Vol. 2, No. 2, September 2025

IJAIS International Journal of Artificial Intelligence and Science

Publsihed By Asosiasi Doktor Sistem Informasi Indonesia

Web-Based Electric Bicycle Fault Diagnosis Using the Backward Chaining Method

Satrio Wicaksono Nugroho[1], Dwi Nor Amadi[2], Pradityo Utomo[3], Candra Budi Susila[4]

[1,2,3,4]Informatics Management Department. Faculty of Engineering, Merdeka Madiun University, Madiun, Indonesia

Email: satriown10@gmail.com[1], dwinor@unmer-madiun.ac.id[2], pradityo@unmer-madiun.ac.id[3], candra.budi89@gmail.com[4]

Received: July 24, 2025

Revised: August 5, 2025

Accepted: Sept. 1, 2025

Published: Sept. 20, 2025

Corresponding Author:

Author Name*:

Pradityo Utomo

Email*:

pradityo@unmer-madiun.ac.id

DOI: 10.63158/IJAIS.v2i2.35

© 2025 The Authors. This open access article is distributed under a (CC-BY License)

Abstract. This paper introduces a web-based expert system for diagnosing electric-bicycle faults that operationalizes backward-chaining inference within an extensible, domain-specific knowledge base. The platform supports both symptom-driven and hypothesis-driven workflows: users may input observable symptoms and/or a suspected fault, after which the engine tests candidate rules, gathers disconfirming evidence, and returns an explainable diagnosis with a transparent reasoning trace. The knowledge base—structured around fault, symptom, and rule schemas curated from literature, service documentation, and practitioner input—enables disciplined updates and governance. A responsive interface implements guided data entry, contradiction checks, and bilingual labels to reduce input variance and improve usability. Functional validation via black-box testing confirmed conformance across end-user and administrative flows, including rule invocation and CRUD persistence. The discussion addresses design trade-offs (confirmation bias, rule granularity), privacy and maintainability considerations, and deployment pathways for workshops, manufacturers, and consumers. Future work includes field evaluation, probabilistic uncertainty handling, telemetry integration, and interoperability with parts inventories to strengthen external validity and scalability.

Keywords: electric bicycle diagnostics; expert systems; backward chaining; explainable AI; web-based decision support

IJAIS International Journal of Artificial Intelligence and Science
Publsihed By Asosiasi Doktor Sistem Informasi Indonesia

1. INTRODUCTION

The contemporary innovation landscape is increasingly shaped by the Fourth Industrial Revolution (Industry 4.0), where pervasive digitization, cyber-physical integration, and automation are redefining production, services, and everyday mobility. Yet amid accelerating technological capability, a first-order constraint remains unchanged: progress is ultimately bounded by the availability, reliability, and quality of energy [1][2]. This dependency is particularly visible in electrified transportation, where system performance, cost, and user experience are mediated by power electronics, storage technologies, and intelligent control that together translate energy into practical utility.

Against this backdrop, electric bicycles (e-bikes) represent a compelling, human-scale expression of electro-mobility. By pairing compact traction motors with rechargeable accumulators, e-bikes deliver propulsion independent of fossil fuels and offer a pragmatic pathway to decarbonizing short- and mid-range travel [3]. Their market momentum is propelled not only by environmental benefits but also by tangible improvements in accessibility and comfort: reduced physical exertion—especially valuable for older adults—and intuitive, user-friendly ergonomics that lower adoption barriers across age and fitness levels [4][5]. As e-bikes scale from niche to mainstream, however, their hybrid electro-mechanical nature introduces diagnostic opacity for end-users and novice technicians, for whom even minor malfunctions can be difficult to recognize, localize, and remedy.

Knowledge-based approaches have been explored to bridge this gap, with mixed success. The expert system introduced in [6] sought to assist users in identifying e-bike faults but relied exclusively on forward chaining, constraining interaction to preselected symptoms and, crucially, prohibiting the entry of hypothesized faults—despite the fact that users often reason from suspected causes. Adjacent work in the motorcycle domain underscores both promise and limitation: study [7] applied backward chaining to fuel-injection motorcycles yet covered only ten fault types; research [8] realized a web-based backward-chaining system achieving 88% diagnostic accuracy but employed a knowledge representation not transferrable to e-bike architectures. Moreover, the web implementation in [9] omitted persistent data storage, impeding iterative curation and

expansion of the knowledge base—an essential capability in fast-evolving product ecosystems.

Methodologically, the comparative analysis in [10] clarifies when each inference paradigm confers advantage: forward chaining excels when progressing from observed symptoms to candidate faults, whereas backward chaining is efficient for testing or refuting a suspected fault by seeking corroborating evidence. The implication for e-bike support tools is that real-world diagnostic workflows should embrace both modes—symptom-driven and hypothesis-driven—to align with how users naturally alternate between noticing effects and proposing causes. Yet, current web-based systems [11][12] do not admit user-entered fault hypotheses, limiting ecological validity, user agency, and the capacity to converge rapidly on plausible diagnoses.

