IMPLEMENTASI CERTAINTY FACTOR PADA SISTEM PAKAR DETEKSI PREEKLAMSIA BERBASIS FRAMEWORK FLASK

Elsa Vania[1*], Agung Brastama Putra[2], Tri Luhur Indayanti Sugata[3]

[1,2,3] Program Studi Sistem Informasi, Fakultas Ilmu Komputer, Universitas Pembangunan Nasional "Veteran" Jawa Timur
Surabaya, Indonesia

Email : [1] velsas.res@gmail.com, [2] agungbp.si@upnjatim.ac.id,
[3]tri.luhur.fasilkom@upnjatim.ac.id

ABSTRAK

Sistem pakar dapat digunakan untuk membantu proses penalaran pada kasus yang membutuhkan penggabungan beberapa gejala dan parameter pemeriksaan. Pada deteksi preeklamsia, data yang digunakan tidak hanya berupa keluhan klinis, tetapi juga parameter laboratorium seperti proteinuria, trombosit, SGOT, dan SGPT. Penelitian ini bertujuan mengimplementasikan sistem pakar deteksi Preeklamsia dan Preeklamsia Berat menggunakan metode *Certainty Factor* berbasis *framework* Flask. *Certainty Factor* digunakan untuk merepresentasikan tingkat keyakinan pakar terhadap bukti yang aktif, sedangkan Flask digunakan untuk membangun aplikasi *website* dengan pemisahan route, service, model, template, dan basis data. Sistem dikembangkan melalui tahapan analisis kebutuhan, akuisisi pengetahuan, desain sistem, implementasi Flask, dan pengujian *black box*. Hasil pengujian menunjukkan bahwa fungsi utama, meliputi *log in*, pengelolaan data pasien, input pemeriksaan, proses diagnosis, detail hasil, riwayat, cetak PDF, basis pengetahuan, serta manajemen user dan *log out* berjalan sesuai skenario. Hasil tersebut menunjukkan bahwa implementasi Flask dapat mendukung penerapan sistem pakar berbasis *Certainty Factor* secara terstruktur.

Kata Kunci : Sistem Pakar, Preeklamsia, *Certainty Factor,* Flask, *Website*

ABSTRACT

Expert systems can be used to assist in the reasoning process for cases that require the integration of multiple symptoms and examination parameters. In the detection of preeclampsia, the data used includes not only clinical symptoms but also laboratory parameters such as proteinuria, platelet count, SGOT, and SGPT. This study aims to implement an expert system for the detection of preeclampsia and severe preeclampsia using the Certainty Factor method based on the Flask framework. The Certainty Factor is used to represent the level of an expert's confidence in active evidence, while Flask is used to build a

web application with separation of routes, services, models, templates, and databases. The system was developed through the stages of requirements analysis, knowledge acquisition, system design, Flask implementation, and black box testing. Test results show that the main functions including login, patient data management, examination input, the diagnostic process, result details, history, PDF printing, the knowledge base, as well as user management and logout operate according to the scenario. These results indicate that the Flask implementation can support the structured application of a Certainty Factor-based expert system.

***Key words**: Expert System, Preeclampsia, Certainty Factor, Flask, Website*

Pendahuluan

Angka Kematian Ibu (AKI) masih menjadi indikator penting dalam menilai kualitas pelayanan kesehatan maternal. Target *Sustainable Development Goals* (SDG) 3.1 menekankan penurunan rasio kematian ibu menjadi kurang dari 70 kematian per 100.000 kelahiran hidup pada tahun 2030 (World Health Organization, 2025). Di Indonesia, hasil *Long Form* Sensus Penduduk 2020 menunjukkan AKI sebesar 189 kematian per 100.000 kelahiran hidup, sehingga capaian tersebut masih berada di atas target SDG (Badan Pusat Statistik, 2023). Selain itu, Profil Kesehatan Indonesia 2024 mencatat jumlah kematian ibu sebanyak 4.150 kasus pada tahun 2024, menurun dari 4.482 kasus pada tahun 2023, tetapi tetap menunjukkan perlunya penguatan deteksi dini komplikasi kehamilan (Kementerian Kesehatan Republik Indonesia, 2024).

