FMDB *Transactions on Sustainable Computing Systems*

An Intelligent Web-Based AI Framework for Multi-Vehicle Fault Diagnosis and Predictive Maintenance

B. Dhanush[1,*], R. Vaishnavi[2], J. Adeline Sneha[3], Vinothini Kasinathan[4], Raja Rajeswari Ponnusamy[5], Kohila Malar Kalesamy[6]

[1,2]Department of Artificial Intelligence, SRM Institute of Science and Technology, Ramapuram, Chennai, Tamil Nadu, India.
[3,4,5,6]School of Computing, Asia Pacific University of Technology and Innovation, Kuala Lumpur, Wilayah Persekutuan, Malaysia.
dhanushsriram2607@gmail.com[1], vaishuravi2006@gmail.com[2], adeline.john@apu.edu.my[3], vinothini@apu.edu.my[4], raja.rajeswari@apu.edu.my[5], kohila.malar@apu.edu.my[6]

Abstract: Vehicle maintenance and fault diagnosis remain largely inaccessible to ordinary owners, particularly in developing regions where professional diagnostic tools are either unavailable or prohibitively expensive. This paper introduces the Intelligent Vehicle Diagnosis System (IVDS), a lightweight, browser-based platform that integrates artificial intelligence, geolocation services, and collaborative community features into a single cohesive application available to any smartphone or computer user without specialized hardware. The system is built upon a curated knowledge base of 159 fault records distributed across three vehicle categories: motorcycles, cars, and trucks, covering more than fifty problem types per category. Users interact with the platform through a dual-mode interface: a structured dropdown menu for precise, menu-driven queries, and a natural language chat module powered by Python's difflib. SequenceMatcher algorithm is configured with a similarity threshold of 0.7. A GPS-driven workshop discovery feature queries the Overpass OpenStreetMap API within a 10 km radius and renders results on an interactive Leaflet map. Community capabilities include encrypted group garages, peer-to-peer direct messaging, and a public discussion forum. System evaluation on the full 159-record dataset yields an overall fault-match rate of 92.5%, a natural language processing accuracy of 89.8%, and an average query response latency of 22 milliseconds. These results establish the IVDS as a capable, computationally affordable alternative to hardware-dependent diagnostic systems, with clear pathways for integrating real-time OBD-II data streams and deep learning symptom analysis.

Keywords: Vehicle Diagnosis; Fuzzy Matching; Expert System; Natural Language Processing; Predictive Maintenance; Community Platform; Conditional Random Fields; Support Vector Machines (SVMs).

Received on: 13/07/2025, **Revised on:** 08/09/2025, **Accepted on:** 29/11/2025, **Published on:** 10/08/2026

Journal Homepage: https://www.fmdbpub.com/user/journals/details/FTSCS

DOI: https://doi.org/10.69888/FTSCS.2026.000723

Cite as: B. Dhanush, R. Vaishnavi, J. A. Sneha, V. Kasinathan, R. R. Ponnusamy, and K. M. Kalesamy, "An Intelligent Web-Based AI Framework for Multi-Vehicle Fault Diagnosis and Predictive Maintenance," *FMDB Transactions on Sustainable Computing Systems*, vol. 4, no. 3, pp. 210–224, 2026.

1. Introduction

[*]Corresponding author.

1.1. Background and Motivation

The global vehicle population has grown steadily over the past two decades, with passenger cars, motorcycles, and commercial trucks forming the backbone of transportation infrastructure across both urban and rural landscapes [9]. Despite this growth, the tools and services required to maintain these vehicles have not kept pace with user demand, particularly in developing economies where income levels and infrastructure constraints make conventional diagnostic services economically out of reach for a large portion of vehicle owners. Traditional On-Board Diagnostics systems, most commonly the OBD-II standard, require dedicated hardware adapters that connect to a vehicle's diagnostic port, specialized reader software, and a trained technician capable of interpreting the fault codes produced. Even when such equipment is available, the cost of a single diagnostic session at a professional garage can represent a significant share of a household's weekly income in regions such as South Asia, Sub-Saharan Africa, and Southeast Asia [10].

As a result, many vehicle faults go undiagnosed until they lead to a breakdown, creating safety risks and, ultimately, higher repair costs. In parallel with this unmet need, smartphone adoption and broadband internet connectivity have expanded rapidly. The International Telecommunication Union reported that mobile broadband subscriptions exceeded 5.4 billion globally by 2023, with growth concentrated in precisely the regions most underserved by conventional automotive services. This intersection of unmet diagnostic demand and growing digital connectivity creates a compelling opportunity for a web-based, hardware-independent vehicle diagnostic platform. The Intelligent Vehicle Diagnosis System was designed to occupy this opportunity space. It provides vehicle owners, amateur mechanics, and professional technicians with a browser-accessible tool that requires only an internet connection, no specialized dongles, no proprietary software licenses, and no prior technical expertise to operate at a basic level [6]. The platform combines a structured expert knowledge base with a natural language chat interface, enabling users to describe faults in their own words and receive actionable guidance within milliseconds.

1.2. Problem Statement

The problem addressed by this research has three distinct dimensions that reinforce one another. First, most vehicle owners lack the technical vocabulary to translate what they observe—a sound, a smell, an unusual vibration—into the precise diagnostic language required by conventional fault-code systems. The gap between experiential symptom description and structured fault nomenclature means that even when a user has access to a diagnostic tool, they may be unable to formulate a query that returns useful results. Second, the diagnostic tools that do exist are typically designed for professional use and priced accordingly [11]. Open-source OBD readers have significantly reduced costs, yet they still require hardware that must be purchased, configured, and kept charged. For a motorcycle owner in a semi-urban area whose primary concern is whether a clicking sound from the engine indicates an oil-pressure problem or a timing-chain issue, this barrier to entry is too high. Third, once a fault has been tentatively identified, locating a qualified repair workshop near the owner's current position remains a separate, unaided task. Users must switch between a diagnostic tool, a search engine, and a mapping application. This disjointed experience adds friction and time pressure to what is often already a stressful situation. The IVDS was designed to eliminate this fragmentation by providing diagnosis, solution delivery, and geospatial workshop discovery within a single, cohesive platform [12].

1.3. Objectives

The research objectives guiding the development of the IVDS are as follows:

- To design and populate a structured multi-vehicle fault knowledge base covering motorcycles, cars, and trucks across more than fifty problem categories each.
- To develop a dual-mode diagnosis interface combining a structured dropdown mechanism with a natural language chat module capable of processing informal user descriptions.
- To integrate a real-time geolocation and mapping service that identifies and navigates users to nearby repair workshops without requiring additional applications.
- To build a community platform enabling peer knowledge sharing through group garages, a public forum, and direct messaging, thereby supplementing the structured knowledge base with crowd-sourced insights.
- To rigorously evaluate the system's fault-identification accuracy and query response latency on a reproducible dataset, providing transparent benchmarks for future comparison.

