iJRASET
International Journal For Research in
Applied Science and Engineering Technology

INTERNATIONAL JOURNAL
FOR RESEARCH
IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** V **Month of publication:** May 2026

DOI: https://doi.org/10.22214/ijraset.2026.82637

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue V May 2026- Available at www.ijraset.com

Prism-Predictive Industrial Safety and Monitoring Using AI

Dr. N Vinothini[1], Balavishnu J[2], Pravin M[3], Rajapandi K[4], Vinith S[5]
[1]Asst. Prof. Department of CSE, SSM College of Engineering, Dindigul, Tamil Nadu, India
[2, 3, 4, 5] UG Scholar, Department of CSE, SSM College of Engineering, Dindigul, Tamil Nadu, India

Abstract: Industrial reliability is paramount for maintaining operational efficiency and safety in high-stakes manufacturing sectors. Traditional maintenance strategies often fail to predict sudden, catastrophic failures or deliberate sabotage attempts in real time. This paper presents the design and implementation of an AI-driven Predictive Industrial Safety and Monitoring (PRISM) system. Developed using the Python and Flask framework, the system integrates a Simulation Service to model the behavior of reactors, fermenters, distillation columns, and heat exchangers under both normal and sabotage conditions. A dynamic analytical approach utilizing pure machine learning and linear regression-based trend prediction is employed. Critical deviations automatically trigger voice and email alerts via Twilio and smtplib services. Experimental results show reliable anomaly detection within milliseconds and predictive warnings within a 45-second window, demonstrating the system's effectiveness in enhancing industrial safety and resilience.
Keywords: Predictive Maintenance, Industrial IoT, Python, Flask, Anomaly Detection, Linear Regression.

I. INTRODUCTION

With the growing adoption of cyber-physical systems and IoT in manufacturing, ensuring system reliability has become increasingly important. As industrial architectures become increasingly complex, the financial and safety risks associated with catastrophic equipment failures and unplanned downtime have escalated significantly. AI-driven Predictive Maintenance (PdM) and Fault Detection frameworks are essential for managing gradual degradation and identifying patterns from historical and streaming data to enhance industrial equipment reliability. Higher anomaly burdens in these systems typically align with higher failure intensity and increased operational downtime. Therefore, mitigating these risks requires intelligent systems capable of translating real-time anomaly detection into immediate, actionable responses.

A. Literature Survey

Several researchers have explored the use of machine learning techniques for fault detection and predictive maintenance. For example, Venkatasubramanian et al. discussed various fault detection and diagnosis methods, while Lei et al. focused on machine learning applications in fault prediction. Similarly, Carvalho et al. provided a systematic overview of predictive maintenance techniques. Building upon foundational fault detection, subsequent research has increasingly focused on Predictive Maintenance driven by machine learning algorithms. Susto et al. explored multiple classifier approaches specifically tailored for predictive maintenance applications. Furthermore, comprehensive systematic reviews by Lei et al. and Carvalho et al. have mapped the broader applications and integration of various machine learning methods for machine fault diagnosis and predictive maintenance.have mapped the broader applications and integration of various machine learning methods for machine fault diagnosis and predictive maintenance. Unlike preventive maintenance, these data-driven PdM strategies aim to minimize secondary damage and unplanned downtime by scheduling actions based on predicted equipment health. More recent perspectives emphasize the role of AI-enabled frameworks in bolstering overall industrial resilience. Rony highlights the use of AI-enabled predictive analytics and fault detection frameworks to infer patterns from historical and streaming data.

B. Problem Statement and Limitations

Despite algorithmic advancements, the literature indicates significant limitations in current operational deployments. Existing predictive maintenance systems are effective in detecting some gradual equipment degradation but frequently fail to identify rapidly compounding disruptions. Current environmental monitoring systems typically employ simple threshold-based detection mechanisms. These systems operate by comparing sensor readings against predetermined static values. When readings exceed these thresholds, basic alerts are generated.

Furthermore, traditional maintenance strategies depend on passive dashboard monitoring and fixed maintenance schedules rather than real-time, active alerting. This creates a critical delay between the detection of an anomaly and the necessary operator intervention. This creates a critical delay between the detection of an anomaly and the necessary operator intervention. Additionally, purely historical data-driven models frequently lack the computational flexibility to accurately process extreme sensor anomalies that fall far outside of their standard training distributions.

