PENERAPAN METODE *NAIVE BAYES CLASSIFIER* PADA SISTEM PAKAR UNTUK MENDIAGNOSIS PENYAKIT AYAM *BROILER* DAN PETELUR (Studi Kasus pada Peternakan Ayam *Broiler* dan Petelur H.Anas di Kaliwining, Jember)

SKRIPSI

diajukan guna melengkapi tugas akhir dan memenuhi salah satu syarat untuk menyelesaikan Program Studi Sistem Informasi (S1) dan mendapatkan gelar Sarjana Komputer

Oleh
Fidia Indriana
NIM 112410101076

PROGRAM STUDI SISTEM INFORMASI
FAKULTAS ILMU KOMPUTER
UNIVERSITAS JEMBER
2018

PENERAPAN METODE *NAIVE BAYES CLASSIFIER* PADA SISTEM PAKAR UNTUK MENDIAGNOSIS PENYAKIT AYAM *BROILER* DAN PETELUR (Studi Kasus pada Peternakan Ayam *Broiler* dan Petelur H.Anas di Kaliwining, Jember)

SKRIPSI

diajukan guna melengkapi tugas akhir dan memenuhi salah satu syarat untuk menyelesaikan Program Studi Sistem Informasi (S1) dan mendapatkan gelar Sarjana Komputer

Oleh
Fidia Indriana
NIM 112410101076

PROGRAM STUDI SISTEM INFORMASI
FAKULTAS ILMU KOMPUTER
UNIVERSITAS JEMBER
2018

PERSEMBAHAN

Skripsi ini saya persembahkan untuk:

1. Allah SWT, Tuhan Yang Maha Pengasih dan Penyayang yang senantiasa memberikan kemudahan dan kelancaran dalam menyelesaikan skripis ini;
2. Bapak Jumadi dan Ibu Supiana tersayang yang tiada hentinya memberikan doa, dukungan dan motivasi yang luar biasa;
3. Ketiga Saudara Saya Sofiana, Jainul Huda, dan Arifi efendi untuk perhatian, dukungan dan semangat yang diberikan;
4. Sahabat terdekat saya serta teman seperjuangan Siti Fatmawati, Merry Hadiyahwati, Eka Suptyowati, Laura Yohana yang selalu memberikan dukungan dan motivasi terbaik;
5. Kakak dan teman-teman angkatan yang selalu memberikan semangat agar terus bersabar;
6. Bapak Yanuar, Andre Bhaskoro, Riska Arimanudin yang telah membantu dalam menyelesaikan skripsi;
7. Guru – guruku sejak sekolah dasar hingga perguruan tinggi;
8. Almamater Fakultas Ilmu Komputer Unviersitas Jember;

MOTO

"Kesalahan akan membuat orang belajar dan lebih baik"

"Jawaban sebuah keberhasilan adalah terus belajar dan tak kenal putus asa"

PERNYATAAN

Saya yang bertanda tangan di bawah ini:

Nama : Fidia Indriana

NIM : 112410101076

menyatakan dengan sesungguhnya bahwa karya tulis ilmiah dengan judul "Penerapan Metode *Naive Bayes Classifier* Pada Sistem Pakar Untuk Mendiagnosis Penyakit Ayam *Broiler* dan Petelur" adalah benar-benar karya sendiri, kecuali kutipan yang sudah saya cantumkan pada referensi daftar pustaka, belum pernah diajukan pada institusi manapun, dan bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

Demikian pernyataan ini saya buat dengan sebenar-benarnya, tanpa ada tekanan dan paksaan dari pihak manapun, serta bersedia mendapat sanksi akademik apabila di kemudian hari pernyataan ini tidak benar.

Jember, 23 Januari 2018

Yang menyatakan,

Fidia Indriana

NIM 112410101076

SKRIPSI

PENERAPAN METODE *NAIVE BAYES CLASSIFIER* PADA SISTEM PAKAR UNTUK MENDIAGNOSIS PENYAKIT AYAM *BROILER* DAN PETELUR (Studi Kasus pada Peternakan Ayam Broiler dan Petelur H.Anas di Kaliwining, Jember)

Oleh

Fidia Indriana

NIM 112410101076

Pembimbing :

Dosen Pembimbing Utama : Anang Andrianto., S.T., M.T.

Dosen Pembimbing Pendamping : Yanuar Nurdiansyah S.T., M.Cs.

PENGESAHAN PEMBIMBING

Skripsi berjudul "Penerapan Metode *Naive Bayes Classifier* Pada Sistem Pakar Untuk Mendiagnosis Penyakit Ayam *Broiler* dan Petelur" telah diuji dan disahkan pada:

hari, tanggal : Selasa, 23 Januari 2018
tempat : Program Studi Sistem Informasi Universitas Jember.

Disetujui oleh :

Pembimbing I,	Pembimbing II,
Anang Andrianto.,S.T.,M.T	Yanuar Nurdiansyah S.T., M.Cs.
NIP 196906151997021002	NIP 198201012010121004

PENGESAHAN PENGUJI

Skripsi berjudul "Penerapan Metode *Naive Bayes Classifier* Pada Sistem Pakar Untuk MendiagnosisPenyakit Ayam *Broiler* dan Petelur" telah diuji dan disahkan pada:

hari, tanggal : Selasa, 23 januari 2018

tempat : Program Studi Sistem Informasi Universitas Jember.

Tim penguji:

Penguji I,	Penguji II,
Nelly Oktavia A, S.Si., MT.	Nova El Maidah, S.Si., M.Cs
NIP. 198410242009122008	NIP. 198411012015042001

Mengesahkan

Ketua Program Studi,

Prof. Drs. Slamin,M.Comp.Sc.,Ph.D

NIP 196704201992011001

RINGKASAN

Penerapan Metode *Naive Bayes Classifier* Pada Sistem Pakar Untuk Mendiagnosis Penyakit Ayam *Broiler* dan Petelur; Fidia Indriana, 112410101076; 2016: 147 halaman; Progam Studi Sistem Informasi Fakultas Ilmu Komputer Universitas Jember.

Peternakan merupakan kegiatan mengembangbiakkan dan membudidayakan hewan ternak untuk mendapat manfaat atau keuntungan dari kegiatan tersebut. Setiap peternak ayam, baik dalam skala kecil maupun skala besar, tentu sangat memperhatikan kesehatan ayam.Kesehatan ayam berpengaruh pada keuntungan yang akan didapat peternak, oleh sebab itu Sistem Pakar untuk mendiagnosis penyakit ayam menggunakan metode *Naive Bayes Classifier* dikembangkan untuk membantu pengguna khususnya peternak ayam *broiler* dan petelur dalam mendiagnosis penyakit beserta saran atau solusi penanggulangan yang dapat direkeondasikan. Pada penelitian ini, menggunakan sebanyak 13 penyakit yang didalamnya memiliki gejala sebanyak 50. Proses diagnosis penyakit dilakukan dengan memilih gejala yang tampak pada ayam dan sesuaikan dengan gejala yang terjadi pada ayam. Nilai probabilitas yang dihasilkan dari perhitungan Teorema *Bayes* yang diterapkan pada sistem pakar akan diurutkan berdasarkan nilai *bayes* pada tiap-tiap penyakit yang terbesar, kemudian mengambil nilai yang terbesar sebagai hasil diagnosis.

Sistem ini telah diuji tingkat akurasi hasil diagnosanya, tingkat ketepatan diagnosa yang didapatkan hingga mencapai 80% dengan menggunakan 10 data sebagai sampel. Pengujian dilakukan dengan cara mendiagnosa beberapa sampel (ganguan yang valid/jumlah seluruh percobaan*100%) pada sampel yang berbeda dalam setiap pengujiannya. Sistem juga melalui uji Validitas dan Reliabilitas Hasil uji validitas menunjukkan 10 item yang diajukan sebagai pertanyaan terhadap 30 responden seluruhnya menghasilkan nilai t hitung > t tabel yang keseluruhan item tersebut menghasilkan nilai yang valid. Hasil uji reliabilitas menunjukkan koefisien

reliabilitas sebesar 0,654545 lebih besar daripada 0,60 yang merupakan nilai minimal tingkat reliabilitas suatu instrumen yang reliabel sehingga dapat disimpulkan bahwa instrumen tersebut memiliki ingkat reliabilitas yang baik.

PRAKATA

Puji syukur kehadirat Allah SWT atas segala rahmat dan karunia-Nya, sehingga penulis dapat menyelesaikan skripsi dengan judul Penerapan Metode *Naive Bayes Classifier* Pada Sistem Pakar Untuk Mendiagnosa Penyakit Pada Aym Broiler dan Petelur. Skripsi ini disusun untuk memenuhi salah satu syarat menyelesaikan pendidikan Strata Satu (S1) pada Program Studi Sistem Informasi, Fakultas Ilmu Komputer Universitas Jember. Penyusunan skripsi ini tidak lepas dari bantuan berbagai pihak. Oleh karena itu, penulis menyampaikan terima kasih kepada:

1. Prof. Drs. Slamin, M.Comp.Sc., Ph.D., selaku Ketua Program Studi Sistem Informasi Universitas Jember;
2. Anang Andrianto, S.T., M.T., selaku Dosen Pembimbing Utama dan Yanuar Nurdiansyah S.T., M.Cs. selaku Dosen Pembimbing Anggota yang telah meluangkan waktu, pikiran, dan perhatian dalam penulisan skripsi;
3. Nurdiansyah S.T., M.Cssebagai dosen pembimbing akademik, yang telah mendampingi penulis sebagai mahasiswa;
4. Bapa Jumadi dan Ibu Supiana tercinta yang selalu mendukung dan mendoakan;
5. Ketiga Saudara Saya Sofiana, Jainul Huda, dan Arifi efendi serta keluarga.

Penulis menyadari bahwa laporan ini masih jauh dari sempurna, oleh sebab itu penulis mengharapkan adanya masukan yang bersifat membangun dari semua pihak. Penulis berharap skripsi ini dapat bermanfaat bagi semua pihak.

Jember, 19 januari 2018

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I. PENDAHULUAN

1.1 Latar Belakang

Peternakan merupakan kegiatan mengembangbiakkan dan membudidayakan hewan ternak untuk mendapat manfaat atau keuntungan dari kegiatan tersebut. Di Kabupaten Jember terdapat 27 kecamatan yang melakukan usaha peternakan khususnya peternakan ayam baik ayam broiler maupun ayam ras peterlur, dari 27 kecamatan memperoleh data jumlah peternakan sbb: jumlah peternakan kemitraan ayam pedaging berjumlah 202, jumlah peternakan mandiri ayam pedaging 66, dan jumlah peternakan mandiri ayam ras petelur 222 peternakan (Dinas Peternakan, Perikanan dan Kelautan Kabupaten Jember, 2016).

Setiap peternak ayam, baik dalam skala kecil maupun skala besar, tentu sangat memperhatikan kesehatan ayam. Kesehatan ayam berpengaruh pada keuntungan yang akan didapat peternak, dari data yang didapat hanya peternakan kemitraan ayam pedaging saja yang mempunyai dr.Hewan atau tenaga PPL untuk mengatasi ternaknya yang mengalami masalah kesehatan, sedangkan untuk peternakan mandiri tidak mempunyai dr.Hewan khusus atau tenaga kesehatan lainnya dikarenakan alasan biaya. Padahal, kebutuhan informasi yang cepat dan tepat dari seorang pakar kesehatan hewan sangatlah dibutuhan untuk meningkatkan kesehatan ayam.

Penyakit merupakan salah satu resiko yang tinggi dan harus selalu dihadapi, antisipasi untuk mencegah dan mengenali gejala penyakit yang berbahaya sangatlah penting. Proses untuk mengenali dengan cepat dan tepat dari serangan jenis penyakit sangatlah sulit karena gejala yang ditimbulkan umumnya mirip dan sama. Akan tetapi, biasanya ada beberapa gejala yang khas untuk setiap jenis penyakit pada ayam. Penyakit tersebut dapat disebabkan oleh serangan virus bakteri, jamur, dan parasit. (Zulkarnaen, 2013).

Penyakit-penyakit unggas baik penyakit ringan maupun penyakit berat yang bisa menular kepada manusia merupakan kendala terberat dalam usaha budidaya ternak

unggas. Serangan penyakit pada ternak unggas merupakan penyebab utama gagalnya suatu usaha budidaya. Selain penyakit-penyakit menular yang tidak mematikan pun perlu mendapatkan perhatian, mengingat penyakit-penyakit tersebut juga menimbulkan kerugian ekonimi yang cukup besar. Pengamanan terhadap penyakit harus mendapatkan prioritas dan perhatian khusus, dimana pengendalian tersebut terdiri dari usaha pencegahan, pengobatan, dan pembasmian.

Berdasarkan permasalahan peternakan ayam diatas, maka diperlukan pengetahuan tentang berbagai jenis penyakit yang menyerang peternakan tersebut, sehingga peternak dapat melakukan penanganan lebih awal untuk mengatasi masalah yang dihadapi. Kesimpulan dari berbagai masalah yang telah dijelaskan diatas yaitu diperlukan suatu sistem yang dapat membantu dan mempermudah peternak untuk mengetahui penyakit yang diderita ayam dengan menggunakan metode *Naive Bayes Classifier*.

1.2 Rumusan Masalah

Dengan mempertimbangkan latar belakang diatas, dapat dirumuskan beberapa rumusan masalah yaitu:

1. Bagaimana menerapkan metode *Naive Bayes* sebagai mesin inferensi untuk sistem pakar mendiagnosis penyakit pada ayam petelur dan *broiler*?
2 Bagaimana membangun sistem pakar berbasis web untuk mendiagnosis penyakit pada ayam petelur dan *broiler*?

2.1 Tujuan dan Manfaat

Tujuan dan manfaat merupakan jawaban dari rumusan masalah yang telah diuraikan di atas serta tujuan yang ingin dicapai pada penelitian ini.

2.1.1 Tujuan

Berdasarkan rumusan masalah yang telah disampaikan sebelumnya maka dapat ditetapkan tujuan yang ingin dicapai dari penelitian ini, yaitumerancangdan membangun sebuah aplikasi untuk mendiagnosis penyakit pada ayam *broiler* dan

ayam ras petelur menggunakan metode *naive bayes classifier* pada peternakan ayam *broiler* dan peternakan ayam ras petelur H.Anas di Kaliwining Jember, untuk mencapai tujuan tersebut peneliti menggunakan model *waterfall* sebagai metode pengembangan aplikasi dalam melakukan penelitian.

1.3.2 Manfaat

Manfaat penelitian ini adalah

1. Manfaat Akademis

   Hasil penelitian ini diharapkan dapat memberikan informasi dan studi literatur bagi dunia pendidikan, khususnya di bidang sistem informasi.

2. Manfaat bagi peternakan

   Sistem ini bermanfaat bagi peternakan dalammendiagnosis penyakit pada ternaknya yaitu ayam *broiler* dan ayam ras petelur, dengan menerapkan metode *naive bayes classifier* diharapkan dapat menentukan jenis penyakit dan memberi solusi pada peternak yang mengalami masalah dengan ternaknya.

3. Manfaat bagi penulis

   Penulis dapat mengetahui bagaimana proses penerapan metode *naivebayes classifier*dalammendiagnosis penyakit pada ayam *broiler* dan ayam ras petelur.

1.4 Batasan Masalah

1. Aplikasi hanya mencatat data-data gejala penyakit.
2. Aplikasi hanya membantu mendiagnosis penyakit pada ayam broiler dan ayam ras petelur serta memberi solusi terhadap penyakit tersebut.
3. Aplikasi hanya mendiagnosis penyakit yang sering menjangkit ayam petelur dan ayam *broiler*
4. Aplikasi dapat menambahkan data tes menjadi dataset.

5. Mendiagnosis penyakit ini menggunakan metode *naive bayes classifier* berbasis web.

1.5 Sistematika Penulisan

Adapun sistematika penulisan skripsi ini adalah sebagai berikut:

1. Pendahuluan.

   Bab ini memuat uraian mengenai latar belakang, rumusan masalah, tujuan dan manfaat, batasan masalah, dan sistematika penulisan skripsi yang masing-masing tertuang pada subbab tersendiri.

2. Tinajuan Pustaka.

   Bab ini memaparkan tinjauan terhadap hasil-hasil penelitian terdahulu yang berkaitan dengan masalah yang dibahas, landasan materi dan konsep untuk mendiagnosis penyakit ayam broiler dan petelur, dan kajian teori metode analisis data yang berkaitan dengan masalah dalam penelitian.

3. Metodologi Penelitian.

   Bab ini menguraikan tentang tempat dan waktu penelitian, metode penelitian, metode pengumpulan data, metode analisis data dan teknik pengembangan sistem yang digunakan dalam penelitian.

4. Pengembangan Sistem.

   Bab ini berisi uraian tentang proses pengembangan sistem. Pengembangan sistem dimulai dari ananlisis kebutuhan fungsional dan non-fungsional sistem, kemudian merancang *Business Process, Usecase Diagram, Scenario, Activity Diagram, Class Diagram* dan *Entity Relationship Diagram* (ERD), kemudian penulisan kode program, lalu proses pengujian sistem.

5. Hasil dan Pembahasan.

   Bab ini memaparkan secara rinci pemecahan masalah melalui analisis yang disajikan dalam bentuk deskripsi dibantu dengan ilustrasi berupa table dan gambar untuk memperjelas hasil penelitian.

6. Penutup.

   Bab ini berisi kesimpulan penelitian yang telah dilakukan dan saran untuk penelitian selanjutnya.

BAB 2. TINJAUAN PUSTAKA

Pada bagian ini menjelaskan teori – teori serta pustaka yang digunakan untuk penelitian. Teori – teori ini diambil dari berbagai literature, buku dan jurnal.

2.1 Penelitihan Terdahulu

Sebelumnya yang berjudul "Sistem Pakar Diagnosis Penyakit Mata Menggunakan *Naive Bayes Classifier*" yang dilakukan oleh Wahyudi Setiawan dan Shofie Ratnasari, Universitas Trunojoyo Madura, melakukan penelitian tentang sistem pakar untuk mendiagnosis penyakit mata. Data yang digunakan untuk penelitian terdiri dari 52 gejala dan 15 penyakit mata, sistem pakar yang digunakan metode *Naive Bayes Classifier*. Terdapat dua tahapan kerja dari aplikasi tersebut, sistem meminta pasien untuk menginputkan gejala-gejala yang dialami. Kedua, sistem secara otomatis menampikan hasil *Naive Bayes Classifier* diagnosis dari penyakit mata yang diderita pasien menggunakan perhitungan *Naive Bayes Classifer*. (Setiawan dan Ratnasari, 2014).

Sebelumnya yang berjudul "Sistem Pakar Diagnosa Penyakit Ayam" yang dilakukan oleh Meilany Nonsi Tentua Dosen Program Studi Teknik Informatika Universitas PGRI Yogyakarta, melakukan perhitungan mengenai identifikasi penyakit ayam berdasarkan gejalanya dengan menerapkan metode *Forward Chaining* dan *Backward Chaining*. Sistem pakar tersebut memberikan informasi yang cepat tentang penyakit yang diderita oleh ayam dan cara penanggulangannya. (Nonsi Tentua, 2009).

Berdasarkan hal tersebut, peneliti akan membandingkan juga dalam hal perbedaan Objek yang diteliti (Penyakit Mata) oleh peneliti diatas apakah dengan Metode Teorema *Bayes* juga dapat diterapkan pada Objek Penyakit Pada Ayam hingga aplikasi yang dikembangkan mampu memberikan solusi layaknya seorang pakar seperti penelitian yang pernah dilakukan oleh peneliti sebelumnya. Selain itu,

hal lain yang mendorong penulis menggunakan metode teorema bayes yaitu ayam memiliki gejala-gejala yang nampak dan memiliki tahapan yang sama dalam hal mengidentifikasi penyakit seperti halnya saat mengidentifikasi penyakit Mata.

2.2 Sistem Pakar

Menurut Giarratano dan Riley (2005), Sistem pakar adalah cabang kecerdasan buatan yang menggunakan pengetahuan / *knowledge k*husus untuk memecahkan masalah pada level human *expert*/pakar. Sistem pakar banyak dikembangkan dalam berbagai ilmu, salah satu diantaranya dalam bidang kedokteran untuk melakukan diagnosis penyakit. Sistem pakar digunakan untuk menentukan diagnosis penyakit akan membantu mengkonfirmasi diagnosa dan menentukan saran dan terapinya.

penjelasan diatas dapat disimpulkan bahwa sistem pakar merupakan suatu program komputer yang dibuat lebih menyerupai seorang pakar dan membantu memecahkan masalah seorang pengguna.

2.3 Peternakan

Peternakan adalah kegiatan mengembangbiakkan dan membudidayakan hewan ternak untuk mendapatkan manfaat dan hasil dari kegiatan tersebut. Pengertian peternakan tidak terbatas pada pemeliharaan saja, memelihara dan peternakan perbedaannya terletak pada tujuannya yang ditetapkan. Tujuan peternakan adalah mencari keuntungan dengan penerapan prinsip-prinsip manajemen pada faktor-faktor produksi yang telah dikombinasikan secara optimal. Kegiatan di bidang peternakan dapat dibagi atas dua golongan, yaitu peternakan hewan besar seperti sapi, kerbau, dan kuda, sedangkan kelompok kedua yaitu peternakan hewan kecil seperti ayam,kelinci, dan hewan ternak kecil lainnya.

2.4 Ayam

Ayam meruupakan hewan unggas yang paling banyak dipelihara masyarakat baik secara tradisional yang sering disebut ayam kampung sampai peternakan besar yang

berupa ayam pedaging atau petelur, karena populasinya yang cukup banyak bila dibandingkan hewan lainnya.

Ayam boiler adalah ayam yang tujuan pemeliharaannya menghasilkan daging tipe ayam ini mempunyai ciri-ciri yaitu mempunyai ukuran badan yang besar dan kokoh,timbangan badannya berat, banyak daging dan lemak,ototkaki sebelah belakang tebal, dan produksi telur sedikit. Ayam-ayam seperti ini contohnya misal: *Cornish, Sussex,* berasal dari Inggris dan ayam-ayam kelas Asia seperti: *Brahma, Langshans, Cornish juga Hybro, Starbro, Cobb* dan lain sebagainya. Klasifikasi berdasarkan tipe dari ayam pedaging ini adalah bersifat tenang,bentuk tubuhnya besar, pertumbuhannya cepat, bulu merapat ke tubuh, kulit putih, dan produksi telurnya sedikit.

Sedangkan ayam petelur adalah ayam yang tujuan pemeliharaannya untuk menghasilkan telur. Tipe ayam petelur ini adalah memiliki karakteristik bersifat *nervous* atau mudah terkejut, bentuk tubuh ramping, telinga berwarna putih, dan kerabang telur berwarnah putih. Karakteristik lainnya yaitu produksitelur tinggi (200butir/ekor/tahun), efisien dalam ransum untuk membentuk telur, dan tidak memiliki sifat mengeram. Berdasarkan nilai tujuan dan nilai ekonomisnya tipe ayam petelur memiliki tubuh yang langsing atau berukuran kecil,Timbangan badan ringan,Jengger dan pial baik pada yang jantan ataupun pada yang betina relatif besar. Hampir tidak ada sifat yang mengeram,Produksi telur tinggi dan besar-besar. Yang termasuk tipe ini misalnya ayam-ayam dari klas lalut Tengah seperti *Leghorn, Ancomas, Minorca.*

2.5 Penyakit Ayam

Para peternak ayam menghadapi beberap kendala dalam beternak ayam. Kendalanya antara lain adalah banyaknya serangan dari berbagai penyakit, mulai dari penyakit yang ringan sampai yang mematikan bahkan penyakit yang dapat menular kepada manusia. Kerugian yang disebabkan oleh penyakit tersebut dapat berupa kematian atau penurunan produksi (Zulkarnaen, 2013).

Penyakit pada ternak secara umum terbagi menjadi penyakit infeksi dan penyakit non infeksi. Penyakit infeksius adalah penyakit yang disebabkan oleh agen-agen infeksi. Agen-agen infeksi penyebab penyakit antara lain virus, bakteri, mikal, parasit. Sedangkan penyakit non infeksius adalah penyakit yang disebabkan selain agen infeksi misalnya akibat defisien nutrisi, defisiensi vitamin, disiensi mineral dan keracunan pakan (Tariakoso, 2009).

