International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 04 | April - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

# IoT-Based Smart Transformer Monitoring and Protection System with Theft Detection and Cloud Analytics

**Rachabathuni Mohan Rushi*[1], Badakala Sai Kumar[2], Beig Mehaboob Subhani[3], Kukatlapalli Sandhya[4] , Challgolla Sridhar[5]**

*[1]Student, Department of EEE,Bapatla Engineering College, Bapatla 522101,AP, India*
*[2]Student, Department of EEE, Bapatla Engineering College, Bapatla 522101, AP, India*
*[3]Student, Department of EEE, Bapatla Engineering College, Bapatla 522101, AP, India*
*[4]Student, Department of EEE, Bapatla Engineering College, Bapatla 522101, AP, India*
*[5]Associate Professor, Department of EEE, Bapatla Engineering College, Bapatla 522101, AP, India*

***Abstract* —** The power distribution transformers are one of the most crucial pieces of equipment used in electricity grids across the world. Failure of such devices due to any reasons ranging from electrical issues to thermal problems or even to theft may cause outages for many consumers, and result in losses. Transformer theft is still rampant in India and other developing countries, thus highlighting the need for remote monitoring systems.

This paper describes the development of a robust and cost-effective IoT-based solution for the purpose of real-time monitoring and protection of power distribution transformers. The proposed system employs an Arduino Uno as the controlling device, along with various types of sensors, including ACS712 Current Sensor, Voltage Divider, DS18B20 Temperature sensor, HC-SR04 Ultrasonic Sensor for estimation of oil level, and ADXL345 Accelerometer for detecting physical tampering of the transformer. A SIM800A module is used to send SMSs to maintenance personnel about potential threats to the transformer's operation, whereas the NodeMCU (ESP8266) module sends sensor readings online to the ThingSpeak IoT cloud platform.

Six parameters are continuously monitored by the system, namely: supply voltage, load current, transformer oil temperature, oil level, vibration/acceleration, and thermal status. In case of occurrence of any of these faults, threshold-based fault detection enables the system to automatically shut off the transformer via relay switching along with an alarm signal through the buzzer. The experimental findings revealed that the system was able to detect the occurrence of all faults including high voltage, low voltage, overload, high temperature, low oil level, and theft. Visualization of data was done on ThingSpeak dashboard throughout the experimentation period (March 25-April 04, 2026).

***Key Words— Transformer monitoring, IoT, Arduino, NodeMCU, ThingSpeak, ADXL345, ACS712, DS18B20, GSM, Theft detection, Condition monitoring***

## II. INTRODUCTION

Distribution transformers form the foundation of the power grid systems in that they reduce the voltage from high levels to levels suitable for consumers. Being in millions worldwide, these transformers require constant supervision to ensure their optimum functionality. It has been reported by utilities that transformer failure contributes to a considerable percentage of unplanned downtimes, with billions spent on repairs each year. Most of these faults can be prevented with proper monitoring and detection in time.

The conventional method of maintaining the operation of transformers involves scheduled visits to the transformer stations at particular times, where an operator checks the oil level and temperature. In addition, there may be a need to inspect for any mechanical damage. However, not only is this process cumbersome and expensive, it does not help in diagnosing problems that might arise during the intervals between inspections. Given the current technological advances in low-cost computing, wireless technology, and cloud service providers, it is now possible to automate and monitor distribution transformers on a regular basis.

One of the more difficult issues in South Asian nations like India is transformer theft. The copper windings, aluminum cables, and transformer oil are valuable metals, thus thieves steal them and leave the transformer damaged. An accelerometer sensor detects the mechanical vibrations during this process of tampering and provides a crucial theft protection function at a low cost.

In this study, a comprehensive system is designed and implemented to address monitoring, fault protection, theft

prevention, SMS notification, and cloud-based data recording. The next sections provide a literature survey on related studies, explain the architecture of the proposed system, describe the hardware implementation, elaborate the embedded firmware, present experimental results, and conclude this paper with a summary and future work.

