

BAB II

TINJAUAN PUSTAKA

2.1. Landasan Teori

Halaman ini membahas konsep dasar dari literatur pengetahuan ilmiah tentang teori dan fakta yang digunakan dalam proses analitis Implementasi perancangan sistem pada topik masalah yang diajukan.

2.1.1. Sistem Pakar

Program AI yang dikenal sebagai "sistem pakar" menggunakan basis pengetahuan (*Knowledge Base*) yang diperoleh dari pengalaman atau pengetahuan ahli atau pakar dalam memecahkan masalah tertentu. Basis pengetahuan ini didukung oleh mesin interaksi/interferensi, yang melakukan penalaran atau pelacakan terhadap sesuatu atau fakta-fakta serta aturan kaidah yang ada di dalamnya setelah pencarian dilakukan. (Hayadi, 2018).

2.1.2. *Forward Chaining*

Forward chaining juga dikenal sebagai pencarian yang dimotori data atau runut maju. Jika, pencarian dimulai dengan premis atau informasi masukan, dan setelah itu, informasi turunan atau kesimpulan dihasilkan. Menggunakan set aturan kondisi-aksi dikenal sebagai *forward chaining*. Data digunakan dalam metode ini untuk menentukan aturan yang akan digunakan, atau untuk memasukkan data ke memori, memprosesnya, dan menemukan hasil. (Kuriansih, S, & Hardian, 2010).

2.1.3. *Breadth-first search*

Algoritma pencarian lebar pertama adalah algoritma yang mengunjungi simpul secara *preorder*, artinya mengunjungi suatu simpul kemudian mengunjungi semua simpul yang bertetangga dengan simpul tersebut terlebih dahulu. Selanjutnya, simpul yang belum anaknya (misalnya, prioritas

penelusuran berdasarkan anak pertama [simpul sebelah kiri]), maka penelusuran dilakukan melalui simpul anak pertama dari simpul anak pertama level sebelumnya hingga mencapai level terdalam. Setelah mencapai level terdalam, penelusuran kembali ke satu level sebelumnya untuk menelusuri simpul anak kedua pada pohon biner [simpul sebelah kanan], lalu kembali ke langkah sebelumnya untuk menelusuri simpul anak kedua pada pohon biner. (Zai, Budiati, & Berutu, 2016).

2.1.4. Tanaman Tomat

Family Solanaceae mencakup tanaman sayuran tomat. (Dewi & Jumini, 2012). Salah satu suku Indian, *xitomate* atau *xitotomate*, mengatakan bahwa kata tomat berasal dari bahasa Aztek. Tomat berasal dari Amerika Latin dan tumbuh di Amerika Tengah dan Selatan. Pada awal abad ke-16, tanaman ini mulai masuk ke Eropa, dan kemudian menyebar ke Benua Asia melalui jalur Amerika Selatan dari Filipina. Sedangkan menurut (Maulida & Zulkarnaen, 2010) Sangat banyak jenis tomat dalam hal bentuk, warna, rasa, dan tekstur. Ada yang bulat, bulat pipih, keriting, atau mirip bola lampu. Tergantung pada jenis pigmen yang dominan, warna buah masak bervariasi dari kuning hingga oranye sampai merah. Mereka memiliki rasa dari masam hingga manis. Buahnya terdiri dari tandan. Buahnya berdaging dan banyak air.

2.1.5. Hypertext Markup Language (HTML)

HTML, singkatan dari *Hypertext Markup Language* atau *Markup Language*, adalah bahasa pemrograman umum yang digunakan untuk membuat halaman web yang dapat diakses melalui Internet.

Menurut (Arief, 2011) Salah satu format yang digunakan untuk membuat dokumen dan aplikasi yang berjalan di halaman web adalah HTML, juga dikenal sebagai *Hyper Text Markup Language*.

Dengan mempertimbangkan definisi di atas, kita dapat mengatakan bahwa bahasa *HyperText Markup Language* (HTML) adalah bahasa yang digunakan untuk membuat dokumen dan aplikasi yang dapat menampilkan

berbagai informasi pada web browser.

2.1.6. Website

Web, juga dikenal sebagai *World Wide Web*, adalah kumpulan halaman web yang berisi berbagai informasi yang dapat diakses oleh pengguna Internet melalui mesin pencari. Jenis informasi yang dapat ditemukan di situs web biasanya termasuk gambar, ilustrasi, video, dan konten teks yang digunakan untuk berbagai tujuan. (Yuhfizar & Hidayat, 2009).