Banyak jenis-jenis penyakit yang sering ditemukan pada ternak unggas. Penyakit unggas yang disebabkan oleh beberapa jenis mikroorganisme, seperti virus, bakteri, jamur, dan parasit. Gejala dan akibat yang ditimbulkan oleh serangan mikroorganisme tersebut berbeda-beda, baik penyakit ringan, penyakit menular maupun penyakit yang mematikan. Berdasarkan penyebabnya, penyakit pada unggas dibedakan menjadi 4 kelompok, yaitu sebagai berikut:

1. Penyakit Viral adalah penyakit unggas yang disebabkan oleh virus.
2. Penyakit Bakteri adalah penyakit unggas yang disebabkan oleh bakteri.
3. Penyakit Mikal adalah penyakit unggas yang disebabkan oleh jamur.
4. Penyakit Parasit adalah penyakit uanggas yang disebabkan oleh Organisme Parasit.

Terdapat 13 jenis penyakit utama yang sering menyerang pada ayam ternak. Beberapa contohnya adalah Snot/Coryza, Berak Kapur, CRD dan Gumboro selengkapanya dapat dilihat pada lampiran A. Dari 13 penyakit tersebut gejala yang paling sering muncul adalah nafsu makan turun, sesak nafas, dan diare. Nafsu makan berkurang dapat dilihat dari pola makan ayam yang berubah. Biasanya ayam dewasa makan 125 gram per satu ekor ayam setiap harinya, jika terjadi penumpukan makanan maka dapat dilihat bahwa nafsu makan yang berkurang. Produksi telur menurun dapat dilihat dari menurunnya persentase telur ayam setiap harinya. Produksi telur ayam yang optimal biasanya sekitar 85% dari jumlah ayam dan dalam 1 tahun ayam dapat menghasilkan kurang lebih 330 butir telur. Diare dapat dilihat dari banyaknya kotoran

ayam pada kandang serta biasanya disertai dengan mencret hijau, mencret putih atau mencretdarah.

Penanganan terhadap penyakit ayam terdiri dari upaya pencegahan dan pembasmian penyakit. Penanganan terhadap penyakit ayam merupakan prioritas utama dan harus mendapatkan perhatian khusus. Tujuan pengendalian penyakit adalah mengurangi terjangkitnya suatu penyakit seminimal mungkin sehingga kerugian yang ditimbulkan dapat ditekan sekecil mungkin, sedangkan tujuan pembasmian penyakit adalah menghilangkan penyakit tertentu secara tuntas sehingga sumber penyakit tersebut dapat di musnakan dan tidak menular ke ternak yang lain (Zulkarnaen, 2013).

2.6 Metode *Naive Bayes Classifier*

Naive Bayes Classifer merupakan pengklasifikasi probabilitas sederhana berdasarkan pada *teorema Bayes*. *Teorema Bayes* dikombinasikan dengan "*Naive*" yang berarti setiap atribut/variabel bersifat bebas (*independent*). *Naive Bayes Classifier* dapat dilatih dengan efisien dalam pembelajaran terawasi (*supervised learning*). Dalam prosesnya, *Naive Bayes Classifier* mengasumsikan bahwa ada atau tidaknya suatu fitir pada suatu kelas tidak berhubungan dengan ada atau tidaknya fitur lain dikelas yang sama.

Teorema ini menjadi dasar dalam pengambilan keputusan modern. Formula Bayes dinyatakan dalam persamaan 1

$$P(H|E) = \frac{P(E|H).P(H)}{P(E)} \quad \text{..... (1)}$$

dimana :

a. E = Data dengan class yang belum diketahui
b. H = Hipotesis data merupakan suatu class spesifik
c. P(H|E)=Probabalitas hipotesis berdasar kondisi*(Posterio probability)*
d. P(E|H) = Probabilitas berdasarkan kondisi pada hipotesis
e. P(H)= Probabilitas hipotesis *(Prior Probability)*

f. P(E) = Probabilitas E

Untuk menjelaskan teorema *naive bayes,* perlu diketahui bahwa proses klasifikasi memerlukan sejumlah petunjuk untuk menentukan kelas apa yang cocok bagi sampel yang dianalisis tersebut. Karena itu, teorema *bayes* di atas disesuaikan sebagai berikut

$$P(C|F_1 \ldots F_n) = \frac{P(C)P(F_1 \ldots . F_n|C)}{P(F_1 . \ldots F_n)}$$

Dimana variabel C merepresentasikan kelas, sementara variabel $F_1 \ldots F_n$ merepresentasikan karakteristik petunjuk yang dibutuhkan untuk melakukan klasifikasi. Maka rumus tersebut menjelaskan bahwa peluang masuknya sampel karakteristik tertentu dalam kelas C (*Posterior*) adalah peluang munculnya kelas C (sebelum masuknya sampel tersebut, seringkali disebut *prior*), dikali dengan kemunculannya karakteristik-karakteristik sampel pada kelas C (disebut juga *likelihood*), di bagi dengan peluang kemunculan karakteristik-karakteristik sampel secara global (disebut *evidience*). Karena itu, rumus diatas dapat ditulis secara sederhana sebagai berikut.

$$Posterior = \frac{Prior \times likelihood}{evidence}$$

Nilai *evidience* selalu tetap untuk setiap kelas pada satu sampel. Nilai dari *posterior* tersebut nantinya akan dibandingkan dengan nilai-nilai *posterior* kelas lainnya untuk menentukan ke kelas apa suatu sampel akan diklasifikasikan. Penjabaran lebih lanjut rumus *bayes* tersebut dilakukan dengan menjabarkan $(C|F_1, \ldots, F_n)$ menggunakan aturan perkalian sebagai berikut.

$$\begin{aligned} P(C|F_1, \ldots, F_n) &= P(C)\, P(F_1, .., F_n|C) \\ &= P(C)P(F_1|C)P(F_2, \ldots, F_n|C, F_1) \\ &= P(C)P(F_1|C)P(F_2|C, F_1)P(F_3, \ldots, F_n|C, F_1, F_2) \\ &= P(C)P(F_1|C)P(F_2|C, F_1)P(F_3|C, F_1, F_2), P(F_4, \ldots, F_n|C, F_1, F_2, F_3) \end{aligned}$$

$$= P(C)P(F_1|C)P(F_2|C,F_1)P(F_3|C,F_1,F_2) \dots P(F_n|C,F_1,F_2,F_3,..,F_{n-1})$$

Dapat dilihat bahwa hasil penjabaran tersebut menyebabkan semakin banyak dan semakin kompleksnya faktor-faktor syarat yang mempengaruhi nilai probabilitas, yang hampir mustahil untkl dianalisis satu persatu. Akibatnya, perhitungan tersebut menjadi sulit untuk dilakukan. Disinilah digunakan asumsi *independensi* yang sangat tinggi (*naif*), bahwa masing-masing petunjuk $(F_1, F_2 \dots F_n)$ saling bebas *(independen*) satu sama lain. Dengan asumsi tersebut, maka berlaku suatu kesamaan sebagai berikut.

$$P(P_i|F_j) = \frac{P(F_i \cap F_j)}{P(F_j)} = \frac{P(F_i)P(F_j)}{P(F_j)} = P(F_i)$$

Untuk $i \neq j$, sehingga

$$P(F_i|C,F_j) = P(F_i|C)$$

Dari persamaan diatas dapat disimpulkan bahwa asumsi *independensi naif* tersebut membuat syarat peluang menjadi sederhana, sehingga perhitungan menjadi mungkin untuk dilakuka. Selanjutkan penjabaran $P(C|F_1, \dots, F_n)$ dapat disederhanakan menjadi,

$$P(C|F_1,..,F_n) = P(C)P(F_1|C)P(F_2|C)P(F_3|C) \dots$$
$$= P(C)\prod_{i=1}^{n} P(F_i|C)$$

Persamaan diatas merupakan model dari teorema *naive bayes* yang selanjutnya akan digunakan dalam proses klsifikasi.

2.6.1 Cara Kerja Algoritma *Naive Bayes Classifier*

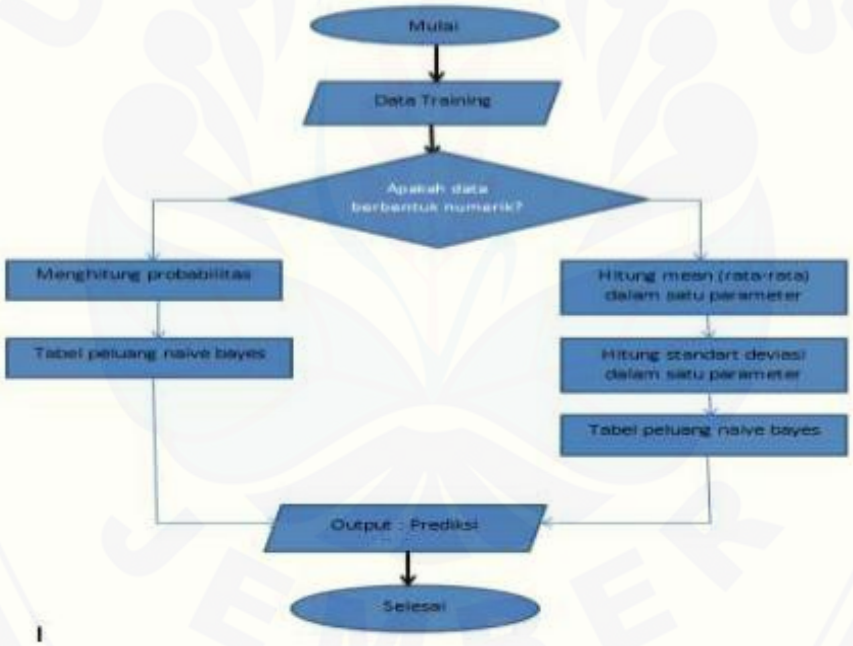
Adapun alur Algoritma Naive Bayes Classifier adalah sebagai berikut

1. Membaca data training

   Pada tahap ini sistem akan menganalisa pola data yang ada pada data yang sudah pernah ada.

2. Menghitung jumlah dan probabilitas, namun bila data numerik maka
   1. Cari nilai mean dan standar deviasi dari masing-masing parameter yang merupakan data numerik.
   2. Cari nilai probabilitas dengan cara menghitung jumlah data yang sesuai dari kategori yang sama dibagi dengan jumlah data pada kategori tersebut.
3. Mendapat nilai mean, standar deviasi dan probabilita

   Sehingga dapat digambarkan dengan skema pada Gambar 2.1



Gambar 2.1*Flowchart*Algoritma *Naive Bayes* (Bustomi, 2014)

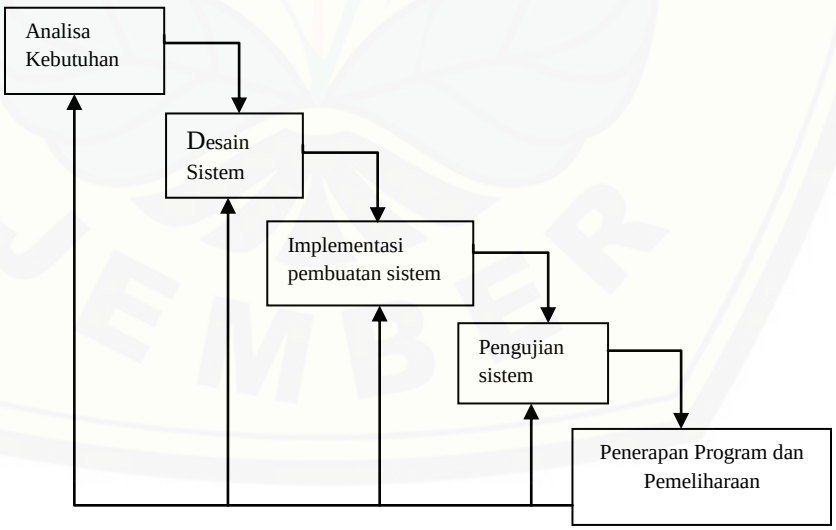
Gambar 2.1 Prose dimulai dari membaca data training untuk menemukan pola data, kemudian penentuan atribut termasuk data numerik atau tidak, jika maka dihitung jumlah dan probabilitasnya sehingga menghasilkan tabel probabilitas, jika ya (data numerik) maka hitung mean dan standar deviasi masing-masing atribut pada

setiap kelas, sehingga menghasilkan tabel mean dan standar deviasi. Pada tahap solusi masing-masing probabilitas atribut dikalikan sehingga ditemukan solusi termasuk pada kelas yang sesuai.

2.6 Model *Waterfall*

Penelitian ini akan dilakukan dalam beberapa tahap yang disesuaikan dengan metode *Software Development Life Cycle* (SDLC) waterfall yang di bagi menjadi beberapa tahapan, yaitu analisis kebutuhan, desain sistem, implementasi, pengujian dan pemeliharaan.

Pembangunan perangkat lunak pada penelitian ini adalah model *waterfall.Waterfall* merupakan salah satu model proses perangkat lunak yang mengambil kegiatan proses dasar seperti spesifikasi, pengembangan, validasi dan evolusi dengan mempresentasikannya sebagai fase-fase proses yang berbeda seperti analisis dan definisi persyaratan, perancangan perangkat lunak, implementasi dan unit testing, integrasi dan pengujian serta operasi dan pemeliharaan (Sommervile, 2011). Sebagaimana terlihat pada Gambar 2.2



Gambar 2.2 Model *Waterfall*(Darmawan, 2015)

1. Analisis Kebutuhan

   Tahap ini menganalisa kebutuhan sistem yang akan dibuat dalam bentuk yang dapat dimengerti oleh klien dan staf pengembang. Dalam tahap ini klien atau pengguna menjelaskan segala kendala dan tujuan serta mendefinisikan apa yang diinginkan dari sistem.

2. Desain Sistem

   Tahap ini pengembang akan menghasilkan sebuah arsitektur sistem secara keseluruhan dan menentukan alur perangkat lunak hingga pada tahap algoritma yang detil

3. Implementasi Pembuatan Sistem

   Tahap ini mengimplementasikan desain yang telah dibuat ke dalam kode program.

4. Pengujian Sistem

   Pengujian Sistem dilakukan untuk menemukan kesalahan-kesalahan yang mungkin terjadi, serta melakukan perbaikan untuk menyempurnakan sistem yang dibuat.

5. Pemeliharaan

   Pemeliharaan suatu software sangat diperlukan, termasuk di dalamnya adalah pengembangan, karena *software* yangdibuat tidak selamanyahanyaseperti itu, ketika dijalankan mungkin saja masih ada error kecil yang tidak ditemukan sebelumnya, atau ada penambahan fitur-fitur yang belum ada pada software tersebut. Pengembangan diperlukan ketika adanya perubahan dari eksternal perusahaan seperti ketika ada pergantian sistem operasi, atau perangkat lainnya.

2.6.1 Kelebihan Model *Waterfall*

Kelebihan model *waterfall* adalah pencerminan kepraktisan rekayasa, yang membuat kualitas *software* tetap terjaga karena pengembangannya yang terstruktur dan terawasi. Model ini merupakan jenis model yang bersifat dokumen lengkap,

sehingga proses pemeliharaan dapat dilakukan dengan mudah. Dokumentasi kode program yang lengkap secara tak langsung menghapus ketergantungan pengembang terhadap pemrogram yang keluar dari tim pengembang, hal ini menguntungkan bagi pihak pengembang karena proses pengembangan perangkat lunak tetap dapat dilanjutkan tanpa bergantung pada pemrogram tertentu.

2.6.2 Kekurangan Model *Waterfall*

Kekurangan model *waterfall* yang utama adalah lambatnya proses pengembangan perangkat lunak, karena prosesnya yang tidak bisa diloncat-loncat menjadikan model klasik ini sangat memakan waktu dalam pengembangannya. Disisi lain, pihak klien tidak dapat mencoba sistem sebelum sistem benar-benar selesai pembuatannya. Kekurangan yang lain adalah kinerja personil yang tidak optimal dan efisien karena terdapat proses menunggu suatu tahapan selesai terlebih dahulu.

BAB 3. METODOLOGI PENELITIAN

Pada bab ini menjelaskan tentang metode yang digunakan terhadap penelitian yang akan digunakan untuk mengembangkan sistem.

3.1 Jenis Penelitian

Jenis penelitian yang akan dilakukan merupakan pengembangan. Penelitian pengembangan dilakukan untuk menghasilkan suatu produk atau mengembangkan suatu produk. Produk dalam penelitian ini adalah "Penerapan Metode *Naive Bayes Classifier* Pada Sistem Pakar Untuk Mendiagnosis Penyakit Ayam *Broiler* dan Petelur". Studi kasus dan pengumpulan data gejala dan penyakit pada ayam didapat dari Dinas Peternakan.

3.2 Tempat dan Waktu Penelitian

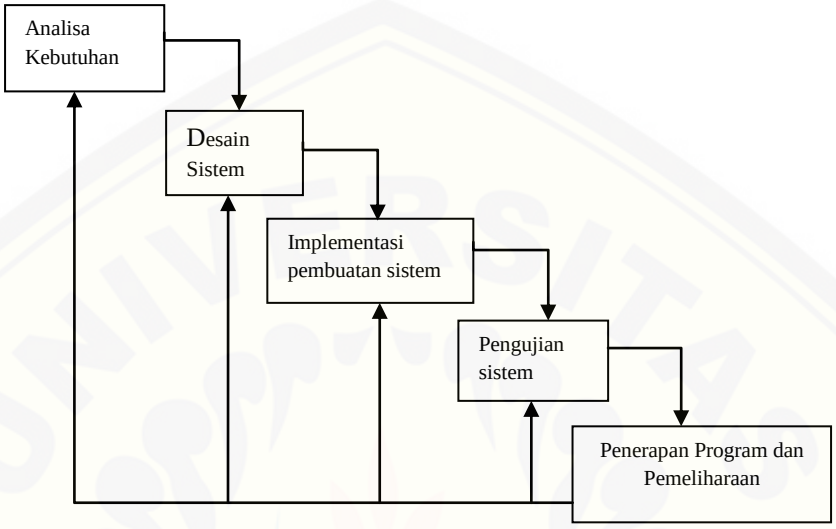
Tempat yang dilaksanakan untuk penelitian adalah Program Studi Sistem Informasi. Waktu penelitian dimulai pada bulan Juli 2016 sampai dengan selesai pengerjaan skripsi dan pengembangan sistem.

3.3 Tahapan Penelitian

penelitian dipaparkan beberapa hal yang akan dilakukan dalam beberapa tahap yang disesuaikan dengan metode Software Defelopment Life Cycle (SDLC) waterfall, yang dibagi menjadi beberapa tahap yaitu : analisis kebutuhan, desain sistem, pengkodean sistem, dan pengujian sistem.

Metode *System Development Life Cycle* (SDLC) yang digunakan dalam pembangunan perangkat lunak dalam penelitian ini adalah model *waterfall*, sering juga disebut dengan model sekuensial atau alur hidup klasik. Model ini dimulai dengan tahap analisis, desain, kode, test dan pemeliharaan sistem (Pressman, 2009). Model ini dipilih karena pembangunan sistem ini masih dalam skala kecil, sehingga

dokumentasi pengembangan dapat terorganisir dengan baik. Tahapan Model *Waterfall* dijelaskan dalam gambar 3.1 dibawah ini:



Gambar 3.1 Diagram *SDLC* Waterfall

3.3.1 Analisis Kebutuhan

Analisis kebutuhan yang diperlukan untuk menyelesaikan tujuan penelitian ini didapat dari tahapan pengumpulan data yang akan dijelaskan pada sub bab berikutnya, setelah pengumpulan data tersebut akan didapat gambaran sistem yang dapat digunakan sebagai acuan dalam pembuatan aplikasi.

Teknik Pengumpulan Data:

Pada teknik Pengumpulan data dilakukan untuk memenuhi kebutuhan data yang diperlukan untuk menyelesaikan tujuan penelitian, pengumpulan data tersebut dapat diperoleh dengan beberapa cara sebagai berikut

a. Studi literatur

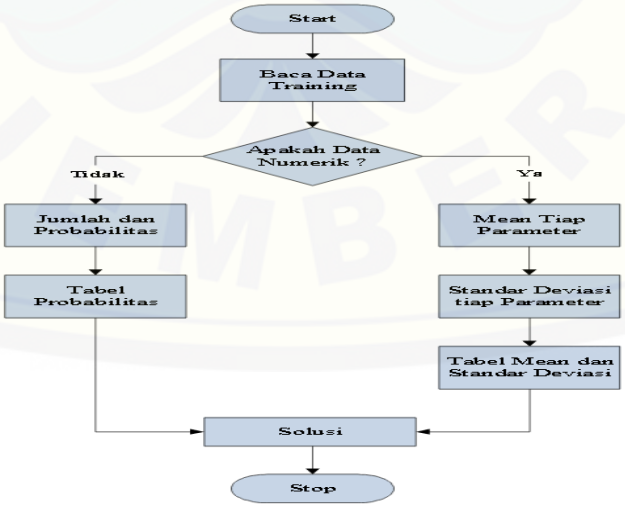
   Studiliteratur digunakan untuk mendukung teori-teori yang berkaitan dengan masalah yang akan dibahas. Studi literatur dalam penelitian ini diperoleh melalui data yang berasal dari buku, karya ilmiah, jurnal *online*, *website* dan penelitian yang dilakukan sebelumnya, dengan studi literatur tersebut peneliti

mendapat referensi untuk menyelesaikan tujuan penelitian dalam pengumpulan data yang dibutuhkan untuk membangun aplikasi.

b. Wawancara

Kegiatan ini dilakukan denganmengumpulkan data kualitatif dan kuantitatif, data kualitatif di peroleh dengan mengajukan pertanyaan langsung kepada narasumberDinas Peternakan, Perikanan dan Kelautan Kabupaten Jember dan peternakan ayam Broiler dan peternakan ras petelur H.Anas,sedangkan data kuantitatif dalam penelitian ini diperoleh dari tahap pengumpulan dan analisis data dalam bentuk angka, data ini merupakan data yang diperoleh dari hasil wawancara.

Data kualitatif yang diperoleh dari wawancara tersebut digunakan sebagai acuan dalam memilih metode yang tepat untuk mendiagnosis penyakit pada ayam *broiler* dan ras pertelur, dalam penelitian ini metode *naive bayes classifier* merupakan metode yang tepat diterapkan dalam aplikasi untuk melakukan pendiagnosisan penyakit pada ayam. Proses/alur dari proses mendiagnosis penyakit ayam menggunakan metode *Naive Bayes classifier* dapat di lihat pada Gambar 3.2



Gambar 3.2 Alur Proses *Naive Bayes*

3.3.2 Desain Aplikasi

Desain dibangun menggunakan *United Modeling language* (UML), penggunaan UML ini sudah menerapkan konsep *Object Oriented Design* yang tentunya sangat memudahkan developer untuk membangun sebuah sistem. Berikut beberapa diagram yang dibuat dalam UML :

1. *Business Process*

   Digunakan untuk menggambarkan masukan data, keluaran dari aplikasi dan tujuan dari pembuatan aplikasi, dirancang sesuai dengan analisis kebutuhan aplikasi.

2. *Usecase Diagram*

   Digunakan untuk mendeskripsikan hak akses antara aktor dengan aplikasi. Dirancang sesuai dengan hasil gambaran *bussiness process* yang telah dibuat.

3. *Scenario Diagram*
   Digunakan untuk pembuatan *interface* suatu sistem dan mempermudah pembuatan *interface*.

4. *Activity Diagram*
   Digunakan untuk menggambarkan *scenario* atau aktifitas dari aplikasi untuk dapat mengetahui alur yang dilakukan aktor serta respon aplikasi sesuai yang ditentukan.

5. *Sequence Diagram*

   Digunakan untuk menggambarkan interaksi-interaksi antar objek di dalam aplikasi. Dirancang sesuai dengan *activity diagram* aplikasi optimalisasi yang telah dibuat, agar dapat mengetahui *method* yang berjalan ketika terjadi aksi.

6. *Class Diagram*

   Digunakan untuk menggambarkan struktur dari segi pendefinisian kelas – kelas yang ada pada aplikasi sesuai dengan *sequence* yang telah dibuat

7. *Entity Relationship Diagram*

   Digunakan untuk menggambarkan *database* aplikasi yang dibangun, agar dapat mengetahui tabel *database* yang diperlukan aplikasi sesuai yang telah ditentukan.

3.3.3 Implementasi Pembuatan Aplikasi

Mengimplementasikan desain yang telah dibuat ke dalam kode program. Penulisan kode program sistem ini menggunakan bahasa pemrogaman *Hypertext Prepocessor* (PHP) dengan bantuan *Framework Code Igniter* (CI). Manajemen basisdata yang digunakan adalah *My SQL*.

