ASHESI

ASHESI UNIVERSITY COLLEGE

REAL-TIME CONDITION MONITORING OF ELECTRICAL MACHINES USING IOT

CAPSTONE PROJECT

B.Sc. Computer Engineering

Eugenia Mawuenya Akpo

2021

ASHESI UNIVERSITY COLLEGE

REAL-TIME CONDITION MONITORING OF ELECTRICAL MACHINES USING IOT

CAPSTONE PROJECT

Capstone Project submitted to the Department of Engineering, Ashesi University College in partial fulfilment of the requirements for the award of Bachelor of Science degree in Computer Engineering.

Eugenia Mawuenya Akpo

2021

DECLARATION

I hereby declare that this capstone is the result of my own original work and that no part of it has been presented for another degree in this university or elsewhere.

Candidate's Signature:

……………..………… Eugenia Mawuenya Akpo ………………………………………

Candidate's Name:

……….. ………… Eugenia Mawuenya Akpo ……………………………………………

Date:

……………………………27th April 2021………………………………………………

I hereby declare that preparation and presentation of this capstone were supervised in accordance with the guidelines on supervision of capstone laid down by Ashesi University College.

Supervisor's Signature:

……………………………………………………………………………………………

Supervisor's Name:

……………………………………………………………………………………………

Date:

……………………………………………………………………………………………

Acknowledgements

Thank God on High for his immense blessings and care and for seeing us through this project in our highs and lows.

To my supervisor and role model, Dr Nathan Amanquah, whose encouragement and academic advice helped me undertake this project, I appreciate you so much. A warm thank you to all Faculty for their great feedback and for guiding us through thinking out solutions.

I extend m gratitude to the Head Hostel Manager of Ashesi University, Kingsley, and security personnel at Ashesi University checkpoint for all the help extended to me. You all made this journey worth it.

I am grateful to all my family and friends outside Ashesi University for their incredible support through this project phase. God bless you all!

Abstract

Electrical Machines are heavily used in industry today as they form an integral part of the production. In the occurrence of a breakdown, processing and production are delayed hence the importance of motor condition monitoring. Unfortunately, manual condition monitoring techniques are not entirely reliable. IoT's emergence proves to be reliable and decreases downtime in the daily use of the induction motor. In this project, an IoT platform is built using Bluetooth technology, temperature, current, voltage and accelerometer sensors for data collection, storing data on the cloud and building a machine learning model to predict faults based on prevalent faults diagnosis techniques on induction.

Table of Content

List of Figures

1. Chapter 1: Introduction

Electrical motors are an integral part of industrial functions. Condition monitoring of electrical motors is an essential part of maintaining the constant healthy state of the motors. It involves monitoring parameters of vibration, motor's temperature and current consumption to identify significant changes that indicate a developing fault. Using this technique helps to prevent failure by taking necessary actions. Condition monitoring techniques have improved over the years. Internet of Things(IoT) technologies have a tremendous impact and involvement in this sphere through data collection and storage with Machine learning advantages of predictive analysis in time series. Thus, Condition Monitoring today heavily involves predictive maintenance.

1.1. Background

It is reported that faults in induction motors include bearing defects of 42%, stator defects of 28%, rotor defects of 8% and others of 22% [1]. Electric motors are electromechanical machines used to convert electrical energy to mechanical energy. The two types of motors, AC and DC, work on magnetic and electric fields. Their working principles combined with the mechanical interactions gradually generates some form of faults. Induction motors are undoubtedly reliable, robust, simple to construct; thus, the possibility of failure cannot be ignored. It can be inferred from **Fig.1.1** [25] that induction motor faults can be classified into electrical and mechanical faults. Stator, bearing, and rotor faults are highly classified. Early detection of failures is critical for industrial purposes as these faults, when they occur, delay the production cycle until corrected. Early detection involves the condition monitoring process of electrical motors [2]. A few monitoring techniques are available, such as Ultrasound Testing, Lubricant Analysis, Vibration Analysis, and Thermography.

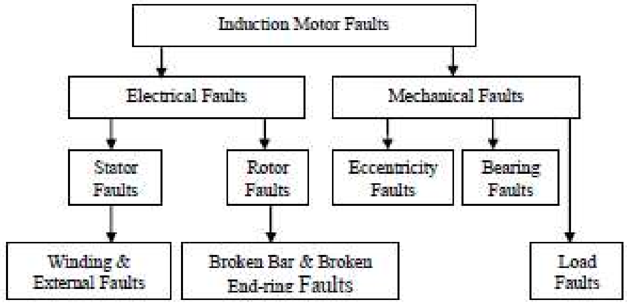


Figure 1.1 Classification of Induction Motor Faults

IoT provides a remedy where real-time analysis of various parameter measurements occurs to improve systems and increase detection and perform predictive maintenance. The Internet of Things (IoT) is a network of interconnected computing devices, events, and individuals with unique identifiers and the capacity to exchange data without involving human-to-human or human-to-computer contact [3]. IoT is a sensor network of billions of intelligent devices that connect people, systems, and other applications to collect and share data. An IoT framework comprises web-enabled intelligent devices that capture, transmit, and function on data from their surroundings using embedded processors, sensors, and communication hardware. [4]. IoT system for health monitoring system for electrical motors (induction motors) considers popular condition parameters such as vibration, thermal profiles and running parameters like speed and acceleration. An accelerometer for measuring vibration data is attached to bearing points as bearings often operate under time-varying rotational speed conditions. Current sensors measure the amount of current and voltage running through the load to determine common electrical faults like unbalanced voltage supply, locked rotor, over-voltage, and a locked rotor faults [5]. Researchers used thermocouples to measure the temperatures of the bearings as high temperatures than allowed is a reaction to an underlying problem from either loss of lubrication or bearing defect [6]. Similar temperature measurements were obtained from the rotor and stator to help identify electrical faults from the winding.

1.2. Problem Statement

Industries or residential settings that use electrical motors look for ways to monitor these devices with few in-person check-ups. For instance, Ashesi University has motors with impellers located at vantage points on the school's premises and hostels used to control water flow. Hostel and Campus Managers usually check these pumps daily to ensure they are working well. They also have other duties to attend; hence the opportunity cost of this is that the time spent on monitoring and predictions could be used on essential businesses. Hostel Managers also have a PC/Work-station at their disposal, which with the appropriate additions could have been used to monitor the state of the machines. Using an IoT-based system to monitor motors automatically goes a long way in [2]:

- Reducing the cost of maintenance
- Predicting the equipment failure
- Improving equipment and component reliability
- Optimizing the equipment performance
- Improving the accuracy in failure prediction
- Preventing downtime if detected early.
- Guiding servicing and how much grease is applied to the motor.

1.3. Project Objectives

2. To design an affordable IoT system for data collection, storage and analysis of induction motor parameters.
3. To determine predictive approaches to extend the life and good health of electrical motors.
4. To build a real-time monitoring system for users of the platform to monitor motor parameter and conditions.

5. Use Machine Learning Algorithms for predictive analysis of common induction motor faults as part of predictive maintenance of condition monitoring.

2. Chapter 2: Literature Review

Condition Monitoring is an essential process in the full utilization of an electrical machine or motor. Health monitoring of a motor has been traditionally done by checking temperatures, allowing cooldowns, checking loose bolts and bearings until automation, data collection and analysis were applied efficiently by introducing the Internet of Things (IoT). Many authors and engineers have thought of creating an IoT embedded platform to support condition monitoring. This process reduces maintenance cost, helps predict equipment failure, improves equipment reliability, optimizes equipment performance, and improves accuracy in failure prediction. The Literature review explores possible motor faults and their various ways of monitoring the health of electrical motors.

2.1. Motor Faults

For induction motors, faults can be classified into mechanical or electrical faults. The significant faults include bearing and gearbox failures, stator faults, broken rotor bar, static and dynamic air gap regularities, bent shaft and others. All these faults require a monitoring technique. Continuous stress on the bearing results in fatigue failure at the inner or outer races of the bearings. Failure is accompanied by increased levels of noise, corrosion, inadequate lubrication, and others. Another critical cause of bearing failure is temperature; thus, the temperature should not exceed a threshold value [2]. Stator defects make up 28% of all defects, and they mainly occur due to inter-turn winding faults caused by insulation breakdown. The unbalanced magnetic field causes excessive vibration causing premature bearing failures. Rotor defects are caused mainly by rotor winding. Here, rotor bars are broken; hence a pulsating effect results in fluctuation of speed, torque pulsation, vibration, overheating, arcing in the rotor and damaged rotor laminations [2]. It is essential to note how each fault occurs, the parameter measurements associated with each fault breakdown and the required modes of repair and maintenance.

2.2. Condition Monitoring Techniques

Condition Monitoring techniques include thermal monitoring, magnetic flux monitoring, vibration monitoring, partial discharge monitoring, air gap monitoring, noise monitoring, stator voltage monitoring and current monitoring. Thermal monitoring is done by measuring temperature stages which can be used for bearing and stator fault detections. Vibration monitoring, a prevalent method, is used to detect bearing failures and other mechanical faults with an accelerometer. Noise and vibration from electrical machines are monitored against standards like the ISO 10816 [6] to detect faults of this nature. The traditional methods of monitoring take much time and increase the cost of operation. Sometimes, predictions might be wrong or too late, which leads us to the concept of IoT based health monitoring systems with Machine Learning [2].

2.3. IoT Involvement

Many researchers have introduced IoT into condition monitoring processes. This method is cost-effective, provides enough data for the analysis and detection of faults in real-time. Khan et al. [2] have proposed step by step process of IoT machine monitoring approaches. It sought to measure parameters such as vibration, electric current and temperature using their various sensors (accelerator, current sensor, and thermocouple) to predict any functional abnormality to the timely performance of an Induction motor. Signal acquired is transmitted over GSM or Wi-Fi protocols for easy access globally. Gounder et al. [7] also proposed similar models but slight differences in the components and sensors. They all used IoT methods to store data using cloud storage capabilities. Ganga et al. [8] have gone further to test other means of transmitting data across the protocols and making it accessible using RESTful HTTP, XMPP, DDS and AMQP, which follow specified format or mode of communication. It also receives data from different locations via cloud services and uses data for various analysis, such as monitoring power consumption levels. Since IoT involves the use of sensors in data collection, sensor placement is a technique that should not be ignored in the process. Placing the suitable sensor at appropriate points on the motor logs very accurate values in the system and reduces false-positive results. Acceleration or speed parameters need to log at the 3-axial directions shown in Figure2.1 to obtain data for accurate analysis [9]. Temperature loggings are best diagnosed alongside the bearings and rotors.

