Software Requirements Specification

for

Fog Adaptive Vehicular Monitoring and Control System

Version 1.0 approved

Prepared by:
Pushkal Gupta 23BCT0253
Shagnik Paul 23BCT0266
Vaibhav Jain 23BAI0033
Adrivid Mishra 23BCT0264

Software Engineering Lab

10th February, 2026

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1.Introduction

Purpose

The scope of this SRS covers the complete system, which is a Fog-enabled vehicular cyber-physical system using non-intrusive AI energy fingerprinting for predictive maintenance, autonomous safety response, and intelligent after-sales monitoring. This system functions as a local 'Vehicular Fog Node' to handle millisecond-level safety decisions autonomously.

Document Conventions

We have printed an SRS that follows the structured, requirement-driven IEEE software specification format to provide clarity, traceability, and verifiability of the requirements provided. Mandatory behavior is considered as being designated by the use of the keyword “shall.” Each functional and non-functional requirement will be individually assigned a unique identifier (for example, REQ-x) and each will have a priority assigned to it independently; that is, priorities assigned at a higher level do not necessarily flow down into the lower levels. A logical system decomposition is indicated through the use of headers and numbering; use of stimulus-response and I/P-O/P formats encourages standardization of behavioral descriptions. Explicit marks will be displayed on all incomplete and/or placeholder information in order to prevent ambiguity.

Intended Audience and Reading Suggestions

1.3.1 Intended Audience: *The primary audience for this Software Requirements Specification (SRS) includes:*

1. ***Developers and Implementers:*** *To understand the functional and non-functional requirements for designing and coding the system.*
2. ***Testers:*** *To define test cases and ensure the system meets all specified criteria.*
3. ***Project Managers:*** *To track scope, estimate effort, and manage project risks and dependencies.*
4. ***System Integrators:*** *To manage the interfaces and communication protocols between the Fog Node, Edge Hardware, and Cloud services.*

1.3.2 Reading Suggestions:

1. ***Overview:*** *All readers should begin with Section 1 (Introduction) and Section 2 (Overall Description) to understand the product's scope, perspective, and major functions.*
2. ***Technical Staff (Developers, System Integrators, Testers):*** *Should focus on Section 3 (External Interface Requirements) and Section 4 (System Features) for detailed functional specifications, followed by Section 5 (Other Nonfunctional Requirements) for constraints and quality attributes.*
3. ***Management and Stakeholders (Project Managers):*** *Should focus on Section 1 (Introduction) and Section 2 (Overall Description) for the business context, and Section 5 (Other Nonfunctional Requirements) for constraints, performance, safety, and security.*
4. ***Glossary:*** *Appendix A should be consulted for definitions of specialized terminology (e.g.,*

Fog Node, NIVLM, Health Vector).

5. ***Incomplete Information:*** *Readers should note all "TBD" (To Be Determined) items listed in Appendix C for requirements that are currently placeholders.*

Product Scope

It is designed to overcome ***cloud latency*** *by using a local* ***'Vehicular Fog Node'*** *for* ***millisecond-level autonomous safety decisions****. It employs* ***Non-Intrusive AI Energy Fingerprinting (NIVEF)*** *for* ***multi-subsystem health assessment****.*

Key Objectives:

1. ***Autonomous Safety:*** *Executes* ***Latency-Locked Vehicular Fog Actuation (LLVFA)*** *to ensure critical actions during network blackouts.*
2. ***Predictive Maintenance:*** *Calculates* ***Remaining Useful Life (RUL)*** *using a* ***Deep-Learning-Based Vehicle Aging Index (VAI)*** *for proactive servicing.*
3. ***Operational Efficiency:*** *Reduces data costs by transmitting* ***Edge-Compressed Vehicular Health Vectors****.*
4. ***Monitoring:*** *Provides a* ***Live Digital Twin Visualization*** *for real-time state viewing.*

The system ensures ***operational independence*** *for safety-critical functions, regardless of cloud connectivity.*

