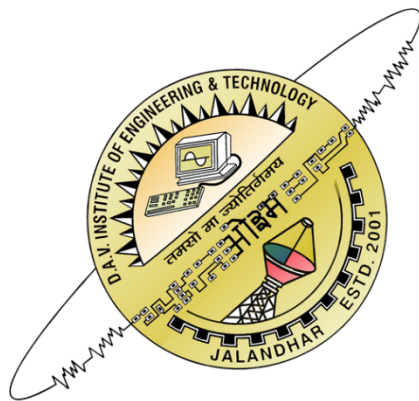


‘PATIENT HEALTH MONITORING SYSTEM’

MAJOR PROJECT REPORT

**SUBMITTED IN PARTIAL FULFILLMENT OF REQUIREMENTS FOR
THE AWARD OF THE DEGREE**



DAVIET

OF

**BACHELOR OF TECHNOLOGY IN
ELECTRONICS AND COMMUNICATION ENGINEERING**

SUBMITTED BY:

AVNISH NAGPAL (12/17) 1704545

AYUSH KUMAR (13/17) 1704546

CHIRAG NARANG (15/17) 1704548

CERTIFICATE

This is to certify that the declaration statement made by this group of students is correct to the best of my knowledge and belief. They have completed this major Project under my guidance and supervision. The present work is the result of their original investigation, effort and study. The Major Project is fit for the submission and partial fulfillment of the conditions for the award of B.Tech. degree in Electronics and Communication Engineering from DAV Institute of Engineering and Technology, Jalandhar.

.....

Signature of Project Guide

Mr. Vishav Kapoor

DECLARATION

We hereby declare that the project report entitled "**Patient Health Monitoring System**" submitted to DAV Institute of Engineering and Technology impartial fulfillment of the requirement for the award of degree of B. Tech in Electronics and Communication Department is a record of project work carried out by us. All the information furnished in this capstone project report is based on our own intensive work and is genuine.

.....

AVNISH NAGPAL

.....

AYUSH KUMAR

.....

CHIRAG NARANG

ACKNOWLEDGEMENT

First and foremost, we sincerely salute over estimated institute DAV INSTITUTE OF ENGINEERING AND TECHNOLOGY for giving this golden opportunity for working on this major project that is also the part of the University Curriculum.

We wish to express our heartfelt gratitude and thanks to our Head of Department Dr. Neeru Malhotra and our guide Mr. Vishav Kapoor for their valuable suggestions and indebted help to complete our project on time successfully.

We thank to our honorable Principal Dr. Manoj Kumar for his kind cooperation and for providing the department facilities like computer lab and internet. We are much thankful to our staff for their valuable suggestions and lab technicians for their cooperation.

AVNISH NAGPAL

AYUSH KUMAR

CHIRAG NARANG

TABLE OF CONTENTS

Chapter 1: Introduction to Project

1.1	Introduction to Project.....	1
1.2	Objective of the Project.....	1
1.3	Scope of the Project.....	2
1.4	Proposed System	2
1.5	Features of the Project	3

Chapter 2: Basics of Internet of Things

2.1	Introduction to Internet of Things	4
2.2	IoT in Health Care System.....	5
2.3	Discussion for Future Development	5

Chapter 3: Design & Requirements

3.1	Design.....	6
3.2	Software Requirements	7
3.3	Hardware Requirements....	9

Chapter 4: Implementation & Testing

4.1	Circuit diagram & Connections.....	17
4.1.1	Sensors Connection with Arduino.....	17
4.1.2	Sensors Connection NodeMCU	18
4.2	Configuring ThingSpeak to record Patient's Data online	19
4.3	Arduino Code.....	29
4.4	Testing	30

Chapter 5: Results & Discussion

5.1	Block Diagram & Connection Representation	31
5.2	Project Snapshots.....	32
5.3	Patient Monitoring through Thingspeak & Ubidot graph	32

Chapter 6: Summary of the Study, Conclusion and Recommendations

6.1	Summary of the Study	36
6.2	Conclusion.....	37
6.3	Recommendations on Future Work.....	37

CHAPTER: 1

INTRODUCTION TO PROJECT

1.1 INTRODUCTION TO PROJECT

Internet of Things (IoT) is now a reliable technological standard and a heavily researched field. Sensors are being used almost everywhere in the present time, from everyday products to industrial monitoring systems. The use of IoT and sensor-based intensive health care systems are increasing rapidly. IoT makes our life smarter, more efficient, and easier. Using a smartphone as the data computing platform, the prototype model provides user-friendly voice recognition and alert functionalities.

