

**PENGEMBANGAN SISTEM PAKAR BERBASIS ANDROID UNTUK
MENDIAGNOSA KERUSAKAN PADA PERANGKAT KOMPUTER**

TUGAS AKHIR SKRIPSI

Diajukan Kepada Fakultas Teknik Universitas Negeri Yogyakarta

Untuk Memenuhi Sebagian Persyaratan

Guna Memperoleh Gelar Sarjana Pendidikan



Disusun Oleh:

Nafis Akhsan

NIM. 09520241027

PROGRAM STUDI PENDIDIKAN TEKNIK INFORMATIKA

JURUSAN PENDIDIKAN TEKNIK ELEKTRONIKA

FAKULTAS TEKNIK

UNIVERSITAS NEGERI YOGYAKARTA

2016

LEMBAR PERSETUJUAN

Tugas Akhir Skripsi dengan Judul

PENGEMBANGAN SISTEM PAKAR BERBASIS ANDROID UNTUK MENDIAGNOSA KERUSAKKAN PADA PERANGKAT KOMPUTER

Disusun Oleh:

Nafis Akhsan

NIM. 09520241027

Telah Memenuhi Syarat dan disetujui oleh Dosen Pembimbing untuk dilaksanakan Ujian Akhir Tugas Akhir Skripsi bagi yang bersangkutan.

Yogyakarta, 17/03/2016

Mengetahui,

Ketua Program Studi Pendidikan Teknik

Informatika



Handaru Jati, Ph.D

NIP. 19740511 199903 1 002

Disetujui,

Pembimbing Skripsi



Prof. Herman Dwi Surjono, Ph.D

NIP. 19640205 198703 1 001

LEMBAR PERNYATAAN

Saya yang bertanda tangan dibawah ini :

Nama : Nafis Akhsan
NIM : 09520241027
Program Studi : Pendidikan Teknik Informatika
Judul TAS : *"Pengembangan Sistem Pakar Berbasis Android Untuk Mendiagnosa Kerusakan Pada Perangkat Komputer"*

Menyatakan bahawa skripsi ini merupakan karya sendiri. Sepanjang sepengetahuan saya tidak ada karya atau pendapat yang ditulis atau diterbitkan orang lain sebagai acuan atau kutipan dengan mengikuti pada tata tulis yang telah lazim.

Yogyakarta, 17 maret 2016

Yang Menyatakan,



Nafis Akhsan

NIM. 09520241027

HALAMAN PENGESAHAN

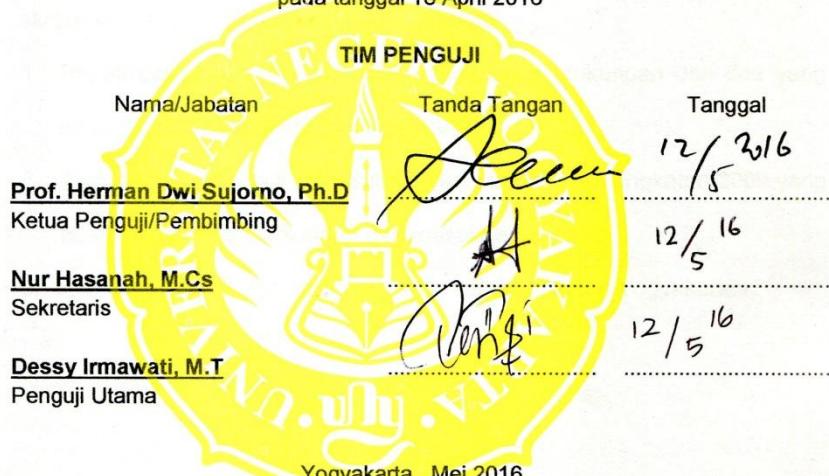
Tugas Akhir Skripsi

PENGEMBANGAN SISTEM PAKAR BERBASIS ANDROID UNTUK MENDIAGNOSA KERUSAKAN PADA PERANGKAT KOMPUTER

Disusun oleh:

Nafis Akhsan
NIM. 09520241027'

Telah dipertahankan di depan Tim Pengaji Tugas Akhir Skripsi Program Studi
Pendidikan Teknik Informatika Fakultas Teknik Universitas Negeri Yogyakarta
pada tanggal 18 April 2016



Fakultas Teknik Universitas Negeri Yogyakarta

Dekan,



MOTTO DAN PERSEMBAHAN

A. MOTTO

“Sesungguhnya setelah kesulitan akan datang kemudahan”

B. PERSEMBAHAN

Segala puji bagi Allah SWT, Rabb semesta alam yang senantiasa memberikan karunia sehingga penulis mampu menyelesaikan penulisan skripsi ini. Karya ini saya persembahkan untuk:

1. Bapak dan Ibu saya yang telah memberikan dukungan dan doa yang tiada henti demi terselesaiannya karya ini.
2. Teman-teman kelas E pendidikan teknik informatika angkatan 2009 yang telah memberikan dukungan dan masukan.
3. Almamater saya, Fakultas Teknik Universitas Negeri Yogyakarta.

PENGEMBANGAN SISTEM PAKAR BERBASIS ANDROID UNTUK MENDIAGNOSA KERUSAKKAN PADA PERANGKAT KOMPUTER

Oleh:

Nafis Akhsan

NIM. 09520241027

ABSTRAK

Perbaikan kerusakan pada perangkat komputer dapat menjadi sesuatu yang sulit bila tidak diketahui penyebab masalah secara pasti. Oleh sebab itu perlu dilakukan diagnosa penyebab kerusakan terlebih dahulu sebelum melakukan perbaikan. Diagnosa penyebab kerusakan dapat dilakukan dengan menggunakan bantuan sebuah aplikasi sistem pakar. Dengan menggunakan aplikasi sistem pakar seseorang dapat dengan mudah menganalisa penyebab kerusakan seperti layaknya ketika sedang melakukan konsultasi dengan seorang ahli. Selain itu dengan aplikasi sistem pakar seseorang juga dapat belajar untuk mengetahui pola-pola atau alur diagnosa suatu permasalahan pada computer. Oleh sebab itu penelitian ini bertujuan untuk mengembangkan sebuah aplikasi sistem pakar untuk melakukan diagnosa guna memperbaiki kerusakan pada komputer atau PC.

Jenis penelitian yang dilakukan adalah *Research and Development* (R&D). Proses R&D menggunakan metode sekuensial linier atau *waterfall* yang meliputi tahap analisis kebutuhan, desain, implementasi, dan pengetesan. Aplikasi sistem pakar dibangun untuk bekerja pada sebuah perangkat *smartphone android*. Guna menghasilkan sebuah aplikasi yang baik, penelitian ini menggunakan standar ISO 9126 sebagai standar tingkat kelayakan aplikasi. Standar ISO 9126 yang digunakan meliputi aspek *functionality, reliability, maintainability, portability, dan usability*.

Berdasarkan prosedur pengetesan yang dilakukan, diperoleh hasil: 1) perancangan aplikasi telah melalui serangkaian proses pengembangan perangkat lunak. 2) hasil pengetesan diperoleh hasil aspek *functionality* telah terpenuhi dengan indikator semua funsionalitas yang direncanakan dapat disajikan pada hasil akhir aplikasi dan berjalan dengan baik, aspek *reliability* telah terpenuhi dengan indikator besar nilai *defect density* berada atau lebih baik dari standar yang ada, aspek *portability* dihasilkan bahwa aplikasi dapat dipasang dan dijalankan pada setiap perangkat yang ada saat pengetesan dilakukan, aspek *maintainability* menunjukkan hasil yang baik dengan setiap sub-kriteria yang ada berada pada tingkat yang baik. aspek *usability* memperoleh hasil yang baik dengan presentase *usability* sebesar 76,64%.

Kata kunci : Aplikasi Sistem Pakar, Aplikasi Android, Komputer, *Research And Development*, ISO 9126

KATA PENGANTAR

Puji Syukur keharidat Allah SWT, yang telah melimpahkan rahmat serta hidayah-Nya sehingga penulis dapat menyelesaikan skripsi sebagai salah satu persyaratan dalam menyelesaikan program S1 program studi Pendidikan Teknik Informatika Universitas Negeri Yogyakarta.

Penelitian ini memberikan banyak pelajaran – pelajaran mengenai apa yang menjadi fokus materi yang penulis kembangkan yaitu **Pengembangan Sistem Pakar Berbasis Android Untuk Mendiagnosa Kerusakan Pada Perangkat Komputer**. Selama melaksanakan penelitian ini, penulis banyak mendapatkan bimbingan, arahan serta dukungan dari berbagai pihak, untuk itu penulis ingin mengucapkan terima kasih kepada:

1. Bapak Prof. Dr. Rochmat Wahab, M.Pd. M.A. selaku Rektor Universitas Negeri Yogyakarta.
2. Bapak Dr. Moch Bruri Triyono selaku Dekan Fakultas Teknik Universitas Negeri Yogyakarta.
3. Bapak Dr. Fatchul Afirin M.T selaku Ketua Jurusan Pendidikan Teknik Elektronika.
4. Bapak Handaru Jati, Ph.D selaku Ketua Program Studi Pendidikan Teknik Informatika.
5. Bapak Prof. Herman Dwi Surjono Ph.D selaku Dosen Pembimbing Tugas Akhir Skripsi.

6. Rekan-rekan akademik kelas E Pendidikan Teknik Informatika UNY 2009 yang telah banyak membantu, mendukung, dan memberikan semangat untuk menyelesaikan Tugas Akhir Skripsi ini.
7. Siswa-siswi SMK N 2 Depok, Sleman Khususnya kelas X Teknik Komputer dan Jaringan.
8. Semua pihak yang telah membantu Tugas Akhir Skripsi ini.

Penulis menyadari bahwa dalam penyusunan laporan ini masih terdapat kekurangan. Saran dan kritik yang bersifat membangun sangat penulis harapkan. Semoga Tugas Akhir Skripsi ini dapat memberikan manfaat bagi semua pihak.

Yogyakarta, 17 maret 2016

Penulis

DAFTAR ISI

| | Halaman |
|--|---------|
| Halaman Judul | i |
| Lembar Persetujuan | ii |
| Lembar Pernyataan | iii |
| Halaman Pengesahan | iv |
| Motto dan Persembahan | v |
| Abstrak | vi |
| Kata Pengantar | vii |
| Daftar Isi | ix |
| Daftar Tabel | xii |
| Daftar Gambar | xiii |
| Daftar Lampiran | xv |
| BAB I PENDAHULUAN | 1 |
| A. Latar Belakang | 1 |
| B. Identifikasi Masalah | 6 |
| C. Pembatasan Masalah | 6 |
| D. Rumusan Masalah | 7 |
| E. Tujuan Penelitian | 7 |
| F. Manfaat | 8 |
| BAB II KAJIAN PUSTAKA | 9 |
| A. Sistem Pakar | 9 |
| 1. Pengertian Sistem Pakar | 9 |
| 2. Konsep Dasar Sistem Pakar | 10 |
| 3. Struktur Sistem Pakar | 12 |
| 4. Pengetahuan | 17 |
| 5. Model Repesentasi Pengetahuan | 19 |
| 6. Penalaran | 29 |
| 7. Perunutan | 31 |

| | |
|--|----|
| B. Komputer | 32 |
| 1. Cara Kerja Komputer | 32 |
| 2. Klasifikasi Komputer | 33 |
| 3. Perangkat Keras Pada Komputer | 35 |
| 4. Proses Diagnosa Pada PC | 37 |
| C. Android | 37 |
| D. Kualitas Perangkat Lunak | 39 |
| E. Model Pengembangan Sekuensial Linier | 49 |
| F. Penelitian Yang Relevan | 51 |
| BAB III METODOLOGI PENELITIAN | 53 |
| A. Jenis Penelitian | 53 |
| B. Langkah-Langkah Penelitian | 54 |
| C. Waktu dan Tempat Penelitian | 55 |
| D. Definisi Variabel | 55 |
| E. Desain Penelitian | 56 |
| F. Teknik Pengumpulan Data | 57 |
| G. Subjek Penelitian | 58 |
| H. Instrumen Penelitian | 59 |
| I. Teknik Analisis data | 60 |
| BAB IV HASIL DAN PEMBAHASAN | 62 |
| A. Hasil Penelitian | 62 |
| 1. Tahap Analisis Kebutuhan | 62 |
| a. Analisis Kebutuhan Fungsi | 62 |
| b. Analisis Kebutuhan Hardware dan Software | 63 |
| 2. Desain Perangkat Lunak | 64 |
| a. Desain UML (<i>Unified Modeling Language</i>) | 64 |
| b. Desain <i>User Interface</i> | 69 |
| c. Desain Database | 78 |
| 3. Implementasi | 79 |
| a. Halaman Awal | 79 |
| b. Halaman Info bantuan | 80 |

| | | |
|-----------------------------------|---|-----|
| c. | Halaman pengaturan | 82 |
| d. | Halaman tambah data | 84 |
| e. | Halaman Ubah Data | 85 |
| f. | Halaman Detail Data | 86 |
| g. | Halaman Hapus Data | 87 |
| h. | Halaman Buat Aturan | 88 |
| i. | Halaman Ubah Aturan | 89 |
| j. | Halaman Detail Aturan | 90 |
| k. | Halaman hapus Aturan | 91 |
| I. | Halaman Konsultasi | 92 |
| 4. | Pengetesan Pada Aplikasi Sistem pakar | 95 |
| a. | Pengetesan Functionality | 95 |
| b. | Pengetesan Reliability | 98 |
| c. | Pengetesan Maintanibility | 101 |
| d. | Pengetesan Portability | 105 |
| e. | Pengetesan Usability | 108 |
| B. | Pembahasan | 108 |
| 1. | Rangkuman Penelitian | 108 |
| 2. | Keterbatasan Penelitian | 114 |
| BAB V KESIMPULAN DAN SARAN | | 116 |
| A. | Kesimpulan | 116 |
| B. | Saran | 117 |
| DAFTAR PUSTAKA | | 118 |
| LAMPIRAN | | 121 |

DAFTAR TABEL

| | Halaman |
|---|---------|
| Tabel 1. Hubungan antara pengguna dan fungsi sistem pakar | 16 |
| Tabel 2. Bingkai Sepeda Motor | 21 |
| Tabel 3. Bingkai Sepeda Motor Honda..... | 21 |
| Tabel 4. Struktur kaidah produksi | 23 |
| Tabel 5. Contoh Format Tabel Keputusan | 24 |
| Tabel 6. Tabel Keputusan Identifikasi Pesawat | 25 |
| Tabel 7. Contoh kaidah produksi | 29 |
| Tabel 8. Metode inferensi | 30 |
| Tabel 9. Perangkat keras PC | 36 |
| Tabel 10. <i>Android Version History</i> | 37 |
| Tabel 11. Beberapa jenis layar yang didukung oleh Android | 39 |
| Tabel 12. Perkiraan Jumlah <i>Error McConnell</i> | 45 |
| Tabel 13. Kisi-kisi instrument <i>usability</i> | 59 |
| Tabel 14. Interpretasi Presentase | 60 |
| Tabel 15. Penyesuaian Interpretasi Presentase | 61 |
| Table 16. Standar Defect Density | 61 |
| Tabel 17. Desain database pada SQLite | 79 |
| Tabel 18. Hasil pengetesan fungsi-fungsi | 95 |
| Tabel 19. Perbandingan hasil perhitungan jumlah defect density pada aplikasi dengan standar yang ada untuk <i>Industry Average</i> dan <i>Microsoft Application</i> | 100 |
| Tabel 20. Kriteria jumlah besaran KLOC | 101 |
| Tabel 21. Kategori standar <i>cyclomatic complexity</i> | 102 |
| Tabel 22. Kategori penilaian duplikasi kode | 104 |
| Table 23. Penilaian ketercakupan <i>unit testing</i> | 105 |
| Tabel 24. Pengetesan manual aplikasi sistem pakar | 106 |
| Tabel 25. Pengetesan secara <i>cloud testing</i> dengan testdroid | 106 |
| Table 26. Perhitungan hasil pengetesan <i>portability</i> | 107 |

DAFTAR GAMBAR

Halaman

| | |
|--|----|
| Gambar 1. Pangsa pasar perangkat mobile dari desember 2011 sampai september 2014 di Indonesia | 4 |
| Gambar 2. 10 Kategori teratas aplikasi android | 5 |
| Gambar 3. Struktur sistem pakar | 13 |
| Gambar 4. Hirarki pengetahuan | 17 |
| Gambar 5. Contoh jaringan semantic | 20 |
| Gambar 6. Pohon keputusan identifikasi pesawat | 26 |
| Gambar 7. Proses reduksi atribut | 27 |
| Gambar 8. Pohon keputusan hasil reduksi | 28 |
| Gambar 9. Persentase pengguna versi android | 38 |
| Gambar 10. Model kualitas perangkat lunak menurut McCall | 40 |
| Gambar 11. <i>Maintainability metric</i> | 48 |
| Gambar 12. Model sekuensial linier | 49 |
| Gambar 13. Model pengembangan perangkat lunak sekuensial linier sebagai tahapan untuk model penelitian dan pengembangan..... | 53 |
| Gambar 14. Metrik <i>maintainability</i> | 57 |
| Gambar 15. Diagram use case sistem pakar perbaikan komputer berbasis android | 65 |
| Gambar 16. Diagram aktivitas melakukan konsultasi | 66 |
| Gambar 17. Diagram sekuensial melakukan konsultasi | 68 |
| Gambar 18. Diagram kelas untuk melakukan konsultasi | 69 |
| Gambar 19. Desain <i>user interface</i> untuk halaman utama | 70 |
| Gambar 20. Desain <i>user interface</i> list menu dan item | 71 |
| Gambar 21. Desain <i>user interface</i> halaman konsultasi | 72 |
| Gambar 22. Desain halaman info bantuan | 73 |
| Gambar 23. Desain <i>user interface</i> halaman tambah dan ubah data | 74 |
| Gambar 24. Desain <i>user interface</i> hapus data (a) dan detail data (b) | 75 |
| Gambar 25. Desain <i>user interface</i> halaman buat node aturan (a) dan ubah | |

| | |
|---|-----|
| node aturan (b) | 76 |
| Gambar 26. Desain <i>user interface</i> halaman hapus node aturan | 77 |
| Gambar 27. Desain <i>user interface</i> detail node aturan | 78 |
| Gambar 28. <i>Splash Screen</i> (a) dan halaman utama (b) | 80 |
| Gambar 29. List menu halaman bantuan | 81 |
| Gambar 30. Halaman tentang aplikasi (a) dan petunjuk pemakaian (b) | 82 |
| Gambar 31. Halaman pengaturan [a], halaman opsi basis pengetahuan [b], halaman opsi basis aturan [c] | 83 |
| Gambar 32. List menu proses diagnose yang disediakan | 84 |
| Gambar 33. Halaman tambah data | 85 |
| Gambar 34. list data yang telah dibuat [a] dan halaman ubah data [b] | 86 |
| Gambar 35. halaman detail data | 87 |
| Gambar 36. opsi penghapusan [a] dan hapus 1 buah data [b] | 88 |
| Gambar 37. Halaman buat aturan | 89 |
| Gambar 38. List hubungan node yang ada [a] dan halaman untuk mengubah aturan [b] | 90 |
| Gambar 39. Detail node aturan yang terhubung | 91 |
| Gambar 40. Halaman hapus node-node yang terhubung pada aturan | 92 |
| Gambar 41. Halaman konsultasi dalam menampilkan pertanyaan diagnosa | 93 |
| Gambar 42. Halaman konsultasi menampilkan penjelasan terkait data pertanyaan/hasil diagnosa yang ditampilkan | 94 |
| Gambar 43. Halaman konsultasi dalam menampilkan hasil diagnose | 94 |
| Gambar 44. Hasil perhitungan banyaknya <i>lines of code</i> pada plugin Metric | 98 |
| Gambar 45. Persiapan pencarian error aplikasi sistem pakar pada program findbugs | 99 |
| Gambar 46. Hasil pencarian error aplikasi sistem pakar pada program findbugs | 99 |
| Gambar 47. Hasil perhitungan <i>cyclomatic complexity</i> | 102 |
| Gambar 48. Hasil perhitungan duplikasi kode | 103 |
| Gambar 49. Hasil perhitungan besarnya LOC dari method-method yang ada | 104 |

DAFTAR LAMPIRAN

| | Halaman |
|---|---------|
| Lampiran Definisi Use Case | 122 |
| Lampiran Desain Diagram Aktifitas | 135 |
| Lampiran Desain Diagram Sekuensial | 147 |
| Lampiran Desain Diagram Kelas..... | 154 |
| Lampiran Pengujian Portability | 157 |
| Lampiran Pengujian Usability | 164 |
| Lampiran Pengujian Functionality | 165 |
| Lampiran Unit Test | 174 |
| Lampiran Daftar Pohon Keputusan Atau <i>Rule Base</i> | 176 |
| Lampiran Surat-Surat | 186 |

BAB I

PENDAHULUAN

A. Latar Belakang

Komputer adalah perangkat yang saat ini tidak lagi menjadi barang yang mewah, namun sudah menjadi kebutuhan. Hampir disetiap aktifitas kita saat ini dipermudah dengan adanya komputer atau dikenal juga dengan sebutan *personal computer* (PC). Mulai dari aktivitas perkantoran, kesehatan, perbankan, pendidikan, dan lain-lain. Perkembangan komputer terus berlangsung sampai saat ini dengan kualitas dan inovasi yang semakin baik.

Kerusakan pada komputer merupakan sebuah masalah yang mengganggu. Terutama bagi seseorang yang pekerjaannya banyak bergantung kepada komputer. Banyak aktivitas yang akan tertunda akibat kerusakan tersebut.

Kerusakan komputer dapat diperbaiki bila diketahui dengan jelas penyebabnya. Penyebab kerusakan dapat diketahui berdasarkan gejala-gejala yang muncul atau ditemukan. Gejala-gejala tersebut umumnya dapat diamati. Proses pengamatan dilakukan lewat beberapa proses pengecekan. Meski demikian proses pengecekan yang digunakan untuk mengamati atau mengetahui sumber masalah tersebut tidaklah selalu sama.

Proses pengecekan sumber masalah untuk gangguan atau kerusakan komputer dapat berbeda-beda untuk setiap komponen atau hardware komputer. Proses pengecekan terkadang dilakukan dalam beberapa tahap. Namun biasanya pengecekan dimulai dari gejala yang

terlihat jelas. Berdasarkan pengecekan gejala awal tersebut, kemudian dapat menuntun proses diagnosa selanjutnya.

Proses perbaikan baru dapat dilakukan bila didapati gejala-gejala yang cukup untuk mengambil kesimpulan. Masalah untuk masing-masing hardware komputer memiliki gejala-gejala tersendiri. Hal tersebut membantu untuk melakukan pengecekan secara lebih spesifik. Meski demikian diagnosa kerusakan pada komputer tidaklah selalu menjadi hal yang mudah.

Diagnosa kerusakan sebuah sistem merupakan pekerjaan yang komplek. Kompleks karena tidak hanya diperlukan pengetahuan yang cukup, namun diperlukan juga pengalaman yang memadai.

“For many engineering systems, fault diagnosis is a very knowledge-intensive task. Expert troubleshooters emerge only after many years of experience in the operation and maintenance of the same system. Fault diagnosis in personal computers is no exception.” (Leng dan Teng, 1992:121).

Bagi teknisi pemula seperti pelajar SMK , maupun mahasiswa teknik informatika atau elektronika terkadang kesulitan melakukan diagnosa kerusakan. Sehingga kesalahan dan kelalaian dalam melakukan perbaikan pada kerusakan komputer dapat terjadi. Meski demikian hal tersebut masih dapat diperbaiki seiring dengan banyaknya pengalaman dan pembelajaran.

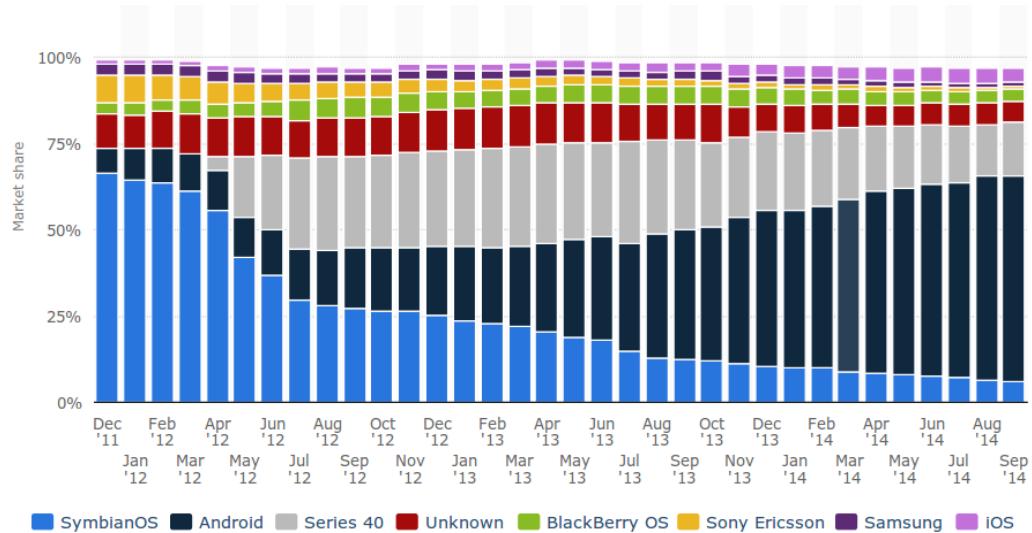
Berdasarkan pengalaman peneliti ketika mengampu mata pelajaran produktif praktik perakitan komputer ketika melakukan praktik pengajaran langsung (PPL) pada tahun 2012 di salah satu SMK. Peneliti melihat bahwa masih ada siswa yang tidak dapat mendiagnosa penyebab kerusakan pada unit komputer praktik yang tidak menyala.

Antara pelajar dan teknisi yang sudah ahli atau pakar terdapat rentang jarak pengalaman yang cukup lebar. Diperlukan suatu cara untuk mempersempit batas tersebut. Belajar dari pengalaman orang lain merupakan salah satu caranya. “...*Although experience may be a good teacher, someone else's experience is a far better teacher...*”(Golden, 2012). Dengan menggunakan pengalaman yang orang lain sebagai batu pijakan, diharapkan dapat mempercepat proses pembelajaran. Sebab dapat menghindari kesalahan yang sudah diperbuat orang lain.

Berdasarkan masalah tersebut dibutuhkan sebuah solusi, program seperti sistem pakar dapat menjadi salah satu cara yang efektif untuk belajar dari pengalaman orang lain. Sebab “sistem pakar adalah sistem yang menggunakan pengetahuan manusia yang terekam dalam komputer untuk memecahkan persoalan yang biasanya memerlukan keahlian manusia” (Turban dkk, 2005:708). Hal tersebut berarti bahwa sistem pakar dibuat agar orang bukan pakar dapat mengakses pengetahuan seorang pakar dalam menyelesaikan masalahnya. Sehingga dapat diperlajari bagaimana metode yang dilakukan seorang pakar. Sistem pakar memiliki beberapa fitur khusus yang memiliki kemungkinan untuk dapat mendukung proses tersebut. Sebagai contoh sistem pakar memiliki kemampuan penjelasan (*Explanation Capability*). *Explanation Capability* berfungsi memberi penjelasan kepada pengguna, bagaimana suatu kesimpulan dapat diambil (Sutojo dkk, 2011:169). Dengan demikian diharapkan pengguna sistem pakar dapat memperoleh wawasan yang lebih luas.

Disisi lain, perkembangan perangkat *mobile* di indonesia beberapa tahun terakhir berkembang dengan pesat, terutama perangkat *smartphone*

basis android. Dilihat dari sisi jumlah pengguna, *smartphone* berbasis android merupakan perangkat mobile yang paling diminati. Pada September 2014 (gambar 1), android menguasai setidaknya 60% pangsa pasar *smartphone* di indonesia.



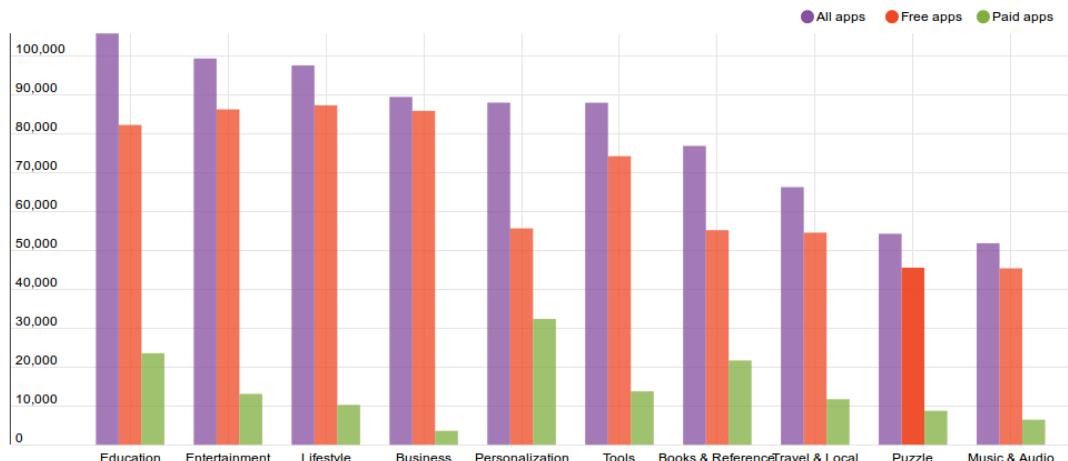
Gambar 1. Pangsa Pasar Perangkat Mobile Dari Desember 2011 Sampai September 2014 Di Indonesia (statista, 2014)

Android tidak hanya dijadikan sebagai alat komunikasi dan hiburan semata, namun juga untuk keperluan pendidikan. AppBrain(2014) menyatakan bahwa Pada Oktober 2014 setidaknya terdapat 105.677 aplikasi pendidikan baik itu gratis maupun berbayar di *Google Play*, sebagaimana dapat dilihat pada gambar 2. Hal ini dapat menunjukkan antusias pengembang perangkat lunak untuk mengembangkan aplikasi pendidikan cukup tinggi.

Most popular Google Play categories

The total number of apps, free apps and paid apps per Google Play category is shown here:

Top 10 Google Play categories



Gambar 2. 10 Kategori Teratas Aplikasi Android (AppBrain, 2014)

Melihat antusiasme masyarakat terhadap perangkat android, itu berarti terdapat lebih banyak peluang bagi sistem pakar dapat digunakan oleh banyak orang bila dikembangkan pada android. Terutama bagi pelajar dan calon tekniksi, sebagai target utama pengguna sistem pakar yang nanti akan dikembangkan. Selain itu android sebagai perangkat *mobile*, menawarkan keleluasaan bagi penggunanya. Bentuk yang tidak terlalu besar dan ringan, memudahkan android untuk dibawa kemana-mana.

Pengembangan sistem pakar pada android dewasa ini masih sedikit. Namun Setidaknya terdapat beberapa sistem pakar yang telah dikembangkan pada perangkat ini, salah satu contoh ICare (Singh, Suraj. Dkk, 2014). ICare merupakan sistem pakar yang berfokus dibidang kesehatan. Meski demikian aplikasi sistem pakar untuk diagnosa kerusakan komputer pada perangkat android belum ada.

Oleh karena itu melihat peluang yang ditawarkan oleh android dan kebutuhan akan sistem pakar untuk mendiagnosa kerusakan pada komputer. Maka penulis melakukan penelitian mengenai sistem pakar pada perangkat android dengan judul "*Pengembangan Sistem Pakar Berbasis Android Untuk Mendiagnosa Kerusakan Pada Perangkat Komputer*".

B. Identifikasi Masalah

Berdasarkan latar belakang yang telah disampaikan sebelumnya. Maka dapat diidentifikasi masalah yang dihadapi yaitu:

1. Kerusakan pada perangkat komputer dapat mengganggu rutinitas pekerjaan seseorang.
2. Kerusakan pada perangkat *hardware* komputer memiliki gejala-gelaja tersendiri.
3. Diagnosa permasalahan pada *hardware* komputer dapat menjadi hal yang sulit, terutama bagi seorang pelajar SMK atau mahasiswa yang masih dalam tahap belajar.
4. Pengalaman (*experience*) yang dibutuhkan untuk menjadi seorang yang ahli dibidangnya membutuhkan waktu yang lama.
5. *Smartphone* berbasis android merupakan perangkat mobile yang banyak digunakan di indonesia.
6. Belum adanya pengembangan aplikasi sistem pakar terkait diagnosa kerusakan komputer pada perangkat android.

C. Pembatasan Masalah

Dari masalah yang telah diuraikan, agar proses penelitian dan pembahasannya tidak terlalu luas. Maka masalah yang ada perlu dibatasi. Adapun batasan masalahnya adalah sebagai berikut:

1. Gejala-gejala yang belum diketahui akan mempersulit identifikasi kerusakan pada komputer.
2. Gejala-gejala kerusakan komputer sulit untuk diidentifikasi oleh orang yang masih awam atau pengalaman yang dibutuhkan masih kurang.
3. Belum adanya pengembangan aplikasi sistem pakar untuk perangkat berbasis android yang terkait dengan diagnosa kerusakan pada perangkat komputer..

D. Rumusan Masalah

Berdasarkan latar belakang, identifikasi dan batasan masalah yang ada. Maka dapat diperoleh rumusan masalah yang nantinya akan digunakan sebagai subjek penelitian. Adapun rumusan masalah tersebut yaitu:

1. Bagaimana mengembangkan sistem pakar untuk mendiagnosa kerusakan komputer pada perangkat *smartphone* dengan sistem operasi android ?
2. Bagaimana tingkat kelayakan sistem pakar untuk mendiagnosa kerusakan komputer pada perangkat *smartphone* dengan sistem operasi android?

E. Tujuan Penelitian

Tujuan yang diharapkan dapat dicapai dalam penelitian ini adalah sebagai berikut :

1. Menghasilkan sistem pakar untuk mendiagnosa kerusakan komputer pada perangkat *smartphone* dengan sistem operasi android.
2. Mengetahui tingkat kelayakan sistem pakar untuk mendiagnosa kerusakan komputer pada perangkat *smartphone* dengan sistem operasi android.

F. Manfaat

1. Manfaat Teoritis

Manfaat teoritis yang dapat diambil dari hasil penelitian ini antara lain:

- a. Dapat memperoleh tambahan wawasan pengetahuan tentang sistem pakar.
- b. Dapat mengetahui bagaimana cara mengukur kelayakan sebuah sistem pakar pada *smartphone* berbasis android.
- c. Dapat dijadikan sebagai bahan referensi bagi penelitian yang serupa.

2. Manfaat Praktis

Manfaat praktis yang dapat diambil dari hasil penelitian ini antara lain:

- a. Sistem pakar yang dihasilkan dapat berfungsi sebagai pemandu dan tutor bagi pelajar.
- b. Diagnosa permasalahan pada hardware komputer dapat lebih mudah dan cepat.

BAB II

KAJIAN PUSTAKA

A. Sistem Pakar

1. Pengertian Sistem Pakar

Kecerdasan buatan merupakan salah satu bidang ilmu komputer yang mendayagunakan kemampuan pada komputer supaya dapat berperilaku cerdas layaknya manusia. Kecerdasan buatan memiliki beberapa bidang-bidang lain yang dipelajari. Termasuk didalamnya yaitu sistem pakar, pengolahan bahasa alami, pengenalan ucapan, *computer vision*, robotika dan sistem sensor, sistem syaraf buatan, pengenalan pola, dan *game playing*.

Sistem pakar merupakan suatu sistem yang dirancang untuk dapat meniru keahlian seorang pakar manusia dapat memecahkan sebuah masalah. Beberapa definisi sistem pakar adalah sebagai berikut:

- a. "Sistem pakar adalah sistem yang menggunakan pengetahuan manusia yang terekam dalam komputer untuk memecahkan persoalan yang biasanya memerlukan keahlian manusia" (Turban dkk, 2005:708).
- b. "Sistem pakar merupakan bidang yang dicirikan oleh sistem berbasis pengetahuan (Knowledge Base System), memungkinkan komputer dapat berfikir dan mengambil kesimpulan dari sekumpulan kaidah" (Ignizio dalam hartati dan iswanti, 2008:3)
- c. "Sistem pakar adalah program yang berbasiskan pengetahuan yang menyediakan solusi 'kualitas pakar' kepada masalah-masalah dalam bidang (domain) yang spesifik" (Luger dan Stubblefield dalam Sutojo dkk, 2011:160).

2. Konsep Dasar Sistem Pakar

a. Kepakaran (*Expertise*)

Kepakaran merupakan suatu yang didapatkan melalui kegiatan pelatihan, membaca, serta pengalaman. Dengan kepakaran inilah para pakar memiliki kemampuan untuk mengambil keputusan lebih cepat, baik dan tepat. Kepakaran itu sendiri menurut Sutojo dkk (2011:163), meliputi pengetahuan tentang hal-hal berikut:

1. Fakta-fakta tentang bidang permasalahan tertentu.
2. Teori-teori tentang bidang permasalahan tertentu.
3. Aturan-aturan dan prosedur-prosedur menurut bidang permasalahan umumnya.
4. Aturan heuristic yang harus dikerjakan dalam suatu situasi tertentu.
5. Strategi global untuk memecahkan permasalahan.
6. Pengetahuan tentang pengetahuan (meta knowledge).

b. Pakar (*Expert*)

Menurut Turban dkk (2005:714), pakar adalah orang yang memiliki pengetahuan, penilaian, pengalaman, dan metode khusus, serta kemampuan untuk menerapkan bakat ini dalam memberi nasehat dan memecahkan persoalan. Merupakan tugas seorang pakar untuk menyediakan pengetahuan tentang bagaimana melaksanakan suatu tugas yang akan dijalankan oleh sistem berbasis-pengetahuan. Pakar mengetahui fakta mana yang penting dan memahami arti hubungan di antaranya. Seorang pakar juga harus mampu menjelaskan dan mempelajari hal-hal baru yang berkaitan dengan topik permasalahan. jika perlu pakar harus mampu menyusun kembali pengetahuan-pengetahuan yang didapatkan dan dapat memecahkan aturan-aturan serta menentukan relevansi kepakarannya.

Oleh karena itu menurut Hartanti dan Iswati (2008:11) seorang pakar memiliki kemampuan kepakaran, yaitu:

1. Dapat mengenali dan merumuskan suatu masalah.
2. Menyelesaikan masalah dengan cepat dan tepat.
3. Menjelaskan solusi dari suatu masalah
4. Restrukturisasi pengetahuan.
5. Belajar dari pengalaman.
6. Memahami batas kemampuan.

c. Pemindahan Kepakaran (Transferring Experise)

Sutojo dkk (2011:164) mengatakan, tujuan dari sistem pakar adalah memindahkan kepakaran dari seorang pakar ke dalam komputer, kemudian ditransfer kepada orang lain yang bukan pakar. Selanjutnya Sutojo dkk (2011:164) menambahkan bahwa proses ini melibatkan 4 kegiatan, yaitu:

1. Akusisi pengetahuan (dari pakar atau sumber lain)
2. Representasi pengetahuan (pada komputer)
3. Inferensi pengetahuan
4. Pemindahan pengetahuan ke pengguna

d. Inferensi (*Inferencing*)

“Inferensi adalah sebuah prosedur (program) yang mempunyai kemampuan dalam melakukan penalaran” (Sutojo dkk, 2011:164). Inferensi ditampilkan pada suatu komponen yang disebut mesin inferensi yang mencakup prosedur-prosedur mengenai pemecahan masalah. Semua pengetahuan yang dimiliki oleh seorang pakar disimpan pada basis pengetahuan oleh sistem pakar. Dengan demikian tugas mesin dari inferensi adalah mengambil kesimpulan berdasarkan basis pengetahuan yang dimilikinya.

e. Aturan-Aturan (*Rules*)

“Kebanyakan software sistem pakar komersial adalah sistem yang berbasis *rule*(*rule-based system*), yaitu pengetahuan disimpan terutama dalam bentuk rule, sebagai prosedur-prosedur pemecahan masalah” (Sutojo dkk, 2011:165). Prosedur-prosedur pemecahan masalah yang ada dalam basis pengetahuan disimpan dalam bentuk aturan. Dengan adanya aturan-aturan tersebut dapat mempermudah proses penalaran oleh sistem pakar. Sehingga hasil yang diperoleh pengguna dapat sesuai dengan yang diharapkan.

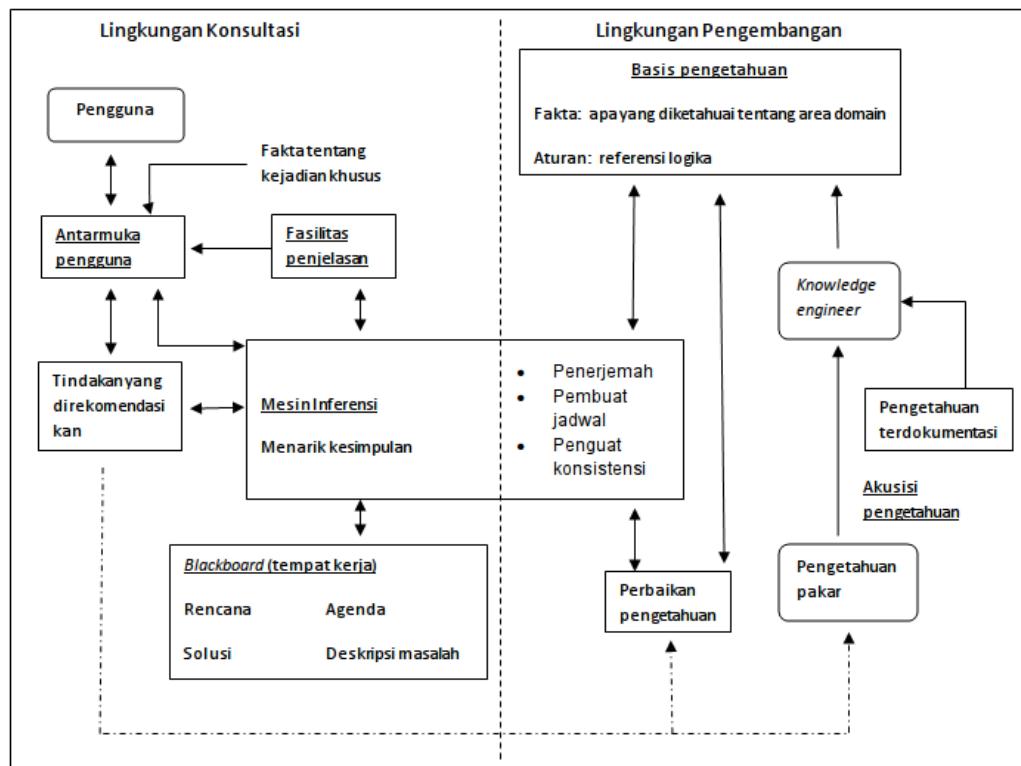
f. Kemampuan Menjelaskan (*Explanation Capability*)

Explanation Capability berfungsi memberi penjelasan kepada pengguna, bagaimana suatu kesimpulan dapat diambil (Sutojo dkk, 2011:169). Penjelasan dilakukan dalam subsistem yang disebut subsistem penjelasan. Bagian dari sistem ini memungkinkan sistem untuk memeriksa penalaran yang dibuatnya sendiri dan menjelaskan operasi-operasinya.

3. Struktur Sistem Pakar

“Ada dua bagian penting dari sistem pakar, yaitu lingkungan pengembangan (*development environment*) dan lingkungan konsultasi (*consultation environment*)” (Sutojo dkk, 2011:166). Lingkungan pengembangan digunakan oleh pengembang sistem pakar untuk membangun komponen dan memasukkan pengetahuan ke dalam basis pengetahuan. Lingkungan konsultasi digunakan oleh nonpakar untuk memperoleh pengetahuan dan nasihat pakar. lingkungan ini dapat

dipisahkan setelah sistem lengkap. Struktur sistem pakar dapat dilihat pada gambar 3.



Gambar 3. Struktur Sistem Pakar (Turban.dkk, 2005:722)

a. Subsistem Akusisi Pengetahuan

“Akusisi pengetahuan adalah akumulasi, transfer, dan transformasi keahlian pemecahan masalah dari pakar atau sumber pengetahuan terdokumentasi ke program komputer, guna membangun atau memperluas basis pengetahuan”(Turban.dkk, 2005:722). Sumber pengetahuan potensial antara lain pakar manusia, buku teks, dokumen multimedia, basis data, laporan riset khusus, dan informasi yang terdapat di web. Mendapatkan pengetahuan dari pakar adalah tugas komplek yang sering menimbulkan kemacetan dalam konstruksi sistem pakar.

Dalam pembangunan sistem yang besar seseorang memerlukan bantuan *knowledge engineer*. Menurut hartati dan Iswanti (2008:12), tugas utama seorang *knowledge engineer* adalah menterjemahkan dan merepresentasikan pengetahuan yang diperoleh dari pakar, baik berupa pengalaman pakar dalam menyelesaikan masalah atau sumber terdokumentasikan lain kedalam bentuk yang dapat diterima oleh sistem pakar.

b. Basis Pengetahuan

Basis pengetahuan berisi pengetahuan relevan yang diperlukan untuk memahami, merumuskan, dan memecahkan persoalan. Menurut Turban dkk (2005:723) basis pengetahuan mencakup dua elemen dasar yaitu fakta dan aturan. Fakta misalnya berisi tentang situasi persoalan dan teori area persoalan. Sedangkan Aturan bertujuan untuk mengarahkan penggunaan pengetahuan yang ada untuk memecahkan suatu masalah pada *domain* tertentu.

c. Mesin Inferensi

“Mesin inferensi adalah sebuah program yang berfungsi untuk memandu proses penalaran terhadap suatu kondisi berdasarkan pada basis pengetahuan yang ada” (Sutojo dkk, 2011 168). Kemudian memanipulasi dan mengarahkan kaidah, model, dan fakta yang disimpan dalam basis pengetahuan untuk mencapai solusi atau kesimpulan. Pada prosesnya, mesin inferensi menggunakan strategi pengendalian. Ada berapa teknik pengendalian yang umum digunakan yaitu infrensi runut maju (*forward chaining*) dan inferensi runut balik (*backward chaining*).

d. Antarmuka Pengguna

Antarmuka pengguna merupakan media komunikasi antara pengguna dan sistem pakar. Guna mempermudah komunikasi yang ada, antarmuka pada sistem pakar disajikan menggunakan bahasa alami (*natural language*). Namun karena keterbatasan teknologi, acap kali sistem pakar yang ada menggunakan pendekatan pertanyaan dan jawaban untuk berinteraksi dengan pengguna (Turban dkk, 2005:723).

e. Blackboard (Tempat Kerja)

“*Blackboard* adalah area kerja memori yang disimpan sebagai basis data untuk deskripsi persoalan terbaru yang ditetapkan oleh data input, digunakan juga untuk merekan hipotesis dan keputusan sementara” (Turban dkk, 2011:723). Disamping itu Sutujo dkk (2011:168) menambahkan terdapat tiga tipe keputusan yang dapat direkam pada *blackboard*, yaitu:

1. Rencana : bagaimana menghadapi masalah.
2. Agenda : aksi-aksi potensial yang sedang menunggu untuk dieksekusi.
3. Solusi : calon aksi yang akan dibangkitkan.

f. Subsistem Penjelasan (*Justifier*)

Kemampuan untuk melacak tanggung jawab suatu kesimpulan terhadap sumbernya adalah penting untuk transferse keahlian dan dalam pemecahan masalah. Dalam sistem pakar yang sederhana, penjelasan menunjukkan aturan yang digunakan untuk memperoleh rekomendasi tertentu. Menurut Turban dkk (2005:724), subssitem penjelasan dapat melacak tanggung jawab tersebut dan menjelaskan perilaku sistem pakar dengan menjawab pertanyaan berikut secara interaktif:

- a. Mengapa suatu pertanyaan ditanyakan oleh sistem pakar?
- b. Bagaimana suatu kesimpulan dicapai?
- c. Mengapa suatu alternatif ditolak?

- d. Apa rencana untuk mencapai solusi? Misalnya apa yang tetap tersisa sebelum diagnosis akhir diterapkan?

g. Sistem perbaikan pengetahuan

Kemampuan memperbaiki pengetahuan dari seorang pakar diperlukan untuk menganalisis pengetahuan, belajar dari pengalaman, kemudian memperbaikinya, sehingga perbaikan pengetahuan yang telah diperbaiki dapat dipakai untuk masa yang akan datang (Sutojo dkk, 2011:169). Kemampuan evaluasi yang ada diperlukan oleh program agar dapat menganalisis alasan-alasan kesuksesan dan kegagalan dalam mengambil kesimpulan. Dengan cara ini basis pengetahuan yang lebih baik dan penalaran yang lebih efektif akan dihasilkan.

h. Pengguna (User)

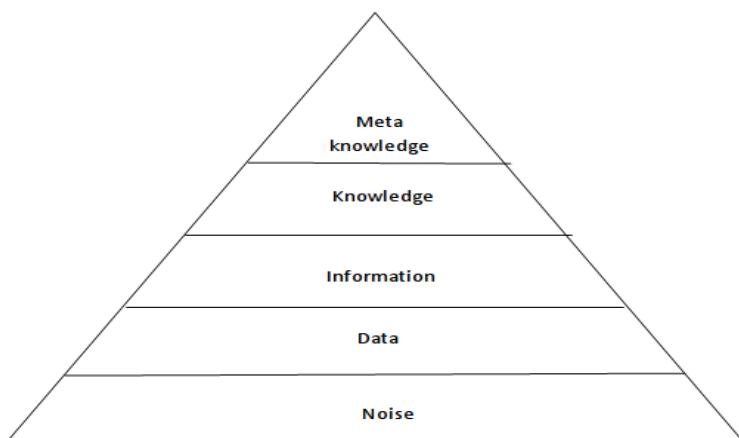
Menurut Sutojo dkk (2011:169), pada umumnya pengguna sistem pakar bukanlah seorang bukan pakar (non-pakar) yang membutuhkan solusi, saran, atau pelatihan dari berbagai permasalahan yang ada. Pengguna mungkin tidak terbiasanya dengan computer dan mungkin pada domain masalah. Pakar dan *knowledge engineer* harus mengantisipasi kebutuhan-kebutuhan pengguna dan membuat batasan-batasan ketika membuat system pakar. Hartati dan Iswanti (2008:13) menjelaskan hubungan antara pengguna dan fungsi sistem pakar sesuai seperti yang dapat dilihat pada tabel 1.

Tabel 1. Hubungan Antara Pengguna dan Fungsi Sistem Pakar

| Pengguna | Kepentingan | Fungsi sistem pakar |
|-------------------|----------------------------|----------------------------|
| Klien bukan pakar | Mencari saran atau nasehat | Konsultasi atau nasehat |
| Mahasiswa | Belajar | Instruktur |
| Pembangun sistem | Memperbaiki atau | Rekan (partner) |

| Pengguna | Kepentingan | Fungsi sistem pakar |
|----------|---|--------------------------|
| | menambah basis pengetahuan. | |
| Pakar | Membantu analisis rutin atau proses komputasi, mencari (mengklasifikasi) informasi, alat bantu diagnose | Rekan kerja atau asisten |

4. Pengetahuan



Gambar 4. Hirarki Pengetahuan (Hartati dan Iswanti, 2008:18).

Pengetahuan merupakan suatu hirarki seperti terlihat pada gambar 4. Menurut Lebih lanjut Hartati dan Iswanti (2008:20) mengatakan bahwa pengetahuan dapat digolongkan menjadi 3 kategori yaitu: pengetahuan deklaratif, pengetahuan prosedural, dan pengetahuan tacit (*tacit knowledge*). Berikut merupakan penjelasan mengenai 3 kategori pengetahuan tersebut:

a. Pengetahuan Deklaratif

“Pengetahuan deklaratif terkait dengan nilai kebenaran, apakah suatu itu bernilai benar atau salah”(Hartati & Iswanti, 2008:21). Sedangkan menurut

Turban dkk (2011:758), pengetahuan deklaratif adalah representasi deskriptif pengetahuan. Pengetahuan deklaratif mengacu pada fakta dan assersi serta diasosiasikan/dihubungkan dengan apa yang terlibat dalam pemecahan masalah .Pengetahuan ini dalam penyajiannya menggunakan basis logika dan pendekatan relasi. Representasi logika menggunakan logika proporsional dan logika predikat, sedangkan pendekatan relasi menggunakan model jaringan semantik, graphs dan pohon keputusan (*decision tree*).

b. Pengetahuan Prosedural

Menurut Hartati & Iswanti (2008:21) dalam bukunya menyatakan kategori pengetahuan prosedural mengacu pada serangkaian tindakan dan konsekuensinya,kemudian diasosiasikan dengan bagaimana menerapkan strategi atau prosedur penggunaan pengetahuan yang tepat untuk memecahkan masalah. Dalam pengetahuan prosedural algoritma digunakan sebagai prosedur pemecahan masalah. Pengetahuan procedural melibatkan respon otomatis untuk stimuli. Pengetahuan tersebut dapat pula menyatakan pada kita bagaimana menggunakan pengetahuan deklaratif dan bagaimana membuat inferensi.

c. Pengetahuan Tacit

Menurut Hartati dan Iswanti (2008:21) pengetahuan tacit disebut juga pengetahuan “tidak sadar” (*unconscious knowledge*), karena tidak dapat diekspresikan dengan bahasa. Istilah “tacit” sendiri mengandung arti dipahami tetapi tidak dapat dikatakan. Contoh pengetahuan tacit adalah bagaimana menggerakkan tangan. Pertanyaan ini dapat dijawab dengan cara mengencangkan atau mengendorkan otot dan tendon tertentu. Pertanyaan

akan berkembang, bagaimana untuk membuat otot dan tendon tertentu tersebut menjadi kencang atau kendur. Hal ini sulit diungkapkan dengan bahasa dalam format tertentu, tetapi menjadi mudah dipahami jika dilakukan atau ditunjukkan contohnya.

