BAB III
METODE PENELITIAN

3.1 Desain Penelitian

Agar penelitian yang dilakukan lebih terarah maka akan digunakan suatu metode. Dalam penelitian ini metode yang digunakan adalah *Backward Chaining*, langkah yang dilakukan dimulai dengan pengumpulan data, pengembangan dan hasil. Tahap identifikasi dan analisis kebutuhan sistem dilakukan untuk mengetahui kebutuhan pengguna terhadap sistem yang akan dikembangkan. Hal ini perlu dilakukan agar sistem yang dikembangkan sesuai dengan kebutuhan pengguna.

Gambar 3.1 Desain Penelitian
Sumber : Pengolahan data penelitian (2019)

1. Identifikasi Masalah

Identifikasi masalah ini dilakukan untuk mengetahui permasalahan ataupun kendala-kendala yang dihadapi oleh masyarakat dalam perbaikan komputer secara umum. Dengan permasalahan yang dihadapi oleh pengguna maka perlu dipersiapkan beberapa masalah dan solusi yang akan diberikan.

2. Menentukan tujuan masalah

Dalam penentuan tujuan permasalahan adalah perancang sistem harus mengetahui permasalahan dan keluhan yang dihadapi oleh masyarakat dalam melakukan perbaikan komputer secara umum dan merancang, membangun sistem yang dapat membantu pengguna dalam mengambil tindakan perbaikan komputer secara umum.

3. Teknik Pengumpulan Data

Untuk teknik pengumpulan data adalah sebuah teknik yang akan dilakukan dan diberikan untuk mendapatkan informasi ataupun data-data pendukung untuk melakukaan perbaikan pada komputer.

4. Mengolah data dengan menggunakan Metode *Backward Chaining*

Penentuan metode *Backward Chaining* adalah sebuah metode yang biasa digunakan oleh perancang sistem informasi baik berbasis *web* maupun Android atau dengan aplikasi lainnya. Dengan adanya metode pada sistem pakar, diharapkan dapat membantu perancang dan pengguna dalam menerapkan sistem yang akan dibangun.

5. Sistem Pakar Berbasis *Web*

Sistem Pakar berbasis *web* ini bertujuan untuk menerapkan sistem yang akan dirancang, dibangun dengan metode *Backward Chaining* dan diimplementasikan dengan *web.*

6. Pengujian Hasil

Tujuan dari pengujian ini adalah untuk melakukan pengujian sistem yang akan dibangun dan akan diuji kelayakkan sistemnya dengan berbantuan *software XAMPP* dalam pengujian sistem secara *Offline.*

3.2 Teknik Pengumpulan Data

Untuk teknik pengambilan data pada penelitian ini adalah dijabarkan sebagai berikut:

1. Observasi dan wawancara

Proses observasi dan wawancaranya dilakukan untuk memperoleh data dan informasi pakar dibidang perbaikan komputer adalah pada CV Duta Komputer dan untuk pakar dibidang perbaikan adalah Bapak Riced. Wawancara dilakukan dengan mempertanyakan beberapa permasalahan dan kendala yang sering dihadapi oleh masyarakat pada saat menggunakan komputer.

2. Studi Literatur

Studi literature ini dilakukan bertujuan untuk memperoleh perbandingan sumber untuk penguatan baik secara teoritis maupun praktikum. Dengan

proses ini maka peneliti bisa membandingkan beberapa informasi yang didapatkan baik secara *online* maupun berdasarkan buku yang ada.

3.3 Operasional Variabel

Definisi operasional merupakan bagian yang mendefinisikan sebuah konsep/variabel agar dapat diukur, dengan cara melihat pada dimensi (indikator) dari suatu konsep/variabel. Dimensi indikator dapat berupa: perilaku, aspek, atau sifat/karakteristik. Dengan demikian, definisi operasional tidak boleh mempunyai makna yang berbeda dengan definisi konseptual. Dengan demikian, definisi operasional bukan berarti definisi/pengertian/makna seperti yang terlihat pada teori dibuku teks, namun lebih menekankan kepada hal-hal yang dapat dijadikan sebagai ukuran/indikator dari suatu variabel dan tidak abstrak (Noor, 2012:97).

Tabel 3.1 *Operasional Variabel* Kerusakan Pada *hardware* komputer

Variabel	Indikator
Power Supply	Korslet
	Lemah/Overhead
RAM	Overload
	Kotor
Motherboard	Korslet
Hardisk	Bad Sector
	Kabel Lemah
	Piringan/Jarum rusak
Processor	Overhead
	Kotor

Sumber : Pengolahan Data Penelitian (2019)

3.4 Metode Perancangan Sistem

Perancangan sistem yang dilakukan dalam membangun sistem pakar mendiagnosa kerusakan pada *hardware* komputer menggunakan metode *backward chaining* adalah pengkodean (Nama Kerusakan, Gejala), memberikan aturan (*Rule*), membuat pohon keputusan. Adapun yang menjadi tahapan yang dilakukan sebagai berikut:

3.4.1 Pengkodean

Pada penelitian ini penulis merancang pengkodean untuk nama kerusakan dan gejala kerusakan yang terjadi untuk mempermudah perancangan *database* yang ada pada sistem. Pengkodean tersebut dapat dilihat dalam tabel dibawah ini.

