

BAB II

TINJAUAN PUSTAKA

2.1 Penyakit Tanaman Karet

Karet merupakan salah satu produk utama perkebunan Indonesia. Tanaman karet (*Hevea brasiliensis*) merupakan tanaman perkebunan yang tumbuh di berbagai wilayah Indonesia dan merupakan tanaman semusim. Perkembangan perkebunan karet mempunyai peranan penting dalam perekonomian nasional, terutama sebagai sumber devisa negara dan bahan baku industri, serta berperan dalam menjaga lingkungan hidup. Di beberapa daerah, karet ditanam per kapita dan merupakan tanaman pendapatan penting bagi keluarga (Sulistiani dan Muludi, 2018). Penyakit tanaman dapat menghambat pertumbuhan dan merusak bagian-bagian tanaman, secara langsung mengurangi hasil atau secara tidak langsung mematikan tanaman. Permasalahan yang muncul di kalangan petani adalah kurangnya pengetahuan terhadap gejala penyakit yang sering menyerang tanaman karet. Penanganan yang cepat dan hati-hati diperlukan untuk menghindari downtime produksi. Penyakit tanaman karet yang menimbulkan kerugian besar biasanya disebabkan oleh jamur. (Arifsyah, 2019)

Penyebab buruknya kualitas karet adalah adanya serangan penyakit yang seringkali menimbulkan kerugian besar pada pabrik karet. Kerugiannya bisa mencapai jutaan rupee per hektar perkebunan karet setiap tahunnya. Jamur merupakan penyebab penyakit yang banyak ditemukan pada tanaman karet. Sebaliknya, bakteri dan virus jarang terjadi dan tidak menyebabkan kerusakan berarti.

Untuk meningkatkan produksi pohon karet perlu diketahui kondisi tanaman karet, cara tumbuhnya, dan terserang penyakit atau tidak. Hal ini berguna untuk menerapkan teknik pengendalian penyakit di pabrik karet. Tanaman yang sakit sebenarnya merupakan hasil interaksi antara faktor pendukung yaitu tanaman inang, lingkungan dan patogen yang dikenal dengan segitiga penyakit. Patogen adalah mikroorganisme patogen. Berbagai jenis mikroorganisme diketahui menyebabkan penyakit tanaman yang merugikan karena tanaman karet didominasi

oleh kelompok jamur, bakteri dan virus. Sedangkan gangguan fisiologis umumnya disebabkan oleh kekurangan unsur mikro dan makro, serta keracunan yang dapat menghambat pertumbuhan tanaman.

2.2 Sistem Pakar

Sistem pakar adalah suatu sistem komputer yang menggunakan fakta, penalaran, dan pengetahuan manusia untuk memecahkan masalah layaknya seorang pakar atau pakar di bidangnya. Ketika sistem pakar dibuat dan dikembangkan, mereka menggabungkan prinsip-prinsip inferensi dengan basis pengetahuan spesifik yang disediakan oleh satu atau lebih pakar di bidang tertentu. Kombinasi keduanya disimpan di komputer dan kemudian digunakan dalam proses diagnosis dan pengambilan keputusan untuk memecahkan masalah tertentu.

Sistem pakar diimplementasikan untuk mendukung aktivitas debugging. Kegiatan pemecahan masalah yang terkait meliputi pengambilan keputusan, manajemen pengetahuan, perancangan, perencanaan, peramalan, pengaturan, pengendalian, penjelasan), diagnosis (*diagnosis*), perumusan (*prescribing*), konsultasi (*konsultan*), dan pelatihan (*mentoring*). Selain itu, sistem pakar juga dapat berperan sebagai asisten cerdas bagi pakar. Sistem pakar pertama kali dikembangkan oleh komunitas AI pada tahun 1960an. Sistem pakar pertama yang dibuat disebut General Purpose Problem Solver (GPS) dan dikembangkan oleh Newel Simon (Pratiwi, 2019)

2.2.1 Pakar

Pakar adalah seseorang yang memiliki pengetahuan tertentu dan mampu menjelaskan suatu tanggapan, mempelajari hal-hal baru seputar topik permasalahan, menyusun kembali pengetahuan-pengetahuan yang didapatkan dan dapat memilah aturan serta menentukan relevan kepakarannya. Seorang pakar memiliki kemampuan kepakaran, yaitu :

1. Dapat mengenali dan merumuskan suatu masalah.
2. Dapat menyelesaikan masalah dengan cepat dan tepat.
3. Menjelaskan solusi dari suatu masalah.
4. Restrukturisasi pengetahuan.

5. Belajar dari pengalaman.
6. Memahami batas kemampuan.

Selain itu pakar juga memiliki kemampuan untuk mengaplikasikan pengetahuannya dan memberikan saran serta pemecahan masalah pada domain tertentu. Ini merupakan pekerjaan pakar, memberikan pengetahuan tentang bagaimana seseorang melaksanakan tugas untuk menyelesaikan masalah (Yulian, 2021)

2.2.2 Kelebihan dan kekurangan Sistem Pakar

Kelebihan sistem pakar, yaitu:

1. Informasi apapun yang diterima tetap dapat membuat sistem pakar bekerja
2. Sistem Pakar dapat bekerja lebih cepat sehingga meningkatkan produktivitas.
3. Sistem Pakar selalu aktif dan konsisten dalam memberikan jawaban dan perhatian terhadap hasil dari masukan pengguna.
4. Sistem Pakar dapat menjangkau jarak yang luas dan jauh. Dengan menggunakan sistem pakar, pengguna seolah-olah berkonsultasi langsung dengan si pakar. Meskipun si pakar telah tiada.
5. Sistem Pakar memiliki kemampuan dalam memecahkan masalah yang kompleks dan rumit yang hanya dikuasai oleh si pakar.

