BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

AI Predictive Maintenance for Smart Factories

[1]Ankita Pokale, [2]Varun Yadav, [3]Yash Vadak, [4]Aditya Salve, [5]Harsh Waghchoure

[1–5]Students, Dept. of Artificial Intelligence and Data Science
[1–5]Ajeenkya DY Patil School of Engineering (SPPU), Pune, India
Emails: ankita.pokale@dypic.in, varun.yadav@dypic.in, yash.vadak@dypic.in, aditya.salve@dypic.in, harsh.waghchoure@dypic.in

Abstract

Unexpected equipment stoppages can interrupt production, increase maintenance costs and reduce manufacturing consistency. This paper presents a low-cost Internet of Things (IoT) prototype for machine-condition monitoring as a foundation for predictive maintenance in smart factories. The prototype uses an ESP32 microcontroller connected to a DHT22 temperature-and-humidity sensor, an SW-420 digital vibration detector and an ACS712 current-sensor module. A low-voltage DC motor is driven through an L298N module, while LEDs and a buzzer provide local threshold-based indications. The ESP32 periodically sends a JSON payload over Wi-Fi to a FastAPI service hosted on a Mac computer. The backend exposes endpoints for receiving the latest sensor data and retrieving historical records, which are stored in an SQLite database. A Streamlit dashboard displays the latest measurements and supports historical visualization. During implementation, sensor readings were visible through the serial monitor, the HTTP POST request returned status code 200, and the history endpoint returned stored records. These observations demonstrate the operation of the core data pipeline. The current prototype performs monitoring and rule-based alerts; a trained and validated artificial-intelligence model has not yet been implemented. In addition, the current field from the ACS712 represents a raw analogue-to-digital converter (ADC) reading rather than calibrated amperes. Future work includes sensor calibration, longer-term data collection, fault-labelled experiments, anomaly detection, and validation of machine-learning methods against documented operating conditions.

Keywords: predictive maintenance, smart factory, Industrial Internet of Things, ESP32, condition monitoring, FastAPI, SQLite, Streamlit, sensor data, anomaly detection.

I. Introduction

Manufacturing equipment is expected to operate safely, reliably and consistently. Motors and other rotating machinery can experience overheating, vibration, electrical loading, loosened components or mechanical wear. If changes in operating condition are not noticed early, equipment may stop unexpectedly, production schedules can be disrupted, and emergency maintenance may be required. Regular manual inspection remains useful, but it provides only periodic snapshots and may not capture changes occurring between inspections.

Maintenance strategies are commonly described as corrective, preventive and predictive. Corrective maintenance takes place after a failure; preventive maintenance follows a predetermined schedule; predictive maintenance uses observed equipment condition and relevant historical information to help determine when intervention may be needed. Predictive maintenance depends on trustworthy

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

measurements, appropriate context and validation. A dashboard that displays sensors or a rule that activates an alarm is an important monitoring step, but it does not by itself predict a future failure.

The Industrial Internet of Things (IIoT) enables physical equipment to be connected to software systems through sensors, controllers and communication networks. In a small-scale system, a microcontroller can gather readings, transmit them wirelessly, and send them to a server for storage and analysis. A dashboard can make current values and historical trends accessible to a user. Once enough representative data have been collected, statistical analysis or machine-learning methods can be explored to detect deviations from normal operation.

This paper presents an educational prototype that integrates an ESP32, environmental and machine-condition sensors, a low-voltage DC motor, a FastAPI backend, SQLite storage and a Streamlit dashboard. The contribution is an end-to-end monitoring workflow: sensor acquisition, periodic network transmission, server-side reception, persistent history, visualization and simple local alerts. The paper distinguishes the functionality already demonstrated from AI-based prediction that remains future work.

A. Problem Statement

In a small workshop or educational smart-factory environment, sensor observations may be difficult to track over time, and operators may not have a convenient way to compare current machine behaviour with previous readings. A low-cost connected system is therefore needed to acquire several condition indicators, transmit them to a computer, store records and display trends. This project addresses that monitoring and data-management need. It does not yet estimate remaining useful life or provide a validated probability of machine failure.

B. Objectives and Contributions

- To monitor machine parameters such as temperature, vibration and electrical current using IoT sensors.
- To collect and transmit real-time sensor data using an ESP32-based IoT device.
- To provide a machine health score and failure-risk indication through a dashboard.
- To generate alerts when the machine shows abnormal operating conditions.
- To help reduce unexpected machine downtime and improve maintenance planning.

