Project Report

On

“Connected Fleet : A one step solution for fleet operations with optimized security and productivity”

Submitted in partial fulfillment for the award of

Post Graduate Diploma in Internet of Things

(PG-DIOT) from C-DAC ACTS (Pune)

Guided by:

Mr. Bhupendra Pratap Singh

Presented by:

Mr. Abdur Kidwai	**Prn: 220340126001**
Mr. Abhishek Sahu	**Prn: 220340126002**
Mr. Sharukh Atar	**Prn: 220340126005**
Mr. Nimesh Nagar	**Prn: 220340126018**
Mr. Tushar Jadhav	**Prn: 220340126030**

Centre of Development of Advanced Computing (C-DAC) ACTS, Pune

CERTIFICATE

TO WHOMSOEVER IT MAY CONCERN

This is to certify that

Mr. Abdur Kidwai	**Prn: 220340126001**
Mr.Abhishek Sahu	**Prn: 220340126002**
Mr. Sharukh Atar	**Prn:220340126005**
Mr. Nimesh Nagar	**Prn:220340126018**
Mr. Tushar Jadhav	**Prn:220340126030**

have successfully completed their project

on

“Connected Fleet : A one step solution for fleet operations with optimized security and productivity”

Under the guidance of

Mr. Bhupendra Pratap Singh

ABSTRACT

Fleet management systems are increasingly changing in various transport companies. Although according to claims of different producers, the usage of such systems can achieve great savings in the business operation, the studies show that for optimal functioning of such systems it is necessary to carry out the evaluation of the company and the very systems according to several criteria.

The methodology that allows the FM implementation within the transport companies starts from the analysis of the characteristics of carriers, characteristics of program packages, which is followed by the evaluation according to several criteria which influence the final choice.

The objective of such approach is the adaptation to the users' requirements which eventually results in the improvement and savings in the operation.

KeyWords : FM (Fleet Management), IoT (Internet of Things).

ACKNOWLEDGEMENT

This project **“Connected Fleet: A One Step Solution for Fleet Operations with Optimized Security and Productivity”** was a great learning experience for me and I am submitting this work to Advanced Computing Training School **(CDAC ACTS)**.

I am very glad to mention the name of ***Mr. Bhupendra Pratap Singh*** for his valuable guidance to work on this project. His guidance and support helped me to overcome various obstacles and intricacies during the course of project work.

I am highly grateful to Ms. Risha P.R. (Joint Director (ACTS training Centre), C-DAC, for her guidance and support whenever necessary while doing this course Post Graduate Diploma in ***Internet of Things (PG-DIOT)*** through C-DAC ACTS, Pune.

My heartfelt thanks go to ***Ms. Seema Sajeevan*** (Course Coordinator, PG-*DIOT*) who gave all the required support and kind coordination to provide all the necessities like required hardware, internet facility and extra Lab hours to complete the project and throughout the course up to the last day here in C-DAC ACTS, Pune.

From:
Mr. Abdur Kidwai (220340126001)
Mr. Abhishek Sahu (220340126002)
Mr. Sharukh Atar (220340126005)
Mr. Nimesh Nagar (220340126018)
Mr. Tushar Jadhav (220340126030)

Chapter 1

Introduction

1.1 Fleet Management

Fleet management is an administrative approach that allows companies to organize and coordinate work vehicles to improve efficiency, reduce costs, and provide compliance with government regulations. While most commonly used for vehicle Tracking, fleet management includes other use cases such as mechanical diagnostics and driver behavior. Automated fleet management solutions to connect vehicles and monitor driver activities, allowing managers to gain insight into fleet performance and driver behavior. This enables managers to know where vehicles and drivers are at all times, identify potential problems and mitigate risks before they become larger issues that can jeopardize client satisfaction, impact driver safety, or increase costs.

Fleet Management System, as described above, can lead to error free, secure and reliable fleet management system. It can assist the user to concentrate on their other activities rather to concentrate on the record keeping. Thus, it will help organization in better utilization of resources.

Every organization, whether big or small, has challenges to overcome and managing the information of Driver, Fleet, Maintenance, Fleet Type, Speed. Every Fleet Management Systems that are adapted to your managerial requirements.

This is designed to assist in strategic planning, and will help you ensure that your goals. Also, for those busy execute who are always on the go, our systems come with remote access features, which will allow you to manage your workforce anytime, at all times. These systems will ultimately allow you to better manage resources.

1.2 Purpose

Our project aims at business process automation, I.e., we have tried to computerize various processes of Fleet Management system.

- In computer system the person has to fill the various forms & number of copies of the forms can be easily generated at a time.
- In computer system, it is not necessary to create the manifest but we can directly print it, which saves our time.
- To assist the staff in capturing the effort spent on their respective working areas.
- To utilize resources in an efficient manner by increasing their productivity through automation.
- The system generates types of information that can be used for various purposes.
- It satisfies the user requirement.
- Be easy to understand by the user and operator.
- Be easy to operate.
- Have a good user Interface (UI).
- Be Expandable
- Delivered on schedule within budget.

1.3 Project Flow :

The key feature of this project is:

Users can monitor all these parameters on Grafana which is the data visualization tool. Users can adjust or modify accordingly. If a user does not follow the alert on Grafana, email will be sent on his registered mail id. Below fig.1.1 is the use case diagram.

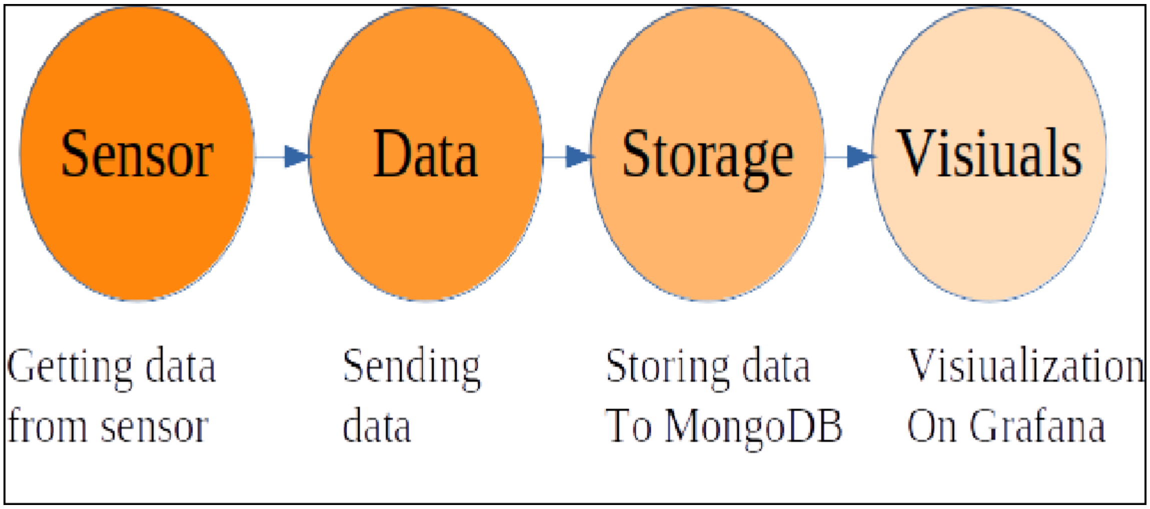


Fig.1.1 Project flow

1.4 Block Diagram:

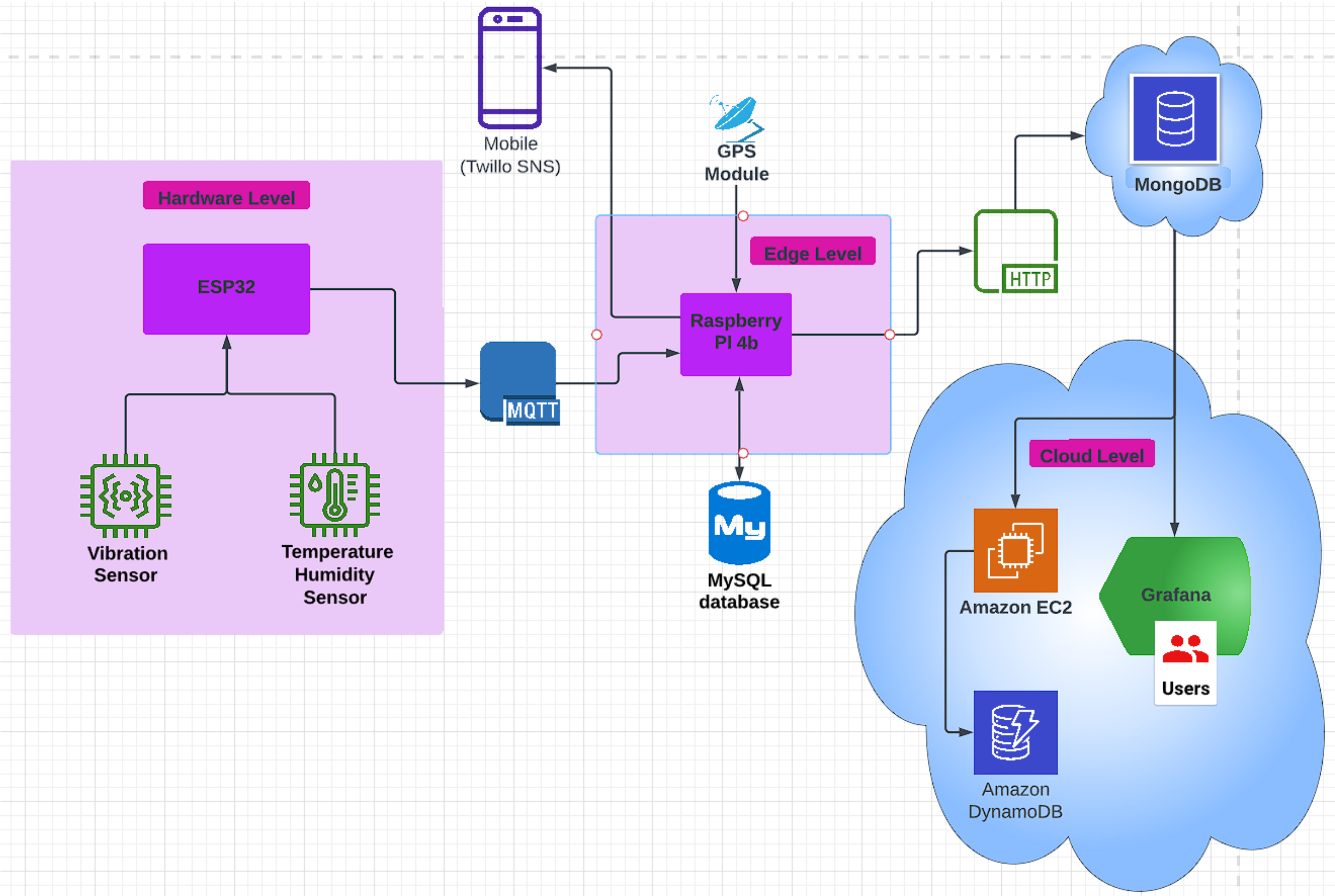


Fig. 1.2 Block diagram

In above fig. 1.2 It shows the block diagram of the project. ESP32 which is connected over Wi-Fi to mosquito broker, all the data received from the sensors is published on a topic then the user side system subscribed to topic and stored at MySQL database and then Raspberry Pi is pushing data in MongoDB through HTTPs connected which will further used for analysis as well it is live streamed to Grafana to display fleet parameters. With the help of AWS EC2, login user data will be stored in AWS DynamoDB .

