IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

MiDAS : Real-Time Machine Insights and Data-Driven Maintenance System

Prof. Sweta Wankhade[1], Kimberly Anthony[2], Siddhi Auti[3], Manjiri Chaudhary[4], David Verma[5]

[1]Professor, Department of Artificial Intelligence and Data Science,Ajeenkya D.Y Patil School of Engineering, Pune

[2]Student, Department of Artificial Intelligence and Data Science,Ajeenkya D.Y Patil School of Engineering, Pune

[3]Student, Department of Artificial Intelligence and Data Science,Ajeenkya D.Y Patil School of Engineering, Pune

[4]Student, Department of Artificial Intelligence and Data Science,Ajeenkya D.Y Patil School of Engineering, Pune

[5]Student, Department of Artificial Intelligence and Data Science,Ajeenkya D.Y Patil School of Engineering, Pune

Abstract: MiDAS is a cost-effective and scalable automation platform developed to enhance the operational efficiency of small and medium-sized manufacturing enterprises (SMEs). Many SMEs still rely on manual methods for monitoring machine performance, which often leads to delayed maintenance, unexpected equipment failures, and suboptimal production efficiency. Although conventional industrial automation solutions such as PLC and SCADA systems provide robust capabilities, their high cost and implementation complexity make them less accessible to smaller industries.

To address this gap, MiDAS integrates affordable Internet of Things (IoT) technologies with practical industrial monitoring needs. The system utilizes low-cost hardware components, including ESP32 microcontrollers, sensors, and a Raspberry Pi functioning as a local server. It captures real-time machine parameters such as punch count, vibration, and energy consumption, and transmits the data wirelessly using the MQTT protocol. The collected data is processed using analytical techniques such as aggregation, threshold-based anomaly detection, moving averages, and linear regression to generate predictive maintenance insights. A web-based dashboard, developed using React, provides real-time visualization of machine status, health metrics, and predictive alerts.

By enabling continuous monitoring, early fault detection, and data-driven decision-making at a significantly lower cost than traditional systems, MiDAS empowers SMEs to reduce downtime, improve productivity, and adopt Industry 4.0 practices effectively.

Keywords - Predictive maintenance, IIoT, ESP32, Raspberry Pi 4, proximity sensor, edge computing, real time monitoring, machine health, production output, maintenance notifications, industrial automation.

I. INTRODUCTION

In the current era of rapid industrial transformation, small and medium-scale manufacturing enterprises face significant challenges in adopting modern automation technologies. Conventional solutions such as PLC–SCADA systems, while effective, often involve high implementation costs, extended deployment timelines, and considerable technical complexity. As a result, many manufacturing units continue to rely on manual or semi-digital methods, including spreadsheet-based or paper-based record-keeping. These

practices limit real-time visibility, delay fault detection, and increase the likelihood of unplanned machine downtime, ultimately affecting productivity and competitiveness.

The proposed Mini Distributed Automation System (MiDAS) aims to address these challenges by providing a cost-effective and scalable Industrial Internet of Things (IIoT) solution tailored for small and medium enterprises. The system is envisioned to utilize affordable hardware components such as ESP32 microcontrollers, Raspberry Pi-based edge devices, and industrial sensors for monitoring machine activity and operational conditions.

At its core, the framework is designed to enable continuous data acquisition from machines, capturing parameters such as operational counts, vibration levels, and temperature variations. This data may be transmitted using lightweight communication protocols such as MQTT to a centralized edge or cloud-based system for further processing. Analytical routines can be applied to extract meaningful insights, which are then stored in a structured database for both real-time monitoring and historical analysis.

A web-based dashboard interface is proposed to present the processed data in an intuitive and accessible manner. Through this interface, operators and supervisors can monitor machine performance, analyze trends, and receive early alerts for maintenance requirements, thereby supporting proactive decision-making.