These gaps motivate the present study, which contributes three advances. First, to our knowledge, it delivers the inaugural web-based backward-chaining expert system dedicated to e-bike fault diagnosis, addressing a domain not served by prior implementations. Second, it introduces an interactive interface that accommodates both symptom input and user-supplied fault hypotheses, operationalizing hybrid diagnostic workflows and mitigating the limitations identified in [6][11][12]. Third, it provides a comprehensive, extensible knowledge base spanning more than fifteen e-bike fault classes, coupled with an optimized rule-based inference strategy for multi-level reasoning, thereby improving transparency, traceability, and updateability. A web-centric architecture further enhances reach and maintainability, enabling streamlined data handling and continuous accuracy improvements [13], while modular, interlinked page designs support clear navigation, scalability, and efficient lifecycle management of knowledge assets [14][15].

1. METHODS

This study followed a structured, four-stage research flow—literature study, data collection, backward-chaining workflow design, and system testing—as summarized in Figure 1. The sequence was chosen to (i) ground the work in established theory and prior art, (ii) assemble a domain-specific knowledge base for e-bike diagnosis, (iii) formalize a goal-driven inference mechanism tailored to that knowledge, and (iv) verify functional

correctness through black-box evaluation. Each stage is detailed below to ensure transparency, reproducibility, and methodological rigor.

Figure 1. Research Flow

2.1. Literature Study

The literature study synthesized evidence from both national and international journals selected for methodological relevance and domain proximity. National publications were reviewed to capture local operating contexts, component availability, and usage patterns that shape fault prevalence and symptom expression in e-bike systems. International sources were used to consolidate theoretical underpinnings in expert systems, diagnostic reasoning, and human–computer interaction, while also enabling cross-study comparisons that inform transferable design choices. Selection emphasized peer-reviewed articles, clear methodological reporting, and applicability to rule-based diagnosis. This dual-source strategy strengthened construct validity, provided triangulation across settings, and supported the derivation of design requirements and evaluation criteria for the proposed system [16].

2.2. Data Collection

Data collection produced three curated datasets that constitute the system's knowledge base: (1) a fault dataset enumerating distinct e-bike failure classes and sub-classes (e.g., powertrain, battery management, sensor and controller faults), including standardized identifiers and brief operational definitions; (2) a symptom dataset cataloging observable indicators (e.g., motor non-response, intermittent cut-out, abnormal voltage readings) and contextual qualifiers (operating conditions, recurrence, severity); and (3) a rule dataset encoding diagnostic relations that map symptoms (and symptom combinations) to candidate faults with associated conditions and priorities. Data were compiled from technical references, service manuals, and practitioner inputs aligned with the literature review. Each entry was normalized to a consistent schema and cross-checked to reduce ambiguity and duplication. Together, these datasets enable evidence-based inference,

ensure consistent terminology across modules, and support maintainable updates as components and fault patterns evolve.

2.3. Backward Chaining Workflow

The diagnostic engine implements a backward-chaining strategy, beginning with a target conclusion (a suspected fault) and recursively seeking facts that confirm or refute it through applicable rules. Concretely, the workflow proceeds as follows: (i) Goal selection—the system initializes one or more diagnostic goals based on user input (hypothesized fault) or system suggestions; (ii) Rule retrieval—it identifies rules whose consequents match the current goal; (iii) Condition checking—for each candidate rule, required symptoms and context conditions are queried from the knowledge base or elicited from the user; (iv) Subgoal decomposition—unverified conditions are transformed into subgoals, spawning additional checks down the rule tree; (v) Resolution and backtracking—if a rule's conditions are satisfied, the goal is provisionally concluded; otherwise, the engine backtracks to alternative rules; and (vi) Termination—the process halts when goals are confirmed, disproven, or no further applicable rules remain. This goal-driven approach aligns with user workflows that often begin from an intuitive hypothesis and then seek corroborating evidence, thereby complementing symptom-first strategies prevalent in forward chaining [17], [18], [19]. An illustrative decision-tree fragment used during knowledge engineering is shown in Figure 2, adapted from prior work in motorcycle diagnostics to guide rule structuring and branching logic [7].

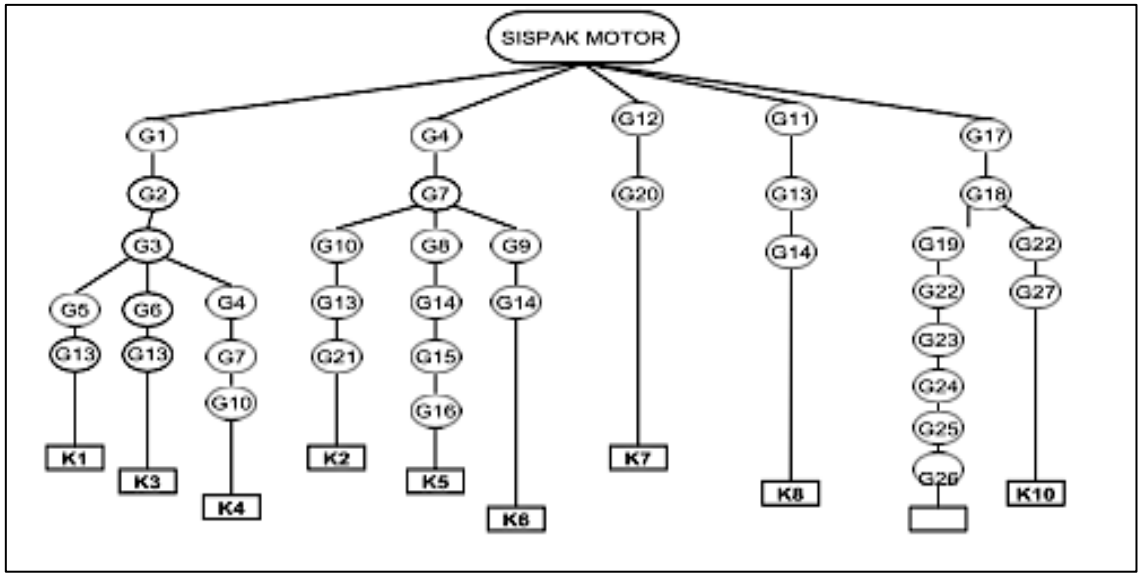

Figure 2. Decision Tree Illustration [7]