Alat yang digunakan untuk proses penelitian ini meluputi *hardware* berupa satu laptop dan *software*sebagai berikut:

1. *Windows 7*
2. *DBMS MySQL*
3. *Xampp*
4. *Google Chrome*
5. *Ms. Office*
6. *Ms. Visio*
7. *Ms. Excel*
8. *Unified Modeling Language*
9. *yEd Graph Editor*

3.3.4 *Testing* dan Evaluasi

Testing dan evaluasi digunakan untuk mengetahui sejauh mana sistem ini dapat berjalan. *Testing* berfungsi untuk mengetahui apakah sistem ini dapat berfungsi dengan baik sesuai dengan yang diharapkan. Serta untuk mengetahui letak kekurangan yang ada pada sistem. Pengujian dilakukan oleh tim penguji dari pengembang sistem. Selanjutnya dilakukan evaluasi serta perbaikan terhadap kekurangan-kekurangan yang ada pada sistem ini dilakukan beberapa metode untuk pengujian yaitu diantaranya:

a. *White Box Testing*

White box testing merupakan cara pengujian dengan melihat modul yang telah dibuat dengan program-program yang ada. Pengujian ini, dilakukan oleh *developer*. Jika ada modul yang menghasilkan output yang tidak sesuai maka baris-baris program, variabel dan parameter yang terlibat pada unit tersebut satu persatu akan dicek dan diperbaiki, kemudian akan di *compile* ulang. Teknik pengujian ini menggunakan pengujian jalur dasar (*basis path testing*) dimana kompleksitas dari perangkat lunak yang dibangun akan dihitung menggunakan *Cyclomatic Complexity.*

b. Black Box Testing

Pengujian *Black Box* melibatkan pengguna/u*ser,* dimana hanya memperhatikan fungsionalitas yang berkaitan dengan masukan/keluaran (I/O) apakah sesuai dengan sistem yang dijalankan.

c. Uji Akurasi Sistem

Uji Akurasi sistem dilakukan setelah diterapkannya metode teorema *bayes* pada sistem pakar identifikasi hama dan penyakit pada tanaman kedelai bertujuan untuk mengetahui akurasi yang didapatkan dengan membandingan dengan hasil yang diporelah dengan pakar.

d. Uji Validitas dan Reliabilitas

Uji Validitas merupakan cara pengujian yang dilakukan untuk mengetahui ketepatan /kelayakan instrumen yang digunakan untuk mengukur apa yang akan di ukur. Uji validitas ini menggunakan metode *Bivariate Pearson* yang dapat dilakukan dengan menggunakan angket/kuisioner dan dapat dihitung menggunakan rumus:

$$r_{xy} = \frac{n\sum XY - (\sum X)(\sum Y)}{\sqrt{\{n\sum X^2 - (\sum X^2)^2\}\{n\sum Y^2 - (\sum Y^2)^2\}}}$$

r_{xy}= Koefisien Korelasi

N = Jumlah responden uji coba

X = Skor tiap Item

Y = Skor seluruh Item Responden

Sedangkan Uji Reliabilitas adalah suatu ukuran yang menunjukkan sejauh mana hasil pengukuran tetap konsisten bila diukur beberapa kali dengan alat ukur yang sama. bertujuan untuk mengetahui apakah alat ukur/instrument memiliki tingkat konsistensi dan dapat dipercaya. Tinggi rendahnya reliabilitas secara empirik dapat ditunjukkan dengan nilai koefisien reliabilitas mendekati angka 1. Kesepakatan secara umum reliabilitas dianggap cukup memuaskan jika ≥ 0,6. Pengujian reliabilitas instrumen dengan menggunakan rumus *Alpha Cronbach* sebagai berikut:

$$r = \left(\frac{n}{n-1}\right)\left(1 - \frac{\sum \sigma_t^2}{\sigma_t^2}\right)$$

r = reliabilitas yang dicari

n = jumlah item pertanyaan yang diuji

$\sum \boldsymbol{\sigma_t^2}$ = jumlah varians skor tiap-tiap item

$\boldsymbol{\sigma_t^2}$ = varians total

3.3.5 Pemeliharaan

Perangkat lunak yang sudah selesai akan mengalami perubahan. Perubahan biasanya berupa *error* sehingga diperlukan perbaikan dan pemeliharaan kepada sistem. Perubahan ini dilakukan agar sistem bersifat dinamis.

BAB 4. DESAIN DAN IMPLEMENTASI

Bab ini menguraikan tentang analisis kebutuhan, desain sistem, implementasi, dan pengujian sistem yang digunakan dalam penerapan metode *naive bayes classifier* untuk mendiagnosis penyakit ayam *broiler* dan petelur. Tahap analisis hingga tahap pengujian menggunakan metode pengembangan *waterfall*.

4.1 Analisis Kebutuhan

Analisis kebutuhan dalam penelitian ini di lakukan dengan cara mengidentifikasi masalah yang ada, dengan mengidentifikasi permasalahan maka analisis kebutuhan dapat ditentukan dalam bentuk kebutuhan fungsional dan non fungsional

Kebutuhan fungsional aplikasi pada penelitian ini adalah sebagai berikut :

1. Aplikasi dapat menyimpan dan melakukan tambah, edit, dan simpan data penyakit
2. Aplikasi dapat melakukan tambah,simpan dan edit data gejala
3. Aplikasi dapat melakukan diagnosa penyakit pada ayam
4. Aplikasi dapat menambahkan dataset dari hasil diagnosa yang dilakukan oleh user
5. Aplikasi menampilkan detail perhitungan dari diagnosa menggunakan *naive bayes classifier*

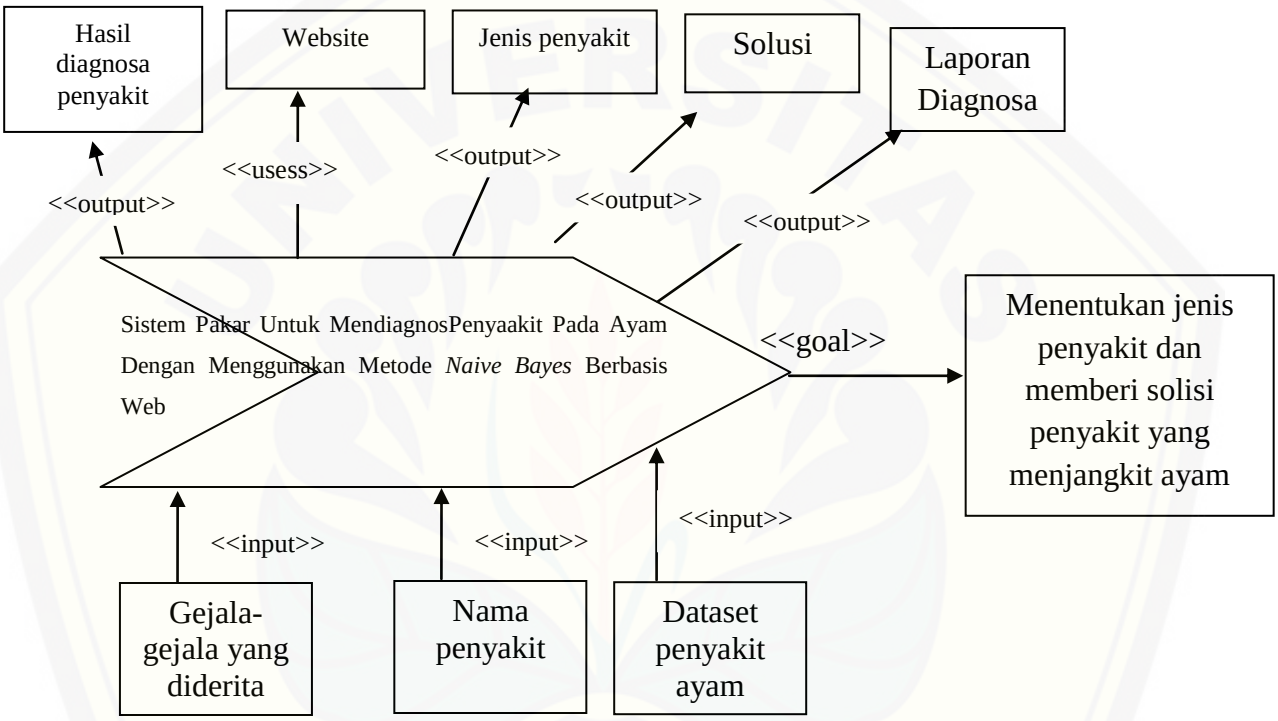
Kebutuhan non-fungsional aplikasi pada penelitian ini adalah sebagai berikut:

1. Sistem dapat diakses 24 jam
2. Sistem menggunakan tampilan yang user friendly, sehingga pengguna tidak kesulitan untuk mengoperasikannya.

4.2 *Bussiness Process*

Bussiness process merupakan tahapan yang digunakan untuk memodelkan proses dilakukan sistem untuk mencapai hasil yang dibutuhkan pengguna. Pada

business process terdapat beberapa komponen meliputi *input*, media yang digunakan (*uses*), keluaran (*output*) dan tujuan (*goal*) yang akan dicapai. Pada Gambar 4.1 dijelaskan mengenai komponen *input*, *output*, *uses* dan *goal* sistem yang akan dibangun oleh peneliti agar jelas apa saja data yang dibutuhkan untuk masukan atau input, hasil dan tujuan pembuatan sistem.



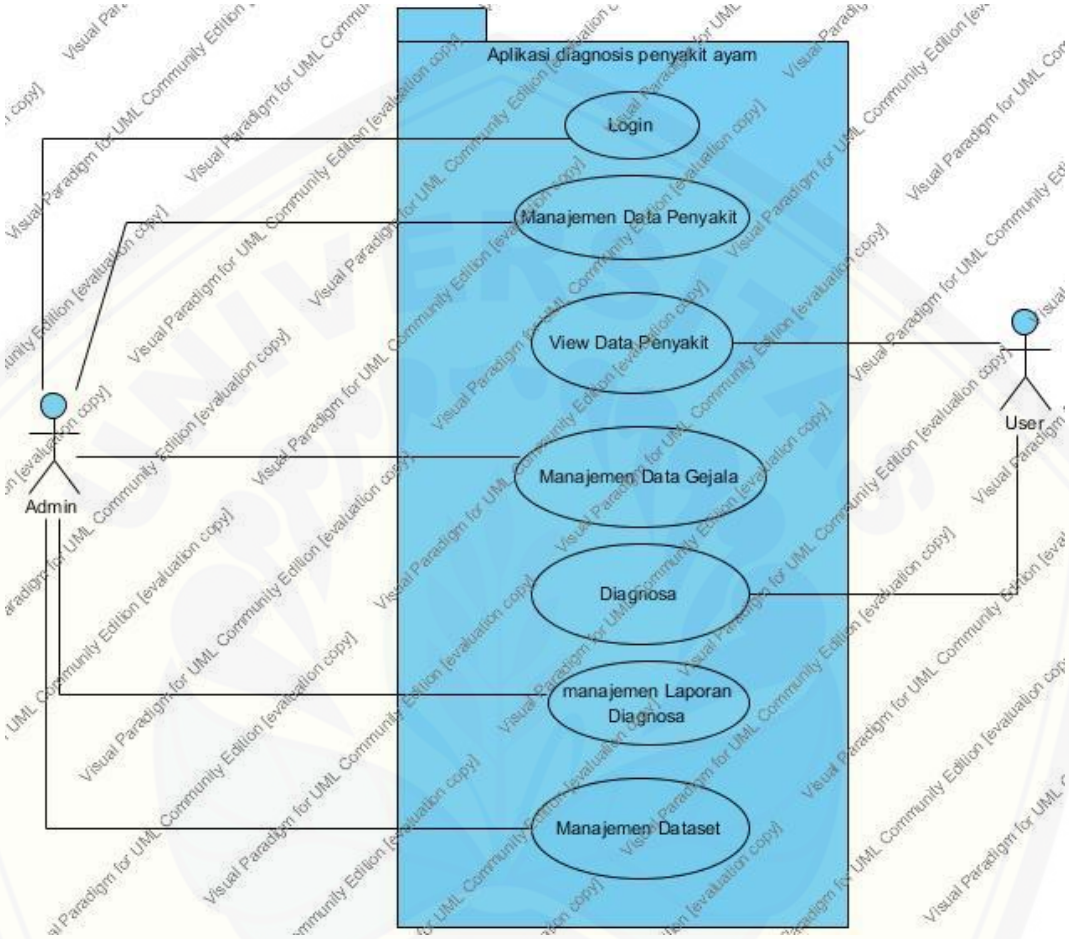
Gambar 4.1*Bussiness Process*

4.3 *Usecase* Diagram

Use case diagram merupakan model atau diagram yang menggambarkan fungsi atau tugas yang dilakukan *user*, baik manusia maupun mesin/komputer. *Use case* digambarkan dari beberapa aktor, *use-case*, dan interaksi diantara komponen tersebut yang dapat memberikan informasi dari sistem yang akan dibangun. Fitur-fitur pada sistem ini terdapat 7 fitur yang digambarkan dengan *elips* dan terdapat 2 tipe *user*.

Usecase berguna untuk menggambarkan dialog antara aktor dengan aplikasi serta menentukan fitur apa saja yang dapat di akses oleh masing-masing aktor.

*Usecase*aplikasi sistem pakaruntuk mendiagnosa penyakit ayam dapat dilihat pada Gambar 4.2



Gambar 4.2*Usecase Diagram* Sistem Pakar Diagnosa Penyakit Ayam

Pada *usecase diagram* tersebut terdapat dua aktor yang dapat mengakses aplikasi diagnosa penyakit ayam. Dua aktor tersebut adalah Pengelola sistem (sebagai admin) dan peternak/pengguna sistem (sebagai user). Deskripsi aktor serta *usecase* akan dijelaskan pada tabel 4.1 dan 4.2

Tabel 4. 1 Definisi Aktor Use Case

No.	Aktor	Definisi Tugas
1.	Admin/dr. Hewan	Admin dapat melakukan : a. Login b. Melihat dan melakukan manajemen data penyakit c. Melihat dan melakukan manajemen data gejala d. Melakukan manajemen dataset e. Mencetak laporan diagnosis
2.	User	User dapat melakukan a. Melihatdata Penyakit yang bisa didiagnosa oleh aplikasi b. Melakukan diagnosa penyakit ayam

Tabel 4. 2 Definisi *Use Case*

No.	Usecase	Deskripsi
1.	Login	Login merupakan langkah awal untuk aktor dapat masuk ke halaman aplikasi
2.	*View*Data Penyakit	Fitur ini berisi data – data atau nama – nama dari penyakit ayam, aktor dapat melakukan aksi *view*/melihat list tabel data penyakit.
3.	Manajemen Data Penyakit	Penyakit merupakan Fitur yang berisi data – data atau nama – nama penyakit ayam. Aktor dapat menambah,menyimpan,

		mengubah serta menghapus data penyakit.
4.	Manajemen Data Gejala	Gejala merupakan Fitur yang berisi data – data gejala dari penyakit ayam. Aktor dapat menambah,menyimpan, mengubah serta menghapus data gejala.
5.	Diagnosa	Diagnosis merupakan fitur yang berisi pertanyaan mengenai gejala yang di derita ayam tersebut. Aktor dapat melakukan aksi *view/*melihat hasil dari diagnosis tersebut.
4	Manajemen Lapora Diagnosa	Manajemen laporan diagnosa merupakann hasil dari diagnosis yang dilakukan oleh user dan hanya bisa diakses oleh admin atau pakar.
5	Dataset	Dataset merupakan Fitur yang berisi data – data atau nama – nama penyakit ayam yang sedah ada sebelumnya. Aktor dapat menambah,menyimpan, mengubah serta menghapus data dataset.

4.4 *Scenario*

Scenario berguna untuk menjabarkan alur kerja tiap step-step *usecase* yang sudah di tentukan. *Scenariousecase* ini terdiri dari nama *usecase*, aktor, pra-kondisi, *pasca*-kondisi, skenario utama dan skenario alternatif.

Scenario sistem dibangun berdasarkan masing-masing *usecase* diagram dan berdasarkan kebutuhan fungsional yang dibutuhkan oleh sistem. Berikut adalah penggunaan skenario sistem yang di gunakan oleh *user*dan admindengan interksi ke

sistem. Untuk lebih jelasnya skenario sistem bisa dilihat pada tabel 4.3 skenario berikut :

4.4.1 *Scenario Login*

Scenario login menjelaskan alur proses *login* dari admin dan pakar. *Scenariologin* dapat dilihat pada table 4.3

Tabel 4. 3 Scenario Login

Nama *Use case*	Login
Aktor	Admin atau Pakar
Pre-Kondisi	Admin atau Pakar harus mempunyai*username* dan *password*untuk masuk kedalamsistem dan mengakses menu halaman Admin.
Post-Kondisi	Admin dan Pakar berhasil login dan masuk ke dalam system.
SKENARIO UTAMA LOGIN	
admin atau pakar	Sistem
1. Membuka Web Sistem Pakar Untuk Mendiagnosa Penyakit Ayam *Broiler* dan Petelur.	
	2. Menampilkan halaman utama system yang berisi nemu : - Home - Penyakit - Diagnosa - Login
3. Memilih menu login.	
	4. Menampilkan pop up login yang berisi(*username* dan *password*)
5. Mengisi *username* dan *password*	
6. Klik tombol*Submit*	
	7. Menampilkan halaman home admin yang berisi menu : - Home - Penyakit - Gejala

	- Dataset - Laporan Diagnosa - Logout
SKENARIO ALTERNATIVE	
Jika aktor mengisi *pop up login* secara tidak lengkap	
admin atau pakar	Sistem
5a.*Username*atau*password* (kosong)	
6a. Klik tombol *submit*	
	7a. Memeriksa kelengkapan masukan *username* dan *password* pada sistem.
	8a. Menampilkan Halaman Home yang berisi menu: - Home - Penyakit - Diagnosa - Login
SKENARIO ALTERNATIVE	
Jika aktor mengisi *pop up login* salah	
admin atau pakar	Sistem
5b. Mengisi u*sername* dan *password* (salah)	
6b. Klik tombol *submit*	
	7b. Memeriksa ketepatan masukan *username* dan *password* pada sistem.
	8b. Menampilkan Halaman Home yang berisi menu: - Home - Penyakit - Diagnosa - Login

4.4.2 *ScenarioView* Data Penyakit

Scenarioview data penyakit menjelaskan alur proses *view*penyakit yang dapat diakses oleh peternak. *Scenarioview* penyakit dapat dilihat pada table 4.4.

Tabel 4. 4 Scenario Melihat Data Penyakit

Nama Usecase	*View*Data Penyakit.
Aktor	Peternak
Pre-Kondisi	- Sistem aktif. - Data Penyakitayam belum di lihat.
Post-Kondisi	Data Penyakit Ayam berhasil di lihat.
SKENARIO UTAMA *VIEW* DATA PENYAKIT AYAM *BROILER* DAN PETELUR	
Peternak	Sistem
1. memilih menu penyakit	
	2. mengambil data penyakit dari database pada tabel penyakit(IdPenyakit, KdPenyakit,Penyakit, Deskripsi, Foto)
	3. menampilkan data penyakit (No, Kode Penyakit, Penyakit)

4.4.3 *Scenario* Manajemen Data Penyakit

Scenario manajemen data penyakit menjelaskan alur proses mengelola data penyakit yang dapat diakses oleh adminatau pakar. *Scenario*manajemen data penyakit dapat dilihat pada table 4.5.

Tabel 4. 5 Scenario Manajemen Data Penyakit

Nama Usecase	Manajemen Data Penyakit.
Aktor	Admin atau Pakar
Pre-Kondisi	- Sistem aktif (admin atau pakar telah melakukan *login*). - Data Penyakit belum di tambahkan, di lihat, diperbarui.
Post-Kondisi	Data Penyakit berhasil di tambahkan, di lihat, diperbarui.
SKENARIO UTAMA MANAJEMEN PENYAKIT	
Admin atau Pakar	Sistem
1. Memilih menu penyakit	
	2. mengambil data penyakit dari datbase tabel penyakit (IdPenyakit, KdPenyakit, Penyakit, Deskripsi, Foto)
	3. menampilkan data penyakit pada halaman View penyakit (No, Kode Penyakit, Penyakit, Action)

SKENARIO UTAMA MENAMBAH DATA PENYAKIT AYAM	
Aksi Aktor	Reaksi Sistem
4. Klik tombol “Tambah Penyakit”.	
	5. Menampilkan form input penyakit pada halaman input penyakit (Kode Penyakit, Penyakit, Deskripsi, Foto Penyakit)
	6. Menampilkan form input penyakit pada halaman input penyakit (Kode Penyakit, Penyakit, Deskripsi, Foto Penyakit)
7. mengisi data input penyakit (Penyakit, Deskripsi, Foto)	
8. Menekan tombol “Submit”	
	9. menyimpan data penyakit pada database tabel penyakit
	10. menampilkan data penyakit pada halaman View penyakit (No, Kode Penyakit, Penyakit, Action)
SKENARIO UTAMA TAMBAH DATA, KEMBALI KE HALAMAN PENYAKIT	
Aksi Aktor	Reaksi Sistem
7. Menekan tombol “Back”	
	8. menampilkan data penyakit pada halaman View penyakit (No, Kode Penyakit, Penyakit, Action)
SKENARIO UTAMA MENGUBAH DATA PENYAKIT	
Aksi Aktor	Reaksi Sistem
4. Klik tombol “*edit*” pada baris data yang akan dirubah.	
	5. mengambil data penyakit ayam berdasarkan id yang dipilih
	6. menampilkan data penyakit ayam pada halaman view penyakit ayam berdasarkan id yang dipilih (Kode Penyakit, Penyakit, Deskripsi, Foto Penyakit)
7. merubah data penyakit ayam (Penyakit, Deskripsi, Foto)	
8. Menekan tombol “Submit”	
	9. Menyimpan perubahan data penyakit pada database tabel penyakit (Penyakit, Deskripsi, Foto)
	10. menampilkan data penyakit pada halaman View penyakit (No, Kode Penyakit, Penyakit, Action)
SKENARIO UTAMA MENGUBAH, KEMBALI KE HALAMAN PENYAKIT	
Aksi Aktor	Reaksi Sistem

8. Menekan tombol “Back”	
	9. Menampilkan data penyakit pada halaman View penyakit (No, Kode Penyakit, Penyakit, Action)
	9. menampilkan data penyakit pada halaman View penyakit (No, Kode Penyakit, Penyakit, Action)
SKENARIO ALTERNATIVE MENAMBAH PENYAKIT	
Jika aktor mengisi *form* tambah data Penyakit kosong	
Aksi Aktor	Reaksi Sistem
6a.Mengisi data input penyakit (Penyakit, Deskripsi, Foto)	
7a.Menekan tombol “submit”	
	8a. Menampilkan popup pesan "please fill out this field" pada field yang kosong
SKENARIO ALTERNATIVE MERUBAH PENYAKIT	
Jika aktor merubah *formedit* data Penyakit gagal input kosong	
Aksi Aktor	Reaksi Sistem
6b. Mengubah isi *formedit* data Penyakit(nama penyakit, deskripsi, foto penyakit)	
7b. Menekan tombol “Submit”	
	8b. Memeriksa ubah data Penyakit
	9b. menampilkan popup pesan "please fill out this field" pada field yang kosong

4.4.4 *Scenario* Manajemen Data Gejala

Scenario manajemen data gejala menjelaskan alur proses mengelola data gejala yang dapat diakses oleh admin atau pakar. *Scenario* manajemen data gejala dapat dilihat pada table 4.6.