Sedangkan, Kekurangan dari penggunaan sistem pakar yaitu:

1. Sistem Pakar hanya dapat menangani pengetahuan yang sudah dimasukkan ke dalam sistem dan hasilnya sudah pasti sesuai dengan alur inferensi yang dimasukkan. Agar dapat bersifat dinamis dan berubah dari waktu ke waktu maka basis pengetahuan harus selalu diperbaharui.
2. Sistem Pakar hanya menangani hal yang bersifat pasti berupa saran ataupun rekomendasi, bukan bersifat keputusan.
3. Format basis pengetahuan bersifat terbatas dan berisi aturan-aturan yang ditulis dalam bentuk pernyataan *if then*. (Pratiwi, 2019)

2.2.3 Karakteristik Sistem Pakar

Karakteristik sistem pakar yaitu:

- a. Kinerja sistem pakar sesuai dengan kemampuan khusus/khusus sistem pakar.
- b. Pengetahuan sistem pakar adalah sebuah konsep dan bukan bentuk numerik. Hal ini karena komputer memproses data secara numerik, sedangkan pengetahuan pakar didasarkan pada fakta dan prinsip, bukan angka.
- c. Informasi dalam sistem pakar tidak selalu lengkap, subyektif, tidak konsisten, materi terus berubah dan bergantung pada kondisi lingkungan, pengambilan keputusan yang tidak pasti dan bukan jawaban “ya” atau “tidak” yang mutlak, tetapi sesuai dengan derajat kebenaran tertentu.
- d. Kemungkinan penyelesaian suatu masalah dalam sistem pakar bervariasi dan mencakup banyak pilihan jawaban yang dapat diterima.
- e. Perubahan atau kemajuan pengetahuan pada sistem pakar dapat terjadi kapan saja, bahkan kapan saja, sehingga perlu dilakukan perubahan sistem dengan mudah untuk mengakomodasi pengetahuan yang semakin banyak tersebut.
- f. Pandangan dan pendapat semua siswa berbeda-beda, sehingga tidak ada jaminan solusi sistem pakar merupakan jawaban yang benar
- g. Berdasarkan kaidah atau asas tertentu, menurut arah penciptaan dan pengembangan keahlian.
- h. Basis pengetahuan dan mesin penalaran terpisah
- i. Mampu membenarkan masukan data yang tidak lengkap dan tidak pasti.
- j. Dimaksudkan untuk dikembangkan secara bertahap berdasarkan pengetahuan para ahli.
- k. Hasil pengoperasian sistem berupa rekomendasi atau saran (Pratiwi, 2019)

2.3 Naïve Bayes

Menurut jurnal (Linda et al., 2020) *Naïve Bayes* merupakan salah satu metode klasifikasi yang menggunakan metode probabilistik dan metode statistik yang ditemukan oleh ilmuwan Inggris Thomas Bayes. Probabilitas algoritma *Naïve Bayes* memprediksi kemungkinan masa depan berdasarkan pengalaman masa lalu, yang sering dikenal dengan teorema Bayes. Probabilitas merupakan interpretasi kalkulus yang mencakup konsep probabilitas sebagai derajat probabilitas suatu pernyataan tertentu. (Yulian, 2021).

Persamaan *Naïve Bayes* adalah:

$$P(A \vee B) = \frac{P(B|A).P(A)}{P(B)} \quad (2.1)$$

Dengan persamaan 1 maka dapat dijelaskan sebagai berikut:

$P(A|B)$ = Peluang kejadian A dengan syarat B, dalam sistem ini dapat diartikan dengan peluang A jika diketahui keadaan jenis penyakit B, dan ditentukan dari

$P(B|A)$ = Peluang evidence B jika diketahui hipotesis A

$P(A)$ = Probabilitas hipotesis A tanpa memandang evidence manapun

$P(B)$ = Peluang evidence B

Penggunaan persamaan teorema bayes diatas, maka dapat ditulis sebagai berikut:

$$V_{map} = \underset{v_j \in V}{\operatorname{argmax}} \frac{P(a_1 a_2 a_3 \dots a_n | v_j) P(v_j)}{P(a_1 a_2 a_3 \dots a_n)} \quad (2.2)$$

Karena $P = (a_1 a_2 a_3 \dots a_n)$ bernilai konstan sehingga persamaan diatas dapat ditulis menjadi persamaan sebagai berikut:

$$V_{map} = \underset{v_j \in V}{\operatorname{argmax}} P(a_1 a_2 a_3 \dots a_n | v_j) \quad (2.3)$$

Dimana:

V_{map} = Probabilitas kelas V atau nilai probabilitas tertinggi dari penyakit

$P(v_j)$ = Peluang jenis kelas V atau penyakit ke-j $(a_1 a_2 a_3 \dots a_n) =$ Peluang atribut (inputan) jika diketahui keadaan v_j .