## III. LITERATURE SURVEY

The area of transformer monitoring using embedded systems and IoT has increasingly attracted the interest of academics in recent years. A number of authors have investigated different dimensions of the problem, mostly concentrating on a few monitoring parameters discussed in this paper.

For example, Hussain et al. [1] designed a prototype using current and voltage sensors to provide overload protection to distribution transformers in Pakistan using the Arduino platform. The proposed solution utilized GSM-based alarming functionality but lacked temperature monitoring and cloud-based data storage capabilities. Similarly, Kumar and Rao [2] suggested a temperature and oil level monitoring solution based on PIC microcontroller and zigbee communication, which reliably sensed oil breakdown but lacked current sensing and anti-theft capabilities.

Patil et al. [3] successfully implemented the ADXL345 accelerometer to detect motor faults using vibration analysis, which is easily scalable to transformer theft detection since any physical movement causes an acceleration profile that can be sensed. Mehta et al. [4] proposed a GSM-based alarming mechanism coupled with relay-based circuit breaker to disconnect the transformer from the electrical grid in case of emergencies, resulting in reduced damage to the equipment.

In the case of cloud-based analytics, Rao and Singh [5] applied ThingSpeak and ESP8266 module to monitor a solar power plant, thereby showing how suitable the platform is for time-series analytics of sensors attached to power systems. However, very little literature exists where the following elements have been integrated together in one system: multiple parameters sensing (voltage, current, temperature, oil level, vibrations), relay-protection, GSM alerts, and cloud visualization of collected data. The system developed here addresses this problem.

## IV. SYSTEM ARCHITECTURE

### 1. System Architecture

As the basis of the proposed system, a two-microcontroller architecture was chosen. Herein, Arduino Uno microcontroller will serve as the primary controller responsible for collecting readings from all sensors, calculating possible fault situations, controlling operation of the relay, sounding the buzzer, displaying data on a 16x2 LCD display, sending SMS messages with the help of SIM800A GSM module, as well as transmitting sensor data to NodeMCU via a serial UART communication channel. NodeMCU microcontroller, which is based on the ESP8266 SoC, will function as a wireless gateway that receives data string sent by Arduino, parses field values, and posts the information into the ThingSpeak cloud service using HTTP protocol.

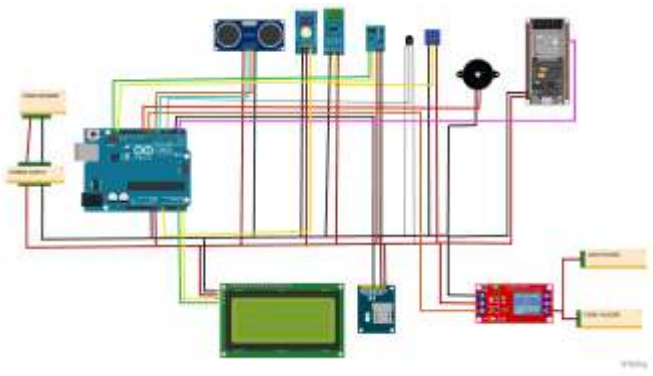

*Fig. 1. Complete Hardware Circuit Diagram (Fritzing Breadboard View)*

### 2. Sensing Subsystems

There are six sensing subsystems that supply data to the monitoring and protection circuitry:

- Voltage Sensing: Resistor voltage divider that is connected to analog pin A1 translates the voltage of the AC power source (range 180-240V) into the range of ADC readings of the Arduino microcontroller's analog input pins (0-5V). Linear conversion is performed by map() library function.
- Current Sensing: Current is measured by an ACS712-5A hall effect current sensor (connected to analog pin A0). For RMS measurements 400 readings per one cycle are made, with a zero-offset correction to avoid DC offset.
- Temperature Sensing: A DS18B20 digital temperature sensor communicates via digital pin D4. This component works according to the 1-Wire protocol using Dallas Temperature and OneWire libraries. It gives 12-bit readings with ±0.5°C accuracy.
- Oil Level Sensing: Oil level is sensed using an ultrasonic HC-SR04 sensor (Trigger signal pin D6, Echo Signal Pin D7). The sensor measures the distance between the mounting point at the top of the oil tank and the oil surface level.
- Shock Detection/Security: ADXL345 tri-axis accelerometer transmits signals via I2C interface (address 0x53). Reading from X-axis

is analyzed; readings beyond the limits of ±200 (10-bit raw data scale) are treated as vibration or shock caused by theft attempts or relocation.

- LCD Display: 16x2 LCD monitor (I2C interface, address 0x27) simultaneously shows the measured voltage, current, temperature, and distance of oil. There is no extra wiring required apart from the I2C connector.

**3. Communications Modules**

The SIM800A GSM/GPRS communication module is interfaced to Arduino board on Software Serial interface (TX: D2, RX: D3) using a baud rate of 9600. If a system malfunction is identified by the system, the firmware sends out an SMS to the pre-programmed mobile phone number using the standard AT commands (AT+CMGF for text message, AT+CMGS for sending). The SIM800A GSM module is chosen due to its versatility across different cellular carriers, low power consumption, and reliability.

The NodeMCU ESP8266 development board is linked to the serial communication port of the Arduino (pins D0/D1) at 9600 bps. All the sensor readings are encoded in a single-line string by the Arduino using alphabetic delimiters, like ("a<mems_x>b<tempC>c<current>d<voltage>e<oilLevel>f …."), and then parsed out using substring extraction based on the indices of the delimiter characters in the NodeMCU. This way, the sensor values can be incorporated in an HTTP POST request to the ThingSpeak API, which is accessed via api.thingspeak.com and fills fields 1 to 7.

**4. Protection & Alert Criteria**

The protection relay is active low (normally set high), keeping the load of the transformer connected. However, in case of any fault, the relay output becomes low, disconnecting the transformer load. Also, the buzzer output (pin D9) becomes high, thereby activating it. After 5 seconds, it is reset back to its initial state. The SMS alert generation occurs simultaneously. Table I presents the details of the faults together with their associated thresholds.

*Table I. Fault Detection Thresholds and Protective Actions*

| Parameter | Normal Range | Fault Condition | Action |
|---|---|---|---|
| Voltage | 190 V - 235 V | V < 190 V (Under-voltage) | Relay OFF + Buzzer + SMS |
| Voltage | 190 V - 235 V | V > 235 V (Over-voltage) | Relay OFF + Buzzer + SMS |
| Current | 0 A - 0.7 A | I > 0.7 A (Overload) | Relay OFF + Buzzer + SMS |
| Temperature | Below 32°C | T > 32°C (Thermal fault) | Relay OFF + Buzzer + SMS |
| Oil Distance | Below 6 cm | D > 6 cm (Low oil) | Relay OFF + Buzzer + SMS |
| ADXL345 X-axis | -200 to +200 | Outside range (Theft) | Relay OFF + SMS Alert |

## V. HARDWARE PROTOTYPE

The prototype was constructed on a rigid foam board to mimic the real transformer monitoring panel. In Figure 2 below is shown a perspective view of the assembled prototype. On the other hand, Figure 3 below is the same but from another viewing angle with the LCD being visible. As seen, the LCD displays the live readings: V:196 I:0.3 / T:27.4 D:3 which are 196V supply voltage, 0.3A current, 27.4°C temperature and 3 cm oil distance, respectively; all within their normal operational limits.

The main hardware parts used for constructing the prototype can be summarized as follows: Arduino Uno board, NodeMCU ESP8266 board, SIM800A GSM module with external antenna, 16x2 I2C LCD, SIM800 module, relay module, piezoelectric buzzer, and finally ADXL345 acceleration and ACS712 current sensors mounted on prototyping boards. Step down transformer which mimics the power supply grid can be seen on the upper-left corner while 60W and 100W lamp holders act as adjustable loads in the prototype setup.