Tabel 4. 6 Scenario Manajemen Data Penyakit

Nama Usecase	Manajemen Data Gejala
Aktor	Admin atau Pakar
Pre-Kondisi	- Sistem aktif (admin atau pakar telah melakukan *login*). - Data gejala belum di tambahkan, diperbarui.
Post-Kondisi	Data Gejala berhasil di tambahkan, diperbarui.
SKENARIO UTAMA MENAMBAH DATA GEJALA	
Admin atau Pakar	Sistem
1. Memilih menu gejala	
	2. mengambil data gejala dari datbase tabel gejala

	(IdGejala, KdGejala, Gejala)
	3. menampilkan data gejala pada halaman View gejala (No, Kode Gejala, Gejala, Action)
SKENARIO UTAMA MENAMBAH DATA GEJALA	
Aksi Aktor	Reaksi Sistem
11. Klik tombol "Tambah Gejala".	
	12. Menampilkan form input Gejala pada halaman input Gejala(Kode Gejala, Gejala)
13. mengisi data input Gejala (Kode Gejala, Gejala)	
14. Menekan tombol "Submit"	
	15. menyimpan data Gejala pada database tabel Gejala
	16. menampilkan data gejala pada halaman View gejala (No, Kode Gejala, Gejala, Action)
SKENARIO UTAMA TAMBAH DATA, KEMBALI KE HALAMAN GEJALA	
Aksi Aktor	Reaksi Sistem
10. Menekan tombol "Back"	
	17. menampilkan data gejala pada halaman View gejala (No, Kode Gejala, Gejala, Action)
SKENARIO UTAMA MENGUBAH DATA GEJALA	
Aksi Aktor	Reaksi Sistem
11. Klik tombol "edit" pada baris yang akan dirubah	
	12. mengambil data Gejala ayam berdasarkan id yang dipilih
	13. menampilkan data Gejala ayam pada halaman view Gejala ayam berdasarkan id yang dipilih (Kode Gejala, Gejala)
14. merubah data Gejala ayam (Gejala)	
15. Menekan tombol "Submit"	
	16. Menyimpan perubahan data gejala pada database tabel gejala (gejala)
	17. menampilkan data gejala pada halaman View gejala (No, Kode Gejala, Gejala, Action)
SKENARIO UTAMA MENGUBAH, KEMBALI KE HALAMAN GEJALA	
Aksi Aktor	Reaksi Sistem

18. Menekan tombol “Back”	
	19. menampilkan data gejala pada halaman View gejala (No, Kode Gejala, Gejala, Action)
SKENARIO ALTERNATIVE MENAMBAH GEJALA	
Jika aktor mengisi *form* tambah data Gejalakosong	
Aksi Aktor	Reaksi Sistem
6a. mengisi data input Gejala (Kode Gejala, Gejala)	
7a.Menekan tombol “submit”	
	8a. Memeriksa masukan data gejala
	9a. menampilkan popup pesan "please fill out this field" pada field yang kosong
SKENARIO ALTERNATIVE MERUBAH GEJALA	
Jika aktor merubah *formedit* data Gejala gagal input kosong	
Aksi Aktor	Reaksi Sistem
6b. mengisi data input Gejala (Kode Gejala, Gejala)	
7b. Menekan tombol “Submit”	
	8b. Memeriksa ubah data gejala
	9b. Menampilkan pesan dengan menunjuk lokasi isian yang belum diisi pada form *edit* data gejalamenampilkan popup pesan "please fill out this field" pada field yang kosong.
	10b. Menampilkan data gejala pada halaman View gejala (No, Kode Gejala, Gejala, Action)

4.4.5 *Scenario* Diagnosa

Scenario diagnosa penyakit ayam menjelaskan alur proses diagnosa penyakit ayam yang dapat diakses oleh user. *Scenario* diagnosa penyakit ayam dapat dilihat pada Tabel 4.7.

Tabel 4. 7 Scenario diagnosa penyakit ayam

Nama Usecase	Diagnosa.
Aktor	Peternak ayam
Pre-Kondisi	- Sistem aktif . - Data Diagnosa penyakit ayam belum di tambahkan.
Post-Kondisi	Data Diagnosa penyakit ayam berhasil di tambahkan.
SKENARIO UTAMA DIAGNOSA PENYAKIT AYAM	

Petani	Sistem
1. Memilih menu diagnosa.	
	2. Mengambil data dari database pada tabel gejala (IdGejala, KdGejala, Gejala)
	3. Menampilkan data gejala pada view diagnosa penyakit ayam (No.Diagnosa,Tanggal, Gejala)
4. Memilih gejala penyakit yang menyerang ayam	
5. Klik "Diagnosa"	
	6. menampilkan halaman hasil diagnosa (kode diagnosa, tanggal diagnosa, penyakit ayam yang diderita)
SKENARIO UTAMA DETAIL DIAGNOSA	
Aksi Aktor	Reaksi Sistem
7. Klik tombol "Detail"	
	8. menampilkan halaman detail hasil diagnosa (kode diagnosa, tangal diagnosa, gejala yang dipilih, hasil perhitungan naive bayes, Probabilitas tinggi)

4.4.6 *Scenario* Manajemen Dataset

Scenario manajemen dataset menjelaskan alur proses manajemen dataset yang dapat diakses oleh admin atau pakar. *Scenario* manajemendataset dapat dilihat pada table 4.8.

Tabel 4. 8 Scenario Manajemen Dataset

Nama Usecase	Manajemen dataset.
Aktor	Admin atau Pakar
Pre-Kondisi	- Sistem aktif (admin atau admin telah melakukan *login*). - Data dataset belum di tambahkan, di lihat, diperbarui, atau di hapus.
Post-Kondisi	Data datasetberhasil di tambahkan, di lihat, diperbarui, atau di hapus.
SKENARIO UTAMA MELIHAT DATASET	
Admin	Sistem
1. memilih menu dataset	
	2. mengambil data dataset dari datbase tabel dataset (IdDataset, NoDIagnosa, IdPenyakit)
	3. menampilkan data dataset pada halaman View dataset (No.Dtaset, Gejala, Penyakit, Action)
SKENARIO UTAMA EDIT DATA DATASET	

Aksi Aktor	Reaksi Sistem
4. Klik tombol "edit"	
	5. mengambil data dataset berdasarkan id yang dipilih
	6. menampilkan data dataset pada halaman view dataset berdasarkan id yang dipilih (No.Dataset, Penyakit, Gejala)
7. merubah data dataset (Penyakit, Gejala)	
8. klik submit	
	9. Menyimpan perubahan data dataset pada database tabel dataset (IdDataset, Penyakti, Gejala)
	10. menampilkan data dataset pada halaman View dataset (No.Dtaset, Gejala, Penyakit, Action)
SKENARIO UTAMA MENGUBAH DATASET BATAL	
Aksi Aktor	Reaksi Sistem
11. klik batal	
	12. menampilkan data dataset pada halaman View dataset (No.Dtaset, Gejala, Penyakit, Action)
SKENARIO UTAMA DELETE DATASET	
Aksi Aktor	Reaksi Sistem
13. Klik tombol "delete"	
	14. Menampilkan popup "apakan anda yakin akan menghapus data?"
15. klik "OK"	
	16. menghapus data dataset pada database tabel dataset
	17. menampilkan data dataset pada halaman View dataset (No.Dtaset, Gejala, Penyakit, Action)
SKENARIO ALTERNATIF EDIT (GAGAL INPUT KOSONG)	
Aksi Aktor	Reaksi Sistem
8a. klik submit	
	9a. Menampilkan popup pesan "please fill out this field" pada field yang kosong
	10a. menampilkan data dataset pada halaman View dataset (No.Dtaset, Gejala, Penyakit, Action)
SKENARIO ALTERNATIF DELETE (BATAL)	
Aksi Aktor	Reaksi Sistem
13a. Klik tombol "delete"	
	14a. Menampilkan popup "apakan anda yakin akan menghapus data?"
15a. Klik Batal	
	16a. menampilkan data dataset pada halaman View dataset (No.Dtaset, Gejala, Penyakit, Action)

4.4.7 *Scenario*Laporan Diagnosa

*Scenario*laporan diagnosa menjelaskan alur proses laporan diagnosa yang dapat diakses oleh admin atau pakar. *Scenario*laporan diagnosa dapat dilihat pada Table 4.9.

Tabel 4. 9 Scenario Laporan Diagnosa

Nama Usecase	Laporan Diagnosa
Aktor	Admin atau pakar
Pre-Kondisi	- Sistem aktif (admin atau pakar telah melakukan *login*).
Post-Kondisi	Data hasil diagnosa sudah masuk di dalam fitur laporan diagnosa dan berhasil di tambahkan ke dataset.
SKENARIO UTAMA LAPORAN DIAGNOSA	
Admin	Sistem
1. Memilih menu Laporan Diagnosa	
	2. Mengambil data dari *database.*
	3. Menampilkan Laporan Diagnosa (Tanggal diagnosa, No.Registrasi, Hasil Diagnosa, Tombol Simpan Dataset, Tombol Report Detail).
SKENARIO UTAMA SIMPAN DATASET	
Aksi Aktor	Reaksi Sistem
4. Klik tombol “Simpan Dataset”.	
	5. Menyimpan dataset ke menu dataset dan Menampilkan tabeldataset yang berisi - Kolom search - Tambah dataset - No dataset - Gejala
	- Penyakit - Tombol edit - Tombol delete
SKENARIO UTAMA REPORT DETAIL	
Aksi Aktor	Reaksi Sistem
7. Menekan tombol “Report Detail”	
	8. Menampilkan detail hasil diagnosa yang berisi penjelasan penyakit yang terdiagnosa tombol detail dan tombol back

SKENARIO UTAMA DETAIL	
Aksi Aktor	Reaksi Sistem
9. Klik tombol "*Detail"*	
	10. Menampilkan pop up detail diagnosa yang berisi - Kode diagnosa - Tanggal - Gejala yang dipilih - Hasil perhitungan naive bayes - Probabilitas tertinggi - Tombol close
SKENARIO UTAMA	
Aksi Aktor	Reaksi Sistem
11. Menekan tombol "Report Detail"	
	12. Menampilkan detail hasil diagnosa yang berisi penjelasan penyakit yang terdiagnosa tombol detail dan tombol back
13. Menekan tombol "Back"	
	14. Menampilkan Laporan Diagnosa (Tanggal diagnosa, No.Registrasi, Hasil Diagnosa, Tombol Simpan Dataset, Tombol Report Detail).

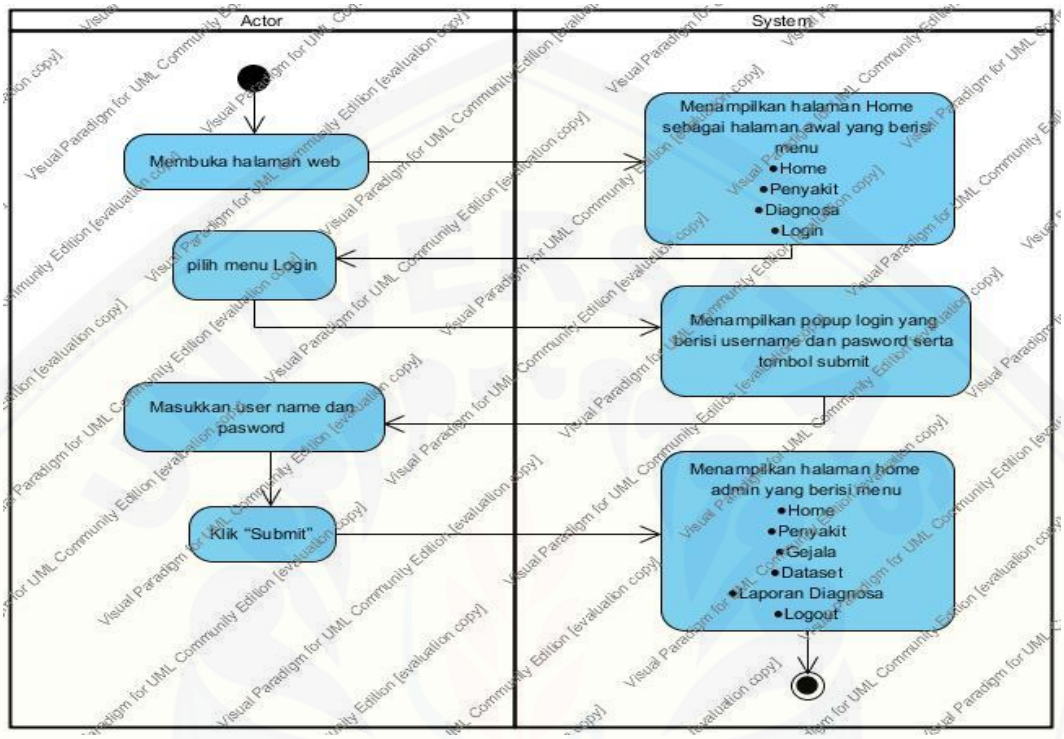
4.5 *ActivityDiagram*

Activity diagram merupakan gambaran tentang alur aktifitas dalam aplikasi yang akan dibangun, bagaimana masing-masing alur berasal, *decision* yang mungkin terjadi, serta bagaimana alur berakhir. Pembuatan *activity diagram*ini mengacu pada *usecase* dan *scenario* yang telah dibuat sebelumnya. Berikut adalah *activity diagram* dari aplikasi yang akan dibangun

4.5.1 *Activity Diagram Login*

Activity Diagram login menggambarkan alur aktivitas dari proses login agar pengguna dapat mengakses fitur sistem. Aktor yang melakukan login adalah admin, atau pakar. Setiap pengguna harus memiliki *user name* dan *password* yang terdaftar pada sistem kecuali peternak. Hanya pengguna berstatus aktif yang memiliki hak

akses terhadap sistem. Jika berhasil *login*, maka aktor akan memasuki tampilan menu utama. *Activity diagram login* dapat dilihat pada Gambar 4.3.



Gambar 4.3 *Activity Diagram Login* Admin

Activity diagram manajemen data penyakit berikut menjelaskan alur kerja antara admin dan sistem kedalam bentuk diagram saat manajemen data penyakit. Manajemen data penyakit ini meliputi *tambah penyakit*, dan*edit*pada database. Alur kerja dimulai dari membuka *web server* dan diakhiri dengan pembaruan database. *Activity Diagram* manajemen data penyakit dapat dilihat pada Lampiran B1.

4.5.2 *Activity Diagram* Manajemen Gejala

Activity diagram manajemen data gejala menggambarkan alur aktifitas aktor untuk melakukan tambah, dan edit gejala. Aktor yang dapat mengakses fitur ini adalah admin. *Activity Diagram* manajemen data gejala dapat dilihat pada Lampiran B2.

4.5.3 *Activity Diagram* Diagnosa

Activity diagram diagnosamenggambarkan alur aktifitas untuk melakukan diagnosa dan akan menghasilkan hasil diagnosa. Aktor yang dapat mengakses fitur ini adalah user. *Activity Diagram* hasil diagnosa dapat dilihat pada Lampiran B3.

4.5.4 *Activity Diagram* View Penyakit

Activity diagram View penyakitmenggambarkan alur aktifitas untuk melihat list penyakit yang sudah diinputkan. Aktor yang dapat mengakses fitur ini adalah user. *Activity Diagram* View Penyakit dapat dilihat pada Lampiran B4.

4.5.5 *Activity Diagram Manajemen* Laporan Diagnosa

Activity diagram manajemen laporan diagnosa berikut menjelaskan alur kerja antara admin dan sistem kedalam bentuk diagram saat manajemen laporan diagnosa. Manajemen laporan diagnosameliputi view hasil diagnosa,tambah dataset, dan *detail* perhitungan diagnosapada database. Alur kerja dimulai dari membuka *web server* dan diakhiri dengan pembaruan database. *Activity Diagram* manajemen dataset dapat dilihat pada Lampiran B5.

4.5.6 *Activity Diagram Manajemen* Dataset

Activity diagrammanajemen dataset berikut menjelaskan alur kerja antara admin dan sistem kedalam bentuk diagram saat manajemen dataset. Manajemen dataset meliputi edit, dan *delete*pada database. Alur kerja dimulai dari membuka *web server* dan diakhiri dengan pembaruan database. *Activity Diagram* manajemen dataset dapat dilihat pada Lampiran B7.

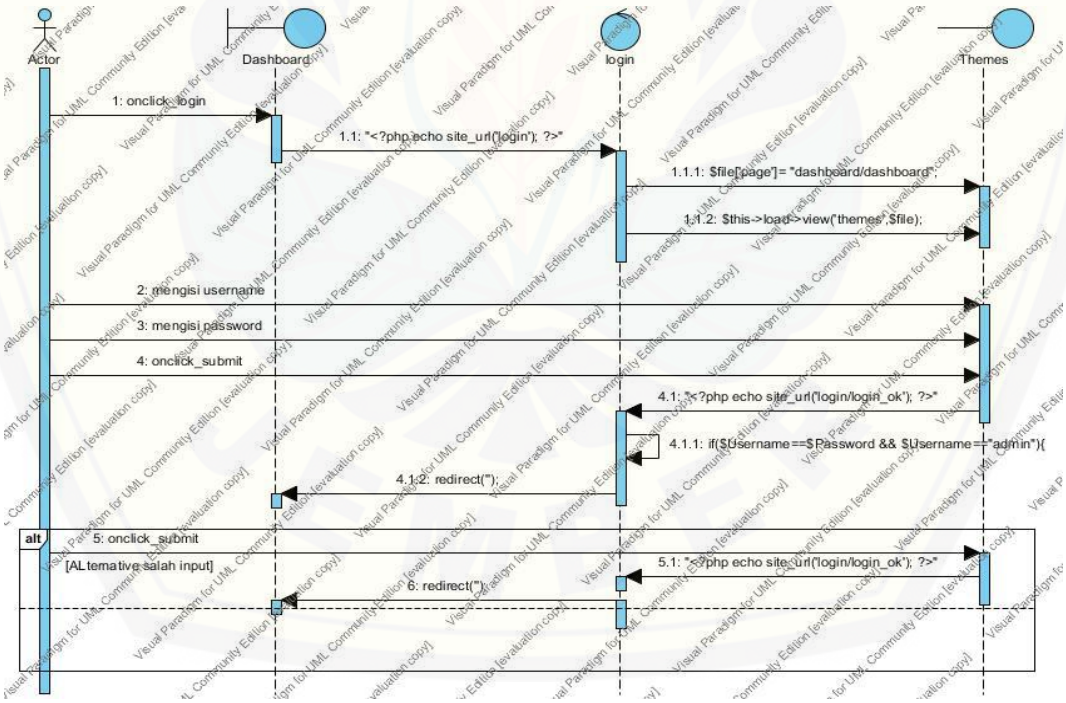
4.6 Squence Diagram

Sequence Diagram adalah dokumentasi suatu diagram terurut yang menampilkan interaksi - interaksi antar objek di dalam sistem. *Sequence* diagram digunakan untuk menggambarkan skenario dan memodelkan aliran logika dalam sistem dengan cara

visual. *Sequence diagram* pada Sistem Pakar Untuk Mendiagnosis Penyakit Ayam Broiler dan Petelur ini digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai sebuah respon dari suatu kejadian/even untuk menghasilkan *output* dari sistem.

4.6.1 *Squence Diagram Login* admin

Sequence Diagram login admin dapat dilihat melalui alur perjalanan suatu fitur login aktor (admin) dengan pertama melakukan login melalui klik menu login yang ada pada index kemudian controller login akan melanjutkan ke model m_query untuk melakukan pengecekan apakah username dan password dari aktor sudah benar atau tidak, jika benar maka akan menuju ke menu halaman home milik admin, jika tidak maka akan kembali ke halaman utama.Gambaran alur *Sequence Diagram login* admin terdapat pada Gambar 4.4.



Gambar 4.4 *Squence Diagram Login* Admin

4.6.2 *Squence* Diagram Manajemen penyakit(Admin)

Admin harus melakukan login terlebih dahulu, setelah itu admin bisa melakukan *view*, *input*, *edit*data penyakit ayam. Setelah melakukan aksi berupa *input* atau *edit*, data tersebut tersimpan dalam database. Gambaran alur *Sequence Diagram* mengelola data penyakit ayam terdapat pada Lampiran C.1 (Sequence Diagram).

4.6.3 *Squence Diagram* Manajemen Gejala (Admin)

Admin harus melakukan login terlebih dahulu, setelah itu admin bisa melakukan *view,input*, *edit*data gejala . Setelah melakukan aksi berupa *input* atau *edit*, data tersebut tersimpan dalam database. Gambaran alur *Sequence Diagram* mengelola data gejala terdapat pada Lampiran C.2 (Sequence Diagram).

4.6.4 *Squence Diagram* Diagnosa

User tidak harus melakukan login terlebih dahulu, user bisa langsung melakukan aksi untuk mendiagnosa penyakit ayam. Untukmelakukan diagnosa tersebut user hanya perlu memilih gejala yang sudah tersedia di sistem sesuai dengan gejala yang terjadi pada ternaknya. Setelah melakukan diagnosa data akan tersimpan di dalam database, dan hasil diagnosa akan ditampilkan pada view hasildiagnosa. Gambaran alur *Sequence Diagram* diagnosa penyakit ayam terdapat pada Lampiran C.3(*Sequence* Diagram).

4.6.5 *Squence Diagram* View Penyakit

Sequence diagram lihat data penyakitpada ayam ini menggambarkan skenario yang dilakukan sebagai aksi untuk melihat data penyakit ayam. Aktor yang dapat melihat data penyakit ayam pada *Sequence Diagram* ini adalah user. *Sequence Diagram* lihat data penyakitayam terdapat pada Lampiran C.4 (Sequence Diagram).

4.6.6 *Squence* Diagram Manajemen Dataset

Sequence diagram Manajemen Dataset menggambarkan alur logika untuk mengelola dataset dan dapat mengedit dan menghapus dataset yang sudah ada pada sistem,aktor yang mengakses fitur ini adalah admin. Gambar alur *sequence* diagram manajemen dataset terdapat pada Lampiran C.5 (*Sequence* Diagram).

4.6.7 *Squence* Diagram Laporan Diagnosa

Sequence diagram laporan diagnosa menggambarkan alur logika untuk melihat Laporan hasil diagnosa dan dapat ditambahkan pada menu dataset,data gejala yg sudah diinputkan oleh aktor, aktor yang mengakses fitur ini adalah user. Gambar alur *sequence* diagram laporan diagnosa terdapat pada Lampiran C.6 (*Sequence* Diagram).

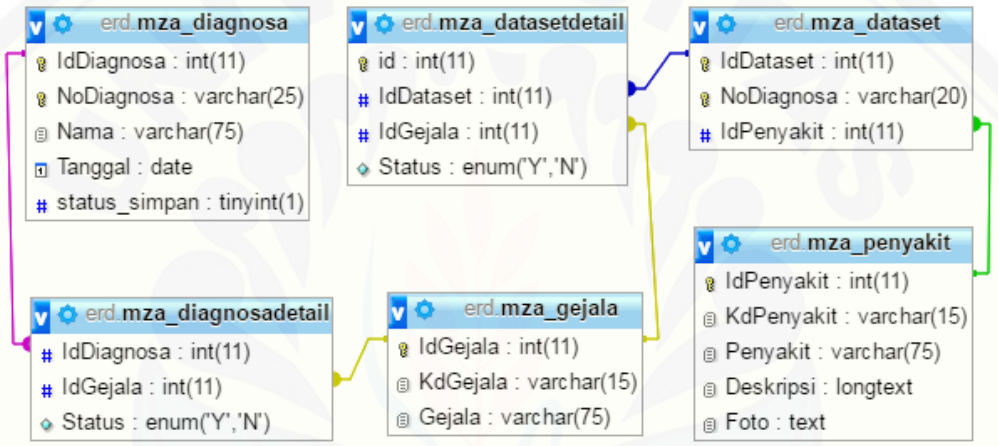
4.7 *Class Diagram*

Class Diagram merupakan suatu diagram yang menggambarkan struktur dan penjelasan *class*, *package*, dan objek serta hubungan satu sama lain. Pada *Class Diagram* sistem ini dibagi menjadi tiga yaitu *controller, view, dan model*. *Controller* menggambarkan struktur *class* dari logika sistem dan menghubungkan antara *model* dan *view*, *model* menggambarkan struktur *class* yang menghubungkan sistem dengan *database*, sedangkan *view* menggambarkan struktur *class* yang menangani tampilan dari sistem. Pada sistem pakar untuk mendiagnosa gangguan jeruk, *Class Diagram* terdiri dari 3 class utama yaitu *CI_controller*, *CI Model* dan *loader_view*. Pada class utama tersebut memiliki fungsi masing – masing yaitu *CI_controller* menghubungkan *controller* satu dengan lainnya. *Controller* pada sistem pakar untuk mendiagnosa penyakit ayam terdiri dari login, home, diagnosa dan lain lain. Gambaran *Class Diagram* pada penelitian ini dapat dilihat pada Lampiran D.

4.8 *EntityRelationship Diagram* (ERD)

Entity Relationship Diagram merupakan suatu model untuk menjelaskan hubungan antar data dalam basis data berdasarkan objek-objek dasar data yang

mempunyai hubungan antar relasi. *Entity Relationship Diagram* digunakan untuk memodelkan struktur data dan hubungan antar data, untuk menggambarkannya digunakan beberapa notasi dan simbol. Pada sistem pakar untuk mendiagnosa penyakit ayam terdapat 6 tabel yaitu mza_dataset, mza_datasetdetail, mza_diagnosa, mza_diagnosadetail, mza_gejala, mza_penyakit.*Entity Relationship Diagram (ERD)* pada penelitian ini dapat dilihat pada gambar 4.5.



Gambar 4.5 ERD Diagnosa Ayam

4.9 Implementasi Perancangan dan Penulisan Kode Program

Setelah tahap desain perancangan selesai, tahap selanjutnya dalam penelitian ini adalah tahap pengimplementasian desain perancangan ke dalam bahasa pemrograman. Bahasa pemrograman yang dipakai adalah bahasa pemrograman PHP (*Hypertext Preprocessor*) dengan memanfaatkan *framework CodeIgniter*, dan *database* yang digunakan adalah MySql (PhpMyadmin).