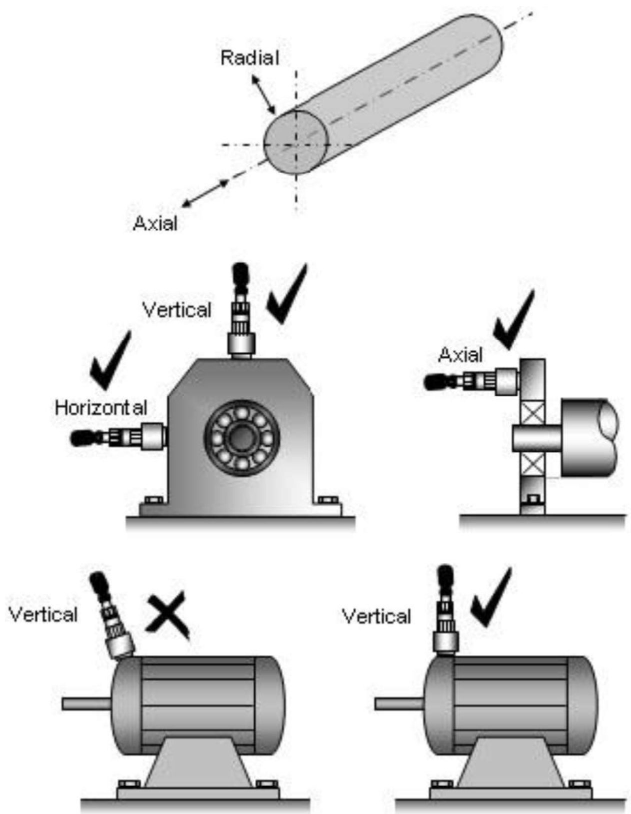


Figure 2.1 Sensor placement technique for induction motors

2.4. Design Choices

The type of sensors to select is based on the kind of data to be analyzed. The accuracy of the sensor is essential to avoid errors. Easy installation and calibration are also crucial for this system. Most researchers used the LM35 temperature sensor, the ACS712 hall effect sensor for voltage and current and the accelerometer for vibration analysis. Several considerations involving the design of the sensor platform include the microcontroller point and where the data has been sent. Some researchers used the Arduino development board and a gateway for collecting and sending data to the cloud. Most researchers focused on the use of open-source web applications for their designs. Most of the open-source software allows limited functionality. For instance, open-source web applications do no allow plots of heatmaps or thermal profiles of an induction motor except monitoring the values and levels of rising and fall. Other researchers built an attachment for alerting based on threshold limits [7]. The choice of monitoring software and storage options should consider the project's objectives and the volume of data to be collected. Data types and data sizes determine the amount of storage space and processing time, and test runs help to produce very effective and efficient programs [7],[8],[10],[11].

2.5. Data Analytics and Time Series Analysis

Time series analysis is a technique that statistically deals with data on the same scale indexed by a time-like parameter [12]. The four main types of data analysis results are; descriptive, diagnostic, predictive, and prescriptive. Descriptive data analysis tells what is happening, either now or in the past. This part focuses more on understanding the type of data collected. From a descriptive data analysis point of view, data can be pulled and analyzed at any point. Diagnostic data analysis tells the user why something happens. The predictive analysis aims to foretell problems or issues before they occur. The prescriptive analysis goes beyond predictive and recommends solutions for future problems [13]

The first part of this project aims at designing an IoT platform for collecting data and storing it in a cloud database using the RESTful HTTP protocol, which is later retrieved for monitoring on a dashboard. The second part of this project also deals with retrieving data from the database and producing predictive analysis based on common induction faults integrated into the system to identify the near occurrence quickly. This project uses Time Series Analysis and machine learning models to predict seasonal trends in the vibration of machines.

2.6. Gaps Identified.

- Very few machine learning algorithms have been used in the IoT sphere with traditional Fourier Transforms for analysis
- Few IoT platform proposals aim at predictive maintenance

2.7. Conclusion

Using IoT for condition monitoring has proven effective with the appropriate additions, and the review provides significant insights in building the system. The following chapters explore the system's requirements, propose a solution, design the proposed model, experiments and analyze results. It also requires the creation of a simple dashboard for the users of the system.

3. Chapter 3: Requirements

The requirement stage is crucial to understanding the necessities for building the system. Generally, the project aims to design a system that can collect data, process and store the data, and access data for monitoring. It also seeks to help users identify the faults and practice predictive maintenance.

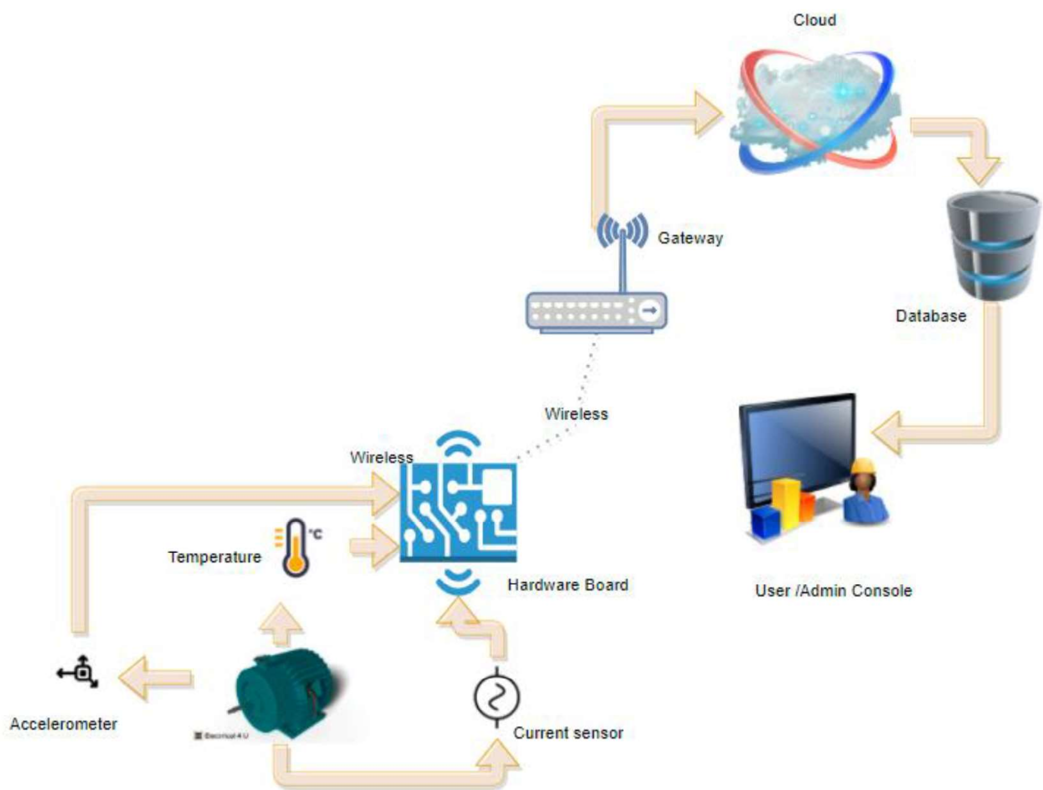


Figure 3.1 Block diagram showing IoT System Interactions

The block diagram above specifies the process of the IoT model. Vibration, Thermal, and Power parameters are analyzed for condition monitoring; hence, temperature, acceleration, current, and voltage is collected using appropriate sensors. These sensors are attached to the bearing, stator, and rotor parts of the induction motor as they produce more significant percentages of most motor faults. The data is collected by the gateway and sent to the cloud via wireless means, and the end-user views readings from their computer and make appropriate conclusions. An algorithm will be embedded in the dashboard or gateway to predict common motor faults.

3.1. System Requirements and Architecture

- The system should read data from sensors frequently to reduce pressure on connectivity and gateway options.
- The system should have a temperature sensor with a small-time constant that gives an accurate reading of every sample. The temperature sensor should be placed on bearings, stator, and rotor to read temperatures from those parts of the system.
- The system should have a vibration sensor that can read vibration values in the required axis of placement, as seen in the diagrams below. The accelerometer should mostly be placed on bearing points for accurate results [6].
- Current readings should be logged into the system.
- The system gateway should collect sensor readings and connect to the cloud.
- The system sensor platform should be easy to attach to the motor, and the platform should be portable.
- The system should be able to collect sensor readings from different motors.
- Web monitoring should be designed for users.
- Cloud storage options should be considered in the design, so storage space is not a problem.
- Users should find it easy to tell faults with alerts integrated based on the machine learning model.

4. Chapter 4: Design and Implementation

4.1. Design

The project design process can be grouped into hardware design and software design.

4.1.1. Hardware Design

Hardware Design includes the sensor platform and circuitry needed for collecting data and passing it to the gateway, which is grouped into three parts: Sensor selection, Microcontroller selection and Connectivity options.

4.1.1.1. Sensor selection

According to requirements, the system should be lightweight and inexpensive. Simultaneously, features worth measuring on induction motors include temperature for thermal profile analysis, current and voltage levels of the induction motor and internal acceleration for vibration analysis.

a. **Thermistors (Figure 4.1 (a))** are thermally sensitive resistors that show a significant, consistent, and specific change in electrical resistance in response to a change in temperature. They have a long-term stability of 0.0009°C. They exhibit a temperature range of 50°C given a base temperature and is very cheap to afford and replace. They have a time constant of about 6-14 seconds and are highly sensitive compared to LM35. The LM35 is a temperature sensor that takes analogue readings. It has a temperature range of -40°C to 100°C in its production. It also has a 1-3 second time constant and 0.01°C stability. Its linear performance makes it very simple to use, but it is moderately costly compared to thermistors. Thermistors usually are the best for measuring single point temperature, very durable, and minimal. Unfortunately, they have a non-linear output with a slow response time [14]. In this project, the thermistor is chosen to log temperature readings because of its long-term stability and accuracy.

Readings will be logged every 10 seconds hence the time constant of change has little effect on the accuracy of the reading.

b. **Accelerometer:** MPU6050 in **Figure 4.1(b)** is a Micro-Electro-Mechanical System that operates on the 3-axis accelerometer and 3-axis gyroscope principle. It is used to measure acceleration in this case. It can also measure temperature, which is helpful for analysis. The sensor is used for measuring motor vibration; hence, the gyroscope capability is not needed. It is moderately cheap and easy to get [15].

c. **ACS712 current sensor:** The current sensor in Figure 4.1 (c) operates from 5V and provides an output analogue voltage proportional to the current. It can read currents for loads operating at high voltages like 230V AC. It has a 100mV/A output sensitivity and is connected to the stator mostly to read results and detect faults [16].

d. **AMG8833 Infrared Camera:** For modification and contactless design, the AMG8833 IR Camera in Figure 4.1 (d) is a sensor that works on the principle of infrared thermography. It takes in the thermal profiles of an object and provides temperature and pixel correspondence for plotting out the image. For heatmap processing, the higher the temperature, the brighter the colour [17]. This IR camera model only shows an eight eight-pixel plotting and may not fully show images as wanted. It has a 60° range of vision capture. Capturing thermal profiles with an IR camera aids the thermography process and helps to identify faults on an induction motor quickly. Machine learning image classifications may help to identify faults in the motor.

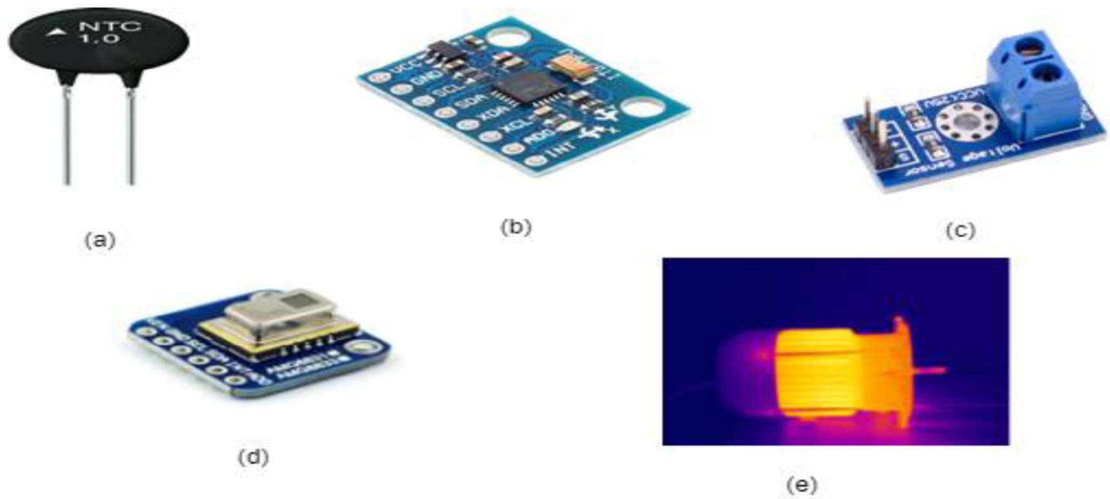


Figure 4.1 Summary of sensors and justification (a)Thermistor (b) MPU6050 Accelerometer (c) ACS712 current sensor (d) IR Camera (e) Heat Map Sample.

