

PENERAPAN ALGORITMA *FORWARD CHAINING* DAN *CERTAINTY FACTOR* PADA APLIKASI *SELF ASSESMENT SMARTPHONE REPAIR SERVICE*

Hilmy Sukma Nurmakarim¹, Aries Suharso², Asep Jamaludin³

^{1,2,3}Program Studi Informatika, Fakultas Ilmu Komputer, Universitas Singaperbangsa Karawang, JL. HS. Ronggowaluyo, Puseurjaya, Telukjambe Timur, Karawang 41361, Jawa Barat, Indonesia. Email: 2110631170133@student.unsika.ac.id

Keywords:

Forward Chaining; Certainty Factor; Sistem Pakar; Self Assesment; Smartphone Repair Service.

Correspondent Email:

2110631170133@student.unsika.ac.id

Abstrak. Meningkatnya penggunaan smartphone di Indonesia berbanding lurus dengan meningkatnya kebutuhan layanan perbaikan akibat kerusakan perangkat. Proses diagnosis awal di konter servis konvensional sering memerlukan waktu lama karena pengguna tidak memahami gejala kerusakan secara teknis. Penelitian ini menerapkan algoritma *Forward Chaining* dan *Certainty Factor* pada aplikasi *self assessment* berbasis Android untuk membantu pengguna melakukan diagnosis awal kerusakan smartphone secara mandiri. Aplikasi dikembangkan menggunakan metode *Software Development Life Cycle* (SDLC) model *Waterfall* dengan *framework Flutter* dan basis data *Firebase*. Basis pengetahuan disusun melalui wawancara dengan pakar teknisi servis di Toko Pertemuan Cell, Karawang, sehingga dihasilkan aturan IF-THEN beserta bobot *Certainty Factor* pakar. *Forward Chaining* berperan sebagai mesin inferensi yang mencocokkan gejala dengan aturan, sedangkan *Certainty Factor* menghitung tingkat kepastian diagnosis dari kombinasi nilai keyakinan pengguna dan pakar. Pengujian dilakukan menggunakan *Black-Box Testing* dan *User Acceptance Testing* (UAT) terhadap 15 responden. Hasil pengujian *black box* menunjukkan seluruh fitur utama berjalan sesuai kebutuhan fungsional, sedangkan hasil UAT memperoleh skor rata-rata 4,49 dari skala 5 yang berada pada kategori setuju hingga sangat setuju. Hasil menunjukkan aplikasi dapat diterima dengan baik sebagai alat bantu diagnosis awal kerusakan smartphone yang informatif dan mudah digunakan oleh pengguna awam.



Copyright © 2026 The Author(s), Published by Jurnal Informatika dan Teknik Elektro Terapan. This is an open-access article distributed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (CC BY-NC 4.0).

Abstract. The increasing use of smartphones in Indonesia has been accompanied by a growing demand for repair services resulting from device malfunctions. Initial diagnostic process at conventional repair shops often requires considerable time because users generally do not understand technical symptoms of smartphone damage. This study implements *Forward Chaining* and *Certainty Factor* algorithms in an Android-based self-assessment application to assist users in performing smartphone damage pre-diagnosis independently. Application developed using *Software Development Life Cycle* (SDLC) *Waterfall* model with *Flutter framework* and *Firebase*. Knowledge base was constructed through interviews with an *Pertemuan Cell* expert technician, Karawang, resulting in a set of IF-THEN rules along with expert-assigned *Certainty Factor* weights. *Forward Chaining* functions as inference engine that matches user-reported symptoms with predefined rules, while *Certainty Factor* calculates level of diagnostic certainty by combining confidence values provided by user and expert. Testing was conducted using *Black-Box Testing* and *User Acceptance Testing* (UAT) on 15 respondents. *Black box* testing results show that all main features function according to

specified functional requirements, while UAT achieved an average score of 4.49 out of 5, corresponding to agree to strongly-agree category. Results indicate application is well accepted as an informative and user-friendly tool for smartphone pre-diagnosis, particularly for non-technical users.

1. PENDAHULUAN

Di era digital saat ini, penggunaan *smartphone* telah menjadi kebutuhan dasar bagi banyak orang, tidak hanya sebagai alat komunikasi tetapi juga sebagai sarana pendukung pekerjaan, hiburan, dan akses informasi. Data GoodStats mencatat sekitar 209,3 juta pengguna *smartphone* di Indonesia pada tahun 2023 [1], sedangkan laporan DataReportal mencatat 353,3 juta koneksi *smartphone* aktif di Indonesia pada awal tahun 2023, atau setara 126,8% dari total populasi [2]. Bahkan, kepemilikan *smartphone* kini meluas mulai dari anak-anak hingga orang dewasa, dengan sebagian pengguna memiliki lebih dari satu perangkat [3].

Meningkatnya kepemilikan *smartphone* berbanding lurus dengan meningkatnya kebutuhan layanan perbaikan akibat kesalahan penggunaan maupun kerusakan perangkat, seperti layar retak, baterai melemah, hingga kerusakan komponen internal. Beberapa penelitian telah mengembangkan sistem informasi berbasis web untuk mendukung proses bisnis reparasi *smartphone* [4], namun dalam praktiknya banyak pelanggan tetap datang ke konter servis tanpa memahami gejala kerusakan secara teknis, sehingga proses identifikasi masalah memerlukan waktu lama dan komunikasi berulang. Hasil wawancara dengan pakar teknisi servis di Toko Pertemuan Cell, Karawang, menunjukkan bahwa lebih dari 266 unit *smartphone* ditangani untuk keperluan servis dalam periode satu bulan (September 2025), dengan jenis kerusakan yang paling dominan berupa layar LCD, baterai, *port charging*, dan IC *power*. Kondisi ini memperkuat urgensi pengembangan aplikasi yang dapat membantu pengguna mengenali indikasi awal kerusakan secara mandiri sebelum melakukan perbaikan lebih lanjut.

