# IMPLEMENTASI APLIKASI SISTEM PAKAR UNTUK DETEKSI KERUSAKAN MESIN KENDARAAN SEPEDA MOTOR MENGGUNAKAN METODE CASE-BASED REASONING BERBASIS WEBSITE PADA BENGKEL BERKAT MOTOR CASTROL TANGERANG

SKRIPSI



OLEH:

KHAIRULLAH TULFAH

211011400354

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS PAMULANG**

**TANGERANG SELATAN**

**2025**

# IMPLEMENTASI APLIKASI SISTEM PAKAR UNTUK DETEKSI KERUSAKAN MESIN KENDARAAN SEPEDA MOTOR MENGGUNAKAN METODE CASE-BASED REASONING BERBASIS WEBSITE PADA BENGKEL BERKAT MOTOR CASTROL TANGERANG

**SKRIPSI**

Diajukan Untuk Melengkapi Salah Satu Syarat  
Memperoleh Gelar Sarjana Komputer



OLEH:

KHAIRULLAH TULFAH

211011400354

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS PAMULANG**

**TANGERANG SELATAN**

**2025**

# LEMBAR PERNYATAAN

Yang bertanda tangan di bawah ini:

Nama : KHAIRULLAH TULFAH

NIM : 211011400354

Program Studi : Teknik Informatika

Fakultas : Ilmu Komputer

Jenjang Pendidikan : Strata 1

Menyatakan bahwa skripsi yang saya buat dengan judul:

IMPLEMENTASI APLIKASI SISTEM PAKAR UNTUK DETEKSI KERUSAKAN MESIN KENDARAAN SEPEDA MOTOR MENGGUNAKAN METODE CASE-BASED REASONING BERBASIS WEBSITE PADA BENGKEL BERKAT MOTOR CASTROL TANGERANG

1. Merupakan hasil karya tulis ilmiah sendiri, bukan merupakan karya yang pernah diajukan untuk memperoleh gelar akademik oleh pihak lain, dan bukan merupakan hasil plagiat.
2. Saya ijinkan untuk dikelola oleh Universitas Pamulang sesuai dengan norma hukum dan etika yang berlaku.

Pernyataan ini saya buat dengan penuh tanggung jawab dan saya bersedia menerima konsekuensi apapun sesuai aturan yang berlaku apabila di kemudian hari pernyataan ini tidak benar.

|  |
| --- |
| Tangerang Selatan, .........................2025 |
|  |
|  |
| (Khairullah Tulfah) |

# LEMBAR PERSETUJUAN

|  |  |  |
| --- | --- | --- |
| NIM | : | 211011400354 |
| Nama | : | KHAIRULLAH TULFAH |
| Program Studi | : | TEKNIK INFORMATIKA |
| Fakultas | : | ILMU KOMPUTER |
| Jenjang Pendidikan | : | STRATA 1 |
| Judul Skripsi | : | IMPLEMENTASI APLIKASI SISTEM PAKAR UNTUK DETEKSI KERUSAKAN MESIN KENDARAAN SEPEDA MOTOR MENGGUNAKAN METODE CASE-BASED REASONING BERBASIS WEBSITE PADA BENGKEL BERKAT MOTOR CASTROL TANGERANG |

Skripsi ini telah diperiksa dan disetujui oleh pembimbing untuk persyaratan sidang skripsi

Tangerang Selatan, ..............................

Pembimbing

|  |
| --- |
| Bani, S.Kom., M.Kom. |
| NIDN: 0429097505 |

Mengetahui,  
Ketua Program Studi Teknik Informatika

|  |
| --- |
| Dr. Ahmad Musyafa, S.Kom., M.Kom. |
| NIDN: 0425018609 |

# LEMBAR PENGESAHAN

|  |  |  |
| --- | --- | --- |
| NIM | : | 211011400354 |
| Nama | : | KHAIRULLAH TULFAH |
| Program Studi | : | TEKNIK INFORMATIKA |
| Fakultas | : | ILMU KOMPUTER |
| Jenjang Pendidikan | : | STRATA 1 |
| Judul Skripsi | : | IMPLEMENTASI APLIKASI SISTEM PAKAR UNTUK DETEKSI KERUSAKAN MESIN KENDARAAN SEPEDA MOTOR MENGGUNAKAN METODE CASE-BASED REASONING BERBASIS WEBSITE PADA BENGKEL BERKAT MOTOR CASTROL TANGERANG |

Skripsi ini telah dipertahankan di hadapan dewan penguji ujian skripsi fakultas Ilmu Komputer, program studi Teknik Informatika dan dinyatakan LULUS.

Tangerang Selatan, ..............................

|  |  |
| --- | --- |
| Penguji I | Penguji II |
|  |  |
|  |  |
| Nama Penguji 1 | Nama Penguji 2 |
| NIDN: - | NIDN: - |

Pembimbing

|  |
| --- |
| Bani, S.Kom., M.Kom. |
| NIDN: 0429097505 |

Mengetahui,  
Ketua Program Studi Teknik Informatika

|  |
| --- |
| Dr. Ahmad Musyafa, S.Kom., M.Kom. |
| NIDN: 0425018609 |

# *ABSTRACT*

*Keywords:*

xi+111 pages; 72 figures; 19 tables; 4 attachments

Bibliography: 40 (2012-2023)

# ABSTRAK

Kata Kunci:

xi+111 halaman; 72 gambar; 19 tabel; 4 lampiran  
Daftar acuan: 40 (2012-2023)

# KATA PENGANTAR

Puji syukur Alhamdulillah ke hadirat Allah SWT yang telah melimpahkan segala rahmat dan karunia-Nya, sehingga penulis dapat menyelesaikan skripsi ini yang berjudul **“IMPLEMENTASI APLIKASI SISTEM PAKAR UNTUK DETEKSI KERUSAKAN MESIN KENDARAAN SEPEDA MOTOR MENGGUNAKAN METODE CASE-BASED REASONING BERBASIS WEBSITE PADA BENGKEL BERKAT MOTOR CASTROL TANGERANG”**.

Skripsi ini disusun untuk memenuhi salah satu syarat kelulusan program studi strata satu (S1) pada program studi Teknik Informatika di Universitas Pamulang.

Penulis mengucapkan terima kasih kepada:

1. **Allah SWT** atas nikmat iman, Islam, kesehatan, dan umur panjang.
2. Bapak **Dr. Pranoto, S.E., M.M.,** selaku Ketua Yayasan Sasmita Jaya.
3. Bapak **Dr. E. Nurzaman A.M., MM., M.Si.,** selaku Rektor Universitas Pamulang.
4. Bapak **Yan Mitha Djaksana, S.Kom., M.Kom.,** selaku Dekan Fakultas Ilmu Komputer Universitas Pamulang.
5. Bapak **Dr. Ahmad Musyafa, S.Kom., M.Kom.,** selaku Ketua Program Studi Teknik Informatika Universitas Pamulang.
6. Bapak **Bani, S.Kom., M.Kom.,** selaku Dosen Pembimbing.
7. Kedua **orang tua** yang selalu mendoakan dan mendukung.
8. Seluruh **dosen**, **kerabat** dan **sahabat**, yaitu **Adit**, **Teguh**, **Rizky**, **Ammar**, **Pai**, serta **teman-teman** **UNPAM** seperjuangan kelas **07TPLP005**.

Semoga Allah SWT membalas segala kebaikan. Penulis menyadari bahwa skripsi ini masih jauh dari sempurna. Semoga skripsi ini bermanfaat bagi pembaca.

|  |
| --- |
| Tangerang Selatan, 9 Desember 2024 |
| Khairullah Tulfah |

# DAFTAR ISI

# DAFTAR GAMBAR

# DAFTAR TABEL

# DAFTAR LAMPIRAN

# BAB I PENDAHULUAN

## Latar Belakang

Di eramodern ini, perkembangan teknologi informasi telah memberikan kontribusi yang signifikan dalam berbagai aspek kehidupan, termasuk di bidang otomotif. Salah satu aspek yang mengalami kemajuan pesat adalah pemeliharaan dan perbaikan kendaraan bermotor. Seiring dengan meningkatnya kebutuhan akan layanan perawatan dan perbaikan mesin sepeda motor, seperti yang dialami oleh Bengkel Berkat Motor CastrolTangerang, muncul tantangan baru. Banyak pengguna belum memiliki pemahaman yang memadai tentang kerusakan mesin sepeda motor, sehingga sering kali mengalami kesulitan dalam mengidentifikasi sumber masalah pada kendaraan mereka.

Bengkel Berkat Motor Castrol Tangerang adalah bengkel yang berlokasi di Jalan Muhammad Toha. KM 2.9 Sebrang Pom Bensin, Jl. Komp. Pd. Arum, RT.001/RW.008, Bugel, Kec. Karawaci, Kota Tangerang dan telah lama dipercaya oleh masyarakat dalam menyediakan layanan perawatan dan perbaikan kendaraan roda dua. Bengkel ini dikenal dengan kecepatan dan akurasi dalam menangani kerusakan mesin, serta didukung oleh mekanik berpengalaman yang berkomitmen memberikan layanan profesional. Namun, tingginya permintaan layanan kadang membuat proses diagnosis dan perbaikan memerlukan waktu yang lebih lama, terutama bagi pengguna yang memerlukan penjelasan rinci mengenai masalah teknis kendaraan mereka.

Untuk mengatasi tantangan ini, penerapan sistem pakar untuk mendeteksi kerusakan mesin sepeda motor berpotensi menjadi solusi yang efektif bagi Bengkel Berkat MotorCastrol Tangerang. Sistem pakar berbasis *web* dengan metode *Case-Based Reasoning (CBR)* dapat membantu mekanik dan pengguna sepeda motor untuk mendiagnosis kerusakan mesin berdasarkan kasus-kasus serupa yang pernah terjadi sebelumnya. Metode *CBR* bekerja dengan mencari solusi untuk kasus baru berdasarkan pengalaman dari kasus yang mirip, sehingga dapat memberikan rekomendasi yang relevan dan akuratbagi setiap permasalahan.

Dengan adanya sistem ini, pengguna sepeda motor dapat menghemat waktu serta mengurangi ketergantungan pada pengetahuan teknis yang rumit. Di sisi lain, bengkel juga dapat meningkatkan efisiensi operasionalnya dengan sistem pakar yang dapat diakses kapan saja melalui *website*. Oleh karena itu, penelitian ini bertujuan untuk merancang dan mengimplementasikan aplikasi sistem pakar berbasis *web* yang dapat mendeteksi kerusakan mesin sepeda motor secara efektif dan efisien di Bengkel Berkat Motor Castrol Tangerang. Diharapkan sistem ini mampu memberikan manfaat nyata, baik bagi pengguna yang membutuhkan informasi akurat tentang perbaikan mesin, maupun bagi bengkel dalam meningkatkan layanan secara lebih optimal.

Berdasarkan permasalahan tersebut, maka penulis akan merancang dan mengimplementasikan sebuah aplikasi sistem pakar yang dapat membantu proses perkembangan Bengkel Berkat Motor Castrol Tangerang berbasis *website* yang berjudul **“****IMPLEMENTASI APLIKASI SISTEM PAKAR UNTUK DETEKSI KERUSAKAN MESIN KENDARAAN SEPEDA MOTOR MENGGUNAKAN METODE CASE-BASED REASONING BERBASIS WEBSITE PADA BENGKEL BERKAT MOTOR CASTROL TANGERANG”**. Diharapkan dari pembuatan aplikasi ini, memberikan kemudahan bagi para pengguna.

## Identifikasi Masalah

Berdasarkan latar belakang di atas, terdapat beberapa masalah yang dapat diidentifikasi yaitu sebagai berikut:

1. Tingginya permintaan layanan perbaikan di Bengkel Berkat Motor CastrolTangerang sering menyebabkan keterlambatan mekanik dalam menganalisis kerusakan mesin sepeda motor secara tepat waktu.
2. Proses penyampaian informasi mengenai kerusakan saat ini hanya bergantung pada analisa mekanik, sehingga berisiko terjadi kesalahan analisis.
3. Kurangnya pemahaman pengguna tentang kerusakan mesin sepeda motor, yang membuat mereka kesulitan dalam mengidentifikasi atau memahami penyebab kerusakan yang terjadi.

## Rumusan Masalah

Berdasarkan latar belakang dan identifikasi masalah yang telah diuraikan sebelumnya, rumusan masalah dalam penelitian ini adalah:

1. Bagaimana merancang dan mengimplementasikan aplikasi sistem pakar berbasis *web* yang dapat mendeteksi kerusakan mesin sepeda motor menggunakan metode *Case-Based Reasoning* di Bengkel Berkat Motor Castrol Tangerang?
2. Bagaimana efektivitas sistem pakar ini dalam mengurangi ketergantungan pada mekanik dan meningkatkan efisiensi operasional bengkel?
3. Bagaimana sistem ini dapat membantu pengguna dalam memahami kerusakan mesin serta memperkirakan kebutuhan perbaikan?

## Batasan Penelitian

Dalam penelitian ini, terdapat sejumlah batasan yang telah dirinci dan difokuskan sebagai berikut:

1. Penelitian ini difokuskan pada penerapan metode *Case-Based Reasoning (CBR)* untuk diagnosis kerusakan mesin kendaraan sepeda motor di Bengkel Berkat Motor Castrol Tangerang.
2. Aplikasi ini hanya dirancang untuk diakses melalui *website*.
3. Data kerusakan yang digunakan hanya mencakup jenis-jenis kerusakan mesin sepeda motor yang umum terjadi sesuai pengalaman bengkel.
4. Aplikasi tidak menggantikan peran mekanik secara langsung tetapi memberikan diagnosis awal.
5. Aplikasi ini hanya mencakup diagnosis kerusakan mesin sepeda motor *Honda Beat.*

## Tujuan Penelitian

Penelitian ini bertujuan untuk mengatasi permasalahan yang telah diidentifikasi. Tujuan dari penelitian ini adalah sebagai berikut:

1. Merancang dan mengimplementasikan aplikasi sistem pakar berbasis *web* untuk membantu diagnosis kerusakan mesin sepeda motor di Bengkel Berkat Motor Castrol Tangerang.
2. Meningkatkan efisiensi pelayanan bengkel melalui sistem yang dapat diakses pelanggan untuk diagnosis awal.
3. Memberikan informasi yang relevan dan akuratkepada pengguna mengenai kerusakan mesin sepeda motor serta langkah perbaikan yang mungkin diperlukan.

## Manfaat Penelitian

Penelitian ini diharapkan dapat diambil beberapa manfaat yang mencakup dua hal pokok berikut:

1. Manfaat Bagi Penulis
2. Penulis mendapatkan pengalaman langsung dalam merancang dan mengimplementasikan aplikasi berbasis *web* untuk tujuan spesifik, yaitu diagnosis kerusakan, yang melibatkan tahapan pengumpulan data, analisis, desain, dan pengujian sistem.
3. Penelitian ini dapat memberikan kontribusi berharga bagi bidang studi penulis, yang juga dapat dijadikan referensi oleh mahasiswa atau peneliti lain dalam mengembangkan sistem serupa di masa depan, baik dalam bidang otomotif maupun bidang lain yang relevan.
4. Penelitian ini adalah salah satu syarat kelulusan dalam menyelesaikan program Strata 1 (S1).
5. Manfaat Bagi Pengguna
6. Pengguna akan memperoleh diagnosis awal terkait kerusakan mesin sepeda motor, sehingga mereka dapat memahami masalah sebelum berkonsultasi langsung dengan mekanik.
7. Pengguna dapat menghemat waktu karena tidak perlu menunggu pemeriksaan manual dari mekanik untuk memperoleh informasi awal tentang kerusakan.
8. Pengguna akan memiliki gambaran data biaya dan waktu perbaikan berdasarkan kerusakan serupa yang telah didiagnosis sebelumnya oleh sistem.