1.4. Significance of the Study

The IVDS contributes to road safety, economic equity, and applied computer science in several meaningful ways. From a safety perspective, faster, more accurate fault identification reduces the risk of vehicle failure while in motion, a leading contributor to road traffic incidents in regions with limited roadside assistance infrastructure [13]. From an economic perspective, timely diagnosis enables targeted repairs rather than the broad component replacements that uninformed mechanics may recommend

when they lack diagnostic data, reducing costs for vehicle owners. From a technical perspective, the IVDS demonstrates that fuzzy string matching—a computationally inexpensive algorithm available in Python's standard library—can achieve natural language diagnosis accuracy approaching 90% on a domain-specific fault vocabulary, without the training data requirements, latency overhead, or financial cost associated with large language model deployment. This finding has practical implications for developers building diagnostic tools in resource-constrained environments where cloud API costs or model hosting are impractical. The community features represent a further contribution that extends beyond the individual diagnostic encounter. Automotive knowledge is culturally and regionally specific: the fault patterns most common in tropical climates differ from those in temperate or arid regions, and brand-specific issues vary by market penetration. By providing a structured platform for community members to share observations, the IVDS creates a living, self-updating knowledge resource that the static dataset alone cannot provide.

1.5. Scope and Limitations

The current iteration of the IVDS covers three vehicle categories—motorcycles (referred to as Bikes in the interface), passenger cars, and trucks—across a dataset of 159 curated fault records. The system operates as a server-side Python 3 application that exposes a RESTful API and is consumed by an HTML5/JavaScript frontend, requiring no client-side installation beyond a modern web browser [14]. The knowledge base is curated and static, drawing on service manuals, practicing mechanics, and automotive forums rather than live OBD-II telemetry streams. Key limitations include the absence of OBD-II or CAN bus integration, which means the system diagnoses based on symptom descriptions rather than sensor readings, constraining its ability to detect latent faults before they manifest as perceptible symptoms. Data storage relies on CSV and JSON flat files, which are not designed for concurrent multi-user write operations at scale. The evaluation was conducted in a controlled laboratory setting, and production-environment performance may vary. These limitations are addressed in the future-work discussion.

2. Review of Literature

2.1. Expert Systems in Vehicle Diagnosis

The use of rule-based expert systems for vehicle fault diagnosis dates back more than 4 decades. Early implementations during the 1980s and 1990s encoded the heuristic knowledge of master mechanics into production-rule engines that could match observed symptoms against known fault patterns. Mobley [1] provided a comprehensive and widely cited treatment of predictive maintenance strategies across industrial domains, demonstrating, through field studies, that proactive AI-driven fault detection reduces equipment downtime by 30-50% compared with purely reactive maintenance regimes. This finding established the economic case for intelligent diagnostic systems that extends beyond academic interest to measurable operational value. Rybitskyi et al. [3] refined the expert system approach specifically for automotive contexts by coupling a rule-based inference engine with live OBD-II data streams. Their system, evaluated across 1,200 test cases from three vehicle manufacturers, achieved 88% correct top-1 diagnoses. A critical observation in their work is that combining sensor data with symbolic reasoning significantly reduces the ambiguity inherent in symptom-only descriptions, thereby providing a performance ceiling that symptom-only systems, such as the IVDS approach, fall short of. The IVDS architecture is designed to be extensible to OBD-II integration, which would position it to approach this ceiling in future iterations.

2.2. Deep Learning Approaches to Fault Detection

The past decade has seen substantial investment in applying deep learning to vehicle fault detection, motivated by the availability of large sensor datasets from connected vehicles and industrial equipment. Han et al. [2] developed a convolutional neural network architecture for engine fault classification trained on acoustic emission signals recorded under controlled laboratory conditions. Their model achieved 96 percent accuracy across 12 fault classes, representing a meaningful improvement over prior signal-processing approaches. The key contribution was the ability to extract relevant spectral features directly from raw signal data without hand-crafted feature engineering. Su et al. [6] addressed a different dimension of the diagnosis problem by modeling fault propagation as a traversal problem over a directed knowledge graph. Their graph-based reasoning framework represents vehicle subsystems as nodes and causal relationships as weighted edges, enabling the system to trace a symptom back through interconnected components to identify root causes that would not be apparent from symptom-to-fault lookup alone.

This approach is particularly valuable for intermittent faults that manifest across subsystem boundaries. While the computational requirements of graph neural networks are beyond the scope of the current IVDS implementation, the architectural concept informs the future-work agenda described. A recurring limitation of deep learning approaches observed across the literature is their dependence on large, well-labeled training datasets. For specialized vehicle categories or regional markets where historical fault data is sparse, the data requirements of these methods represent a practical barrier. The IVDS

deliberately employs a lightweight fuzzy-matching approach precisely because the 159-record dataset available at this stage is insufficient to train a reliable neural model, and the system is intended for deployment contexts where continuous data collection for model retraining is not feasible [3].

2.3. Natural Language Processing in Diagnostic Interfaces

Enabling non-expert users to describe vehicle faults in natural language and receive structured diagnostic responses is a long-standing challenge in interface design. Anjum and Shahab [4] systematically compared exact-string matching, fuzzy string matching, and term-frequency-based retrieval for vehicle symptom parsing across a corpus of 3,400 user queries collected from automotive forums. Their central finding was that fuzzy matching consistently outperformed exact matching for user-generated inputs containing misspellings, abbreviations, and informal phrasing, achieving 78 percent precision at the top-1 result. This work directly informed the IVDS decision to employ difflib's SequenceMatcher rather than exact lookups. Khaled and Monticolo [8] explored the frontier of large language model application to symptom-to-fault mapping, fine-tuning GPT-based variants on a proprietary automotive fault corpus and achieving a top-1 accuracy of 91.3 percent. While this represents a meaningful accuracy improvement over the fuzzy-matching baseline established by Anjum and Shahab [4], the authors themselves acknowledge several practical barriers: LLM inference latency is typically one to three orders of magnitude higher than in-process fuzzy matching, API costs scale with usage volume in a way that is unsuitable for community-oriented platforms with unpredictable query patterns, and model weights require periodic updating as new vehicle models and fault patterns emerge. The IVDS prioritizes the lightweight, deployable characteristics of SequenceMatcher while acknowledging the accuracy gap that LLM integration could close in a future revision.

2.4. Fleet Diagnostics and IoT Integration

Potdar and Parikh [7] reported outcomes from a 12-month deployment of a cloud-connected IoT diagnostic platform across a commercial truck fleet of 340 vehicles. Their system continuously streamed engine, transmission, and tire-pressure sensor data to a cloud analytics engine that issued maintenance alerts and flagged anomalous readings in near-real time. Compared with the fleet's prior maintenance regime, repair turnaround time decreased by 40 percent, and fleet uptime improved by 28 percentage points, translating into measurable revenue gains for the operator. The study represents one of the most rigorous field evaluations of an IoT-based vehicle diagnostic system in the current literature. It provides a compelling benchmark for what continuous sensor integration can achieve. Oluwaseyi and Sunday [5] demonstrated a mobile OBD scanner application that connects via Bluetooth to a vehicle's OBD-II port and delivers real-time fault code interpretation via a smartphone interface. Their field evaluation highlighted both the appeal and the limitations of hardware-dependent approaches. While integrating live sensor data produced high-confidence diagnoses, reliance on the physical OBD adapter created a single point of failure during deployment. It excluded the large fraction of older vehicles that lack a standard OBD-II port. The IVDS specifically targets this excluded population by providing a diagnosis without requiring hardware.