Pemrosesan yang dilakukan oleh sistem pakar merupakan pemrosesan pengetahuan. Bukan pemrosesan data seperti yang dikerjakan dengan pemrograman secara konvensional. "Pengetahuan adalah pemahaman secara praktis maupun teoritis terhadap suatu obyek atau domain tertentu" (Hartati dan Iswanti, 2008:18). Pemahaman terhadap bidang tertentu dapat diperoleh dari pelatihan, membaca, pendidikan, maupun dari pengalaman.

Pengetahuan yang digunakan pada sistem pakar merupakan serangkaian informasi mengenai gejala-diagnosa, sebab-akibat, aksi-reaksi tentang suatu bidang tertentu.

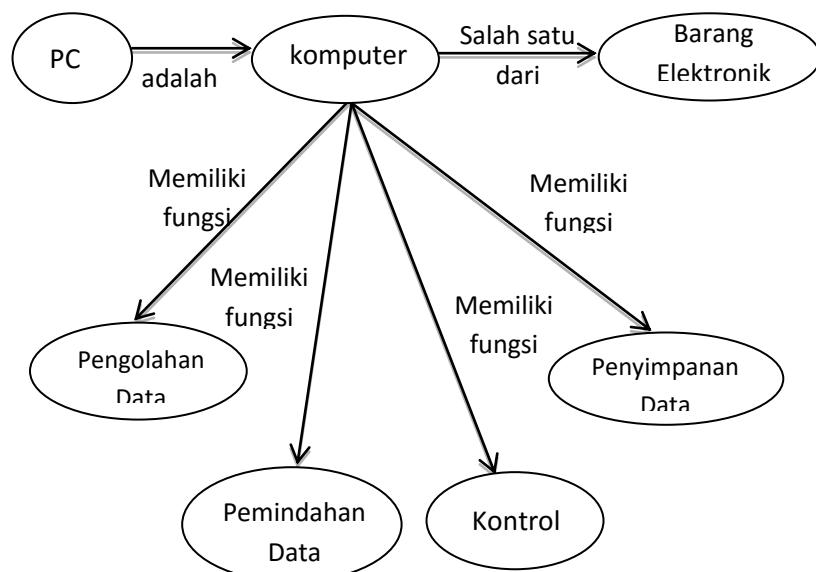
5. Model Representasi Pengetahuan

Representasi pengetahuan merupakan langkah yang penting dalam pembuatan sebuah sistem pakar. Sistem pakar merupakan program yang menggunakan pengetahuan seorang pakar untuk memecahkan sebuah permasalahan tertentu. Representasi pengetahuan dimaksudkan untuk mengorganisasikan pengetahuan dalam bentuk dan format tertentu untuk bisa dimengerti oleh komputer. Oleh karena itu perlu memilih model representasi pengetahuan yang tepat.

Menurut Hartati dan Iswanti (2008:22) menyatakan terdapat beberapa model pengetahuan yaitu jaringan semantik, bingkai, logika predikat, kaidah produksi.

a. Jaringan Semantik

“Jaringan semantik adalah teknik representasi pengetahuan yang digunakan untuk informasi proposisional, sedangkan yang dimaksud dengan informasi proporsional adalah pertanyaan yang mempunyai nilai benar atau salah”(Hartati dan iswanti, 2008:22-23). Komponen dasar untuk merepresentasikan pengetahuan dalam bentuk jaringan semantik adalah simpul (node) dan penghubung (link). Simpul merepresentasikan obyek dan hubungan antar obyek dinyatakan oleh penghubung yang diberi label untuk menyatakan hubungan yang direpresentasikan. Jaringan ini digunakan untuk informasi proporsional. Informasi proporsional merupakan bahasa deklaratif karena menyatakan fakta. Contoh jaringan semantik dapat dilihat pada Gambar 5.



Gambar 5. Contoh Jaringan Semantik (Hartati dan Iswanti, 2008:24)

b. Bingkai (*Frame*)

“*Frame* adalah struktur data yang menyertakan semua pengetahuan tentang objek tertentu”(Turban.dkk, 2005:792). *Frame* berupa kumpulan slot-slot yang berisi atribut untuk mendeskripsikan pengetahuan. Pengetahuan yang termuat dalam slot dapat berupa kejadian, lokasi, situasi, ataupun elemen-elemen lainnya.

Frame digunakan untuk representasi pengetahuan deklaratif. *Frame* memuat deskripsi sebuah objek dengan menggunakan tabulasi informasi yang berhubungan dengan obyek. Dengan kata lain frame mengelompokkan atribut sebuah obyek dan membantu menirukan cara seseorang mengorganisasi informasi tentang sebuah obyek menjadi kumpulan data. Representasi pengetahuan dengan bingkai ini sesuai untuk jenis pengetahuan yang memiliki subyek sempit, lebih bersifat pasti dan jarang berubah-ubah isinya kecuali terdapat kondisi khusus. Contoh representasi pengetahuan dengan bingkai terlihat pada tabel 2 dan tabel 3.

Tabel 2. Bingkai Sepeda Motor (Hartati dan Iswanti, 2008:24)

| Slots | Fillers |
|--------------|-----------------------------------|
| Nama | Sepeda motor |
| Spesialisasi | Jenis kendaraan beroda dua |
| Produk | Honda, Yamaha, Kawasaki, Daiheiyō |
| Bahan Bakar | Bensin |

Tabel 3. Bingkai Sepeda Motor (Honda Hartati dan Iswanti, 2008:25)

| Slots | Fillers |
|-------------------|-----------------------------|
| Nama | Honda |
| Spesialisasi dari | Produk sepeda motor |
| Type | Supra, Karisma, Vario, Revo |
| Buatan | Jepang |

c. Logika predikat

Hartati & Iswanti (2008:39) menyatakan pada tahun 1847 George Boole mengemukakan konsep logika simbolis yang mengenal aksioma yaitu symbol-simbol yang merepresentasikan obyek dan kelas serta operasi aljabar untuk memanipulasi simbol-simbol tersebut. Selanjutnya Hartati & Iswanti (2008:39) juga menyatakan logika proporsional adalah logika simbolis yang memanipulasi proposisi. Logika proporsional akan menangani kalimat deklaratif namun hanya mampu menangani pernyataan yang komplit dan tidak bisa menganalisa struktur internal sebuah pernyataan. Maka dikembangkan logika predikat untuk menganalisa kasus yang lebih umum dan dapat menganalisis struktur internal kalimat.

Bentuk paling sederhana dari logika predikat adalah logika derajat pertama (*first order logic*) yang terbentuk dengan menambahkan fungsi atau analisis lain pada kalkulus predikat. Logika predikat berdasarkan pada kebenaran dan kaidah inferensi untuk merepresentasikan simbol-simbol dan hubungannya satu dengan yang lain. Logika predikat selain digunakan untuk menentukan kebenaran (*truthfulness*) atau kesalahan (*falsity*) sebuah pernyataan juga dapat digunakan untuk merepresentasikan pernyataan tentang obyek tertentu.

Contoh logika proporsional : Bujur sangkar mempunyai empat sisi. Kalimat tersebut merupakan logika proporsional karena mengandung pernyataan yang mempunyai nilai kebenaran. Contoh logika predikat : Semua segitiga adalah poligon. Logika predikat menganalisa struktur internal kalimat tersebut, ditunjukkan dengan penggunaan kata “semua” yang merupakan quantifier.

d. Kaidah produksi

“Cara merepresentasikan pengetahuan berbasis kaidah produksi adalah dengan memanfaatkan apa yang disebut dengan kaidah. Kaidah yang tidak lain adalah pernyataan IF-THEN” (Hartati dan Iswanti, 2008:41). Bagian then akan bernilai benar jika satu atau lebih sekumpulan fakta atau hubungan antar fakta diketahui benar, memenuhi bagian IF. Secara umum dalam bentuk kaidah produksi IF premis THEN konklusi, maka premis yang lebih dari satu dapat dihubungkan dengan operator and atau or. Sedangkan bagian konklusi dapat berupa kalimat tunggal, beberapa kalimat yang dihubungkan and, dan dimungkinkan dikembangkan dengan else.

Menurut Adedeji dalam Hartati dan Iswanti (2008:25) berbagai struktur kaidah IF-THEN yang menghubungkan obyek atau atribut adalah seperti yang tercantum pada tabel 4.

Tabel 4. Struktur Kaidah Produksi

| No | Struktur Kaidah |
|----|--------------------------------|
| 1 | If masukan Then Keluaran |
| 2 | If kondisi Then tindakan |
| 3 | If antesenden Then konsekuensi |
| 4 | If data Then hasil |
| 5 | If tindakan Then tujuan |
| 6 | If aksi Then reaksi |
| 7 | If sebab Then akibat |
| 8 | If gejala Then diagnosa |
| 9 | If premis Then konklusi |

Sebelum sampai pada bentuk kaidah produksi , terdapat langkah-langkah yang harus ditempuh dari pengetahuan yang didapatkan dalam domain tertentu. Langkah-langkah tersebut adalah menyajikan pengetahuan yang berhasil didapatkan dalam bentuk tabel keputusan (*decision table*) kemudian dari tabel keputusan dibuat pohon keputusan (*decision tree*).

1) Tabel Keputusan dan Pohon Keputusan

Tabel keputusan merupakan suatu cara untuk mendokumentasikan pengetahuan. “Tabel keputusan merupakan matrik kondisi yang dipertimbangkan dalam pendeskripsi kaidah” (Hartati dan Iswanti, 2008:26). Tabel 5 menunjukkan contoh sebuah format tabel keputusan.

Tabel 5. Contoh Format Tabel Keputusan

| KONDISI | GOAL 1 | GOAL 2 |
|-----------|--------|--------|
| KONDISI 1 | ✓ | |
| KONDISI 2 | ✓ | ✓ |
| KONDISI 3 | | ✓ |

Kaidah yang disajikan dalam bentuk kaidah produksi disusun dari tabel keputusan (dibentuk dari pengubahan tabel keputusan). Pembuatan suatu kaidah dilakukan dengan beberapa tahapan. Sebagai contoh perhatikan pembuatan kaidah 1. Pertama, kita lihat Goal 1 merupakan konklusi dari kaidah 1. Konlusi ini akan dapat dicapai bila kondisi-kondisi yang mendukungnya terpenuhi. Kedua, tanda centang (✓) pada kolom dibawah Goal 1 menunjukan kondisi mana yang harus dipenuhi untuk mencapai konklusi tersebut. Pada Goal 1, terlihat tanda centang berada

pada Kondisi 1 dan 2. Ketiga, pembuatan kaidah 1 menggunakan goal dan kondisi yang telah diperoleh dari langkah 1 dan 2, seperti berikut ini:

| | |
|------------|-----------------------------------|
| Kaidah 1 : | Goal 1 IF Kondisi 1 AND Kondisi 2 |
|------------|-----------------------------------|

Kaidah 2 dapat diperoleh dengan cara yang sama :

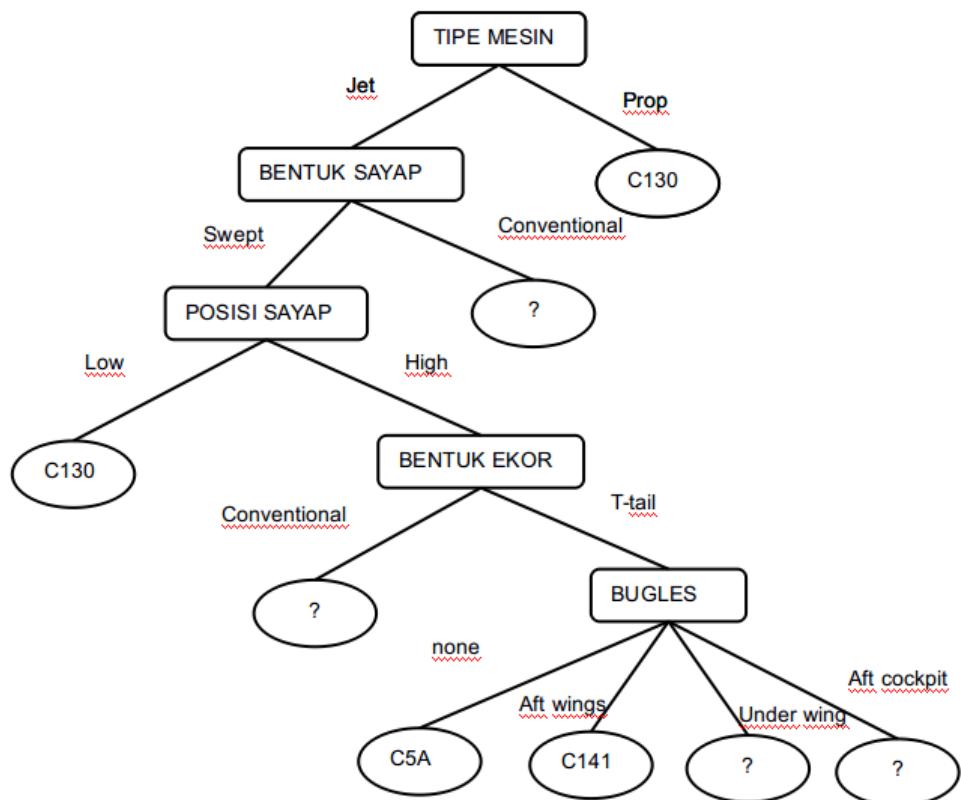
| | |
|------------|-----------------------------------|
| Kaidah 2 : | Goal 2 IF Kondisi 2 AND Kondisi 3 |
|------------|-----------------------------------|

Meskipun kaidah secara langsung dapat dihasilkan dari tabel keputusan. Namun untuk menghasilkan kaidah yang lebih efisien perlu membuat terlebih dahulu pohon keputusan. Dari pohon keputusan dapat diketahui atribut (kondisi) yang dapat yang dapat direduksi sehingga dapat dihasilkan kaidah yang efisien dan optimal. Sebagai contoh akan disampaikan penyajian pengetahuan untuk mengidentifikasi jenis pesawat terbang. Pengetahuan yang didapatkan disajikan dalam bentuk tabel keputusan seperti yang dapat dilihat pada tabel 6.

Tabel 6. Tabel Keputusan Identifikasi Pesawat (Hartati dan Iswanti, 2008:27)

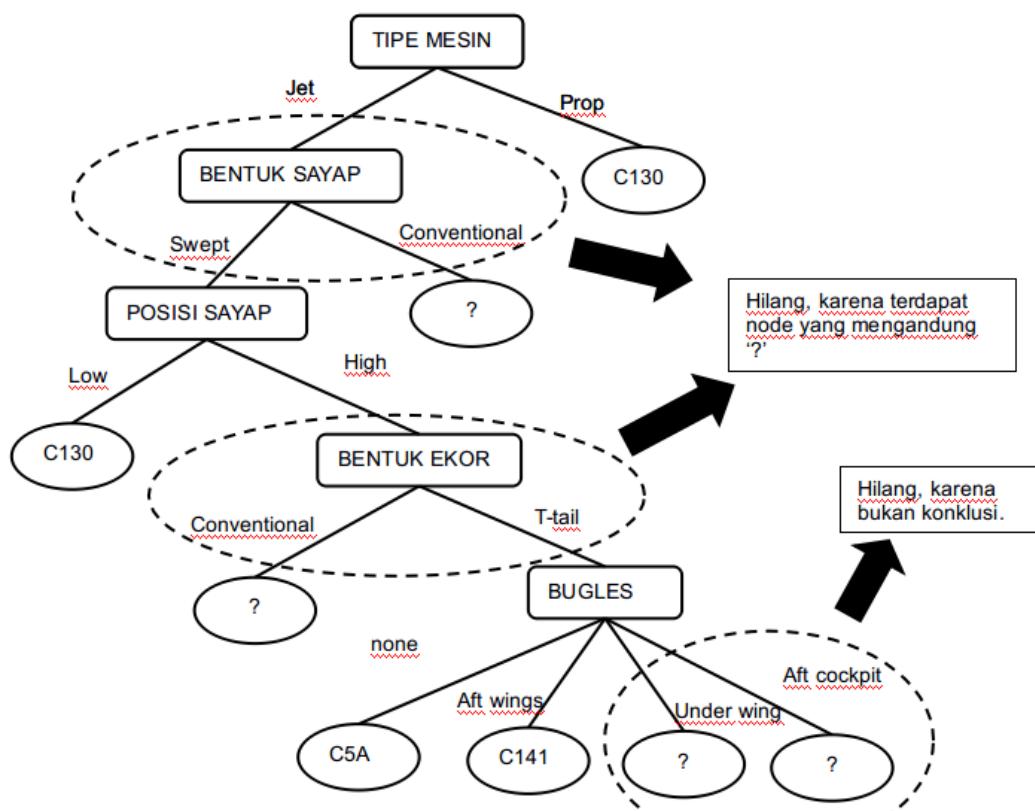
| Atribut Tipe | C130 | C141 | C5A | B747 |
|-----------------|--------------|------------|------------|--------------|
| Tipe mesin | Prop | Jet | Jet | Jet |
| Posisi sayap | High | High | High | Low |
| Bentuk sayap | Conventional | Swept-back | Swept-back | Swept-back |
| Bentuk ekor | Conventional | T-tail | T-tail | Conventional |
| Bulges | Under wings | Aft wings | None | Aft cockpit |

Identifikasi pesawat tertentu dapat dengan mudah diketahui jika melihat tabel keputusan pada tabel 6. Identifikasi pesawat dilakukan dengan memperhatikan 5 atribut penentu yang ada. Tetapi dapat saja identifikasi dilakukan hanya dengan memperhatikan atribut-atribut yang benar-benar membedakan tipe pesawat yang satu dengan yang lain. Hal ini akan diperlihatkan secara lebih jelas dalam bentuk pohon keputusan. Pohon keputusan dibuat dengan mengacu dari tabel keputusan yang ada. Dari pengetahuan akusisi dari tabel 6 tersebut kemudian dibuat menjadi sebuah pohon keputusan seperti pada gambar 6.



Gambar 6. Pohon Keputusan Identifikasi Pesawat (Hartati dan Iswanti, 2008:28)

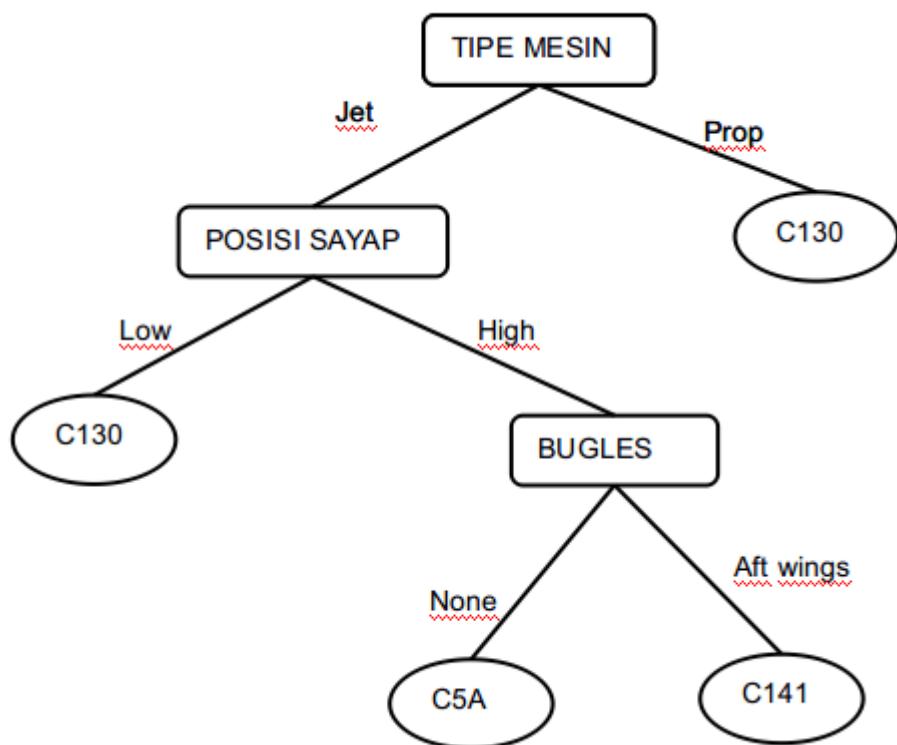
Pohon keputusan yang telah dibuat dengan mengacu dari tabel keputusan dapat digunakan sebagai acuan untuk mereduksi atribut-atribut yang sebenarnya dapat dihilangkan dalam proses identifikasi. Atribut yang dapat dihilangkan adalah atribut-atribut yang mengandung node dengan tanda tanya (?). Kecuali dari atribut tersebut dapat disimpulkan suatu konklusi. Proses reduksi pohon keputusan pada gambar 6 dapat dilihat seperti pada gambar 7 berikut.



Gambar 7. Proses Reduksi Atribut (Hartati dan Iswanti, 2008:30)

Setelah proses reduksi dilakukan maka akan dihasilkan pohon keputusan yang lebih sederhana seperti terlihat pada gambar 8. Pada

gambar 8, pohon keputusan hanya menggunakan 3 buah atribut untuk mengidentifikasi tipe pesawat. Sehingga tidak perlu semua atribut yang didapatkan dari akusisi pengetahuan dioergunakan saat identifikasi. Hal tersebut akan membantu pada saat proses yang dilakuakn komputer dari sisi penyimpanandan kecekpatan proses terkait dengan penyajian kaidah yang dihasilkan dari pohon keputusan. Dari sistem pakar sendiri dengan adanya atribut yang direduksi maka premis akan berkurang dengan sendirinya. Hal tersebut membantu mengurangi lamanya proses konsultasi, sehingga pengguna dapat lebih cepat mengetahui hasil identifikasi masalahnya.



Gambar 8. Pohon Keputusan Hasil Reduksi (Hartati dan Iswanti, 2008:31)

Menurut Hartati dan Iswanti (2008:37) kaidah yang efisien adalah kaidah yang memiliki atribut (premis) lebih sedikit dan dalam sesi konsultasi

dari sisi implementasi sistem akan memunculkan pertanyaan-pertanyaan yang potensial saja kepada pengguna. Perlu disadari juga bahwa tidak selalu pohon keputusan yang minimal menjadi yang terbaik. Pohon keputusan minimal atau paling minimal dari sejumlah alternatif pohon keputusan dapat digunakan jika secara realita situasinya bersifat *deterministic* (pasti, tidak mengandung unsur probabilitas). Secara teori, diperoleh hasil optimal jika jumlah atribut dalam suatu pohon keputusan dan dalam kasus yang sifatnya statis.

Pohon keputusan yang dihasilkan digunakan sebagai acuan dalam menyusun kaidah produksi. Kaidah-kaidah yang dihasilkan dari contoh pohon keputusan mengenai identifikasi pesawat dapat dilihat pada tabel 7.

Tabel 7. Contoh Kaidah Produksi (Hartati dan Iswanti, 2008:36)

| Pohon Keputusan | Himpunan Kaidah |
|-----------------|---|
| Gambar 11 | Kaidah 1: If tipe mesin prop Then tipe pesawat C130 |
| | Kaidah 2: If tipe mesin jet And posisi sayap low Then tipe pesawat B747 |
| | Kaidah 3: If tipe mesin jet And posisi sayap high And bugles none Then tipe pesawat C5A |
| | Kaidah 4: If tipe mesin jet And posisi sayap high And bugles aft wings Then tipe pesawat C141 |

6. Penalaran

“Penalaran adalah proses untuk menghasilkan inferensi dari fakta yang diketahui atau diasumsikan. Inferensi adalah konklusi logis (logical conclusion) atau implikasi berdasarkan informasi yang tersedia”(Hartati dan

Iswanti, 2008:43). Selanjutnya Giarantano dan Riley dalam Hartati dan Iswanti, (2008:43) menjelaskan bahwa terdapat 10 metode inferensi seperti yang disajikan dalam tabel 8.

Tabel 8. Metode Inferensi

| Metode Inferensi | Penjelasan |
|---------------------------------------|---|
| Deduksi | Proses penalaran dimana konklusi mengikuti premise. |
| Induksi | Inferensi dari hal-hal yang khusus menuju ke umum. |
| <i>Abduction</i> | Penalaran balik dari konklusi yang benar menuju ke premis yang menyebabkan terjadinya konklusi tersebut. |
| Intuisi | Jawaban atau hasil yang didapat kemungkinan berasal dari pengenalan pola yang ada tanpa disadari. Inferensi jenis ini berdasarkan pada intuisi dan sistem pakar belum mengimplementasikan tipe inferensi ini. |
| <i>Default</i> | Pengetahuan yang khusus kurang, secara <i>default</i> menggunakan pengetahuan yang umum. |
| Analogi | Menghasilkan konklusi berdasarkan kesamaan pada situasi yang lain. |
| <i>Heuristic</i> | Proses penalaran yang berdasarkan kesamaan pada situasi yang lain. |
| <i>Autoepistemic = self-knowledge</i> | Konklusi yang dihasilkan sistem, akan dipakai sebagai bagian premis dari suatu kaidah baru secara mandiri. Biasanya berlaku untuk sistem pakar yang komplek. |
| Nonmonotomic | Ketika bukti baru didapatkan pengetahuan yang sebelumnya kemungkinan menjadi salah. |
| <i>Generate and test</i> | Inferensi berdasarkan pada <i>trial and error</i> , tipe inferensi ini sering digunakan untuk perencanaan yang bertujuan efisiensi. |

7. Perunutan

Dalam melakukan inferensi diperlukan adanya proses pengujian kaidah-kaidah dalam urutan tertentu guna mencari kesesuaian antara kondisi awal dengan kondisi yang berjalan yang sudah dimasukkan pada basis data. “Perunutan adalah proses pencocokan fakta, pernyataan atau kondisi berjalan yang tersimpan pada basis pengetahuan maupun pada memori kerja dengan kondisi yang dinyatakan pada premis atau bagian kondisi pada kaidah” (Hartati dan Iswanti, 2008:45).

a. Runut Maju (*Forward Chaining*)

“Runut maju (*Forward Chaining*) adalah teknik pencarian yang dimulai dengan fakta yang diketahui, kemudian mencocokkan fakta-fakta tersebut dengan bagian IF dari aturan IF-THEN”(Sutojo dkk, 2011:171). Bila ada fakta yang cocok dengan bagian IF, maka aturan tersebut dieksekusi. Sebuah aturan dieksekusi, maka sebuah fakta baru (bagian THEN) ditambahkan ke dalam database. Runut maju juga disebut sebagai pencarian yang dimotori data (*data driven search*). Runut maju dapat dimodelkan sebagai berikut (Hartati dan Iswanti, 2008:45) :

IF (informasi masukkan)
THEN (konklusi)

Informasi masukan dapat berupa data, bukti, temuan, atau pengamatan. Sedangkan konklusi dapat berupa tujuan, hipotesa, penjelasan, atau diagnosis. Sehingga jalannya penalaran runut maju dapat dimulai dari data menuju tujuan, dari bukti menuju hipotsa, dari temuan menuju penjelasan, atau dari pengamatan menuju diagnosa.

b. Runut Mundur (Backward Chaining)

“Runut mundur atau (Backward Chaining) adalah metode inferensi yang bekerja mundur ke arah kondisi awal”(Sutojo dkk, 2011:178). Proses diawali dari Goal (bagian THEN dari aturan IF-THEN), kemudian pencarian mulai dijalankan untuk mencocokkan apakah fakta-fakta yang ada cocok dengan premis-premis di bagian IF. Jika cocok, aturan dieksekusi, kemudian hipotesis di bagian Then ditempatkan di basis data sebagai fakta baru. Jika tidak cocok, simpan premis di bagian IF ke dalam stack subGoal. Proses berakhir jika ditemukan atau tidak ada aturan yang bisa membuktikan kebenaran dari subGoal atau Goal.

Jadi secara umum runut balik diaplikasikan ketika tujuan atau hipotesis yang dipilih sebagai titik awal penyelesaian masalah. Runut balik disebut juga sebagai *goal-driven search*. Runut balik dimodelkan sebagai berikut (Hartati dan Iswanti, 2008:46) :

*Tujuan,
IF (kondisi).*

B. Komputer

1. Cara kerja Komputer

Komputer adalah mesin yang terprogram. Hal tersebut mengijinkan pengguna untuk menyimpan berbagai macam informasi dan kemudian memproses informasi atau data tersebut. Komputer juga dapat menjalankan tugas sesuai informasi yang masuk seperti menghitung angka atau menyusun kata. Lebih jelasnya menurut Barata (1999:10-11), cara computer bekerja adalah sebagai berikut :

- 1) **Computer menerima input.** Input computer adalah segala yang masuk atau dimasukkan ke dalam system computer. Input dapat diberikan oleh seseorang, komputer lain, atau perangkat lain (seperti diskette atau CD-ROM).
- 2) **Komputer melakukan operasi yang berguna,** memanipulasi data dalam berbagai cara. Manipulasi dalam computer ini disebut dengan pemrosesan. Contoh pemrosesan termasuk didalamnya mengolah perhitungan, mengurutkan daftar kata atau angka, mengubah dokumen dan gambar sesuai perintah pemakai, serta menggambarkan grafik. Data dalam computer diproses dalam *Central Processing Unit* (CPU).
- 3) **Komputer menyimpan data.** Komputer harus menyimpan data yang ada untuk diproses. Kebanyakan computer memiliki lebih dari satu lokasi untuk menyimpan data. Tempat dimana computer menyimpan data tergantung bagaimana data tersebut digunakan. Komputer menempatkan data disuatu tempat selama menunggu untuk diproses dan tempat lain ketika tidak dibutuhkan untuk diproses.
- 4) **Komputer menghasilkan output.** Output computer adalah informasi yang telah diproduksi oleh computer. Sebagai contoh output computer termasuk didalamnya laporan, dokumen, music, grafik, dan gambar. Output dapat muncul didalam beberapa format yang berbeda seperti kertas, diskette, atau berada di layar monitor.

2. Klasifikasi Komputer

Komputer secara umum dapat diklasifikasikan berdasarkan ukuran dan kemampuan. Menurut Barata (1999:5-6), berikut klasifikasi dari

beberapa jenis computer yaitu *mainframe computer*, *mini-computer*, *workstation*, *personal computer* (PC).

a. *Mainframe Computer*

Barata (1999:5) menerangkan bawah, *mainframe computer* berukuran besar, komputer multi-pengguna yang berkemampuan besar dan dapat mengukung kerja program yang berjalan bersamaan. Hal tersebut berarti *mainframe computer* dapat menjalankan aksi atau proses yang berbeda dalam waktu yang bersamaan. *mainframe computer* dapat digunakan oleh ratusan atau ribuan pengguna dalam waktu yang bersamaan pula. Organisasi besar biasanya menggunakan *mainframe computer* untuk menjalankan proses berukuran besar.

b. *Mini-computer*

Menurut Barata (1999:5), *Mini-computer* adalah computer multi-proses yang berukuran sedang. *Mini-computer* juga dapat melakukan beberapa proses sesacara bersamaan dan dapat mendukung 4 sampai 200 pengguna secara serentak. Dalam beberapa tahun terahir perbedaan antara *mini-computer* dan *mainframe computer* menjadi kabur. Sering kali perbedaan didasarkan pada bagaimana cara produsen ingin memasarkan produknya. Organisasi dapat menggunakan *mini-computer* untuk melakukan tugas seperti memanajemen informasi dalam system keuangan yang kecil atau mengelola basis data informasi yang kecil.

c. *Workstations*

Menurut Barata (1999:5), *Workstation* adalah komputer dengan pengguna tunggal yang berkemampuan besar. *Workstation* memiliki kapasitas penyimpanan dan memproses jumlah data yang besar, tetapi

hanya digunakan oleh satu orang pada waktu yang bersamaan. Meskipun demikian, workstation biasanya saling berhubungan satu dengan yang lain guna membentuk sebuah jaringan computer atau *local area network*. Hal tersebut berlaku beberapa orang, seperti karyawan dalam perusahaan dapat berkomunikasi satu dengan yang lain dan berbagi data atau file elektronik.

d. Personal computer (PC)

Barata (1999:6) menjelaskan bahwa, PC atau disebut juga dengan komputer mikro merupakan tipe yang popular digunakan saat ini. PC memiliki ukuran yang kecil, relative tidak mahal yang dirancang untuk pengguna individual. Sekarang ini, PC didunia secara mendasar dibagi menjadi *IBM-compatible* dan *macintosh-compatible*, yang dinamakan berdasarkan dua produsen komputer. Komputer juga terkadang disebut dengan komputer desktop. Organisasi dan individual menggunakan PC untuk banyak tugas, termasuk pengolahan kata, akuntansi, penerbitan, persiapan dan pengantaran presentasi, dan manajemen basis data. *Entry-level* PC sekarang memiliki kemampuan yang lebih baik dibanding beberapa tahun lalu. Selain itu sekarang ini hanya terdapat perbedaan tipis antara PC dan *workstation*.

3. Perangkat keras pada computer

Sebuah komputer merupakan satu kesatuan dari beberapa perangkat keras. Komputer tidak dapat bekerja dengan baik bila terdapat satu atau lebih perangkat keras yang hilang atau rusak. Menurut Muller (2013:59), berikut peripheral atau hardware yang dibutuhkan untuk membangun sebuah basis PC modern sebagaimana dapat dilihat pada tabel 9.

Tabel 9. Perangkat Keras PC

| Hardware atau Peripheral | Deskripsi singkat |
|---------------------------------|---|
| 1. Motherboard | Merupakan salah satu komponen penting dalam PC. Segala komponen lain terhubung pada motherboard. |
| 2. Processor | Processor merupakan otak dari computer. Processor juga disebut dengan <i>central processing unit</i> (CPU). Kebanyakan prosessos juga terdapat <i>graphics processing unit</i> (GPU) didalamnya. |
| 3. Memory (RAM) | System memory dalam computer sering kali disebut dengan RAM (<i>random access memory</i>). RAM merupakan memori primer yang menyokong semua program dan data yang diproses pada processor. |
| 4. Case/chasis | Merupakan <i>frame</i> atau chasis yang didalamnya memuat atau terpasang motherboard, power supply, disk drive, adapter card, dan fisikal komponen lain di PC. |
| 5. Power supply | <i>Power supply</i> berfungsi untuk menyediakan tenaga listrik pabi komponel internal dalam PC. |
| 6. Hard drive | Hard drive merupakan media penyimpanan utama bagi system PC. |
| 7. Optical drive | CD (<i>compact disk</i>), DVD (<i>digital versatile discs</i>), dan BD (<i>Blu-ray discs</i>) drive relatif memiliki kapasiatas yang besar dan merupakan <i>removable-media</i> . Kebanyakan system sekarang ini menyediakan <i>optical drive</i> yang memiliki kemampuan <i>write</i> dan <i>rewrite</i> . |
| 8. Keyboard | Keyboard merupakan perangkat pada PC yang digunakan manusia untuk berkomunikasi dan mengontrol system. |
| 9. Mouse | Mouse merupakan satu satunya jenis dari pointing device yang digunakan untuk mengoperasikan PC. |
| 10. Video card | Video card berisi GPU yang bertugas mengontrol informasi yang pengguna dapat lihat pada monitor. |
| 11. Monitor | Monitor menampilkan informasi dilayar dan dikontrol oleh GPU atau <i>video card</i> . |
| 12. Sound card | Sound card berfungsi untuk membantu PC menghasilkan suara yang komplek. |
| 13. Network | Kebanyak PC dipaketkan dengan <i>wired</i> atau <i>wireless network interface</i> . Sehingga PC mampu terhubung ke jaringan computer. |

4. Proses Diagnosa Pada PC

Proses diagnosa pada buku yang penulis rujuk sebagai sumber pengetahuan digambarkan sebagai sebuah *flowchart*. Proses diagnosa kerusakan komputer untuk setiap perangkat keras yang ada tidaklah selalu sama. Masing-masing perangkat keras memiliki *flowchart* tersendiri. Daftar *flowchart* untuk proses diagnosa masing-masing perangkat keras dapat dilihat pada lampiran bagian daftar pohon keputusan atau *rule base*. Proses diagnosa selalu dimulai dengan memunculkan sebuah pertanyaan. Pertanyaan yang muncul pada umumnya terkait dengan gejala yang mungkin terjadi. Proses selanjutnya adalah memunculkan pertanyaan lain sesuai dengan jawaban atau tanggapan pertanyaan sebelumnya. Proses yang sama akan berlanjut sampai pada akhirnya gejala-gejala yang ditanyakan atau diindikasikan ada, akan mengasilkan sebuah kesimpulan dimana kemungkinan letak permasalahan.

C. Android

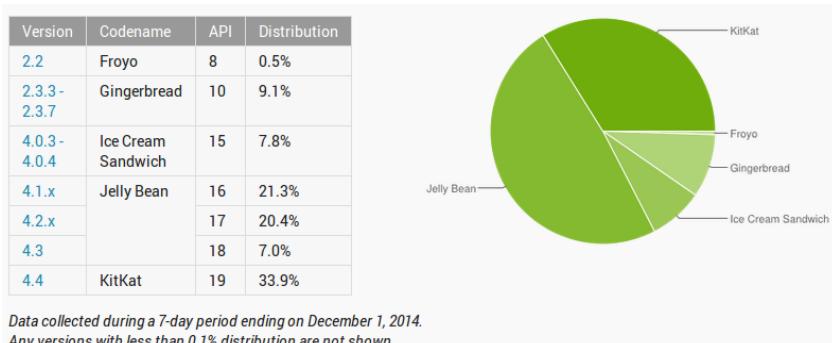
Android diperkenalkan pertama kali oleh Google pada tahun 2007, android saat ini menjadi sistem operasi yang banyak digunakan pada telepon pintar. Android telah beberapa kali memperbarui versinya, Gargenta dan Nakamura (2014:5) menyatakan dalam bukunya urutan versi android yang ada sebagai mana dapat dilihat pada tabel 10.

Tabel 10. *Android Version History*

| Android Version | API Level | Codename |
|-----------------|-----------|----------|
| Android 1.0 | 1 | |
| Android 1.1 | 2 | |
| Android 1.5 | 3 | Cupcake |
| Android 1.6 | 4 | Donut |

| Android Version | API Level | Codename |
|-----------------|-----------|--------------------|
| Android 2.0 | 5 | Eclair |
| Android 2.01 | 6 | Eclair |
| Android 2.1 | 7 | Eclair |
| Android 2.2 | 8 | Froyo |
| Android 2.3 | 9 | Gingerbread |
| Android 2.3.3 | 10 | Gingerbread |
| Android 3.0 | 11 | Honeycomb |
| Android 3.1 | 12 | Honeycomb |
| Android 3.2 | 13 | Honeycomb |
| Android 4.0 | 14 | Ice Cream Sandwich |
| Android 4.0.3 | 15 | Ice Cream Sandwich |
| Android 4.1 | 16 | Jelly Bean |
| Android 4.2 | 17 | Jelly Bean |
| Android 4.3 | 18 | Jelly Bean |
| Android 4.4 | 19 | KitKat |

Dalam membuat sebuah aplikasi di android, pemilihan versi android dan API level adalah penting. Hal tersebut terkait dengan fitur yang ada pada masing-masing versi. Serta kompatibilitas aplikasi terhadap perangkat-perangkat android yang digunakan oleh masing-masing pengguna. Dalam situs resminya android merilis persentase pengguna masing-masing versi android sebagaimana dapat dilihat pada gambar 9 (android, 2014)



Gambar 9. Persentase Pengguna Versi Android

Tabel 11. Beberapa Jenis Layar Yang Didukung Oleh Android

| | Low density (120), <i>ldpi</i> | Medium density (160), <i>mdpi</i> | High density (240), <i>hdpi</i> | Extra-high-density (320), <i>xhdpi</i> |
|---------------------------|--|--|--|--|
| <i>Small screen</i> | QVGA (240x320) | | 480x640 | |
| <i>Normal screen</i> | WQVGA400 (240x400) WQVGA432 (240x432) | HVGA (320x480) | WVGA800 (480x800) WVGA854 (480x854) 600x1024 | 640x960 |
| <i>Large screen</i> | WVGA800** (480x800) WVGA854** (480x854) | WVGA800* (480x800) WVGA854* (480x854) 600x1024 | | |
| <i>Extra-Large screen</i> | 1024x600 | WXGA (1280x800)[†] 1024x768 1280x768 | 1536x1152 1920x1152 1920x1200 | 2048x1536 2560x1536 2560x1600 |

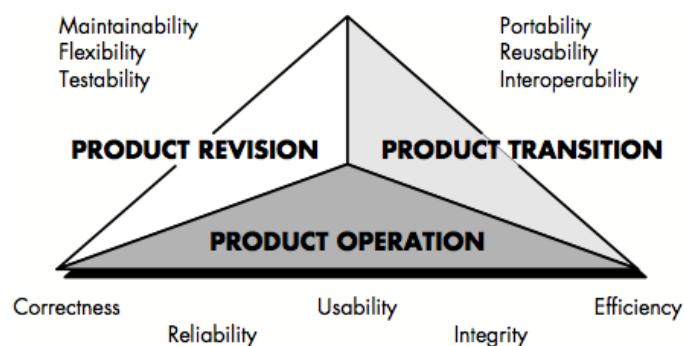
Selain itu Android juga mendukung beberapa jenis layar pada perangkat *smartphone* dan tablet sebagai mana terdapat pada tabel 11 (android, 2014). Dimana layar yang didukung oleh android terdiri dari *small screen*, *normal screen*, *large screen*, dan *extra-large screen*. Selain itu masing-masing ukuran layar memiliki ukuran densitas yang berbeda beda.

D. Kualitas Perangkat Lunak

Kualitas perangkat lunak memiliki banyak definisi dan menimbulkan berbagai perdebatan. Kontroversi tersebut muncul karena orang-orang tidak sepakat dengan definisi mengenai kualitas tersebut (Cote dkk, 2006). Salah satu definisi yang disepakati banyak pihak adalah dari Pressman (2010:400) yang mendefinisikan kualitas perangkat lunak sebagai “*An effective software process applied in a manner that creates a useful product that provides measureable value for those who produce it and those who use it*”. Dengan demikian kualitas perangkat lunak dapat diartikan sebagai proses yang efektif yang diwujudkan dalam bentuk produk yang dapat memberikan manfaat dan dapat diukur.

Penelitian tentang kualitas perangkat lunak sudah dimulai sejak lama.

Salah satu pendekatan yang digunakan dalam penelitian tersebut adalah tentang model yang digunakan. Oleh karena itu model kualitas telah menjadi instrumen yang umum digunakan untuk mengukur kualitas perangkat lunak (Deissenboeck dkk, 2009). Ada beberapa model kualitas perangkat lunak yang telah dikembangkan. McCall (McCall dkk, 1977) mengenalkan model kualitas perangkat lunak pada tahun 1977. Menurut Pressman (2010:402) model kualitas perangkat lunak McCall memiliki fokus pada 3 aspek penting dari perangkat lunak, yaitu: karakteristik operasional, kemampuan dalam menerima perubahan dan adaptabilitas terhadap lingkungan baru. Gambar 10 menjelaskan ketiga aspek tersebut:



Gambar 10. Model Kualitas Perangkat Lunak Menurut Mccall (McCall dkk, 1977)

Penjelasan faktor kualitas perangkat lunak menurut McCall (1977:15) adalah sebagai berikut:

- a. *Correctness*, berkaitan dengan kemampuan program memenuhi spesifikasi dan tujuan yang diinginkan pengguna.
- b. *Reliability*, berkaitan dengan kemampuan program untuk menjalankan fungsinya sesuai dengan tingkat presisi yang telah ditentukan.

- c. *Efficiency*, berkaitan dengan jumlah sumber daya komputer dan kode yang dibutuhkan oleh program untuk menjalankan fungsinya.
- d. *Integrity*, berkaitan dengan kontrol akses terhadap perangkat lunak atau data.
- e. *Usability*, berkaitan dengan usaha yang dibutuhkan oleh pengguna untuk mempelajari, mengoperasikan, menyiapkan *input*, dan menginterpretasikan *output* dari program.
- f. *Maintainability*, berkaitan dengan usaha yang diperlukan untuk menemukan dan mengatasi kesalahan di dalam program.
- g. *Flexibility*, berkaitan dengan usaha yang diperlukan untuk mengubah program yang beroperasi.
- h. *Testability*, berkaitan dengan usaha yang diperlukan untuk menguji sebuah program untuk memastikan bahwa program tersebut dapat berjalan sesuai dengan fungsinya.
- i. *Portability*, berkaitan dengan usaha yang diperlukan untuk dapat mentransfer program dari suatu lingkungan perangkat keras atau lunak tertentu ke lingkungan yang lain.
- j. *Reusability*, berkaitan dengan bagaimana suatu bagian dari program dapat digunakan kembali di dalam program lain.
- k. *Interoperability*, berkaitan dengan usaha yang diperlukan untuk menghubungkan sebuah sistem dengan sistem yang lain.

Masalah ditemukan dalam implementasi karena cukup susah, bahkan dalam beberapa kasus tidak mungkin dilakukan pengukuran secara langsung terhadap faktor kualitas tersebut (Cote dkk, 2006). Oleh karena itu muncul model kualitas perangkat lunak yang dirilis oleh ISO 9126. Standar ISO 9126 dikembangkan dengan tujuan mengidentifikasi faktor kunci dalam kualitas perangkat lunak.

Dalam bukunya, Pressman (2010:403) menjelaskan 6 faktor tersebut sebagai berikut:

- a. *Functionality*, kemampuan perangkat lunak untuk memenuhi kebutuhan pengguna yang diindikasikan pada sub faktor berikut : *suitability, accuracy, interoperability, security , dan functionality compliance*.
- b. *Reliability*, berkaitan dengan kapabilitas sebuah perangkat lunak untuk mampu menjaga level performa yang dimilikinya. Faktor ini dapat ditunjukkan oleh beberapa sub faktor yaitu : *maturity, fault tolerance, recoverability, dan reliability compliance*.
- c. *Usability*, berkaitan dengan kemudahan perangkat lunak untuk digunakan yang diindikasikan pada sub faktor berikut : *understandability, learnability, operability, attractiveness, dan usability compliance*.
- d. *Efficiency*, kemampuan perangkat lunak memanfaatkan sumber daya yang tersedia secara optimal, diindikasikan pada beberapa sub faktor yaitu : *time behavior, resource utilization, dan efficiency compliance*.
- e. *Maintainability*, berkaitan dengan kemudahan suatu perangkat lunak untuk diperbaiki di kemudian hari, diindikasikan oleh sub faktor berikut

ini : *analyzability, changeability, stability, testability*, dan *Maintainability compliance*.

- f. *Portability*, berkaitan dengan kemudahan perangkat lunak untuk dipindahkan atau diakses dari satu lingkungan tertentu ke lingkungan yang lain yang diindikasikan pada sub faktor berikut : *adaptability, installability, conformance, replaceability*, dan *portability compliance*.

1. Faktor Kualitas *Functionality*

Pressman (2010:403) mendefinisikan *functionality* sebagai kemampuan perangkat lunak untuk memenuhi kebutuhan pengguna. Sementara ISO/IEC (1991) mendefinisikan *functionality* sebagai “*the capability of the software product to provide functions which meet stated and implied needs when the software is used under specified condition*”. Dari beberapa definisi tersebut *functionality* dapat diartikan sebagai kemampuan perangkat lunak untuk menyediakan fungsi sesuai kebutuhan pengguna saat digunakan dalam kondisi tertentu. Selain itu Menurut ISO 9126 (ISO/IEC/1991), *functionality* memiliki beberapa sub-karakteristik yaitu :

- a. *Suitability*, merupakan sub-karakteristik yang penting dari *functionality* dimana karakteristik ini merujuk pada kesesuaian dari fungsi-fungsi perangkat lunak.
- b. *Accuracy*, sub-karakteristik ini menunjukkan seberapa benar fungsi dari perangkat lunak.
- c. *Interoperability*, sub-karakteristik ini diutamakan pada kemampuan perangkat lunak dalam berinteraksi antar komponen dalam system.

- d. Security, sub-karakteristik ini berkaitan dengan akses tanpa izin ke fungsi perangkat lunak.
- e. Compliance, sub-karakteristik ini diutamakan dalam perhatiannya mengenai pemenuhan syarat secara hukum dan aturan yang berlaku dari fungsi yang ada di dalam perangkat lunak.

Faktor kualitas *functionality* dapat diuji dengan melakukan analisis fungsionalitas dari setiap komponen pada suatu perangkat lunak. Metode yang digunakan adalah *black-box testing*. Pressman (2010) menjelaskan bahwa *black-box testing* atau *behavioral testing* merupakan pengujian yang memiliki fokus pada kebutuhan fungsional dari suatu perangkat lunak. Dengan melakukan pengujian ini, analis sistem dapat memperoleh kumpulan kondisi *input* yang akan mengerjakan seluruh keperluan fungsional program dan *output* yang akan dihasilkan pada kondisi *input* tertentu. Lebih lanjut Niknejad (2011:8) menyatakan bahwa pengujian yang dilakukan adalah dengan menghitung jumlah fitur fungsional yang terdapat pada aplikasi, kemudian dibandingkan dengan fitur fungsional yang berjalan. Hasil dari pengujian tersebut kemudian dianalisis dengan metode analisis deskriptif.

2. Faktor Kualitas *Reliability*

Pressman (2010: 403) mendefinisikan *reliability* sebagai aspek yang berkaitan dengan kapabilitas sebuah perangkat lunak untuk mampu menjaga level performa yang dimilikinya. *Reliability* merupakan salah satu elemen penting dalam kualitas perangkat lunak secara keseluruhan. Jika suatu program berulangkali gagal untuk menjalankan operasi pada tingkat performansi tertentu maka program tersebut memiliki kualitas yang buruk.

Tidak seperti faktor kualitas yang lain, *reliability* dari perangkat lunak dapat diukur secara langsung dengan menggunakan beberapa metrik.

Menurut Malaiya (2005:3) *software reliability* dapat diukur dengan menggunakan *Defect density*. *Defect density* diukur dengan menentukan berapa jumlah *defect* per 1000 *line of code*(KLOC). Defect sendiri menurut Malaiya (2005:1) “*Defect (fault or bug): An error in system implementation that can cause a failure during execution.*”

McConnell (2004:652) dalam bukunya menjelaskan bahwa jumlah *error* yang terjadi dalam pengembangan perangkat lunak, terutama yang kaitannya dengan penulisan kode, dapat diperkirakan berdasarkan besar kecilnya *project* perangkat lunak yang sedang dikembangkan. Rentang kemungkinan *error* yang terjadi dalam suatu *project* digambarkan dalam tabel 12.

Tabel 12.Perkiraan Jumlah *Error* McConnell

| Ukuran Project (Line of Code/LOC) | Perkiraan Jumlah <i>Error</i> |
|--|--------------------------------------|
| <2K | 0 - 25 <i>error</i> / KLOC |
| 2K – 16K | 0 - 40 <i>error</i> / KLOC |
| 16K – 64K | 0,5 - 50 <i>error</i> / KLOC |
| 64K – 512K | 2 - 70 <i>error</i> / KLOC |
| >512K | 4 - 100 <i>error</i> / KLOC |

McConnell (2004:558)juga menjelaskan bahwa kemungkinan *error* yang dapat ditemukan dalam sebuah *project* tergantung pada kualitas pengembangan perangkat lunak yang dilakukan. Semakin baik kualitas pengembangan perangkat lunak tersebut, maka semakin kecil ditemukan *error* dalam *project* tersebut. Berikut adalah beberapa rentang kemungkinan *error* tersebut :

- a. *Industry Average* : 1-25 error tiap 1 KLOC
- b. *Microsoft Application* : 10-20 error tiap 1 KLOC pada tiap tahap pengujian *in house* dan 0.5 error tiap KLOC pada tahap peluncuran

3. Faktor Kualitas *Usability*

Usability merupakan faktor kualitas yang berhubungan langsung dengan pengguna. Pressman (2010: 404) mendefinisikan *usability* sebagai kemudahan perangkat lunak untuk digunakan. Sementara ISO 9126 (ISO/IEC, 1991) mendefinisikan *usability* sebagai kemampuan perangkat lunak untuk dipahami, dipelajari, digunakan, dan menarik bagi pengguna, ketika digunakan dalam kondisi tertentu. Dari dua definisi tersebut dapat disimpulkan bahwa *usability* merupakan faktor yang berhubungan dengan kemampuan perangkat lunak untuk dipahami oleh pengguna. Suatu program yang memiliki kualitas *usability* bagus akan mudah dipahami dan digunakan oleh pengguna.

