**BAB II**

**LANDASAN TEORI**

**2.1. Tinjauan Pustaka**

**A. Sistem Pakar**

 Menurut Yakub (2012:86) “Sistem pakar (*expert system*/ES) yaitu program komputer yang memiliki fungsi sama dengan keahlian manusia untuk menyediakan pemecahan masalah”.

Sedangkan menurut Merlina, Nita dan Rahmat (2012:1) “Sistem pakar adalah suatu program komputer yang dirancang untuk memodelkan kemampuan penyelesaian masalah yang dilakukan seorang pakar”.

 Menurut Hayadi, B, Herawan (2016:3) menyimpulkan bahwa ciri-ciri sistem pakar adalah sebagai berikut:

1. Terbatas pada domain keahlian tertentu.
2. Dapat memberikan penalaran untuk data yang tidak pasti.
3. Dapat mengemukakan rangkaian alasan yang diberikannya dengan cara yang dapat dipahami.
4. Berdasarkan pada kaidah atau rule tertentu.
5. Dirancang untuk dapat dikembangkan secara bertahap.
6. Pengetahuan dan mekanisme inferensi jelas terpisah.
7. Keluarannya bersifat anjuran.
8. System dapat mengaktifkan kaidah secara searah yang sesuai yang dituntun oleh dialog dengan pemakai.

Menurut Hayadi, B, Herawan (2016:2) menyatakan bahwa sistem pakar menjadi sangat popular karena sangat banyak kemampuan dan manfaat yang diberikannya T. Sutojo, e.t. 2010, diantaranya:

1. Meningkatkan produktivitas, karena sistem pakar dapat bekerja lebih cepat daripada manusia.
2. Membuat seseorang yang awam bekerja seperti layaknya seorang pakar.
3. Meningkatkan kualitas, dengan memberi nasehat yang konsisten dan mengurangi kesalahaan.
4. Mampu menangkap pengetahuan dan kepakaran seseorang.
5. Memudahkan akses pengetahuan seorang pakar.
6. Bias digunakan sebagai media pelengkap dalam pelatihan. Pengguna pemula yang bekerja dengan sistem pakar akan menjadi lebih berpengalaman karena adanya fasilitas penjelas yang berfungsi sebagai guru.
7. Meningkatkan kemampuan untuk menyelesaikan masalah karena system pakar mengambil sumber pengetahuan dari banyak pakar.

Menurut Hayadi, B, Herawan (2016:3) menyatakan bahwa selain manfaat, ada juga beberapa kekurangan yang ada pada sistem pakar, diantaranya:

1. Biaya yang sangat mahal untuk membuat dan memeliharanya.
2. Sulit dikembangkan karena keterbatasan keahlian dan ketersediaan pakar.
3. Sistem pakar tidak 100% bernilai benar.

**B. Metode Inferensi (*Inference Method*)**

Menurut Kusrini (2008:8) “*Inferensi* merupakan proses untuk menghasilkan informasi dari fakta yang diketahui atau di asumsikan. Inferensi adalah konklusi logis (*logical conclusion*) atau implikasi berdasarkan informasi yang tersedia. Dalam system pakar proses inferensi dilakukan dalam suatu modul yang disebut *Inference Engine* (Mesin Inferensi)”.

 Menurut Kusrini (2008:8) “Ada dua pendekatan dalan menentukan metode inferensi yang penting dalam sistem pakar yaitu runut maju (*forward chaining*) dan runut balik (*backward chaining*)”.

1. Runut Maju (*Forward Chaining*)

Menurut Kusrini (2008:8) “Runut Maju berarti menggunakan himpunan aturan kondisi-aksi. Dalam metode ini, data diguanakan untuk menentukan aturan mana yang akan dijalankan, kemudian aturan tersebut dijalankan. Mungkin proses menambahkan data ke memori kerja. Proses diulang sampai ditemukan suatu hasil (Wilson, 1998)”.

Sedangkan menurut Merlina, Nita dan Rahmat (2012:22) “*Forward chaining* adalah pendekatan *data-driven* yang dimulai dari informasi yang tersedia atau dari ide dasar, kemudian mencoba menarik kesimpulan”.