## Metodologi Penelitian



Dalam implementasi aplikasisistem pakaruntuk Deteksi Kerusakan Mesin Sepeda Motor menggunakan metode *Case-Based Reasoning (CBR)*, penulis menerapkan metode penelitian sebagai berikut:

1. Metode Pengumpulan Data

Dalam memperoleh informasi yang akurat, penulis akan menggunakan metode penelitian kuantitatif yang menggunakan data berupa angka untuk menjawab hipotesis penelitian. Metode ini dipilih karena memungkinkan pengumpulan data yang objektif dan terukur serta lebih efisien dalam hal waktu dan sumber daya, terutama saat melakukan kuesioner yang dapat diakses secara *online*. Metode pengumpulan data yang digunakan meliputi:

1. Metode Kuesioner

Kuesioner disebarkan secara *online* melalui *WhatsApp* kepada 10 mekanik Bengkel Berkat Motor *Castrol* Tangerang untuk mengumpulkan data mengenai kerusakan mesin sepeda motor berdasarkan pengalaman mereka dalam perbaikan.

1. Metode Studi Pustaka

Pengumpulan data sekunder dari buku, jurnal, dan sumber-sumber lainnya yang relevan untuk mendukung analisis dan pengembangan aplikasi sistem pakar. Lokasi studi pustaka dilakukan di Perpustakaan Universitas Pamulang, yang beralamat di Jl. Raya Puspitek, Buaran, Kec. Pamulang, Kota Tangerang Selatan, Banten 15310.

1. Metode Perancangan Sistem

Dalam proses perancangan aplikasi “Sistem Pakar untuk Deteksi Kerusakan Mesin Sepeda Motor”, penulis menerapkan Metode *Rapid Application Development (RAD), RAD* merupakan suatu metode pengembangan perangkat lunak dengan pendekatan berorientasi objek (*object oriented approach*) terhadap pengembangan sistem. Metode ini bertujuan untuk mempersingkat waktu dalam perencanaan dan penerapan suatu sistem bila dibandingkan dengan metode tradisional.

Dalam penelitian ini, digunakan metode *RAD* yang melibatkan lima tahap penelitian, antara lain perancangan dan analisis kebutuhan, desain *prototipe*, pengembangan cepat, pengujian dan integrasi, serta implementasi dan pemeliharaan.

Tahapan penelitian yang dilakukan berdasarkan pada proses *RAD* terdiri dari sebagai berikut:

1. Perencanaan dan Analisis Kebutuhan

Pengembang dan pemangku kepentingan bekerja sama untuk mendefinisikan tujuan proyek, ruang lingkup sistem, dan kebutuhan pengguna. Fokus pada identifikasi kebutuhan inti tanpa terlalu terperinci.

1. Desain *Prototipe*

*Prototipe* awal sistem dibuat berdasarkan kebutuhan yang telah dikumpulkan. *Prototipe* ini berfungsi sebagai representasi awal sistem, memungkinkan pengguna memberikan umpan balik sejak dini.

1. Pengembangan Cepat

Pengembang memperbaiki *prototipe* berdasarkan umpan balik yang diterima, lalu mengimplementasikan fungsionalitas inti menggunakan alat atau *platform* pengembangan cepat.

1. Pengujian dan Integrasi

Sistem diuji untuk memastikan fungsionalitas, kompatibilitas, dan kestabilan. Pengujian dilakukan secara intensif, termasuk pengujian sistem secara keseluruhan.

1. Implementasi dan Pemeliharaan

Sistem diterapkan dalam lingkungan pengguna dan dilakukan pelatihan bagi pengguna akhir. Pemeliharaan berkelanjutan dilakukan untuk memperbaiki masalah yang mungkin muncul setelah implementasi.

## Sistematika Penulisan

Dalam penyusunan skripsi ini secara sistematis diatur dan disusun dalam 5 bab, yang masing-masing terdiri dari sub bab. Adapun urutan sistematika penulisan adalah sebagai berikut:

**BAB I PENDAHULUAN**

Bab ini berisi pemaparan yang terdiri dari latar belakang, identifikasi masalah, rumusan masalah, batasan penelitian, tujuan penelitian, manfaat penelitian, metodologi penelitian, dan terakhir adalah sistematika penulisan.

**BAB II LANDASAN TEORI**

Bab ini menguraikan teori-teori yang menunjang penulisan atau penelitian, yang *bias* diperkuat dengan menunjukan hasil penelitian sebelumnya dan kerangka pemikiran.

**BAB III ANALISA DAN PERANCANGAN**

Bab ini menjelaskan tentang proses menganalisa dan merancang sistem aplikasi, seperti analisa sistem, perancangan basis data, *ERD* (*Entity Relationship Diagram*), transformasi *ERD* ke *Logical Record Structure* *(LRS),* *UML* (*Unified Modelling Language*) seperti *use case diagram*, *activity diagram*, *sequence diagram*, *class diagram*, dan perancangan *user interface*.

**BAB IV IMPLEMENTASI DAN PENGUJIAN**

Bab ini menjelaskan tentang implementasi dan pengujian dari aplikasi yang telah dibuat, seperti implementasi sistem, implementasi perangkat keras, implementasi perangkat lunak, implementasi aplikasi, implementasi antarmuka, implementasi program, pengajuan sistem, kasus dan hasil penelitian.

**BAB V PENUTUP**

Bab ini adalah bagian terakhir yang berisikan kesimpulan dan saran yang berhubungan dengan penulisan skripsi.

# BAB II LANDASAN TEORI



## Penelitian yang Relevan

Dalam bab ini, akan dibahas penelitian-penelitian sebelumnya yang berkaitan dengan topik penelitian yang mendukung implementasi aplikasi deteksi kerusakanmesin sepeda motor. Berikut adalah beberapa penelitian yang terkait pada penelitian ini:

Penelitian pertama yang dilakukan oleh (Yuliyanto et al., 2022) dengan judul "Sistem Pakar Mendeteksi Kerusakan Komponen Kelistrikan Sepeda Motor Matic Injeksi Menggunakan *Fuzzy Sugeno*". Penelitian ini dilakukan di Bengkel Aji Motor untuk menangani permasalahan kerusakan komponen kelistrikan akibat minimnya pengetahuan pemilik kendaraan. Sistem *fuzzy* memungkinkan penanganan data yang tidak pasti dan memberikan diagnosis berbasis aturan linguistik. Dengan metode ini, sistem mampu memberikan prediksi perbaikan yang lebih fleksibel, tetapi memerlukan aturan *fuzzy* yang lebih kompleks agar diagnosis lebih akurat.

Penelitian kedua yang dilakukan oleh (Ilyas et al., 2022) dengan judul "Penerapan Metode *Forward Chaining* Pada Sistem Pakar Untuk Diagnosa Kerusakan Sepeda Motor Berbasis *Web*". Masalah utama yang diangkat adalah kurangnya pemahaman teknisi pemula dalam mengidentifikasi kerusakan sepeda motor, yang membuat mereka bergantung pada teknisi senior. Sistem ini menggunakan pendekatan aturan *IF-THEN* untuk memberikan solusi berdasarkan gejala yang dimasukkan oleh teknisi pemula. Hasil penelitian menunjukkan bahwa sistem dapat membantu teknisi dalam mempercepat proses diagnosis, meskipun sistem bergantung sepenuhnya pada data gejala dari teknisi.

Penelitian ketiga yang dilakukan oleh (Nahar, 2022) dengan judul "Aplikasi Sistem Pendeteksi Kerusakan Mesin Sepeda Motor Satria Fu Dengan Metode *Certainty Factor*". Masalah utama yang dihadapi adalah keterbatasan teknisi yang memahami kerusakan khusus pada jenis motor Satria FU, sehingga diperlukan sistem yang dapat memberikan informasi terkait penanganan kerusakan. Sistem pakar yang dikembangkan bertujuan untuk membantu pengguna dan teknisi dengan mendiagnosis kerusakan berdasarkan tingkat keyakinan dari gejala yang dimasukkan oleh pengguna. Hasil penelitian menunjukkan bahwa metode ini berhasil memberikan hasil diagnosis yang cukup akurat dalam kondisi gejala yang spesifik, namun akurasi sistem dapat menurun apabila data yang dimasukkan tidak lengkap.

## Tinjauan Pustaka

Tinjauan Pustaka (*Literature Review*) merupakan salah satu bab yang hampir selalu ditemukan dalam proposal penelitian dan laporan penelitian, termasuk skripsi, tesis, dan disertasi. Tinjauan Pustaka tidak ditemukan dalam sebuah artikel jurnal ilmiah atau prosiding seminar ilmiah, dan fungsi Tinjauan Pustaka di sini diambil alih oleh bagian Pendahuluan. Di luar negeri, orang sering juga menerbitkan Literature Review sebagai artikel dalam jurnal ilmiah (Mahanum, 2021).

Tinjauan pustaka (*literature review*) secara sederhana dapat diartikan sebagai kajian terhadap publikasi yang telah diterbitkan sebelumnya terkait suatu topik. Tinjauan pustaka sebagai kajian literatur bertujuan memberikan landasan teori dan kerangka literatur yang relevan sehingga memberikan konteks pemahaman bagi peneliti maupun pembaca. Dalam hal ini, tinjauan pustaka digunakan untuk menyintesis berbagai publikasi yang telah ada untuk memberikan konteks pemahaman terkait suatu topik penelitian, mengidentifikasi kesenjangan dalam pengetahuan, dan memperkuat keberadaan masalah penelitian (Hansen, 2024).

Tinjauan pustaka adalah menampilkan berbagai hasil penelitian sebelumnya yang memiliki kesamaan permasalahan yang akan diteliti. Di sisi lain juga untuk menjawab berbagai tantangan yang muncul ketika memulai sebuah penelitian. Rangkuman tertulis mengenai artikel dari jurnal, buku, dan dokumen lain yang mendeskripsikan teori serta informasi baik masa lalu maupun saat ini juga salah satu makna dari tinjauan pustaka. Ia juga bisa bermakna sebagai usaha yang dilakukan oleh peneliti untuk mencari dan menghimpun berbagai informasi yang relevan dengan permasalahan yang akan diteliti (Hadi & Afandi, 2021).

### Aplikasi

Aplikasi merupakan perangkat lunak proses data yang berpacu pada sebuah komputasi. Aplikasi berasal dari bahasa Inggris *application* yang berarti penerapan, lamaran ataupun penggunaan. Sedangkan secara istilah, pengertian aplikasi adalah suatu program yang siap untuk digunakan yang dibuat untuk melaksanakan suatu fungsi bagi pengguna jasa aplikasi serta penggunaan aplikasi lain yang dapat digunakan oleh suatu sasaran yang akan dituju (Parjito et al., 2023).

Aplikasi adalah program siap pakai yang dapat digunakan untuk menjalankan perintah- perintah dari pengguna aplikasi tersebut dengan tujuan mendapatkan hasil yang lebih akurat sesuai dengan tujuan pembuatan aplikasi tersebut, aplikasi mempunyai arti yaitu pemecahan masalah yang menggunakan salah satu teknik pemrosesan data aplikasi yang biasanya berpacu pada sebuah komputansi yang diinginkan atau diharapkan maupun pemprosesan data yang diharapkan (Sari et al., 2022).

Aplikasi adalah suatu proses dari cara manual yang ditransformasikan ke komputer dengan membuat sistem atau program agar data diolah lebih berdaya guna secara optimal agar menghasilkan program yang berisikan perintah-perintah untuk melakukan pengelolahan data. Selain itu, aplikasi juga merupakan sebuah perangkat lunak yang berperan untuk membantu pekerjaan manusia agar lebih mudah, efektif dan efisien. Berbagai kegiatan yang pada awalnya dilakukan secara manual, sekarang dengan adanya aplikasi maka proses pengolahan data dan penyimpanan data dapat terselesaikan dengan waktu yang relatif lebih cepat (Nendya et al., 2023).

### Sistem Pakar

Sistem pakar adalah sistem yang dirancang untuk meniru kemampuan seorang ahli pakar agar bisa dimanfaatkan untuk memecahkan permasalahan permasalahan dalam bidang tertentu. Selain itu, sistem pakar juga merupakan suatu sistem yang dibuat atau di desain untuk menerapkan pengetahuan seperti seorang ahli pakar dalam memecahkan suatu permasalahan tertentu.( (Dewi, 2023).

Sistem pakar adalah suatu sistem yang dirancang agar dapat menyelesaikan suatu permasalahan tertentu dengan meniru kerja dari para ahli dalam menjawab pertanyaan dan memecahkan suatu masalah. Dengan sistem pakar ini orang awam pun dapat menyelesaikan masalah yang cukup rumit yang sebenarnya hanya dapat diselesaikan dengan bantuan para ahli. Bagi para ahli sistem pakar ini juga membantu aktivitasnya sebagai asisten yang sangat berpengalaman (Nahar, 2022).

Sistem pakar (*Expert System*) adalah sistem yang berusaha mengadopsi pengetahuan manusia ke komputer, agar komputer dapat menyelesaikan masalah seperti yang dilakukan oleh para ahli (pakar). Sistem pakar yang baik dirancang agar dapat menyelesaikan suatu masalah tertentu dengan meniru cara kerja dari para ahli (pakar) (Yuliyanto et al., 2022).



### Sepeda Motor

Sepeda motor merupakan alat transportasi yang kerap digunakan masyarakat untuk bepergian kesuatu tempat, baik berada diluar dan didalam kota maupun ditempat pelosok atau terpencil. Sepeda motor memiliki keistimewaan yaitu ukuran yang lebih mungil dibandingkan kendaraan lain atau mobil, karena itu pengemudi menjadi lebih mudah melesat bergerak di keramaian lalu lintas (Surahman et al., 2022).

Sepeda motor adalah kendaraan beroda dua yang ditenagai oleh sebuah mesin. Sepeda motor terdiri dari beberapa bagian, yaitu mesin sebagai sumber tenaga, transmisi untuk memindahkan dan mengatur torsi ke roda, suspensi untuk meningkatkan kenyamanan, kemudi untuk mengendalikan arah sepeda motor, rem untuk mengurangi dan menghentikan laju kendaraan, serta rangka untuk menopang dan menempatkan semua kelengkapan (Yuliyanto et al., 2022).

Sepeda motor merupakan alat transportasi yang digerakkan oleh mesin. Sepeda motor ada dua jenis , yaitu mesin 2 -tak dan mesin 4-tak. Disebut mesin 4-tak karena satu siklus motor bensin terdiri dari empat langkah torak. Perbedaan mesin 2-tak dan 4-tak terletak pada siklus yang terjadi pada motor bensin tersebut. Siklus yang terjadi mulai dari menghisap campuran bahan bakar dengan udara, mengkompresikan, meneruskan gaya tekan hasil pembakaran kemudian membuang gas sisa pembakaran dari ruang pembakaran (Ilyas et al., 2022).

### Metode *Case-Based Reasoning (CBR)*

Metode *Case Based Reasoning* adalah salah satu metode untuk membangun sistem dengan pengambilan keputusan dari kasus yang baru dengan berdasarkan solusi dari kasus–kasus sebelumnya. Konsep dari metode *Case Based Reasoning* ditemukan dari ide untuk menggunakan pengalaman– pengalaman yang terdokumentasi untuk menyelesaikan masalah yang baru (Tamara & Khair, 2024).

*Metode Case-based reasoning* (*CBR*) adalah pendekatan pemecahan masalah yang berbasis pengalaman, menggunakan repositori kasus yang telah diselesaikan sebelumnya untuk mengatasi masalah baru melalui empat fase: *Retrieve* (Mengambil), *Reuse* (Menggunakan Kembali), *Revise* (Memperbaiki), dan *Retain* (Menyimpan) (Marom, 2025).

Metode *Case-based reasoning* (*CBR*) adalah metode dalam kecerdasan buatan (*AI*) yang menggunakan pendekatan berbasis pengalaman dengan membuat analogi antara keputusan baru dengan contoh pelatihan sebelumnya yang relevan. Data pelatihan diperlakukan sebagai kasus hukum dengan hasil tertentu untuk membantu pengambilan keputusan (Van Woerkom et al., 2022).

### *Website*

*Website* adalah suatu media yang bisa digunakan untuk menampung berbagai jenis informasi seperti teks, suara, gambar, dan animasi yang dimana bisa diakses oleh komputer melalui *internet.* Selain itu, website juga merupakan media informasi berbasis jaringan komputer yang bisa diakses kapan saja dan di mana saja dengan biaya yang murah. (Firmansyah & Herman, 2023).