References

- Hossain et al., "Survey on AI-Based Diagnostic Techniques Across Subsystems for Vehicle Fault Diagnosis," *Elsevier*, 2025.
- Mahale et al., "Review of AI-Driven Predictive Maintenance Architectures for Vehicle Maintenance Planning," *Springer*, 2025.
- Min et al., "Multi-Sensor Fusion with ML Classifiers for High-Accuracy Vehicle Fault Classification," *Elsevier*, 2023.
- Zhang et al., "Cloud-Synchronized Digital Twin Implementation for Real-Time Vehicle Monitoring," *IEEE*, 2023.
- Behravan et al., "Resource Pooling Mechanisms for Efficient Utilization of Fog Resources in Vehicular Networks," *Springer*, 2021.
- Hossain et al., "AI-Based Fault Detection Models for Improved Rule-Based Diagnostics," *Elsevier*, 2024.
- Bharatheedasan et al., "Hybrid MLP–LSTM Deep Learning Model for Superior Remaining Useful Life (RUL) Prediction," *Elsevier*, 2025.
- Ucar et al., "Review of AI Validation and Explainability Methods for Trust in Predictive Maintenance," *Elsevier*, 2024.
- Kapoor et al., "Analysis of Edge–Fog–Cloud Deployment Strategies to Reduce Cloud-Only Monitoring Latency," *Springer*, 2025.
- Shah et al., "Vibration Signal Processing for Early Mechanical Fault Detection in Powertrains," *Elsevier*, 2025.

2.Overall Description

Product Perspective

EcoSmart Vehicle Intelligence is a new, self-contained vehicular monitoring system that augments existing vehicle architectures by adding a fog-enabled intelligence layer without replacing OEM ECUs or control logic. It uses non-intrusive power-line sensing to infer vehicle health and execute real-time safety and predictive decisions locally through a Fog-Adaptive Vehicular Control Loop (FAVCL), ensuring operation even during network loss. The system follows a layered design with clearly defined interfaces between edge sensing, fog-based decision logic, optional cloud analytics, and a real-time web dashboard.

- New standalone product, not a replacement for existing ECUs.
- Non-intrusive integration via main power-line monitoring.
- Fog node is the primary, low-latency decision authority.
- Cloud layer is optional and non-critical.

Product Functions

2.2 Product Functions

The product is designed to perform a closed-loop control and monitoring process, with core functions centered on local, real-time intelligence at the Vehicular Fog Node to ensure safety and efficiency.

The major functions include:

1. ***Real-Time Data Processing:*** *Acquires raw sensor data and extracts features (FFT, statistics) at the fog level for immediate analysis.*
2. ***Autonomous Safety (LLVFA):*** *Instantly detects critical conditions and issues Latency-Locked Actuation commands to hardware, ensuring safety independent of cloud connection.*
3. ***Predictive Maintenance (VAI/RUL):*** *Assesses subsystem health and calculates Remaining Useful Life (RUL) using the Deep-Learning-Based Vehicle Aging Index (VAI) for proactive servicing.*
4. ***Dynamic Resource & Behavior Management:*** *Arbitrates electrical loads (EDVTA) and enforces operational limits based on driver behavior (DBESP) to protect components and ensure critical systems have power.*
5. ***Data Efficiency:*** *Compresses health information into* ***Edge-Compressed Vehicular Health Vectors*** *for cost-effective transmission to the cloud.*
6. ***User Visualization:*** *Provides a* ***Live Digital Twin Visualization*** *for real-time, lag-free monitoring of the vehicle's state.*

User Classes and Characteristics

<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>

Operating Environment

The Fog-Based Vehicular Monitoring System operates in a heterogeneous cyber-physical environment consisting of embedded hardware, fog-level computation infrastructure, optional cloud services, and user-facing web applications.

Hardware Environment

- ***Edge Sensor Node**:*
    - *ESP32 microcontroller for non-intrusive electrical signal acquisition and preprocessing*
    - *Current Transformer (CT) sensors for power-line sensing*
    - *Signal conditioning circuits, ADCs, and isolation components*
    - *Electromechanical relays for fail-safe hardware actuation*

- ***Vehicular Fog Node**:*
    - *Raspberry Pi or equivalent single-board computer installed within the vehicle or at a nearby edge location*
    - *Acts as the primary real-time decision authority*

- ***Vehicle Platform**:*
    - *Standard 12V/24V automotive electrical systems*
    - *Existing OEM ECUs remain unchanged and operate independently*

Software Environment

- ***Edge Firmware**:*
    - *C++ firmware running on ESP32*
    - *Handles sensor acquisition, noise filtering, FFT preprocessing, and data transmission*

- ***Fog Layer Software**:*
    - *Python-based middleware executing Fog-Adaptive Vehicular Control Loop (FAVCL)*
    - *TensorFlow Lite for low-latency AI inference*
    - *Eclipse Mosquitto MQTT broker for inter-layer communication*

    - *Docker for containerized deployment of fog services*

- ***Cloud & Storage (Non-Critical)**:*
    - *MongoDB Atlas (or equivalent NoSQL database) for long-term health data storage*
    - *Cloud services are optional and not involved in safety-critical decisions*
- ***User Interface Environment**:*
    - *Web-based dashboard developed using React.js*
    - *Hosted on a cloud platform such as Vercel*
    - *Uses WebSockets for real-time Digital Twin visualization*

Communication Environment

- *MQTT for edge-to-fog data streaming*
- *HTTPS/WebSockets for dashboard communication*
- *Cellular or Wi-Fi connectivity for optional cloud synchronization*

The system is designed to operate safely and autonomously even in the absence of network connectivity, ensuring uninterrupted safety enforcement and monitoring.