Salah satu pendekatan yang dapat digunakan untuk membantu proses identifikasi

tersebut adalah sistem pakar (*expert system*) berbasis *Rule-Based System*, yaitu kerangka kerja yang mengeksekusi aturan *if-then* untuk mereplikasi penalaran seorang pakar [5]. Penelitian terdahulu menunjukkan bahwa kombinasi algoritma *Forward Chaining* dan *Certainty Factor* efektif digunakan pada sistem pakar diagnosis kerusakan *smartphone* karena mampu menelusuri gejala menuju kesimpulan sekaligus mengakomodasi ketidakpastian pakar dan pengguna [6], [7].

Namun demikian, kedua penelitian tersebut serta beberapa penelitian layanan reparasi sejenis masih diimplementasikan pada platform web [4], [8], sehingga belum banyak diterapkan pada aplikasi *mobile* Android yang lebih mudah diakses oleh pengguna awam kapan saja dan di mana saja. Celah inilah yang mendasari kebaruan penelitian ini, yaitu penerapan *Forward Chaining* dan *Certainty Factor* pada aplikasi *self-assessment* berbasis Android dengan *Flutter* dan *Firebase*, yang basis pengetahuannya disusun dari hasil akuisisi langsung dari pakar teknisi lapangan.

Berdasarkan permasalahan tersebut, penelitian ini bertujuan merancang dan membangun aplikasi *self-assessment smartphone repair service* berbasis Android yang menerapkan algoritma *Forward Chaining* sebagai mesin inferensi berbasis aturan dan *Certainty Factor* untuk menghitung tingkat kepastian diagnosis. Aplikasi dikembangkan menggunakan metode *Software Development Life Cycle* (SDLC) model *Waterfall* agar proses pembangunan sistem berjalan terstruktur dan terdokumentasi, dengan *Flutter* sebagai *framework* pengembangan *front-end* dan *Firebase* sebagai *back-end* basis data. Melalui aplikasi ini, pengguna dapat memilih gejala kerusakan beserta frekuensi kemunculannya, kemudian sistem menganalisis data tersebut untuk menentukan jenis dan tingkat kerusakan

smartphone beserta rekomendasi solusi perbaikan

2. TINJAUAN PUSTAKA

2.1 Sistem Pakar dan Rule-Based System

Sistem pakar merupakan penerapan pengetahuan seorang pakar ke dalam komputer, sehingga pengguna non-pakar dapat memperoleh kemampuan pemecahan masalah selayaknya berkonsultasi dengan seorang pakar [6]. Salah satu pendekatan yang umum digunakan dalam pengembangan sistem pakar adalah *Rule-Based System* (RBS), yaitu kerangka kerja yang mengeksekusi aturan berbentuk pernyataan *if-then* berdasarkan kondisi tertentu; pendekatan berbasis aturan ini terbukti mampu meningkatkan efisiensi dan kepatuhan proses pada penerapan aplikasi penjadwalan dan pelaporan operasional [5].

2.2 Forward Chaining

Forward Chaining adalah teknik inferensi yang bekerja dari fakta awal menuju kesimpulan, dengan mencocokkan fakta terhadap aturan yang tersedia pada basis pengetahuan; apabila kondisi pada suatu aturan terpenuhi, sistem mengeksekusi kesimpulan yang bersesuaian [9]. Komponen utama pendekatan ini terdiri atas basis pengetahuan (*knowledge base*) berupa kumpulan aturan dan fakta, mesin inferensi (*inference engine*) yang mencocokkan fakta dengan aturan, serta *working memory* sebagai ruang penyimpanan sementara fakta aktif selama proses inferensi berlangsung. Penelitian oleh Nengsih dan Putra menerapkan *Forward Chaining* untuk diagnosis kerusakan smartphone dan menunjukkan bahwa metode ini mampu menelusuri gejala menuju kandidat kerusakan secara sistematis [7], sedangkan Hasanah . menerapkan pendekatan serupa pada diagnosis penyakit ayam, menegaskan fleksibilitas *Forward Chaining* pada berbagai domain diagnosis [9].

2.3 Certainty Factor

Certainty Factor (CF) digunakan untuk mengukur tingkat keyakinan terhadap suatu kesimpulan berdasarkan bukti atau gejala yang

diberikan. Fitriyani menerapkan kombinasi *Forward Chaining* dan *Certainty Factor* pada sistem pakar diagnosis kerusakan smartphone Android dan menunjukkan bahwa pendekatan ini meningkatkan akurasi diagnosis karena turut mempertimbangkan tingkat keyakinan pakar dan pengguna secara bersamaan [6]. Metode *Certainty Factor* digunakan untuk mengakomodasi ketidakpastian dengan menggabungkan tingkat keyakinan pakar dan pengguna terhadap suatu gejala [24]. Kombinasi kedua metode inilah yang memungkinkan sistem tetap memberikan hasil diagnosis meskipun gejala yang dimasukkan pengguna tidak sepenuhnya cocok dengan aturan pada basis pengetahuan, dengan tetap menampilkan nilai kepastian sebagai indikator seberapa kuat diagnosis tersebut dapat dipercaya.

2.4 Aplikasi Mobile dan Metode Waterfall

Aplikasi dapat diartikan sebagai program siap pakai yang dirancang untuk menjalankan fungsi tertentu bagi pengguna, termasuk aplikasi *mobile* yang berjalan pada perangkat bergerak seperti smartphone [10]. *Flutter* merupakan *framework open-source* yang dikembangkan Google berbasis bahasa pemrograman *Dart*, yang memungkinkan satu basis kode digunakan untuk berbagai platform sehingga mempercepat proses pengembangan aplikasi [11]; penerapan *Flutter* dan *Dart* pada aplikasi layanan bengkel berbasis Android juga terbukti mempermudah pengembangan antarmuka yang responsif [12], sebagaimana juga diterapkan pada pengembangan sistem informasi *tracer study* berbasis *mobile* [13]. Sebagai basis data, penggunaan layanan *backend* berbasis *cloud* dinilai efektif dalam meningkatkan efisiensi pengelolaan data secara terpusat dan *real-time* [14].