Menurut ISO 9126(ISO/IEC, 1991) *usability* memiliki sub karakteristik sebagai berikut:

- a. *Understandability*, kemampuan perangkat lunak dalam kemudahan untuk dipahami.
- b. *Learnability*, kemampuan perangkat lunak dalam kemudahan untuk dipelajari.
- c. *Operability*, kemampuan perangkat lunak dalam kemudahan untuk dioperasikan.
- d. *Attractiveness*, kemampuan perangkat lunak dalam menarik perhatian pengguna.

Pengujian faktor kualitas *usability* pada penelitian ini dilakukan dengan melakukan survei terhadap pengguna menggunakan angket kuisioner J.R. Lewis (Lewis, 1993) yang telah dipublikasikan dalam *paper IBM Computer Usability Satisfaction Questionnaires: Psychometric Evaluation and Instructions for Use*. Paper ini telah dipublikasikan dalam *International Journal of Human Computer Interaction* pada tahun 1993. Angket kuisioner J.R. Lewis sudah banyak digunakan sebagai instrumen untuk melakukan penilaian terhadap faktor kualitas *usability* karena sudah memenuhi sub karakteristik dari aspek *usability*.

4. Faktor Kualitas *Efficiency*

Efficiency merupakan faktor kualitas perangkat lunak yang terkait dengan sumber daya. Pressman (2010: 404) mendefinisikan *efficiency* sebagai kemampuan perangkat lunak memanfaatkan sumber daya yang tersedia secara optimal. Sementara ISO 9126 (ISO/IEC, 1991) mendefinisikan *efficiency* sebagai kemampuan perangkat lunak untuk memberikan kinerja yang sesuai dan relatif terhadap jumlah sumber daya yang digunakan pada keadaan tersebut. Kedua definisi tersebut hampir sama sehingga *efficiency* dapat disimpulkan sebagai kemampuan perangkat lunak untuk memanfaatkan sumber daya yang ada secara optimal. *Efficiency* memiliki dua sub karakteristik, yaitu:

- a. *Time behavior*, kemampuan perangkat lunak dalam memberikan respon dan waktu pengolahan yang sesuai saat melakukan fungsinya.
- b. *Resource Utilization*, kemampuan perangkat lunak dalam menggunakan sumber daya yang dimilikinya ketika melakukan fungsi yang ditentukan.

5. Faktor Kualitas *Maintainability*

Pressman (2010: 404) mendefinisikan *maintainability* sebagai aspek pada perangkat lunak yang berkaitan dengan kemudahan suatu perangkat lunak untuk diperbaiki di kemudian hari. Sementara ISO 9126 (ISO/IEC, 1991) mendefinisikan *maintainability* sebagai kemampuan perangkat lunak untuk dimodifikasi. Modifikasi meliputi koreksi perbaikan atau adaptasi terhadap perubahan lingkungan, persyaratan, dan spesifikasi fungsional.

Pengujian untuk aspek *maintainability* pada aplikasi Android belum ada standar pengukuran khusus. Sehingga pada pengujian ini menggunakan ukuran-ukuran (*metrics*) yang dilakukan peneliti dengan diuji secara fungsi. Langkah-langkah uji *maintainability* menurut Heitlager, dkk (2007), pengujian *maintainability* dilakukan terhadap source code program. Gambar 11 menunjukkan keterkaitan antara properti pada *source code* program dan juga sub karakteristik aspek *maintainability* pada standar ISO 9126:

| ISO 9126 maintainability | source code properties | | | | |
|-----------------------------|------------------------|---------------------|-------------|-----------|--------------|
| | volume | complexity per unit | duplication | unit size | unit testing |
| analysability | x | | x | x | x |
| changeability | | x | x | | |
| stability | | | | | x |
| testability | | x | | x | x |

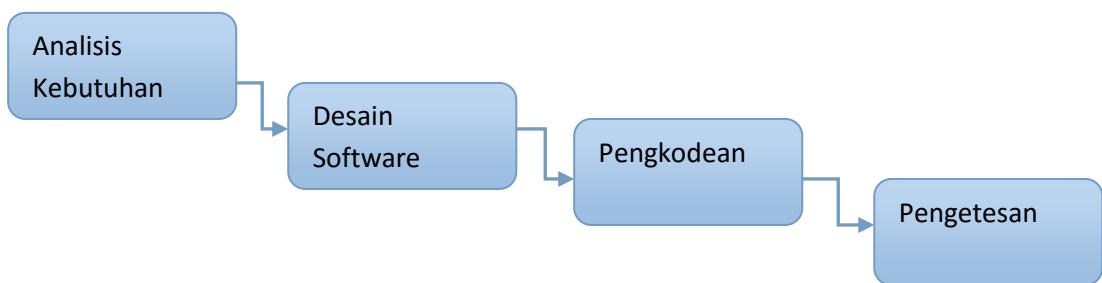
Gambar 11. *Maintainability Metric*

6. Faktor Kualitas *Portability*

Portability merupakan faktor kualitas yang berkaitan dengan media untuk mengakses perangkat lunak. Pressman (2010:404) mendefinisikan *portability* sebagai kemudahan perangkat lunak untuk dipindahkan atau diakses dari satu lingkungan tertentu ke lingkungan yang lain. Aspek *portability* dalam penelitian ini dilakukan dengan menguji aplikasi yang dibuat pada beberapa perangkat *smartphone android* yang berbeda

E. Model Pengembangan Sekuensial Linier

Sekuensial linier mengusulkan sebuah pendekatan perkembangan perangkat lunak yang sistematik dan sekuensial yang mulai pada tingkat dan kemajuan sistem pada seluruh analisis, desain, kode, pengujian, dan peneliharaan. Model sekuensial linier sering disebut juga dengan siklus kehidupan klasik atau model air terjun (*Waterfall*). Gambar model pengembangan sekuensial linier menurut Pressman (2002:37) dapat dilihat pada gambar 12.



Gambar 12. Model Sekuensial Linier atau *Waterfall*

Pressman (2002:37-38) dalam bukunya menjelaskan bahwa model sekuensial linier melingkupi aktivitas-aktivitas berikut:

- a. **Rekayasa dan pemodelan sistem/informasi.** Karena perangkat lunak selalu merupakan bagian dari sebuah sistem yang lebih besar, kerja dimulai dengan membangun syarat dari semua elemen sistem dan mengalokasikan beberapa subset dari kebutuhan ke perangkat lunak tersebut. Pandangan sistem ini penting ketika perangkat lunak harus berhubungan dengan elemen-elemen yang lain seperti perangkat lunak, manusia, dan database.
- b. **Analisis kebutuhan perangkat lunak.** Proses pengumpulan kebutuhan diintensifkan dan difokuskan, khususnya pada perangkat lunak. Untuk memahami sifat program yang dibangun, perekayasa perangkat lunak (analisis) harus memahami domain informasi, tingkah laku, unjuk kerja, dan antar muka yang diperlukan.
- c. **Desain.** Desain perangkat lunak sebenarnya adalah proses multi langkah yang berfokus pada empat atribut sebuah program yang berbeda, struktur data, arsitektur perangkat lunak, representasi interface, dan detail (algoritma) prosedural. Proses desain menerjemahkan syarat/kebutuhan ke dalam sebuah representasi perangkat lunak yang dapat diperkirakan demi kualitas sebelum representasi pemunculan kode.
- d. **Generasi kode.** Desain harus diterjemahkan ke dalam bentuk mesin yang bisa dibaca. Langkah pembuatan kode melakukan tugas ini. Jika

desain dilakukan dengan cara yang lengkap, pembuatan kode dapat diselesaikan secara mekanis.

- e. **Pemeliharaan.** Perangkat lunak akan mengalami perubahan setelah disampaikan kepada pelanggan. Perubahan akan terjadi karena kesalahan-kesalahan ditentunkan, karena perangkat lunak harus disesuaikan untuk mengakomodasi perubahan-perubahan didalam lingkungan eksternalnya.
- f. **Pengujian.** Sekali kode dibuat, pengujian program dimulai. Proses pengujian berfokus pada logika internal perangkat lunak, memastikan bahwa, semua pernyataan sudah diuji, dan pada eksternal fungsional yaitu mengarahkan pengujian untuk menemukan kesalahan-kesalahan dan memastikan bahwa input yang dibatasi akan memberikan hasil aktual yang sesuai dengan hasil yang dibutuhkan.

F. Penelitian Yang Relevan

1. *A New Approach for Developing Diagnostic Expert Systems on Mobile Phones.* Oleh Yasser Abdelhamid dan Mohammed El-Helly, 2013. Jurnal, *Communications in Information Science and Management Engineering* bulan Agustus 2013, Volume. 3 Issue. 8 pp 374-384. Dalam membangun sistem pakar dengan model desain *Expert System as a Standalone Application*, terdapat beberapa persyaratan yang harus dipenuhi. Persyaratan tersebut antara lain representasi pengetahuan yang ringan, antarmuka pengguna dan mesin inferensi yang dapat beradaptasi dengan keterbatasan perangkat moble phone, serta kemampuan untuk berjalan pada perangkat yang

- berbeda. Model *Expert System as a Standalone Application* berhasil diimplementasikan dalam sistem pakar pada bidang pertanian mengenai pemanenan stroberi.
2. ESPCRM - *an Expert System for Personal Computer Repair and Maintenance.* Oleh Goh Wee Leng dan Lau Kim Teen, 1992. Jurnal, *Engineering Applications of Artificial Intelligence.* Volume. 5, Issue. 2, pp. 121-133. Hasil penelitian berupa sistem pakar yang dinamakan ESPCRM. ESPCRM dikembangkan menggunakan *Personal Consultant Plus 4.0 expert system shell.* Sistem pakar tersebut dapat membantu teknisi dalam rangka penanganan masalah pada komputer dengan cara yang sistematik dan akurat.
 3. *Pc Diagnosis And Troubleshooting Expert System Based On Computer Response During Power On Self Test (Post) - PCDIASHOOT.* Oleh Mohd Daud Bin Isa dan Othman Bin Sidek, 2000. Jurnal, *Intelligent Systems and Technologies for the New Millennium,* diterbitkan Oleh TENCON Proceedings. Penelitian menghasilkan sebuah sistem pakar yang dinamakan PCDIASHOOT. Sistem pakar tersebut memiliki keunggulan diantaranya sistem pakar tersebut mampu mendiagnosa berbagai masalah pada komputer baik berupa permasalahan pada hardware maupun software. Sistem pakar tersebut juga memiliki kemampuan sebagai media pelatihan bagi teknisi baru maupun pengguna komputer dalam mendiagnosa dan menangani masalah pada komputernya tanpa bantuan seorang ahli.

Dari penelitian yang relevan tersebut, belum ada yang mengembangkan sistem pakar perbaikan dan diagnosa komputer pada perangkat *smartphone* berbasis android.

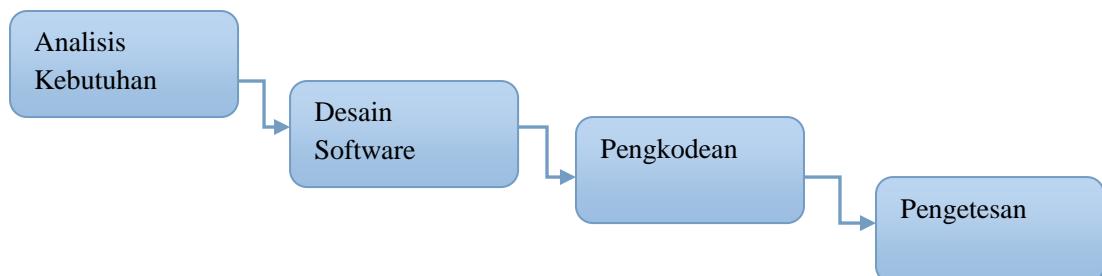
BAB III

METODOLOGI PENELITIAN

A. Jenis Penelitian

Metode penelitian yang digunakan adalah metode penelitian dan pengembangan (*Research and Development*). Hal ini berarti Hal ini berarti bahwa penelitian *research and development* merupakan suatu proses dalam mengembangkan sebuah produk serta melakukan pengujian terhadap validitas produk yang dikembangkan.

Dalam mengembangkan sistem pakar ini penulis menggunakan model pengembangan perangkat lunak sekuensial linier. Adapun langkah-langkah pengembangannya dapat dilihat pada gambar 16.



Gambar 13. Model Pengembangan Perangkat Lunak Sekuensial Linier Sebagai Tahapan Untuk Model Penelitian Dan Pengembangan (Pressman, 2002:37)

B. Langkah-Langkah Penelitian

1. Analisis Kebutuhan

Pada tahap ini analisis terhadap kebutuhan perangkat lunak dan perangkat keras untuk dapat mengembangkan dan menjalankan aplikasi sistem pakar yang nantinya akan berjalan pada perangkat *smartphone* Android. Analisis kebutuhan juga dilakukan terhadap kebutuhan fitur pada aplikasi. Sementara pengumpulan data untuk menyusun aturan yang akan digunakan pada sistem pakar nanti dilakukan dengan melakukan studi literatur.

2. Desain Software

Setelah dilakukan analisis kebutuhan dan pengumpulan data langkah selanjutnya adalah pembuatan desain aplikasi. Desain yang dibuat meliputi desain UML (*Unified Modelling Language*), desain kaidah yang digunakan, desain antar muka baik untuk *knowledge engineer* maupun pengguna akhir, flow chart program, serta desain database.

3. Pengkodean

Pada proses pengkodean atau implementasi, sistem pakar mulai kerjakan dengan mengacu pada desain yang telah dibuat sebelumnya. Sistem pakar dibuat dengan menggunakan bahasa pemrogramaan java yang disesuaikan untuk android. Serta menggunakan database SQLite untuk menyimpan data dan susunan aturan yang dibuat. Pada tahap ini pula kaidah yang telah dibuat diimplementasikan kedalam program.

4. Pengetesan

Tahap pengetesan dilakukan setelah aplikasi selesai dibuat. Pengetesan dilakukan dengan menggunakan faktor kualitas ISO 9126 sebagai panduan. Faktor kualitas yang digunakan terdiri dari faktor *functionality, reliability, maintainability, portability*, dan *usability*. Pengetesan faktor *functionality, reliability, maintainability, portability* dilakukan oleh pengembang. Setelah keempat faktor tadi selesai dites, aplikasi kemudian diujikan kepada pengguna untuk dilakukan pengetesan terkait faktor *usability*. Tujuan dari semua pengetesan yang telah dilakukan adalah guna menghasilkan sebuah aplikasi yang baik dan dapat diandalkan.

C. Waktu dan Tempat Penelitian

Penelitian rencananya akan dilakukan dari bulan januari 2015 sampai dengan november 2015. Lokasi penelitian untuk proses pembuatan aplikasi, validasi, dan revisi dilakukan di Jurusan Pendidikan Teknik Elektronika, Fakultas Teknik, Universitas Negeri Yogyakarta. Serta SMKN 2 Depok Sleman untuk melakukan pengujian *beta test* aplikasi ke pengguna.

D. Definisi Variabel

Definisi variabel penelitian yang gunakan adalah sebagai berikut:

1. *Functionality*, berkaitan dengan kemampuan perangkat lunak untuk memenuhi kebutuhan pengguna.
2. *Reliability*, berkaitan dengan kapabilitas sebuah perangkat lunak untuk mampu menjaga level performa yang dimilikinya.

3. *Usability*, berkaitan dengan kemudahan perangkat lunak untuk digunakan oleh pengguna.
4. *Maintainability*, berkaitan dengan kemudahan suatu perangkat lunak untuk diperbaiki atau dimodifikasi di kemudian hari.
5. *Portability*, berkaitan dengan kemudahan perangkat lunak untuk dipindahkan atau diakses dari satu lingkungan tertentu ke lingkungan yang lain.

E. Desain Penelitian

Desain penelitian yang digunakan adalah sebagai berikut:

1. *Functionality* diteliti dengan menggunakan metode *black-box* berupa checklist daftar fungsi yang dimiliki oleh aplikasi yang diisi oleh ahli dan analisis deskriptif terhadap fungsionalitas yang ada dalam setiap komponen perangkat lunak.
2. *Portability* diteliti dengan melakukan observasi dan analisis terhadap aplikasi sistem pakar yang telah dibuat. Sistem pakar tersebut dites apakah dapat berjalan pada beberapa perangkat android dan emulator yang berbeda.
3. *Reliability* diteliti dengan melakukan pengujian terhadap aplikasi sistem pakar untuk mengetahui tingkat *defect density*.
4. *Usability* diteliti dengan menggunakan angket usability dari J. R Lewis (Lewis, 1993). *Usability* diujikan ke pengguna untuk mendapat respon dari pengguna berkaitan dengan kemudahan dalam menggunakan

aplikasi.

5. *Maintainability* diteliti dengan menggunakan *matrix maintainability* sesuai yang dapat dilihat pada gambar 14. Dengan mengukur persentase *source code properties*.

| ISO 9126 maintainability | source code properties | | | | |
|-----------------------------|------------------------|---------------------|-------------|-----------|--------------|
| | volume | complexity per unit | duplication | unit size | unit testing |
| analysability | x | | x | x | x |
| changeability | | x | x | | |
| stability | | | | | x |
| testability | | x | | x | x |

Gambar 14. Metrik *Maintainability* oleh Heitlager, dkk (2007)

F. Teknik Pengumpulan Data

Teknik pengumpulan data dalam penelitian ini adalah sebagai berikut;

1. **Studi literatur**, dilakukan guna mendapatkan pemahaman mengenai sistem pakar yang dibuat. Termasuk didalamnya bagaimana model sistem pakar, teknik inferensi, dan pengembangan kaidah mengenai proses diagnosa perbaikan perangkat komputer. Sumber pengetahuan yang digunakan dalam membuat kaidah/aturan adalah dengan menggunakan pengetahuan terbukukan. Buku yang digunakan adalah buku karangan Morris Rosenthal berjudul “*Computer Repair with Diagnostic Flowchart, 3rd Edition*”. Studi literatur juga digunakan pada materi *quality model* ISO 9126. Hal tersebut dilakukan untuk lebih

memahami masing-masing faktor kualitas yang ada. Serta untuk menentukan proses validasi yang sesuai untuk masing-masing faktor kualitas.

2. **Observasi**, dilakukan untuk mengumpulkan data terkait dengan pengujian kualitas perangkat lunak pada faktor kualitas: *reliability*, *Maintainability* dan *portability*.
3. **Kuisisioner**, digunakan untuk pengumpulan data pada proses pengujian terkait faktor *functionality* dan *usability*. Kuisisioner untuk *functionality* ditujukan untuk ahli. Sedangkan kuisioner untuk *usability* ditujukan ke target pengguna aplikasi.

G. Subjek Penelitian

Subjek dalam penelitian ini pada aspek *functionality*, *reliability*, *Maintainability* dan *portability* adalah aplikasi sistem pakar perbaikan komputer yang berbentuk aplikasi pada perangkat android. Sedangkan untuk aspek *usability*, subjek penelitiannya adalah siswa kelas X jurusan teknik komputer dan jaringan (TKJ) SMK N 2 Depok. Di SMK N 2 Depok untuk jurusan TKJ kelas X terdiri dari 2 kelas, dengan total jumlah siswa kurang lebih 64 siswa. Besar jumlah sampel yang diambil untuk pengetesan aspek *usability* mengacu pada pernyataan Gay (1987:114) yang menerangkan bahwa “minimal 20% untuk populasi yang dirasa kecil”. Meski demikian jumlah 20%, masih dirasa kurang. Sehingga Pengambilan jumlah sampel mengikuti aturan *rule of thumb* yang menyatakan ukuran sampel, yaitu antara 30-500 (Guritno, Sudaryono, Raharja, 2011:159). Sehingga diambil sebanyak 32 siswa.

H. Instrumen Penelitian

Instrumen-instrumen penelitian yang digunakan dalam penelitian ini antara lain :

1. Instrumen penelitian faktor *functionality* menggunakan angket berbentuk *checklist*. Angket berisi daftar fungsi yang dimiliki oleh aplikasi
2. Instrumen penelitian faktor *portability* menggunakan beberapa perangkat nyata android dan *cloud test* dengan testdroid.
3. Instrumen penelitian faktor *usability* menggunakan angket usability yang mengacu pada *Computer System Usability Questionnaire* J.R Lewis.

Tabel 13. Kisi-kisi instrument *usability*

| Aspek | Indikator | Jumlah Soal | No. Soal |
|--------------------------|--|-------------|----------|
| <i>Operability</i> | Kemampuan perangkat lunak dalam kemudahan untuk dioperasikan | 6 | 1-6 |
| <i>Learnability</i> | Kemampuan perangkat lunak dalam kemudahan untuk dipelajari | 6 | 7-12 |
| <i>Understandability</i> | Kemampuan perangkat lunak dalam kemudahan untuk dipahami | 3 | 13-15 |
| <i>Attractiveness</i> | Kemampuan perangkat lunak dalam menarik perhatian pengguna | 4 | 16-19 |

4. Instrumen penelitian faktor *maintainability* adalah serangkaian metrik yang digunakan untuk mengetes aplikasi secara operasional. Metrik untuk faktor *maintainability* dapat dilihat pada gambar 22. Dalam mengukur tiap poin dari *source code properties* tersebut digunakan

bantuan program Eclipse dengan tambahan plugin matric dan program PMD *copy paste detector*.

5. Instrumen penelitian faktor *reliability* menggunakan program Eclipse dengan bantuan *plugin* matric untuk menghitung jumlah *line of code* (LOC). Kemudian digunakan program Findbugs untuk mengecek adanya *defect* atau *error*.

I. Teknik Analisis Data

1. Analisis Faktor *Functionality* dan *usability*

Analisis aspek *functionality* dan *usability* dilakukan dengan menggunakan teknik analisis deskriptif, yaitu menganalisis persentase hasil pengujian untuk tiap fungsi yang dilakukan oleh ahli. Sedangkan untuk *usability* dilakukan oleh target pengguna. Persentase tersebut diperoleh dengan menggunakan perhitungan sebagai berikut:

$$\text{Presentase kelayakan} = \frac{\text{Jumlah skor yang didapat}}{\text{Jumlah skor ideal}} \times 100\%$$

Hasil perhitungan kemudian dikonversi menjadi pernyataan predikat. Pernyataan predikat tersebut digunakan untuk menjelaskan kelayakan aplikasi yang dibuat. Jika hasil maksimal yang dapat dicapai adalah 100%, maka nilai tersebut dibagi rata berdasarkan 5 kategori. Pembagian kategori kelayakan dapat dilihat pada tabel 14 (Guritno, Sudaryono, Raharja, 2011:122).

Tabel 14. Interpretasi Presentase

| Nomor | Persentase | Interpretasi |
|-------|------------|--------------|
| 1 | 0% - 20% | Sangat Lemah |
| 2 | 21% - 40% | Lemah |

| Nomor | Persentase | Interpretasi |
|-------|------------|--------------|
| 3 | 41% - 60% | Cukup |
| 4 | 61% - 80% | Kuat |
| 5 | 81% - 100% | Sangat Kuat |

Supaya sesuai dengan penelitian yang dilakukan, maka pernyataan persentase pada tabel 14 perlu disesuaikan. Sehingga pernyataan persentase yang digunakan menjadi seperti pada tabel 15.

Tabel 15. Penyesuaian Interpretasi Presentase

| Nomor | Persentase | Interpretasi |
|-------|------------|--------------|
| 1 | 0% - 20% | Sangat Buruk |
| 2 | 21% - 40% | Buruk |
| 3 | 41% - 60% | Cukup |
| 4 | 61% - 80% | Baik |
| 5 | 81% - 100% | Sangat Baik |

2. Analisis Faktor *Reliability*

Analisis faktor reliability dilakukan dengan menentukan defect density pada software. Defect density dihitung dengan rumus:

$$\text{Defect Density} = \frac{\text{Number of Defects}}{\text{Size (KLOC)}}$$

Kemudian perhitungan tersebut dicocokkan dengan standar densitas error untuk setiap seribu baris kode menurut Steve McConnel pada tabel 16.

Tabel 16. Standar Defect Density

| Ukuran Project (<i>Line of Code/LOC</i>) | Density |
|--|------------------------|
| < 2K | 0-25 Defect per KLOC |
| 2K - 16K | 0-40 Defect per KLOC |
| 16K - 64K | 0.5-50 Defect per KLOC |
| 64K - 512K | 2-70 Defect per KLOC |
| > 512K | 4-100 Defect per KLOC |

BAB IV

HASIL DAN PEMBAHASAN

A. Hasil Penelitian

1. Tahap Analisis Kebutuhan

a. Analisis Kebutuhan Fungsi

Dalam menentukan fungsi-fungsi yang ada dalam sistem pakar yang dibuat, terlebih dahulu penulis melakukan observasi dan studi literatur yang terkait dengan sistem pakar. Adapun fungsi-fungsi atau fitur yang terdapat di dalam sistem pakar yang dikembang dapat dijabar sebagai berikut:

- 1) Sistem pakar memiliki 3 fungsi pokok yaitu fungsi konsultasi, pengaturan, dan bantuan.
- 2) Fungsi konsultasi dapat menampilkan daftar diagnosa kerusakan PC yang telah disediakan. Proses konsultasi dalam fungsi konsultasi memiliki konsep tanya jawab.
- 3) Fungsi konsultasi memiliki kolom untuk menampilkan pertanyaan serta hasil akhir diagnosa. Terdapat tombol Ya dan Tidak sebagai tanggapan terhadap pertanyaan yang ditampilkan.
- 4) Fungsi konsultasi memiliki fitur penjelasan untuk menjelaskan maksud dari pertanyaan yang disampaikan.
- 5) Fungsi konsultasi juga terdapat fitur untuk mengulang ke pertanyaan sebelumnya jika dirasa pertanyaan-pertanyaan yang telah diberikan dirasa tidak sesuai. Fitur tersebut yang direpresentasikan dalam bentuk sebuah tombol kembali.

- 6) Fungsi pengaturan berguna untuk memanajemen data dan menyusun data menjadi alur atau aturan diagnosa yang nantinya digunakan dalam proses konsultasi.
- 7) Fungsi pengaturan memiliki 2 opsi pokok yaitu pengaturan untuk basis pengetahuan (data) dan basis aturan.
- 8) Opsi pengaturan basis pengetahuan terdapat fitur untuk menambah data, mengubah data, melihat detail data, dan menghapus data.
- 9) Opsi pengaturan basis aturan terdapat fitur untuk menyusun aturan dari data yang ada, mengubah susunan aturan, melihat detail susunan aturan, dan menghapus aturan yang telah dibuat.
- 10) Fungsi bantuan bertujuan memberikan penjelasan tentang sistem pakar yang dibuat dan tentang cara melakukan konsultasi.

b. Analisis Kebutuhan *Hardware* dan *Software*

Dalam proses pengembangan sistem pakar yang dibuat, diperlukan beberapa hardware dan software pendukung. Adapun hardware dan software yang diperlukan antara lain :

1) Hardware

Hardware yang diperlukan dalam pengembangan sistem pakar ini antara lain:

1. Seperangkat PC atau Laptop dengan spesifikasi :
 - a. Prosessor AMD atau Intel dual core minimal 1.4 GHz.
 - b. RAM 2 GB atau lebih
 - c. Hard disk minimal 80GB.
 - d. Terdapat port USB minimal versi 2.0
2. 1 buah *smartphone* android.

3. Kabel konektor USB untuk menyambungkan perangkat *smartphone* android ke PC.

2) Software

Software yang diperlukan untuk proses pengembangan antara lain:

1. Sistem operasi baik windows atau distro linux (contoh ubuntu, debian) untuk PC.
2. JDK java minimal versi 7
3. SDK android
4. Eclipse
5. Star UML
6. SQLlite browser
7. Android minimal versi 4.0 pada perangkat *smartphone*

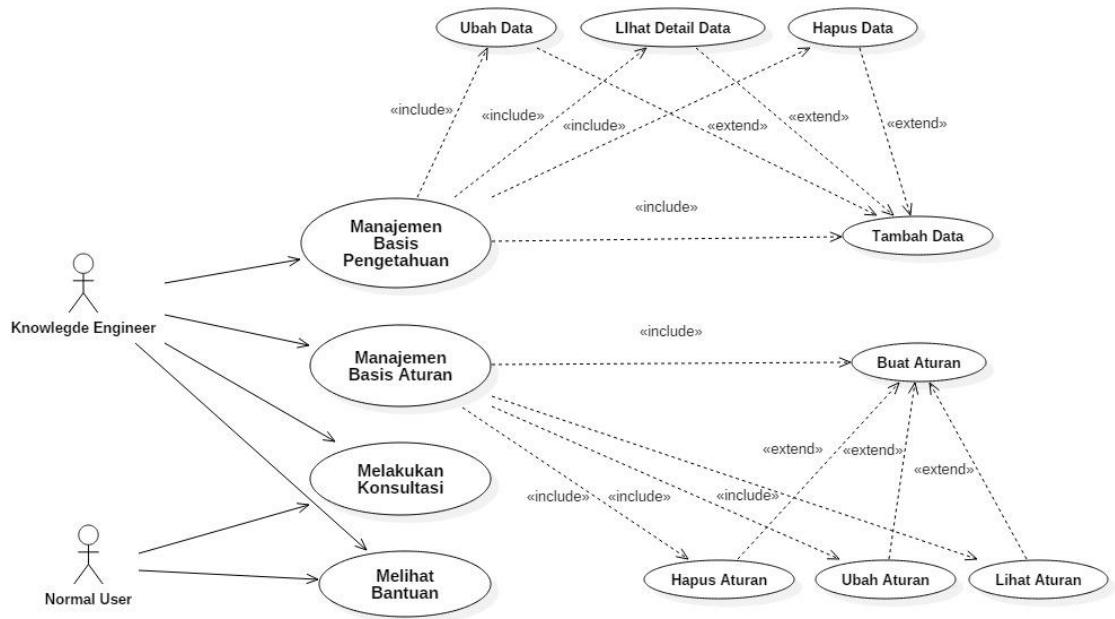
2. Desain Perangkat Lunak

a. Desain UML (*Unified Modeling Language*)

Desain UML digunakan untuk menyusun dan menggambarkan sistem pakar yang akan dibuat. Desain UML meliputi beberapa diagram yaitu diagram *use case*, diagram aktivitas, diagram sekuensial, dan diagram kelas.

1) Diagram *Use Case*

Diagram *use case* menggambarkan fungsionalitas yang disediakan oleh sistem. Adapun diagram *use case* untuk sistem pakar ini dapat dilihat pada gambar 15.



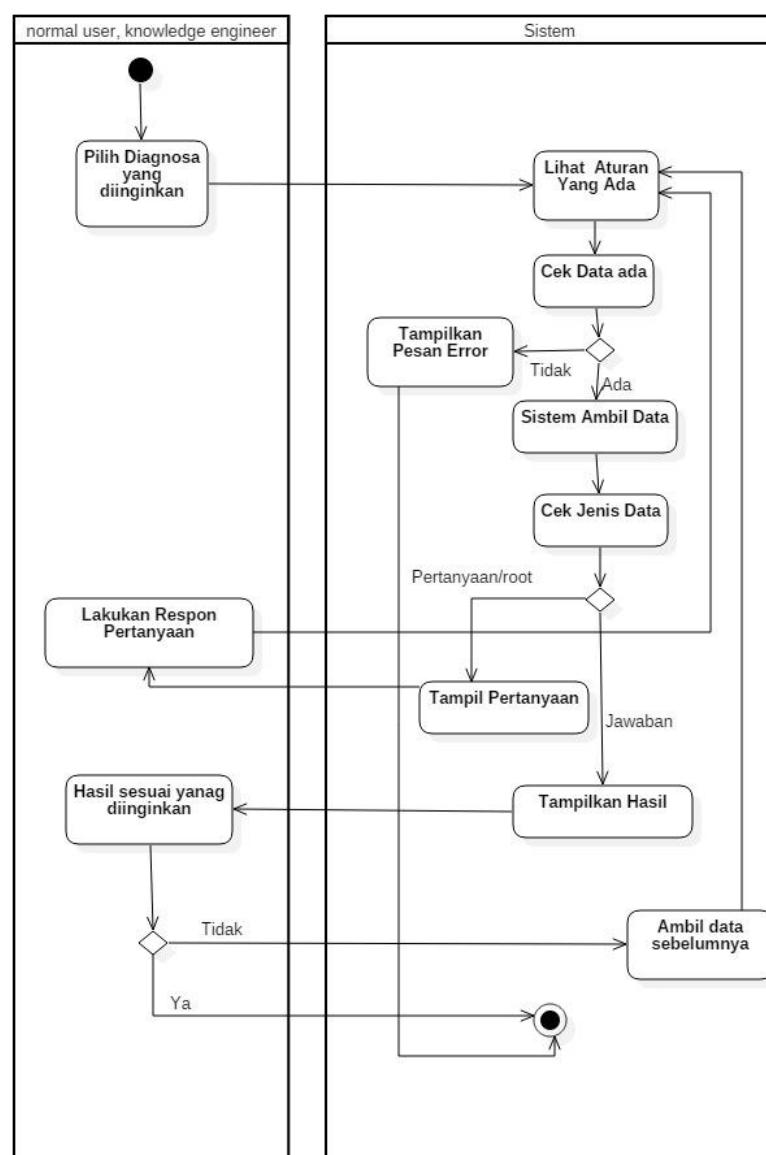
Gambar 15. Diagram use case sistem pakar perbaikan komputer berbasis android

Pada gambar 15, diagram *use case* sistem pakar mempunyai 2 aktor dan beberapa *use case* yang terlibat di dalam sistem. Aktor tersebut adalah aktor *knowlegde engineer* dan *normal user*. Adapun untuk aliran kejadian (*flow of events*) dari *use case* yang ada, akan digambarkan pada diagram aktivitas. Sedang untuk definisi aktor yang ada adalah sebagai berikut :

- *Knowlegde engineer*, merupakan aktor yang terlibat dalam sistem dan bertugas untuk memanajemen data dari sumber pengetahuan yang ada dan menyusunnya menjadi sebuah aturan atau alur konsultasi.
- *Normal user*, merupakan aktor yang terlibat dalam sistem dan menggunakan sistem pakar yang ada hanya bertujuan untuk melakukan konsultasi pada sistem.

2) Diagram Aktivitas

Diagram aktivitas pada desain UML bertujuan untuk menggambarkan aliran kejadian yang ada pada *use case*. Terdapat beberapa diagram aktivitas yang dibuat pada desain UML ini (**daftar lengkap dapat dilihat pada lampiran**), sebagai contoh sebagaimana yang dapat dilihat pada gambar 16.



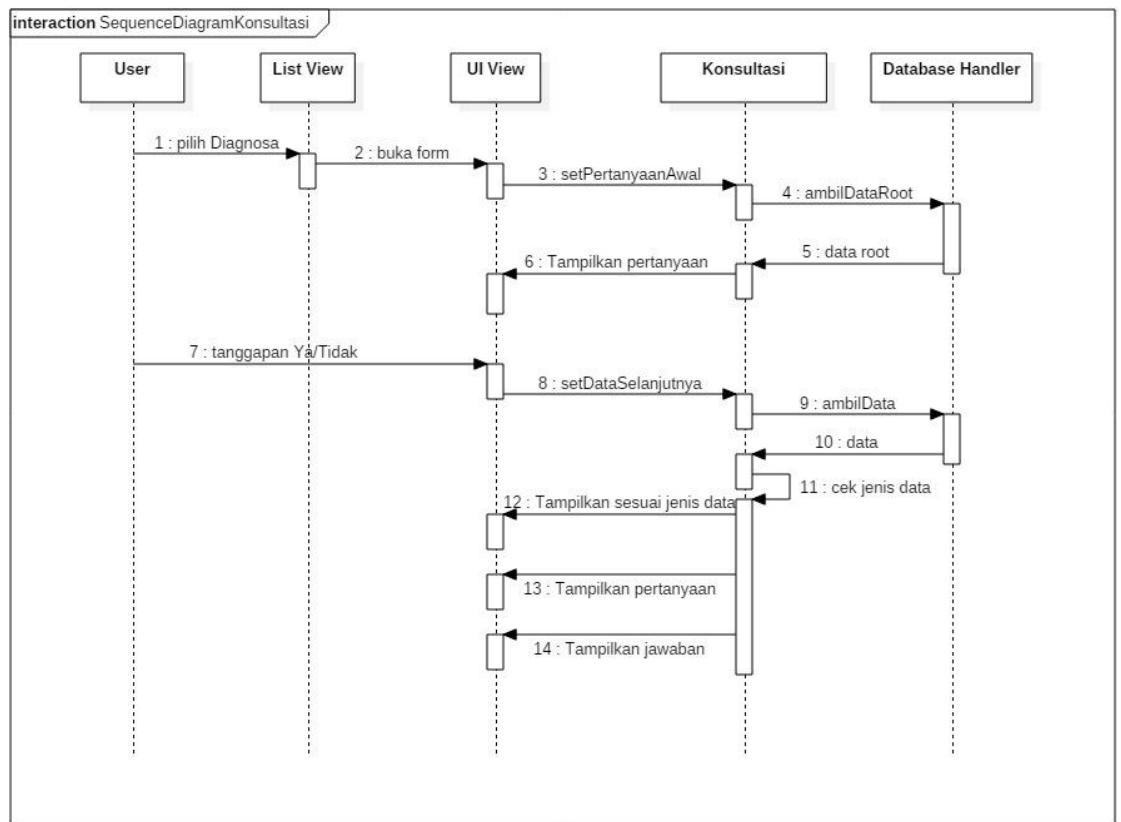
Gambar 16. Diagram aktivitas melakukan konsultasi.

Aliran kejadian diagram aktivitas melakukan konsultasi pada gambar 16 dapat dijabarkan sebagai berikut :

- a. Pengguna mula-mula memilih proses diagnosa yang diinginkan.
- b. Sistem kemudian akan melihat aturan yang sudah ada.
- c. Sistem akan mengecek keberadaan data yang dimaksud dalam aturan. Bila tidak terdapat data yang dimaksud sistem akan menampilkan pemberitahuan.
- d. Sistem akan mengambil data yang dimaksud.
- e. Sistem akan mengecek jenis data yang dimaksud apakah meruoakan sebuah pertanyaan atau jawaban.
- f. Bila data adalah sebuah pertanyaan, sistem akan menampikannya dan mengijinkan pengguna untuk menanggapinya.
- g. Prosedur pada poin b sampai dengan f akan berulang hingga jawaban ditemukan.
- h. Alternaif scenario, bila jawaban yang dihasilkan tidak sesuai. Pengguna dapat kembali ke pertanyaan-pertanyaan sebelumnya dan mencoba alur yang berbeda.

3) Diagram Sekuensial

Diagram sekuensial pada desain UML disusun untuk menerangkan hubungan objek-objek yang ada pada suatu *use case*. Terdapat beberapa diagram sekuensial yang disusun pada desain UML ini (**daftar lengkap dapat dilihat pada lampiran**). Sebagai contoh sebagaimana dapat dilihat pada gambar 17.

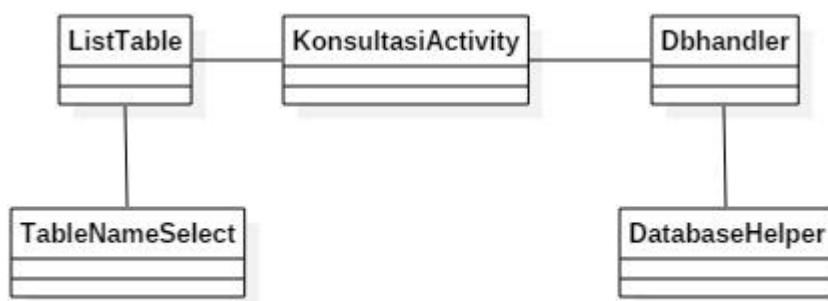


Gambar 17. Diagram sekuensial melakukan konsultasi

Diagram sekuensial pada gambar 17 digunakan untuk menggambarkan objek-objek yang ada untuk menjalankan *use case* melakukan konsultasi. Objek *list view* merupakan objek yang bertugas untuk menampilkan daftar proses diagnosa-diagnosa yang disediakan oleh aplikasi sistem pakar. Ketika user memilih daftar diagnosa yang diinginkan sistem akan memunculkan objek *ui view* yang bertugas sebagai antar muka pada proses konsultasi. Objek *konsultasi* dan *database handler* bertugas untuk menangani proses dilakukannya konsultasi untuk mendiagnosa sumber kerusakan pada PC.

4) Diagram Kelas

Diagram kelas disusun untuk menggambarkan hubungan antar kelas yang ada pada desain UML. Pada desain UML untuk sistem pakar ini terdapat beberapa diagram yang dibuat (**daftar lengkap ada dilampiran**). Contoh diagram kelas yang ada dapat dilihat pada gambar 18.



Gambar 18. Diagram kelas untuk melakukan konsultasi

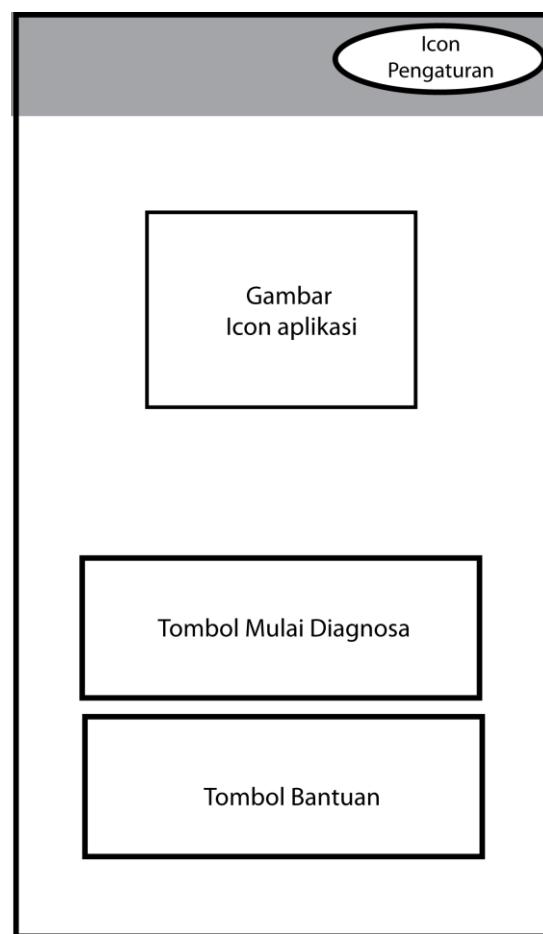
Gambar 18 menunjukkan hubungan dari kelas-kelas yang terbentuk untuk melakukan mekanisme melakukan konsultasi. Terdapat beberapa kelas yang saling terhubung yaitu kelas TableNameSelect, ListTable, KonsultasiActivity, Dbhandler, DatabaseHelper.

b. Desain *User Interface*

Desain *user interface* pada proses pengembangan ini disusun sebagai panduan untuk membuat tampilan sistem pakar nantinya. Terdapat beberapa desain user interface yang dibuat untuk masing-masing tampilan fungsi pada aplikasi sistem pakar ini. Meski pun demikian terdapat juga 1 desain user interface yang dipakai untuk beberapa fungsi berbeda.

1) Desain Halaman Utama

Desain *user interface* untuk halaman utama untuk aplikasi sistem pakar ini dapat dilihat pada gambar 19. Konsep awal dari desain *user interface* sistem pakar yang dibuat adalah sederhana dan langsung ke tujuan. Pada halaman utama dari sistem pakar ini terdapat 2 buah tombol yang nantinya dipakai untuk tombol mulai diagnosa dan menu bantuan. Selain itu terdapat ikon sekaligus juga berfungsi sebagai tombol untuk mengarahkan pengguna ke menu pengaturan. Terdapat pula logo aplikasi yang diletakkan dihalaman utama.



Gambar 19. Desain *user interface* untuk halaman utama

2) Desain List Menu dan Item

List merupakan fitur yang banyak digunakan dalam desain user interface. List digunakan untuk menampilkan menu dan daftar item, diantaranya untuk menu pilihan diagnosa, menu-menu pada opsi pengaturan, menu untuk pilihan info bantuan, serta daftar data dan node-node aturan. Desain untuk list menu dan item dapat dilihat pada gambar 20.



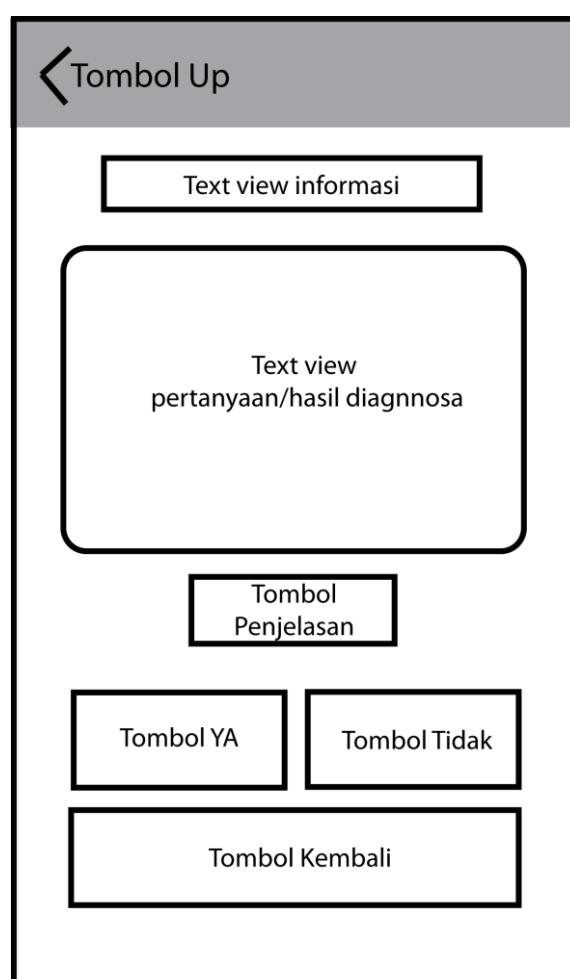
Gambar 20. Desain *user interface* list menu dan item

3) Desain Halaman Konsultasi

Halaman konsultasi pada aplikasi sistem pakar ini merupakan salah satu bagian yang penting. Oleh sebab itu pengembang mendesain agar halaman konsultasi dapat dipahami semudah mungkin pada layar

smartphone yang terbatas sebagaimana yang dapat dilihat pada gambar 21.

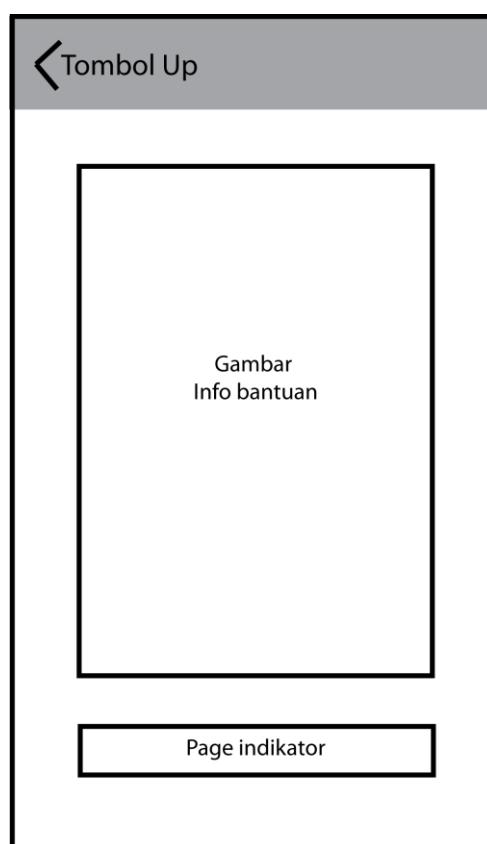
Pada halaman konsultasi terdapat *text view* untuk menampilkan pertanyaan dan hasil diagnosa. Terdapat pula 4 buah tombol, masing-masing untuk memunculkan penjelasan, tanggapan ya, tanggapan tidak, serta untuk proses kembali ke pertanyaan sebelumnya. Tombol kembali tidak akan aktif pada saat pertanyaan awal muncul dan baru aktif bila sudah menuju ke pertanyaan selanjutnya. Ketika hasil diagnosa muncul maka tombol tanggapan ya dan tidak akan dinonaktifkan.



Gambar 21. Desain *user interface* halaman konsultasi

4) Desain Halaman Bantuan

Halaman bantuan berguna untuk memberitahu pengguna tentang bagai mana cara menggunakan aplikasi sistem pakar ini. Info bantuan akan ditampilkan dalam bentuk *slide show* gambar cara pengoperasian. Desain user interface untuk halaman bantuan dapat dilihat pada gambar 22. Pada desain terdapat ruang yang cukup lebar untuk menampilkan gambar. Selain itu terdapat indicator halaman untuk mengetahui posisi tampilan dan jumlah gambar pada *slide show*.

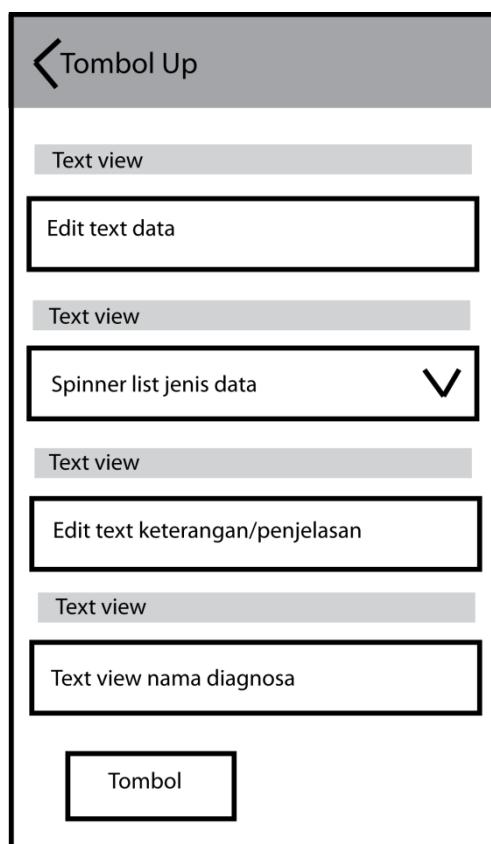


Gambar 22. Desain halaman info bantuan

5) Desain Halaman Tambah dan Ubah Data

Desain untuk halaman tambah dan ubah data menggunakan sedain yang sama. Hal tersebut dikarenakan kebutuhan yang diperlukan tidak

berbeda dalam memanipulasi data. Terdapat 2 buah form *edit text* untuk menulis pertanyaan atau hasil diagnose dan penjelasannya. Terdapat pula form *spinner list* untuk memilih tipe data yang akan disimpan. Selanjutnya terdapat *text view* untuk menampilkan pada proses diagnose mana data akan disimpan. Serta sebuah tombol untuk melakukan konfirmasi penyimpanan. Desain untuk halaman tambah dan ubah data dapat dilihat pada gambar 23.

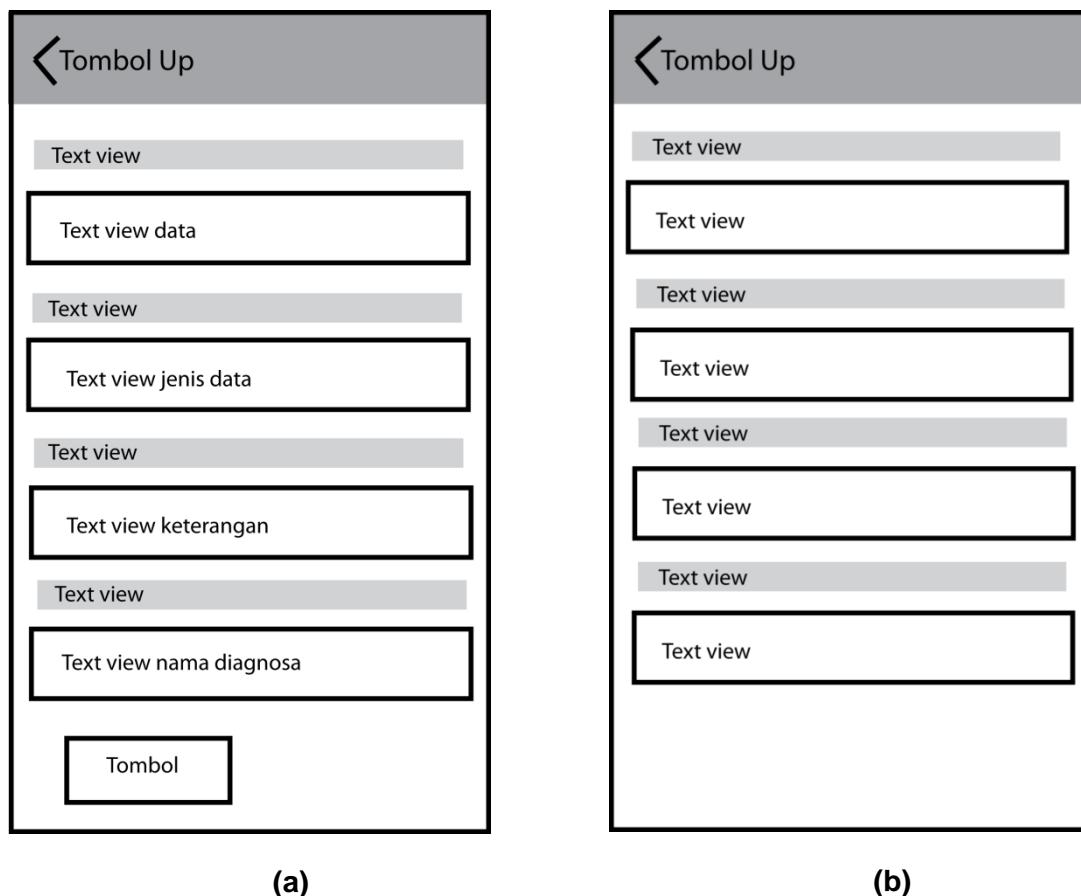


Gambar 23 . Desain *user interface* halaman tambah dan ubah data

6) Desain Halaman Detail dan Hapus Data'

Halaman detail dan hapus data memiliki desain yang hampir serupa. Terdapat beberapa *text view* pada layout desain untuk menampilkan isi data yang bersangkutan. Masing-masing *text view* berfungsi untuk menampilkan isi pertanyaan, jenis tipe data yang disimpan, isi penjelasan, dan info dimana

data tersimpan. Hal pokok yang membedakan desain detail dan hapus data adalah pada desain halaman hapus data terdapat tombol untuk melakukan konfirmasi proses penghapusan data, sedang pada halaman detail data tidak ada terdapat tombol. Desain user interface detail dan hapus data dapat dilihat pada gambar 24.

