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/ML-Based IoT Health Monitoring and Risk Prediction Using an ESP32 Sensor Node, Random Forest Classification, and a Streamlit Dashboard

Aditya Deshmane

Department of Artificial Intelligence and Data Science Engineering
Ajeenkya DY Patil School of Engineering, Pune, India
TE, Academic Year 2026–27

Pravin Giri

Department of Artificial Intelligence and Data Science Engineering
Ajeenkya DY Patil School of Engineering, Pune, India
TE, Academic Year 2026–27

Parth Ghag

Department of Artificial Intelligence and Data Science Engineering
Ajeenkya DY Patil School of Engineering, Pune, India
TE, Academic Year 2026–27

Vishal Puranik

Department of Artificial Intelligence and Data Science Engineering
Ajeenkya DY Patil School of Engineering, Pune, India
Email : vishalpura@gmail.com

Abstract

Health problems are easier to manage when abnormal physiological trends are noticed early.. Right now routine monitoring still depends mostly on periodic manual checks. These checks happen at set times rely on the person doing them and are hard to use in homes or remote areas. This paper introduces Smart Health Monitor, a low-cost prototype that brings sensing, machine learning and visualization together in one process. An ESP32 microcontroller collects data from sensors and sends it through a serial link to a Python application. That application cleans the data loads a Random Forest classifier that was trained before and gives a health-risk class: low, medium or high. A Streamlit dashboard shows the real-time readings along with the predicted risk level. The paper looks at work on IoT-based healthcare, wearable monitoring and machine learning models that predict health risks. It points out the gap between showing

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)

raw values and offering affordable real-time analysis. It also outlines a method for training models comparing them and validating results. The prototype is meant as a decision-support tool and a demonstration system. It is not a certified device. No performance numbers are given here. All numerical results will be shared after the experiments, in Section VIII are completed.

Keywords: Internet of Things, ESP32, Health Monitoring, Machine Learning, Random Forest, Risk Prediction, Streamlit, Edge Sensing

I. Introduction

Health risks that come from living are going up. Because of that we need to check a persons health often than a single visit to a clinic can give us. A nurse or caregiver who writes down readings by hand can only record one instant. That method is slow different people may record it differently. It does not work well when many people need to be monitored or when they live far from a hospital.

Two technologies have made a new way possible. Cheap sensors that connect to a network can now. Send body signals all the time. Machine-learning models can look at those signals. Find patterns that a human eye might miss. Surveys of the Internet of Things or IoT in healthcare explain how sensing, communication and cloud or local processing together make it possible to observe patients from a distance and all the time [1], [2]. Reviews of devices show that measuring vital signs has become small and cheap [3].

A common weakness still exists. Many cheap devices only show numbers and the user has to decide if a number is a problem. Systems that add analysis are usually linked to hospital systems or private platforms. Therefore a simple system made by students or at home that connects sensors directly to a trained classifier and shows clear results is a good engineering goal.

This paper talks about Smart Health Monitor, a prototype built around that idea. An ESP32 sensor node collects data a Python backend runs a Random Forest model and a Streamlit interface shows the result. The rest of the paper explains the problem reviews the related literature describes the design and machine-learning process and shows how the system should be tested before making any claims, about accuracy.

II. Motivation and Objectives

The motivation starts with an idea: a reading is only helpful if someone understands it in time. A never-ending flow of numbers doesn't help a person who doesn't have the training or the focus to read it. A system that turns the flow into a few to-understand risk groups reduces that problem. The work is organized around four objectives:

17. Collect physiological and environmental readings with an ESP32 microcontroller and attached sensors.
18. Transfer the readings reliably to a Python application for processing.
19. Process the data and produce a machine-learning-based health-risk prediction.
20. Present live readings and predictions through an interactive Streamlit dashboard.

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 aim is assistance, not replacement of medical judgement. The output should indicate when closer attention or professional consultation may be appropriate.