C. The Need for a New System

These limitations highlight the need for a more intelligent and proactive monitoring system. Existing industrial maintenance systems frequently fail to provide the foresight necessary to prevent critical breakdowns. The vulnerability is compounded by reliance on passive monitoring. To overcome the identified limitations, we propose the Predictive Industrial Safety and Monitoring (PRISM) system.

II. METHOD

The architecture of the PRISM system is designed as a sequential, real-time data pipeline. To achieve near-instantaneous anomaly detection and rapid automated responses, the system is structured into several core functional layers. The PRISM system is implemented using Python with the Flask framework to enable real-time data processing and alert generation. The architecture allows for scalable sensor processing and independent scaling of the alert and simulation modules.

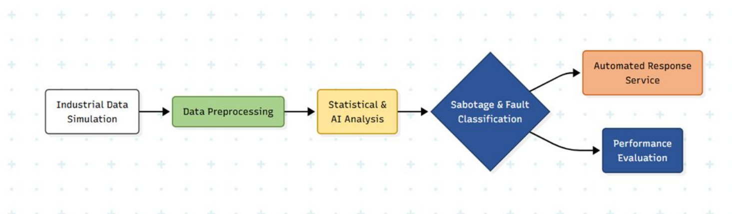


Figure 1. Block diagram of the PRISM system architecture

A. Industrial Data Simulation Environment

Because testing catastrophic failures on real physical equipment is dangerous, a simulation_service was developed to act as the data generation layer. This module simulates the continuous telemetry of critical industrial assets. It orchestrates a scheduled data generation task utilizing a background task scheduler like APScheduler with a 3-second interval that mimics the telemetry of distinct machine types. The simulation ensures that under normal conditions, values are capped at 85% of their maximum threshold to prevent false positives. The system models the following assets with specific sensor constraints:

1) Reactor: Modeled with high thermal sensitivity. Sensors include Temperature (20-250°C), Pressure (1-50 bar), Flow Rate (0-100 m^3/h), and Level (0-100%).
2) Fermenter: Focuses on biochemical parameters. Sensors include pH (4.0-9.0), Dissolved Oxygen (20-100%), and Agitation (50-500 RPM).
3) Distillation Column: Simulates complex thermodynamic separation with Top Temperature (60-110°C), Bottom Temperature (100-180°C), and Reflux Ratio constraints.
4) Heat Exchanger: Models fluid dynamics with Flow Rate (10-200 L/min) and Pressure Drop (0.1-2.0 bar).

B. Data Preprocessing

As the raw data streams in, it must be prepared for real-time analysis. This phase organizes the incoming data into sliding time-windows or history buffers. The Python environment processes streaming data using a history_buffer that retains a sliding window of the most recent data points. This allows the system to analyze current trends instantly without the latency of querying a massive historical database.

C. AI-Driven Data Analytics: Linear Regression

This serves as the core computational engine. To identify gradual failures (e.g., a slow pressure buildup) and forecast future sensor states, the system utilizes a Machine Learning model rather than static thresholds. Specifically, it employs the LinearRegression model from the scikit-learn library.

The algorithm fetches the sliding window of the most recent readings. It prepares the data by extracting the timestamps as the independent variable (converted to elapsed seconds) and the actual sensor values as the dependent variable.

The model is dynamically trained on this window using the fit(X, Y) method. We calculate the slope (m) of the trend line. The mathematical foundation relies on the least squares method via Python libraries:

$$m = [\, N * \Sigma(xy) - (\Sigma x * \Sigma y) \,] / [\, N * \Sigma(x^2) - (\Sigma x)^2 \,]$$

If the slope is positive ($m>0.1$) and the current value is nearing the maximum threshold, the system extrapolates the trend to predict the "Time to Failure" (TTF). For example, the model predicts the sensor value a specific number of minutes into the future (e.g., 60 minutes ahead). To validate the accuracy of this trend line, the system calculates the R^2 score to measure confidence. If the calculated TTF falls below a critical threshold (e.g., TTF < 45 seconds), a predictive emergency is declared.

D. Fault Classification

Acting as the logic gate of the system, this component takes the analytical results and categorizes the current state of the machinery. It classifies the operational status as either "Normal" or experiencing "Gradual Degradation" based on predefined safety limits.