Tabel 3.2 Kode dan Nama Kerusakan

Variabel	Kode Kerusakan	Indikator
Power Supply	I01	Korslet
	I02	Lemah/Overhead
RAM	I03	Overload
	I04	Kotor
Motherboard	I05	Korslet
Hardisk	I06	Bad Sector
	I07	Kabel Lemah
	I08	Piringan/Jarum rusak
Processor	I09	Overhead
	I10	Kotor

Sumber : Pengolahan Data Penelitian (2019)

Pada Tabel 3.2, dapat dilihat bahwa setiap nama kerusakan diwakili dengan kode berdasarkan indikator masing-masing. Kode I01 dan I05 mewakili kerusakan korslet, kode I02 dan I09 mewakili kerusakan *Overhead*, kode I04 dan I10 mewakili *hardware* kotor, kode I06 mewakili kerusakan *Bad Sector*, I07 mewakili kerusakan kabel lemah, I08 mewakili rusaknya piringan atau jarum dan I09 mewakili *Overload*.

Untuk pengkodean gejala kerusakan dapat dilihat pada tabel dibawah ini :

Tabel 3.3 Kode dan Gejala Kerusakan

Kode Kerusakan	**Kode Gejala**	**Gejala**
I01	G01	Bau tidak sedap
	G02	Tidak menyala
I02	G03	Tidak tampil gambar dimonitor
	G04	Menyala sebentar lalu mati
	G05	Cepat panas
	G06	Kipas tidak berputar
I03	G07	Tidak tampil gambar dimonitor
	G08	Bunyi *beep* beberapa kali
I04	G09	Tidak tampil gambar dimonitor
	G10	Tidak ada bunyi bios indikator

	G11	Lampu indikator menyala tetapi fan prosesor tidak menyala/merespon
I05	G12	Mati total
	G13	Chipset panas
	G14	Tidak tampil gambar dimonitor
	G15	Tidak ada suara *alarm* saat RAM dilepas
I06	G16	Respon lambat
	G17	Saat booting muncul *checkdisk*
	G18	Susah untuk diformat (instal ulang OS)
I07	G19	Tidak terdeteksi dibios
	G20	Jika terdeteksi, proses lama
I08	G21	Tidak terdeteksi dibios
	G22	Ada suara pada hardisk
I09	G23	Tidak tampil gambar pada monitor
	G24	Tidak ada suara *alarm* saat RAM dilepas
I10	G25	Tidak tampil gambar pada monitor

Sumber : Pengolahan Data Penelitian (2019)

Pada Tabel 3.3, dapat dilihat bahwa gejala kerusakan diwakili dengan pengkodean G01-G25. Masing-masing kode terdapat gejala yang akan dipergunakan untuk perancangan *database* sehingga akan lebih mudah

diimplementasikan. Setelah pengkodean nama kerusakan dan gejala kerusakan dibuat, maka perlu aturan (*rule*) untuk mengabungkan keduanya. Berikut dapat dilihat aturan (*rule*) dalam tabel dibawah ini.

Tabel 3.4 Aturan (*Rule*)

Kode Kerusakan	Kode Gejala
I01	G01,G02
I02	G03,G04,G05,G06
I03	G07,G08
I04	G09,G10
I05	G11,G12,G13,G14,G15
I06	G16,G17,G18
I07	G19,G20
I08	G21,G22
I09	G23,G24
I10	G25

Sumber : Pengolahan Data Penelitian (2019)

Pada Tabel 3.4 dapat dilihat bahwa kode kerusakan dan kode gejala dipasangkan dengan aturan (*rule*). Untuk mendapatkan hasil atau kode kerusakan yang terjadi, maka kode gejala dipasangkan sesuai dengan gejala kerusakan sebenarnya. Kode kerusakan I01 dipasangkan dengan kode gejala G01 dan G02, kode kerusakan I02 dipasangkan dengan kode gejala G03-G06, kode kerusakan I03 dipasangkan dengan kode gejala G07 dan G08, kode kerusakan I04

dipasangkan dengan kode gejala G09 dan G10, kode kerusakan I05 dipasangkan dengan kode gejala G11-G15, kode kerusakan I06 dipasangkan dengan kode gejala G16-G18, kode kerusakan I07 dipasangkan dengan kode gejala G19 dan G20, kode kerusakan I08 dipasangkan dengan kode gejala G21 dan G22, kode kerusakan I09 dipasangkan dengan kode gejala G23 dan G24 sedangkan kode kerusakan I10 dipasangkan dengan kode gejala G25.

3.4.2 Aturan (*Rule)*

Dalam penelitian ini pengetahuan direpresentasikan dengan menggunakan aturan berbentuk: *IF-THEN*. Berikut dapat dilihat *rule* yang berkaitan dengan variabel yang digunakan dalam tabel dibawah ini.