Website berfungsi untuk menyediakan informasi. Bisnis saat ini dapat menggunakan situs web sebagai platform untuk memasarkan produk mereka dan menjangkau demografi pelanggan yang lebih luas. Selain itu, bagi individu, situs web sangat digunakan untuk berkomunikasi, berbagi informasi, dan menjalankan bisnis online.

2.1.7. Flowchart

Flowchart menggambarkan secara grafik langkah-langkah dan urutan prosedur suatu program. Ini membagi masalah ke dalam bagian yang lebih kecil dan membantu analisis menganalisis alternatif pengoperasian lainnya. (Ridlo, 2017).

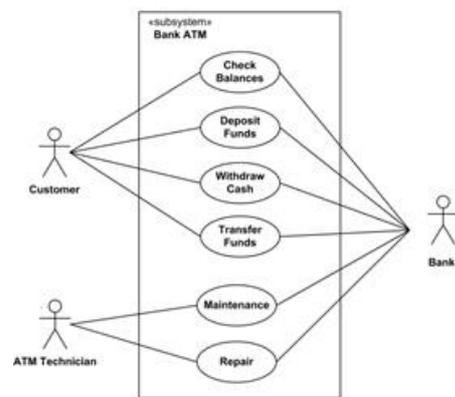
Flowchart sangat penting untuk menentukan langkah atau fungsionalitas dari sebuah proyek pembuatan program yang melibatkan banyak orang pada saat yang sama. Selain itu, membuat diagram alur proses program lebih mudah dipahami, membuatnya lebih ringkas, dan mengurangi kemungkinan salah pahami. Dalam pemrograman, *flowchart* adalah cara yang bagus untuk menghubungkan kebutuhan teknis dan non-teknis. (Adelia & Setiawan, 2011).

2.1.8. Use case Diagram

Menurut (Shalahuddin & Rosa, 2013) Diagram *use case* menggambarkan interaksi antara sistem informasi dengan satu atau lebih aktor. *Use case* diagram menunjukkan elemen-elemen berikut:

- a. Sistem menunjukkan batasan sistem dalam hal interaksi dengan aktor yang menggunakannya di luar sistem dan fitur yang harus disediakan di dalam sistem.

- b. Aktor: Aktor adalah segala hal di luar sistem yang dimaksudkan untuk melakukan fungsi tertentu. Bisa berupa individu, sistem, atau perangkat yang berkontribusi pada keberhasilan operasi sistem.
- c. Penggunaan Kasus: Kasus adalah gambaran fungsional sistem. Dengan demikian, semua orang yang menggunakan sistem, baik konsumen maupun pengguna, akan memiliki pemahaman tentang fungsi sistem utama.



An example of use case diagram for Bank ATM subsystem - top level use cases.

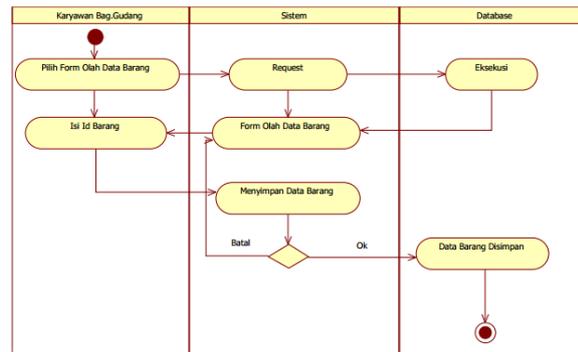
Sumber: socs.binus.ac.id

Gambar 2. 1 Contoh *Use Case Diagram*.

2.1.9. Activity Diagram

Salah satu contoh diagram UML untuk pengembangan *Use Case* adalah aktivitas diagram, yang digambarkan secara vertikal dan menyerupai runtutan proses yang terjadi dalam suatu sistem.

Menurut (Shalahuddin & Rosa, 2013) Sebuah sistem atau proses bisnis yang ada pada perangkat lunak disebut diagram aliran kerja.