The MiDAS framework integrates multiple technological domains, including embedded systems, IoT communication, data analytics, and web application development. Its modular design allows flexible deployment and scalability, enabling manufacturers to adapt the system based on their specific operational requirements.

Aligned with the principles of Industry 4.0, the proposed system seeks to bridge the gap between traditional manual monitoring practices and fully automated smart manufacturing environments. By offering an affordable and adaptable alternative to conventional automation systems, MiDAS has the potential to enhance operational efficiency, improve reliability, and support data-driven manufacturing for small and medium-scale industries.

II. LITERATURE SURVEY

Industrial automation is undergoing significant transformation due to the proliferation of affordable IoT hardware and advancements in lightweight communication protocols, along with edge cloud computing infrastructures. Traditional methods relying on PLCs and proprietary SCADA systems are increasingly being supplanted by innovative IoT solutions that offer flexibility and cost-effectiveness, particularly benefiting smaller manufacturing operations. Current research lays the groundwork for systems like MIDAS, emphasizing the necessity of real-time monitoring and predictive maintenance on low-cost platforms.

In their review, Sharma and Patel [1] analyzed low-cost IoT systems tailored for industrial applications, highlighting the role of sensor deployments for data acquisition, wireless networking, and the avoidance of conventional PLCs. Their findings indicate that small enterprises can achieve substantial cost reductions by utilizing microcontrollers, cloud solutions, and modular components. Building on this framework, Shinde and Kulkarni [2] proposed an IoT-driven predictive maintenance model employing machine learning techniques to evaluate equipment health based on sensor data, demonstrating how IoT analytics facilitate failure predictions. This trend suggests a shift towards economical and adaptable monitoring strategies.

Effective communication within industrial IoT ecosystems among devices and gateways is paramount. Singh et al. [3] explored the use of MQTT in manufacturing settings, noting its efficacy for low-latency applications and bandwidth constraints. The lightweight publish-subscribe architecture of MQTT supports distributed devices, allowing rapid event dissemination. Furthermore, Hong and Zhao [11] validated the reliability of real-time data transmission using Raspberry Pi alongside MQTT in their experiments. Menon [19] highlighted that MQTT effectively manages real-time updates from distributed sensor streams, reinforcing its suitability for IoT environments.

Regarding hardware for monitoring machinery, Kumar and Ghosh [8] integrated ESP32 and Raspberry Pi configurations for device monitoring, emphasizing the inherent Wi-Fi capabilities, interrupt handling for sensory input, and lightweight processing. This aligns with the MIDAS model, where the ESP32 performs edge computing tasks, while the Raspberry Pi aggregates and validates data. Additionally, Patel and Shah [17] demonstrated the feasibility of employing IoT sensors for vibration energy harvesting, proving that low-cost sensors can effectively contribute to behavioral analysis. Yadav and Singh [18] integrated IoT and machine learning methodologies for fault detection, utilizing datasets to forecast equipment failures. Collectively, these advancements underscore the potential of affordable sensors and controllers for monitoring and assessing machine states effectively.

III. METHODOLOGY

The proposed MiDAS framework is designed as an integrated system combining sensing, edge computing, backend processing, and visualization. The methodology outlines the functional components and data flow involved in enabling real-time monitoring and predictive maintenance.

A. System Components

1. ESP32 Microcontroller Units

The ESP32 modules are envisioned to operate in conjunction with sensors deployed near industrial machines. These microcontrollers are intended to serve as the primary interface for data acquisition and communication.

The ESP32 units may be responsible for:

- Capturing machine events such as punch cycles in real time
- Applying signal conditioning techniques (e.g., debouncing) to reduce noise
- Structuring event data with timestamps
- Transmitting data to the edge gateway via wireless communication protocols such as Wi-Fi or MQTT

2. Proximity Sensors

Proximity sensors, based on optical or inductive principles, are proposed for detecting machine operations. These sensors are expected to generate signals corresponding to the completion of each machine cycle (e.g., punch events).