C. Scope

The current scope is a low-voltage laboratory prototype for monitoring and visualization. It is not a certified industrial control or safety system. The implemented alert rules are demonstration thresholds and must be calibrated to the equipment and operating context before practical deployment. AI prediction, fault classification and remaining-useful-life estimation are not claimed as completed or validated results.

II. Related Work

Research on manufacturing diagnostics and prognostics describes the importance of matching monitoring methods to the machine, the failure mode and the maintenance decision. Vogl, Weiss and Helu review diagnostic and prognostic capabilities for manufacturing and discuss practices relevant to the selection and

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

evaluation of such methods [1]. This motivates the distinction between identifying a present abnormal condition and forecasting a future failure.

Sharp, Ak and Hedberg survey the growing use of machine learning in smart manufacturing [2]. Machine-learning methods can identify patterns in complex data, but their value depends on measurement quality, relevant features, representative training examples and independent evaluation. For this project, a consistent sensor history is a necessary prerequisite before a predictive model can be developed.

Samatas, Moumgiakmas and Papakostas discuss predictive maintenance at the intersection of artificial intelligence and IoT [3]. IoT components enable collection and communication of measurements, while AI methods may analyse those measurements for signs of changing equipment condition. The two parts complement one another, but connectivity alone does not constitute prediction.

Xu and colleagues review the Industrial Internet of Things from a cyber-physical systems perspective [4], highlighting the interaction of physical processes, computing and communication. The present prototype follows a simplified version of this architecture: physical measurements are acquired by the ESP32, transmitted through Wi-Fi, stored by a backend service and presented through a web dashboard.

Dadfarnia, Sharp and Herrmann review condition-monitoring-based technologies in industrial maintenance [5]. Their work supports a structured evaluation approach: the suitability of a sensor or analytical technique depends on the maintenance task and evidence available. The present prototype combines several indicators, but experiments are still needed to determine how reliably they reveal specific faults. Table 1 summarizes the related research and its relevance.

Table 1: Related Research and Relevance

Reference	Topic	Relevance to this project
[1] Vogl et al.	Manufacturing diagnostics and prognostics	Motivates separating condition monitoring from validated prediction.
[2] Sharp et al.	Machine learning in smart manufacturing	Highlights data quality, features and evaluation requirements.
[3] Samatas et al.	AI and IoT for predictive maintenance	Supports a connected sensor-to-analysis architecture.
[4] Xu et al.	Industrial IoT and cyber-physical systems	Provides context for linking physical devices and software.
[5] Dadfarnia et al.	Review of condition-monitoring technologies	Supports systematic assessment of monitoring methods.
This work	ESP32 + FastAPI + SQLite + Streamlit	Demonstrates a working prototype data pipeline and rule-based local alerts.

Compared with periodic manual observation, the prototype offers recurring measurements and a retrievable history. Compared with a complete predictive-maintenance platform, it remains an early-stage monitoring system: fault-labelled datasets, validated models, performance metrics and industrial deployment testing are not yet available.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

III. System Architecture

The system follows the sequence Sense → Transmit → Receive → Store → Visualize → Review. Sensors provide signals to the ESP32. The controller packages measurements into a JSON object and sends the object over Wi-Fi using an HTTP POST request. FastAPI receives the payload and exposes endpoints for the latest measurement and historical records. SQLite stores readings in a local database file. Streamlit requests data from the API and presents current values and historical plots. The overall architecture is shown in Fig. 1, and the operating flow is shown in Fig. 2.