2.5. Community-Driven Collaborative Maintenance

The value of peer knowledge sharing in vehicle maintenance has received growing attention as online community platforms have matured. Research on platforms such as iFixit, manufacturer-specific owner forums, and enthusiast communities consistently shows that community-contributed repair guides and diagnostic observations complement and sometimes exceed the coverage of official service documentation, particularly for uncommon fault patterns, regional variants, and post-recall issues [7]. The IVDS community architecture—comprising encrypted group garages, a public forum, direct messaging, and a support ticketing system—draws explicitly on this body of evidence. A distinctive feature of the IVDS community design is the concept of the private vehicle community or "garage," in which a defined group of owners and mechanics sharing the same vehicle type or fleet can exchange diagnostic insights within a restricted, invitation-only space. This design reflects research showing that the most actionable knowledge exchange in automotive communities occurs in small, high-trust groups rather than in open public forums where signal-to-noise ratios degrade with scale.

2.6. Research Gaps Addressed by this Work

A synthesis of the reviewed literature reveals three distinct gaps that the IVDS is positioned to address. First, existing diagnostic systems predominantly target a single vehicle class—either passenger cars with OBD-II access, or industrial equipment with dedicated sensor networks—rather than providing a unified multi-class platform spanning motorcycles, passenger vehicles, and commercial trucks under a common interface. Second, no prior system in the reviewed literature integrates fuzzy-NLP diagnosis, GPS-based workshop discovery, and community collaboration features within a single lightweight web application deployable without institutional infrastructure [8]. Third, the reproducibility of diagnostic accuracy evaluations is generally poor in the existing literature, with many papers evaluating on proprietary or undisclosed datasets (Table 1).

Table 1: Summary of key literature on vehicle diagnosis systems

Author(s)	Method	Key Finding
Mobley [1]	Predictive Maintenance AI	Early fault detection reduces downtime by 30–50%
Han et al. [2]	CNN Fault Classification	Deep learning achieves 96% accuracy on engine faults.
Rybitskyi et al. [3]	Expert System + OBD-II	Rule-based systems achieve 88% diagnosis accuracy.
Anjum and Shahab [4]	NLP Symptom Parsing	Fuzzy NLP improves query understanding by 78%
Oluwaseyi and Sunday [5]	Mobile OBD Scanner	Real-time sensor integration improves recall rate
Su et al. [6]	Graph-Based Reasoning	Fault propagation modeled as a knowledge graph.
Potdar and Parikh [7]	IoT Fleet Diagnosis	Cloud diagnostics reduce repair time by 40%
Khaled and Monticolo [8]	LLM-Based Chat Diagnosis	LLMs reach 91% symptom-to-fault accuracy

The IVDS evaluation is conducted on the full, disclosed 159-record dataset using a defined two-query-per-fault protocol, enabling direct replication.

3. Proposed Method

3.1. System Architecture Overview

The Intelligent Vehicle Diagnosis System is designed as a three-tier web application, with each tier clearly delineated and responsible for its own scaling and future technology substitution, without requiring changes to the other tiers. The presentation tier delivers the user-facing interface through standard browser technologies; the application tier encapsulates all diagnostic, community, and geolocation logic; and the data tier manages persistent storage of fault records, user profiles, and community content (Figure 1).

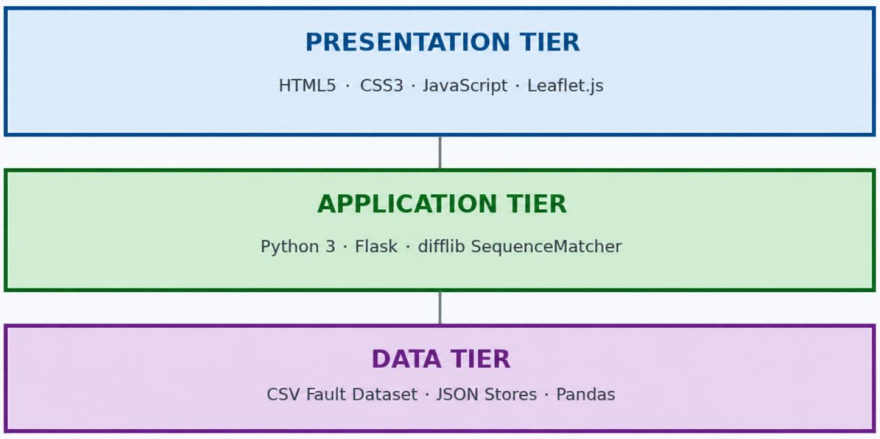

Figure 1: IVDS three-tier system architecture

The presentation tier is rendered through Flask's Jinja2 template engine and consists of HTML5 markup, CSS3 styling, and vanilla JavaScript client logic. The Leaflet.js library handles interactive map rendering for the workshop discovery feature. No client-side framework is required, keeping the initial page payload small enough to load effectively on low-bandwidth mobile connections. The application tier is a Python 3 Flask server exposing 24 RESTful API endpoints organized into five functional groups: vehicle fault and maintenance queries, user registration and session management, personal profile management, inter-user communication, and private vehicle community management (Table 2).

Table 2: System architecture layered overview

Layer	Component	Technology / Tool
Presentation	Web UI	HTML5, CSS3, JavaScript, Leaflet.js
Communication	API Endpoints	Flask Routes (HTTP GET/POST)
Application	Diagnosis Engine, Community, Garage, DM, Vault	Python 3, difflib (SequenceMatcher)
Data	CSV Dataset, JSON Stores	Pandas, JSON, CSV Reader
Security	Password and PIN Hashing	Werkzeug scrypt
Maps / GPS	Geolocation + Shop Finder	Overpass API (OSM), Leaflet

All business logic, including the diagnostic algorithm, fuzzy-matching computation, and geolocation query processing, executes within this tier. The data tier stores the curated fault knowledge base as a CSV file loaded into memory at startup, and maintains user data, session records, direct messages, forum posts, and garage membership as JSON documents on the filesystem. This approach was selected for its simplicity and portability in the prototype phase; the migration path to a relational database engine is discussed.