Each detected event can be translated into a digital signal, enabling accurate tracking of machine activity over time. The use of such sensors allows continuous monitoring without direct physical interaction with machine components.

3. Raspberry Pi 4 Model B (Edge Gateway)

The Raspberry Pi is proposed as an edge gateway responsible for aggregating data from multiple ESP32 nodes. It acts as an intermediate processing unit between the sensing layer and the cloud infrastructure.

The edge gateway may perform tasks such as data collection, preprocessing, temporary storage, and communication with backend systems, thereby reducing latency and improving system efficiency.

4. Central Backend and Database

The backend infrastructure is intended to be implemented using server-side technologies such as Node.js, which can handle incoming data streams and execute system-level logic.

A relational database system, such as PostgreSQL, may be utilized to store both real-time and historical data. The database structure can include:

- Machine-wise event logs
- Maintenance and alert records
- Threshold configurations and derived analytics

5. Dashboard Interface

A web-based dashboard is proposed for system visualization and monitoring. It can be developed using modern frontend frameworks such as React.js to provide real-time updates on machine status, alerts, and production metrics.

The interface is expected to present data in an intuitive format, enabling users to monitor system performance and respond to maintenance requirements effectively.

B. Data Acquisition Method

In the proposed system, proximity sensors generate signals corresponding to machine activity, which are captured by ESP32 modules. These modules may process the incoming signals and maintain a running count of machine cycles.

The processed data can be transmitted at regular intervals using lightweight communication protocols such as MQTT. This approach allows efficient data distribution from multiple sensing nodes to the edge gateway or cloud infrastructure.

C. Edge Processing

The edge gateway is expected to perform preliminary data processing before forwarding the data to the backend. This preprocessing stage enhances data reliability and reduces noise.

Key edge processing functions may include:

1. **Signal Validation:**
Filtering out invalid or noisy signals caused by electrical interference or sensor anomalies.
2. **Counter Consistency Check:**
Monitoring irregularities such as unexpected resets or abnormal jumps in event counts.
3. **Derived Metrics Calculation:**
Computing parameters such as punch rate, cycle interval variation, and machine activity status.
4. **Threshold Evaluation:**
Comparing operational metrics with predefined limits to identify potential maintenance requirements.

D. Backend Processing Pipeline

The backend system is designed to perform advanced data processing, analysis, and decision-making. It may incorporate rule-based and analytical methods to ensure data quality and support predictive maintenance.

The backend processing pipeline may include:

1. **Data Cleaning:**
Ensuring completeness, consistency, and validity of incoming data.
2. **Maintenance Logic:**
Generating alerts when machine parameters exceed predefined warning or critical thresholds.
3. **Predictive Indicators:**
Estimating metrics such as rate of usage, time-to-threshold, and potential maintenance windows.
4. **Report Generation:**
Producing summarized insights, including daily and line-wise production trends.

E. Data Storage and Retrieval

The proposed system utilizes a structured database approach for storing machine-related data. A relational database such as PostgreSQL can be used to organize information into well-defined tables, including machine logs, maintenance records, and alert histories.

Data access and communication between system components may be facilitated through RESTful APIs, ensuring efficient and secure data exchange. This structured approach supports both real-time monitoring and long-term analytical applications.

IV. IMPLEMENTATION

The Mini Distributed Automation System (MiDAS) is proposed as a cost-effective and scalable alternative to conventional PLC–SCADA-based industrial automation systems. The framework aims to support real-time machine monitoring, production data acquisition, and predictive maintenance through an integrated hardware–software architecture. The proposed approach is intended to reduce reliance on manual record-keeping, improve maintenance response time, and enhance overall operational efficiency in small and medium-scale manufacturing environments.

A. Hardware Architecture

The proposed hardware architecture is centered around an edge computing device such as a Raspberry Pi 4 Model B, which serves as a local processing unit. This device is envisioned to interface with inductive and proximity sensors deployed on industrial machines to detect operational events such as punch or cycle completion.