Dari sisi metode pengembangan, *Software Development Life Cycle* (SDLC) model *Waterfall* banyak diterapkan pada pengembangan aplikasi layanan berbasis Android karena tahapannya yang berurutan, terintegrasi, dan terdokumentasi dengan baik pada setiap fase [15], [16]. Penelitian

pengembangan aplikasi layanan jasa berbasis Android dan web, seperti Prasetyo yang menerapkan metode *Scrum* untuk aplikasi Mitra Obre-Jek [17] dan Maulana yang membangun layanan reparasi elektronik berbasis web dengan *Laravel* [8], menunjukkan bahwa domain layanan jasa reparasi maupun antar-jemput banyak diadaptasi ke platform digital, tetapi penerapannya pada layanan diagnosis mandiri (*self-assessment*) kerusakan smartphone berbasis *mobile* masih terbatas.

Berbeda dari penelitian-penelitian tersebut, penelitian ini berfokus pada aplikasi *mobile* Android yang dibangun dengan *framework* *Flutter* dan basis data *Firebase*, dengan basis pengetahuan disusun dari hasil akuisisi pengetahuan langsung terhadap pakar teknisi lapangan di Toko Pertemuan Cell, Karawang, serta menyediakan penjelasan proses inferensi (*explanation facility*) agar hasil diagnosis lebih transparan bagi pengguna awam

3. METODE PENELITIAN

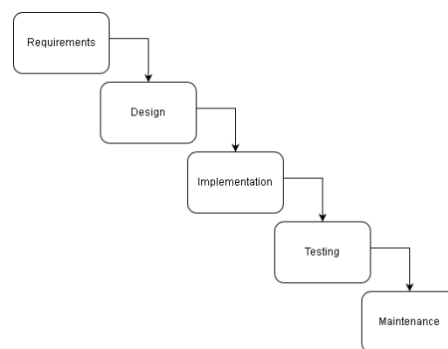
3.1 Objek dan Sumber Data Penelitian

Objek penelitian ini adalah aplikasi *Self Assessment Smartphone Repair Service* berbasis Android yang menerapkan algoritma *Forward Chaining* dan *Certainty Factor* untuk diagnosis awal kerusakan smartphone. Akuisisi pengetahuan dilakukan melalui wawancara terstruktur dengan seorang pakar teknisi servis di Toko Pertemuan Cell, Karawang, yang berpengalaman menangani berbagai jenis kerusakan smartphone. Pada proses wawancara, peneliti mengajukan daftar kerusakan yang telah terdokumentasi, kemudian pakar diminta mengidentifikasi dan mengonfirmasi gejala yang umumnya muncul pada setiap jenis kerusakan, sekaligus menentukan tingkat keyakinan hubungan antara gejala dan kerusakan sebagai bobot *Certainty Factor* pakar (CFexpert). Gambaran volume dan pola kerusakan yang paling sering ditangani bersifat estimasi berdasarkan pengalaman pakar, digunakan semata sebagai gambaran pendukung urgensi penelitian dan pertimbangan cakupan basis pengetahuan,

bukan sebagai data kuantitatif populasi kerusakan smartphone secara umum.

3.2 Metode Pengembangan Sistem

Penelitian ini menggunakan metode *Software Development Life Cycle* (SDLC) model *Waterfall*, yang mencakup tahap analisis kebutuhan, desain, implementasi, pengujian, dan pemeliharaan secara berurutan [15]. Metode ini dipilih karena memungkinkan dokumentasi yang jelas dan terperinci pada setiap fase pengembangan aplikasi layanan berbasis Android [16]. Tahapan penelitian ditunjukkan pada Gambar 1.



Gambar 1. Tahapan Penelitian Metode Waterfall

1) *Requirement Gathering and Analysis*. Tahap ini mengumpulkan dan menganalisis kebutuhan sistem melalui wawancara dan observasi dengan teknisi smartphone berpengalaman, guna memperoleh gambaran proses diagnosis manual serta menyusun dokumen Spesifikasi Kebutuhan Perangkat Lunak (SKPL) sebagai acuan tahap berikutnya.

2) *Design*. Perancangan dilakukan terhadap antarmuka pengguna (UI) [18] dan diagram *Unified Modeling Language* (UML) berupa *Use Case Diagram*, *Activity Diagram*, dan *Class Diagram* [19] untuk memetakan alur kerja sistem, serta perancangan arsitektur basis data yang menyimpan gejala, aturan (*rule*), jenis kerusakan, dan bobot *Certainty Factor*.

3) *Implementation*. *Front-end* dikembangkan menggunakan *Flutter*, sedangkan *Firebase Firestore* digunakan sebagai basis data untuk menyimpan basis pengetahuan dan riwayat diagnosis. Mekanisme inferensi *Forward Chaining* dan perhitungan

Certainty Factor diimplementasikan agar sistem mampu memproses input gejala dan menghasilkan diagnosis kerusakan.

4) *Testing*. Pengujian dilakukan menggunakan *Black-Box Testing* untuk memvalidasi fungsionalitas sistem berdasarkan masukan dan keluaran tanpa menelaah struktur internal kode [20], dikombinasikan dengan *User Acceptance Testing* (UAT) untuk menilai penerimaan pengguna.

5) *Maintenance*. Tahap pemeliharaan dilakukan untuk menjaga kestabilan sistem, memperbaiki kesalahan yang ditemukan setelah penggunaan, serta mencatat rekomendasi pengembangan lanjutan berdasarkan hasil evaluasi.