Salah satu komplikasi kehamilan yang perlu dikenali sejak awal adalah preeklamsia. Kementerian Kesehatan menjelaskan bahwa preeklamsia termasuk gangguan hipertensi dalam kehamilan yang muncul setelah usia kehamilan lebih dari 20 minggu dan dapat disertai proteinuria atau tanda perburukan lain (Azizah M. Ranggis, 2024). Pedoman Nasional Pelayanan Kedokteran Preeklamsia juga menjelaskan bahwa penilaian preeklamsia melibatkan tekanan darah, proteinuria, trombositopenia, gangguan fungsi hati, gejala neurologis, dan nyeri epigastrium (Perkumpulan Obstetri dan Ginekologi Indonesia, Himpunan Kedokteran Feto Maternal, 2016).

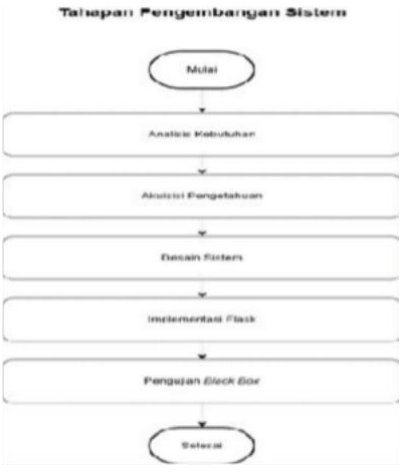
Penelitian terdahulu menunjukkan bahwa *Certainty Factor* telah digunakan dalam sistem pakar preeklamsia pada beberapa *platform*. Pada penelitian yang menerapkan Certainty Factor berbasis Python untuk deteksi penyakit Preeklamsia menggunakan tujuh gejala memperoleh hasil perhitungan sebesar 96,35% pada kasus uji yang digunakan (Hanif et al., 2022). Tak hanya itu , penelitian sebelumnya juga membangun sistem pakar deteksi dini preeklamsia berbasis web (Prasetya et al., 2022), sedangkan penelitian mengembangkan

sistem peringatan awal risiko preeklamsia berbasis Android (Fitrilina et al., 2021). Penelitian-penelitian tersebut menunjukkan bahwa *Certainty Factor* dapat merepresentasikan tingkat keyakinan pakar, tetapi pembahasan mengenai integrasi parameter laboratorium seperti trombosit, SGOT, dan SGPT serta struktur implementasi aplikasi web berbasis Flask belum menjadi fokus utama.

Berdasarkan kondisi tersebut, penelitian ini berfokus pada implementasi sistem pakar deteksi Preeklamsia dan Preeklamsia Berat berbasis framework Flask. Flask dipilih karena bersifat ringan dan fleksibel untuk membangun aplikasi web, serta memungkinkan pemisahan alur program menjadi *route*, *service*, *model*, *template*, dan *database*. Pemisahan tersebut dibutuhkan agar logika diagnosis *Certainty Factor* tidak bercampur langsung dengan kode tampilan. Dengan demikian, kontribusi artikel ini terletak pada penerapan Flask untuk menghubungkan input pemeriksaan, basis pengetahuan, proses diagnosis, penyimpanan hasil, riwayat diagnosis, dan laporan PDF pada sistem pakar preeklamsia.

Materi dan Metode

Penelitian dilakukan melalui tahapan pengembangan sistem yang disesuaikan dengan kebutuhan implementasi sistem pakar. Tahap pertama adalah analisis kebutuhan untuk menentukan aktor, fitur utama, dan data yang harus dikelola sistem. Tahap kedua adalah akuisisi pengetahuan, yaitu penyusunan parameter klinis dan laboratorium, relasi penyakit-gejala, nilai *Measure of Belief* (MB), nilai *Measure of Disbelief* (MD), serta aturan klasifikasi diagnosis. Tahap ketiga adalah desain sistem yang meliputi *use case*, struktur basis data, dan arsitektur implementasi Flask. Tahap keempat adalah implementasi sistem menggunakan Python, Flask, MySQL, Flask-Login, Flask-SQLAlchemy, dan Jinja2. Tahap terakhir adalah pengujian *black box* untuk memeriksa kesesuaian masukan, proses, dan keluaran pada fungsi utama sistem.