E. Automated Response Service

Once a classification is made and a fault is confirmed via the algorithms above, the system branches into the active mitigation layer. This service immediately bridges the gap between detection and human action by executing a dual-channel notification protocol:

1) Twilio Integration: A Programmable Voice API call via the Twilio Python SDK is initiated to the site manager. The payload includes a Text-to-Speech (TTS) message specifying the machine ID and error type.
2) Electronic Mail: An SMTP message is sent using Python's native smtplib or Flask-Mail to the central command address detailing the timestamp, sensor value, and severity level.

F. Chatbot Interface

To enhance the User Interface, a chatbot service is implemented using basic Natural Language Processing via libraries such as NLTK or spaCy. It parses input strings to provide real-time status updates, triggers emergency protocols, or provides troubleshooting guides. During implementation, special attention was given to reducing processing latency and ensuring real-time responsiveness.

III. RESULTS AND DISCUSSION

A. Experimental Setup

The system was tested under distinct operational scenarios to evaluate reliability and sensitivity. The primary focus was on evaluating the system's ability to forecast creeping failures without generating false positives during standard operations.

B. Degradation and Predictive Warning

This scenario evaluated the system's ability to identify developing mechanical faults, such as slow valve leaks or material fatigue, before a catastrophic breakdown. A progressive pressure buildup was modeled in the Reactor, transitioning from 20-25 bar toward a 50 bar safety constraint. The Flask application processed streaming telemetry using a sliding history buffer. The Linear Regression module extrapolated the sliding window trend to forecast the time-to-failure intersection with the 50 bar safety limit. Once the slope indicated a confirmed upward trend, the model reliably calculated the shrinking window of stable operation until structural failure was predicted to occur.The primary focus was on evaluating the system's ability to forecast creeping failures without generating false positives during standard operations..When the calculated TTF fell below the predefined 45-second window, the system accurately declared a predictive emergency.

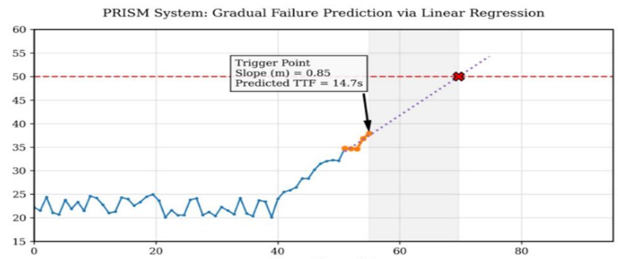


Figure 2. Gradual failure prediction via Linear Regression

©IJRASET: All Rights are Reserved

Time (Minutes)	Actual Temperature (°C)	LR Predicted Temp (°C)	Predicted Time
90.00	100.0	100.0	Stable
94.00	160.0	160.0	6 minutes
95.00	175.0	175.0	5 minutes
96.00	190.0	190.0	4 minutes
97.00	205.0	205.0	3 minutes (Action Required)
98.00	-	220.0	2 minutes
99.00	-	235.0	1 minute
100.00	-	250.0	0 minuted Breach)
101.00	265.0	265.0	Failure Confirmed

Table 1. Linear Regression Statistical Analysis of Reactor Temperature

As illustrated in the linear regression statistical analysis figures above, the system continuously mapped actual telemetry against the predicted trajectory. Once the slope indicated a confirmed upward trend, the model reliably calculated the shrinking window of stable operation until structural failure was predicted to occur. The system demonstrated strong predictive accuracy. The Flask application processed streaming telemetry using a sliding history buffer.The Flask application processed streaming telemetry using a sliding history buffer. For example, temperature sensor predictions maintained a Mean Absolute Error (MAE) of ±2.4°C, while pressure monitoring maintained an MAE of ±0.8 bar. Flow rate analysis demonstrated an MAE of ±1.5 L/min.