*Website* adalah Halaman *web* yang saling berhubungan yang berisi kumpulan informasi berupa teks, gambar, animasi, audio dan video bisa diakses melalui jalur koneksi *internet* yang dibuat untuk personal, organisasi dan perusahaan. Kumpulan dokumen-dokumen yang sangat banyak yang berada pada komputer *server* (*web server*), dimana *server-server* ini tersebar di lima benua termasuk Indonesia, dan terhubung menjadi satu melalui jaringan *internet* (Rina Noviana, 2022).

*Website* adalah salah satu aplikasi yang berisikan dokumen-dokumen *multimedia* (teks, gambar, suara, animasi, video) di dalamnya yang menggunakan protokol *HTTP* (*hyper transfer protokol*) dan untuk mengakses menggunakan perangkat lunak yang disebut *browser*. Fungsi *website* diantaranya : Media promosi, pemasaran, informasi, pendidikan dan komunikasi (Nurlailah & Nova Wardani, 2023).

### *Hypertext Markup Language (HTML)*

*Hypertext Markup Language (HTML)* adalah *script* pemrograman yang mengatur bagaimana kita menyajikan informasi di dunia *internet* dan bagaimana informasi itu membawa kita melompat dari satu tempat ke tempat lainnya. *HTML* dibuat oleh Tim *Berners-Lee* ketika masih bekerja dengan *CERN* dan dipopulerkan pertama kali oleh *browser Mosaic*. Awal tahun 1990 *HTML* mengalami perkembangan yang sangat maju. Setiap pengembangan *HTML* pasti akan menambahkan kemampuan dan fasilitas yang lebih baik dari versi sebelumnya. (Rina Noviana, 2022).

*Hypertext Markup Language (HTML)* adalah bahasa dasar untuk *web* scripting bersifat *client side* yang memungkinkan untuk menampilkan informasi dalam bentuk teks, grafik, serta *multimedia* dan juga untuk menghubungkan antartampilan *web page* (*hyperlink*). *HTML* digunakan untuk membangun struktur dasar sebuah halaman *web* dengan elemen-elemen seperti *heading*, *paragraph*, *link*, dan tabel. Bahasa ini bekerja bersama dengan *CSS* untuk tampilan dan *JavaScript* untuk interaktivitas (Sari et al., 2022).

*Hypertext Markup Language (HTML)* adalah sebuah bahasa *markah* yang digunakan untuk membuat sebuah halaman *web*, menampilkan berbagai informasi di dalam sebuah penjelajah *web Internet* dan pemformatan *hiperteks* sederhana yang ditulis dalam berkas format *ASCII* agar dapat menghasilkan tampilan wujud yang terintegrasi. *HTML* (*Hyper* *Text Markup Language*), yaitu suatu bahasa yang menggunakan tanda – tanda tertentu (disebut sebagai *Tag*) untuk menyatakan kode – kode yang harus ditafsirkn oleh *browser* agar halaman tersebut dapat ditampilkan secara benar (Parjito et al., 2023).

### *Cascading Style Sheet (CSS)*

*Cascading Style Sheet* (*CSS*) merupakan salah satu kode pemrograman yang bertujuan untuk menghias dan mengatur gaya tampilan atau *layout* halaman *web* agar lebih elegan dan menarik. *CSS* adalah sebuah dokumen yang berdiri sendiri dan dapat dimasukkan dalam kode *HTML* atau sekedar menjadi rujukan oleh *HTML* dalam pendefinisian *style*. Ada banyak hal yang dapat di lakukan menggunakan *CSS* dibandingkan dengan bahasa pemrograman inti seperti *HTML* dan *PHP*. (Rina Noviana, 2022).

*CSS* adalah singkatan dari *Cascading Style Sheet* merupakan pengembangan atas kode *HTML* yang sebelumnya. *CSS* dapat menentukan suatu struktur dasar halaman *web* secara cepat dan mudah. *CSS* merupakan Salah satu bahasa desain *web* (*Style Sheet Languange*) yang mengontrol format tampilan sebuah halaman *web* yang ditulis dengan menggunakan penanda *markup languange*. Biasanya *CSS* digunakan untuk mendesain sebuah halaman *HTML* dan *XHTML*, tetapi sekarang *CSS* bisa di aplikasikan untuk segala dokumen *XML*, termasuk *SVG* dan *XUL* bahkan *android* (Parjito et al., 2023).

*CSS* dapat diartikan sebagai lembar berjenjang untuk format tata letak halaman *web*. *CSS* merupakan sebuah temuan untuk membantu pengembang *web* yang tadinya hanya bergantung pada *HTML* sebagai bahasa *markup*. *HTML* di awal mula digunakan untuk menandai objek di halaman *web* seperti tabel, gambar, paragraf, dan sebagainya (Sari et al., 2022).

### *Hypertext Preprocessor (PHP)*

*Hypertext Preprocessor (PHP)* adalah bahasa pemrograman *script server-side* yang didesain untuk pengembangan *web*. Selain itu, *PHP* juga bisa digunakan sebagai bahasa pemrograman umum. *PHP* diciptakan oleh *Rasmus Lerdorf* pertama kali tahun 1994. Saat ini *PHP* adalah singkatan dari *PHP*: *Hypertext Preprocessor*, sebuah kepanjangan rekursif, yakni permainan kata dimana kepanjangannya terdiri dari singkatan itu sendiri: *PHP*: *Hypertext Preprocessor*. *PHP* dapat digunakan dengan gratis (*free*) dan bersifat *Open Source*. *PHP* dirilis dalam lisensi *PHP License*, sedikit berbeda dengan lisensi *GNU* *General Public License* (*GPL*) yang biasa digunakan untuk proyek *Open Source* (Rina Noviana, 2022).

*PHP* merupakan singkatan dari ”*Hypertext Preprocessor*”, *PHP* adalah sebuah bahasa *scripting* yang terpasang pada *HTML* Sebagian besar sintaksnya mirip dengan bahasa pemrograman *C*, *Java*, *asp* dan *Perl*, ditambah beberapa fungsi *PHP* yang spesifik dan mudah dimengerti. Sejarah *PHP*, awalnya *PHP* merupakan kependekan dari *Personal Home Page* (situs *personal*) dan *PHP* itu sendiri pertama kali dibuat oleh *Rasmus Lerdorf* pada tahun 1995, dan pada saat *PHP* masih bernama *FI* (*Form Interpreted*), yang wujudnya berupa sekumpulan *script* yang digunakan untuk mengolah data *form* dari *web* dan selanjutnya *Rasmus* merilis kode sumber tersebut untuk umum dan menamakannya *PHP* (Parjito et al., 2023).

*PHP* merupakan bahasa *scripting* yang memiliki banyak fitur sehingga mudah merancang sebuah *website*. *PHP* merupakan bahasa berbentuk *script* yang ditempatkan dalam *server* dan diproses di *server*. Hasilnya akan dikirimkan ke *client*, tempat pemakai menggunakan *browser*. *PHP* dikenal sebagai sebuah bahasa *scripting*, yang menyatu dengan *tag-tag HTML*, dieksekusi di *server*, dan digunakan untuk membuat halaman *web* yang dinamis seperti halnya *Active Server* *Pages* (*ASP*) atau *Java Server Pages* (*JSP*). *PHP* merupakan sebuah *software Open* *Source* (Mahdalena et al., 2023).

### *MySQL*

*MySQL* adalah salah satu jenis *database server* yang terkenal yang termasuk ke dalam jenis *RDBMS* (*Relational Database Management System*). *MySQL* merupakan *RDBMS* (*Relational Database Management System*) *server*. *RDBMS* adalah program yang memungkinkan pengguna *database* untuk membuat, mengelola, dan menggunakan data pada suatu model *relational*. Dengan demikian, tabel-tabel yang ada pada *database* memiliki relasi antara satu tabel dengan tabel lainnya (Mahdalena et al., 2023).

*MySQL* adalah sebuah perangkat lunak sistem manajemen basis data *SQL* (bahasa Inggris: *database management system*) atau *DBMS* yang *multi* alur, *multi* pengguna, dengan sekitar 6 juta instalasi di seluruh dunia. *MySQL* tersedia sebagai perangkat lunak gratis dibawah lisensi *GNU General Public License* (*GPL*), tetapi mereka juga menjual di bawah lisensi komersial untuk kasus-kasus di mana penggunaannya tidak cocok dengan penggunaan *GPL* (Parjito et al., 2023).

*MySQL* merupakan *Database Management System* (*DBMS*) *tools open source* yang mendukung *multiuser*, *multithreaded*, populer, dan *free*. Berdasarkan teori diatas maka dapat disimpulkan bahwa *SQL* adalah bahasa permintaan *database* tertentu dimana subbahasa dapat membuat dan memanipulasi data di dalam *database*. *SQL* digunakan untuk melakukan tugas-tugas seperti melalukan *update* terhadap database, yang merujuk pada konsep *Relational Database* *Management System* (*RDBMS*) (Rina Noviana, 2022).

### Kebijakan Privasi

Kebijakan Privasi adalah deskripsi lengkap mengenai tanggung jawab dan penegakan kebijakan ini untuk melindungi privasi pengguna yang mengungkapkan informasi pribadi mereka. Kebijakan ini mencakup cara pengumpulan, penggunaan, penyimpanan, dan perlindungan data pribadi pengguna. Informasi yang dikumpulkan dapat mencakup nama, alamat email, riwayat aktivitas, atau data lainnya sesuai kebutuhan layanan. (Fahri, 2023).

Kebijakan privasi adalah aturan dan prinsip yang mengatur bagaimana data pribadi pengguna dikumpulkan, digunakan, dan dilindungi dalam lingkungan digital. Dalam konteks populasi digital Indonesia yang beragam dan berkembang pesat, kebijakan privasi berperan dalam membentuk perilaku pengguna dalam melindungi privasi online. Platform digital yang mengelola sejumlah besar data pengguna harus memahami bagaimana kebijakan ini serta tingkat kepercayaan pengguna memengaruhi tindakan mereka dalam menjaga keamanan informasi pribadi (Fathni et al., 2023).

kebijakan privasi merupakan salah satu bentuk self regulation untuk melindungi data pribadi pengguna platform digital. Kebijakan privasi tersebut dibuat oleh perusahaan-perusahaan perdagangan secara elektronik di Amerika Serikat dengan tujuan untuk memperoleh kepercayaan pengguna platform digital marketplace. Kebijakan privasi sebagai model pengaturan dalam bentuk self regulation, sebenarnya akan menguntungkan pihak perusahaan dalam hal ini adalah penyelenggara platform digital marketplace, karena akan membentuk citra positif terhadap sistem perdagangan secara elektronik dalam perlindungan dan penghargaan terhadap hak-hak pribadi pengguna (Dhianty, 2022).

### Metode *Rapid Application Development (RAD)*

*Rapid Application Development* (*RAD*) merupakan suatu metode pengembangan perangkat lunak dengan pendekatan berorientasi objek (*object oriented approach*) terhadap pengembangan sistem. Metode ini bertujuan untuk mempersingkat waktu dalam perencanaan, perancangan, dan penerapan suatu sistem bila dibandingkan dengan metode tradisional (Slivnik, 2022).

*Rapid Aplication Development* (*RAD*) merupakan model proses pengembangan perangkat lunak secara *linear sequential* yang menekankan pada siklus pengembangan yang sangat singkat. *RAD* dapat dijadikan acuan untuk mengembangkan suatu sistem informasi yang unggul dalam hal kecepatan, ketepatan dan biaya yang lebih rendah (Nurman Hidayat & Kusuma Hati, 2021).

*Rapid Application Development* (*RAD*) adalah model proses pengembangan perangkat lunak yang bersifat *incremental* terutama untuk waktu pengerjaan yang pendek. *RAD* merupakan model proses perangkat lunak yang menekankan pada daur pengembangan hidup yang singkat, dan versi adaptasi cepat dari metode *Waterfall* dengan menggunakan kontruksi komponen (Hariyanto et al., 2021).

## *Unified Modeling Language (UML)*

*Unified Modelling Language (UML)* merupakan sebuah bahasa yang divisualisasikan dalam bentuk gambar atau grafik yang berfungsi untuk memberikan gambaran dan spesifikasi dalam pembangunan dan dokumentasi dari sebuah pengembangan sistem berorientasi objek (*object oriented*). *UML* memberikan sebuah standar pembuatan *blue print* sistem, yang dapat terdiri dari konsep proses bisnis, pembuatan *class* yang dapat dituangkan pada bahasa pemrograman tertentu, rancangan basis data, serta komponen-komponen yang dibutuhkan dalam pengembangan sistem (Siska Narulita et al., 2024).

*Unified Modelling Language (UML)* adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement*, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek. Perancangan *UML* pada *website* terdiri dari *activity diagram* dan *class diagram*. *UML* juga digunakan untuk memodelkan perilaku sistem, interaksi antar komponen, serta struktur data yang digunakan (Rina Noviana, 2022).

*Unified Modeling Language* (*UML*) adalah suatu metode dalam pemodelan secara *visual* yang digunakan sebagai sarana perancangan sistem berorientasi objek. *Diagram* *UML* yang sering digunakan adalah *Use Case* *Diagram*, *Activity* *Diagram*, *Sequence* *Diagram*, *Class* *Diagram*, *Statemachine* *Diagram* dan *Component Diagram* (Noneng Marthiawati et al., 2024).



### *Use Case Diagram*

*Use case diagram* merupakan visualisasi dari beberapa komponen, seperti *actor*, *use case*, dan relasi antar komponen. Beberapa simbol atau notasi digunakan dalam penggambaran fungsionalitas sebuah sistem dalam *use case diagram*. Melalui *use case diagram*, dapat membantu analis dalam penyusunan kebutuhan (*requirement*) pengembangan sistem. *Use case diagram* dipakai untuk menjelaskan perancangan sistem kepada *user* dan melakukan perancangan semua fitur yang ada pada sistem yang akan dibangun (Siska Narulita et al., 2024).

*Use case* adalah deskripsi dari sebuah sistem dari *perspektif* pengguna. *Use case* bekerja dengan cara mendeskripsikan tipikal interaksi antara *user* (pengguna) sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai. *Use case* membantu dalam memahami kebutuhan fungsional sistem serta bagaimana pengguna berinteraksi dengannya. Dengan *diagram use case*, pengembang dapat mengidentifikasi aktor, alur interaksi, dan fungsi utama yang harus disediakan oleh sistem (Nahar, 2022).

*Use case diagram* merupakan pemodelan untuk melakukan (*behavior*) sistem informasi yang akan dibuat. *Use case diagram* digunakan untuk mengetahui fungsi apa saja yang ada didalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. *Diagram* ini membantu dalam mengidentifikasi aktor yang terlibat serta hubungan mereka dengan *use case* dalam sistem. Dengan *use case diagram*, pengembang dapat memahami batasan sistem dan interaksi antara pengguna dan sistem secara lebih jelas (Slivnik, 2022).

Tabel 2. 1 tabel simbol use case diagram

|  |  |  |
| --- | --- | --- |
| **Simbol** | **Nama** | **Deskripsi** |
|  | *Use Case* | *Use case* menggambarkan deskripsi atau skenario dari fungsionalitas yang dapat dilakukan oleh sistem dari sudut pandang mahasiswa. |
|  | *Actor*/Aktor | *Actor* atau Aktor adalah entitas atau elemen di luar sistem yang berinteraksi dengan sistem. |
|  | *Boundary System* | *Boundary System* adalah garis kotak yang mengelilingi kumpulan *use case* untuk menunjukkan batas sistem di mana *kumpulan use case* tersebut beroperasi. |
|  | *Association* / hubungan | *Association* adalah hubungan antara dua atau lebih *use case* dalam suatu sistem yang memiliki ketergantungan satu sama lainnya. |
|  | *Include* | *Include* menggambarkan situasi di mana fungsionalitas dari suatu *use case* selalu diikutsertakan *(included)* dalam fungsionalitas *use case* lainnya. |
|  | *Extend* | *Extend* menggambarkan situasi di mana suatu fungsionalitas tambahan dapat ditambahkan ke dalam *use case* utama berdasarkan suatu kondisi atau skenario tertentu. |
|  | *Depedency* | *Depedency* menggambarkan ketergantungan antara dua elemen. Ketergantungan ini menunjukkan bahwa satu elemen bergantung pada elemen lainnya dalam konteks tertentu. |
|  | *Generalization* | *Generalization* menggambarkan hubungan hierarki antara *use case*, di mana *use case* yang lebih umum menyediakan fungsionalitas dasar, dan *use case* yang lebih khusus mewarisi fungsionalitas tersebut serta menambahkan fungsionalitas tambahan. |

### *Activity Diagram*

*Activity diagram* merepresentasikan aliran proses atau aktivitas dalam sebuah sistem yang akan dibangun, mulai dari proses awal, keputusan-keputusan yang terjadi di dalam sistem, hingga bagaimana sebuah proses berakhir. *Activity diagram* juga memvisualisasikan proses-proses *paralel* yang terjadi ketika sistem dieksekusi. Tahapan atau langkah-langkah yang terjadi di dalam sistem digambarkan dalam *diagram* ini. Setiap *use case* minimal terdapat satu *activity diagram*. *Activity diagram* dirancang berdasarkan satu atau beberapa *use case* yang ada pada *use case diagram* (Siska Narulita et al., 2024).