3.3 Rancangan Alur Inferensi Sistem Pakar

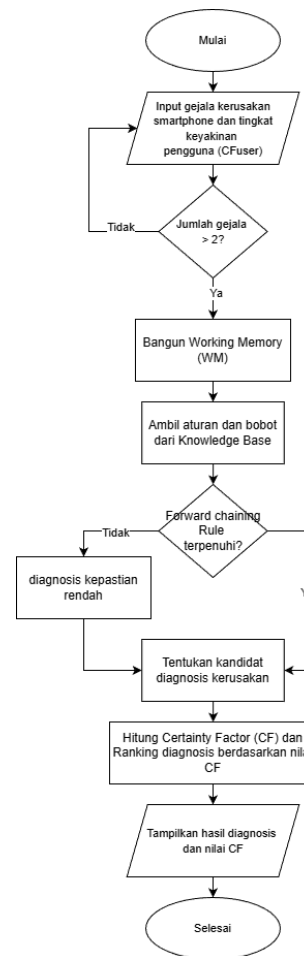
Rancangan alur inferensi sistem pakar digambarkan dalam bentuk *flowchart* pada Gambar 2, yang menjelaskan proses mulai dari input gejala oleh pengguna hingga dihasilkannya diagnosis kerusakan smartphone beserta tingkat kepastiannya.

Proses inferensi dimulai dari input gejala kerusakan beserta frekuensi kemunculannya (CFuser) yang disimpan sebagai fakta awal pada *working memory*. Sistem selanjutnya mengakses *knowledge base* yang berisi aturan (*rule*) dan bobot pakar (CFexpert) untuk melakukan pencocokan menggunakan algoritma *Forward Chaining*. Setiap aturan berbentuk Persamaan (1):

$$IF (G1, G2, ..., Gn) THEN Di \quad (1)$$

dengan G merupakan gejala dan D merupakan jenis kerusakan. Suatu aturan dinyatakan terpenuhi apabila tingkat kecocokan (*match ratio*) mencapai nilai ambang (*threshold*) yang ditetapkan, dihitung dengan Persamaan (2):

$$Match Ratio = \frac{(jumlah\ gejala\ cocok)}{(jumlah\ gejala\ rule)} \quad (2)$$



Gambar 2. Flowchart Inferensi Sistem Pakar Self Assessment Smartphone Repair Service

Apabila $Match Ratio \geq Threshold$, kerusakan terkait ditetapkan sebagai kandidat diagnosis. Apabila tidak terdapat aturan yang memenuhi *threshold* sistem tetap menghasilkan diagnosis dengan mempertimbangkan seluruh jenis kerusakan sebagai kandidat, kemudian melakukan perangkingan berdasarkan nilai *Certainty Factor* tertinggi.

Tingkat kepastian setiap kandidat diagnosis dihitung menggunakan metode *Certainty Factor*. Untuk setiap pasangan gejala G dan kerusakan D dihitung nilai $CF evidence$ pada Persamaan (3):

$$CF evidence = CFuser(G) \times CFexpert(G,D) \quad (3)$$

Apabila terdapat lebih dari satu gejala relevan terhadap satu kerusakan, nilai CF digabungkan secara bertahap menggunakan persamaan kombinasi (4):

$$CF_{combine} = CF_a + CF_b \times (1 - CF_a) \quad (4)$$

Proses kombinasi dilakukan berulang hingga seluruh gejala relevan tergabung, sehingga diperoleh nilai CF akhir yang merepresentasikan tingkat kepastian sistem terhadap suatu diagnosis kerusakan.

4. HASIL DAN PEMBAHASAN

4.1 Analisis Kebutuhan Sistem

Hasil analisis kebutuhan menunjukkan bahwa sistem harus mampu menampilkan daftar gejala kerusakan smartphone yang dapat dipilih pengguna, menerima input frekuensi kemunculan gejala (*CFuser*), membentuk *working memory*, melakukan inferensi *Forward Chaining*, menghitung *Certainty Factor*, serta menampilkan hasil diagnosis berupa jenis kerusakan, nilai kepastian, tingkat keparahan, dan rekomendasi solusi perbaikan disertai penjelasan proses inferensi (*explanation facility*). Selain kebutuhan fungsional tersebut, sistem juga dirancang untuk memenuhi kebutuhan non-fungsional berupa kompatibilitas pada perangkat Android versi 6.0 (*Marshmallow*) API level 23 ke atas dengan RAM minimal 2 GB dan koneksi internet 4G yang stabil.

Berdasarkan hasil akuisisi pengetahuan dari pakar, disusun sepuluh jenis kerusakan sebagai cakupan basis pengetahuan sistem sebagaimana ditunjukkan pada Tabel 1.

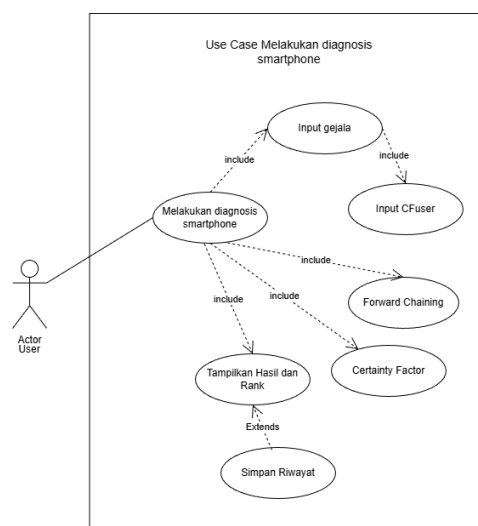
Tabel 1. Jenis Kerusakan pada Basis Pengetahuan

Kode	Jenis Kerusakan
D01	Baterai Drop / Health Menurun
D02	Kerusakan Port Charging / Konektor Charger
D03	Kerusakan IC Power / PMIC
D04	Kerusakan LCD / Backlight
D05	Kerusakan Touchscreen / Digitizer
D06	Gangguan Sistem / OS Crash
D07	Bootloop / Stuck Logo
D08	Kerusakan Modul Sinyal / Antena / Baseband

Kode	Jenis Kerusakan
D09	Kerusakan Audio (Speaker / Mic)
D10	Kerusakan Kamera (Modul / Fleksibel)