IJAIS International Journal of Artificial Intelligence and Science
Publsihed By Asosiasi Doktor Sistem Informasi Indonesia

2.4. System Testing

Functional verification employed black-box testing, focusing on externally observable behavior without inspecting internal code. Test cases were derived from the rule base and common user scenarios, covering nominal diagnoses, conflicting symptoms, incomplete information, and negative cases where no rule should fire. Inputs were varied to exercise different rule paths and backtracking behaviors, while outputs were checked for correctness of diagnosed fault sets, explanation traces, and user prompts. Acceptance criteria included accurate rule activation, consistent goal resolution, and robust handling of missing or contradictory inputs. This method is widely used for validating software components against specifications and is well-suited to expert-system interfaces where correctness is defined at the input–output boundary rather than by internal implementation details [20], [21].

3. RESULTS AND DISCUSSION

3.1. Expert System

Forward Chaining and Backward Chaining are the two principal rule-based reasoning strategies used in expert systems. Forward Chaining is data-driven: inference begins from known facts or symptoms and matches them to the IF part of rules to derive conclusions. This approach is well suited to users who do not yet know the specific malfunction but can recognize observable symptoms. In the context of electric bicycles, for example, an Android-based expert system developed by [6] guides users to a diagnosis simply by selecting the symptoms they observe. In contrast, Backward Chaining is goal-driven: the process starts from a hypothesized malfunction, and the system then traces backward to find facts or symptoms that support or refute that hypothesis. This method is more efficient when users already have an initial suspicion and want to validate it through targeted evidence gathering. Practically, a user first selects a suspected fault, and the system queries for corroborating symptoms. In general, Forward Chaining tends to suit novice users who can only identify symptoms, whereas Backward Chaining better serves users who possess an initial understanding of the problem. Both strategies can be employed complementarily, depending on user needs and their technical literacy with the developed system.

1) Data Collection

The literature review examined decision-making methods commonly used in expert systems, with a focus on Backward Chaining. The evaluation followed predefined criteria—traceable logic, rule structure, and consistent terminology—to ensure the system can draw conclusions systematically from available data and symptoms. Accordingly, Backward Chaining is expected to contribute to decision-making that is more accurate, well-structured, and logically defensible.

Data collection for diagnosing electric bicycle malfunctions was conducted through observation and interviews. Three datasets were compiled: (1) a fault dataset listing potential failure types; (2) a symptom dataset describing initial indicators of faults; and (3) a rule dataset linking symptom combinations to specific fault types. Together, these datasets form the system's knowledge base, supporting inference and evidence-based decision-making throughout the study.

Table 1. Malfunction Data [6]

No	Fault Code	Malfunction Description
1	K1	Damaged battery
2	K2	Damaged motor
3	K3	Broken or loose cable
4	K4	Restricted electrical flow
5	K5	Worn mechanical component
6	K6	Battery not supplying power
7	K7	Poor battery connection
8	K8	Electrical flow disturbance
9	K9	Worn battery
10	K10	Motor not receiving power
11	K11	Faulty power regulator
12	K12	Worn motor
13	K13	Damaged cable
14	K14	Faulty control module
15	K15	Unstable connection
16	K16	Problematic power system
17	K17	Widespread electrical flow issue

IJAIS International Journal of Artificial Intelligence and Science
Published By Asosiasi Doktor Sistem Informasi Indonesia

No	Fault Code	Malfunction Description
18	K18	Damaged battery and motor
19	K19	Poor electrical connection
20	K20	Battery damage and power disturbance
21	K21	Combined power and mechanical problem
22	K22	Motor and cable issues
23	K23	Worn motor and control module
24	K24	Worn motor and components
25	K25	Worn cables and components
26	K26	System-wide power problem
27	K27	Damaged power and mechanical systems
28	K28	System-wide malfunction
29	K29	Complex power and connection problems
30	K30	Damaged motor, cables, and mechanics
31	K31	Overall bicycle system failure

Table 1 lists various types of electric bicycle faults along with their unique ID codes, using the symbol (K) to denote faults. Serving as the expert system's primary reference, this table will assist in diagnosing issues based on user-input symptoms.

Table 2 Symptom Data [6]

No	Symptom Code	Symptom Description
1	G1	Bicycle cannot be powered on
2	G2	Bicycle will not move
3	G3	Indicator light does not turn on
4	G4	Bicycle shuts off suddenly during operation
5	G5	Unusual noise while riding

Table 2 contains a list of symptoms that may occur in malfunctioning electric bicycles, each with a unique ID code. These symptoms (denoted as (G)) will serve as user input to initiate the system's diagnostic process.

