

FALL DETECTION USING IFTTT

A MINI PROJECT REPORT

Submitted by

M.S. LAKSHMI SRI (910620103036)

J. LATIKAA TEJUS (910620103037)

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

**ELECTRONICS AND COMMUNICATION
ENGINEERING**

**K.L.N. COLLEGE OF ENGINEERING,
POTTAPALAYAM**

An Autonomous Institution

May 2023

BONAFIDE CERTIFICATE

Certified that this report titled “**Fall Detection using IFTTT**”, is the Bonafide work of

LATIKAA TEJUS.J (910620103037)

LAKSHMI SRI M.S (910620103036)

who carried out the project work under my supervision.

SIGNATURE

Dr.V. KEJALAKSHMI, M.E., Ph.D.,
Professor
Head of the Department
Department of Electronics and
Communication Engineering
K.L.N.College of Engineering
Pottapalayam ,
Sivagangai- 630311

SIGNATURE

MRS.S. INDUMATHI., M. E
Assistant professor
Supervisor
Department of Electronics and
Communications Engineering
K.L.N. College of Engineering
Pottapalayam,
Sivagangai-630311

Submitted for the project viva-voce held on _____

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We extend our gratitude to the Founder, Late **Thiru. K.L.N. KRISHNAN**, K.L.N. College of Engineering and Management Members for making us march towards the glory of success. We express our sincere thanks to our respected **Principal Dr. A.V.RAM PRASAD, M.E, Ph.D., MISTE.FIE**, for all the facilities offered.

We would like to express our profound gratitude and heartfelt thanks to **Dr.V. KEJA LAKSHMI, M.E., Ph.D.**, Head of the Department of Electronics and Communication Engineering, who motivated and encouraged us to do this out-ri-val project for this academic year.

Our delightful and sincere thanks to our respected Project Coordinator **MRS.S. INDUMATHI M.E** Assistant Professor, whose support was inevitable during the entire period of our work.

We thank our teaching staffs for sharing their knowledge and view to enhance our project. We also thank our non-teaching staff for extending their technical support to us. I thank my parents for giving me such a wonderful life and my friends for their friendly encouragement throughout the project.

Finally, we thank the Almighty for giving the full health to finish the project successfully

FALL DETECTION USING IFTTT (Abstract)

- According to the Report of the Technical Group on Population Projections for India there are nearly **38 million** elderly persons in India in 2021 (67 million males and 71 million females) and is further expected to increase by around 56 million elderly persons in 2031.
- This is why we propose this prototype; a health band that assist old people in their daily lives, leaving the family stress free. Especially for people with Alzheimer's.
- This prototype basically detects the pulse of the wearer using the pulse sensor we have attached to the NodeMCU and detects the fall of a person by the use of the MPU6050 an accelerometer, which is a key component of this prototype.
- Instead of using a GPS module for the communication between two devices we have introduced a concept where we use the software IFTTT for efficient communication. This IFTTT is abbreviated as “**IF THIS THEN THAT**” this concept is very easy for simple automation works, and found to be easily accessible and simple to work with.
- The final goal is to alert the caregiver when the wearer falls and to generate a dataset of the wearer's heartbeat in ThingSpeak cloud. For this we use the IFTTT

TABLE OF CONTENTS

CHAPTER	CONTENT	PG.NO
1.	INTRODUCTION What is Fall detection	1
2	Hardware Requirements	2
	2.1 NodeMCU-esp8266	2
	2.2 MPU6050	5
	2.3 Pulse Sensor	8
	2.4 Arduino IDE	10
	2.4.1. step-by-step procedure for adding the ESP8266 board to the Arduino IDE	11
	2.5 Why NodeMCU is the best option?	15
3.	3.1.1 What is IFTTT?	17
	3.2 Keywords	18
	3.2.1 Action:	
	3.2.2 Applets	19
	3.2.3 Ingredients	21

	3.2.4 Query and query field	22
	3.2.5 Services:	23
	3.2.6 Trigger:	24
	3.2.7 Applet ID	25
	3.2.8 Step by step Procedure for creating anApplet	26
	3.2.9 Why is IFTTT better than usingTransmitter and Receiver Module?	28
4	The block Diagram (fall detection)	30
	4.1 Explanation of this Block Diagram	31
	4.2 Simple representation of how a Pulse sensor work	
5	Literature survey	33
	5.1. Online Fall Detection Using WristDevices	33
	5.3 Machine learning algorithms - used forfall detection	35
6	ThingSpeak	36
	6.1 What is ThingSpeak	
	6.2Key Features of ThingSpeak Cloud	37

	6.3Architecture of ThingSpeak Cloud	39
	6.4 Step by step procedure for creating a channel in ThingSpeak	42
	6.5. Step by step procedure to download a dataset from ThingSpeak	45
7	APPLICATIONS (of Fall detection in real lives)	46
8	EXISTING METHODS	48
	8.1 PROPOSED METHOD (For this fall detection)	50
	8.2. OUTPUTS FROM IFTTT AND THINGSPEAK	51
9	CONCLUSION	52

LIST OF FIGURES

- Fig 2.1.1 NODEMCU (esp8266)
- Fig 2.2.1 MPU6050
- Fig 2.3.1 Circuit Diagram of a Pulse Sensor
- Fig 2.3.2 Working of Pulse sensor using IR rays
- Fig 2.4.1 The Arduino ide console
- Fig 2.4.2 The URL for connecting Arduino IDE with NodeMCU
- Fig 2.4.3 The boards in the esp8266 is shown here
- Fig 2.4.4 The Actual picture of our Project with MPU6050
- Fig 2.4.5 The connection of Pulse sensor with NodeMCU
- Fig 3.2.2.1 Applets
- Fig 3.2.3.1. Ingredients
- Fig 3.2.1 Services provided by IFTTT
- Fig.3.2.2 Trigger in IFTT (IF block)
- Fig 3.2.3 Applet ID in IFTTT
- Fig 3.2.4 Applets created on IFTTT
- Fig 3.2.5 Transmitter Receiver Module VS IFTTT
- Fig 4.1.1 The Alert message received on the mobile device
- Fig 5.3.1 The future scope of fall detection due to machine learning
- Fig 6.1.1 Simple representation of the working of ThingSpeak
- Fig 6.2.1 Key features of ThingSpeak
- Fig 6.3.1 Simple Architecture of a Sensor data Acquisition
- Fig 6.4.1 ThingSpeak webpage
- Fig 6.5.1 Downloaded Excel sheet
- Fig 8.1 Types of sensors used in fall detection

- Fig 8.1.1 Visual representation of our prototype
- Fig 8.2.1 Alert on Android device
- Fig 8.2.2. Output on ThingSpeak cloud

LIST OF TABLES

- Table 2.1.1 Pin details
- Table 2.1.2 Peripherals and I/O
- Table 2.2.1 Pin description of MPU6050T
- Table 2.6 Commands in Arduino IDE

ABBREVIATIONS

- NODEMCU (Node micro-controller unit)
- ESP8266 (Espressif 8266)
- MPU 6050 (Motion Processing unit 6050)
- MEMS (Micro Electro-Mechanical system)
- Arduino IDE (Arduino Integrated Development Environment)
- IFTTT (If This Then That)
- GPIO (General-Purpose Input/Output)
- I2C (Serial 2-wire bus for Communicating)
- API (Application Programming Interface)
- SCL (Serial Clock)
- SDA (Serial Data)
- XDA (Auxiliary Serial Data)
- XCL (Auxiliary Serial Clock)
- INT (Interrupt)
- GSM (Global System for Mobile Communication)

CHAPTER 1-INTRODUCTION

WHAT IS FALL DETECTION?

Fall detection is a technology that is designed to detect when an individual fall and is unable to get up on their own. The technology typically utilizes sensors and algorithms to detect sudden changes in motion or orientation that are indicative of a fall. Once a fall is detected, the technology can trigger an alarm or alert to notify caregivers or emergency services to provide assistance.

Fall detection technology can be implemented in a variety of forms, such as wearable devices, smart home systems, and healthcare monitoring systems. In wearable devices, such as smartwatches or pendants, the fall detection sensors are typically accelerometer and gyroscope sensors that can detect sudden changes in movement and orientation. In smart home systems, the technology can be integrated into sensors that are placed throughout the home to monitor the individual's movements and detect falls. Healthcare monitoring systems can use fall detection sensors to monitor patients in hospitals or other healthcare facilities.

Fall detection is particularly important for seniors or individuals with mobility or health issues who are at an increased risk of falls. Falls can cause serious injuries, such as fractures or head trauma, and can lead to long-term disability or even death. By detecting falls quickly, fall detection technology can help reduce the risk of injury and provide prompt medical attention.

CHAPTER 2- HARDWARE REQUIREMENTS

2.1 NodeMCU

ESP8266 is a low-cost Wi-Fi microchip that has become very popular in recent years, especially in the field of Internet of Things (IoT). It was developed by Espressif Systems, a Chinese company, and has quickly gained a reputation for being versatile and easy to use.

One of the main advantages of the ESP8266 is its low cost. This makes it accessible to hobbyists and small businesses who may not have the budget for more expensive microcontrollers. Despite its low price, the ESP8266 is still very powerful and has a full TCP/IP stack and microcontroller capabilities.

The ESP8266 can be programmed using a variety of programming languages, including the Arduino IDE, Micro-Python, Lua or C/C++. This makes it very flexible and allows developers to choose the language that they are most comfortable with. It can also be used as a standalone microcontroller or as a Wi-Fi module in conjunction with other microcontrollers.

One of the key features of the ESP8266 is its Wi-Fi connectivity. This makes it easy to connect to the internet and control IoT devices remotely. For example, the ESP8266 could be used in a smart home automation system, allowing users to control their lights, thermostat, and other devices using a smartphone app.



Fig 2.1.1 NODEMCU (esp8266)

One of the challenges with the ESP8266 is its limited memory and processing power. This means that it may not be suitable for more complex applications, and developers may need to optimize their code to make the most of the available resources. However, for many IoT applications, the ESP8266 is more than powerful enough.

Pin Type	Pin Name	Pin Description
Power	Micro-USB, 3.3V, Vin, GND	Micro-USB: It can power the NodeMCU. 3.3V: A regulated 3.3V supply can also power the NodeMCU. Vin: External power supply pin GND: Ground pins
Control pins	EN, RST	The button and pin can reset the microcontroller.
Analog pins	A0	It measures analog voltage within the range of 0-3.3v.
GPIO pins	GPIO1 to GPIO16	There are sixteen general-purpose input-output pins on the NodeMCU board.
SPI pins	SD1, CMD, SD0, CLK	There are four pins that work for SPI communication on NodeMCU.
UART pins	TXD0, RXD0, TXD2, RXD2	NodeMCU features two UART interfaces including UART0 (RXD0 and TXD0) and UART1 (RXD1 and TXD1). You can use UART1 to upload programs or firmware
I2C pins.		You can get I2C functionality support on NodeMCU. However, you have to find the I2C pin because of these pins' internal functionality.

Table 2.1.1 Pin details

1 ADC channel	1 channel of 10-bit precision SAR ADC
2 UART interfaces	2 UART interfaces with support for flow control
4 PWM outputs	4 PWM pins to control things like motor speed or LED brightness
2 SPI and 1 I2C interfaces	Two SPI and one I2C interfaces for connecting various sensors and peripherals

Table 2.1.2 Peripherals and I/O

UART Pins:

The ESP8266 has two UART interfaces, UART0 and UART2, that support asynchronous communication (RS232 and RS485) at up to 4.5 Mbps. UART0 (TXD0, RXD0, RST0, and CTS0 pins) is used for communication. UART1 (TXD1 pin) only features a data transmit signal and is typically used for printing logs.

ADC Pins:

The ESP8266 is embedded with a 10-bit precision SAR ADC, which means it can detect 1024 (2^{10}) discrete analog levels. In other words, it will convert input voltages ranging from 0 to 3.3V (operating voltage) into integer values ranging from 0 to 1024. This results in a resolution of 3.3 volts / 1024 units, or 0.0032 volts (3.2 mV) per unit

2.2. MPU6050

The MPU6050 accelerometer is a small and versatile sensor that is used in a variety of electronic devices to measure acceleration, orientation, and rotation. It is a highly integrated sensor that combines a 3-axis accelerometer and a 3-axis gyroscope in a single package, which makes it a popular choice for many applications.

It has a built-in 16-bit ADC that provides high-resolution data for accurate measurement of acceleration and rotation.

Pin Number	Pin Name	Description
1	Vcc	Provides power for the module, can be +3V to +5V. Typically +5V is used
2	Ground	Connected to Ground of system
3	SerialClock (SCL)	Used for providing clock pulse for I2C Communication
4	SerialData (SDA)	Used for transferring Data through I2C communication
5	Auxiliary Serial Data (XDA)	Can be used to interface other I2C modules with MPU6050. It is optional
6	Auxiliary Serial Clock (XCL)	Can be used to interface other I2C modules with MPU6050. It is optional
7	AD0	If more than one MPU6050 is used a single MCU, then this pin can be used to vary the address
8	Interrupt (INT)	Interrupt pin to indicate that data is available for MCU to read.

Table 2.2.1 Pin description of MPU6050

One of the key features of the MPU6050 is its motion detection capabilities. The sensor can detect motion in six different directions, which makes it useful in applications where motion sensing is critical. For example, it can be used to detect when a device is being moved or shaken, which is useful in gaming applications where the player's movements need to be tracked.