Internet of Things (IoT) based smart health monitoring system is a patient monitoring system in which a patient can be monitored 24 hours. In the present world, IoT is changing the infrastructure of technologies. By facilitating effortless interaction among various modules, IoT has enabled us to implement various complex systems such as smart home appliances, smart traffic control systems, smart office systems, smart environment, smart vehicles, and smart temperature control systems and so on in very little space. Health monitoring systems are one of the most notable applications of IoT. Many types of designs and patterns have already been implemented to monitor a patient's health condition through IoT.

1.2 OBJECTIVE OF THE PROJECT

The objective of patient monitoring is to have a quantitative assessment of the important physiological variables of the patients during critical periods of their biological functions. For diagnostic and research purposes, it is necessary to know their actual value or trend of change. With tons of new healthcare technology start-ups, IoT is rapidly revolutionizing the healthcare industry. In this project, we have designed the **IoT Based Patient Health Monitoring System using ESP8266 & Arduino**. The IoT platform used in this project is Thing Speak. Thing Speak is an open-source Internet of Things (IoT) application and API to store and retrieve data from things using the HTTP protocol over the Internet or via a Local Area Network. This IoT device could read the pulse rate and measure the surrounding temperature. It continuously monitors the pulse rate and surrounding temperature and updates them to an IoT platform.

The Arduino Sketch running over the device implements the various functionalities of the project like reading sensor data, converting them into strings, passing them to the IoT platform, and displaying measured pulse rate and temperature on character LCD.

1.3 SCOPE OF THE PROJECT

Patient health monitoring systems are useful due to the possibility of timely and efficient healthcare services. These systems are based on advanced wireless and wearable sensor technologies. The rapid growth in technology has remarkably enhanced the scope of remote health monitoring systems.

The scope of this study is the development and implementation of real-time monitoring system for remote patients using wireless technology. The developed system would inform the doctor in case of emergency through alarms; however, delay in alarms might occur due to weak signals of 3G networks in some remote areas. Though the delayed alarming time is still within the golden period of time it should be considered in future research. As wireless technology is emerging day by day, the use of latest wireless technology may overcome these issues which ultimately increases the applicability and usefulness of the proposed remote monitoring system. Furthermore, false alarms can be generated due to the battery issues of sensors and smartphone. The research can be extended to overcome these battery and false alarm limitations.

1.4 PROPOSED SYSTEM

Health monitoring is the major problem in today's world. Due to lack of proper health monitoring, patient suffer from serious health issues. There are lots of IoT devices now days to monitor the health of patient over internet. Health experts are also taking advantage of these smart devices to keep an eye on their patients. With tons of new healthcare technology start-ups, IoT is rapidly revolutionizing the healthcare industry.

Here in this project, we will make an **IoT based Health Monitoring System** which records the patient heartbeat rate and body temperature and also send an email/SMS alert whenever those readings goes beyond critical values. Pulse rate and body temperature readings are recorded over ThingSpeak and Google sheets so that patient health can be monitored from anywhere in the world over internet. A panic will also be attached so that patient can press it on emergency to send email/SMS to their relatives.

System health monitoring is a set of activities undertaken to maintain a system in operable condition and may be limited to an observation of current system states, with maintenance and repair being prompted by these observations. Many sensors are required to provide real-time, onboard structural integrity assessments for the integrated system health management (ISHM) system. The health monitoring system is employed as the method to measure the observation data. The experimental modal analysis is one of the effective health monitoring systems for the case where the modal parameters such as natural frequencies are adopted as the observations.