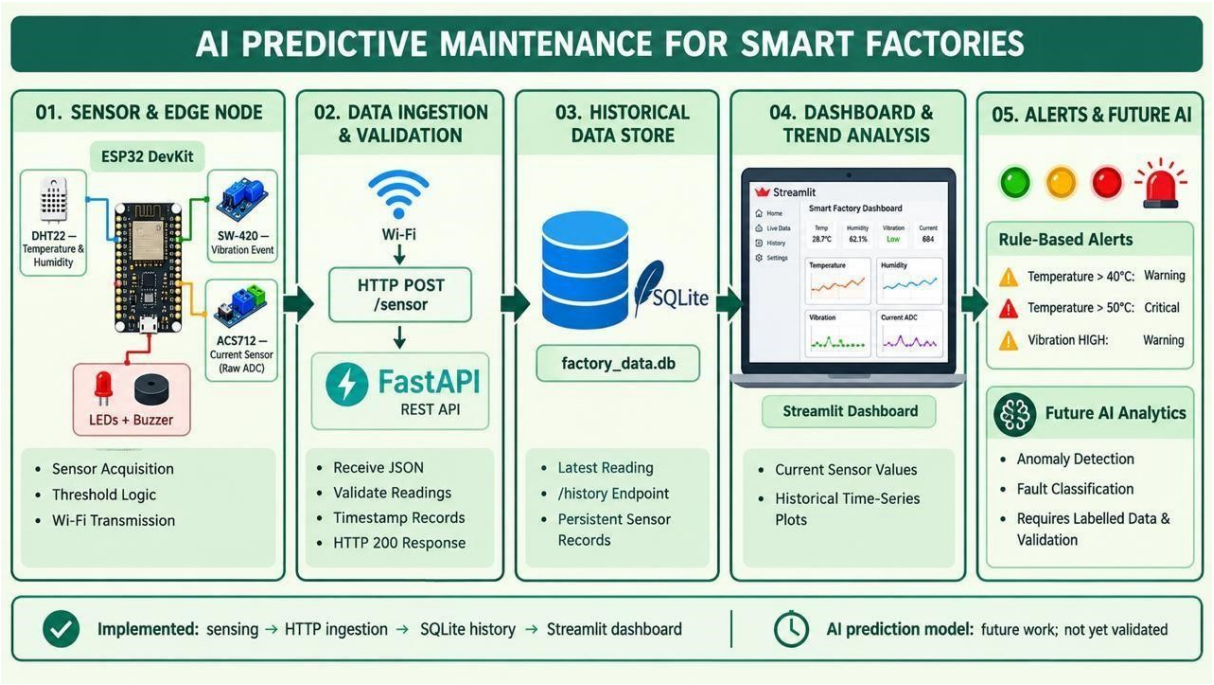


Fig. 1. System architecture.

The architecture can be divided into four layers. The sensing layer contains the DHT22, SW-420 and ACS712 modules. The edge layer contains the ESP32 firmware and local alert logic. The server and storage layer contains FastAPI and SQLite. The application layer contains the Streamlit dashboard used to inspect current and historical readings.

Data flow: DHT22 / SW-420 / ACS712 → ESP32 → Wi-Fi and HTTP → FastAPI API → SQLite database → Streamlit dashboard.

The current API routes are POST /sensor for incoming readings, GET /sensor for the latest reading and GET /history for stored readings. The API can also be inspected using FastAPI's interactive documentation page. The server runs on a Mac computer connected to the same reachable network as the ESP32. If the computer's local IP address changes, the endpoint configured in the ESP32 firmware must be updated.

The current architecture does not include an AI inference service. A future analytics layer could consume historical records, calculate rolling features and produce anomaly scores or fault-risk estimates after sufficient data and a valid evaluation method are available.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

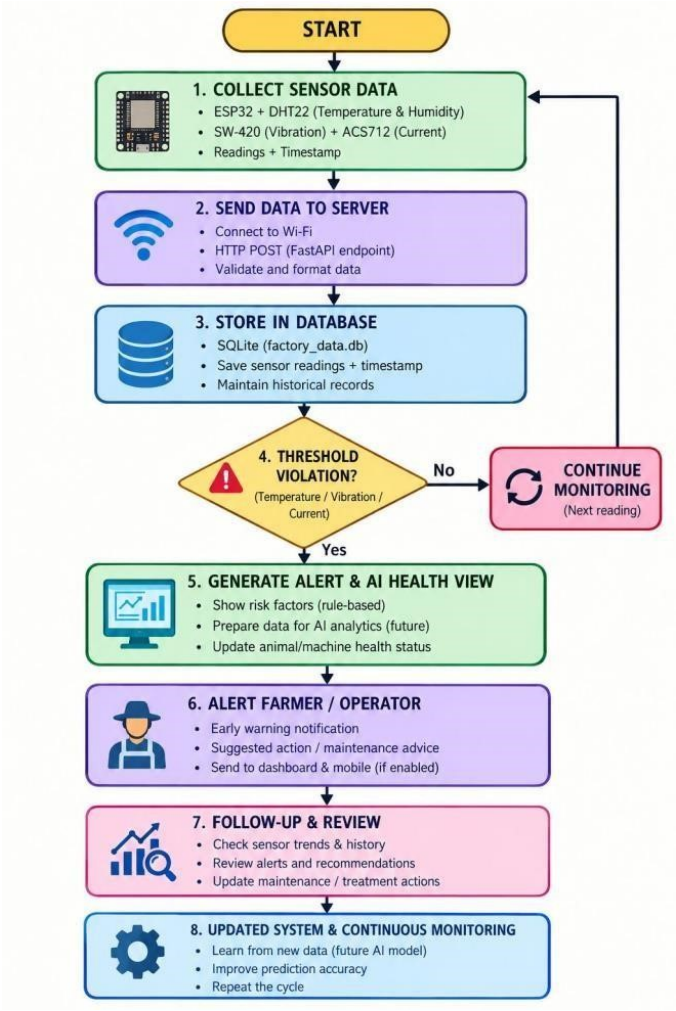


Fig. 2. System flowchart.