Another important feature of the MPU6050 is its ability to provide accurate orientation data. This is achieved by combining the data from the accelerometer and gyroscope to calculate the device's pitch, roll, and yaw. This information is useful in applications such as virtual reality, where precise orientation data is essential.

The MPU6050 is also highly customizable, with a range of configurable settings that allow users to optimize the sensor for their specific application. For example, users can adjust the sampling rate, filter settings, and power management options to achieve the best performance for their particular use case.

The MPU6050 can be used in a wide range of applications, including robotics, drones, gaming, and motion tracking. In robotics, the MPU6050 can be used to measure the orientation and acceleration of a robot, which is essential for ensuring its stability and control.

In drones, the MPU6050 can be used to stabilize the drone during flight and to measure the drone's orientation and acceleration.

Another application of the MPU6050 is in motion detection and gesture recognition. The sensor can detect when a device is being shaken or tilted, which can be useful for detecting user input or for triggering certain actions in the device. The MPU6050 can also be used for gesture recognition, such as detecting when a user performs a specific gesture or movement.

The MPU6050 is a low-cost and efficient sensor that is easy to integrate into electronic devices. It is also highly accurate and reliable, which makes it a popular choice for many designers and engineers. Furthermore, its small size and low power consumption make it ideal for use in portable and battery-powered devices.

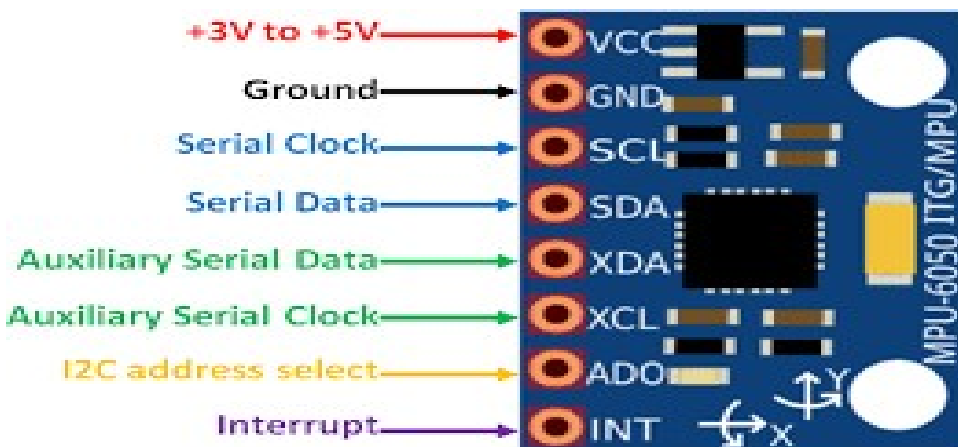


Fig 2.2.1 MPU6050

2.3 Pulse detector

A pulse detector is a device that is used to detect the pulse rate of an individual. It is commonly used in medical settings to monitor the vital signs of patients. The pulse detector works by measuring the changes in blood flow in the arteries. This information is then converted into an electrical signal, which is then displayed on a monitor or recorded for further analysis.

The pulse detector is a vital tool in the healthcare industry. It is used to monitor the pulse rate of patients who are critically ill or undergoing surgery. The device helps doctors and nurses keep track of changes in a patient's condition and adjust treatment accordingly. In addition, the pulse detector is used in fitness and wellness settings to monitor the pulse rate of athletes during exercise or training sessions.

The most common type of pulse detector is the fingertip pulse oximeter. This device is placed on the fingertip of the patient and uses light to measure the changes in blood flow. The pulse oximeter is non-invasive and painless, making it ideal for use in medical settings. It is also small and portable, allowing doctors and nurses to easily move it from one patient to another.

Another type of pulse detector is the ECG or electrocardiogram. This device is used to measure the electrical activity of the heart. The ECG is commonly used to diagnose heart problems such as arrhythmia, heart attack, and heart failure. The device consists of several electrodes that are placed on the chest, arms, and legs of the patient. The electrical signals from the electrodes are recorded and displayed on a monitor for analysis.

In addition to medical settings, the pulse detector is also used in fitness and wellness settings. Athletes use heart rate monitors to monitor their pulse rate during training and competition. The heart rate monitor consists of a chest strap that is worn around the chest and a watch or monitor that displays the pulse rate. The heart rate monitor is used to help athletes optimize their training by ensuring that they are exercising at the right intensity.



Fig 2.3.1 Circuit Diagram of a Pulse Sensor

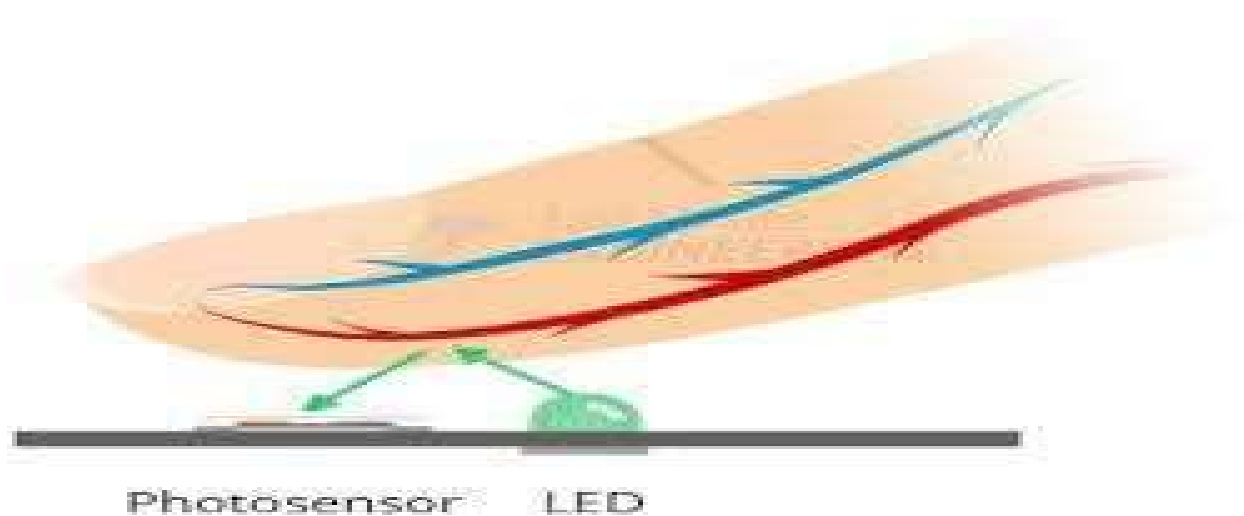


Fig 2.3.2 Working of Pulse sensor using IR rays

2.4 Arduino IDE

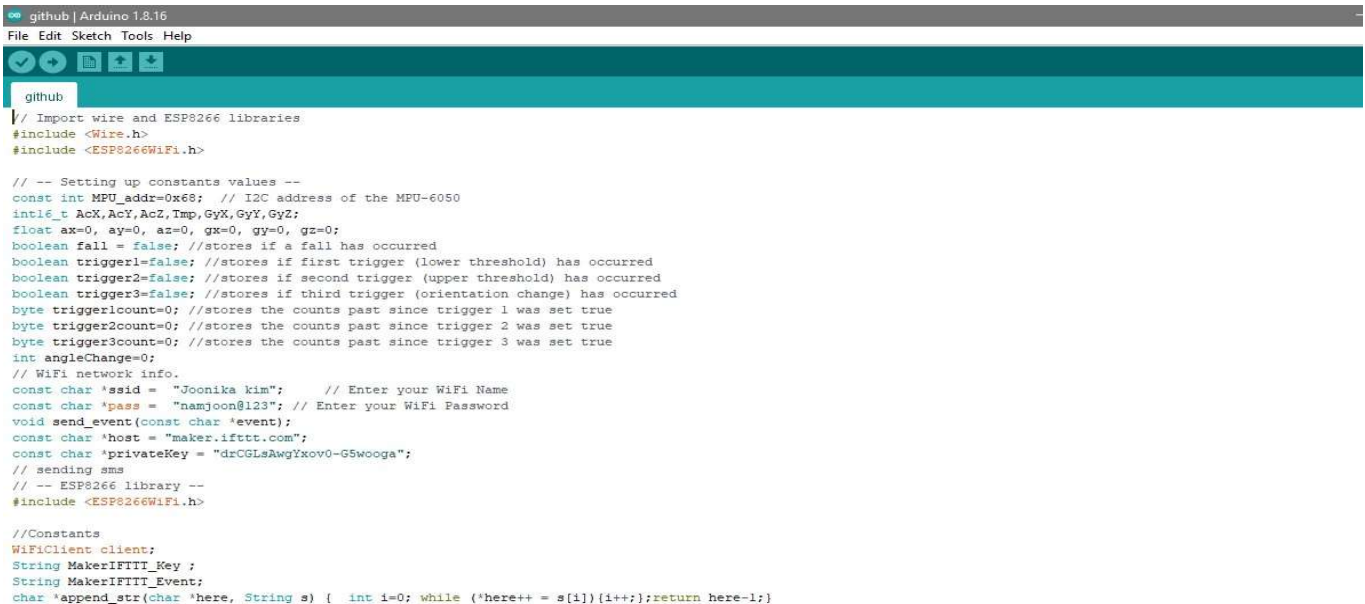
Arduino IDE (Integrated Development Environment) is a software platform used to program and develop projects using the Arduino microcontroller board. Arduino IDE is an open-source software that is freely available to download and use on multiple operating systems such as Windows, Mac OS, and Linux. The software is designed to provide an easy-to-use platform for beginners and experts to create, upload, and debug programs for the Arduino boards.

The Arduino IDE features a user-friendly interface that provides a text editor for writing code, a message console to display output and error messages, and a toolbar to compile and upload code to the connected Arduino board. Additionally, the IDE includes a library manager that allows users to easily download and manage external libraries, making it easy to add additional functionality to their projects.

One of the main advantages of using the Arduino IDE is that it simplifies the programming process by abstracting away the complexity of low-level programming tasks such as register settings and communication protocols. This allows users to focus on the high-level logic of their projects, making it easier to prototype and iterate.

Moreover, the Arduino IDE supports a wide range of programming languages, including C and C++, making it easy to write complex programs for the Arduino board. Additionally, the software includes a robust set of debugging and testing tools, allowing users to quickly identify and fix bugs in their code.

Overall, the Arduino IDE is an essential tool for anyone interested in creating projects using the Arduino board. Its ease of use, flexibility, and wide range of features make it an excellent platform for both beginners and experts to develop projects of varying complexity.



```
github | Arduino 1.8.16
File Edit Sketch Tools Help

github
// Import wire and ESP8266 libraries
#include <Wire.h>
#include <ESP8266WiFi.h>

// -- Setting up constants values --
const int MPU_addr=0x68; // I2C address of the MPU-6050
int16_t AcX,AcY,AcZ,Tmp,GyX,GyY,GyZ;
float ax=0, ay=0, az=0, gx=0, gy=0, gz=0;
boolean fall = false; //stores if a fall has occurred
boolean trigger1=false; //stores if first trigger (lower threshold) has occurred
boolean trigger2=false; //stores if second trigger (upper threshold) has occurred
boolean trigger3=false; //stores if third trigger (orientation change) has occurred
byte trigger1count=0; //stores the counts past since trigger 1 was set true
byte trigger2count=0; //stores the counts past since trigger 2 was set true
byte trigger3count=0; //stores the counts past since trigger 3 was set true
int angleChange=0;
// WiFi network info.
const char 'ssid' = "Joonika kim"; // Enter your WiFi Name
const char 'pass' = "namjoon@123"; // Enter your WiFi Password
void send_event(const char 'event');
const char 'host' = "maker.ifttt.com";
const char 'privateKey' = "drCGLsAwgYxov0-G5wooga";
// sending sms
// -- ESP8266 library --
#include <ESP8266WiFi.h>

//Constants
WiFiClient client;
String MakerIFTTT_Key ;
String MakerIFTTT_Event;
char 'append_str(char 'here, String s) { int i=0; while ('here++ = s[i]){i++;}return here-1;}
```

Fig 2.4.1 The Arduino ide console

2.4.1. Step-by-step procedure for adding the ESP8266 board to the Arduino IDE:

1. Open the Arduino IDE.
2. Click on the "File" menu, and select "Preferences".
3. In the "Additional Boards Manager URLs" field, paste the following URL:

http://arduino.esp8266.com/stable/package_esp8266com_index.json.
If there are other URLs in the field, separate them with commas.