CHAPTER 2

2.1 Data storage:

MongoDB :-

In this project we are using MongoDB atlas for data storage. MongoDB Atlas provides an easy way to host and manage your data in the cloud. This allows us to create an Atlas cluster, connecting to it, inserting data, and querying data. MongoDB is a true data platform with a comprehensive suite of tools to make working with data remarkably easy for everyone.

Following steps guide to use MongoDB atlas:

1. Create an Atlas Account: You can register for an Atlas account using your GoogleAccount or an email address.
2. Deploy a Free Tier Cluster: Atlas Free Tier clusters provide a small-scale development environment to host your data.
3. Add Your Connection IP Address to Your IP Access List: An IP is a unique numeric identifier for a device connecting to a network.
4. Create a Database User for Your Cluster: You must create a database user to access your cluster. For security purposes, Atlas requires clients to authenticate asMongoDB database users to access clusters.
5. Connect to Your Cluster: You can connect to your cluster in a variety of ways. We can connect to the cluster using the mongo shell, the Node.js driver, the PyMongo driver, and Compass.
6. Insert and View Data in Your Cluster.

Clusters are Atlas-managed MongoDB deployments. A cluster can be either a replica set or a sharded cluster. Use these resources to create, configure, and manage Atlas clusters.

The apparent advantages of this database system are:

Open-source:

The open-source database provides you complete peace of mind. It also offers you ample scope to learn about custom development.

Flexibility:

MongoDB got created to cater to online transaction processing. Hence, the database system's structure complements the same as well. Several successful organizations make use of MongoDB as its open-source and free. When you use MongoDB, you can seamlessly manage customer data management, mobile application management, and content management.

The data structure doesn't have a schema:

Usually, the relational database systems comprise of strict schema which can restrict the data structure flexibility. MongoDB gets best defined as a document store database, and the schema here isn't rigid.

It is developer-friendly:

MongoDB provides multiple resources for helping the developers to get started making use of the database.

2.2. Communication protocol:

2.2.1 MOSQUITTO MQTT :

The MQTT protocol was invented in 1999 by Andy Stanford-Clark (IBM) and Arlen Nipper. They needed a protocol for minimal battery loss and minimal bandwidth to connect with oil pipelines via satellite. The acronym for MQTT is MQ telemetry transport or simply MQTT protocol.

MQTT would be standardized under the wings of OASIS [The Organization for the Advancement of Structured Information Standards (OASIS)]. 2014 MQTT became an officially approved OASIS Standard. OASIS ratified the new MQTT 5 specification. This new MQTT version introduced new features to MQTT that are required for IoT applications deployed on cloud platforms, and those that require more reliability and error handling to implement mission-critical messaging.

MQTT is a Client-Server publish/subscribe messaging transport protocol. It is an application layer protocol built on TCP protocol. It is lightweight, open, simple, and designed so as to be easy to implement.

It is a very lightweight and binary protocol, and due to its minimal packet overhead, MQTT excels when transferring data over the wire in comparison to protocols like HTTP. Another important aspect of the protocol is that MQTT is extremely easy to implement on the client-side. Ease of use was a key concern in the development of MQTT and makes it a perfect fit for constrained devices with limited resources today.

2.2.1.1 Pub-sub-Pattern:

The publish/subscribe pattern (also known as pub/sub) provides an alternative to a traditional client-server architecture. In the client-server model, a client communicates directly with an endpoint. The pub/sub model decouples(separates) the client that sends a message (the publisher) from the client or clients that receive the messages (the subscribers).

The publishers and subscribers never contact each other directly. In fact, they are not even aware that the other exists. The connection between them is handled by a third component (the broker). The job of the broker is to filter all incoming messages and distribute them correctly tosubscribers.

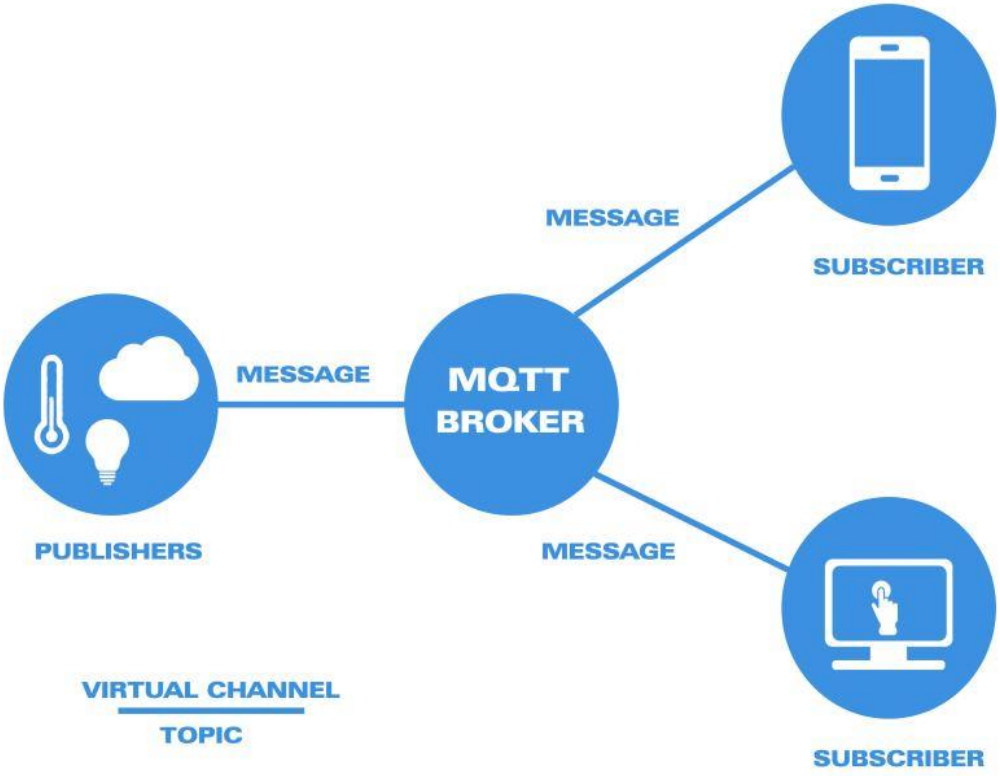


Fig. 2.1 MQTT pub-sub architecture

The most important aspect of pub/sub is the decoupling of the publisher of the message from therecipient (subscriber).

Pub/Sub scales better than the traditional client-server approach. This is because operations onthe broker can be highly parallelized and messages can be processed in an event-driven way.

2.2.1.2 Client, Broker / Server, and Connection Establishment :

Client:

- Both publishers and subscribers are MQTT clients.
- An MQTT client is any device (from a Microcontroller up to a full-fledged server) thatruns an MQTT library and connects to an MQTT broker over a network.
- Basically, any device that speaks MQTT over a TCP/IP stack can be called an MQTTclient.

Broker:

- The broker is at the heart of any publish/subscribe protocol.
- Depending on the implementation, a broker can handle up to thousands of concurrentlyconnected MQTT clients.
- The broker is responsible for receiving all messages, filtering the messages, determiningwho is subscribed to each message, and sending the message to these subscribed clients.
- The broker also holds the session data of all clients that have persistent sessions,including subscriptions and missed messages.
- Another responsibility of the broker is the authentication and authorization of clients.
- Usually, the broker is extensible, which facilitates custom authentication, authorization,and integration into backend systems.

MQTT Connection:

The MQTT protocol is based on TCP/IP. Both the client and the broker need to have a TCP/IP stack.The MQTT connection is always between one client and the broker. Clients never connect to each other directly.

To initiate a connection, the client sends a CONNECT message to the broker. The broker responds with a CONNACK message and a status code. Once the connection is established, the broker keeps it open until the client sends a disconnect command or the connection breaks.

Publish:

- An MQTT client can publish messages as soon as it connects to a broker. MQTT utilizesthe topic-based filtering of the messages on the broker.
- Each message must contain a topic that the broker can use to forward the message to

  interested clients.
- Typically, each message has a payload that contains the data to transmit in byte format.Subscribe:

Publishing a message doesn’t make sense if no one ever receives it. In other words, if there are no clients to subscribe to the topics of the messages. To receive messages on topics of interest, the client sends a SUBSCRIBE message to the MQTT broker.

2.2.2 HTTPS

2.2.2.1 What is HTTPs :

Hypertext transfer protocol secure (HTTPS) is the secure version of HTTP, which is the primary protocol used to send data between a web browser and a website. HTTPS is encrypted in order to increase security of data transfer. This is particularly important when users transmit sensitive data, such as by logging into a bank account, email service, or health insurance provider.

