Information System Journal (INFOS)
Vol. 8, No. 2, November 2025, pp. 85-94
E-ISSN : 2655-142X , P-ISSN : 2655-190X, DOI:https://doi.org/10.24076/infosjournal.2025v8i02.2365

IMPLEMENTASI ALGORITMA *RULE-BASED* DAN *CLUSTERING* UNTUK APLIKASI PERAWATAN MOTOR MATIK ADAPTIF "MOTOCARE"

Rizky Alifian Ilham [1)], **Moh. Ali Romli** [2)]
[1)2)] *Informatika, Sains & Teknologi, Universitas Teknologi Yogyakarta*

email: rizky.5220411098@student.uty.ac.id [1)], ali.romli@uty.ac.id [2)]

INFO ARTIKEL

Riwayat Artikel:

Diterima September, 2025
Revisi Oktober, 2025
Terbit November, 2025

ABSTRAK

Jadwal perawatan motor matik yang bersifat statis seringkali tidak sesuai dengan kondisi penggunaan kendaraan, sehingga menyebabkan perawatan menjadi kurang efektif. Penelitian yang dilakukan bertujuan untuk mengembangkan model perawatan yang adaptif pada aplikasi "Motocare" yang mampu mempersonalisasi *interval* servis berdasarkan perilaku berkendara pengguna. Metode yang diusulkan mengombinasikan algoritma *Clustering* untuk mengelompokkan pengguna ke dalam kategori intensitas (tinggi, normal, ringan) dan algoritma *rule-based* untuk menyesuaikan target kilometer servis secara dinamis. Hasil pengujian menunjukkan model berhasil mempercepat interval servis untuk pengguna intensitas tinggi dan memperlambatnya untuk pengguna ringan. Kontribusi utama penelitian yaang dilakukan adalah sebuah sistem rekomendasi cerdas yang proaktif dan personal tanpa memerlukan instalasi perangkat keras tambahan pada kendaraan.

Kata Kunci :
Perawatan Motor; *Machine Learning*; *Rule-Based System*; *Clustering*; Aplikasi *Mobile*

ABSTRACT

Static maintenance schedules for automatic motorcycles are often misaligned with actual vehicle usage conditions, leading to less effective maintenance. The research conducted aims to develop an adaptive maintenance model in the "Motocare" application that can personalize service intervals based on user riding behavior. The proposed method combines a Clustering algorithm to group users into intensity categories (high, normal, and low) and a Rule-Based algorithm to dynamically adjust the service kilometer targets. Testing results demonstrate that the model successfully shortens the service for high-intensity users and extends it for low-intensity users. The main contribution of the research conducted is an intelligent, proactive, and personalized recommendation system that does not require additional hardware installation on the vehicle.

***Keywords*:**
Motorcycle Maintenance; Machine Learning; Rule-Based System; Clustering; Mobile Application

Penulis Korespondensi:

Rizky Alifian Ilham
Informatika, Sains & Teknologi, Universitas Teknologi Yogyakarta

Email:
rizky.5220411098@student.uty.ac.id

1. PENDAHULUAN

Transformasi digital telah menjadi pendorong inovasi di berbagai sektor, tidak terkecuali pada industri otomotif yang telah mengadopsi teknologi informasi. Di Indonesia, sepeda motor matik mendominasi pasar secara signifikan, dengan persentase penjualan mencapai 93,75% pada awal tahun 2025 [1]. Popularitas motor matik didorong oleh kemudahan penggunaan dan efisiensi bahan bakar, menjadikannya pilihan utama bagi berbagai kalangan, mulai dari pekerja hingga ibu rumah tangga [2].

Continuously Variable Transmission (CVT) memiliki kompleksitas yang lebih tinggi dibandingkan motor manual, sehingga memerlukan perhatian khusus untuk menjaga performa optimal dan mencegah kerusakan [3]. Perawatan motor matik tidak hanya melibatkan penggantian oli mesin, tetapi juga oli gardan serta servis berkala pada komponen-komponen rumit di dalam rumah CVT. Adanya beberapa item perawatan dengan interval yang berbeda menciptakan sebuah jadwal yang berlapis dan kompleks. Banyak pengguna, terutama dari kalangan awam, menghadapi kesulitan dalam mengingat dan memahami jadwal perawatan serta kerusakan yang kompleks terhadap motor matik [4]. Akibatnya, kelalaian dalam melakukan servis rutin seringkali tidak terhindarkan, yang pada akhirnya dapat berujung pada penurunan performa dan biaya perbaikan yang lebih mahal.

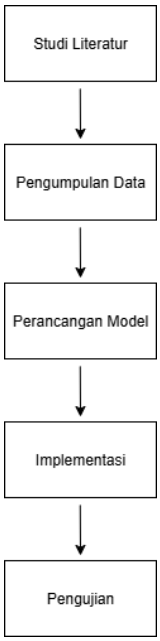
Pada penelitian sebelumnya, beberapa solusi digital telah dikembangkan untuk mengatasi masalah yang terjadi. Studi oleh Driesa dan Somya [5], berhasil membangun aplikasi pengingat servis berbasis *Android Mobile*, namun sistem pengingatnya masih bersifat manual. Penelitian lain berfokus pada pendekatan berbasis *Internet of Things* (IoT), seperti penelitian yang dilakukan oleh Alhadidi [6], yang membangun sistem pengingat pergantian minyak pelumas pelumas sepeda motor berdasarkan jarak tempuh dan waktu dibuat berbasis *Internet of Things* dengan notifikasi *telegram* dan menggunakan sensor *Real Time Clock DS3231* sebagai acuan waktu serta sensor *hall effect A3144* untuk mengetahui jarak tempuh kendaraan kemudian modul *NodeMCU ESP8266* digunakan untuk mengolah hasil pembacaan sensor. Pengembangan yang dilakukan sangat rumit untuk pemasangan bagi pengguna yang tidak mengerti dalam dunia otomotif dan pemasangan alat sensor yang kompleks. Kesenjangan inilah yang coba diatasi oleh penelitian yang dilakukan untuk menciptakan sistem cerdas yang mampu mengatur jadwal perawatan sesuai dengan perilaku berkendara pengguna tanpa memerlukan perangkat keras tambahan.