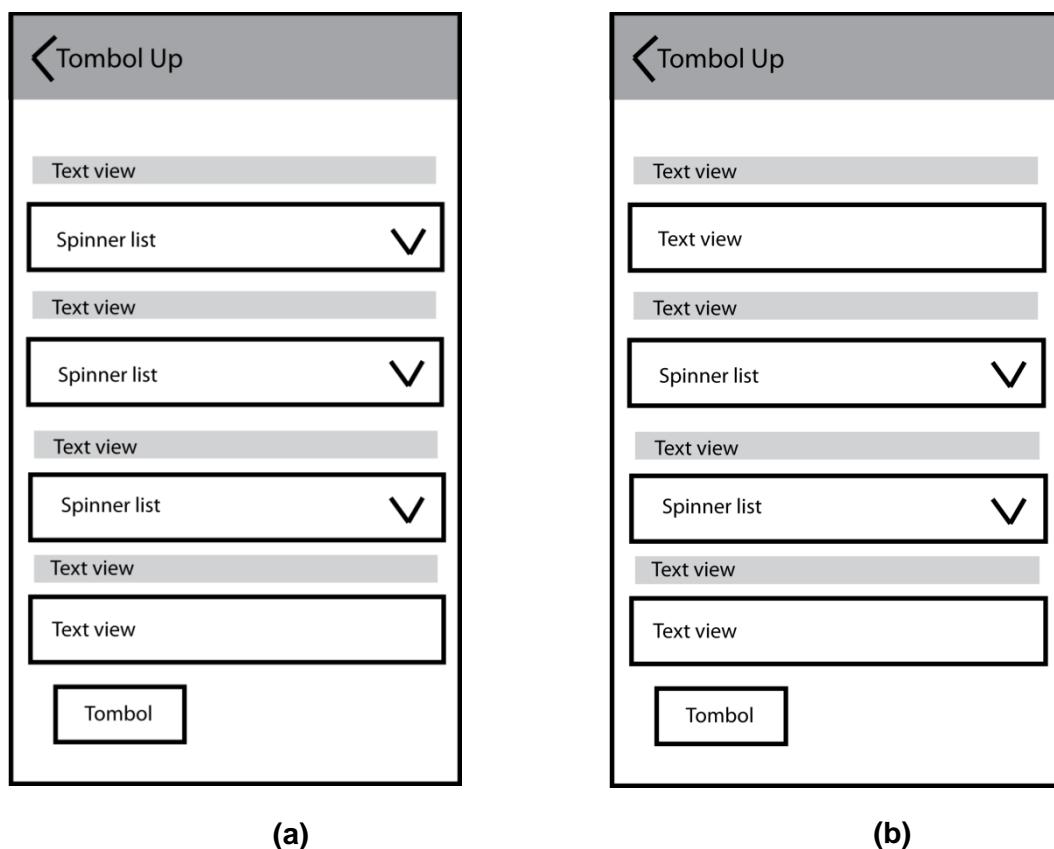


Gambar 24. Desain *user interface* hapus data (a) dan detail data (b)

7) Desain Halaman Buat dan Ubah Node Aturan

Desain untuk halaman membuat dan mengubah node aturan memiliki desain yang hamper mirip. Pada desain untuk membuat node aturan, terdapat 3 buah *spinner list* yang nantinya berfungsi menentukan data mana yang akan digunakan sebagai node induk dan 2 data untuk node anak.

Selain itu terdapat tombol untuk melakukan konfirmasi penyimpanan. Sedangkan pada desain halaman untuk mengubah node aturan hanya terdapat 2 buah *spinner list* untuk mengubah isi dari 2 node anak. Sedangkan untuk node induk sebagai gantinya menggunakan *text view* untuk menampilkan isinya. Desain selengkapnya untuk halaman buat dan ubah node aturan dapat dilihat pada gambar 25. Terdapat pula sebuah *text view* untuk menampilkan diproses diagnose mana node aturan disimpan.

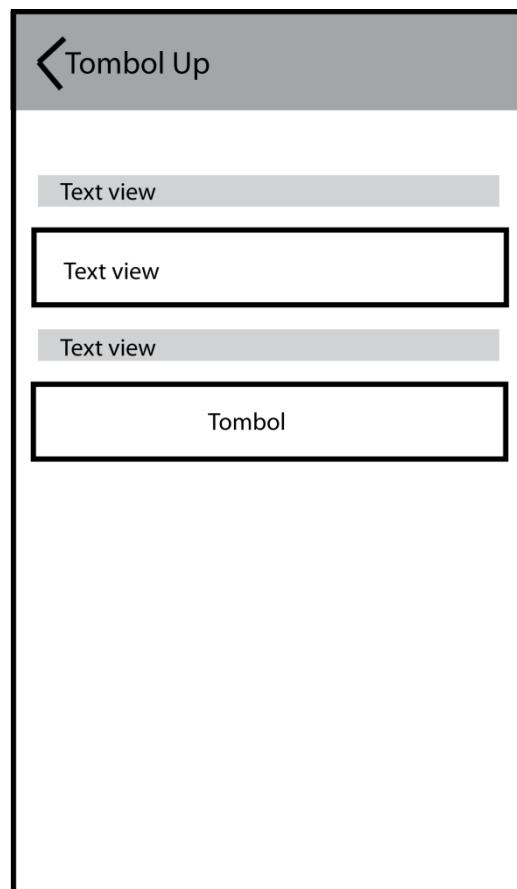


Gambar 25. Desain *user interface* halaman buat node aturan (a) dan ubah node aturan (b)

8) Desain Halaman Hapus Node Aturan

Halaman hapus node aturan memiliki desain yang sederhana. Hal tersebut terkait fungsi halaman tersebut adalah untuk menghapus seluruh

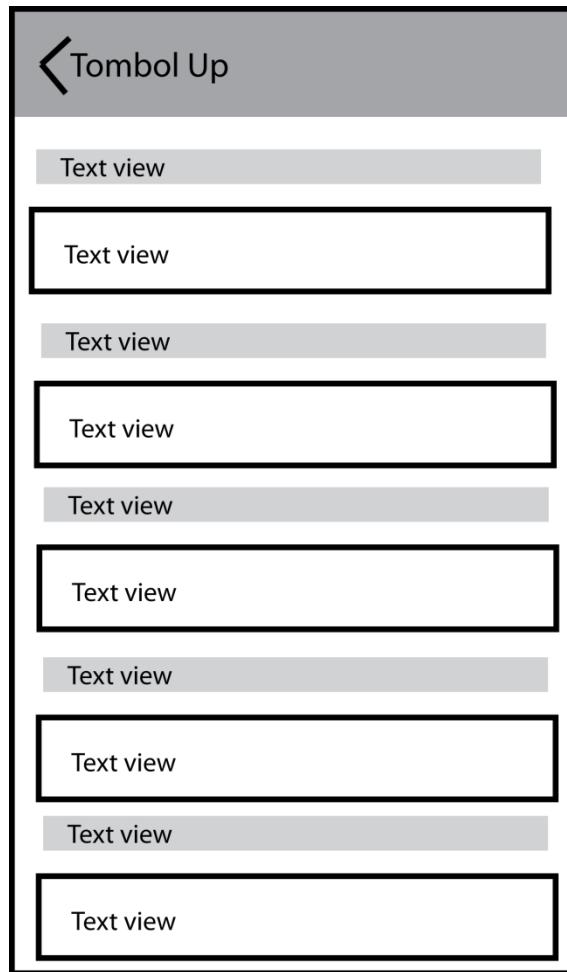
node yang terhubung pada proses diagnose yang ada. Terdapat 1 buah *text view* pokok untuk menampilkan proses diagnose mana yang akan dihapus. Serta 1 tombol untuk melakukan konfirmasi proses penghapusan. Desain selengkapnya dapat dilihat pada gambar 26.



Gambar 26. Desain *user interface* halaman hapus node aturan

9) Desain Halaman Detail Node Aturan

Desain halaman detail node aturan menampilkan daftar detail data yang digunakan pada node tersebut. Tampilan halaman ini berupa daftar *text view* yang nantinya akan diisi dengan detail masing-masing data. Desain *user interface* untuk halaman detail node aturam dapat dilihat pada gambar 27.



Gambar 27. Desain *user interface* detail node aturan

c. Desain Database

Desain database dibuat untuk menentukan rancangan database yang akan digunakan nanti. Database diperlukan sebagai sarana penyimpanan data dan susunan node aturan yang ada. Database yang akan digunakan pada aplikasi sistem pakar ini adalah SQLite. Pemilihann SQLite sebagai jenis database yang digunakan Karena android memiliki dukungan yang baik terhadap SQLite. Selain itu SQLlite termasuk database yang ringan, sehingga cocok digunakan untuk keperluan pembuatan aplikasi pada perangkat *smartphone*. Desain untuk database yang dbuat dapat dilihat pada tabel 17.

Tabel 17. Desain database pada SQLite

| Field name | Type | Null | Primary Key | Autoincrement |
|------------|---------|------|-------------|---------------|
| _id | Integer | No | Yes | Yes |
| fakta | Text | Yes | No | No |
| penjelasan | Text | Yes | No | No |
| kode | Text | Yes | No | No |
| Ya | Integer | Yes | No | No |
| tidak | Integer | Yes | No | No |

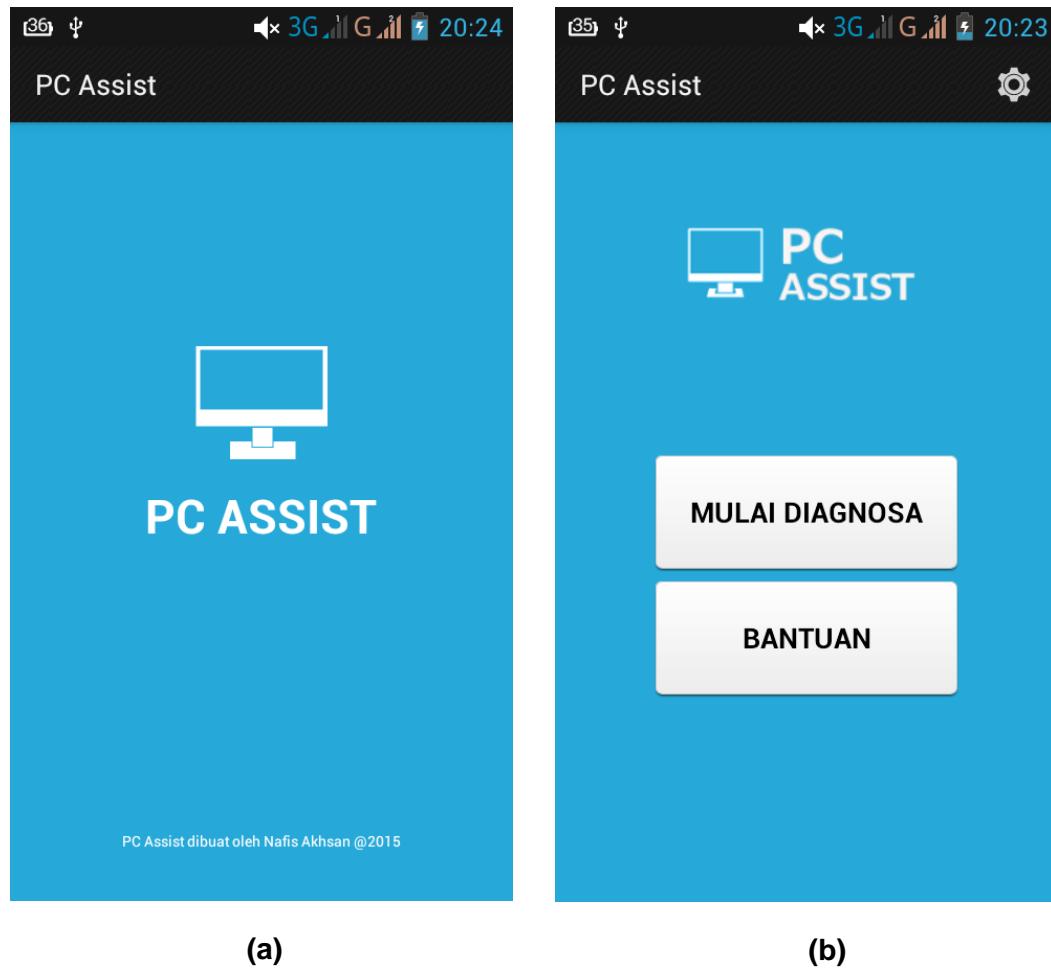
3. Implementasi

Proses selanjutnya pada tahapan pengembangan aplikasi ini adalah implementasi. Pada proses implementasi, aplikasi telah mulai dikerjakan berdasarkan desain yang telah dibuat sebelumnya. Proses pembuatan berlangsung sampai fungsi-fungsi pada aplikasi siap untuk digunakan dan diuji. Berikut proses hasil dari proses implementasi pada aplikasi sistem pakar.

a. Halaman Awal

Aplikasi sistem pakar ini dibuka dengan menekan logo dari aplikasi ini pada perangkat android. Setelah dibuka akan muncul tampilan *splash screen* seperti yang ditunjukan pada gambar 28. *Splash Screen* akan muncul selama beberapa detik, kemudian akan secara otomatis dibawa ke halaman utama aplikasi ini. Halaman utama seperti yang dapat dilihat pada gambar 28, terdapat tombol untuk memulai proses diagnose dan melihat bantuan yang tersedia. Bila tombol mulai diagnose di tekan aplikasi akan akan membuka daftar proses diagnosa yang tersedia. Bila tombol bantuan ditekan aplikasi akan membuka daftar info bantuan yang tersedia. Selain itu pada pojok

kanan atas terdapat icon berbentuk roda gigi yang berfungsi sebagai tombol yang bila ditekan akan menuju ke menu pengaturan.

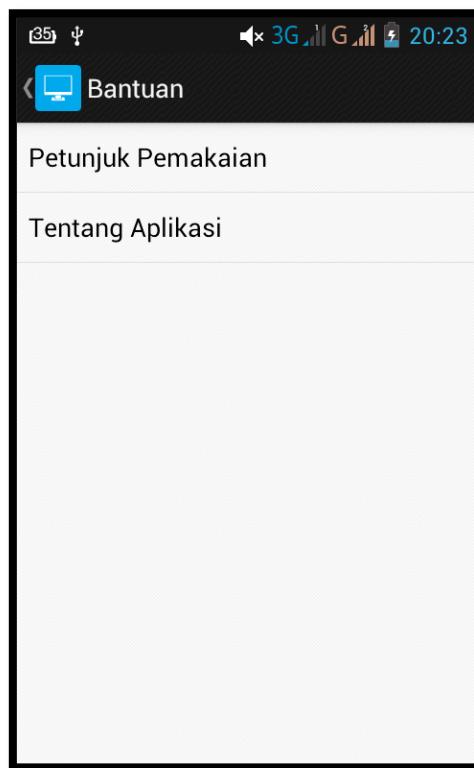


Gambar 28. *Splash Screen* (a) dan halaman utama (b)

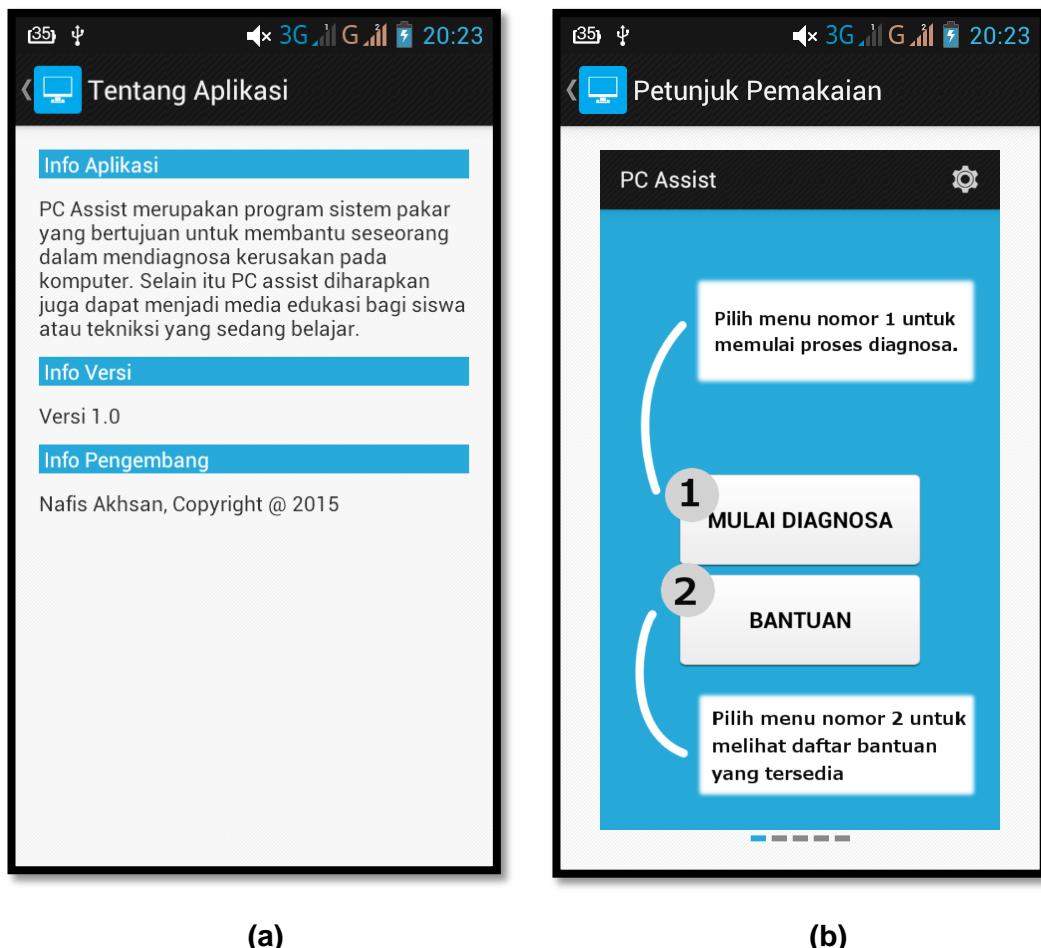
b. Halaman Info Bantuan

Halaman bantuan berisi informasi yang dapat digunakan oleh pengguna untuk mempermudah dalam hal pemakaian aplikasi. Halaman bantuan diakses menggunakan tombol yang ada pada halaman utama. Terdapat dua menu bantuan yaitu tentang aplikasi dan petunjuk pemakaian (gambar 29). Menu tentang aplikasi bila dipilih selanjutnya akan menampilkan informasi tentang tujuan sistem pakar ini dan info pengembang (gambar 30). Sedangkan menu petunjukan pemakaian menampilkan cara pengguna untuk

melakukan proses konsutasi diagnose (gambar 30). Halaman petunjuk pemakaian merupakan tampilan berupa *slide show* gambar cara pemakaian. Proses untuk melihat gambar yang ada adalah dengan menggeser jari ke kiri atau kanan pada layar perangkat *smartphone*. Selain itu terdapat indicator halaman untuk mengidikasikan pada gambar keberapa yang berada di layar. indicator halaman berupa titik-titik dibawah gambar yang ikut bergerak bila gambar di geser.



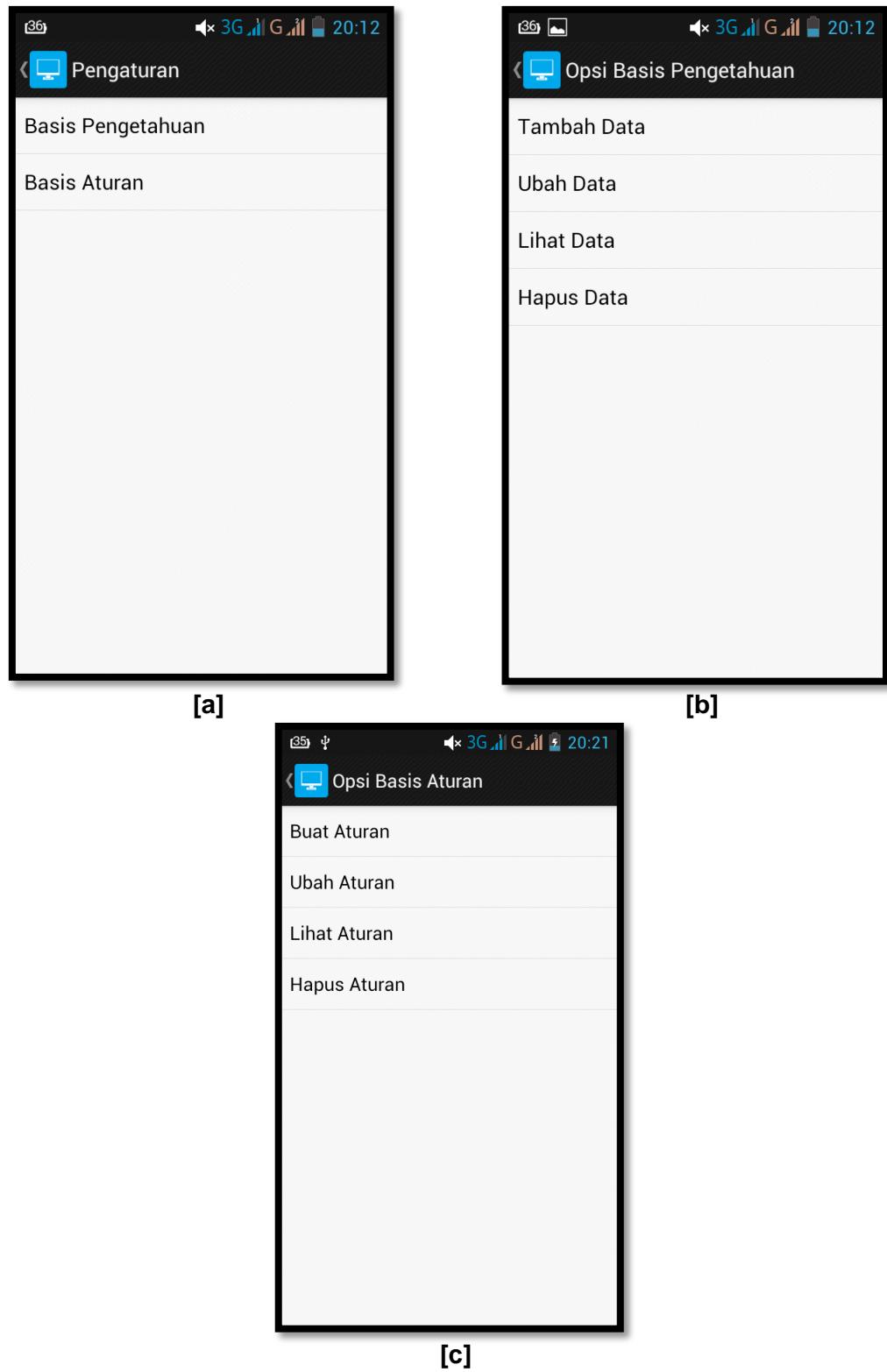
Gambar 29. List menu halaman bantuan



Gambar 30. Halaman tentang aplikasi (a) dan petunjuk pemakaian (b)

c. Halaman Pengaturan

Halaman pengaturan diakses melalui tombol ikon pengaturan dari halaman utama. Terdapat 2 menu pokok untuk pengaturan, yaitu pengaturan untuk basis data atau pengetahuan dan basis aturan. Pengaturan basis pengetahuan berisi kumpulan data-data pertanyaan dan hasil diagnosa yang akan digunakan untuk menyusun node-node aturan. Pengaturan basis aturan berisi hubungan antara data-data yang ada, disusun menjadi sebuah pohon aturan yang nantinya digunakan untuk proses konsultasi. Tampilan selengkapnya terkait halaman pengetaran dapat dilihat pada gambar 31.



Gambar 31. Halaman pengaturan [a], halaman opsi basis pengetahuan [b], halaman opsi basis aturan [c]

d. Halaman Tambah Data

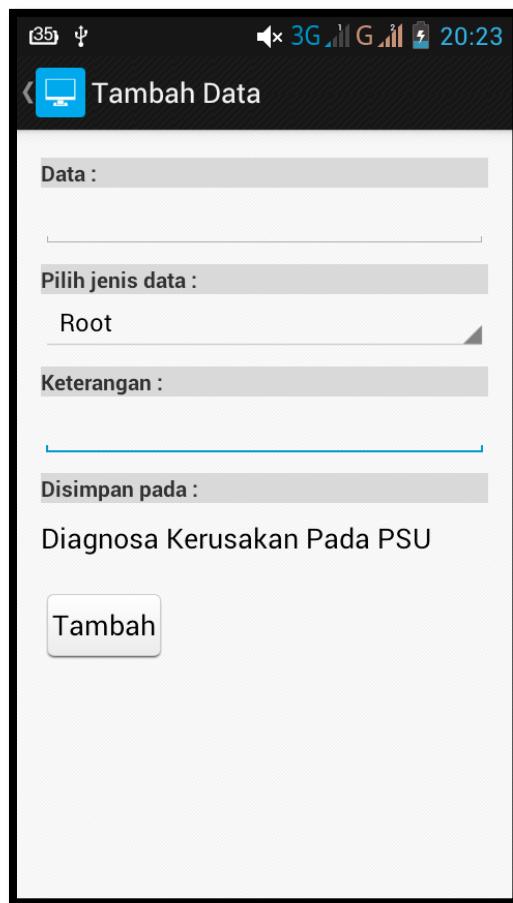
Halaman tambah data berfungsi untuk memasukan data baik sebagai pertanyaan atau hasil diagnosa. Sebelum memasuki halaman tambah data, terlebih dulu memilih pada proses diagnosa mana nantinya data akan disimpan pada list menu daftar diagnosa (gambar 32).



Gambar 32. List menu proses diagnose yang disediakan

Di halaman tambah data (gambar 33), pokok data ditulis pada kolom data, kemudian memilih tipe data yang sesuai. Tipe data root merupakan pertanyaan yang akan ditampilkan pertama kali pada saat membuka proses diagnosa. hanya ada 1 data root yang diperbolehkan. Tipe data lain adalah pertanyaan dan jawaban. Pertanyaan merupakan percabangan pada suatu pohon keputusan. Sedang jawaban merupakan akhir dari setiap pohon keputusan yang berupa hasil diagnosa. terdapat juga kolom keterangan yang merupakan penjelasan lebih lanjut dari data yang dimasukkan. Bila

semua kolom dirasa sudah benar kemudian menekan tombol tambah untuk menyimpan data ke database.

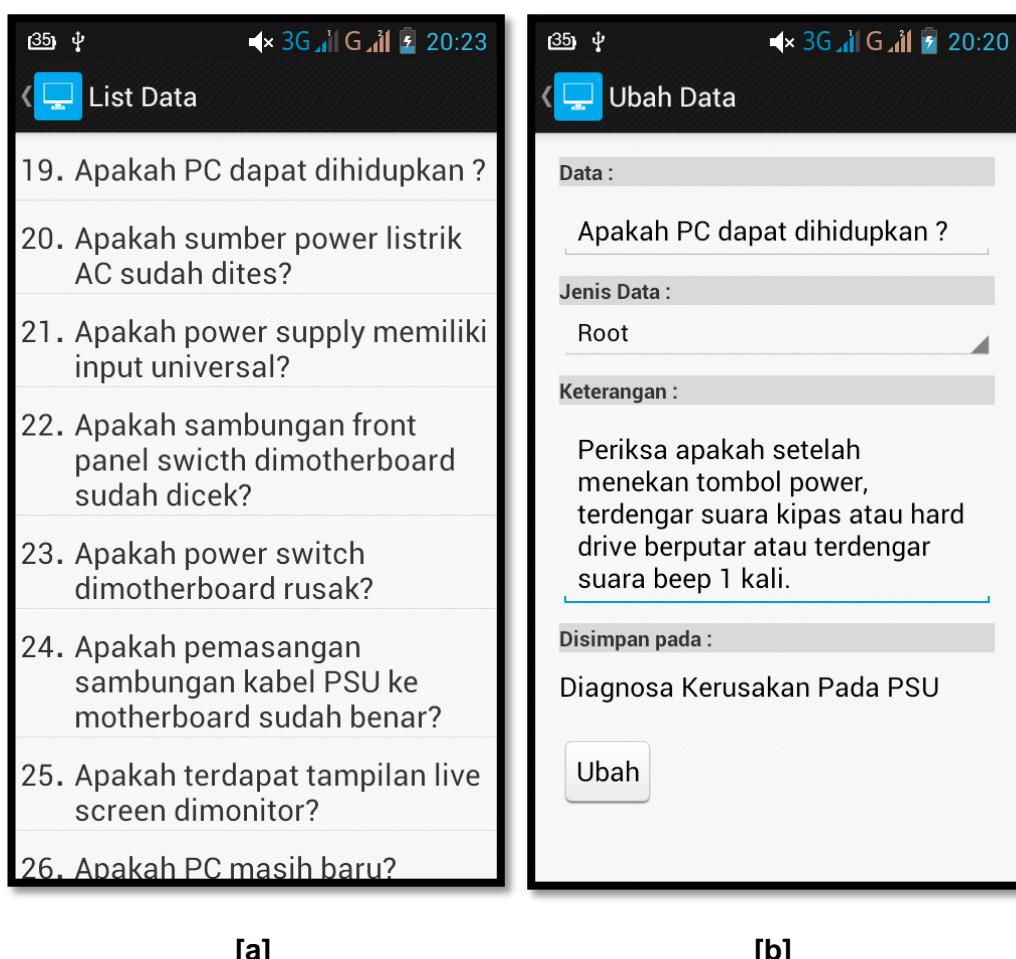


Gambar 33. Halaman tambah data

e. Halaman Ubah Data

Halaman ubah data berfungsi untuk melakukan perubahan terhadap data yang telah disimpan. Akses ke halaman ubah data pertama-tama dengan memilih menu ubah data pada list menu opsi basis pengetahuan. Kemudian memilih pada proses diagnosa mana data yang ada disimpan (gambar 32). Aplikasi kasi kemudian akan menampilkan daftar data yang telah disimpan (gambar 34). Pilih satu data yang akan diedit kemudian halaman edit data akan muncul. Lakukan perubahan yang diperlukan pada

kolom data, jenis data, atau keterangan. Setelah itu pilih tombol ubah untuk menyimpan perubahan. Dalam proses melakukan perubahan terhadap data yang ada, mengubah jenis data dapat mempengaruhi susunan pada pohon keputusan bila data tersebut telah dipakai. Sebagai contoh mengubah jenis data pertanyaan menjadi jawaban dapat memutus hubungan dengan data yang tersambung dibawahnya di pohon keputusan. Oleh sebab itu perlu dilakukan perhitungan terlebih dulu sebelum melakukan perubahan.



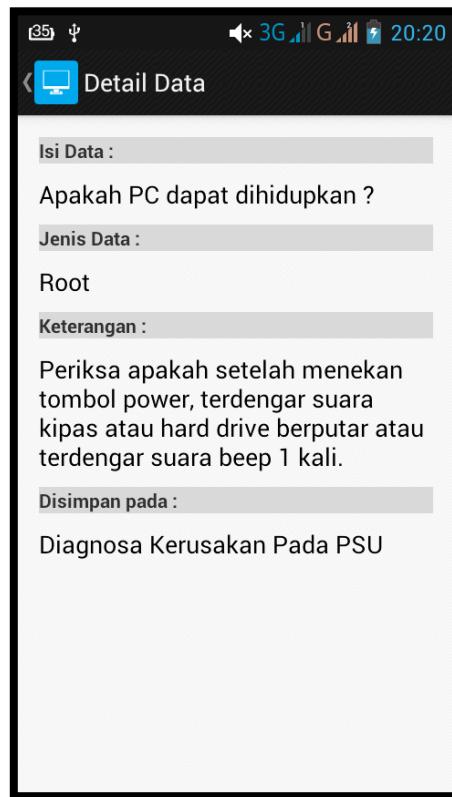
Gambar 34. list data yang telah dibuat [a] dan halaman ubah data [b]

f. Halaman Detail Data

Halaman detail data berfungsi utnuk melihat isi data yang tersimpan.

Pertama pilih menu lihat data pada opsi basis pengetahuan. Pilih proses

diagnosa mana yang ingin dilihat datanya. Aplikasi kemudian mengeluarkan list data yang telah tersimpan (gambar 34). Pilih salah satu data yang diinginkan untuk melihat detailnya. Halaman detail data akan muncul dan menampilkan isi data berupa informasi detail data pokok, jenis data yang dipakai, keterangan, dan dimana data tersimpan (gambar 35).

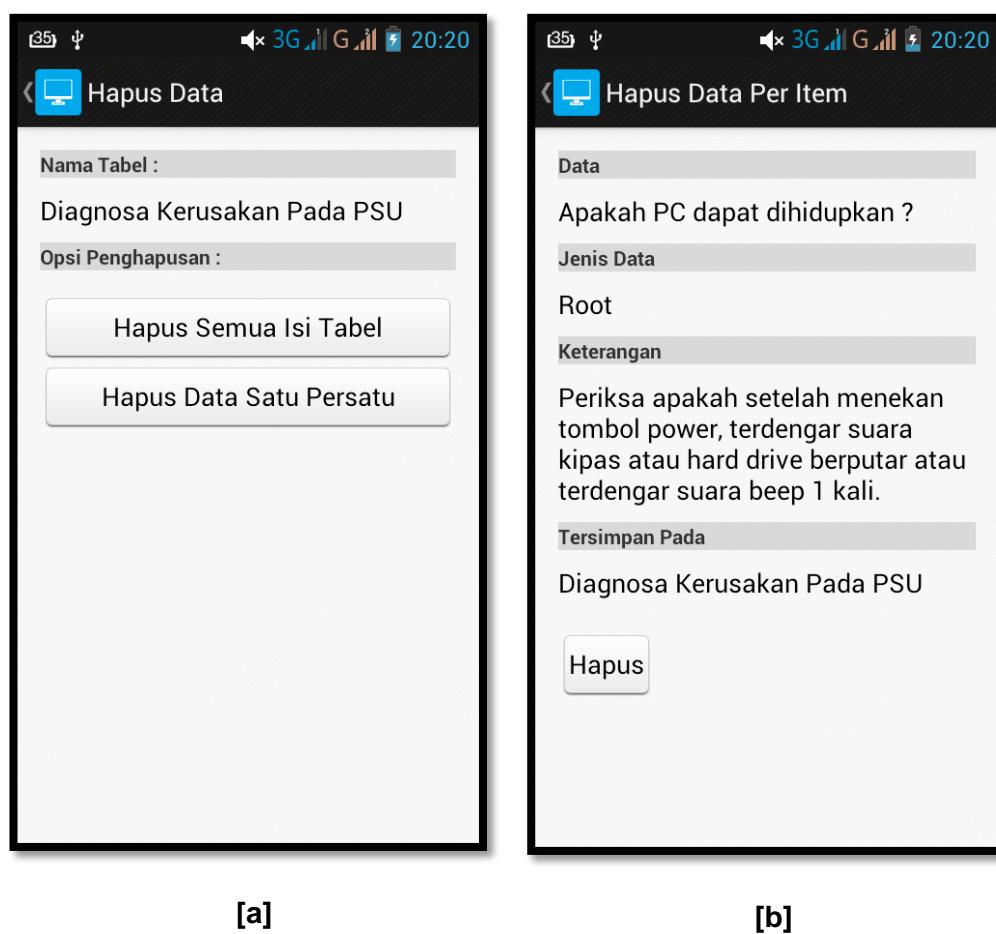


Gambar 35. halaman detail data

g. Halaman Hapus Data

Halaman hapus data berfungsi untuk menghapus data yang sekiranya tidak diperlukan lagi. Halaman ini dapat diakses melalui menu hapus data kemudian memilih proses diagnosa mana data yang akan dihapus berada. Terdapat 2 pilih dalam menghapus data, yaitu menghapus seluruh data yang ada dan menghapus satu data saja (gambar 36). Ketika data yang ada telah dipakai dalam pohon keputusan, menghapus data dapat memutus hubungan

dalam pohon keputusan tersebut. Oleh sebab itu perlu dipikirkan sebaik mungkin bila akan melalukan penghapusan. Dalam opsi menghapus satu data, aplikasi sebelumnya akan menampilkan daftar semua data yang ada dip roses diagnosa bersangkutan untuk nantinya dipilih mana yang akan dihapus. Aplikasi selanjutnya akan menampilkan detail data yang dipilih guna memastikan bahwa data yang dipilih sudah sesuai yang diinginkan.

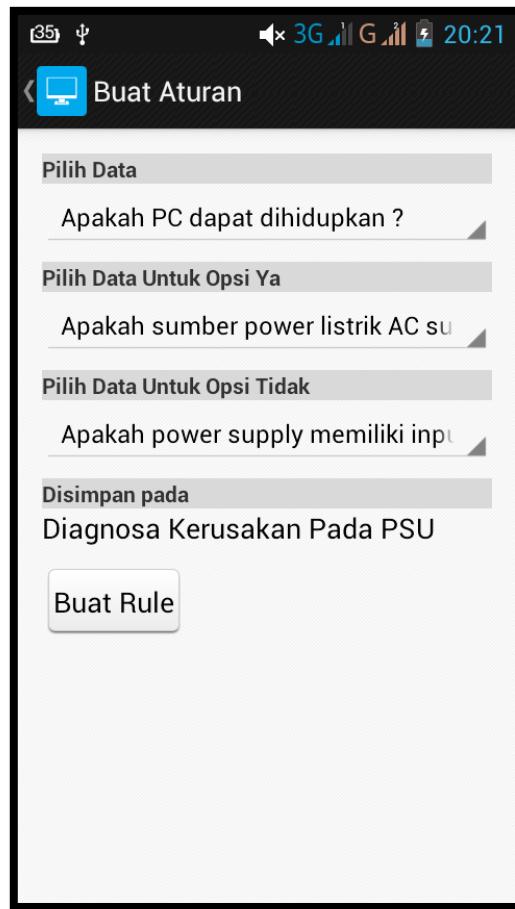


Gambar 36. opsi penghapusan [a] dan hapus 1 buah data [b]

h. Halaman Buat Aturan

Halaman buat aturan (gambar 37) berfungsi untuk menghubungkan data-data yang ada menjadi sebuah pohon aturan. Proses pembuatan pohon aturan dilakukan dengan menghubungkan tiap-tiap 3 buah data. 1 data untuk

node induk dan 2 data untuk node anak. Demikian seterusnya hingga pohon keputusan yang diinginkan terbentuk. Dalam membentuk hubungan dari 3 buah data diharuskan menggunakan data yang berbeda. Bila terdapat data yang sama hubungan tidak dapat terbentuk.

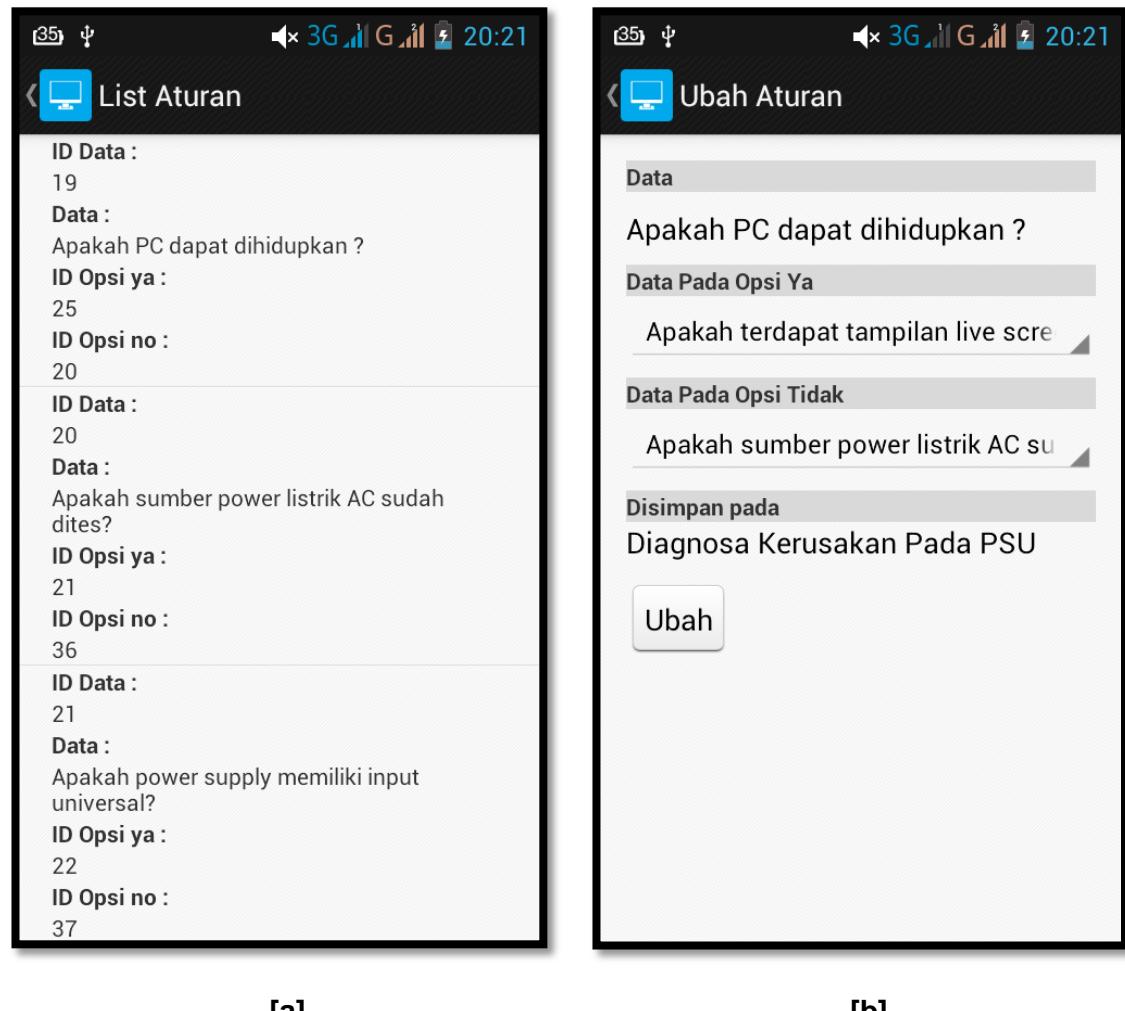


Gambar 37. Halaman buat aturan

i. Halaman ubah aturan

Halaman ubah aturan berfungsi untuk mengubah hubungan dari data-data yang telah disusun. Aplikasi pertama akan menampilkan list node-node yang terhubung pada proses diagnosa yang dipilih. Pada proses mengubahan data yang boleh diubah hanya 2 node anak saja. Sedangkan

untuk node induk tetap. Sama halnya dengan proses membuat hubungan pada halaman buat aturan, dalam mengubah hubungan tidak tiperbolehkan untuk memilih data yang sama untuk node yang berbeda. Tampilan selengkapnya untuk halaman mengubah aturan dapat dilihat pada gambar 38.



[a]

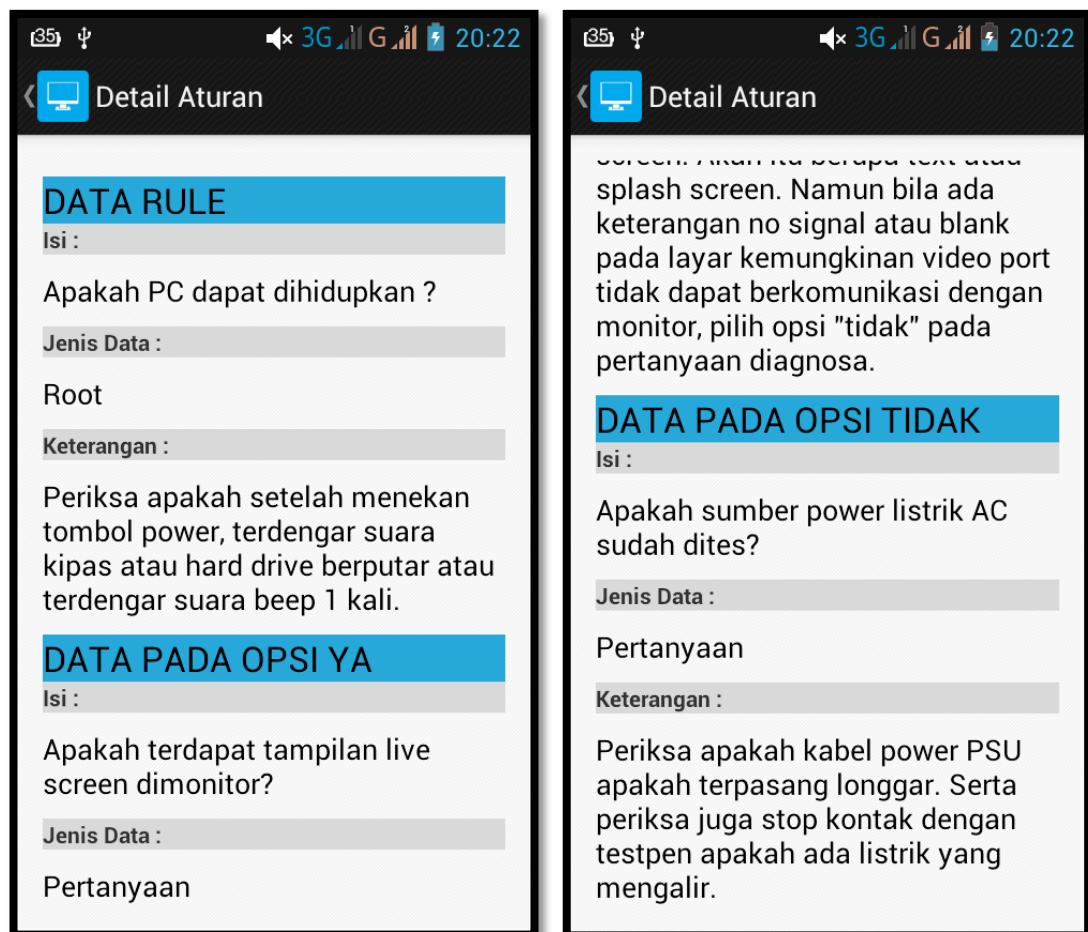
[b]

Gambar 38. List hubungan node yang ada [a] dan halaman untuk mengubah aturan [b]

j. Halaman Detail Aturan

Halaman detail aturan (gambar 39) berfungsi untuk menampilkan detail dari data-data yang terhubung. Halaman ini akan menampilkan detail dari

data pada node induk dan detail dari data-data pada 2 node anak yang terhubung. Aplikasi pertama akan menampilkan list hubungan node-node yang ada pada proses diagnose yang dipilih. Halaman detail aturan akan muncul pilih satu potong node dipilih dari list yang ada.

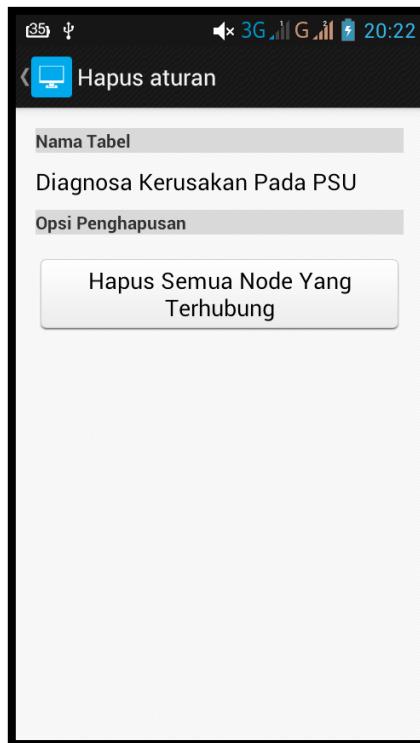


Gambar 39. Detail node aturan yang terhubung

k. Halaman Hapus Aturan

Halaman hapus aturan (gambar 40) berfungsi untuk menghapus node-node yang terhubung dalam sebuah pohon keputusan. Pertama pilih opsi hapus aturan pada list menu basis aturan. Kemudian memilih pada proses diagnosa mana node-node yang terhubung akan dihapus. Halaman hapus aturan akan muncul kemudian memilih tombol hapus. Menghapus node-node

yang terhubung tidak akan menghilangkan data-data yang ada. Aplikasi hanya menghilangkan hubungan yang ada.

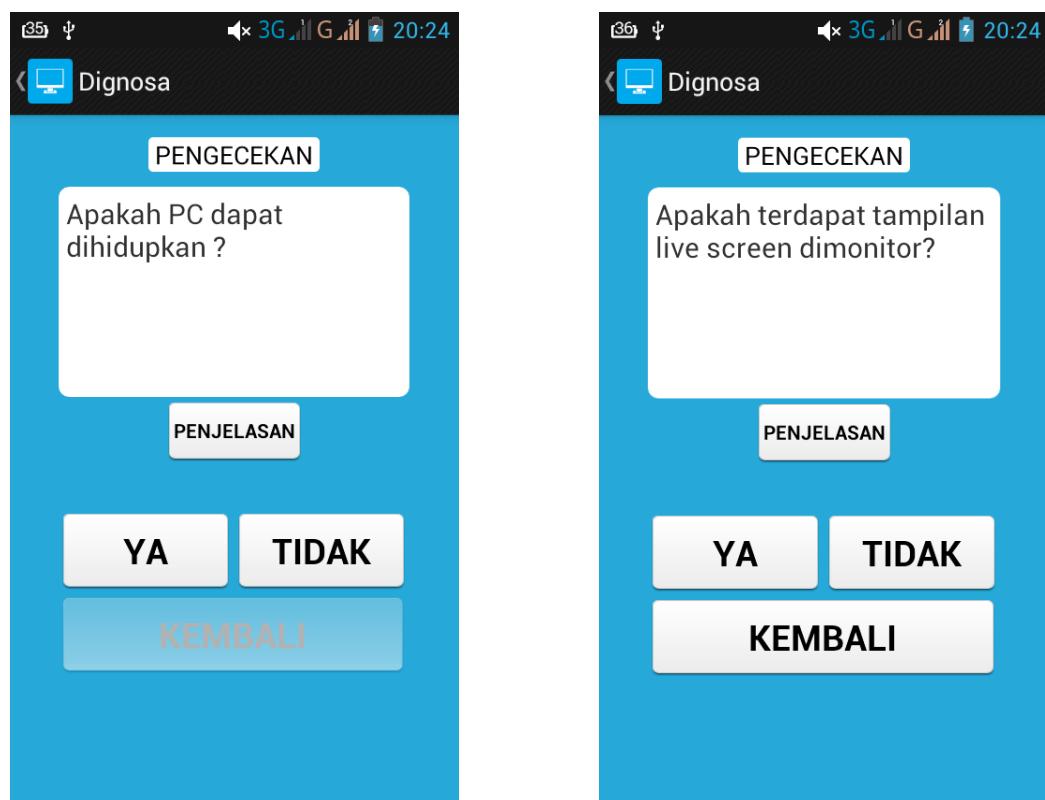


Gambar 40. Halaman hapus node-node yang terhubung pada aturan

I. Halaman Konsultasi

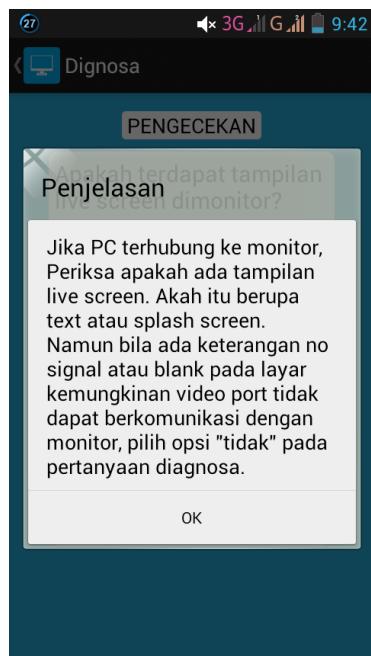
Halaman konsultasi merupakan halaman dimana pengguna melakukan proses diagnosa dengan aplikasi. Proses diagnosa dilakukan dengan model Tanya jawab antara aplikasi dengan pengguna. Proses diagnosa dimulai dengan memilih tombol mulai diagnosa pada halaman utama. Kemudian aplikasi akan menampilkan list proses diagnosa yang disediakan (gambar 32). Aplikasi akan membuka halaman diagnosa (gambar 41), kemudian akan mengambil data dengan jenis root untuk ditampilkan sebagai pertanyaan pertama dilayar. Selanjutnya pengguna memberika tanggapan terhadap pertanyaan tersebut dengan memilih salah satu tombol, ya atau tidak. Terkait tanggapan yang diberikan aplikasi akan mengecek data yang tersambung ke

pertanyaan yang ditanyakan sebelumnya. Kemudian menampilkan lagi data selanjutnya dilayar untuk ditanyakan kepada pengguna, lalu diberikan tanggapan.