4.2 Desain Sistem

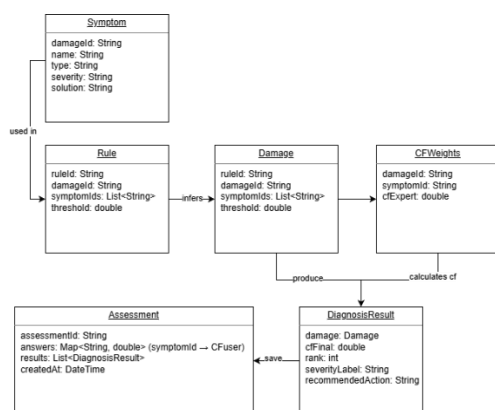
Perancangan sistem dimodelkan melalui *Use Case Diagram*, *Activity Diagram*, dan *Class Diagram* [19]. *Use Case Diagram* pada Gambar 3 menempatkan pengguna (*user*) sebagai satu-satunya aktor, karena aplikasi dikembangkan sebagai aplikasi *self-assessment* satu sisi tanpa modul admin/pakar.



Gambar 3. Use Case Diagram Melakukan Diagnosis Smartphone

Use case utama “Melakukan diagnosis *smartphone*” memiliki dua *use case* yang disertakan (*<<include>>*) karena selalu terjadi dalam proses diagnosis, yaitu input gejala dan input *CFuser*, serta satu *use case* opsional “Simpan riwayat” yang dimodelkan sebagai *<<extend>>* karena hanya dijalankan apabila pengguna memilih untuk menyimpan hasil diagnosis.

Struktur data inti sistem dimodelkan melalui *Class Diagram* pada Gambar 4, yang terdiri atas kelas *Symptom* (gejala), *Rule* (aturan IF-THEN), *Damage* (jenis kerusakan), *CFWeights* (bobot) *CFexpert*, serta *Assessment* yang menyimpan riwayat jawaban dan hasil diagnosis pengguna.



Gambar 4. Class Diagram Domain Entity

Basis pengetahuan diimplementasikan pada *Firestore* dengan pendekatan *NoSQL* berbasis dokumen, terdiri atas koleksi *symptoms*, *damages*, *rules*, dan *cf_weights*, karena pendekatan ini mendukung sinkronisasi data secara *real-time* serta mudah diintegrasikan dengan aplikasi berbasis *Flutter* [14].

4.3 Implementasi Forward Chaining dan Certainty Factor

Sebagai ilustrasi penerapan algoritma pada aplikasi, disimulasikan seorang pengguna memilih empat gejala kerusakan beserta frekuensi kemunculannya, sebagaimana ditunjukkan pada Tabel 2.

Tabel 2. Gejala Terpilih dan Nilai CFuser

Kode	Nama Gejala	Frekuensi	CFuser
G01	Baterai cepat habis	Sering	1,0
G05	Perangkat cepat panas	Kadang	0,5
G06	Performa lambat / lag	Sering	1,0
G03	Perangkat sering mati sendiri	Kadang	0,5

Salah satu aturan yang dievaluasi sistem adalah *Rule R01*: IF G01 AND G05 AND G06 AND G03 THEN D01 (Baterai Drop/Health Menurun). Karena keempat gejala prasyarat *rule* dipilih pengguna, *match ratio* dihitung menggunakan Persamaan (2):

$$\text{Match Ratio} = 4 / 4 = 1,0 \quad (-)$$

Nilai tersebut melebihi *threshold* yang ditetapkan pada aturan, sehingga *rule R01* dinyatakan terpenuhi dan kerusakan D01 ditetapkan sebagai kandidat diagnosis. Selanjutnya, dihitung nilai *CF evidence* untuk setiap gejala menggunakan Persamaan (3) berdasarkan bobot *CFexpert* dari pakar, sebagaimana ditunjukkan pada Tabel 3.

Tabel 3. Perhitungan CF Evidence untuk Kerusakan D01

Gejala	CFuser	CFexpert	CF Evidence
G01	1,0	0,9	0,90
G05	0,5	0,6	0,30
G06	1,0	0,5	0,50
G03	0,5	0,4	0,20

Karena seluruh nilai *CF evidence* bersifat positif, penggabungan dilakukan secara bertahap menggunakan Persamaan (4):

$$CF_{12} = 0,90 + 0,30 \times (1 - 0,90) = 0,930 \quad (-)$$

$$CF_{123} = 0,930 + 0,50 \times (1 - 0,930) = 0,965 \quad (-)$$

$$CF_{1234} = 0,965 + 0,20 \times (1 - 0,965) = 0,972 \quad (-)$$

sehingga diperoleh nilai CF akhir untuk kerusakan D01 sebesar 0,972 atau 97,2%. Sistem juga menghitung nilai *Certainty Factor* untuk kandidat kerusakan lain yang memiliki relasi bobot dengan gejala yang sama, kemudian mengurutkannya berdasarkan nilai CF secara menurun sebagaimana ditunjukkan pada Tabel 4.

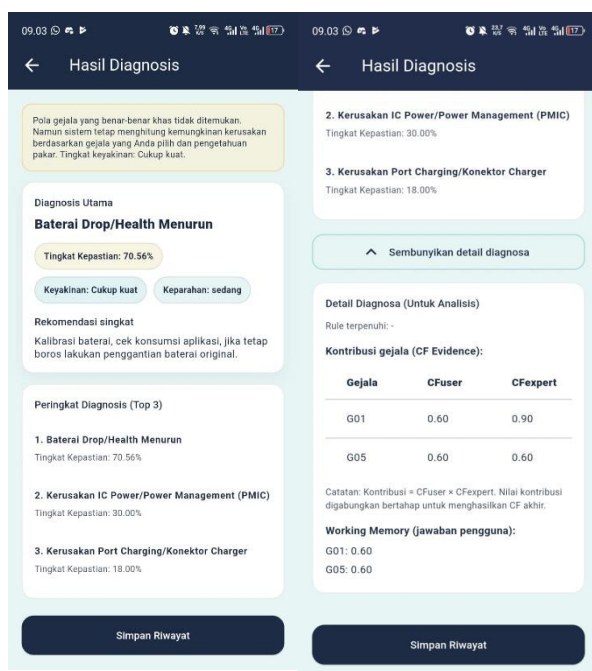
Tabel 4. Peringkat Hasil Diagnosis Berdasarkan Nilai CF