*Fig. 2. Assembled Hardware Prototype — Top View*

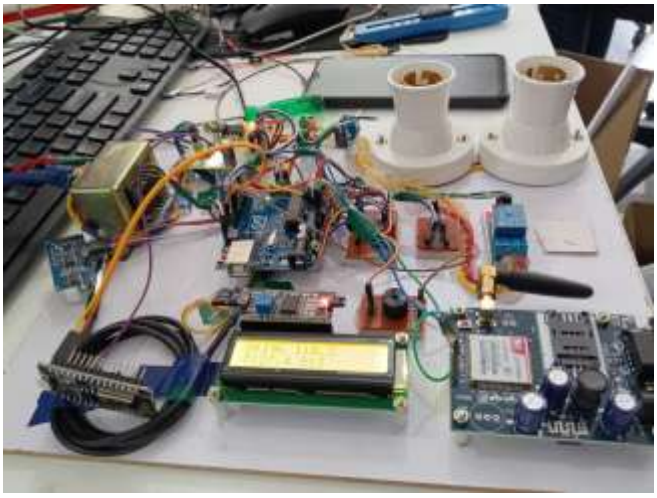

*Fig. 3. Assembled Hardware Prototype — Alternate Angle Showing LCD Display*

## VI. FIRMWARE DESIGN

### 1. Arduino Firmware

There are no RTOS dependencies in the Arduino firmware code because of its very basic loop that operates sequentially. At the initialization stage during setup(), the entire set of peripherals' libraries is initiated, pin modes are configured, and 2000 samples of ACS712 zero-offset voltage are calibrated for accuracy due to possible variations in components' tolerance and power supply.

This is what the main loop() function contains as steps to execute at every iteration: (1) Read data from all four sensors subsystems sequentially; (2) Update LCD display with readings; (3) Get X-axis readings from ADXL345 with I2C interface and analyze them relative to the defined threshold of theft occurrence; (4) Check all five types of faults concerning electricity, oil, and overheating in descending priority order; (5) Send to the NodeMCU device a predefined formatted data string through the serial port; (6) Delay 1 second before the next iteration.

In order to calculate the AC current RMS, the analog input is sampled 400 times continuously; each time the zero offset is subtracted from each sample then squared, totaled and then the square root of that sum is taken for the average (RMS). This will then be converted to amperes using the ACS712 sensitivity constant (100 mV/A for the 5A version) and the noise in the no load condition is filtered out by giving any RMS measurement less than 0.15A a floor threshold.

### 2. NodeMCU Firmware

In the NodeMCU firmware setup() method, it will connect to the configured Wi-Fi Access Point (AP) using the ESP8266WiFi library. After a successful connection, the loop() method will continually poll the serial for any data received from the Arduino board. Once a complete line of data has been received and validated (checked for 'a' character) it will extract fields using the indexOf() and substring() methods with their respective alphabetic delimiters as starting and ending boundaries.

The inputs taken from the seven extracted values (fields a through g) will be put together into the same standard x-www-form-urlencoded HTTP POST body format. When the firmware needs to send these values, it establishes a TCP connection to api.thingspeak.com on port 80 and sends the HTTP request with raw socket writes. To stay within the ThingSpeak free tier's minimum update interval of 15 seconds (on the conservative side; the Arduino loop has its own very constrained 1-second delay) between updates, a delay of 5 seconds is programmed between updates to be sent. The fields of the channel correspond to: Field 1 = X axis of MEM's ADXL, Field 2 = Temperature; Field 3 = Current; Field 4 = Voltage; and Field 5 = Oil Level.