Any website, especially those that require login credentials, should use HTTPS. In modern web browsers such as Chrome, websites that do not use HTTPS are marked differently than those that are. Look for a padlock in the URL bar to signify the webpage is secure. Web browsers take HTTPS seriously; Google Chrome and other browsers flag all non-HTTPS websites as not secure.

2.2.2.2 Scope of using HTTPs :

HTTPS prevents websites from having their information broadcast in a way that’s easily viewed by anyone snooping on the network. When information is sent over regular HTTP, the information is broken into packets of data that can be easily “sniffed” using free software. This makes communication over the an unsecure medium, such as public Wi-Fi, highly vulnerable to interception. In fact, all communications that occur over HTTP occur in plain text, making them highly accessible to anyone with the correct tools, and vulnerable to on-path attacks.

With HTTPS, traffic is encrypted such that even if the packets are sniffed or otherwise intercepted, they will come across as nonsensical characters. Let’s look at an example:

Before encryption:

This is a string of text that is completely readable

After encryption:

ITM0IRyiEhVpa6VnKyExMiEgNveroyWBPlgGyfkflYjDaaFf/Kn3bo3OfghBPDWo6AfSHlNtL8N7ITEwIXc1gU5X73xMs

In websites without HTTPS, it is possible for Internet service providers (ISPs) or other intermediaries to inject content into webpages without the approval of the website owner.

This commonly takes the form of advertising, where an ISP looking to increase revenue injects paid advertising into the webpages of their customers.

Unsurprisingly, when this occurs, the profits for the advertisements and the quality control of those advertisements are in no way shared with the website owner.

HTTPS eliminates the ability of unmoderated third parties to inject advertising into web content.

Chapter 3

Implementation:

3.1 Connection diagram

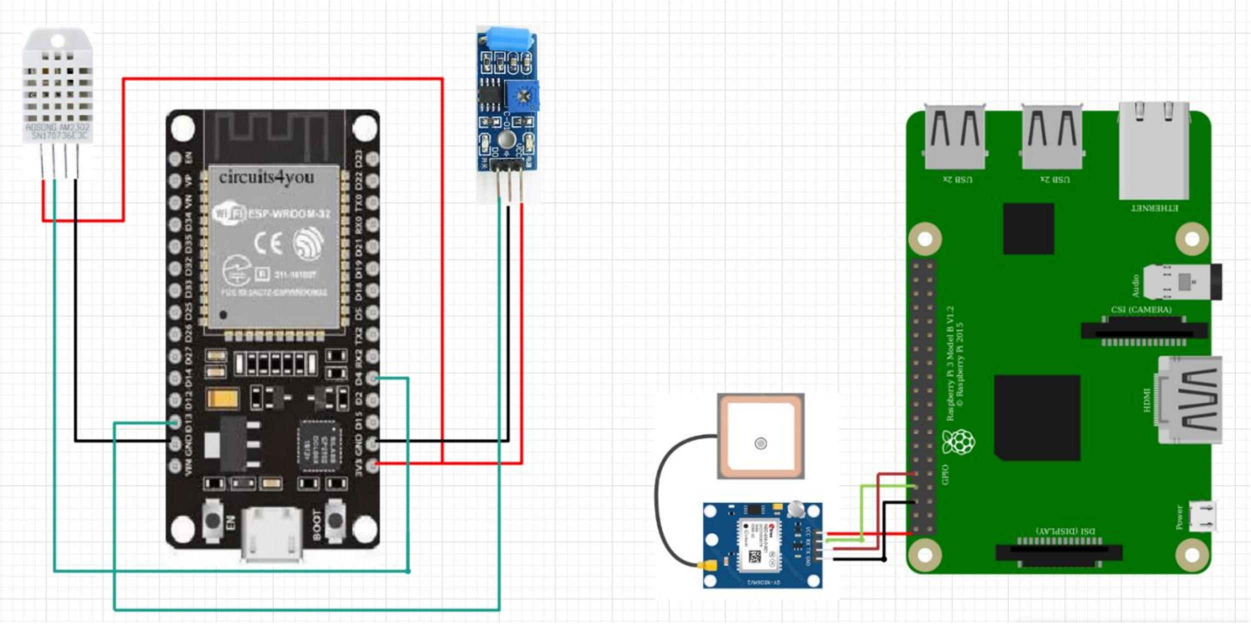


Fig. 3.1 Circuit Connection

The above fig. 3.1 shows the connection diagram of the proposed system. The ESP32 which will collect all sensor data and send it to the Edge database (Raspberry PI).GPS module is directly connected to the Raspberry Pi and its value is directly stored into Edge database. Main components of the system are:

i. ESP32
ii. Raspberry PI (4 model b)
iii. Temperature & Humidity sensor (DHT22)
iv. Vibration sensor (SW420)
v. GPS module (Neo-6M)

3.2 Sensors:

3.2.1. Digital Temperature and Humidity sensor (DHT22):

DHT22 output calibrated digital signal. It utilizes exclusive digital-signal-collecting-technique and humidity sensing technology, assuring its reliability and stability.Its sensing elements is connected with 8-bit single-chip computer. Every sensor of this model is temperature compensated and calibrated in accurate calibration chamber and the calibration-coefficient is saved in type of programme in OTP memory, when the sensor is detecting, it will cite coefficient from memory.

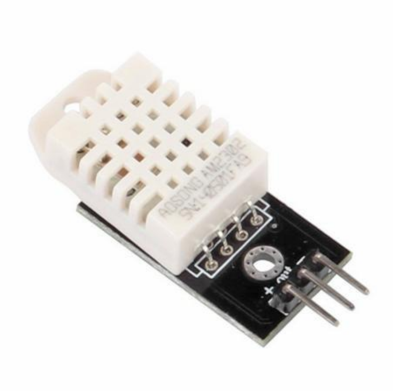


Fig. 3.2 DHT22Temperature and Humidity

Specifications:

Operating Voltage(V)	3.3- 5.5V
Temprature Range	0°c ~ 50°c
Humidity Range	20% - 90%
Sensor working temperature(Degree C)	0°c ~ 50°c
Sensing temperature range(%)	20% - 90%

3.2.2. Neo-6M GPS Module

It can track up to 22 satellites over 50 channels and achieve the industry’s highest level of tracking sensitivity i.e. -161 dB, while consuming only 45 mA current.

Unlike other GPS modules, it can perform 5 location updates in a second with 2.5m horizontal position accuracy. The U-blox 6 positioning engine also has a Time-To-First-Fix (TTFF) of less than 1 second.

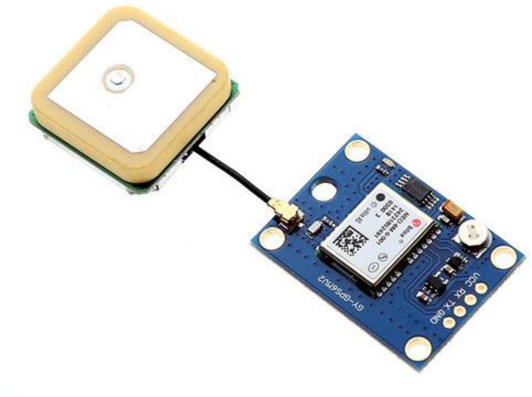


Fig. 3.3 Neo-6M GPS Module

One of the best features offered by the chip is Power Save Mode (PSM). This allows a reduction in system power consumption by selectively switching certain parts of the receiver on and off. This dramatically reduces the power consumption of the module to just 11mA making it suitable for power sensitive applications such as GPS wristwatches.

The required data pins of the NEO-6M GPS chip are broken out to a 0.1″ pitch headers. It contains the pins needed for communication with the micro controller over the UART. The module supports baud rates from 4800bps to 230400bps with a default baud of 9600.

Neo-6M GPS Module Working Principle:

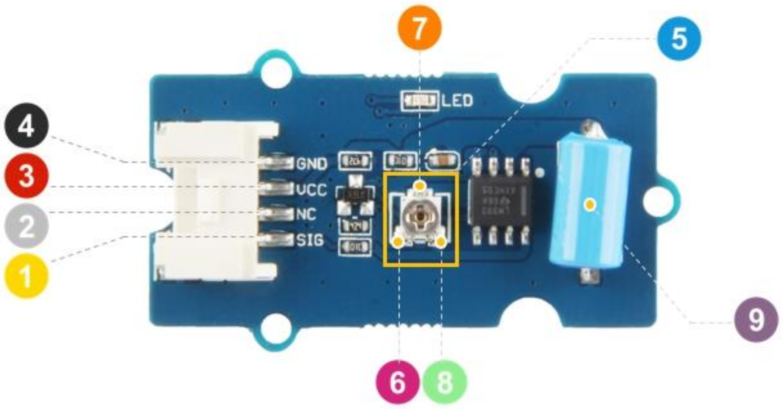
The NEO-6M GPS module is a well-performing complete GPS receiver with a built-in 25 x 25 x 4mm ceramic antenna, which provides a strong satellite search capability. With the power and signal indicators, you can monitor the status of the module. Thanks to the data backup battery, the module can save the data when the main power is shut down accidentally. Its 3mm mounting holes can ensure easy assembly on your aircraft, which thus can fly steadily at a fixed position, return to Home automatically, and automatic waypoint flying, etc. Or you can apply it on your smart robot car for automatic returning or heading to a certain destination, making it a real "smart" bot!

Specifications:

Reciver Type	50 channels, GPS L1(1575.42Mhz)
Horizontal Position Accuracy	2.5m
Navigation Update Rate	1HZ (5Hz maximum)
Capture Time	Cool start: 27sHot start: 1s
Navigation Sensitivity	-161dBm
Communication Protocol	NMEA, UBX Binary, RTCM
Serial Baud Rate	4800-230400 (default 9600)
Operating Temperature	-40°C ~ 85°C
Operating Voltage	2.7V ~ 3.6V
Operating Current	45mA
TXD/RXD Impedance	510Ω

3.2.3. Vibration sensor (SW420):

Vibration Sensor (SW-420) is a high sensitivity non-directional vibration sensor. When the module is stable, the circuit is turned on and the output is high. When the movement or vibration occurs, the circuit will be briefly disconnected and output low. At the same time, you can also adjust the sensitivity according to your own needs.