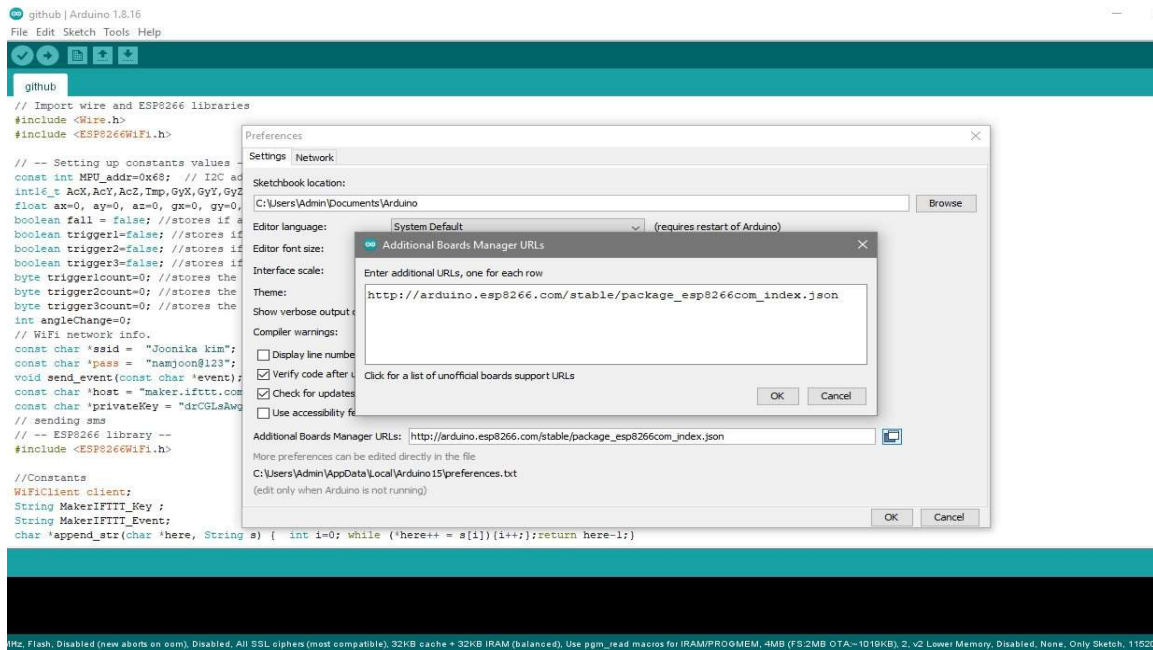


Fig 2.4.2 The URL for connecting Arduino IDE with NodeMCU

4. Click "OK" to save the preferences.
5. Next, click on the "Tools" menu, and select "Board: ...".
6. From the drop-down menu, select "Boards Manager".
7. In the search bar, type "esp8266". This should bring up the "esp8266" platform by ESP8266 Community.
8. Click on "Install" to install the ESP8266 platform.
9. Wait for the installation to complete.
10. After installation, select the ESP8266 board you want to use from the "Tools" > "Board" menu. For example, if you are using the ESP8266 NodeMCU board, select "NodeMCU 1.0 (ESP-12E Module)".

11. Next, select the correct port from the "Tools" > "Port" menu. The port may vary depending on your system, but it should have the word "ESP" or "Arduino" in the name.

12. Now you are ready to write and upload code to the ESP8266 board using the Arduino IDE.

That's it! By following these steps, you should now be able to program the ESP8266 board using the Arduino IDE.

By following these steps our Arduino ide will look something similar the figure given below

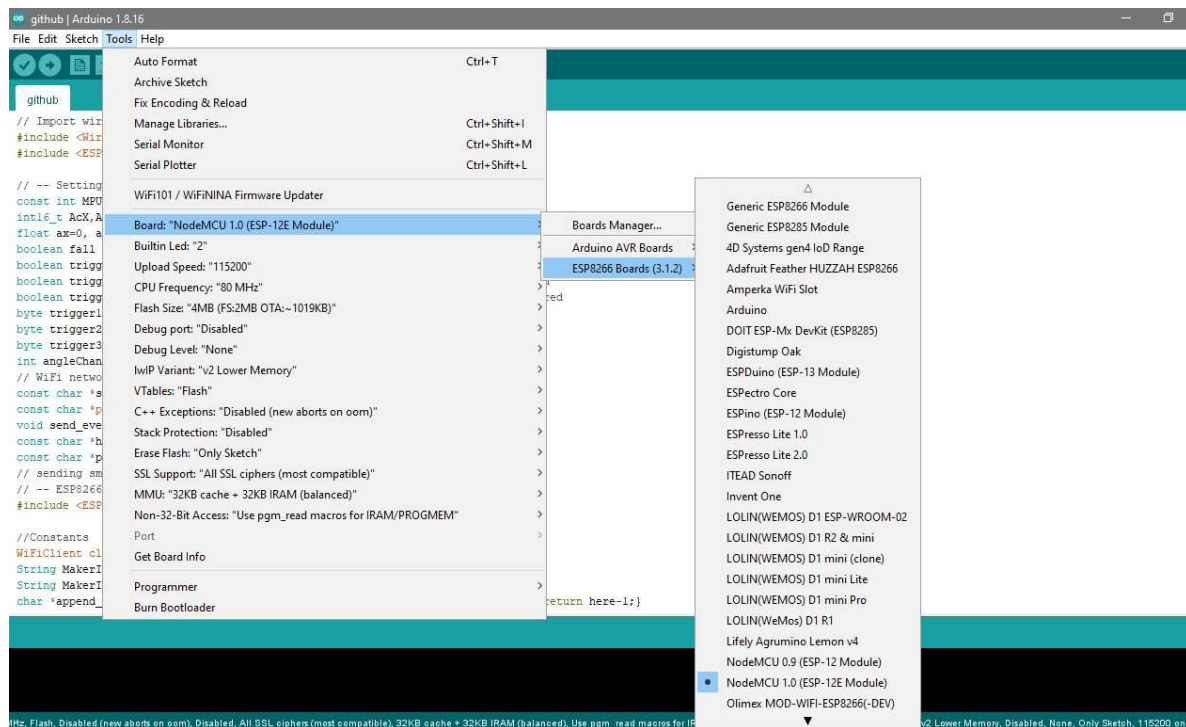


Fig 2.4.3 The boards in the esp8266 is shown here

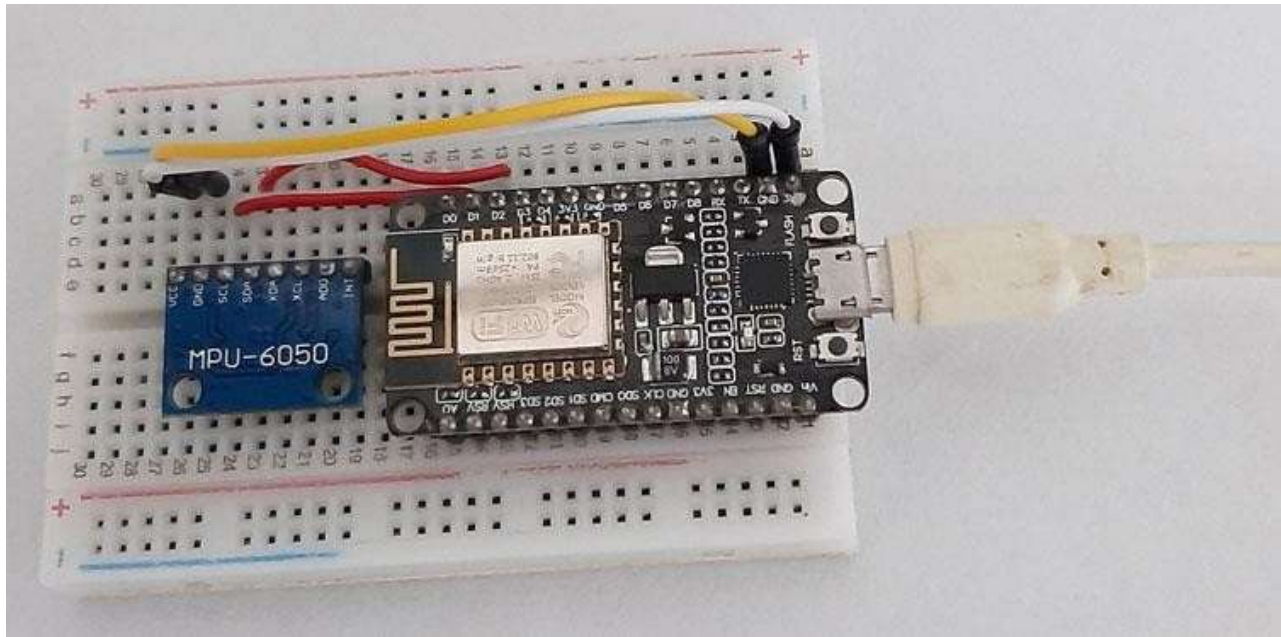


Fig 2.4.4 The Actual picture of our Project with MPU6050

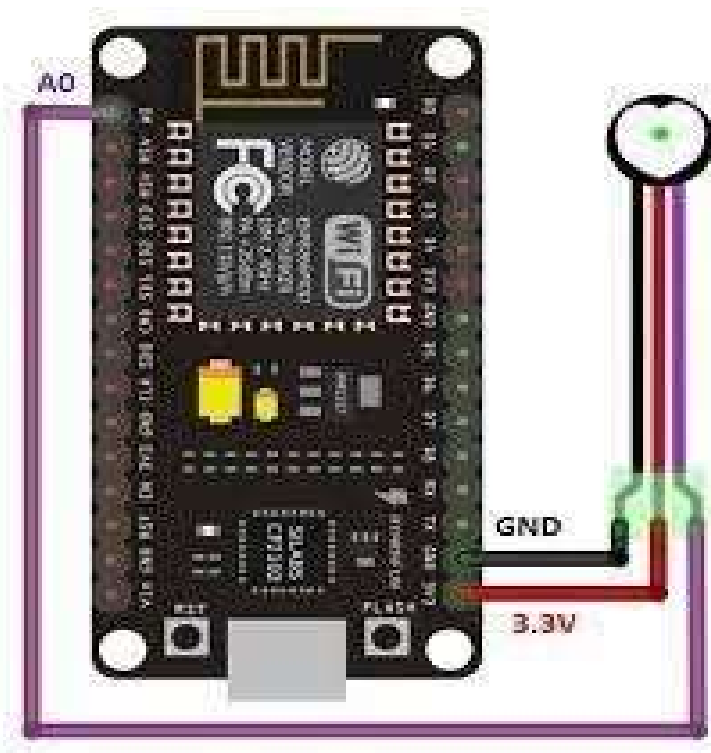


Fig 2.4.5. The connection of Pulse sensor with NodeMCU

2.5 Why NodeMCU is the best option?

There are many microcontrollers that are well-suited for IoT projects, depending on your specific requirements and constraints. Here are a few popular options:

Arduino: Arduino is the best choice for IoT projects because its boards are durable and simpler than others, making them easier to maintain and program. With the backing of the Arduino IoT Cloud and its programming environments.

Raspberry Pi: The Raspberry Pi is a credit card-sized computer that can be used as a microcontroller for IoT projects. It has built-in Wi-Fi and Bluetooth connectivity and can run a variety of operating systems, including Linux.

ESP32: The ESP32 is a low-cost, low-power microcontroller that is designed specifically for IoT applications. It has built-in Wi-Fi and Bluetooth connectivity, as well as a dual-core processor and a range of sensors and peripherals.

Particle Photon: The Particle Photon is a Wi-Fi-enabled microcontroller that is designed for IoT applications. It has a built-in cloud platform that makes it easy to build and deploy IoT applications.

STM32: The STM32 is a family of microcontrollers that are well-suited for IoT projects. They are low-power and have a range of connectivity options, including Wi-Fi, Bluetooth, and Lora-Connectivity options, and cost.

NodeMCU and Arduino are both popular microcontroller platforms used in DIY electronics and IoT projects. However, each platform has its strengths and weaknesses, and the choice between them depends on the specific needs of the project.

Here are some reasons why NodeMCU might be considered better than Arduino for certain projects:

1. **Built-in Wi-Fi:** NodeMCU has built-in Wi-Fi capabilities, which makes it a better choice for IoT projects that require wireless connectivity. Arduino boards, on the other hand, require an additional Wi-Fi module or shield to connect to a wireless network.
2. **Lua scripting language:** NodeMCU is programmed using Lua, a scripting language that is easy to learn and use. This makes it a good choice for beginners who want to get started with programming.
3. **More memory:** NodeMCU has more memory than most Arduino boards, which allows it to handle more complex tasks and larger programs.
4. **Lower cost:** NodeMCU boards are often cheaper than Arduino boards, which makes them a more cost-effective choice for projects with a tight budget.

Ultimately, the choice between NodeMCU and Arduino depends on the specific needs of the project, and both platforms have their own strengths and weaknesses that should be considered.

CHAPTER 3 - IFTTT

3.1 What is IFTTT?

IFTTT stands for "**If This Then That**," and is a web-based service that allows users to create applets, which are small programs that automate tasks across different web services. The service has gained popularity for its ability to connect and automate a wide range of apps, smart devices, and web services.

The IFTTT platform works based on conditional statements that trigger an action when a certain event occurs. These conditional statements are called "applets" and consist of two parts: the trigger and the action. The trigger is the event that initiates the action, while the action is the resulting task that is carried out.

Users can create their own applets or choose from a library of pre-built applets that are designed to connect various services, such as social media, email, productivity tools, and smart devices. For example, an applet can be created to automatically save all Gmail attachments to a Dropbox folder or to turn on the lights when a motion sensor detects movement.

IFTTT has become popular for its ability to connect smart home devices such as Amazon Echo, Google Home, and Philips Hue lights. By using applets, users can control their smart home devices from a single app, or automate tasks such as turning off lights when leaving the house or adjusting the thermostat based on the weather outside.

The service has also been used to automate business workflows and improve productivity. For example, an applet can be created to automatically add new leads from a Facebook Lead Ad campaign to a Google Sheet or to send a Slack notification when a Trello task is completed.

One of the key advantages of IFTTT is its simplicity and user-friendly interface. Users do not need to have any programming skills to create applets and automate tasks. The service also provides a wide range of integrations, making it easy to connect different web services and smart devices.

3.2 Keywords

3.2.1 Action:

In IFTTT, an action is the task that is carried out as a result of a specific trigger. When an applet is triggered, the action is what happens next. For example, if a user creates an applet that is triggered when they post a photo on Instagram, the action could be to automatically save that photo to a specific folder in their Dropbox account.