Penelitian yang dilakukan bertujuan untuk merancang dan mengimplementasikan sebuah sistem penjadwalan perawatan adaptif pada aplikasi *mobile* "Motocare" dengan memanfaatkan *machine learning*. Metode yang diusulkan adalah kombinasi algoritma *clustering* untuk mengelompokkan pengguna berdasarkan pola berkendara dan *rule-based system* untuk menyesuaikan interval servis secara dinamis. Kontribusi utama dari penelitian yang dilakukan adalah menyajikan sebuah model yang mengubah sistem pengingat manual menjadi sistem yang personal dan otomatis, di mana target kilometer servis misalnya, penggantian oli dapat dipercepat atau diperlambat sesuai dengan intensitas penggunaan kendaraan sehari-hari. Penelitian yang dilakukan memiliki batasan yaitu, pengembangan aplikasi hanya berfokus pada perawatan motor matik yang digunakan di Indonesia; sistem mengandalkan data GPS untuk kalkulasi jarak tempuh serta memanfaatkan *package flutter_activity_recognition* untuk validasi aktivitas. *Package* ini bekerja dengan menginterpretasi data dari sensor internal *smartphone* (seperti akselerometer), sehingga aplikasi 'Motocare' tidak mengolah data mentah dari sensor tersebut secara langsung. Batasan lainnya adalah sistem beroperasi tanpa perangkat tambahan serta aplikasi tidak mendiagnosis kerusakan motor yang kompleks misalnya, motor tersendat-sendat atau motor tidak bisa dinyalakan.

2. METODOLOGI PENELITIAN

Penelitian dilakukan menggunakan pendekatan kuantitatif dengan menerapkan algoritma *machine learning* untuk membangun model perawatan adaptif. Tahapan penelitian disusun secara sistematis untuk memastikan proses pengembangan berjalan terstruktur, seperti yang digambarkan pada Gambar 1.

Information System Journal (INFOS)
Vol. 8, No. 2, November 2025, pp. 85-94
E-ISSN : 2655-142X , P-ISSN : 2655-190X, DOI:https://doi.org/10.24076/infosjournal.2025v8i02.2365



Gambar 1. Alur Kerja Metode *Waterfall*.

2.1 Studi Literatur

Tahap awal penelitian berfokus pada pengumpulan dan analisis mendalam terhadap penelitian terdahulu yang relevan. Kajian pustaka mencakup topik aplikasi perawatan kendaraan, sistem pengingat statis, penerapan algoritma *clustering* untuk segmentasi pengguna, serta implementasi *rule-based system* dalam pengambilan keputusan. Tujuan dari tahap studi literatur adalah untuk memahami kelebihan dan kekurangan solusi yang sudah ada, sehingga dapat diidentifikasi celah penelitian (*research gap*) yang menjadi dasar dari inovasi yang diusulkan.