 R-9 R-10

Fakta R-4

 R-5

 R-3

R-6

Fakta

Sumber (Merlina, Nita dan Rahmat, 2012:22)

Gambar II.1 Cara Kerja Mesin Inferensi *Forward Chaining*

2. Runut Balik (*Backward Chaining*)

 Menurut Kusrini (2008:11) “Runut balik merupakan metode penalaran kebalikan dari runut maju. Dalam runut balik penalaran dimulai dengan tujuan kemudian merunut balik ke jalur yang akan mengarahkan ke tujuan tersebut (Giarattano dan Riley, 1994)”.

 Sedangkan menurut Merlina, Nita dan Rahmat (2012:21) “*Backward chaining* adalah pendekatan *goal-driven* yang dimulai dari harapan apa yang akan terjadi (hipotesis) dan kemudian mencari bukti yang mendukung (atau berlawanan) dengan harapan”.

 R-10

 R-8 R-7 R-1

Pertama : Gagal

 R-10 R-9 R-4

Kedua : Sukses

Sumber (Nita dan Rahmat, 2012:22)

Gambar II.2 Cara Kerja Mesin Inferensi *Backward Chaining*

**C. Model Pengembangan Sistem**

**Model Air Terjun (*Water Fall*)**

Menurut Rosa dan Shalahuddin (2013:28) “Model air terjun menyediakan pendekatan alur hidup perangkat lunak secara sekuensial atau terurut dimulai dari analisis desain pengodean, pengujian dan tahap pendukung (*support*)”.

Analisis

Desain

Pengodean

Pengujian

Sumber (Rosa dan Shalahuddin, 2013:28)

Gambar II.3 Ilustrasi model *waterfall*

Menurut Rosa dan Shalahuddin (2013:29) adapun tahapannya sebagai berikut

1. Analisis kebutuhan perangkat lunak

Proses pengumpulan kebutuhan dilakukan secara intensif untuk mespesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh *user*. Spesifikasi kebutuhan perangkat lunak pada tahap ini perlu didokumentasikan. (Rosa dan Shalahuddin, 2013:29)

1. Desain

Desain perangkat lunak adalah proses multi langkah yang fokus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antarmuka dan prosedur pengodean. Tahap ini mentranslasi kebutuhan perangkat lunak dari tahap analisis kebutuhan ke representasi desain agar dapat diimplementasikan menjadi program pada tahap selanjutnya. Desain perangkat lunak yang dihasilkan pada tahap ini perlu didokumentasikan. (Rosa dan Shalahuddin, 2013:29)

1. Pembuatan kode program

Desain harus ditranslasikan ke dalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai dengan desain yang telah dibuat pada tahap desain. (Rosa dan Shalahuddin, 2013:30)

1. Pengujian

Pengujian fokus pada perangkat lunak secara dari segi lojik dan fungsional dan memastikan bahwa semua bagian sudah diuji. Hal ini dilakukan untuk meminimalisir kesalahan (*error*) dan memastikan keluaran yang dihasilkan sesuai dengan yang diinginkan. (Rosa dan Shalahuddin, 2013:30)

1. Pendukung (*support*) atau pemeliharaan (*maintenance*)

Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah dikirimkan ke *user.* Perubahan bisa terjadi karena adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian atau perangkat lunak harus diadaptasi dengan lingkungan baru. Tahap pendukung atau pemeliharaan dapat mengulangi proses pengembangan mulai dari analisis spesifikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidah untuk membuat perangkat lunak baru. (Rosa dan Shalahuddin, 2013:30)

**D. Bahasa Permodelan Sistem**

**UML (*Unified Modeling Language*)**

Menurut Rosa dan Shalahuddin (2013:133) “UML (*Unified Modeling Language*) adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement*, membuat analis & desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek”.