For scalable deployment, multiple machines can be interconnected through an industrial Ethernet switch, enabling a single edge device to monitor several machines within a network segment. The exact number of machines supported may vary depending on system configuration, data rates, and sampling requirements.

To ensure reliable communication in industrial environments, shielded Ethernet cables (e.g., Category 6) may be utilized to minimize interference caused by electromagnetic disturbances. The edge device can maintain connectivity with cloud services through wired or wireless communication channels, enabling continuous data synchronization.

The system may incorporate regulated power supplies and optional backup mechanisms such as Uninterruptible Power Supplies (UPS) to maintain uninterrupted operation during power fluctuations. Additional peripherals, including cooling mechanisms, can be integrated to ensure system stability under continuous workloads.

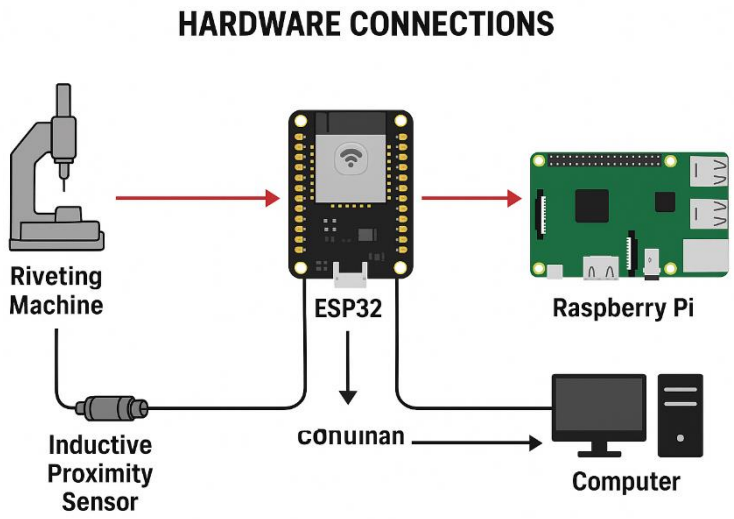

Fig. 7.1 Proposed Hardware Architecture of MiDAS

B. Software Architecture

The MiDAS framework is designed as a multi-layered system comprising the Edge Layer, Cloud Layer, and User Interface Layer. This modular architecture enables efficient handling of data acquisition, processing, storage, and visualization while maintaining system scalability and flexibility.

1. Edge Layer (Data Acquisition and Preprocessing)

The Edge Layer is responsible for real-time data acquisition and preliminary processing at the source. It is envisioned to utilize embedded platforms running lightweight scripts for interfacing with machine sensors.

Sensor signals can be captured through GPIO interfaces, and signal conditioning techniques such as software debouncing may be applied to eliminate noise and false triggers. Valid machine events may be recorded as incremental counts and associated with timestamps for further analysis.

The collected data can be temporarily stored in local buffers and periodically transmitted to higher layers in structured formats such as JSON. In addition to data acquisition, basic monitoring logic may be incorporated at the edge by comparing machine activity against predefined maintenance thresholds. This enables early identification of abnormal conditions while reducing unnecessary data transmission and network overhead.

2. Cloud Layer (Data Management and Analytics)

The Cloud Layer is intended to function as the central hub for data storage, processing, and analytics. Machine data received from edge devices can be organized using cloud-based database systems in a structured and hierarchical manner.

Backend processing may be implemented using serverless computing models or cloud functions, which can perform tasks such as data validation, anomaly detection, and maintenance alert generation. These processes are expected to ensure data consistency while filtering out erroneous or redundant inputs.

The system may also support automated notification mechanisms, where alerts are triggered when machine parameters approach predefined thresholds. Integration with messaging services can facilitate real-time communication with maintenance personnel, enabling proactive decision-making.

