mewakili satu kelurahan. Atribut pada tabel ini dijelaskan pada Tabel A.11.

Tabel A.11 Tabel kelurahan

Atribut	Tipe	*Null*	*Default*	Keterangan
id_ kelurahan	int	Tidak	*autoincrement*	PK
nama_ kelurahan	varchar (45)	Ya	*null*	nama kelurahan

12. Tabel kecamatan

Tabel kecamatan digunakan untuk menyimpan data kecamatan yang ada di Kota Kediri. Dimana setiap baris pada tabel mewakili satu kecamatan. Atribut pada tabel ini dijelaskan pada tabel Tabel A.12

Tabel A.12 Tabel kecamatan

Atribut	Tipe	*Null*	*Default*	Keterangan
id_ kecamatan	int	Tidak	*autoincrement*	PK
nama_ kecamatan	varchar (45)	Ya	*null*	nama kecamatan

13. Tabel *upload*

Tabel *upload* digunakan untuk menyimpan data berupa *path* dari *file* yang disimpan di *server*. *File* yang dimaksud adalah lampiran yang diunggah warga ketika mengisi aduan melalui *website*. Dimana setiap baris pada tabel mewakili satu *path* dari *file* lampiran. Atribut pada tabel ini dijelaskan pada Tabel A.13.

Tabel A.13 Tabel *upload*

Atribut	Tipe	*Null*	*Default*	Keterangan
id_upload	varchar (128)	Tidak	-	PK
orig_name	varchar (256)	Ya	*null*	nama asli dari *file*
path_upload	varchar (256)	Ya	*null*	*path* dari *file*
file_type	varchar (256)	Ya	*null*	tipe *file*

14. Tabel sms_tidak_valid

Tabel sms_tidak_valid digunakan untuk menyimpan data pesan singkat yang masuk namun tidak valid. Kriteria tidak valid yang dimaksud adalah ketika pesan singkat tidak sesuai dengan format penulisan yang sudah ditentukan pemerintah Kota Kediri. Dimana setiap baris pada tabel mewakili satu pesan singkat yang tidak valid. Atribut pada tabel ini dijelaskan pada tabel Tabel A.14

Tabel A.14 Tabel sms_tidak_valid

Atribut	Tipe	*Null*	*Default*	Keterangan
id	int	Tidak	-	PK
nomor_ pengirim	varchar (20)	Ya	-	nomor pengirim pesan singkat
urutan	varchar (20)	Ya	-	urutan pesan singkat

Atribut	Tipe	*Null*	*Default*	Keterangan
waktu	timestamp	Ya	*current timestamp*	waktu pesan singkat diterima
isi	text		-	isi pesan singkat

15. Tabel all_app

Tabel all_app digunakan untuk menyimpan *url* dari aplikasi *web* suara warga. Dimana setiap baris pada tabel mewakili satu buah *url*. Atribut pada tabel ini dijelaskan pada Tabel A.15.

Tabel A.15 Tabel all_app

Atribut	Tipe	*Null*	*Default*	Keterangan
id_all_app	int	Tidak	*autoincrement*	PK
url_app	varchar (256)	Ya	*null*	nama kategori
desc_app	varchar (45)	Ya	*null*	Deskripsi aplikasi

16. Tabel user_app

Tabel user_app digunakan untuk menyimpan *log* dari petugas yang mengakses aplikasi *web* suara warga. Dimana setiap baris pada tabel mewakili satu *user*. Atribut pada tabel ini dijelaskan pada tabel Tabel A.16.

Tabel A.16 Tabel user_app

Atribut	Tipe	*Null*	*Default*	Keterangan
id_all_app	int	Tidak	*autoincrement*	PK
id_app	int	Tidak	-	*url web*
petugas	int	Tidak	-	Petugas yang mengakses *url*