Actions in IFTTT can be customized to meet the user's needs. There are many different actions available in IFTTT, such as sending an email, creating a calendar event, sending a text message, and turning on a smart home device. Users can choose the specific action they want to occur when a trigger is activated.

Actions can also be combined with filters and other tools to create more complex applets. For example, a user can create an applet that is triggered when they receive an email with an attachment.

The action could be to automatically save that attachment to a specific folder in their Google Drive account. The user can also add filters to the applet so that it only triggers for emails from specific senders or with certain keywords in the subject line.

In addition to the pre-built actions available in IFTTT, users can also create their own custom actions using the IFTTT platform. This allows users to create applets that are tailored to their specific needs and workflows.

3.2.2 Applets

Applets are small programs that automate tasks across different web services and smart devices in IFTTT. They consist of two parts: a trigger and an action. The trigger is the event that initiates the action, while the action is the resulting task that is carried out.

Users can create their own applets or choose from a library of pre-built applets that are designed to connect various services, such as social media, email, productivity tools, and smart devices. These pre-built applets are called "recipes" in older versions of IFTTT.

Applets in IFTTT can be used to automate a wide range of tasks. For example, a user can create an applet that is triggered when they receive an email from their boss. The action could be to automatically send a text message to the user's phone with the subject line of the email. Another example is an applet that is triggered when the user arrives at their office.

The action could be to automatically turn on the lights and adjust the thermostat to a comfortable temperature.

Applets can also be used to automate business workflows and improve productivity. For example, a user can create an applet that is triggered when a new lead is added to their Salesforce account. The action could be to automatically create a task in their Trello board to follow up with that lead.

Users can also customize their applets with filters and other tools to create more complex workflows. For example, a user can create an applet that is triggered when they receive an email with an attachment. The action could be to automatically save that attachment to a specific folder in their Google Drive account. The user can also add filters to the applet so that it only triggers for emails from specific senders or with certain keywords in the subject line.

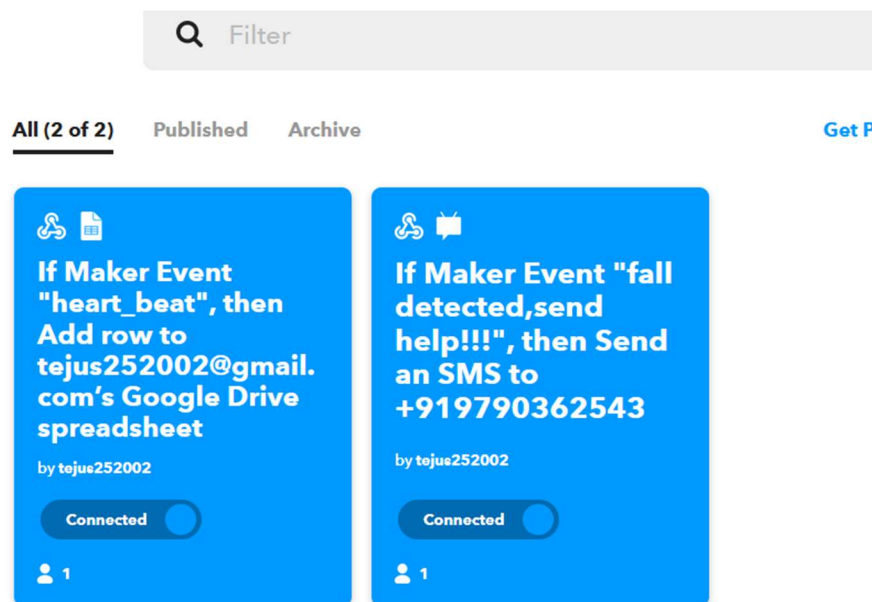


Fig 3.2.2.1 Applets

3.2.3 Ingredients:

In IFTTT, an ingredient is a placeholder value that is used to represent dynamic data within an applet. Ingredients allow users to create flexible and customizable applets that can handle different types of data.

For example, if a user creates an applet that is triggered when they post a tweet on Twitter, they can use ingredients to customize the resulting action. They can use the ingredient "TweetText" to insert the text of the tweet into the action, or they can use the ingredient "TweetURL" to insert a link to the tweet.



Fig 3.2.3.1 Ingredients

3.2.4 Query and query field

In IFTTT, a query is a request for data from a web service or smart device that is used in an applet. The query field is where users input the specific parameters of their query to customize the data that is returned.

For example, if a user creates an applet that is triggered when a new email is received in their Gmail account, they can use a query to filter the resulting action to only include emails that meet specific criteria. The query field would allow the user to input parameters such as the sender's email address, specific keywords in the subject line, or certain labels or categories.

Queries are specific to the web service or smart device that is being used in the applet, and the available query fields depend on the specific service. Some common query fields in IFTTT include:

- **Date and time:** allow users to customize the date and time format in an applet.
- **Location:** allows users to customize the location data in an applet.
- **Weather:** allows users to customize the weather data in an applet.
- **Email:** allows users to customize the email data in an applet, such as the subject line, sender, and message body.

3.2.5 Services:

In IFTTT, a service is a web application or smart device that provides data and functionality that can be integrated with other services to create applets. Examples of popular services in IFTTT include Gmail, Twitter, Amazon Alexa, Philips Hue, and Nest.

Each service has its own set of triggers and actions that can be used in applets. Triggers are events that occur within a service, such as a new email being received or a motion sensor detecting movement.

To create an applet in IFTTT, users select a trigger from one service and an action from another service, and then customize the parameters to create a workflow that meets their specific needs. For example, a user might create an applet that is triggered when they receive a new email in their Gmail account, and then sends a notification to their phone using the Push bullet service.

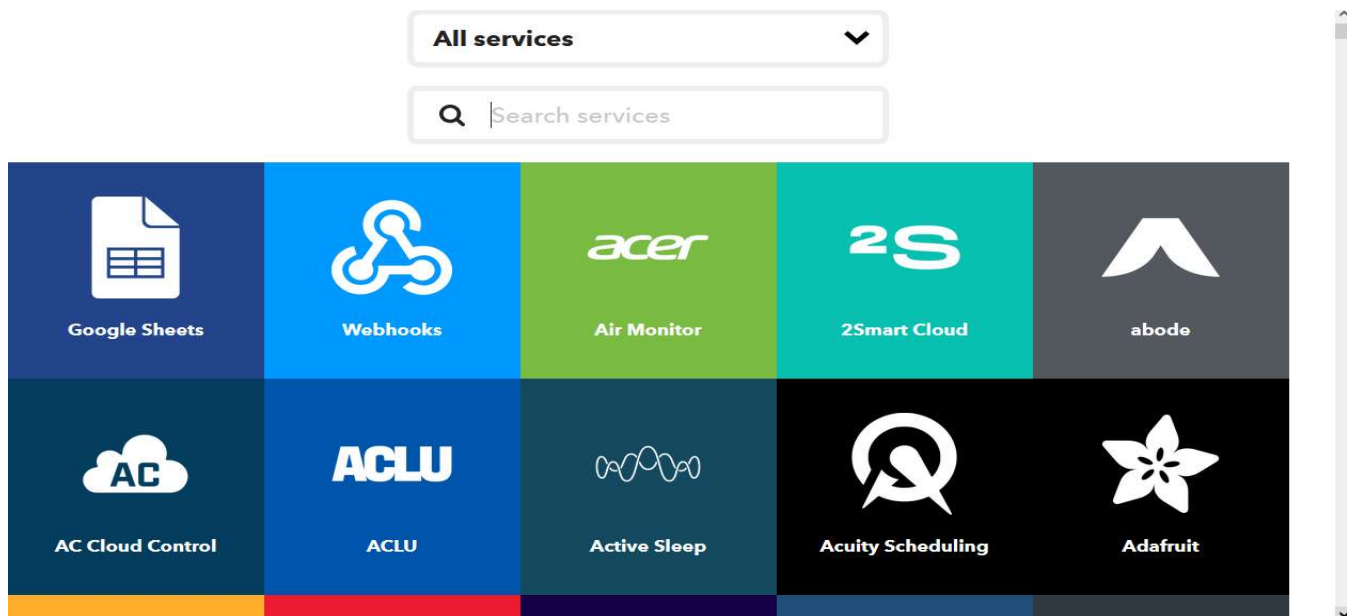


Fig 3.2.1 Services provided by IFTT

3.2.6 Trigger:

In IFTTT, a trigger is an event that initiates an applet. A real-time trigger is a specific type of trigger that is designed to immediately initiate an applet as soon as the trigger event occurs.

Real-time triggers are used for time-sensitive or mission-critical workflows, where it is important for the applet to be triggered as quickly as possible after the event occurs. For example, a real-time trigger could be used to automatically turn on the lights in a room as soon as motion is detected, or to receive a push notification as soon as a specific email is received.

Trigger events are the specific events that initiate a trigger in an applet. For example, a trigger event could be a new email being received, a specific keyword being mentioned on Twitter, or a motion sensor detecting movement. IFTTT supports a wide range of trigger events across many different services, allowing users to create highly customized applets.

Trigger checks are used to verify that a trigger event has occurred before initiating an applet. This helps to ensure that applets are only triggered under the desired conditions, and can prevent false positives or unnecessary actions. For example, if an applet is triggered by a new email being received, the trigger check could verify that the email was received from a specific sender or contains a certain keyword before performing the action.

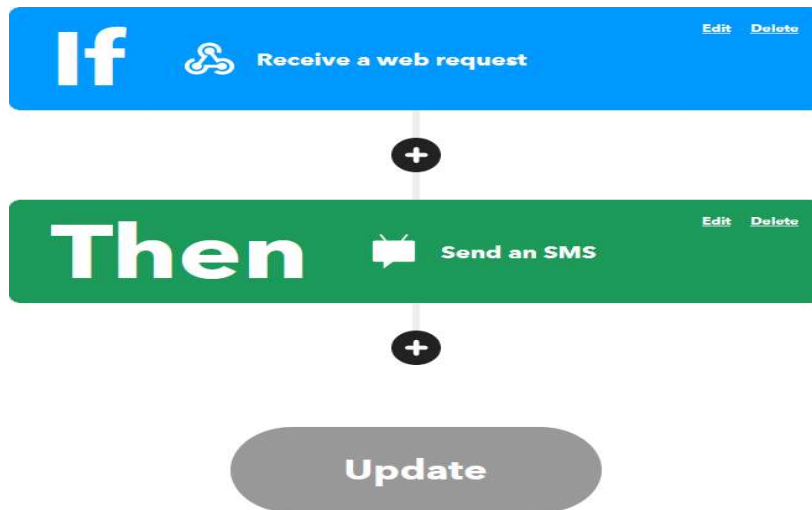


Fig.3.2.2 Trigger in IFTT (IF block)

Triggers, trigger events, real-time triggers, and trigger checks are important components of the IFTTT platform that allow users to create powerful automations and integrations.

3.2.7 Applet ID

The Applet ID is an 8-character alphanumeric identifier given to each Applet on IFTTT. For example, this Applet's ID is UMzGVYr9

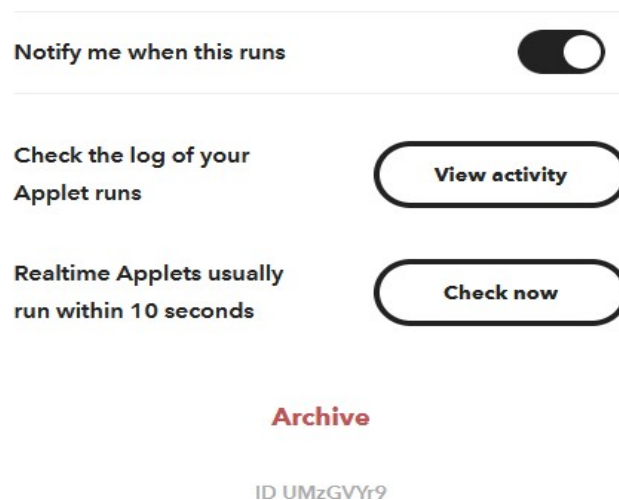


Fig 3.2.3 Applet ID in IFTTT

3.2.8 Step by step Procedure for creating an Applet

1. Go to the IFTTT website (<https://ifttt.com/>) and create an account. If you already have an account, log in.
2. Once you are logged in, click on your username in the top right corner of the screen and select "Create" from the dropdown menu.
3. You will be taken to a page where you can choose between creating a "Personal Applet" or a "Service". For this guide, we will be creating a "Personal Applet".
4. Click on "Personal Applet" and then click on the blue "This" button to choose a trigger for your applet. A trigger is the event that will cause your applet to run. You can choose from a variety of services such as Gmail, Facebook, Twitter, etc.
5. Once you have selected a trigger, you will need to connect the service to IFTTT by logging in and authorizing IFTTT to access your account.
6. Once the service is connected, you will need to configure the trigger by specifying the details such as the specific keyword, hashtag or the content that you want the applet to trigger on.
7. After you have configured the trigger, click on the blue "That" button to choose an action. An action is the task that your applet will perform when the trigger occurs. You can choose from a variety of services such as Evernote, Dropbox, Slack, etc.

8. Once you have selected an action, you will need to connect the service to IFTTT by logging in and authorizing IFTTT to access your account.
9. Once the service is connected, you will need to configure the action by specifying the details such as the folder or label to which the content should be saved, the message that should be sent, etc.
10. Once you have configured the action, click on the blue "Create Action" button to create your applet.
11. Your applet is now created and will run automatically when the trigger occurs.
12. You can go to "My Applets" to see a list of all the applets you have created. You can edit or delete an applet by clicking on the three dots on the right side of the applet.