1.5 FEATURES OF THE PROJECT

- Speed up and extend the communication coverage to increase the freedom for enhance patient quality of life.
- It gives immediate information to the belonging one e.g., doctors.
- Easy to monitor in the case of emergency.
- It reduced the death percentages in accidents.



Fig 1.1 Features of the project

CHAPTER: 2

BASICS OF INTERNET OF THINGS

2.1 INTRODUCTION TO INTERNET OF THINGS

With tons of new healthcare technology start-ups, IoT is rapidly revolutionizing the healthcare industry. In this project, we have designed the **IoT Based Patient Health Monitoring System using ESP8266 & Arduino**. The IoT platform used in this project is ThingSpeak. ThingSpeak is an open-source Internet of Things (IoT) application and API to store and retrieve data from things using the HTTP protocol over the Internet or via a Local Area Network. This IoT device could read the pulse rate and measure the surrounding temperature. It continuously monitors the pulse rate and surrounding temperature and updates them to an IoT platform.

Block Diagram:

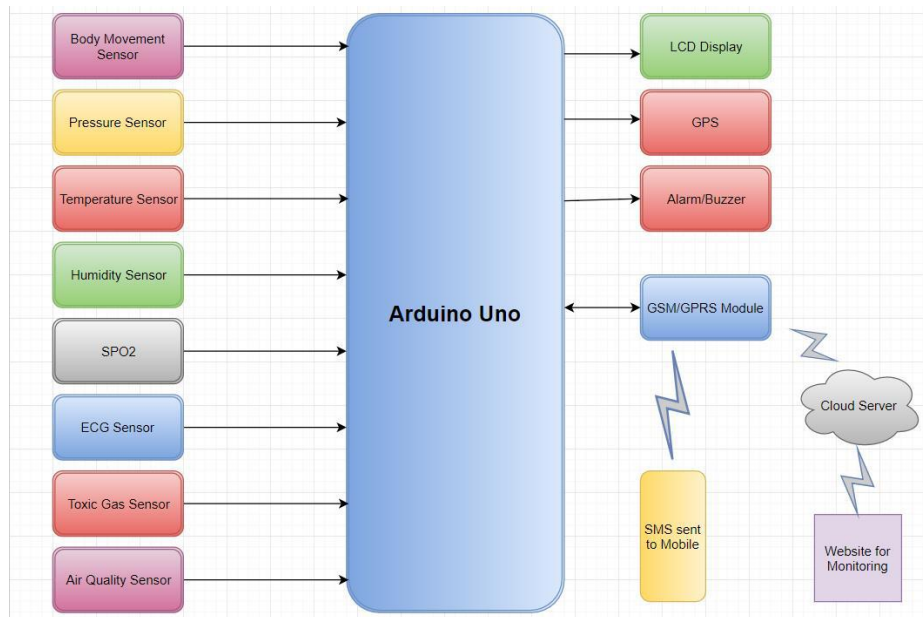


Fig 2.1 Block Diagram of Health Monitoring System

This is a simple block diagram that explains the **IoT Based Patient Health Monitoring System using ESP8266 & Arduino**. Pulse Sensor and LM35 Temperature Sensors measure BPM & Environmental Temperature respectively. The Arduino processes the code and displays it to 16*2 LCD Display. **ESP8266 Wi-Fi module** connects to Wi-Fi and sends the data to IoT device server. The IoT server used here is ThingSpeak. Finally, the data can be monitored from any part of the world by logging into the ThingSpeak channel.

2.2 IOT IN HEALTH CARE SYSTEM

IoT in the health monitoring system has given us a big advantage in the development of modern medical treatment. Due to advances in VLSI technology, the sensors have become smaller which has enabled the development of wearable solutions. Due to consistent internet connectivity, the devices are becoming more efficient and powerful. IoT based health monitoring devices monitor a patient 24/7. At any crucial moment, the devices generate necessary signals by analyzing statistical data. As IoT based devices are constantly connected to the internet, the patients can be remotely monitored, and necessary measures can be taken in case of an emergency. IoT based devices can thus provide both detection and emergency response services. There are significant differences between normal health monitoring systems and IoT based health monitoring systems. Incorporating IoT in health monitoring systems is a challenging task.