*Activity diagram* mendeskripsikan proses dan aliran kerja sistem. *Diagram* ini memperlihatkan aliran dari suatu aktivitas ke aktivitas lainnya dalam suatu sistem. *Activity diagram* digunakan untuk memodelkan logika bisnis, alur kerja, atau proses dalam suatu sistem. *Diagram* ini menampilkan bagaimana suatu aktivitas dimulai, keputusan yang diambil, dan bagaimana aktivitas berakhir. Dengan *activity diagram*, pengembang dapat memahami urutan proses dan identifikasi potensi perbaikan dalam sistem (Nahar, 2022).

*Activity diagram* atau diagram aktivitas menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak, yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan yang dilakukan aktor, jadi aktivitas dilakukan oleh sistem (Malius et al., 2021).

Tabel 2. 2 tabel simbol activity diagram

|  |  |  |
| --- | --- | --- |
| **Simbol** | **Nama** | **Deskripsi** |
|  | *Initial Node* | *Initial Node* merupakan titik awal dalam aliran kerja. Ini adalah langkah pertama dalam proses dan menandakan tempat di mana aliran kerja dimulai. |
|  | *Swimlane* | *Swimlane* digunakan untuk membagi aliran kerja menjadi bagian-bagian yang terkait dengan unit atau entitas tertentu. |
|  | *Activity* | *Activity* merupakan tugas yang harus dilakukan dalam aliran kerja. |
|  | *Control Flow* | *Control Flow* digunakan untuk menghubungkan *activity* dan menunjukkan urutan dalam aliran kerja. |
|  | *Decision* | *Decision* adalah titik di mana keputusan harus diambil dalam aliran kerja. |
|  | *Fork* | *Fork* adalah percabangan atau pembagian jalur eksekusi dalam suatu proses. *Fork* memungkinkan *activity* untuk dibagi menjadi beberapa jalur yang dapat dieksekusi secara bersamaan atau paralel. |
|  | *Join* | *Join* adalah tempat di mana jalur-jalur yang terbagi oleh *Fork* harus bergabung kembali. Ini mengindikasikan penggabungan dari aktivitas-aktivitas yang berjalan secara paralel. |
|  | *End Node* | *End Node* adalah akhir dari aliran kerja. Setelah mencapai *End Node*, *activity* dianggap selesai. |

### *Sequence Diagram*

*Sequence diagram* menggambarkan pesan (*message*) yang melewati antar *use case* setiap waktu. *Sequence diagram* memvisualisasikan semua objek yang berkaitan dalam sebuah *use case*. Selain itu, *sequence* *diagram* juga merepresentasikan kolaborasi yang dinamis antar beberapa objek dan memperlihatkan rangkaian pesan yang dikirimkan antar objek dan juga interaksi yang terjadi antar objek dalam sistem yang dibangun (Siska Narulita et al., 2024).

*Sequence Diagram* memberikan gambaran kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Diagram ini menunjukkan interaksi antar objek dalam urutan waktu yang berurutan. Setiap objek berkomunikasi melalui pesan yang menggambarkan bagaimana suatu proses atau fitur dieksekusi dalam sistem. Dengan *sequence diagram*, pengembang dapat memahami alur eksekusi dan *dependensi* antar komponen dalam sistem (Slivnik, 2022).

*Sequence diagram* menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. Diagram ini menampilkan bagaimana objek berinteraksi dalam suatu skenario tertentu melalui pertukaran pesan. Setiap pesan dikirim dalam urutan waktu tertentu, menunjukkan proses yang terjadi dalam sistem (Nurman Hidayat & Kusuma Hati, 2021).

Tabel 2. 3 tabel simbol sequence diagram

|  |  |  |
| --- | --- | --- |
| **Simbol** | **Nama** | **Deskripsi** |
|  | *Actor*/Aktor | Aktor adalah entitas di luar sistem yang berinteraksi dengan objek-objek dalam sistem melalui pertukaran pesan atau interaksi. |
|  | *Boundary* | *Boundary* adalah representasi antarmuka atau batasan antara sistem dan aktor. |
|  | *Control* | *Control* adalah komponen dalam sistem yang mengontrol alur eksekusi atau mengoordinasikan aktivitas sistem. |
|  | *Entity* | *Entity* adalah objek yang berisi data atau informasi dalam sistem. |
|  | *Object Lifeline* | *Object Lifeline* adalah menggambarkan seberapa lama objek atau entitas tertentu ada selama interaksi dalam sistem. *Object Lifeline*, mewakili kehidupan objek selama interaksi dan digambarkan sebagai garis vertikal yang menghubungkan objek dengan waktu. |
|  | *Activation* | *Activation* adalah kapan objek melakukan sesuatu atau menjalankan tindakan tertentu selama interaksi. |
|  | *Message* | *Message* adalah cara objek berkomunikasi satu sama lain dengan mengirim pesan yang berisi informasi tentang apa yang harus dilakukan. |
|  | *Return* | *Return* adalah kapan objek memberikan hasil atau respons setelah menerima pesan. |
|  | *Callback* | *Callback* adalah saat sistem menjalankan suatu tindakan khusus saat kondisi tertentu terpenuhi. |
|  | *Self-Call* | *Self-Call* adalah saat objek melakukan tindakan pada dirinya sendiri dengan cara memanggil metodenya sendiri. |

### *Class Diagram*

*Class diagram* adalah *diagram*  yang menggambarkan struktur sistem dari segi pendefenisian kelas-kelas yang akan dibuat untuk membangun sistem. *Class* adalah kumpulan objek-objek yang mempunyai struktur umum, *behavior* umum, relasi umum, dan *semantic*/kata yang umum. *Class-class* ditentukan dengan cara memeriksa objek-objek dalam *sequence diagram* dan *collaboration diagram* (Malius et al., 2021)*.*

*Class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Atribut merupakan *variable-variable* yang dimiliki oleh suatu kelas, sedangkan operasi atau metode adalah fungsi-fungsi yang dimiliki dari suatu kelas. *Diagram* ini membantu dalam memahami hubungan antar kelas, seperti asosiasi, pewarisan, dan *dependensi* dalam sistem (Nahar, 2022).

*Class Diagram* adalah hubungan antar kelas dan penjelasan *detail* pada tiap-tiap kelas didalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dari tanggung jawab entitas yang menentukan prilaku sistem. *Diagram* ini menunjukkan bagaimana kelas-kelas berinteraksi dan berbagi data dalam sistem. Dengan *class diagram*, pengembang dapat merancang struktur sistem yang efisien dan memastikan keterkaitan antar komponen berjalan sesuai kebutuhan (Slivnik, 2022).

Tabel 2. 4 tabel simbol class diagram

|  |  |  |
| --- | --- | --- |
| **Simbol** | **Nama** | **Deskripsi** |
|  | *Class*/Kelas | *Class* digambarkan sebagai kotak dengan tiga bagian: bagian atas untuk nama *class*, bagian tengah untuk *property*/atribut, dan bagian bawah untuk *method*/metode. |
|  | *Association*/ Asosiasi | *Association* digunakan untuk menunjukkan hubungan antar *class*. Diwakili oleh garis lurus yang menghubungkan dua *class*. |
|  | *Generalization*/ Pewarisan | *Generalization* digunakan untuk menunjukkan hubungan hierarki antar *class*, dengan *child class* mewarisi atribut dan metode dari *parent class*. Digambarkan sebagai garis lurus dengan segitiga putih di ujungnya, mengarah ke *parent class*. |
|  | *Aggregation*/ Agregasi | *Aggregation* digunakan untuk menunjukkan hubungan “bagian-dari” di mana satu kelas terdiri dari objek-objek *class* lain. Digambarkan sebagai garis dengan berlian putih di ujungnya. |
|  | *Composition*/ Komposisi | Mirip dengan *aggregation* tetapi hubungan ini lebih kuat; objek bagian tidak dapat berdiri sendiri tanpa objek utama. Digambarkan dengan berlian hitam di ujung garis. |
|  | *Depedency*/ Ketergantungan | *Depedency* digunakan untuk menunjukkan bahwa satu *class* bergantung pada *class* lain (menggunakan *class* lain dalam operasi). Diwakili oleh garis putus-putus dengan panah mengarah ke kelas yang menjadi tumpuan. |

## Aplikasi Pendukung

Dalam penyusunan skripsi ini, penulis menggunakan beberapa aplikasi pendukung yang berperan penting dalam perancangan aplikasi“Sistem Pakar Untuk Deteksi Kerusakan Mesin Sepeda Motor”. Berikut adalah beberapa aplikasi pendukung yang digunakan:

### *Draw.io*

*Draw.io* adalah sebuah *website* yang didesain khusus untuk menggambarkan *diagram UML* secara *online*. Dimana punya tampilan yang sangat *responsive* dan terintergrasi dengan layanan penyimpanan *file* milik *google* yaitu *Google Drive* sehingga *draw.io* menjadi alternatif dalam pembuatan *diagram UML* dengan waktu yang lebih singkat (Noneng Marthiawati et al., 2024).

*Draw.io* merupakan sebuah situs yang didesain khusus untuk menggambar *diagram* secara *online*. Untuk mengaksesnya hanya diperlukan *browser* yang mendukung *HTML5* dan juga koneksi *internet*. *Draw.io* sudah terintegrasi dengan *Google Drive* untuk penyimpanan *file* selain meng*ekspor* dalam bentuk *JPG/PNG/SVG/XML* (Eko, 2022).

*Draw.io* merupakan sebuah aplikasi yang digunakan untuk membuat rancangan *diagram* tanpa diperlukan instalasi aplikasi, cukup dengan sambungan *internet*. *Draw.io* merupakan aplikasi yang bersifat fleksibel, datanya dapat disimpan dimana saja dan dapat menggunakan media penyimpanan sendiri. Pada aplikasi ini menawarkan banyak *package* untuk merancang dan membuat *diagram*, seperti *UML*, *Flowchart*, dan 12 *Entity Relation* (Berliana & Purtiningrum, 2023).

### *Visual Studio Code*

*Visual Studio Code* adalah sebuah teks *editor* ringan dan handal yang dibuat oleh *Microsoft* untuk sistem operasi *multiplatform*, artinya tersedia juga untuk versi *Linux*, *Mac*, dan *Windows.* Teks *editor* ini secara langsung mendukung bahasa pemrograman *Javascript*, *Typescript*, dan *Node. Js*, serta bahasa pemrograman lainnya dengan bantuan *plugin* yang dapat dipasang via *marketplace Visual Studio* *Code* seperti : *C++,* *C#,* *Python*, *Go*, *Java*, *PHP*, dst(Nendya et al., 2023)*.*

*Visual studio code* yang biasa disingkat dengan *VSCode* adalah perangkat lunak yang digunakan untuk menuliskan kode-kode atau *coding* yang dapat diakses pada berbagai sistem operasi seperti *Windows*, *Linux*, maupun *macOS*. *VSCode* dikembangkan oleh *Microsoft* yang diperkenalkan pertama kali pada tanggal 29 April 2015 di konferensi *Build* 2015. *VSCode* mendukung berbagai macam bahasa pemrograman seperti *Java*, *JavaScript*, *C*, *C++,* *Java*, *Python*, dan masih banyak lagi (Salendah et al., 2022).

*Visual Studio Code* adalah *editor* perangkat lunak yang sangat ringan namun kuat. Kode sumbernya berjalan dari *desktop*. Dukungan bawaan Untuk *JavaScript*, *Scripts*, *Node.js*, ada berbagai ekstensi. Tersedia dalam bahasa lain seperti *C++*, *C#*, *Python* dan *PHP*. Hal ini Berdasarkan versi lintas *platform* dari *Githubs Electron* Pengeditan kode untuk komponen Atom berdasarkan *JavaScript* dan *HTML5*. *Editor* ini Adalah lingkungan pengembangan terintegrasi (*IDE*) berfitur lengkap. Pengembang menggunakan teknologi *cloud* terbuka *Microsoft* (Noviantoro et al., 2022).

### *Figma*

*Figma* adalah sebuah *software* yang memungkinkan penggunanya untuk bekerja mendesain tampilan secara kolaboratif pada waktu yang *real time. Figma* dapat digunakan kapanpun dan dimanapun melalui *internet* serta dapat juga digunakan pada *desktop* yang berbasis *Windows* atau *MAC OS.* kelebihan yang dimiliki oleh *Figma* antara lain dapat membuat desain *prototype* dan *mock up* yang sangat cepat, dapat berkolaborasi dengan *team* saat mendesain *prototype* (Wardhanie & Lebdaningrum, 2023).

*Figma* adalah salah satu *design tool* yang biasanya digunakan untuk membuat tampilan aplikasi *mobile*, *desktop*, *website* dan lain-lain. *Figma* bisa digunakan di sistem operasi *windows*, *linux* ataupun *mac* dengan terhubung ke internet. Umumnya *Figma* banyak digunakan oleh seseorang yang bekerja dibidang *UI/UX*, *web design* dan bidang lainnya yang sejenis.*UI* dan *UX* adalah singakatan dari *User Interface* dan *User experience* yakni merupakan sebuah tampilan *visual* dalam sebuah aplikasi atau alat pemasaran digital dalam bentuk *website* yang dapat meningkatkan *brand* di suatu perusahaan (Suparman et al., 2023).

*Figma* adalah aplikasi desain *UI* dan *UX* yang dapat digunakan untuk membuat situs *web*, aplikasi, atau komponen antarmuka. *Figma* adalah sebuah *software design tool* yang umumnya digunakan untuk mendesain, merancang, melakukan *wireframing*, *prototyping*, desain antarmuka, tampilan *website* dan aplikasi *mobile*. Umumnya penggunaan aplikasi ini banyak digunakan oleh seorang yang bekerja dibidang desain aplikasi dan sejenisnya, seperti *UI/UX designer*, *web* *designer* (Ratna Dwi Pratiwi, 2023).

### *XAMPP*

*XAMPP* adalah *web server open source* yang berjalan pada sistem operasi *cross-platform* (*Windows*, *Linux*, *MacOS*). Semua yang diperlukan untuk mengelola *website* tersedia di *XAMPP* seperti *Apache*, *MySQL/MariaDB*, *PHP*, dan *Perl*. Meski program di dalamnya lengkap, *XAMPP* tetap merupakan *web server* yang sederhana dan ringan. *XAMPP* banyak digunakan oleh *developer* untuk membuat dan menguji *website* secara lokal sebelum dipublikasikan ke *server online*. Dengan antarmuka yang *user-friendly*, *XAMPP* memudahkan instalasi dan konfigurasi komponen *web server* (Sari et al., 2022).

*XAMPP* merupakan perangkat lunak berbasis *web server* yang bersifat *open source* (bebas), serta mendukung di berbagai sistem operasi, baik *Windows*, *Linux*, atau *Mac OS*. *XAMPP* digunakan sebagai *standalone server* atau biasa disebut dengan *localhost*. Hal tersebut memudahkan dalam proses pengeditan, desain, dan pengembangan aplikasi. Selain itu, *XAMPP* juga dilengkapi dengan *phpMyAdmin* untuk mempermudah pengelolaan *database MySQL/MariaDB* (Noviantoro et al., 2022).