3. User Interface Layer (Monitoring Dashboard)

The User Interface Layer is proposed as an interactive platform for monitoring and visualization. It can be implemented as a web-based or progressive web application, allowing accessibility across multiple devices including desktops, tablets, and mobile systems.

This layer is expected to present key operational metrics such as machine status, production counts, and maintenance alerts in an intuitive format. Role-based access control mechanisms may be incorporated to ensure secure and controlled usage.

Visualization libraries and graphical tools can be utilized to display both real-time and historical data, enabling users to analyze trends and make informed operational decisions.

C. Communication and Data Flow

The proposed system follows a structured data flow beginning at the machine level, where sensor signals represent operational events. These signals are captured and processed at the edge, after which the processed data may be transmitted to the cloud using secure communication protocols such as HTTP or MQTT.

To ensure reliability, the system may include temporary local storage mechanisms to handle network disruptions. In such cases, data can be buffered locally and synchronized with the cloud once connectivity is restored.

The cloud database acts as an intermediary between the edge layer and the user interface, enabling real-time updates through event-driven mechanisms. This ensures that the dashboard reflects the latest machine status with minimal latency.

D. System Evaluation (Proposed)

The system is intended to be evaluated based on parameters such as reliability, responsiveness, and scalability. Simulated or real-time industrial signals may be used to assess system performance under varying operational conditions.

Performance metrics such as data transmission latency, event detection accuracy, and system throughput can be analyzed to validate the effectiveness of the proposed architecture. Additionally, fault tolerance mechanisms such as data buffering during network failures can be assessed for reliability.

E. Expected Advantages

The proposed MiDAS framework offers several potential advantages over traditional industrial automation systems:

- Reduced implementation and maintenance cost due to the use of low-cost hardware components
- Scalability through modular architecture
- Real-time monitoring and data-driven maintenance capabilities
- Reduced dependency on manual data recording
- Improved operational transparency and efficiency

The system is particularly suited for small and medium-scale industries seeking affordable and adaptable automation solutions aligned with Industry 4.0 principles.

F. System Workflow (Conceptual)

The overall workflow of the proposed system follows a sequential process involving data acquisition, preprocessing, validation, storage, and visualization. Machine events are captured through sensors,

processed at the edge, and transmitted to the cloud for storage and analysis. The processed information is then presented through the user interface for monitoring and decision-making.