Gambar 1. Tahapan Pengembangan Sistem

Perangkat yang digunakan meliputi laptop dengan sistem operasi Windows, Visual Studio Code sebagai editor, Python sebagai bahasa pemrograman, Flask sebagai framework web, MySQL sebagai basis data, serta pustaka pendukung seperti Flask-Login, Flask-SQLAlchemy, dan Jinja2. Data penelitian berupa basis pengetahuan preeklamsia yang terdiri dari parameter klinis, parameter laboratorium, nilai *Measure of Belief* (MB), nilai *Measure of Disbelief* (MD), dan aturan klasifikasi diagnosis.

Sistem pakar merupakan sistem yang menyimpan pengetahuan pakar dalam bentuk fakta dan aturan, kemudian menggunakan mesin inferensi untuk menghasilkan kesimpulan pada domain tertentu (Joseph C. Giarratano and Gary D Riley, 2005). Pada sistem ini, basis pengetahuan disimpan pada tabel basis_pengetahuan yang memuat relasi penyakit, gejala, nilai MB, dan nilai MD. Mesin inferensi ditempatkan pada *service* Diagnosis *Engine*, sedangkan antarmuka pengguna ditampilkan melalui template HTML/Jinja2 pada Flask.

Certainty Factor digunakan untuk menghitung tingkat keyakinan sistem terhadap diagnosis berdasarkan gejala aktif. Nilai CF gejala dihitung dari *measure of belief* (MB) dan *measure of disbelief* (MD) (Shortliffe and Buchanan, 1975). Jika terdapat lebih dari satu bukti yang mendukung diagnosis yang sama, nilai CF dikombinasikan secara bertahap menggunakan rumus kombinasi CF (Heckerman, 1992).

Rumus Certainty Factor:

$$CF = MB(E,H) - MD(E,H) \quad \text{.....(1)}$$

Di mana:

E adalah bukti atau gejala yang teramati.

H adalah hipotesis atau diagnosa.

MB(E,H) adalah tingkat keyakinan terhadap hipotesis H berdasarkan bukti E.

MD(E,H) adalah tingkat ketidakpercayaan terhadap hipotesis H berdasarkan bukti E.

Jika terdapat lebih dari satu bukti yang mendukung hipotesis, CF dari setiap bukti akan dikombinasikan dengan rumus sebagai berikut:

$$CF\ Combine = CF1 + CF2 * (1 - CF1) \quad \text{.....(2)}$$

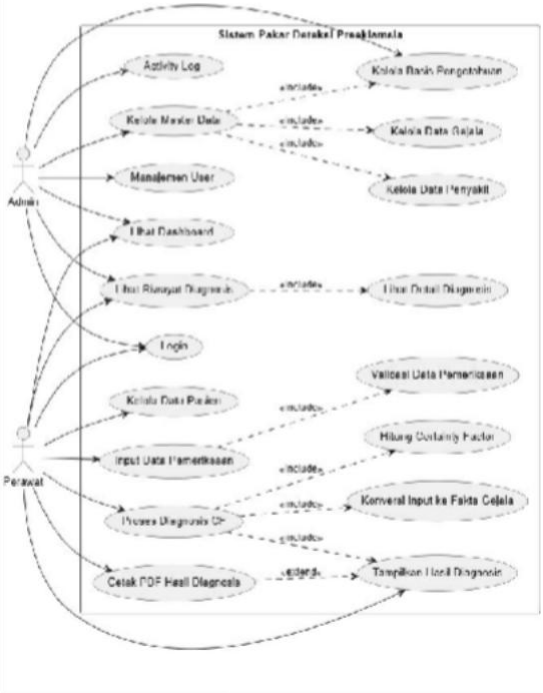
Di mana CF1 dan CF2 adalah nilai Certainty Factor dari masing-masing bukti.

Rincian parameter yang digunakan pada sistem disajikan pada Tabel 1 di bawah ini. Parameter laboratorium tetap ditampilkan dalam tabel tersebut karena menjadi pembeda utama dari sistem yang hanya menggunakan gejala klinis dasar.