Dikarenakan nilai $P = (a_1 a_2 a_3 \dots a_n) | v_j$ sulit untuk dihitung, maka dapat diasumsikan bahwa setiap atribut pada gejala tidak mempunyai keterkaitan sehingga:

$$V_{map} = \underset{v_j \in VP(v_j)}{\operatorname{argmax}} \prod P(a_i | v_j) \quad (2.4)$$

Kemudian didapat perhitungan naïve bayes classifier yaitu menghitung $P(a_i | v_j)$ dan didapatkan rumus persamaan:

$$(a_i | v_j) = \frac{n.c+m.p}{n+m} \quad (2.5)$$

Dimana:

nc = Jumlah record pada data learning dimana $v = v_j$ dan $a = a_i$

p = 1/banyaknya class (penyakit)

m = Jumlah parameter (total gejala)

n = jumlah record pada data learning yang $v = nc$ tiap class (penyakit)

Berdasarkan penjelasan diatas didapatkan Langkah-langkah perhitungan naïve bayes classifier dapat diselesaikan sebagai berikut:

1. Menentukan nilai nc untuk tiap kelas
2. Menghitung nilai $(a_i | v_j)$ dan menghitung nilai v_j
3. Menghitung $(a_i | v_j) \times v_j$ untuk tiap v
4. Menentukan hasil klasifikasi yaitu nilai yang memiliki hasil perkalian terbesar. (Mangande, 2020).

2.3.1 Karakteristik Naïve Bayes

Pada karakteristik klasifikasi dengan menggunakan metode *Naïve Bayes* bekerja berdasarkan teori probabilitas yang memandang semua fitur dari data-data sebagai bukti dalam probabilitas. Karakteristik *Naïve Bayes* sebagai berikut:

1. Metode N bekerja teguh terhadap data-data yang terisolasi yang biasanya merupakan data dengan karakteristik berbeda (*outliner*). N juga bisa menangani nilai atribut yang salah dengan mengabaikan data latih selama proses pembangunan model dan prediksi.
2. Tangguh menghadapi atribut yang tidak relevan.
3. Atribut yang mempunyai korelasi bisa mendegradasi kinerja klasifikasi *Naïve Bayes* karena asumsi independensi atribut tersebut sudah tidak ada.

2.3.2 Penerapan *Naïve Bayes*

Contoh kasus perhitungan *naïve bayes*:

Tabel 2.1 Data jenis penyakit selama kehamilan

Kode Penyakit	Jenis Penyakit Selama Kehamilan
P001	Abortus (keluarnya janin se belum masa visibilitas)
P002	Hamil ektrauteri ektopik terganggu (hamil yang berkembang diluar rahim)
P003	Solusio plasenta (lepasnya plasenta dari dinding rahim)
P004	Preeklampsia berat (suatu komplikasi ditandai dengan hipertensi)
P005	Rupture uteri (robeknya dinding rahim)

Tabel 2.2 Data gejala penyakit selama kehamilan

Kode	Nama Gejala
G1	Usia 7 bulan/ lebih
G2	Mual
G3	Muntah
G4	Kejang
G5	Proteinuria lebih dari 3g/liter
G6	Tekanan darah $\geq 160/110$ mmHg
G7	Pusing
G8	Nyeri ulu ati
G9	Nyeri perut bagian bawah
G10	Nyeri perut pada satu sisi kanan/kiri
G11	Nyeri hebat tiba – tiba
G12	Perdarahan dari jalan lahir
G13	Perdarahan dari jalan lahir warna kehitaman
G14	Kematian janin
G15	Syok

G16	Pemeriksaan dinding Rahim
G17	Tampak Pucat
G18	Air kencing berwarna kemerahan
G19	Tekanan darah turun sampai dibawah 90/60 mmHg
G20	Nadi cepat
G21	Kontraksi dari Rahim
G22	Jumlah pendarahan sedikit
G23	Produksi urin sedikit
G24	Kontraksi Rahim yang hilang
G25	Kesadaran menurun

Tabel 2.3 Data Training

Kondisi	Diagnosis				
	P1	P2	P3	P4	P5
G1	-	✓	-	-	-
G2	-	✓	-	-	✓
G3	-	✓	-	-	-
G4	-	-	-	✓	-
G5	-	-	-	✓	-
G6	-	-	-	✓	-
G7	-	-	-	✓	-
G8	-	-	-	✓	-
G9	✓	✓	-	✓	✓
G10	✓	-	-	-	-
G11	-	✓	-	-	-
G12	✓	-	-	-	-
G13	-	-	✓	-	-
G14	✓	-	✓	-	-
G15	-	✓	✓	-	-
G16	-	-	-	-	✓

G17	-	-	-	-	✓
G18	-	-	-	-	✓
G19	-	-	✓	-	✓
G20	-	-	✓	-	✓
G21	-	-	✓	-	-
G22	-	-	✓	-	-
G23	-	-	-	✓	✓
G24	-	-	-	-	✓
G25	-	-	-	-	✓

Tabel 2.4 Data penyakit selama masa kehamilan dan gejalanya.

No	Penyakit	Gejala
1.	Abortus (keluarnya janin sebelum masa visibilitas)	G9,G10,G12,G14
2.	Hamil ektrauteri ektopik terganggu (hamil yang berkembang diluar Rahim)	G1, G2, G3, G9, G11, 15
3.	Solusio plasenta (lepasnya plasenta dari dinding rahim)	G13,G14, G15,G19, G20,G21, G22
4.	Preeklampsia berat (suatu komplikasi ditandai dengan hipertensi)	G4,G5,G6, G7,G8,G9, G23
5.	Rupture uteri (robeknya dinding rahim)	G2,G9, G16,G17,G18 ,G20,G23, G24, G25

Langkah perhitungan dengan *naïve bayes* sebagai berikut :

- a. Menentukan nilai n_c

Tahap pertama yang dilakukan yaitu menentukan nilai n_c untuk masing masing penyakit berdasarkan gejala yang muncul.