Tabel 3.5 Aturan (*Rule*) Kerusakan Dengan Gejala

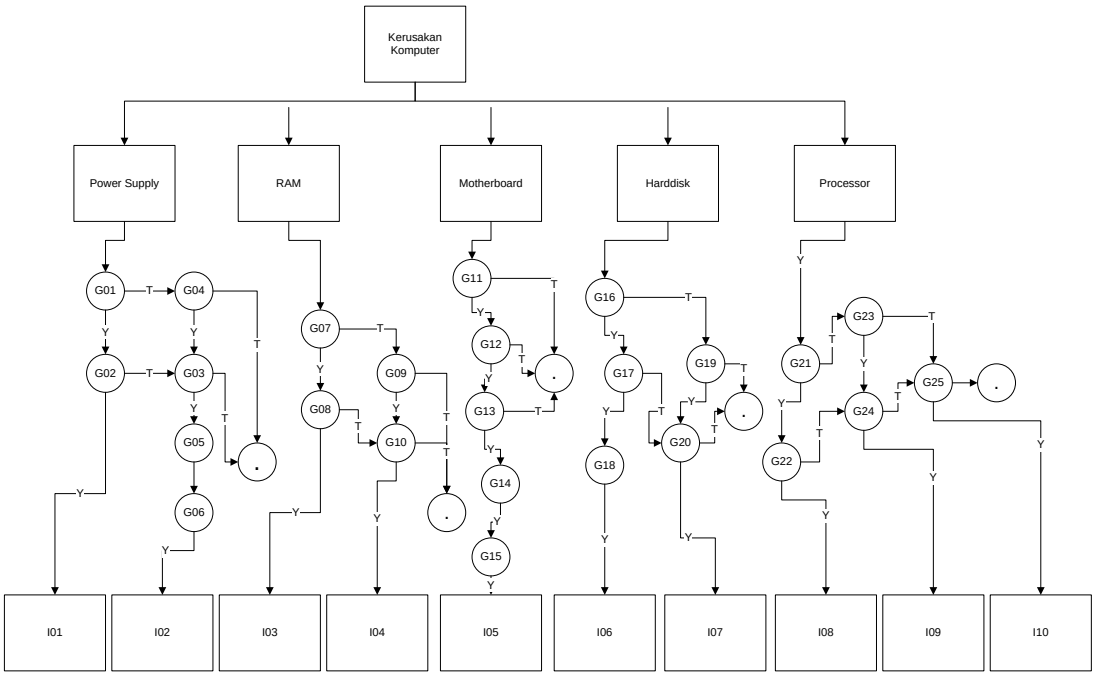
No	Aturan (*Rule*)
1	*IF* G01 *AND* G02 *TRUE THEN* I01 *ELSE* G03
2	*IF* G03,G04,G05 *AND* G06 *TRUE THEN* I02 *ELSE* G07
3	*IF* G07 *AND* G08 *TRUE THEN* I03 *ELSE* G9
4	*IF* G9 *AND* G10 *TRUE THEN* I04 *ELSE* G11
5	*IF* G11,G12,G13,G14 *AND* G15 *TRUE THEN* I05 *ELSE* G16
6	*IF* G16,G17 *AND* G18 *TRUE THEN* I06 *ELSE* G19
7	*IF* G19 *AND* G20 *TRUE THEN* I07 *ELSE* G21
8	*IF* G21 *AND* G22 *TRUE THEN* I08 *ELSE* G23
9	*IF* G23 *AND* G24 *TRUE THEN* I09 *ELSE* G25
10	*IF* G25 *TRUE THEN* I10

Sumber : Pengolahan Data Penelitian (2019)

Pada Tabel 3.4 dapat dilihat bahwa tujuan dari aturan ini sebagai prosedur pemecahan masalah kerusakan. Perlunya mengurutkan langkah-langkah pada setiap permasalahan kerusakan pada tiap *variable hardware* komputer adalah untuk mendapatkan solusi. Jika aturan *IF THEN* bernilai *True,* maka akan menghasilkan solusi dari masing-masing kerusakan sesuai pada tabel yang telah dibuat

3.4.3 Pohon Keputusan

Dalam penelitian ini penulis merancang pohon keputusan berdasarkan aturan *rule* dan fakta-fakta yang ada. Adapun pohon keputusan tersebut adalah :



Gambar 3.2 Pohon Keputusan
Sumber *:* Pengolahan Data Penelitian (2019)

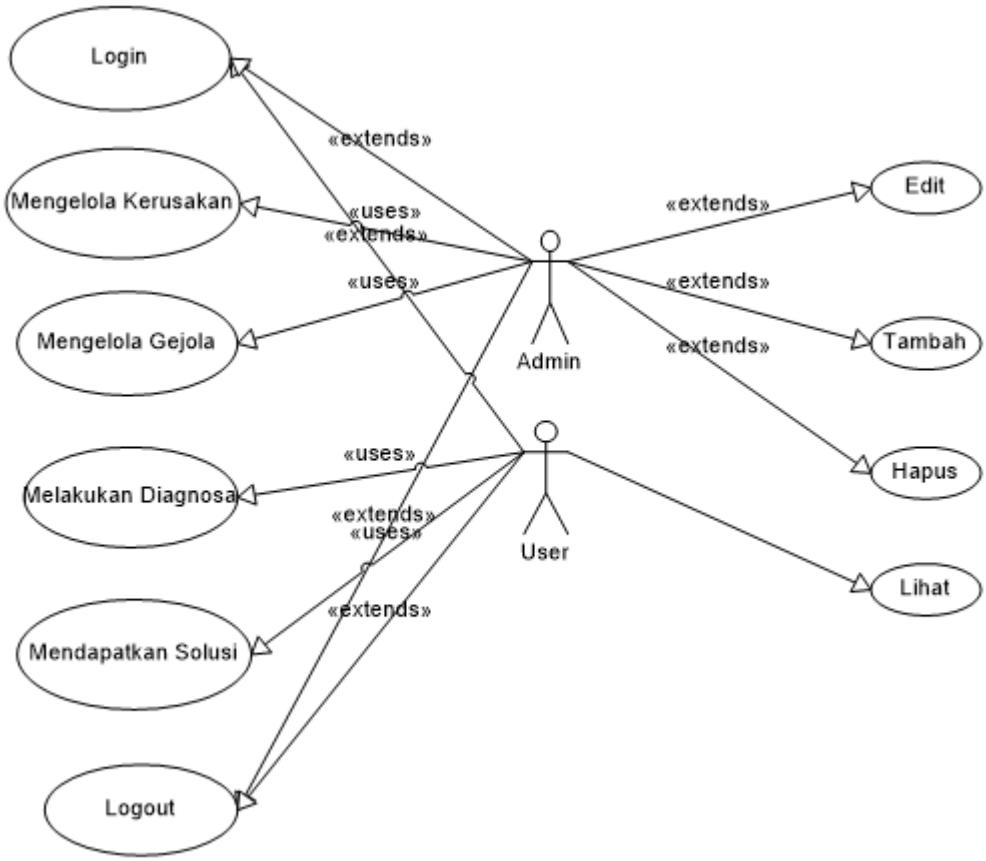
Pada Gambar 3.2 dapat dilihat bahwa pada pohon keputusan menjelaskan alur dari pengetahuan pakar yang akan diimplementasikan ke dalam *database*. Mulai dari G01, G03, G07, G09, G11, G16, G19, G21, G23 dan G 25 menjawab

Tidak maka hasilnya tidak terdiagnosa atau dengan tanda * dan dilanjutkan pada pertanyaan gejala kerusakan yang lainnya.