4.1.1.2. Microcontroller Choice

The microcontroller unit considered is the Atmgea328P chip shown in Figure 4.2 that readily interfaces with the sensors described earlier. It is a low consumption 8-bit AVR and has very high performance achieving high single clock cycle execution of 131 instructions [18]. The considerations for selecting the best microprocessor are the size of the chip and the number of pinout options available to connect all the sensors and peripherals for connectivity options. It is also minimal and lightweight.

Figure 4.2 ATmega328P Microchip

4.1.1.3. Connectivity Options

An IoT platform on collecting sensors readings requires a storage option. In designing, one of the system's requirements is to collect all readings from all motors under monitoring. One can connect to the cloud only via the internet; hence the Wi-Fi module is an excellent choice for connecting our platform to the cloud. The ESP8266 Wi-Fi module (Figure 4.3 (a)) has a range of about 100m strong connection; hence is suitable for end-users whose motors are not located in the same place of a range of 100m. It has a speed transfer rate of about 7Mbps. The Wi-Fi module needs a power source to operate and consumes so much power. Unfortunately, a Wi-Fi module for ten induction motors each is quite expensive.

Another option considered is the Bluetooth Module. The HC-05 module has an ideal range of 10m. It has a transfer rate of about 1Mbps with a goodput of about 720Kbps. Data to be transferred from the sensors include float data types of three temperature values, three acceleration values, current and voltage values, and an array of sixty-four pixels. If data is to be read every 10-15s and a float data type has 4 bytes (32 bits), data to be logged is about **230bps** suitable for an HC-05 BLE module. Unfortunately, Bluetooth cannot connect directly to the cloud and considering that the design holds for more motors to be monitored, a gateway system would be required [19], [20].

(a)

(b)

Figure 4.3 Summary of connectivity options (a) HC-05 Bluetooth Module (b) ESP8266 Wi-Fi Module

The requirement for a cluster of motors is very crucial in this project. It is very costly to provide a Wi-Fi module for all the motors; hence Bluetooth connectivity to a gateway is the selected option for the system.

4.1.2. Gateway Choice

The gateway choice includes the ability to connect several Bluetooth networks links simultaneously. It should also have Wi-Fi/Ethernet properties and a constant source of power and cloud connectivity. The Raspberry Pi Model 4 has most of the properties already mentioned. Its latest models from 3B+ have Bluetooth/Wi-Fi properties and do not need extra physical connections for activation.

Figure 4.4 Raspberry Pi as Gateway

4.2. Final Design

The Final Design for the physical build includes three design choices: Sensor Node Design and the gateway. Consideration for the data types, size and volume, and the variety of the data gave a final flow and design for the system. Data is sampled every 10-15s for the test run to read all design faults and trends accurately.

4.2.1. Circuit Design

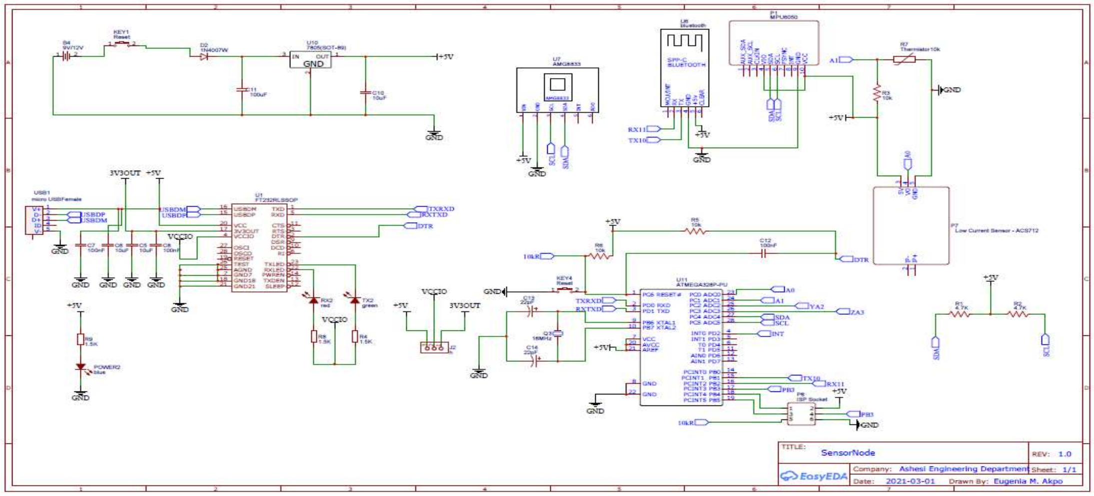


Figure 4.5 Sensor circuit

The diagram in Figure 4.5 is a circuit showing a design of sensor connections to the Atmega328P. The sensors are connected to the analogue pins of the MCU. The circuit also has the blue tooth module attached to the RX and TX of the pin. The circuitry also has an FTDI chip with a USB connector designed to allow easy programming to the microcontroller. This circuit is designed to read the temperature, current and vibration from the motor.

4.3. Software Design

Software design requires programming the Raspberry Pi or Gateway functionality. It also involves selecting the suitable transfer protocol for the system. The HTTP protocol for python interactions on the Raspberry Pi was employed for the Software Design. A server script is run to connect a Bluetooth serial and allow TCP/IP HTTP posts to the database that has already been created.

4.3.1. MCU Software

The Arduino based system runs a program written in the Arduino IDE. It imports all the libraries needed for the sensors to run in the IDE. First, the microcontroller unit is boot loaded using the Arduino IDE system to set it up. The program is then loaded on the ATmega Chip through the FTDI connections or the SPI functions of the board. It collects the readings and stores them as a JSON, which is serialized so the system's serial port or the Bluetooth device can transfer as one piece. The Bluetooth device connects to the gateway over the same baud rate. It is essential to ensure gateway and serial baud rate are the same.

4.3.2. Gateway Software

Raspberry Pi requires installing all the packages for Bluetooth activation, including the Blueman package. A python script is needed to connect to the serial port of the ATmega-based MCU, where a serialized JSON form is received. The byte message is decoded using the proper encoding techniques required for the version of Python installed. The decoded message is then deserialized, and a function is called to connect to the PostgresSQL database and insert it into the database. A copy of the script is found in Appendix A.

4.4. Database Design

The database design is a relational database for collecting information on the Machines. For this project, the PostgresSQL database is used because of the ease of storing arrays. It stores temperature, current, voltage, and an array of pixels from the AMG8833 IR camera and the time of collection. It also has admin details for access and monitoring. It also has letter rating information for each machine under observation.

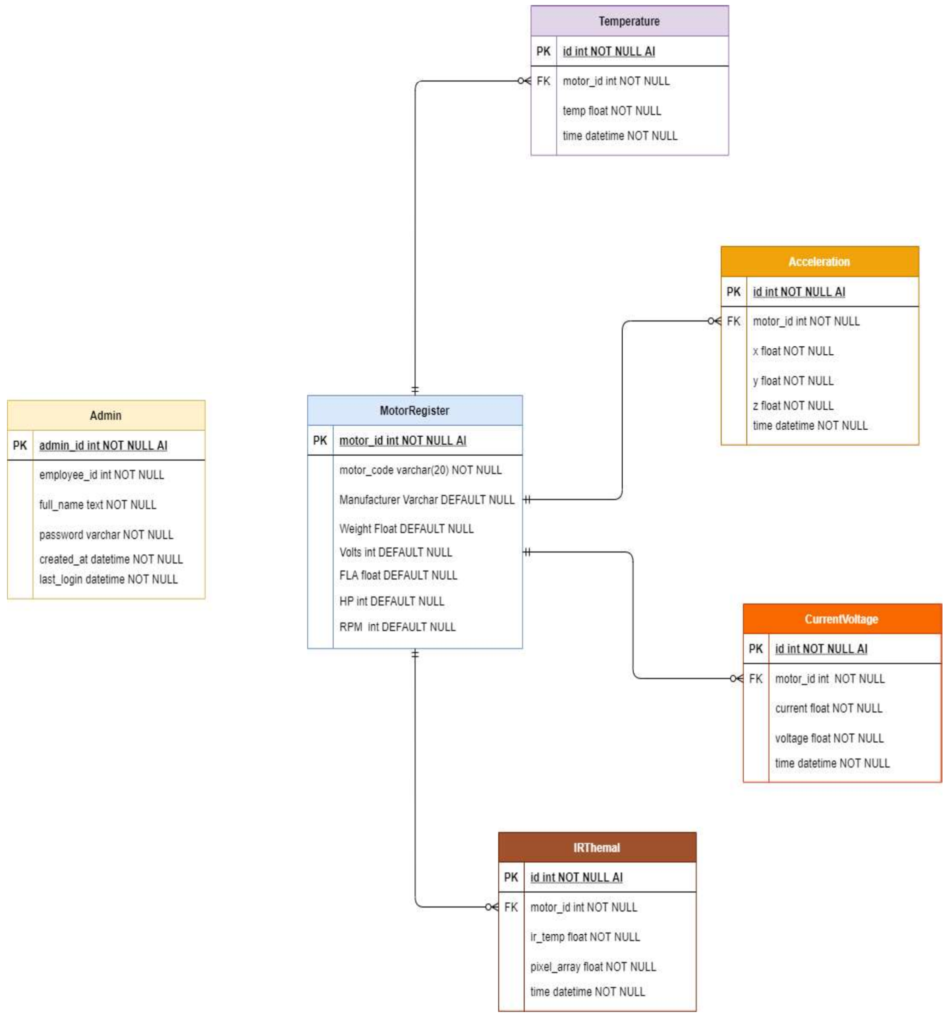


Figure 4.6 Database Interactions

4.5. Dashboard Application Design

The Dashboard design is purely based on Python as it makes the flow between the gateway, database, and user console very smooth. The dashboard is built with an Anvil framework that relies purely on Python. It allows the user to view loggings on vibration velocity (Vrms), temperature and heatmap of the system. It also shows power consumption parameters used for machine learning classification. The essential requirement of the dashboard is to monitor the motor conditions in real-time and make fault inferences from it.