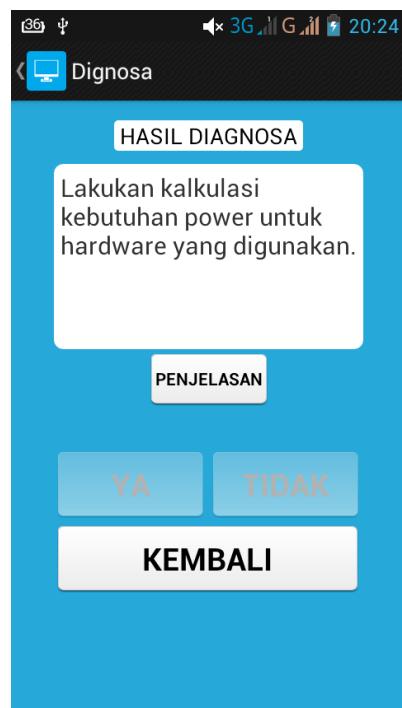


Gambar 41. Halaman konsultasi dalam menampilkan pertanyaan diagnosa

Selain itu terdapat fitur penjelasan terkait pertanyaan atau hasil yang ditampilkan bila dirasa kurang jelas (gambar 42) . Begitu seterusnya hingga pada akhirnya aplikasi mengeluarkan hasil diagnosa (gambar 43). Pada halaman konsultasi juga terdapat tombol kembali yang berguna untuk kembali ke data yang sebelumnya. Sehingga pengguna dapat memilih alur diagnosa yang berbeda dari yang sebelumnya diambil. Hal ini terkait bila hasil diagnose sebelumnya tidak sesuai dengan yang diharapkan.



Gambar 42. Halaman konsultasi menampilkan penjelasan terkait data pertanyaan/hasil diagnosa yang ditampilkan



Gambar 43. Halaman konsultasi dalam menampilkan hasil diagnosa

4. Pengetesan Pada Aplikasi Sistem pakar

a. Pengetesan *Functionality*

Pengetesan aspek *functionality* pada aplikasi sistem pakar ini dilakukan dengan serangkaian uji coba terhadap fungsi-fungsi yang tersedia. Pengetesan dilakukan oleh 2 orang responden. Responden diminta untuk mencoba aplikasi sistem pakar yang ada, kemudian memberikan tanggapannya melalui kuisioner. Hasil dari kuisioner tersebut adalah untuk memastikan bahwa fungsi-fungsi yang ada dapat berjalan dengan baik dan sesuai dengan rencana yang disusun sebelumnya pada awal pengembangan aplikasi. Hasil pengetesan dapat dilihat pada tabel 18.

Tabel 18. Hasil pengetesan fungsi-fungsi

| No | Prosedur Tes | Banyak Tes | Jumlah Hasil | |
|----|---|------------|--------------|-------|
| | | | Berhasil | Gagal |
| 1 | Pengguna memilih aplikasi PCassist pada daftar aplikasi yang ada di smartphone. | 2 kali | 2 | 0 |
| 2 | Pengguna memilih menu Mulai Diagnosa pada halaman utama. | 2 kali | 2 | 0 |
| 3 | Pengguna memilih proses diagnosa yang diinginkan pada Daftar diagnosa yang tersedia. | 2 kali | 2 | 0 |
| 4 | Pengguna memilih opsi Ya atau Tidak untuk merespon pertanyaan yang ada. | 2 kali | 2 | 0 |
| 5 | Pengguna memilih menu Penjelasan yang tersedia dihalaman konsultasi | 2 kali | 2 | 0 |
| 6 | Pengguna memilih menu Kembali yang ada dihalaman konsultasi | 2 kali | 2 | 0 |
| 7 | Pengguna memilih menu Bantuan pada halaman utama | 2 kali | 2 | 0 |

| No | Prosedur Tes | Banyak Tes | Jumlah Hasil | |
|----|---|------------|--------------|-------|
| | | | Berhasil | Gagal |
| 8 | Pengguna memilih menu Tentang PCassist dihalaman utama. | 2 kali | 2 | 0 |
| 9 | Pengguna memilih menu Pengaturan yang berupa icon roda gigi dipojok kanan atas dihalaman utama | 2 kali | 2 | 0 |
| 10 | Pengguna memilih menu Basis Pengetahuan pada halaman diagnosa | 2 kali | 2 | 0 |
| 11 | Pengguna memilih menu Tambah data , kemudian memilih pada proses diagnosa mana data akan ditambah. Selanjutnya pengguna mengisi form yang telah disedia dan memilih tombol Tambah . | 2 kali | 2 | 0 |
| 12 | Pengguna memilih menu Lihat data , kemudian milih proses diagnosa mana yang akan dilihat datanya. Kemudian mengklik daftar data yang tampil untuk melihat detailnya. | 2 kali | 2 | 0 |
| 13 | Pengguna memilih menu Edit data , kemudian memilih diproses diagnosa mana data berada. Lalu mengklik data yang akan diedit pada daftar data yang ada. Selanjutnya mengisi form yang ada dan memilih tombol Edit . | 2 kali | 2 | 0 |
| 14 | Pengguna memilih menu Hapus Data , kemudian memilih diproses diagnosa mana data berada. Lalu memilih tombol Hapus . | 2 kali | 2 | 0 |
| 15 | Pengguna memilih menu Basis aturan pada halaman pengeturan. | 2 kali | 2 | 0 |
| 16 | Pengguna memilih menu Buat aturan , kemudian memilih pada proses diagnosa mana aturan akan disusun. Selanjutnya menyusun data yang ada pada form yang telah disediakan dan memilih tombol Buat Aturan . | 2 kali | 2 | 0 |

| No | Prosedur Tes | Banyak Tes | Jumlah Hasil | |
|---------------|--|------------|--------------|-------|
| | | | Berhasil | Gagal |
| 17 | Pengguna memilih menu Lihat aturan , kemudian memilih proses diagnosa mana yang akan dilihat aturan yang telah disusun. | 2 kali | 2 | 0 |
| 18 | Pengguna memilih menu Edit aturan , kemudian memilih proses diagnosa mana yang akan diedit aturannya. Selanjutnya memilih node aturan yang ada dan melakukan pengeditan pada form yang telah disediakan. Setelah itu memilih tombol Edit . | 2 kali | 2 | 0 |
| 19 | Pengguna memilih menu up yang berupa icon anak panah dan icon aplikasi dipojok kiri atas. | 2 kali | 2 | 0 |
| 20 | Pengguna melakukan konsultasi pada halaman konsultasi dengan bantuan menu opsi Ya, Tidak, Kembali , serta Penjelasan. | 2 kali | 2 | 0 |
| Jumlah | | 40 kali | 40 | 0 |

Berdasarkan hasil pengetesan pada tabel 18, dapat diketahui bahwa dari 60 kali (2x20) tes yang dilakukan diperoleh bahwa tes yang berhasil sejumlah 40. Sedangkan untuk jumlah tes yang gagal adalah sebesar 0. Kemudian dilakukan penghitungan persentase untuk mengetahui besar persentase untuk aspek *functionality* pada aplikasi sistem pakar yang dibuat, perhitungannya adalah sebagai berikut:

$$\text{Persentase Tes Berhasil} = \frac{\text{Banyaknya tes berhasil}}{\text{Total jumlah tes}} \times 100\%$$

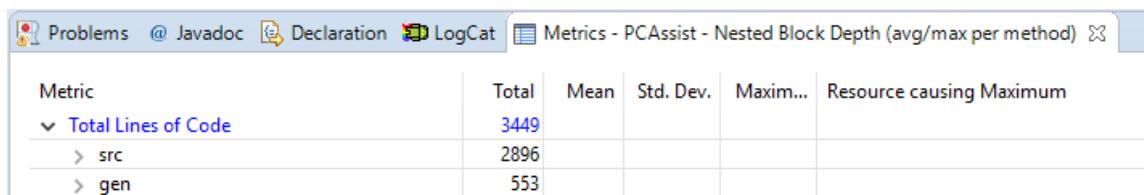
$$\text{Persentase Tes Berhasil} = \frac{40}{40} \times 100\%$$

$$\text{Persentase Tes Berhasil} = 100\%$$

Berdasarkan perhitungan yang telah dilakukan yang diperoleh persentase *functionality* sebesar 100%. Hasil tersebut kemudian dikonversi ke data kualitatif dengan membandingkan dengan nilai pada tabel . Sehingga diperoleh bahwa aspek *functionality* aplikasi sistem pakar memiliki skala “**sangat baik**” dan telah sesuai dengan aspek *functionality*.

b. Pengetesan *Reliability*

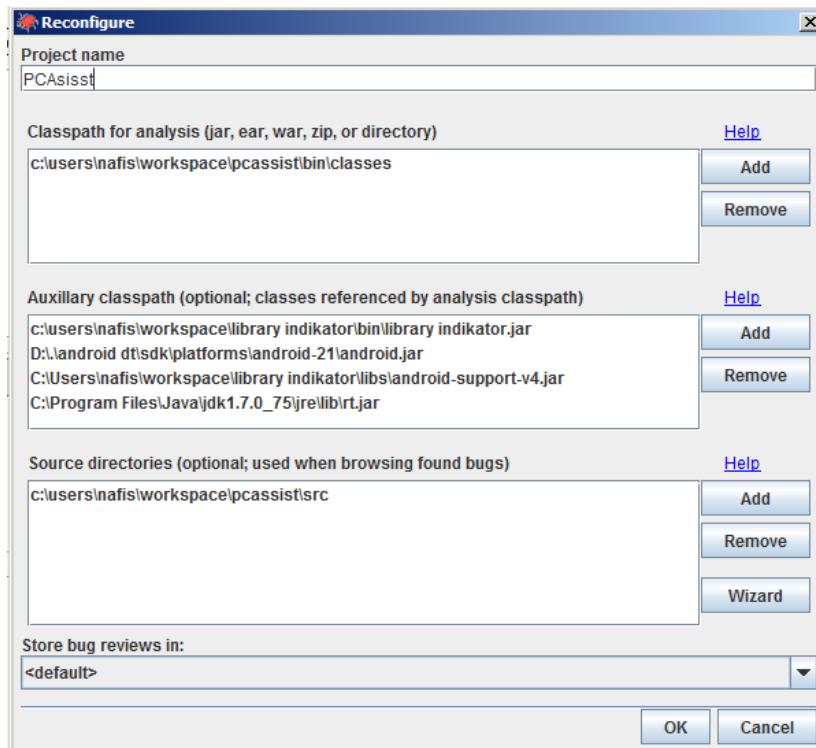
Pengetesan aspek *reliability* pada aplikasi sistem pakar dilakukan dengan mengukur jumlah *defect density* yang ada. *Defect density* diukur dengan membanding banyaknya *defect* atau *error* yang ditemukan dengan jumlah *lines of code* (LOC) yang ada. Jumlah LOC dihitung dengan bantuan plugin metric pada program eclipse yang digunakan dalam pengembangan aplikasi sistem pakar. Sedangkan untuk jumlah banyaknya *defect* atau *error* yang ada dihitung dengan bantuan program findbugs.



| Metric | Total | Mean | Std. Dev. | Maxim... | Resource causing Maximum |
|-----------------------|-------|------|-----------|----------|--------------------------|
| ▼ Total Lines of Code | 3449 | | | | |
| > src | 2896 | | | | |
| > gen | 553 | | | | |

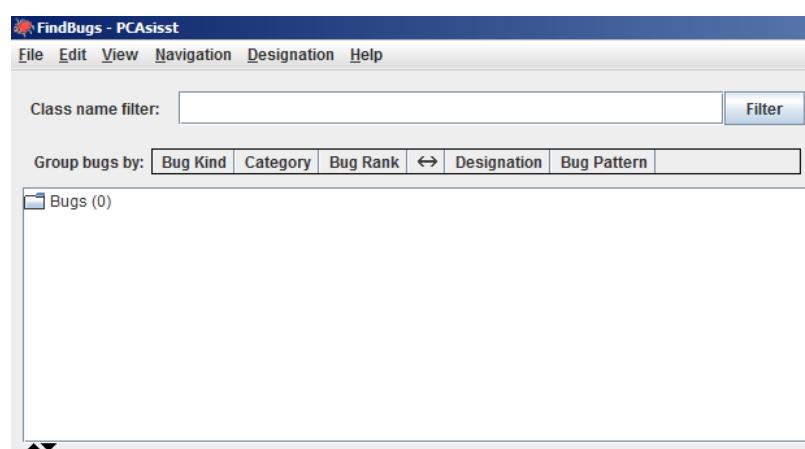
Gambar 44. hasil perhitungan banyaknya *lines of code* pada plugin metric

Berdasarkan hasil yang peroleh dari penghitungan plugin tersebut jumlah penghitungan LOC adalah sebesar 3449 LOC (gambar 44). Jumlah tersebut nantinya akan digunakan untuk mengetahui berapa besar nilai *defect density* pada aplikasi sistem pakar ini.



Gambar 45. Persiapan pencarian error aplikasi sistem pakar pada program findbugs

Proses pada program findbugs diawali dengan memasukkan sumber kode aplikasi sistem pakar dan sumber dependensinya (gambar 45). Setelah itu langkah selanjutnya adalah proses pemindaian oleh program findbugs untuk mengetahui banyaknya *defect*.



Gambar 46. Hasil pencarian error aplikasi sistem pakar pada program findbugs

Berdasarkan hasil perhitungan yang dihasilkan dari program findbugs (gambar 46), didapatkan hasil sebesar 0 *defect atau error*. Sehingga hasil dari *defect density* yang ada dapat dilihat pada perhitungan berikut :

$$\text{Defect Density} = \frac{\text{Jumlah defect/error}}{\text{Jumlah LOC}} = \frac{0}{3449} = 0$$

Hasil perhitungan tersebut kemudian dibandingkan dengan standar perkiraan jumlah *defect density* yang ada pada tabel 16. Berdasarkan standar tersebut, untuk aplikasi dengan jumlah *line of code* sebesar 2000 sampai 16000 baris, setidaknya terdapat nilai *defect density* 0 sampai dengan 40. Oleh karena itu aplikasi sistem pakar yang dibuat masih berada pada rentang kriteria tersebut. Kemudian jika dibandingkan dengan standar kriteria jumlah *defect density* untuk *Industry Average* dan *Microsoft Application* (tabel 19), maka aplikasi sistem pakar ini telah sesuai atau lolos standar yang ada.

Tabel 19. Perbandingan hasil perhitungan jumlah *defect density* pada aplikasi dengan standar yang ada untuk *Industry Average* dan *Microsoft Application*.

| Nama Standar | Nilai Standar | Hasil Pengetesan Aplikasi | Keterangan |
|------------------------------|-----------------------------|---------------------------|--|
| <i>Industry Average</i> | 1 -25 <i>defect density</i> | 0 <i>defect density</i> | Lolos , dengan jumlah <i>defect density</i> lebih kecil dari standar. Lebih Baik |
| <i>Microsoft Application</i> | 0,5 <i>defect density</i> | | Lolos , dengan jumlah <i>defect density</i> lebih kecil dari standar. Lebih Baik |

c. Pengetesan *Maintainability*

Pengetesan pada aspek *maintainability* pada aplikasi sistem pakar ini menggunakan matrik *maintainability* sesuai yang terdapat pada gambar 14. Terdapat 5 hal yang dilakukan pengetesan yaitu tentang *volume*, *complexity per unit*, *duplication*, *unit size*, *unit testing*. Berikut rincian hasil pengetesan untuk matrik maintainability yang digunakan.

1) *Volume*

Pengukuran *volume* terkait dengan sub-karakteristik *analyzability* pada aspek *maintainability*. Volume yang besar dapat menyebabkan *analyzability* rendah (sistem sulit untuk dipahami). Volume diukur dengan menggunakan besarnya LOC. Sama halnya dengan proses yang dilakukan pada pengetesan aspek *reliability*, LOC dihitung dengan bantuan plugin metric pada program eclipse. Hasil perhitungan LOC sebagaimana yang dapat dilihat pada gambar 44 , didapatkan hasil sebesar 3449 LOC (3,449 KLOC).

Hasil perhitungan jumlah KLOC tersebut kemudian dibandingkan dengan standar yang ada pada tabel 20. Berdasarkan standar tersebut aplikasi sistem pakar ini berada pada rentang 0-66 KLOC untuk aplikasi berbahasa java. Oleh karena itu aplikasi sistem pakar ini memiliki peringkat “++” atau ber volume kecil.

Tabel 20. Kriteria jumlah besaran KLOC

| Rank | KLOC | | |
|------|----------|-----------|---------|
| | Java | Cobol | PL/SQL |
| ++ | 0-66 | 0-131 | 0-46 |
| + | 66-246 | 131-491 | 46-173 |
| O | 246-665 | 491-1310 | 173-461 |
| - | 665-1310 | 1310-2621 | 461-922 |
| -- | >1310 | >2621 | >922 |

2) Complexity Per Unit

Complexity per unit mempengaruhi sub-karakteristik *changeability* dan *testability* pada aspek *maintainability* aplikasi. Hal tersebut *complexity* per unit dapat memberikan dampak negatif bagi 2 sub-karakteristik tersebut. *Complexity* per unit diukur dengan *cyclomatic complexity* (CC) per unit. *Cyclomatic complexity* dihitung dengan bantuan plugin matric pada program eclipse. Hasil perhitungan *cyclomatic complexity* dapat dilihat pada gambar 47.

| Metric | Total | Mean | Std. Dev. | Maximum | Resource causing Maximum |
|---|-------|-------|-----------|---------|--|
| McCabe Cyclomatic Complexity (avg/max per method) | 1.872 | 2.118 | | 19 | /PCAssist/src/com/nafis/pcassist/EditData.java |
| src | 1.872 | 2.118 | | 19 | /PCAssist/src/com/nafis/pcassist/EditData.java |
| com.nafis.pcassist | 1.872 | 2.118 | | 19 | /PCAssist/src/com/nafis/pcassist/EditData.java |
| gen | 0 | 0 | | | |

Gambar 47. Hasil perhitungan *cyclomatic complexity*

Berdasarkan hasil perhitungan didapatkan rata-rata *cyclomatic complexity* pada aplikasi sistem pakar ini adalah sebesar 1,872. Hasil tersebut kemudian dibandingkan dengan standar pada tabel 21.

Tabel 21. Kategori standar *cyclomatic complexity*

| CC | Risk Evaluation |
|-------|-----------------------------|
| 1-10 | Simple, without much risk |
| 11-20 | More complex, moderate risk |
| 21-50 | Complex, high risk |
| >50 | Untestable, very high risk |

Berdasarkan perbandingan antara hasil *cyclomatic complexity* yang ada dengan tabel 21 dapat diperoleh kesimpulan bahwa aplikasi sistem pakar yang dibuat memiliki *complexity* per unit dengan kategori “**Simple**”.

3) Duplication

Duplikasi dalam *source code* merupakan fenomena yang sering terjadi pada setiap sistem. Meskipun duplikasi dalam jumlah kecil merupakan hal yang natural. Namun bila terjadi dengan jumlah yang besar dapat mengganggu *maintainability* sistem. Duplikasi mempengaruhi sub-karakteristik *analyzability* dan *changeability*. Duplikasi pada aplikasi sistem pakar ini dihitung dengan bantuan program PMD Duplicate Code Detector. Hasil perhitungan duplikasi kode pada aplikasi sistem pakar ini dapat dilihat pada gambar 48.

The screenshot shows the PMD Duplicate Code Detector application window. The root source directory is set to C:\Users\nafis\workspace\PCAssist\src. The report shows chunks larger than 75 lines. The 'Also scan subdirectories' option is checked. The language is set to Java, and the extension is .JAVA. The file encoding is Cp1252. The results table lists file pairs with their respective line counts:

| Source | Matches | Lines |
|--------------------|---------|-------|
| (2 separate files) | | 17 |
| (3 separate files) | 3 | 11 |
| (2 separate files) | | 19 |
| (2 separate files) | | 17 |
| ...IDbhandler.java | | 13 |
| (5 separate files) | 5 | 26 |
| ...IDbhandler.java | | 14 |
| ...IDbhandler.java | 3 | 14 |

The right pane displays the Java code for the identified duplicate blocks. One block is shown in full:

```
@Override  
protected void onCreate(Bundle savedInstanceState) {  
    super.onCreate(savedInstanceState);  
    getActionBar().setDisplayHomeAsUpEnabled(true);  
    Intent intent = getIntent();  
  
    no = intent.getIntExtra("opsi", 0);  
    String[] values1 = new String[] {"Diagnosa Kerusakan Pada P...  
  
    ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,  
        R.layout.rowlist,R.id.label, values1);  
    setListAdapter(adapter);  
  
    protected void onListItemClick(android.widget.ListView l,  
        android.view.View v, int position, long id) {  
        String item = (String) getListAdapter().getItem(position);  
    }  
}
```

Gambar 48. Hasil perhitungan duplikasi kode

Dari hasil perhitungan pada gambar 48 ditemukan sebanyak 131 LOC yang mengalami duplikasi. Jika dilakukan persentase duplikasi, nilainya adalah:

$$\text{Persentase duplikasi} = \frac{131}{3449} \times 100\% = 3,79\%$$

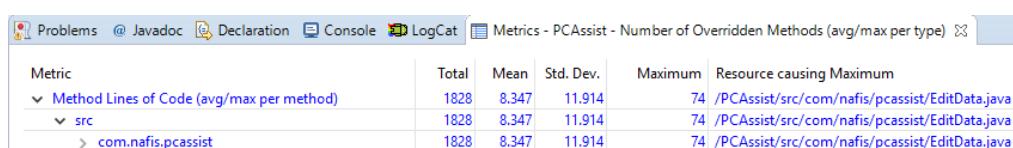
Hasil persentase sebesar 3,79% kemudian dibandingkan dengan kriteria rangking duplikasi yang ada pada tabel 22. Maka aplikasi sistem pakar yang dikembangkan memiliki kategori duplikasi “+” atau **baik**.

Tabel 22. Kategori penilaian duplikasi kode

| Rank | Percentase Duplikasi | Kategori |
|------|----------------------|---------------|
| ++ | 0-3% | Sangat baik |
| + | 3-5% | Baik |
| O | 5-10% | Cukup |
| - | 10-20% | Kurang |
| -- | 20-100% | Sangat kurang |

4) Unit Size

Unit merupakan bagian terkecil dalam sebuah program. Dalam bahasa pemrograman java, unit terkecilnya adalah method. Besarnya unit yang ada berpengaruh pada sub-karakteristik *analyzability* dan *testability*. Besarnya unit dalam aplikasi sistem pakar yang dibuat diukur dengan banyaknya LOC dalam method-method yang ada.



Gambar 49. Hasil perhitungan besarnya LOC dari method-method yang ada

Penghitungan jumlah LOC dalam method-method dilakukan dengan bantuan plugin metric pada program eclipse. Berdasarkan hasil perhitungan yang didapatkan pada gambar 49, Jumlah unit size untuk aplikasi sistem pakar ini adalah sebesar 1872 LOC (1,872 KLOC). Jika hasil tersebut

dibandingkan dengan kriteria pada tabel 17, maka besarnya unit dalam aplikasi sistem pakar ini mendapat peringkat “++” atau **Kecil**.

5) **Unit Testing**

Unit testing merupakan sebuah test yang ditulis oleh pengembang aplikasi bersangkutan, yang bertujuan secara otomatis menguji kode aplikasi.

Unit testing pada aplikasi sistem pakar ini digunakan *framework JUnit* sebagai sarana untuk melakukan *unit test*. Dari Sejumlah unit test yang dilakukan, diperoleh total ketercakupan tes sebesar 100% (data lengkap dapat dilihat pada lampiran). Hasil tersebut kemudian dibandingkan skala pada tabel 23, didapatkan hasil untuk unit testing pada peringkat “+”.

Table 23. Penilaian ketercakupan *unit testing*

| Peringkat | Ketercakupan <i>Unit Testing</i> |
|------------------|---|
| ++ | 95%-100% |
| + | 80%-95% |
| O | 60%-80% |
| - | 20%-60% |
| -- | 0%-20% |

d. **Pengetesan *Portability***

Pengetesan aspek *portability* aplikasi sistem pakar ini dilakukan dengan melakukan instalasi dan menjalankan aplikasi pada perangkat *smartphone android*. Karena keterbatasan perangkat untuk uji coba, peneliti se bisa mungkin menggunakan beberapa perangkat dengan versi android dan layar yang bervariasi, dengan harapan setidaknya dapat menunjukkan bahwa aplikasi dapat dijalankan pada perangkat-perangkat yang berbeda-beda. Dalam pengetesan dilakukan pengetesan manual dengan beberapa perangkat fisik android berbeda dan pengetesan *cloud testing* dengan bantuan testdroid. Hasil pengetesan manual aplikasi dapat dilihat pada tabel

24. Sedangkan untuk gambar hasil pengetesan *portability* secara manual dapat dilihat pada lampiran.

Tabel 24. Pengetesan manual aplikasi sistem pakar

| No | Tipe Smartphone | Versi OS Android | Ukuran Layar | Proses Instalasi | Proses Aplikasi Berjalan |
|----|--------------------------------|------------------|---|--------------------|-------------------------------|
| 1 | Nexus 4 | 5.0 | 4.7 inches , 768 x 1280 pixels, (318 ppi pixel density) | Instalasi berhasil | Aplikasi berjalan dengan baik |
| 2 | Samsung Galaxy Fame (GT-S6810) | 4.1.2 | 3.5 inches, 320 x 480 pixels, (165 ppi pixel density) | Instalasi berhasil | Aplikasi berjalan dengan baik |
| 3 | Oppo neo 3 | 4.2.2 | 4.5 inches, 480 x 854 pixels, (218 ppi pixel density) | Instalasi berhasil | Aplikasi berjalan dengan baik |
| 4 | Lenovo A820 | 4.1.2 | 4.5 inches, 540 x 960 pixels, (245 ppi pixel density) | Instalasi berhasil | Aplikasi berjalan dengan baik |

Sedangkan untuk *cloud testing* dengan testdroid menggunakan beberapa pilihan smartphone yang disediakan, hasilnya dapat dilihat pada tabel 25. Sedangkan untuk gambar hasil pengetesan *portability* secara *cloud testing* dapat dilihat pada lampiran.

Tabel 25. Pengetesan secara *cloud testing* dengan testdroid

| No | Tipe Smartphone | Versi OS Android | Ukuran Layar | Proses Instalasi | Proses Aplikasi Berjalan |
|----|-----------------------|------------------|---|--------------------|-------------------------------|
| 1 | Acer iconia Tab 8 | 4.4.2 | 8.0 inches, 1200 x 1920 pixels, (283 ppi pixel density) | Instalasi berhasil | Aplikasi berjalan dengan baik |
| 2 | Asus Fonepad ME371MG | 4.1 | 7.0 inches, 800 x 1280 pixels, (216 ppi pixel density) | Instalasi berhasil | Aplikasi berjalan dengan baik |
| 3 | Dell Venue 7 3730 | 4.2.2 | 7.0 inches, 800 x 1280 pixels, (216 ppi pixel density) | Instalasi berhasil | Aplikasi berjalan dengan baik |
| 4 | Motorola RAZR I XT890 | 4.1.2 | 4.3 inches, 540 x 960 pixels, (256 ppi pixel density) | Instalasi berhasil | Aplikasi berjalan dengan baik |
| 5 | Samsung Galaxy Tab 3 | 4.4.2 | 7.0 inches, 600 x 1024 pixels, (170 ppi pixel density) | Instalasi berhasil | Aplikasi berjalan dengan baik |

| No | Tipe Smartphone | Versi OS Android | Ukuran Layar | Proses Instalasi | Proses Aplikasi Berjalan |
|----|-------------------------------|------------------|---|--------------------|-------------------------------|
| 6 | NVIDIA Shield Tablet | 4.4.2 | 8.0 inches, 1920 x 1200 pixels, (283 ppi pixel density) | Instalasi berhasil | Aplikasi berjalan dengan baik |
| 7 | Samsung Galaxy Nexus GT-I9250 | 4.2.2 | 4.65 inches, 720 x 1280 pixels, (316 ppi pixel density) | Instalasi berhasil | Aplikasi berjalan dengan baik |
| 8 | Samsung Galaxy Nexus SPH-L700 | 4.3 | 4.65 inches, 720 x 1280 pixels, (316 ppi pixel density) | Instalasi berhasil | Aplikasi berjalan dengan baik |

Table 26. Perhitungan hasil pengetesan *portability*

| No | Pengetesan | Banyak Tes | Berhasil | Gagal |
|--------------|-------------------------------------|------------|----------|-------|
| 1 | Instalasi aplikasi pada perangkat | 12 | 12 | 0 |
| 2 | Menjalankan aplikasi pada perangkat | 12 | 12 | 0 |
| Total | | 24 | 24 | 0 |

$$\text{Persentase} = \frac{\text{Skor tes berhasil}}{\text{Skor total tes}} \times 100\%$$

$$\text{Persentase} = \frac{24}{24} \times 100\%$$

$$\text{Persentase} = 100\%$$

Hasil pengetesan *portability* pada tes yang dilakukan pada perangkat yang tersedia baik secara manual dengan perangkat sebenarnya dan secara *cloud testing* tersebut jika dipersentasekan didapatkan hasil sebesar **100%**. Hasil tersebut menunjukan bahwa aplikasi sistem pakar memiliki tingkat usability yang sangat baik dengan indikator aplikasi dapat terpasang dan dijalankan pada setiap perangkat yang ada

e. Pengetesan *Usability*

Pengetesan aspek *usability* dilakukan dengan melakukan uji coba aplikasi pada 32 responden. Responden diminta untuk mencoba aplikasi sistem pakar yang dibuat kemudian mengisi kuisioner yang telah dibagikan sebelumnya. Kuisioner dibuat berdasarkan pada kuisioner *usability* yang dibuat oleh J.R Lewis dengan jumlah soal sebanyak 19 butir soal. Hasil pengetesan dapat dilihat pada lampiran.

Perhitungan skor total hasil penilian *usability* diperoleh hasil sebesar 2330. Sedangkan untuk hasil maksimal yang diharapkan adalah sebesar 3040. Berdasarkan hasil tersebut kemudian dilakukan perhitungan persentase, dari perhitungan tersebut dihasilkan persentase sebesar **76,64%** untuk aspek *usability*. Hasil persentase tersebut lalu dibandingkan dengan hasil konversi nilai persentase pada tabel 15, maka diperoleh tingkat *usability* untuk aplikasi sistem pakar yang dibuat berada pada tingkat **Baik**.

$$\text{Persentase } \textbf{\textit{Usability}} = \frac{\text{Total hasil}}{\text{Total Hasil maksimal}} \times 100\%$$

$$\text{Persentase } \textbf{\textit{Usability}} = \frac{2330}{3040} \times 100\%$$

$$\text{Persentase } \textbf{\textit{Usability}} = 76,64\%$$

B. Pembahasan

1. Rangkuman Penelitian

Penelitian dilakukan dalam rangka mengembangkan sebuah aplikasi sistem pakar pada perangkat *smartphone android*. Aplikasi sistem pakar yang dibuat bertujuan untuk melakukan diagnosa sumber kerusakan yang ada pada perangkat komputer atau PC. Hal tersebut dilatar belakangi bahwa diagnosa sumber kerusakan dapat menjadi hal yang sulit dan dapat

menghambat proses perbaikan pada komputer atau PC. Selain itu dipilihnya perangkat *smartphone* android sebagai media sistem pakar ini akan berjalan, tidak terlepas dari perkembangan pesat android beberapa tahun terahir. Melihat dukungan dari pengembang sistem operasi android dan semakin matangnya android dari versi ke versi. Oleh karena itu peneliti melihat sebagai sebuah peluang untuk menjadi solusi praktis dalam pengembangan sistem pakar.

Model penelitian yang digunakan adalah model *Research and Developpent* (R&D). Tahapan penelitian mengacu pada model pengembangan perangkat lunak sekuensial linier atau *waterfall*. Tahap-tahap pengembangan tersebut meliputi tahap analisis kebutuhan, desain, implementasi, dan pengetesan.

Tahap analisis kebutuhan merupakan tahap dimana peneliti mengumpulkan informasi terkait pengembangan aplikasi sistem pakar. Pada tahap ini peneliti mempelajari bagaimana sistem pakar bekerja. Serta mendefinisikan fungsi-fungsi yang akan ada pada sistem pakar nanti. Selain itu peneliti juga mencari dan mempelajari sumber-sumber literatur yang terkait dengan diagnosa kerusakan pada PC.

Tahap desain merupakan tahap dimana pengembangan perangkat lunak memasuki awal realisasi. Pada tahap ini peneliti membuat desain UML aplikasi sistem pakar. Desain UML ini nantinya akan digunakan untuk panduan membuat aplikasi sistem pakar terutama dari sisi bagaimana sistem bekerja dan fungsi-fungsi yang ada. Desain UML terdiri dari beberapa model diagram. Diagram-diagram tersebut yaitu diagram *use case*, diagram aktifitas, diagram sekuensial, dan diagram kelas. Selain terdapat desain

UML, juga terdapat desain untuk antarmuka aplikasi (*user interface*) dan desain database. Desain *user interface* digunakan sebagai panduan untuk membuat tampilan pada aplikasi sistem pakar. Terdapat beberapa desain *user interface* untuk masing-masing fungsi, meski demikian terdapat desain yang sama untuk fungsi yang berbeda. Desain database digunakan untuk panduan membuat database yang nantinya digunakan untuk menyimpan data dan alur diagnosa yang ada.

Tahap implementasi atau pengkodean merupakan tahap dimana aplikasi direalisasikan. Aplikasi dibuat dengan menggunakan program eclipse dengan tambahan plugin *android development tool*. Sedangkan bahasa pemrograman yang digunakan adalah bahasa java dan xml. Aplikasi dibuat berdasarkan dengan desain UML dan *user interface* yang telah dibuat sebelumnya. Proses penggerjaan aplikasi kurang lebih memakan waktu 3 bulan dari februari sampai dengan april 2015.

Tahap pengetesan merupakan tahap akhir dari proses pengembangan aplikasi sistem pakar. Pada tahap ini aplikasi sistem pakar yang telah dibuat diuji untuk mengetahui tingkat kelayakan dan agar menghasilkan aplikasi yang baik. Sebagai dasar acuan kelayakan perangkat lunak, penulis menggunakan standar ISO 9126. Standar ISO 9126 meliputi aspek *functionality, reliability, maintainability, portability, dan usability*.

a. Pembahasan Pengetesan *functionality*

Pengetesan *functionality* dilakukan dengan metode *black-box testing*. *Black-box testing* dilakukan dengan membandingkan tingkat keberhasilan dari fungsi-fungsi yang direncanakan pada awal proses pengembangan dengan fungsi-fungsi yang terdapat pada aplikasi yang telah jadi. Selanjutnya

proses pengecekan dilakukan dengan memperhatikan input yang dimasukkan dengan hasil output yang diharapkan muncul.

Pengetesan secara teknis dilakukan dengan bantuan 2 orang responden yaitu M. Rifqi Atsani,S.Pd, M.Kom dan Handika Tri Aswara S.Pd yang berprofensi sebagai programmer. Hasil pengetesan didapatkan bahwa semua fungsi yang direncanakan pada awal pengembangan dapat direalisasikan pada aplikasi akhir dan bekerja dengan baik. Selain itu juga responden memberikan masukkan terkait fungsi yang ada pada aplikasi. Masukkan yang diperoleh yaitu agar ditambahkan fungsi untuk menghapus data satu per satu. Sebab pada awalnya fungsi menghapus data berfungsi menghapus seluruh data yang ada sekaligus.

b. Pembahasan Pengetesan *reliability*

Pengetesan *reliability* pada aplikasi sistem pakar ini merupakan salah satu aspek yang penting. Aspek *reliability* yang baik menunjukan bahwa aplikasi yang dibuat dapat diandalkan atau mampu mengatasi kesalahan yang muncul. *Reliability* diukur dengan menggunakan nilai *defect density*. Secara garis besar semakin kecil nilai *defect density* suatu aplikasi semakin baik. Nilai *defect density* dihitung dengan membandingkan jumlah *bugs* atau *defect* yang ditemukan dengan banyaknya jumlah baris kode (*line of code*) aplikasi. *Defect* pada aplikasi sistem pakar ini dicari dengan menggunakan program findbugs. Findbugs merupakan program statik analisis yang akan memindai *source* aplikasi sistem pakar untuk mencari *defect*. Pemindaian dilakukan dengan mencari pola-pola yang dapat menyebabkan terjadinya *defect* pada aplikasi. Dalam proses pemindaian juga dilakukan beberapa filter untuk menjegah munculnya *false positif*. Jumlah baris kode dihitung

dengan plugin matric pada program eclipse. Hasil nilai defect density yang dihasilkan lebih baik (lebih kecil) dari standar yang diacu.

c. Pembahasan Pengetesan *maintainability*

Pengetesan *maintainability* aplikasi sistem pakar ini dilakukan dengan menggunakan *maintainability matrix* (gambar 19) sebagai acuan. Pengetesan dilakukan dengan menguji *source code properties* untuk mengetahui pengaruhnya terhadap sub-karakteristik *maintainability*. Sedangkan *source code properties* yang diuji meliputi *volume*, *complexity per unit*, *duplication*, *unit size*, dan *unit testing*. Hasil pengetesan terhadap *source code properties* dihasilkan bahwa aplikasi sistem pakar memiliki *volume* yang kecil, jumlah *complexity per unit* termasuk kategori *simple*, persentase *duplication* sebesar 3,79%, besar *unit size* yang kecil, dan hasil ketercakupan pengetesan *unit test* sebesar 100%. Hasil tersebut jika dikaitkan dengan sub-karakteristik dari *maintainability* menunjukkan bahwa *analyzability* dari aplikasi sistem pakar ini baik dengan indikator *volume*, *duplication*, *unit size* yang kecil dan hasil *unit testing* sebesar 100%. Sub-karakteristik *changeability* menunjukkan bahwa aplikasi dapat diubah seaktu-waktu bila diperlukan tanpa masalah dengan indikator nilai rata-rata *complexity per unit* yang kecil (*simple*) dan banyaknya *duplication* yang kurang dari 5%. Sub-karakteristik *stability* menunjukkan bahwa aplikasi memiliki tingkat stabilitas yang baik dengan indikator ketercakupan nilai *unit testing* sebesar 100%. Sub-karakteristik *testability* menunjukkan bahwa aplikasi ini mudah dilakukan pengetesan dengan indikator rata-rata nilai *complexity per unit* dan *unit size* yang kecil, serta besar *unit testing* sebesar 100%.

d. Pembahasan Pengetesan *portability*

Pengetesan *portability* dilakukan dengan melakukan tes aplikasi sistem pakar pada beberapa perangkat *smartphone* yang berbeda-beda. Tes yang dilakukan adalah dengan mengamati berhasil atau tidaknya aplikasi dipasang dan dijalankan pada perangkat tersebut. Terdapat 4 perangkat fisik yang digunakan, masing-masing memiliki spesifikasi yang berbeda. Kemudian untuk menambah variasi jumlah perangkat yang digunakan, peneliti menggunakan bantuan layanan testdroid. Testdroid merupakan layanan jasa yang menyediakan perangkat *smartphone* android untuk melakukan tes terhadap aplikasi yang ada. Sayangnya peneliti hanya dapat menggunakan 8 buah perangkat *smartphone* dari testdroid. Hal tersebut terkait dengan biaya sewa bila ingin menggunakan perangkat lebih banyak lagi. Pengetesan pada testdroid merupakan pengetesan secara *cloud testing*, peneliti hanya tinggal mengunggah file aplikasi ke situs testdroid dan akan secara otomatis dilakukan tes. Baik secara fisik maupun *cloud* dengan testdroid, seluruh tes berjalan dengan baik, aplikasi dapat dipasang dan dijalankan pada seluruh perangkat tes yang tersedia.

e. Pembahasan Pengetesan *usability*

Pengetesan *usability* dilakukan dengan mengujikan aplikasi yang telah dibuat kepada pengguna akhir. Pengetesan dilakukan di SMK N 2 Depok, Sleman dengan jumlah responden sejumlah 32 orang siswa kelas X jurusan Teknik Komputer dan Jaringan. Dalam praktiknya siswa diminta mencoba aplikasi sistem pakar yang telah dibuat. Kemudian mengisi angket yang telah diberikan sebelumnya. Angket yang digunakan dibuat berdasarkan angket *usability* milik J. R. Lewis. Hasil perhitungan angket dihasilkan persentase

usability sebesar 76,64%. Hal tersebut menunjukkan tingkat *usability* aplikasi sistem pakar ini dalam kategori baik. Berarti aplikasi ini mudah untuk dipahami, dipelajari, dan digunakan oleh pengguna.

2. Keterbatasan Penelitian

Penelitian ini bertujuan untuk mengembangkan sebuah aplikasi sistem pakar yang baik dan dapat diandalkan. Meskipun demikian pada praktiknya terdapat beberapa keterbatasan yang terjadi selama proses penelitian berlangsung. Keterbatasan yang dihadapi tersebut antara lain:

- a. Pada proses akusisi pengetahuan untuk menjadi sebuah aturan yang digunakan dalam proses konsultasi, peneliti hanya menggunakan sumber pengetahuan terdokumentasi. Peneliti tidak membuat secara langsung aturan yang dipakai, sebab pada buku yang dirujuk sudah terdapat *flowchart* atau bagan alur diagnosa untuk masing-masing *hardware* yang terdapat pada PC dan lengkap dengan penjelasan masing masing poin pada *flowchart* yang ada.
- b. Pada pengetesan untuk faktor *portability*, perangkat yang digunakan jumlahnya masih terbatas. Oleh karena itu peneliti berfokus bahwa perangkat yang digunakan setidaknya dapat mewakili versi android yang berbeda. Aplikasi sistem pakar dirancang untuk beroperasi pada android versi 4.0 dan di atasnya.
- c. Pada pengetesan aplikasi dilapangan atau ketika diujicobakan pada pengguna, pengetesan yang dilakukan hanya dapat sebatas untuk mengetahui tingkat *usability* aplikasi saja. Pengetesan tidak dilakukan dengan menghadapi permasalahan langsung pada perangkat PC. Hal tersebut dikarenakan tidak adanya perangkat PC yang mengalami

kerusakan pada saat penelitian dilakukan. Meski demikian peneliti sempat melakukan diskusi dengan Guru pengampu mata pelajaran yang bersangkutan terkait dengan aturan yang digunakan pada aplikasi.

BAB V

KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan penelitian yang telah dilakukan dalam pengembangan aplikasi sistem pakar untuk diagnosa kerusakan pada komputer, maka peneliti dapat mengambil beberapa kesimpulan yaitu:

1. Aplikasi sistem pakar dikembangkan dengan menggunakan model pengembangan perangkat lunak sekuensial linier atau dikenal juga dengan model *waterfall*. Tahapan pengembangan dalam model ini meliputi proses analisis pengetahuan, desain, pengkodean atau implementasi, dan pengetesan.
2. Kelayakan aplikasi diuji dalam tahap pengetesan untuk menentukan bahwa aplikasi yang dibuat layak untuk digunakan. Kelayakan aplikasi diukur dengan menggunakan kriteria kelayakan perangkat lunak ISO 9126. Pengujian kelayakan aplikasi oleh pengembang meliputi aspek *functionality*, *reliability*, *Maintainability*, serta *portability*. Aspek *functionality* berdasarkan pengetesan yang telah dilakukan mendapatkan hasil tes sangat baik. Semua fungsi yang ada dapat berjalan dengan baik. Aspek *reliability* berdasarkan hasil pengetesan yang dilakukan didapatkan hasil sangat baik. Hal tersebut dapat dilihat dengan indikator besarnya nilai *defect density* yang dihasilkan adalah sebesar 0 *defect density*. Hasil pengetesan aspek *portability* didapatkan bahwa aplikasi dapat dipasang dan dijalankan pada perangkat-perangkat berbeda yang telah disediakan. Oleh karena itu aspek *portability* untuk aplikasi sistem pakar ini mendapat

hasil sangat baik. Sedangkan untuk aspek *maintability*, aplikasi sistem pakar ini mendapat hasil baik. Aplikasi sistem pakar yang dibuat memiliki tingkat *usability* yang baik. Hal ini berarti bahwa aplikasi sistem pakar yang dibuat mudah untuk digunakan, dipelajari, dipahami, dan menarik bagi pengguna. Pernyataan tersebut berdasarkan pada hasil pengetesan terhadap 32 responden. Hasil pengetesan tersebut dihasilkan presentase *usability* sebesar 76,64 %.

B. Saran

Penelitian yang telah dilakukan masih terdapat kekurangan dan masih dapat dikaji lebih lanjut. Oleh sebab itu peneliti memberikan saran-saran terkait pengembangan penelitian ini yaitu:

1. Aplikasi sistem pakar yang dikembangkan dalam penelitian ini masih berupa aplikasi *offline*. Aplikasi masih dapat dikembangkan dengan model *client-server* agar dapat lebih mudah dan fleksibel dalam melakukan manajemen data.
2. Proses diagnosa yang ada masih terbatas, oleh karena itu masih dapat ditambah lagi dengan proses diagnosa periperal lain seperti input-output device, networking dan sebagainya.
3. Desain aplikasi yang ada saat ini masih sederhana. Oleh karena itu perbaikan desain masih diperlukan agar lebih menarik dan memudahkan pengguna. Terlebih lagi terkait dukungan untuk jenis layar yang berbeda dan perangkat tablet.
4. Penelitian masih dapat dikembangkan untuk meneliti bagaimana pengaruh aplikasi sistem pakar ini terhadap peningkatan kinerja teknisi dalam melakukan perbaikan perangkat komputer.

DAFTAR PUSTAKA

- Abdelhamid, Y & El-Helly, M. (2013) *A New Approach for Developinh Diagnostic Expert Systems on Mobile Phones*. *Jurnal Communications in Information Science and Management Enginerring*. (Volume 3, Issue 8). Hlm. 374-384.
- Android. (2014). *Platform Versions*. Diakses pada 20 desember 2014, dari <http://developer.android.com/about/dashboards/index.html#Platform>
- Android. (2014). *Supporting Multiple Screens*. Diakses pada 20 desember 2014 dari http://developer.android.com/guide/practices/screens_support.html
- AppBrain. (2014). *Most popular Android market categories*. Retrieved 7 Agustus, 2014, dari <http://www.appbrain.com/stats/android-market-app-categories>.
- Barata, Kimberly et al. (1999). *Understanding Computers: an Overview for Records and Archives Staff*. London: International Records Managent Trust.
- Cote, M.A. Suryn, W & Georgiadou, E. (2006). *Software Quality Model Requirements for Software Quality Engineering*.
- Deissenboeck, F et al. (2009). *Software Quality Models: Purposes, Usage Scenarios and Requirements*. Munchen: Technische Universitat Munchen.
- Gay, L. R. (1987). Educational research competencies for analysis & application edition. Ohio: A Bell & Howell Company.
- Gargenta, M & Nakamura, M. (2014). *Learning Android, Second Edition*. Amerika: O'Reilly Media , Inc.
- Guritno, S., Sudaryono, & Raharja, U. (2011). *Theory and application of IT Research: Metodologi Penelitian Teknologi Informasi*. Yogyakarta: Penerbit ANDI.
- Golden, Bryan. (2012) *Learn from the experience of others*. Diaskses pada 7 agustus 2014, dari www.presspublications.com/opinionscolumns/146-dare-to-livewithout-limits-/8521-learn-from-the-experience-of-others.
- Hartati, S & Iswanti, S. (2008). *Sistem Pakar & Pengembangannya*. Yogyakarta : Graha Ilmu.

- Heitlager, Ilja., Kuipers, Tobias., & Visser, Joost . (2007). *A Practical Model for Measuring Maintainability*. QUATIC '07 Proceedings of the 6th International Conference on Quality of Information and Communications Technology. Hlm. 30-39.
- Isa, M.D & Sadek, O. (2000). *PC Diagnosis and Troubleshooting Expert System based On Computer Response Power On Self Test (POST)*. Jurnal Intelligent Systems and Technologies for The new millennium. (Tencon Proceedings). Hlm. 458-463.
- ISO/IEC. (1991). *Information Technology - Software Product Evaluation - Quality Characteristics and Guidelines for Their Use 9126*.
- Leng, G.W & Teen, L.K. (1992). *ESPCRM an Expert System for Personal Computer Repair and Maintenance*. Jurnal Engineering Applications of Artificial Intelligence. (Volume 5, Issue 2). Hlm. 121-133.
- Lewis, J. R. (1993). *IBM Computer Usability Satisfaction Questionnaires: Psychometric Evaluation and Instructions for Use*. International Journal of Human Computer Interaction. Hlm. 1-39
- Malaiya, Yashwant K. (2005). *Software Reliability and Security*. Jurnal Encyclopedia of Library and Information Science. Hlm 1-12.
- McCall, J. A., Richards, P. K., & Walters, F. G. (1977). *Factors in Software Quality US Rome Air Development Center Reports*.
- McConnell, Steve. (2004). *Code Complete, Second Edition*. Amerika: Microsoft Press.
- Mueller, Scot. (2013). *Upgrading and Repairing PCs 21st Edition*. Indiana : Pearson Education, Inc.
- Niknejad, Aida. (2011). *A Quality Evaluation of an Android Smartphone Application*. Sweden: University of Gothenburg.
- Pressman, Roger S. (2002). *Rekayasa Perangkat Lunak Pendekatan Praktisi (Buku I)*. Penerjemah: LN Harnaningrum. Yogyakarta: Penerbit ANDI.
- Pressman, R. S. (2010). *Software engineering: a practitioner's approach*. New York: The McGraw-Hill Companies, Inc.
- Rosenthal, Morris. (2013). *Computer Repair With Diagnostic Flowcharts Third Edition: Troubleshootig PC Hardware Problems from Boot failure to Poor Performance*. Amerika: Foner Books.

- Singh, Suraj et al. (2014). *Healthcare Services using Android Devices. The Internationa; Journal of Engineering and Science.* (Volume 3, issue 4). Hlm 41-45.
- Statista. 2014. *Mobile OS: Market Share in Indonesia 2011-2014*. Diakses pada 31 Oktober 2014, dari <http://www.statista.com/statistics/262205/market-share-held-by-mobile-operating-systems-in-indonesia/>
- Sutojo, T. Mulyanto, E & Suhartono, V. (2011). *Kecerdasan Buatan*. Yogyakarta: Penerbit ANDI.
- Turban, E. Aronson, J.E & Liang Peng Ting. (2005). *Decision Support Systems and Intelligent Systems – 7th Ed. Jilid 2 (Sistem pendukung Keputusan dan Sistem Cerdas)*. Penerjemah: Siska Primaningrum. Yogyakarta: Penerbit ANDI.