Sebagaimana proses pembuatan sebuah perangkat lunak, pembuatan sebuah sistem informasi juga memerlukan beberapa buah tahapan. Tahapan-tahapan tersebut dimulai dari desain, perancangan, hingga implementasi dan pengujian. Pembuatan sebuah sistem informasi diawali dengan kebutuhan pengguna berdasarkan permasalahan yang terjadi. Berdasarkan sudut pandang keilmuwan informatika, langkah-langkah tersebut termasuk dalam kajian penelitian. *UML* (*Unified Modelling Language*) adalah standarisasi internasional untuk notasi dalam bentuk grafik, yang menjelaskan tentang analisis dan desain perangkat lunak yang dikembangkan dengan pemrograman berorientasi objek (Pratama, 2014:27).

3.4.4 *Use Case Diagram*

Use Case atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case diagram* menggambarkan fungsionalitas yang diharapkan dari sistem dan merepersentasikan interaksi antara aktor dengan sistem.



Gambar 3.3 *Use Case Diagram*
Sumber : Pengolahan Data Penelitian (2019)

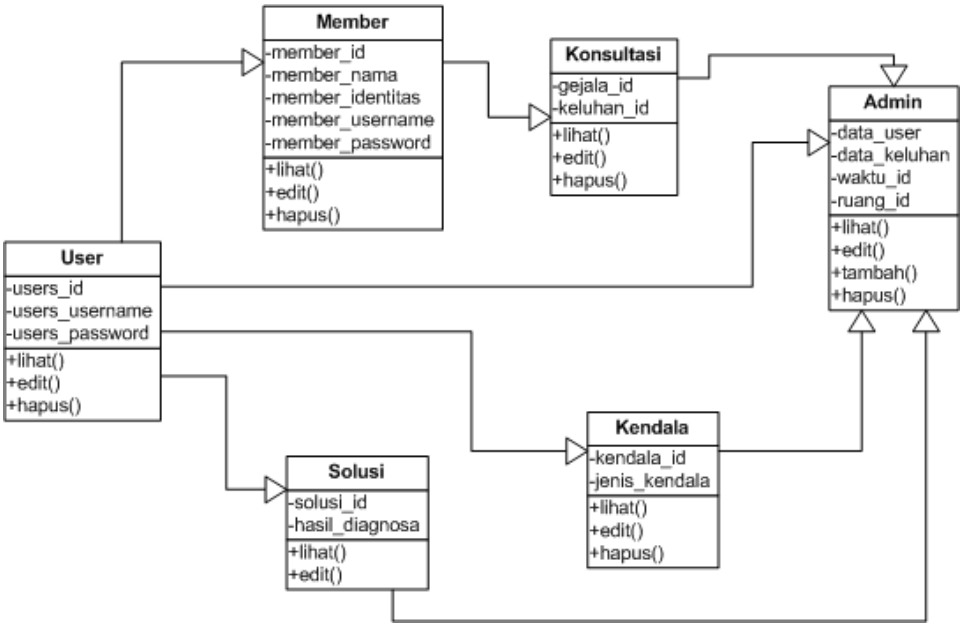
Penjelasan dari Gambar 3.3 *Use Case Diagram*, aplikasi sistem pakar sebagai berikut:

1. Pakar adalah aktor yang menjadi sumber pengetahuan kerusakan mesin *hardware* komputer, dan juga menjadi *administrator* dari sistem aplikasi yang dibangun.
2. *User* adalah aktor yang menggunakan aplikasi sistem pakar.
3. *Login* adalah pintu masuk untuk pakar sebagai *administrator*.
4. Isi data adalah pendaftaran pengguna untuk menjadi *user* dari aplikasi sistem pakar.

5. *Logout* adalah pintu keluar untuk pakar sebagai *administrator.*
6. Mengelola Diagnosa adalah pengetahuan pakar mendiagnosa kerusakan *hardware* komputer.
7. Mengelola Gejala adalah gejala-gejala dari setiap jenis kerusakan *hardware* komputer.
8. Mengelola Solusi adalah solusi atas kerusakan *hardware* komputer.
9. Melakukan Diagnosa adalah pengguna menggunakan fungsi diagnosa kerusakan *hardware* komputer.
10. Mendapatkan Solusi adalah hasil dari diagnosa kerusakan *hardware* komputer.

3.4.5 *Class Diagram*

Class Diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat atribut dan membangun sistem.