*XAMPP* adalah salah satu paket instalasi *apache*, *PHP*, dan *MySQL* secara instan yang dapat digunakan untuk membantu proses instalasi ketiga produk tersebut. Pengertian *XAMPP* sendiri adalah perangkat lunak (*free software*) bebas, yang mendukung untuk banyak sistem operasi, yang merupakan kompilasi dari beberapa program. *XAMPP* sering digunakan oleh *developer* untuk pengembangan dan pengujian aplikasi *web* secara lokal sebelum diunggah ke *server online* (Parjito et al., 2023).

## Teori Pengujian Sistem

Pengujian Sistem merupakan sebuah metode dalam proses memeriksa dan mengevaluasi sebuah sistem untuk memastikan bahwa ia berfungsi sesuai dengan spesifikasi yang ditentukan. Proses ini melibatkan berbagai jenis pengujian untuk mengidentifikasi kekurangan, kesalahan, atau kelemahan dalam sistem. Tujuan utama dari pengujian sistem adalah untuk memastikan bahwa sistem yang dikembangkan aman, andal, dan berfungsi seperti yang diharapkan sebelum diluncurkan atau digunakan dalam lingkungan produksi (Hajizah, 2024).

Pengujian Sistem merupakan bagian dari pengukuran yang artinya memiliki jawaban yang benar atau salah. Pengujian Sistem bertujuan untuk memastikan bahwa sistem berfungsi sesuai dengan spesifikasi yang telah ditentukan. Proses ini mencakup verifikasi dan validasi terhadap setiap komponen sistem. Dengan melakukan pengujian sistem, kesalahan atau *bug* dapat ditemukan dan diperbaiki sebelum sistem diterapkan. Pengujian Sistem yang baik akan meningkatkan keandalan dan performa sistem secara keseluruhan (Malius et al., 2021).

Pengujian Sistem merupakan bagian yang penting dalam siklus pengembangan perangkat lunak. Pengujian Sistem dilakukan untuk menjamin kualitas dan juga mengetahui kelemahan dari perangkat lunak. Tujuan dari pengujian sistem adalah untuk menjamin bahwa perangkat lunak yang dibangun memiliki kualitas yang handal. Dengan pengujian sistem yang tepat, kesalahan dapat dideteksi lebih awal sehingga meminimalkan risiko kegagalan saat digunakan (Slivnik, 2022).

### Sistem *Black Box Testing*

*Black box Testing* adalah metode pengujian perangkat lunak yang meneliti fungsi (*Functional Testing*) dari aplikasi tanpa melihat ke dalam struktur *internal* atau kinerja aplikasi. Metode uji ini dapat diterapkan untuk hampir setiap tingkat pengujian perangkat lunak seperti unit, integrasi, sistem dan penerimaan. Pengujian ini berfokus pada *input* dan *output* yang dihasilkan untuk memastikan bahwa sistem berfungsi sesuai dengan yang diharapkan (Fahrezi et al., 2022).

*BlackBox Testing* (pengujian kotak hitam) yaitu menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian dimaksud untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan. Pengujian *Black Box* dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak apakah sesuai dengan spesifikasi yang dibutuhkan (Nahar, 2022).

Pengujian dengan metode *black box* merupakan metode pengujian dimana data tersebut berasal dari pernyataan fungsional yang ditentukan tanpa memperhatikan struktur program akhir. Pengujian *black box* mengacu pada uji fungsional, metode pengujian menekankan pada menjalankan fungsi, pemeriksaan dan data keluaran. Metode ini membantu menemukan kesalahan dalam fungsi perangkat lunak tanpa perlu memahami kode sumbernya (Eko, 2022).

### *User Response* (Kuesioner)

# Kuesioner adalah instrumen yang digunakan untuk mengukur suatu peristiwa atau kejadian yang berisi kumpulan pertanyaan untuk memperoleh informasi terkait penelitian yang dilakukan. Tidak jarang, peneliti dituntut untuk membuat kuesioner sendiri. Data yang baik dan benar tergantung pada baik tidaknya alat ukur dalam penelitian, dalam hal ini adalah kuesioner (Amalia et al., 2022).

Kuesioner adalah serangkaian pertanyaan tertulis yang telah diformulasi sebelumnya untuk mencatat jawaban responden. Pertanyaan yang diajukan biasanya dinyatakan dalam bentuk pertanyaan tertutup. Kuesioner umumnya dirancang untuk mengumpulkan sejumlah besar data kuantitatif. Kuesioner tersebut dapat diberikan langsung ke responden atau didistribusikan secara elektronik atau dikirim ke responden (Yusianto, 2022).

Kusioner merupakan salah satu alat ukur yang dipergunakan sebagai pengukur kejadian. Kusioner juga disebut sebagai sekumpulan pertanyaan atau pernyataan yang digunakan untuk mendapatkan informasi dari seseorang terkait dengan penelitian yang akan dilakukan. Kusioner memiliki peran penting untuk menentukan kebenaran data yang didapatkan pada setiap penelitian (Rosita et al., 2021).

# BAB III ANALISA DAN PERANCANGAN



## Analisa Sistem

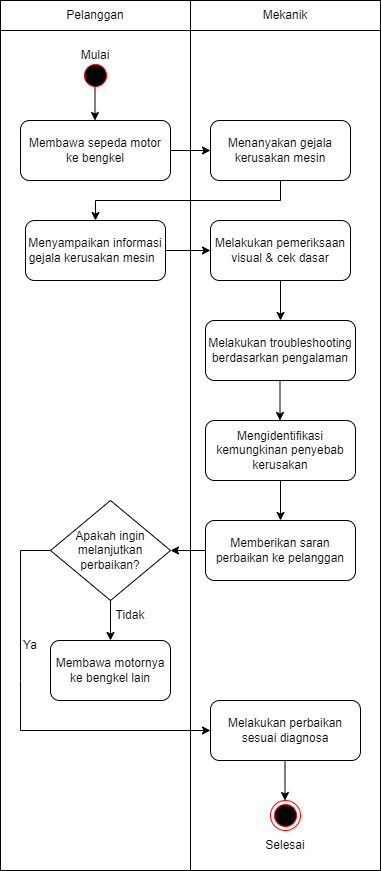
Analisa sistem merupakan fase penting dalam pengembangan sistem yang bertujuan untuk menentukan sistem informasi apa yang harus dibangun untuk menyelesaikan suatu masalah. Tahapan ini melibatkan studi terhadap sistem dan proses kerja yang sudah ada untuk mengidentifikasi kelebihan, kekurangan, dan peluang perbaikan. Analisa sistem yang kurang akurat dapat menyebabkan hasil pengembangan perangkat lunak tidak sesuai dengan harapan. Tahap ini dilakukan setelah perancangan sistem dan sebelum desain sistem, dan merupakan tahap kritis karena kesalahan dalam analisa akan berdampak pada tahap selanjutnya.

Untuk merancang sistem yang baik dan sesuai dengan harapan, diperlukan data dan informasi tentang sistem yang sudah berjalan. Data tersebut diperoleh dengan menganalisa sistem yang ada, sehingga dapat diketahui kekurangan yang perlu disempurnakan dalam sistem yang akan dirancang. Dari hasil analisa sistem berjalan, penulis dapat mengetahui kebutuhan informasi yang akan dibuat.

### Analisa Sistem Berjalan

Pada sistem yang berjalan di Bengkel Berkat Motor Castrol Tangerang, proses deteksi kerusakan mesin sepeda motor Beat dilakukan secara manual oleh mekanik. Mekanik akan memeriksa kondisi mesin, mendengarkan suara mesin, dan melakukan tes sederhana untuk mengidentifikasi masalah. Proses ini memerlukan waktu yang cukup lama dan sangat bergantung pada pengalaman dan keahlian mekanik. Selain itu, proses ini rentan terhadap kesalahan manusia (human error) karena mekanik mungkin melewatkan gejala tertentu atau salah menginterpretasikan gejala yang ada. Sistem yang berjalan saat ini juga tidak memiliki basis data yang terstruktur untuk menyimpan kasus-kasus sebelumnya, sehingga mekanik tidak dapat dengan mudah merujuk pada kasus serupa yang pernah ditangani. Hal ini menyebabkan proses diagnosa menjadi kurang efisien dan memakan waktu lebih lama, terutama ketika menghadapi kasus yang kompleks atau tidak biasa.

Sistem yang berjalan saat ini juga memiliki keterbatasan dalam hal konsistensi diagnosa. Karena diagnosa sangat bergantung pada pengalaman individu mekanik, hasil diagnosa bisa berbeda-beda antara satu mekanik dengan mekanik lainnya. Selain itu, tidak adanya dokumentasi yang terstruktur membuat sulit untuk melacak riwayat perbaikan kendaraan, sehingga pelanggan mungkin mengalami masalah yang sama berulang kali tanpa solusi yang tuntas. Hal ini dapat mengurangi kepuasan pelanggan dan mempengaruhi reputasi bengkel. Oleh karena itu, diperlukan sistem yang lebih terstruktur dan terotomatisasi untuk meningkatkan akurasi dan efisiensi proses diagnosa. Berikut adalah *activity diagram* tahapan yang sedang berjalan dalam mendeteksi kerusakan mesin sepeda motor:

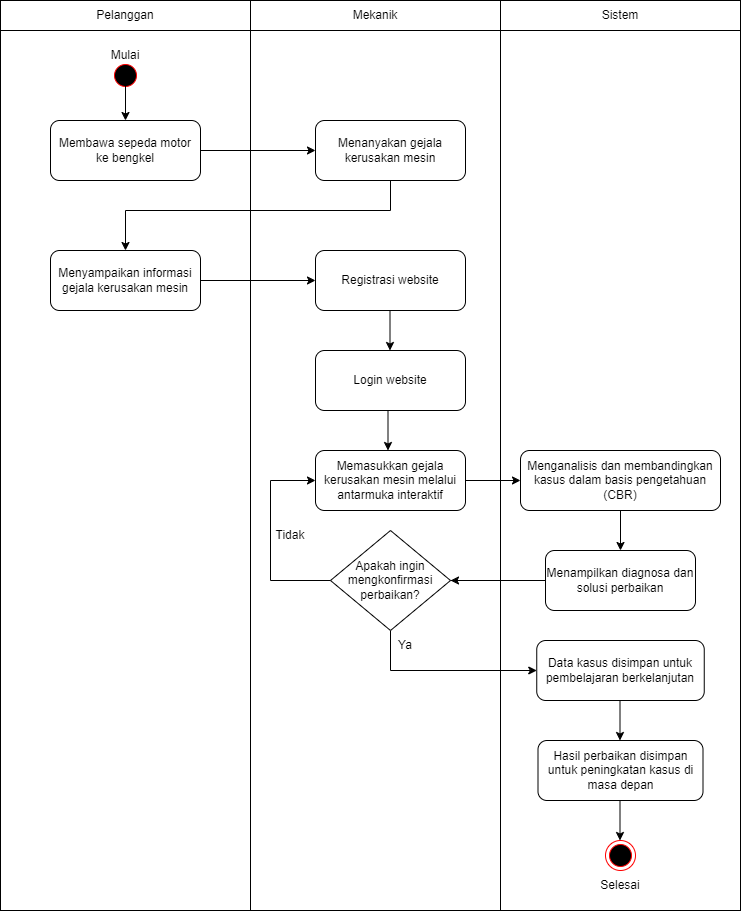


Gambar 3. 1 Activity Diagram Sistem Berjalan

### Analisa Sistem Usulan

Untuk mengatasi kekurangan pada sistem yang berjalan, diusulkan sebuah sistem berbasis *Case-Based Reasoning* (*CBR*) yang dapat membantu mekanik dalam mendiagnosa kerusakan mesin sepeda motor dengan lebih cepat dan akurat. Sistem ini akan menggunakan basis data yang terstruktur untuk menyimpan kasus-kasus sebelumnya, sehingga dapat digunakan sebagai referensi untuk mendiagnosa kasus baru. Dengan sistem ini, mekanik hanya perlu memasukkan gejala yang dialami oleh sepeda motor, dan sistem akan secara otomatis menganalisa gejala tersebut serta memberikan rekomendasi solusi berdasarkan kasus-kasus yang paling mirip. Sistem ini juga akan mengurangi ketergantungan pada pengalaman mekanik, sehingga diagnosa dapat dilakukan dengan lebih konsisten dan akurat, bahkan oleh mekanik yang kurang berpengalaman.

Selain itu, sistem usulan ini akan meningkatkan efisiensi waktu dan mengurangi risiko kesalahan manusia, sehingga proses perbaikan dapat dilakukan dengan lebih cepat dan tepat. Sistem ini juga akan menyediakan dokumentasi yang terstruktur untuk setiap kasus, memungkinkan mekanik dan pelanggan untuk melacak riwayat perbaikan kendaraan dengan mudah. Dengan adanya sistem ini, diharapkan kepuasan pelanggan akan meningkat karena masalah kerusakan mesin dapat didiagnosa dan diperbaiki dengan lebih akurat dan efisien. Sistem ini juga akan membantu bengkel dalam meningkatkan reputasi dan kepercayaan pelanggan, karena proses diagnosa dan perbaikan menjadi lebih transparan dan terpercaya. Berikut ini merupakan activity diagram dari analisa sistem usulan:



Gambar 3. 2 Activity Diagram Sistem Usulan

### 3.1.3 Analisa Kerusakan Mesin Kendaraan Sepeda Motor

Dalam melakukan pengembangan sistem pakar, terlebih dahulu dilakukan penentuan masalah yang akan dikaji. Hal ini dilakukan karena akan menentukan pengetahuan yang diperlukan tentang gejala kerusakan mesin yang dialami oleh sepeda motor Beat. Berikut adalah beberapa jenis kerusakan mesin yang umum terjadi pada sepeda motor:

1. Mesin Sulit Hidup

Mesin sulit hidup adalah kondisi di mana mesin sepeda motor Beat tidak dapat dihidupkan dengan mudah. Penyebabnya bisa berupa masalah pada sistem pengapian, bahan bakar, atau kompresi mesin.

Gejalanya meliputi:

1. Mesin tidak menyala saat distarter
2. Suara mesin tidak stabil
3. Busi tidak menghasilkan percikan api
4. Mesin Overheat

Mesin overheat adalah kondisi di mana suhu mesin terlalu tinggi dan melebihi batas normal. Penyebabnya bisa berupa masalah pada sistem pendingin, oli mesin, atau sirkulasi udara.

Gejalanya meliputi:

1. Suhu mesin sangat panas
2. Mesin mati secara tiba-tiba
3. Asap keluar dari mesin
4. Mesin Berisik

Mesin berisik adalah kondisi di mana mesin mengeluarkan suara yang tidak normal. Penyebabnya bisa berupa masalah pada bagian dalam mesin seperti piston, ring piston, atau bearing.

Gejalanya meliputi:

1. Suara ketukan dari mesin
2. Getaran yang tidak normal
3. Performa mesin menurun
4. **Mesin Tidak Bisa Berakselerasi**

Mesin tidak bisa berakselerasi adalah kondisi di mana mesin tidak dapat meningkatkan kecepatan dengan baik. Penyebabnya bisa berupa masalah pada karburator, sistem bahan bakar, atau transmisi.

Gejalanya meliputi:

1. Mesin tersendat-sendat
2. Kecepatan tidak stabil
3. Tenaga mesin berkurang

### Analisa Pengetahuan

Keberhasilan suatu sistem pakar terletak pada pengetahuan serta bagaimana mengelola pengetahuan tersebut agar dapat dilakukan penarikan kesimpulan. Pengetahuan tersebut kemudian dikonversikan ke dalam beberapa tabel untuk mempermudah proses pencarian dan solusi.