IV. Hardware Design

A. Controller and Local Functions

The ESP32 DevKit is the main controller. It reads digital and analogue inputs, executes simple threshold logic, updates LEDs and the buzzer, connects to Wi-Fi and transmits measurements to the backend. The prototype reports data at an interval of approximately three seconds. The ESP32 provides an economical way to combine sensor acquisition and network communication in one device.

B. Sensors and Their Roles

The DHT22 measures ambient temperature and relative humidity. Its data signal is connected to GPIO 4. Temperature can provide environmental context for a motor-monitoring experiment, while humidity describes the surrounding environment. These readings should not be interpreted as direct measurements of internal motor winding or bearing temperature.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

The SW-420 vibration detector provides a digital output connected to GPIO 34. Its output indicates that vibration has crossed the module's configured threshold. It does not provide a calibrated vibration amplitude or a frequency spectrum. Detailed vibration diagnosis would generally require a suitable accelerometer, appropriate sampling and signal processing.

The ACS712 current-sensor module provides an analogue output connected to GPIO 35. In the current firmware, the value sent to the server is a raw ADC reading. It is not expressed in amperes and should not be presented as a calibrated current measurement until the sensor is calibrated and a conversion is implemented. The signal voltage must remain within the ESP32's safe input range.

The L298N motor driver controls a low-voltage DC motor. The documented control connections are ENA on GPIO 25, IN1 on GPIO 26 and IN2 on GPIO 27. The motor should use an appropriate separate supply, with a common ground where required. The system is intended for low-voltage demonstration and must not be connected directly to mains equipment.

Three LEDs provide local status indication: green on GPIO 18, yellow on GPIO 19 and red on GPIO 23, using appropriate current-limiting resistors. A buzzer is controlled from GPIO 5. If the buzzer current exceeds what a GPIO can safely supply, a transistor or MOSFET driver should be used. Table 2 summarizes the components and pin allocation.

Table 2: Hardware Components and Pin Allocation

Component	Pin / Interface	Function
ESP32 DevKit	Controller	Reads sensors, applies rules and sends data
DHT22	GPIO 4	Ambient temperature and relative humidity
SW-420 vibration module	GPIO 34 (DO)	Digital vibration-threshold event
ACS712 current module	GPIO 35 (analogue)	Raw current-sensor ADC reading
L298N ENA	GPIO 25	Motor enable / speed-control input
L298N IN1	GPIO 26	Motor direction control
L298N IN2	GPIO 27	Motor direction control
Green LED	GPIO 18	Normal/status indication
Yellow LED	GPIO 19	Warning indication
Red LED	GPIO 23	Critical indication
Buzzer	GPIO 5	Audible alert

C. Power and Safety Considerations

The motor supply must be appropriate for the motor and driver and should not be routed through the ESP32's GPIO pins. All grounds that need a common reference should be connected correctly. The ACS712 output voltage must be checked against the ESP32 analogue-input limits before connection. Sensor wiring should be secure, the motor should be mounted to prevent movement, and the prototype should remain a low-voltage system. These precautions are important because motor switching can introduce electrical noise and because an incorrect sensor voltage can damage the controller. The connection diagram is shown in Fig. 3.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