Gambar 3.4 *Class Diagram*
Sumber : Pengolahan Data Penelitian (2019)

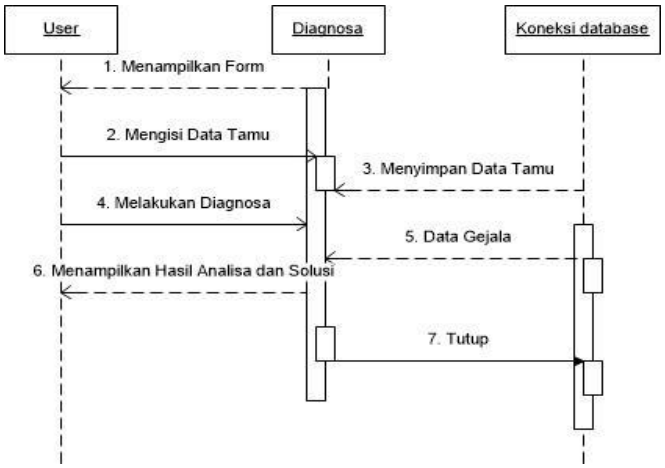
Penjelasan dari Gambar 3.4 *Class diagram*, aplikasi sistem pakar sebagai berikut:

1. *Login* adalah kelas proses yang diambil dari pendefinisian *use case login* untuk pintu masuk pakar *administrator* ke sistem aplikasi pakar.
2. Tamu adalah kelas proses yang diambil dari pendefinisian *use case* untuk aktor yang menggunakan aplikasi sistem pakar.
3. Kelola Diagnosa adalah kelas proses yang diambil dari pendefinisian *use case* mengelola diagnosa yang didalamnya menangani proses gejala dan proses solusi.
4. Kelola Gejala adalah kelas proses yang diambil dari pendefenisian *use case* mengelola gejala yang didalamnya menangani proses pertanyaan gejala untuk setiap jenis kerusakan.
5. Kelola Solusi adalah kelas proses yang diambil dari pendefinisian *use case* mengelola solusi yang didalamnya menangani proses kesimpulan atas proses mengelola diagnosa dan proses mengelola gejala.
6. Pakar adalah kelas data yang digunakan untuk memproses segala pengaksesan terhadap proses mengelola diagnosa, proses mengelola gejala, dan proses mengelola solusi.
7. Koneksi *database* adalah kelas utilitas untuk koneksi ke *database*.

3.4.6 *Sequence Diagram*

Squence Diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek.

1. *Squence Diagram User*

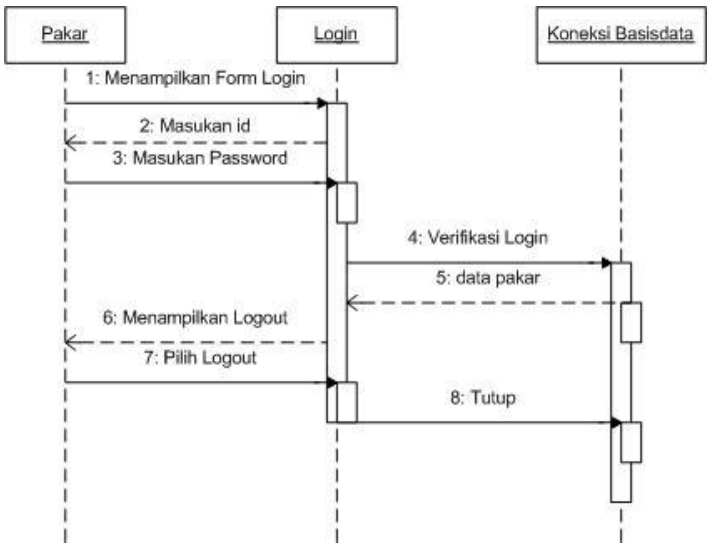


Gambar 3.5 *Squence Diagram Login User*
Sumber : Pengolahan Data Penelitian (2019)

Penjelasan dari Gambar 3.5 *Squence Diagram User*, aplikasi sistem pakar sebagai berikut:

1. Untuk melakukan diagnosa *user* akan mengklik menu diagnosa pada sistem aplikasi pakar dan seterusnya akan tampil form untuk diisi.
2. *Diagnosa* menampilkan *form* yang berisi nama, *hardware* komputer dan *type*. *User* akan mengisi kolom hardware komputer dan *type* berupa angka atau huruf serta mengisi nama pengguna pada kolom nama.
3. Setelah tombol mulai diklik, maka data *User* akan disimpan ke dalam *database*.
4. Proses diagnosa akan dijalankan dengan menjawab pertanyaan.
5. Pertanyaan diagnosa akan muncul satu persatu.
6. Menampilkan hasil diagnosa berdasarkan jawaban pertanyaan.
7. Proses *user* selesai.

2. *Sequence Diagram Login* Pakar



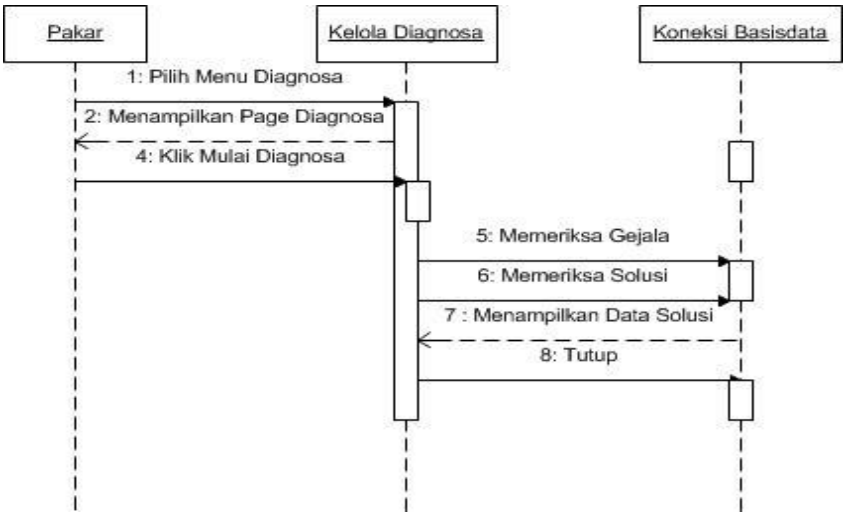
Gambar 3.6 *Squence Diagram Login* Pakar
Sumber : Pengolahan Data Penelitian (2019)