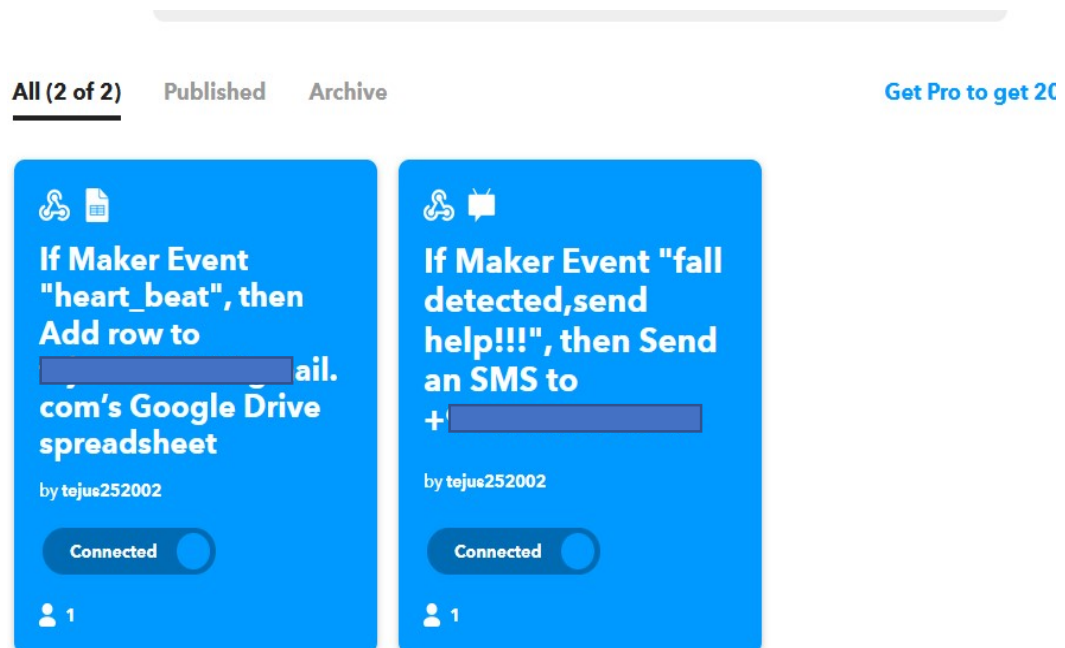


Fig 3.2.4 Applets created on IFTTT

3.2.9 Why is IFTTT better than using Transmitter and Receiver Module?

IFTTT (If This Then That) and Transmitter/Receiver modules serve different purposes, so it's difficult to make a direct comparison between the two.

IFTTT is a web-based service that enables you to create custom automations between various internet-connected devices and services. It allows you to set up triggers (the "if this" part) and actions (the "then that" part) to automate tasks. For example, you can set up an automation to turn on your smart lights when your smart door lock is unlocked.

One of the key benefits of IFTTT is its versatility. It supports a wide range of devices and services, including smart home devices, social media platforms, and productivity apps. This means you can create automations that span multiple devices and services, which can be difficult to achieve with a Transmitter/Receiver module. IFTTT also has a user-friendly interface that makes it easy to set up and manage automations, even if you have no programming experience.

Another advantage of IFTTT is that it's a cloud-based service, which means you don't need to have physical hardware to create automations. This can save you time and money, as you don't need to purchase and set up a Transmitter/Receiver module for each device you want to automate. Instead, you can use IFTTT to create automations that work across all your devices.

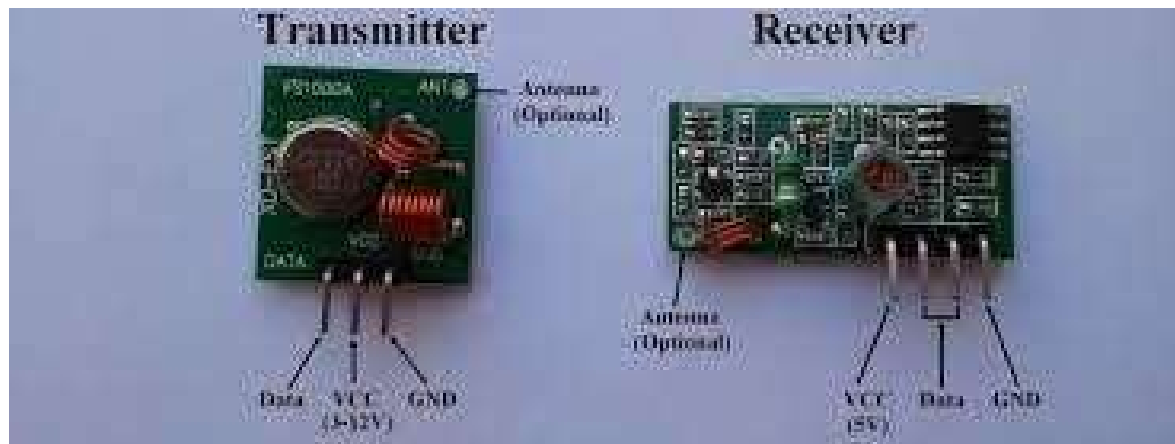


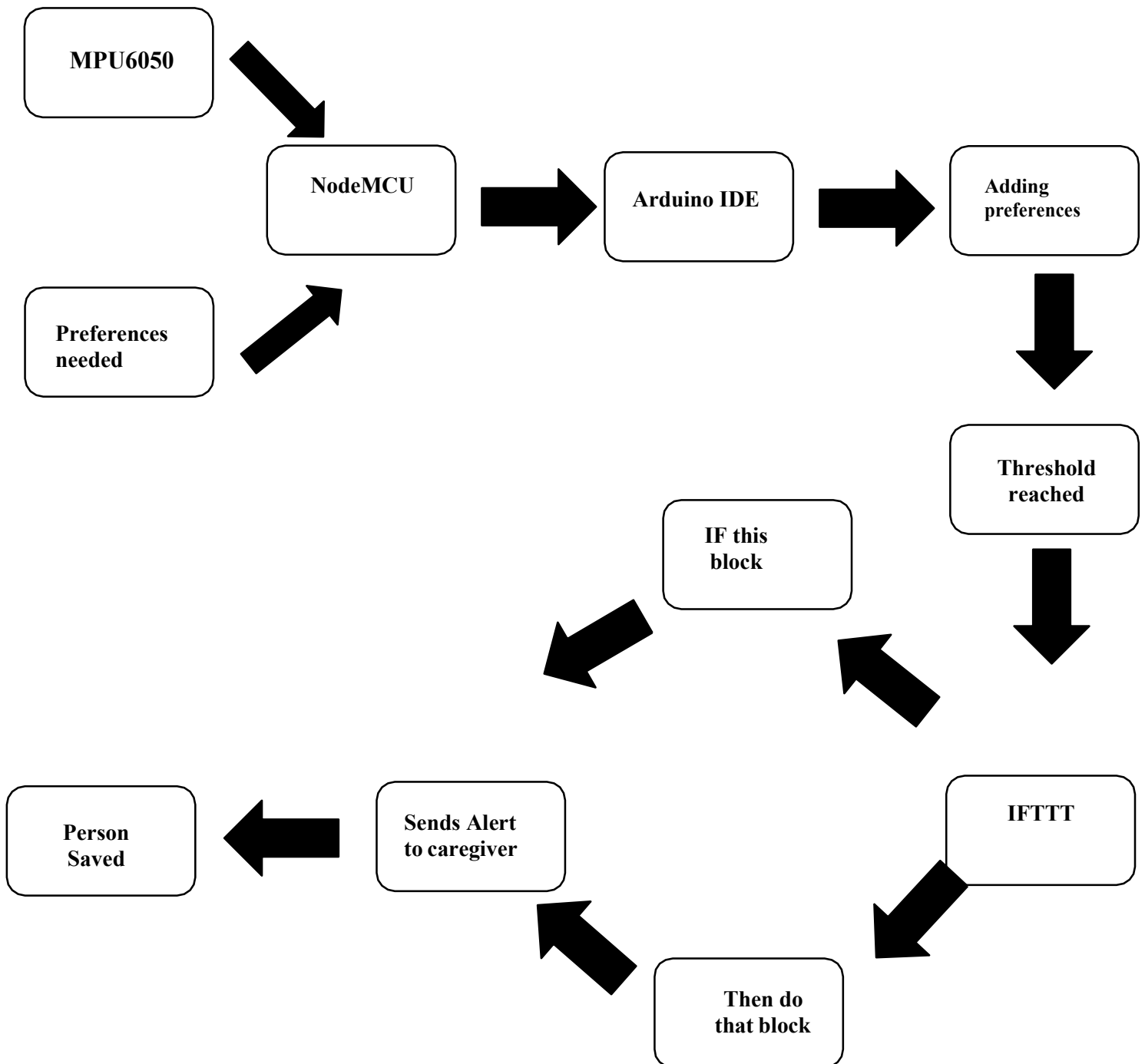
Fig 3.2.5 Transmitter Receiver Module VS IFTTT

Finally, IFTTT has a large community of users who create and share their own automations. This means you can benefit from the expertise and creativity of others, and find inspiration for new automations that you might not have thought of otherwise.

IFTTT offers a more versatile and user-friendly way to automate tasks across multiple devices and services without the need for physical hardware.

Chapter 4 The block diagram

The complete project idea of the fall detection is given here in simple block.



4.1 Explanation of this Block Diagram

The MPU6050 is a commonly used accelerometer and gyroscope sensor module that can be connected to a NodeMCU microcontroller board. By combining these two components, it is possible to create a fall detection system that can help prevent falls in the elderly.

The MPU6050 can detect changes in motion and orientation in real-time and send this data to the NodeMCU board. The NodeMCU can then process this data and analyze it to determine if a fall has occurred. If a fall is detected, the system can alert a caregiver or family member via a notification on their smartphone or other connected device, through hardware like the Transmitter and Receiver Module. In some cases using Software like IFTTT, Zapier etc.

The system can also be programmed to send an emergency alert to a monitoring center or emergency services if a fall is detected, ensuring that the elderly person receives immediate assistance.

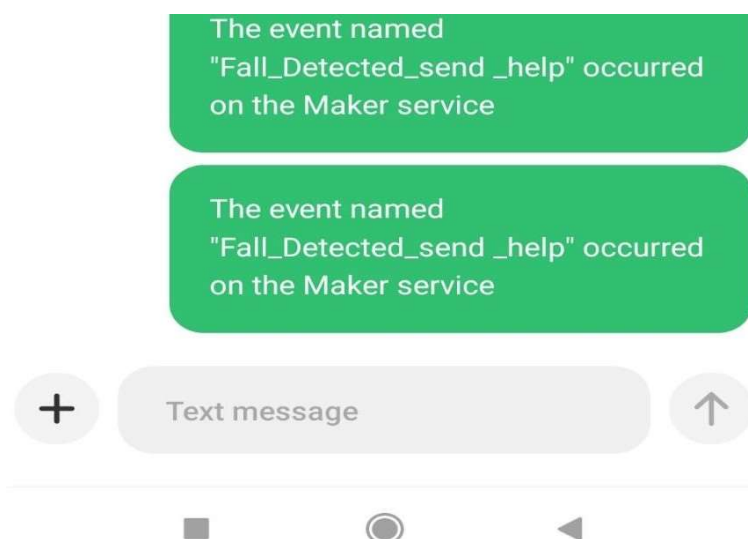
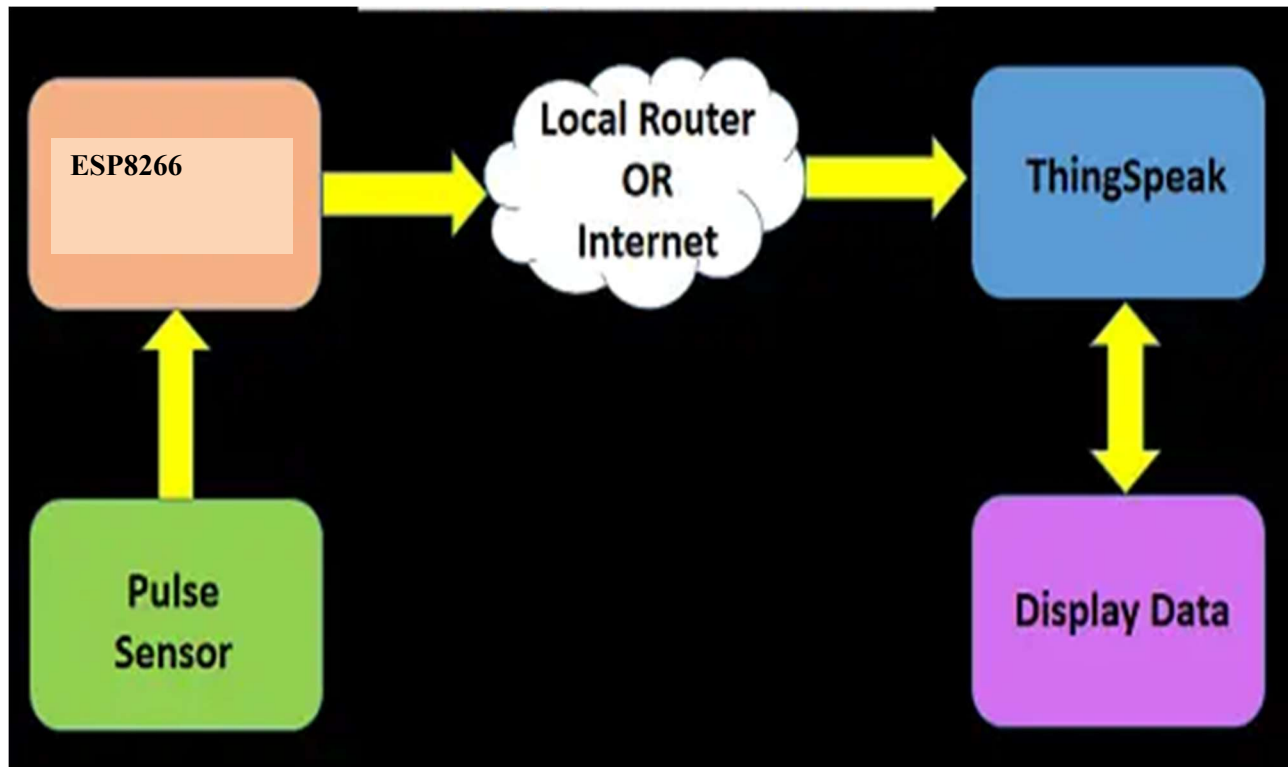


Fig 4.1.1 The Alert message received on the mobile device

4.2 Simple representation of how a Pulse sensor work



- The pulse sensor is connected to the ESP8266 and by adding several libraries needed to run a pulse sensor.
- The sensor works and gives us an output on the serial monitor.
- By Creating the Channel in the ThingSpeak cloud we can send data from the NodeMCU board to the ThingSpeak console.
- The data will be on the Visualization board for easy understanding

Chapter 5 Literature Survey

5.1. Online Fall Detection Using Wrist Devices

João Marques and Plinio Moreno

More than 37 million falls that require medical attention occur every year, mainly affecting the elderly. Besides the natural consequences of falls, most aged adults with a history of falling are likely to develop a fear of falling, leading to a decrease in their mobility level and impacting their overall quality of life. Previous wrist-based datasets revealed limitations such as unrealistic recording set-ups, lack of proper documentation and, most importantly, the absence of elderly people's movements.