III. Technical Foundations

A. Sensing and the ESP32 Edge Node

The ESP32 is a low-cost microcontroller with built-in wireless capability and enough input pins to read several sensors [11]. In this prototype it acts as the edge device: it samples the attached sensors at fixed intervals and sends the values to a host computer through a USB serial connection. Keeping the node simple limits power use and cost, which matters for any system meant for home or small-scale use. General surveys of IoT architectures stress that the sensing layer, the transport layer, and the application layer should be kept clearly separate so that each can be replaced independently [8].

B. Serial Transfer and Data Handling

The host application reads the serial stream, parses each line into numeric fields, and stores the values in tabular form. Pandas and NumPy are used for this stage. Parsing must tolerate partial lines, dropped characters, and sensor spikes, because a wired link can still deliver malformed records.

C. Random Forest Classification

A Random Forest builds many decision trees, each fitted to a random sample of the training data, and considers only a random subset of features at every split [9]. Each tree votes for a class, and the most frequent vote becomes the prediction. The method suits this project for four reasons: it handles non-linear relationships between sensor values, it is comparatively resistant to overfitting, it is light enough for near-real-time prediction on ordinary hardware, and it provides a feature-importance ranking. The Scikit-learn library supplies a maintained implementation [10].

D. Model Persistence and Dashboard

After training, the model is saved with Joblib as health_model.pkl, so the dashboard can load it without retraining. Streamlit is used to build the interface because it allows a data application to be written in Python alone [12]. The dashboard therefore shares one language and one environment with the processing code.

IV. Literature Review

Research on IoT-enabled healthcare is broad. Islam et al. surveyed network architectures, platforms, applications, and open issues in IoT-based care, and highlighted security and interoperability as persistent concerns [1]. Baker et al. examined technologies, challenges, and opportunities for smart healthcare and proposed a general architecture connecting sensors, gateways, and cloud services [2].

Wearable monitoring has been reviewed separately. Dias and Cunha discussed devices and systems that measure vital signs and noted that the main practical questions concern comfort, accuracy, and data handling rather than sensing alone [3].

Edge-oriented designs aim to reduce delay and bandwidth use. Pathinarupothi et al. described an IoT-based smart edge for remote monitoring that estimates severity and sends alerts, illustrating how some analysis can be done close to the patient [4].

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)

Machine learning for risk or disease prediction has been studied on healthcare data. Chen et al. applied learning methods to large healthcare datasets for disease-risk prediction [5]. Esteva et al. presented an overview of deep learning in medicine, covering both the promise and the obstacles to clinical use [6]. An important caution comes from Rudin, who argues that for high-stakes decisions, models whose reasoning can be inspected are preferable to opaque ones [7]. This supports the choice of a tree-based model whose feature importance can be examined.

Taken together, the literature shows strong progress on sensing, on architectures, and on prediction, but these threads are often discussed separately. Few accessible designs combine all three in a compact prototype.

V. Literature Comparison and Research Gap

Table I summarizes how the reviewed work relates to the present project.

Table I: Comparison of Representative Studies

Study	Main Focus	Contribution	Limitation / Gap
Islam et al. [1]	IoT in healthcare	Survey of architectures, platforms, and applications	Broad overview; no concrete low-cost prototype
Baker et al. [2]	Smart healthcare	Technologies, challenges, and a generic architecture	Architecture-level; limited implementation detail
Dias and Cunha [3]	Wearable devices	Review of vital-sign monitoring systems	Emphasis on measurement rather than risk analysis
Pathinarupothi et al. [4]	Edge monitoring	Severity detection and alerts near the patient	Focused on specific deployment settings
Chen et al. [5]	Disease prediction	Learning over healthcare data	Relies on large institutional datasets
Esteva et al. [6]	Deep learning in medicine	Overview of opportunities and challenges	General guide; not a monitoring system

Four gaps follow from this comparison.

21. **Raw display versus interpretation:** many affordable devices show values without any risk assessment.

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)

22. **Affordable real-time prediction:** few low-cost designs connect live sensing directly to a trained classifier.
23. **Lightweight models at the edge:** the use of compact models for real-time classification of locally collected data is still limited.
24. **Validation:** prototypes are often demonstrated without a defined evaluation procedure, which makes their reliability unclear.