Peringkat	Jenis Kerusakan	CF Final
1	Baterai Drop / Health Menurun	97,2%
2	Gangguan Sistem / OS Crash	61,0%
3	Kerusakan IC Power (PMIC)	48,0%

Berdasarkan hasil tersebut, sistem menampilkan diagnosis utama berupa jenis

kerusakan Baterai Drop/*Health* Menurun dengan tingkat kepastian 97,2% disertai rekomendasi solusi berupa kalibrasi baterai, pemeriksaan konsumsi aplikasi, dan penggantian baterai apabila diperlukan. Nilai $CF \geq 50\%$ pada hasil ini menunjukkan bahwa diagnosis memiliki tingkat keyakinan yang cukup kuat, karena didukung kombinasi gejala relevan dan bobot pakar yang memadai, sedangkan kandidat kerusakan pada peringkat kedua dan ketiga ditampilkan sebagai kemungkinan tambahan untuk mengakomodasi ketidakpastian gejala yang tumpang tindih antar-kerusakan.

Gambar 5 menunjukkan tampilan antarmuka hasil diagnosis dan *explanation facility* yang menjelaskan proses inferensi kepada pengguna, sehingga pengguna awam dapat memahami dasar dari setiap diagnosis yang dihasilkan sistem.



Gambar 5. Tampilan Detail Hasil Diagnosis dan Explanation Facility

4.4 Pengujian Sistem

4.4.1 Black-Box Testing

Pengujian fungsional dilakukan menggunakan metode *Black-Box Testing* terhadap delapan fitur utama aplikasi, meliputi registrasi, *login* (skenario valid dan tidak valid),

proses *self-assessment*, inferensi *Forward Chaining*, perhitungan *Certainty Factor*, tampilan hasil diagnosis, dan penyimpanan riwayat diagnosis [20]. Rekapitulasi hasil pengujian ditunjukkan pada Tabel 5.

Tabel 5. Rekapitulasi Pengujian Black-Box Testing

No	Fitur	Skenario Pengujian	Hasil
1	Register	User register email & password	Sesuai / Valid
2	Login	Login dengan data valid	Sesuai / Valid
3	Login	Login dengan data tidak valid	Sesuai / Valid
4	Self-Assessment	User memilih minimal 2 gejala	Sesuai / Valid
5	Forward Chaining	Rule memenuhi nilai threshold	Sesuai / Valid
6	Certainty Factor	Gejala memiliki bobot pakar	Sesuai / Valid
7	Hasil Diagnosis	Sistem menampilkan hasil diagnosis	Sesuai / Valid
8	Riwayat Diagnosis	User menyimpan hasil diagnosis	Sesuai / Valid

Seluruh skenario pengujian pada Tabel 5 menunjukkan hasil “Sesuai” dengan hasil yang diharapkan, sehingga seluruh fitur utama sistem, termasuk mekanisme inferensi *Forward Chaining* dan perhitungan *Certainty Factor*, berjalan sesuai dengan kebutuhan fungsional yang telah ditetapkan pada tahap analisis kebutuhan.

4.4.2 User Acceptance Testing (UAT)

Pengujian penerimaan pengguna dilakukan dengan menyebarkan kuesioner berbasis *Google Form* kepada 15 responden yang mencoba seluruh fitur utama aplikasi secara langsung, menggunakan skala *Likert* 1–5 (Sangat Tidak Setuju hingga Sangat Setuju). Rekapitulasi hasil kuesioner UAT ditunjukkan pada Tabel 6.

Tabel 6. Rekapitulasi Hasil Kuesioner UAT

No	Pernyataan	Rata-rata Skor
1	Aplikasi dapat dijalankan dengan baik di perangkat Android	4,6
2	Proses login dan registrasi berjalan dengan lancar	4,5
3	Daftar gejala mudah dipahami oleh pengguna	4,4
4	Pengisian frekuensi gejala mudah dilakukan	4,5
5	Proses diagnosis berjalan tanpa kendala	4,6
6	Hasil diagnosis ditampilkan dengan jelas	4,5
7	Tingkat kepastian diagnosis mudah dipahami	4,4
8	Penjelasan hasil diagnosis membantu pemahaman pengguna	4,3

Rata-rata keseluruhan skor UAT adalah 4,49 dari skala 5, yang berada pada kategori Setuju hingga Sangat Setuju. Skor tertinggi diperoleh pada pernyataan mengenai kelancaran proses diagnosis dan kompatibilitas aplikasi pada perangkat Android (4,6), sedangkan skor terendah namun tetap positif diperoleh pada pernyataan mengenai kemudahan pemahaman penjelasan hasil diagnosis (4,3). Temuan ini menunjukkan bahwa pengguna awam dapat memahami proses pengisian gejala, pembacaan tingkat kepastian diagnosis, serta penjelasan hasil diagnosis dengan baik.

4.5 Pembahasan

Hasil pengujian *Black-Box* dan UAT secara konsisten menunjukkan bahwa kombinasi *Forward Chaining* dan *Certainty Factor* dapat diterapkan pada aplikasi *mobile* Android tanpa mengurangi transparansi proses inferensi, sejalan dengan temuan penelitian terdahulu yang menerapkan kombinasi kedua metode ini untuk diagnosis kerusakan smartphone [6], [7]. Perbedaan utama penelitian ini terletak pada penyediaan *explanation facility* yang menampilkan rincian perhitungan CF *evidence* dan proses kombinasinya, sehingga pengguna awam tidak hanya menerima label kerusakan, tetapi juga memahami dasar keyakinan sistem terhadap diagnosis tersebut.