Therefore, this work proposes a new wrist-based dataset to tackle this problem. With this dataset, exhaustive research is carried out with the low computational FS-1 feature set (maximum, minimum, mean and variance) with various machine learning methods. This work presents an accelerometer-only fall detector streaming data at 50 Hz, using the low computational FS-1 feature set to train a 3NN algorithm with Euclidean distance, with a window size of 9 s.

REFERENCE

- Legters, K. Fear of falling. *Phys. Ther.* **2002**, 82, 264–272. [[Google Scholar](#)] [[CrossRef](#)] [[PubMed](#)][[Green Version](#)]
- Vellas, B.J.; Wayne, S.J.; Romero, L.J.; Baumgartner, R.N.; Garry, P.J. Fear of falling and restriction of mobility in elderly fallers. *Age Ageing* **1997**, 26, 189–193. [[Google Scholar](#)]

5.2 Machine learning algorithms for fall detection

The application of machine learning techniques to detect and classify falls is a prominent area of research in the domain of intelligent assisted living systems. Machine learning (ML) based solutions for fall detection systems built on wearable devices use various sources of information such as inertial motion units (IMU), vital signs, acoustic or channel state information parameters. Most existing research rely on only one of these sources;

however, a need to do more experimentation to observe the efficiency of the ML classifiers while coupling features from diverse sources, was felt. In addition, fall detection systems based on wearable devices, require intelligent feature engineering and selection for dimensionality reduction, so as to reduce the computational complexity of the devices.

In this paper we have a comprehensive performance analysis of ML classifiers for fall detection, on a dataset we collected. The analysis includes the impact of the following aspects on the performance of ML classifiers for fall detection:

- (i) Using a combination of features from 2 sensors-an IMU sensor and a heart rate sensor
- (ii) Feature engineering and feature selection based on statistical methods
- (iii) Using ensemble techniques for fall detection.

5.3 MACHINE LEARNING ALGORITHMS (used for fall detection)

Support Vector Machines (SVMs): SVMs are a popular algorithm used for classification tasks, including fall detection. SVMs are effective at separating two classes of data by finding.

Random Forests: Random forests are an ensemble algorithm that combines multiple decision trees to make predictions. They are commonly used for fall detection because they can handle high dimensional data and are resistant to overfitting.

Convolutional network (CNNs): Convolutional network is a one type of neural network that are commonly used for image recognition they can handle for high dimensional data and are resistant to overfitting.

Recurrent Neural Networks (RNNs): RNNs are a type of neural network that are commonly used for time-series data, such as sensor data. They can be used for fall detection by analyzing data from wearable sensors.

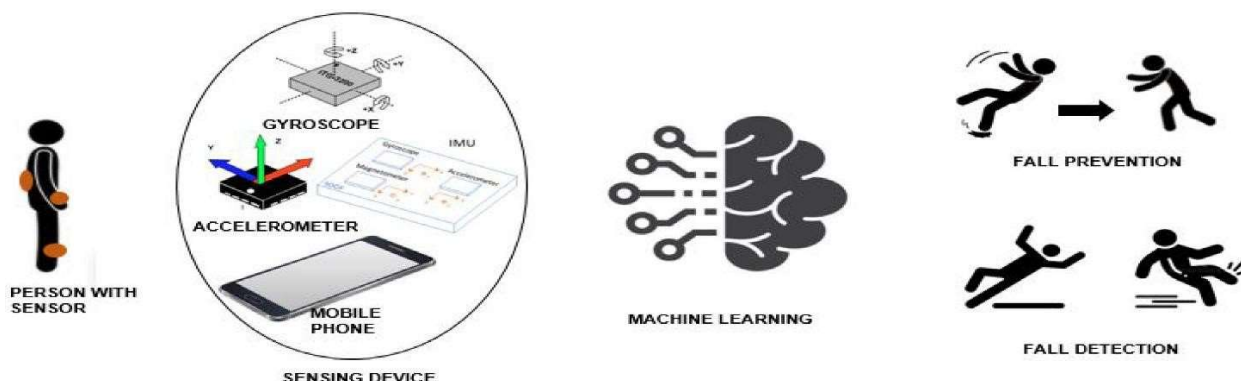


Fig 5.3.1 The future scope of fall detection due to machine learning

CHAPTER -6 THINGSPEAK

6.1 What is ThingSpeak

- ThingSpeak™ is an IoT analytics platform service that allows you to aggregate, visualize and analyze live data streams in the cloud. ThingSpeak provides instant visualizations of data posted by your devices to ThingSpeak. With the ability to execute MATLAB® code in ThingSpeak you can perform online analysis and processing of the data as it comes in. ThingSpeak is often used for prototyping and proof of concept IoT systems that require analytics.
- You can send data from any internet-connected device directly to ThingSpeak using a Rest API or MQTT. In addition, cloud-to-cloud integrations with The Things Network, Senet, the Libelium Meshlium gateway.
- With ThingSpeak, you can store and analyze data in the cloud without configuring web servers, and you can create sophisticated event-based email alerts that trigger based on data coming in from your connected devices.



Fig 6.1.1 Simple representation of the working of ThingSpeak

6.2 Key Features of ThingSpeak Cloud:

1. **Data Collection:** ThingSpeak Cloud allows users to collect data from various IoT devices and sensors. It supports multiple communication protocols such as MQTT, HTTP, and HTTPS, enabling seamless integration with a wide range of IoT devices.
2. **Channel Architecture:** ThingSpeak organizes data into channels, which act as containers for incoming data streams. Each channel consists of multiple fields that store different types of data, such as temperature, humidity, or GPS coordinates. Users can define the number and type of fields based on their specific application requirements.
3. **Real-time Data Handling:** ThingSpeak Cloud is designed to handle real-time data streams efficiently. It provides an MQTT broker that allows devices to publish data using the publish-subscribe pattern, ensuring instant updates and notifications. Real-time data visualization and analysis are integral components of ThingSpeak Cloud.
4. **Data Storage:** ThingSpeak Cloud offers built-in data storage capabilities. It retains data for a configurable period, allowing users to access historical data for analysis, visualization, and comparison. The storage capacity depends on the user's subscription level, with higher tiers providing larger storage limits.

5. **Data Analysis:** ThingSpeak Cloud provides an integrated MATLAB® Analytics environment, which allows users to apply powerful data analysis techniques. Users can write MATLAB® code directly within ThingSpeak to perform custom calculations, filtering, statistical analysis, and more on the collected data.
6. **Visualization:** ThingSpeak Cloud offers a variety of visualization options to represent data effectively. Users can create custom visualizations using MATLAB® or choose from a range of predefined visualizations such as line plots, scatter plots, gauges, and maps. These visualizations can be embedded in websites, shared with others, or used for real-time monitoring.
7. **Alerts and Notifications:** ThingSpeak Cloud allows users to set up alerts based on specific conditions or thresholds. When a condition is met, such as a sensor reading exceeding a predefined threshold, an alert can be triggered. Alerts can be sent via email, SMS, or custom webhooks, enabling users to take immediate action based on received notifications.
8. **Integration with External Services:** ThingSpeak Cloud supports integration with external services and platforms through webhooks and APIs. This allows for seamless interaction with third-party services, such as sending data to other cloudplatforms, triggering actions on external systems, or receiving data from externalsources.

9. **User Management and Access Control:** ThingSpeak Cloud provides user management features, allowing multiple users to collaborate on projects. Access control mechanisms enable users to define permissions and roles, ensuring that data and channels are securely shared with the appropriate individuals or groups.

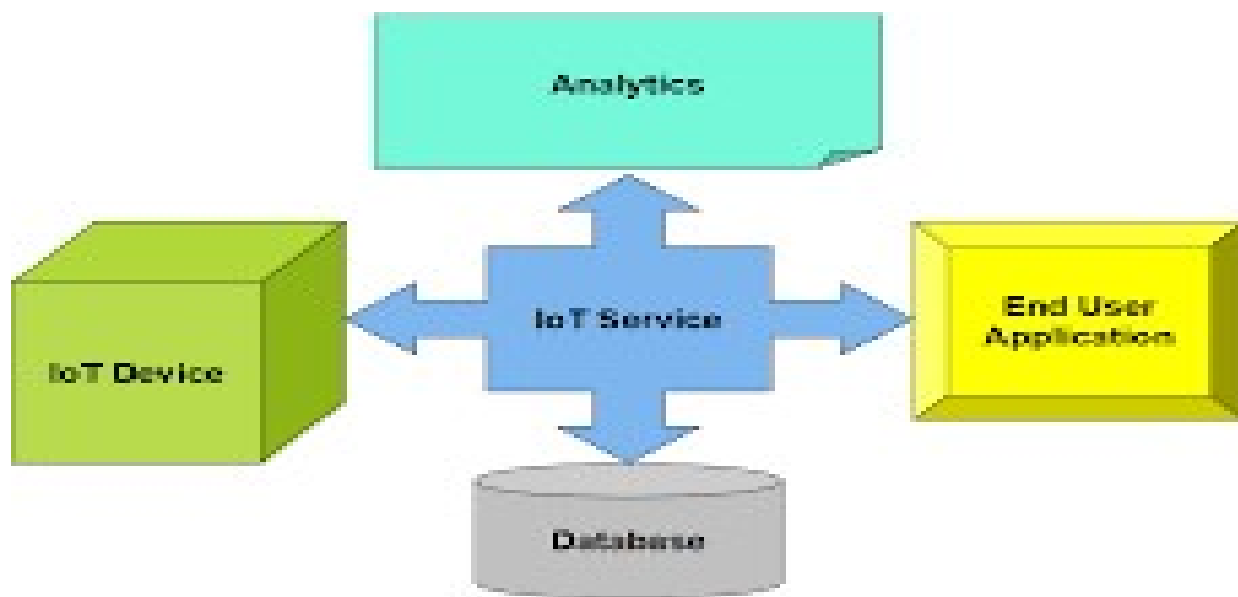


Fig 6.2.1 Key features of ThingSpeak

6.3 Architecture of ThingSpeak Cloud:

The architecture of ThingSpeak Cloud consists of several components working together:

1. **IoT Devices:** These are the physical devices or sensors that collect data from the environment. IoT devices can connect to ThingSpeak Cloud using supported communication protocols and publish data to specific channels.

10. **ThingSpeak Cloud Server:** The ThingSpeak Cloud server is the central component that receives, stores, and processes the data sent by IoT devices. It provides the necessary infrastructure to handle data streams, store data in a time-series database, and perform real-time analytics.
11. **Channels:** Channels are the entities within ThingSpeak Cloud that act as containers for data. Each channel can have multiple fields that represent different data types. Users can create and configure channels based on their specific application requirements.
12. **Analytics Engine:** ThingSpeak Cloud incorporates an integrated MATLAB® Analytics engine. This engine allows users to write MATLAB® code within ThingSpeak to perform complex data analysis, apply algorithms, create mathematical models, and generate insights from the collected data.
13. **Storage:** ThingSpeak Cloud provides built-in data storage capabilities. It stores the collected data in a time-series database, allowing users to access historical data for analysis and visualization purposes. The storage capacity depends on the user's subscription level.
14. **Visualization Tools:** ThingSpeak Cloud offers a range of visualization tools and widgets. Users can choose from pre-built visualizations or create custom visualizations using MATLAB®. These visualizations enable users to understand the data patterns, trends, and anomalies.

15. **APIs and Webhooks:** ThingSpeak Cloud provides APIs and webhooks to enable seamless integration with external services and platforms. Users can interact with ThingSpeak Cloud programmatically, send data to other cloud platforms, trigger actions on external systems, or receive data from external sources.