IJAIS International Journal of Artificial Intelligence and Science
Publsihed By Asosiasi Doktor Sistem Informasi Indonesia

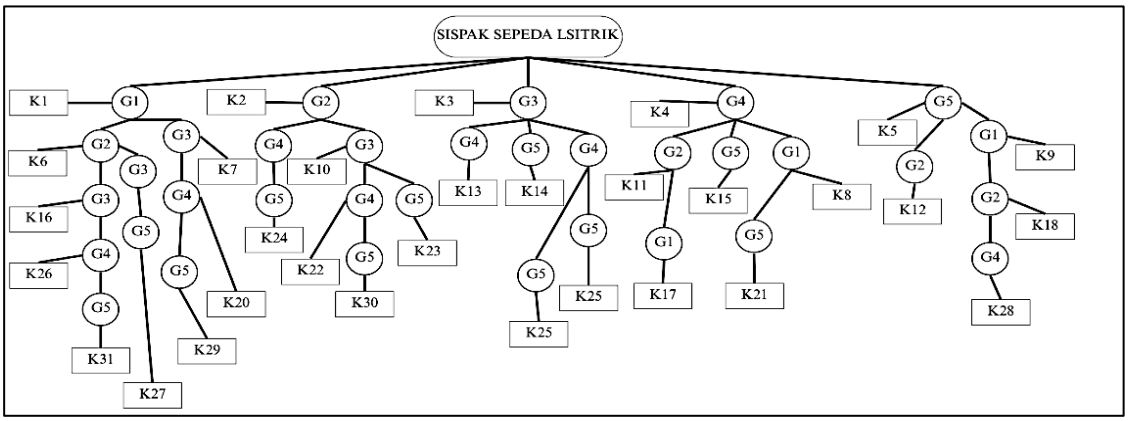

Figure 1. Example of a Backward-Chaining Decision Tree

As illustrated in Figure 3, the expert system applies a backward-chaining strategy to diagnose electric bicycle faults by starting from a suspected fault (the goal node) and tracing backward through rule links to the required evidence (symptom nodes). In practice, the user's input initializes one or more diagnostic goals (codes K), and the inference engine navigates the decision tree to seek confirming or disconfirming symptoms (codes G). The branching in Figure 3 clarifies how alternative rules are explored when a required symptom is absent, ensuring systematic backtracking until a consistent explanation is found or all candidate paths are exhausted.

The diagnostic flow begins with Fault Selection, where users specify the malfunction, they wish to verify (e.g., K7 – Poor battery connection, K11 – Faulty power regulator). This step, depicted at the top of Figure 3, anchors the reasoning at a concrete hypothesis so the system can retrieve only those rules whose consequents match the selected fault. By constraining the search space in this way, the tree in Figure 3 reduces unnecessary exploration and accelerates convergence on likely causes.

Next, the system proceeds to Symptom Identification. Guided by the branches in Figure 3, the engine queries the user for observable indicators that correspond to the rule antecedents. Symptoms are encoded to streamline matching, for example: G1 (Bicycle cannot be powered on), G2 (Bicycle will not move), G3 (Indicator light does not turn on), G4 (Sudden shutdown during use), and G5 (Unusual noise while riding). In the decision tree of Figure 3, each required symptom appears as a branching condition leading toward or away from the hypothesized fault.

IJAIS International Journal of Artificial Intelligence and Science
Publsihed By Asosiasi Doktor Sistem Informasi Indonesia

With the hypothesis and symptoms in hand, the Rule Processing stage aligns user inputs to predefined diagnostic rules derived from empirical data and technician interviews. As exemplified by the branches and leaf nodes in Figure 3, the system evaluates rules such as: A1—if G1 and G3 are present, then the likely malfunction is K7 (Poor battery connection); A2—if G2 and G4 are present, then the likely malfunction is K11 (Faulty power regulator); and A3—if G5 alone is present, then the likely malfunction is K5 (Worn mechanical component). When a rule's antecedents are satisfied, the corresponding branch in Figure 3 terminates at a fault node, signaling provisional confirmation; otherwise, the engine backtracks to evaluate alternative branches.

The system delivers the Diagnostic Output, presenting the most probable fault(s) and the evidence path used to reach them—mirroring the successful path through Figure 3. Typical results include K5 (Worn mechanical component), K7 (Poor battery connection), and K11 (Faulty power regulator). By explicitly mapping each conclusion to its supporting rules and symptoms—as visually organized in Figure 3—the system maintains a transparent explanation trail, facilitating technician review, user trust, and iterative refinement of the knowledge base.

2) Diagnosis

The diagnosis modules shown in Figure 4 implements a guided backward-chaining workflow through a clean, four-step interface optimized for clarity, speed, and data integrity. A compact header displays the title "Bycle Fault Diagnose" and a tagline indicating that the engine reasons via backward chaining. The layout follows a left-to-right hierarchy—identity → (optional) fault hypothesis → symptom capture → execution—with responsive behavior for desktop, tablet, and mobile. Inline validation, context tooltips, and status cues (neutral/valid/error) ensure inputs are complete and machine-parsable before the reasoning engine runs.