Design and Implementation Constraints

The design and implementation of the Fog-Based Vehicular Monitoring System are subject to the following constraints, which influence architectural, technological, and operational decisions.

Hardware Constraints

- *The system must operate within the **computational, memory, and power limitations** of embedded platforms such as ESP32 and Raspberry Pi.*
- *Sensor acquisition must be **non-intrusive**, prohibiting direct modification of OEM ECUs, CAN bus wiring, or proprietary vehicle control logic.*
- *Actuation mechanisms must support **fail-safe behavior**, ensuring safe operation during software or hardware faults.*

Performance Constraints

- *Fog-level safety decisions must be completed within **strict real-time bounds (≤100 ms)**.*
- *Hardware actuation must be triggered within **≤50 ms** after a critical safety decision.*
- *Continuous operation must be supported without degradation under sustained sensor input rates.*

Software and Technology Constraints

- *AI models must be deployable using **TensorFlow Lite** to meet real-time inference and memory requirements.*

- *Inter-process and inter-device communication shall use **lightweight protocols (MQTT)** to minimize latency and overhead.*
- *The fog software stack must be compatible with **Linux-based operating systems**.*

Security Constraints

- *Safety-critical actuation logic must execute **exclusively at the fog layer** and shall not be cloud-controlled.*
- *Authentication and encryption mechanisms must be applied to all inter-layer communications.*
- *Unauthorized override of safety decisions is strictly prohibited.*

Regulatory and Maintainability Constraints

- *The system must comply with **automotive safety and electrical isolation best practices**.*
- *Modular design is required to allow independent updates to sensing, AI models, fog logic, and UI components.*
- *Programming standards and documentation must support long-term maintainability by third-party developers.*

*These constraints ensure that the system remains **safe, reliable, low-latency, and deployable in real-world vehicular environments** while maintaining compatibility with existing vehicle architectures.*

User Documentation

<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>

Assumptions and Dependencies

- **Assumed Factors**
  - **Hardware & Data Quality:** Non-intrusive power-line sensing hardware is reliable, correctly installed, and provides accurate sensor data.
  - **Model Validation:** The pre-trained AI models (VAI/RUL) are accurate, and the Fog-Level Autonomous Safety Logic (LLVFA) is correctly defined and validated.
  - **Compatibility:** The Fog Node software is compatible with the vehicle's embedded operating environment.
- **External Dependencies**
  - **Core Vehicle Systems:** The system relies on the correct, continued operation of existing OEM ECUs and control logic.
  - **Cloud Services:** Dependence on a third-party or internal cloud platform for long-term data storage, analytics, and Digital Twin visualization (non-safety functions).
  - **Connectivity:** Availability of a functional wireless communication link (e.g., cellular

network) for non-safety data transmission (Health Vectors to the cloud).
- **Power:** The entire system requires a continuous and stable power supply from the vehicle's electrical system.

3.External Interface Requirements

User Interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

4.System Features

This section describes the major system features of the Fog-Based Vehicle Monitoring System. Each feature is specified with a clear description, priority, stimulus–response behavior, and detailed functional requirements. All requirements are unambiguous, complete, consistent, and verifiable.

4.1 Real-Time Vehicle Sensor Data Acquisition

4.1.1 Description and Priority

This feature enables continuous acquisition of raw vehicular sensor data from embedded hardware installed in the vehicle. The data represents the physical state of vehicle subsystems and serves as the foundation for all higher-level analysis and decision-making.

Priority: High

Priority Component Ratings (1–9):

Benefit: 9
Penalty: 9
Cost: 5
Risk: 6

4.1.2 Stimulus / Response Sequences

Stimulus 1:
Vehicle sensors generate analog or digital signals during normal operation.

System Response 1:
The system samples sensor values, timestamps them, and forwards the raw data to the fog node.

4.1.3 Functional Requirements

REQ-1: The system shall continuously sample vehicle sensor data from the embedded hardware layer.

I/P: Analog/digital sensor signals
O/P: Raw sensor samples
Processing: ADC sampling and buffering

REQ-2: The system shall attach a timestamp and device identifier to each sensor sample.