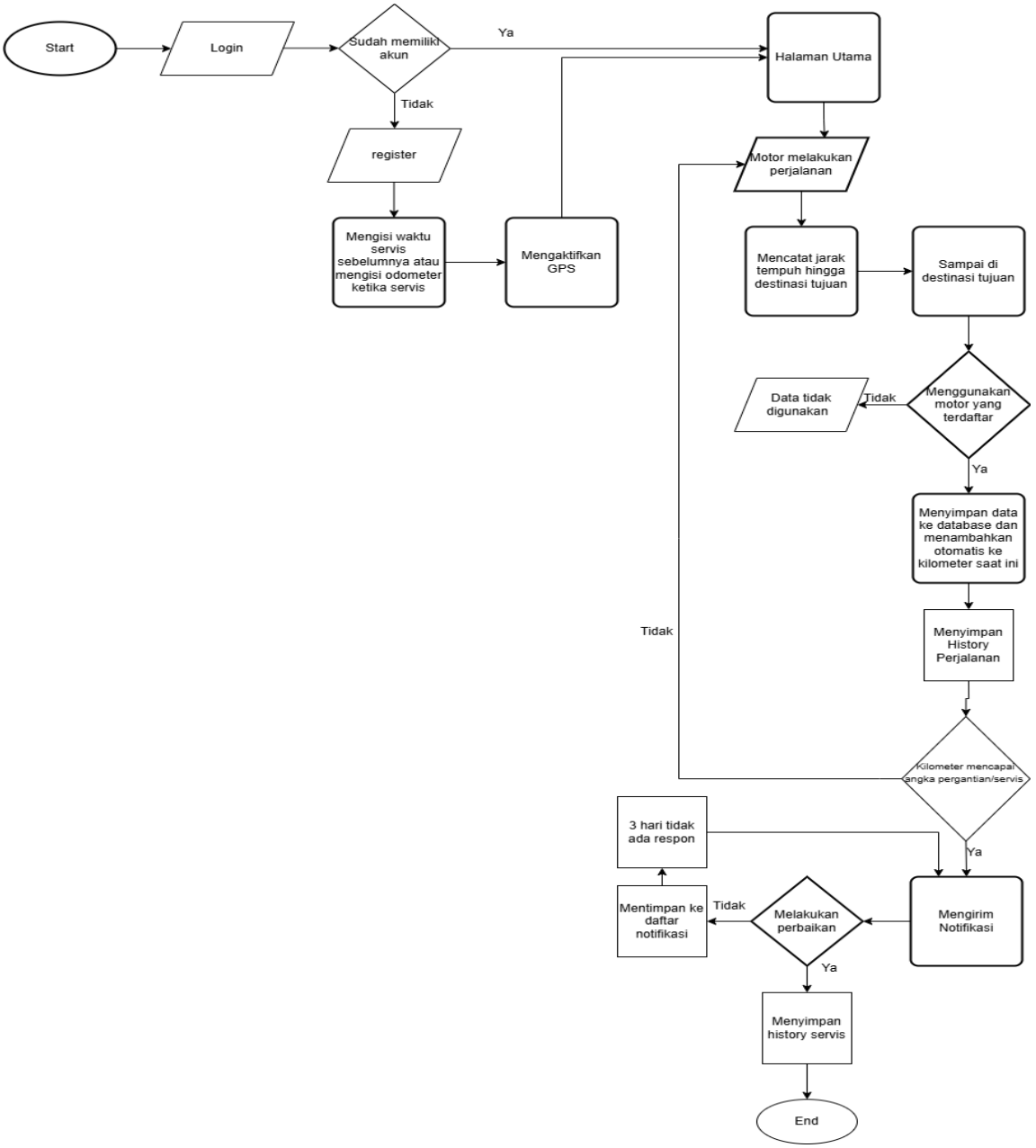
2.2 Pengumpulan Data

Tahap untuk memastikan model yang dibangun, data dikumpulkan dari 60 pengguna dengan berbagai profil berkendara selama periode 30 hari. Dataset ini mencakup lebih dari 800 data perjalanan yang terekam, dengan variasi rata-rata jarak tempuh mingguan yang signifikan (mulai dari 50 km hingga 350 km per minggu). Variasi ini penting untuk melatih model *clustering* agar dapat mengenali pola yang beragam dan representatif dari pengguna di dunia nyata. Disajikan data tersebut pada Tabel 1., berikut.

Tabel 1. Data Perjalanan Mingguan Pengguna

Id Perjalanan	Plat Nomor	Tanggal Perjalanan	Jarak Tempuh (km)
1	K 5036 AZF	2025-09-06	5.3
2	K 5723 AMF	2025-09-06	9.2
3	N 2420 PV	2025-09-08	4.8
4	AD 3256 GW	2025-09-09	14.8
...	...	...	...
795	AB 2636 DK	2025-10-05	6.6
796	K 5036 AZF	2025-10-05	8.1
797	H 3497 NH	2025-10-05	1.3
798	H 2092 FS	2025-10-06	9.4
799	K 5723 AMF	2025-10-06	3.7
800	K 5036 AZF	2025-10-06	5.5

Tahap pengumpulan data dapat disimulasikan menggunakan *flowchart* seperti pada Gambar 2. *Flowchart* merupakan penggambaran grafis dari langkah-langkah dan urutan prosedur suatu program, yang terdiri dari *flowchart sistem* yang menunjukkan urutan proses beserta media *input*, *output*, dan penyimpanannya, serta *flowchart program* yang menggunakan simbol-simbol tertentu untuk menggambarkan detail hubungan antar instruksi dalam program tersebut [7].



Gambar 2. *Flowchart* Alur Kerja Sistem

Pada tahap pengumpulan data, dilakukan pengumpulan data primer yang menjadi input bagi model *machine learning*. Seperti yang digambarkan dalam alur kerja sistem, proses pengumpulan data dimulai saat pengguna melakukan perjalanan, di mana aplikasi "Motocare" secara otomatis mengaktifkan GPS untuk mencatat parameter utama secara *real-time*, meliputi titik koordinat, kecepatan, dan waktu tempuh. Untuk memastikan validitas data, sistem melakukan verifikasi penting: data perjalanan hanya akan dihitung jika dilakukan menggunakan motor yang terdaftar.

Proses verifikasi ini tidak mengolah data mentah dari akselerometer secara langsung. Sebaliknya, sistem memanfaatkan *package Flutter 'flutter_activity_recognition'*. *Package* ini menggunakan logika internal yang sudah teruji untuk menganalisis berbagai data sensor dan secara otomatis mengidentifikasi aktivitas pengguna (misalnya, *'IN_VEHICLE'*, *'ON_FOOT'*, atau *'STILL'*). Dengan bantuan *package* ini, sistem dapat secara cerdas mencegah pencatatan yang tidak relevan, seperti saat pengguna sedang berjalan kaki atau berhenti, dan hanya fokus pada data saat pengguna benar-benar sedang berkendara. Jika perjalanan terverifikasi (dikonfirmasi sebagai aktivitas *'IN_VEHICLE'*), data akan disimpan ke dalam basis data, dan jaraknya diakumulasikan untuk memperbarui total kilometer motor. Data kilometer inilah yang kemudian diolah untuk mendapatkan metrik utama bagi model, yaitu rata-rata jarak tempuh harian dan mingguan setiap pengguna.

Information System Journal (INFOS)
Vol. 8, No. 2, November 2025, pp. 85-94
E-ISSN : 2655-142X , P-ISSN : 2655-190X, DOI:https://doi.org/10.24076/infosjournal.2025v8i02.2365

2.2.1 Pra-pemrosesan Data

Tahap pra-pemrosesan data ini berfokus secara eksklusif pada pembersihan data yang secara teknis tidak valid (*error*) dan data yang tidak logis (*outlier*) seperti, data perjalanan yang terekam oleh sistem namun gagal mencatat jarak tempuh (misalnya, menghasilkan nilai 0.0 km atau *null* akibat kegagalan kunci GPS) diidentifikasi sebagai *error* dan dihapus dari *dataset* agar tidak memengaruhi kalkulasi rata-rata.

Faktor lain selanjutnya merupakan penanganan data yang tidak logis. Perjalanan yang terekam dengan kecepatan rata-rata yang mustahil (misalnya, di atas 150 km/jam, yang mengindikasikan lompatan sinyal GPS, bukan kecepatan berkendara nyata) dianggap sebagai *outlier* teknis. Data ini dihapus agar tidak merusak total akumulasi mingguan dengan nilai yang tidak realistis.