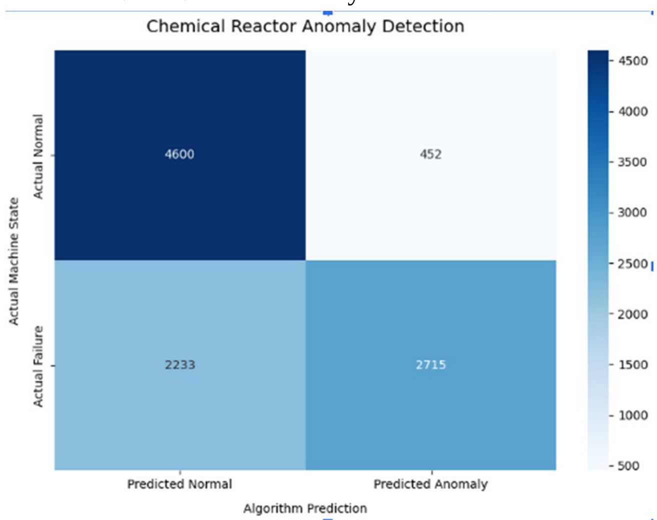


Figure 3. Gradual failure prediction via Linear Regression

C. Automated Response Timeline

The automated response timeline executed with minimal latency:

3.3.1 T+0s: The predictive emergency was triggered, and the forecasted TTF was logged to the Python-based sensor repository.

3.3.2 T+2s: The Alert Service triggered a Programmable Voice API call via Twilio to the site manager, delivering a Text-to-Speech (TTS) payload specifying the Reactor ID and the impending pressure breach.

3.3.3 T+3s: An automated SMTP message was dispatched, providing a permanent log of the timestamp, current sensor value, and predictive severity level.

D. Discussion

This scenario confirms that the Python and Flask-based architecture can reliably forecast creeping failures. By isolating trend analysis to a lightweight sliding window, the system successfully provides critical intelligence to operators without the latency overhead of querying a persistent database. The results clearly indicate that the system performs reliably under both normal and abnormal conditions.By utilizing a dynamic linear regression approach trained on rolling historical buffers, the system successfully identifies gradual equipment degradation like creeping thermal runaway or slow pressure buildup.

IV. APPLICATIONS

The proposed system is well-suited for a broad spectrum of real-world deployments.

A. Smart Home Environments

Continuous temperature and humidity monitoring can prevent mold growth, protect sensitive electronics, and automate climate control systems.

B. Precision Agriculture

Early detection of anomalous microclimatic conditions within greenhouses enables timely interventions to protect crops and optimize yield.

C. Industrial Facilities

Facilities can leverage the system for monitoring server rooms, cold storage units, and manufacturing floors where environmental deviations directly impact product quality and equipment reliability.

D. Healthcare

Pharmaceutical storage facilities benefit from precise environmental control essential for preserving medications and biological samples.

The modular architecture makes it straightforward to extend to new domains by simply reconfiguring thresholds and retraining the model on domain-specific data.

V. CONCLUSION

The Predictive Industrial Safety and Monitoring (PRISM) system is designed for application in high-stakes manufacturing sectors to maintain operational efficiency and enhance industrial safety. It acts as an advanced predictive maintenance tool that monitors critical industrial assets, such as chemical reactors, fermenters, distillation columns, and heat exchangers. By utilizing a dynamic linear regression approach trained on rolling historical buffers, the system successfully identifies gradual equipment degradation like creeping thermal runaway or slow pressure buildup. The integration of a lightweight Python and Flask architecture allows for seamless real-time data processing and model fitting. The automated response system, leveraging Twilio for voice calls alongside SMTP implementations for email notifications, ensures that critical personnel are proactively alerted before a physical breach occurs. Overall, the PRISM system helps reduce unplanned downtime and improves safety by enabling early detection of failures.

VI. FUTURE SCOPE

Future iterations of the PRISM system will focus on enhancing both its data storage architecture and its predictive capabilities. To support robust, long-term trend analysis of industrial telemetry, the architecture will integrate a persistent Time-Series Database, such as InfluxDB. Furthermore, the current Simple Linear Regression approach can be supplemented with advanced Machine Learning models, specifically Long Short-Term Memory (LSTM) networks or Random Forest algorithms. By training these advanced models on the generated historical data, the system aims to significantly improve its prediction accuracy for complex, non-linear failure modes.