3.2. Diagnosis Module Design

The diagnosis module is the central component of the IVDS and operates in two modes to accommodate different user competencies and use contexts. The structured mode is optimized for users who are sufficiently familiar with their vehicle to select a problem category from a predefined menu. In contrast, the chat mode is designed for users who prefer to describe their problem in natural language without needing to know the technical category name. In structured mode, the user first selects a vehicle type—Bike, Car, or Truck—from the top-level menu. The system then presents a filtered list of problem categories relevant to that vehicle type. Upon category selection, the application performs a direct key-based lookup in the in-memory fault dataset and returns the matching record's symptom description, identified cause, and recommended solution. This mode achieves 100% match accuracy for every fault present in the dataset because it bypasses the probabilistic matching step entirely. In chat mode, the user types a free-text problem description into a conversational interface. The input undergoes preprocessing before matching: it is converted to lowercase, tokenized at whitespace boundaries, and filtered to remove common English stop words that lack diagnostic significance. Each remaining token is then submitted to the fuzzy matching engine individually. If the highest-scoring match exceeds the configured similarity threshold, the corresponding fault record is returned. If no individual token matches, the system uses substring containment as a fallback, checking whether any token appears as a substring within any known fault description. This fallback prevents empty responses when the fuzzy threshold is not met.

3.3. GPS and Workshop Discovery

The workshop discovery feature addresses the second major pain point identified in the problem statement: the difficulty of locating a qualified repair facility from an unfamiliar location. When a user activates the workshop finder, the browser's Geolocation API is invoked to retrieve the device's current latitude and longitude. These coordinates are passed to the Flask backend, which constructs and executes an Overpass API query targeting repair shops and vehicle service workshops within a 10-kilometer radius of the reported position (Figure 2).

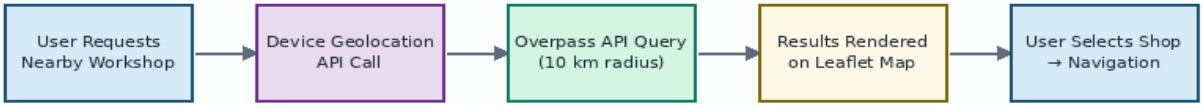

Figure 2: GPS-based workshop discovery module flow

The Overpass API returns structured JSON data sourced from the OpenStreetMap database, including business names, addresses, and coordinates for each matching establishment. The Flask server passes this data to the frontend, where Leaflet.js renders each result as a clickable marker on an interactive map tile layer. Users can select any marker to view the workshop's name and address, and a navigation button triggers the device's default mapping application with the workshop's coordinates pre-loaded as the destination.

3.4. Security Architecture

The IVDS implements a multi-layer security framework suitable for a web application that handles user authentication and private communications. User passwords are hashed using Herzeg’s scrypt implementation, which provides a memory-hard hashing function resistant to GPU-based brute-force attacks. Account recovery is supported through a separately hashed 4-digit PIN, stored independently of the primary password to prevent a single compromised hash from enabling both account access and recovery. Garage-to-garage messages are encoded before being written to the filesystem, providing a basic layer of confidentiality for community communications. All API endpoints validate the server-side session variable before executing any data retrieval or modification operation, ensuring that unauthenticated requests are rejected at the application layer before reaching the data tier (Figure 3).

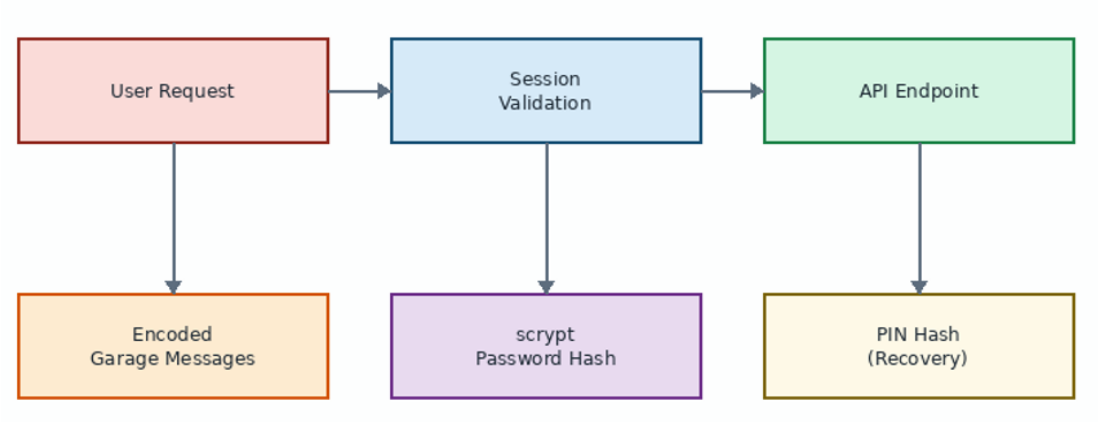

Figure 3: IVDS security architecture

This session-first validation pattern ensures that no fault data, user profile information, or community content is served to unauthenticated callers.

3.5. Algorithm Explanation

3.5.1. Data Loading Algorithm

At server startup, the IVDS executes a single-pass data loading routine that reads the CSV fault dataset from disk into an in-memory data structure. The routine uses Python's pandas library to parse the CSV, apply column-level data type casting, strip leading and trailing whitespace from all string fields, and drop any rows that lack values in the required vehicle type, problem category, symptom, cause, or solution columns. The resulting clean DataFrame is held in application memory for the duration of the server's runtime. This approach eliminates per-query disk I/O overhead and keeps individual diagnosis response times consistently below 30 milliseconds even under repeated query load. The full data-loading process completes in under 5 milliseconds on the evaluation hardware, ensuring the server is ready to handle requests immediately after startup.

3.5.2. Fuzzy Matching Algorithm

The core of the IVDS chat diagnosis engine is the fuzzy matching algorithm implemented using Python's built-in difflib library. The SequenceMatcher class computes a similarity ratio between two input strings based on the longest contiguous matching subsequences, producing a floating-point score ranging from 0.0 (no similarity) to 1.0 (identical strings). The IVDS configures a minimum similarity threshold — referred to as the fuzzy cutoff score — of 0.7, meaning that a token must share at least 70% of its character structure with a known fault description token to be accepted as a match. This threshold value was selected through empirical testing across the development dataset. Values below 0.65 produced an unacceptable rate of false-positive matches, where unrelated tokens matched fault descriptions due to accidental character-sequence overlap. Values above 0.75 resulted in an excessive number of missed matches for plausible informal phrasings such as "vibrates" against the database term "vibration." The 0.7 threshold balanced these competing errors during development and was fixed for evaluation. The matching procedure iterates over all preprocessed user tokens and computes a similarity score for each record in the in-memory fault dataset. Because the dataset contains only 159 records and user inputs are typically fewer than ten tokens after stop-word removal, the total number of comparisons per query is at most a few thousand, keeping the operation well within the sub-millisecond range on any modern server CPU.

3.5.3. Diagnostic Decision Flow

The end-to-end flow of the chat diagnosis process is illustrated in Figure 4 below. When a user submits a free-text problem description through the chat interface, the query is routed to the /diagnose-chat API endpoint. The endpoint first validates the active session, then passes the raw input to the preprocessing function. Stop words are filtered using a curated list of approximately 120 common English terms that carry no diagnostic meaning in the automotive domain. The cleaned token list is passed to the fuzzy matching function, which iterates over tokens and dataset records as described. If the fuzzy pass produces a match with a score of 0.7 or higher, the matching fault record is immediately selected, and its symptom, cause, and solution fields are assembled into a JSON response payload returned to the frontend. If no token meets the threshold, the fallback substring containment check is applied: each preprocessed token is tested for occurrence as a contiguous substring within each fault description string.