1. Tabel kerusakan Mesin Sepeda Motor

Berikut ini terdapat tabel kerusakan mesin yang di dalamnya terdapat informasi kode kerusakan dan nama kerusakan:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Kode Kerusakan** | **Nama Kerusakan** | **Deskripsi Kerusakan** | **Solusi** |
| 1 | K001 | Mesin Sulit Hidup | Mesin tidak dapat dihidupkan dengan mudah, biasanya disebabkan oleh masalah pada sistem pengapian, bahan bakar, atau kompresi mesin. | 1. Periksa sistem pengapian (busi, koil, kabel busi). 2. Periksa sistem bahan bakar (pompa bahan bakar, karburator, injektor). 3. Periksa kompresi mesin. |
| 2 | K002 | Mesin Overheat | Suhu mesin terlalu tinggi dan melebihi batas normal, disebabkan oleh masalah pada sistem pendingin, oli mesin, atau sirkulasi udara. | 1. Periksa sistem pendingin (radiator, selang, cairan pendingin). 2. Ganti oli mesin jika diperlukan. 3. Pastikan sirkulasi udara di sekitar mesin lancar. |
| 3 | K003 | Mesin Berisik | Mesin mengeluarkan suara tidak normal, biasanya disebabkan oleh masalah pada bagian dalam mesin seperti piston, ring piston, atau bearing. | 1. Periksa kondisi piston dan ring piston. 2. Periksa bearing dan komponen internal mesin. 3. Lakukan perbaikan atau penggantian komponen yang rusak. |
| 4 | K004 | Mesin Tidak Bisa Berakselerasi | Mesin tidak dapat meningkatkan kecepatan dengan baik, disebabkan oleh masalah pada karburator, sistem bahan bakar, atau transmisi. | 1. Periksa karburator atau sistem injeksi bahan bakar. 2. Periksa sistem transmisi (rantai, gir, kopling). 3. Bersihkan atau ganti komponen yang bermasalah. |
| 5 | K005 | Mesin Mogok Saat Digunakan | Mesin tiba-tiba mati saat sedang digunakan, biasanya disebabkan oleh masalah pada sistem bahan bakar atau pengapian. | 1. Periksa sistem bahan bakar (pompa bahan bakar, filter bahan bakar). 2. Periksa sistem pengapian (busi, koil, kabel busi). 3. Pastikan tidak ada kebocoran bahan bakar. |
| 6 | K006 | Mesin Boros Bahan Bakar | Konsumsi bahan bakar berlebihan, disebabkan oleh masalah pada karburator, sistem injeksi, atau kebocoran bahan bakar. | 1. Periksa karburator atau sistem injeksi bahan bakar. 2. Periksa kebocoran bahan bakar. 3. Lakukan penyetelan ulang sistem bahan bakar. |

### **Tabel 3.1 Kerusakan Mesin Sepeda Motor**

1. Tabel Gejala

Berikut ini merupakan tabel gejala yang di dalamnya terdapat informasi mengenai kode gejala dan jenis gejala pada kerusakan mesin:

Tabel 3.2 Gejala

|  |  |  |  |
| --- | --- | --- | --- |
| **No.** | **Kode Gejala** | **Gejala** | **Deskripsi Gejala** |
| 1 | G001 | Mesin tidak menyala saat distarter | Mesin tidak bereaksi saat distarter, tidak ada tanda-tanda hidup. |
| 2 | G002 | Suara mesin tidak stabil | Mesin mengeluarkan suara yang tidak konsisten, kadang tinggi kadang rendah. |
| 3 | G003 | Busi tidak menghasilkan percikan api | Busi tidak mengeluarkan percikan api saat distarter. |
| 4 | G004 | Suhu mesin sangat panas | Mesin terasa panas berlebihan saat disentuh atau terlihat dari indikator suhu. |
| 5 | G005 | Mesin mati secara tiba-tiba | Mesin tiba-tiba mati saat sedang digunakan. |
| 6 | G006 | Asap keluar dari mesin | Asap putih atau hitam keluar dari bagian mesin atau knalpot. |
| 7 | G007 | Suara ketukan dari mesin | Mesin mengeluarkan suara ketukan yang tidak normal. |
| 8 | G008 | Getaran yang tidak normal | Mesin bergetar secara berlebihan saat dihidupkan atau digunakan. |
| 9 | G009 | Performa mesin menurun | Mesin terasa lemah dan tidak bertenaga saat digunakan. |
| 10 | G010 | Mesin tersendat-sendat | Mesin tidak berakselerasi dengan lancar, tersendat saat gas ditarik. |
| 11 | G011 | Kecepatan tidak stabil | Kecepatan mesin naik turun tanpa diatur. |
| 12 | G012 | Tenaga mesin berkurang | Mesin tidak menghasilkan tenaga maksimal saat digunakan. |
| 13 | G013 | Konsumsi bahan bakar berlebihan | Bahan bakar habis lebih cepat dari biasanya. |
| 14 | G014 | Bau bahan bakar menyengat | Bau bahan bakar terasa sangat kuat di sekitar mesin. |
| 15 | G015 | Oli mesin cepat habis | Oli mesin berkurang secara signifikan dalam waktu singkat. |
| 16 | G016 | Mesin sulit distarter saat dingin | Mesin sulit dihidupkan saat kondisi mesin masih dingin. |
| 17 | G017 | Mesin sulit distarter saat panas | Mesin sulit dihidupkan saat kondisi mesin sudah panas. |
| 18 | G018 | Suara kasar saat mesin hidup | Mesin mengeluarkan suara kasar saat dihidupkan. |
| 19 | G019 | Indikator oli menyala | Lampu indikator oli pada dashboard menyala. |
| 20 | G020 | Indikator suhu mesin menyala | Lampu indikator suhu mesin pada dashboard menyala. |

1. Tabel Bobot

Berikut ini merupakan tabel bobot yang di dalamnya terdapat informasi mengenai kode kerusakan, kode gejala, jenis gejala, dan bobot pada kerusakan mesin:

Tabel 3.3 Bobot

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Kode Kerusakan** | **Kode Gejala** | **Jenis Gejala** | **Bobot** |
| 1 | K001 | G001 | Ringan | 1 |
| 2 | G002 | Sedang | 3 |
| 3 | G003 | Sedang | 3 |
| 4 | K002 | G004 | Sedang | 3 |
| 5 | G005 | Sedang | 3 |
| 6 | G006 | Berat | 5 |
| 7 | K003 | G007 | Sedang | 3 |
| 8 | G008 | Sedang | 3 |
| 9 | G009 | Berat | 5 |
| 10 | K004 | G010 | Sedang | 3 |
| 11 | G011 | Sedang | 3 |
| 12 | G012 | Berat | 5 |
| 13 | K005 | G005 | Sedang | 3 |
| 14 | G016 | Sedang | 3 |
| 15 | G017 | Berat | 5 |
| 16 | K006 | G013 | Sedang | 3 |
| 17 | G014 | Sedang | 3 |
| 18 | G015 | Berat | 5 |

### 3.1.5 Akuisisi Pengetahuan

Akuisisi pengetahuan adalah proses ekstraksi dan pengumpulan pengetahuan dari satu atau lebih pakar yang kemudian pengetahuan tersebut dimasukkan ke dalam sistem untuk melakukan perancangan dan pengembangan sistem. Berikut ini tabel yang mengolah informasi atau data yang diperoleh ke dalam struktur pengetahuan yang akan mempermudah pada saat tahap representasi pengetahuan:

**Tabel 3.4 Struktur Pengetahuan Gejala dan Kerusakan**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Gejala** | **K001** | **K002** | **K003** | **K004** | **K005** | **K006** |
| G001 | ✓ |  |  |  |  |  |
| G002 | ✓ |  |  |  |  |  |
| G003 | ✓ |  |  |  |  |  |
| G004 |  | ✓ |  |  |  |  |
| G005 |  | ✓ |  |  | ✓ |  |
| G006 |  | ✓ |  |  |  |  |
| G007 |  |  | ✓ |  |  |  |
| G008 |  |  | ✓ |  |  |  |
| G009 |  |  | ✓ |  |  |  |
| G010 |  |  |  | ✓ |  |  |
| G011 |  |  |  | ✓ |  |  |
| G012 |  |  |  | ✓ |  |  |
| G013 |  |  |  |  |  | ✓ |
| G014 |  |  |  |  |  | ✓ |
| G015 |  |  |  |  |  | ✓ |
| G016 |  |  |  |  | ✓ |  |
| G017 |  |  |  |  | ✓ |  |
| G018 |  |  | ✓ |  |  |  |
| G019 |  |  |  |  |  | ✓ |
| G020 |  | ✓ |  |  |  |  |

### 3.1.6 Penalaran Berbasis Aturan (Rule-Based Reasoning)

Penalaran berbasis aturan merupakan salah satu bentuk representasi pengetahuan yang populer dalam pengembangan sistem pakar. Aturan yang digunakan berbentuk IF-THEN. Berikut ini tabel aturan kerusakan mesin dan gejala:

Tabel 3.5 Aturan Kerusakan dan Gejala

|  |  |  |  |
| --- | --- | --- | --- |
| **Kerusakan** | **Gejala** | | |
| K001 | G001 | G002 | G003 |
| IF | Mesin tidak menyala saat distarter <AND>  Suara mesin tidak stabil <AND>  Busi tidak menghasilkan percikan api <AND> | |
| THEN | Mesin Sulit Hidup | |
| K002 | G004 | G005 | G006 |
| IF | Suhu mesin sangat panas <AND>  Mesin mati secara tiba-tiba <AND>  Asap keluar dari mesin <AND> | |
| THEN | Mesin Overheat | |
| K003 | G007 | G008 | G009 |
| IF | Suara ketukan dari mesin <AND>  Getaran yang tidak normal <AND>  Performa mesin menurun <AND> | |
| THEN | Mesin Berisik | |
| K004 | G010 | G011 | G012 |
| IF | Mesin tersendat-sendat <AND>  Kecepatan tidak stabil <AND>  Tenaga mesin berkurang <AND> | |
| THEN | Mesin Tidak Bisa Berakselerasi | |
| K005 | G005 | G016 | G017 |
| IF | Mesin mati secara tiba-tiba <AND>  Mesin sulit distarter saat dingin <AND>  Mesin sulit distarter saat panas <AND> | |
| THEN | Mesin Mogok Saat Digunakan | |
| K006 | G013 | G014 | G015 |
| IF | Konsumsi bahan bakar berlebihan <AND>  Bau bahan bakar menyengat <AND>  Oli mesin cepat habis <AND> | |
| THEN | Mesin Boros Bahan Bakar | |

### 3.1.7 Penerapan Metode Case-Based Reasoning (CBR)

Metode Case-Based Reasoning (CBR) adalah metode yang bekerja dengan cara mendiagnosa kasus baru berdasarkan kasus lama yang pernah terjadi dan memberikan solusi pada kasus baru berdasarkan kasus lama yang memiliki nilai kemiripan tertinggi.

Rumus untuk menghitung kemiripan kasus adalah:

Similarity (problem, case)

Keterangan:

* S*S* = Similarity (nilai kemiripan) antara 1 (sama) dan 0 (beda).
* p*p* = Kasus baru (problem).
* q*q* = Kasus yang ada dalam penyimpanan (case).
* w*w* = Bobot yang diberikan pada setiap gejala.

Menyiapkan Data Kasus Lama

Kasus lama adalah kasus-kasus yang sudah tersimpan dalam basis data sistem. Setiap kasus lama memiliki gejala dan solusi yang terkait dengan jenis kerusakan tertentu. Berikut adalah contoh kasus lama untuk 6 jenis kerusakan mesin sepeda motor:

1. Kasus Lama: Mesin Sulit Hidup (K001)

Gejala:

* Mesin tidak menyala saat distarter (G001) - Bobot = 1
* Suara mesin tidak stabil (G002) - Bobot = 3
* Busi tidak menghasilkan percikan api (G003) - Bobot = 3

2. Kasus Lama: Mesin Overheat (K002)

Gejala:

* Suhu mesin sangat panas (G004) - Bobot = 3
* Mesin mati secara tiba-tiba (G005) - Bobot = 3
* Asap keluar dari mesin (G006) - Bobot = 5

3. Kasus Lama: Mesin Berisik (K003)

Gejala:

* Suara ketukan dari mesin (G007) - Bobot = 3
* Getaran yang tidak normal (G008) - Bobot = 3
* Performa mesin menurun (G009) - Bobot = 5

4. Kasus Lama: Mesin Tidak Bisa Berakselerasi (K004)

Gejala:

* Mesin tersendat-sendat (G010) - Bobot = 3
* Kecepatan tidak stabil (G011) - Bobot = 3
* Tenaga mesin berkurang (G012) - Bobot = 5

5. Kasus Lama: Mesin Mogok Saat Digunakan (K005)

Gejala:

* Mesin mati secara tiba-tiba (G005) - Bobot = 3
* Mesin sulit distarter saat dingin (G016) - Bobot = 3
* Mesin sulit distarter saat panas (G017) - Bobot = 5

6. Kasus Lama: Mesin Boros Bahan Bakar (K006)

Gejala:

* Konsumsi bahan bakar berlebihan (G013) - Bobot = 3
* Bau bahan bakar menyengat (G014) - Bobot = 3
* Oli mesin cepat habis (G015) - Bobot = 5

Langkah 2: Menyiapkan Kasus Baru

Kasus baru adalah gejala yang dimasukkan oleh pengguna (mekanik) untuk didiagnosa oleh sistem. Misalkan kasus baru memiliki gejala berikut:

* Mesin tidak menyala saat distarter (G001)
* Suara mesin tidak stabil (G002)
* Busi tidak menghasilkan percikan api (G003)
* Mesin tersendat-sendat (G010)

Perhitungan Similarity untuk Setiap Kerusakan

1. Kerusakan Mesin Sulit Hidup (K001)

Gejala yang terkait dengan K001:

* Mesin tidak menyala saat distarter (G001) - Bobot = 1
* Suara mesin tidak stabil (G002) - Bobot = 3
* Busi tidak menghasilkan percikan api (G003) - Bobot = 3

Perbandingan Gejala:

|  |  |  |  |
| --- | --- | --- | --- |
| **Gejala Kasus Lama** | **Bobot** | **Gejala Kasus Baru** | **Bobot** |
| Mesin tidak menyala saat distarter (G001) | 1 | Mesin tidak menyala saat distarter (G001) | 1 |
| Suara mesin tidak stabil (G002) | 3 | Suara mesin tidak stabil (G002) | 3 |
| Busi tidak menghasilkan percikan api (G003) | 3 | Busi tidak menghasilkan percikan api (G003) | 3 |

Perhitungan:

Similarity

Jadi, Kasus baru memiliki kemiripan**100%**dengan kasus lama**Mesin Sulit Hidup (K001).**

2. Kerusakan Mesin Overheat (K002)

Gejala yang terkait dengan K002:

* Suhu mesin sangat panas (G004) - Bobot = 3
* Mesin mati secara tiba-tiba (G005) - Bobot = 3
* Asap keluar dari mesin (G006) - Bobot = 5

Perbandingan Gejala:

|  |  |  |  |
| --- | --- | --- | --- |
| **Gejala Kasus Lama** | **Bobot** | **Gejala Kasus Baru** | **Bobot** |
| Suhu mesin sangat panas (G004) | 3 | - | - |
| Mesin mati secara tiba-tiba (G005) | 3 | - | - |
| Asap keluar dari mesin (G006) | 5 | - | - |

Perhitungan:

Similarity

Jadi, Kasus baru memiliki kemiripan 0% dengan kasus lama Mesin Overheat (K002).

3. Kerusakan Mesin Berisik (K003)

Gejala yang terkait dengan K003:

* Suara ketukan dari mesin (G007) - Bobot = 3
* Getaran yang tidak normal (G008) - Bobot = 3
* Performa mesin menurun (G009) - Bobot = 5

Perbandingan Gejala:

|  |  |  |  |
| --- | --- | --- | --- |
| **Gejala Kasus Lama** | **Bobot** | **Gejala Kasus Baru** | **Bobot** |
| Suara ketukan dari mesin (G007) | 3 | - | - |
| Getaran yang tidak normal (G008) | 3 | - | - |
| Performa mesin menurun (G009) | 5 | - | - |

Perhitungan:

Similarity

Jadi, Kasus baru memiliki kemiripan 0% dengan kasus lama Mesin Berisik (K003).

4. Kerusakan Mesin Tidak Bisa Berakselerasi (K004)

Gejala yang terkait dengan K004:

* Mesin tersendat-sendat (G010) - Bobot = 3
* Kecepatan tidak stabil (G011) - Bobot = 3
* Tenaga mesin berkurang (G012) - Bobot = 5

Perbandingan Gejala:

|  |  |  |  |
| --- | --- | --- | --- |
| **Gejala Kasus Lama** | **Bobot** | **Gejala Kasus Baru** | **Bobot** |
| Mesin tersendat-sendat (G010) | 3 | Mesin tersendat-sendat (G010) | 3 |
| Kecepatan tidak stabil (G011) | 3 | - | - |
| Tenaga mesin berkurang (G012) | 5 | - | - |

Perhitungan:

Similarity

Jadi, Kasus baru memiliki kemiripan 27% dengan kasus lama Mesin Tidak Bisa Berakselerasi (K004).