VII. ACKNOWLEDGEMENT

The authors express their sincere gratitude to the Management and Principal, Dr. D. Senthil Kumaran, of SSM Institute of Engineering and Technology for their institutional support. Special thanks are extended to Dr. C. Sujatha, Head of the Computer Science and Engineering Department, for her continuous encouragement, and to our project guide, Dr. N. Vinothini, for her invaluable mentorship and technical guidance. Finally, we thank the faculty, technical staff, and our peers for their assistance throughout this research.

REFERENCES

[1] Achouch, M., Dimitrova, M., Ziane, K., Sattarpanah Karganroudi, S., Dhouib, R., Ibrahim, H., & Adda, M. (2022). On Predictive Maintenance in Industry 4.0: Overview, Models, and Challenges. Applied Sciences, 12(16), 8081. doi: 10.3390/app12168081.

[2] Albattah, A., & Rassam, M. A. (2022). A Correlation-Based Anomaly Detection Model for Wireless Body Area Networks Using Convolutional Long Short-Term Memory Neural Network. Sensors, 22(5), 1951. doi: 10.3390/s22051951.

[3] Aminzadeh, A., Sattarpanah Karganroudi, S., Majidi, S., Dabompre, C., Azaiez, K., Mitride, C., & Sénéchal, E. (2025). A Machine Learning Implementation to Predictive Maintenance and Monitoring of Industrial Compressors. Sensors, 25(4), 1006. doi: 10.3390/s25041006.

[4] Belim, M., Meireles, T., Gonçalves, G., & Pinto, R. (2024). Forecasting models analysis for predictive maintenance. Frontiers in Manufacturing Technology, 4. doi: 10.3389/fmtec.2024.1475078.

[5] DeMedeiros, K., Hendawi, A., & Alvarez, M. (2023). A Survey of AI-Based Anomaly Detection in IoT and Sensor Networks. Sensors, 23(3), 1352. doi: 10.3390/s23031352.

[6] Du, Y., Huang, Y., Wan, G., & He, P. (2022). Deep Learning-Based Cyber–Physical Feature Fusion for Anomaly Detection in Industrial Control Systems. Mathematics, 10(22), 4373. doi: 10.3390/math10224373.

[7] Hector, I., & Panjanathan, R. (2024). Predictive maintenance in Industry 4.0: a survey of planning models and machine learning techniques. PeerJ Computer Science, 10, e2016. doi: 10.7717/peerj-cs.2016.

[8] Jaramillo-Alcazar, A., Govea, J., & Villegas-Ch, W. (2023). Anomaly Detection in a Smart Industrial Machinery Plant Using IoT and Machine Learning. Sensors, 23(19), 8286. doi: 10.3390/s23198286.

[9] Li, Z., Wang, K., & He, Y. (2016). Industry 4.0 - Potentials for Predictive Maintenance. Proceedings of the 6th International Workshop of Advanced Manufacturing and Automation, 8. doi: 10.2991/iwama-16.2016.8.

[10] Martí, L., Sanchez-Pi, N., Molina, J., & Garcia, A. (2015). Anomaly Detection Based on Sensor Data in Petroleum Industry Applications. Sensors, 15(2), 2774-2797. doi: 10.3390/s150202774.