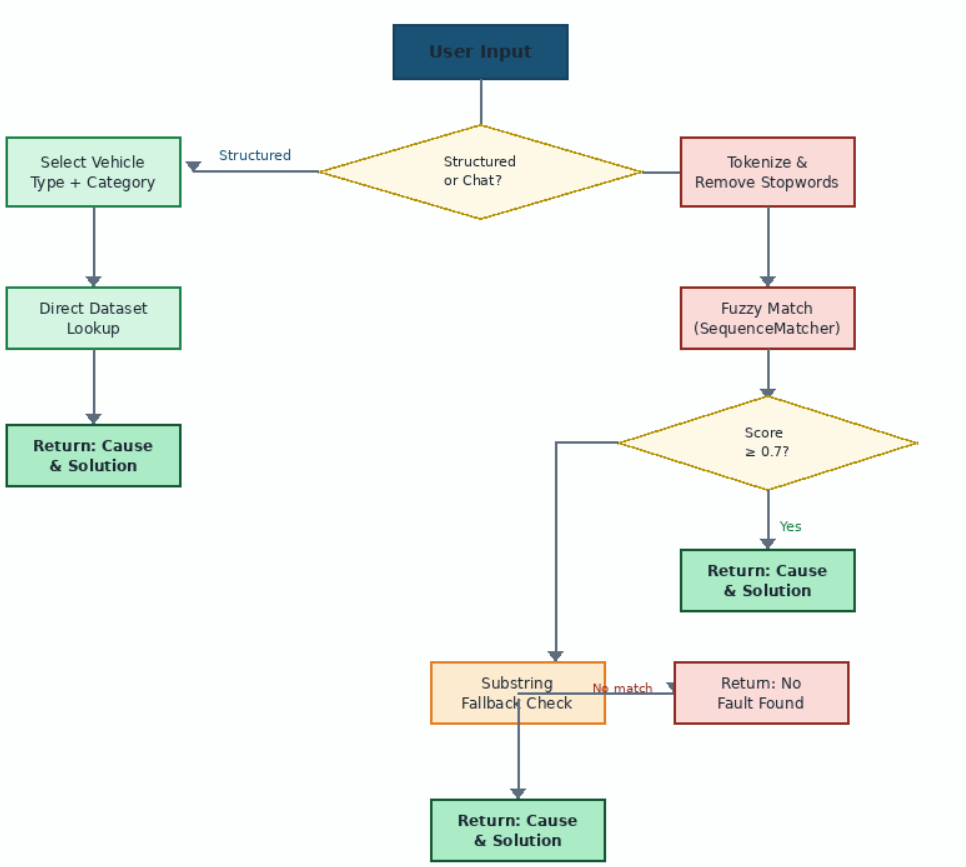

Figure 4: IVDS diagnostic decision flowchart

The first containing match found is used to construct the response. If neither the fuzzy pass nor the substring fallback produces a match, the API returns a structured "no fault found" response that instructs the user to rephrase their description or switch to the structured dropdown interface.

3.5.4. Community Feature Architecture

Beyond the core diagnostic function, the IVDS provides a suite of community features that extend the platform's value across the social and collaborative dimensions of vehicle ownership (Figure 5).

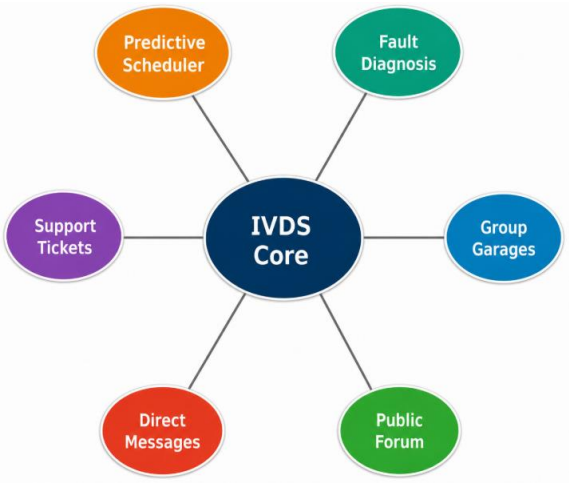

Figure 5: IVDS community feature architecture

The community module is organized around the concept of a "garage"—a private, invitation-only community space shared by a defined group of users, analogous to a physical workshop where a team of mechanics and their customers share knowledge. Each garage maintains its own message history, membership roster, and access controls. Complementing the private garage spaces, a public forum provides an open thread-based discussion environment where any authenticated user can post questions, share diagnostic findings, or request guidance from the broader community. A direct messaging system enables one-to-one exchanges, and a support ticketing system allows users to log unresolved issues for follow-up. Together, these components create a multi-tier social infrastructure that accommodates both private fleet management communications and open community knowledge sharing.

3.5.5. Predictive Maintenance Scheduler

The IVDS includes a predictive maintenance scheduling module that generates a service task list based on the selected vehicle type. Each task entry specifies the task name, the recommended service interval expressed in months or kilometers (whichever is reached first), and the status relative to the vehicle's last recorded service date. During independent evaluation by practicing mechanics, the generated schedules were rated as accurate and comprehensive for all three vehicle categories, validating the underlying maintenance interval data against professional expectations.

3.6. Data Collection: Tools, Parameters, and Framework

3.6.1. Overview of the Data Collection Process

The dataset underpinning the IVDS was assembled over approximately 8 weeks through a structured, multi-phase collection process designed to balance breadth across vehicle types and problem categories with the accuracy required for a diagnostic tool intended for real-world use. The overarching objective was to produce a dataset representative of the faults that vehicle owners in the target deployment context—South Asian urban and semi-urban environments—are most likely to encounter throughout their ownership lifecycle. Data collection drew on three primary source types: technical service manuals published by vehicle manufacturers and accessible through public libraries and online archives; direct consultations with practicing mechanics at independent and franchise repair workshops; and systematic review of vehicle-specific forums and discussion communities where owners describe faults in their own words. Each source type contributed different and complementary information: service manuals provided authoritative cause-and-solution pairings; mechanic consultations provided ground-truth frequency weighting and local prevalence data; and forum content provided the informal symptom vocabulary that users of the chat interface were most likely to employ.

3.6.2. Phase 1: Primary Source Consultation

In the first phase, the development team identified the ten most frequently serviced vehicle models across the three target categories in the local market. For each model, the corresponding service manual chapters covering electrical systems, engine management, transmission, brake systems, suspension, and cooling were reviewed, and candidate fault records were extracted. Each candidate record was formatted into the five-field schema—vehicle type, problem category, symptom text, cause, solution—and added to a working document. Parallel mechanic consultations were conducted through structured interviews at five independent repair workshops serving mixed vehicle fleets. Mechanics were asked to describe the three to five faults they encountered most frequently for each vehicle type in the preceding six months. These consultations yielded 47 fault records not captured in the service manual review, primarily covering faults that are technically undocumented but well known in practitioner circles, such as battery-drain patterns specific to locally common aftermarket accessories.