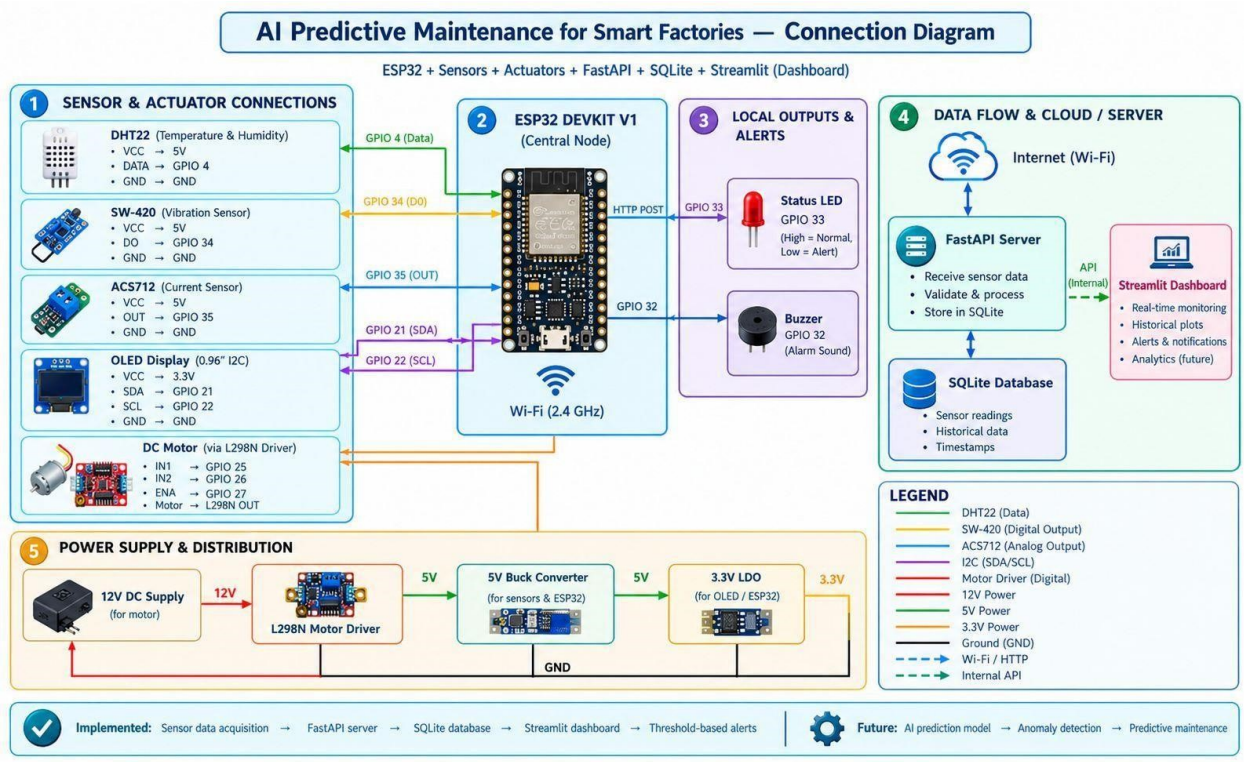


Fig. 3. Connection diagram.

D. Prototype Cost and Practicality

The design uses widely available development hardware and open-source software, making it suitable for a student prototype. A verified total bill of materials and cost breakdown have not been supplied in the current project record, so no exact total prototype cost is claimed here. Future documentation should list each component, quantity, unit price, power supply and enclosure cost.

V. Data Pipeline, Software and Dashboard

A. Sensor Acquisition and JSON Payload

The ESP32 reads the connected sensors and assembles a structured JSON object. The current payload contains machine_id, temperature, humidity, vibration and current. The current field represents the raw ACS712 ADC reading in the present version. Periodic transmission creates a sequence of observations that can be examined over time. A more robust implementation should attach consistent timestamps, validate sensor values and explicitly record sensor faults or missing readings.

B. Communication and FastAPI

The ESP32 sends an HTTP POST request to the FastAPI /sensor endpoint over the local Wi-Fi network. During testing, the POST request returned HTTP status code 200. This is evidence that the server handled the request successfully. The GET /sensor endpoint returns the latest measurement, while GET /history returns previously stored readings. Communication can fail if the ESP32 and computer are not on a reachable network, if the server is stopped, or if the computer's IP address changes.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

C. SQLite Historical Storage

SQLite stores sensor records in the local factory_data.db database file. The history endpoint makes these records available to the dashboard and for later inspection. Retaining records is important because maintenance analysis depends on patterns and changes over time rather than isolated readings. For future model development, records should include reliable timestamps, operating state, sensor calibration information and, where available, confirmed maintenance events.

D. Streamlit Dashboard

The Streamlit application requests the latest sensor record and historical data from FastAPI. It displays recent measurements and provides plots that help the user inspect trends. Temperature and humidity are environmental measurements, vibration is a digital state, and current is a raw ADC reading until calibration is completed. Clear labels are essential so that a user does not confuse raw sensor counts with physical units.

A future dashboard version could add last-update time, communication status, selectable time ranges, CSV export, per-machine views and alerts when readings are missing or stale. These are potential enhancements and should not be described as implemented unless they are tested. Table 3 summarizes the software stack.