LAMPIRAN

LAMPIRAN

DEFINISI USE CASE

DEFINISI USE CASE MELAKUKAN KONSULTASI

| | |
|-------------------------|--|
| Use Case Name | Melakukan Konsultasi |
| Actors | <i>Knowledge Engginner, Normal User</i> |
| Description | Use case ini berfungsi untuk menangani proses konsultasi diagnosa pada sistem pakar |
| Precondition | Pengguna yang akan melakukan konsultasi, memilih terlebih dulu proses konsultasi mana yang akan dipakai dari daftar proses konsultasi yang telah disediakan. |
| Basic Path | <ol style="list-style-type: none"> 1. Sistem mengecek alur aturan yang ada. 2. Sistem menampilkan pertanyaan awal diaplikasi. 3. Pengguna memasukan input sebagai tanggapan terhadap pertanyaan yang ditampilkan. 4. Sistem mengecek alur aturan yang ada dan menampilkan pertanyaan berikutnya. 5. Nomor 3 dan 4 berulang sampai ditemukan jawaban. |
| Alternative Path | <ol style="list-style-type: none"> 1. Sistem mengecek alur aturan yang ada. 2. Sistem menampilkan pertanyaan awal diaplikasi. 3. Pengguna memasukan input sebagai tanggapan terhadap pertanyaan yang ditampilkan. 4. Sistem mengecek alur aturan yang ada dan menampilkan pertanyaan berikutnya. 5. Nomor 3 dan 4 berulang sampai ditemukan jawaban. 6. Jika jawaban tidak sesuai yang diharapkan, pengguna dapat kembali kepertanyaan sebelumnya dan mengambil alur yang berbeda. |
| Postcondition | Sistem menampilkan hasil dari proses diagnosa yang telah dilakukan. |
| Exception Path | - |
| Extend | - |
| Include | - |

DEFINISI USE CASE MELIHAT BANTUAN

| | |
|-------------------------|--|
| Use Case Name | Melihat Bantuan |
| Actors | <i>Knowledge Engginer, Normal User</i> |
| Description | Use case ini berfungsi untuk menampilkan informasi bantuan terkait penggunaan dan cara melakukan konsultasi. |
| Precondition | Pengguna memilih menu bantuan pada halaman utama aplikasi. |
| Basic Path | <ol style="list-style-type: none">1. Pengguna memilih daftar bantuan yang ada2. Sistem menampilkan informasi bantuan yang dipilih |
| Alternative Path | - |
| Postcondition | Informasi bantuan ditampilkan diaplikasi |
| Exception Path | - |
| Extend | - |
| Include | - |

DEFINISI USE CASE MANAJEMEN BASIS PENGETAHUAN

| | |
|-------------------------|--|
| Use Case Name | Manajemen Basis Pengetahuan |
| Actors | <i>Knowledge Engginer</i> |
| Description | Use case ini berfungsi untuk mengatur manajemen terkait data-data yang akan digunakan dalam basis aturan. |
| Precondition | Pengguna memilih menu pengaturan basis pengetahuan pada halaman pengaturan. |
| Basic Path | <ol style="list-style-type: none"> 1. Pengguna memilih menu pengaturan basis pengetahuan 2. Sistem menampilkan menu-menu pengaturan untuk manajemen data pada basis pengetahuan. |
| Alternative Path | - |
| Postcondition | Menu-menu terkait pengaturan data basis pengetahuan ditampilkan |
| Exception Path | - |
| Extend | - |
| Include | Tambah data, Ubah data, Lihat detail data, Hapus data |

DEFINISI USE CASE TAMBAH DATA

| | |
|-------------------------|--|
| Use Case Name | Tambah Data |
| Actors | <i>Knowledge Engginer</i> |
| Description | Use case ini berfungsi untuk menambah data pada basis pengetahuan. |
| Precondition | Pengguna memilih menu tambah data pada halaman pengaturan basis pengetahuan. |
| Basic Path | <ol style="list-style-type: none"> 1. Pengguna memilih daftar diagnosa yang akan ditambahkan data 2. Pengguna mengisi form yang ada 3. Pengguna memilih tombol tambah data untuk menyimpan data 4. Sistem akan mengecek bila ada data yang sama sebelum disimpan 5. Sistem menyimpan data |
| Alternative Path | - |
| Postcondition | Data yang dimasukkan oleh pengguna disimpan pada database basis pengetahuan |
| Exception Path | - |
| Extend | - |
| Include | - |

DEFINISI USE CASE UBAH DATA

| | |
|-------------------------|---|
| Use Case Name | Ubah Data |
| Actors | <i>Knowledge Engginer</i> |
| Description | Use case ini berfungsi untuk mengubah data yang ada pada basis pengetahuan. |
| Precondition | Pengguna memilih menu ubah data |
| Basic Path | <ol style="list-style-type: none"> 1. Pengguna memilih daftar diagnosa yang diinginkan 2. Sistem menampilkan data-data yang terdapat di daftar diagnose yang dipilih 3. Pengguna memilih data yang diinginkan 4. Pengguna melakukan perubahan pada data di form yang ada 5. Pengguna memilih tombol ubah untuk menyimpan data 6. Sistem mengecek bila ada data yang sama 7. Sistem meyimpan data |
| Alternative Path | - |
| Postcondition | Data yang diubah disimpan didatabase |
| Exception Path | - |
| Extend | Tambah Data |
| Include | - |

DEFINISI USE CASE LIHAT DETAIL DATA

| | |
|-------------------------|--|
| Use Case Name | Lihat Detail Data |
| Actors | <i>Knowledge Engginer</i> |
| Description | Use case ini berfungsi untuk melihat detail dari data yang tersimpan pada database basis pengetahuan. |
| Precondition | Pengguna memilih menu lihat data |
| Basic Path | <ol style="list-style-type: none">1. Pengguna memilih daftar diagnosa yang ada2. Sistem menampilkan data-data yang ada didaftar diagnosa yang dipilih3. Pengguna memilih data yang akan dilihat detailnya4. Sistem menampilkan form detail data |
| Alternative Path | - |
| Postcondition | Detail data dari data yang dipilih ditampilkan diaplikasi |
| Exception Path | - |
| Extend | Tambah Data |
| Include | - |

DEFINISI USE CASE HAPUS DATA

| | |
|-------------------------|--|
| Use Case Name | Hapus Data |
| Actors | <i>Knowledge Engginer</i> |
| Description | Use case ini berfungsi untuk menghapus data yang ada. |
| Precondition | Pengguna memilih menu hapus data pada menu pengaturan basis pengetahuan. |
| Basic Path | <ol style="list-style-type: none"> 1. Pengguna memilih diagnosa yang akan dihapus datanya 2. Sistem menampilkan data-data yang ada 3. Pengguna memilih data yang akan dihapus 4. Sistem menampilkan detail data 5. Pengguna memilih tombol hapus 6. Sistem memberikan konfirmasi 7. Pengguna melakukan konfirmasi |
| Alternative Path | <ol style="list-style-type: none"> 1. Pengguna memilih diagnosa yang akan dihapus datanya 2. Pengguna menekan tombol hapus 3. Sistem memberikan konfirmasi penghapusan seluruh data 4. Pengguna melakukan konfirmasi |
| Postcondition | Data dihapus dari database |
| Exception Path | - |
| Extend | Tambah data |
| Include | - |

DEFINISI USE CASE MANAJEMEN BASIS ATURAN

| | |
|-------------------------|--|
| Use Case Name | Manjemen Basis Aturan |
| Actors | <i>Knowledge Engginer</i> |
| Description | Use case ini berfungsi untuk memanajemen aturan-aturan yang ada. |
| Precondition | Pengguna memilih menu pengaturan basis aturan di menu pengaturan. |
| Basic Path | <ol style="list-style-type: none">1. Pengguna memilih menu pengaturan basis aturan2. Sistem menampilkan daftar menu pengaturan untuk basis aturan |
| Alternative Path | - |
| Postcondition | Menu-menu terkait pengaturan basis aturan tertampilkan dilayar |
| Exception Path | - |
| Extend | - |
| Include | Buat aturan, ubah aturan, lihat aturan, hapus aturan. |

DEFINISI USE CASE BUAT ATURAN

| | |
|-------------------------|---|
| Use Case Name | Buat aturan |
| Actors | <i>Knowledge Engginer</i> |
| Description | Use case ini berfungsi untuk menyusun data-data menjadi sebuah aturan. |
| Precondition | Pengguna memilih menu buat aturan pada pengeturan basis aturan |
| Basic Path | <ol style="list-style-type: none"> 1. Sistem menampilkan daftar diagnosa yang akan dibuat aturannya 2. Pengguna memilih daftar diagnose yang diinginkan 3. Sistem menampilkan form untuk menyusun aturan 4. Pengguna menyusun aturan dari data-data yang sudah ada 5. Pengguna memilih tombol buat untuk menyimpan aturan yang telah disusun 6. Sistem memberikan konfirmasi 7. Pengguna melakukan konfirmasi 8. Sistem mengecek bila aturan sudah pernah dibuat atau belum 9. Sistem mengecek bila data yang dipakai sama |
| Alternative Path | - |
| Postcondition | Aturan yang dibuat disimpan di database |
| Exception Path | - |
| Extend | - |
| Include | - |

DEFINISI USE CASE UBAH ATURAN

| | |
|-------------------------|---|
| Use Case Name | Ubah Aturan |
| Actors | <i>Knowledge Engginer</i> |
| Description | Use case ini berfungsi untuk mengubah aturan yang telah dibuat. |
| Precondition | Pengguna memilih menu ubah aturan pada menu pengaturan basis aturan. |
| Basic Path | <ol style="list-style-type: none"> 1. Pengguna memilih daftar diagnosa yang akan diubah aturan 2. Sistem menampilkan node-node aturan yang telah dibuat 3. Pengguna memilih node yang akan diubah 4. Sistem menampilkan form ubah aturan 5. Pengguna mengubah node aturan yang ada 6. Pengguna memilih tombol ubah 7. Sistem memberikan konfirmasi 8. Pengguna melakukan konfirmasi 9. Sistem melakukan pengecekan bila ada data yang sama dibuatkan untuk membuat node aturan |
| Alternative Path | - |
| Postcondition | Node aturan disimpan didatabase |
| Exception Path | - |
| Extend | Buat aturan |
| Include | - |

DEFINISI USE CASE LIHAT ATURAN

| | |
|-------------------------|---|
| Use Case Name | Lihat Aturan |
| Actors | <i>Knowledge Engginer</i> |
| Description | Use case ini berfungsi untuk melihat detail node-node aturan yang telah dibuat dan data yang menyusunnya. |
| Precondition | Pengguna memilih menu lihat aturan pada menu pengaturan basis aturan. |
| Basic Path | <ol style="list-style-type: none"> 1. Sistem menampilkan daftar diagnosa yang ada. 2. Pengguna memilih daftar diagnosa yang diinginkan. 3. Sistem menampilkan node-node aturan yang telah dibuat 4. Pengguna memilih node yang diinginkan. 5. Sistem menampilkan detail node yang dipilih. |
| Alternative Path | - |
| Postcondition | Detail node aturan dan data-data yang terhubung ditampilkan di aplikasi. |
| Exception Path | - |
| Extend | Buat aturan |
| Include | - |

DEFINISI USE CASE HAPUS ATURAN

| | |
|-------------------------|--|
| Use Case Name | Hapus Aturan |
| Actors | <i>Knowledge Engginer</i> |
| Description | Use case ini berfungsi untuk menghapus aturan yang telah dibuat. |
| Precondition | Pengguna memilih menu hapus aturan pada menu pengaturan basis aturan. |
| Basic Path | <ol style="list-style-type: none">1. Sistem menampilkan daftar diagnosa2. Pengguna memilih diagnosa yang diinginkan3. Pengguna memilih tombol hapus4. Sistem memberikan konfirmasi5. Pengguna melakukan konfirmasi |
| Alternative Path | - |
| Postcondition | Aturan yang ada dihapus dari database |
| Exception Path | - |
| Extend | Buat aturan |
| Include | - |