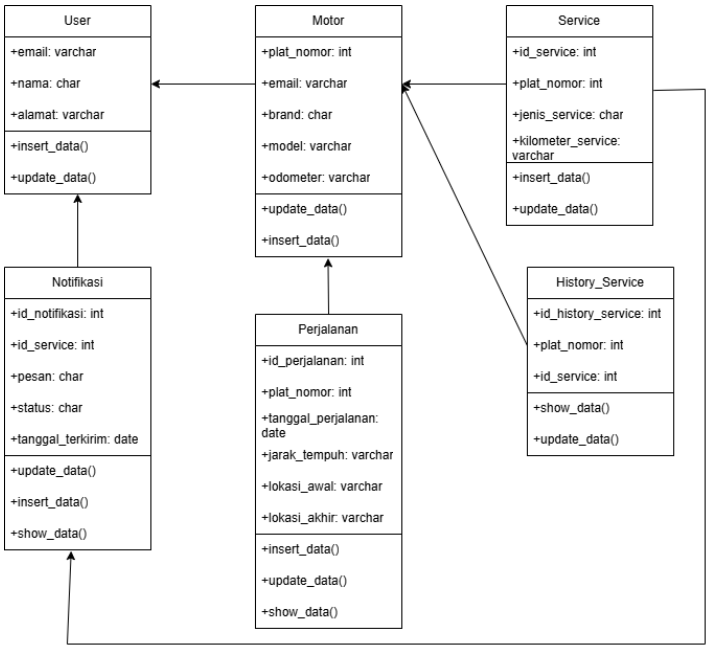
2.3 Perancangan Model

Setelah data terkumpul, dilakukan tahap perancangan yang mencakup perancangan alur kerja sistem, struktur data, dan model adaptif untuk memastikan semua komponen dapat bekerja secara terintegrasi.

2.3.1 Diagram

A. *Class Diagram* Sistem Motocare

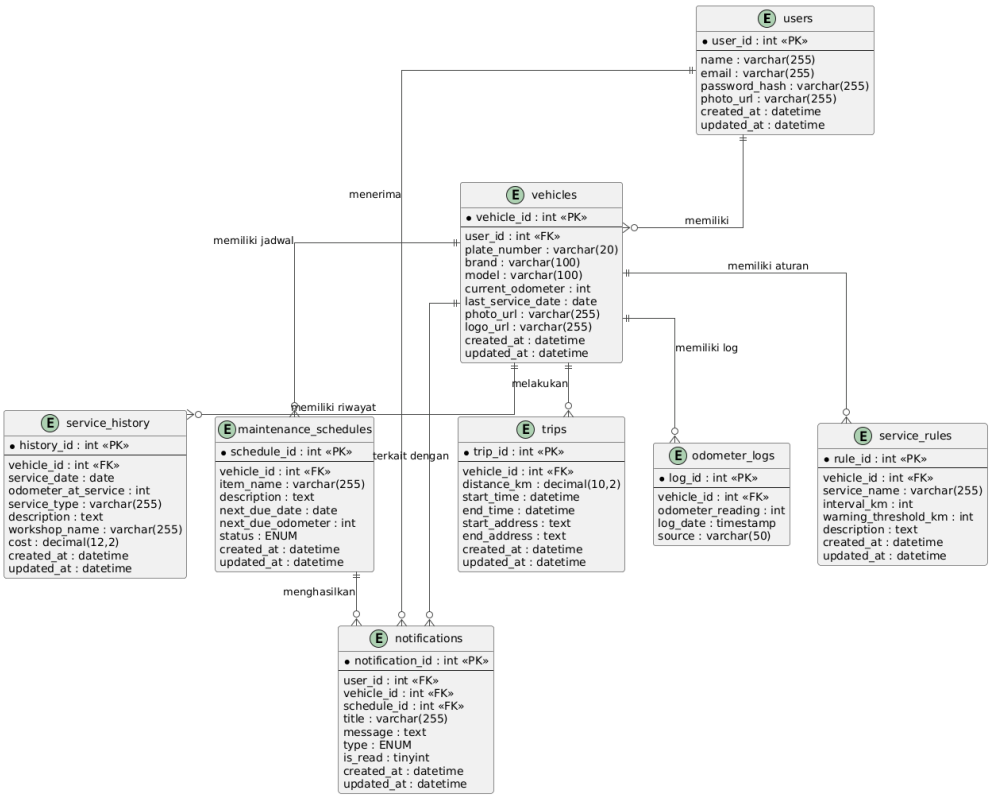
Untuk merancang struktur perangkat lunak dan interaksi antar objek dalam aplikasi, digunakan perancangan *Class Diagram* seperti yang terlihat pada Gambar 3. *Class Diagram* menggambarkan struktur sistem dengan mendefinisikan kelas-kelas yang akan dibangun untuk membentuk sistem tersebut [8]. Setiap kelas memiliki atribut dan metode atau operasi yang menjadi bagian dari fungsionalitasnya [8]. Pada sistem aplikasi "Motocare" terdapat kelas seperti, *User*, Motor, Perjalanan, *Service*, *History_Service* dan Notifikasi. Setiap kelas memiliki atribut (data) dan metode (fungsi) yang spesifik, dan relasi antar kelas digambarkan untuk memastikan struktur aplikasi yang logis dan terorganisir dengan baik pada level kode program.



Gambar 3. *Class Diagram* Sitem Motocare

B. *Entity Relationship Diagram* (ERD) Relasi Pada Sistem

Untuk memodelkan struktur basis data secara konseptual, digunakan perancangan *Entity Relationship Diagram* (ERD) seperti yang ditunjukkan pada Gambar 4. ERD mendefinisikan entitas-entitas utama yang saling berelasi [9], di mana entitas *User* dapat memiliki satu atau lebih Motor. Setiap Motor kemudian terhubung dengan data transaksionalnya, seperti Perjalanan yang mencatat riwayat perjalanan dan *Service_History* yang menyimpan catatan perawatan. Perancangan ERD krusial untuk memastikan integritas data dan menciptakan skema basis data yang efisien dan terstruktur sebelum diimplementasikan menggunakan *MySQL*.