- 4 GND: connect this module to the system GND
- 3 VCC: you can use 5V or 3.3V for this module
- 2 NC: none connected in this module
- 1 SIG: output the Vout voltage signal

- 5 Potentiometer: you can use a screwdriver to rotate the potentiometer which controls the sensitivity of this sensor.
- 6 GND: the GND pin of the potentiometer.
- 7 Vsen: the lower the voltage of Vsen, the higher the sensitivity
- 8 VCC: the VCC pin of the potentiometer.
- 9 SW-420: the Vibration Sensor

Fig. 3.4 SW420

Specifications:

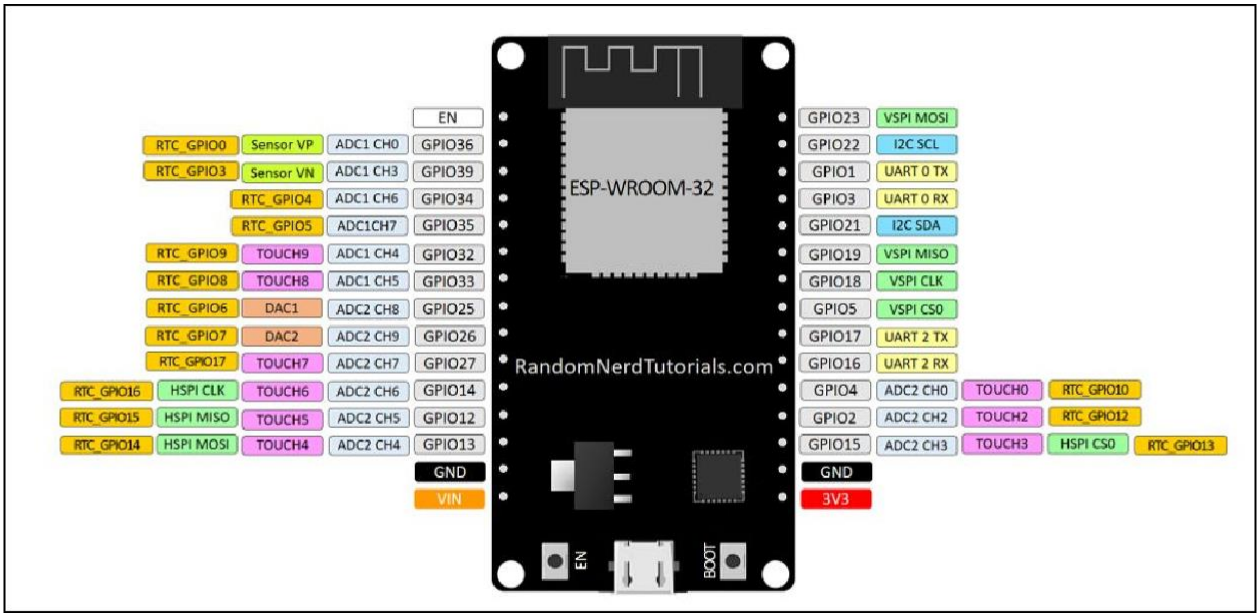
Operating voltage	3.3V / 5V
Interface	Digital
Size	L: 40mm W: 20mm H: 10mm
Weight	4.3g
Package size	L: 140mm W: 85mm H: 10mm
Gross Weight	10g

3.3 Hardware Components and Platforms Used:

3.3.1. ESP32

ESP32 is a low-cost, low-power system on a chip (SoC) series with Wi-Fi & dual-mode Bluetooth capabilities. The ESP32 is a development board that combines Wi-Fi and Bluetooth wireless capabilities, and it's dual core.

Fig. 3.7 ESP32 Model



- Processors:
  - Main processor: Tensilica Xtensa 32-bit LX6 microprocessor
  - Cores: 2 or 1 (depending on variation)
  - Clock frequency: up to 240 MHz
  - Performance: up to 600 DMIPS
  - Ultra low power co-processor: allows you to do ADC conversions, computation,and level thresholds while in deep sleep.

- Wireless connectivity:
  - Wi-Fi: 802.11 b/g/n/e/i (802.11n @ 2.4 GHz up to 150 Mbit/s)
  - Bluetooth: v4.2 BR/EDR and Bluetooth Low Energy (BLE)
- Memory:
  - Internal memory:
  - ROM: 448 KiB

    For booting and core functions.
  - SRAM: 520 KiB

    For data and instruction.
  - RTC fast SRAM: 8 KiB

    For data storage and main CPU during RTC Boot from the deep-sleep mode.
  - RTC slow SRAM: 8 KiB

    For co-processor access during deep-sleep mode.
  - eFuse: 1 Kibit

    Of which 256 bits are used for the system (MAC address and chip configuration) and the remaining 768 bits are reserved for customer applications, including Flash-Encryption and Chip-ID.
  - Embedded flash:

    Flash connected internally via IO16, IO17, SD_CMD, SD_CLK, SD_DATA_0and SD_DATA_1 on ESP32-D2WD and ESP32-PICO-D4.
    - 0 MiB (ESP32-D0WDQ6, ESP32-D0WD, and ESP32-S0WD chips)
    - 2 MiB (ESP32-D2WD chip)
    - 4 MiB (ESP32-PICO-D4 SiP module)

- Peripheral input/output: Rich peripheral interface with DMA that includes capacitive touch, ADCs (analog-to-digital converter), DACs (digital-to-analog converter), I²C (Inter-Integrated Circuit), UART (universal asynchronous receiver/transmitter), CAN 2.0 (Controller Area Network), SPI (Serial Peripheral Interface), I²S (Integrated Inter-IC Sound), RMII (Reduced Media-Independent Interface), PWM (pulse width modulation), and more.

3.3.2. Raspberry PI (4 model B):

Raspberry Pi 4 Model B is the latest product in the popular Raspberry Pi range of computers. It offers ground-breaking increases in processor speed, multimedia performance, memory, and connectivity compared to the prior-generation Raspberry Pi 3 Model B+, while retaining backwards compatibility and similar power consumption.

For the end user, Raspberry Pi 4 Model B provides desktop performance comparable to entry-level x86 PC systems. This product’s key features include a high-performance 64-bit quad-core processor, dual-display support at resolutions up to 4K via a pair of micro-HDMI ports, hardware video decode at up to 4Kp60, up to 4GB of RAM, dual-band 2.4/5.0 GHz wireless LAN, Bluetooth 5.0, Gigabit Ethernet, USB 3.0, and PoE capability (via a separate PoE HAT add-on).

The dual-band wireless LAN and Bluetooth have modular compliance certification,

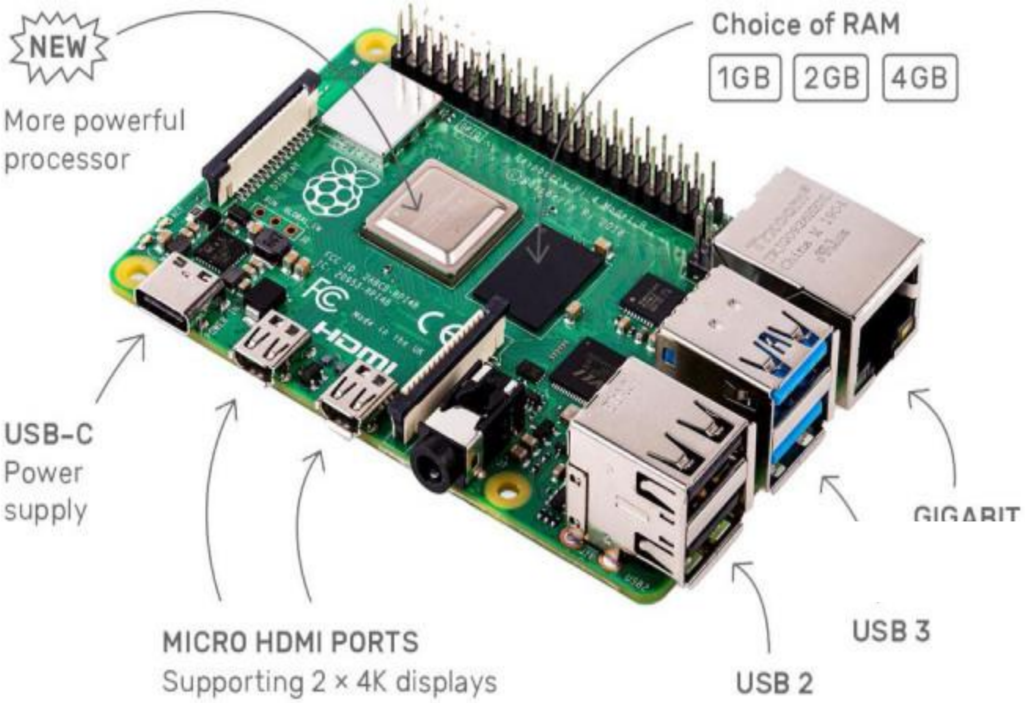


Fig. 3.8 Raspberry PI 4 model B

Specifications:

Processor	Broad Com (BCM2711), quad-core Cortex-A72 (ARMv8) 64-bit SoC @ 1.5GHz
Memory:	1GB, 2GB or 4GB LPDDR4 (depending on model)
Connectivity:	⬥ 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE. ⬥ Gigabit Ethernet over USB 2.0 (maximum throughput 300Mbps). ⬥ 2 × USB 3.0 ports. ⬥ 2 × USB 2.0 ports.
Access:	Standard40-pin GPIO header (Fully backwards-compatiable with previous boards)
Video & sound:	⬥ 2 × micro HDMI ports (up to 4k60fps supported) ⬥ 2-lane MIPI DSI display port ⬥ 2-lane MIPI CSI camera port ⬥ 4 pole stereo audio and composite video port

Multimedia:	H.264 (4k 60fps);
	H.264 encode (1080p 60fps encode);
	OpenGL ES 3.0 graphics

SD card support:	Micro SD card slot for loading operating system and data storage
Input power:	⬥ 5V DC via USB-C connector (minimum 3A) ⬥ 5V DC via GPIO header (minimum 3A) ⬥ Power over Ethernet (PoE)–enabled (requires separate PoE HAT)
Environment:	Operating temperature, 0–50°C
Production lifetime:	The Raspberry Pi 4 Model B will remain in production until at least January 2026.