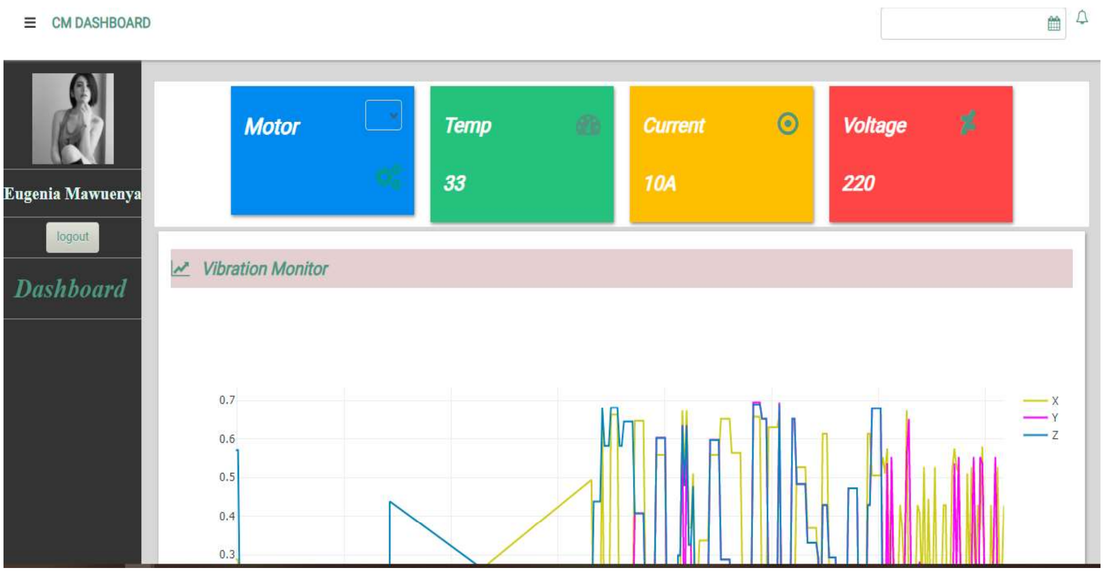


Figure 4.7 Landing page with cards to show summary of current motor under analysis

The figure above shows four different cards. The first contains a dropdown button that allows you to select the motor to observe. Upon selection, the other three cards show the temperature, current and voltage values, respectively. The follow-up diagrams give a brief of other functionalities to expect in the dashboard design.

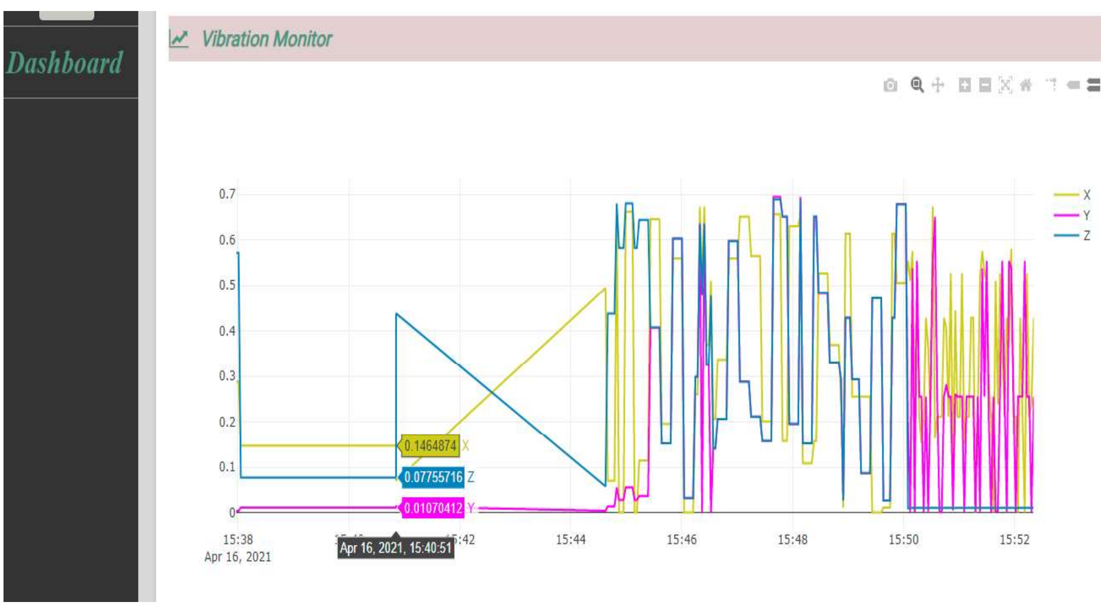


Figure 4.8 Vibration Velocity against time of 3-axis direction

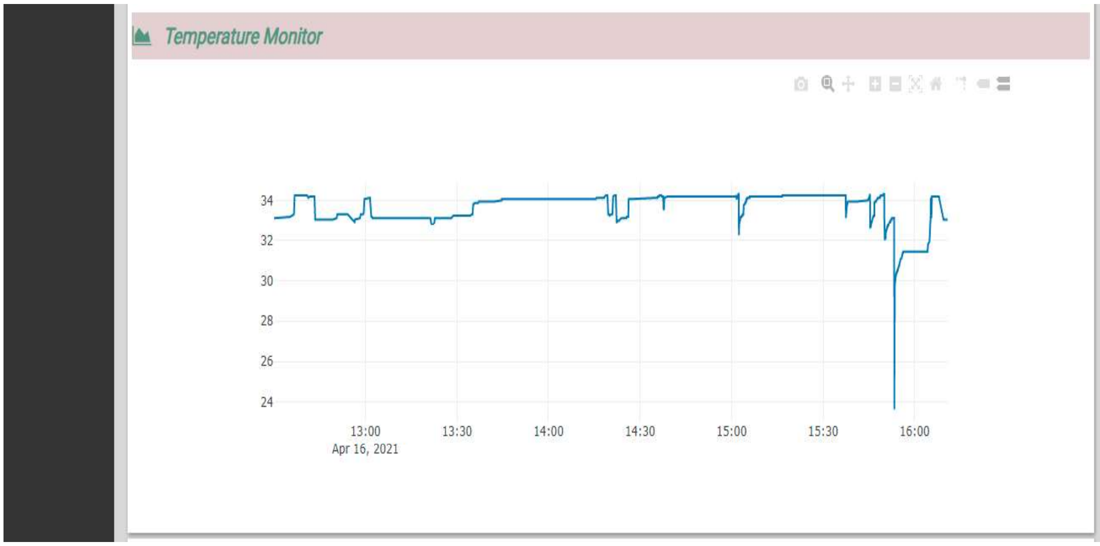


Figure 4.9 Temperature graph

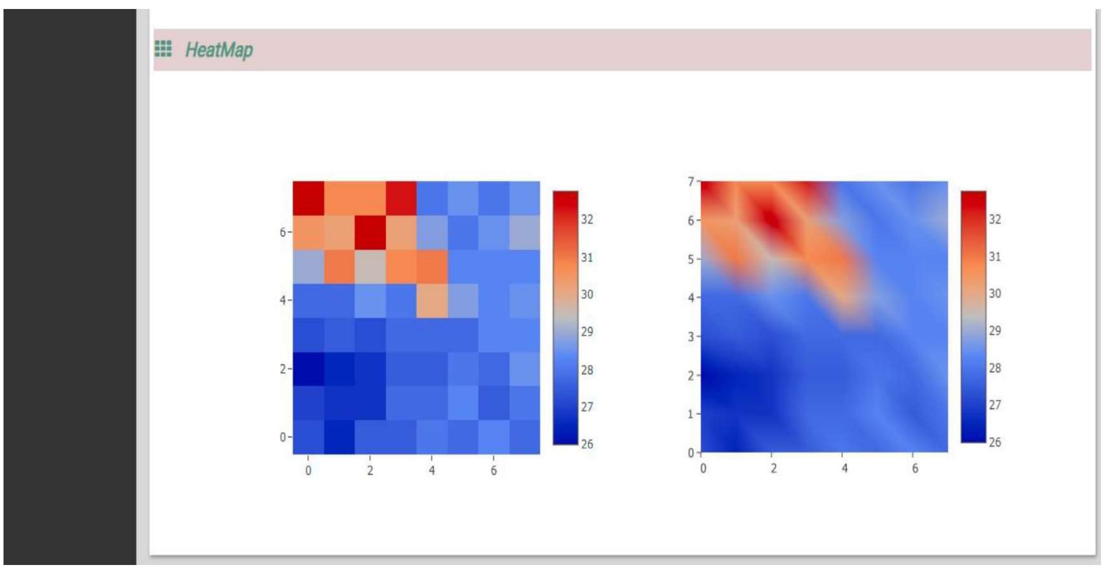


Figure 4.10 Heatmap showing points of high heat

4.6. Machine Learning Predictive Approach

For this project, a machine learning approach towards fault diagnosis and prediction is employed to make the platform more accessible. Based on different machine faults, as explained above, a time series prediction approach is used. The algorithm combines different models on time series forecasting and multinomial classification methods to identify faults that are likely to happen. The diagram in Figure 4.11 explains the machine learning algorithm process.

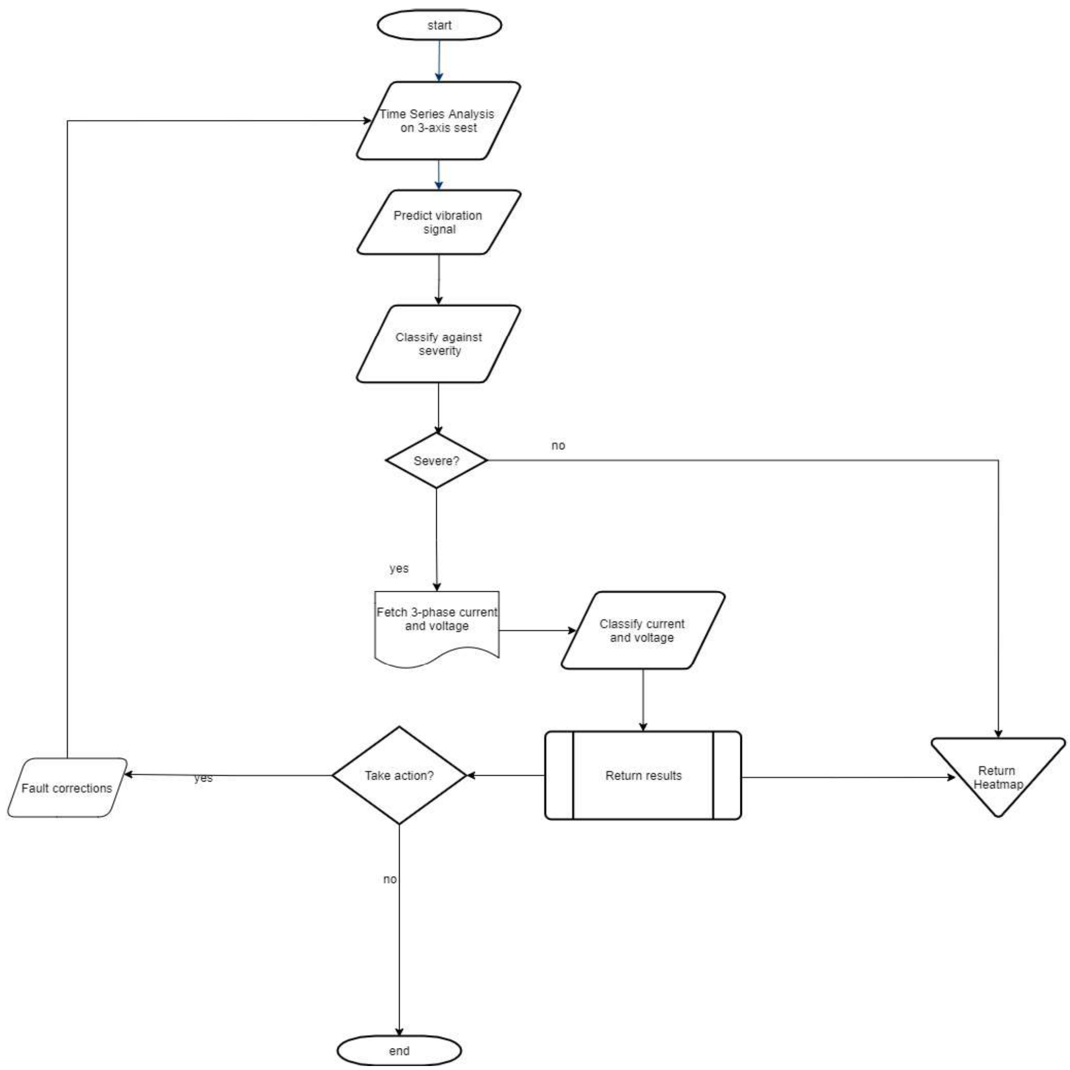


Figure 4.11 Machine Learning Proposed Approach

First, it runs the transformed vibration velocity signal through the ***statsmodel*** library in machine learning forecasting to predict the following few sets of vibration points [5]. It compares the expected values to the ISO 10816 vibration velocity severity norm [6], built on Support Vector Machine (SVM), and returns the severity rating. [21]. The statsmodel library has saved models for running any machine learning programme. In this project, the ARIMA/SARIMAX library will be called to predict values. A follow-up classification is requested based on the current and voltage predictions of the 3-phase motor and produces a type of fault.