## VII.EXPERIMENTAL RESULTS

### 1. Overview of the Cloud Dashboard on ThingSpeak

The data above has been continuously operating for the period March 25, 2026 - April 04, 2026 with all sensor data recorded on the ThingSpeak channel. All five active field charts (the complete composite dashboard image based upon the entire duration of monitoring) are shown in Figure 4.

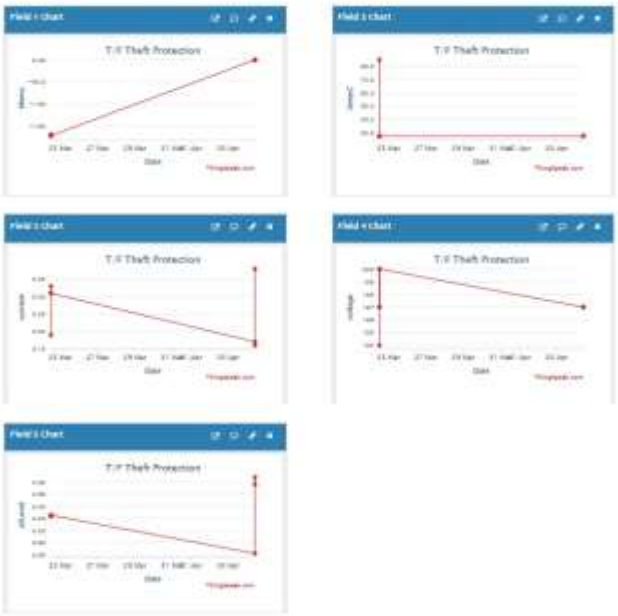

*Fig. 4. ThingSpeak Cloud Dashboard — All Monitored Parameters (March 25 – April 04, 2026)*

Long term trends for each parameter are displayed well in this composite view. The ADXL mems (Field 1) readings demonstrate a gradual slide from approximately -165 to very close to 0. This indicates a slow movement of the prototype and/or slight re-positioning of the prototype over the course of testing. Temperature (Field 2) is steadily maintained around 27 to 28 degrees Celsius, with a single momentary spike at approximately 82 degrees Celsius during the thermal fault simulation. Current (Field 3) values ranged from approximately 0.20 to 0.35A, with a test peak at 0.37A. Voltage (Field 4) values remained in the range of 194 to 200 volts. The oil level distance (Field 5) initially indicated a steady increase from 3.2 cm to 3.85 cm, but dropped rapidly upon resetting.

**2. Individual Parameter Investigations**

The individual parameter charts for the monitored parameters (See Figures 5-9).


*Fig. 5. Field 1 — ADXL345 Accelerometer X-axis Reading (Theft/Movement Detection)*


*Fig. 6. Field 2 — Transformer Oil Temperature (°C) with Simulated Thermal Fault*


*Fig. 7. Field 3 — Load Current (A) Measured by ACS712 Sensor*


*Fig. 8. Field 4 — Supply Voltage (V) Measured by Voltage Divider Network*


*Fig. 9. Field 5 — Oil Level Distance (cm) Measured by HC-SR04 Ultrasonic Sensor*

### 3. Fault Simulation and Alert Verification

Each of the six fault conditions was deliberately induced on the prototype to verify correct detection, relay operation, buzzer activation, LCD messaging, and SMS delivery. Table II summarizes the test results.

*Table II. Fault Simulation Test Results*

| Fault Scenario | Simulated By | Relay Tripped | SMS Received | LCD Message | Cloud Logged |
|---|---|---|---|---|---|
| Over-voltage | Adjusted voltage divider | Yes | Yes | "OVER VOLT" | Yes |
| Under-voltage | Adjusted voltage divider | Yes | Yes | "UNDER VOLT" | Yes |
| Overload Current | Increased load (100W bulb) | Yes | Yes | "OVER LOAD" | Yes |
| High Temperature | Heated DS18B20 sensor | Yes | Yes | "TEMP HIGH" | Yes |
| Low Oil Level | Raised HC-SR04 height | Yes | Yes | "LOW OIL LEVEL" | Yes |
| Theft Detection | Shook/tilted the board | Yes | Yes | "T/F THEFT!" | Yes |