1. Penyakit ke-1: Abortus

$$\begin{aligned} n &= 1 & p &= 1/5=0,2 \\ m &= 25 \\ 8.n_c &= 0 & 12.n_c &= 1 \end{aligned}$$

10. $nc=0$ 14. $nc =1$
 2. Penyakit ke-2: Hamil ektrauteri *ektopik* terganggu

$$n= 1 \quad p= 1/5 = 0,2$$

$$m= 25$$

$$8. nc = 0 \quad 12. nc = 0$$

$$10. nc = 0 \quad 14. nc = 0$$

3. Penyakit ke-3 : *Solusio plasenta*

$$n = 1 \quad p=1/5 = 0,2$$

$$m= 25$$

$$8. nc = 0 \quad 12. nc = 0$$

$$10. nc = 0 \quad 14. nc = 1$$

4. Penyakit ke-4 : *Preeklampsia berat*

$$n=1 \quad p=1/5=0,2$$

$$m= 25$$

$$8. nc = 0 \quad 12.nc = 0$$

$$10. nc = 0 \quad 14. nc = 0$$

5. Penyakit ke-5 : *Rupture uteri*

$$n= 1 \quad p= 1/5 = 0,2$$

$$m= 25$$

$$8. nc = 0 \quad 12. nc = 0$$

$$10. nc = 0 \quad 14. nc = 0$$

- b. Menghitung nilai $P(a_i|v_j)$ dan menghitung nilai $P(v_j)$

Tahap kedua yang dilakukan yaitu menghitung nilai probabilitas untuk masing-masing penyakit berdasarkan gejala.

1. *Abortus*

$$P(8|A) = \frac{0+25 \cdot 0,2}{1+25} = 0,1923$$

$$P(10|A) = \frac{1+25 \cdot 0,2}{1+25} = 0,2$$

$$P(12|A) = \frac{1+25 \cdot 0,2}{1+25} = 0,2$$

$$P(14|A) = \frac{1+25 \cdot 0,2}{1+25} = 0,2$$

$$P(A) = 0,2$$

2. Hamil *ektrauteri ektopik* terganggu

$$P(8|HE) = \frac{0+25 \cdot 0,2}{1+25} = 0,1923$$

$$P(10|HE) = \frac{0+25 \cdot 0,2}{1+25} = 0,1923$$

$$P(12|HE) = \frac{0+25.0,2}{1+25} = 0,1923$$

$$P(14|HE) = \frac{0+25.0,2}{1+25} = 0,1923$$

$$P(HE) = 0,2$$

3. *Solusio plasenta*

$$P(8|SP) = \frac{0+25.0,2}{1+25} = 0,1923$$

$$P(10|SP) = \frac{0+25.0,2}{1+25} = 0,1923$$

$$P(12|SP) = \frac{0+25.0,2}{1+25} = 0,1923$$

$$P(14|SP) = \frac{0+25.0,2}{1+25} = 0,1923$$

$$P(SP) = 0,2$$

4. *Preeklampsia berat*

$$P(8|PB) = \frac{0+25.0,2}{1+25} = 0,1923$$

$$P(10|PB) = \frac{0+25.0,2}{1+25} = 0,1923$$

$$P(12|PB) = \frac{0+25.0,2}{1+25} = 0,1923$$

$$P(14|PB) = \frac{0+25.0,2}{1+25} = 0,1923$$

$$P(PB) = 0,2$$

5. *Rupture uteri*

$$P(8|RU) = \frac{0+25.0,2}{1+25} = 0,1923$$

$$P(10|RU) = \frac{0+25.0,2}{1+25} = 0,1923$$

$$P(12|RU) = \frac{0+25.0,2}{1+25} = 0,1923$$

$$P(14|RU) = \frac{0+25.0,2}{1+25} = 0,1923$$

$$P(RU) = 0,2$$

c. Menghitung $P(a_i|v_j) \times P(v_j)$ untuk tiap v

Tahap ketiga adalah mengalikan nilai probabilitas setiap penyakit dengan masing-masing gejala.

1. Penyakit ke - 1 = *Abortus*

$$\begin{aligned} &= P(A) \times [P(8|A) \times P(10|A) \times P(12|A) \times P(14|A)] \\ &= 0,2 \times 0,1923 \times 0,2 \times 0,2 \times 0,2 \end{aligned}$$

- = 0,00030768
2. Penyakit ke - 2 = Hamil *ekstrauteri ektopik* terganggu
 = $P(\text{HE}) \times [P(8|\text{HE}) \times P(10|\text{HE}) \times P(12|\text{HE}) \times P(14|\text{HE})]$
 = $0,2 \times 0,1923 \times 0,1923 \times 0,1923 \times 0,1923$
 = 0,000273493578
3. Penyakit ke - 3 = *Solusio plasenta*
 = $P(\text{SP}) \times [P(8|\text{SP}) \times P(10|\text{SP}) \times P(12|\text{SP}) \times P(14|\text{SP})]$
 = $0,2 \times 0,1923 \times 0,1923 \times 0,1923 \times 0,2$
 = 0,000284444699
4. Penyakit ke - 4 = *Preeklampsia* berat
 = $P(\text{PB}) \times [P(8|\text{PB}) \times P(10|\text{PB}) \times P(12|\text{PB}) \times P(14|\text{PB})]$
 = $0,2 \times 0,1923 \times 0,1923 \times 0,1923 \times 0,1923$
 = 0,000273493578
5. Penyakit ke - 5 = *Rupture uteri*
 = $P(\text{RU}) \times [P(8|\text{RU}) \times P(10|\text{RU}) \times P(12|\text{RU}) \times P(14|\text{RU})]$
 = $0,2 \times 0,1923 \times 0,1923 \times 0,1923 \times 0,1923$
 = 0,000273493578

Hasil v yang memiliki perkalian terbesar didapatkan pada tabel 5.