5. Chapter 5: Design Testing and Results

This section covers all the test cases and their outcomes, precision, and features. It describes how data was acquired, the design of experiments, the accuracy of data collected, machine learning proposals on fault diagnosis and predictions, and each experiment's results.

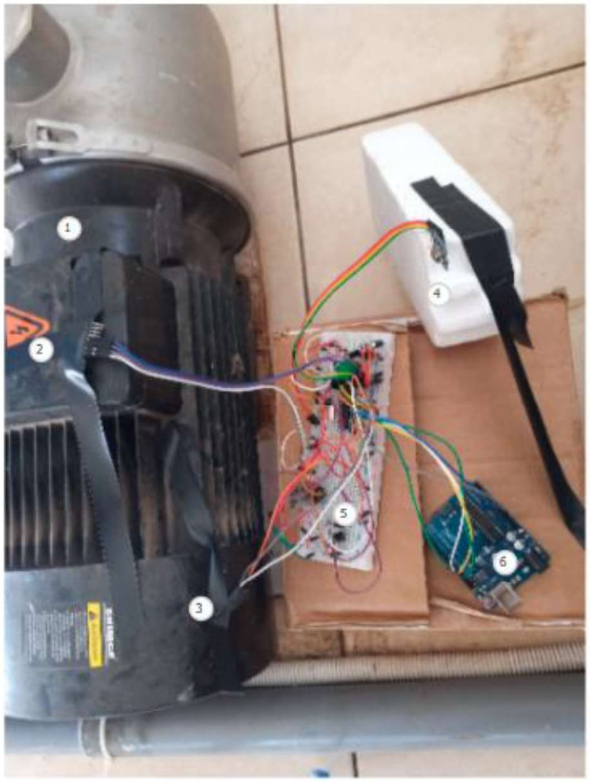
5.1. Data Acquisition and Data Sources

The system is tested on a 3-phase induction motor located at the checkpoint of Ashesi University to ensure that the design system was working without issues and to sample some data for analysis. The motor had a cooling fan; hence temperatures recorded were within a specific range and not high. The motor's rating is as follows:

Table 1 showing the letter rating of the test motor

Type	100L-2
Phase	3
Rated Voltage	380 V
Rated full-load Amperage	6.31 A
Frequency	50 Hz
Weight	24 kg
Power	3kW
Speed	2680r/min
INS.Class	F

Due to restrictions on interfering with wiring, the ACS712 current sensor was not attached to the motor—the motor run from 12:30 PM to pump water to the school. Acceleration and temperature values were recorded and saved into the cloud database. Sensor values were logged every 10s. To ensure that the expected temperature value is known, a multimeter is used to obtain a reading average of 37.8°, 34° and 35.6° on different instances at stability.



1----- 3--Phase Induction motor
2------MPU6050 Accelerometer
3........Thermistor
4.........AMG8833 IR Camera
5.......Sensor Circuit on breadboard
6-......Arduino UNO as ISP

Figure 5.1 Caption of Data Collection

The accelerometer was placed face down and to the hottest part of the motor found, which is the terminal box, because the motor already had a cooling system. The thermistor was also attached to the other side of the terminal box, and the infrared camera was placed a little outward facing the terminal box at about 30cm, as shown in **Figure 5.1**. It is important to remember that the camera specification only captures within 60° of placement. This placement method ensures that data collection and conditions are similar for accurate analysis, as discussed earlier.

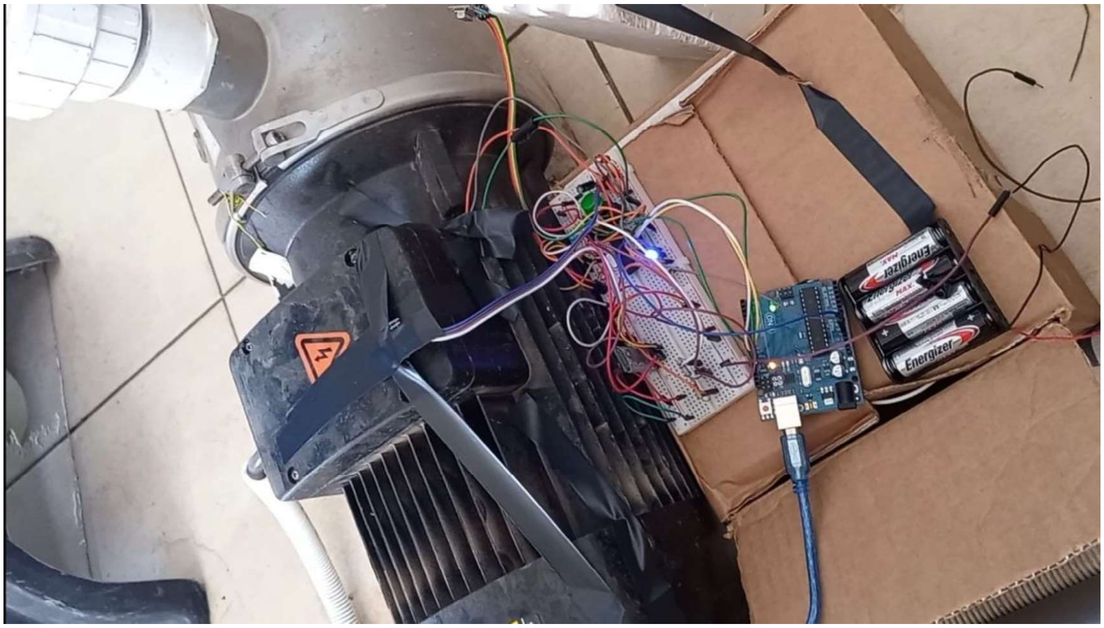


Figure 5.2 Capture of System Logging Values

5.2. Results

The data collected is stored in the PostgreSQL database.

a. **Accelerometer:** Vibration values were read in units of m/s^2. The accelerometer is a triaxial one and takes measurements in three directions. A negative measure indicates movement in the opposite direction of the axis. The sensor placed downwards means a state of 0g, 1g and 0g in the x, y and z directions. The MPU6050 sensor also has the capabilities of reading temperature values; thus, temperature readings were also logged from the accelerometer for analysis. Time of logging is also recorded in the database from vibration analysis to be done. When the circuitry is restarted, initial values of temperature read negative or zero with spikes in the accelerometer, which primarily affects the mean values logged.

b. **Thermistor:** The thermistor reading showed a varying difference from the accelerometer readings by some 3°C which may be due to the placement of the thermistor or poor connections to the leads of its jumper wires. A start in the system logged negative readings as in the case of the accelerometer.

c. The Infrared Camera from observation logged entirely accurate values. In cases of forced start, the pixels returned a zero reading, but the system's temperature is accurately read reads averagely 38°C.

5.3. Statistical Test on Temperature Values

With the temperatures, it is necessary to analyze the variations between them to make the best predictions as, especially, the infrared camera takes temperature readings without contact. A summary and boxplot to analyze the data distribution are shown in **Table 2** and **Figure 5.3**, respectively.

Table 2 Temperature Data Descriptions

	temp_acc	temp_therm	temp_cam
count	230.000000	230.000000	230.000000
mean	34.654717	30.346567	33.582065
std	8.352987	11.001406	0.939049
min	-27.690590	15.324830	30.250000
25%	35.880000	22.435940	33.125000
50%	36.685880	28.487270	34.000000
75%	37.062350	34.299160	34.187500
max	38.027060	98.723940	34.312500

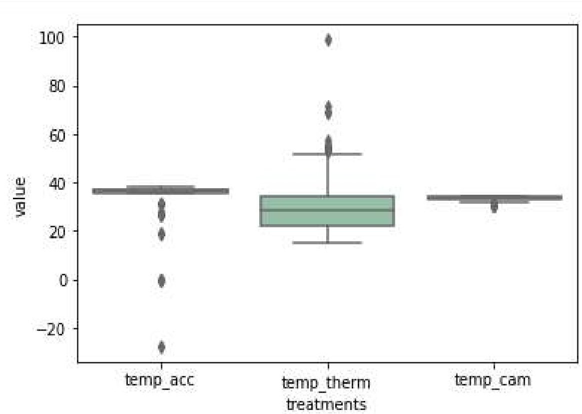


Figure 5.3 Boxplot showing temperature distribution from different sensors.

An ANOVA analysis test for significance in the mean using Ordinary Least Squares (OLS) regression model results from p = 2.1726598040461474e-08. The p-value shows a statistical difference in the mean values of the three sets of data.

18.10577150936454 2.1726598040461474e-08

	df	sum_sq	mean_sq	F	PR(>F)
C(treatments)	2.0	2313.738179	1156.869089	18.105772	2.172660e-08
Residual	687.0	43895.896067	63.895045	NaN	NaN

Figure 5.4 Ordinary Least Squares (OLS) results

With that, a post-hoc analysis (Tukey-HSD) is conducted to find out exactly where the difference is. Table 3 shows a statistical difference between all groupings except between the accelerometer temperature and the camera temperature. This means the accelerometer and IR camera gave very close readings and show high precision compared to the known temperature value of 38°C.

Table 3 Tukey's honestly significantly differenced (HSD) test

	group1	group2	Diff	Lower	Upper	q-value	p-value
0	temp_acc	temp_therm	4.308150	2.557327	6.058974	8.173747	0.001000
1	temp_acc	temp_cam	1.072652	-0.678172	2.823476	2.035116	0.321958
2	temp_therm	temp_cam	3.235499	1.484675	4.986322	6.138631	0.001000

The Shapiro-Wilk test is then used to check the test statistics, which returned from p=3.3667967366824627e-34, which shows that the data is not drawn from a normal distribution; hence each one cannot be used to substitute the other. By observation, data from the accelerometer temperatures and the camera lack a wide distribution. Temperature readings from the thermistor have a wide distribution and cannot be relied on to make tests inferences on the induction motor.