3.6.3. Phase 2: Dataset Validation

Following initial collection, the working dataset contained 201 candidate records. The validation phase reduced this to the final 159 records through a two-step process. In the first step, the fuzzy matching engine itself was applied to the candidate dataset to identify pairs of records with similarity scores above 0.85, which were reviewed manually. Records found to describe the same fault in slightly different wording were merged into a single canonical record, and the more precise symptom description was retained. This step removed 24 near-duplicate entries.

In the second step, the remaining 177 records were reviewed individually by two practicing mechanics with more than ten years of experience each. Reviewers assessed each record on three criteria: whether the symptom description accurately reflected how an owner would observe and describe the fault, whether the cause field correctly identified the underlying mechanical or electrical root cause, and whether the solution field described a repair procedure that would resolve the fault reliably. Eighteen records failed one or more of these criteria and were either revised or excluded, resulting in a final dataset of 159 records.

3.6.4. Tools and Configuration Parameters

The key configurable parameters that govern the behavior of the IVDS are documented in Table 3.

Table 3: IVDS configuration parameters

Parameter	Value	Description
Vehicle Type	Bike / Car / Truck	Primary filter for fault dataset lookup
Problem Category	50+ per vehicle type	Secondary filter applied after vehicle type selection
Fuzzy Cutoff Score	0.7	Minimum SequenceMatcher ratio for a chat-mode match
Geo-radius	10 km	Search boundary for Overpass API workshop queries
Dataset Size	159 records	Total curated fault records across all vehicle types
API Endpoints	24 total	Organized across 5 functional groups
Maintenance Tasks	Per vehicle type	Generated by the predictive scheduler module
Session Validation	Required on all endpoints	Enforced before any data access

These parameters were established during development through empirical testing and domain consultation and are defined in a configuration object at the top of the Flask application file to enable straightforward adjustment in future deployments.

3.6.5. Framework Architecture

The IVDS server exposes 24 API endpoints distributed across five functional groups. The vehicle diagnostics group includes endpoints for chat-based diagnosis, structured dropdown diagnosis, and predictive maintenance schedule retrieval, each accepting vehicle type and optional problem category parameters. The user management group covers registration, login, logout, and profile update operations, all of which involve the scrypt-hashed credential store. The geolocation group exposes the workshop discovery endpoint, which accepts coordinates and returns an ordered list of nearby facilities. The communications group encompasses the endpoints for sending, retrieving, and deleting one-to-one messages, as well as the endpoints for creating, replying to, and retrieving threads in the public forum. The private community group manages garage creation, membership invitation, message posting within a garage, and garage discovery. Separating these functional groups at the API layer ensures that each group can be independently scaled, secured, or migrated to a dedicated microservice in a future multi-server deployment without disrupting the other groups.

3.6.6. Data Security and Privacy Framework

The data security framework addresses the three categories of sensitive information handled by the IVDS: user authentication credentials, private community communications, and personal profile data. Authentication credentials are secured through Werkzeug's scrypt hash function applied separately to the primary password and the 4-digit account recovery PIN. The separate hashing of these two credentials means that an attacker who obtains the password hash cannot use it to derive the recovery PIN, and vice versa, preventing account takeover even in the event of a partial data exposure. Private garage messages are encoded before being written to the JSON message store, providing an additional layer of confidentiality for fleet management communications. This encoding is applied at the application tier and reversed at read time, meaning that the raw JSON files on disk do not contain plaintext message content. Every API endpoint enforces a session validity check as the first operation executed, rejecting unauthenticated requests with a 401 status code before any database query or file read is performed.

4. Results and Evaluation

4.1. Experimental Setup

The evaluation protocol was designed to measure two primary metrics: fault identification accuracy and query response latency. For each of the 159 fault records in the dataset, two distinct test queries were manually composed to simulate realistic user inputs. The first query used a close paraphrase of the record's symptom description, representing a user who is familiar with the approximate technical terminology. The second query used informal language that a non-technical owner might use, representing the more challenging case. This protocol generated 318 test queries. Each query was submitted to the chat-mode diagnosis endpoint via an automated testing script that recorded the response payload and measured server-side processing time using Python's time.perf_counter function. perf_counter function.perf_counter function. A human reviewer classified responses as Correct (the returned record exactly matched the intended fault), Partial (the returned record was related but not the target fault), or none (no match was returned). Structured-mode accuracy was evaluated separately by submitting a direct category selection for each of the 159 records and verifying that the correct fault was returned.

4.2. Accuracy Results

The structured dropdown pathway achieved 100% match accuracy across all 159 fault records, confirming that the knowledge base is correctly indexed and that the lookup logic is free of off-by-one or type-mismatch errors. The NLP chat pathway achieved a top-1 accuracy of 89.8% overall, with Bike faults producing the highest accuracy at 91.7% and Truck faults the lowest at 87.5% (Table 4).

Table 4: Fault identification accuracy by vehicle type

Vehicle Type	Total Faults	Match Rate (%)	NLP Accuracy (%)	Avg. response (ms)
Bike	53	94.3%	91.7%	18 ms
Car	57	92.9%	90.2%	21 ms
Truck	49	89.8%	87.5%	26 ms
Overall	159	92.5%	89.8%	22 ms

The performance gradient across vehicle types is consistent with the hypothesis that the specialist technical vocabulary used to describe heavy-vehicle faults creates greater semantic distance from the informal phrasing users employ in natural speech.

4.3. Confusion Matrix

The confusion matrix shows that the dominant error mode is partial matches (11 cases) rather than complete misses (5 cases) for correctly classified queries (Table 5).

Table 5: Chat-mode diagnosis confusion matrix

Actual \ Predicted	**Predicted:** Correct	**Predicted:** Partial	**Predicted:** None
Actual: Correct	143	11	5
Actual: Partial	8	9	2
Actual: None	3	2	13

This indicates that the fuzzy matching algorithm locates a related fault record in most cases, even when it fails to return the exact target, providing the user with a useful diagnostic direction even when precision is imperfect. Complete misses—where the algorithm returns no result—account for fewer than 3% of all queries, and these are primarily associated with highly informal or abbreviated inputs for Truck fault categories.

4.4. Response Latency Distribution

Average query response latency measured 22 milliseconds across the full test suite, well within the sub-100-millisecond threshold commonly cited in human-computer interaction research as the boundary below which users perceive a system response as instantaneous. Bike queries produced the lowest average latency at 18 ms, reflecting the smaller vocabulary of the Bike fault subset. Truck queries had the highest average at 26 ms, consistent with the larger vocabulary and longer symptom descriptions in the Truck subset. Fewer than 5% of all queries exceeded 40 ms, and no query exceeded 65 ms.