3.3.3 Arduino Software(IDE):

The Arduino Integrated Development Environment - or Arduino Software (IDE) contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them.Arduino also simplifies the process of working with microcontrollers, but it offers some advantage for teachers, students, and interested amateurs over other systems:

- Inexpensive
- Cross-platform
- Simple, clear programming environment
- Open source and extensible software
- Open source and extensible hardware

3.3.4 Amazon Web Services(EC2 instance):

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, re-sizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers. Amazon EC2's simple web service interface allows you to obtain and configure capacity with minimal friction. It provides you with complete control of your computing resources and lets you run on Amazon's proven computing environment. Amazon EC2 offers the broadest and deepest compute platform with choice of processor, storage, networking, operating system, and purchase model.

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instance types comprise varying combinations of CPU, memory, storage, and networking capacity and give you the flexibility to choose the appropriate mix of resources for your applications. Each instance type includes one or more instance sizes, allowing you to scale your resources to the requirements of your target workload.

General purpose instances provide a balance of compute, memory and networking resources, and can be used for a variety of diverse workloads. These instances are ideal for applications that use these resources in equal proportions such as web servers and code repositories.

The purpose of using this EC2 instance is to manage and deploy NodeJS server so that user can login to his account and monitor his fleet with the help of dashboard.

3.3.5 Twillo (Notification Service):

Twilio MessagingX has everything you need to engage with customers, as one of the world’s most widely-used, trusted, and accessible communication tools. You can use MessagingX to send and receive SMS, MMS, and OTT messages at scale with purpose-built APIs like Programmable Messaging and Conversations, using phone numbers in more than 180 countries. MessagingX is used by companies like Dell to generate more online sales, and by nonprofits like The Trevor Project to deliver essential communications.

3.4 Data Visualization:

Grafana is open source visualization and analytics software. It allows you to query, visualize, alert on, and explore your metrics no matter where they are stored. In plain English, it provides you with tools to turn your time-series database (TSDB) data into beautiful graphs and visualizations.

Getting started with Grafana:

Step 1 : Install Grafana

Step 2 : Login

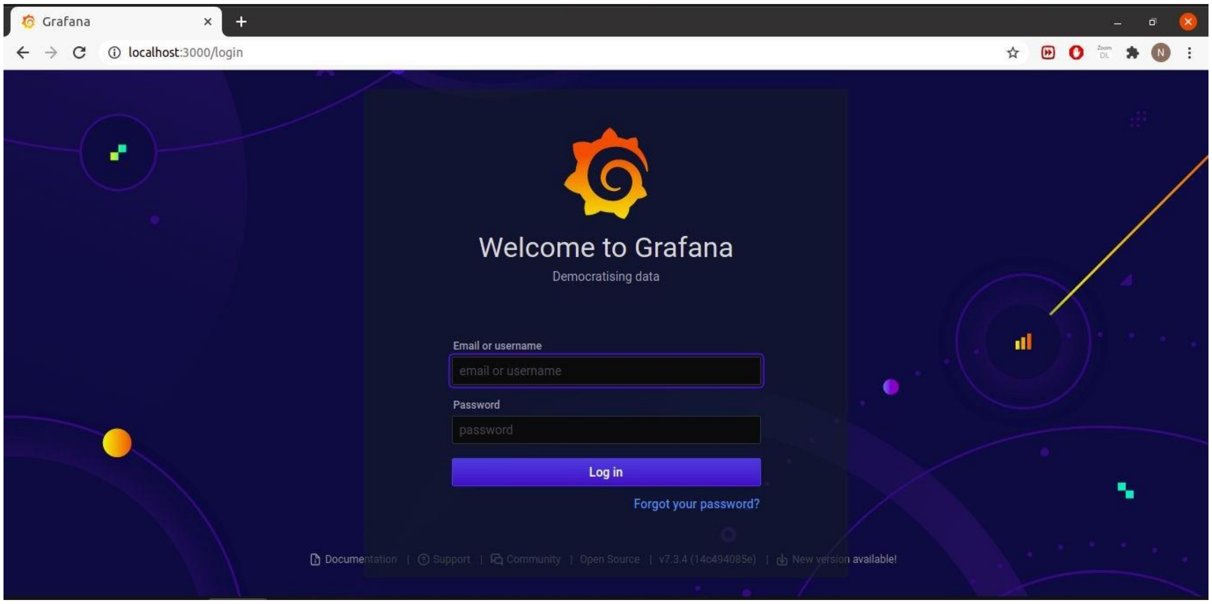


Fig. 3.9. Grafana Login page

To log in to Grafana for the first time:

1. Open your web browser and go to http://localhost:3000/. The default HTTP port thatGrafana listens to is 3000 unless you have configured a different port.
2. On the login page, enter admin for username and password.
3. Click Log In. If login is successful, then you will see a prompt to change the password.

4. Click OK on the prompt, then change your password.

Step 3 : Create a Dashboard

1. Click the + icon on the left panel, select Create Dashboard, and then click Add newpanel.
2. In the New Dashboard/Edit Panel view, go to the Query tab.
3. Configure your query by selecting -- Grafana -- from the data source selector. Thisgenerates the Random Walk dashboard.
4. Click the Save icon in the top right corner of your screen to save the dashboard.
5. Add a descriptive name, and then click Save.

3.5 Work Description:

Fig 3.10 shows a detailed diagram of the project. All sensors data is taken with the help of ESP32. Collected data is published with the help of MQTT protocol. Here the proposed system is using AWS EC2 as a broker. Host machine is used as a subscriber which will collect the sensor data and store it on the cloud of MongoDB atlas.

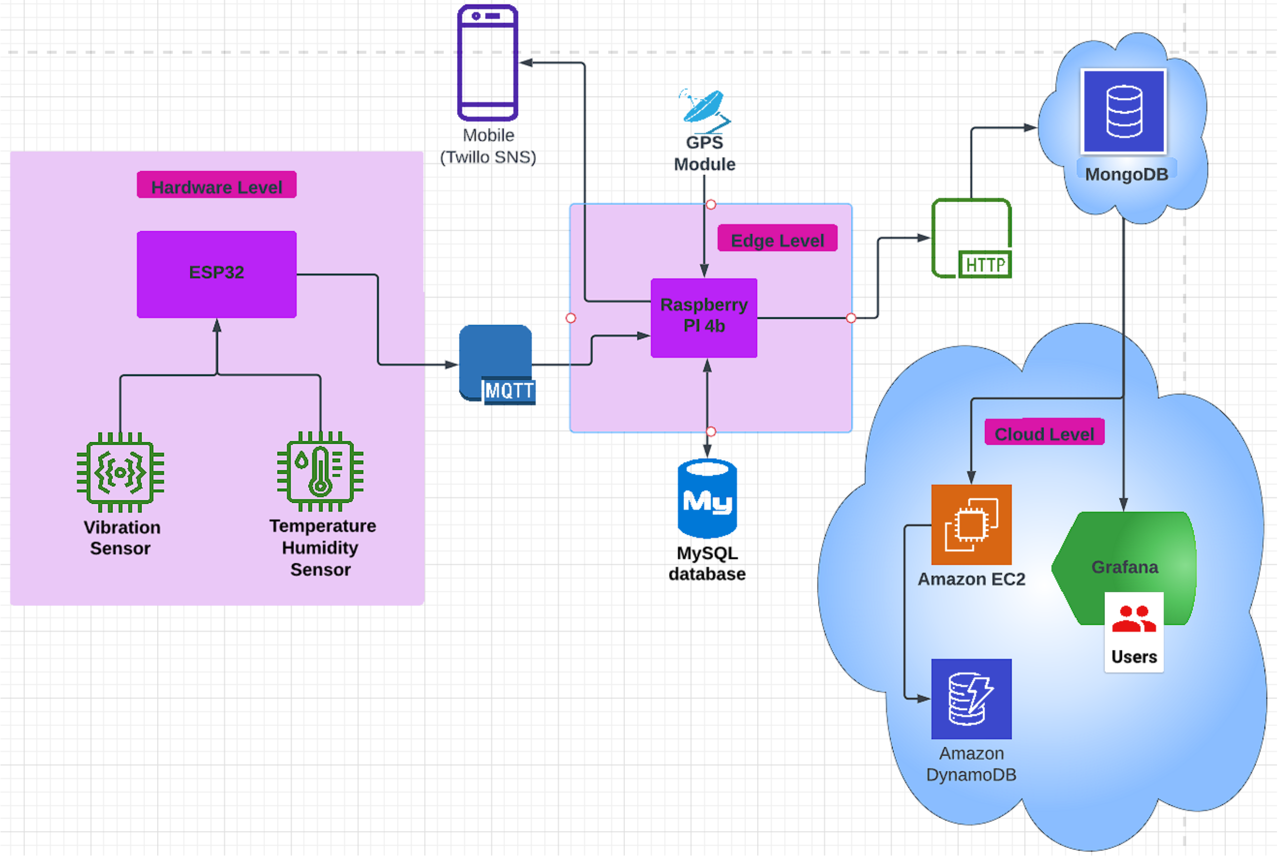


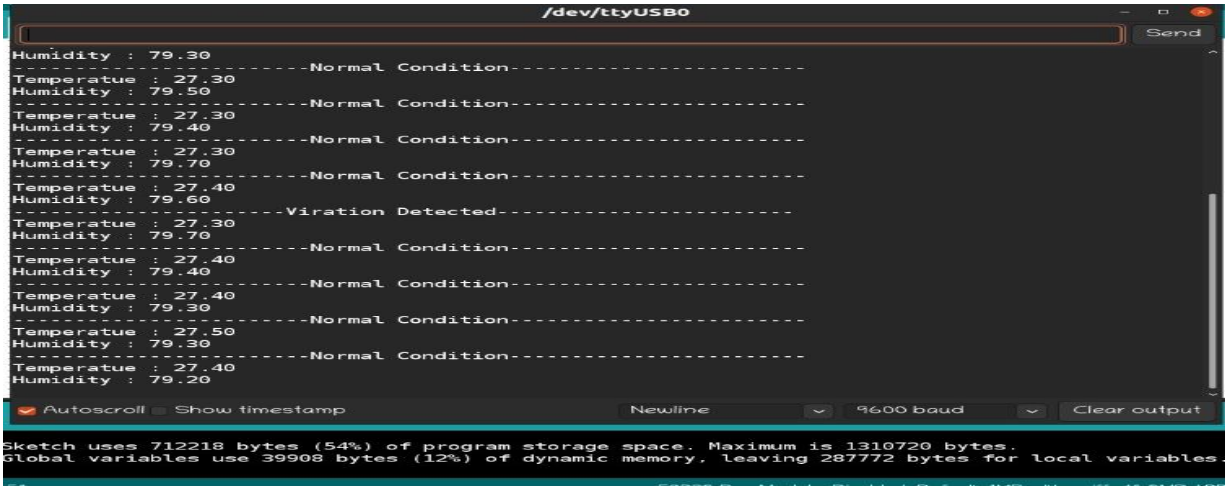
Fig.3.10 Work Flow diagram

All sensor data is shown on Grafana. On Grafana maximum, minimum, current result and threshold results are shown. Logistic regression is used for prediction of remaining life of the motor.