Tabel 1 Parameter Diagnosis yang Digunakan Pada Sistem

Kode	Parameter	Jenis Data
G001 – G002	Tekanan darah sistolik dan diastolik >= 140/90 mmHg	Klinis
G003	Proteinuria positif	Laboratorium
G004 – G005	Tekanan darah sistolik/diastolik >= 160/110 mmHg	Klinis
G006	Trombosit < 100.000	Laboratorium
G007 – G008	SGOT dan SGPT meningkat	Laboratorium
G009 – G011	Nyeri kepala berat, pandangan kabur, nyeri ulu hati	Klinis

Use case diagram merupakan bagian dari *Unified Modeling Language* (UML) yang digunakan untuk menggambarkan interaksi antara aktor dan fungsi di dalam sistem. Pada artikel ini, *use case sistem* disederhanakan agar lebih fokus pada proses diagnosis. Sistem ini melibatkan dua aktor utama, yaitu Admin dan Perawat. Admin memiliki hak akses untuk mengelola data master penyakit, gejala, basis pengetahuan, manajemen pengguna, serta memantau *log* aktivitas pengguna. Sementara itu, Perawat berwenang mengelola data pasien, menginput hasil pemeriksaan klinis, menjalankan proses diagnosis, serta melihat dan mencetak laporan hasil diagnosis ke format PDF. Struktur interaksi fungsional kedua aktor tersebut dapat dilihat pada Gambar 2 berikut ini.

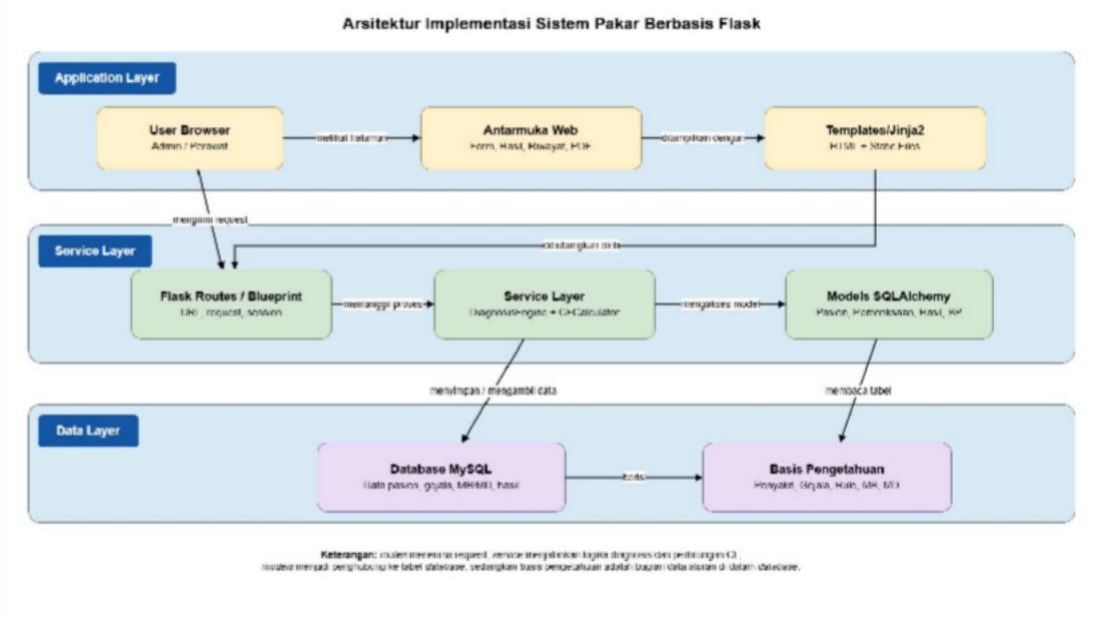


Gambar 2 Use Case Sistem Pakar Deteksi Preeklamsia

Flask merupakan *framework* WSGI (*Web Server Gateway Interface*) berbasis lightweight yang dirancang untuk mempercepat pengembangan awal sekaligus mempertahankan fleksibilitas saat dikembangkan menjadi sistem yang kompleks (Pallets