Overall, ThingSpeak Cloud simplifies the development of IoT applications by providing a robust and scalable infrastructure for data collection, analysis, visualization, and integration with other services. Its rich set of features and user-friendly interface make it an attractive choice for IoT developers and enthusiasts.

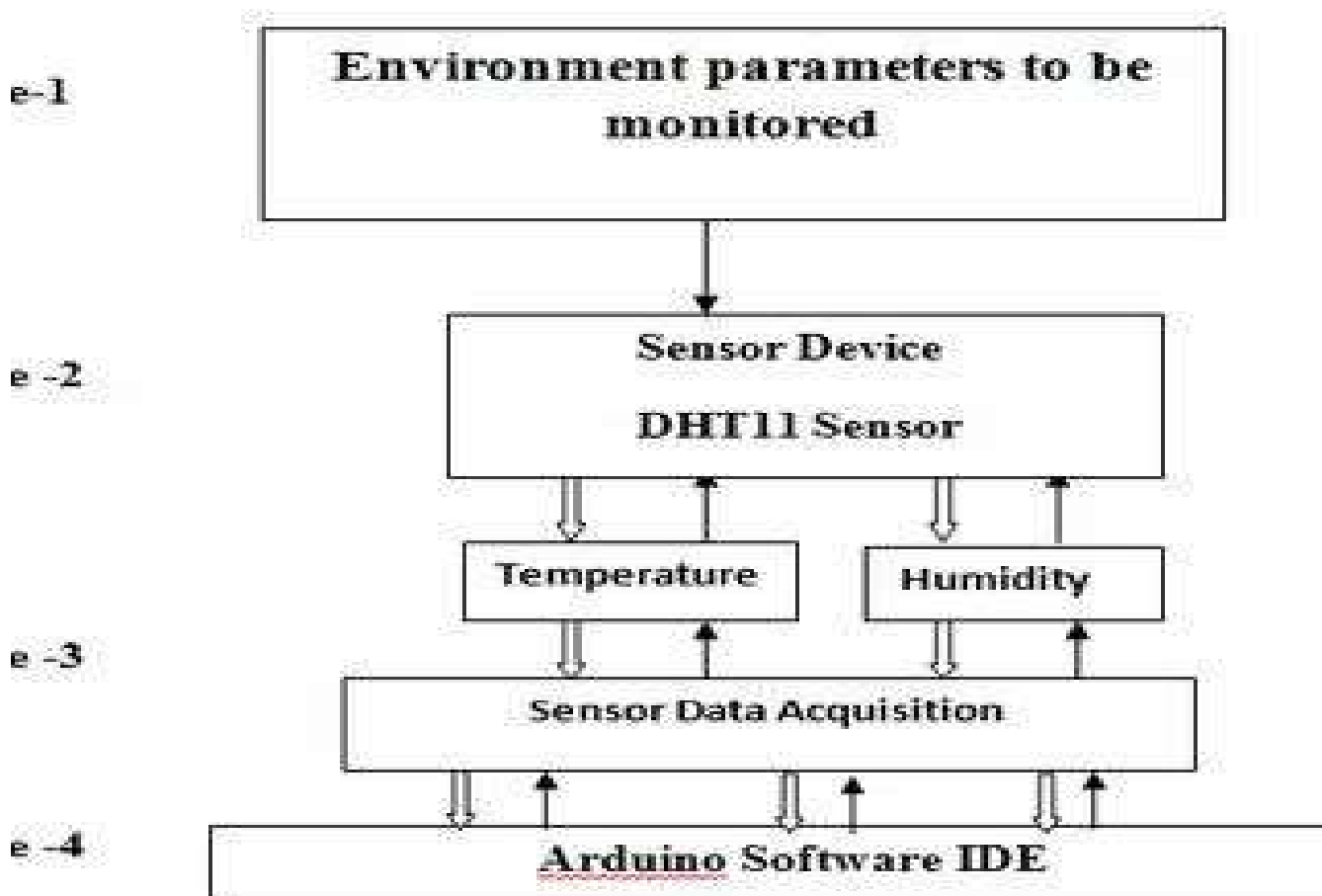


Fig 6.3.1 Simple Architecture of a Sensor data Acquisition

6.4 Step by step procedure for creating a channel in ThingSpeak

To create a channel in ThingSpeak, follow these step-by-step instructions:

1. **Go to the ThingSpeak website:** Open your preferred web browser and navigate to the ThingSpeak website at <https://thingspeak.com/>.
2. **Sign up or log in:** If you don't already have an account, click on the "Sign Up" button and create a new account. If you have an existing account, click on the "Log In" button and enter your credentials to log in.
3. **Access the Channels page:** After signing in, you will be redirected to the ThingSpeak platform. Click on the "Channels" tab located in the navigation menu at the top of the page.
4. **Create a new channel:** On the Channels page, click on the green "New Channel" button. This will open the channel creation form.
5. **Fill in channel details:** In the channel creation form, you need to provide the following information:
 - **Name:** Enter a descriptive name for your channel.
 - **Description:** Provide a brief description of what the channel will be used for.
 - **Field labels:** Specify labels for up to eight fields. These labels will be used to identify the data you send to each field.

- **Field tags (optional):** You can add tags to categorize your channel and make it easier to find in searches.
 - **Metadata (optional):** Add any additional information or metadata related to your channel.
6. **Configure channel settings:** Below the channel details, you can configure various settings for your channel:
- **Public/Private:** Choose whether your channel should be public or private. Public channels can be accessed by anyone, while private channels require an API key for access.
 - **License:** Select the appropriate license for your channel's data.
 - **Advanced settings (optional):** You can customize additional settings such as geolocation, data deletion, and MATLAB® analysis.
7. **Save the channel:** Once you have filled in the required information and configured the desired settings, click on the "Save Channel" button at the bottom of the form. Your channel will be created, and you will be redirected to the channel view page.
8. **Obtain API keys (optional):** If you plan to send data to your channel using an IoT device or an application, you will need API keys. To generate API keys, click on the "API Keys" tab on the channel view page and follow the instructions to create.

9. Start using your channel: You can now start sending data to your ThingSpeak channel using various methods such as HTTP requests, MQTT, or by using the ThingSpeak API. Refer to the ThingSpeak documentation or explore the platform's features to learn more about data collection, visualization, and analysis.

By following these steps, you will successfully create a channel in ThingSpeak and be ready to start capturing and analyzing your IoT data.

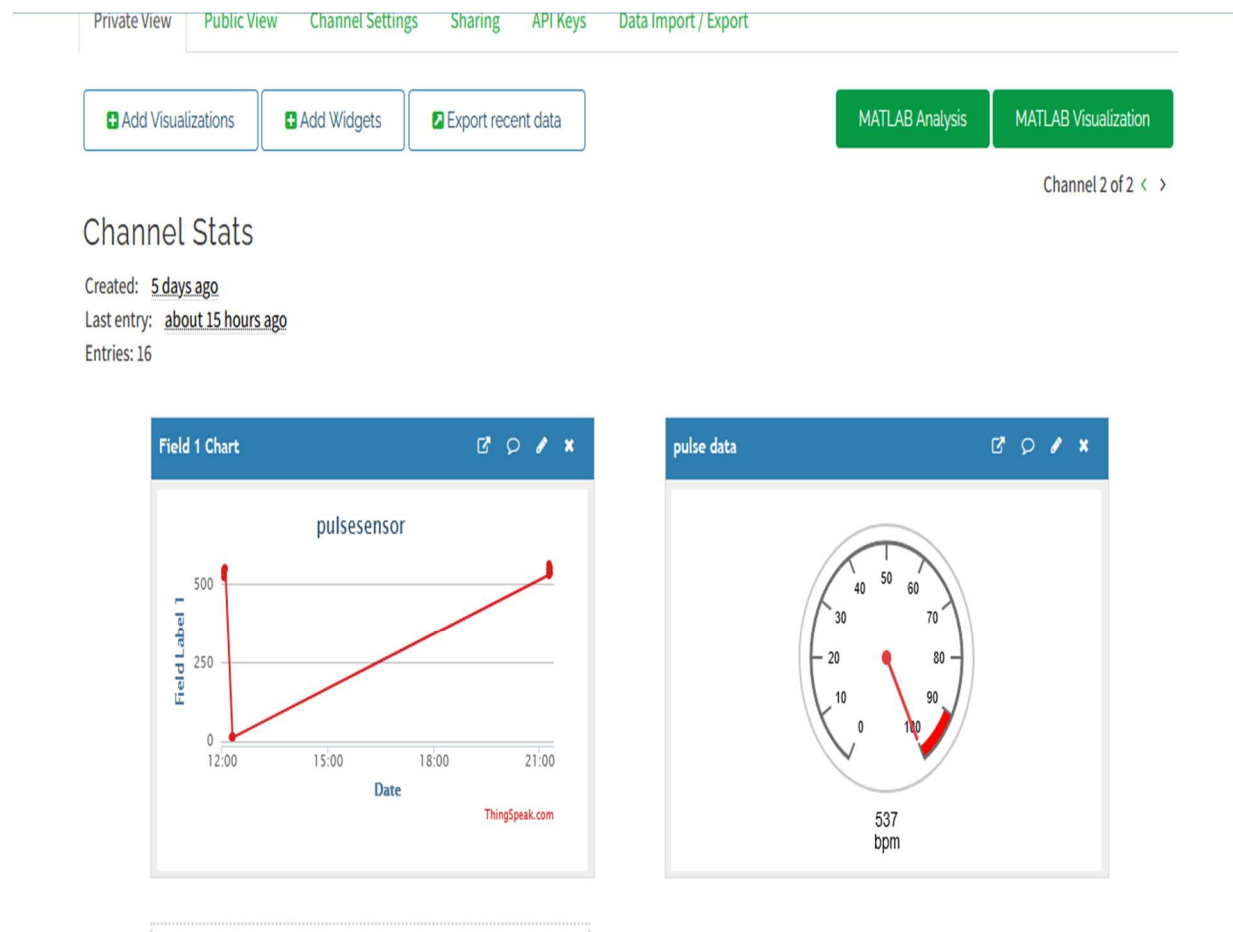


Fig 6.4.1 ThingSpeak webpage

6.5. Step by step procedure to download a dataset from ThingSpeak

To export all data from a ThingSpeak channel:

- Select **Channels > My Channels**.
- Select the channel.
- Select the **Data Import / Export** tab.
- Use the default or choose a time zone for the export.

The feed data is exported in the time zone you select in your account profile as the default. You can select a different time zone for the export using the drop-down box. See **Account > My Profile** to change your default time zone. See Time Zones Reference for a list of standard time zones.

- Click **Download**.

	A	B	C	D
1	created_at	entry_id	field1	latitud
2	2023-05-23T12:03:30+05:30		1	538
3	2023-05-23T12:03:58+05:30		2	544
4	2023-05-23T12:04:22+05:30		3	522
5	2023-05-23T12:04:46+05:30		4	550
6	2023-05-23T12:05:14+05:30		5	548
7	2023-05-23T12:17:21+05:30		6	13
8	2023-05-23T12:17:57+05:30		7	13
9	2023-05-23T12:18:21+05:30		8	14
10	2023-05-23T12:18:45+05:30		9	13
11	2023-05-23T21:15:43+05:30		10	529
12	2023-05-23T21:16:06+05:30		11	548
13	2023-05-23T21:16:30+05:30		12	562
14	2023-05-23T21:16:54+05:30		13	536
15	2023-05-23T21:17:17+05:30		14	534
16	2023-05-23T21:17:41+05:30		15	552
17	2023-05-23T21:18:05+05:30		16	537
18				
19				
20				
21				
22				
23				

Fig 6.5.1 Downloaded Excel sheet

CHAPTER -7 APPLICATIONS (of Fall detection in real lives)

1. **Elderly care:** Fall detection systems play a crucial role in monitoring and ensuring the safety of elderly individuals who are prone to falls. By using wearable devices or sensors placed around the home, caregivers or family members can be alerted in real-time when a fall occurs, enabling prompt assistance and medical attention.
2. **Assisted living facilities:** Fall detection technology is commonly employed in assisted living facilities and nursing homes to enhance resident safety. It helps staff members respond quickly to falls and provides an added layer of protection for vulnerable individuals.
3. **Home healthcare:** Fall detection systems enable remote monitoring of patients in their own homes. Healthcare professionals can receive alerts when a fall is detected, allowing them to assess the situation and provide necessary assistance or medical intervention.
4. **Hospital settings:** Fall detection technology can be integrated into hospital environments to improve patient safety. It helps healthcare providers identify potential fall risks and take preventive measures. In cases where a fall does occur, immediate notifications can be sent to the relevant medical staff, ensuring prompt response and minimizing the impact of falls.

5. **Wearable devices:** Many wearable devices, such as smartwatches or fitness trackers, include fall detection capabilities. These devices can detect sudden movements or impacts indicative of a fall and send alerts to designated contacts or emergency services. Fall detection features in wearable devices provide an added sense of security for individuals during daily activities or exercise.
6. **Emergency response systems:** Fall detection systems are often integrated into emergency response systems, especially for individuals living alone or those with medical conditions that make them more prone to falls. When a fall is detected, the system can automatically trigger emergency calls, notifying paramedics or caregivers to provide immediate assistance.
7. **Workplace safety:** Fall detection systems are employed in industrial or construction settings to enhance worker safety. By utilizing sensors and algorithms, these systems can detect falls or accidents at worksites and trigger alarms or emergency protocols to mitigate potential injuries or accidents.

Overall, fall detection technology has the potential to prevent injuries, reduce response times, and enhance the overall well-being of individuals in various settings, especially those at higher risk of falls.