Dibandingkan dengan penelitian pengembangan layanan jasa berbasis Android atau web yang berfokus pada aspek transaksional seperti pemesanan atau pemantauan status perbaikan [21], [22], aplikasi ini berfokus pada tahap pra-servis, yaitu membantu pengguna mengenali indikasi awal kerusakan sebelum memutuskan untuk melakukan perbaikan. Pendekatan ini relevan mengingat kepuasan pelanggan pada layanan reparasi juga banyak dipengaruhi oleh kejelasan informasi dan komunikasi sejak tahap awal layanan [23], sehingga diagnosis awal yang informatif berpotensi mengurangi kesalahpahaman antara pengguna dan teknisi pada saat konsultasi lanjutan.

Meskipun hasil pengujian menunjukkan penerimaan yang baik, penelitian ini masih memiliki keterbatasan. Basis pengetahuan yang digunakan bersumber dari satu orang pakar, sehingga cakupan gejala dan bobot *Certainty Factor* masih terbatas pada pengalaman pakar tersebut. Selain itu, evaluasi UAT hanya melibatkan 15 responden sehingga generalisasi hasil penerimaan pengguna perlu diperkuat dengan jumlah responden yang lebih besar pada penelitian selanjutnya. Basis pengetahuan yang lebih luas dari beberapa pakar dan pengujian akurasi diagnosis dengan jumlah kasus uji yang lebih besar akan menjadi arah penting bagi pengembangan lanjutan aplikasi ini.

5. KESIMPULAN

- Aplikasi *self assessment smartphone repair service* berhasil dirancang dan dibangun menggunakan *framework Flutter* dan *Firestore* sebagai *backend*, sehingga dapat dijalankan pada perangkat Android dan digunakan pengguna untuk melakukan diagnosis awal kerusakan smartphone secara mandiri.
- Algoritma *Forward Chaining* berhasil diterapkan sebagai mesin inferensi berbasis aturan yang menelusuri gejala menuju kandidat diagnosis berdasarkan nilai ambang (*threshold*) kecocokan *rule*, sedangkan metode *Certainty*

- Factor* berhasil menghitung tingkat kepastian diagnosis dengan mengombinasikan nilai keyakinan pengguna (CFuser) dan bobot keyakinan pakar (CFexpert).
- c. Metode *Software Development Life Cycle* (SDLC) model *Waterfall* berhasil diterapkan secara sistematis mulai dari analisis kebutuhan hingga pemeliharaan, sehingga proses pengembangan aplikasi berjalan terstruktur dan terdokumentasi dengan baik.
- d. Hasil pengujian *Black-Box* menunjukkan seluruh delapan fitur utama aplikasi berjalan sesuai kebutuhan fungsional, sedangkan hasil *User Acceptance Testing* terhadap 15 responden memperoleh skor rata-rata 4,49 dari skala 5 yang berada pada kategori setuju hingga sangat setuju, menunjukkan aplikasi diterima baik sebagai alat bantu diagnosis awal yang informatif dan mudah digunakan oleh pengguna awam.
- e. Penelitian ini masih memiliki keterbatasan pada cakupan basis pengetahuan yang bersumber dari satu pakar dan jumlah responden pengujian yang terbatas, sehingga pengembangan lanjutan disarankan mencakup penambahan basis pengetahuan dari beberapa pakar, evaluasi akurasi dengan kasus uji yang lebih besar, serta pengembangan fitur pendukung seperti integrasi lokasi layanan servis terdekat.

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Bapak Setya Rinaldi, teknisi ahli servis di Toko Pertemuan Cell, Karawang, atas kesediaannya menjadi narasumber akuisisi pengetahuan dalam penelitian ini, serta kepada Program Studi Informatika, Fakultas Ilmu Komputer, Universitas Singaperbangsa Karawang atas dukungan yang diberikan selama pelaksanaan penelitian.

DAFTAR PUSTAKA

- [1] G. Data, "209,3 juta orang di Indonesia menggunakan smartphone pada tahun 2023," GoodStats Data, Jun. 12, 2024. [Online]. Available: <https://data.goodstats.id/statistic/2093-juta-orang-di-indonesia-menggunakan-smartphone-pada-tahun-2023-cbha0>
- [2] S. Kemp, "Digital 2023: Indonesia," DataReportal – Global Digital Insights, Feb. 8, 2023. [Online]. Available: <https://datareportal.com/reports/digital-2023-indonesia>
- [3] A. Fathah, "Evaluasi fitur tracking pada pengalaman pelanggan di bisnis reparasi smartphone," JKBM (Jurnal Konsep Bisnis dan Manajemen), vol. 9, no. 1, pp. 120–134, 2022, doi: 10.31289/jkbm.v9i1.8432.
- [4] W. G. Aedi, "Rancang bangun sistem informasi perbaikan smartphone berbasis web (Studi kasus pada iKids Phone Service)," BULLET: Jurnal Multidisiplin Ilmu, vol. 1, no. 3, pp. 342–351, 2022.
- [5] Z. S. Al Ayubi, S. A. Syuhada, D. Prasetyo, and R. Djutalov, "Implementasi algoritma rule-based dalam aplikasi penjadwalan dan pelaporan menggunakan metode Agile (Studi kasus PT. Sukses Mitra Pest)," Jurnal Ilmiah Pengabdian pada Masyarakat, vol. 2, no. 2, pp. 555–565, 2024. Retrieved from <https://jurnal.ittc.web.id/index.php/jipm/article/view/1985>
- [6] M. Fitriyani, H. Rohayani, and H. Santoso, "Sistem pakar mendiagnosa kerusakan smartphone Android menggunakan metode certainty factor," Jurnal Rekayasa Sistem Informasi dan Teknologi, vol. 1, no. 3, pp. 137–149, 2024, doi: 10.59407/jrsit.v1i3.484.
- [7] Y. G. Nengsih and N. Putra, "Sistem pakar menggunakan forward chaining dan certainty factor untuk diagnosa kerusakan smartphone," JURSIMA, vol. 8, no. 2, pp. 61–69, 2020, doi:10.47024/js.v8i2.205.
- [8] A. Maulana, I. Purnamasari, and I. Maulana, "Rancang bangun website layanan jasa reparasi alat elektronik rumah tangga menggunakan framework Laravel (Studi kasus: CV. XYZ)," Jurnal Informatika dan Teknik Elektro Terapan, vol. 12, no. 3, 2024, doi:10.23960/jitet.v12i3.4859.
- [9] N. A. Hasanah, R. Rodianto, and Y. Yuliadi, "Rancang bangun aplikasi sistem pakar menggunakan metode forward chaining untuk mendiagnosa penyakit pada ayam," JATISI (Jurnal Teknik Informatika dan Sistem Informasi), vol. 12, no. 2, 2025, doi: 10.35957/jatisi.v12i2.10890.