Chapter 4

Results :

Below image 4.1 shows a live ESP32 Publishing data.

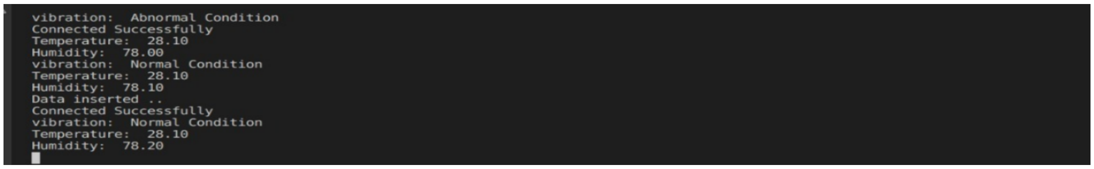


```
Humidity  :  79.30
------------------------Normal Condition------------------------
Temperatue  :  27.30
Humidity  :  79.50
------------------------Normal Condition------------------------
Temperatue  :  27.30
Humidity  :  79.40
------------------------Normal Condition------------------------
Temperatue  :  27.30
Humidity  :  79.70
------------------------Normal Condition------------------------
Temperatue  :  27.40
Humidity  :  79.60
----------------------Viration Detected----------------------
Temperatue  :  27.30
Humidity  :  79.70
------------------------Normal Condition------------------------
Temperatue  :  27.40
Humidity  :  79.40
------------------------Normal Condition------------------------
Temperatue  :  27.40
Humidity  :  79.30
------------------------Normal Condition------------------------
Temperatue  :  27.50
Humidity  :  79.30
------------------------Normal Condition------------------------
Temperatue  :  27.40
Humidity  :  79.20
```

```
Sketch uses 712218 bytes (54%) of program storage space. Maximum is 1310720 bytes.
Global variables use 39908 bytes (12%) of dynamic memory, leaving 287772 bytes for local variables.
```

Fig. 4.1

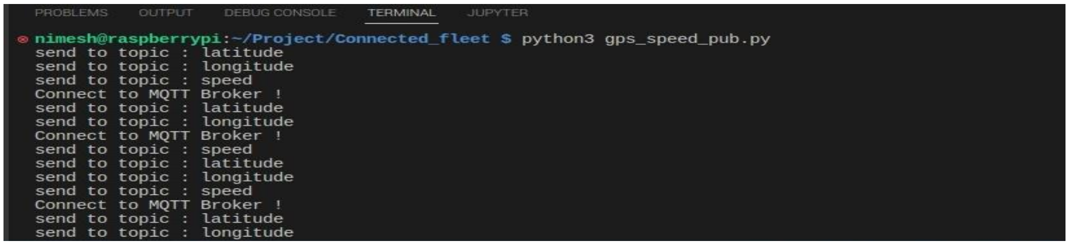
Subscriber Raspberry PI is storing data into MySQL data base in Fig.4.2 :



```
vibration:  Abnormal Condition
Connected Successfully
Temperature:  28.10
Humidity:  78.00
vibration:  Normal Condition
Temperature:  28.10
Humidity:  78.10
Data inserted ..
Connected Successfully
vibration:  Normal Condition
Temperature:  28.10
Humidity:  78.20
```

Fig. 4.2

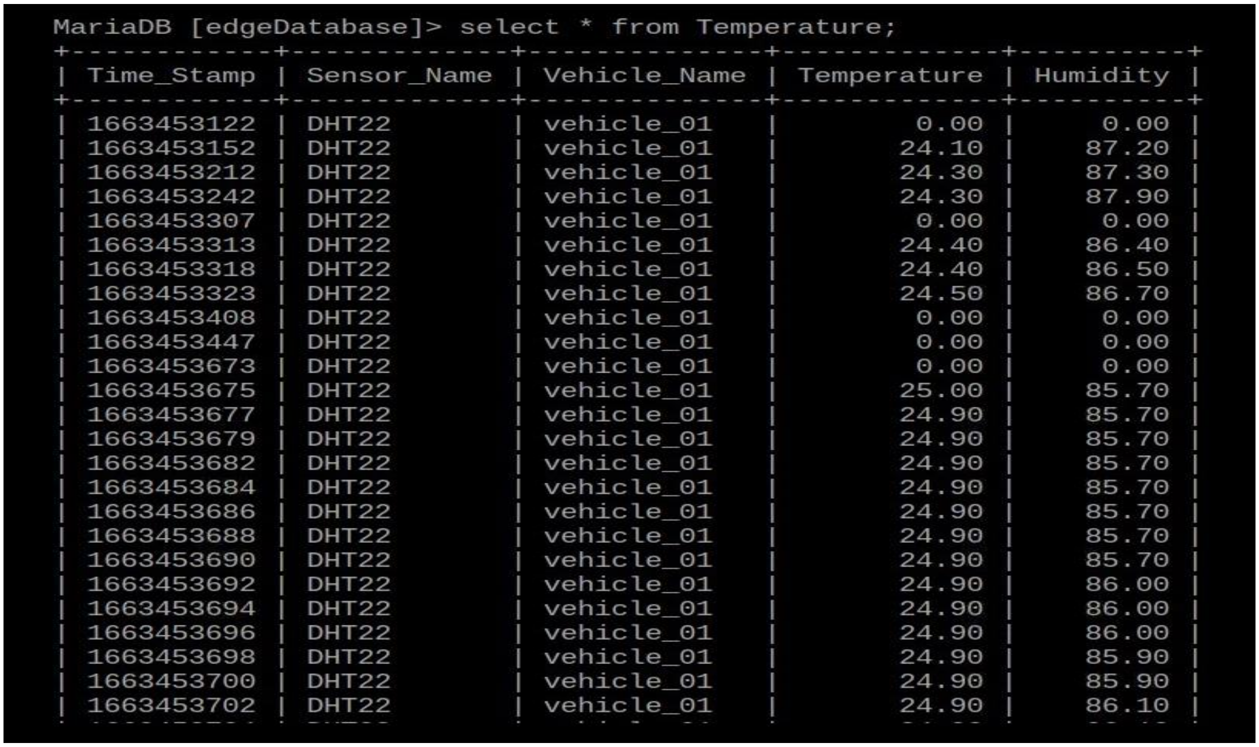
Raspberry PI is publishing Geo-location in Fig.4.3 :



```
nimesh@raspberrypi:~/Project/Connected_fleet $ python3 gps_speed_pub.py
send to topic : latitude
send to topic : longitude
send to topic : speed
Connect to MQTT Broker !
send to topic : latitude
send to topic : longitude
Connect to MQTT Broker !
send to topic : speed
send to topic : latitude
send to topic : longitude
send to topic : speed
Connect to MQTT Broker !
send to topic : latitude
send to topic : longitude
```

Fig. 4.3

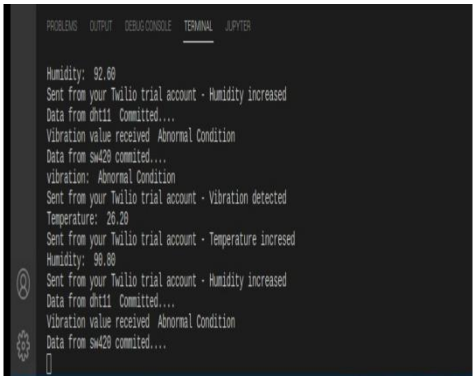
Sensor Data Showing up on Edge MySQL Database in Fig.4.4:



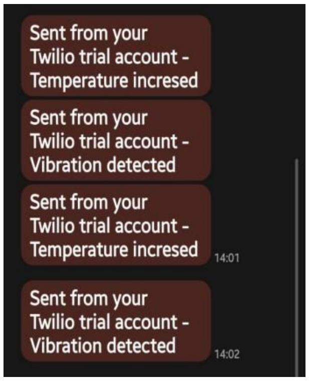
```
MariaDB [edgeDatabase]> select * from Temperature;
+------------+------------+-------------+-------------+----------+
| Time_Stamp | Sensor_Name | Vehicle_Name | Temperature | Humidity |
+------------+------------+-------------+-------------+----------+
| 1663453122 | DHT22      | vehicle_01  |        0.00 |     0.00 |
| 1663453152 | DHT22      | vehicle_01  |       24.10 |    87.20 |
| 1663453212 | DHT22      | vehicle_01  |       24.30 |    87.30 |
| 1663453242 | DHT22      | vehicle_01  |       24.30 |    87.90 |
| 1663453307 | DHT22      | vehicle_01  |        0.00 |     0.00 |
| 1663453313 | DHT22      | vehicle_01  |       24.40 |    86.40 |
| 1663453318 | DHT22      | vehicle_01  |       24.40 |    86.50 |
| 1663453323 | DHT22      | vehicle_01  |       24.50 |    86.70 |
| 1663453408 | DHT22      | vehicle_01  |        0.00 |     0.00 |
| 1663453447 | DHT22      | vehicle_01  |        0.00 |     0.00 |
| 1663453673 | DHT22      | vehicle_01  |        0.00 |     0.00 |
| 1663453675 | DHT22      | vehicle_01  |       25.00 |    85.70 |
| 1663453677 | DHT22      | vehicle_01  |       24.90 |    85.70 |
| 1663453679 | DHT22      | vehicle_01  |       24.90 |    85.70 |
| 1663453682 | DHT22      | vehicle_01  |       24.90 |    85.70 |
| 1663453684 | DHT22      | vehicle_01  |       24.90 |    85.70 |
| 1663453686 | DHT22      | vehicle_01  |       24.90 |    85.70 |
| 1663453688 | DHT22      | vehicle_01  |       24.90 |    85.70 |
| 1663453690 | DHT22      | vehicle_01  |       24.90 |    85.70 |
| 1663453692 | DHT22      | vehicle_01  |       24.90 |    86.00 |
| 1663453694 | DHT22      | vehicle_01  |       24.90 |    86.00 |
| 1663453696 | DHT22      | vehicle_01  |       24.90 |    86.00 |
| 1663453698 | DHT22      | vehicle_01  |       24.90 |    85.90 |
| 1663453700 | DHT22      | vehicle_01  |       24.90 |    85.90 |
| 1663453702 | DHT22      | vehicle_01  |       24.90 |    86.10 |
```

Fig.4.4

Critical Data is showing up on Twillo (SNS) shown in Fig.4.5, 4.6, 4.7 :



```
Humidity:  92.60
Sent from your Twilio trial account - Humidity increased
Data from dht11  Committed....
Vibration value received  Abnormal Condition
Data from sw420 commited....
vibration:  Abnormal Condition
Sent from your Twilio trial account - Vibration detected
Temperature:  26.20
Sent from your Twilio trial account - Temperature incresed
Humidity:  90.80
Sent from your Twilio trial account - Humidity increased
Data from dht11  Committed....
Vibration value received  Abnormal Condition
Data from sw420 commited....
```

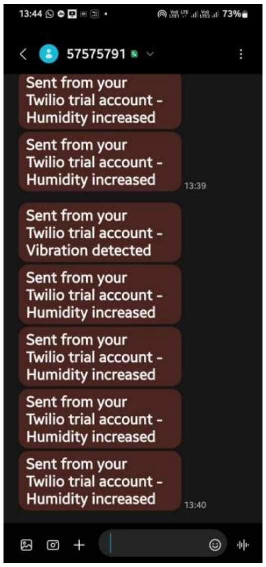


Sent from your Twilio trial account - Temperature incresed

Sent from your Twilio trial account - Vibration detected

Sent from your Twilio trial account - Temperature incresed 14:01

Sent from your Twilio trial account - Vibration detected 14:02



Sent from your Twilio trial account - Humidity increased

Sent from your Twilio trial account - Humidity increased 13:39

Sent from your Twilio trial account - Vibration detected

Sent from your Twilio trial account - Humidity increased

Sent from your Twilio trial account - Humidity increased

Sent from your Twilio trial account - Humidity increased

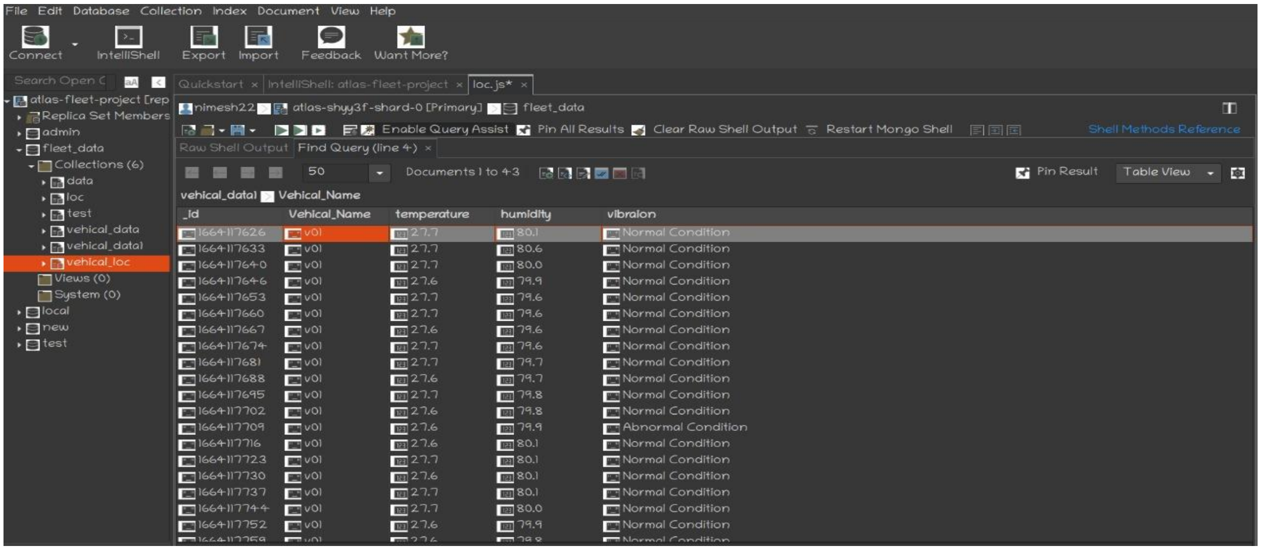
Sent from your Twilio trial account - Humidity increased 13:40

Fig.4.5

Fig.4.6

Fig.4.7

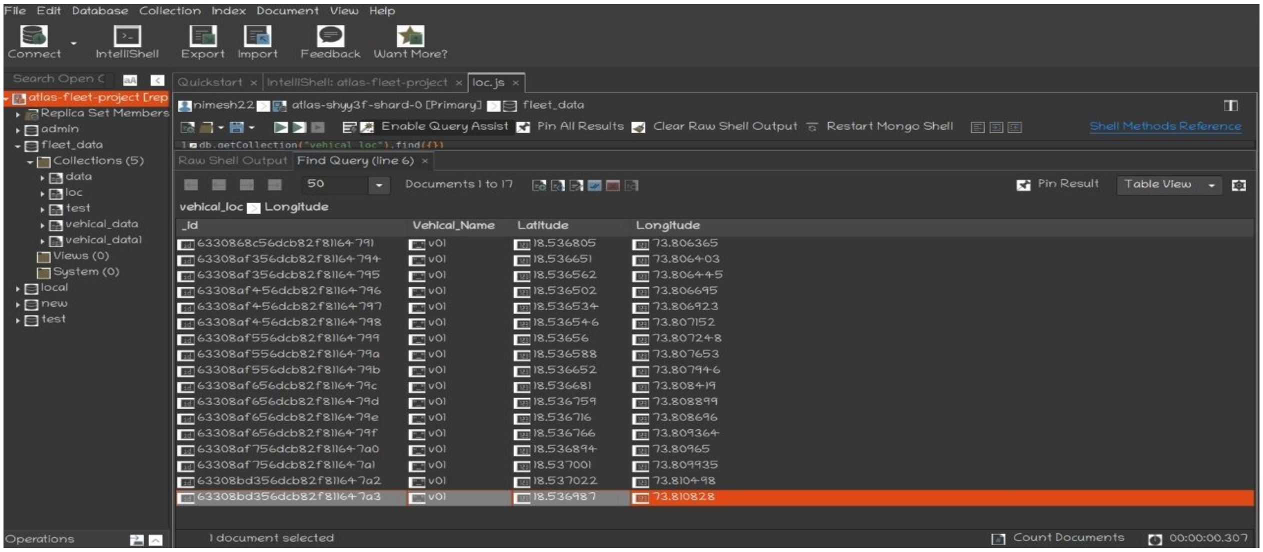
Through MongoDB the Sensor Data is showing up on Cloud in Fig. 4.8 :



_id	Vehical_Name	temperature	humidity	vibraion
1664117626	v01	27.7	80.1	Normal Condition
1664117633	v01	27.7	80.6	Normal Condition
1664117640	v01	27.7	80.0	Normal Condition
1664117646	v01	27.6	79.9	Normal Condition
1664117653	v01	27.7	79.6	Normal Condition
1664117660	v01	27.7	79.6	Normal Condition
1664117667	v01	27.6	79.6	Normal Condition
1664117674	v01	27.7	79.6	Normal Condition
1664117681	v01	27.7	79.7	Normal Condition
1664117688	v01	27.6	79.7	Normal Condition
1664117695	v01	27.7	79.8	Normal Condition
1664117702	v01	27.6	79.8	Normal Condition
1664117709	v01	27.6	79.9	Abnormal Condition
1664117716	v01	27.6	80.1	Normal Condition
1664117723	v01	27.7	80.1	Normal Condition
1664117730	v01	27.6	80.1	Normal Condition
1664117737	v01	27.7	80.1	Normal Condition
1664117744	v01	27.7	80.0	Normal Condition
1664117752	v01	27.6	79.9	Normal Condition
1664117759	v01	27.6	79.8	Normal Condition

Fig.4.8

Through MongoDB the location data is showing up on Cloud in Fig. 4.9 :



```
db.getCollection("vehical_loc").find({})
```

_id	Vehical_Name	Latitude	Longitude
6330868c56dcb82f81164791	v01	18.536805	73.806365
63308af356dcb82f81164794	v01	18.536651	73.806403
63308af356dcb82f81164795	v01	18.536562	73.806445
63308af456dcb82f81164796	v01	18.536502	73.806695
63308af456dcb82f81164797	v01	18.536534	73.806923
63308af456dcb82f81164798	v01	18.536546	73.807152
63308af556dcb82f81164799	v01	18.53656	73.807248
63308af556dcb82f8116479a	v01	18.536588	73.807653
63308af556dcb82f8116479b	v01	18.536652	73.807946
63308af656dcb82f8116479c	v01	18.536681	73.808419
63308af656dcb82f8116479d	v01	18.536759	73.808899
63308af656dcb82f8116479e	v01	18.536716	73.808696
63308af656dcb82f8116479f	v01	18.536766	73.809364
63308af756dcb82f811647a0	v01	18.536894	73.80965
63308af756dcb82f811647a1	v01	18.537001	73.809935
63308bd356dcb82f811647a2	v01	18.537022	73.810498
63308bd356dcb82f811647a3	v01	18.536987	73.810828