I/P: Raw sensor samples
O/P: Timestamped sensor data
Processing: Metadata tagging

REQ-3: The system shall transmit raw sensor data to the fog node.

I/P: Timestamped sensor data
O/P: Sensor data packets sent to fog
Processing: Network transmission

REQ-4: The system shall discard or flag corrupted sensor readings.

I/P: Invalid or incomplete sensor samples
O/P: Error flag or discarded sample
Processing: Data validation

4.2 Fog-Level Feature Extraction

4.2.1 Description and Priority

This feature extracts meaningful physical and statistical features from raw sensor data at the fog layer to reduce data volume and enable real-time analysis.

Priority: High

Priority Component Ratings (1–9):

Benefit: 8
Penalty: 8
Cost: 6
Risk: 6

4.2.2 Stimulus / Response Sequences

Stimulus 1:
Raw sensor data is received at the fog node.

System Response 1:
The system computes derived features such as RMS, variance, and thermal indicators.

4.2.3 Functional Requirements

REQ-1: The system shall compute statistical features from raw sensor data.
I/P: Raw sensor data
O/P: Feature vectors
Processing: RMS, variance, FFT

REQ-2:The system shall normalize extracted features to defined operational ranges.
I/P: Feature vectors
O/P: Normalized feature vectors
Processing: Range normalization

REQ-3:The system shall reject feature computation when input data is insufficient.

I/P: Incomplete data window
O/P: Feature computation error
Processing: Window validation

4.3 Subsystem Health Assessment

4.3.1 Description and Priority

This feature evaluates the health of critical vehicle subsystems such as braking, engine, and battery using fog-level computations.

Priority: High

Priority Component Ratings (1–9):
Benefit: 9
Penalty: 9
Cost: 6
Risk: 7

4.3.2 Stimulus / Response Sequences

Stimulus 1:
Normalized feature vectors are generated.

System Response 1:
The system computes subsystem-specific health indices.

4.3.3 Functional Requirements

REQ-1: The system shall compute health indices for each monitored subsystem.

I/P: Normalized feature vectors
O/P: Subsystem health scores
Processing: Formula-based evaluation

REQ-2: The system shall update health indices in real time.

I/P: New feature vectors
O/P: Updated health scores
Processing: Rolling computation

REQ-3: The system shall mark a subsystem as degraded when health falls below thresholds.

I/P: Health score
O/P: Degradation flag
Processing: Threshold comparison

4.4 Fog-Level Autonomous Safety Decision Making

4.4.1 Description and Priority

This feature enables the fog node to autonomously determine vehicle safety states and required actions without relying on cloud connectivity.

Priority: High

4.4.2 Stimulus / Response Sequences

Stimulus 1:
A subsystem health score crosses a critical threshold.

System Response 1:
The system classifies the condition and decides corrective actions.

4.4.3 Functional Requirements

REQ-1: The system shall classify vehicle conditions as normal, warning, or critical.
I/P: Subsystem health scores
O/P: Safety classification
Processing: Rule-based logic

REQ-2: The system shall generate a safety decision event for critical conditions.
I/P: Critical classification
O/P: Safety decision event

4.5 Hardware Actuation and Enforcement

4.5.1 Description and Priority

This feature performs actual physical actuation on vehicle hardware to mitigate damage and ensure safety.

Priority: High

4.5.2 Stimulus / Response Sequences

Stimulus 1:
A critical safety decision is generated.

System Response 1:
The system sends actuation commands to vehicle hardware.

4.5.3 Functional Requirements

REQ-1: The system shall issue actuation commands to the hardware layer.

I/P: Safety decision event
O/P: Actuation command

REQ-2: The system shall enforce speed limiting, braking control, or cooling activation.
I/P: Actuation command
O/P: Physical subsystem response

REQ-3: The system shall override non-critical operations during safety enforcement.
I/P: Critical actuation state
O/P: Disabled non-critical actions

4.6 Edge-to-Cloud Health Vector Transmission

4.6.1 Description and Priority

This feature compresses and transmits summarized vehicle health information to the cloud.

Priority: Medium

4.6.3 Functional Requirements

REQ-1: The system shall generate compact health vectors.
I/P: Subsystem health scores
O/P: Health vector

REQ-2:The system shall transmit health vectors to the cloud.
I/P: Health vector
O/P: Cloud-stored data

4.7 Digital Twin Visualization

4.7.1 Description and Priority

This feature provides real-time visualization of vehicle state to users.