Gambar 4. *Entity Relationship Diagram* Relasi Pada Sistem

2.3.2 Pemilihan Algoritma

Setelah struktur sistem dan data dirancang, fokus beralih pada perancangan model perawatan adaptif. Model dirancang sebagai sistem yang menggabungkan dua algoritma utama. Pertama, algoritma *Clustering* (*K-Means*) dirancang untuk mengelompokkan pengguna secara otomatis ke dalam segmen-segmen tertentu [10], (misalnya, 'Pengguna Intensitas Tinggi', 'Pengguna Normal', 'Pengguna Ringan') berdasarkan pola data perjalanan mereka.

Untuk memvalidasi pemilihan *K-Means*, dilakukan analisis komparatif dengan algoritma *clustering* lain yang umum digunakan, yaitu *K-Medoids* dan *Fuzzy C-Means*.

Setiap algoritma diuji menggunakan jumlah *cluster* (K) yang bervariasi dari 2 hingga 4. Kualitas setiap hasil *cluster* diukur menggunakan *Silhouette Score*, di mana skor yang lebih tinggi menunjukkan *cluster* yang lebih baik. Hasil perbandingan disajikan pada Tabel 2.

Tabel 2. Perbandingan Hasil *Silhouette Score* Antar Algoritma

Jumlah *Cluster* (K)	*K-Means* (Skor)	*K-Medoids* (Skor)	*Fuzzy C-Means* (Skor)
2	0.680	0.665	0.670
3	0.720	0.695	0.708
4	0.655	0.640	0.650

Hasil pengujian pada *dataset* penelitian ini menunjukkan model *clustering K-Means* (pada K=3) mencapai *Silhouette Score* sebesar 0.720. Skor ini mengindikasikan bahwa *cluster* yang terbentuk memiliki kualitas yang baik, padat, dan terpisah secara signifikan, sehingga *valid* untuk digunakan pada tahap *rule-based system*.

Perbandingan lain antara algoritma *K-Means* dengan algoritma seperti *DBSCAN*, juga telah dilakukan oleh Saputra [11] yang menyatakan algoritma *K-Means* lebih unggul dalam hal kualitas *clustering* dan waktu pemrosesan yang lebih cepat serta konsisten dibanding *DBSCAN*. Pernyataan tersebut yang membuat *K-Means* sangat cocok digunakan oleh sistem aplikasi “Motocare”.

Information System Journal (INFOS)
Vol. 8, No. 2, November 2025, pp. 85-94
E-ISSN : 2655-142X , P-ISSN : 2655-190X, DOI:https://doi.org/10.24076/infosjournal.2025v8i02.2365

Kedua, algoritma *rule-based system*, menurut Muzakkir [12], algoritma *rule-based* dapat meningkatkan akurasi dalam pengelolaan proses yang kompleks dan dinamis. Algoritma *rule-based* dirancang dengan serangkaian aturan kondisional (*IF-THEN*) untuk menyesuaikan interval servis standar berdasarkan hasil *cluster* setiap pengguna.

2.4 Implementasi

Tahap implementasi adalah proses penerjemahan hasil perancangan model dan desain antarmuka menjadi sebuah prototipe aplikasi yang fungsional. Aplikasi "Motocare" dikembangkan menggunakan *framework flutter* untuk sisi klien (*front-end*) yang berjalan di platform *Android*. Sementara itu, untuk pengelolaan data pengguna, data kendaraan, dan riwayat servis, digunakan sistem manajemen basis data *MySQL* serta penggunaan *express js* pada sisi server (*back-end*).

2.5 Pengujian

Tahap terakhir adalah pengujian untuk memvalidasi fungsionalitas dan keandalan sistem yang telah diimplementasikan. Pengujian dilakukan menggunakan metode *black box testing*, di mana fokusnya adalah pada kesesuaian *input* dan *output* tanpa melihat struktur kode internal [13]. Skenario pengujian dirancang untuk mensimulasikan berbagai pola penggunaan (intensitas tinggi dan rendah) untuk memastikan model adaptif dapat memberikan penyesuaian interval servis yang benar dan sesuai dengan aturan yang telah dirancang.

3. HASIL DAN PEMBAHASAN

Pengembangan aplikasi perawatan motor matik “Motocare” dengan menggunakan teknologi algoritma *rule-based* serta memanfaatkan perangkat GPS dam akselerometer yang sudah ada pada *smartphone* menghasilkan aplikasi yang mengubah sistem pengingat manual menjadi sistem yang personal dan otomatis.

3.1. Hasil Perancangan Model

Pengujian model adaptif dilakukan dalam dua langkah utama sesuai dengan alur kerja algoritma: pengelompokan pengguna (*clustering*) dan penyesuaian interval servis (*rule-based adjustment*).

A. Pengelompokkan Pengguna dengan Algoritma K-Means

Langkah pertama adalah mengelompokkan pengguna berdasarkan rata-rata jarak tempuh mingguan mereka. Proses pengelompokkan menggunakan algoritma K-Means karena implementasi yang mudah dan waktu komputasi yang efisien [14], K-Means juga untuk mempartisi data pengguna ke dalam tiga *cluster*: 'Intensitas Rendah', 'Intensitas Normal', dan 'Intensitas Tinggi' yang sebelumnnya telah dijelaskan pada Tabel 1. Algoritma K-Means bekerja dengan menghitung jarak setiap titik data (pengguna) ke pusat *cluster* (*centroid*) menggunakan rumus Jarak Euclidean karena merupakan cara paling sederhana [15], seperti pada Persamaan (1), rumus yang juga digunakan oleh yadav [16], untuk melakukan ujicoba perhitungan rumus jarak euclidean pada java yang menghitung jarak antara dua titik.