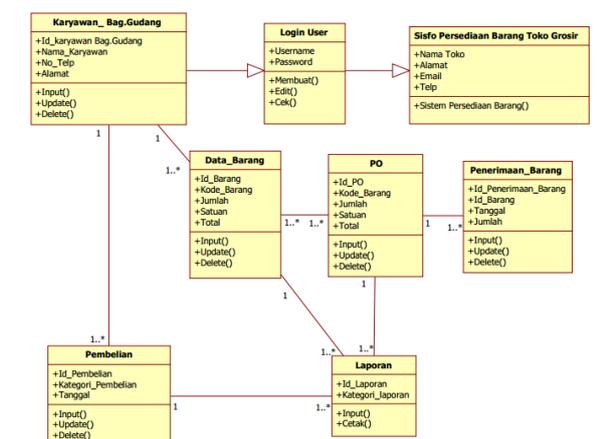
Sumber: tutorialkampus.com

Gambar 2. 2 Contoh *Activity Diagram*.

2.1.10. Class Diagram

Diagram kelas adalah jenis diagram struktur statis UML yang menunjukkan struktur sistem dengan menunjukkan hubungan antara kelas sistem, atribut, metode, dan objeknya. Ini adalah jenis diagram struktur karena menunjukkan apa yang harus dimasukkan dalam sistem yang dimodelkan oleh berbagai komponennya.

Menurut (Shalahuddin & Rosa, 2013) Dalam struktur sistem, kelas diagram mendefinisikan kelas yang digunakan untuk membangun sistem. Kelas memiliki atribut dan metode operasi.



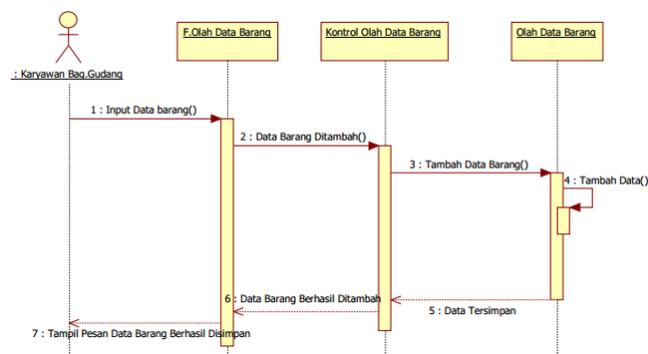
Sumber: tutorialkampus.com

Gambar 2. 3 Contoh *Class Diagram*

2.1.11. Sequence Diagram

Menurut (Hendini, 2016) *Sequence diagram* adalah proses yang menunjukkan perilaku objek dalam *use case* dengan menunjukkan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek.

Sequence diagram menunjukkan pemanggilan metode dengan panah horizontal dari pemanggil ke objek yang dipanggil, dan dapat menampilkan parameter, tipe, dan tipe kembalian yang dipilih secara opsional. (Maxim & Pressman, 2014).



Sumber: tutorialkampus.com

Gambar 2. 4 Contoh *Sequence Diagram*.

2.2. Penelitian Terdahulu

Dalam penyusunan proposal skripsi ini, penulis sedikit banyak terinspirasi dan mereferensi dari penelitian-penelitian sebelumnya yang berkaitan dengan latar belakang masalah pada proposal skripsi ini. Berikut ini penelitian terdahulu yang berhubungan dengan penulisan ini antara lain :

Tabel 2. 1 Penelitian Terdahulu

No	Peneliti	Judul	Tahun	Tujuan Penelitian
1.	Mirna Niawati, Ferly Ardhy, Tahta Herdian Andika dan	Aplikasi Diagnosis Penyakit Pada Tumbuhan	2023	penelitian ini bertujuan untuk Aplikasi sistem pakar untuk diagnosa penyakit tanaman tomat berbasis website

	Panji Bintoro	Tomat Berbasis Website		Dalam penelitian ini, peneliti menerapkan fungsi <i>if-then</i> yang didalamnya menentukan jenis penyakit berdasarkan kesesuai dengan data dan fakta dari seorang pakar. Penelitian ini menghasilkan aplikasi diagnosis penyakit tanaman tomat menggunakan metode pengembangan sistem <i>extreme programming</i> , dengan tujuan aplikasi ini dapat membantu pera petani untuk mendiagnosis tahap awal terhadap jenis penyakit berdasarkan gejala yang di alami (Niawati et al., 2023)
2.	Prahasti, Nofi Qurniati dan Venny Novita Sari	Metode <i>Forward Chaining</i> Pada Sistem Pakar Diagonis Penyakit Tanaman Tomat	2024	penelitian ini bertujuan untuk menghasilkan aplikasi sistem pakar dalam mendiagnosa penyakit pada tanaman tomat sebagai upaya membantu kesulitan penyuluh pertanian dan petani tomat akan permasalahan terhadap tanaman tomat dengan memanfaatkan teknologi informasi melalui Bahasa pemrograman visual basic dengan <i>datatabase</i> mysql