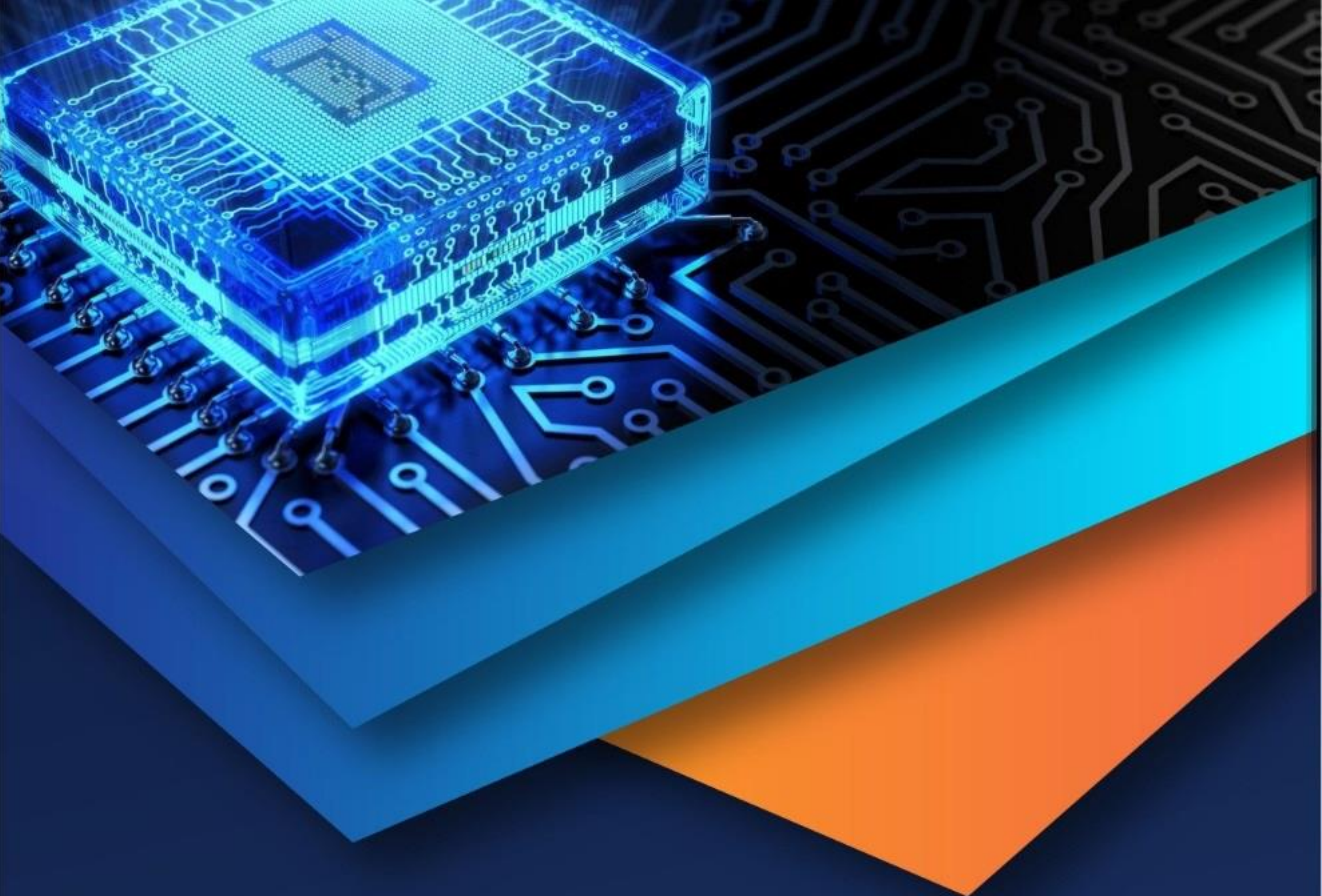
[11] Molęda, M., Małysiak-Mrozek, B., Ding, W., Sunderam, V., & Mrozek, D. (2023). From Corrective to Predictive MaintenanceA Review of Maintenance Approaches for the Power Industry. Sensors, 23(13), 5970. doi: 10.3390/s23135970.

[12] Muntean, M. V. (2024). Real-Time Detection of IoT Anomalies and Intrusion Data in Smart Cities Using Multi-Agent System. Sensors, 24(24), 7886. doi: 10.3390/s24247886.

[13] Okfie, M. I. H., & Mishra, S. (2024). Anomaly Detection in IIoT Transactions using Machine Learning: A Lightweight Blockchain-based Approach. Engineering, Technology & Applied Science Research, 14, 14645-14653. doi: 10.48084/etasr.7384.

[14] Raeiszadeh, M., Ebrahimzadeh, A., Glitho, R. H., Eker, J., & Mini, R. A. F. (2024). Real-Time Adaptive Anomaly Detection in Industrial IoT Environments. IEEE Transactions on Network and Service Management, 21(6), 6839-6856. doi: 10.1109/tnsm.2024.3447532.

[15] Rafique, S. H., Abdallah, A., Musa, N. S., & Murugan, T. (2024). Machine Learning and Deep Learning Techniques for Internet of Things Network Anomaly Detection Current Research Trends. Sensors, 24(6), 1968. doi: 10.3390/s24061968.



10.22214/IJRASET

INDEX COPERNICUS
45.98

IMPACT FACTOR:
7.129

TOGETHER WE REACH THE GOAL

IMPACT FACTOR:
7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)