VI. Proposed System

The proposed system has five stages, shown in Fig. 1: sensors, ESP32, Python processing, ML model, and Streamlit dashboard. The ESP32 acts as the edge device, while the Python application handles processing, prediction, and visualization.

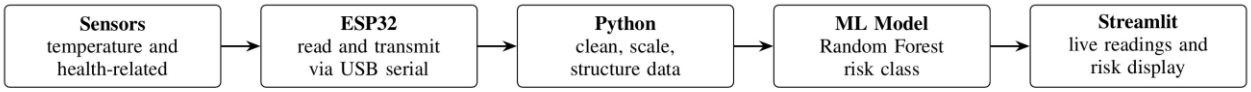


Fig. 1. Proposed Smart Health Monitor architecture, from physical sensing to the user dashboard.

A. Hardware and Software

The hardware consists of an ESP32 development board, temperature and other health-related sensors, a USB serial connection, and the required wiring and power supply. The software stack consists of the Arduino IDE for programming the board, Python for integration, Pandas and NumPy for data handling, Scikit-learn for modelling, Joblib for storing the trained model, and Streamlit for the interface.

B. Data Flow

Each sensor reading is time-stamped on arrival so that the dashboard can show trends and so that stored records can later be used for retraining. The Python application passes each cleaned record to the loaded model, receives a class label, and sends both the reading and the label to the display.

VII. Machine-Learning Workflow

The model passes through five steps: data, preprocessing, training, saving, and prediction.

25. **Data:** readings from the ESP32 (temperature and heart or other sensor values) are combined with a labelled health dataset.
26. **Preprocessing:** missing and noisy values are removed, features are scaled, and the data are divided into training and test sets.
27. **Training:** a Random Forest classifier is fitted using Scikit-learn and its accuracy is checked on the held-out set.
28. **Saving:** the fitted model is exported with Joblib as health_model.pkl.
29. **Prediction:** the application loads the stored model and classifies incoming readings as low, medium, or high risk.

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)

Let **x** be the feature vector of one reading and let T be the number of trees. If tree t outputs class $h_t(\mathbf{x})$, the forest prediction is the majority vote

$$\hat{y}(\mathbf{x}) = \arg\max_c \sum_{t=1}^{T} \mathbb{1}\,[\,h_t(\mathbf{x}) = c\,], \tag{1}$$

where c ranges over the risk classes. A reading on which most trees choose the low-risk class is displayed as low risk.

VIII. Experimental Methodology

The experimental study should show whether the classifier is dependable enough to be useful.

A. Dataset Preparation

Training data may come from a documented public health dataset combined with readings collected through the ESP32 node. The data should be divided into training, validation, and test subsets, and the test subset must not be used during model tuning. Each record should store the sensor values, the observation time, and the assigned risk label, along with a note on how the label was obtained.

B. Model Comparison

The Random Forest should be compared with simpler baselines, for example logistic regression, a single decision tree, and a support vector machine, using the same split. Cross-validation should be used when the dataset is small. Hyperparameters such as the number of trees and the maximum depth should be selected on the validation data only.

C. Evaluation Metrics

Accuracy, precision, recall, and F1-score should be reported for each class, together with a confusion matrix. Accuracy alone can hide poor behaviour when the classes are unbalanced. Recall for the high-risk class deserves special attention, because labelling a risky reading as safe is more harmful than raising a false alarm. The study should also record the time taken to produce one prediction and the size of the stored model, since these determine whether the system works smoothly in real time.

IX. Results Analysis and Discussion

At the time of writing, the prototype demonstrates the complete pipeline from sensing to display. The values visible in the demonstration dashboard are illustrative and do not constitute measured performance. Table II is therefore given as a reporting template, to be completed only with figures produced by the experiments above.