Table 3: Software Stack by Layer

Layer	Technology	Role
Firmware	Arduino IDE / ESP32 code	Sensor reading, threshold logic and HTTP transmission
Communication	Wi-Fi, HTTP, JSON	Transfers readings from ESP32 to server
API / backend	Python, FastAPI	Receives data and serves latest/history endpoints
Database	SQLite	Stores historical sensor records
Dashboard	Streamlit	Displays current values and historical plots
Future analytics	Python ML libraries (planned)	Potential feature engineering and anomaly detection after data collection

VI. Monitoring Logic and Future AI Analytics

The present prototype uses threshold-based rules for immediate indication. Temperature above 40 °C is treated as a warning, and temperature above 50 °C is treated as a critical condition. A HIGH signal from the SW-420 module is treated as a vibration event. The configured state determines which LEDs and buzzer indications are activated. These values are demonstration settings, not universal industrial limits; appropriate thresholds depend on the motor, load, sensor location, environment and baseline behaviour.

A. Current Rule-Based Monitoring

Rule-based monitoring is transparent and easy to test. It can inform a user when a measured value crosses a chosen limit, but it has important limitations. A fixed threshold may not suit different operating loads; a digital vibration signal does not quantify severity; and the current-sensor ADC value cannot be interpreted as amperes without calibration. A threshold alert identifies a condition under the chosen rule and does not establish that a future failure is certain.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

B. Planned Anomaly Detection

A future analytics stage could establish a baseline of normal operation using a sufficiently long and representative dataset. Rolling averages, standard deviations, rates of change and correlations between sensor channels could be calculated over time windows. Statistical limits or anomaly-detection algorithms could then flag observations that differ from the learned baseline. Such alerts should be evaluated against controlled tests and documented operating conditions before being treated as evidence of a fault.

C. Planned Predictive Modelling

Supervised machine learning would require representative labelled examples of normal operation and relevant fault conditions. Potential features could include temperature trend, vibration-event frequency, calibrated current, operating duration and motor load. The data should be separated into training and test sets in a way that avoids leakage between measurements from the same run. Evaluation could include precision, recall, false-alarm rate and detection lead time. A remaining-useful-life model would require appropriate degradation or run-to-failure data and is outside the current implementation. Table 4 summarizes the current capability and planned analytics.

Table 4: Current Capability and Planned Analytics

Function	Method / Output	Status
Temperature alert	Fixed thresholds at 40 °C and 50 °C	Implemented in prototype firmware
Vibration alert	SW-420 digital output state	Implemented in prototype firmware
Current monitoring	Raw ACS712 ADC reading	Reading transmitted; calibration required
Historical analysis	SQLite records and dashboard plots	Basic data pipeline demonstrated
Anomaly detection	Baseline or statistical / ML method	Future work; not validated
Fault prediction	Trained model evaluated on labelled events	Future work; no performance claim
Remaining useful life	Degradation/run-to-failure modelling	Outside current scope

VII. Implementation Evidence and Testing Methodology

A. Verified Implementation

During testing, the ESP32 produced sensor readings visible in the serial monitor. The server's /sensor endpoint handled an incoming POST request and returned HTTP status code 200. The /history endpoint returned saved sensor records, demonstrating that the backend could retrieve historical data. The Streamlit dashboard was opened and displayed the latest sensor values. These observations demonstrate the basic connection between the sensor node, server, database and user interface. Table 5 summarizes the verified tests.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

Table 5: Summary of Verified Tests

Test	Observed result	Interpretation
ESP32 acquisition	Readings visible in Serial Monitor	Firmware is producing sensor output
HTTP POST /sensor	HTTP 200 response	Server handled the request successfully
GET /history	Stored records returned	Historical data retrieval is working
Streamlit dashboard	Latest readings displayed	Dashboard can retrieve and present current data
Predictive model evaluation	Not performed	No AI accuracy or fault-prediction metric is claimed

B. Recommended Test Plan

Further evaluation should be performed using repeatable tests. First, sensor readings should be compared with appropriate reference instruments and the current-sensor conversion should be calibrated. Second, the motor should be run under documented safe operating conditions while readings and timestamps are recorded. Third, communication interruptions should be introduced in a controlled manner to determine whether missing packets are identified and whether recovery is reliable. Fourth, alert transitions should be checked against the configured thresholds. Finally, any future anomaly or prediction model should be evaluated on separate data not used for training.

No quantitative sensor-accuracy results, long-duration reliability figures, fault-detection metrics or AI model scores are reported in the current project evidence. These should be added only after experiments have been completed and results documented.

C. Validation Record to Complete

Table 6 lists the evaluation items that remain to be completed.