Menurut Rosa dan Shalahuddin (2013:134) berikut ini merupakan macam-macam diagram UML :

A. *Use Case* *Diagram*

 *Use case* atau diagram use case merupakan pemodelan untuk kelakuan (*behaviour*) sistem informasi yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. (Rosa dan Shalahuddin, 2013:134)

B. *Activity* *Diagram*

 Diagram aktivitas atau *activity* *diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, melainkan yang dilakukan oleh sistem. (Rosa dan Shalahuddin, 2013:134)

C. *Sequence* *Diagram*

 Diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek. Dalam membuat diagram sekuen harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. (Rosa dan Shalahuddin, 2013:135)

D. *Class* *Diagram*

 Diagram kelas atau *class* *diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Diagram kelas dibuat agar pembuat program dan *programmer* membuat kelas-kelas sesuai rancangan di dalam diagram kelas agar antara dokumentasi perancangan dan perangkat lunak sinkron. (Rosa dan Shalahuddin, 2013:135)

E. *Component* *Diagram*

 Diagram komponen atau *component* *diagram* dibuat untuk menunjukkan organisasi dan ketergantungan diantara kumpulan komponen dalam sebuah sistem. Diagram komponen fokus pada komponen sistem yang dibutuhkan dan ada di dalam sistem. (Rosa dan Shalahuddin, 2013:136)

F. *Deployment* *Diagram*

 Diagram *deployment* atau *deployment diagram* menunjukan konfigurasi komponen dalam proses eksekusi aplikasi. Diagram *deployment* juga dapat digunakan untuk memodelkan sistem tambahan (*embedded system*) yang menggambarkan rancangan *device, node* dan *hardware*, sistem *client server,* sistem terdistribusi dan rekayasa ulang aplikasi. (Rosa dan Shalahuddin, 2013:136)

**E. Desain Database**

**ERD (*Entity Relationship Diagram*)**

A. Pengertian ERD

Menurut Rosa dan Shalahuddin (2010:212) “ERD (*Entity Relationship Diagram*) merupakan sebuah diagram yang digunakan untuk merancang hubungan antar tabel-tabel dalam basis data”.

Menurut Rosa dan Shalahuddin (2010:212) Simbol-simbol yang boleh digunakan adalah:

1. Persegi Panjang, berungsi untuk menyatakan suatu *entity*.
2. *Elips*, berfungsi untuk menyatakan *attribute*, jika diberi garis bawah menandakan bahwa *attribute* tersebut merupakan *attribute*/field kunci.
3. Belah ketupat, menyatakan jenis relasi
4. Garis, penghubungan antara relasi dengan *entity* dan antara *entity* dengan *attribute*.

B. Jenis Kerelasian Antar Entitas *( Relationship)*

Menurut Silberschatz dalam Sutanta (2011:102) kerelasian antar entitas dikelompokkan dalam tiga jenis:

1. Kerelasian jenis 1 ke 1 *(one to one)*

Kerelasian jenis ini terjadi jika kejadian atau transaksi diantara dua entitas yang berhubungan hanya memungkinkan terjadi sebuah kejadian atau transaksi pada kedua entitas. Secara lebih teknis, jika nilai yang digunakan sebagai penghubung pada entitas pertama hanya dimungkinkan muncul satu kali saja pada entitas kedua yang saling berhubungan. (Silberschatz dalam Sutanta, 2011:102)

2. Kerelasian jenis m ke 1 *(many to one)* atau 1 ke m *(one to many)*

Kerelasian jenis ini terjadi jika kejadian atau transaksi diantara dua entitas yang berhubungan hanya memungkinkan terjadi satu kali dalam entitas pertama dan dapat terjadi lebih dari satu kali kejadian atau transaksi pada entitas kedua. Secara lebih teknis, jika nilai yang digunakan sebagai penghubung pada entitas pertama dimungkinkan muncul lebih dari satu kali pada entitas kedua yang berhubungan. (Silberschatz dalam Sutanta, 2011:102)

3. Kerelasian jenis m ke m *(many to many)*

Kerelasian jenis ini terjadi jika kejadian atau transaksi diantara dua entitas yang berhubungan memungkinkan terjadi lebih dari satu kali dalam entitas pertama dan entitas kedua. Secara lebih teknis, jika nilai yang digunakan sebagai penghubung pada entitas pertama dimungkinkan muncul lebih dari satu kali, baik pada entitas pertama maupun pada entitas kedua yang saling berhubungan, dan sebaliknya. (Silberschatz dalam Sutanta, 2011:102)

**F. Database Management System (DBMS)**

 Menurut Yanto (2016:16) “Database Management System (DBMS) merupakan paket program (*Software*) yang dibuat agar memudahkan dan mengefisienkan pemasukan, pengeditan, penghapusan dan pengambilan informasi terhadap databse. *Software* yang tergolong kedalam DBMS antara lain, *Microsoft SQL, MySQL, Oracle, MS.Access,* dan lain-lain”.