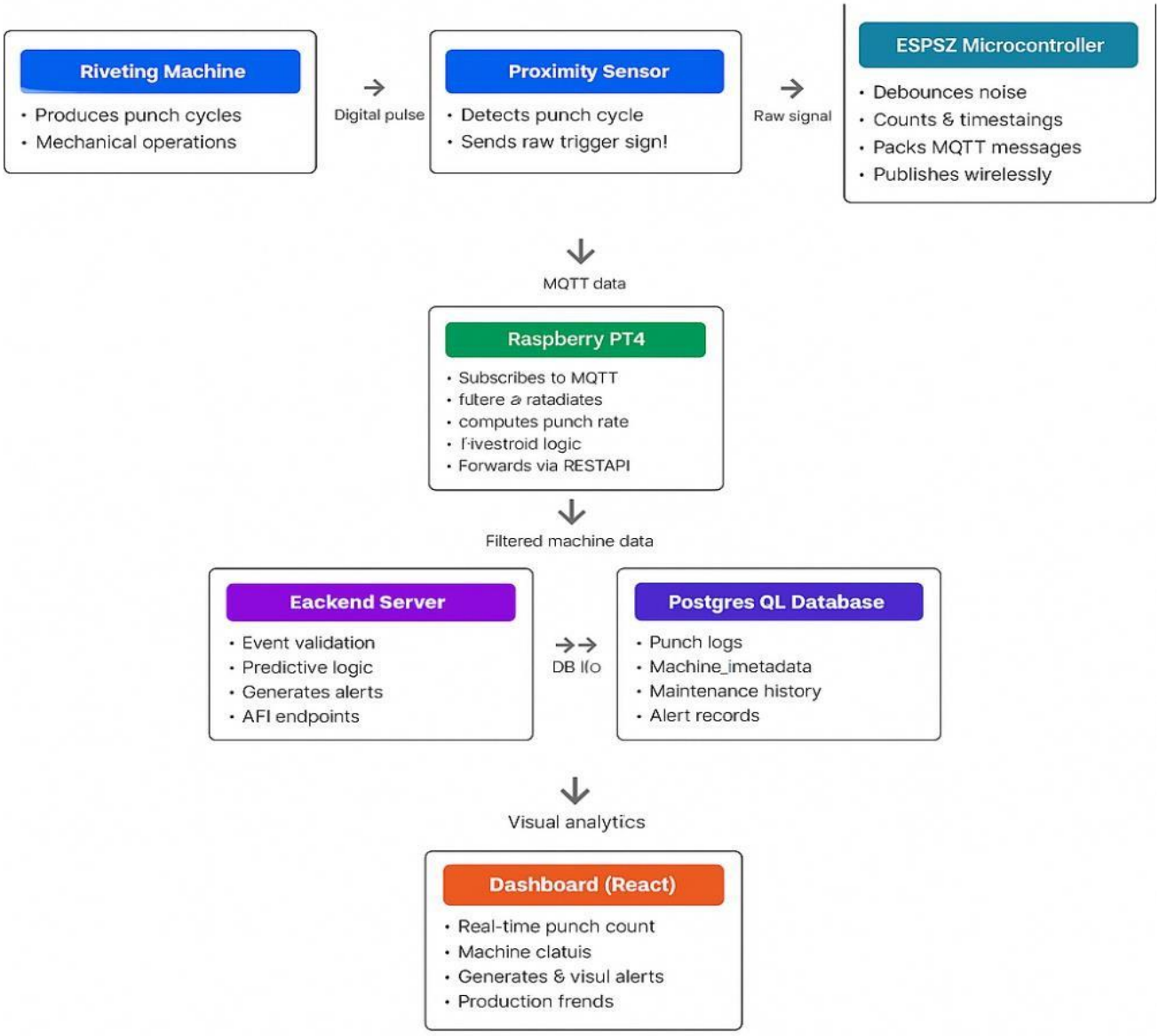

Fig 7.2 System Workflow

V. CONCLUSIONS

The proposed MiDAS framework presents a scalable and cost-effective approach for enabling predictive maintenance in small and medium-scale manufacturing environments. By leveraging a compact IIoT-based architecture as an alternative to conventional PLC–SCADA systems, the model aims to provide an accessible pathway toward data-driven industrial automation.

The system is designed to integrate components such as proximity sensors, ESP32 microcontrollers, and a Raspberry Pi-based edge gateway, along with a centralized backend infrastructure. This architecture is intended to support accurate event detection and structured data processing through stages such as validation, filtering, and threshold evaluation. The use of lightweight communication protocols, such as MQTT, is expected to facilitate efficient real-time data transmission with minimal latency, making the system suitable for continuous monitoring applications.

Furthermore, the incorporation of backend analytics and alert mechanisms is envisioned to enable proactive maintenance strategies by identifying early signs of machine degradation. Long-term data storage solutions, such as relational databases, can support historical analysis and the development of predictive models for improved maintenance planning.

The proposed dashboard interface is expected to transform raw machine data into meaningful insights through real-time visualization, status monitoring, and trend analysis. Overall, the system is designed with a focus on affordability, modularity, and ease of deployment, making it particularly suitable for industries where traditional automation solutions may be impractical due to cost or complexity constraints.

By automating machine monitoring and maintenance tracking, the proposed approach has the potential to reduce manual intervention, minimize human error, and enhance operational transparency. In the long term, such a system can contribute to improved machine performance, reduced downtime, and more efficient, data-driven decision-making in manufacturing processes.