Tabel 2.5 Nilai v tiap class

Penyakit	Nilai v
Abortus	0,00030768
Hamil ekstrauteri ektopik terganggu	0,000273493578
Solusio Plasenta	0,000284444699
Preeklampsia	0,000273493578
Rupture uteri	0,000273493578

Berdasarkan tabel diatas dapat dilihat nilai v yang terbesar adalah 0,00030768, maka contoh kasus pada penyakit ibu hamil yaitu penyakit abortus (Ridho Handoko, 2021)











2.4 Unified Modelling Language (UML)

UML adalah seperangkat alat untuk mengabstraksi sistem atau perangkat lunak berorientasi objek. UML adalah singkatan dari *Unified Modeling Language*. UML juga merupakan cara untuk memungkinkan pengembangan aplikasi berkelanjutan. Aplikasi atau sistem yang tidak berdokumen biasanya dapat menghambat pengembangan karena pengembang perlu mencari dan mempelajari kode program dan diagram dasar yang dapat digunakan dalam analisis dan desain, misalnya *Use case diagram*, *class diagram*, dan *diagram aktivitas*. Diagram UML menggunakan berbagai simbol grafis dan komponen untuk mewakili berbagai aspek dari suatu sistem perangkat lunak. Tujuan UML adalah membantu dalam semua tahap siklus hidup pengembangan perangkat lunak, dari pemahaman awal kebutuhan hingga implementasi dan dokumentasi akhir. Beberapa diagram UML yang paling umum digunakan meliputi:

a. *Use Case Diagram*

Use Case Diagram digunakan untuk menjelaskan tindakan yang dapat dilakukan pengguna pada sistem yang sedang berjalan. Diagram use case menunjukkan interaksi antara use case dan aktor, dimana aktor dapat berupa orang, perangkat, atau sistem lain yang berinteraksi dengan sistem yang Anda bangun. Kasus penggunaan menggambarkan fungsionalitas sistem atau persyaratan yang harus dipenuhi sistem dari sudut pandang pengguna. Tujuan utama dari use case diagram adalah untuk menggambarkan fungsionalitas sistem dari sudut pandang pengguna. Simbol-simbol use case diagram meliputi:






Tabel 2.6 Simbol-Simbol *Use Case Diagram*(I.P.Sari, 2021)

NO	GAMBAR	NAMA	KETERANGAN
1		<i>Actor</i>	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan <i>use case</i> .
2		<i>Dependency</i>	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (<i>independent</i>) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (<i>independent</i>).
3		<i>Generalization</i>	Hubungan dimana objek anak (<i>descendent</i>) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (<i>ancestor</i>).
4		<i>Include</i>	Menspesifikasikan bahwa <i>use case</i> sumber secara eksplisit.
5		<i>Extend</i>	Menspesifikasikan bahwa <i>use case</i> target memperluas perilaku dari <i>use case</i> sumber pada suatu titik yang diberikan.
6		<i>Association</i>	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7		<i>Sistem</i>	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		<i>Use Case</i>	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
9		<i>Collaboration</i>	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya (<i>sinergi</i>).
10		<i>Note</i>	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.

b. *Activity Diagram*

Activity Diagram merupakan suatu model yang menggambarkan suatu sistem kerja yang dimulai dari suatu objek atau sistem, *Activity Diagram* menggambarkan alur proses kerja yang terstruktur dari use case yang dicakup dari titik awal hingga titik akhir, setiap aktivitas digambarkan dengan markup sesuai dengan fungsinya. Simbol diagram aktivitas meliputi:

Tabel 2.7 Simbol-Simbol *Activity Diagram*
(I.P.Sari, 2021)

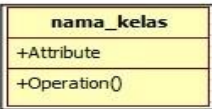





NO	GAMBAR	NAMA	KETERANGAN
1		<i>Actifity</i>	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		<i>Action</i>	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		<i>Initial Node</i>	Bagaimana objek dibentuk atau diawali.
4		<i>Actifity Final Node</i>	Bagaimana objek dibentuk dan dihancurkan
5		<i>Fork Node</i>	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

c. *Class Diagram*

Class Diagram adalah salah satu model UML yang paling penting, fungsinya untuk membuat model logis dari sistem. *Class diagram* menunjukkan seperti apa arsitektur sistem yang dirancang. *Class Diagram* ditampilkan dengan kelas yang berisi atribut dan metode. Setiap kelas dihubungkan oleh sebuah garis yang disebut asosiasi. Simbol- simbol *class diagram* dapat dilihat pada table dibawah ini:

Tabel 2.8 Simbol-Simbol *Class Diagram*

(I.P.Sari, 2021)