Pada tahap implementasi perancangan ini menjelaskan tentang fitur-fitur yang terdapat pada aplikasi mendiagnosis penyakit ayam. Fitur-fitur tersebut meliputi login, View Penyakit,Data diagnosa, Manajemen gejala, Manajemen penyakit,

Manajemen Dataset, Manajemen Laporan diagnosa. Dalam tahap ini juga mengimplementasikan metode *Naive Bayes Classifier* didalam kode program.

Kode program diagnosa penyakit ayam pada sistem pakar dengan menerapkan metode *naive bayes classifier*terdapat pada *class* diagnosa pada *package controllers*, dan pada *class*m_query pada *package models,* kemudian tampilan diagnosa dan hasil diagnosa ditampilkan pada *package* diagnosa_form dan diagnosa_hasil yang dapat dilihat dilampiran E.

4.10 Pengujian Sistem

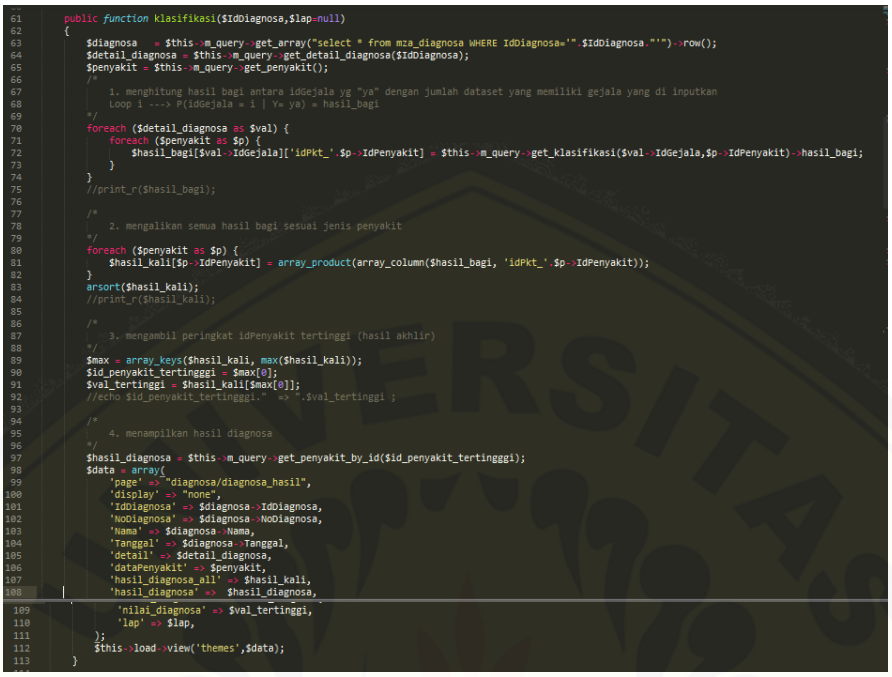
Tahapan pengujian aplikasi merupakan suatu tahapan yang dilakukan secara sistematis untuk menguji dan mengevaluasi sistem dengan menggunakan sebuah metode pengujian sistem. Hal tersebut dilakukan dengan tujuan untuk mengevaluasi apakah kebutuhan sistem telah terpenuhi dan sistem layak untuk digunakan oleh pengguna. Agar pengujian yang dilakukan lebih valid, maka tahap pengujian sistem ini dilakukan dengan menggunakan dua metode, yaitu *white box* dan *black box*.

4.10.1 *White Box Testing*

White Box Testing merupakan pengujian pada modul pengkodean program untuk menjamin kode program bebas dari kesalahan sintaks maupun logika. Dalam pengujian *white box* terdapat beberapa tahapan pembuatan dokumentasi pengujian yaitu *cyclomatic complexity* (CC), *listing program*, penentuan jalur *independen*, dan *test case*. Tahapan-tahapan pengujian dengan metode *white box* ini akan diterapkan pada fitur yang dinilai dapat mewakili sistem ini. Tahapan pengujian jalur dasar meliputi:

a. *Listing Program*

Listing program merupakan baris-baris kode yang akan diuji. Setiap langkah dari kode-kode yang ada diberi nomor baik menjalankan *statement* biasa atau penggunaan kondisi dalam program. *Listing program* diagnosa penyakit ayam dapat dilihat pada gambar 4.6.

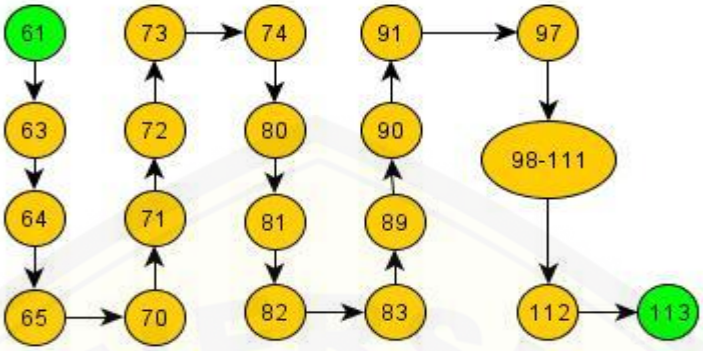


```php
public function klasifikasi($IdDiagnosa,$lap=null)
{
    $diagnosa    = $this->m_query->get_array("select * from mza_diagnosa WHERE IdDiagnosa='".$IdDiagnosa."'")->row();
    $detail_diagnosa = $this->m_query->get_detail_diagnosa($IdDiagnosa);
    $penyakit = $this->m_query->get_penyakit();
    /*
        1. menghitung hasil bagi antara idGejala yg "ya" dengan jumlah dataset yang memiliki gejala yang di inputkan
        Loop i ---> P(idGejala = i | Y= ya) = hasil_bagi
    */
    foreach ($detail_diagnosa as $val) {
        foreach ($penyakit as $p) {
            $hasil_bagi[$val->IdGejala]['idPkt_'.$p->IdPenyakit] = $this->m_query->get_klasifikasi($val->IdGejala,$p->IdPenyakit)->hasil_bagi;
        }
    }
    //print_r($hasil_bagi);

    /*
        2. mengalikan semua hasil bagi sesuai jenis penyakit
    */
    foreach ($penyakit as $p) {
        $hasil_kali[$p->IdPenyakit] = array_product(array_column($hasil_bagi, 'idPkt_'.$p->IdPenyakit));
    }
    arsort($hasil_kali);
    //print_r($hasil_kali);

    /*
        3. mengambil peringkat idPenyakit tertinggi (hasil akhlir)
    */
    $max = array_keys($hasil_kali, max($hasil_kali));
    $id_penyakit_tertinggi = $max[0];
    $val_tertinggi = $hasil_kali[$max[0]];
    //echo $id_penyakit_tertinggi."  => ".$val_tertinggi ;

    /*
        4. menampilkan hasil diagnosa
    */
    $hasil_diagnosa = $this->m_query->get_penyakit_by_id($id_penyakit_tertinggi);
    $data = array(
        'page' => "diagnosa/diagnosa_hasil",
        'display' => "none",
        'IdDiagnosa' => $diagnosa->IdDiagnosa,
        'NoDiagnosa' => $diagnosa->NoDiagnosa,
        'Nama' => $diagnosa->Nama,
        'Tanggal' => $diagnosa->Tanggal,
        'detail' => $detail_diagnosa,
        'dataPenyakit' => $penyakit,
        'hasil_diagnosa_all' => $hasil_kali,
        'hasil_diagnosa' =>  $hasil_diagnosa,
        'nilai_diagnosa' => $val_tertinggi,
        'lap' => $lap,
    );
    $this->load->view('themes',$data);
}
```

Gambar 4.6 *Listing* Program Proses Diagnosa

b. Diagram Alir

Diagram alir merupakan notasi sederhana yang digunakan untuk merepresentasikan aliran kontrol. Aliran kontrol yang digambarkan merupakan hasil penomoran dari *listing program.* Diagram alir digambarkan dengan *node-node* (simpul) yang dihubungkan dengan *edge-edge* (garis) yang menggambarkan alur jalannya program. Diagram alir *Fuction* Klasifikasi dapat dilihat pada gambar 4.7



Gambar 4.7 Diagram Alir *Function* Klasifikasi

c. Kompleksitas Siklomatik (*cyclomatic complexity*).

Kompleksitas siklomatik merupakan matriks perangkat lunak yang menyediakan ukuran kuantitatif dari kompleksitas logis suatu program. Bila digunakan dalam konteks teknik pengujian jalur dasar, nilai yang dihitung untuk kompleksitas siklomatik mendefinisikan jumlah jumlah jalur independen dalam basis set suatu program. Perhitungan kompleksitas siklomatik menggunakan rumus dan dapat dilihat pada persamaan 4.1

$$V(G) = E - N + 2 \quad \text{persamaan....(4.1)}$$

Keterangan :

V (G) = kompleksitas siklomatik

E = jumlah *edge* (garis)

N = jumlah *node* (simpul)

CC *function*klasifikasi= 19 - 20 + 2

CC = 1

d. Jalur Independen

Jalur independen adalah setiap jalur yang melalui program, menunjukkan setidaknya satu kumpulan pernyataan-pernyataan pemrosesan atau kondisi baru. Bila

dinyatakan dalam grafik alir, jalur independen harus bergerak setidaknya sepanjang satu edge yang belum dilintasi sebelum jalur tersebut didefinisi.

*Function*Klasifikasi diagnosa penyakit ayam

Jalur 1 = 61 – 63 – 64 – 65 – 70 – 71 – 72 – 73 – 74 – 80 – 81 – 82 – 83 – 89 – 90 – 91 – 97 – 78 – [98 – 111] –112 – 113

4.10.2 *Black Box Testing*

Pengujian *black box* menitik beratkan pada fungsionalitas sistem. Pengujian ini tidak melihat kinerja internal dari sistem, jadi hanya berfokus pada kinerja sistem sesuai dengan spesifikasi dan kebututuhan yang dianalisis pada bab perancangan. Hasil pengujian *black box* dapat dilihat pada Lampiran F.

4.10.3 Uji Validitas dan Reliabilitas

Pada Uji Validitas dan Reliabilitas dengan menggunakan angket atau kuisioner sebagai alat ukur instrumen pada aplikasi identifikasi penyakit pada ayam petelur dan *broiler*. Pada pertanyaan yang diajukan dengan menggunakan skala likert lima pilihan jawaban yaitu:

a. Skor 1 = Sangat Tidak Paham (STP)
b. Skor 2 = Tidak Paham (TP)
c. Skor 3 = Cukup Paham (CP)
d. Skor 4 = Paham (P)
e. Skor 5 = Sangat Paham (SP)

Adapun daftar pertanyaan yang diajukan terdapat pada Tabel 4.10.

Tabel 4. 10 Daftar pertanyaan pada angket atau kuisioner

No.	Pertanyaan	SP	P	CP	TP	STP
1.	Apakah anda paham manfaat dan kegunaan dari sistem pakar ini?					
2.	Apakah anda paham kenapa peternakan ini di buatkan sistem pakar?					
3.	Apakah anda sudah paham cara penggunaan sistem pakar ini?					

No.	Pertanyaan	SP	P	CP	TP	STP
4.	Apakah anda paham penjelasan solusi yang di berikan sistem pakar?					
5.	Apakah anda paham dengan kalimat ini? "Apakah anda setuju untuk mendiagnosa penyakit pada ayam menggunakan sistem pakar ini?"					
6.	Apakah anda paham dengan kalimat ini "Apakah sistem pakar ini cukup membantu peternak mengidentifikasi penyakit pada ayam?"					
7.	Apakah anda paham dengan kalimat ini "Apakah sistem pakar ini mudah di gunakan dan dimengerti?"					
8.	Apakah anda paham dengan kalimat ini "Apakah sistem pakar ini sudah cukup jelas dalam memberikan solusi terhadap penyakit yang sudah teridentifikasi?"					
9.	Apakah anda paham dengan kalimat ini "Apakah sistem pakar ini bermanfaat bagi peternak ayam?"					
10.	Apakah anda paham dengan kalimat ini "Apakah dengan menggunakan sistem pakar untuk mendiagnosa penyakit ayam dapat dengan mudah mengidentifikasi penyakit pada ayam di banding tanpa sistem?"					

Setelah pertanyaan diajuakan, didapatkan sejumlah 30 responden dengan yang masing-masing 6 item pertanyaan dengan skor yang didapatkan berdasarkan jawaban yang diberikan responden dapat dilihat pada Tabel 4.11.

Tabel 4. 11 Hasil Rekap Kuisioner untuk aplikasi mendiagnosis Penyakit pada ayam

No.	Responden	No. Item					
		Item 1	Item 2	Item 3	Item 4	Item 5	Item 6
1	1	4	4	5	4	5	4
2	2	5	4	5	4	5	4
3	3	4	4	5	5	5	3

4	4	5	5	4	4	5	5
5	5	5	4	5	5	5	4
6	6	3	3	3	5	5	3
7	7	5	5	5	5	5	5
8	8	4	5	5	4	4	5
9	9	5	5	4	4	5	5
10	10	5	5	5	5	5	5
11	11	5	5	5	5	5	5
12	12	4	3	4	3	4	4
13	13	4	5	5	3	4	5
14	14	5	5	5	5	5	5
15	15	5	3	3	4	4	3
16	16	4	4	4	4	4	4
17	17	3	3	3	4	3	3
18	18	4	4	5	3	4	4
19	19	4	3	4	3	4	3
20	20	5	4	5	4	4	4
21	21	5	4	5	5	5	5
22	22	4	3	4	4	5	5
23	23	5	4	4	5	4	5
24	24	4	3	5	4	4	5
25	25	3	4	4	3	4	3
26	26	4	4	4	4	4	4
27	27	5	5	5	5	5	5
28	28	4	3	4	4	4	5

29	29	5	5	5	5	5	5
30	30	5	3	4	4	4	5

BAB 6. PENUTUP

Bab ini berisi mengenai kesimpulan dan saran dari peneliti tentang penelitian yang telah dilakukan. Kesimpulan dan saran tersebut diharapkan dapat digunakan sebagai acuan pada penelitian selanjutnya.

6.1 Kesimpulan

Kesimpulan dari hasil penelitian yang telah dilakukan oleh peneliti adalah sebagai berikut:

1. Berdasarkan hasil penelitian yang di lakukan hasil perhitungan *naive bayes* dengan menginputkan gejala jumlah minimum yaitu 11 gejala adapun gejala yang di inputkan adalah diare, nafas sesak, nafas ngorok, bersin-bersin, batuk, nafsu makan berkurang, mati mendadak, Mencret putih, tidur paruh diletakkan di lantai, muka dan mata bengkak, mata lesu menghasilkan 2 hasil perhitungan *naive bayes* yaitu penyakit Avian Influenza dengan nilai 8,1333169E-11 dan penyakit Tetelo 2,1857879E-14, tetapi yang mendapat nilai tertinggi adalah penyakit Avian Influenza dan diteteapkan sebagai hasil diagnosa dari gejala yang di inputkan. Untuk penelitian menggunakan inputan gejala dengan jumlah maksimal yaitu 18 gejala, dan gejala yang di inputkan adalah diare, nafas sesak, nafas ngorok, batuk, nampak membiru, kepala bengkak, tampak lesu, sempoyongan, kedinginan, mencret putih, jengger merah bengkak, kaki bengkak, mencret darah, lendir bercampur dengan darah pada rongga mulut, kaki pincang, kotoran banyak menempel dianus, pertumbuhan lambat, kornea menjadi keruh menghasilkan perhitungan naive bayes yaitu penyakit Tetelo dengan nilai 2,60056602182E-21 karna penyakit tersebut yang memiliki banyak gejala.

2. Berdasarkan hasil pengujian sistem terdapat 3 hasil yaitu diantaranya:
   a. Hasil perhitungan akurasi sistem dengan menginputkan beberapa gejala yang berbeda, didapatkan sejumlah 8 hipotesa yang valid dari 10 jumlah hipotesa. Sehingga didapatkan persentase sebesar 80%.
   b. Hasil uji validitas menunjukkan 10 item yang diajukan sebagai pertanyaan terhadap 30 responden seluruhnya menghasilkan nilai t hitung > t tabel yang keseluruhan item tersebut menghasilkan nilai yang valid.
   c. Hasil uji reliabilitas menunjukkan koefisien reliabilitas sebesar 0,654545 lebih besar daripada 0,60 yang merupakan nilai minimal tingkat reliabilitas suatu instrumen yang reliabel sehingga dapat disimpulkan bahwa instrumen tersebut memiliki ingkat reliabilitas yang baik.

6.2 Saran

Pengembangan lebih lanjut pada penelitian ini diharapkan dapat menambahkan hasil akurasi yang lebih maksimal dengan menambahkan jumlah dataset dengan jumlah data yang lebih dari penelitian ini sehingga menghasilkan hasil klasifikasi yang lebih maksimal dan diharapkan dapat lebih banyak penyakit yang di diagnosa.

DAFTAR PUSTAKA

Giarratano,J. Dan Riley,G.2005. *Expert System Principles dan Programming, PWS Publishing Company*, Boston.

Hardika P, Angga. (Tanpa Tahun). *Aplikasi Sistem Pakar Untuk Identifikasi Hama dan Penyakit Tanaman Tebu Dengan Metode Naive Bayes Berbasis Web*. Malang, Universitas Brawijaya.

Karina, Nia Esti. dan Yamasari, Yuni. 2013. *Aplikasi Diagnosa Kanker Kandungan Dengan Metode Naive Bayes Berbasis Web*. Surabaya, Universitas Negeri Surabaya.

Rasyaf. 2009. Panduan Beternak Ayam Bertelur, Jakarta : Swadaya.

Setiawan,W. Dan Ratnasari,S. 2014. *Sistem Pakar Diagnosis Penyakit Mata Menggunakan Naive Bayes Classifier*. Madura, Universitas Trunojoyo Madura.

Sommervile, I. 2011. Software Engineering (Rekasyasa Perangkat Lunak). Jakarta: Erlangga.

Tentua, Meilany Nonsi. 2009. *Sistem Pakar Diagnosis Penyakit Ayam*. Yogyakarta, Universitas PGRI Yogyakarta.

Turban,E.,Aronson,J. Dan Peng L.,2005. *Decision Support System and Intellegence System-7th Ed, Pearson education, New Jersey*.

Yakub, Suardin. 2008. *Sistem Pakar Deteksi Penyakit Diabetes Mellitus Dengan Menggunakan Pendekatan Naive Bayesian Berbasis Web.* Malang, Universitas Ialam Negeri (UIN) Maulana Malik Ibrahim Malang.

Zulkarnaen, D. 2013. *Lebih Sukses & Untung Beternak Ayam Broiler*. Surabaya: Dafa Publising.

LAMPIRAN

A. Tabel Penyakit

No	Nama Penyakit	Deskripsi	Gejala
1.	Avian Influenza	- Dikenal sebagai flu burung - Disebabkan oleh virus H5N1 - Disebut juga penyakit Fowl Plaque - Penyakit yang berbahaya karena dapat menular pada manusia dan dapat menyebabkan kematian.	- Diare - Sesak nafas - Nafas ngorok - Bersin-bersin - Batuk - Nafsu makan berkurang - Produksi telur menurun - Kepala kebiruan - Keluar cairan dari berbusa dari mata - Kepalabengkak - Mati mendadak
2.	Newcastle atau Tetelo	- Menyerang bagian pernafasan - Disebabkan oleh virus Paramyxo - Penyakit menular yang biasanya dalam 3-4 hari seluruh ternak akan terinfeksi	- Sesak nafas - Nafas ngorok - Bersin-bersin - Batuk - Nafsu makan berkurang - Produksi telur menurun - Kelihatan ngentukdan bulu berdiri - Tampak lesu - Mencret hijau - Sempoyongan - Kepala berputar
3.	Gumboro	- Menyerang sistem kekebalan tubuh - Menyebabkan kerusakan yang parah karenaantibodi tubuh tidak terbentuk.	- Bulu kusam - Nafsu akan berkurang - Kedinginan - Tampak lesu - Mencret putih - Tidur paruh diletakkan dilantai - Duduk membungkuk
4.	Kolera	- Disebabkan oleh bakteri Pasteurella Gallinarum atau Pasteurella Multocida - Biasanya menyerang ayam pada usia 12 minggu - Menyerang pernafasan dan	- Diare - Sesak nafas - Nafas ngorok - Batuk - Bulu kusam - Nafsu makan

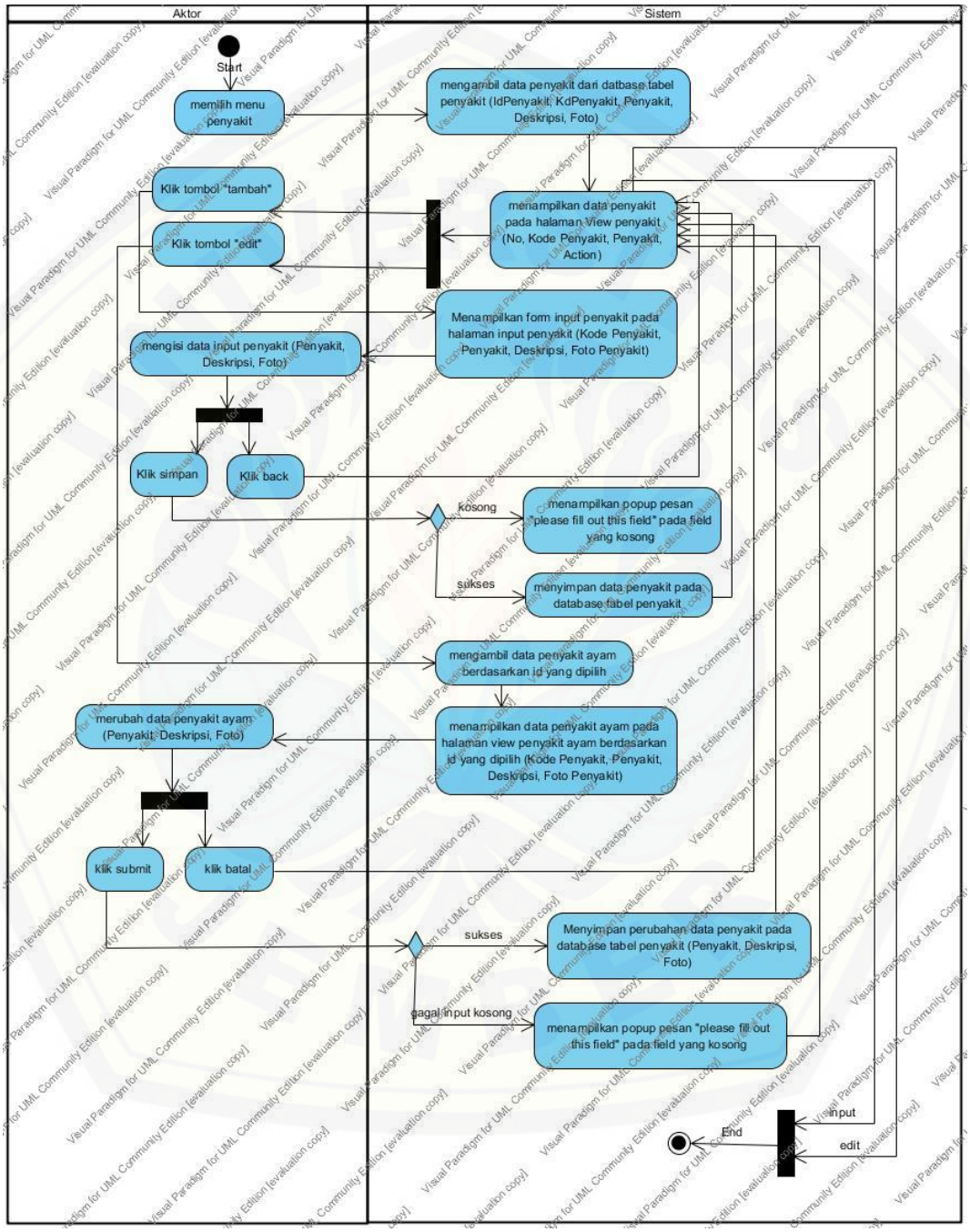
		pencernaan	berkurang - Produksi telur menurun - Kelihatan ngantuk dan bulu berdiri - Tampak lesu - Mencret hijau - Banyak minum - Jengger merah bengkak - Kaki meradang atau lumpuh - Keluar cairan dari mata dan hidung
5.	Berak Kapur	- Disebabkan oleh bakteri Salmonella Pollurum - Sering ditemukan pada anak ayam - Dapat menular - Biasanya mulai menjangkit sejak menetas dan menyebabkan kematian pada anak ayam	- Diare - Sesak nafas - Nafas cepat - Badan kurus - Bulu kusam - Nafsu makan berkurang - Produksi telur menurun - Kedinginan - Tampak lesu - Mencret putih - Kaki bengkak - Lotoran putih - menempel dianus
6.	Berak Darah	- Disebabkan oleh bakteri Haemophillus Gallinarum - Biasanya menyerang ayam pada saat perubahan musim - Menyerang ayam semua umur - Untuk ayam petelur yang terkena penyakit ini biasanya produktivitas telur akan menurun sampai 25%	- Badan kurus - Nafsu makan berkurang - Produksi telur menurun - Mencret darah - Muka pucat
7.	Batuk ayam menahun atau Infectious Bronchitis	- Disebabkan oleh Corana Virus - Menyerang sistem pernafasan - Penularan dapat terjadi melalui udara, minuman, makanan, peralatan, dan pakaian. - Virus ini hidup selama	- Diare - Nafas ngorok - Bersin-bersin - Batuk - Nafsu makan berkurang - Produksi telur menurun - Kelihatan

		kurang satu minggu jika tidak terdapat pada ternak karena jenis virus ini mudah mati jika terkena panas atau desinfektan	ngantuk dan bulu berdiri - Tampak lesu - Nampak membiru
8.	Mareks	- Menyerang organ dalam bagian tubuh ayam - Disebabkan oleh virus marek - Biasanya terkena kelumpuhan pada bagian tubuhnya seperti sayap atau kaki	- Diare - Nafas cepat - Badan kurus - Nafsu makan berkurang - Muka pucat - Sempoyongan - Kaki pincang - Sayap menggantung
9.	Tipus Ayam	- Penyakit yang menular - Disebabkan oleh bakteri Salminella Gallinarum - Cukup berbahaya - Penyebaran penyakit ini biasanya terdapat pada kotoran ayam yang sudah terkontaminasi dan pada bangkai ayam	- Diare - Badan kurus - Bulu kusam - Nafsu makan berkurang - Kelihatan ngantuk dan bulu berdiri - Tampak lesu - Mencret hijau - Jengger pucat
10.	Berak Hujau	- Penyakit yang menular - Disebabkan oleh bakteri Salmonella pullorum - Penularan penyakit ini melaluikontak langsung	- Jengger pucat - Mata lesu - Nafsu makan menurun - Sekitar pantat terlihat memutih dan lengket
11.	Chronicrespiratory desease (CRD)	- Disebabkan oleh bakteri Mycoplasma galisepticum - Menyerang ayam usia 4-9 minggu - Penyakit menular - Penularan melalui kontak langsung	- Batuk - Nafas ngorok - Keluar cairan dari hidung - Nafsu makan menurun - Produksi telur menurun - Ayam suka menggeleng-gelengkan kepala
	Colibacillosis	- Disebabkan oleh Escherichia coli - Penyakit berbahaya - Menyebabkan infeksi akut berat dan mati tiba-tiba - Disebabkan oleh bakteri Mycoplasma galisepticum	- Nafsu makan menurun - Tampak lesu - Bulu kasar - Nafas sesak - Kotoran banyak menempel di