Table 6: Validation Record to Complete

Evaluation item	Reference / method	Result
Temperature measurement	Reference thermometer	[To be measured]
Humidity measurement	Reference hygrometer	[To be measured]
Current conversion	Known load / suitable reference meter	[To be calibrated]
Vibration alert	Repeatable vibration test	[To be measured]
Data completeness	Compare transmitted and stored records	[To be measured]
Dashboard update delay	Timestamped packet-to-display test	[To be measured]
Future anomaly model	Independent test dataset	[Not yet implemented]

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

VIII. Feasibility, Benefits and Potential Impact

A. Technical Feasibility

The prototype is technically feasible as a student project because it combines common sensor modules, a microcontroller and open-source software. The architecture separates device firmware from the backend and dashboard, allowing each layer to be tested independently. FastAPI provides a simple interface for data exchange, SQLite provides local storage without requiring a separate database server, and Streamlit provides a practical way to visualize measurements.

B. Operational Benefits

For an operator, recurring measurements and historical plots can make changes more visible than a single manual observation. Local LEDs and a buzzer provide immediate indications under the configured rules, while the dashboard provides a convenient view of recent and previous readings. For maintenance planning, the stored history can support later comparisons between operating runs and help identify what additional sensors or tests may be required.

These benefits are potential operational advantages of monitoring, not evidence that downtime or maintenance cost has already been reduced. Demonstrating such outcomes would require a field trial, a baseline comparison and documented maintenance or production records.

C. Economic and Deployment Considerations

A low-cost prototype can be used to explore the data pipeline before investing in industrial-grade sensors or deployment infrastructure. A realistic deployment estimate must include not only the controller and sensors, but also enclosure, wiring, power supply, calibration, installation, maintenance, networking and software support. No quantified return on investment is claimed because the current project has not measured avoided failures, saved labour or reduced downtime.

D. Potential Industrial Applications

With further calibration and validation, a similar architecture could be adapted for small motors, fans, pumps or other equipment where temperature, vibration and current are relevant condition indicators. Each application would require appropriate sensor selection, mounting, operating limits and fault-specific validation. The current demonstration should not be treated as a ready-made industrial product or as a replacement for established safety controls.

IX. Limitations

The current implementation is a monitoring prototype with fixed threshold rules; it does not contain a validated trained AI model. The ACS712 reading is a raw ADC value and is not calibrated current in amperes. The SW-420 reports a digital threshold event rather than a detailed vibration waveform. The DHT22 measures local ambient conditions and may not reflect internal motor temperature. The demonstration thresholds may not be appropriate for all operating conditions.

The tests establish basic data transmission, history retrieval and dashboard display, but do not quantify sensor accuracy, long-term reliability, detection sensitivity, false alarms or predictive performance. Wi-Fi availability and the computer-hosted backend are dependencies. Robust buffering, authentication,

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

timestamp validation, data backup and production deployment security need further work. Finally, the prototype is intended for low-voltage educational use and is not a certified industrial safety or control system.

X. Conclusion and Future Work

This paper presented an IoT-based machine-condition monitoring prototype intended as a foundation for predictive maintenance in smart factories. The ESP32 collects temperature, humidity, vibration-state and current-sensor readings and transmits them through Wi-Fi to a FastAPI service. SQLite retains historical records, while a Streamlit dashboard displays current values and historical data. Local LEDs and a buzzer provide threshold-based indications. The observed HTTP 200 response, successful history retrieval and dashboard display demonstrate that the core monitoring pipeline is operating.

The project provides a useful starting point for data-driven maintenance, but the current evidence supports monitoring rather than AI-based prediction. Future work should calibrate sensors, convert the current reading into a validated physical unit, improve timestamps and communication-loss handling, collect longer-term datasets under documented conditions, and perform controlled tests. Once representative labelled data are available, anomaly-detection or supervised machine-learning methods can be evaluated using independent test data and clearly reported metrics. Only after such validation should predictive performance or maintenance benefits be claimed.

References

[1] G. W. Vogl, B. A. Weiss, and M. Helu, "A review of diagnostic and prognostic capabilities and best practices for manufacturing," *J. Intell. Manuf.*, doi: 10.1007/s10845-016-1228-8.

[2] M. Sharp, R. Ak, and T. Hedberg, "A survey of the advancing use and development of machine learning in smart manufacturing," *J. Manuf. Syst.*, vol. 48, 2018, doi: 10.1016/j.jmsy.2018.02.004.