Simbol	Deskripsi
<p>Kelas</p> 	Kelas pada stuktur sistem.
<p>Antarmuka (Interface)</p> 	Sama dengan konsep interface dalam pemrograman berorientasi objek.
<p>Asosiasi (Association)</p> 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga di sertai dengan multiplicity.
<p>Asosiasi berarah (Directed Association)</p> 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi berarah biasanya juga disertai dengan multiplicity.
<p>Generalisasi (Generalization)</p> 	Relasi antar kelas dengan makna generalisasi-spesialisasi (Umum-khusus)
<p>Kebergantungan (Dependency)</p> 	Relasi antar kelas dengan makna kebergantungan antar kelas.
<p>Agregasi (Aggregation)</p> 	Relasi antar kelas dengan makna semua-bagian (Whole-part)

2.5 Database

Menurut jurnal (Jayanti, 2018) Database adalah sekumpulan data yang saling berhubungan atau data yang saling bergantung yang disimpan bersama-sama dalam suatu media tanpa saling memandang atau memerlukan struktur data apa pun redundansi terkendali dengan cara tertentu, membuat masukan lebih mudah digunakan atau dilihat. Mereka dapat digunakan atau ditampilkan kembali, mereka dapat disimpan secara independen dari program yang akan menggunakannya. Data disimpan sedemikian rupa sehingga dapat ditambahkan, diambil, dan dimodifikasi dengan mudah dan terkendali.

2.6 Aplikasi Mobile

Aplikasi *mobile* merupakan istilah yang digunakan untuk mendeskripsikan aplikasi internet yang berjalan pada *smartphone* atau piranti *mobile* lainnya. Aplikasi *mobile* biasanya membantu para penggunanya untuk terkoneksi dengan layanan internet yang biasa diakses pada PC atau mempermudah *user* untuk menggunakan aplikasi internet pada piranti yang biasa dibawa dan dirancang untuk melakukan fungsi spesifik perangkat komputasi *mobile* (Alda, 2020)

2.7 Java

Java merupakan sekumpulan teknologi yang digunakan untuk membangun dan menjalankan perangkat lunak pada komputer yang berdiri sendiri atau di lingkungan jaringan. Java 2 adalah generasi kedua dari platform Java. Java terdiri dari mesin interpretasi yang disebut Java Virtual Machine (JVM). Jvm membaca bytecode file, kelas program sebagai representasi bahasa mesin langsung dari program tersebut. Oleh karena itu, Java disebut sebagai bahasa pemrograman yang bersifat portable karena dapat dijalankan pada sistem operasi yang berbeda selama sistem operasi tersebut memiliki JVM. Java adalah bahasa pemrograman yang murni berorientasi objek, karena seluruh kode program dienkapsulasi dalam kelas. Java digunakan dalam berbagai aplikasi, termasuk pengembangan perangkat lunak berbasis desktop, perangkat lunak server, aplikasi web (Murduko, 2019).

2.8 Android

Android merupakan sistem operasi yang banyak digunakan pada perangkat bergerak yang dewasa ini sangat terkenal dan populer digunakan pada ponsel cerdas. *Android* juga merupakan platform pemrograman yang dikembangkan oleh Google untuk ponsel cerdas dan perangkat seluler lainnya, misalnya tablet. *Android* bisa berjalan di beberapa macam perangkat yang dikembangkan oleh banyak vendor ponsel cerdas yang berbeda. *Android* menyertakan paket pengembangan perangkat lunak untuk penulisan kode asli dan perakitan modul perangkat lunak dalam membuat aplikasi bagi pengembang *android*. Selain menyediakan paket pengembangan aplikasi *android*, *android* juga menyediakan pasar untuk mendistribusikan aplikasi yang telah selesai dikembangkan.

Android pertama kali dikembangkan oleh perusahaan kecil di Silicon Valley yang bernama Android Inc. Pada tahun 2005, sistem operasi tersebut diambil alih oleh Google dan menjadikan sistem operasi tersebut bersifat "Open Source" sehingga siapa pun dapat menggunakannya dengan gratis, termasuk penggunaan kode sumber yang digunakan dalam pengembangan sistem operasi tersebut. (Herlina, 2019)

2.8.1 Android Studio

Android Studio merupakan Lingkungan Pengembangan Perangkat Lunak Terpadu *Integrated Development Environment (IDE)* untuk pengembangan aplikasi *Android*, berdasarkan IntelliJ IDEA. Selain merupakan editor kode IntelliJ dan alat pengembang yang berdaya guna, *Android Studio* juga menawarkan banyak fitur untuk meningkatkan produktivitas Anda saat membuat aplikasi *Android*, misalnya:

1. Sistem versi berbasis Gradle yang fleksibel.
2. Emulator yang cepat dan kaya fitur.
3. Lingkungan yang menyatu untuk pengembangan bagi semua perangkat *Android*.
4. Instant Run untuk mendorong perubahan ke aplikasi yang berjalan tanpa membuat APK baru.

5. Kode Templat dan Integrasi GitHub untuk membuat fitur aplikasi yang sama dan mengimpor kode contoh.
6. Memiliki alat pengujian dan kerangka kerja yang ekstensif.