		- Menyerang semua umur ayam - Paling banyak menyerang ayam usia muda	anus - Diare - Batuk - Banyak minum
3.	Snot / Coryza	- Disebabkan oleh bakteri Heamophillus gallinarum - Menyerang akibat perubahan musim - Menyerang pada semua usia ayam - Paling rentang menyerang ayam betina usia 18-23 minggu - Penyakit mematikan - Penyakit menular - Penularan melalui kontak langsung	- Kelihatan mengantuk - Keluar lendir dari hidung - Muka dan mata bengkak - Terdapat kerak dihidung - Nafsu makan menurun - Nafas ngorok - Nafas sesak - Pertumbuhan lambat

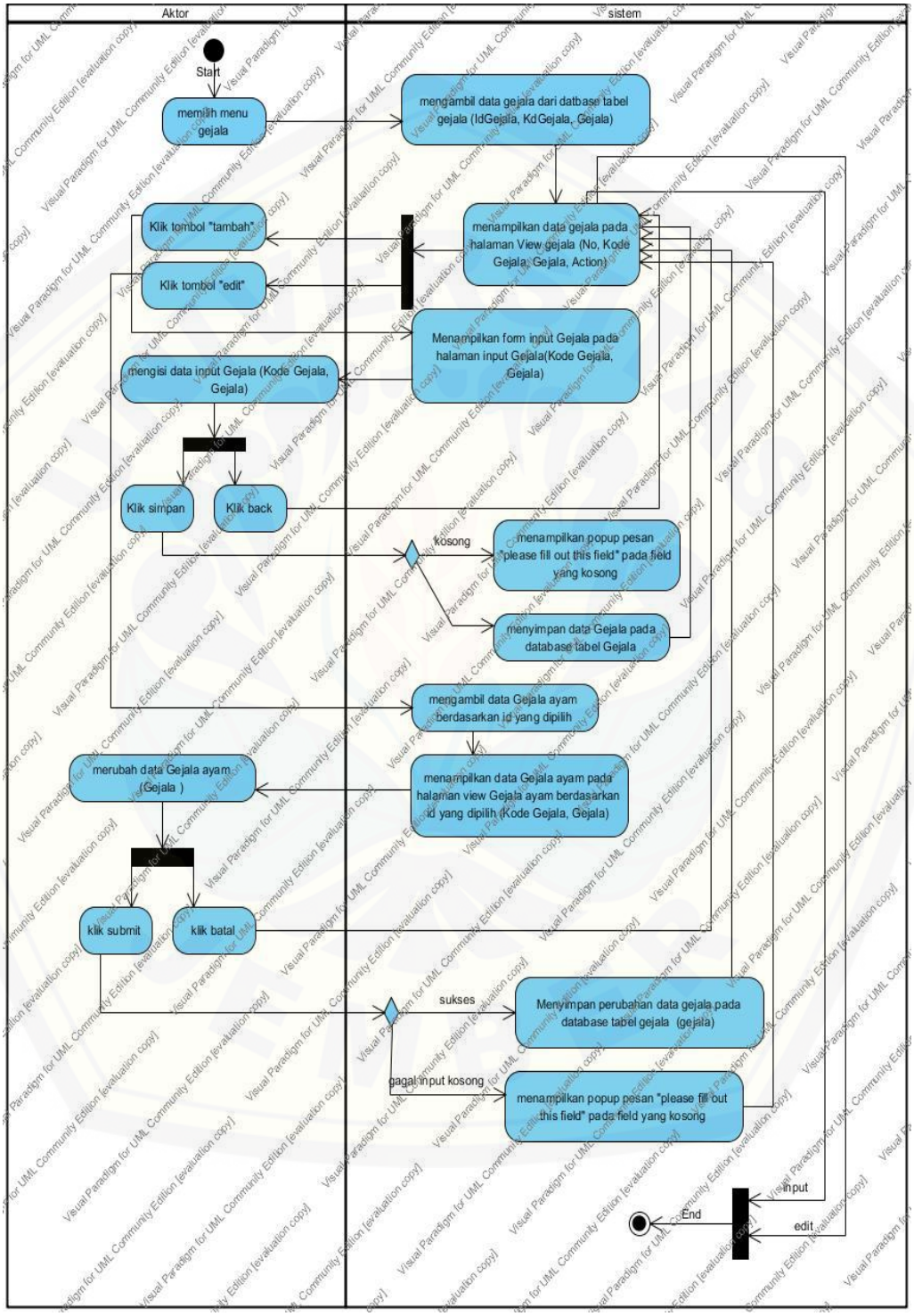
B. *Activity* Diagram

B.1.*Activity Diagram* Manajemen Penyakit



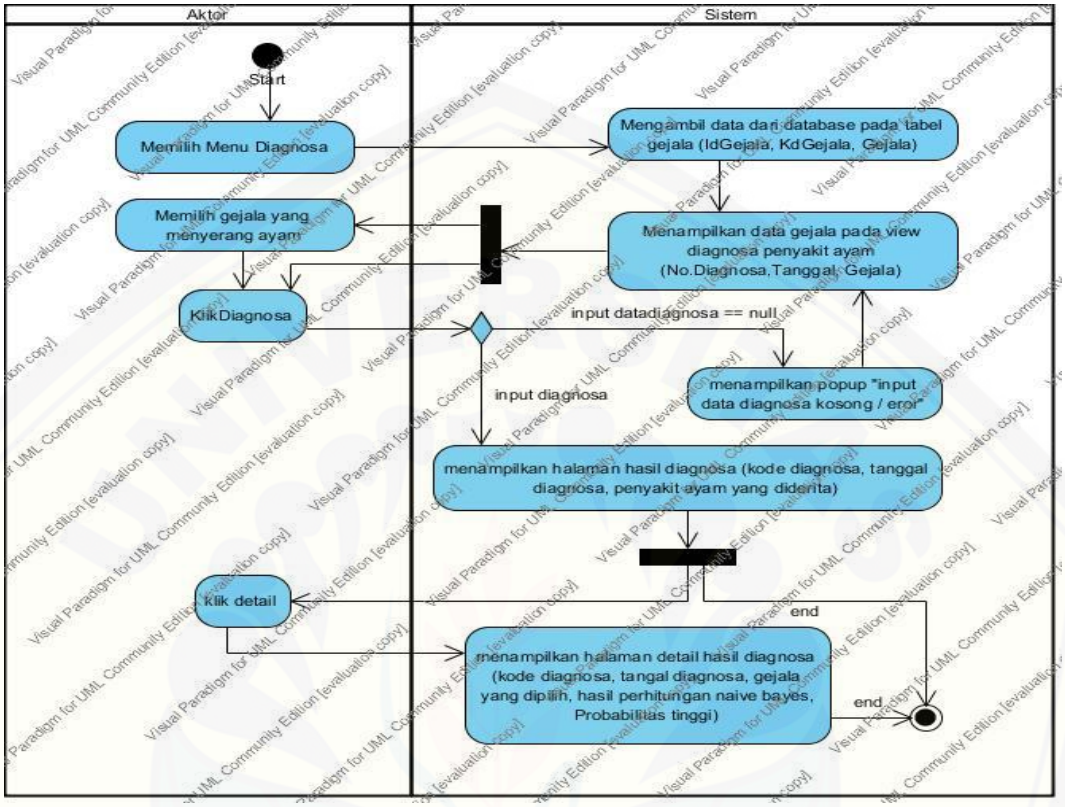
Gambar B.1*Activity* Diagram Manajemen Penyakit

B.2*Activity Diagram*Manajemen Gejala



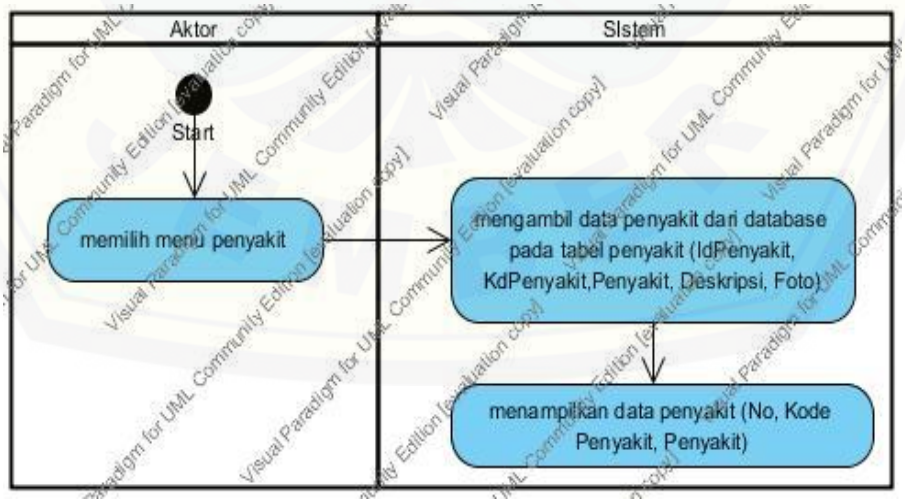
Gambar B.2 *Activity* Diagram Manajemen Gejala

B.3 *Activity* Diagram Diagnosa



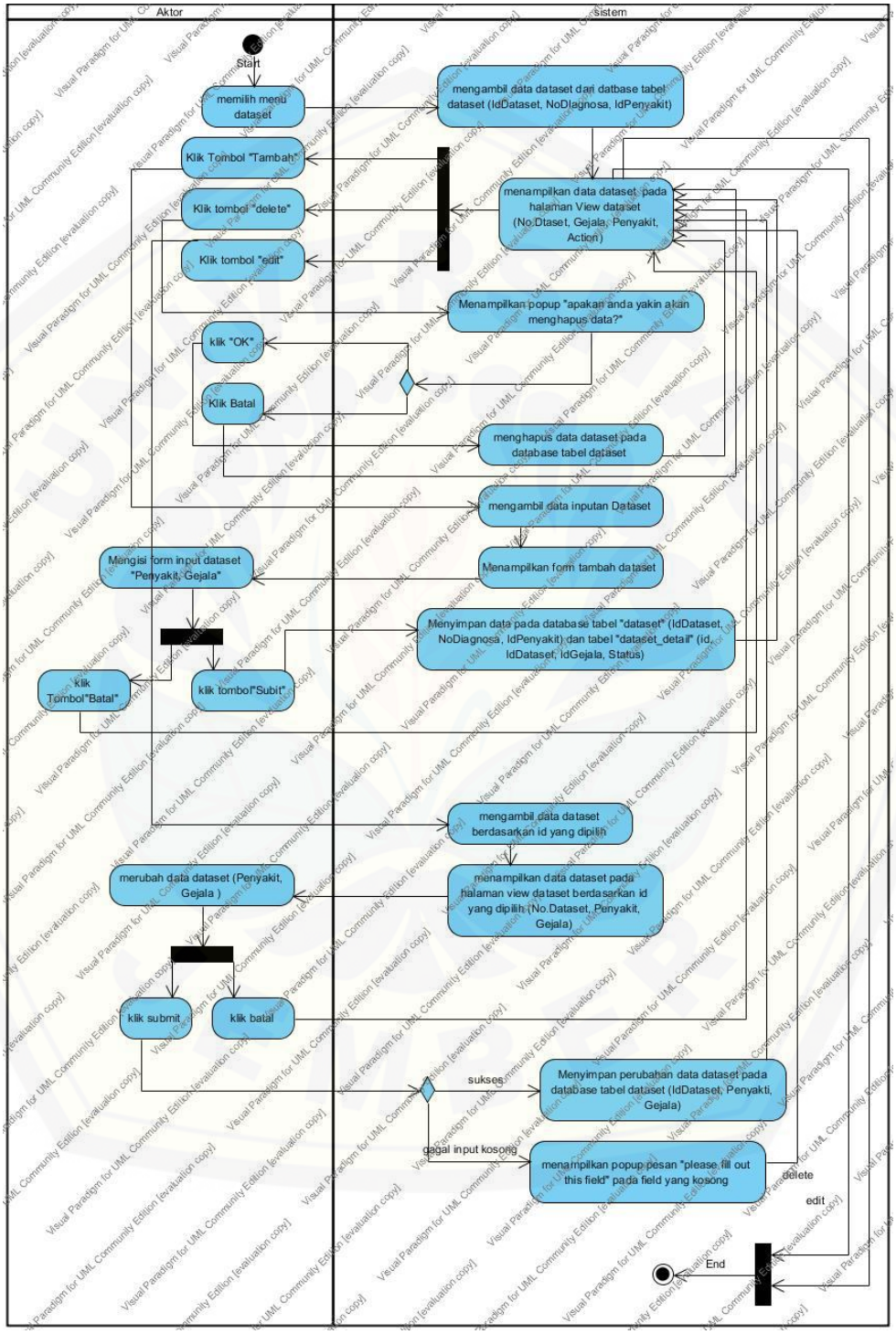
Gambar B.3 *Activity* Diagram Diagnosa

B.4 *Activity* Diagram View Penyakit



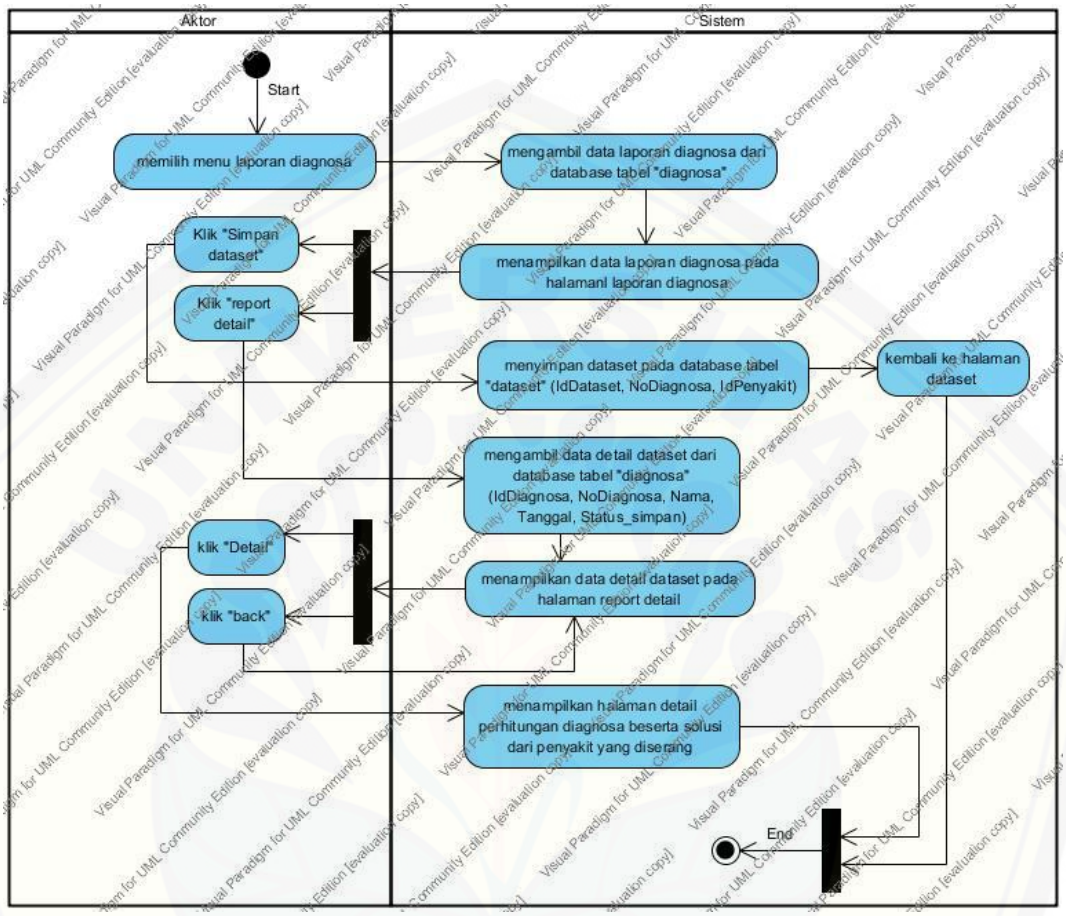
Gambar B.4 *Activity* Diagram View Penyakit

B.5*Activity Diagram* Manajemen Dataset



Gambar B. 5 *Activity* Diagram Manajemen Dataset

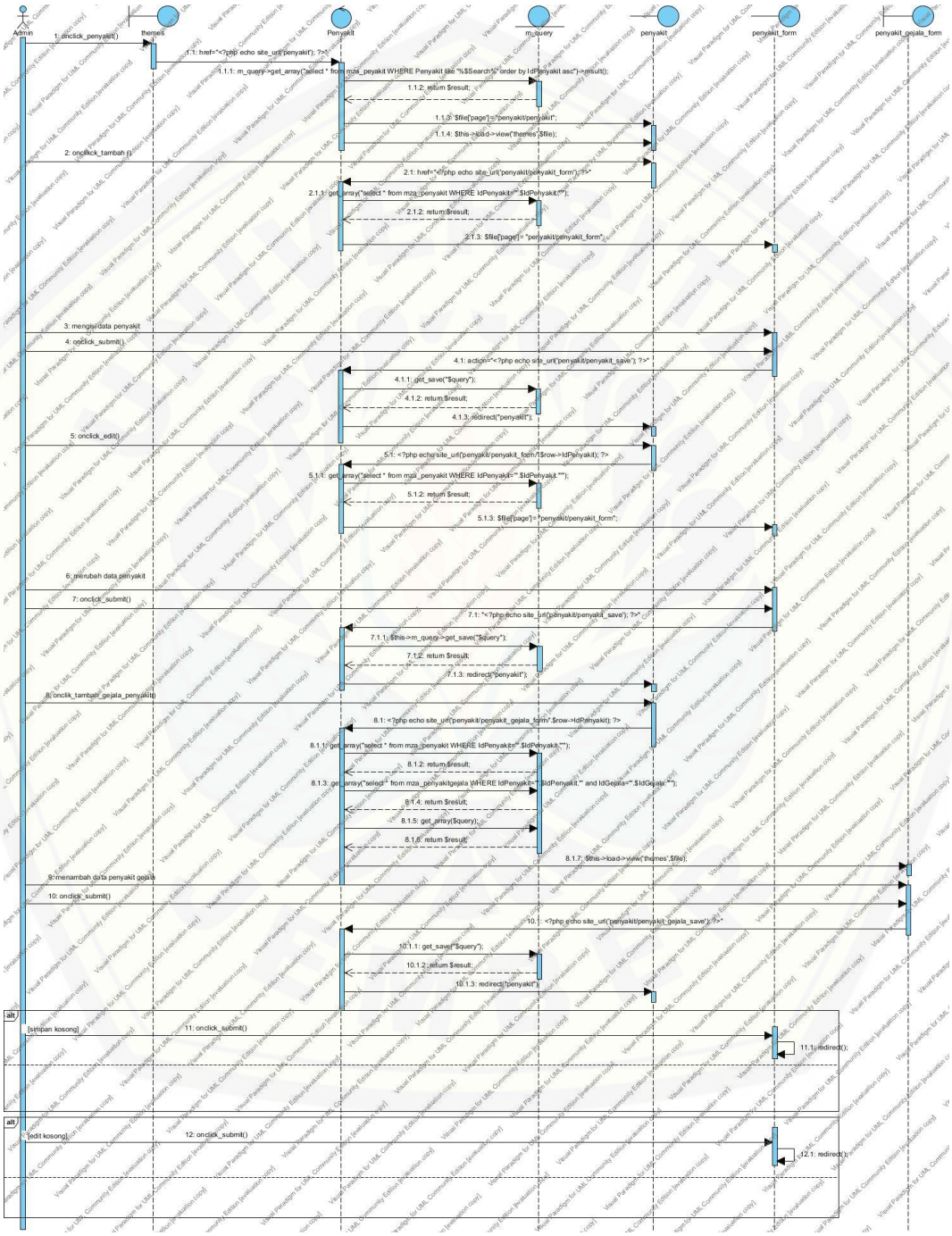
B.6 *Activity* Diagram Manajemen Laporan Diagnosa



Gambar B.6 *Activity* Diagram Manajemen Laporan Diagnosa

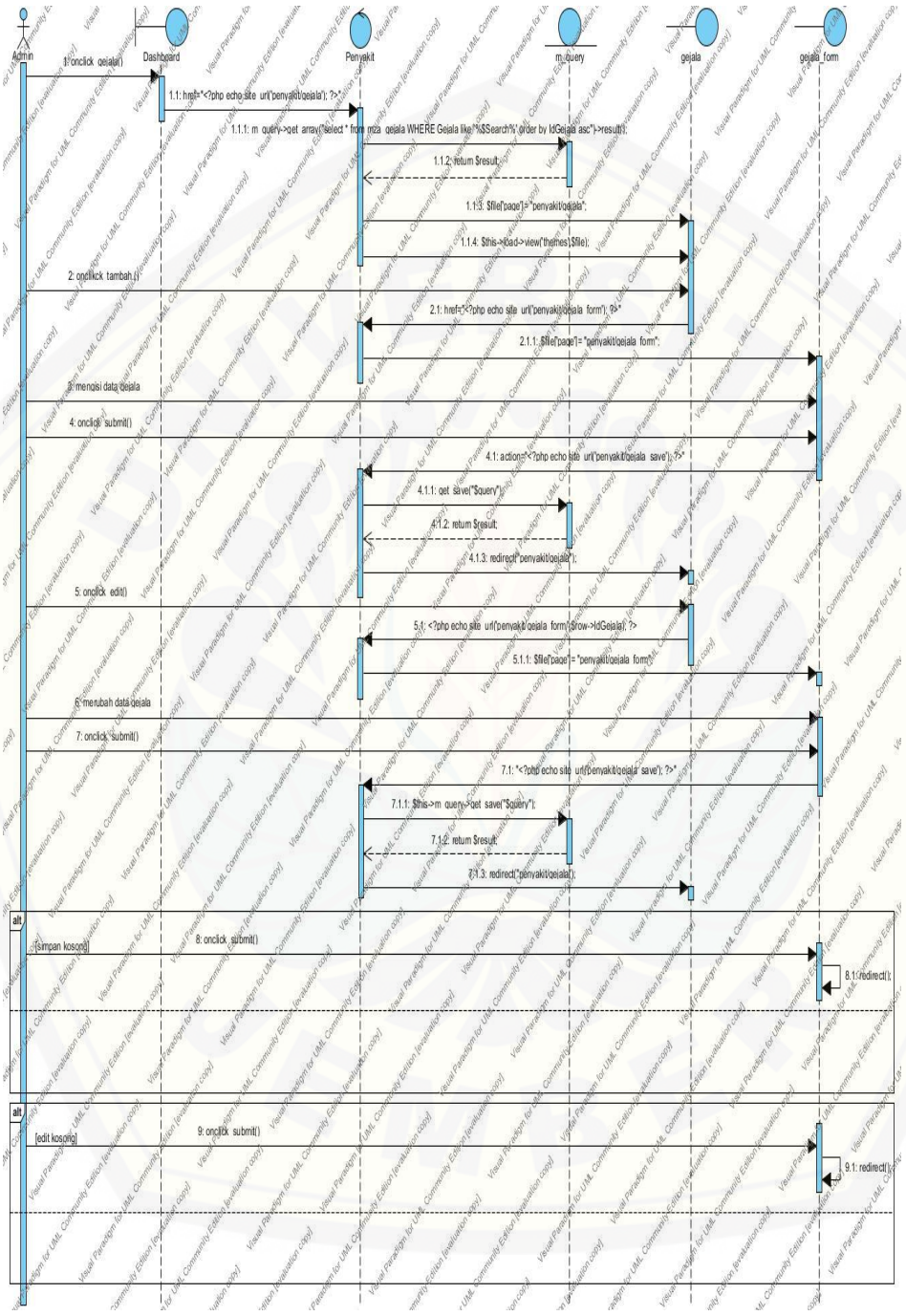
C. *Sequence* Diagram

C.1 *Sequence* Diagram ManajemenPenyakit (admin)