Projects, 2010). Dalam implementasinya, Flask menyediakan mekanisme rute (routing) untuk menghubungkan alamat URL dengan fungsi Python yang dieksekusi saat halaman diakses (Pallets Projects, 2010). Pada sistem pakar ini, konsep tersebut diterapkan melalui pemisahan struktur *routes*, *service*, *models*, *templates*, dan *database*. Pemisahan komponen ini bertujuan agar alur input pemeriksaan, kalkulasi diagnosis, penyimpanan data hasil, dan visualisasi antarmuka pengguna dapat dikelola secara modular serta terstruktur.

Arsitektur sistem ini dibagi menjadi tiga bagian utama. Bagian pertama adalah *Application Layer* yang berisi antarmuka *website* yang diakses pengguna dan template HTML/Jinja2 untuk menampilkan halaman. Bagian kedua adalah *Service Layer* yang berisi Flask *routes*, servis diagnosis, dan model SQLAlchemy untuk menghubungkan kode Python dengan tabel database. Bagian ketiga adalah *Data Layer* yang berisi *database* MySQL dan basis pengetahuan. Diagram ini bukan arsitektur resmi bawaan Flask, melainkan rancangan yang disusun sendiri berdasarkan struktur kode aplikasi.



Gambar 3 Arsitektur Implementasi Sistem Pakar Berbasis Flask

Hasil dan Pembahasan

Hasil implementasi menunjukkan bahwa Flask dapat digunakan untuk membangun sistem pakar berbasis web dengan struktur terpisah antara *route*, *service*, *model*, *template*, dan konfigurasi. *Route* menerima permintaan pengguna dari *browser*, *service* menjalankan logika diagnosis dan perhitungan CF, *model* merepresentasikan tabel database, sedangkan *template* menampilkan *form* dan hasil diagnosis kepada pengguna.

```python
@diagnosis_bp.route("/form/<id_pasien>", methods=["GET", "POST"])
@login_required
@role_required("perawat")
def form_pemeriksaan_preeklamsia(id_pasien):
    pasien = PemeriksaanService.get_pasien_or_404(id_pasien)
    if request.method == "POST":
        try:
            form_data = request.form.copy()
            form_data["id_pasien"] = str(id_pasien)
            pemeriksaan, warnings = PemeriksaanService.simpan_pemeriksaan(form_data, current_user)
            for warning in warnings:
                flash(warning, "warning")
            hasil, _ = DiagnosisEngine.diagnosis_dari_pemeriksaan(pemeriksaan.id_pemeriksaan, current_user)
            flash("Diagnosis berhasil dijalankan.", "success")
            return redirect(url_for("diagnosis.hasil_diagnosis", id_hasil=hasil.id_hasil))
        except Exception as error:
            db.session.rollback()
            flash(f"Gagal memproses pemeriksaan: {error}", "danger")
    return render_template("perawat/pemeriksaan_preeklamsia/form.html", pasien=pasien)
```

Gambar 4. Potongan Kode Route Input Pemeriksaan Dan Diagnosis

Potongan kode pada Gambar 4 menunjukkan bahwa route tidak menghitung nilai CF secara langsung. Route bertugas menerima input dan memanggil service, sehingga logika diagnosis tidak bercampur dengan kode tampilan. Pola ini sejalan dengan penggunaan route pada Flask yang memetakan URL ke fungsi Python (Pallets Projects, 2010).

```python
@staticmethod
def convert_input_to_facts(pemeriksaan):
    sistolik = DiagnosisEngine._to_int(getattr(pemeriksaan, "sistolik", 0))
    diastolik = DiagnosisEngine._to_int(getattr(pemeriksaan, "diastolik", 0))
    usia_kehamilan = DiagnosisEngine._to_int(getattr(pemeriksaan, "usia_kehamilan_minggu", 0))
    trombosit = DiagnosisEngine._to_int(getattr(pemeriksaan, "trombosit", 0))
    sgot = DiagnosisEngine._to_int(getattr(pemeriksaan, "sgot", 0))
    sgpt = DiagnosisEngine._to_int(getattr(pemeriksaan, "sgpt", 0))

    return {
        "G001": sistolik >= 140,
        "G002": diastolik >= 90,
        "G003": DiagnosisEngine._proteinuria_positif(getattr(pemeriksaan, "proteinuria", None)),
        "G004": sistolik >= 160,
        "G005": diastolik >= 110,
        "G006": trombosit > 0 and trombosit < 100000,
        "G007": sgot > 70,
        "G008": sgpt > 82,
        "G009": DiagnosisEngine._to_bool(getattr(pemeriksaan, "nyeri_kepala_berat", False)),
        "G010": DiagnosisEngine._to_bool(getattr(pemeriksaan, "pandangan_kabur", False)),
        "G011": DiagnosisEngine._to_bool(getattr(pemeriksaan, "nyeri_ulu_hati", False)),
        "USIA_GT_20": usia_kehamilan > 20,
    }
```