Fig.4.9

With the help of EC2 instances Grafana is showing Dashboard using NodeJS app in Fig.5.0 :

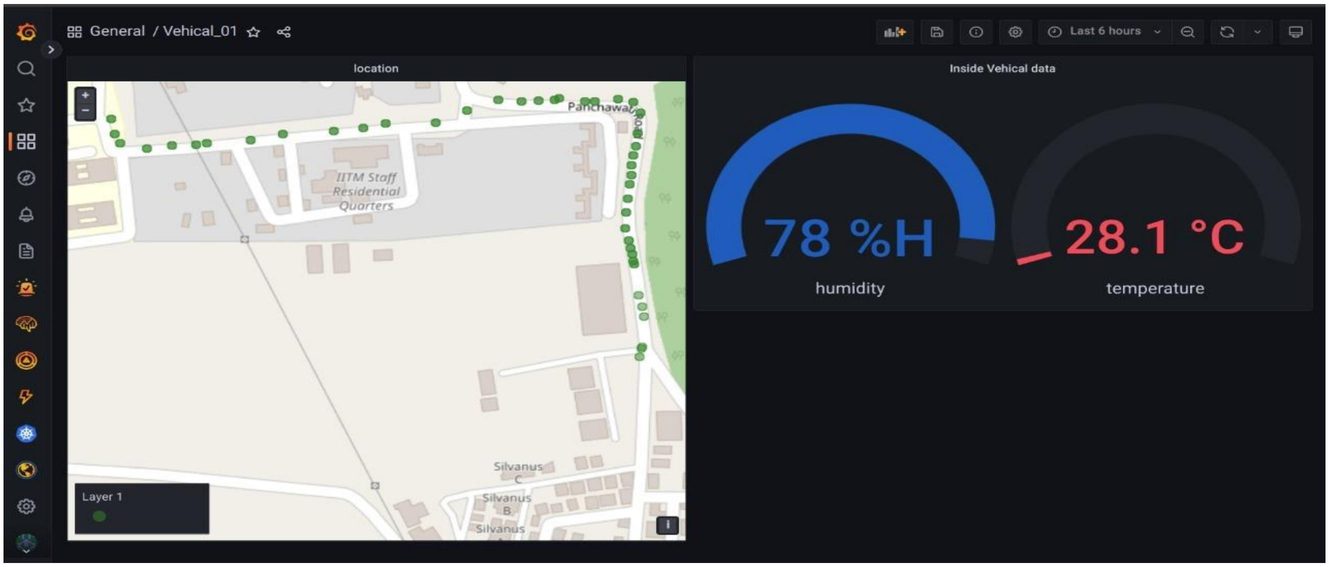


Fig.5.0

Fig.5.1 Shows Final Prototype of proposed system for Fleet Management :

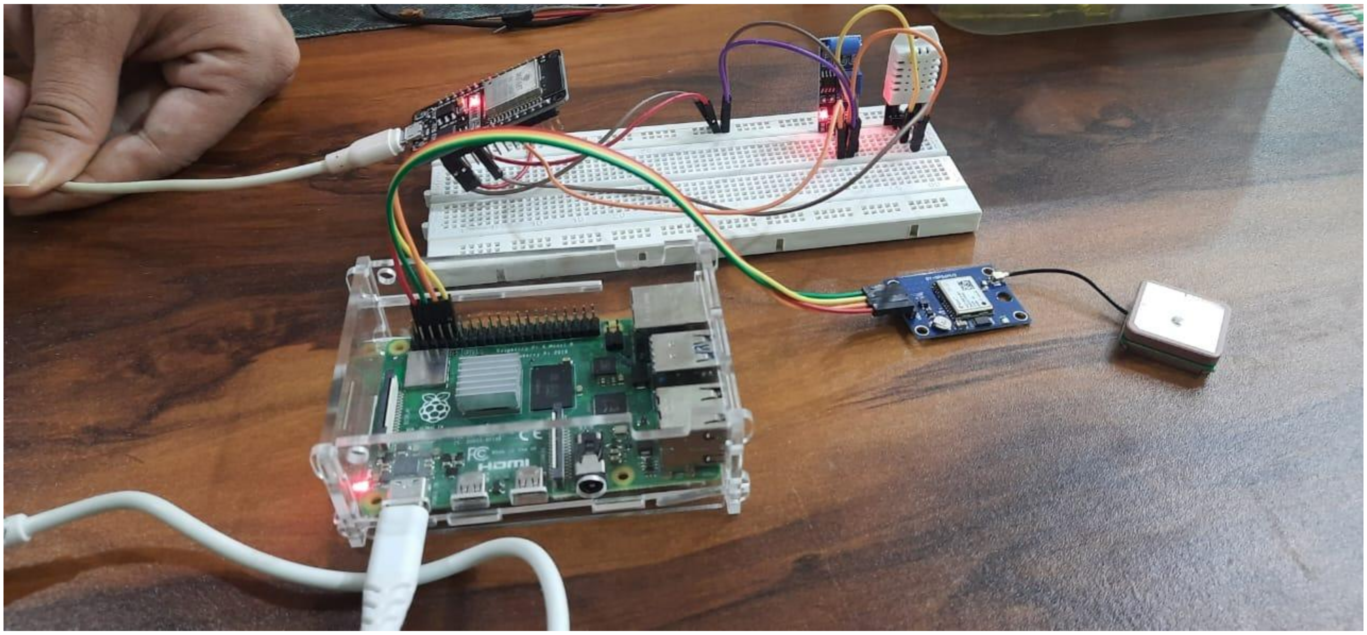


Fig.5.1

Conclusion:

The goal of this project is to develop error free, secure and reliable fleet management system. which can assist the user to concentrate on their other activities rather to concentrate on the record keeping. Thus, it will help organization in better utilization of resources.

Fleet management is essential to a smooth operating fleet of vehicles, no matter the size. Tracking the location and condition of the vehicles, maintenance schedules, and fuel usage helps manage costs and keeps the equipment working longer.

This project demonstrates the idea of monitoring and analyzing fleet's parameters like- goods temperature, humidity, location and vibration by rendering support for remote field deployments and monitoring.

This use case is offering high performance, useful data parameter, real time data collection, and data visualization .

Internet of Things (IoT) aims to extend the Internet to a large number of distributed devices by defining standard, interoperable communication protocols. The major objective of IoT is to create a smart environment using enabling technologies such as sensors, embedded devices, communication protocols etc and this project successfully implements that.

Challenges:

- Security

As fleets become autonomous and digitally powered, maintenance teams will be part of the critical line of defense to ensure transportation and the use of vehicles in plants are safe. Not only will technicians need to inspect and ensure that back-up systems and safeguards are in place, but they also must monitor the technologies involved in fleet operation, from dispatch to sensors monitoring the levels of brake fluid.

- The driver shortage

The driver shortage has been among the top concerns for fleet managers in the past decade. As the demands for freight transport increase and the number of new driver applicants continues to stagnate, experts predict the shortage of delivery drivers may approach 250,000 by 2022. Recruiting inexperienced drivers can also lead to safety issues and the need for driver monitoring tactics.

- Digitization of Vehicles

The adoption of digital applications can enable managers to track vehicle routes, access data on service history down to the component and part level, maintain compliance records, and receive notifications when preventive maintenance is due. Increased sophistication of telemetry and internet of things (IoT) technology are making it possible for managers to track all of these processes and more.

- Route Optimization

Software built especially for dispatching service technicians can track fleet vehicle location, identify the optimal vehicle to dispatch for an urgent call and calculate the best route. For the fleet manager and the maintenance team, this GPS-centric data can be valuable in monitoring vehicles and pinpointing environmental factors that lead to unusual maintenance requirements.

Future Scope:

Fleet management is evolving as the world of technology grows. Not only is it embracing the idea of sustainability as a key tracking metric, but it is becoming more attuned to customer needs for ease of use and technological advancement. As we move forward, we will begin to see more advancement in:

- Blockchain technology

With blockchain, data is decentralized and disseminated, bringing anonymity to help strengthen security while allowing businesses to easily access the information they require.

- Mobility as a service

This consumer-centric form of mobility can also be applied to the transportation industry. It is an on-demand, real-time platform that can incorporate any combination of modes of transportation and offers the consumer everything from journey planning to payment.

- Keyless vehicle access

Digital keys, like Geotab keyless, make it easy for customers and employees to access shared vehicles at the touch of a button or through the use of an app. This opens up opportunities for fleets to get the most out of their vehicles.

- Sustainable vehicle solutions

With the world beginning to lean more towards sustainable solutions, new makes and models of EVs are being released for even more purposes. From electric long haul trucks to additional consumer models, it will soon become easier than ever to access the right EVs for your fleet needs.

- Autonomous vehicles

Though self-driving cars may still feel like a far-off idea, there are opportunities in trucking to take advantage of these assets due to the predictability of traffic patterns on highways.

Bibliography

MQTT Official Documentation

Link: http://mqtt.org/

MQTT

Link: https://www.hivemq.com/blog/how-to-get-started-with-mqtt

Arduino IDE documentation

Link: https://www.arduino.cc/en/Guide

Python 3 documentation:

Link: https://docs.python.org/3/

Grafana server documentation :

Link: https://grafana.com/docs/

EC2 instance

Link:

https://aws.amazon.com/ec2/

instance-types/

MongoDB:

Link: https://docs.atlas.mongodb.com/

Wi-Fi manager:

Link: https://github.com/khoih-prog/ESP_WiFiManager

Grafana mongoDB setup:

Link: https://github.com/JamesOsgood/mongodb-grafana