				<p>melalui pendefinisian terlebih dahulu oleh pakar. Metode yang digunakan dalam penelitian adalah model <i>waterfall</i>, sementara itu dalam pembuatan aplikasi sistem pakar basis pengetahuan yang digunakan adalah metode <i>forward chaining</i>. Aplikasi sistem pakar dibuat menghasilkan data-data yang di masukkan ke dalam aplikasi terdiri dari data gejala, data penyakit, data solusi, data solusi, data rule, serta data diagnosa (konsultasi) oleh petani. Output dari aplikasi yang dibuat menghasilkan hasil diagnosa terhadap penyakit tanaman tomat berdasarkan gejala yang diinputkan dan solusi terhadap diagnosa yang telah dilakukan. Hasil dari penelitian yaitu aplikasi sistem pakar yang dibuat dapat dioperasikan dengan baik dengan basis pengetahuan dengan metode <i>forward chaining</i> serta rule sebagai basis pengetahuan</p>
--	--	--	--	---

				telah sesuai dengan tujuan (Prahasti et al., 2024).
3.	Alberta Maria Mbagho, Meliana O. Meo dan Gregorius Rinduh Iriane	Sistem Pakar Mendiagnosa Hama Penyakit Tanaman Tomat Menggunakan Metode <i>Forward Chaining</i>	2023	penelitian ini bertujuan untuk membangun aplikasi sistem pakar yang dapat mendiagnosa hama dan penyakit tanaman tomat menggunakan metode <i>forward chaining</i> . Aplikasi ini diharapkan dapat membantu masyarakat dalam mendiagnosa jenis hama penyakit tanaman tomat. Implementasi aplikasi sistem pakar hama dan penyakit tanaman tomat ini diharapkan memberikan kemudahan akses bagi pengguna dan masyarakat luas, melalui penggunaan media sarana berbasis web (Mbagho et al., 2023).

2.3. Teori Perancangan Basis Data

Pada sub bab berikut akan membahas mengenai teori basis data yang akan digunakan berdasarkan penelitian penulis.

2.3.1. Basis Data

Basis data adalah kumpulan data yang dihubungkan bersama menurut skema atau struktur tertentu yang disimpan dalam *software* dan *hardware* komputer. *Software* dan *hardware* ini memungkinkan data dimanipulasi untuk tujuan tertentu, seperti diperbarui, dicari, diolah dengan perhitungan tertentu, dan

dihapus.

Menurut (Shalahuddin & Rosa, 2013) Tujuan utama sistem basis data terkomputerisasi adalah untuk menjaga data atau informasi yang sudah diolah tetap tersedia saat dibutuhkan. Secara umum, basis data adalah media penyimpanan data yang dapat diakses dengan mudah dan cepat.

2.3.2. Tujuan dan Manfaat Basis Data

Database digunakan untuk tujuan utama untuk mempermudah dan mempercepat pengambilan data yang dicari. Tujuan-tujuan berikut digunakan oleh basis data:

1. Kecepatan dan kemudahan (*Speed*) *Database* memungkinkan Anda menyimpan, membuat perubahan (manipulasi), dan menyegarkan data dengan cepat dan mudah.
2. Efisien ruang penyimpanan (*Space*): *Database* data dapat mengurangi jumlah iterasi data dengan menekankan jumlah pengulangan data dengan menggunakan sejumlah pengkodean.
3. Keakuratan (*Acuracy*): Pemanfaatan pengkodean atau pembentukan relasi antar data dengan menegakkan aturan atau batasan tipe data dapat diterapkan ke database untuk membantu menemukan ketidakakuratan input atau penyimpanan.
4. Keamanan (*Security*) : Pengelolah *database* mungkin tidak memperhatikan aspek keamanan saat menggunakan *database* dalam beberapa sistem (apilkasi), tetapi pertimbangan keamanan juga berlaku untuk sistem yang besar dan serius. Ini membantu Anda menentukan siapa yang boleh menggunakan *database* dan apa yang boleh dilakukan.
5. Menjaga konsistensi data (*Consitant*) Jika terjadi perubahan pada data dalam beberapa aplikasi, perubahan tersebut secara otomatis berlaku untuk semua aplikasi.
6. Data dapat digunakan secara bersama (*shared*) : data dapat digunakan oleh beberapa program aplikasi secara bersamaan, baik secara *batch* maupun secara online.