2.3 DISCUSSION FOR FUTURE DEVELOPMENT

The summary of this review is done based on some criteria such as feedback devices, major hardware components, uses, and cost-effectiveness. Different frameworks employ different feedback systems. It detects heart problems and body temperature. Some scholars discussed that an accelerometer, a voice sensor, and a microphone have been used for detecting the hyper-functional disorder and the system detected cardiovascular disease through ECG and heart rate sensor. The system discussed used a pulse oximeter, blood glucometer, and accelerometer for detecting chronic disease progression and used a Wi-Fi module for data transmitting. The Arduino Uno based system has been used to detect hypothermia. The high-cost device detected heart diseases. Smartphone, laptop, VGA display have been used as a feedback device. The system detects abnormalities in the heart. The respiration rate was monitored by using a respirator and accelerometer. The system used various gas sensors to provide the health monitoring facility. Though extensive works have been done to implement smart healthcare systems that are summarized in this paper, various sensors can be employed for health system monitoring for further development. The future developed systems can employ Wi-Fi and IR sensors to overcome the range limitations of Bluetooth devices. Smartphones can be used as a health monitoring system as it makes the interaction between multiple sensors very easy. Various machine learning algorithms can be used to make the systems more accurate. In microcontroller-based systems, raspberry pi can be used for easy presentation of the monitoring data on websites.

CHAPTER: 3

DESIGN & REQUIREMENTS

3.1 DESIGN

This is a simple block diagram that explains the **IoT Based Patient Health Monitoring System using ESP8266 & Arduino**. Pulse Sensor and LM35 Temperature Sensors measure BPM & Environmental Temperature, respectively. The Arduino processes the code and displays it to 16*2 LCD Display. **ESP8266 Wi-Fi module** connects to Wi-Fi and sends the data to IoT device server. The IoT server used here is ThingSpeak. Finally, the data can be monitored from any part of the world by logging into the ThingSpeak channel.