Priority: Medium

4.7.3 Functional Requirements

REQ-1: The system shall display real-time vehicle health on a dashboard.
I/P: Health vectors
O/P: UI updates

4.8 Predictive Maintenance and Lifecycle Estimation

4.8.1 Description and Priority

This feature estimates the remaining useful life of vehicle components.

Priority: Medium

4.8.3 Functional Requirements

REQ-1: The system shall estimate the remaining useful life of components.
I/P: Historical health data
O/P: RUL percentage

5.Other Nonfunctional Requirements

5.1 Product Requirements

5.1.1 Performance Requirements

REQ-1: Fog-Level Decision Latency

Description: The system shall meet strict real-time performance constraints to ensure safety-critical decisions and actuation are performed without delay.

R1.1: *The system shall complete fog-level safety decision-making within a bounded time limit.*
I/P: *Subsystem health scores and feature vectors*
O/P: *Safety decision (normal / warning / critical)*
Processing:
Rule evaluation and threshold comparison , end-to-end fog decision latency ≤ 100 milliseconds

---

R1.2: *The system shall trigger hardware actuation within a fixed delay after a critical decision.*
I/P: *Critical safety decision*
O/P: *Actuation command sent to hardware*
Processing:
Command validation , Actuation initiation ≤ 50 milliseconds after decision

---

R1.3: *The system shall support continuous monitoring without performance degradation.*
I/P: *Continuous sensor data stream*
O/P: *Stable feature extraction and decision rate*
Processing:
Sustained operation at ≥ 10 sensor updates per second

5.2 Safety Requirements

The system shall be designed to prevent loss, damage, or harm to the vehicle and its occupants by enforcing safety-critical controls at all times. When a critical safety condition is detected, the system shall automatically initiate appropriate hardware actuation to mitigate risk without requiring human intervention. To ensure system integrity, safety actions shall not be suppressible or overridden by unauthorized entities. In the event of internal failures such as fog node malfunction or software errors, the system shall transition into a predefined safe operational state that restricts non-essential functionality while maintaining safety monitoring. The system shall also preserve full safety functionality during loss of cloud connectivity by operating autonomously at the fog layer. These safeguards ensure that safety enforcement remains effective under normal operation, fault conditions, and network disruptions, in accordance with accepted safety-critical system design principles.

5.3 Security Requirements

The system shall ensure the confidentiality, integrity, and authenticity of all vehicular data generated, processed, or transmitted during operation. To prevent unauthorized access and malicious control, all edge devices and fog nodes shall be authenticated prior to any data exchange. Data transmitted between the edge, fog, and cloud layers shall be protected using encryption mechanisms to safeguard against interception and tampering. Safety-critical actuation commands shall be generated exclusively by authorized fog-level logic to prevent unauthorized or unsafe control actions. Additionally, all security-relevant events, including authentication attempts and access violations, shall be securely logged to support auditing, traceability, and forensic analysis. These measures collectively ensure secure operation, protect user privacy, and maintain trustworthiness of the system under both normal and adverse conditions.

5.4 Software Quality Attributes

The system shall be designed to meet high standards of availability, reliability, maintainability, scalability, portability, usability, robustness, and testability. The fog node shall maintain an availability of at least 99.5% during vehicle operation and support continuous system operation for a minimum of eight hours without requiring restart. The system shall support modular updates to sensing components, fog-level logic, and user interface modules, with a mean time to repair not exceeding 30 minutes. The architecture shall allow monitoring of multiple vehicles without requiring architectural changes and shall enable the fog logic to operate across multiple hardware platforms with minimal configuration effort. Safety alerts shall be presented in a manner that is easily understandable by non-technical users within five seconds of display. The system shall tolerate intermittent sensor noise without generating false critical safety triggers. All functional and non-functional requirements shall be independently testable using system logs, metrics, or observable outputs to ensure verifiability and correctness.

5.5 Business Rules

The system shall operate under clearly defined role-based control principles to ensure safe and predictable behavior. Only the fog node is authorized to make safety decisions and issue hardware actuation commands, while the hardware layer is restricted to executing received commands

without participating in safety logic. The cloud layer shall be limited to analytics, storage, and long-term intelligence functions and shall not perform real-time safety actuation. Safety-related decisions shall take precedence over all performance, comfort, and infotainment functions to ensure protection of the vehicle and occupants. Raw sensor data generated by the vehicle shall remain under the ownership of the vehicle system, and only processed health vectors may be shared with cloud services for analytical purposes. The system shall retain full safety functionality and enforcement capability regardless of cloud availability, ensuring operational independence under all network conditions.