Step 1 — Personal Data Input. Users provide Full Name, Email, and WhatsApp Contact. Client-side validation enforces correct formats (RFC-compliant email, country-aware phone masks), trims whitespace, and normalizes characters; server-side checks create a short-lived session ID linking identity, inputs, and results for auditability. PII is stored separately from diagnostic data, transported over HTTPS/TLS, and encrypted at rest. Users may proceed without contact details; the engine still executes, though report

IJAIS International Journal of Artificial Intelligence and Science
Publsihed By Asosiasi Doktor Sistem Informasi Indonesia

exports (PDF/CSV) remain disabled unless at least one contact method is provided. A privacy notice explains retention and deletion policies.

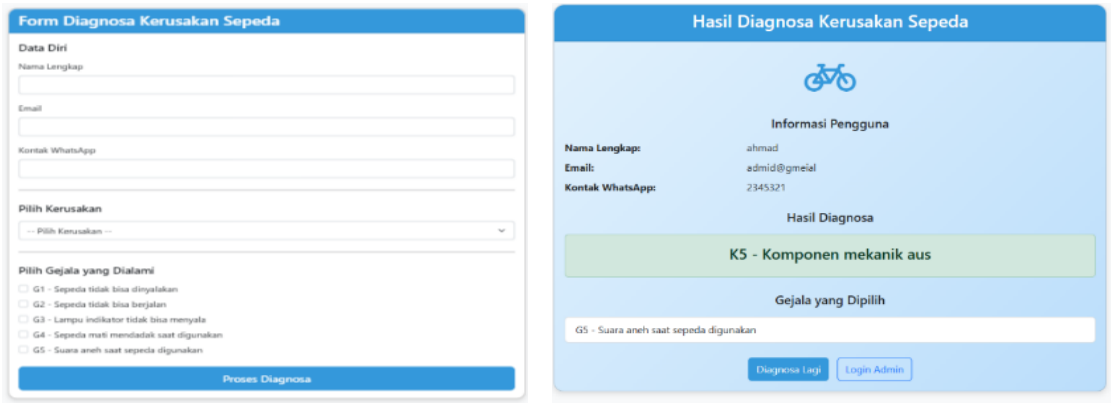

Figure 4. Diagnosis Process

Step 2 — Select Damage Type (Optional Hypothesis). A searchable dropdown lets users select a suspected damage category (K code) to seed the backward-chaining goal. If no hypothesis is supplied, the system starts in neutral mode and proposes goals after a brief symptom triage. The list is grouped semantically (Power System, Motor/Drive, Cabling/Connectivity, Control Electronics, Mechanical). When a K code is focused, as shown in Figure 4, the UI displays a short definition and typical manifestations from the knowledge base. Selection does not bias the outcome: the engine will confirm or refute the hypothesis against evidence.

Step 3 — Select Symptoms. Users choose one or more symptoms (G codes) from a filterable checklist. Each item includes a concise label and an expandable description with examples and measurement guidance (e.g., multimeter ranges). Common entries include G1 (Bicycle cannot be powered on), G2 (Bicycle will not move), G3 (Indicator light does not turn on), G4 (Sudden shutdown during use), and G5 (Unusual noise while riding). To reduce bias, the list is ordered by category and recent usage rather than likelihood. The interface flags mutually exclusive selections (e.g., “cannot power on” alongside “indicator light normal”) and suggests adding high-value symptoms if the current set is weak—as shown in Figure 4.

The diagnosis results page of this electric bicycle expert system presents a comprehensive report on identified damage issues. User information—including Full

IJAIS International Journal of Artificial Intelligence and Science
Published By Asosiasi Doktor Sistem Informasi Indonesia

Name, Email, and WhatsApp Contact—provides contextual reporting details. Within the 'Diagnosis Findings' section, the system displays both the symptoms selected by the user and the corresponding identified damage. These results are generated through rule-based correlation between the user-reported symptoms and the system's knowledge base. Interactive controls include a 'New Diagnosis' button to restart the diagnostic process, and an 'Admin Portal' button that redirects administrators to the login interface for system management.

3) Expert System Testing

This subsection details the validation of the web-based expert system for diagnosing electric-bicycle faults that employs a backward-chaining inference engine. We adopted Black-Box Testing to verify externally observable behavior against specifications without inspecting internal code paths. The objectives were to (i) confirm end-to-end correctness of the diagnostic workflow, (ii) validate user-interface behavior and input validation, and (iii) ensure that outputs (diagnoses, confidence cues, and administrative actions) are consistent with user inputs and the encoded rule base. Test design drew on equivalence partitioning, boundary-value analysis, and state-transition checks for the key flows (user diagnosis and admin maintenance), emphasizing reproducibility and clear pass/fail criteria.