Lampiran Diagram Aktivitas

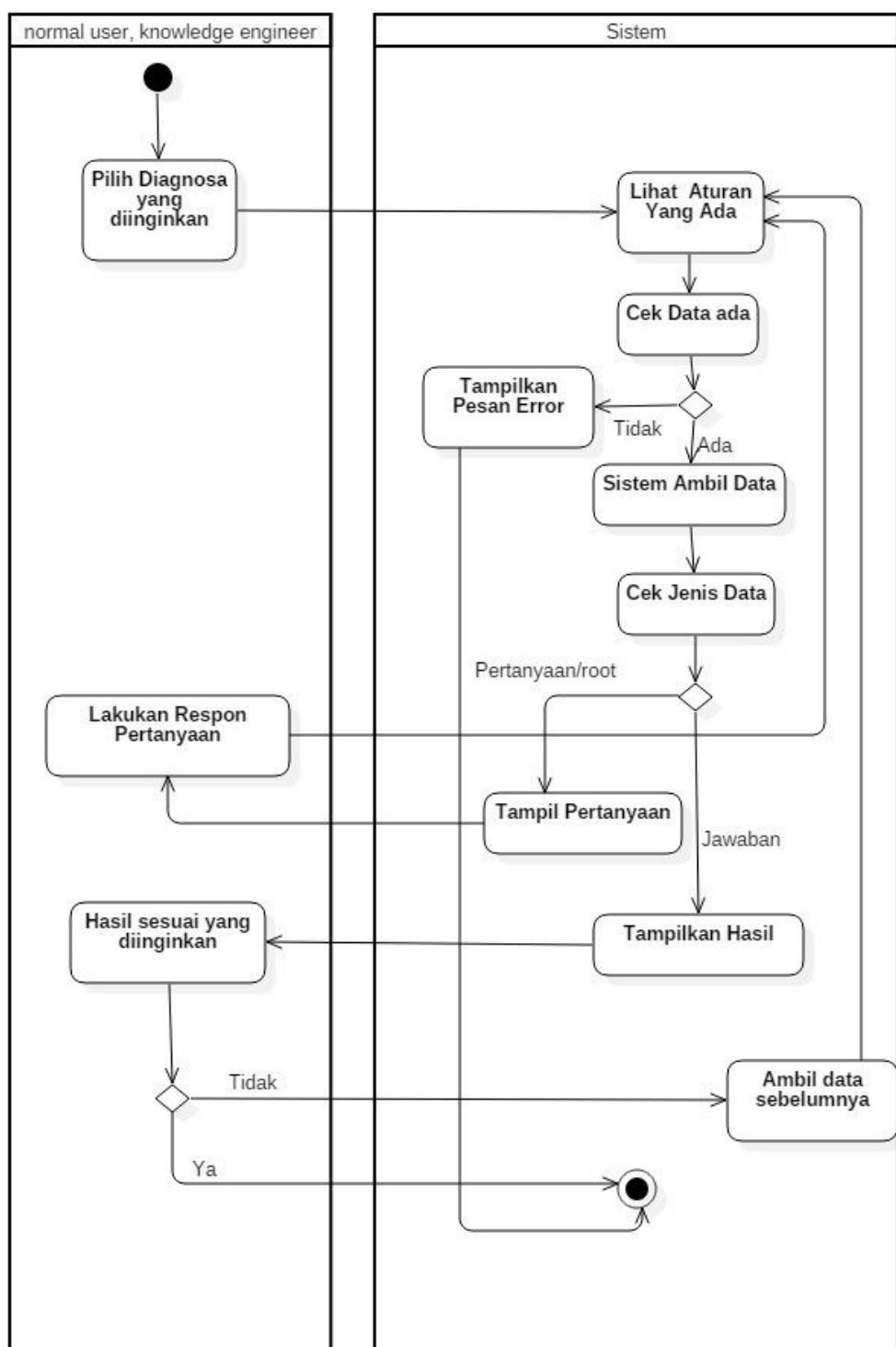


Diagram aktivitas melakukan konsultasi

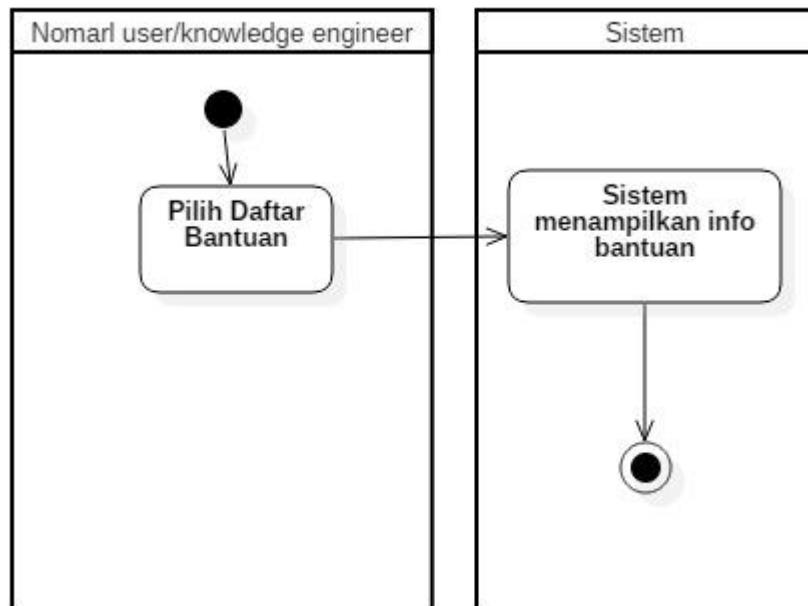


Diagram aktivitas melihat bantuan

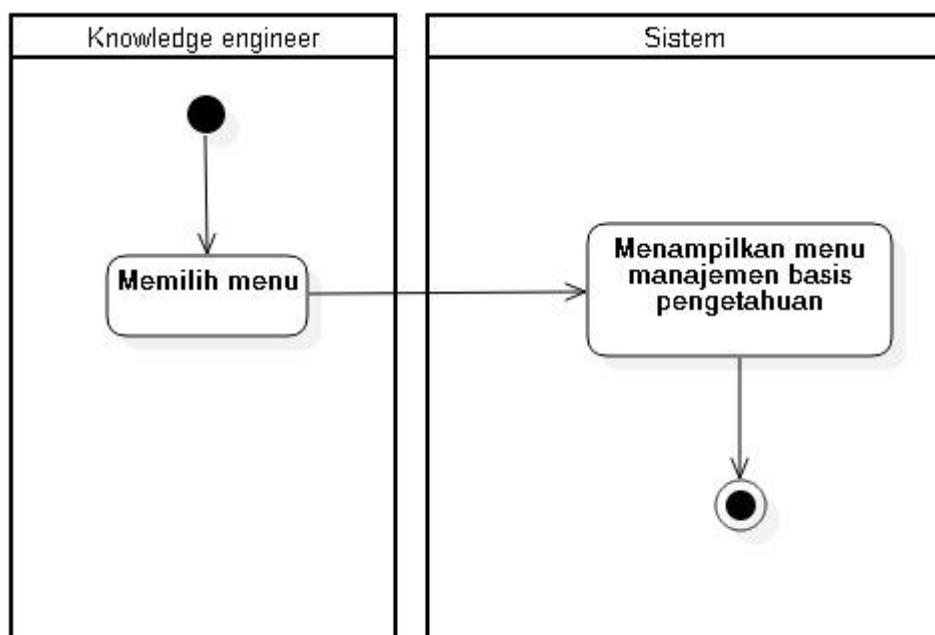


Diagram aktifitas manajemen basis pengetahuan

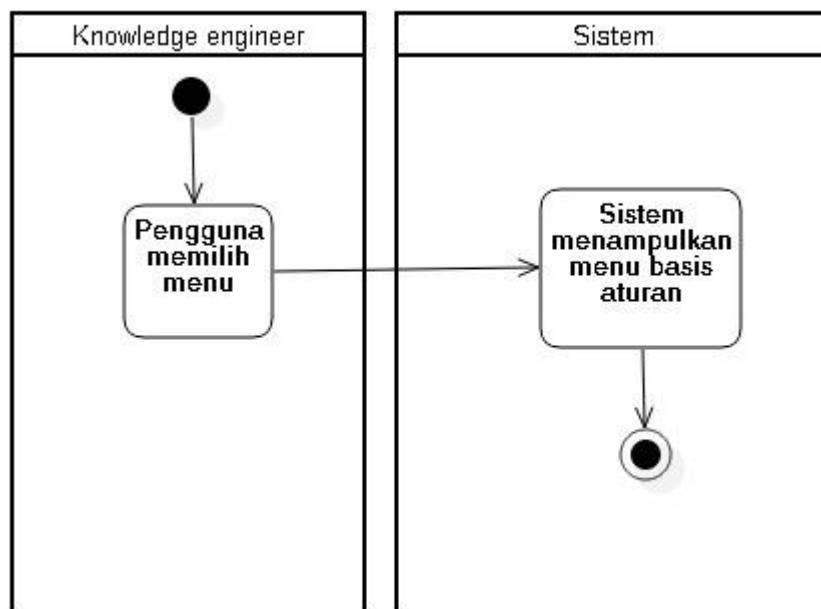


Diagram aktifitas manajemen basis aturan

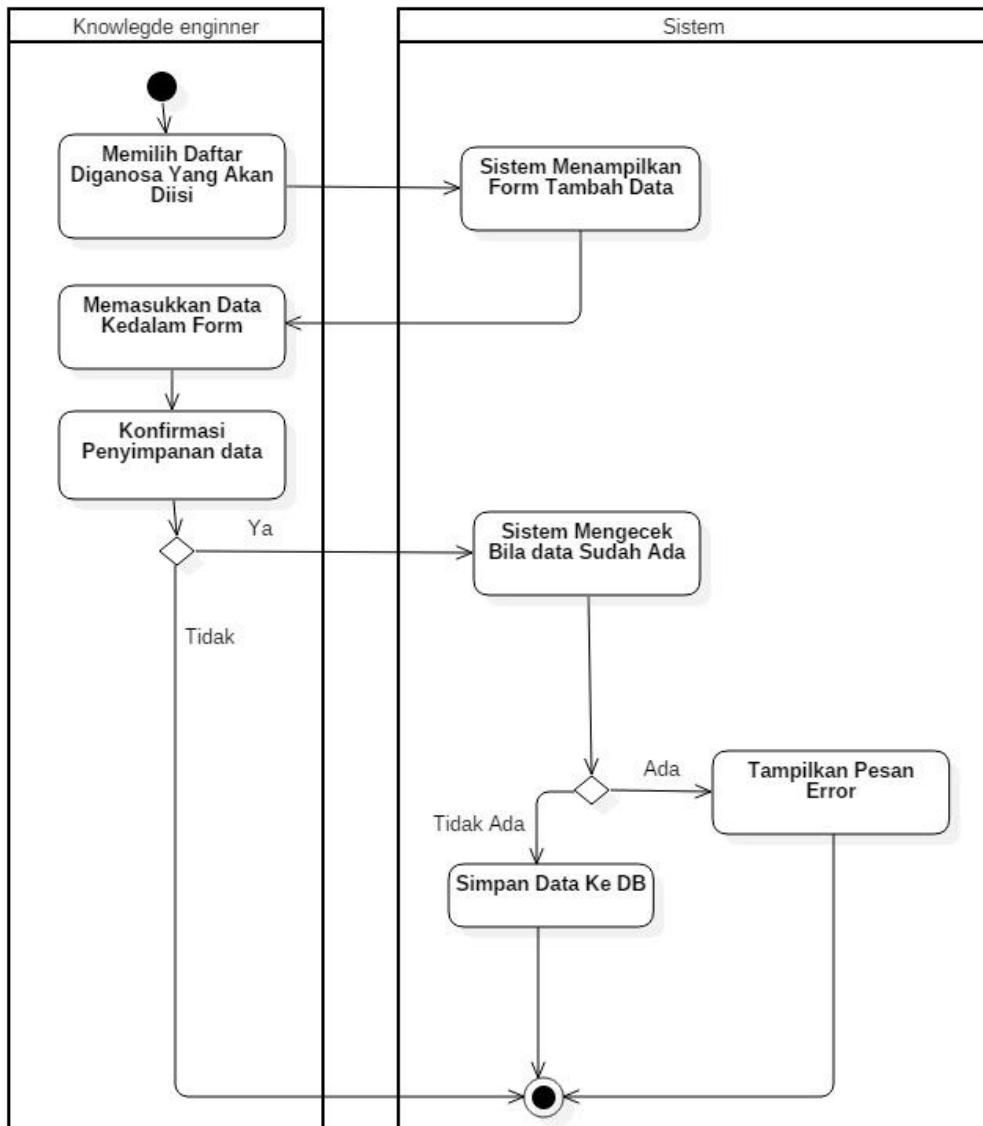


Diagram aktivitas tambah data

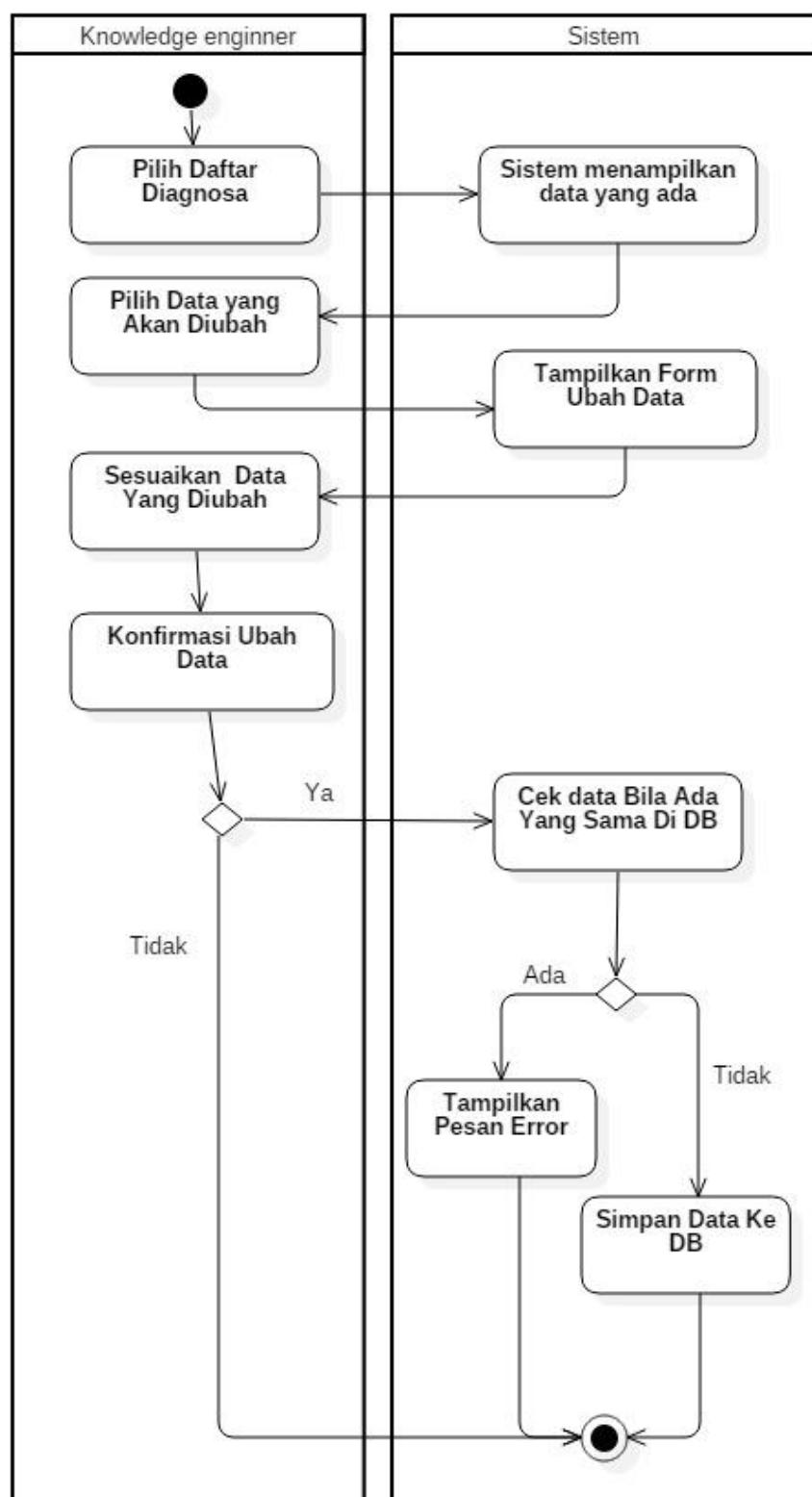


Diagram aktivitas ubah data

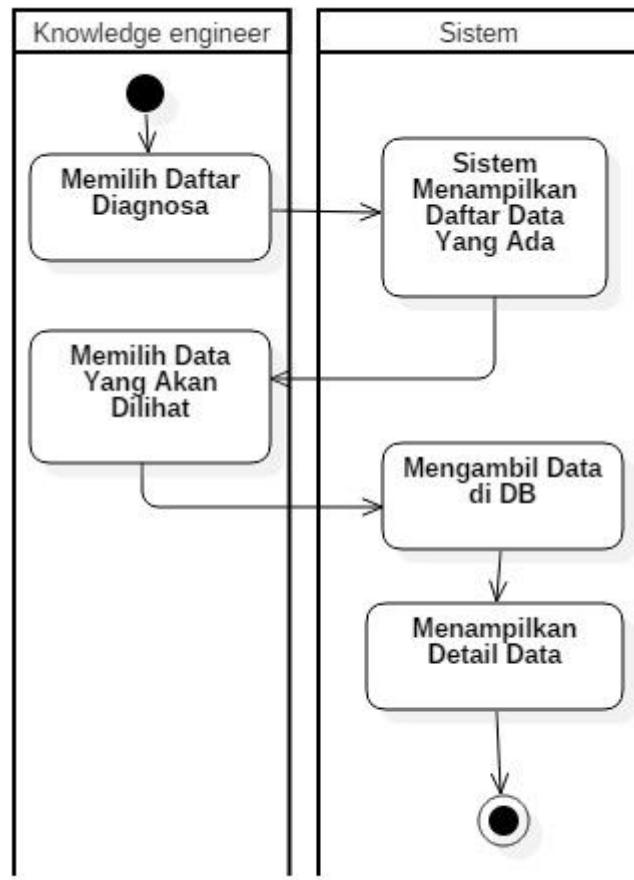


Diagram aktivitas lihat detail data

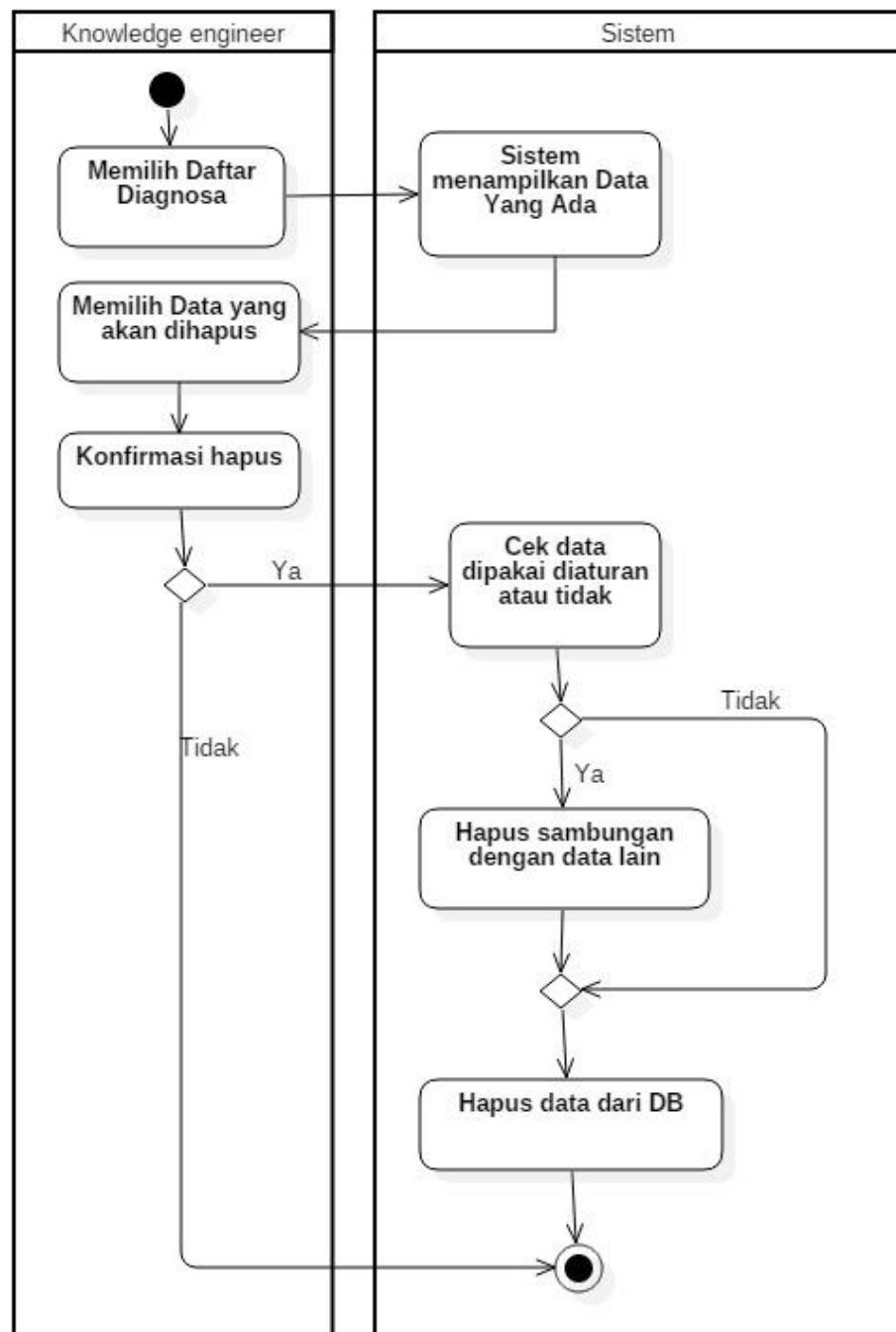


Diagram aktivitas hapus data

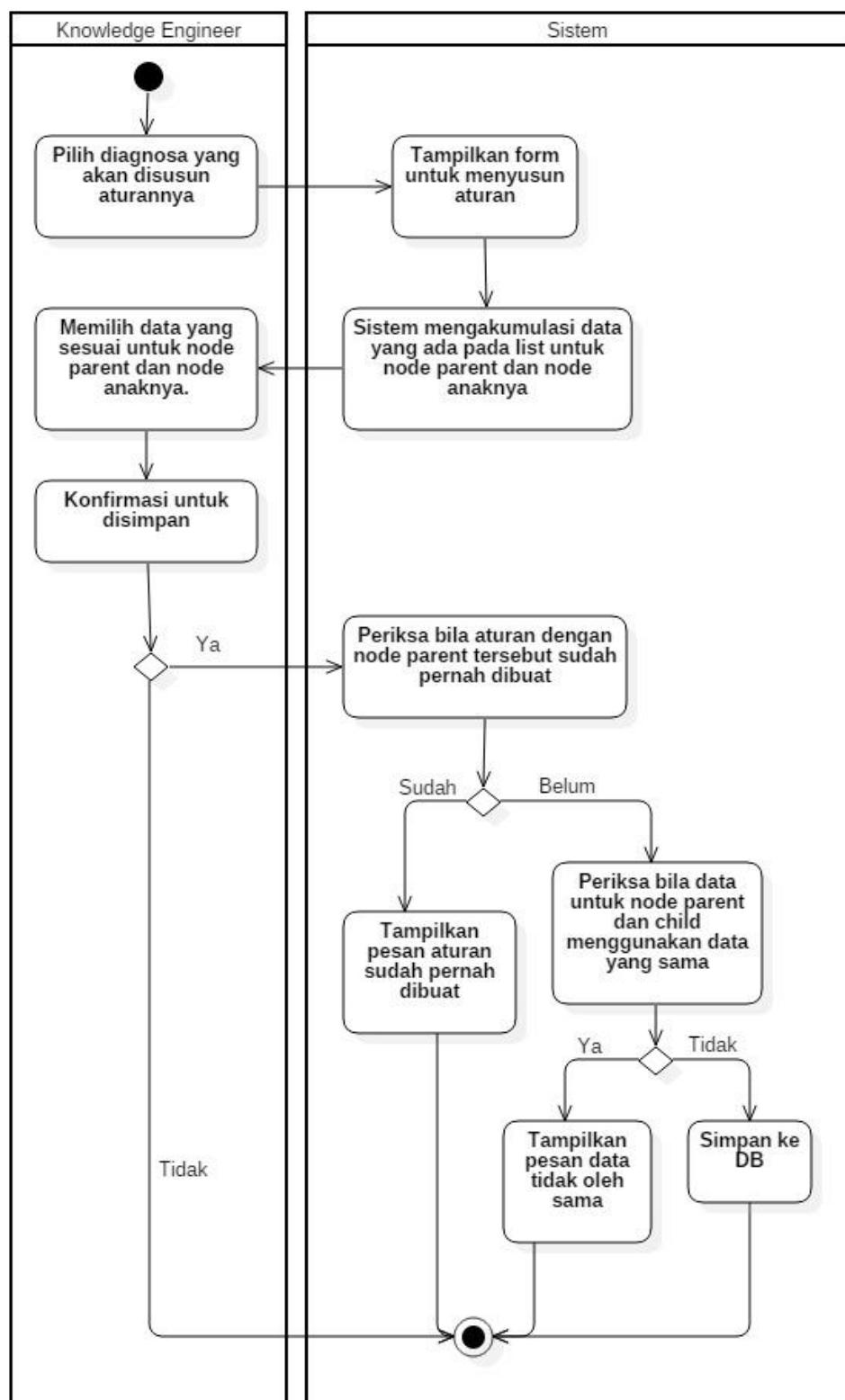


Diagram aktivitas buat aturan

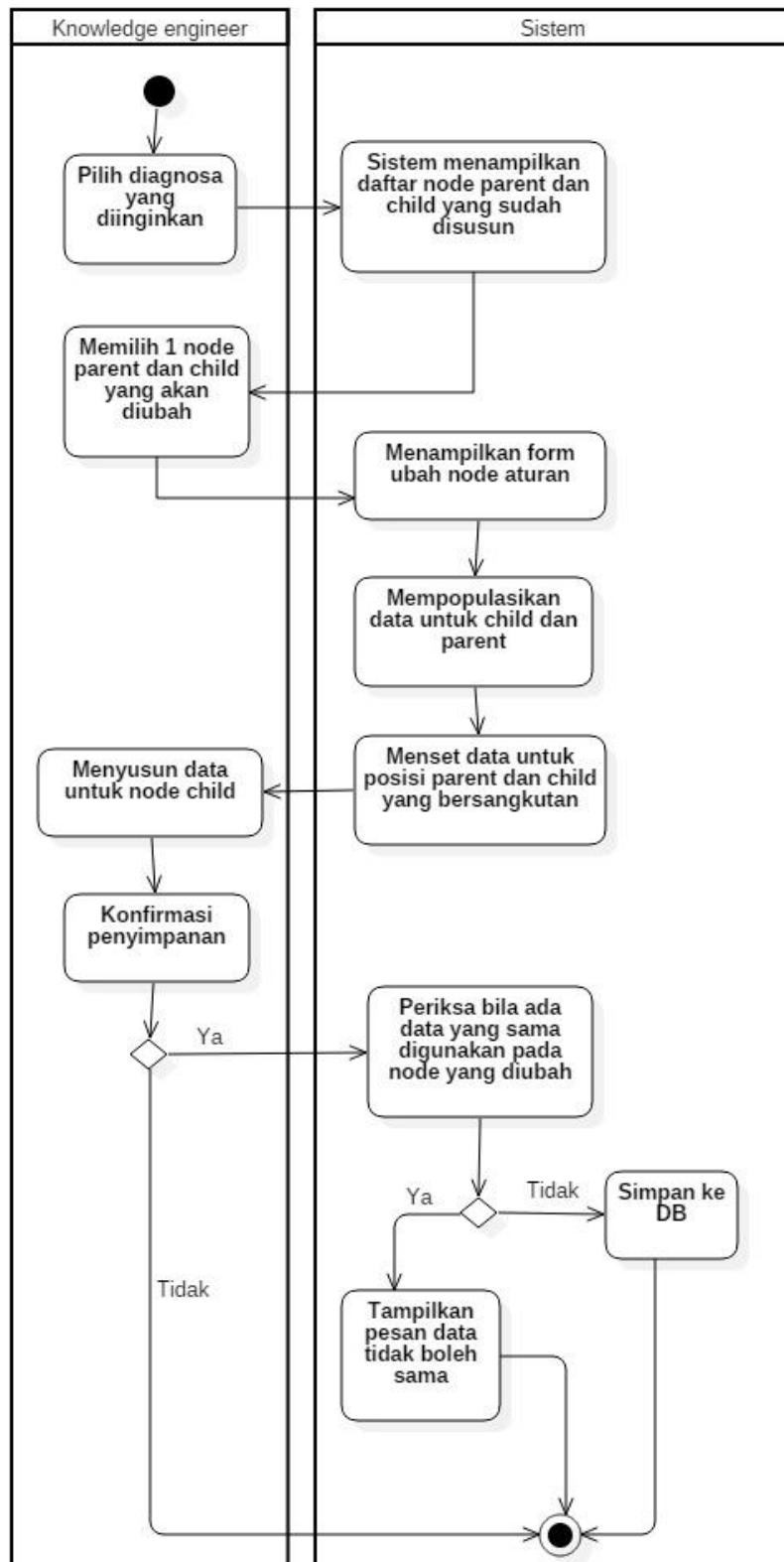


Diagram aktivitas ubah aturan

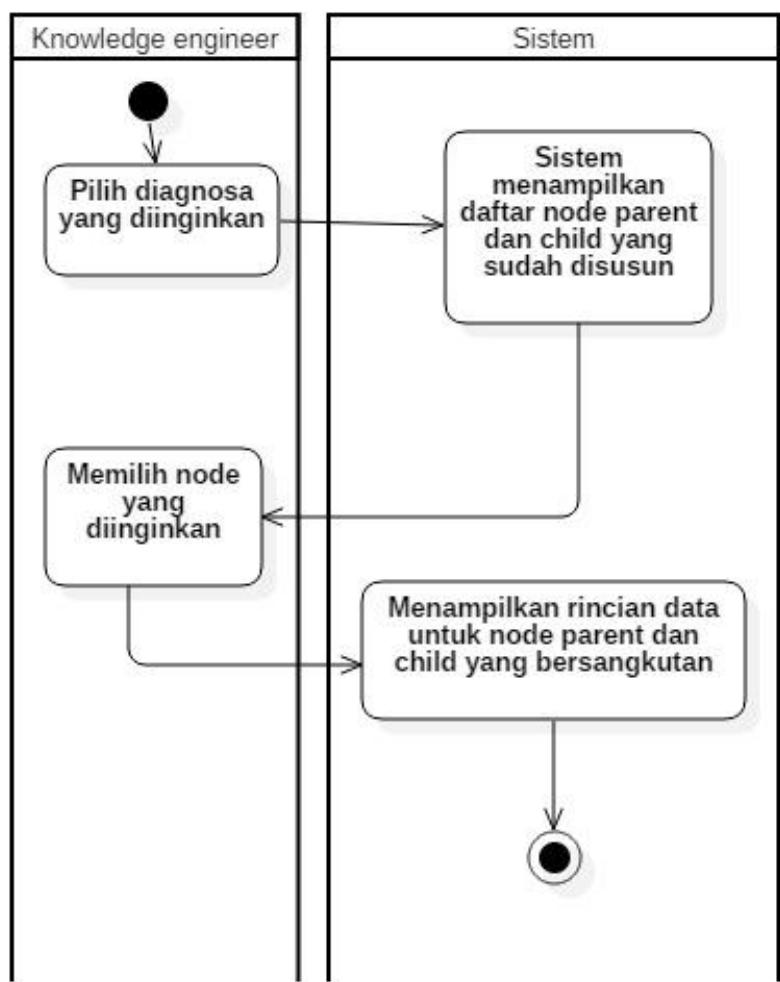


Diagram aktivitas lihat aturan

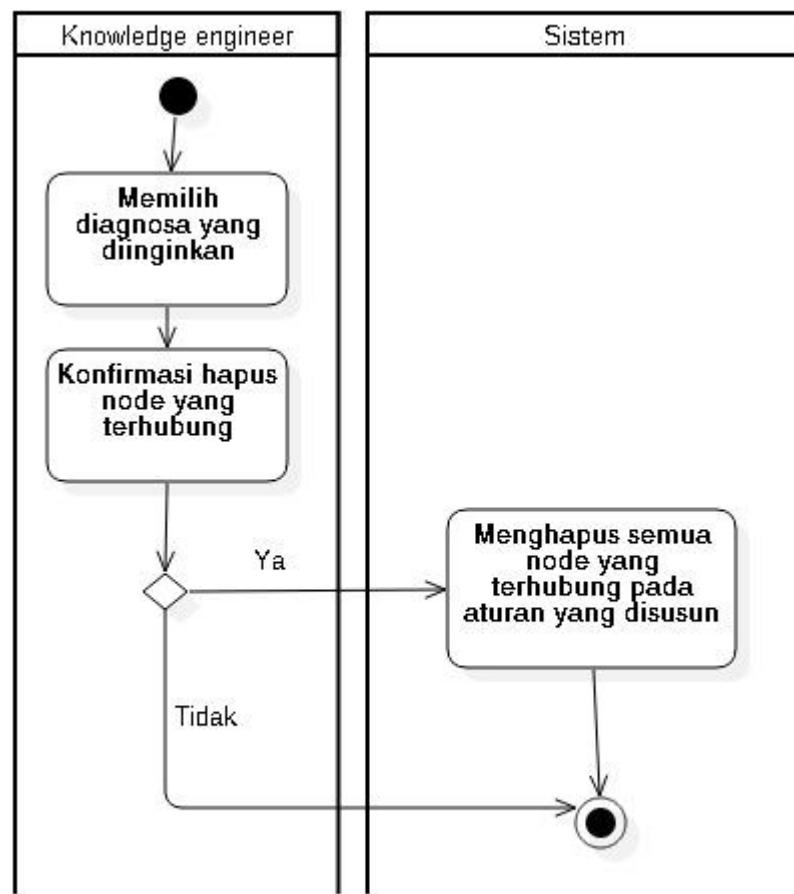


Diagram aktivitas hapus aturan

Lampiran Diagram Sekuensial

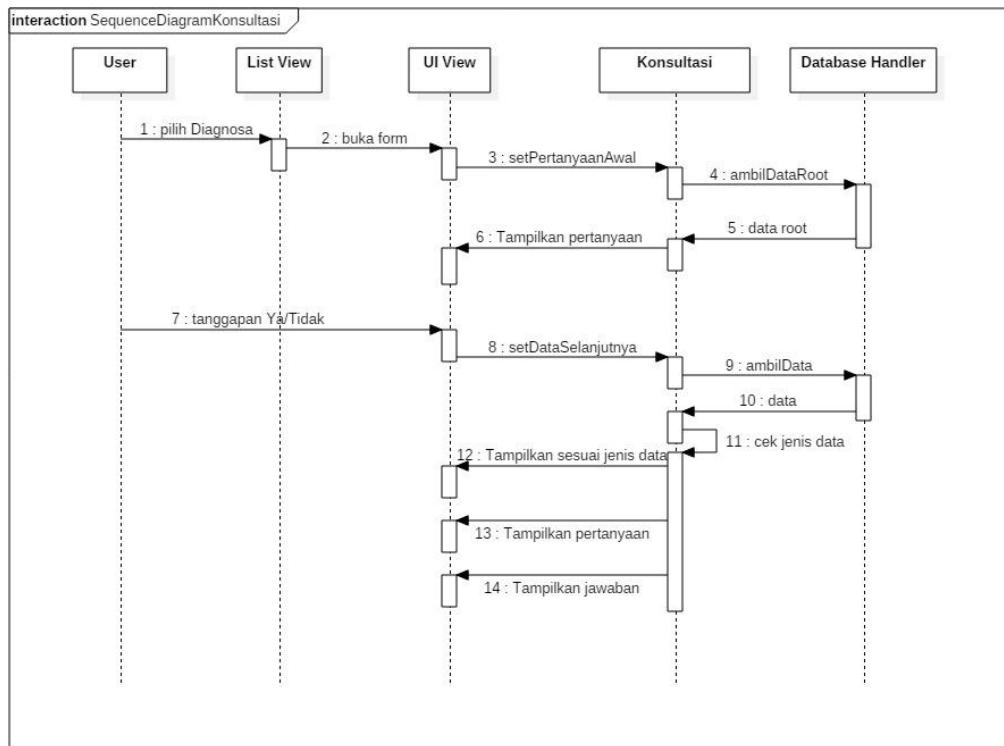


Diagram sekuensial melakukan konsultasi

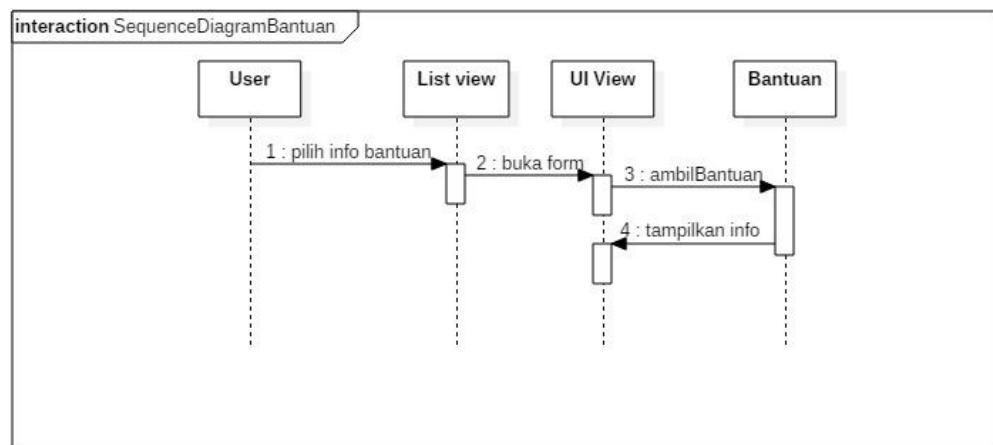


Diagram sekuensial melihat bantuan

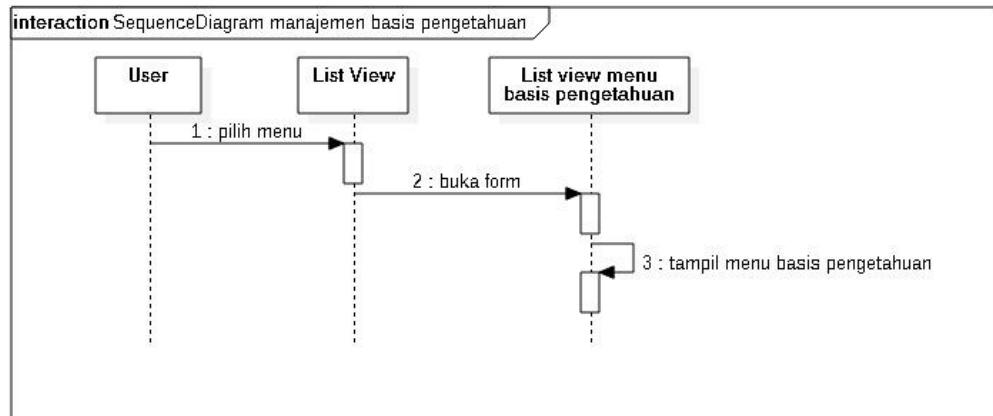


Diagram sekuensial manajemen basis pengetahuan

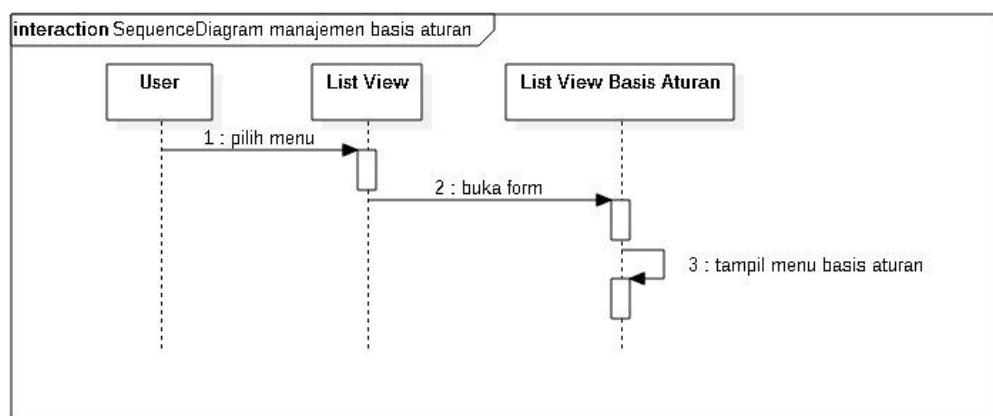


Diagram sekuensial manajemen basis aturan

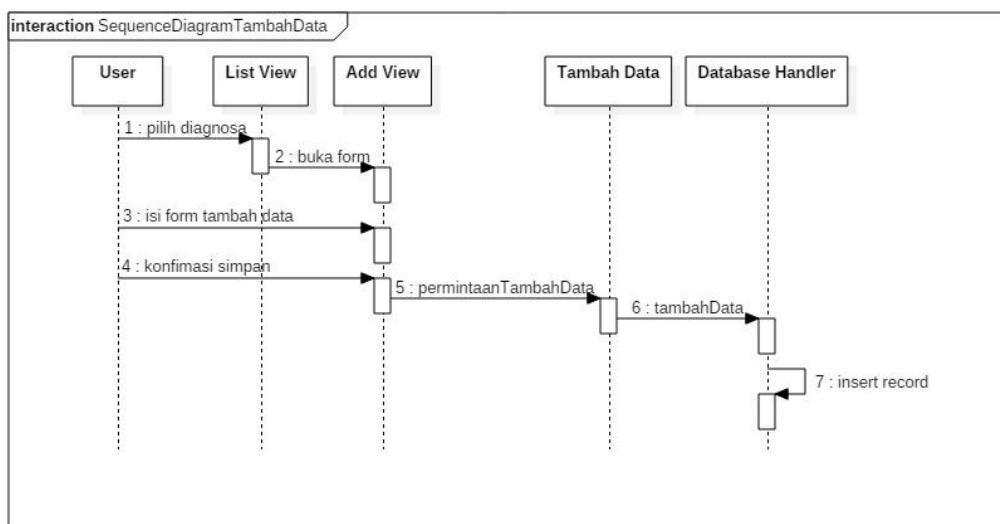


Diagram sekuensial tambah data

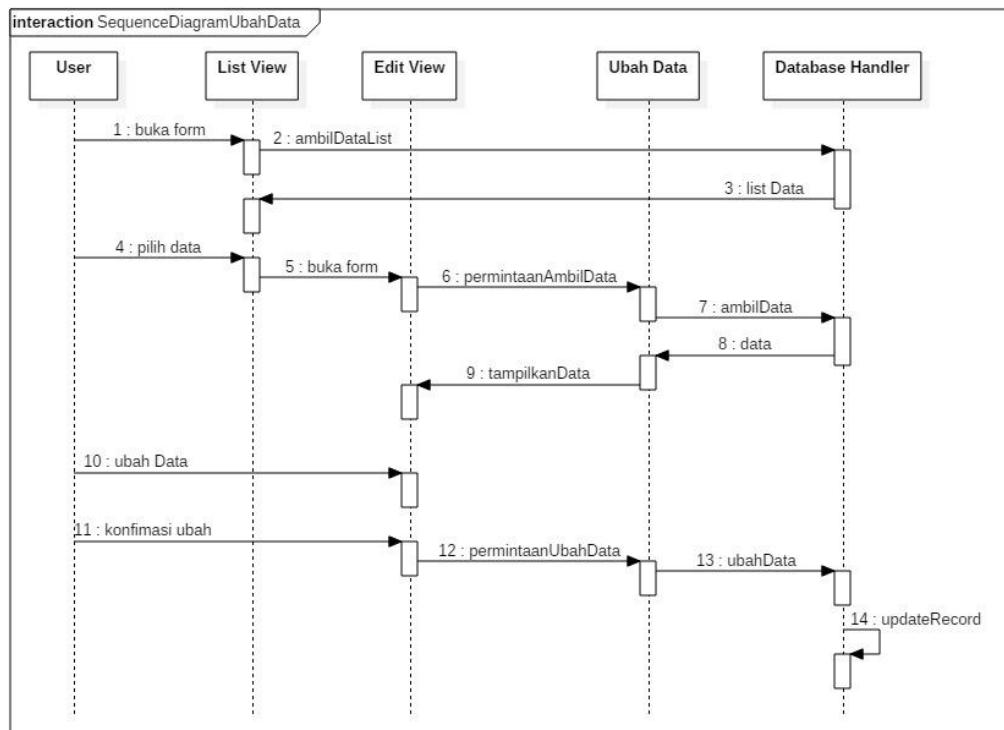


Diagram sekuensial ubah data

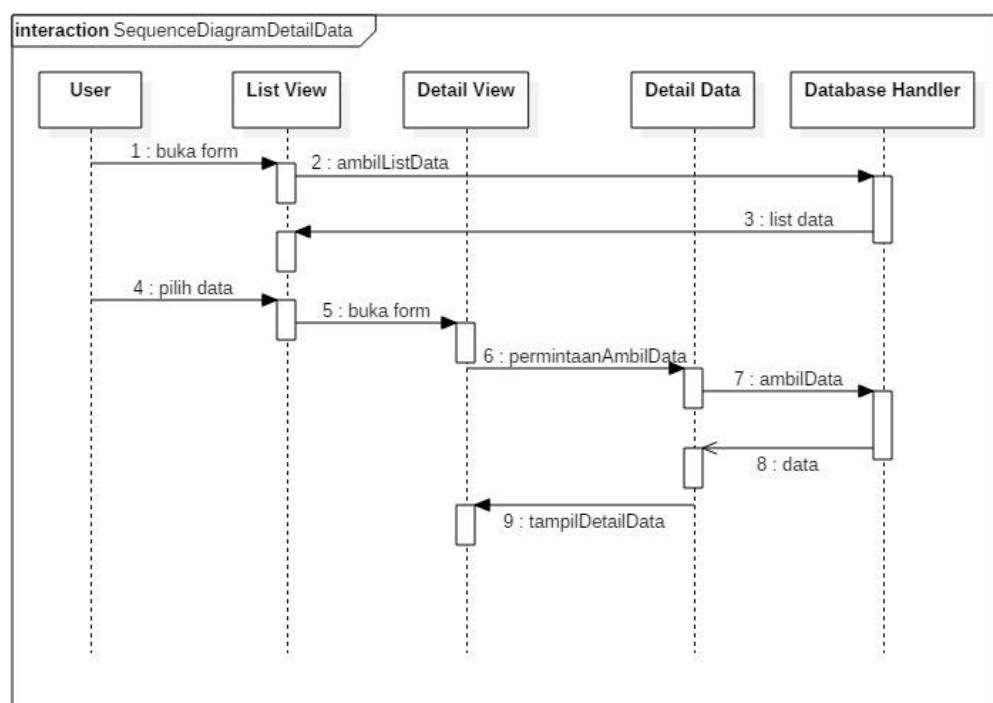


Diagram sekuensial detail data

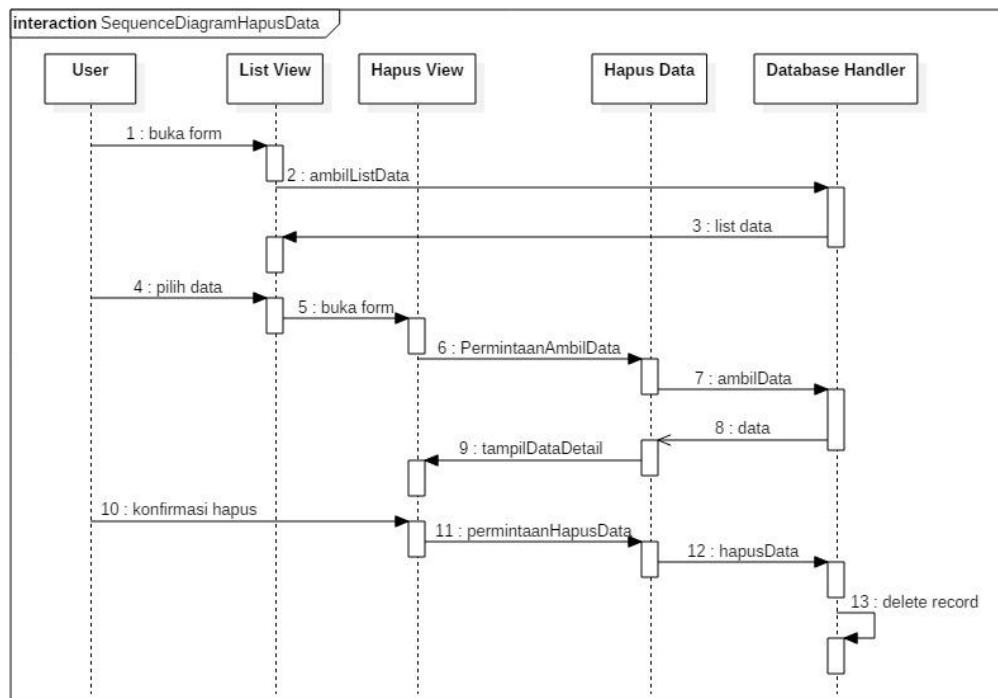


Diagram sekuensial hapus data

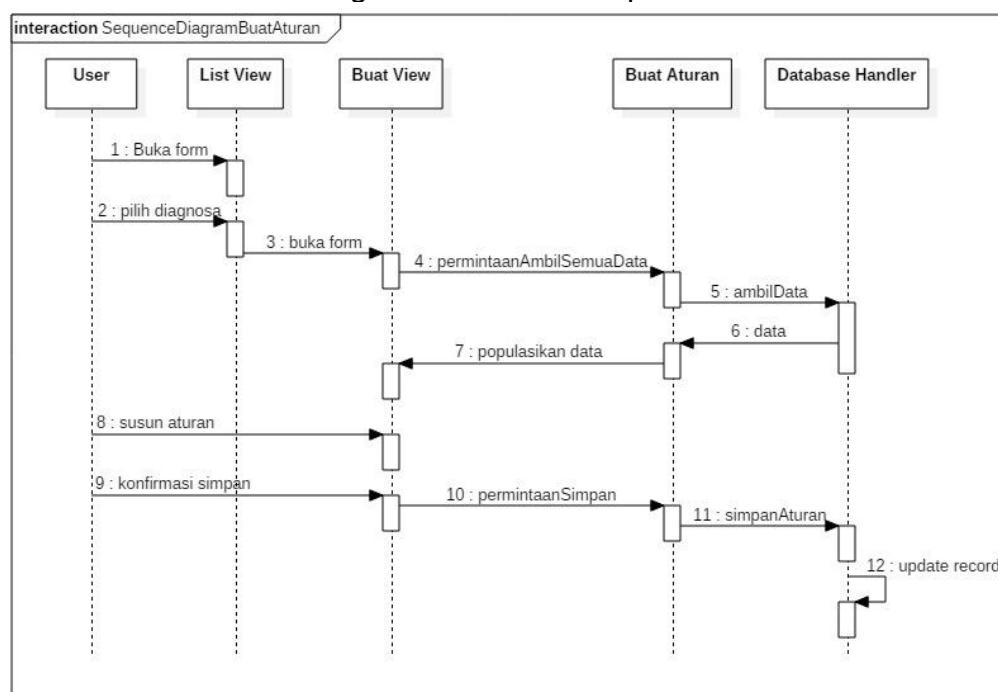


Diagram sekuensial buat aturan

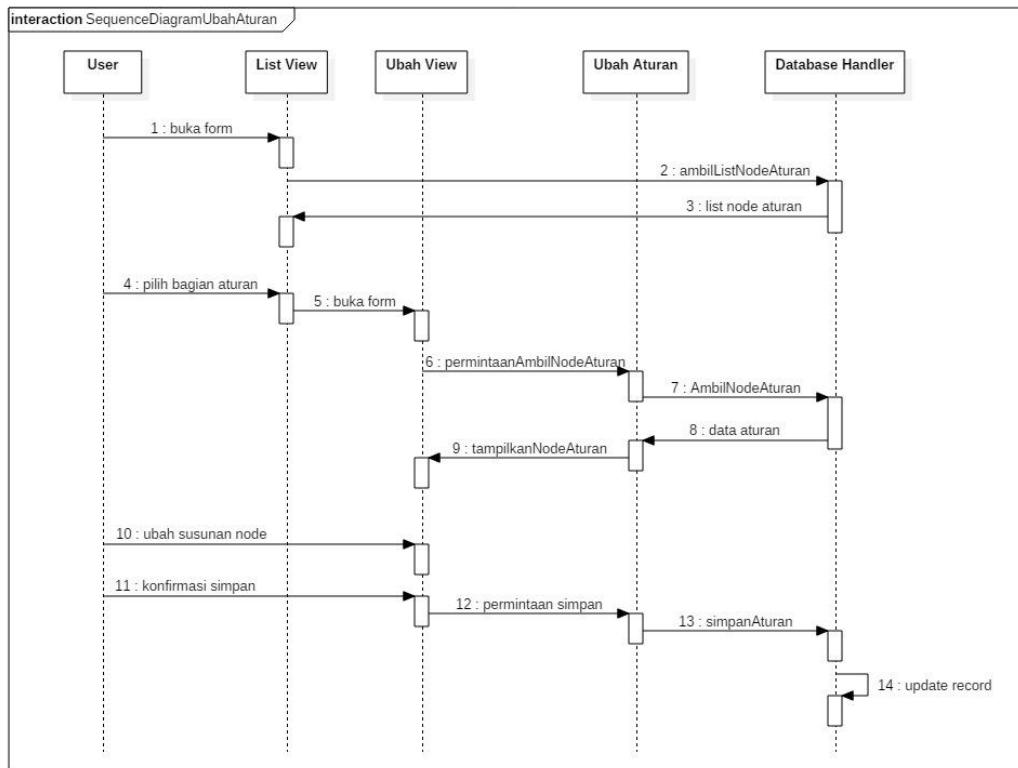


Diagram sekuensial ubah aturan

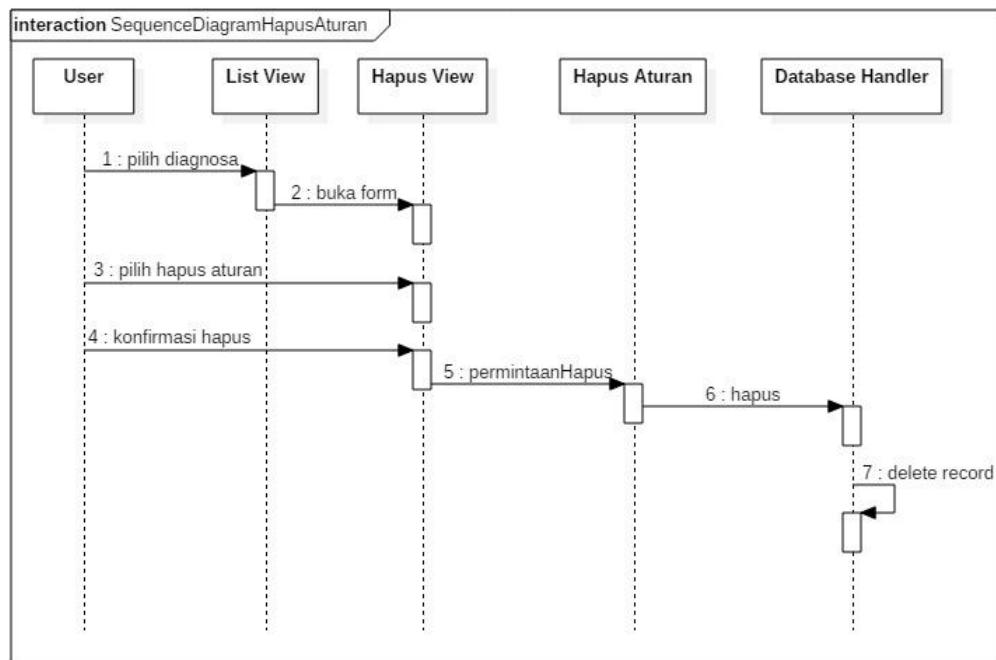


Diagram sekuensial hapus aturan

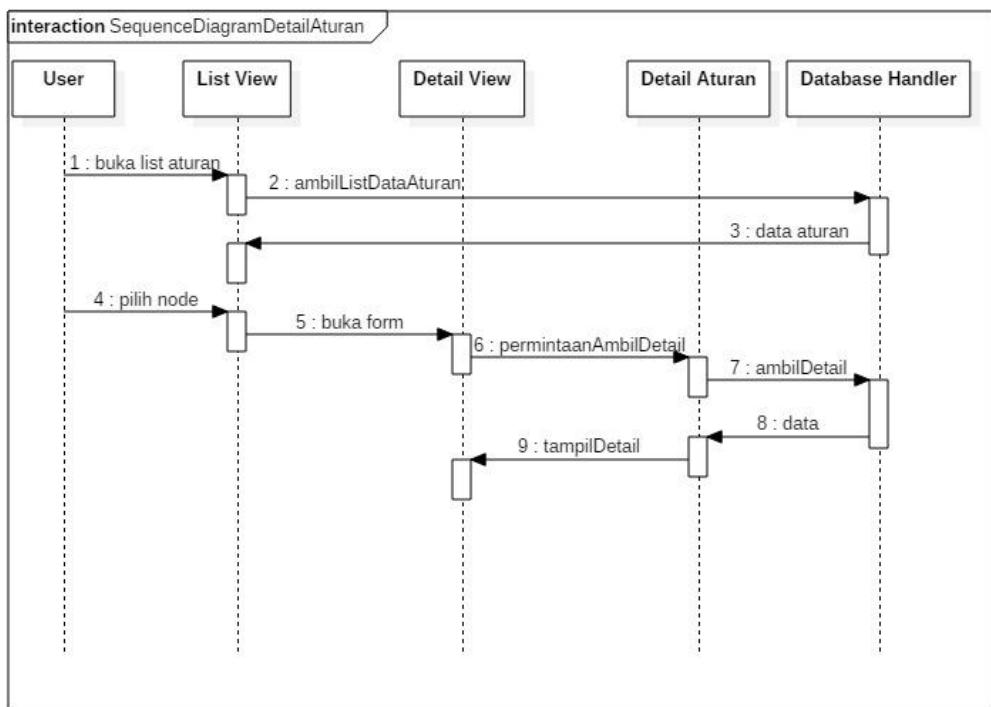
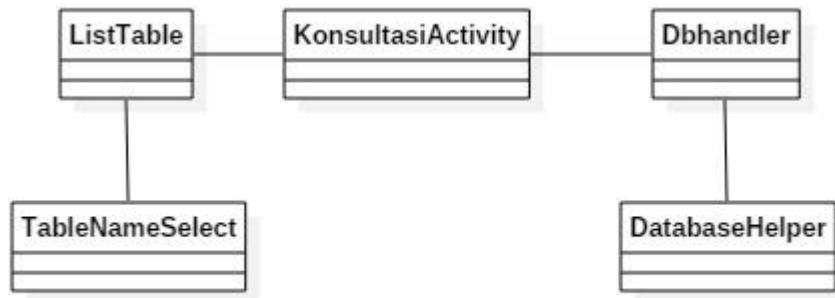


Diagram sekuensial lihat detail aturan

LAMPIRAN DESAIN DIAGRAM KELAS



. Diagram kelas untuk melakukan konsultasi

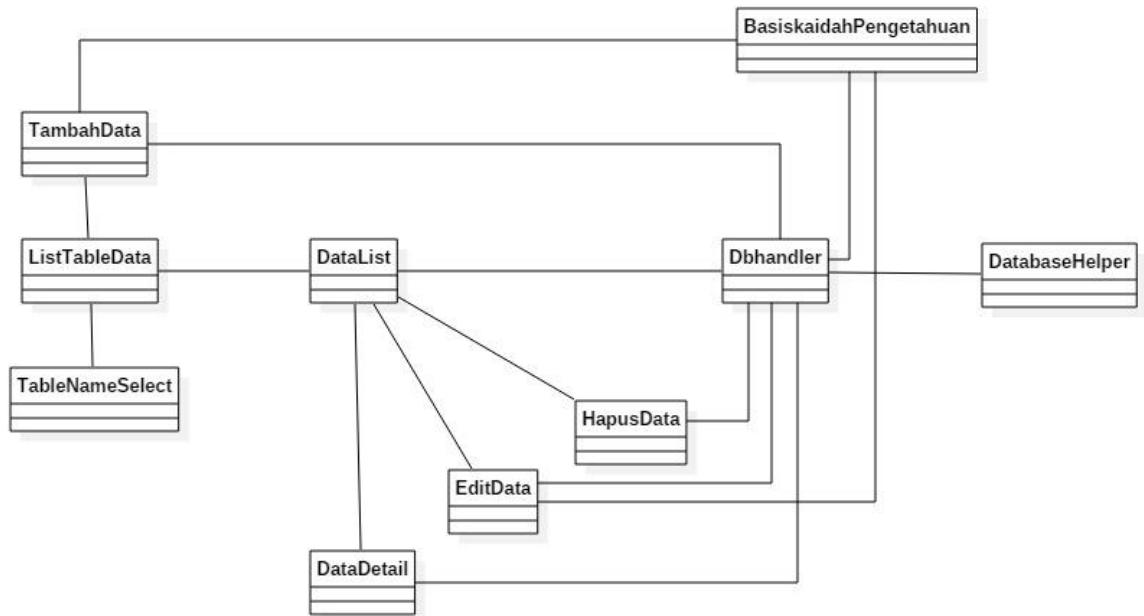


Diagram kelas untuk manajemen basis pengetahuan

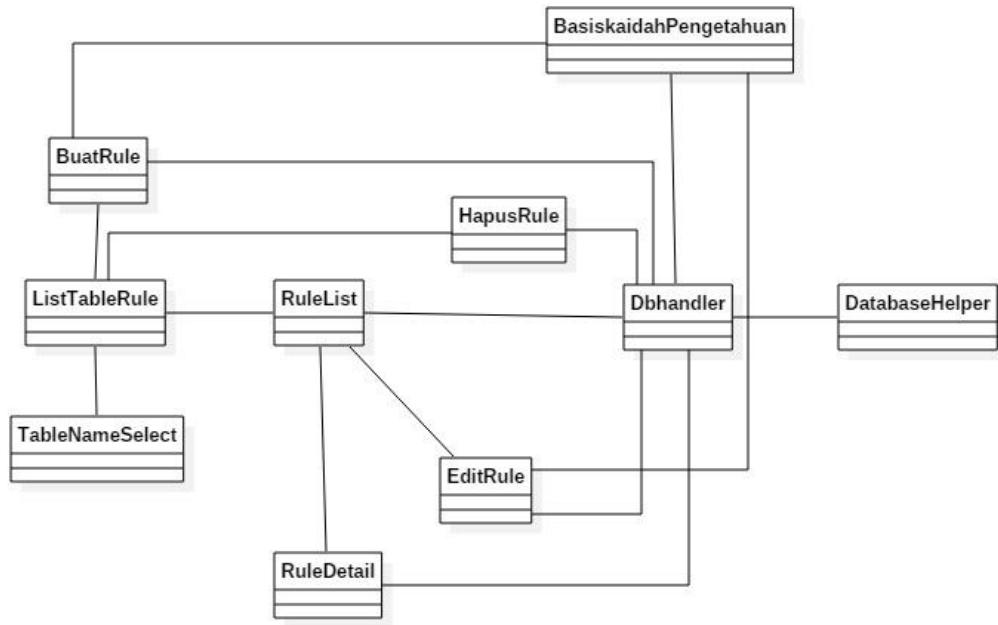


Diagram kelas untuk manajemen basis aturan

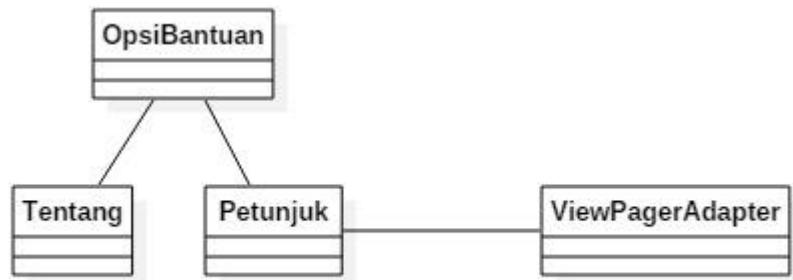
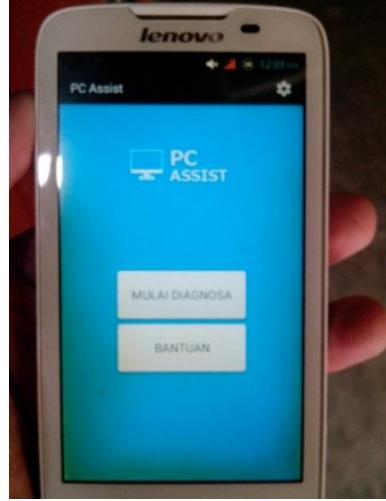
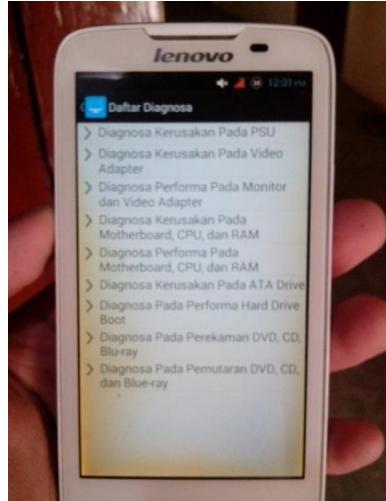
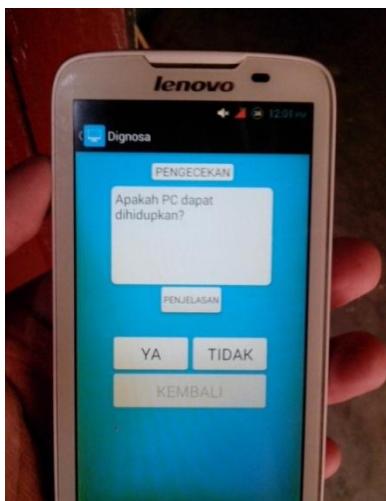
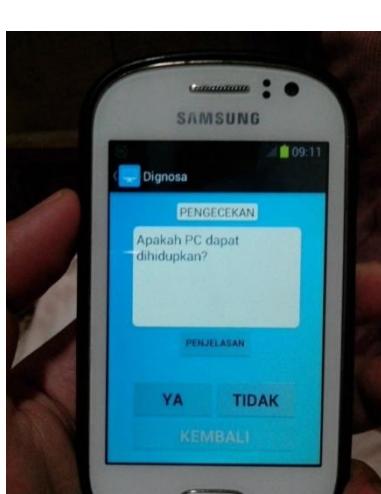


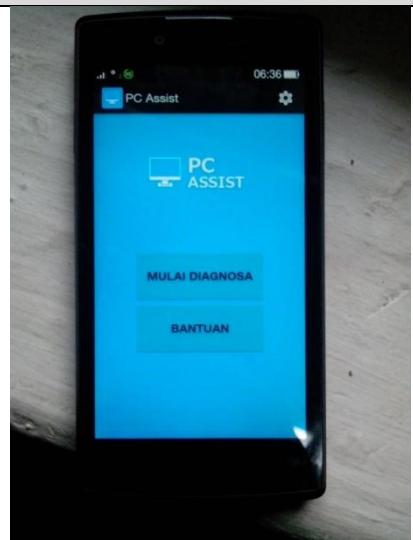
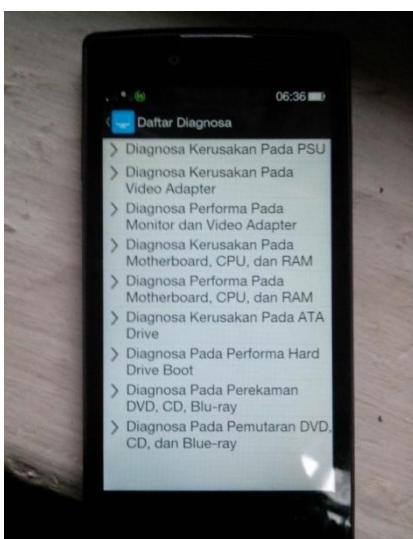
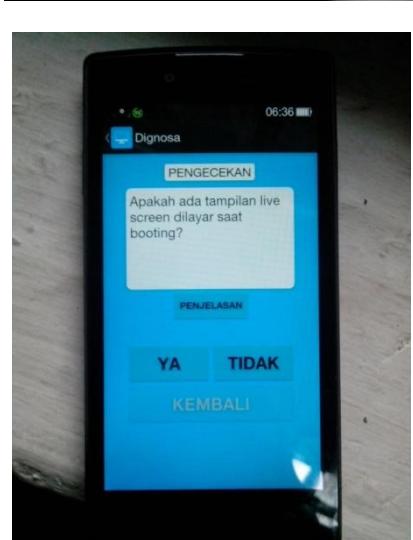
Diagram kelas untuk melihat bantuan

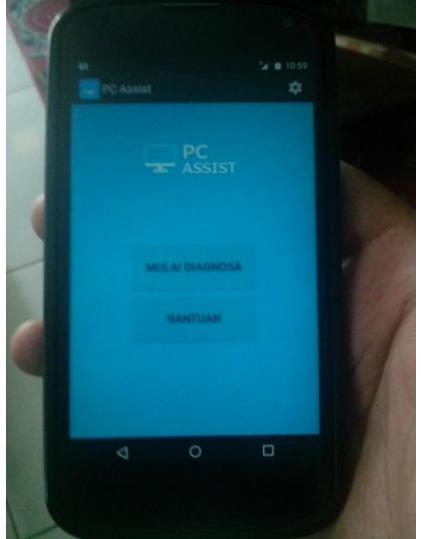
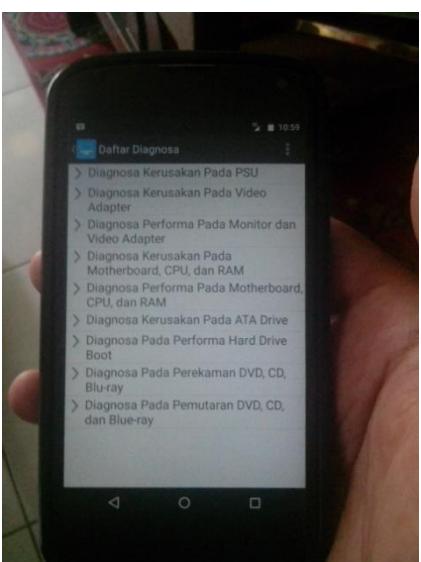
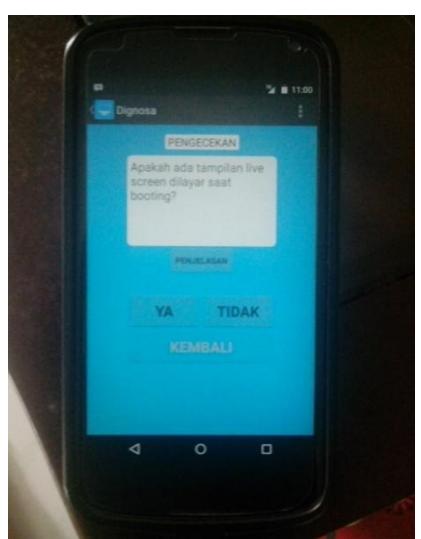
LAMPIRAN PENGUJIAN PORTABILITY

PENGUJIAN PORTABILITY APLIKASI SECARA MANUAL

| No | Nama Perangkat | Gambar Pengujian | Hasil Pengujian | |
|----|----------------|---|-------------------|---------------------|
| | | | Dapat Dipasang | Dapat Dijalankan |
| 1 | Lenovo A820 |    | Berhasil dipasang | Berhasil dijalankan |

| No | Nama Perangkat | Gambar Pengujian | Hasil Pengujian | |
|----|--------------------------------|---|-------------------|---------------------|
| | | | Dapat Dipasang | Dapat Dijalankan |
| 2 | Samsung galaxy fame (GT-S6810) |    | Berhasil dipasang | Berhasil dijalankan |

| No | Nama Perangkat | Gambar Pengujian | Hasil Pengujian | |
|----|----------------|---|-------------------|---------------------|
| | | | Dapat Dipasang | Dapat Dijalankan |
| 3 | Oppo Neo 3 |    | Berhasil dipasang | Berhasil dijalankan |

| No | Nama Perangkat | Gambar Pengujian | Hasil Pengujian | |
|----|----------------|---|-------------------|---------------------|
| | | | Dapat Dipasang | Dapat Dijalankan |
| 4 | Nexus 4 |    | Berhasil dipasang | Berhasil dijalankan |

PENGUJIAN PORTABILITY APLIKASI DENGAN CARA COULD TESTING

Test Run 8 in [Pc Assist](#) project

✔ FINISHED

| | | | | | |
|------------------------|---|--------------------|----------------------------|-------------------|---------------------|
| Executed devices: | 5 | Project type: | Android | Application: | com.nafis.pcass... |
| Devices with warnings: | 0 | Starting time: | 22.12.2015 01:33:51 | Tests: | App Crawler Compare |
| Excluded devices: | 0 | Language used: | English, United States | Screenshots: | Compare |
| Failed devices: | 0 | Device group used: | Free Intel Android Devices | Logs: | Available ⓘ |
| Running devices: | 0 | Tags: | + | Performance logs: | Available ⓘ |
| Waiting devices: | 0 | | | | |

Device statuses

| Device | Status | Installing application | Launching application | Test execution | Test cases passed |
|--------------------------------|-------------|------------------------|-----------------------|----------------|-------------------|
| Acer Iconia Tab 8 A1-840FH... | succeeded ✓ | 3s ✓ | N/A ✓ | 5m 16s ✓ | 1/1 ✓ |
| Asus Fonepad ME371MG | succeeded ✓ | 10s ✓ | N/A ✓ | 5m 16s ✓ | 1/1 ✓ |
| Dell Venue 7 3730 | succeeded ✓ | 8s ✓ | N/A ✓ | 5m 14s ✓ | 1/1 ✓ |
| Motorola RAZR I XT890 | succeeded ✓ | 14s ✓ | N/A ✓ | 5m 15s ✓ | 1/1 ✓ |
| Samsung Galaxy Tab 3 10.1 i... | succeeded ✓ | 6s ✓ | N/A ✓ | 5m 16s ✓ | 1/1 ✓ |

Test Run 4 in [Pc Assist](#) project

✔ FINISHED

| | | | | | |
|------------------------|---|--------------------|------------------------|-------------------|---------------------|
| Executed devices: | 3 | Project type: | Android | Application: | com.nafis.pcass... |
| Devices with warnings: | 0 | Starting time: | 16.12.2015 17:00:36 | Tests: | App Crawler Compare |
| Excluded devices: | 0 | Language used: | English, United States | Screenshots: | Compare |
| Failed devices: | 0 | Device group used: | --- | Logs: | Available ⓘ |
| Running devices: | 0 | Tags: | + | Performance logs: | Available ⓘ |
| Waiting devices: | 0 | | | | |

Device statuses

| Device | Status | Installing application | Launching application | Test execution | Test cases passed |
|------------------------------|-------------|------------------------|-----------------------|----------------|-------------------|
| NVIDIA Shield Tablet | succeeded ✓ | 3s ✓ | 0s ✓ | 5m 54s ✓ | 1/1 ✓ |
| Samsung Galaxy Nexus GT-I... | succeeded ✓ | 8s ✓ | 1s ✓ | 5m 47s ✓ | 1/1 ✓ |
| Samsung Galaxy Nexus SPH... | succeeded ✓ | 8s ✓ | 1s ✓ | 5m 51s ✓ | 1/1 ✓ |

Test case comparison (1/1) of Test Ru...

| Device | Status | Errors |
|--|--------|--------|
| Acer Iconia Tab 8 A1-840FHD EU | PASSED | N/A |
| Asus Fonepad ME371MG | PASSED | N/A |
| Dell Venue 7 3730 | PASSED | N/A |
| Motorola RAZR i XT890 | PASSED | N/A |
| Samsung Galaxy Tab 3 10.1 GT-P5210 4.4.2 | PASSED | N/A |

Privacy Policy Terms of Service Our Devices Testdroid FAQ Contact Us
Copyright © 2015 Bitbar Technologies Ltd. All rights reserved.

Start Testdroid Cloud - Tes... BAB IV [Compatibility Mo... test HeitlagerKuijpersVisser-Q... test2 - Paint 2:02

Test case comparison (1/1) of Test Ru...

| Device | Status | Errors |
|-------------------------------------|--------|--------|
| NVIDIA Shield Tablet | PASSED | N/A |
| Samsung Galaxy Nexus GT-I9250 4.2.2 | PASSED | N/A |
| Samsung Galaxy Nexus SPH-L700 4.3 | PASSED | N/A |

Privacy Policy Terms of Service Our Devices Testdroid FAQ Contact Us
Copyright © 2015 Bitbar Technologies Ltd. All rights reserved.

Start Testdroid Cloud - Tes... BAB IV [Compatibility Mo... test HeitlagerKuijpersVisser-Q... test3 - Paint 2:03

LAMPIRAN

PENGUJIAN *USABILITY*

REKAPITULASI NILAI HASIL PENGUJIAN USABILITY

| NAMA | NILAI | | | | | | | | | | | | | | | | | | | Total |
|------------------------------------|-------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-------|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | |
| Agallio Samai Suhardi | 4 | 4 | 4 | 3 | 4 | 4 | 4 | 4 | 4 | 3 | 4 | 4 | 4 | 4 | 4 | 2 | 2 | 2 | 4 | 68 |
| Agnes Thioida Sigalingging | 3 | 4 | 4 | 4 | 4 | 5 | 4 | 3 | 4 | 3 | 4 | 4 | 4 | 4 | 4 | 3 | 3 | 4 | 4 | 72 |
| Ahmad Khotibul Umam | 4 | 4 | 4 | 3 | 4 | 4 | 4 | 4 | 4 | 3 | 5 | 4 | 5 | 4 | 4 | 4 | 3 | 4 | 4 | 75 |
| Aji Kurniawan | 5 | 5 | 4 | 4 | 5 | 5 | 5 | 3 | 3 | 5 | 4 | 4 | 4 | 4 | 5 | 5 | 5 | 4 | 4 | 83 |
| Anisa Rahmawati | 3 | 4 | 5 | 4 | 4 | 4 | 4 | 3 | 4 | 3 | 3 | 4 | 4 | 4 | 3 | 4 | 4 | 3 | 3 | 70 |
| Azka Fisyah Faza Althaf | 5 | 5 | 4 | 4 | 4 | 5 | 5 | 4 | 4 | 3 | 3 | 4 | 4 | 4 | 4 | 4 | 3 | 3 | 4 | 76 |
| Bayu Khrisnamurti | 5 | 5 | 4 | 5 | 5 | 4 | 5 | 4 | 5 | 4 | 5 | 5 | 4 | 5 | 5 | 3 | 3 | 4 | 4 | 85 |
| Bayu Setiawan | 2 | 4 | 3 | 2 | 4 | 3 | 4 | 3 | 4 | 3 | 4 | 4 | 4 | 3 | 4 | 4 | 3 | 3 | 3 | 64 |
| Bernadetha Mega Devina Ayuningtyas | 4 | 4 | 3 | 3 | 3 | 3 | 4 | 3 | 4 | 3 | 4 | 4 | 4 | 3 | 2 | 3 | 2 | 3 | 3 | 62 |
| Bintang Buana Rajasa | 5 | 5 | 3 | 3 | 3 | 4 | 5 | 3 | 4 | 3 | 5 | 3 | 5 | 4 | 5 | 5 | 3 | 2 | 3 | 73 |
| Cecep Wahyu Cahyana | 5 | 5 | 4 | 4 | 4 | 5 | 5 | 3 | 5 | 5 | 4 | 4 | 4 | 4 | 5 | 5 | 5 | 3 | 4 | 83 |
| Chandra Adam Pratama | 4 | 5 | 4 | 3 | 4 | 3 | 5 | 4 | 3 | 3 | 4 | 4 | 4 | 4 | 5 | 3 | 2 | 5 | 4 | 73 |
| Chatarina Ria Septiana | 3 | 4 | 3 | 3 | 4 | 3 | 4 | 3 | 4 | 4 | 4 | 4 | 4 | 3 | 3 | 4 | 3 | 3 | 3 | 66 |
| Clarita Dwiyanti | 4 | 5 | 4 | 4 | 4 | 5 | 5 | 4 | 5 | 3 | 4 | 3 | 5 | 5 | 5 | 4 | 3 | 3 | 4 | 79 |
| Dana Irfani | 4 | 5 | 4 | 4 | 4 | 4 | 5 | 3 | 5 | 3 | 4 | 4 | 4 | 4 | 4 | 3 | 2 | 4 | 4 | 74 |
| Danar Asmoro Jati | 4 | 4 | 3 | 4 | 5 | 5 | 5 | 4 | 4 | 3 | 4 | 4 | 4 | 5 | 4 | 4 | 3 | 3 | 4 | 77 |
| Danis Nurmansyah | 5 | 5 | 4 | 3 | 4 | 4 | 5 | 4 | 4 | 3 | 4 | 3 | 4 | 3 | 3 | 5 | 5 | 4 | 4 | 76 |
| Dava Mohammad Hamka | 4 | 5 | 3 | 4 | 4 | 3 | 5 | 3 | 4 | 4 | 4 | 4 | 5 | 4 | 4 | 3 | 3 | 4 | 4 | 74 |
| Denny Ramadhan | 3 | 5 | 4 | 3 | 3 | 3 | 4 | 3 | 4 | 4 | 4 | 3 | 4 | 3 | 2 | 3 | 2 | 3 | 4 | 64 |
| Deva Nurmeilita Kusumarani | 3 | 5 | 3 | 3 | 3 | 3 | 5 | 3 | 5 | 5 | 4 | 3 | 5 | 3 | 5 | 5 | 5 | 3 | 3 | 74 |
| Diah Febriyanti Nengrum | 3 | 4 | 3 | 3 | 3 | 2 | 5 | 3 | 4 | 5 | 3 | 3 | 5 | 4 | 5 | 5 | 4 | 2 | 4 | 70 |
| Dwi Sugiyanto | 5 | 5 | 4 | 3 | 4 | 3 | 5 | 4 | 5 | 3 | 3 | 4 | 5 | 4 | 5 | 5 | 4 | 3 | 4 | 78 |
| Enggal Aldiansyah | 2 | 4 | 2 | 2 | 3 | 3 | 3 | 2 | 3 | 3 | 2 | 2 | 3 | 3 | 4 | 4 | 4 | 3 | 3 | 55 |
| Erna Yulinana | 3 | 4 | 3 | 3 | 3 | 3 | 4 | 3 | 4 | 3 | 3 | 3 | 4 | 4 | 3 | 4 | 4 | 3 | 3 | 64 |
| Ervin Setiadi | 4 | 4 | 3 | 3 | 4 | 4 | 4 | 3 | 4 | 4 | 3 | 3 | 4 | 4 | 4 | 4 | 2 | 3 | 3 | 67 |
| Fadhilah Azhar | 5 | 5 | 4 | 4 | 4 | 3 | 5 | 4 | 3 | 4 | 4 | 3 | 3 | 4 | 4 | 5 | 5 | 3 | 4 | 76 |
| Fahrul Ahmad Fauzi | 4 | 5 | 4 | 4 | 5 | 4 | 5 | 4 | 4 | 4 | 4 | 3 | 5 | 4 | 4 | 4 | 3 | 3 | 5 | 78 |
| Faisal Nur Rais | 4 | 5 | 4 | 4 | 4 | 4 | 5 | 5 | 4 | 4 | 5 | 4 | 4 | 4 | 3 | 4 | 3 | 3 | 4 | 77 |
| Galuh Mumpuni Yuniari | 4 | 4 | 3 | 3 | 3 | 3 | 4 | 4 | 4 | 3 | 3 | 3 | 4 | 3 | 4 | 3 | 3 | 3 | 4 | 65 |
| Habib Muhammad Rifqi Rais | 4 | 5 | 3 | 4 | 4 | 3 | 4 | 3 | 4 | 3 | 4 | 4 | 4 | 3 | 3 | 4 | 3 | 4 | 4 | 70 |
| Handy Susanto | 5 | 5 | 4 | 4 | 4 | 5 | 5 | 4 | 5 | 4 | 4 | 4 | 4 | 4 | 5 | 3 | 4 | 4 | 4 | 81 |
| Hangga Nugraha Julian | 4 | 5 | 4 | 3 | 4 | 5 | 5 | 4 | 5 | 4 | 5 | 5 | 4 | 4 | 5 | 4 | 3 | 4 | 4 | 81 |
| Jumlah | 126 | 147 | 115 | 110 | 124 | 121 | 146 | 111 | 132 | 114 | 124 | 117 | 136 | 121 | 128 | 129 | 105 | 104 | 120 | 2330 |

Jumlah skor total yang dihasilkan = 2330

Jumlah skor maksimal yang diharapkan = $5 \times 19 \times 32 = 3040$

LAMPIRAN

PENGUJIAN *FUNCTIONALITY*

UJI FUNGSIONALITAS

SISTEM PAKAR UNTUK PERBAIKAN KOMPUTER BERBASIS ANDROID

Nama : M. Rifqi Atsani, S.pd., M.Kom.

Bidang Keahlian : Programmer

Berilah tanda centang (✓) pada kolom hasil sesuai dengan pendapat Bapak/Ibu.

| No | Aktivitas | Prosedur yang dilakukan | Hasil yang diharapkan | Hasil | |
|----|---|---|---|--------|-------|
| | | | | Sukses | Gagal |
| 1 | Akses halaman utama | Pengguna memilih aplikasi PCassist pada daftar aplikasi yang ada di smartphone. | Aplikasi PCassist dapat terakses dan menunjukkan halaman utama. | ✓ | |
| 2 | Akses menu diagnosa | Pengguna memilih menu Mulai Diagnosa pada halaman utama. | Menu Mulai Diagnosa dapat terakses dan memunculkan Daftar diagnosa yang tersedia. | ✓ | |
| 3 | Akses halaman konsultasi | Pengguna memilih proses diagnosa yang diinginkan pada Daftar diagnosa yang tersedia. | Halaman konsultasi dapat terakses dan menampilkan form konsultasi. | ✓ | |
| 4 | Proses konsultasi, menu opsi Ya dan Tidak | Pengguna memilih opsi Ya atau Tidak untuk merespon pertanyaan yang ada. | Dapat menampilkan pertanyaan selanjutnya atau menampilkan hasil diagnosa pada halaman konsultasi. | ✓ | |
| 5 | Proses konsultasi, menu Penjelasan | Pengguna memilih menu Penjelasan yang tersedia dihalaman konsultasi | Dapat menampilkan info penjelasan dari pertanyaan atau | ✓ | |

| No | Aktivitas | Prosedur yang dilakukan | Hasil yang diharapkan | Hasil | |
|----|--|--|---|--------|-------|
| | | | | Sukses | Gagal |
| | | | hasil pengecekan yang ada. | | |
| 6 | Proses konsultasi, menu Kembali | Pengguna memilih menu Kembali yang ada dihalaman konsultasi | Dapat menampilkan kembali pertanyaan diagnosa sebelumnya dihalaman konsultasi. | ✓ | |
| 7 | Akses menu Bantuan | Pengguna memilih menu Bantuan pada halaman utama | Halaman bantuan dapat terakses dan menampilkan informasi untuk membantu mengoperasikan aplikasi PCassist. | ✓ | |
| 8 | Akses menu Tentang PCassist | Pengguna memilih menu Tentang PCassist dihalaman utama. | Halaman tentang PCassist dapat terakses dan menampilkan info tujuan aplikasi PCassist serta pembuatnya. | ✓ | |
| 9 | Akses menu Pengaturan | Pengguna memilih menu Pengaturan yang berupa icon roda gigi dipojok kanan atas dihalaman utama | Halaman pengaturan dapat terakses yang menampilkan opsi pengaturan yang ada. | ✓ | |
| 10 | Akses menu Basis Pengetahuan | Pengguna memilih menu Basis Pengetahuan pada halaman diagnosa | Menampilkan opsi yang ada untuk pengaturan basis pengetahuan. | ✓ | |
| 11 | Tambah data pada basis pengetahuan | Pengguna memilih menu Tambah data , kemudian memilih pada proses diagnosa mana data akan ditambah. Selanjutnya pengguna mengisi form yang telah disediakan dan memilih tombol Tambah . | Data yang dimasukkan dapat disimpan dibasis pengetahuan yang ada sesuai proses diagnosa yang dipilih. | ✓ | |
| 12 | Lihat data pada basis pengetahuan | Pengguna memilih menu Lihat data , kemudian milih proses diagnosa mana yang akan dilihat datanya. Kemudian mengklik daftar data yang tampil untuk melihat | Dapat menampilkan data yang telah dimasukkan sebelumnya, serta menampilkan detail datanya. | ✓ | |

| No | Aktivitas | Prosedur yang dilakukan | Hasil yang diharapkan | Hasil | |
|----|-----------------------------------|---|---|--------|-------|
| | | | | Sukses | Gagal |
| | | detailnya. | | | |
| 13 | Edit data pada basis pengetahuan | Pengguna memilih menu Edit data , kemudian memilih diproses diagnosa mana data berada. Lalu mengklik data yang akan diedit pada daftar data yang ada. Selanjutnya mengisi form yang ada dan memilih tombol Edit . | Dapat menyimpan data yang telah diedit pada form pengeditan ke basis data. | ✓ | |
| 14 | Hapus data pada basis pengetahuan | Pengguna memilih menu Hapus Data , kemudian memilih diproses diagnosa mana data berada. Lalu memilih tombol Hapus . | Dapat menghapus data proses diagnosa pada basis pengetahuan. | ✓ | |
| 15 | Akses menu Basis aturan | Pengguna memilih menu Basis aturan pada halaman pengetahuan. | Dapat menampilkan opsi pengaturan untuk basis aturan. | ✓ | |
| 16 | Buat aturan pada basis aturan | Pengguna memilih menu Buat aturan , kemudian memilih pada proses diagnosa mana aturan akan disusun. Selanjutnya menyusun data yang ada pada form yang telah disediakan dan memilih tombol Buat Aturan . | Dapat menyusun rangkaian aturan dari data yang sebelumnya telah dimasukkan. | ✓ | |
| 17 | Lihat aturan pada basis aturan | Pengguna memilih menu Lihat aturan , kemudian memilih proses diagnosa mana yang akan dilihat aturan yang telah disusun. | Dapat menampilkan daftar node-node aturan yang telah disusun dan menampilkan detailnya. | ✓ | |
| 18 | Edit aturan pada basis aturan | Pengguna memilih menu Edit aturan , kemudian memilih proses diagnosa mana yang akan diedit aturannya. Selanjutnya memilih node aturan yang ada dan | Dapat mengedit node-node aturan yang telah dibuat sebelumnya. | ✓ | |

| No | Aktivitas | Prosedur yang dilakukan | Hasil yang diharapkan | Hasil | |
|----|-------------------------|--|--|--------|-------|
| | | | | Sukses | Gagal |
| | | melakukan pengeditan pada form yang telah disediakan. Setelah itu memilih tombol Edit . | | | |
| 19 | Navigasi menu up | Pengguna memilih menu up yang berupa icon anak panah dan icon aplikasi dipojok kiri atas. | Dapat kembali ke menu atau halaman sebelumnya. | ✓ | |
| 20 | Proses konsultasi | Pengguna melakukan konsultasi pada halaman konsultasi dengan bantuan menu opsi Ya, Tidak, Kembali , serta Penjelasan. | Proses diagnosa berjalan sesuai dengan aturan yang telah disusun dan draf pohon keputusan yang dibuat. | ✓ | |

Yogyakarta,.....

Validator,



M. Rifahi Atsan, S.pd., M.Kom

UJI FUNGSIONALITAS

SISTEM PAKAR UNTUK PERBAIKAN KOMPUTER BERBASIS ANDROID

Nama : Aandiken Tim Arawan

Bidang Keahlian : Mobile Developer

Berilah tanda centang (✓) pada kolom hasil sesuai dengan pendapat Bapak/Ibu.

| No | Aktivitas | Prosedur yang dilakukan | Hasil yang diharapkan | Hasil | |
|----|---|---|---|--------|-------|
| | | | | Sukses | Gagal |
| 1 | Akses halaman utama | Pengguna memilih aplikasi PCassist pada daftar aplikasi yang ada di smartphone. | Aplikasi PCassist dapat terakses dan menunjukkan halaman utama. | ✓ | |
| 2 | Akses menu diagnosa | Pengguna memilih menu Mulai Diagnosa pada halaman utama. | Menu Mulai Diagnosa dapat terakses dan memunculkan Daftar diagnosa yang tersedia. | ✓ | |
| 3 | Akses halaman konsultasi | Pengguna memilih proses diagnosa yang diinginkan pada Daftar diagnosa yang tersedia. | Halaman konsultasi dapat terakses dan menampilkan form konsultasi. | ✓ | |
| 4 | Proses konsultasi, menu opsi Ya dan Tidak | Pengguna memilih opsi Ya atau Tidak untuk merespon pertanyaan yang ada. | Dapat menampilkan pertanyaan selanjutnya atau menampilkan hasil diagnosa pada halaman konsultasi. | ✓ | |
| 5 | Proses konsultasi, menu Penjelasan | Pengguna memilih menu Penjelasan yang tersedia dihalaman konsultasi | Dapat menampilkan info penjelasan dari pertanyaan atau | ✓ | |

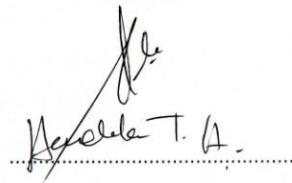
| No | Aktivitas | Prosedur yang dilakukan | Hasil yang diharapkan | Hasil | |
|----|--|--|---|--------|-------|
| | | | | Sukses | Gagal |
| | | | hasil pengecekan yang ada. | | |
| 6 | Proses konsultasi, menu Kembali | Pengguna memilih menu Kembali yang ada dihalaman konsultasi | Dapat menampilkan kembali pertanyaan diagnosa sebelumnya dihalaman konsultasi. | ✓ | |
| 7 | Akses menu Bantuan | Pengguna memilih menu Bantuan pada halaman utama | Halaman bantuan dapat terakses dan menampilkan informasi untuk membantu mengoperasikan aplikasi PCassist. | ✓ | |
| 8 | Akses menu Tentang PCassist | Pengguna memilih menu Tentang PCassist dihalaman utama. | Halaman tentang PCassist dapat terakses dan menampilkan info tujuan aplikasi PCassist serta pembuatnya. | ✓ | |
| 9 | Akses menu Pengaturan | Pengguna memilih menu Pengaturan yang berupa icon roda gigi di pojok kanan atas dihalaman utama | Halaman pengaturan dapat terakses yang menampilkan opsi pengaturan yang ada. | ✓ | |
| 10 | Akses menu Basis Pengetahuan | Pengguna memilih menu Basis Pengetahuan pada halaman diagnosa | Menampilkan opsi yang ada untuk pengaturan basis pengetahuan. | ✓ | |
| 11 | Tambah data pada basis pengetahuan | Pengguna memilih menu Tambah data , kemudian memilih pada proses diagnosa mana data akan ditambah. Selanjutnya pengguna mengisi form yang telah disediakan dan memilih tombol Tambah . | Data yang dimasukkan dapat disimpan dibasis pengetahuan yang ada sesuai proses diagnosa yang dipilih. | ✓ | |
| 12 | Lihat data pada basis pengetahuan | Pengguna memilih menu Lihat data , kemudianilih proses diagnosa mana yang akan dilihat datanya. Kemudian mengklik daftar data yang tampil untuk melihat | Dapat menampilkan data yang telah dimasukkan sebelumnya, serta menampilkan detail datanya. | ✓ | |

| No | Aktivitas | Prosedur yang dilakukan | Hasil yang diharapkan | Hasil | |
|----|-----------------------------------|---|---|--------|-------|
| | | | | Sukses | Gagal |
| | | detailnya. | | | |
| 13 | Edit data pada basis pengetahuan | Pengguna memilih menu Edit data , kemudian memilih diproses diagnosa mana data berada. Lalu mengklik data yang akan diedit pada daftar data yang ada. Selanjutnya mengisi form yang ada dan memilih tombol Edit . | Dapat menyimpan data yang telah diedit pada form pengeditan ke basis data. | ✓ | |
| 14 | Hapus data pada basis pengetahuan | Pengguna memilih menu Hapus Data , kemudian memilih diproses diagnosa mana data berada. Lalu memilih tombol Hapus . | Dapat menghapus data proses diagnosa pada basis pengetahuan. | ✓ | |
| 15 | Akses menu Basis aturan | Pengguna memilih menu Basis aturan pada halaman pengetahuan. | Dapat menampilkan opsi pengaturan untuk basis aturan. | ✓ | |
| 16 | Buat aturan pada basis aturan | Pengguna memilih menu Buat aturan , kemudian memilih pada proses diagnosa mana aturan akan disusun. Selanjutnya menyusun data yang ada pada form yang telah disediakan dan memilih tombol Buat Aturan . | Dapat menyusun rangkaian aturan dari data yang sebelumnya telah dimasukkan. | ✓ | |
| 17 | Lihat aturan pada basis aturan | Pengguna memilih menu Lihat aturan , kemudian memilih proses diagnosa mana yang akan dilihat aturan yang telah disusun. | Dapat menampilkan daftar node-node aturan yang telah disusun dan menampilkan detailnya. | ✓ | |
| 18 | Edit aturan pada basis aturan | Pengguna memilih menu Edit aturan , kemudian memilih proses diagnosa mana yang akan diedit aturnya. Selanjutnya memilih node aturan yang ada dan | Dapat mengedit node-node aturan yang telah dibuat sebelumnya. | ✓ | |

| No | Aktivitas | Prosedur yang dilakukan | Hasil yang diharapkan | Hasil | |
|----|-------------------------|---|---|-------------------------------------|-------|
| | | | | Sukses | Gagal |
| | | mengakses halaman konsultasi dan melakukan pengeditan pada form yang telah disediakan. Setelah itu memilih tombol Edit . | | | |
| 19 | Navigasi menu up | Pengguna memilih menu up yang berupa icon anak panah dan icon aplikasi di pojok kiri atas. | Dapat kembali ke menu atau halaman sebelumnya. | <input checked="" type="checkbox"/> | |
| 20 | Proses konsultasi | Pengguna melakukan konsultasi pada halaman konsultasi dengan bantuan menu opsi Ya, Tidak, Kembali , serta Penjelasan. | Proses diagnosa berjalan sesuai dengan aturan yang telah disusun dan draft pohon keputusan yang dibuat. | <input checked="" type="checkbox"/> | |

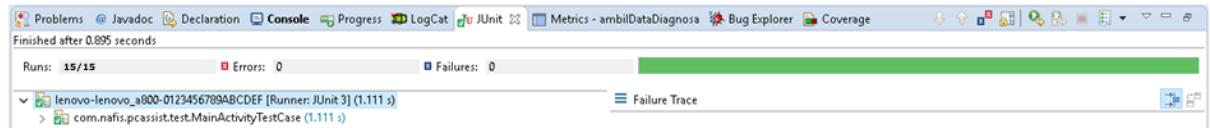
Yogyakarta,.....