5. Kerusakan Mesin Mogok Saat Digunakan (K005)

Gejala yang terkait dengan K005:

* Mesin mati secara tiba-tiba (G005) - Bobot = 3
* Mesin sulit distarter saat dingin (G016) - Bobot = 3
* Mesin sulit distarter saat panas (G017) - Bobot = 5

Perbandingan Gejala:

|  |  |  |  |
| --- | --- | --- | --- |
| **Gejala Kasus Lama** | **Bobot** | **Gejala Kasus Baru** | **Bobot** |
| Mesin mati secara tiba-tiba (G005) | 3 | - | - |
| Mesin sulit distarter saat dingin (G016) | 3 | - | - |
| Mesin sulit distarter saat panas (G017) | 5 | - | - |

Perhitungan:

Similarity

Jadi, Kasus baru memiliki kemiripan 0% dengan kasus lama Mesin Mogok Saat Digunakan (K005).

6. Kerusakan Mesin Boros Bahan Bakar (K006)

Gejala yang terkait dengan K006:

* Konsumsi bahan bakar berlebihan (G013) - Bobot = 3
* Bau bahan bakar menyengat (G014) - Bobot = 3
* Oli mesin cepat habis (G015) - Bobot = 5

Perbandingan Gejala:

|  |  |  |  |
| --- | --- | --- | --- |
| **Gejala Kasus Lama** | **Bobot** | **Gejala Kasus Baru** | **Bobot** |
| Konsumsi bahan bakar berlebihan (G013) | 3 | - | - |
| Bau bahan bakar menyengat (G014) | 3 | - | - |
| Oli mesin cepat habis (G015) | 5 | - | - |

Perhitungan:

Similarity

Jadi, Kasus baru memiliki kemiripan 0% dengan kasus lama Mesin Boros Bahan Bakar (K006).

Hasil Akhir Perhitungan CBR

|  |  |
| --- | --- |
| **Kerusakan** | **Similarity (%)** |
| Kerusakan Mesin Sulit Hidup (K001) | 100% |
| Kerusakan Mesin Overheat (K002) | 0% |
| Kerusakan Mesin Berisik (K003) | 0% |
| Kerusakan Mesin Tidak Bisa Berakselerasi (K004) | 27% |
| Kerusakan Mesin Mogok Saat Digunakan (K005) | 0% |
| Kerusakan Mesin Boros Bahan Bakar (K006) | 0% |

Jadi, Kasus baru memiliki kemiripan tertinggi dengan Mesin Sulit Hidup (K001) dengan nilai similarity 100%. Oleh karena itu, sistem akan merekomendasikan solusi untuk kerusakan Mesin Sulit Hidup.

## Analisa Kebutuhan Data



### Analisa Data Gejala Kerusakan Mesin Sepeda Motor

### Analisa Data Kerusakan Mesin Sepeda Motor

### Analisa Menetapkan *Rule* Metode *Case-Based Reasoning*

### Perakitan (*Assembly*)

Gambar 3. 4 proses pembuatan animasi 3D

Gambar 3. 5 proses pembuatan main menu

### Pengujian Aplikasi (*Testing*)

Gambar 3. 6 scene main menu

Gambar 3. 7 scene character selection

Gambar 3. 8 scene gameplay

### Distribusi (*Distribution*)

Gambar 3. 9 aplikasi tersedia di Google Play Store

## Perancangan *Unified Modeling Language (UML)*

### *Use Case Diagram*

Gambar 3. 10 use case diagram JepangCita

### *Activity Diagram*

Gambar 3. 11 activity diagram main menu

1. *Activity diagram main menu – start game*

Gambar 3. 12 activity diagram main menu – start game

1. *Activity diagram main menu – settings*

Gambar 3. 13 activity diagram main menu – settings

1. *Activity diagram main menu – credits*

Gambar 3. 14 activity diagram main menu – credits

1. *Activity diagram main menu – quit game*

Gambar 3. 15 activity diagram main menu – quit game

1. *Activity diagram gameplay*

Gambar 3. 16 activity diagram gameplay

1. *Activity diagram gameplay – mission*

Gambar 3. 17 activity diagram gameplay – mission

1. *Activity diagram gameplay – pause*

Gambar 3. 18 activity diagram gameplay – pause

1. *Activity diagram pause – resume*

Gambar 3. 19 activity diagram pause – resume

1. *Activity diagram pause – settings*

Gambar 3. 20 activity diagram pause – settings

1. *Activity diagram pause – main menu*

Gambar 3. 21 activity diagram pause – main menu

1. *Activity diagram gameplay – action*

Gambar 3. 22 activity diagram gameplay – action

1. *Activity diagram action – classroom*

Gambar 3. 23 activity diagram action – classroom

1. *Activity diagram action – bed*

Gambar 3. 24 activity diagram action – bed

1. *Activity diagram action – laptop*

Gambar 3. 25 activity diagram action – laptop

1. *Activity diagram action classroom – sit*

Gambar 3. 26 activity diagram action classroom – sit

1. *Activity diagram action laptop – materi bahasa Jepang*

Gambar 3. 27 activity diagram action laptop – materi bahasa Jepang

1. *Activity diagram update game*

Gambar 3. 28 activity diagram update game

### *Sequence Diagram*

Gambar 3. 29 sequence diagram main menu

1. *Sequence diagram – start game*

Gambar 3. 30 sequence diagram start game

1. *Sequence diagram main menu – settings*

Gambar 3. 31 sequence diagram main menu – settings

1. *Sequence diagram main menu – credits*

Gambar 3. 32 sequence diagram main menu – credits

1. *Sequence diagram main menu – quit game*

Gambar 3. 33 sequence diagram main menu – quit game

1. *Sequence diagram start game – gameplay*

Gambar 3. 34 sequence diagram start game – gameplay

1. *Sequence diagram gameplay – mission*

Gambar 3. 35 sequence diagram gameplay – mission

1. *Sequence diagram gameplay – pause*

Gambar 3. 36 sequence diagram gameplay – pause

1. *Sequence diagram pause – resume*

Gambar 3. 37 sequence diagram pause – resume

Pada gambar 3.40, setelah *admin* atau mahasiswa menekan tombol *start game* pada *scene main menu* maka, akan diarahkan ke *scene gameplay*. Jika menekan tombol *pause* maka, akan ditampilkan *panel pause* dan menghentikan permainan. Pada *panel* tersebut jika menekan tombol *resume* maka, akan menutup *panel pause* dan melanjutkan permainan.

1. *Sequence diagram pause – settings*

Gambar 3. 38 sequence diagram pause – settings

1. *Sequence diagram pause – main menu*

Gambar 3. 39 sequence diagram pause – main menu

1. *Sequence diagram gameplay – action*

Gambar 3. 40 sequence diagram gameplay – action

1. *Sequence diagram action – classroom*

Gambar 3. 41 sequence diagram action – classroom

1. *Sequence diagram action – bed*

Gambar 3. 42 sequence diagram action – bed

1. *Sequence diagram action – laptop*

Gambar 3. 43 sequence diagram action – laptop

1. *Sequence Diagram Action Classroom – Sit*

Gambar 3. 44 sequence diagram action classroom – sit

1. *Sequence diagram laptop – materi bahasa Jepang*

Gambar 3. 45 sequence diagram laptop – materi bahasa Jepang

1. *Sequence Diagram Update Game*

Gambar 3. 46 sequence diagram update game



### *Class Diagram*

Gambar 3. 47 class diagram JepangCita

## *User Interface*

1. Rancangan tampilan *main menu*

Gambar 3. 48 rancangan tampilan main menu

1. Rancangan tampilan *credits*

Gambar 3. 49 rancangan tampilan credits

1. Rancangan tampilan *settings*

Gambar 3. 50 rancangan tampilan settings

1. Rancangan tampilan *quit game*

Gambar 3. 51 rancangan tampilan quit game

1. Rancangan tampilan *character selection*

Gambar 3. 52 rancangan tampilan character selection

1. Rancangan tampilan *gameplay*

Gambar 3. 53 rancangan tampilan gameplay

1. Rancangan tampilan *mission*

Gambar 3. 54 rancangan tampilan mission

1. Rancangan tampilan *pause*

Gambar 3. 55 rancangan tampilan pause

1. Rancangan tampilan *action – laptop*

Gambar 3. 56 rancangan tampilan action – laptop

1. Rancangan tampilan *action – bed*

Gambar 3. 57 rancangan tampilan action – bed

1. Rancangan tampilan *action – classroom*

Gambar 3. 58 rancangan tampilan action – classroom

# BAB IV IMPLEMENTASI DAN PENGUJIAN



## Spesifikasi

Spesifikasi adalah serangkaian karakteristik atau persyaratan teknis yang menggambarkan fitur, fungsi, dan kinerja suatu sistem. Spesifikasi ini bertujuan untuk memastikan bahwa sistem tersebut memenuhi kebutuhan dan harapan mahasiswa serta standar yang ditetapkan.



### Spesifikasi Perangkat Lunak

Spesifikasi dalam implementasi dan pengujian *aplikasi* “Sistem Pakar Deteksi Kerusakan Mesin Sepeda Motor”menggunakan perangkat lunak sebagai berikut:

Tabel 4. 1 tabel spesifikasi perangkat lunak

|  |  |  |
| --- | --- | --- |
| **No.** | **Perangkat Lunak** | **Keterangan** |
| 1. | Sistem Operasi | *Windows 10 Pro* |
| 2. | *Integrated Development Environment (IDE)* | *Visual Studio 2019* |
| 3. | Desain Grafis | *Draw.io* |
| 4. | *Web Browser* | *Google Chrome* |
| 5. | Desain *UI/UX* | *Figma* |

### Spesifikasi Perangkat Keras

Spesifikasi dalam implementasi dan pengujian *aplikasi* “Sistem Pakar Deteksi Kerusakan Mesin Sepeda Motor”menggunakan perangkat keras *laptop* sebagai berikut:

Tabel 4. 2 tabel spesifikasi perangkat keras laptop

|  |  |  |
| --- | --- | --- |
| **No.** | **Perangkat Keras** | **Keterangan** |
| 1. | *Model* |  |
| 2. | *Processor* |  |
| 3. | *Memory RAM* |  |
| 4. | Penyimpanan Internal |  |
| 5. | Internet |  |

## Implementasi Program

Implementasi adalah tahap di mana sistem mulai dioperasikan secara nyata, sehingga dapat diketahui apakah sistem tersebut benar-benar dapat mencapai tujuan yang diinginkan.



### Tampilan *Splash Screen*

Gambar 4. 1 tampilan dari splash screen

### Tampilan *Main Menu*

Gambar 4. 2 tampilan dari main menu

### Tampilan Pengaturan

Gambar 4. 3 tampilan dari pengaturan

### Tampilan Kredit

Gambar 4. 4 tampilan dari kredit

### Tampilan Keluar Permainan

Gambar 4. 5 tampilan dari keluar permainan

### Tampilan *Character Selection*

Gambar 4. 6 tampilan dari character selection

### Tampilan *Gameplay*

Gambar 4. 7 tampilan dari gameplay

### Tampilan *Gameplay* – Pembelajaran *Online*

Gambar 4. 8 tampilan dari gameplay – pembelajaran online

### Tampilan *Classroom*

Gambar 4. 9 tampilan dari classroom

### Tampilan *Classroom –* Pembelajaran *Offline*

Gambar 4. 10 tampilan classroom – pembelajaran offline

### Tampilan *Classroom –* Ujian

Gambar 4. 11 tampilan dari classroom – ujian

## Pengujian Sistem

Pengujian sistem merupakan proses evaluasi untuk memastikan bahwa perangkat lunak yang telah dikembangkan dapat beroperasi sesuai dengan standar yang ditetapkan. Jenis pengujian sistem yang digunakan adalah *Functional Test* dan *User Acceptance Test (UAT)*.

### *Functional Testing*

*Black box testing* adalah pendekatan pengujian perangkat lunak yang mengevaluasi fungsionalitas suatu aplikasi tanpa memperhatikan struktur internal atau cara kerjanya. Metode pengujian *black box testing* memungkinkan pengembang perangkat lunak untuk menyusun serangkaian kondisi *input* yang mencakup semua persyaratan fungsional program. Pengujian dilakukan dengan memilih sejumlah modul yang mencakup berbagai jenis data untuk memastikan bahwa program hanya menerima *input* dengan jenis data yang sesuai. Selain itu, pengujian juga memeriksa antarmuka pengguna aplikasi itu sendiri. Proses pengujian *black box* pada *game* “JepangCita: Game Simulasi 3D” adalah sebagai berikut:

1. Pengujian *black box* pada *scene main menu*

Tabel 4. 3 tabel pengujian scene main menu

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Kode** | **Deskripsi** | **Hasil yang Diharapkan** | **Hasil Pengujian** | **Status** |
| PSMM1 |  |  |  | Valid |
| PSMM2 |  |  |  | Valid |
| PSMM3 |  |  |  | Valid |
| PSMM4 |  |  |  | Valid |
| PSMM5 |  |  |  | Valid |
| PSMM6 |  |  |  | Valid |
| PSMM7 |  |  |  | Valid |
| PSMM8 |  |  |  | Valid |
| PSMM9 |  |  |  | Valid |
| PSMM10 |  |  |  | Valid |
| PSMM11 |  |  |  | Valid |
| PSMM12 |  |  |  | Valid |
| PSMM13 |  |  |  | Valid |
| PSMM14 |  |  |  | Valid |

1. Pengujian *black box* pada *scene character selection*

Tabel 4. 4 tabel pengujian scene character selection

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Kode** | **Deskripsi** | **Hasil yang Diharapkan** | **Hasil Pengujian** | **Status** |
| PSCS1 |  |  |  | Valid |
| PSCS2 |  |  |  | Valid |
| PSCS3 |  |  |  | Valid |
| PSCS4 |  |  |  | Valid |

1. Pengujian *black box* pada *scene gameplay*

Tabel 4. 5 tabel pengujian scene gameplay

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Kode** | **Deskripsi** | **Hasil yang Diharapkan** | **Hasil Pengujian** | **Status** |
| PSG1 |  |  |  | Valid |
| PSG2 |  |  |  | Valid |
| PSG3 |  |  |  | Valid |
| PSG4 |  |  |  | Valid |
| PSG5 |  |  |  | Valid |
| PSG6 |  |  |  | Valid |
| PSG7 |  |  |  | Valid |
| PSG8 |  |  |  | Valid |
| PSG9 |  |  |  | Valid |
| PSG10 |  |  |  | Valid |
| PSG11 |  |  |  | Valid |
| PSG12 |  |  |  | Valid |
| PSG13 |  |  |  | Valid |
| PSG14 |  |  |  | Valid |
| PSG15 |  |  |  | Valid |
| PSG16 |  |  |  | Valid |
| PSG17 |  |  |  | Valid |
| PSG18 |  |  |  | Valid |
| PSG19 |  |  |  | Valid |
| PSG20 |  |  |  | Valid |
| PSG21 |  |  |  | Valid |
| PSG22 |  |  |  | Valid |
| PSG23 |  |  |  | Valid |
| PSG24 |  |  |  | Valid |
| PSG25 |  |  |  | Valid |
| PSG26 |  |  |  | Valid |
| PSG27 |  |  |  | Valid |
| PSG28 |  |  |  | Valid |
| PSG29 |  |  |  | Valid |
| PSG30 |  |  |  | Valid |
| PSG31 |  |  |  | Valid |
| PSG32 |  |  |  | Valid |
| PSG33 |  |  |  | Valid |
| PSG34 |  |  |  | Valid |
| PSG35 |  |  |  | Valid |
| PSG36 |  |  |  | Valid |
| PSG37 |  |  |  | Valid |
| PSG38 |  |  |  | Valid |
| PSG39 |  |  |  | Valid |
| PSG40 |  |  |  | Valid |
| PSG41 |  |  |  | Valid |
| PSG42 |  |  |  | Valid |
| PSG43 |  |  |  | Valid |
| PSG44 |  |  |  | Valid |
| PSG45 |  |  |  | Valid |
| PSG46 |  |  |  | Valid |
| PSG47 |  |  |  | Valid |
| PSG48 |  |  |  | Valid |
| PSG49 |  |  |  | Valid |
| PSG50 |  |  |  | Valid |
| PSG51 |  |  |  | Valid |
| PSG52 |  |  |  | Valid |
| PSG53 |  |  |  | Valid |
| PSG54 |  |  |  | Valid |
| PSG55 |  |  |  | Valid |
| PSG56 |  |  |  | Valid |
| PSG57 |  |  |  | Valid |

1. Pengujian *black box* pada *scene classroom*

Tabel 4. 6 tabel pengujian scene classroom

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Kode** | **Deskripsi** | **Hasil yang Diharapkan** | **Hasil Pengujian** | **Status** |
| PSC1 |  |  |  | Valid |
| PSC2 |  |  |  | Valid |
| PSC3 |  |  |  | Valid |
| PSC4 |  |  |  | Valid |
| PSC5 |  |  |  | Valid |

Berdasarkan hasil dari keempat *scene* yang diuji, total skenario uji yang berhasil mencapai 100%. Ini menunjukkan bahwa semua fitur yang diuji dalam *aplikasi* “Sistem Pakar Untuk Deteksi Kerusakan Mesin Sepeda Motor” berfungsi dengan baik sesuai dengan spesifikasi dan ekspektasi yang ditetapkan. Pengujian *functional test* ini memastikan bahwa aplikasi telah memenuhi persyaratan fungsionalnya dan siap untuk tahap pengujian berikutnya, yaitu pengujian *User Acceptance Test (UAT)*, yang akan dilakukan menggunakan kuesioner yaitu mengumpulkan umpan balik dari mahasiswa.