Table 3. Black Box Testing Results

No	Test Scenario	Test Input (Example)	Expected Result	Actual Result
1	User completes diagnosis form (name, contact, email, suspected fault, symptoms)	**Full Name:** ahmad; **Contact:** 123456; **Email:** ahmad@test.com; **Fault:** K1; **Symptom:** G1	System generates diagnosis consistent with selected fault/symptoms and enables result saving	Matches expectation; save enabled
2	Admin logs in with credentials	**Username:** admin; **Password:** admin	Credentials processed; access to Admin menu granted	Matches expectation
3	Symptom management (create, edit, delete)	Create/Write/Update/Delete Symptom	Changes persist to symptoms database and reflect in UI	Matches expectation
4	Fault (damage) management (create, edit, delete)	Create/Write/Update/Delete Fault	Changes persist to faults database and reflect in UI	Matches expectation
5	Rule management (create, edit, delete)	Create/Write/Update/Delete Rule	Changes persist to rules database;	Matches expectation

IJAIS International Journal of Artificial Intelligence and Science
Publsihed By Asosiasi Doktor Sistem Informasi Indonesia

No	Test Scenario	Test Input (Example)	Expected Result	Actual Result
			inference uses updated rules	
6	Admin account management (create, edit, delete)	Create/Write/Update/Delete Account	Changes persist to admin-accounts database	Matches expectation
7	Admin logout	Click **Logout**	Session terminated; redirected to Login page	Matches expectation

Test scope and design. Functional coverage included: user session initiation; personal-data capture; optional hypothesis selection (fault K); symptom selection (G); diagnosis execution; and structured presentation of results. Administrative coverage included authentication, CRUD operations for symptoms, faults, rules, and admin accounts, as well as session termination (logout). Non-functional checks (observability of validation messages, graceful error handling, and role-aware navigation) were assessed qualitatively during execution. The environment comprised a modern standards-compliant browser, server-hosted application, and a persistent datastore; results below reflect behavior at the user–system interface.

3.2. Discussion

This study demonstrates that a web-based, backward-chaining expert system can operationalize hypothesis-driven troubleshooting for e-bikes in a way that more closely mirrors how technicians and informed users' reason in practice. By allowing users to start with a suspected fault (goal) and then elicit corroborating or disconfirming evidence, the workflow aligns with the methodological advantages of backward chaining described in the literature [10] and addresses interaction gaps noted in prior systems that restricted users to symptom-only inputs [6][11][12]. The accompanying knowledge base—structured around distinct fault classes, observable symptoms, and explicit rules—supports transparent, traceable inference. In applied contexts (home users, repair shops, after-sales support), this translates to shorter diagnostic paths, clearer next steps, and a shared vocabulary between users and technicians, thereby reducing miscommunication costs and rework.

The system extends earlier efforts in several ways without duplicating their scope. First, where forward-chaining implementations primarily served novice users by progressing from observations to causes [6], our design inverts the starting point to accommodate hypothesis testing, which is valuable when a user or technician already has a tentative diagnosis in mind [10]. Second, unlike motorcycle-focused implementations that either limited the set of faults [7] or employed knowledge models not transferable to e-bike architectures [8], the present work tailors its ontology (fault, symptom, rule schemas) to e-bike components and typical failure modes. Finally, the decision to deploy on the web with persistent storage directly addresses sustainability and maintainability issues raised by non-persistent prototypes [9], enabling iterative curation of rules and terminology as devices and usage patterns evolve.

Key design choices introduce trade-offs that warrant discussion. Prioritizing backward chaining improves efficiency when a credible hypothesis exists, but it can also bias users toward confirmation; the interface mitigates this with prompts for disconfirming evidence, negative tests, and explanations of why alternatives were rejected. Confidence scores help communicate uncertainty, yet they arise from rule weights and coverage rather than probabilistic calibration; without ground-truth labels across diverse cases, these scores should be interpreted as heuristic guidance rather than absolute likelihoods. Another consideration is knowledge-base granularity: finer-grained rules increase diagnostic precision but also raise authoring and maintenance overhead. The chosen compromise—modular rules with shared antecedents—supports reuse while keeping the authoring burden manageable. From a validity perspective, black-box functional testing ensures conformance to specifications but does not, by itself, establish clinical-style diagnostic accuracy; field evaluation with annotated cases and inter-rater comparisons remains necessary to quantify sensitivity/specificity across fault categories. Finally, user-entered data introduce variance (e.g., inconsistent symptom interpretation); the UI's definitions, examples, and contradiction checks reduce but do not eliminate this source of noise.

The system's explanatory artifacts—goal trees, fired rules, and "why-not" rationales—serve both usability and trust. Unlike opaque recommenders, rule-based traces let users audit how a conclusion was reached and what additional evidence could flip the outcome. This aligns with explainability best practices and is particularly important for safety-adjacent

recommendations (e.g., electrical checks). The symptom capture flow balances guidance with autonomy: plain-language labels are paired with concise technical cues (measurement ranges, operating conditions), assisting non-experts without oversimplifying for technicians. Accessibility considerations (keyboard navigation, screen-reader roles, contrast) increase reach, while bilingual labels help bridge terminology gaps between English-language documentation and local workshop parlance—an issue noted in national vs. international literature synthesis [16]. Collectively, these choices support informed decision-making rather than rote button-clicking.

Persisting cases and rule edits in a versioned datastore enables continuous improvement, but it also raises governance questions. Curation workflows should include change proposals, peer review, and rollback capabilities to prevent degradation from erroneous edits. To manage drift as new e-bike models and components arrive, scheduled audits can compare rule coverage against service bulletins and prevalent support tickets. On privacy, the architecture segregates personally identifiable information from diagnostic content, enforces transport encryption, and provides user-controlled deletion—practices consistent with minimizing data exposure while preserving the ability to reproduce explanations. Ethically, the system must avoid overconfidence: language in the UI should communicate that recommendations are decision support, not guarantees, and should flag cases where professional inspection is advisable (e.g., signs of thermal damage). Finally, inclusive design suggests monitoring for differential performance across user groups (novices vs. technicians) and adapting prompts and flows accordingly.