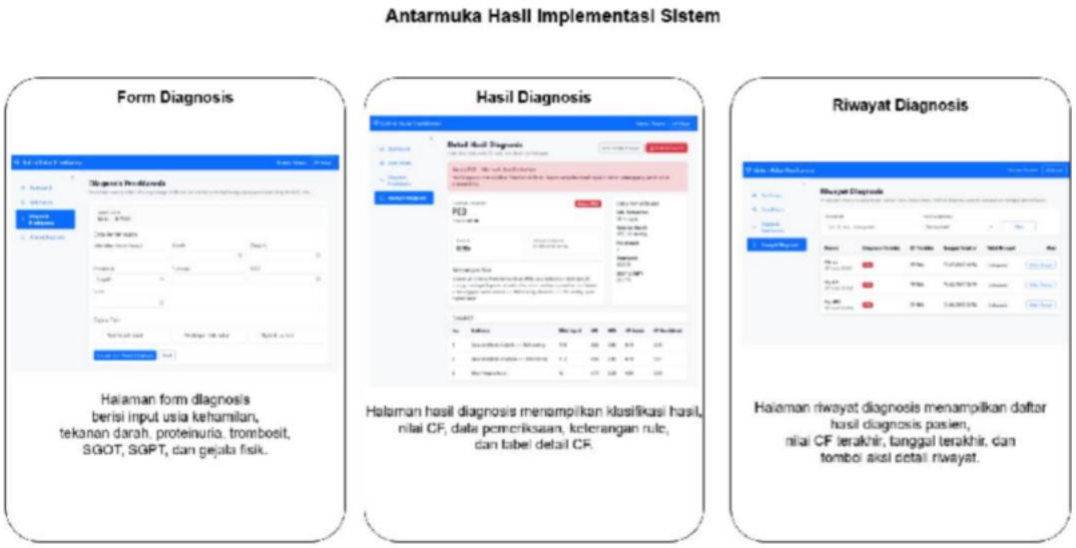
Gambar 5. Potongan Kode Konversi Input Pemeriksaan Menjadi Fakta Gejala Gambar 5 menunjukkan proses konversi data pemeriksaan menjadi fakta gejala. Setiap parameter klinis dan laboratorium dievaluasi berdasarkan ambang batas yang ditentukan, kemudian menghasilkan nilai boolean untuk menandai aktif atau tidaknya suatu gejala. Fakta tersebut menjadi masukan bagi aturan diagnosis sebelum proses perhitungan Certainty Factor.

E-ISSN : 2809-7165

```python
@staticmethod
def predict_from_values(data):
    facts = DiagnosisEngine._facts_from_values(data)
    label = DiagnosisEngine._determine_label(facts)
    penyakit = DiagnosisEngine._get_penyakit(label)
    active_rules = DiagnosisEngine._active_rules_for_penyakit(penyakit, facts)
    cf_list = [CFCalculator.calculate_single_cf(rule.mb, rule.md) for rule in active_rules]
    nilai_cf, _ = CFCalculator.combine_cf(cf_list)
    return {
        "facts": facts,
        "klasifikasi diagnosis": label,
        "nilai_cf": nilai_cf,
        "kasus_peb": label == "PtB",
        "penjelasan_rule": DiagnosisEngine._penjelasan_rule(label, facts),
        "gejala_terdeteksi": [rule.gejala.nama_gejala for rule in active_rules if rule.gejala],
    }
```