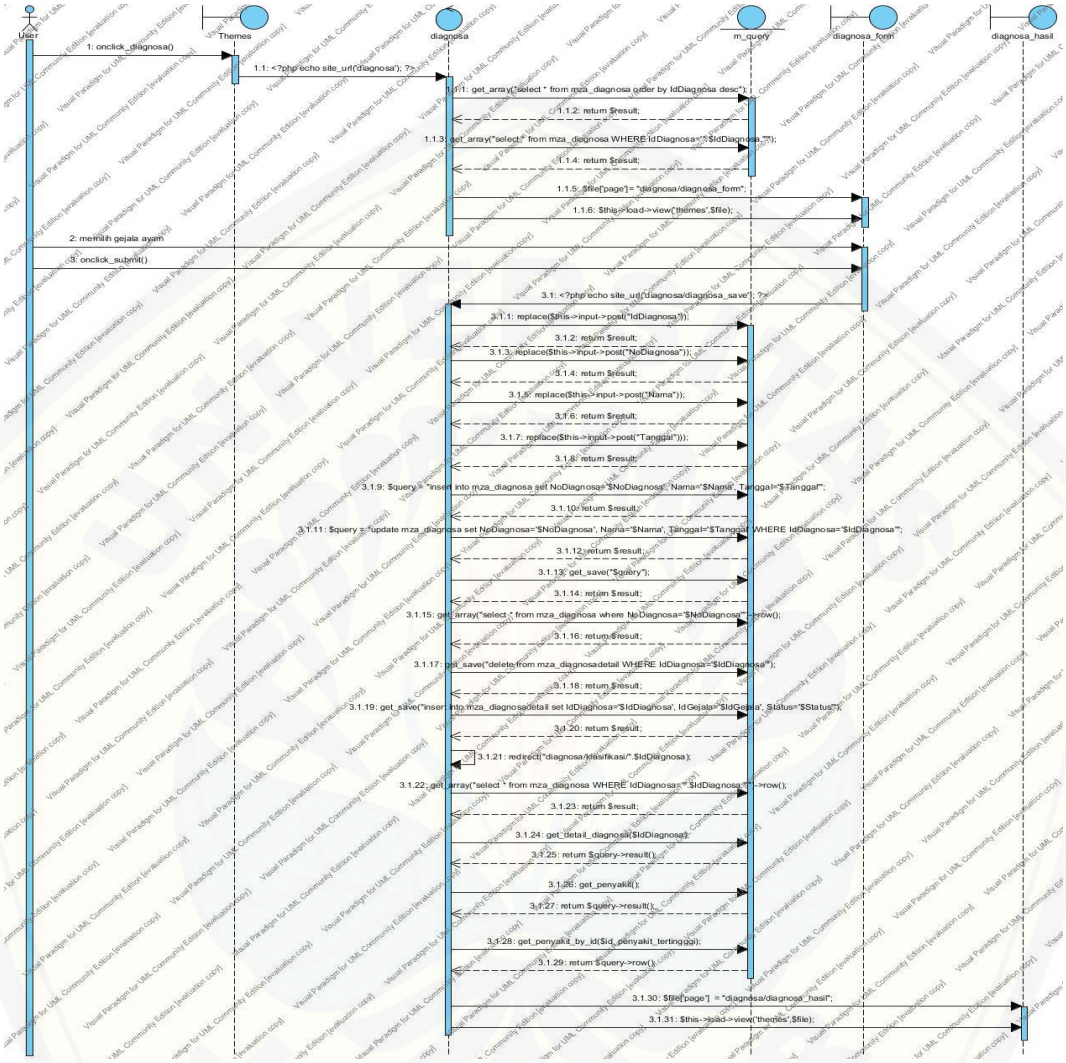
Gambar C.1 *Sequence* Diagram Manajemen Penyakit (admin)

C.2*Sequence Diagram* Manajemen Gejala (admin)



Gambar C.2 *Sequence* Diagram Manajemen Gejala (admin)

C.3 *Sequence Diagram* Diagnosa (User)

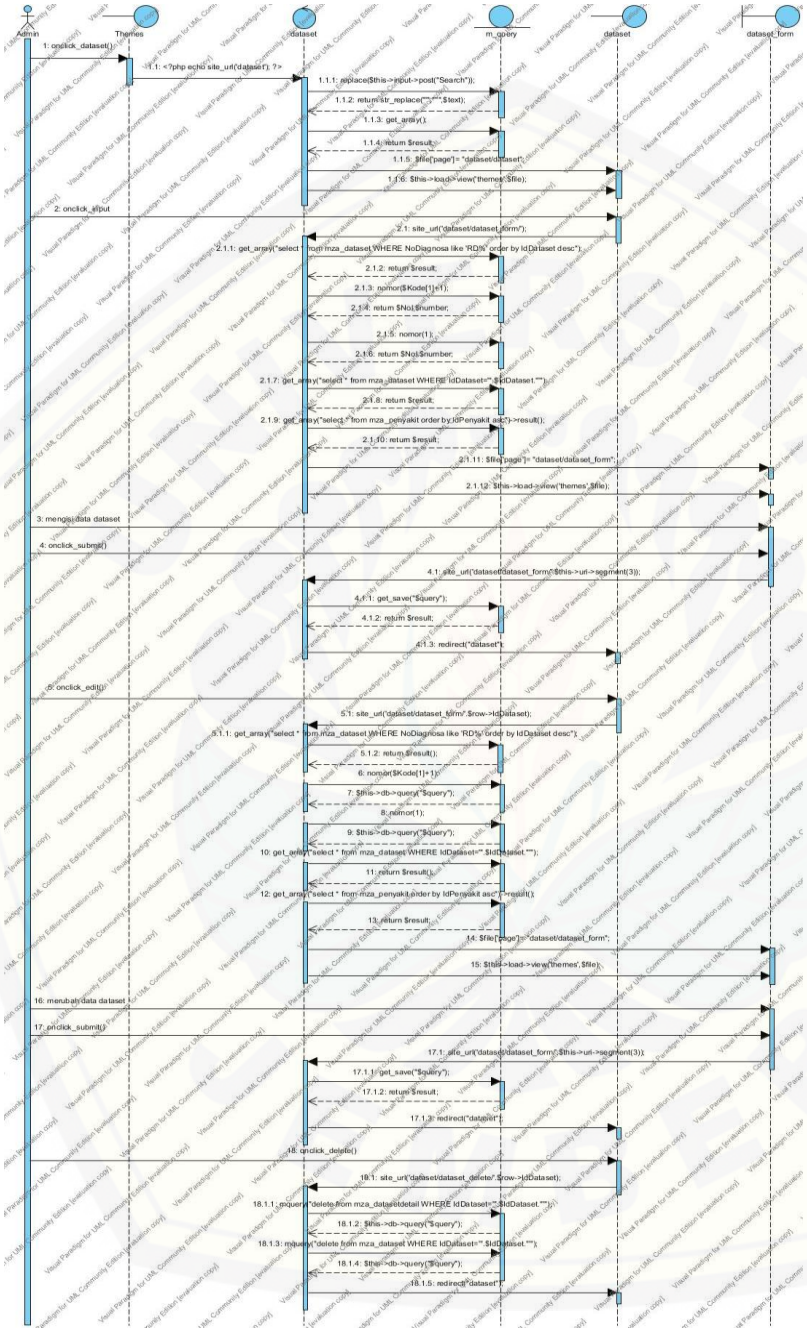


Gambar C.3 *Sequence* Diagram Diagnosa (User)

C.4 *Sequence* Diagram View Penyakit (User)

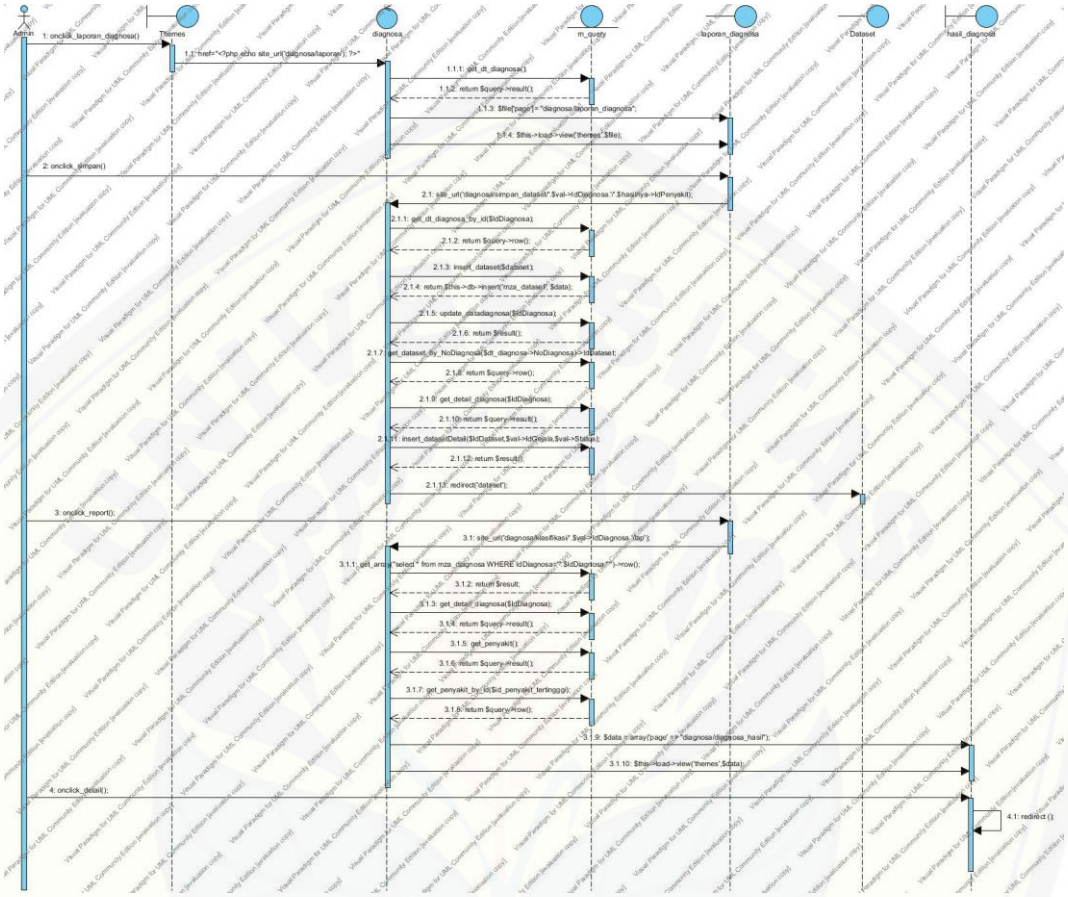
Gambar C.4 *Sequence* Diagram *View* Penyakit (User)

C.5*Sequence Diagram*Manajemen Dataset (admin)



Gambar C.5 Sequence Diagram Manajemen Dataset (admin)

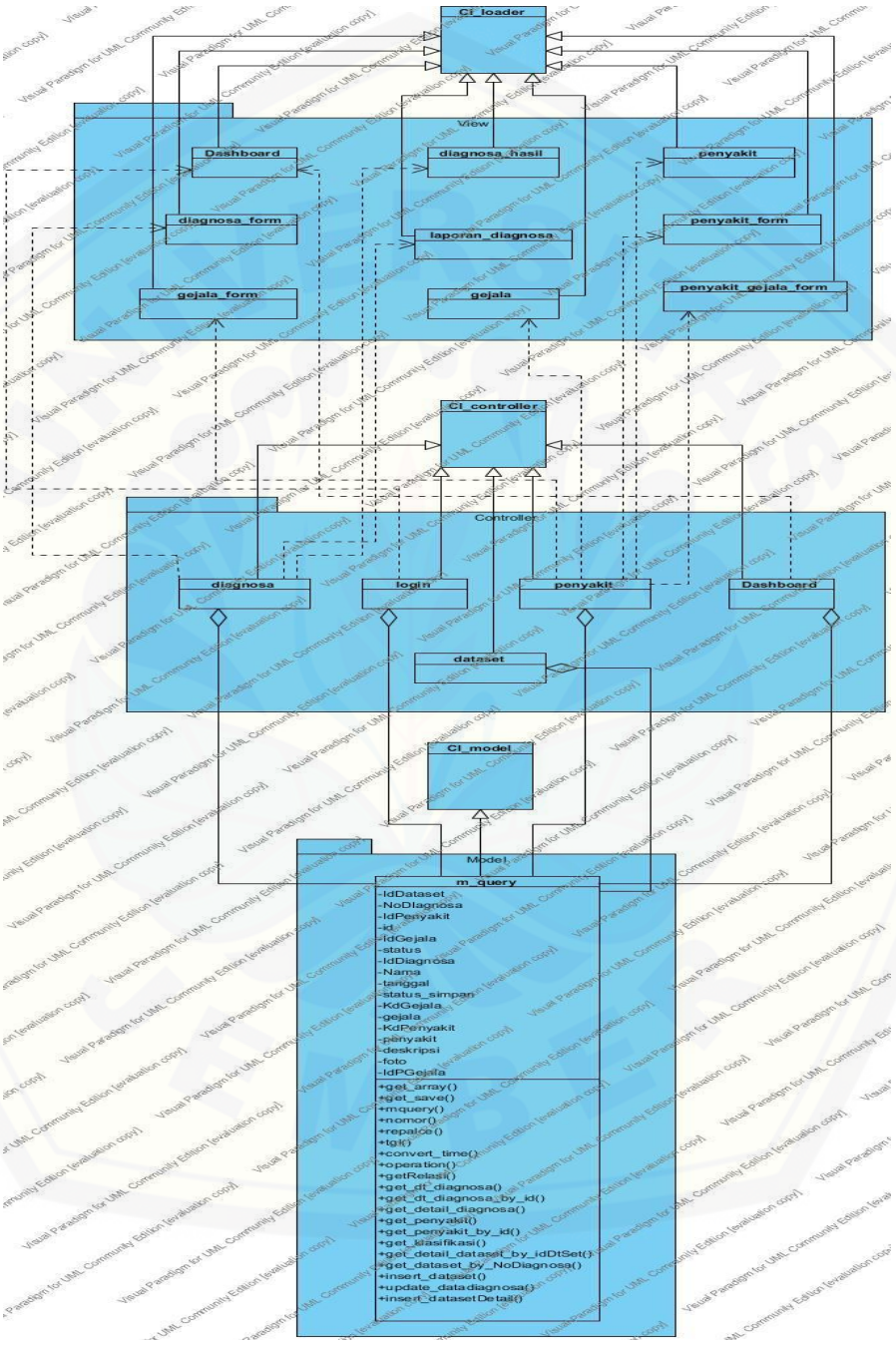
C.6 *Sequence Diagram* Manajemen Laporan Diagnosa (admin)



Gambar C.6 *Sequenc*e Diagram Manajemen Laporan Diagnosa (admin)

D.*Class* Diagram

Class Diagram Diagnosa Penyakit Ayam



Gambar D.*Class Diagram* Diagnosa Penyakit Ayam

E. Implementasi Koding

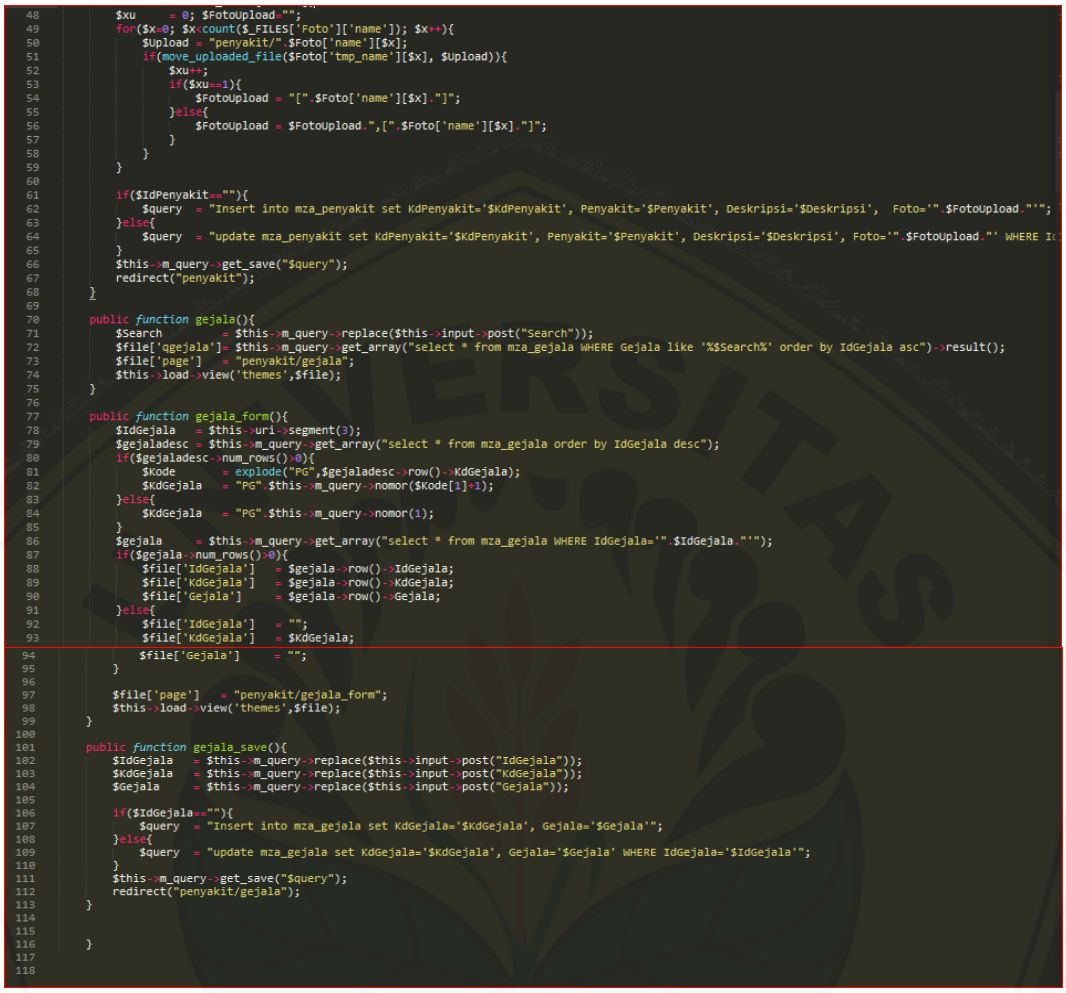
E.1 Login (login.php)

```php
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class login extends CI_Controller {
    public function index(){
        $file['page']    = "dashboard/dashboard";
        $this->load->view('themes',$file);
    }

    public function login_ok(){
        $Username    = $this->input->post('Username');
        $Password    = $this->input->post('Password');
        if($Username==$Password && $Username=="admin"){
            $this->session->set_userdata('login','ok');
        }
        redirect('');
    }

    public function login_no(){
        $this->session->set_userdata('login','no');
        redirect('');
    }
}

```

Gambar E.1*Login* (login.php)

E.2 Controller Penyakit dan Gejala (penyakit.php)

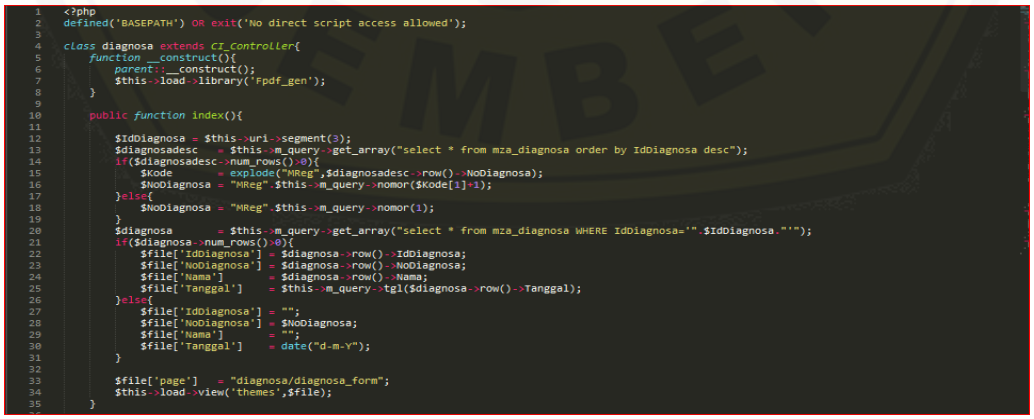
```php
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class penyakit extends CI_Controller{
    public function index(){
        $Search         = $this->m_query->replace($this->input->post("Search"));
        $file['qpenyakit']= $this->m_query->get_array("select * from mza_penyakit WHERE Penyakit like '%$Search%' order by IdPenyakit asc")->result();
        $file['page']   = "penyakit/penyakit";
        $this->load->view('themes',$file);
    }


    public function penyakit_form(){
        $IdPenyakit = $this->uri->segment(3);
        $penyakitdesc   = $this->m_query->get_array("select * from mza_penyakit order by IdPenyakit desc");
        if($penyakitdesc->num_rows()>0){
            $Kode       = explode("PA",$penyakitdesc->row()->KdPenyakit);
            $KdPenyakit = "PA".$this->m_query->nomor($Kode[1]+1);
        }else{
            $KdPenyakit = "PA".$this->m_query->nomor(1);
        }
        $penyakit       = $this->m_query->get_array("select * from mza_penyakit WHERE IdPenyakit='".$IdPenyakit."'");
        if($penyakit->num_rows()>0){
            $file['IdPenyakit'] = $penyakit->row()->IdPenyakit;
            $file['KdPenyakit'] = $penyakit->row()->KdPenyakit;
            $file['Penyakit']   = $penyakit->row()->Penyakit;
            $file['Deskripsi']  = $penyakit->row()->Deskripsi;
            //$file['Solusi']   = $penyakit->row()->Solusi;
        }else{
            $file['IdPenyakit'] = "";
            $file['KdPenyakit'] = $KdPenyakit;
            $file['Penyakit']   = "";
            $file['Deskripsi']  = "";
            //$file['Solusi']   = "";
        }

        $file['page']   = "penyakit/penyakit_form";
        $this->load->view('themes',$file);
    }

    public function penyakit_save(){
        $IdPenyakit = $this->m_query->replace($this->input->post("IdPenyakit"));
        $KdPenyakit = $this->m_query->replace($this->input->post("KdPenyakit"));
        $Penyakit   = $this->m_query->replace($this->input->post("Penyakit"));
        $Deskripsi  = $this->m_query->replace($this->input->post("Deskripsi"));
        $Solusi     = $this->m_query->replace($this->input->post("Solusi"));
        $Foto       = $_FILES['Foto'];
```



```
        $xu     = 0; $FotoUpload="";
        for($x=0; $x<count($_FILES['Foto']['name']); $x++){
            $Upload = "penyakit/".$Foto['name'][$x];
            if(move_uploaded_file($Foto['tmp_name'][$x], $Upload)){
                $xu++;
                if($xu==1){
                    $FotoUpload = "[".$Foto['name'][$x]."]";
                }else{
                    $FotoUpload = $FotoUpload.",[".$Foto['name'][$x]."]";
                }
            }
        }

        if($IdPenyakit==""){
            $query  = "Insert into mza_penyakit set KdPenyakit='$KdPenyakit', Penyakit='$Penyakit', Deskripsi='$Deskripsi',  Foto='".$FotoUpload."'";
        }else{
            $query  = "update mza_penyakit set KdPenyakit='$KdPenyakit', Penyakit='$Penyakit', Deskripsi='$Deskripsi', Foto='".$FotoUpload."' WHERE I
        }
        $this->m_query->get_save("$query");
        redirect("penyakit");
    }

    public function gejala(){
        $Search         = $this->m_query->replace($this->input->post("Search"));
        $file['qgejala']= $this->m_query->get_array("select * from mza_gejala WHERE Gejala like '%$Search%' order by IdGejala asc")->result();
        $file['page']   = "penyakit/gejala";
        $this->load->view('themes',$file);
    }

    public function gejala_form(){
        $IdGejala   = $this->uri->segment(3);
        $gejaladesc = $this->m_query->get_array("select * from mza_gejala order by IdGejala desc");
        if($gejaladesc->num_rows()>0){
            $Kode       = explode("PG",$gejaladesc->row()->KdGejala);
            $KdGejala   = "PG".$this->m_query->nomor($Kode[1]+1);
        }else{
            $KdGejala   = "PG".$this->m_query->nomor(1);
        }
        $gejala     = $this->m_query->get_array("select * from mza_gejala WHERE IdGejala='".$IdGejala."'");
        if($gejala->num_rows()>0){
            $file['IdGejala']   = $gejala->row()->IdGejala;
            $file['KdGejala']   = $gejala->row()->KdGejala;
            $file['Gejala']     = $gejala->row()->Gejala;
        }else{
            $file['IdGejala']   = "";
            $file['KdGejala']   = $KdGejala;
            $file['Gejala']     = "";
        }

        $file['page']   = "penyakit/gejala_form";
        $this->load->view('themes',$file);
    }

    public function gejala_save(){
        $IdGejala   = $this->m_query->replace($this->input->post("IdGejala"));
        $KdGejala   = $this->m_query->replace($this->input->post("KdGejala"));
        $Gejala     = $this->m_query->replace($this->input->post("Gejala"));

        if($IdGejala==""){
            $query  = "Insert into mza_gejala set KdGejala='$KdGejala', Gejala='$Gejala'";
        }else{
            $query  = "update mza_gejala set KdGejala='$KdGejala', Gejala='$Gejala' WHERE IdGejala='$IdGejala'";
        }
        $this->m_query->get_save("$query");
        redirect("penyakit/gejala");
    }


}
```