Dengan beragamnya fitur yang diberikan oleh Google Cloud Platform, akan memudahkan pengintegrasian *Google Cloud Messaging* dan *App Engine* (Herlina, 2019)

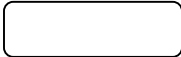
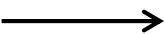
2.9 Black Box

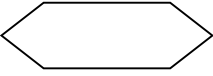

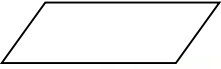
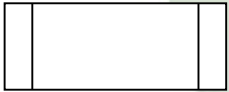
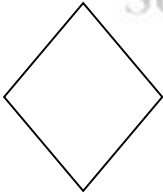
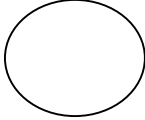
Black box testing merupakan pengujian perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program untuk mengetahui apakah fungsi, masukan dan keluaran dari perangkat sesuai dengan yang dibutuhkan. Metode *Black box testing* merupakan salah satu metode yang mudah digunakan karena hanya memerlukan batas bawah dan batas atas dari data yang diharapkan (Cholifah et al., 2018)

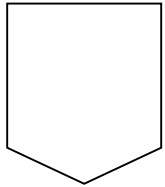
2.10 Flowchart

Flowchart adalah representasi sistematis dari logika dan proses pemrosesan informasi, atau representasi grafis dari fase dan pengaturan teknis suatu program. *Flowchart* membantu para ahli dan pengembang memecah masalah menjadi bagian-bagian yang lebih sederhana dan menganalisis berbagai pilihan pekerjaan. Tujuan dari *flowchart* adalah untuk menggambarkan langkah penyelesaian masalah secara sederhana, terorganisir, bersih, dan jelas dengan menggunakan symbol-simbol yang standar.

Tabel 2.9 Simbol- Simbol *Flowchart*

Simbol	Nama	Fungsi
	Terminator	Permulaan/ akhir Program
	Garis Alir (<i>Flow Line</i>)	Arah Aliran Program

	<i>Preparation</i>	Proses Inisialisasi/ Pemberian harga awal
	Proses	Proses Perhitungan
	<i>Input/Output Data</i>	Proses <i>input/output</i> data, parameter, informasi.
	<i>Predefined proses (sub program).</i>	Permulaan sub program/ proses menjalankan sub program
	<i>Decision</i>	Perbandingan pernyataan, penyeleksian data yang memberikan pilihan untuk langkah selanjutnya
	<i>One Page Connector</i>	Penghubung bagian- bagian <i>flowchart</i> yang berada pada satu halaman

	<i>Off Page Connector</i>	Penghubung bagian-bagian <i>flowchart</i> yang berada pada halaman berbeda
-----------------------------------------------------------------------------------	---------------------------	----------------------------------------------------------------------------

2.11 Penelitian Terdahulu

Tabel 2.10 Penelitian Terdahulu

NO	Peneliti	Judul	Kesimpulan
1.	(Linda et al., 2020)	Penerapan Algoritma Naïve Bayes Untuk Memprediksi Penyakit Malaria Pada Puskesmas Hanura	Metode naive bayes dapat diterapkan untuk memprediksi penyakit malaria pada puskesmas hanura, memprediksi penyakit malaria dalam bentuk website dapat memberikan kemudahan bagi pihak puskesmas hanura. Kemudian dengan adanya penerapan data mining maka pihak puskesmas hanura dapat mengolah data pasien dan mendapatkan informasi secara mudah.
2.	(Hendrawan et al., 2020)	Diagnosis Penyakit Tanaman Karet dengan Metode Fuzzy Mamdani	Dari hasil perhitungan yang dilakukan dalam mendiagnosis penyakit tanaman karet sebanyak 161 data objek tanaman karet yang dilengkapi dengan identitas 33 gejala serta diagnosis dari data perkebunan, kemudian diujikan 60 data tanaman karet yang tidak ada label diagnosis, kami memperoleh nilai akurasi sebesar 81.28%. Begitu juga pengujian dengan pengacakan data training dengan Cross

			Validation diperoleh hasil yang berdekatan.
3.	Arifsyah, Anita Sindar (2019)	Sistem Pakar Diagnosa Penyakit Pohon Karet Dengan Metode Certainty Factor	Rancangan tampilan form hasil diagnosa digunakan melihat hasil mendiagnosa penyakit tanaman karet berdasarkan jawaban-jawaban yang telah diberikan pengguna umum sebelumnya, apabila bagian premise dipenuhi maka bagian konkluksi dapat berhubungan dengan “OR” atau “AND”. Sistem dapat menampilkan informasi data penyakit, gejala, solusi penanganan penyakit dan juga menampilkan hasil diagnosa penyakit pada pohon karet.
4.	(Dwi Yuli Rakhmawati, Salmon Andriano Dangga, 2019)	Sistem Pakar Diagnosa Penyakit Tanaman Cengkeh Menggunakan Metode Naïve Bayes	Sistem pakar diagnosis penyakit pada tanaman cengkeh berhasil diimplementasi dalam bentuk perangkat lunak dengan fungsi melakukan diagnosis penyakit pada tanaman cengkeh. Selain itu terdapat pula menu berupa informasi daftar penyakit pada tanaman cengkeh dan gejalanya. Metode naïve bayes baik digunakan untuk diagnosis penyakit tanaman cengkeh karena menghasilkan tingkat akurasi sebesar 93%
5.	Yuliyana, Anita Sindar, Ros Maryana Sinaga (2019)	Sistem Pakar Diagnosa Penyakit Gigi Menggunakan Metode Naïve Bayes	Metode Naïve Bayes Menggunakan data training untuk menghasilkan probabilitas setiap kriteria, hasil normalisasi nilai tertinggi menjadi hasil akhir diagnosa penyakit. Basis Aturan