Validator,



LAMPIRAN UNIT TEST

HASIL UNIT TEST



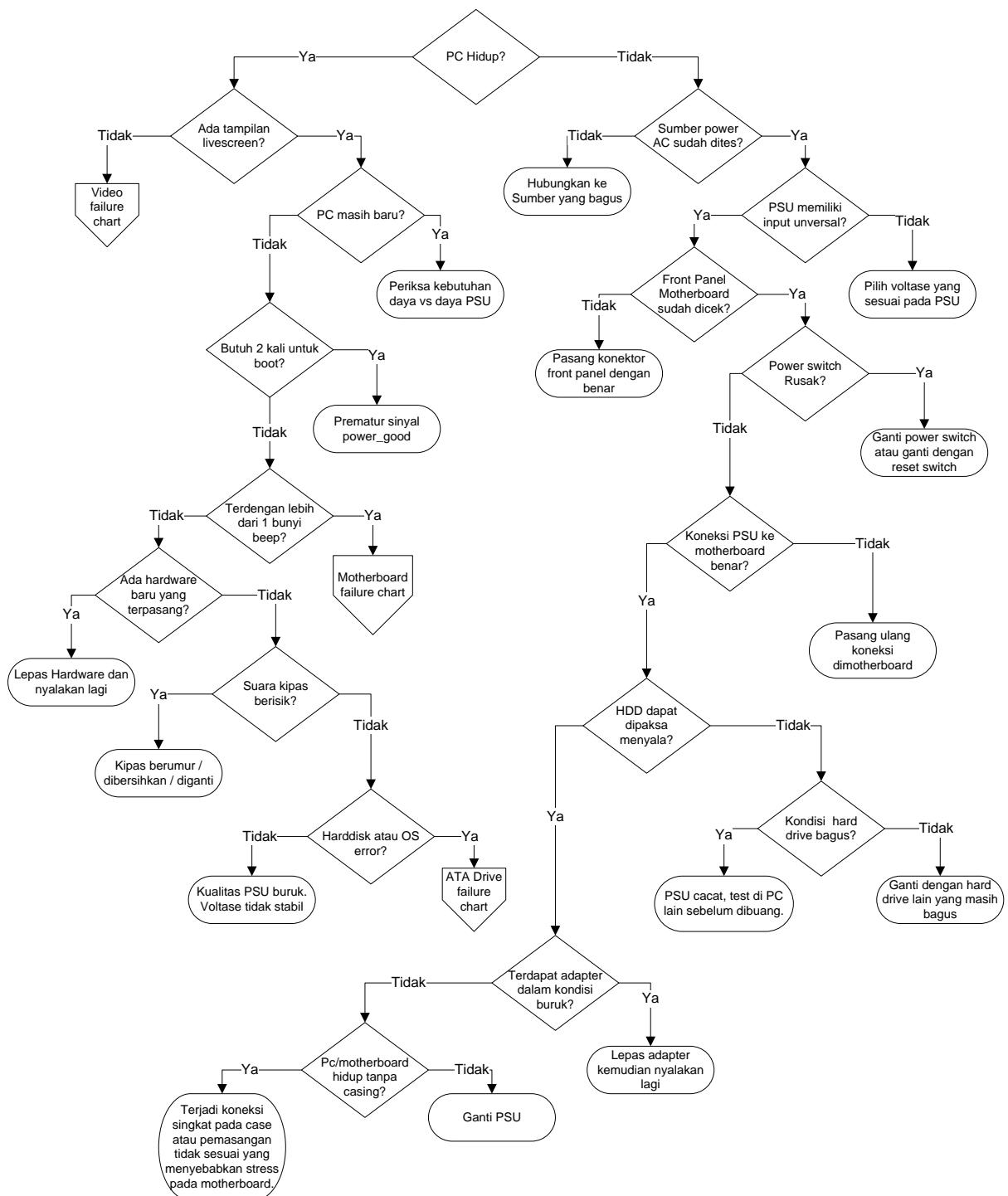
Keterangan :

- Tes berhasil = 15
- Tes error = 0
- Test gagal = 0

$$\begin{aligned}\text{Persentase tes berhasil} &= \frac{\text{jumlah test berhasil}}{\text{total test}} \times 100\% \\ &= \frac{15}{15} \times 100\% \\ &= 100\%\end{aligned}$$

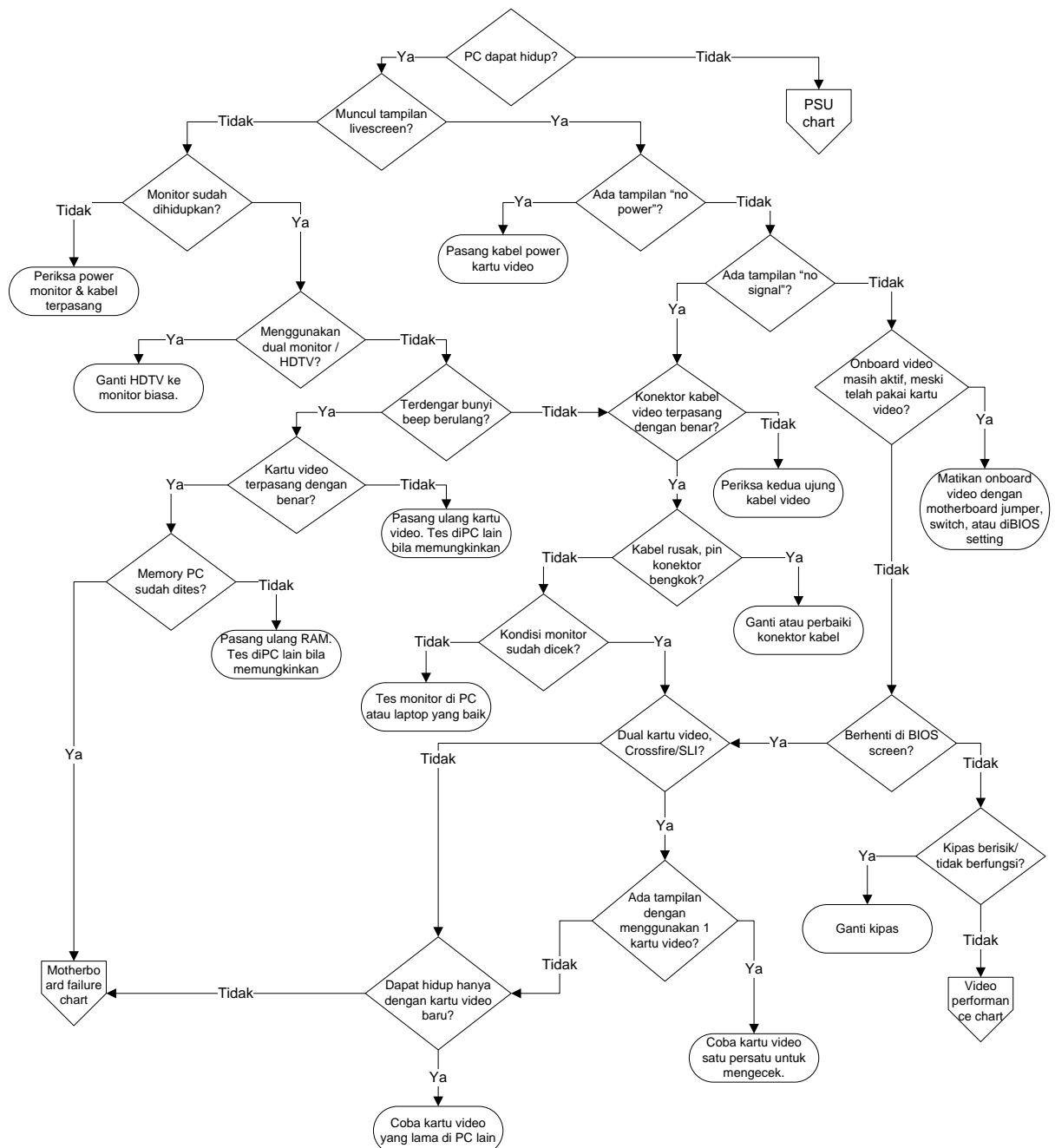
**LAMPIRAN
DAFTAR POHON KEPUTURAN ATAU *RULE
BASE***

RULE BASE DIAGNOSA KERUSAKAN PADA PSU SUMBER, ROSENTHAL (2013:5)

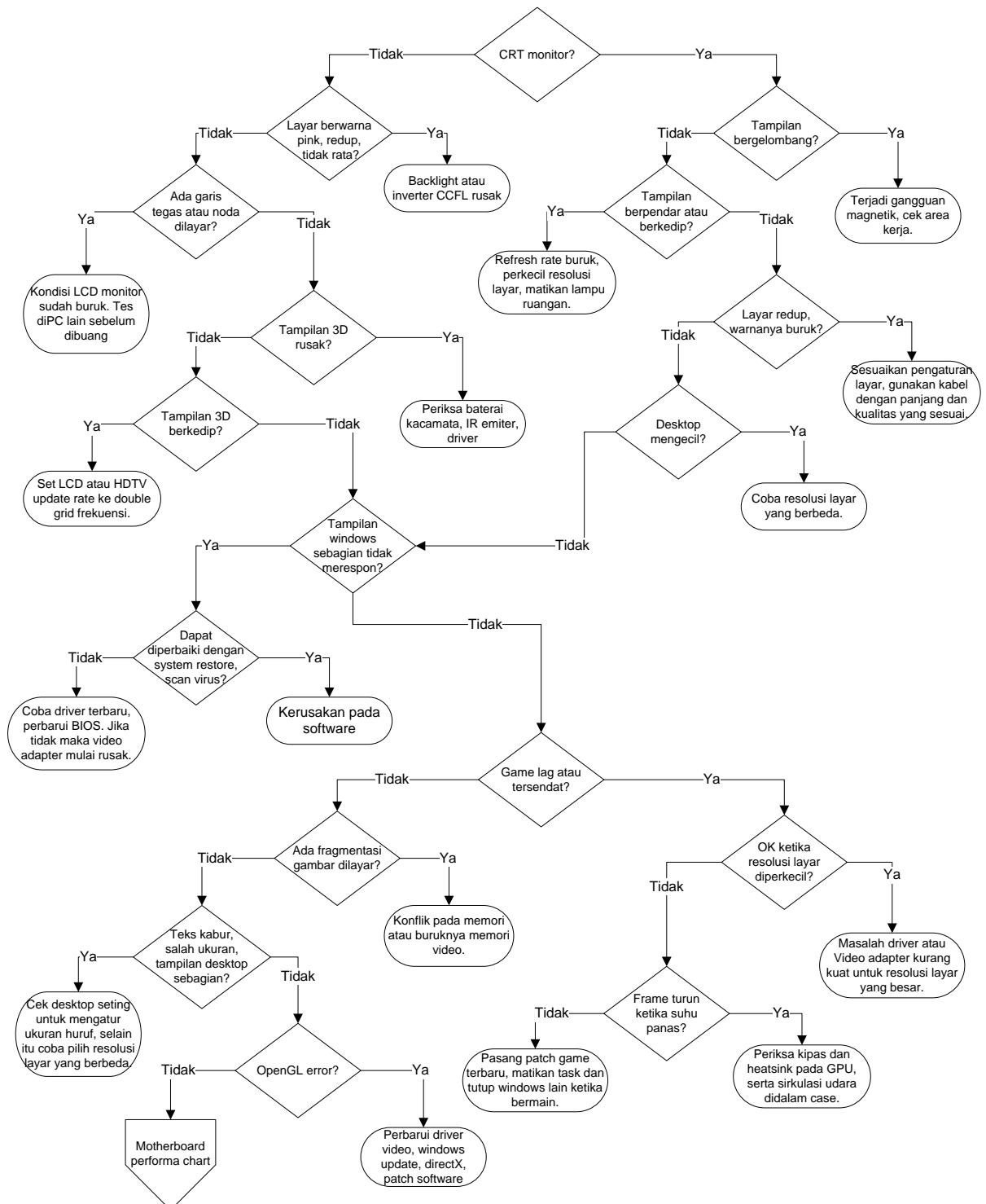


RULE BASE DIAGNOSA KERUSAKAN PADA VIDEO ADAPTER

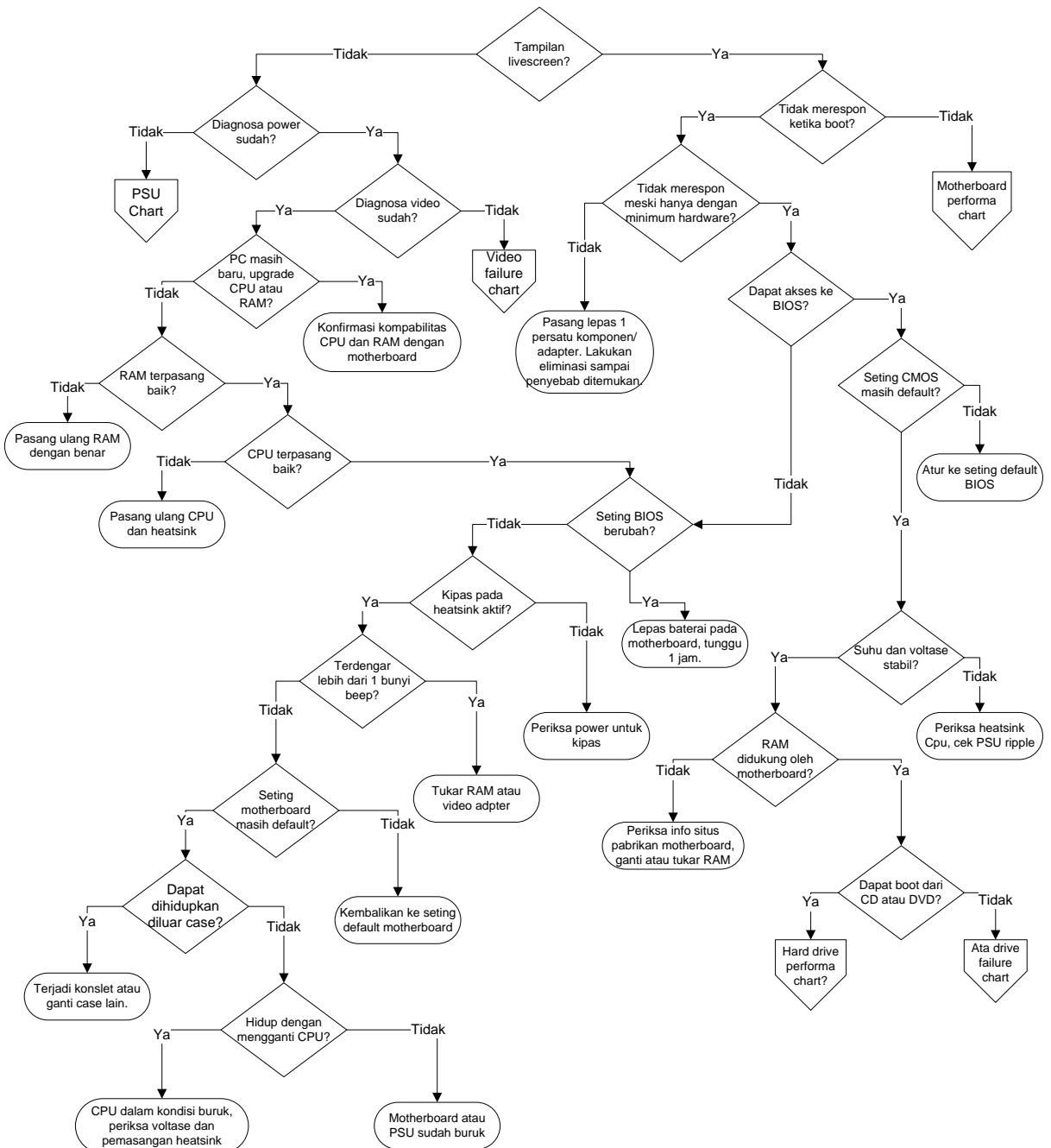
SUMBER, ROSENTHAL (2013:19)



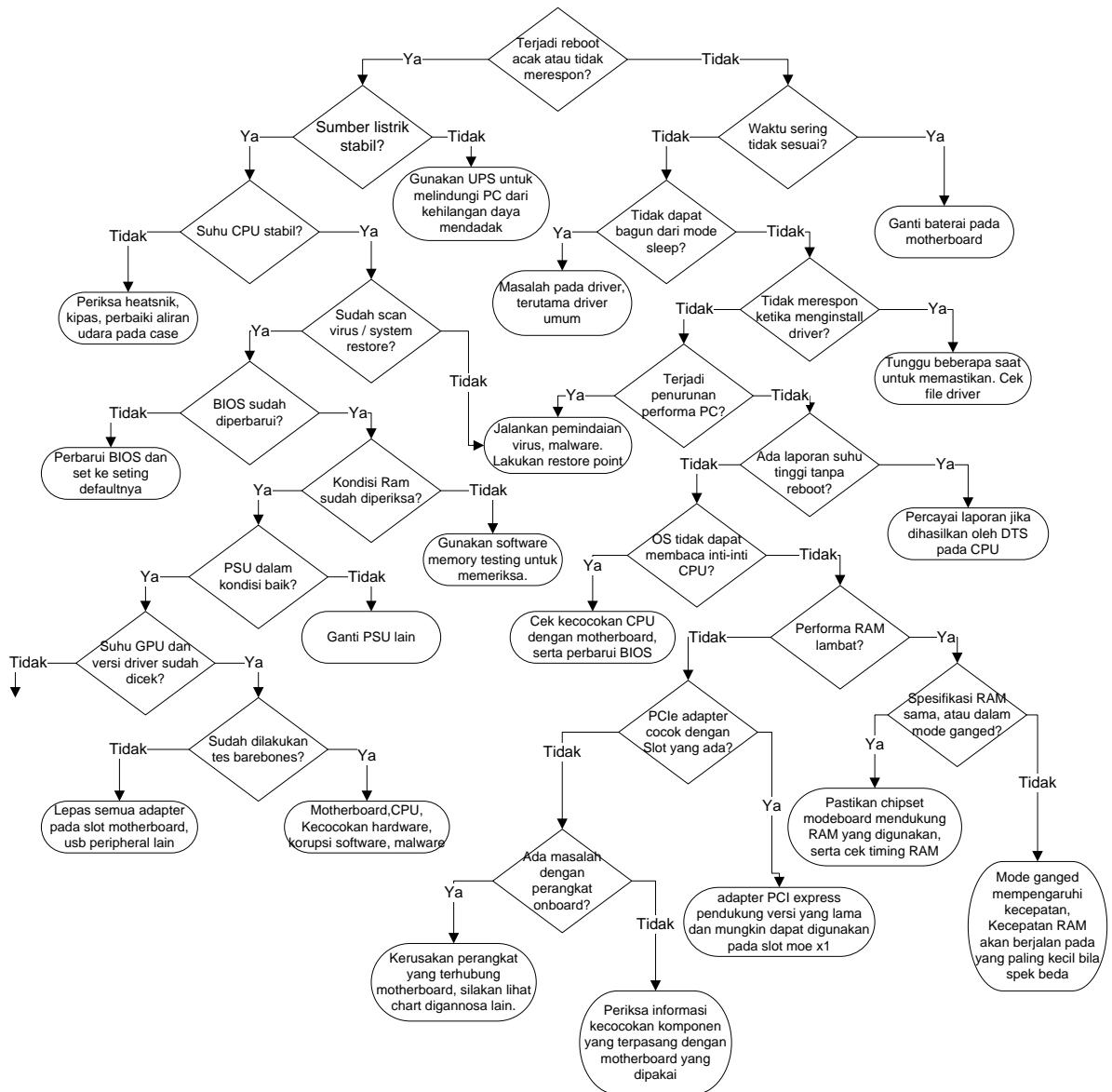
RULE BASE DIAGNOSA PERFORMA PADA MONITOR DAN VIDEO ADAPTER
SUMBER, ROSENTHAL (2013:29)



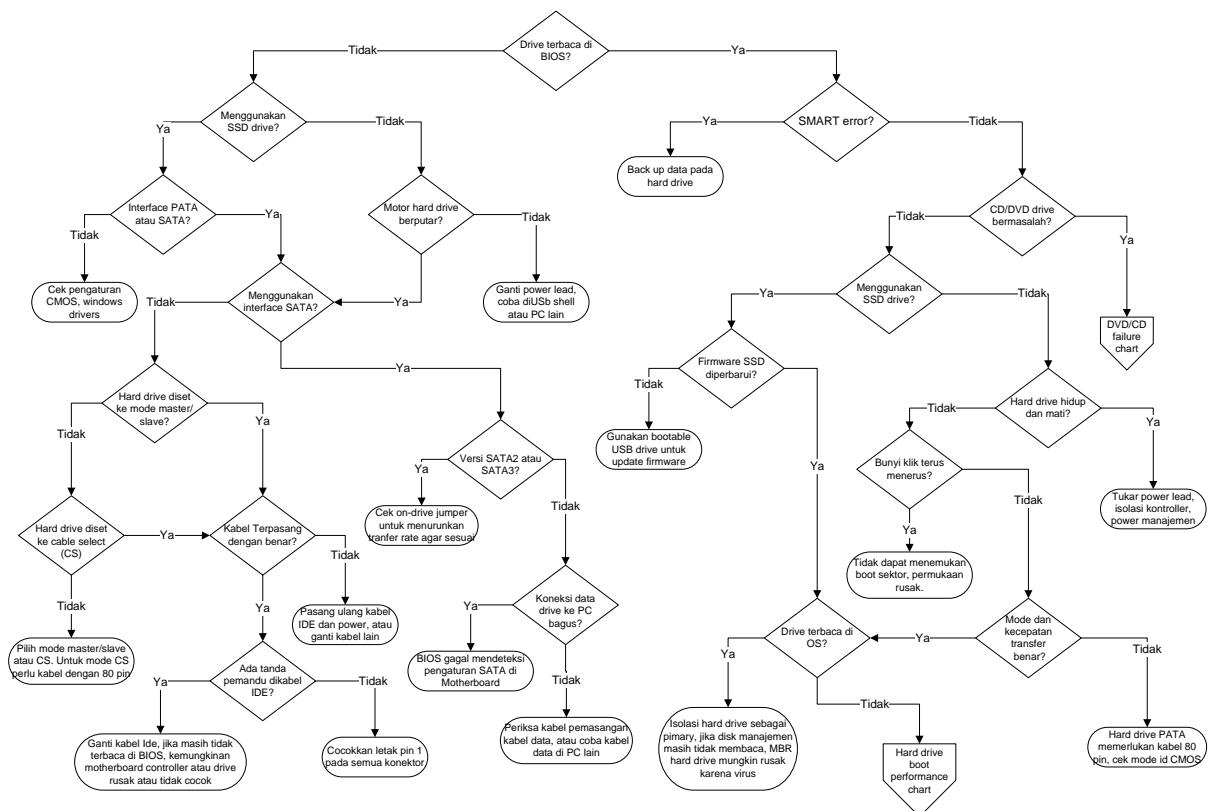
**RULE BASE DIAGNOSA KERUSAKAN PADA MOTHERBOARD, CPU,
DAN RAM**
SUMBER, ROSENTHAL (2013:41)



**RULE BASE DIAGNOSA PERFORMA PADA MOTHERBOARD, CPU,
DAN RAM**
SUMBER, ROSENTHAL (2013:53)

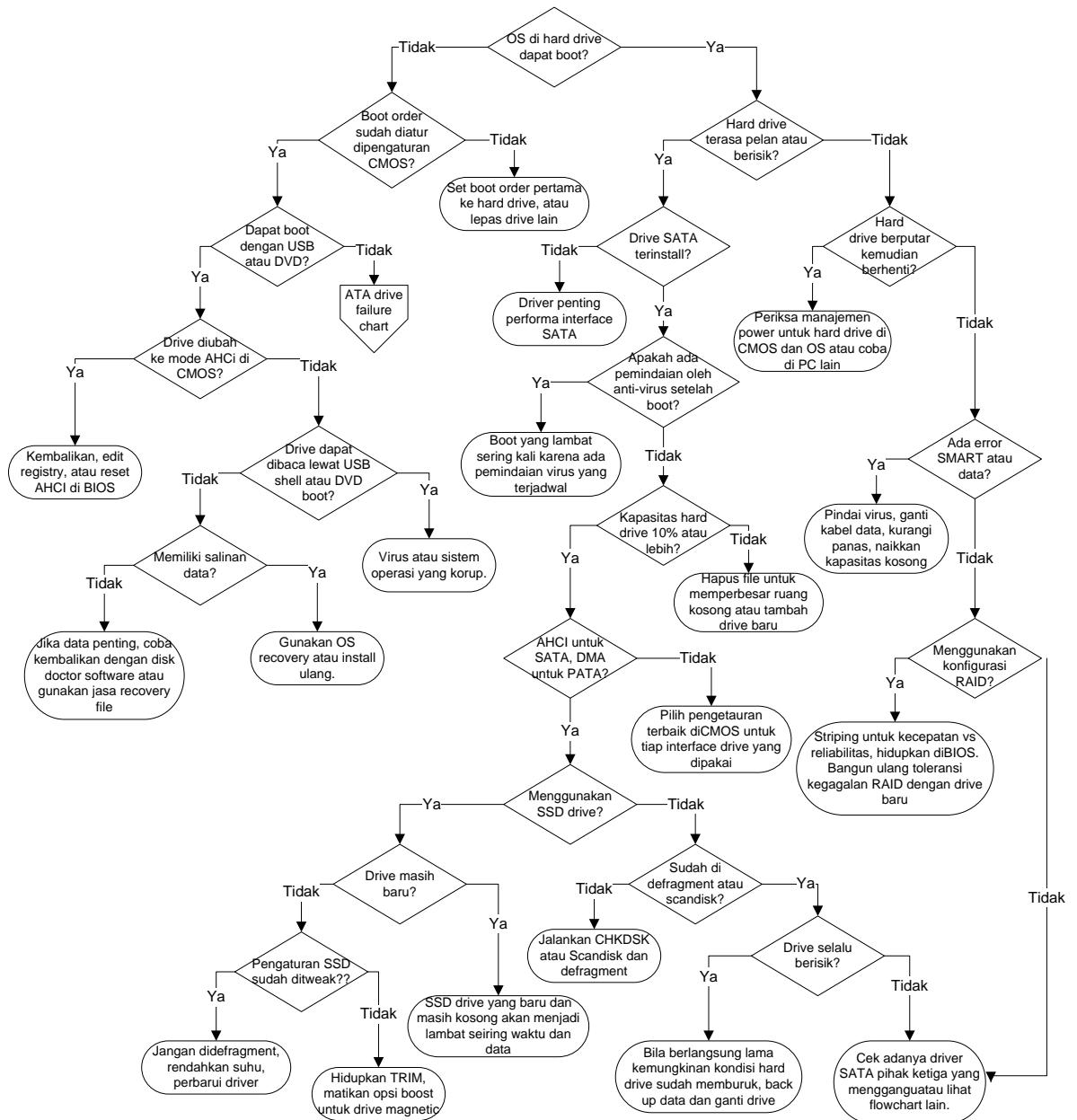


RULE BASE DIAGNOSA KERUSAKAN PADA ATA DRIVE SUMBER, ROSENTHAL (2013:65)



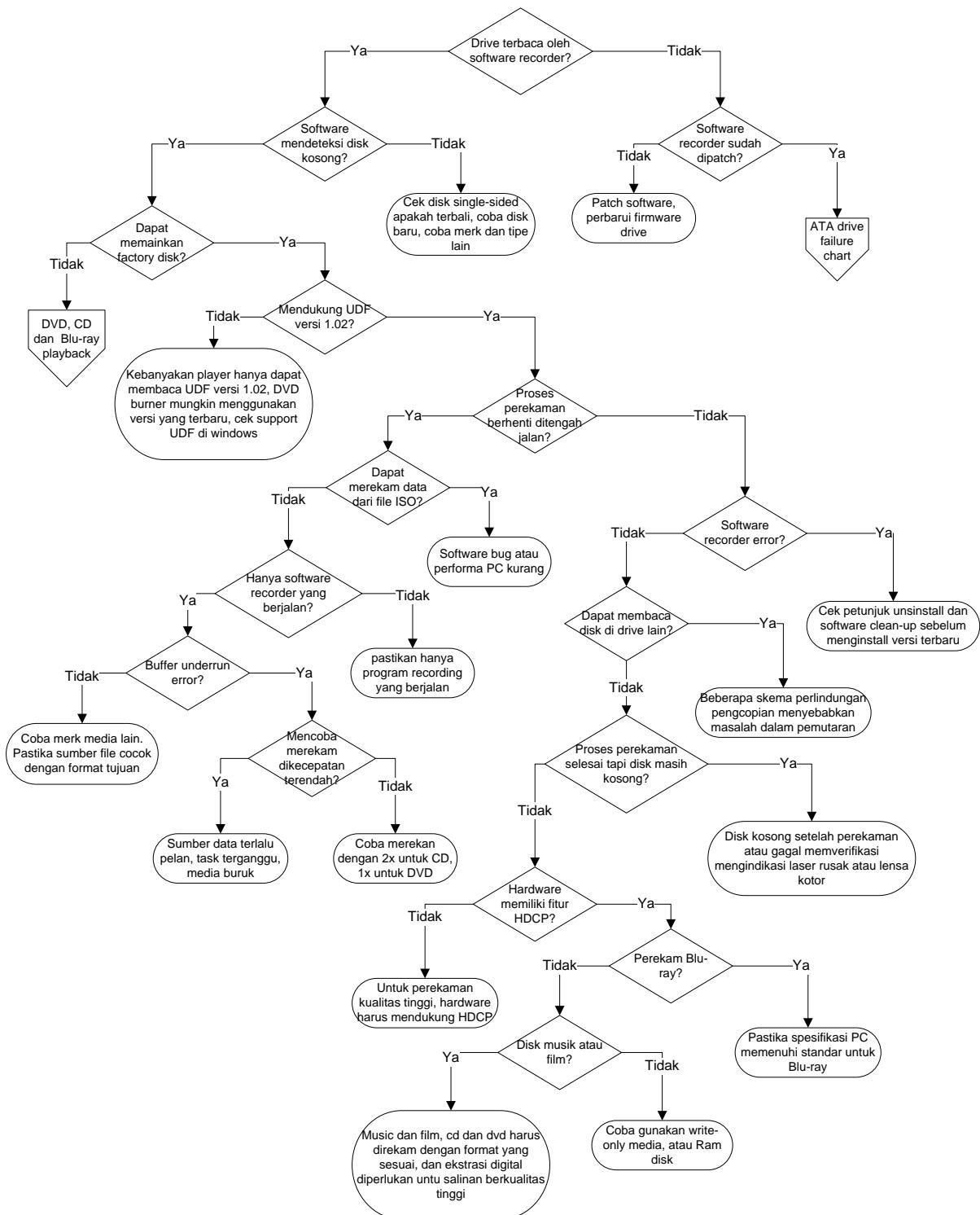
RULE BASE DIAGONSA PADA PERFORMA HARD DRIVE BOOT

SUMBER, ROSENTHAL (2013:75)



RULE BASE DIAGNOSA PADA PEREKAMAN DVD, CD, BLU-RAY

SUMBER, ROSENTHAL (2013:87)



RULE BASE DIAGNOSA PADA PEMUTARAN DVD, CD, DAN BLU-RAY

SUMBER, ROSENTHAL (2013:95)



LAMPIRAN SURAT-SURAT



**PEMERINTAH DAERAH DAERAH ISTIMEWA YOGYAKARTA
SEKRETARIAT DAERAH**

Kompleks Kepatihan, Danurejan, Telepon (0274) 562811 - 562814 (Hunting)
YOGYAKARTA 55213

SURAT KETERANGAN / IJIN

070/REG/V/356/8/2015

Membaca Surat : **WAKIL DEKAN I FAKULTAS TEKNIK** Nomor : **1626/H34/PL/2015**
Tanggal : **22 JUNI 2015** Perihal : **IJIN PENELITIAN/RISET**

Mengingat : 1. Peraturan Pemerintah Nomor 41 Tahun 2006, tentang Perizinan bagi Perguruan Tinggi Asing, Lembaga Penelitian dan Pengembangan Asing, Badan Usaha Asing dan Orang Asing dalam melakukan Kegiatan Penelitian dan Pengembangan di Indonesia;
2. Peraturan Menteri Dalam Negeri Nomor 20 Tahun 2011, tentang Pedoman Penelitian dan Pengembangan di Lingkungan Kementerian Dalam Negeri dan Pemerintah Daerah;
3. Peraturan Gubernur Daerah Istimewa Yogyakarta Nomor 37 Tahun 2008, tentang Rincian Tugas dan Fungsi Satuan Organisasi di Lingkungan Sekretariat Daerah dan Sekretariat Dewan Perwakilan Rakyat Daerah.
4. Peraturan Gubernur Daerah Istimewa Yogyakarta Nomor 18 Tahun 2009 tentang Pedoman Pelayanan Perizinan, Rekomendasi Pelaksanaan Survei, Penelitian, Pendataan, Pengembangan, Pengkajian, dan Studi Lapangan di Daerah Istimewa Yogyakarta.

DIIJINKAN untuk melakukan kegiatan survei/penelitian/pendataan/pengembangan/pengkajian/studi lapangan kepada:

Nama : **NAFIS AKHSAN** NIP/NIM : **09520241027**
Alamat : **FAKULTAS TEKNIK , PENDIDIKAN TEKNIK INFORMATIKA , UNIVERSITAS NEGERI YOGYAKARTA**
.Judul : **PENGEMBANGAN SISTEM PAKAR PERBAIKAN KOMPUTER BERBASIS ANDROID**
Lokasi : **DINAS PENDIDIKAN, PEMUDA DAN OLAHRAGA DIY**
Waktu : **26 AGUSTUS 2015 s/d 26 NOVEMBER 2015**

Dengan Ketentuan

1. Menyerahkan surat keterangan/ijin survei/penelitian/pendataan/pengembangan/pengkajian/studi lapangan *) dari Pemerintah Daerah DIY kepada Bupati/Walikota melalui institusi yang berwenang mengeluarkan ijin dimaksud;
2. Menyerahkan soft copy hasil penelitiannya baik kepada Gubernur Daerah Istimewa Yogyakarta melalui Biro Administrasi Pembangunan Setda DIY dalam compact disk (CD) maupun mengunggah (upload) melalui website adbang.jogjaprov.go.id dan menunjukkan cetakan asli yang sudah disahkan dan dibubuh cap institusi;
3. Ijin ini hanya dipergunakan untuk keperluan ilmiah, dan pemegang ijin wajib mentaati ketentuan yang berlaku di lokasi kegiatan;
4. Ijin penelitian dapat diperpanjang maksimal 2 (dua) kali dengan menunjukkan surat ini kembali sebelum berakhir waktunya setelah mengajukan perpanjangan melalui website adbang.jogjaprov.go.id;
5. Ijin yang diberikan dapat dibatalkan sewaktu-waktu apabila pemegang ijin ini tidak memenuhi ketentuan yang berlaku.

Dikeluarkan di Yogyakarta
Pada tanggal **26 AGUSTUS 2015**
A.n Sekretaris Daerah
Asisten Perekonomian dan Pembangunan
Ub.



Tembusan :

1. GUBERNUR DAERAH ISTIMEWA YOGYAKARTA (SEBAGAI LAPORAN)
2. BUPATI SLEMAN C.Q KA. BAKESBANGLINMAS SLEMAN
3. DINAS PENDIDIKAN, PEMUDA DAN OLAHRAGA DIY
4. WAKIL DEKAN I FAKULTAS TEKNIK , UNIVERSITAS NEGERI YOGYAKARTA
5. YANG BERSANGKUTAN

**KEPUTUSAN DEKAN
FAKULTAS TEKNIK
UNIVERSITAS NEGERI YOGYAKARTA
NOMOR : 223/ELK/Q-I/XII/2015
TENTANG
PENGANGKATAN PEMBIMBING TUGAS AKHIR SKRIPSI
BAGI MAHASISWA FAKULTAS TEKNIK UNIVERSITAS NEGERI YOGYAKARTA**

**DEKAN FAKULTAS TEKNIK
UNIVERSITAS NEGERI YOGYAKARTA**

Menimbang : 1. Bawa sehubungan dengan telah dipenuhi syarat untuk penulisan Tugas Akhir Skripsi bagi mahasiswa Fakultas Teknik Universitas Negeri Yogyakarta, perlu diangkat pembimbing.
2. Bawa untuk keperluan dimaksud perlu ditetapkan dengan Keputusan Dekan.

Mengingat : 1. Undang-undang Nomor 20 tahun 2003.
2. Peraturan Pemerintah RI Nomor 60 tahun 1999.
3. Keputusan Presiden RI: a. Nomor 93 tahun 1999; b. 305/M tahun 1999.
4. Keputusan Menteri Pendidikan dan Kebudayaan RI: Nomor 274/O/1999.
5. Keputusan Mendiknas RI Nomor 003/O/2001.
6. Keputusan Rektor UNY Nomor : 1160/UN34/KP/2011.

M E M U T U S K A N

Menetapkan

Pertama : Mengangkat Pembimbing Tugas Akhir Skripsi bagi mahasiswa Fakultas Teknik Universitas Negeri Yogyakarta sebagai berikut :

Nama Pembimbing : Prof. Herman Dwi Surjono, Ph.D

Bagi mahasiswa :

Nama/No.Mahasiswa : Nafis Ahsan /09520241027

Jurusan/Prodi : Pendidikan Teknik Elektronika / Pendidikan Teknik Informatika

Judul Skripsi : Pengembangan Sistem Pakar Perbaikan Komputer Berbasis Android

Kedua : Dosen pembimbing diberi tugas membimbing penulisan Tugas Akhir Skripsi sesuai dengan Pedoman Tugas Akhir Skripsi.

Ketiga : Keputusan ini berlaku sejak ditetapkan

Keempat : Segala sesuatu akan diubah dan dibetulkan sebagaimana mestinya apabila di kemudian hari ternyata terdapat kekeliruan dalam Keputusan ini.



Tembusan Yth :

1. Wakil Dekan II, FT UNY
2. Ketua Jurusan Pendidikan Teknik Elektronika
3. Kasub. Bag. Pendidikan FT UNY
4. Yang bersangkutan

Tambahan Yth :



PEMERINTAH KABUPATEN SLEMAN
BADAN PERENCANAAN PEMBANGUNAN DAERAH

Jalan Parasamya Nomor 1 Beran, Tridadi, Sleman, Yogyakarta 55511
Telepon (0274) 868800, Faksimilie (0274) 868800
Website: www.bappeda.sleman.go.id, E-mail : bappeda@sleman.go.id

SURAT IZIN

Nomor : 070 / Bappeda / 3172 / 2015

**TENTANG
PENELITIAN**

KEPALA BADAN PERENCANAAN PEMBANGUNAN DAERAH

Dasar : Peraturan Bupati Sleman Nomor : 45 Tahun 2013 Tentang Izin Penelitian, Izin Kuliah Kerja Nyata, Dan Izin Praktik Kerja Lapangan.

Menunjuk : Surat dari Kepala Kantor Kesatuan Bangsa Kab. Sleman

Nomor : 070/Kesbang/3106/2015

Tanggal : 26 Agustus 2015

Hal : Rekomendasi Penelitian

MENGIZINKAN :

| | |
|----------------------------------|---|
| Kepada | : |
| Nama | : NAFIS AKHSAN |
| No.Mhs/NIM/NIP/NIK | : 09520241027 |
| Program/Tingkat | : S1 |
| Instansi/Perguruan Tinggi | : Universitas Negeri Yogyakarta |
| Alamat instansi/Perguruan Tinggi | : Karangmalang Sleman Yogyakarta |
| Alamat Rumah | : Penumbang Kecepit Punggelan Banjarnegara |
| No. Telp / HP | : 085226980728 |
| Untuk | : Mengadakan Penelitian / Pra Survey / Uji Validitas / PKL dengan judul PENGEMBANGAN SISTEM PAKAR PERBAIKAN KOMPUTER BERBASIS ANDROID |
| Lokasi | : SMKN 2 Depok Sleman |
| Waktu | : Selama 3 Bulan mulai tanggal 26 Agustus 2015 s/d 26 Nopember 2015 |

Dengan ketentuan sebagai berikut :

1. Wajib melaporkan diri kepada Pejabat Pemerintah setempat (Camat/ Kepala Desa) atau Kepala Instansi untuk mendapat petunjuk seperlunya.
2. Wajib menjaga tata tertib dan mematuhi ketentuan-ketentuan setempat yang berlaku.
3. Izin tidak disalahgunakan untuk kepentingan-kepentingan di luar yang direkomendasikan.
4. Wajib menyampaikan laporan hasil penelitian berupa 1 (satu) CD format PDF kepada Bupati diserahkan melalui Kepala Badan Perencanaan Pembangunan Daerah.
5. Izin ini dapat dibatalkan sewaktu-waktu apabila tidak dipenuhi ketentuan-ketentuan di atas.

Demikian izin ini dikeluarkan untuk digunakan sebagaimana mestinya, diharapkan pejabat pemerintah/non pemerintah setempat memberikan bantuan seperlunya.

Setelah selesai pelaksanaan penelitian Saudara wajib menyampaikan laporan kepada kami 1 (satu) bulan setelah berakhirnya penelitian.

Dikeluarkan di Sleman

Pada Tanggal : 26 Agustus 2015

a.n. Kepala Badan Perencanaan Pembangunan Daerah

Sekretaris

Kepala Bidang Statistik, Penelitian, dan Perencanaan



DR. MARYATUN, S.I.P, MT
Pembina, IV/a
NIP 19720411 199603 2 003

Tembusan :

1. Bupati Sleman (sebagai laporan)
2. Kepala Dinas Dikpora Kab. Sleman
3. Kabid. Sosial & Pemerintahan Bappeda Kab. Sleman
4. Camat Depok
5. Kepala UPT Pelayanan Pendidikan Kec. Depok
6. Kepala SMKN 2 Depok Sleman
7. Dekan Fak. Teknik - UNY
8. Yang Bersangkutan



KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN
UNIVERSITAS NEGERI YOGYAKARTA
FAKULTAS TEKNIK



Alamat : Kampus Karangmalang, Yogyakarta, 55281
Telp. (0274) 586168 psw. 276,289,292 (0274) 586734 Fax. (0274) 586734
website : <http://ft.uny.ac.id> e-mail: ft@uny.ac.id ; teknik@uny.ac.id

Nomor: 1626/H34/PL/2015

22 Juni 2015

Lamp. :

Hal : Ijin Penelitian

Yth.

- 1 . Gubernur DIY c.q. Biro Administrasi Pembangunan Setda DIY
- 2 . Gubernur Provinsi DIY c.q. Ka. Bappeda Provinsi DIY
- 3 . Bupati Kabupaten Sleman c.q. Kepala Badan Pelayanan Terpadu Kabupaten Sleman
- 4 . Kepala Dinas Pendidikan, Pemuda , dan Olahraga Provinsi DIY
- 5 . Kepala Dinas Pendidikan, Pemuda , dan Olahraga Kabupaten Sleman
- 6 . Kepala SMK Negeri 2 Depok Sleman

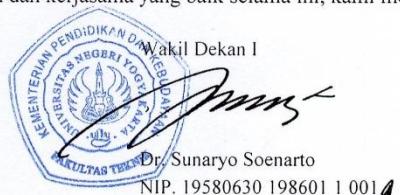
Dalam rangka pelaksanaan Tugas Akhir Skripsi kami mohon dengan hormat bantuan Saudara memberikan ijin untuk melaksanakan penelitian dengan judul Pengembangan Sistem Pakar Perbaikan Komputer Berbasis Android, bagi Mahasiswa Fakultas Teknik Universitas Negeri Yogyakarta tersebut di bawah ini:

| No. | Nama | NIM | Jurusan | Lokasi |
|-----|-----------------|-------------|-------------------------------|---------------------------|
| 1 | Nafis Akhsan .. | 09520241027 | Pend. Teknik Informatika - S1 | SMK Negeri 2 Depok Sleman |

Dosen Pembimbing/Dosen Pengampu :

Nama : Prof. Herman Dwi Surjono, M.Sc, MT.,Ph.D.
NIP : 19640205 198703 1 001

Adapun pelaksanaan penelitian dilakukan mulai Tanggal 1 Agustus 2015 s/d 30 Oktober 2015.
Demikian permohonan ini, atas bantuan dan kerjasama yang baik selama ini, kami mengucapkan terima kasih.



Tembusan :
Ketua Jurusan



PEMERINTAH KABUPATEN SLEMAN
DINAS PENDIDIKAN PEMUDA DAN OLAHRAGA
SMK NEGERI 2 DEPOK
Mrican ,Caturtunggal ,Depok, Sleman Telp. 513515 Fax. 513438
E-mail : simkn2depok@yahoo.com
YOGYAKARTA 55281

SURAT KETERANGAN
Nomor : 070 / 1698

Yang bertanda tangan dibawah ini Kepala Sekolah Menengah Kejuruan Negeri 2 Depok Sleman, menerangkan dengan sesungguhnya bahwa :

N a m a : Nafis Akhsan
No.Induk Mahasiswa : 09520241027
Prodi / Jurusan : Pendidikan Teknik Informatika
: Fakultas Teknik
Universitas Negeri Yogyakarta

Telah melaksanakan Penelitian pada tanggal 6 – 7 Nopember 2015 dengan judul “ Pengembangan Sistem Pakar Perbaikan Komputer Berbasis Android ”.

Demikian surat keterangan ini diberikan untuk dapat digunakan sebagaimana mestinya.

Magan, 12 Nopember 2015
Kepala Sekolah

Dinas Pendidikan, Pemuda, dan Olahraga
SMK NEGERI 2 DEPOK
Drs. Aragani Mizan Zakaria
Penulis, IV/a
NIP. 19630203 198803 1 010

SURAT PERNYATAAN VALIDASI
INSTRUMEN PENELITIAN TUGAS AKHIR SKRIPSI

Saya yang bertanda tangan dibawah ini:

Nama : Drs. Muhammad Munir, M.Pd

NIP : 19630512 198901 1 001

Jurusan : Pendidikan Teknik Elektronika

menyatakan bahwa instrumen penelitian TAS atas nama mahasiswa:

Nama : Nafis Akhsan

NIM : 09520241027

Program Studi : Pendidikan Teknik Informatika

Judul TAS : "Pengembangan Sistem Pakar Perbaikan Komputer Berbasis Android"

Setelah dilakukan kajian atas instrumen penelitian TAS tersebut dapat dinyatakan:

- Layak digunakan untuk penelitian
 Layak digunakan dengan perbaikan
 Tidak layak digunakan untuk penelitian yang bersangkutan

dengan saran/perbaikan sebagaimana terlampir.

Demikian agar dapat digunakan sebagaimana mestinya.

Yogyakarta,.....
Validator

Drs. Muhammad Munir, M.Pd
NIP. 19630512 198901 1 001

Catatan:

- Beri tanda ✓

Hasil Validasi Instrumen Penelitian TAS

Nama Mahasiswa : Nafis Akhsan

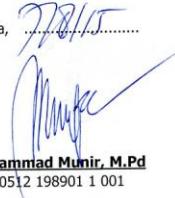
NIM : 09520241027

Judul TAS : "Pengembangan Sistem Pakar Perbaikan Komputer Berbasis Android"

| No. | Variabel | Saran/Tanggapan |
|-----|--------------------------|-------------------|
| | | <i>Siapah OK!</i> |
| | | |
| | | |
| | Komentar Umum/Lain-lain: | |

Yogyakarta, 27/11/.....

Validator,



Drs. Muhammad Munir, M.Pd
NIP. 19630512 198901 1 001

SURAT PERNYATAAN VALIDASI
INSTRUMEN PENELITIAN TUGAS AKHIR SKRIPSI

Saya yang bertanda tangan dibawah ini:

Nama : Dessy Irmawati , M.T

NIP : 19791214 201012 2 002

Jurusan : Pendidikan Teknik Elektronika

menyatakan bahwa instrumen penelitian TAS atas nama mahasiswa:

Nama : Nafis Akhsan

NIM : 09520241027

Program Studi : Pendidikan Teknik Informatika

Judul TAS : "Pengembangan Sistem Pakar Perbaikan Komputer Berbasis Android"

Setelah dilakukan kajian atas instrumen penelitian TAS tersebut dapat dinyatakan:

- Layak digunakan untuk penelitian
- Layak digunakan dengan perbaikan
- Tidak layak digunakan untuk penelitian yang bersangkutan

dengan saran/perbaikan sebagaimana terlampir.

Demikian agar dapat digunakan sebagaimana mestinya.

Yogyakarta, 21 Sept 2015

Validator,



Dessy Irmawati , M.T

NIP. 19791214 201012 2 002

Catatan:

- Beri tanda √

Hasil Validasi Instrumen Penelitian TAS

Nama Mahasiswa : Nafis Akhsan

NIM : 09520241027

Judul TAS

: "Pengembangan Sistem Pakar Perbaikan Komputer Berbasis Android"

| No. | Variabel | Saran/Tanggapan |
|-----|--------------------------|--|
| | Kisi-kisi Instrumen | Tabel kisi-kisi seharusnya blm ke pernyataan. Sebelumnya berisi kriteria & indikator. Pernyataan utk kuisionernya. |
| | | Knp yg ditulis hanya usability ? apakah sudah dibatasi? |
| | | |
| | Komentar Umum/Lain-lain: | |

Yogyakarta, 21 Sept 2015
Validator,



Dassy Irmawati, M.T
NIP. 19791214 201012 2 002