Penjelasan dari Gambar 3.6 *Squence Diagram Login* Pakar, aplikasi sistem pakar sebagai berikut:

1. Pakar memasukan *link* ke *address bar* pada *browser* untuk masuk kedalam *form login* sistem aplikasi pakar.
2. *Login* menampilkan *form* yang berisi *username* dan *password*.
3. Pakar memasukan *username* yang berupa angka atau huruf.
4. Pakar memasukan *password* yang berupa angka atau huruf.
5. Setelah tombol *login* diklik, maka data Pakar diverifikasi kedalam *database*.
6. Jika data *input* benar maka *login* pakar berhasil masuk ke dalam sistem.
7. Tombol *logout* tampil setelah berhasil *login*.
8. Jika ingin keluar dari sistem aplikasi maka silahkan klik tombol *logout*.

9. Proses *login* selesai.

3. *Squence Diagram* Diagnosa

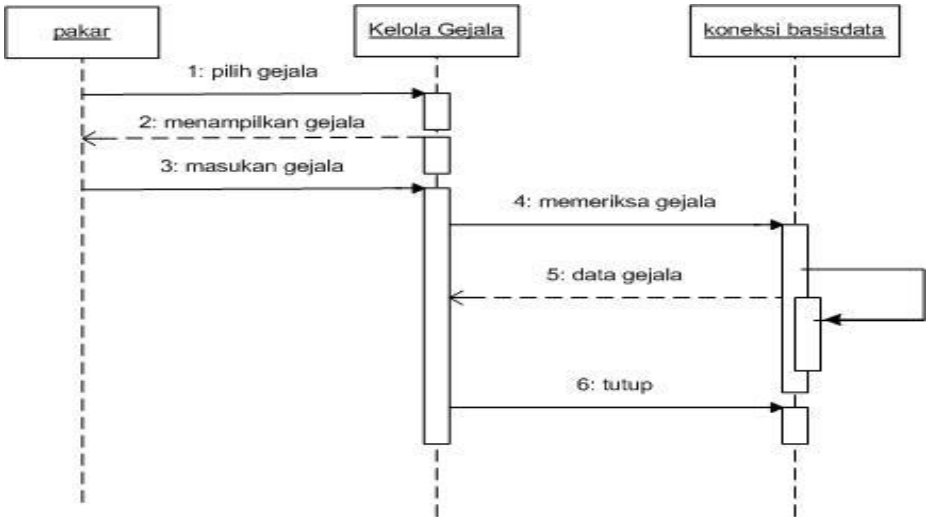


Gambar 3.7 *Squence Diagram* Diagnosa
Sumber : Pengolahan Data Penelitian (2019)

Penjelasan dari Gambar 3.7 *Squence Diagram* Diagnosa, aplikasi sistem pakar sebagai berikut:

1. Pakar memilih *menu* diagnosa untuk mendiagnosa kerusakan.
2. Mengelola Diagnosa terkoneksi ke *database*.
3. Mengelola Diagnosa menampilkan halaman diagnosa.
4. Pakar memulai diagnosa.
5. Mengelola Diagnosa memeriksa gejala dengan mengajukan pertanyaan.
6. Mengelola Diagnosa memeriksa solusi berdasarkan jawaban atas pertanyaan gejala kerusakan.
7. Koneksi basisdata menampilkan solusi kerusakan.
8. Proses Mengelola Diagnosa selesai.

4. *Squence Diagram* Mengelola Gejala

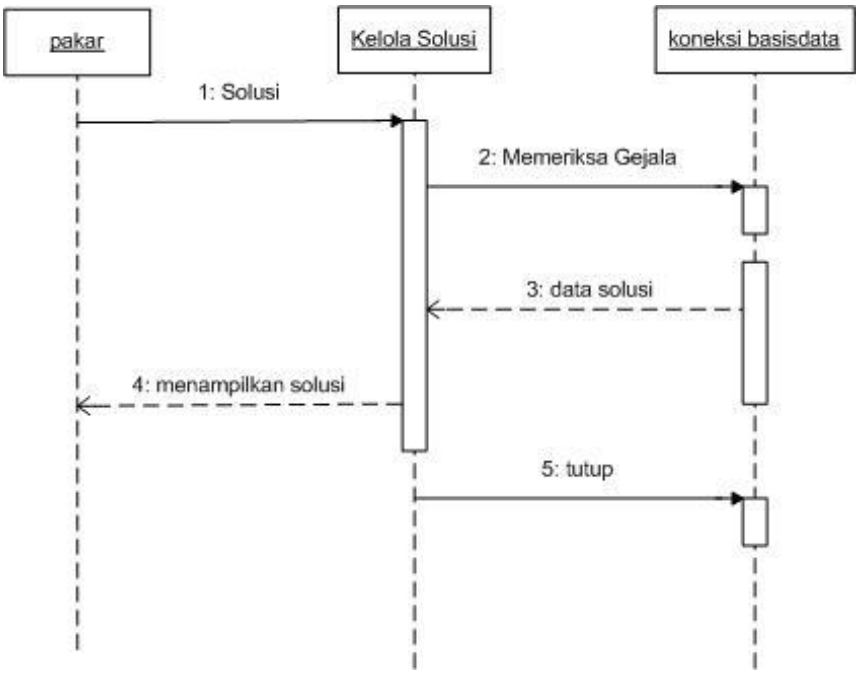


Gambar 3.8 *Squence Diagram* Mengelola Gejala
Sumber : Pengolahan Data Penelitian (2019)

Penjelasan dari Gambar 3.8 *Squence Diagram* Mengelola Gejala, aplikasi sistem pakar sebagai berikut:

1. Pakar menjawab/memilih gejala kerusakan.
2. Mengelola Gejala menampilkan pertanyaan gejala sesuai dengan jenis kerusakan.
3. Pakar memasukan/memilih gejala kerusakan.
4. Mengelola Gejala memeriksa setiap jawaban dari gejala terpilih.
5. Mengelola Gejala memproses dan menampilkan gejala.
6. Proses Mengelola Gejala selesai.