6.Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

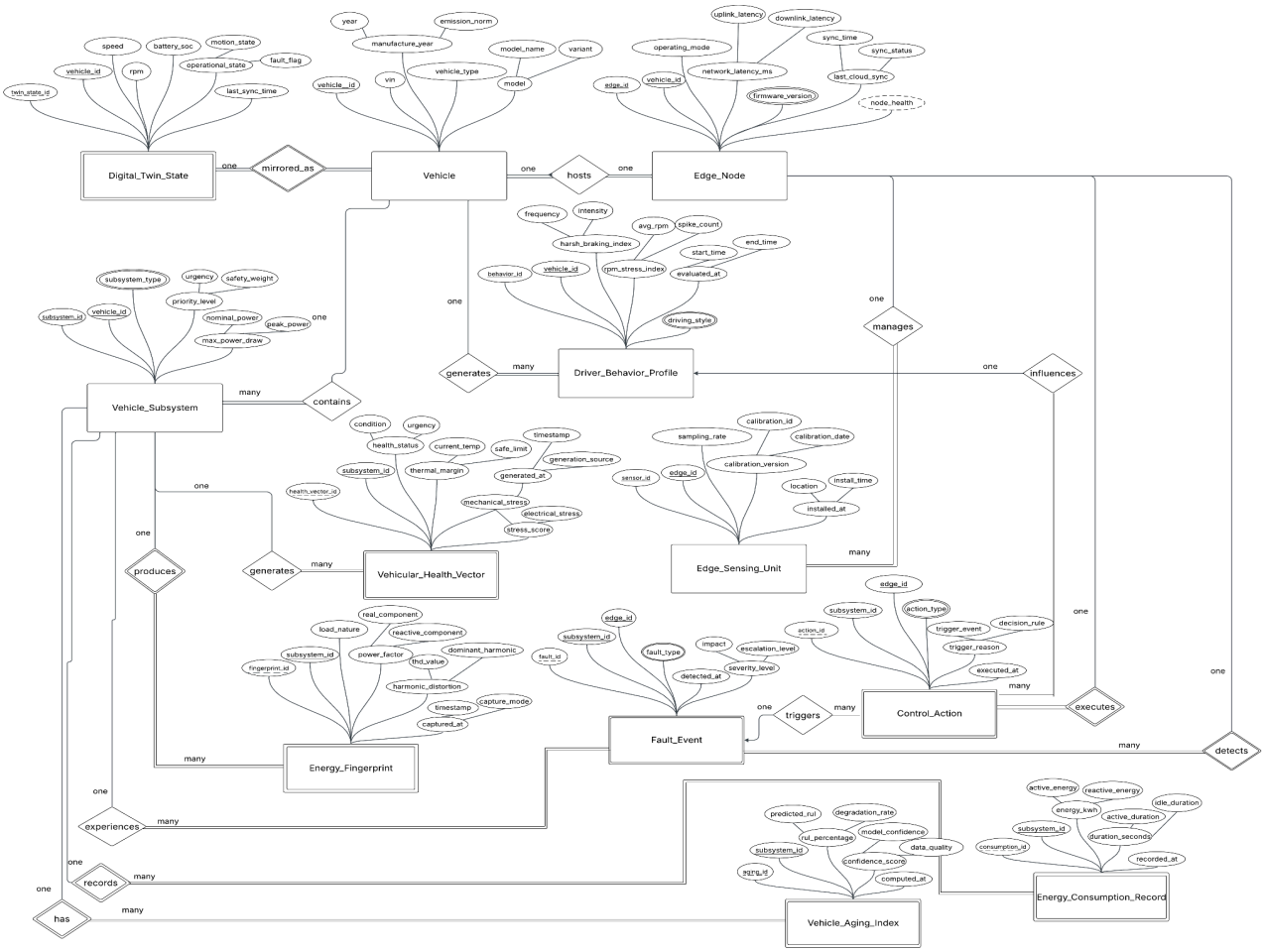
Appendix A: Glossary

Term	Definition
CT Sensors	Current Transformer Sensors; hardware for non-intrusive electrical sensing.
DBESP	Driver Behavior Enforcement Protocol; enforces operational limits based on driver behavior.
Digital Twin	Real-time web dashboard visualization of the vehicle's state.
ECUs	Electronic Control Units; existing, independent OEM control logic.
EDVTA	Dynamic Voltage and Timing Arbitration; arbitrates electrical loads.
Health Vector	Compact, compressed summary of vehicle health data sent to the cloud.
FAVCL	Fog-Adaptive Vehicular Control Loop; Python middleware for real-time decision logic on the Fog Node.
FFT	Fast Fourier Transform; extracts features from sensor data at the fog layer.
Fog Node	Local, low-latency decision-making computer in the vehicle for autonomous safety.
LLVFA	Latency-Locked Vehicular Fog Actuation; autonomous safety function that issues critical hardware commands.
MQTT	Message Queuing Telemetry Transport; lightweight protocol for edge-to-fog communication.