5.4. Fault Diagnosis Based Current and Voltage Parameters

Due to the inability to collect current and voltage data on the motor described above, a sample data collected for Artificial Neural Network (ANN) is used for the current analysis. This fault diagnosis journal provides data on diagnosing seven classes of faults, including overload, ground fault, locked rotor, single phasing, over-voltage, under-voltage, and unbalanced supply voltage [5]. The sample data contains voltage and current values sampled for three different sets of each parameter. A glimpse of the database is seen in Table 4:

Table 4 Sample Data for time series forecasting

	V1	V2	V3	I1	I2	I3	Fault
0	2.639855	2.598443	2.689093	0.405013	0.424213	0.414541	No Fault
1	2.639127	2.597583	2.689994	0.413220	0.432774	0.423409	No Fault
2	2.648514	2.638497	2.716745	0.634211	0.638780	0.008767	Overload
3	2.649291	2.639254	2.690204	0.005936	0.685578	0.689449	Overload
4	2.685974	2.870574	2.956132	0.398647	0.531207	0.473744	Ground Fault
5	2.682843	2.870628	2.953648	0.398442	0.531102	0.472787	Ground Fault
6	1.452803	1.449902	1.441935	0.245231	0.786128	0.234152	Locked Rotor
7	1.086249	1.086596	1.083130	0.231117	0.781319	0.225809	Locked Rotor
8	2.868790	2.875877	2.860353	0.483453	0.838635	0.496879	Unbalanced Voltage
9	2.853745	2.876647	2.861104	0.484121	0.829710	0.497440	Unbalanced Voltage
10	2.640762	2.603945	2.678396	0.848768	0.830192	0.844275	Over Voltage
11	2.640960	2.604793	2.678446	0.848640	0.830029	0.843615	Over Voltage
12	2.661374	2.613395	2.688907	1.416786	1.397752	1.411372	Under Voltage
13	2.657179	2.613409	2.687374	1.671357	1.650515	1.668700	Under Voltage

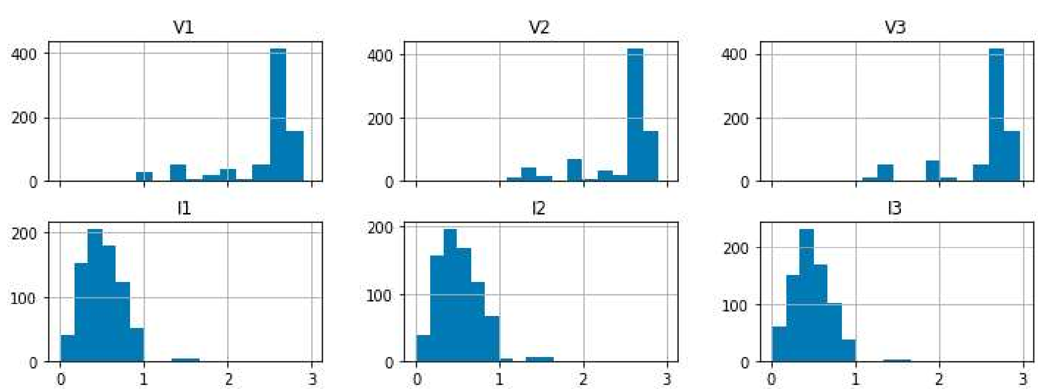


Figure 5.5 Histogram plots of all 6 dimensions

The six columns of data in Table 4 represent the voltages and currents received from the 3-phase line of the motor used for the experiments. The fault diagnosis is based on a simple Support Vector Machine (SVM) in Python. SVM is a supervised machine learning model suitable for classification and regression [21]. Data points are plotted in n-dimensional space with the feature values in a specific coordinate followed by classification by finding the hyper-plane to differentiate the classes. The target feature classification includes the seven faults listed above. The accuracy of this classification method returned 100% for a test size of 30/150 as seen in Figure 5.6, which means the classification aspect of our proposed model is accurate enough to classify the current and voltage distributions coming into a signal.

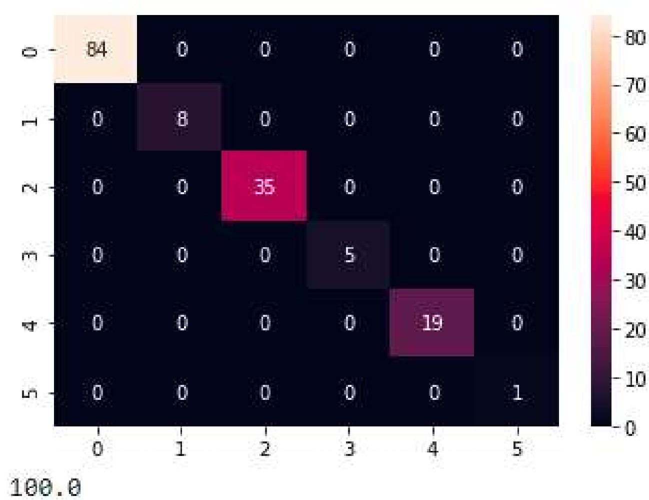


Figure 5.6 Confusion Matrix to visualize accuracy of the model

It is vital to note that the current and voltage values are not from the motor sample described earlier but is data sourced online only to prove the proposed machine learning approach [23].

5.5. Time Series Analysis of the Accelerometer readings

The proposed machine learning predictive maintenance approach requires that the vibration signal is predicted in time series, and the resulting prediction is classified under the ISO 10816 standard [6]. The various standards for vibration monitoring and induction motor condition monitoring classify severity using the Velocity of the signal in in/s or mm/s. Velocity against time graph represents the acceleration of the system. A graph of the Vibration Velocity (Vrms) against the x-direction of the three signals captured is shown in ***Figure 5.7.***

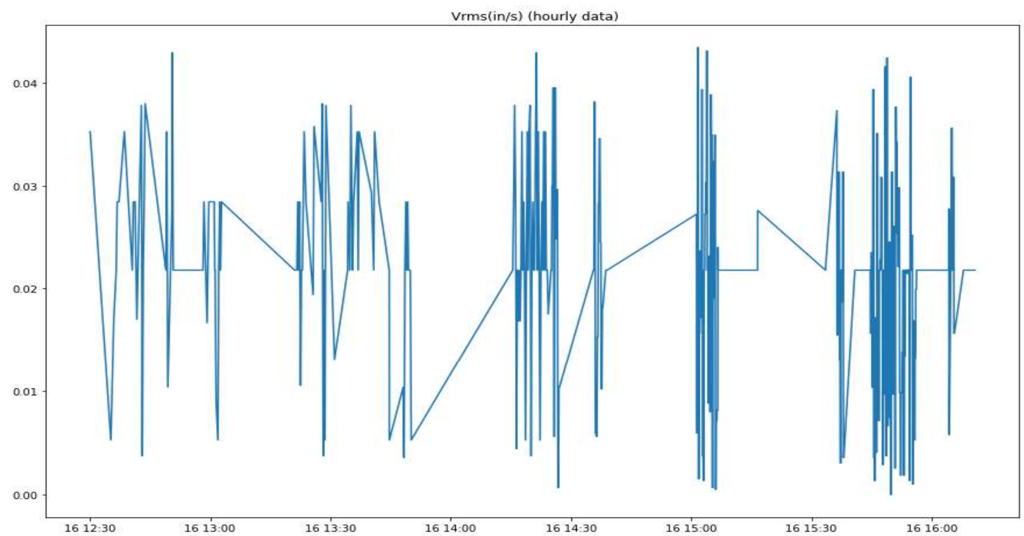


Figure 5.7 Graph of Vrms of the x-direction against time

The time series algorithm uses the ***stats models*** library to produce a forward time graph [24]. The model grows by first getting the Moving Average (MA). It then uses an exponential smoothing formula to get a model value between the current and previous model values. Smoothing options are explored to obtain the best model for the time graph, and then time-series cross-validation is introduced. This is also achieved by trying to catch various anomalies in the system using the HoltWinters approach.

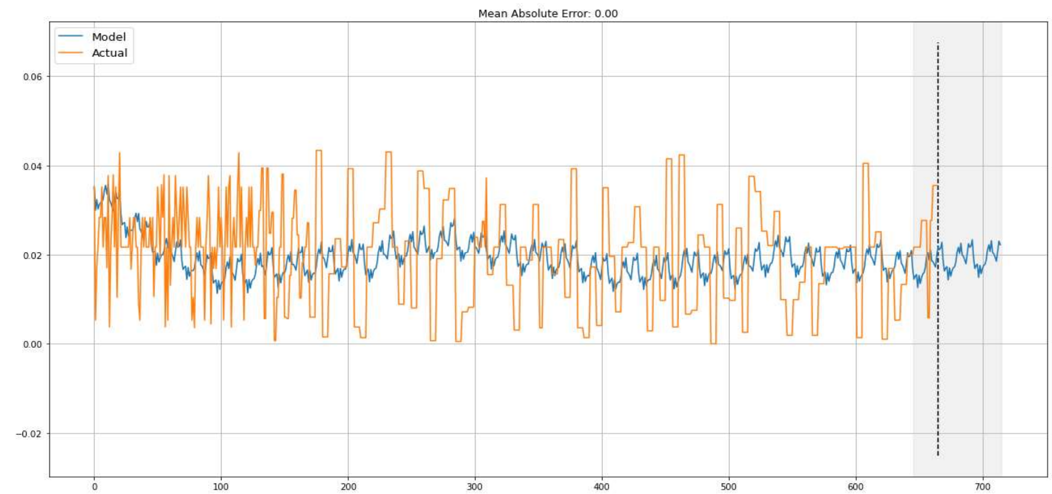


Figure 5.8 The graph of the model and the actual values

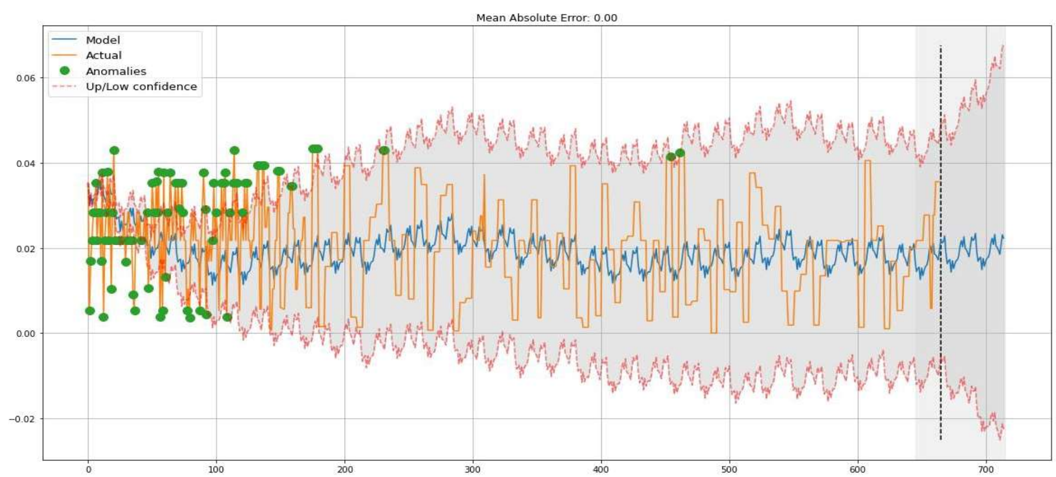


Figure 5.9 The graph of the model, actual and the anomalies with the confidence intervals

The graphs in Figure 5.8 and 5.9 show the model graph and the actual graph values to predict the following twenty. Then stationarity is checked using the Dickey-Fuller test. A stationary signal does not change its statistical properties (mean and variance) over time [24]; hence, the preferred choice is a stationary signal for time series analysis. The test returned a p-value of 0.081 at $\rho=1$, which signifies non-stationarity in the system. The non-stationarity was removed by shifting using seasonality of the data viewed in ACF and PACF, leading to building SARIMA.