**G. Konsep Dasar Pemrograman**

Menurut Sahyar (2016:2) “Program komputer adalah perintah – perintah atau instruksi yang disusun berdasarkan algoritma dengan menggunakan Bahasa pemograman untuk penyelesaian suatu masalah”.

Menurut Rosa dan Shalahuddin (2013:67) “Pemrograman terstruktur adalah konsep atau paradigma atau sudut pandang pemrograman yang membagi-bagi program berdasarkan fungsi-fungsi atau prosedur-prosedur yang dibutuhkan program komputer. Ciri Teknik Pemrograman Terstruktur antara lain:

1. Mengandung algoritma pemecahan masalah yang tepat, benar, sederhana, standar dan efektif.
2. Memiliki struktur logika dan struktur program yang benar dan mudah dipahami serta menghindari penggunaan instruksi GOTO.
3. Membutuhkan biaya testing, pemeliharaan dan pengembangan yang rendah.
4. Memiliki dokumentasi yang baik.

**H. PHP**

Menurut Sibero (2012:49), “PHP (*Personal Home Page*) adalah pemograman (interpreter) adalah proses penerjemahan baris sumber menjadi kode mesin yang di mengerti komputer secara langsung pada saat baris kode dijalankan”.

Menurut Kustiyahningsih (2011:114), “PHP (atau resminya PHP: Hypertext Preprosesor) adalah skrip bersifat *server-side* yang ditambahkan ke dalam HTML. PHP sendiri merupakan singkatan dari *Personal Home Page Tools*. Skrip ini akan membuat suatu aplikasi dapat diintegrasikan ke dalam HTML, sehingga suatu halaman web tidak lagi bersifat statis, namun menjadi bersifat dinamis. Sifat *server side* berarti pengerjaan kode program dilakukan di *server*, baru kemudian hasilnya dikirimkan ke *browser*”.

Menurut Supono dan Vidiandry Putratama (2016:3) menyimpulkan bahwa:

PHP (PHP: *Hypertext Preprocessor*) adalah suatu bahasa pemograman yang digunakan untuk menerjemahkan baris kode program menjadi kode mesin yang dapat dimengerti oleh komputer yang bersifat *server-side* yang dapat ditambahkan ke dalam HTML.

**I. CSS**

Menurut Sulistyawan, Rubianto dan Rahmad Saleh (2008:32), “(CSS) *Cascading Style Sheets* adalah suatu bahasa *stylesheet* yang digunakan untuk mengatur *style* suatu dokumen. Pada umumnya CSS dipakai untuk memformat tampilan halaman web yang dibuat dengan bahasa HTML dan XHTML”.

**2.2. Penelitian Terkait**

Menurut Christoffel dkk (2017:1) Sebagian besar dari pecinta anjing tidak mengetahui penyakit apa yang sedang dialami oleh anjing mereka, bahkan banyak dari mereka yang tidak tahu bahwa anjing mereka sedang sakit, yang akibatnya berujung pada kematian karena penanganan yang terlambat. Oleh sebab itu penelitian ini untuk membantu pengguna dalam mendapatkan informasi tentang pengenalan penyakit anjing dan cara penanganan awal penyakit pada anjing tersebut agar anjing kesayangan pengguna dapat sembuh dari penyakit. Metode yang digunakan yaitu *Forward Chaining*. Hasil dari penelitian ini yaitu untuk membantu masyarakat dan pecinta anjing di kota Manado dalam mendapatkan informasi penyakit anjing dan cara penanganannya.

Menurut KR Witari dkk (2013:45) Permasalahan yang diambil dalam pembuatan sistem pakar pendiagnosa penyakit menular pada anjing berdasarkan ciri-ciri fisik atau gejala – gejala yang dapat dilihat atau dirasakan tanpa melalui pemeriksaan spesifik, dengan menggunakan metode *Forward Chaining dan Certainty Factor*. Dengan tujuan membangun suatu sistem pakar berbasis online untuk mendiagnosa penyakit menular pada anjing melalui perangkat android. Yang dimana hasil nya dapat sebagai penanganan dini terhadap penyakit tersebut dan mengurangi resiko penyakit tersebut dapat berdampak buruk bagi manusia.