For workshops, the platform can function as a triage and training tool: novices can follow prompted checks while experts use it to document rationale and share institutional knowledge. Integration with inventory systems would let rule outputs trigger parts availability checks, shortening repair cycles. For manufacturers and distributors, anonymized aggregate analytics (e.g., co-occurrence of symptoms, seasonal failure spikes) could inform design improvements and warranty policies—provided that data collection is transparent and consent-based. In consumer scenarios, a lightweight mode that supports “quick checks” (battery connectors, basic continuity tests) could reduce unnecessary service visits while still escalating risky conditions. Over time, connecting the rule base to telemetry from smart controllers or battery management systems could

shift some inputs from self-report to measured signals, improving consistency and enabling proactive fault detection.

Three limitations suggest immediate avenues for expansion. First, the current knowledge base, while broader than prior art, remains finite; introducing semi-automated knowledge acquisition (e.g., rule candidates mined from labeled cases) could accelerate coverage growth while keeping human review in the loop. Second, uncertainty handling is rule-centric; augmenting the engine with probabilistic layers (Bayesian updating or Dempster–Shafer for conflicting evidence) would allow more nuanced confidence estimates, particularly under sparse or noisy symptom sets. Third, evaluation to date emphasizes functional correctness; future studies should include prospective field trials with technician ground truth, time-to-diagnosis metrics, and user-perceived workload (e.g., SUS or NASA-TLX) to quantify human-factors benefits. Additional priorities include multilingual term alignment, mobile-first optimization for constrained network environments, and APIs for interoperability with service management and parts catalog systems.

Although tailored to e-bikes, the methodology—domain-specific ontology, transparent rules, hybrid user interaction (symptom-driven and hypothesis-driven), and web-based delivery—transfers to adjacent small-EV platforms (e-scooters, e-mopeds) and other consumer mechatronic products where fault identification relies on a mix of observable symptoms and simple measurements. The key to portability is disciplined separation between core inference logic and domain knowledge; by keeping rules declarative and data schemas explicit, new domains can be onboarded with limited engineering effort while retaining the explainability and governance features emphasized here.

4. CONCLUSION

This study advances diagnostic support for electric bicycles by formalizing a web-based, backward-chaining expert system grounded in an extensible, domain-specific knowledge base and an interface that accommodates both symptom-driven and hypothesis-driven workflows. By coupling transparent rule traces with versioned knowledge governance and privacy-aware data handling, the platform delivers explainable, auditable outcomes suitable for service operations and after-sales support while creating a sustainable

IJAIS International Journal of Artificial Intelligence and Science
Publsihed By Asosiasi Doktor Sistem Informasi Indonesia

pathway for continuous improvement. Although black-box validation establishes functional conformance, rigorous field evaluations remain necessary to quantify diagnostic accuracy, time-to-decision, and user burden across stakeholder groups. Future enhancements should incorporate principled uncertainty handling, multilingual term harmonization, mobile-first optimizations for resource-constrained settings, and integrations with telemetry and parts inventory systems. Collectively, these steps will strengthen external validity, support responsible scaling, and position the system as a durable foundation for decision support across adjacent small-EV domains.

REFERENCES

[1] D. Alladin and I. Mardian, "Pengaruh Gaya Hidup dan Promosi Terhadap Keputusan Pembelian Sepeda Listrik di Duta Elektronik Kota Bima," *Indones. J. Innov. Multidisipliner Res.*, vol. 1, no. 3, pp. 182–196, 2023, doi: 10.31004/ijim.v1i3.19.

[2] A. S. Cahyaningtyas, A. N. Aeni, and H. N. Adipura, "Pengaruh Perkembangan Teknologi Pada Era Revolusi Industri," *Univ. Padjajaran*, no. October, pp. 1–18, 2023.

[3] B. N. Setyanto, P. Budiastuti, M. Y. Rismarinandyo, and R. F. Yulanda, "Pengembangan Alat Peraga Sepeda Listrik Portabel Sebagai Media Pembelajaran Elektronika Daya," *Jupiter (Jurnal Pendidik. Tek. Elektro)*, vol. 8, no. 1, p. 39, 2023, doi: 10.25273/jupiter.v8i1.16061.

[4] V. Tabitha, E. Purwanto, S. Kom, M. Kom, H. Permatasari, and M. Kom, "Sistem Pakar Diagnosa Kerusakan Pada Sepeda Listrik Menggunakan Metode Forward Chaining di PT . Dunia Usaha Mapan Sejahtera," pp. 55–63, 2024.

[5] M. Hermawati, M. H. Nuhi, A. Andari, E. E. Marito, N. Farros, and H. Josua, "Media Hukum Indonesia (MHI) Penegakan Hukum Bagi Pengguna Sepeda Listrik di Jalan Raya Dalam Perspektif Hukum Positif Indonesia (Undang-Undang Lalu Lintas) Media Hukum Indonesia (MHI) sedemikian besar sebagai wujud daripada revolusi industri 4 . 0 .," vol. 2, no. 2, pp. 66–73, 2024.