5. *Squence Diagram* Mengelola Solusi



Gambar 3.9 *Squence Diagram* Mengelola Solusi
Sumber : Pengolahan Data Penelitian (2019)

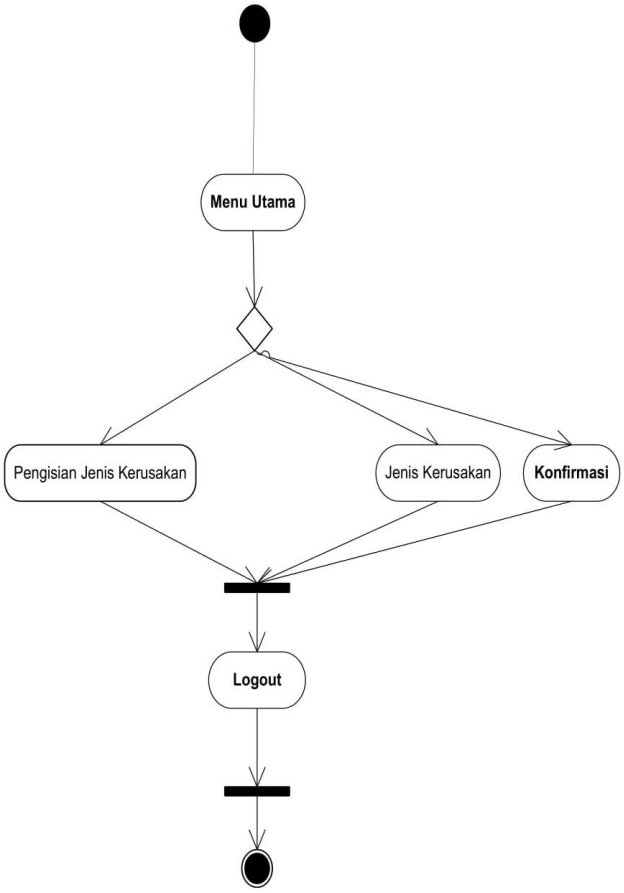
Penjelasan dari Gambar 3.9 *Squence Diagram* Mengelola Solusi, sebagai berikut:

1. Pakar mengikuti proses untuk mendapatkan solusi.
2. Mengelola Solusi memilih gejala yang sudah dipilih pakar.
3. Mengelola Solusi memeriksa hasil pertanyaan gejala sudah dipilih.
4. Mengelola Solusi memproses analisa dari *database*.
5. Mengelola Solusi menampilkan solusi kerusakan.
6. Proses Mengelola Solusi selesai.

3.4.7 *Activity Diagram*

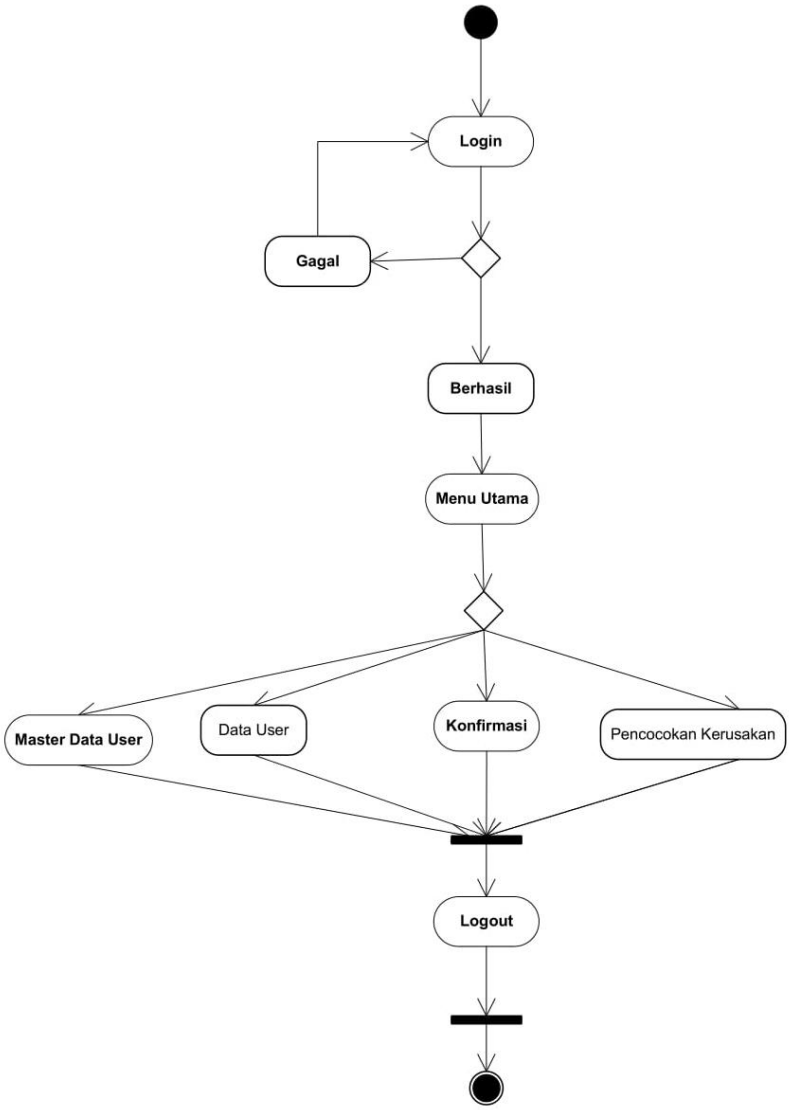
Berikut ini adalah diagaram aktifitas yang menunjukan fungsi *admin* dan *user* pada sistem informasi kerusakan *hardware* pada komputer.