Gambar 6. Potongan Kode Proses Inferensi Diagnosis dan Perhitungan Certainty Factor

Gambar 6 menunjukkan proses inferensi diagnosis menggunakan Certainty Factor. Sistem membentuk fakta gejala, menentukan klasifikasi, memilih aturan aktif, menghitung nilai CF dari MB dan MD, lalu menggabungkan nilai CF untuk memperoleh tingkat kepastian akhir. Hasil proses dikembalikan dalam bentuk data diagnosis, nilai CF, status kasus, penjelasan rule, dan gejala terdeteksi.



Gambar 7. Antarmuka Hasil Implementasi Sistem

Gambar 7 menampilkan antarmuka utama sistem yang terdiri dari Form Diagnosis, Hasil Diagnosis, dan Riwayat Diagnosis. Ketiga halaman tersebut mewakili alur perawat mulai dari pengisian data pemeriksaan, melihat hasil diagnosis dan nilai CF, hingga menelusuri riwayat diagnosis pasien.

Pengujian sistem ini dilakukan menggunakan pendekatan black-box testing untuk memastikan seluruh fungsionalitas berjalan dengan baik. Metode ini dipilih karena fokus utamanya adalah memvalidasi hubungan antara input, proses, dan output sistem agar sesuai dengan kebutuhan pengguna (Myers, 2012). Selain itu, pendekatan ini memungkinkan

evaluasi fitur dilakukan secara independen tanpa harus memeriksa struktur internal kode pemrograman secara langsung (Paul C. Jorgensen, 2018). Pengujian difokuskan pada fungsi utama yang berkaitan dengan sistem pakar deteksi preeklamsia ini.

Tabel 2 Pengujian *Black Box* Fungsi Utama Sistem

No	Fungsi Sistem	Deskripsi Pengujian	Status
1	Autentikasi Pengguna (*Log In)*	Sistem menerima akun valid dan menolak akun tidak valid.	Berhasil
2	Kelola Data Pasien	Sistem menyimpan data pasien dan memberi peringatan jika data belum lengkap.	Berhasil
3	Input Data Pemeriksaan	Sistem menyimpan data klinis dan laboratorium pasien.	Berhasil
4	Proses Diagnosis Menggunakan *Certainty Factor*	Sistem menghitung nilai Certainty *Factor* dan menampilkan hasil diagnosis.	Berhasil
5	Tampilan Detail Hasil Diagnosis	Sistem menampilkan *rule* aktif dan rincian nilai CF.	Berhasil
6	Kelola Riwayat Diagnosis	Sistem menampilkan daftar riwayat dan detail hasil diagnosis pasien.	Berhasil
7	Cetak Laporan PDF	Sistem menghasilkan laporan diagnosis dalam format PDF.	Berhasil
8	Kelola Basis Pengetahuan	Admin dapat mengelola gejala, penyakit, nilai MB, dan nilai MD.	Berhasil
9	Manajemen Akun Pengguna	Admin dapat mengelola akun dan hak akses pengguna.	Berhasil
10	Pencatatan Aktivitas Pengguna (*Log*)	Sistem mencatat aktivitas yang dilakukan pengguna.	Berhasil

Hasil pada Tabel 2 menunjukkan bahwa seluruh fungsi utama berjalan sesuai keluaran yang diharapkan. Sistem mampu menerima input pemeriksaan, menjalankan service diagnosis, menghitung nilai CF, menampilkan hasil, menyimpan riwayat, serta menghasilkan laporan PDF. Dengan demikian, hubungan antara form input, service diagnosis, model database, dan template tampilan telah berjalan sesuai kebutuhan fungsional.