[6] F. N. Rahman, "Sistem Pakar Deteksi Kerusakan Sepeda Lsitrik Menggunakan Metode Forward Chaining Berbasis Android," Universitas Merdeka Madiun, 2025.

[7] A. G. Pratama, R. Rizky, A. M. Yunita, and N. N. Wardah, "Implementasi Metode Backward Chaining untuk Diagnosa Kerusakan Motor Matic Injection," *Explor. Sist. Inf. dan Telemat.*, vol. 11, no. 2, p. 91, 2020, doi: 10.36448/jsit.v11i2.1515.

[8] R. Siregar, "Sistem Pakar Analisa Kerusakan Pada Sepeda Motor Honda Beat Injection Dengan Metode Backward Chaining," *Petir*, vol. 11, no. 1, pp. 1–8, 2018, doi: 10.33322/petir.v11i1.1.

[9] Y. Wijayana, "Sistem Pakar Kerusakan Hardware Komputer Dengan Metode Backward Chaining Berbasis Web," *Media Elektr.*, vol. 12, no. 2, p. 99, 2020, doi: 10.26714/me.12.2.2019.99-107.

[10] S. Ibrahim, D. Paseru, and V. D. Kumenap, "Perbandingan Metode Forward Chaining dan Backward Chaining Dalam Mendiagnosis Perkembangan Anak Usia Dini," *Sisfotek*, vol. 5, pp. 51–58, 2021.

[11] B. Raharjo and F. Agustini, "Metode Forward Chaining pada Sistem Pakar Penilaian Kualitas Biji Kopi Berbasis Web," *Int. J. Nat. Sci. Eng.*, vol. 4, no. 2, pp. 73–82, 2020, doi: 10.23887/ijnse.v4i2.28578.

[12] F. Hamdallah, "Sistem Pakar Diagnosa Gejala Kecanduan Game Online dengan Metode Backward Chaining," *Cakrawala Repos. IMWI*, vol. 3, no. 2, pp. 118–124, 2021, doi: 10.52851/cakrawala.v3i2.56.

[13] M. Arafat, "Rancang Bangun Sistem Informasi Pemesanan Online Percetakan Sriwijaya Multi Grafika Berbasis Website," *Intech*, vol. 3, no. 2, pp. 6–11, 2022, doi: 10.54895/intech.v3i2.1691.

[14] N. Nurmaesah and R. Tullah, "Sistem Pakar Diagnosa Terjadinya Waste Plastik Berbasis Web dengan Metode Backward Chaining Nunung," *Jurnal Sisfotek Global* vol. 10, no. 1, pp. 49–56, 2020.

[15] R. R. Tanjung and E. Hutabri, "Sistem Pakar Mendeteksi Kerusakan Pada Gitar Akustik Menggunakan Metode Backward Chaining Berbasis," *Computer and Science Industrial Engineering (COMASIE),* vol. 12, no. 4, pp. 97–106, 2025.

[16] B. Kurniawan, D. Dwikoranto, and M. Marsini, "Implementasi problem based learning untuk meningkatkan pemahaman konsep siswa: Studi pustaka," *Pract. Sci. Teach. J. J. Prakt. Pendidik.*, vol. 2, no. 1, pp. 27–36, 2023, doi: 10.58362/hafecspost.v2i1.28.

[17] R. A. P. REGI, "Sistem Pakar untuk Mendiagnosa Obesitas pada anak dengan menggunakan metode Backward Chaining," *JEKIN - J. Tek. Inform.*, vol. 1, no. 2, pp. 13–21, 2023, doi: 10.58794/jekin.v1i2.214.

[18] B. Pitaloka, B. Pitaloka, A. Putra, and S. Lestari, "Implementasi Metode Forward Chaining Dan Backward Chaining Dalam Mendeteksi Kerusakan Pada Prasarana Lalu Lintas," *Pros. Semin.*, no. 3, pp. 182–189, 2023.

IJAIS International Journal of Artificial Intelligence and Science
Publsihed By Asosiasi Doktor Sistem Informasi Indonesia

[19] S. Hardianti, A. Tenriawaru, and N. Ransi, "Sistem Pakar Diagnosa Penyakit Menular Pada Anak Menggunakan Metode Forward Chaining dan Backward Chaining," *Just TI (Jurnal Sains Terap. Teknol. Informasi)*, vol. 13, no. 2, p. 111, 2021, doi: 10.46964/justti.v13i2.625.

[20] I. A. Shaleh, J. P. Yogi, P. Pirdaus, R. Syawal, and A. Saifudin, "Pengujian Black Box pada Sistem Informasi Penjualan Buku Berbasis Web dengan Teknik Equivalent Partitions," *J. Teknol. Sist. Inf. dan Apl.*, vol. 4, no. 1, p. 38, 2021, doi: 10.32493/jtsi.v4i1.8960.

[21] S. D. S. Saian, N. L. Kakihary, and T. Wahyono, "Pengujian Content Management System (Cms) Sekolahku Menggunakan Metode Black Box Testing Dengan Teknik Boundary Value Analysis," *IT-Explore J. Penerapan Teknol. Inf. dan Komun.*, vol. 1, no. 2, pp. 100–113, 2022, doi: 10.24246/itexplore.v1i2.2022.pp100-113.