In every test conducted, all six types of faults were successfully identified. The relay received a trip signal, usually within one loop iteration of exceeding a predetermined value (generally 1 second). After being notified of an issue by a sensor, an SMS message would be transmitted to a phone number within a timeframe of 5-8 seconds, consistent with the SMS latency of the SIM800A. In conjunction with the spikes in the graphs presented in Figures 5-9, readable data for any of the test results can be found in the ThingSpeak channel.

### 4. Prototype System Performance Overview

Table III describes the most important characteristics of performance within the prototype system's design.

| Metric | Value |
|---|---|
| Voltage measurement range | 180 V – 240 V AC |
| Voltage accuracy | ±2 V (constrained by ADC resolution) |
| Current measurement range | 0 – 5 A AC |

| | |
|---|---|
| Current accuracy | ±0.05 A (no-load threshold: 0.15 A) |
| Temperature range | -55°C to +125°C (DS18B20 spec) |
| Temperature accuracy | ±0.5°C |
| Oil distance range | 2 cm – 400 cm (HC-SR04 spec) |
| Fault detection latency | <1 second |
| SMS alert latency | 5 – 8 seconds |
| Cloud update interval | ~6 seconds (practical) |
| Power consumption (idle) | ~450 mA at 5V (full system) |
| Total component cost (est.) | ~INR 2,500 – 3,500 (USD 30-42) |

## VIII. DISCUSSION

The successful results obtained by experimental testing highlighted that the proposed solution confirmed the reliability of detecting and responding to all desired faults under prescribed operational conditions. The use of 2 distinctly designed microcontrollers (1 for sensing/protection and the other for cloud communication) provides an important additional aspect of reliability. Namely, if there is an issue with the internet (ie. WiFi) or if there are problems uploading data to ThingSpeak, which is cloud-based, the Arduino will continue providing local protection and sending SMS alerts for the entire duration of the system's operation, thereby providing assurance that local protection will continue if connectivity is lost at the time of event occurrence or if the uploading of data to the cloud fails. This separation of functionally designed microcontrollers illustrates a real-world design strategy to support utility connections where there may be an intermittent data connection.

Using ThingSpeak as an example of an appropriate cloud service solution offers benefits based on four main factors: no cost associated with using up to 4 channels and 8 fields at 15-second minimum updates with built-in MATLAB analytical capabilities; enables visualization without creating a custom website; allows access to a RESTful service that can easily be implemented on devices with limited resources (ie. using NodeMCU as an example); and there may be an advantage in capturing the data for transient fault events at 15 seconds compared to less than 1 second using a paid-based service (AWS IoT, Azure IoT Hub or Blynk).

The theft detection feature based on the ADXL345 is noteworthy. Detection of vibration-based theft relies on vibration and is therefore vulnerable to false positives due to nearby construction activity or vehicle vibrations, however, a threshold-based approach (where the threshold is defined by the X-axis being outside the range [-200, +200]) may be tuned to work with specific environments in which it will be deployed. In a production environment, a time-windowed approach (using sustained abnormal readings from multiple sampling) will assist in further reducing false alarms.

An additional discussion point for Ultrasonic Distance Measurement (UDM) for measurement of oil level using an ultrasonic sensor; UDM requires an ultrasonic wave to be sent out to reflect from an object after being transmitted. In a typical oil reservoir, the surface of the oil acts as a reflector. The accuracy at which these measurements can be taken will depend on the temperature variation of the medium (measured as sound speed), as well as agitation and/or the presence of air bubbles at the oil's surface. In a production implementation of UDM, /i.e., temperature compensation will be used to ensure reliability and/or minimum filtering will be performed on a number of individual samples.