Gambar E.2Controller Penyakit dan Gejala (penyakit.php)

E.3 Controller Diagnosa (diagnosa.php)



```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class diagnosa extends CI_Controller{
    function __construct(){
        parent::__construct();
        $this->load->library('fpdf_gen');
    }

    public function index(){

        $IdDiagnosa = $this->uri->segment(3);
        $diagnosadesc   = $this->m_query->get_array("select * from mza_diagnosa order by IdDiagnosa desc");
        if($diagnosadesc->num_rows()>0){
            $Kode       = explode("MReg",$diagnosadesc->row()->NoDiagnosa);
            $NoDiagnosa = "MReg".$this->m_query->nomor($Kode[1]+1);
        }else{
            $NoDiagnosa = "MReg".$this->m_query->nomor(1);
        }
        $diagnosa       = $this->m_query->get_array("select * from mza_diagnosa WHERE IdDiagnosa='".$IdDiagnosa."'");
        if($diagnosa->num_rows()>0){
            $file['IdDiagnosa'] = $diagnosa->row()->IdDiagnosa;
            $file['NoDiagnosa'] = $diagnosa->row()->NoDiagnosa;
            $file['Nama']       = $diagnosa->row()->Nama;
            $file['Tanggal']    = $this->m_query->tgl($diagnosa->row()->Tanggal);
        }else{
            $file['IdDiagnosa'] = "";
            $file['NoDiagnosa'] = $NoDiagnosa;
            $file['Nama']       = "";
            $file['Tanggal']    = date("d-m-Y");
        }

        $file['page']   = "diagnosa/diagnosa_form";
        $this->load->view('themes',$file);
    }
```

```php
    public function diagnosa_save(){
        $IdDiagnosa = $this->m_query->replace($this->input->post("IdDiagnosa"));
        $NoDiagnosa = $this->m_query->replace($this->input->post("NoDiagnosa"));
        $Nama       = $this->m_query->replace($this->input->post("Nama"));
        $Tanggal    = $this->m_query->tgl($this->m_query->replace($this->input->post("Tanggal")));

        if($IdDiagnosa==""){
            $query = "insert into mza_diagnosa set NoDiagnosa='$NoDiagnosa', Nama='$Nama', Tanggal='$Tanggal'";
        }else{
            $query = "update mza_diagnosa set NoDiagnosa='$NoDiagnosa', Nama='$Nama', Tanggal='$Tanggal' WHERE IdDiagnosa='$IdDiagnosa'";
        }
        $this->m_query->get_save("$query");
        $diagnosa   = $this->m_query->get_array("select * from mza_diagnosa where NoDiagnosa='$NoDiagnosa'")->row();
        $IdDiagnosa = $diagnosa->IdDiagnosa;
        $this->m_query->get_save("delete from mza_diagnosadetail WHERE IdDiagnosa='$IdDiagnosa'");
        $gejala = $this->m_query->get_array("select g.IdGejala, g.KdGejala, g.Gejala from mza_gejala g order by g.IdGejala asc")->result();
        foreach($gejala as $key => $value){
            $IdGejala   = $value->IdGejala;
            $Status     = $this->input->post("IdGejala_".$value->IdGejala);
            if($Status=="Y"){ $Status = "Y"; }else{ $Status = "N"; }
            $this->m_query->get_save("insert into mza_diagnosadetail set IdDiagnosa='$IdDiagnosa', IdGejala='$IdGejala', Status='$Status'");
        }
        redirect("diagnosa/klasifikasi/".$IdDiagnosa);
    }

    public function klasifikasi($IdDiagnosa,$lap=null)
    {
        $diagnosa   = $this->m_query->get_array("select * from mza_diagnosa WHERE IdDiagnosa='".$IdDiagnosa."'")->row();
        $detail_diagnosa = $this->m_query->get_detail_diagnosa($IdDiagnosa);
        $penyakit = $this->m_query->get_penyakit();
        /*
            1. menghitung hasil bagi antara idGejala yg "ya" dengan jumlah dataset yang memiliki gejala yang di inputkan
            Loop i ---> P(idGejala = i | Y= ya) = hasil_bagi
        */
        foreach ($detail_diagnosa as $val) {
            foreach ($penyakit as $p) {
                $hasil_bagi[$val->IdGejala]['idPkt_'.$p->IdPenyakit] = $this->m_query->get_klasifikasi($val->IdGejala,$p->IdPenyakit)->hasil_bagi;
            }
        }
        //print_r($hasil_bagi);

        /*
            2. mengalikan semua hasil bagi sesuai jenis penyakit
        */
        foreach ($penyakit as $p) {
            $hasil_kali[$p->IdPenyakit] = array_product(array_column($hasil_bagi, 'idPkt_'.$p->IdPenyakit));
        }
        redirect('dataset');
    }

    public function laporan(){
        $file['data_diagnosa']  = $this->m_query->get_dt_diagnosa();
        $file['page']   = "diagnosa/laporan_diagnosa";
        $this->load->view('themes',$file);
    }


}
```

Gambar E.3Controller Diagnosa (diagnosa.php)

D.4 Controller Dataset (dataset.php)

```php
<?php
defined('BASEPATH') OR exit('No direct script access allowed');

class dataset extends CI_Controller {


    function __construct(){
        parent::__construct();
        $this->load->library(array('PHPExcel','PHPExcel/IOFactory'));
    }


    public function index(){
        $Search         = $this->m_query->replace($this->input->post("Search"));
        $file['qdataset']= $this->m_query->get_array("select ds.IdDataset, ds.NoDiagnosa, ds.IdPenyakit, p.Penyakit from mza_dataset ds inner join mza
        $file['page']   = "dataset/dataset";
        $this->load->view('themes',$file);
    }

    public function dataset_form(){
        $IdDataset  = $this->uri->segment(3);
        $datasetdesc    = $this->m_query->get_array("select * from mza_dataset WHERE NoDiagnosa like 'RD%' order by IdDataset desc");
        if($datasetdesc->num_rows()>0){
            $Kode       = explode("RD",$datasetdesc->row()->NoDiagnosa);
            $NoDiagnosa = "RD".$this->m_query->nomor($Kode[1]+1);
        }else{
            $NoDiagnosa = "RD".$this->m_query->nomor(1);
        }

        $dataset        = $this->m_query->get_array("select * from mza_dataset WHERE IdDataset='".$IdDataset."'");
        if($dataset->num_rows()>0){
            $file['IdDataset']  = $dataset->row()->IdDataset;
            $file['NoDiagnosa'] = $dataset->row()->NoDiagnosa;
            $file['IdPenyakit'] = $dataset->row()->IdPenyakit;
        }else{
            $file['IdDataset']  = "";
            $file['NoDiagnosa'] = $NoDiagnosa;
            $file['IdPenyakit'] = "";
        }
        $file['qpenyakit']= $this->m_query->get_array("select * from mza_penyakit order by IdPenyakit asc")->result();
        $file['page']   = "dataset/dataset_form";
        $this->load->view('themes',$file);
    }

```

```
    public function dataset_delete(){
        $IdDataset  = $this->uri->segment(3);
        $this->m_query->mquery("delete from mza_datasetdetail WHERE IdDataset='".$IdDataset."'");
        $this->m_query->mquery("delete from mza_dataset WHERE IdDataset='".$IdDataset."'");
        redirect("dataset");
    }

}

```

Gambar E.4Controller Dataset (dataset.php)

D.5 Koding model (m_query.php)

```
<?php
defined('BASEPATH') OR exit('No direct script access allowed');
class m_query extends CI_Model{
    function get_array($query){
        $result = $this->db->query("$query");
        return $result;
    }

    function get_save($query){
        $result = $this->db->query("$query");
        return $result;
    }

    function mquery($query){
        $this->db->query("$query");
    }

    function nomor($number){
        $Length = strlen($number);
        if($Length==1){ $Nol="00";
        }elseif($Length==2){ $Nol="0";
        }else{ $Nol=""; }
        return $Nol.$number;
    }

    function replace($text){
        return str_replace("'","''",$text);
    }

    function tgl($date){
        $Tgl = explode("-",$date);
        return $Tgl[2]."-".$Tgl[1]."-".$Tgl[0];
    }

    function convert_time($second){
        $SJ     = 3600;
        $SM     = 60;

        $J      = $second%$SJ;
        $SJam   = $second-$J;
        $Jam    = round($SJam/$SJ,3);

        $M      = $J%$SM;
        $SMenit = $J-$M;
        $Menit  = round($SMenit/$SM,3);

        if($Jam>0){
            $timer = $Jam." jam, ".$Menit." menit, ".$M." detik";
        }else if($Jam<=0 && $Menit>0){
            $timer = $Menit." menit, ".$M." detik";
        }else{
            $timer = round($second,3)." detik";
        }
        return $timer;
    }

    public function getRelasi($IdPenyakit, $IdGejala){
        $return = $this->db->query("SELECT * FROM mza_penyakitgejala WHERE IdPenyakit='".$IdPenyakit."' and IdGejala='".$IdGejala."'");
        return $return;
    }

    function get_dt_diagnosa()
    {
        $sql = "select * from mza_diagnosa WHERE status_simpan = 0";
        $query = $this->db->query($sql);
        return $query->result();
    }

    function get_dt_diagnosa_by_id($id)
    {
        $sql = "select * from mza_diagnosa WHERE IdDiagnosa = ".$id;
        $query = $this->db->query($sql);
        return $query->row();
    }

    function get_detail_diagnosa($id_diagnosa)
    {
        $sql = "SELECT mza_diagnosadetail.IdDiagnosa, mza_diagnosadetail.IdGejala,
                    mza_diagnosadetail.Status, mza_gejala.kdGejala, mza_gejala.Gejala
                FROM mza_diagnosadetail
                JOIN mza_gejala ON mza_gejala.IdGejala = mza_diagnosadetail.IdGejala
                WHERE `IdDiagnosa` = $id_diagnosa AND Status = 'Y'";
        $query = $this->db->query($sql);
        return $query->result();
    }

    function get_penyakit()
    {
        $sql = "SELECT * FROM mza_penyakit";
        $query = $this->db->query($sql);
        return $query->result();
    }
```

```php
    function get_penyakit_by_id($idP)
    {
        $sql = "SELECT * FROM mza_penyakit where idPenyakit = $idP";
        $query = $this->db->query($sql);
        return $query->row();
    }

    function get_klasifikasi($id_gejala,$id_penyakit)
    {
        $sql = "
            SELECT
                (
                    SELECT COUNT(b.IdPenyakit)
                    FROM mza_datasetdetail a
                    JOIN mza_dataset b ON b.IdDataset = a.IdDataset
                    JOIN mza_penyakit c ON c.IdPenyakit = b.IdPenyakit
                    JOIN mza_gejala d ON d.IdGejala = a.IdGejala
                    where a.IdGejala =".$id_gejala." AND a.Status = 'Y' AND b.IdPenyakit = ".$id_penyakit."
                ) / COUNT(mza_datasetdetail.IdGejala) as hasil_bagi
            FROM mza_datasetdetail
            JOIN mza_dataset ON mza_dataset.IdDataset = mza_datasetdetail.IdDataset
            JOIN mza_penyakit ON mza_penyakit.IdPenyakit = mza_dataset.IdPenyakit
            JOIN mza_gejala ON mza_gejala.IdGejala = mza_datasetdetail.IdGejala
            WHERE mza_datasetdetail.IdGejala =".$id_gejala;
        $query = $this->db->query($sql);
        return $query->row();
    }

    function get_detail_dataset_by_idDtSet($idDataset)
    {
        $sql = "
            SELECT mza_datasetdetail.IdDataset, mza_datasetdetail.IdGejala, mza_datasetdetail.Status, mza_gejala.KdGejala, mza_gejala.Gejala
            FROM mza_datasetdetail
            JOIN mza_gejala ON mza_datasetdetail.IdGejala = mza_gejala.IdGejala
            WHERE mza_datasetdetail.Status = 'Y' AND mza_datasetdetail.IdDataset = ".$idDataset;
        $query = $this->db->query($sql);
        return $query->result();
    }

    function get_dataset_by_NoDiagnosa($noD)
    {
        $sql = "SELECT * FROM `mza_dataset` WHERE `NoDiagnosa` = '$noD'";
        $query = $this->db->query($sql);
        return $query->row();
    }

    function insert_dataset($data)
    {
        $this->db->insert('mza_dataset', $data);
    }

    function update_datadiagnosa($idDiagnosa)
    {
        $this->db->set('status_simpan', 1);
        $this->db->where('IdDiagnosa', $idDiagnosa);
        $this->db->update('mza_diagnosa');
    }

    function insert_datasetDetail($iddataset,$idgejala,$status)
    {
        $sql = "INSERT INTO `mza_datasetdetail` (`IdDataset`, `IdGejala`, `Status`) VALUES ('$iddataset', '$idgejala', '$status')";
        $this->db->query($sql);
    }

}
?>
```

Gambar E.5Koding Model (m_query.php

F. Testing

F.1 Pengujian *Black Box*

Tabel 4. 12Pengujian Black Box

No	Menu	Fungsi	Aksi	Hasil	Ket
1.	Login (admin)	Menu login merupakan menu awal yang digunakan oleh Admin untuk masuk kedalam sistem sesuai hak aksesnya.	Mengisi *username* dan *password* lalumenekan tombol *"Login"*.	*Login* berhasil menampilkan halaman awal actor	√
			Username atau *Password* salah	Menampilkan pop up login dengan form dengan *field username, password,*dan tombol *submit*	√
			Username atau *Password* kosong	Menampilkan pop up login dengan form dengan *field username, password,*dan tombol *submit*	√
2.	Gejala (admin)	Menu gejala merupakan menu yang terdapat pada sistem pakar unutk mendiagnosa penyakit ayam untuk menambah, mengubah data gejala.	Memilih menu gejala	Mengambil data dari *database* mza_gejala (idGejala, KdGejala, Gejala).	√
				Menampilkan data Gejala dari mza_gejala (No, Kode Gejala, Gejala, Tambah Gejala ,Action).	√
			Klik tombol "Tambah Gejala".	Menampilkan *form* isi tambah data gejala penyakit ayam (kode gejala, gejala, action)	√
			Klik tombol "Submit"	Menyimpan data masukan data ke database (idGejala, KdGejala, Gejala).	√
			Field belum diisi saat menambah data gejala, dan menekan tombol "submit".	Menampilkan pesan "*field* harus diisi" dibawah *field* yang kosong	√
			Klik tombol "Back" saat menambah data.	Menampilkan data Gejala dari mza_gejala (No, Kode Gejala, Gejala, Tambah Gejala ,Action).	√

			Klik Tombol "*Edit*" pada salah satu data gejala pada halaman gejala.	Mengambil data gejala penyakit ayam yang dipililih dari database mza_gejala (IdGejala, KdGejala, Gejala)	√
				Menampilkan data gejala penyakit ayam ke dalam *formedit* gejala penyakit ayam (Kode Gejala, Gejala, Action)	√
			Field belum diisi saat merubah data gejala, dan menekan tombol "*submit*".	Menampilkan pesan "*field* harus diisi" dibawah *field* yang kosong	√
			Klik tombol "Back" saat merubah data	Menampilkan data gejala pada tabel di halaman gejala penyakit ayammza_gejala (IdGejala, KdGejala, Gejala)	√
3.	Penyakit (admin)	Menu penyakit merupakan menu yang terdapat pada sistem pakar untuk mendiagnosa penyakit ayam untuk menambah, mengubahdata penyakit ayam.	Memilih menupenyakit.	Mengambil data dari *database* mza_penyakit (IdPenyakit, KdPenyakit, Penyakit, Deskripsi, Foto).	√
				Menampilkan data penyakitdari mza_penyakit (no, Kode Penyakit, Penyakit, action).	√
			Klik tombol "Tambah Penyakit".	Menampilkan *form* isi tambah data penyakit (no, nama jeruk, gambar, ciri jeruk, manfaat jeruk).	√
			Klik tombol "Submit"	Menyimpan data masukan data ke database (kode penyakit, penyakit, deskripsi, Foto)	√
			Field belum diisi saat menambah data penyakit, dan menekan tombol "submit".	Menampilkan pesan "*field* harus diisi" dibawah *field* yang kosong	√
			Klik tombol "Back" saat menambah data.	Menampilkan data penyakit dari mza_penyakit (no, Kode Penyakit,	√

				Penyakit, action).	
			Klik Tombol "*Edit*" pada salah satu data penyakit pada halaman penyakit.	Mengambil data penyakityang dipililih dari database mza_penyakit(IdPenyakit, KdPenyakit, Penyakit, Deskripsi, Foto).	√
				Menampilkan data penyakit ke dalam *formedit*penyakit(Kode penyakit, penyakit, deskripsi, Foto).	√
			Field belum diisi saat merubah data penyakit, dan menekan tombol "*submit*".	Menampilkan pesan "*field* harus diisi" dibawah *field* yang kosong	√
			Klik tombol "Back" saat merubah data penyakit	Menampilkan data penyakit dari mza_penyakit (no, Kode Penyakit, Penyakit, action).	√
4.	Dataset (admin)	Menu dataset merupakan menu yang terdapat pada sistem pakar untuk mendiagnosa penyakit ayam untuk mengubah, dan menghapus data dataset.	Memilih menudataset.	Menampilkan data datasetmza_Dataset, mza_datasetdetail (no dataset, Gejala, Penyakit, Action)	√
			Klik Tombol "*Edit*" pada salah satu data dataset pada halaman dataset.	Menampilkan datadataset ke dalam *formedit* (no dataset, nama penyakit, gejala berserta pilihan "ya" atau "Tidak")..	√
			Klik tombol "Batal" saat merubah data	Menampilkan data datasetmza_Dataset, mza_datasetdetail (no dataset, Gejala, Penyakit, Action)	√
			Klik Tombol "Delete" pada salah satu data dataset pada halaman dataset	Menampikan alert "Apakah anda yakin akan menghapus data ini?"	√
			Klik Tombol "Ya" saat menghapus data.	Menghapus data dataset dari database sesuai id yang dipilih.	√
			Klik Tombol "cancel" saat menghapus data.	Menampilkan data datasetmza_Dataset, mza_datasetdetail (no dataset, Gejala, Penyakit, Action)	√

5.	Laporan Diagnosa (admin)	Menu laporan diagnosa merupakan menu yang terdapat pada sistem pakar untuk mendiagnosa penyakit ayam untuk menampilkan hasil diagnosa yang telah dilalukan user atau peternak, menambah dataset.	Memilih menuLaporan diagnosa.	Menampilkan data hasil diagnosa (tanggal diagnosa, No.Registrasi, action)	√
			Klik Tombol "Simpan Dataset" pada salah satu data hasil diagnosa pada halaman laporan diagnosa.	Menyimpan dan menampilkan halaman dataset yang berisi (No dataset, Gejala, penyakit,action).	√
			Klik Tombol "Report Detail" pada salah satu data hasil diagnosa pada halaman laporan diagnosa	Menampikan hasil diagnosa yang berisi tentang penjelasan penyakit dan action (Detail dan back)	√
			Klik Tombol "Detail" di laporan diagnosa	Menampilkan detail hasil diagnosa yang berisi (kode diagnosa, tanggal, gejala yang dipilih, hasil perhitungan naive bayes, nilai probabilitas tertinggi).	√
			Klik Tombol "Back" di laporan diagnosa	Menampilkan data hasil diagnosa (tanggal diagnosa, No.Registrasi,action)	√
6.	Diagnosa (user atau peternak)	Menu diagnosa penyakit ayam merupakan menu yang terdapat pada sistem pakar untuk mendiagnosa penyakit ayam.	Klik menu diagnosa	Menampilkan gejala penyakit ayam (No diagnosa, tanggal diagnosa, gejala penyakit ayam)	√
			Klik tombol "Diagnosa"	Menampilkan halaman hasil diagnosa yang berupa penjelsan penyakit yang terdiagnosa, tombol detail.	√
			Klik tombol "Detail" pada halaman hasil diagnosa	Menampilkan pop up"Detail hasil Diagnosa" yang berisi (Kode diagnosa, tanggal diagnosa, gejala yang dipilih, hasil perhitungan naive bayes, hasil perhitungan probabilitas tertinggi).	√
7.	Penyakit (user atau peternak)	Menu penyakit pada hak akses user atau peternak merupakan menu yang terdapat pada sistem pakar untuk mendiagnosa	Klik menu penyakit	Mengambil data penyakit dari *database* mza_penyakit (no, kode penyakit, nama penyakit).	√

		penyakit ayam untuk melihat data penyakit ayam.			

G. Transkip Wawancara

No.	Pertanyaan	SP	P	CP	TP	STP
1.	Apakah anda paham manfaat dan kegunaan dari sistem pakar ini?					
2.	Apakah anda paham kenapa peternakan ini di buatkan sistem pakar?					
3.	Apakah anda sudah paham cara penggunaan sistem pakar ini?					
4.	Apakah anda paham penjelasan solusi yang di berikan sistem pakar?					
5.	Apakah anda paham dengan kalimat ini? "Apakah anda setuju untuk mendiagnosa penyakit pada ayam menggunakan sistem pakar ini?"					
6.	Apakah anda paham dengan kalimat ini "Apakah sistem pakar ini cukup membantu peternak mengidentifikasi penyakit pada ayam?"					
7.	Apakah anda paham dengan kalimat ini "Apakah sistem pakar ini mudah di gunakan dan dimengerti?"					
8.	Apakah anda paham dengan kalimat ini "Apakah sistem pakar ini sudah cukup jelas dalam memberikan solusi terhadap penyakit yang sudah teridentifikasi?"					
9.	Apakah anda paham dengan kalimat ini "Apakah sistem pakar ini bermanfaat bagi peternak ayam?"					
10.	Apakah anda paham dengan kalimat ini "Apakah dengan menggunakan sistem pakar untuk mendiagnosa penyakit ayam dapat dengan mudah mengidentifikasi penyakit pada ayam di banding tanpa sistem?"					

Keterangan :

a. Skor 1 = Sangat Tidak Paham (STP)
b. Skor 2 = Tidak Paham (TP)
c. Skor 3 = Cukup Paham (CP)
d. Skor 4 = Paham (P)
e. Skor 5 = Sangat Paham (SP)