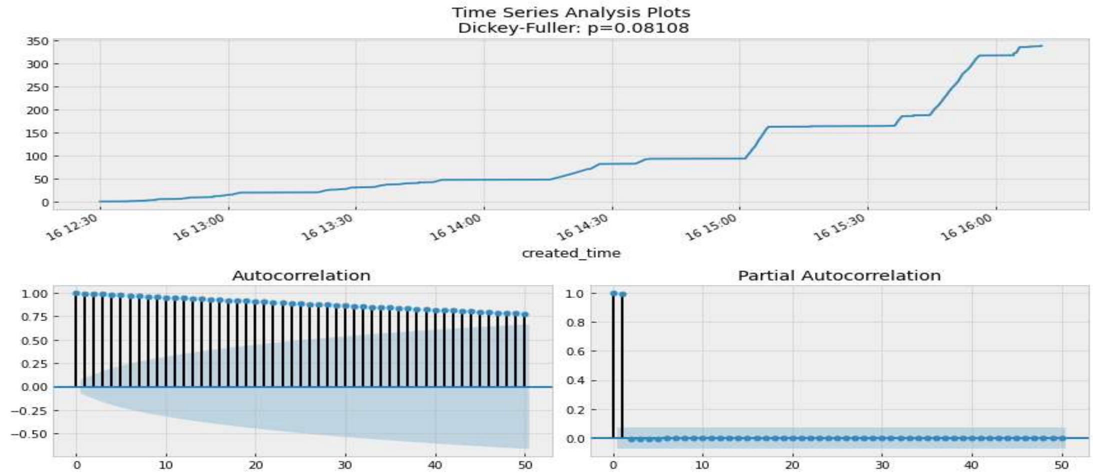


Figure 5.10 Initial graph with p-value=0.081 showing non-stationarity

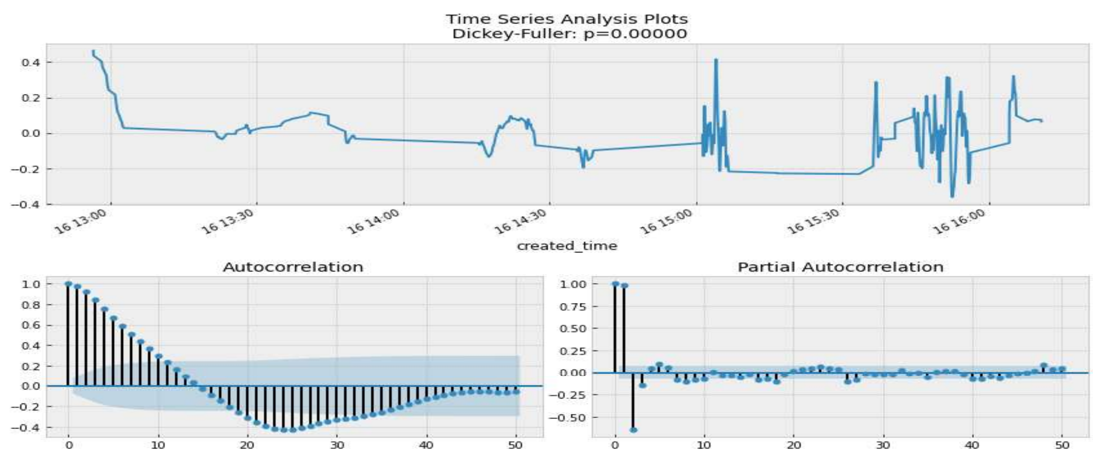


Figure 5.11 Graph showing removal of non-stationarity.

Due to the SARIMAX model, including seasonality, the resulting response is shown and returns accurate PACF and ACF plots. We can confidently move on to forecast and give predictions [24]. The statistics show obtaining accurately predicted signal values classified against ISO standards to show the fault in the system and its severity level.

```
                              Statespace Model Results
==========================================================================================
Dep. Variable:                             vrm_inch   No. Observations:                685
Model:             SARIMAX(2, 1, 4)x(0, 1, 2, 24)   Log Likelihood                2171.841
Date:                            Sat, 24 Apr 2021   AIC                          -4325.682
Time:                                    04:16:38   BIC                          -4285.252
Sample:                                         0   HQIC                         -4310.011
                                            - 685
Covariance Type:                              opg
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          1.5749      0.058     27.211      0.000       1.461       1.688
ar.L2         -0.6292      0.060    -10.485      0.000      -0.747      -0.512
ma.L1          0.6570      0.051     12.796      0.000       0.556       0.758
ma.L2          0.5249      0.056      9.327      0.000       0.415       0.635
ma.L3          0.4441      0.049      9.048      0.000       0.348       0.540
ma.L4          0.3224      0.033      9.805      0.000       0.258       0.387
ma.S.L24      -0.8780      0.042    -20.753      0.000      -0.961      -0.795
ma.S.L48       0.0788      0.040      1.984      0.047       0.001       0.157
sigma2      7.714e-05   3.54e-06     21.774      0.000    7.02e-05    8.41e-05
===================================================================================
Ljung-Box (Q):                       32.70   Jarque-Bera (JB):               177.09
Prob(Q):                              0.79   Prob(JB):                         0.00
Heteroskedasticity (H):               0.35   Skew:                            -0.04
Prob(H) (two-sided):                  0.00   Kurtosis:                         5.54
===================================================================================
```

Figure 5.12 Graph showing removal of non-stationarity.

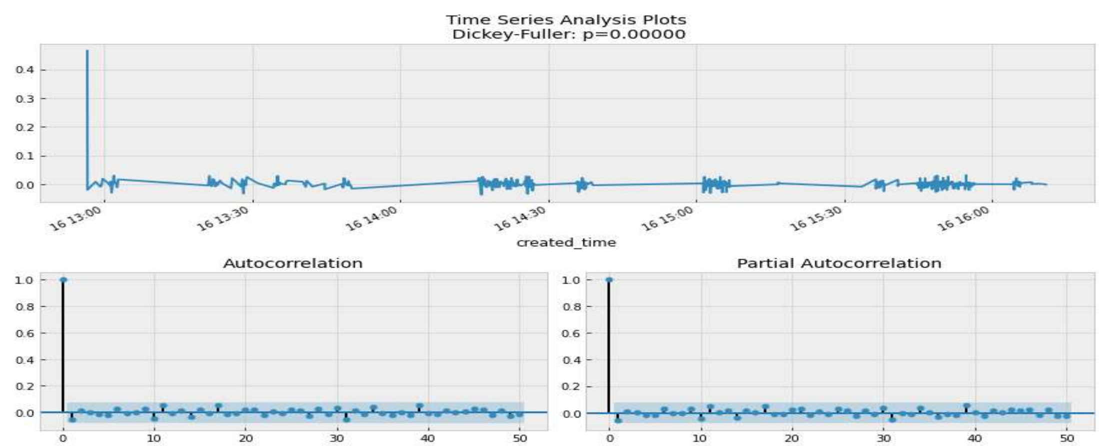
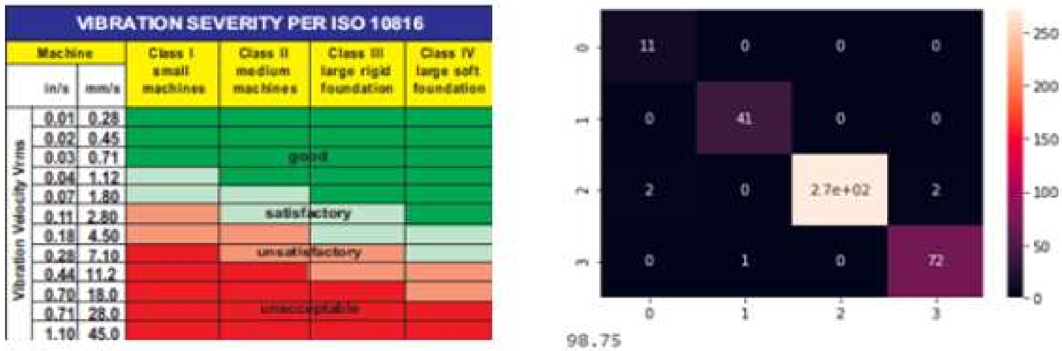


Figure 5.13 Graph showing removal of non-stationarity

5.6. Severity Classifier

The severity classifier uses the same classifier function as the fault classifier. ISO 10816 severity classifier is shown in the figure below.



(a) (b)

Figure 5.14 (a)ISO 10816 Severity Chart (b) Heatmap of Severity Classifier.

The chart in Figure 5.14 shows that severity levels are also based on the Class provisions provided by the standard. The sample motor used for the experiment is a Class 1 motor. The classifier predicts the severity of the signals moves on to classify current and voltage options entering the system at that instant. Sample classification into the good, satisfactory, unsatisfactory and alarm or unacceptable states gives an accuracy of 98.75%, as displayed in the heat map above [23].

5.7. Heatmap Capture

The AMG8833 IR Camera is an eight by eight camera that captures the thermal profiles of objects. It can capture within 60° of spread, as mentioned earlier. **Figure 5.15** shows different placement angles of the motor terminal from which most readings were captured. The first row shows different tests of obtaining the right angle and distance from the object. It is noted that a design for this purpose should consider the distance of placement or attachment and cannot be attached to the sensor board as one whole unit. An enclosure design is therefore needed for exclusive use. Another observation is that the pixel size does not give high visibility but only a quick outline of high-temperature observations. **Figure 4.1(e)** is an image obtained from several motor images captured for fault diagnosis via Image Classification [25]. It is observed that the image provides a more apparent thermal profile compared to the heatmap diagram in Figure 5.15, so higher pixel options can be explored.

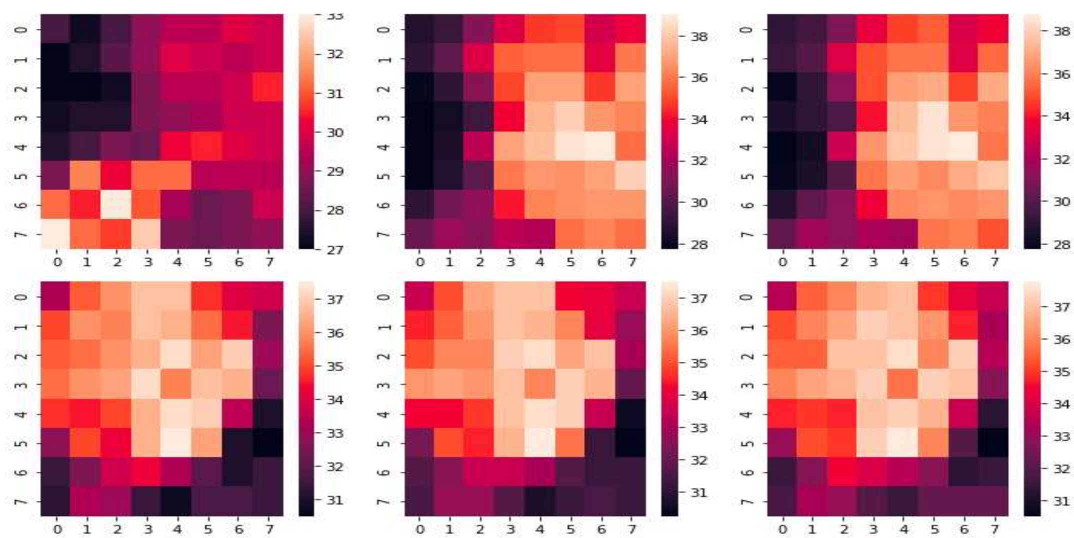


Figure 5.15 Heatmap observations from IR Camera

6. Chapter 6: Conclusion

Integrating the concept of IoT into condition monitoring techniques made the process easier. There is available data for analysis and prediction towards predictive maintenance technique in Condition Monitoring. In this project, a set of sensors was used to collect data from the machine based on everyday parameter interactions from the motor, such as bearing vibration, temperature, current and voltage consumption, and a circuit to collect data from the source. The data was transferred serially to the Raspberry Pi, which is the gateway. The Raspberry pi has a dual Bluetooth and Wi-Fi functionality that allows interaction with the internet. The deserialized data was stored in the cloud database, where the user gets access to the data on the dashboard. A machine learning approach was introduced for fault diagnosis easier and to aid the predictive maintenance process. The IoT system was designed to replace expensive processes and platforms available on the market. The system can measure temperature, current and voltage effectively. The dashboard shows all parameters needed to monitor the computer over the internet effectively. This project proposed an IoT-based design for Condition Monitoring and proposed a Machine Learning approach to aid the process using time series analysis and logistic regression to classify faults. Data collection and logging was successful, and data can be retrieved from the database.