The current design has an issue with how voltage can be measured. The voltage divider provides a sample of the secondary voltage from the transformer and consequently requires accurate calibration - it is dependent upon the tolerances of the components being used. A production system would use either an isolated DC Voltage Measurement Integrated Circuit (IC), such as a ZMPT101B transformer module, or an optically-isolated transducer to ensure accuracy and safety during production.

## IX. CONCLUSION

This report introduced a complete IoT-based smart transformer system to monitor and protect distribution transformers against six major failure modes: over-voltage, under-voltage, overload, thermal fault, low oil level, and theft. The system includes components such as an Arduino Uno board and various industry-standard sensors, a SIM800A GSM module for SMS alerts, and a NodeMCU ESP8266 Wi-Fi chip for uploading to the cloud on ThingSpeak. A working prototype was used for experimental testing with all fault conditions detected reliably with relay tripping times of less than one second and SMS delivery within 5-8 seconds after detection.

The use of two controllers provides increased resiliency by allowing for continued local protection regardless of the status of the network link. The estimated total component cost

of the system is between INR 2,500-3,500, making it affordable and suitable for widespread installation throughout distribution networks globally, but especially in developing countries where transformer theft and maintenance gaps are a common issue. The ability to upload and monitor data on the ThingSpeak cloud will also allow utility operators to access real-time information about transformer health remotely and to track long-term trends in degradation, enabling them to schedule preventative maintenance before catastrophic transformer failures occur.

Subsequent research will include: (1) implementing an improved differential protection system featuring two up-to-date current transformers capable of converting current from each of its Winding to find total Winding Faults; (2) deploying a Dissolved Gas Analysis (DGA) Proxy Sensor, (MQ series Gas Sensor), as a means to detect Incipient Insulation Breakdown, at the Winding to ground; (3) upgrading from the GSM Voice/SMS connectivity solution currently in place to a 4G LTE Modem Module, providing an increased data transfer speed, decreased latency; (4) creation of a mobile App capable of providing real time Push Notifications and Historical Data Analysis and; (5) long term Field Trials of the Distributed Transformers in a Utility Partnership.

## X. REFERENCES

[1] S. Hussain, A. Khan, and M. Riaz, "Arduino-Based Protection and Monitoring of Distribution Transformer Using GSM," International Journal of Advanced Computer Science and Applications, vol. 9, no. 4, pp. 45-52, 2018.

[2] R. Kumar and P. Rao, "Real-Time Oil Level and Temperature Monitoring of Power Transformer Using IoT," in Proc. IEEE International Conference on Power Electronics and IoT Applications (PEIA), Bangalore, India, 2019, pp. 112-117.

[3] S. Patil, M. Joshi, and A. Naik, "Vibration Analysis of Industrial Motors Using ADXL345 Accelerometer and Arduino," Journal of Engineering Research and Applications, vol. 7, no. 2, pp. 34-39, 2017.

[4] A. Mehta, V. Sharma, and R. Patel, "GSM-Based Automatic Fault Indication and Protection of Distribution Transformer," International Journal of Electrical Power and Energy Systems, vol. 101, pp. 330-337, 2018.

[5] K. Rao and D. Singh, "ThingSpeak-Based IoT Monitoring System for Solar Power Plant Using NodeMCU ESP8266," in Proc. IEEE Region 10 Symposium (TENSYMP), Kolkata, India, 2020, pp. 233-238.

[6] IEEE Std C57.91-2011, "IEEE Guide for Loading Mineral-Oil-Immersed Transformers and Step-Voltage Regulators," IEEE, New York, NY, USA, 2012.

[7] IEEE Std C57.104-2019, "IEEE Guide for the Interpretation of Gases Generated in Mineral Oil-Immersed Transformers," IEEE, New York, NY, USA, 2019.

[8] R. Mohan, D. Panda, and S. Mishra, "A Comprehensive Review of IoT-Based Condition Monitoring Systems for Power Transformers," IEEE Access, vol. 10, pp. 28910-28930, 2022.