Table II: Template for Reporting Measured Results

Model	Acc.	Prec.	Recall	F1
Logistic regression	–	–	–	–

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)

Model	Acc.	Prec.	Recall	F1
Decision tree	–	–	–	–
SVM	–	–	–	–
Random Forest	–	–	–	–

Once the table is filled, the discussion should examine whether the Random Forest improves on the baselines for every class or only for some, which features carry most weight in its feature-importance ranking, and how often high-risk readings are missed. Cases in which a single noisy sensor value flips the predicted class should be studied separately, as they show how sensitive the system is to measurement error.

No performance value should be quoted until the corresponding experiment has been run.

X. Applications and Limitations

The system may be applied to home monitoring, remote or rural observation, student and laboratory demonstrations, and as a base for a fuller telehealth service. Its low component cost makes it accessible for small-scale use.

Several limitations apply. The prediction is only as reliable as the training data and the labels attached to it. Low-cost sensors can drift, be badly positioned, or fail, and readings may be affected by movement and ambient conditions. A USB serial link restricts the distance between the node and the host. The system has not been clinically validated and must not be used to diagnose or treat any condition. Any user concerned about their health should consult a qualified professional.

XI. Future Scope

The prototype can be extended in several directions: adding more physiological sensors and patient parameters, storing data in the cloud for long-term monitoring, supporting mobile and remote access, and training on larger real-world datasets. An alert mechanism for abnormal readings is another natural addition. Replacing the serial link with the wireless capability of the ESP32 would remove the distance restriction. Finally, explanations of each prediction, such as showing which sensor value contributed most, would make the output easier to trust [7].

XII. Conclusion

This paper described Smart Health Monitor, a low-cost IoT and machine-learning prototype in which an ESP32 node collects sensor readings, a Python application applies a Random Forest classifier, and a Streamlit dashboard shows live values with a risk class. The reviewed literature shows mature work on healthcare IoT, wearable sensing, and prediction, but few compact designs that join the three in an affordable, understandable workflow. The proposed methodology specifies how the model should be trained, compared against baselines, and evaluated with attention to missed high-risk cases. The value

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)

of the approach will be settled by the measured results and by validation on larger, real-world data. Until then, the system should be regarded as a demonstration of a complete sensing-to-decision pipeline and not as a medical instrument.

References

[1] S. M. R. Islam, D. Kwak, M. H. Kabir, M. Hossain, and K.-S. Kwak, "The Internet of Things for health care: A comprehensive survey," IEEE Access, vol. 3, pp. 678–708, 2015.

[2] S. B. Baker, W. Xiang, and I. Atkinson, "Internet of Things for smart healthcare: Technologies, challenges, and opportunities," IEEE Access, vol. 5, pp. 26521–26544, 2017.

[3] D. Dias and J. Paulo Silva Cunha, "Wearable health devices—vital sign monitoring, systems and technologies," Sensors, vol. 18, no. 8, Art. no. 2414, 2018.

[4] R. K. Pathinarupothi, P. Durga, and E. S. Rangan, "IoT-based smart edge for global health: Remote monitoring with severity detection and alerts transmission," IEEE Internet of Things Journal, vol. 6, no. 2, pp. 2449–2462, 2019.

[5] M. Chen, Y. Hao, K. Hwang, L. Wang, and L. Wang, "Disease prediction by machine learning over big data from healthcare communities," IEEE Access, vol. 5, pp. 8869–8879, 2017.

[6] A. Esteva et al., "A guide to deep learning in healthcare," Nature Medicine, vol. 25, pp. 24–29, 2019.

[7] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," Nature Machine Intelligence, vol. 1, pp. 206–215, 2019.

[8] P. P. Ray, "A survey on Internet of Things architectures," Journal of King Saud University – Computer and Information Sciences, vol. 30, no. 3, pp. 291–319, 2018.

[9] L. Breiman, "Random forests," Machine Learning, vol. 45, no. 1, pp. 5–32, 2001.

[10] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[11] Espressif Systems, "ESP32 Technical Reference Manual," official documentation. [Online]. Available: https://www.espressif.com

[12] Streamlit Inc., "Streamlit documentation," official documentation. [Online]. Available: https://docs.streamlit.io