- [10] N. Permata, S. Safaruddin, and D. Pujiyanto, "Aplikasi pengepul getah karet di Kabupaten Ogan Komering Ulu berbasis Android," *JSIM: Jurnal Sistem Informasi Mahakarya*, vol. 6, no. 1, pp. 82–90, 2022.
- [11] U. Kalsum and B. Rakhim, "Implementasi framework Flutter untuk chatbot informasi sistem berbasis aplikasi mobile," *Innovative: Journal of Social Science Research*, vol. 4, no. 2, pp. 6272–6286, 2024.
- [12] N. Sofi and R. Dharmawan, "Perancangan aplikasi bengkel CSM berbasis Android menggunakan framework Flutter (bahasa Dart)," *Jurnal Teknik dan Science*, vol. 1, no. 2, pp. 53–64, 2022, doi: 10.56127/jts.v1i2.125.
- [13] N. Rachma and F. Ramdhani, "Sistem Informasi Tracer Study Berbasis Mobile Dengan Metode Fountain Menggunakan Bahasa Pemrograman Dart Dan Framework Flutter," *Journal of Comprehensive Science (JCS)*, vol. 3, no. 3, pp. 702–711, Mar. 2024, doi: 10.59188/jcs.v3i3.654.
- [14] N. Noviyana and M. I. P. Nasution, "Implementasi database dalam meningkatkan efektivitas pengelolaan data mahasiswa," *Kohesi: Jurnal Sains dan Teknologi*, vol. 3, no. 11, pp. 51–60, 2024, doi: 10.3785/kohesi.v3i11.4263.
- [15] B. S. Nagara, D. Oetari, Z. Apriliani, and T. Sutabri, "Penerapan metode SDLC (System Development Life Cycle) Waterfall pada perancangan aplikasi belanja online berbasis Android pada CV Widi Agro," *Intecom: Journal of Information Technology and Computer Science*, vol. 6, no. 2, pp. 1202–1210, 2023, doi: 10.31539/intecom.v6i2.8244.
- [16] Y. Nuryamin and F. Risyda, "Perancangan aplikasi kasir pada kedai kopi berbasis web menggunakan model Waterfall," *JSI (Jurnal Sistem Informasi) Universitas Suryadarma*, vol. 11, no. 1, pp. 191–198, 2024, doi: 10.35968/jsi.v11i1.1140.
- [17] N. A. Prasetyo, "Rancang bangun aplikasi layanan jasa pada Mitra Obre-Jek berbasis Android menggunakan metode Scrum," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 12, no. 3, 2024, doi: 10.23960/jitet.v12i3.4505.
- [18] H. Dafitri, E. Panggabean, N. Wulan, A. J. Lubis, S. Khairani, and A. P. Humaira, "Pelatihan pembuatan desain UI/UX website UMKM Profile Labscarpe dengan aplikasi Figma," *Jurnal Pengabdian kepada Masyarakat Nusantara*, vol. 3, no. 2.2, pp. 1972–1980, 2023, doi: 10.47065/josh.v6i1.5987.
- [19] R. S. D. Andiko and M. R. A. Cahyono, "Rancang bangun aplikasi transaksi barcode berbasis Java melalui metode Unified Modeling Language (UML)," *Jurnal Instrumentasi dan Teknologi Informasi (JITI)*, vol. 3, no. 2, pp. 160–166, 2022. Retrieved from <https://jurnal.poltek-gt.ac.id/jiti/id/article/view/31/38>
- [20] A. C. Praniffa, A. Syahri, F. Sandes, U. Fariha, Q. A. Giansyah, and M. L. Hamzah, "Pengujian black box dan white box sistem informasi parkir berbasis web," *Jurnal Test dan Implementasi Sistem Informasi*, vol. 1, no. 1, pp. 1–16, 2023, doi: 10.55583/jtisi.v1i1.321
- [21] A. Darmuli, F. Fauziyah, and R. G. Whendasmoro, "Rancang bangun aplikasi advisor and monitoring service pada Auto2000 Permata Hijau berbasis Android," *Jurnal Tekinkom (Teknik Informasi dan Komputer)*, vol. 6, no. 2, pp. 660–667, 2023, doi: 10.37600/tekinkom.v6i2.749.
- [23] Y. Y. Anis and A. S. Rifa, "Perancangan sistem informasi e-booking jasa steam mobil dan motor berbasis web (Studi kasus Cheers Autocare Solo)," *Bulletin of Information Technology (BIT)*, vol. 4, no. 1, pp. 99–104, 2023, doi: 10.47065/bit.v4i1.516.
- [24] S. Helvina, "Pengaruh komunikasi pelayanan terhadap kepuasan pelanggan CV Apple Team Care Cabang Banjarbaru," *Skripsi, Universitas Islam Kalimantan MAB*, 2024.
- [25] Disantis & A. S. Purnomo, "Sistem Pakar untuk Mendiagnosis Tingkat Keterikatan Remaja terhadap Game Online di Kabupaten Malaka Menggunakan Metode Certainty Factor," *Jurnal Informatika dan Teknik Elektro Terapan*, vol. 13, no. 3, pp. 929–938, Jul. 2025, doi: 10.23960/jitet.v13i3.7091.