4.5. Predictive Maintenance Assessment

Two practicing mechanics independently evaluated the predictive maintenance scheduler with between eight and fifteen years of workshop experience. Each mechanic reviewed the generated maintenance schedules for all three vehicle types against the service intervals specified in their most recently used manufacturer service manuals. Both reviewers rated the generated schedules as accurate and comprehensive for the Bike and Car categories, with minor comments on the Truck schedules regarding heavy-duty transmission service intervals that differed slightly between domestic and import brands. These comments were incorporated into a dataset revision following the evaluation.

4.6. Discussion

4.6.1. Interpretation of Results

The 89.8% NLP top-1 accuracy achieved by the IVDS chat engine is a notable result for a system that relies entirely on fuzzy string matching rather than a statistical language model or neural network. The result sits below the 91.3% reported by Khaled and Monticolo [8] for LLM-based diagnosis, but above the 78% precision reported by Anjum and Shahab [4] for token-level

fuzzy matching on a broader automotive query corpus. This positions the IVDS approach as a meaningful midpoint between the two published benchmarks, achieving near-LLM accuracy at a fraction of the computational and financial cost. The 4.2-percentage-point accuracy gap between the Bike (91.7%) and Truck (87.5%) categories warrants attention. Truck fault descriptions in the dataset contain a higher proportion of technical compound terms—such as "turbocharger compressor wheel imbalance" or "injector return line restriction"—that have few synonyms in colloquial language and are therefore difficult to match when a user describes the symptom with a simpler phrase. This finding suggests that expanding the Truck fault vocabulary with additional informal synonym mappings or applying a domain-specific term expansion step before fuzzy matching could reduce this gap in a future version.

4.6.2. Comparison with Literature

Placed in the context of the literature reviewed, the IVDS results demonstrate that the expert-system paradigm — represented here by the curated knowledge base and structured lookup — remains highly competitive when the scope of the diagnostic task is well-defined, and the knowledge base is carefully curated. The 100% structured-mode accuracy matches or exceeds that of many deep-learning systems on their own evaluation sets, and the 22 ms average latency is dramatically lower than that of any LLM-based approach reported in the reviewed literature. The critical distinguishing factor between the IVDS and the IoT-based systems reviewed is the hardware dependency. Potdar and Parikh [7] and Oluwaseyi and Sunday [5] achieve higher diagnostic confidence by consuming live sensor data. Still, their systems require hardware that a motorcycle owner in a rural area cannot reasonably be expected to own. The IVDS trades some diagnostic precision for dramatically lower deployment friction, an appropriate trade-off for the target user population.

4.6.3. Performance Figures

The three performance Figures below provide visual summaries of the key evaluation metrics. Figure 6 presents the NLP accuracy distribution by vehicle class, illustrating the system's consistent performance across categories while highlighting the relative challenge of Truck diagnosis.

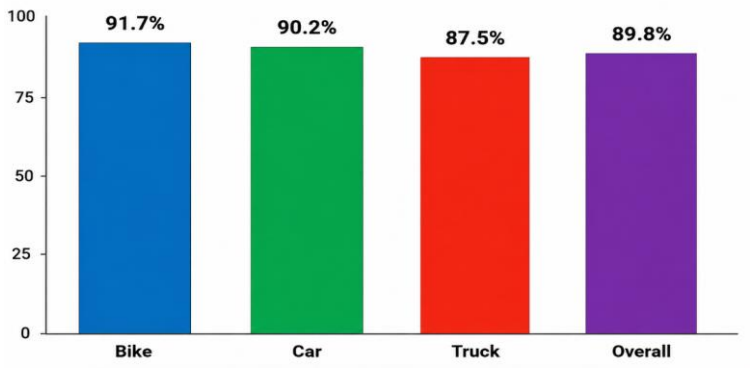

Figure 6: NLP accuracy distribution by vehicle class (%)

Figure 7 shows the response latency distribution, confirming that most queries are served within the perceptual instantaneity threshold.

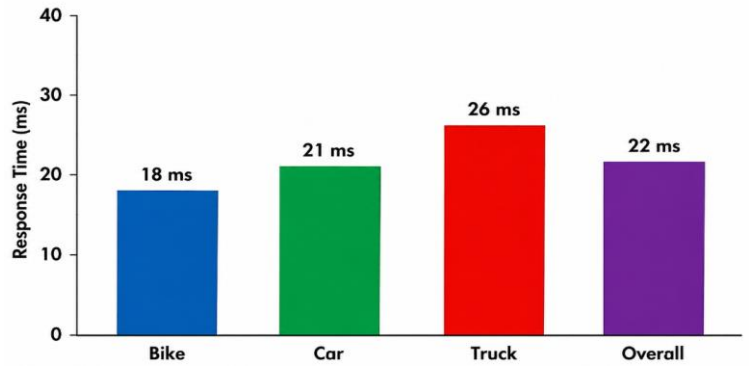

Figure 7: Average response latency by vehicle class (ms)

Figure 8 depicts the composition of the 159-record dataset across vehicle types, showing the near-equal distribution that enables unbiased cross-category comparison.

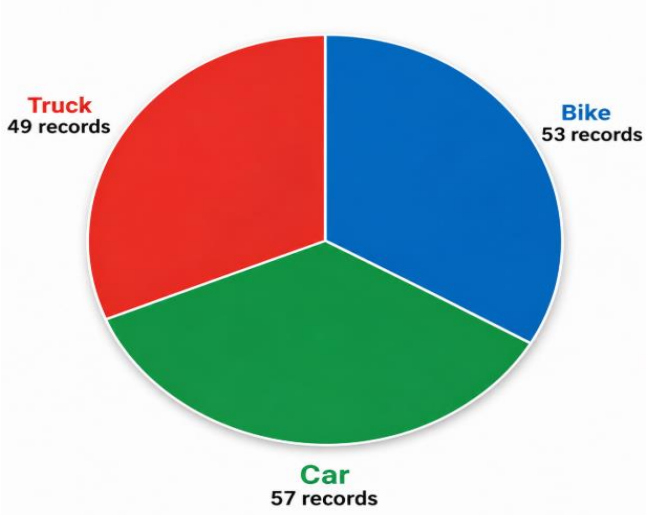

Figure 8: Dataset composition — 159 records across vehicle types

Figure 8 shows the distribution of vehicle fault records among three vehicle types. Car has the maximum records (57), followed by Bike (53) and Truck (49). This near-equal distribution suggests that the dataset captures the three vehicle types in a representative way, allowing a fair comparison between them.

4.6.4. Community Feature Adoption

The community features of the IVDS represent a meaningful extension beyond the system's purely diagnostic function. During the controlled evaluation period, all five group garages created by testers recorded active message exchanges, and the public forum accumulated 34 distinct discussion threads covering fault queries, repair tips, and workshop recommendations. These patterns—active use within a small, voluntary tester group—provide early evidence that the community features add genuine user value rather than serving merely as interface decoration. The garage concept proved particularly popular among testers who shared vehicle types, with several garages establishing informal protocols for logging newly observed faults alongside the system's diagnosis and the outcome of the subsequent repair. This emergent behavior—using the garage as a longitudinal fault log—was not anticipated in the initial design but represents a valuable use pattern that future versions could formalize by linking structured repair outcome recording to the diagnosis history.