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(qi - pi)^2} \quad (1)$$

Keterangan:
$d(p,q)$: Jarak antara titik data p dan *centroid* q
pi : Nilai atribut titik data (rata-rata km/minggu pengguna)
qi : Nilai atribut *centroid cluster*

Pengguna akan dimasukkan ke dalam *cluster* dengan jarak *centroid* terdekat. Tabel 3 menyajikan contoh hasil dari proses *clustering* yang dilakukan.

Tabel 3. Hasil Pengelompokan (Clustering) Pengguna

Id Pengguna	Rata-rata Km/Minggu (p)	Jarak ke *Centroid* Rendah ($q = 100$)	Jarak ke *Centroid* Normal ($q = 250$)	Jarak ke *Centroid* Tinggi ($q = 1000$)	Hasil *Cluster*
User_A	85	15	165	915	Intensitas Rendah
User_B	150	50	100	850	Intensitas Rendah
User_C	310	210	60	690	Intensitas Normal
User_D	977	877	727	23	Intensitas Tinggi

Pada Tabel 3., menyajikan 4 dari 60 pengguna yang dipilih secara khusus untuk mendemonstrasikan proses kalkulasi penetapan *cluster* ‘Intensitas Rendah’ dan ‘Intensitas Tinggi’.

Pengguna akan dimasukkan ke dalam *cluster* dengan jarak *centroid* terdekat. Setelah model K-Means (K=3) dijalankan pada *dataset* penelitian, diperoleh tiga nilai *centroid* akhir yang merepresentasikan setiap *cluster*: Intensitas Rendah (q = 100 km), Intensitas Normal (q = 250 km), dan Intensitas Tinggi (q = 1000 km) berdasarkan data dari Suzuki [17]. Tabel 3., menyajikan contoh demonstrasi kalkulasi Jarak Euclidean dari beberapa pengguna ke tiga *centroid* final tersebut.

B. Penyesuaian Internal Servis dengan *Rule-Based System*

Setelah pengguna dikelompokkan, penyesuaian interval servis dilakukan menggunakan *Rule-Based System* yang menerapkan aturan kondisional (IF-THEN) berdasarkan hasil *cluster* pengguna. Secara matematis, aksi penyesuaian dari aturan tersebut dieksekusi menggunakan Formula Penyesuaian Persentase seperti yang ditunjukkan pada Persamaan (2).

$$\mathcal{T}_{baru} = \mathcal{T}_{standar} \pm \left(\mathcal{T}_{standar} \times \mathcal{P}_{penyesuaian}\right) \quad (2)$$

Keterangan:
T_{baru} : Target kilometer servis yang baru dan adaptif.
$T_{standar}$: Target kilometer servis standar dari pabrikan (misal, 2000 km).
$\mathcal{P}_{penyesuaian}$: Presentase penyesuaian (misal, 25% atau 0.25).

Nilai persentase penyesuaian (P) sebesar ±25% pada Tabel 4 ditetapkan sebagai parameter awal yang konservatif. Nilai ini ditentukan berdasarkan rujukan pada panduan servis serta standar teknis dari pakar pabrikan resmi Suzuki [17], yang bertujuan untuk memberikan perbedaan interval yang signifikan secara operasional (mempercepat atau memperlambat servis) tanpa menyimpang terlalu jauh dari rekomendasi dasar pabrikan, sehingga tetap menjaga keamanan dan garansi mesin.

Aturan spesifik dan persentase penyesuaian untuk setiap *cluster* didefinisikan dalam Tabel 4.

Tabel 4. Aturan Penyesuaian Internal Servis

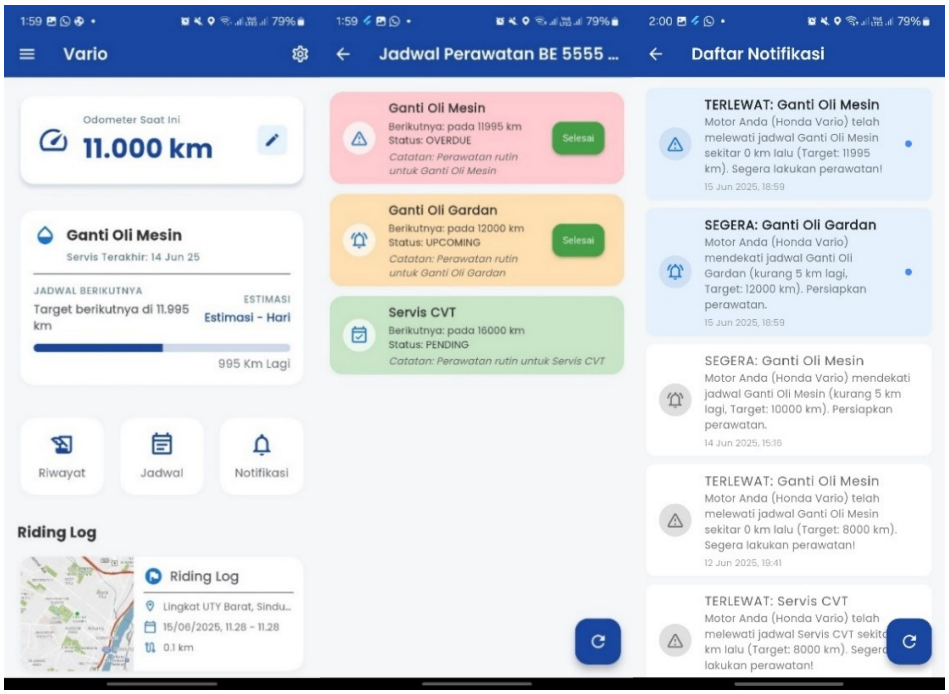
Hasil *Cluster*	Kondisi (*IF*)	Aksi (*THEN*)	Presentase Penyesuaian ($\mathcal{P}$)
Intensitas Rendah	Rata-rata Km/Minggu < 100	Perpanjang Interval Servis	+25%
Intensitas Normal	100 ≤ Rata-rata Km/Minggu < 250	Gunakan Interval Standar	0
Intensitas Tinggi	Rata-rata Km/Minggu ≥ 250	Percepat Interval Servis	-25%