			bertujuan pengklasifikasian penyakit berdasarkan gejala penyakit. Pengklasifikasian menggunakan pendekatan probabilitas. Diagnosis penyakit gigi dengan metode naïve bayes dilakukan beberapa tahapan user melakukan input fakta gejala penyakit, kemudian sistem akan menghitung probabilitas prior, likelihood, dan posterior. Nilai terbesar dari perhitungan posterior akan dijadikan hasil diagnosa.
6.	(Syarifudin et al., 2018)	Sistem Pakar Diagnosis Pada Tanaman Jagung Menggunakan Metode Naïve Bayes Berbasis Android	Sistem yang dihasilkan sesuai dengan perancangan sistem karena keseluruhan kebutuhan fungsional yang diuji dengan blackbox testing hasilnya valid. Metode naïve bayes baik digunakan untuk diagnosa penyakit tanaman jagung karena menghasilkan tingkat akurasi 96%.
7.	(Sulistiani et al., 2020)	Implementasi Metode Case Based Reasoning dan Hama pada Tanaman Karet.	Aplikasi ini diperuntukkan para petani karet khususnya yang ada di wilayah tulang bawang agar masyarakat ataupun petani setempat mengetahui apakah tanaman karet yang mereka tanam terserang penyakit dan hama atau tidak dan juga bisa memberikan penanganan pertama serta edukasi bagaimana cara menanggulangnya, hasil pengujian yang dilakukan, penerapan metode case based reasoning dan k-nearest neighbor dalam sistem pakar

			untuk diagnosa penyakit dan hama pada tanaman karet mempunyai tingkat akurasi sebesar 89 %
8.	Siti Rofiqoh, Dedy Kurniadi, Andi Riansyah (2020)	Sistem Pakar Menggunakan Metode Forward Chaining untuk Diagnosa Penyakit Tanaman Karet	Mampu mendiagnosa penyakit tanaman karet dengan gejala-gejala yang ada pada database dan efektif membantu para petani dalam berkonsultasi mengenai penyakit yang akan yang mungkin dialami oleh tanaman karetnya. Sistem pakar ini dapat menambah pengetahuan petani dalam mencegah terjadinya penyakit yang terjadi pada perkebunan karet milik mereka. Sehingga bertambahnya pengetahuan dan kepedulian petani terhadap perkebunan petani terhadap perkebunan karet, dapat dipastikan hasil produksi karet akan bertambah.
9.	(Sulistiani & Muludi, 2018)	Penerapan Metode <i>Certainty Factor</i> Dalam Mendeteksi Penyakit Tanaman Karet.	Penelitian ini, menunjukkan bahwa akurasi hasil diagnosis sistem pakar menggunakan metode certainty factor sebesar 100% yang diperoleh dari sampel hasil diagnosis pengujian oleh pakar ini dilakukan untuk membuktikan bahwa sistem pakar sudah mewakili pakar. Hasil pengujian oleh pakar menunjukkan bahwa sistem pakar identifikasi penyakit pada pohon karet telah mewakili pakar

10.	Taufik Firdaus, Fitri Yanti (2022)	Implementasi Metode Naïve Bayes Pada Sistem Pakar Diagnosa Penyakit Gigi Berbasis Web	<p>Sistem pakar menggunakan metode Naïve Bayes, mampu mengidentifikasi penyakit gigi berdasarkan gejala-gejala yang telah diinputkan, sehingga masyarakat atau pasien dapat menambah pengetahuan terkait definisi penyakit gigi serta solusi penanganannya tanpa harus mengeluarkan biaya. Untuk merancang sistem pakar menggunakan naïve bayes, dibutuhkan pengumpulan data relevan dari pakar atau dokter tentang penyakit gigi serta gejala-gejala yang terkait dengan setiap penyakit, selanjutnya metode naïve bayes dapat digunakan untuk membangun model yang mampu memprediksi kemungkinan penyakit berdasarkan gejala yang diberikan. Setelah model tersebut dibangun, sistem pakar dapat dibuat kedalam website menggunakan bahasa pemrograman PHP dan database MySQL. Dari hasil uji coba yang dilakukan dengan 10 pasien secara acak menggunakan sitem pakar diagnosa penyakit gigi berbasis web dan di dampingi langsung oleh pakar / dokter gigi, mendapatkan bahwa dari 10 pasien, 8 orang menunjukkan hasil yang sama dengan pakar / dokter gigi. Ketepatan diagnosa yang diperoleh dari</p>
-----	------------------------------------	---------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

			perbandingan antara hasil diagnosa sistem dengan diagnosa dokter adalah dengan presentase nilai 80%, sistem mendapatkan klasifikasi yang layak untuk digunakan pasien atau masyarakat.
11.	(Saipul Maulana & Najiyah, 2022)	Sistem Pakar Diagnosa Penyakit Pohon Karet Menggunakan Metode Dempster Shafer	Aplikasi system pakar diagnosa penyakit pohon karet guna membantu menaikkan nilai jual pohon dan menghindari kegagalan panen dan Aplikasi system pakar diagnosa penyakit pohon karet dan membantu memberikan pengetahuan petani karet tentang penyakit pohon karet dan Aplikasi system pakar dengan melakukan transfer ilmu dari pakar dalam mendeteksi penyakit pohon karet.
12.	(Ananda et al., 2020)	Penerapan Sistem Pakar pada Diagnosa Penyakit Tanaman Karet dengan Metode Forward Chaining (FC)	Dapat disimpulkan bahwa aplikasi sistem pakar dengan metode forward chaining yang dibuat dengan bahasa pemrograman berbasis web dapat digunakan untuk membantu dan mempermudah petani dan perusahaan dalam mendeteksi penyakit tanaman karet berdasarkan gejala-gejala yang diberikan dan dapat mengetahui cara atau solusi dalam mengatasi penyakit pada tanaman karet tersebut.