Kesimpulan

Implementasi Certainty Factor pada sistem pakar deteksi preeklamsia berbasis Flask berhasil dilakukan dengan mengintegrasikan parameter klinis dan laboratorium ke dalam proses diagnosis. Sistem menggunakan data pemeriksaan seperti usia kehamilan, tekanan darah, proteinuria, trombosit, SGOT, SGPT, dan gejala fisik sebagai evidence yang dikonversi menjadi fakta gejala, kemudian dihitung menggunakan Certainty Factor untuk menghasilkan klasifikasi diagnosis dan nilai keyakinan. Flask berperan sebagai kerangka implementasi web yang memisahkan route, service, model, template, dan konfigurasi, sehingga proses input pemeriksaan, perhitungan CF, penyimpanan hasil, riwayat diagnosis, dan cetak laporan dapat

E-ISSN : 2809-7165

dikelola secara terstruktur. Pengujian black box terhadap fungsi utama menunjukkan bahwa alur sistem berjalan sesuai kebutuhan fungsional, mulai dari pengelolaan data pasien, input pemeriksaan, proses diagnosis, tampilan detail hasil, riwayat diagnosis, hingga cetak laporan

Daftar Pustaka

Azizah M. Ranggis. 2024. Hipertensi Dalam Kehamilan. Available from: https://keslan.kemkes.go.id/view_artikel/3646/hipertensi-dalam-kehamilan

Badan Pusat Statistik. 2023. Angka Kematian Ibu/AKI (Maternal Mortality Rate/MMR) Hasil Long Form SP2020 Menurut Provinsi, 2020. Available from: https://www.bps.go.id/id/statistics-table/1/MjIxOSMx/angka-kematian-ibu-aki--maternal-mortality-rate-mmr---hasil-long-form-sp2020-menurut-provinsi--2020.html

Fitrilina, F., M. Albbi, I. Agustian, A. Herawati, and N. A. Massardi. 2021. Sistem Peringatan Awal Resiko Preklamsia pada kehamilan menggunakan metoda Certainty Factor dan Android. JNTE. 10:45. doi:10.25077/jnte.v10n1.896.2021.

Hanif, K. H., N. R. Muntiari, and P. A. Ramadhani. 2022. Penerapan Metode Certainty Factor untuk Mendiagnosa Penyakit Preekslamsia pada Ibu Hamil dengan Menggunakan Bahasa Pemrograman Python. Insect (Informatics and Security): Jurnal Teknik Informatika. 7:63–71. doi:10.33506/insect.v7i2.1818.

Heckerman, D. 1992. The Certainty-Factor Model. Encyclopedia of Artificial Intelligence. 131–138.

Joseph C. Giarratano and Gary D Riley. 2005. Expert Systems : Principles and Programming. 4th ed. Thomson Course Technology.

Kementerian Kesehatan Republik Indonesia. 2024. Profil Kesehatan Indonesia 2024.

Myers, G. J. 2012. The art of software testing. 3rd ed. J. Wiley & Sons, Hoboken, N.J.

Pallets Projects. 2010. Welcome To Flask. Available from: https://flask.palletsprojects.com/

Paul C. Jorgensen. 2018. Software Testing. 4th ed. Auerbach Publications, New York.

Perkumpulan Obstetri dan Ginekologi Indonesia, Himpunan Kedokteran Feto Maternal. 2016. Pedoman Nasional Pelayanan Kedokteran Diagnosis dan Tata Laksana Pre-Eklamsia. Perkumpulan Obstetri dan Ginekologi Indonesia Himpunan Kedokteran Feto Maternal.

Prasetya, N. W. A., L. P. Wanti, L. Sari, and L. Puspitasari. 2022. Sistem Pakar Deteksi Dini Penyakit Preeklamsia pada Ibu Hamil Menggunakan Metode Certainty Factor. infotekmesin. 13:168–177. doi:10.35970/infotekmesin.v13i1.1050.

Shortliffe, E. H., and B. G. Buchanan. 1975. A model of inexact reasoning in medicine. Mathematical Biosciences. 23:351–379. doi:10.1016/0025-5564(75)90047-4.

World Health Organization. 2025. SDG Target 3.1 Reduce the global maternal mortality ratio to less than 70 per 100 000 live births. Available from: https://www.who.int/data/gho/data/themes/topics/sdg-target-3-1-maternal-mortality