3.2. Implementasi

Tahap implementasi berhasil menghasilkan sebuah prototipe aplikasi “Motocare” yang fungsional. Aplikasi yang dikembangkan dibangun menggunakan *framework* Flutter, yang dipilih karena kemampuannya dalam menciptakan antarmuka pengguna yang responsif dan konsisten pada *platform* Android dari satu basis kode tunggal. *Package* flutter_activity_recognition juga digunakan untuk mencegah data tidak relevan yang dihasilkan oleh akselerometer dari *smartphone* pengguna. Untuk menopang arsitektur sistem, MySQL diimplementasikan sebagai sistem manajemen basis data. Basis data yagn digunakan berperan sebagai tulang punggung untuk mengelola seluruh data secara terpusat dan terstruktur, mencakup informasi krusial seperti data pengguna, detail kendaraan yang terdaftar, dan seluruh riwayat perawatan.

Implementasi antarmuka fungsional aplikasi "Motocare" disajikan pada Gambar 5. Tampilan antarmuka ini dirancang secara terintegrasi untuk menyajikan informasi perawatan yang kompleks menjadi visual yang mudah dipahami.

Information System Journal (INFOS)
Vol. 8, No. 2, November 2025, pp. 85-94
E-ISSN : 2655-142X , P-ISSN : 2655-190X, DOI:https://doi.org/10.24076/infosjournal.2025v8i02.2365



Gambar 5. Tampilan Antarmuka Aplikasi (dari kiri ke kanan: Halaman Utama, Halaman Jadwal, dan Halaman Notifikasi)

Antarmuka aplikasi (kiri ke kanan) berupa Halaman Utama berfungsi sebagai dasbor utama, menampilkan informasi krusial seperti odometer *real-time* dan bilah progres visual menuju servis terdekat. Halaman Jadwal Perawatan (tengah) memberikan pandangan terpusat untuk semua item perawatan, menggunakan format kartu (*card*) dengan label status yang jelas (misal: OVERDUE, UPCOMING) dan tombol "Selesai" untuk memperbarui siklus perawatan. Terakhir, Halaman Daftar Notifikasi (kanan) berfungsi sebagai arsip untuk semua pemberitahuan sistem, yang dikategorikan berdasarkan urgensinya (misal: ‘SEGERA’ atau ‘TERLEWAT’) dan menyajikan detail kilometer yang relevan.

3.3. Pengujian

Sesuai dengan metodologi yang telah ditetapkan, tahap pengujian fungsional dilakukan untuk memvalidasi keandalan dan kesesuaian sistem. Pengujian yang dilakukan menggunakan metode *Black Box*, di mana fokus utamanya adalah untuk memastikan bahwa output yang dihasilkan sistem sesuai dengan input yang diberikan, tanpa perlu melihat struktur kode internalnya. Skenario pengujian dirancang untuk mensimulasikan berbagai pola penggunaan (intensitas tinggi dan rendah) guna memverifikasi bahwa model adaptif dapat memberikan penyesuaian interval servis yang benar dan logis. Hasil dari serangkaian skenario pengujian fungsional disajikan pada Tabel 5.

Tabel 5. Hasil Pengujian *Black box*

Id Skenario	Deskripsi Skenario	Input Sistem	Output yang Diharapkan	Output Aktual Sistem	Hasil
Uji_01	Pengguna Intensitas Rendah	Rata-rata Km/Minggu: 85 Km	2500 Km	2500 Km	Berhasil
Uji_02	Pengguna Intensitas Rendah	Rata-rata Km/Minggu: 150 Km	2500 Km	2500 Km	Berhasil
Uji_03	Pengguna Intensitas Normal	Rata-rata Km/Minggu: 310 Km	2000 Km	2000 Km	Berhasil
Uji_04	Pengguna Intensitas Tinggi	Rata-rata Km/Minggu: 977 Km	1500 Km	1500 Km	Berhasil

Berdasarkan hasil pengujian yang dirangkum pada Tabel 5, seluruh skenario yang dijalankan menunjukkan hasil 'Berhasil', yang berarti output aktual dari sistem sepenuhnya sesuai dengan output yang diharapkan. Pada skenario UJI-03, sistem secara akurat mengidentifikasi pengguna sebagai 'Intensitas Tinggi' dan menerapkan aturan untuk mengurangi interval servis menjadi 1500 km. Kondisi yang terjadi memvalidasi bahwa logika *rule-based* untuk perawatan preventif telah berfungsi dengan benar.

Demikian pula pada skenario UJI-01, sistem dengan tepat memperpanjang interval servis menjadi 2500 km untuk pengguna 'Intensitas Rendah', membuktikan bahwa logika untuk efisiensi perawatan juga berjalan

sesuai fungsionalitasnya. Secara keseluruhan, hasil pengujian *Black Box* yang dilakukan mengonfirmasi bahwa sistem aplikasi "Motocare" telah memenuhi semua persyaratan fungsional yang ditentukan. Model adaptif terbukti andal dalam memberikan rekomendasi yang benar dan logis, sehingga valid untuk menjawab permasalahan penelitian.

4. KESIMPULAN

Berdasarkan hasil penelitian dan pengujian, dapat disimpulkan bahwa implementasi model hibrida yang mengombinasikan algoritma *K-Means Clustering* dan *Rule-Based System* berhasil menciptakan aplikasi perawatan motor matik "Motocare" yang adaptif dan personal. Validasi melalui pengujian *Black Box* menunjukkan bahwa model mampu secara andal menyesuaikan interval servis untuk berbagai skenario pengguna, dengan seluruh hasil pengujian fungsional yang menunjukkan kesesuaian dengan *output* yang diharapkan. Kontribusi utama dari penelitian yang dilakukan adalah transformasi sistem pengingat manual menjadi sebuah sistem rekomendasi cerdas dan otomatis yang dapat beroperasi tanpa memerlukan instalasi perangkat keras tambahan pada kendaraan.