### Kuesioner *User Acceptance Testing*

Kuesioner adalah suatu teknik pengumpulan data yang melibatkan pemberian serangkaian pertanyaan atau pernyataan tertulis kepada responden. Tujuan dari kuesioner ini adalah mengevaluasi apakah sistem yang sedang digunakan sudah memenuhi kebutuhan mekanik, mengukur tingkat kepuasan makanik, dan mengidentifikasi *aplikasi* yang perlu ditingkatkan.

Kuesioner dalam *User Acceptance Testing* *(UAT)* berfungsi sebagai alat untuk mengumpulkan data yang penting mengenai pengalaman pengguna, sehingga *developer* dapat membuat perbaikan yang diperlukan sebelum *aplikasi* diluncurkan secara resmi. Dengan menggunakan kuesioner, diharapkan pengujian dapat memberikan wawasan yang berharga tentang bagaimana *aplikasi* diterima oleh *mekanik* dan apakah ada aspek yang perlu diperbaiki atau ditingkatkan.

#### Daftar Pertanyaan Kuesioner

Berikut adalah daftar pertanyaan yang terdapat dalam kuesioner *aplikasi* “Sistem Pakar Untuk Deteksi Kerusakan Mesin Sepeda Motor”. Berikut adalah Keterangan dari jawaban kuesioner:

1. SKB: Sangat Kurang Baik
2. KB: Kurang Baik
3. C: Cukup
4. B: Baik
5. SB: Sangat Baik

Tabel 4. 7 tabel daftar pertanyaan kuesioner

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No** | **Pertanyaan** | **SKB** | **KB** | **C** | **B** | **SB** |
| **1** | **2** | **3** | **4** | **5** |
| 1. |  |  |  |  |  |  |
| 2. |  |  |  |  |  |  |
| 3. |  |  |  |  |  |  |
| 4. |  |  |  |  |  |  |
| 5. |  |  |  |  |  |  |
| 6. |  |  |  |  |  |  |
| 7. |  |  |  |  |  |  |
| 8. |  |  |  |  |  |  |
| 9. |  |  |  |  |  |  |
| 10. |  |  |  |  |  |  |

#### Hasil Kuesioner *User Acceptance Testing*

Hasil kuesioner *User Acceptance Testing* adalah data yang dihasilkan dari pertanyaan yang sudah disebarkan dan sudah diberi jawaban oleh responden melalui *Google Forms*. Data tersebut memberikan gambaran tentang pandangan, opini, atau pengalaman responden terkait topik yang ditanyakan.

Tabel 4. 8 hasil kuesioner user acceptance testing

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Nama Responden** | **Pertanyaan** | | | | | | | | | |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
| 1. |  |  |  |  |  |  |  |  |  |  |  |
| 2. |  |  |  |  |  |  |  |  |  |  |  |
| 3. |  |  |  |  |  |  |  |  |  |  |  |
| 4. |  |  |  |  |  |  |  |  |  |  |  |
| 5. |  |  |  |  |  |  |  |  |  |  |  |
| 6. |  |  |  |  |  |  |  |  |  |  |  |
| 7. |  |  |  |  |  |  |  |  |  |  |  |
| 8. |  |  |  |  |  |  |  |  |  |  |  |
| 9. |  |  |  |  |  |  |  |  |  |  |  |
| 10. |  |  |  |  |  |  |  |  |  |  |  |
| 11. |  |  |  |  |  |  |  |  |  |  |  |
| 12. |  |  |  |  |  |  |  |  |  |  |  |
| 13. |  |  |  |  |  |  |  |  |  |  |  |
| 14. |  |  |  |  |  |  |  |  |  |  |  |
| 15. |  |  |  |  |  |  |  |  |  |  |  |
| 16. |  |  |  |  |  |  |  |  |  |  |  |
| 17. |  |  |  |  |  |  |  |  |  |  |  |
| 18. |  |  |  |  |  |  |  |  |  |  |  |
| 19. |  |  |  |  |  |  |  |  |  |  |  |
| 20. |  |  |  |  |  |  |  |  |  |  |  |
| 21. |  |  |  |  |  |  |  |  |  |  |  |
| 22. |  |  |  |  |  |  |  |  |  |  |  |

Pada tabel 4.6 adalah jawaban dari 22 responden dengan jumlah 10 pertanyaan.

#### Perhitungan Skor Kuesioner Menggunakan Skala *Likert*

Dalam perhitungan skor kuesioner yang menggunakan skala *likert*, terdapat beberapa langkah penting yang harus dilakukan untuk mendapatkan interpretasi yang akurat terhadap hasil kuesioner. Langkah-langkah perhitungannya adalah sebagai berikut:

1. Menentukan bobot skala *likert*

Bobot skala *likert* yang digunakan dalam kuesioner adalah sebagai berikut:

1. Sangat Kurang Baik (SKB): 1
2. Kurang Baik (KB): 2
3. Cukup (C): 3
4. Baik (B): 4
5. Sangat Baik (SB): 5
6. Perhitungan total skor berdasarkan pilihan responden

Adapun perhitungan jumlah responden yang mengisi kuesioner adalah sebagai berikut:

Jumlah responden: 22

Jumlah soal: 10

Total jumlah responden yang terkumpul dari kuesioner adalah:

Total jumlah responden = Jumlah responden × Jumlah soal

T = 22 x 10 = 220

Tabel 4. 9 tabel perhitungan kuesioner

|  |  |  |
| --- | --- | --- |
| **Pilihan Angka Skor Likert** | **Total Jumlah Responden yang memilih** | **Hasil** |
| Responden yang menjawab Sangat Kurang Baik (1) | **935** |  |
| Responden yang menjawab Kurang Baik (2) |  |  |
| Responden yang menjawab Cukup (3) |  |  |
| Responden yang menjawab Baik (4) |  |  |
| Responden yang menjawab Sangat Baik (5) |  |  |
| **Total Skor** | |  |

1. Menentukan skor tertinggi dan terendah

Untuk mendapatkan hasil interpretasi, diperlukan skor tertinggi (Y) dan skor terendah (X) untuk *item* penilaian dengan rumus sebagai berikut:

1. Skor Tertinggi (Y): Skor tertinggi yang dapat diperoleh dalam kuesioner. Dengan skala *likert*, skor tertinggi adalah 5 (Sangat Baik). Untuk seluruh kuesioner, skor tertinggi dihitung dengan:

𝑌 = Skor Tertinggi × Total Jumlah Responden

1. Skor Terendah (X): Skor terendah yang dapat diperoleh dalam kuesioner. Dengan skala *likert*, skor terendah adalah 1 (Sangat Kurang Baik). Untuk seluruh kuesioner, skor terendah dihitung dengan:

𝑋 = Skor Terendah × Total Jumlah Responden

Jumlah skor tertinggi untuk *item* Sangat Baik adalah 5 x 220 = 1100, sedangkan item Sangat Kurang Baik adalah 1 x 220 = 220. Jadi, jika total skor penilaian responden yang diperoleh adalah 935, maka penilaian interpretasi responden terhadap pemahaman belajar bahasa Jepang melalui *game* simulasi 3D adalah hasil nilai yang dihasilkan dengan menggunakan rumusindeks persentase (%).

1. Menghitung indeks persentase (%)

Indeks persentase digunakan untuk menentukan seberapa baik hasil penilaian responden. Rumus untuk menghitung indeks persentase adalah:

Index % = () x 100

1. Menentukan *interval* kriteria skor

*Interval* digunakan untuk mengategorikan hasil skor ke dalam kriteria tertentu. Rumus *interval* dihitung dengan:

I = ()

I = () = 20

Hasil (I) = 20 (ini adalah intervalnya jarak dari terendah 0% hingga 100%)

Berikut kriteria interpretasi skor berdasarkan *interval*:

Angka 0% - 19,99% = Sangat Buruk

Angka 20% - 39,99% = Buruk

Angka 40% - 59,99% = Cukup

Angka 60% - 79,99% = Baik

Angka 80% - 100% = Sangat Baik

1. Menginterpretasikan hasil

Index % = () x 100

Index % = () x 100 = 85%

**Penyelesaian Akhir (%) = 85% (SANGAT BAIK).**

#### Kesimpulan Kuesioner *User Acceptance Testing*

Berdasarkan hasil kuesioner yang telah dilakukan terhadap 22 m yang telah mengisi kuesioner tersebut, maka dapat disimpulkan bahwa IMPLEMENTASI APLIKASI SISTEM PAKAR UNTUK DETEKSI KERUSAKAN MESIN KENDARAAN SEPEDA MOTOR MENGGUNAKAN METODE CASE-BASED REASONING BERBASIS WEBSITE PADA BENGKEL BERKAT MOTOR CASTROL TANGERANG mendapatkan nilai 85% (Sangat Baik) dari skala 100% (Sangat Baik) dari para mahasiswa. Sebagian besar responden menilai *aplikasi* ini mendapat penilaian tinggi dalam aspek grafis, animasi, kontrol, dan antarmuka mekanik. Dalam sisi edukatif, *aplikasi* ini efektif dalam membantu mekanik mempelajari kosakata, tata bahasa, membaca, dan menulis bahasa Jepang dengan penilaian rata-rata baik hingga sangat baik. Banyak responden merasa termotivasi untuk belajar lebih lanjut setelah bermain, menunjukkan minat belajar yang tinggi. Secara keseluruhan, *game* ini dianggap layak dan bermanfaat untuk membantu pelajar dalam mempelajari bahasa Jepang melalui pendekatan permainan.

## Uji Normalitas

Uji normalitas adalah metode yang digunakan untuk menentukan apakah distribusi data mengikuti pola distribusi normal atau tidak. Terdapat banyak metode yang dapat digunakan untuk pengujian normalitas data, dan setiap metode dapat menghasilkan keputusan yang berbeda (Sintia dkk., 2022).

Dalam penelitian ini, terdapat 22 responden mahasiswa, sehingga uji normalitas dilakukan menggunakan uji *Shapiro-Wilk*. Jika *p-value* yang dihasilkan dari uji *Shapiro-Wilk* lebih besar dari tingkat signifikansi yang telah ditentukan yaitu (0,05), maka hipotesis nol () diterima, yang berarti data berdistribusi normal. Namun, jika *p-value* lebih kecil dari tingkat signifikansi, hipotesis nol () ditolak, yang menunjukkan bahwa data tidak berdistribusi normal. Kriteria untuk pengambilan keputusan dalam uji ini adalah sebagai berikut:

1. Jika p > 0,05 (), menunjukkan bahwa tidak ada bukti yang cukup untuk menolak hipotesis nol, sehingga data dianggap berdistribusi normal.
2. Jika p ≤ 0,05 (), menunjukkan bahwa ada bukti yang cukup untuk menolak hipotesis nol, sehingga data dianggap tidak berdistribusi normal.

Berikut adalah uji normalitas terhadap hasil kuesioner yang dilakukan menggunakan uji *Shapiro-Wilk*, sebagai berikut:

Tabel 4. 10 data responden dan jawaban skala likert

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Responden** | **Q1** | **Q2** | **Q3** | **Q4** | **Q5** | **Q6** | **Q7** | **Q8** | **Q9** | **Q10** |
| Mochamad Aminnur |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |

Tabel 4. 11 hasil uji normalitas pertanyaan

|  |  |  |  |
| --- | --- | --- | --- |
| ***Tests of Normality*** | | | |
|  | *Shapiro-Wilk* | | |
| *Statistic* | *df* | *Sig.* |
| Q1 |  |  |  |
| Q2 |  |  |  |
| Q3 |  |  |  |
| Q4 |  |  |  |
| Q5 |  |  |  |
| Q6 |  |  |  |
| Q7 |  |  |  |
| Q8 |  |  |  |
| Q9 |  |  |  |
| Q10 |  |  |  |

Pada tabel 4.12 menampilkan data responden berserta jawaban dalam skala *likert* setiap pertanyaan dan hasil uji normalitas sementara menggunakan uji *Shapiro-Wilk* untuk 22 responden dari 10 pertanyaan (Q1 hingga Q10).

Tabel 4. 12 hasil statistik deskriptif

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Descriptive Statistics** | | | | | |
|  | N | Minimum | Maximum | Mean | Std. Deviation |
| Q1 |  |  |  |  |  |
| Q2 |  |  |  |  |  |
| Q3 |  |  |  |  |  |
| Q4 |  |  |  |  |  |
| Q5 |  |  |  |  |  |
| Q6 |  |  |  |  |  |
| Q7 |  |  |  |  |  |
| Q8 |  |  |  |  |  |
| Q9 |  |  |  |  |  |
| Q10 |  |  |  |  |  |
| Total\_Skor |  |  |  |  |  |

Pada tabel 4.13 menampilkan hasil sistem statistik deskriptif, di mana rata-rata penilaian responden untuk semua pertanyaan berkisar antara 4,0455 hingga 4,4091, yang mengindikasikan evaluasi yang umumnya positif. Nilai standar deviasi yang rendah menunjukkan tingkat konsistensi yang tinggi dalam jawaban responden, dengan variasi yang relatif kecil. Skor total rata-rata sebesar 42,50 dari 50,00 juga memperkuat kesimpulan bahwa penilaian keseluruhan sangat positif. Secara keseluruhan, data ini menunjukkan bahwa responden memiliki pandangan yang seragam dan baik terhadap objek yang dinilai.

Tabel 4. 13 hasil akhir uji normalitas responden

|  |  |  |  |
| --- | --- | --- | --- |
| ***Tests of Normality*** | | | |
|  | *Shapiro-Wilk* | | |
| *Statistic* | *df* | *Sig.* |
| Total\_Skor |  |  |  |

Dengan menjumlahkan semua variabel pertanyaan yang di mana perhitungan setiap responden dari 10 pertanyaan (Q1 hingga Q10), maka tabel 4.14 menampilkan hasil akhir uji *Shapiro-Wilk* pada variabel “Total\_Skor” menunjukkan nilai statistik sebesar 0,922 dengan *p-value* sebesar 0,084. Karena *p-value* ini lebih besar dari ambang batas 0,05, hipotesis nol () yang menyatakan bahwa data berdistribusi diterima. Hal ini berarti bahwa distribusi data dapat dianggap normal, sehingga memenuhi asumsi normalitas yang diperlukan.

# BAB V PENUTUP



## Kesimpulan

## Saran

# DAFTAR PUSTAKA

# LAMPIRAN

Lampiran 1 kartu konsultasi mahasiswa

Lampiran 2 hasil kuesioner

Lampiran 3 data tabel kuesioner

Lampiran 4 game JepangCita dimainkan oleh mahasiswa