7. Dapat diterapkan standarisasi (*standardization*) : DBA dapat menerapkan standarisasi data yang disimpan agar lebih mudah digunakan, pengirim

2.3.3. Entity Relationship Diagram (ERD)

Diagram hubungan entitas, atau ERD, adalah alat yang digunakan dalam desain *database* yang menjelaskan hubungan antara objek dan atributnya.

Menurut Fatta (2009:27) *ERD* adalah model jaringan yang menggunakan kumpulan data sistem yang sederhana.

Menurut (Shalahuddin & Rosa, 2013) *ERD (Entity Relationship Diagram)* Bidang matematika saat ini sedang mengembangkannya, berdasarkan teori himpunan.

Dari uraian di atas, dapat disimpulkan bahwa *ERD (Entity Relationship Diagram)* adalah model jaringan yang didasarkan pada teori himpunan matematika dan menggunakan susunan data yang abstrak yang tersimpan dalam suatu sistem.

2.3.4. Logical Record Structure (LRS)

Tabel yang terdiri dari hasil dari seluruh set entitas merupakan representasi struktur catatan. Ini adalah pemahaman ahli mengenai *Logical Record Structure (LRS)*.

Menurut Ladjamudin (2013:163) , "Pemetaan ERD ke *database* relasional adalah istilah yang sering digunakan untuk menggambarkan transformasi ERD/LRS." Sedangkan menurut Hasugia dan Shidiq (2012:68), "*Logical Record Structure (LRS)* adalah sistem yang dijelaskan oleh diagram ER yang mengikuti pola dan aturan pemodelan tertentu yang terkait dengan transformasi ke LRS.

Berdasarkan uraian di atas, kita dapat menyimpulkan bahwa *Logical Record Structure (LRS)* adalah terjemahan dari pemetaan ERD ke dalam basis data relasional yang ditunjukkan dengan diagram ER yang mengikuti pola dan konvensi pemodelan tertentu.

2.4. Teori Pengujian Sistem

Pada sub bab berikut akan membahas tentang teori pengujian sistem yang digunakan dalam pengujian Sistem Pakar Diagnosa penyakit tanaman tomat menggunakan metode *forward chaining* dan BFS (*Breadth First Search*) :

2.4.1. *Black Box Testing*

Black box testing atau dapat disebut juga *Behavioral Testing* adalah pengujian yang dilakukan pada akhir pembuatan perangkat lunak untuk memastikan bahwa perangkat lunak berjalan dengan benar dengan melihat hasil input dan output tanpa mengetahui struktur kodenya.

Black box testing berpusat pada kebutuhan fungsional perangkat lunak. Teknik pengujian *black box* bukanlah alternatif metode *white box*; sebaliknya, itu adalah pendekatan pelengkap yang dapat mengungkap kelas kesalahan yang berbeda dari metode *white box*. Teknik pengujian *black box* memungkinkan untuk menentukan set kondisi input yang akan sepenuhnya menjalankan semua persyaratan fungsional untuk suatu program. (Maxim & Pressman, 2014).

2.4.2. *White Box Testing*

White box testing adalah pengujian yang dilakukan untuk menguji perangkat lunak dengan menganalisis dan memeriksa struktur internal dan kode perangkat lunak. Tidak seperti pengujian *black box*, yang hanya melihat hasil *input-output* perangkat lunak, pengujian *white box testing* berfokus pada aliran input-output perangkat lunak. Untuk menjalankan tes ini, penguji harus dapat memahami kode program.

White box testing adalah metode desain pengujian yang menggunakan struktur kontrol sebagai bagian dari desain tingkat komponen untuk menurunkan kasus uji. Dengan menggunakan metode pengujian *white box*, dapat diturunkan menjadi kasus uji yang menjamin bahwa setidaknya semua jalur independen dalam modul telah dilakukan, bahwa semua keputusan logis telah dilakukan di sisi yang benar dan salah, bahwa semua *loop* telah dilakukan pada batas-batas mereka dan dalam batas-batas operasional mereka, dan bahwa semua *loop* telah dilakukan di luar batas-batas (Maxim & Pressman, 2014).