[3] G. G. Samatas, S. S. Moumgiakmas, and G. A. Papakostas, "Predictive maintenance—Bridging artificial intelligence and IoT," arXiv:2103.11148, 2021. [Online]. Available: https://arxiv.org/abs/2103.11148

[4] X. Xu, Y. Yu, D. Griffith, and N. Golmie, "A survey on industrial Internet of Things: A cyber-physical systems perspective," *IEEE Access*, vol. 6, 2018, doi: 10.1109/ACCESS.2018.2884906.

[5] M. Dadfarnia, M. Sharp, and J. W. Herrmann, "Comprehensive evaluations of condition monitoring-based technologies in industrial maintenance: A systematic review," *J. Manuf. Syst.*, vol. 82, 2025, doi: 10.1016/j.jmsy.2025.06.015.

[6] A. K. S. Jardine, D. Lin, and D. Banjevic, "A review on machinery diagnostics and prognostics implementing condition-based maintenance," *Mech. Syst. Signal Process.*, vol. 20, no. 7, pp. 1483–1510, 2006, doi: 10.1016/j.ymssp.2005.09.012.

[7] J. Lee, B. Bagheri, and H.-A. Kao, "A cyber-physical systems architecture for Industry 4.0-based manufacturing systems," *Manuf. Lett.*, vol. 3, pp. 18–23, 2015, doi: 10.1016/j.mfglet.2014.12.001.

[8] J. Lee, H. Davari, J. Singh, and V. Pandhare, "Industrial artificial intelligence for Industry 4.0-based manufacturing systems," *Manuf. Lett.*, vol. 18, pp. 20–23, 2018, doi: 10.1016/j.mfglet.2018.09.002.

[9] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, and R. X. Gao, "Deep learning and its applications to machine health monitoring," *Mech. Syst. Signal Process.*, vol. 115, pp. 213–237, 2019, doi: 10.1016/j.ymssp.2018.05.050.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

[10] A. Heng, S. Zhang, A. C. C. Tan, and J. Mathew, "Rotating machinery prognostics: State of the art, challenges and opportunities," *Mech. Syst. Signal Process.*, vol. 23, no. 3, pp. 724–739, 2009, doi: 10.1016/j.ymssp.2008.06.009.
[11] M. Pecht, *Prognostics and Health Management of Electronics*. Hoboken, NJ, USA: Wiley, 2008.
[12] R. K. Mobley, *An Introduction to Predictive Maintenance*, 2nd ed. Oxford, U.K.: Butterworth-Heinemann, 2002.
[13] L. Monostori, "Cyber-physical production systems: Roots, expectations and R&D challenges," *Procedia CIRP*, vol. 17, pp. 9–13, 2014, doi: 10.1016/j.procir.2014.03.115.
[14] S. Mittal, M. A. Khan, D. Romero, and T. Wuest, "Smart manufacturing: Characteristics, technologies and enabling factors," *Proc. Inst. Mech. Eng. B, J. Eng. Manuf.*, vol. 233, no. 5, pp. 1342–1361, 2019, doi: 10.1177/0954405417736547.
[15] J. Dalzochio et al., "Machine learning and reasoning for predictive maintenance in Industry 4.0: Current status and challenges," *Comput. Ind.*, vol. 123, Art. no. 103298, 2020, doi: 10.1016/j.compind.2020.103298.
[16] "Predictive maintenance and intelligent sensors in smart factory: Review," *Sensors*, vol. 21, no. 4, Art. no. 1470, 2021, doi: 10.3390/s21041470.
[17] P. Visconti et al., "Machine learning and IoT-based solutions in industrial applications for smart manufacturing: A critical review," *Future Internet*, vol. 16, no. 11, Art. no. 394, 2024, doi: 10.3390/fi16110394.
[18] P. Mallioris, E. Aivazidou, and D. Bechtsis, "Predictive maintenance in Industry 4.0: A systematic multi-sector mapping," *CIRP J. Manuf. Sci. Technol.*, vol. 50, pp. 80–103, 2024, doi: 10.1016/j.cirpj.2024.02.003.
[19] F. Ramzan and D. Reforgiato Recupero, "A literature review on enhancing predictive maintenance in smart manufacturing industries: Fostering human-technology collaboration and overcoming data scarcity limitations with advanced AI models," *Oper. Res. Forum*, vol. 6, Art. no. 181, 2025, doi: 10.1007/s43069-025-00584-0.
[20] *Condition Monitoring and Diagnostics of Machines—General Guidelines*, ISO 17359:2018, International Organization for Standardization, Geneva, Switzerland, 2018.