1. *Activity Diagram User*



Gambar 3.10 *Activity Diagram User* Kerusakan *Hardware* Komputer
Sumber : Pengolahan Data Penelitian (2019)

2. ***Activity Diagram Admin***



Gambar 3.11 *Activity Diagram Admin* Kerusakan *Hardware* Komputer
Sumber : Pengolahan Data Peneleitian (2019)

3.4.8 Perancangan *Database*

Rancangan *database* menggambarkan bagaimana rancangan didalam sebuah *database* sistem sehingga sistem tersebut bisa berjalan sesuai dengan rancangan dan implementasi.

1. *Database* Gejala

Tabel 3.6 *Database* Gejala

Field Name	***Type***	***Size***	**Keterangan**
Id_indikator	Int	11	Default Not null
Keterangan_gejala	Varchart	100	Default Not null
Pertanyaan	Varchar	100	Default Not null

2. *Database* Indikator

Tabel 3.7 *Database* Indikator

Field Name	***Type***	***Size***	**Keterangan**
Id_variabel	Int	100	Default Null
Ket_indikator	Varchar	100	Not Null

3. *Database* Periksa

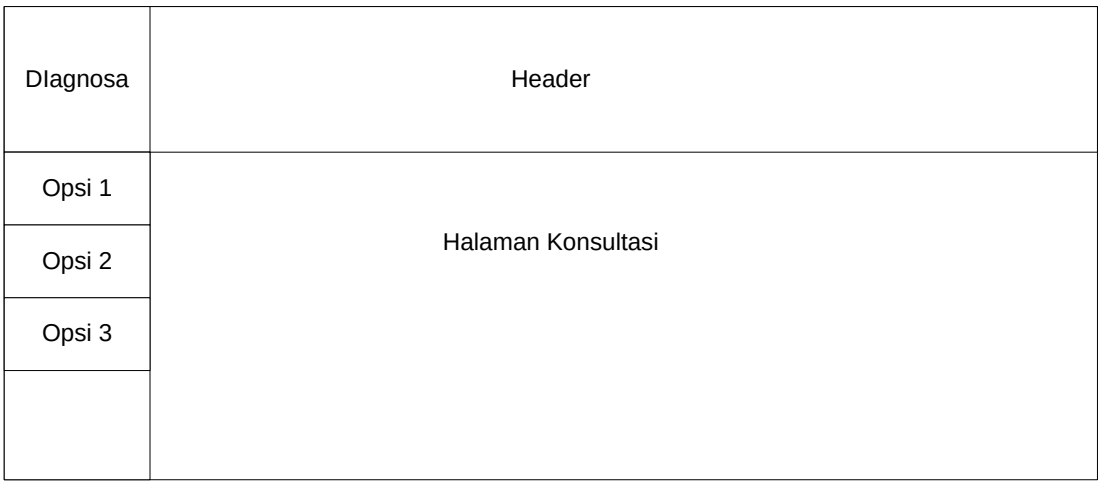
Tabel 3.8 *Database* Periksa

Field Name	***Type***	***Size***	**Keterangan**
Id	Int	11	Not Null
Nama	Varchar	100	Default Null
Gejala	Int	11	Not Null

3.4.9 Perancangan Antar Muka (*Interface*)

Perancangan antarmuka ini bertujuan untuk memberikan gambaran mengenai bentuk antarmuka dari perangkat lunak yang akan digunakan oleh *user* untuk berinteraksi dengan perangkat lunak. Rancangan antar muka ini

mempertimbangkan berbagai kemudahan dan fungsionalitas dari perangkat lunak itu sendiri.



Gambar 3.12 *Interface* Menu Beranda
Sumber : Pengolahan Data Penelitian (2019)

3.5 Lokasi dan Jadwal Penelitian

3.5.1 Lokasi Penelitian

Dalam melakukan penelitian ini penulis meneliti dikota Batam. Penelitian ini dilakukan berdasarkan data-data yang didapatkan dari pihak terkait dengan penelitian ini.

3.5.2 Jadwal Penelitian

Jadwal penelitian untuk memperoleh data dan informasi dilaksanakan pada bulan September 2019 sampai dengan Februari 2020 Sedangkan waktu penelitian ini disesuaikan dengan waktu senggang pembelajaran atau jam tertentu. Berikut jadwal penelitian selengkapnya.

Kegiatan	Jadwal Penelitian																							
	September 2019				Oktober 2019				November 2019				Desember 2019				Januari 2020				Februari 2020			
	I	II	III	IV	I	II	III	IV	I	II	III	IV	I	II	III	IV	I	II	III	IV	I	II	III	IV
Pengajuan Judul																								
Penyusunan Bab I																								
Penyusunan Bab II																								
Penyusunan Bab III																								
Penyusunan Bab IV																								
Penyusunan Bab V, daftar pustaka, lampiran																								

Pengumpulan Skripsi																							

Tabel 3.9 Jadwal Penelitian
Sumber : Pengolahan Data Penelitian (2019)