DAFTAR PUSTAKA

[1] L. Anshori, "Penjualan Motor di Indonesia Januari 2025 Tembus Setengah Juta Unit," detikOto. Accessed: Oct. 30, 2025. [Online]. Available: https://oto.detik.com/motor/d-7771383/penjualan-motor-di-indonesia-januari-2025-tembus-setengah-juta-unit

[2] Rivai, "5 Alasan Motor Matic Mendominasi Jalan Raya | IDN Times," IDN TIMES. Accessed: Oct. 30, 2025. [Online]. Available: https://www.idntimes.com/automotive/motorbike/5-alasan-motor-matic-mendominasi-jalan-raya-c1c2-01-r3f23-sk94j8

[3] A. C. D. Akhsa, "Sistem Pakar Diagnosa Kerusakan Motor Matic Menggunakan Metode Forward Chaining," MALCOM: Indonesian Journal of Machine Learning and Computer Science, vol. 4, no. 2, pp. 452–462, Feb. 2024, doi: 10.57152/malcom.v4i2.1189.

[4] Y. Kristianto, "Sistem Pakar Diagnosa Kerusakan Kendaraan Bermotor Jenis Matic Menggunakan Metode Naive Bayes," JIK), vol. 8, no. 1, 2024, doi: https://doi.org/10.59697/jik.v8i1.473.

[5] Y. Kehat Driesa and R. Somya, "Perancangan Aplikasi Service Reminder Sepeda Motor Berbasis Android Mobile," June25 th, p. 201, 2023, doi: https://doi.org/10.35957/jatisi.v10i1.2705.

[6] M. B. Alhadidi, D. Erwanto, and R. F. Rizal, "Sistem Pengingat Penggantian Minyak Pelumas Sepeda Motor Berbasis IoT," vol. 11, no. 2, pp. 65–71, 2024, doi: https://doi.org/10.52234/tb.v11i2.310.

[7] Erlianda Cibro and Yahfizham, "Mengidentifikasi Dasar Algoritma Pemrograman," Jurnal ilmiah Sistem Informasi dan Ilmu Komputer, vol. 4, no. 1, pp. 194–206, Mar. 2024, doi: 10.55606/juisik.v4i1.763.

[8] S. W. Ramdany, S. Aulia Kaidar, B. Aguchino, C. Amelia, A. Putri, and R. Anggie, "Penerapan UML Class Diagram dalam Perancangan Sistem Informasi Perpustakaan Berbasis Web," Journal of Industrial and Engineering System, vol. 5, no. 1, pp. 30–41, Jun. 2024, doi: https://doi.org/10.31599/2e9afp31.

[9] A. Simanungkalit and A. H. Lubis, "Sistem Informasi Keluar Masuk Barang Berbasis Website Pada Telkom STO Cinta Damai," Jurnal Ilmiah Teknik Informatika & Elektro (JITEK), vol. 2, no. 1, pp. 14–27, Jun. 2023, doi: 10.31289/jitek.v2i1.1895.

[10] P. sari and R. Syahri, "Algoritma K-Means Clustering: Sebuah Studi Literatur," Jurnal Informatika (JURI), pp. 1–7, Jan. 2023, doi: 10.12345/juri.

[11] A. Saputra and R. Yusuf, "Perbandingan Algoritma DBSCAN dan K-MEANS dalam Segmentasi Pelanggan Pengguna Transportasi Publik Transjakarta Menggunakan Metode RFM," MALCOM: Indonesian Journal of Machine Learning and Computer Science, vol. 4, no. 4, pp. 1346–1361, Jul. 2024, doi: https://doi.org/10.57152/malcom.v4i4.1516.

[12] H. H. Muzakkir, "Implementasi Algoritma Rule Based Pada Sistem Pengajuan Cuti Karyawan Berbasis Web Di Cv. Sumber Bahagia Printing," Semarang, Jun. 2024. Accessed: Oct. 30, 2025. [Online]. Available: https://eprints3.upgris.ac.id/id/eprint/4211/1/HUZAIFAH%20HAMYU%20MUZAKKIR.pdf

[13] S. Dika Pratama and M. Noviarsyah Dadaprawira, "Pengujian Black Box Testing Pada Aplikasi Edu Digital Berbasis Website Menggunakan Metode Equivalence Dan Boundary Value," Jurnal Teknologi Sistem Informasi dan Sistem Komputer TGD, vol. 6, no. 2, pp. 560–569, Jul. 2023, doi: https://doi.org/10.53513/jsk.v6i2.8166.

[14] Jaelani, Octaviana, and E. Rilvani, "Studi Literatur: Perbandingan Algoritma K-Means dan Fuzzy C-Means dalam Analisis Clustering," Journal of Computer Science and Technology JCS-TECH, vol. 5, no. 2, pp. 52–58, Aug. 2025, Accessed: Oct. 30, 2025. [Online]. Available: https://mail.journalunwidha.com/index.php/jcstech/article/view/382

[15] A. Rahayu and S. Dewi, "Klasterisasi Pemilihan Paket Umrah Berdasarkan Musim untuk Menentukan Strategi Promosi Menggunakan Algoritma K-Means," DIMAMU, vol. 4, no. 1, pp. 83–96, Dec. 2024, doi: https://doi.org/10.32627/dimamu.v4i1.1326.

[16] S. Yadav, "Calcular la distancia euclidiana en Java | Delft Stack," DelftStack. Accessed: Sep. 07, 2025. [Online]. Available: https://www.delftstack.com/es/howto/java/euclidean-distance-java/

[17] PRSuzuki, "Catat Ini Jarak Ideal Ganti Oli Motor Matic | Suzuki Indonesia," SUZUKI. Accessed: Oct. 30, 2025. [Online]. Available: https://www.suzuki.co.id/tips-trik/catat-ini-jarak-ideal-ganti-oli-motor-matic?pages=all