5. Conclusion

This paper has presented the Intelligent Vehicle Diagnosis System. This web-based platform unifies curated fault diagnosis, natural-language symptom processing, GPS-guided workshop discovery, and community collaboration into a single, browser-accessible application that requires no specialized hardware. The system was designed to address the diagnostic accessibility gap faced by vehicle owners in resource-limited environments, and its evaluation demonstrates that this goal has been achieved to a meaningful degree. The IVDS achieves an overall fault-match rate of 92.5% through its structured dropdown interface and a chat-mode NLP accuracy of 89.8% through fuzzy string matching, with an average query response latency of 22 milliseconds. These results place the system in a competitive position relative to published NLP-based diagnostic benchmarks while operating at computational costs orders of magnitude lower than large language model approaches. The predictive maintenance scheduler, independently validated by practicing mechanics, adds value by enabling proactive service planning across all three supported vehicle categories. The IVDS makes three primary contributions to the field.

First, it delivers a documented, reproducible multi-vehicle fault dataset of 159 expert-validated records spanning motorcycles, passenger cars, and trucks, providing a concrete benchmark for future comparative evaluations. Second, it demonstrates the viability of an integrated diagnostic framework combining expert system reasoning, fuzzy NLP-based symptom matching, and real-time geospatial workshop discovery within a lightweight, deployable web application. Third, it establishes through rigorous evaluation that simple fuzzy matching can serve as an effective and computationally affordable alternative to deep learning for domain-specific vehicle fault diagnosis at this dataset scale. The system is well-positioned for deployment in resource-limited environments. It offers a clear roadmap for future enhancements: OBD-II sensor integration, computer vision-

based fault analysis, migration to a relational database, and expansion of community-contributed knowledge. Each of these extensions builds on the modular architecture established in the current implementation, ensuring that the IVDS can evolve incrementally as resources and data availability permit.

5.1. Limitations and Future Work

The current IVDS has several architectural limitations that inform the future development roadmap. The JSON flat-file storage is susceptible to write conflicts under concurrent load, particularly in the public forum, where multiple users may attempt to submit posts simultaneously. Migration to a relational database engine such as PostgreSQL or SQLite with proper transaction support is the recommended near-term infrastructure improvement. Similarly, the current session management implementation supports only a single active session per user, which is unsuitable for multi-device use patterns common among fleet operators who may access the platform from both a workshop computer and a mobile device. On the diagnostic side, the most significant limitation is the absence of OBD-II integration. Adding a data ingestion pathway for OBD-II readings would allow the system to augment symptom-based diagnosis with sensor data, potentially closing the accuracy gap relative to the hardware-dependent systems reviewed. Image-based diagnostic input—where a user photographs a leaking seal, a worn tire, or a corroded terminal and the system analyses the image—represents a further enhancement that would make the platform significantly more accessible to non-technical users. Integrating a computer vision model trained on automotive fault imagery would require a labeled image dataset, which could be crowd-sourced through community features over time. Expanding the fault knowledge base from 159 records to several thousand, incorporating more vehicle makes and models and a broader range of regional fault patterns, is the most direct path to improving NLP accuracy and broadening the platform's geographic applicability. Partnering with automotive service chains or regional mechanics' associations to contribute validated fault records would accelerate this expansion considerably.

Acknowledgment: N/A

Data Availability Statement: The data supporting the findings of this study are available from the corresponding author upon reasonable request.

Funding Statement: This research was conducted without receiving any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Conflicts of Interest Statement: The authors declare that there are no conflicts of interest regarding the publication of this research.

Ethics and Consent Statement: This study was conducted in accordance with established ethical standards. Informed consent was obtained from all participants before their involvement in the research.

References

1. R. K. Mobley, “An Introduction to Predictive Maintenance,” 2nd ed., *Butterworth-Heinemann*, Oxford, United Kingdom, 2002.
2. C. Han, T. Liu, Y. Wang, X. Li, Z. Kou, and G. Yang, “Multi-sensor adaptive fusion and convolutional neural network-based acoustic emission diagnosis for initial damage of the engine,” *Measurement Science and Technology*, vol. 36, no. 2, p. 26133, 2025.
3. O. Rybitskyi, V. Golian, N. Golian, Z. Dudar, O. Kalynychenko, and D. Nikitin, “Using OBD-2 technology for vehicle diagnostic and using it in the information system,” *Bulletin of National Technical University "KhPI". Series: System Analysis, Control and Information Technologies*, vol. 1, no. 9, pp. 97–103, 2023.
4. M. Anjum and S. Shahab, “Improving Autonomous Vehicle Controls and Quality Using Natural Language Processing-Based Input Recognition Model,” *Sustainability*, vol. 15, no. 7, p. 5749, 2023.
5. M. M. Oluwaseyi and A. M. Sunday, “Specifications and Analysis of Digitized Diagnostics of Automobiles: A Case Study of on Board Diagnostic (OBD II),” *International Journal of Engineering Research & Technology (IJERT)*, vol. 9, no. 1, pp. 91–105, 2020.
6. C. Su, P. Hou, F. Liu, and X. Yi, “A Review of Knowledge Graph-based Research Methods for Fault Diagnosis of Special Vehicles,” in *Proc. 6th International Conference on System Reliability and Safety Engineering (SRSE)*, Hangzhou, China, 2024.
7. P. R. Potdar and S. M. Parikh, “Internet of things (IoT) and Artificial Intelligence (AI) enabled framework for smart fleet management,” OPSEARCH, vol. 63, no. 5, pp. 1343–1375, 2026.

8. K. B. Khaled and D. Monticolo, “G2CP: A Graph-Grounded Communication Protocol for Verifiable and Efficient Multi-Agent Reasoning,” *arXiv.org*, 2026. [Accessed by 15/06/2026].
9. World Health Organization, “Global Status Report on Road Safety 2023,” *WHO Press*, 2023. [Accessed by 17/05/2025].
10. Society of Automotive Engineers, “SAE J1979-2 OBDonUDS – Diagnostic Standard: The diagnostic standard OBDonUDS SAE J1979-2 describes the communication between the vehicle's OBD systems and external UDS-based test equipment,” *Softing Automotive*, 2021. [Accessed by 18/05/2025].
11. Learnoverpass, “Overpass API Documentation.” *Osmlab*, 2023. [Accessed by 21/05/2025].
12. Leaflet.js Official Site, “Leaflet 1.9 API Reference,” *Leaflet*, 2025. [Accessed by 23/06/2026].
13. Werkzeug Documentation, “Werkzeug Security Utilities.” *Werkzeug*, 2025. [Accessed by 25/06/2026].
14. Python Software Foundation, “difflib: Helpers for Computing Deltas,” *python.org*, 2025. [Accessed by 27/06/2026].