CHAPTER -8 EXISTING METHODS

- **Sensor selection:** Choose appropriate sensors based on the requirements of your fall detection system. Accelerometers, gyroscopes, and pressure sensors are commonly used sensors for fall detection.
- **Sensor data collection:** Place the selected sensors on the person or object being monitored. Collect data from the sensors, typically in the form of acceleration, angular velocity, and pressure readings. The sensors may be integrated into wearable devices, such as smartwatches or body-worn sensors.
- **Feature extraction:** Process the collected sensor data to extract relevant features that can help distinguish falls from normal activities. Common features include peak acceleration, orientation changes, impact force, and temporal characteristics of the sensor signals.
- **Algorithm development:** Develop a fall detection algorithm using machine learning or signal processing techniques. This algorithm should analyze the extracted features to determine whether a fall event has occurred. Various algorithms can be used, such as threshold-based methods, pattern recognition algorithms, or machine learning classifiers (e.g., support vector machines, random forests, or artificial neural networks).
- **Training and validation:** If using machine learning techniques, train the algorithm using a labeled dataset that includes both fall and non-fall events. The algorithm should learn to differentiate between the two classes based on the extracted features. Validate the trained algorithm using independent datasets to ensure its effectiveness and generalizability.

- **Real-time implementation:** Implement the fall detection algorithm on the target platform or device. This could be a dedicated fall detection system or an integrated feature within an existing device, such as a smartwatch or a smartphone.
- **Testing and evaluation:** Conduct rigorous testing of the fall detection system in various scenarios to evaluate its performance. Measure the system's accuracy, sensitivity, specificity, and response time. Make necessary adjustments and improvements based on the test results.
- **Alerts and notifications:** Once a fall event is detected, trigger appropriate actions such as generating alerts, sending notifications to caregivers or emergency services, or activating built-in safety mechanisms (e.g., automatic emergency calls).
- **Continuous improvement:** Monitor the performance of the fall detection system over time and collect user feedback. Make updates and refinements to enhance its accuracy, reliability, and usability.

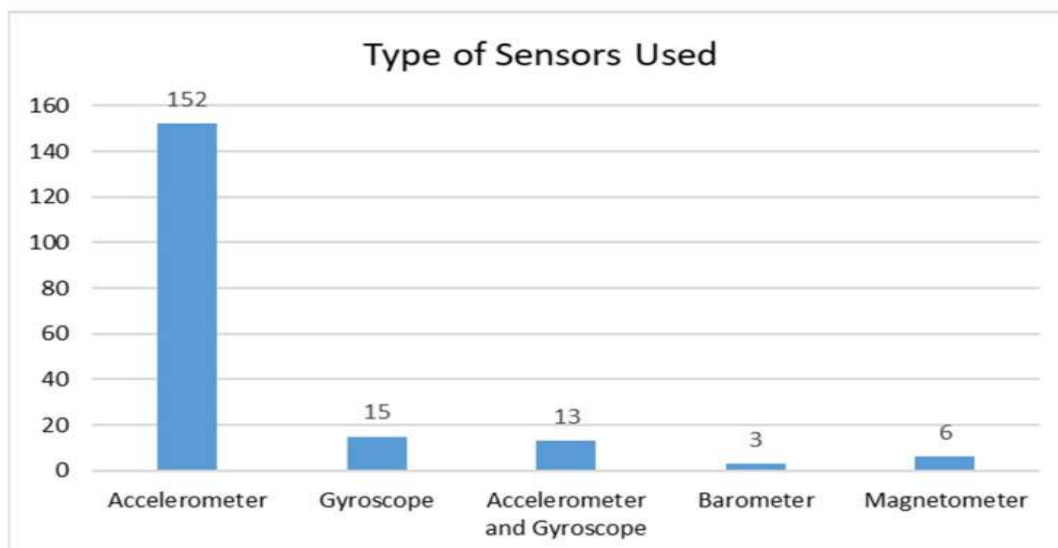


Fig 8.1 Types of sensors used in fall detection

8.1 PROPOSED METHOD (For this fall detection)

- The idea is to make a prototype that uses MPU6050 to detect falling and send alert through IFTTT software which is an alternate to the GSM module, this gives us an alert to our Android mobile when it detects a fall.
- We have also included a pulse sensor to detect our heart beat and send that data to ThingSpeak cloud, from where we can also download the dataset when needed.

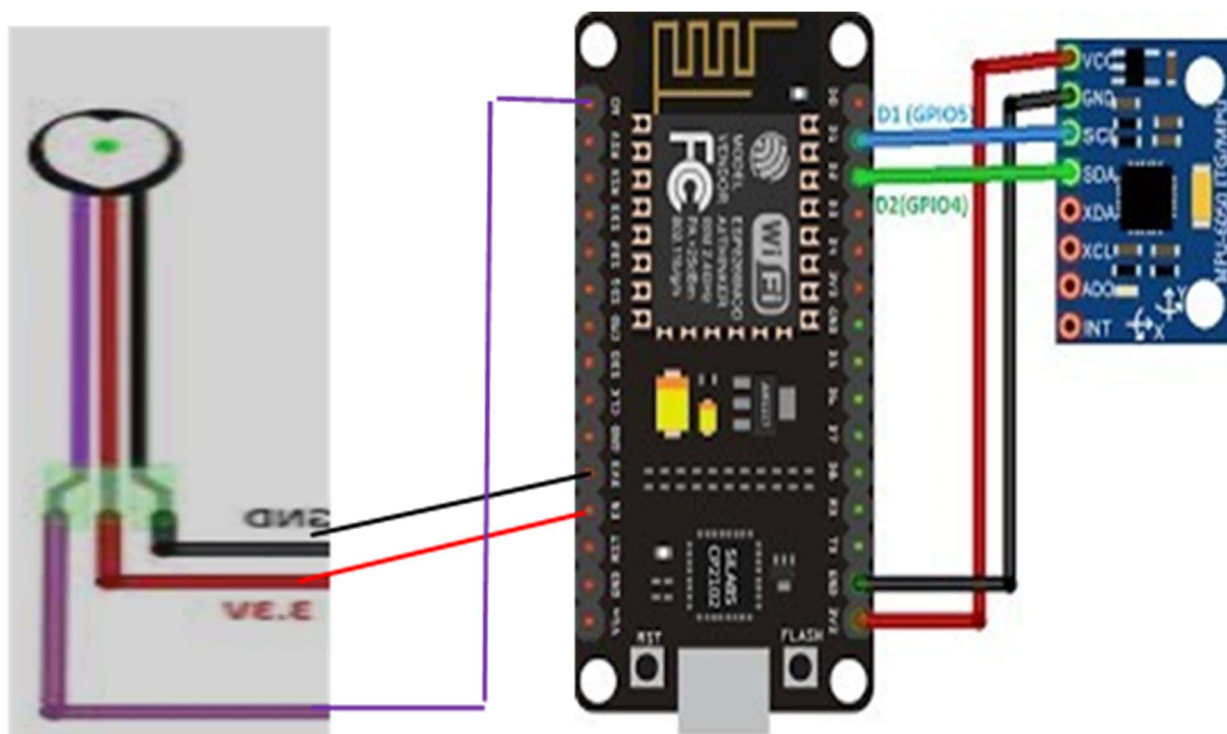


Fig 8.1.1 Visual representation of our prototype

8.2. OUTPUTS FROM IFTTT AND THINGSPEAK

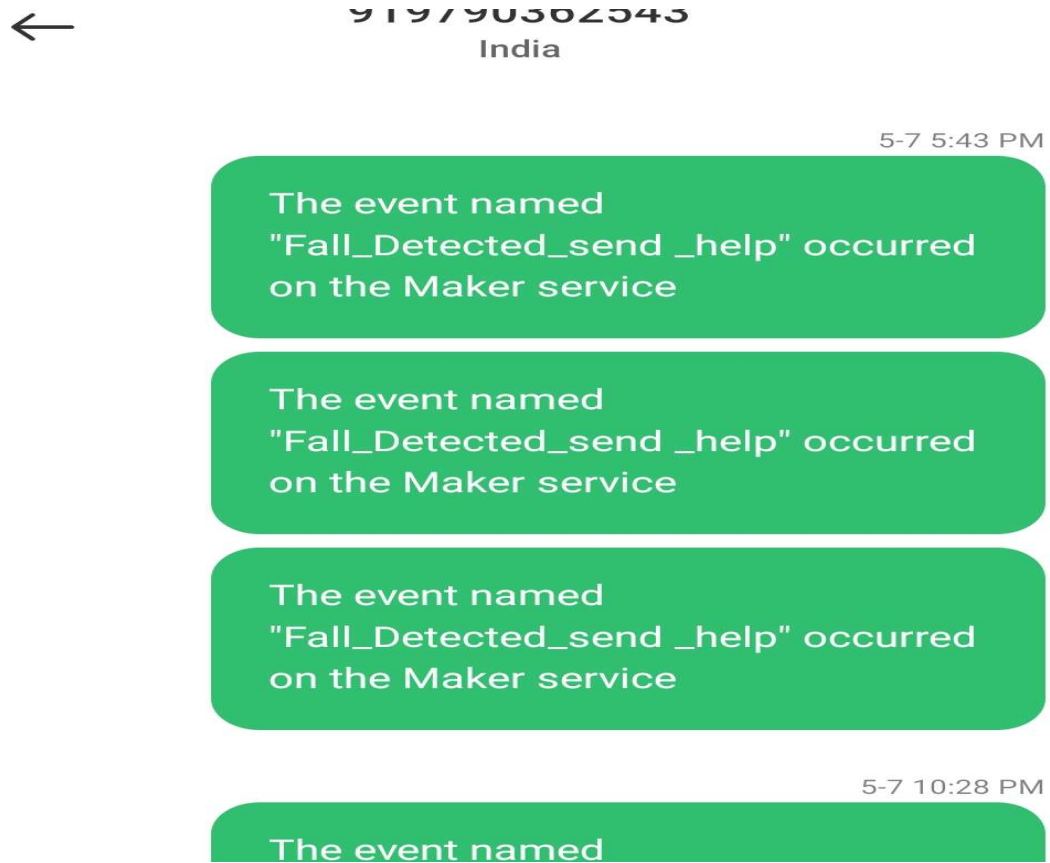


Fig 8.2.1 Alert on Android device

entries: 16

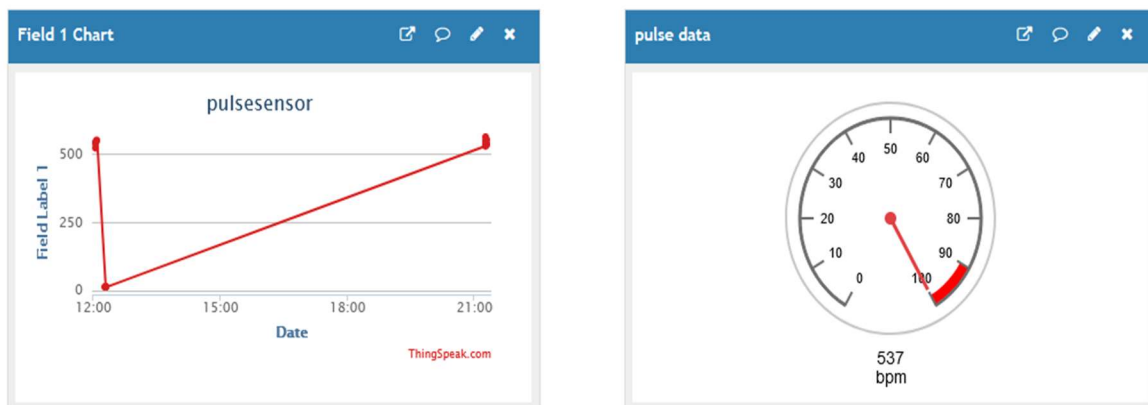


Fig 8.2.2. Output on ThingSpeak cloud

CHAPTER 9 - CONCLUSION

The future scope of a NodeMCU-based fall detection system is quite promising, as it has the potential to improve the quality of life for the elderly population and reduce the burden on caregivers and healthcare providers. Some possible future developments and applications of this technology include:

1. **Improved Accuracy:** Currently, most fall detection systems rely on motion sensors to detect falls. In the future, we may see the integration of more advanced sensors, such as pressure sensors or cameras, to improve the accuracy of fall detection and reduce false positives.
2. **Machine Learning:** Machine learning algorithms can be used to analyze fall detection data and improve the accuracy of fall detection over time. As the system gathers more data, it can learn to recognize patterns and identify new types of falls.
3. **Integration with Smart Homes:** Fall detection systems can be integrated with smart home systems to provide additional support for elderly people living alone. For example, if a fall is detected, the system can automatically turn on lights or unlock doors to make it easier for emergency responders to access the home.

4. **Wearable Devices:** Wearable devices, such as smartwatches or bracelets, can be used to provide continuous fall detection and monitoring. These devices can also be used to track vital signs, such as heart rate and blood pressure, to provide a more comprehensive picture of an elderly person's health.
5. **Telemedicine:** Telemedicine technologies can be used in conjunction with fall detection systems to provide remote monitoring and care for elderly people who live far from medical facilities or who have limited mobility.

References:

- <https://iotdesignpro.com/projects/iot-based-fall-detection-system-using-nodemcu-esp8266-and-accelerometer-mpu6050>
- <https://iotprojectsideas.com/iot-fall-detector-using-mpu6050-esp8266/>
- <https://github.com/ishanjogalekar/Fall-Detection-IoT-device-using-NodeMCU-and-MPU6050>