NIVEF	Non-Intrusive AI Energy Fingerprinting; core technology for multi-subsystem health assessment.
RUL	Remaining Useful Life; estimated component lifespan for predictive maintenance.
SRS	Software Requirements Specification (this document).
TBD	To Be Determined; a placeholder for incomplete information.
TensorFlow Lite	Framework for deploying AI models (like VAI/RUL) on the Fog Node.
VAI	Vehicle Aging Index; deep-learning index for RUL calculation.

Appendix B: Analysis Models

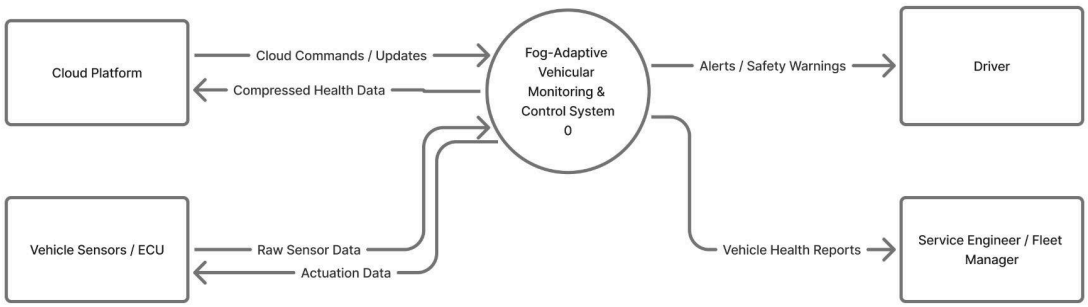
Entity Relationship Diagram:



For a clearer image, please visit this link.

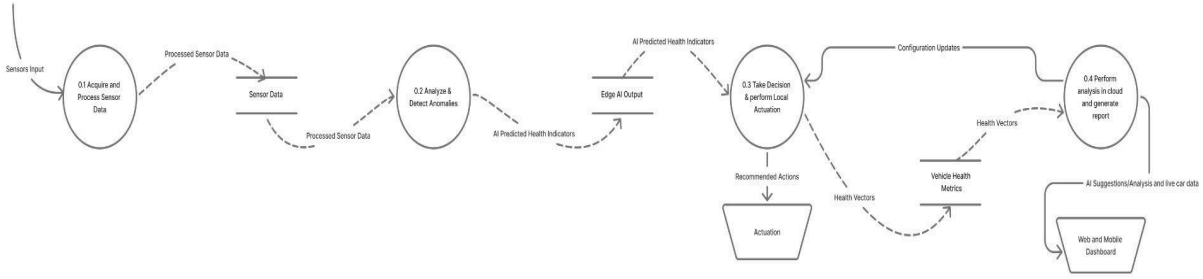
Data Flow Diagram:

Level 0:



Level 0 DFD – Fog-Adaptive Vehicular Monitoring System

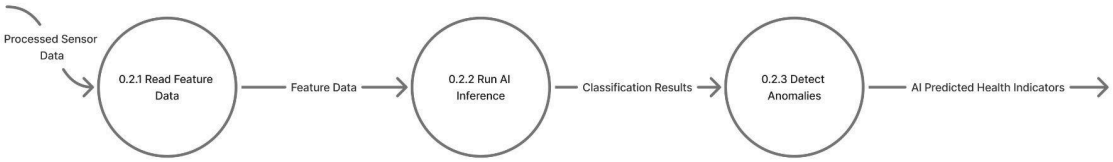
Level 1:



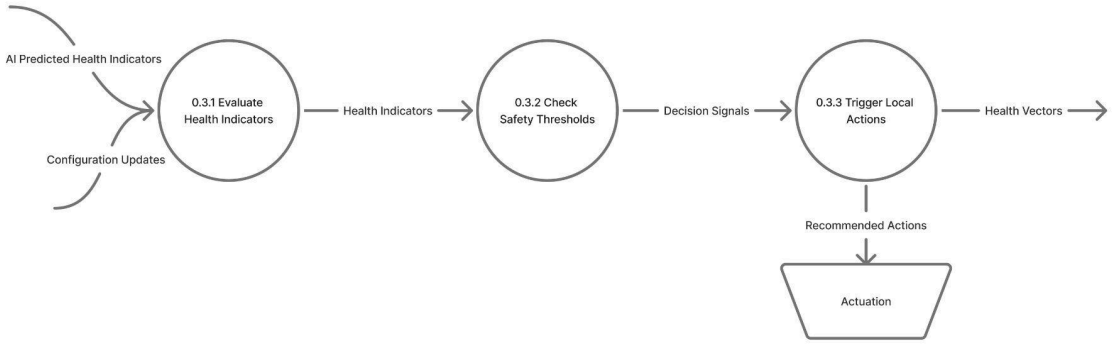
Level 1 DFD – Fog-Adaptive Vehicular Monitoring & Control System

Level 2:

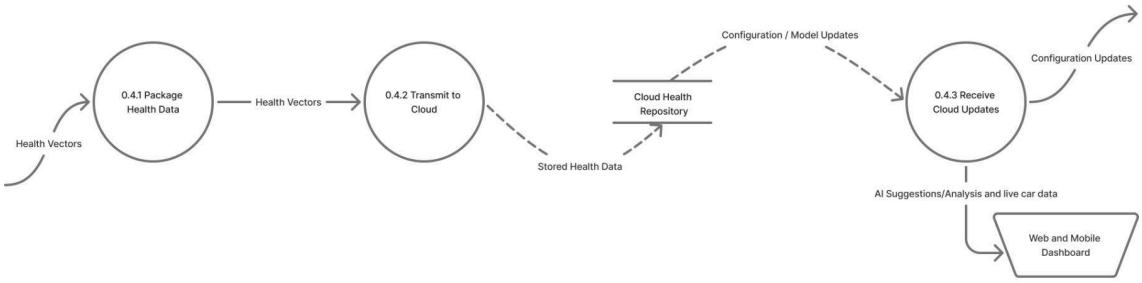
Level 2 DFD – Process 0.1 Acquire Sensor Data



Level 2 DFD – Process 0.2 Analyze & Detect Anomalies



Level 2 DFD – Process 0.3 Decision & Local Actuation



Level 2 DFD – Process 0.4 Cloud Communication & Reporting

Fog-Adaptive Vehicular Monitoring & Control System Data Dictionary

The system's data is structured into the following key components:1. Raw Sensor Data

RawSensorData = ElectricalSignal + ThermalSignal + VibrationSignal + (Timestamp)

- **ElectricalSignal**: `float` (measured current/voltage)
- **ThermalSignal**: `float` (temperature reading)
- **VibrationSignal**: `float` (vibration amplitude)
- **Timestamp**: `datetime` (optional)

2. Processed Sensor Data

ProcessedSensorData = FilteredSignal + FeatureVector

- **FilteredSignal**: `float` (noise-reduced signal)
- **FeatureVector** = RMSValue + FFTComponents + PeakValue
    - **RMSValue**: `float` (signal strength)
    - **FFTComponents**: `{FrequencyMagnitude}`
    - **PeakValue**: `float` (maximum value)

3. AI Output & Health Indicators

AIOutput = AnomalyStatus + HealthIndicators

- **AnomalyStatus**: `[Normal, Warning, Critical]`
- **HealthIndicators** = HealthScore + FaultLabel
    - **HealthScore**: `integer` (0–100)
    - **FaultLabel**: `string` (detected fault)

4. Decision Signals

DecisionSignals = ActionType + SeverityLevel

- **ActionType**: `[Alert, LimitOperation, EmergencyShutdown]`
- **SeverityLevel**: `[Low, Medium, High]`

5. Health Vector & Compressed Data

- **HealthVector** = HealthScore + FaultLabel + RemainingUsefulLife
    - **RemainingUsefulLife**: `integer` (percentage)
- **CompressedHealthData** = VehicleID + HealthVector + Timestamp
    - **VehicleID**: `string` (unique vehicle id)

6. Alerts

Alerts = AlertType + Message

- **AlertType**: `[Warning, Critical]`
- **Message**: `string`

Appendix C: To Be Determined List

These items are tracked to ensure closure before system deployment.

- Final selection and specifications of production-grade sensors and supporting hardware components for the edge node.
- Exact threshold values for subsystem health classification (normal, warning, critical) after extended real-world calibration.
- Final dataset size and training parameters for deep-learning models used in Vehicle Aging Index (VAI) computation.
- Cloud deployment configuration details, including region selection and long-term data retention policies.
- Final user interface layout and visual design standards for the Digital Twin dashboard.
- Regulatory and compliance requirements applicable to on-road vehicle deployment in different regions.