VI. REFERENCES

[1] S. Sharma and R. Patel, "IoT-based industrial automation: Review of economical monitoring solutions," *International Journal of Emerging Technologies in Engineering Research (IJETER)*, vol. 8, no. 5, pp. 45–52, 2020.

[2] M. Shinde and A. Kulkarni, "Design and implementation of an IoT and machine learning-based predictive maintenance system," *International Research Journal of Engineering and Technology (IRJET)*, vol. 7, no. 9, pp. 980–985, 2020.

[3] S. Singh, P. Kumar, and R. Jain, "Use of MQTT protocol in IoT-driven industrial automation," in *IEEE International Conference on Communication and Electronics Systems (ICCES)*, pp. 215–220, 2019.

[4] N. Gupta and A. Kumar, "SCADA system development using Raspberry Pi integrated with IoT," in *IEEE International Conference on Computing, Power and Communication Technologies (GUCON)*, pp. 110–114, 2021.

[5] A. S. Al-Ali, I. Zualkernan, and F. Aloul, "Mobile GPRS-based sensor array for monitoring air pollution," *IEEE Sensors Journal*, vol. 10, no. 10, pp. 1666–1671, 2010.

[6] K. S. Rajasekar and V. Arulmozhi, "Machine learning-based predictive analytics for industrial IoT applications," *International Journal of Scientific & Technology Research*, vol. 9, no. 4, pp. 312–318, 2020.

[7] A. Banerjee, "Smart manufacturing data analytics and visualization using Python and MQTT: A case study," *IEEE Access*, vol. 8, pp. 150987–150995, 2020.

[8] D. Kumar and S. Ghosh, "Industrial data monitoring through integration of ESP32 and Raspberry Pi," *International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering*, vol. 10, no. 3, pp. 2180–2186, 2021.

[9] R. K. Jain and P. Mehta, "Smart factory applications using IoT and machine learning techniques," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4505–4512, 2020.

[10] T. P. Nguyen, "Analysis of low-cost SCADA alternatives based on open-source platforms," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 3, pp. 2055–2063, 2021.

[11] L. Hong and J. Zhao, "MQTT and Raspberry Pi-based real-time data acquisition and monitoring system," in *International Conference on Intelligent Computing and Control Systems (ICICCS)*, pp. 865–870, 2022.

[12] P. R. Raj and S. Thomas, "IoT-enabled low-cost predictive maintenance system using vibration analysis," *Journal of Electrical Systems and Information Technology*, vol. 8, no. 2, pp. 155–163, 2021.

[13] A. Verma, M. Kaur, and S. Sinha, "Edge device and cloud analytics-based smart factory implementation," *IEEE Access*, vol. 9, pp. 105320–105330, 2021.

[14] H. Zhang and Y. Li, "MQTT and Node.js-based industrial IoT data management," in *IEEE International Conference on Industrial Technology (ICIT)*, pp. 450–456, 2019.

[15] P. Bhattacharya, "SCADA-lite solutions for small-scale industries using Raspberry Pi," *International Journal of Engineering Research and Applications (IJERA)*, vol. 11, no. 4, pp. 37–43, 2021.

[16] M. Patel and T. Shah, "IoT sensor-based energy and vibration monitoring in manufacturing systems," in *IEEE International Conference on Intelligent Systems and Control (ISCO)*, pp. 323–327, 2022.

[17] S. Yadav and K. Singh, "Machine learning integrated IoT system for fault detection in manufacturing," in *IEEE International Conference on Smart Technologies and Systems for Next Generation Computing (ICSTSN)*, pp. 78–83, 2022.

[18] R. Menon, "Real-time IoT communication using MQTT protocol implementation," in *International Conference on Smart Electronics and Communication (ICOSEC)*, pp. 1120–1125, 2021.

[19] G. Thomas and L. Mathew, "Web-based industrial monitoring dashboard using React and Flask framework," *International Journal of Computer Applications*, vol. 183, no. 29, pp. 20–26, 2021.