6.1. Limitations and Future Work

1. Due to constraints associated with the data collection process, the machine learning approach was not thoroughly tested on integration with the gateway for effective real-time monitoring. Data acquired from cited sources could only be compared in conditions and may not show the actual output of the test motor in the experiment.
2. Enclosure options could not be explored for the project and could further be explored.

3. IR Camera only provides thermal profile in thin outlines; hence the visibility of faults could not be explored at maximum.
4. Data gathered did not cover over 24 hours of seasonal working of the motor.
5. The design system should include an enclosure for the circuit design for attachment to the testbed.
6. Data can be gathered over an extended to produce higher accuracy of prediction results.
7. A major update is needed on the dashboard to run a real-time analysis while data is being logged.

7. References

[1] N. Khan, F. Rafiq, F. Abedin, and F. U. Khan, "IoT based health monitoring system for electrical motors," in *2019 15th International Conference on Emerging Technologies (ICET)*, 2019.

[2] K. M. Siddiqui, K. Sahay, and V. K. Giri, "Health monitoring and fault diagnosis in induction motor-a review"," *International Journal of Advanced Research in Electrical Electronics and Instrumentation Engineering*, vol. 3, no. 1, pp. 6549–6565, 2014.

[3] I. F. All, "What is IoT? - A simple explanation of the internet of things," *Iotforall.com*, 01-Feb-2021. [Online]. Available: https://www.iotforall.com/what-is-internet-of-things. [Accessed: 27-Apr-2021].

[4] "IoT (Internet of Things) – YeniWave Telecom," *Yeniwave.com*. [Online]. Available: https://www.yeniwave.com/iot-internet-of-things/. [Accessed: 27-Apr-2021].

[5] V. Kavana and M. Neethi, "Fault analysis and predictive maintenance of induction motor using machine learning," in *2018 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*, 2018.

[6] *Intech2000.com*. [Online]. Available: https://www.intech2000.com/downloads/web/54/54.1/Applying%20Examiner.pdf. [Accessed: 27-Apr-2021].

[7] S. S. Goundar, M. R. Pillai, K. A. Mamun, F. R. Islam, and R. Deo, "Real-time condition monitoring system for industrial motors," in *2015 2nd Asia-Pacific World Congress on Computer Science and Engineering (APWC on CSE)*, 2015.

[8] D. Ganga and V. Ramachandran, "IoT-based vibration analytics of electrical machines," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4538–4549, 2018.

[9] Reliabilityweb, "How is vibration measured?," *Reliabilityweb.com*. [Online]. Available: https://reliabilityweb.com/articles/entry/how_is_vibration_measured/. [Accessed: 27-Apr-2021].

[10] H. Tang, S. Lu, G. Qian, J. Ding, Y. Liu, and Q. Wang, "IoT-based signal enhancement and compression method for efficient motor bearing fault diagnosis," *IEEE Sens. J.*, vol. 21, no. 2, pp. 1820–1828, 2021.

[11] A. P. Junfithrana, I. Solikin, J. K. Hadi, and M. Dedy Novianto, "Improving electric motor performance by handling interference using IOT," in *2019 5th International Conference on Computing Engineering and Design (ICCED)*, 2019.

[12] "Time series analysis," *Statisticssolutions.com*, 21-Dec-2010. [Online]. Available: https://www.statisticssolutions.com/time-series-analysis/. [Accessed: 27-Apr-2021].

[13] "IoT: Architecture," *M2mology.com*. [Online]. Available: https://www.m2mology.com/iot-transformation/iot-world-forum/. [Accessed: 27-Apr-2021].

[14] *Electronics-tutorials.ws*. [Online]. Available: https://www.electronics-tutorials.ws/io/thermistors.html. [Accessed: 27-Apr-2021].

[15] "MPU6050 Pinout, Configuration, Features, Arduino Interfacing & Datasheet," *Components101.com*. [Online]. Available: https://components101.com/sensors/mpu6050-module. [Accessed: 27-Apr-2021].

[16] EG Projects, "How to measure current using Arduino and ACS712 Current Sensor," *Engineersgarage.com*, 20-Apr-2021. [Online]. Available: https://www.engineersgarage.com/arduino/acs712-current-sensor-with-arduino/. [Accessed: 27-Apr-2021].

[17] A. B. Haripriya, K. A. Sunitha, and B. Mahima, "Development of low-cost thermal imaging system as a preliminary screening instrument," *Procedia Comput. Sci.*, vol. 172, pp. 283–288, 2020.

[18] Shawn, "ATmega328P, the one microcontroller you should start with," *Seeedstudio.com*, 22-Oct-2019. [Online]. Available: https://www.seeedstudio.com/blog/2019/10/22/atmega328p-the-one-microcontroller-you-should-start-with/. [Accessed: 27-Apr-2021].

[19] "Bluetooth vs Wi-Fi," *Diffen.com*. [Online]. Available: https://www.diffen.com/difference/Bluetooth_vs_Wifi. [Accessed: 27-Apr-2021].

[20] K. S. Amith, A. G. V. Sisir, S. Abhilash, and Y. Rijal, "Comparison of different communication devices for Arduino Uno," *Ijert.org*. [Online]. Available: https://www.ijert.org/research/comparison-of-different-communication-devices-for-arduino-uno-IJERTCONV6IS15033.pdf. [Accessed: 27-Apr-2021].

[21] S. Ray, "Understanding Support Vector Machine(SVM) algorithm from examples (along with code)," *Analyticsvidhya.com*, 12-Sep-2017. [Online]. Available: https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/. [Accessed: 27-Apr-2021].

[22] R. Bedre, "ANOVA using Python (with examples)," *Reneshbedre.com*, 22-Oct-2018. [Online]. Available: https://www.reneshbedre.com/blog/anova.html. [Accessed: 27-Apr-2021].

[23] venkygajjala, "Iris Dataset Classification," *Kaggle.com*, 01-Nov-2018. [Online]. Available: https://www.kaggle.com/venkygajjala548/iris-dataset-classification. [Accessed: 27-Apr-2021].

[24] [Online]. Available: http://ttps://www.kaggle.com/kashnitsky/topic-9-part-1-time-series-analysis-in-python/notebook. [Accessed: 27-Apr-2021].

[25] *Mendeley.com*. [Online]. Available: https://data.mendeley.com/datasets/m4sbt8hbvk/2. [Accessed: 27-Apr-2021].

8. Appendix A

a. Import all packages and module libraries

```python
#!/usr/bin/env python3
#!/usr/bin/python -tt
# -*- coding: utf-8 -*-
"""
A simple Python script to receive messages from a client over
Bluetooth using Python sockets (with Python 3.3 or above).
"""
import socket
import bluetooth
import numpy as np
import time
import json
import datetime
import serial
import os
from datetime import datetime
import psycopg2
import numpy as np


port = "/dev/rfcomm0" #Raspberry pi serial port to bluetoorh
ser = serial.Serial(port, 9600)
hostMACAddress = '98:D3:51:FE:OE:4A' # The MAC address of a Bluetooth adapter o$
```

b. Write a function to connect to the database

```python
def sensor_Data_Handler(data): # Create a connection object
    databaseServerIP = "192.168.43.101"  # IP address of the MySQL database ser$
    databaseUserName = "piuser" #User name of the database server
    databaseUserPassword = "pipasswd" # Password for the database user
    newDatabaseName = "hydropidb" # Name of the database that is to be created
    connectionInstance = psycopg2.connect(database="datadb", user="pi",password="pi", host="127.0.0.1", port="5432")
```

```python
    try:
        cursorInsatnce = connectionInstance.cursor()#Create a cursor object
        try:
            cursorInsatnce.execute(cursorInsatnce.mogrify("""INSERT INTO datab(motor_code,temp_lm35,m_current,voltage,x_acc,y_acc,z_acc,temp_acc,pixels,temp_cam)
             VALUES (%s,%s,%s,%s,%s,%s,%s,%s,ARRAY[%s],%s)""",('M001',data['Temp_lm35'],data['Current'],data['Voltage'],data['X_Acc'],
                                                data['Y_Acc'],data['Z_Acc'],data['Temp_Acc'],arra1,data['Temp_Cam']),))
            cursorInsatnce.execute(cursorInsatnce.mogrify("""INSERT INTO camera(motor_code,pixels,temp_cam)
            VALUES (%s,ARRAY[%s],%s)""",('M001',arra1,data['Temp_Cam']),))
            cursorInsatnce.execute(cursorInsatnce.mogrify("""INSERT INTO temperature(motor_code,temp_lm35)
            VALUES (%s,%s)""",('M001',data['Temp_lm35']),))
            cursorInsatnce.execute(cursorInsatnce.mogrify("""INSERT INTO currvolt(motor_code,m_current,voltage)
             VALUES (%s,%s,%s)""",('M001',data['Current'],data['Voltage']),))
            cursorInsatnce.execute(cursorInsatnce.mogrify("""INSERT INTO acceleration(motor_code,x_acc,y_acc,z_acc,temp_acc)
            VALUES (%s,%s,%s,%s,%s)""",('M001',data['X_Acc'],data['Y_Acc'],data['Z_Acc'],data['Temp_Acc'],),))
            connectionInstance.commit()
            print("Inserted Data into Database.")
            print("")
        finally:
            connectionInstance.close()
    except Exception as e:
        print("Exeception occured:{}".format(e))
```

c. Write necessary functions for conversion of JSON, array and string data types

```
def buildArray(theList):
    arr = '{'
    for i in theList:
        arr = arr + str(i) + ','
    return arr[:-1] + '}'
```

d. Connect to Bluetooth serial and read data from the sensors and Bluetooth module

```
print("Waiting for data...")

data = ""

while (True):
    if (ser.inWaiting() > 0):
        character= ser.read()
        data+= character.decode("utf-8", "ignore")
        try:
            character= json.loads(data)
            print("Received chunk", character)
            arra = np.reshape(character['Pixels'],(8,8))
            arra1 = arra.tolist();
            sensor_Data_Handler(character)
            data = ""
        except json.JSONDecodeError as e:
            continue
```