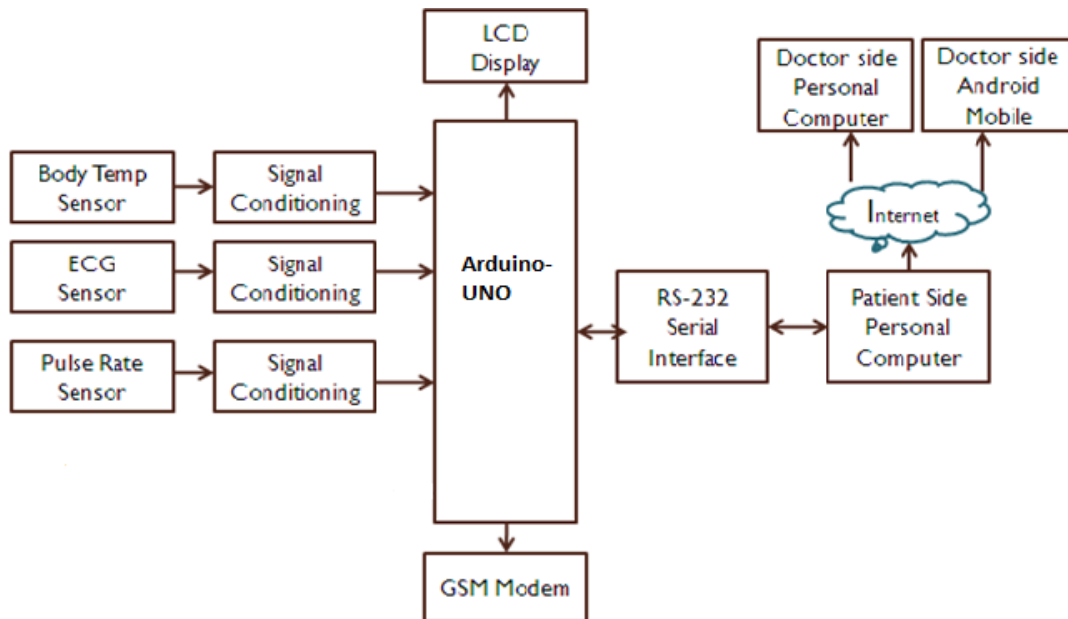


Fig 3.1 Project Block Diagram

3.2 SOFTWARE REQUIREMENTS

1) Arduino IDE: - Arduino is a popular tool for IoT product development as well as one of the most successful tools for STEM/STEAM education. Hundreds of thousands of designers, engineers, students, developers and makers around the world are using Arduino to innovate in music, games, toys, smart homes, farming, autonomous vehicles, and more.

Originally started as a research project by Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, and David Mellis at the Interaction Design Institute of Ivrea in the early 2000s, it builds upon the Processing project, a language for learning how to code within the context of the visual arts developed by Casey Reas and Ben Fry as well as a thesis project by Hernando Barragan about the Wiring board

2) ThingSpeak: - ThingSpeak is an IoT analytics platform service that allows you to aggregate, visualize, and analyze live data streams in the cloud. You can send data to ThingSpeak™ from your devices, create instant visualizations of live data, and send alerts using web services like Twitter® and Twilio®. With MATLAB® analytics inside ThingSpeak, you can write and execute MATLAB code to perform preprocessing, visualizations, and analyses. ThingSpeak enables engineers and scientists to prototype and build IoT systems without setting up servers or developing web software.

3) Ubidot: - Ubidots is an Internet of Things (IoT) data analytics and visualization company. We turn sensor data into information that matters for business-decisions, machine-to-machine interactions, educational research, and increase economization of global resources. Ubidots exists as an easy and affordable means to integrate the power of the IoT into your business or research.

Ubidots technology and engineering stack was developed to deliver a secure, white-glove experience for our users. Device friendly APIs (accessed over HTTP/MQTT/TCP/UDP protocols) provide a simple and secure connection for sending and retrieving data to and from our cloud service in real-time. Ubidots time-series backend services are performance optimized for IoT data storage, computation, and retrieval. Our application enablement platform supports interactive, real-time data visualization (widgets), and an IoT App Builder that allows developers to extend the platform with their own HTML/JS code for private customization when desired. Ubidots exists to empower your data from device to visualization.

4) IFTTT: - **If This Then That** (commonly known as **IFTTT**) is a service that allows a user to program a response to events in the world of various kinds. There is a long list of kinds of events to which IFTTT can respond, all detectable via the Internet. An example event is that Weather Underground reports rain is forecast for tomorrow. Another is that someone tagged the user in a photo on Facebook. There is similarly a long list of kinds of responses that are possible, all executable via the Internet. An example response would be to send an email to the user saying rain is forecast or copy the mentioned photo to the user's archive.

IFTTT has partnerships with hundreds of service providers that supply event notifications to IFTTT and execute commands that implement the responses, but some event and command interfaces are just public APIs. The programs, called applets, are simple and created graphically. User can create programs and otherwise control IFTTT with a web interface or iOS or Android application.

Overview

- Services (formerly known as channels) are the basic building blocks of IFTTT. They mainly describe a series of data from a certain web service such as YouTube or eBay. Services can also describe actions controlled with certain APIs like SMS. Sometimes, they can represent information in terms of weather or stocks. Each service has a particular set of triggers and actions.
- Triggers are the "this" part of an applet. They are the items that trigger the action. For example, from an feed, you can receive a notification based on a keyword or phrase.
- Actions are the "that" part of an applet. They are the output that results from the input of the trigger.
- Applets (formerly known as recipes) are the predicates made from Triggers and Actions. For example, if you like a picture on Instagram (trigger), an IFTTT app can send the photo to your account (action).
- Ingredients are basic data available from a trigger—from the email trigger, for example, subject, body, attachment, received date, and sender's address.

Usage examples

- IFTTT can automate web-application tasks, such as posting the same content on several social networks.
- Marketing professionals can use IFTTT to track mentions of companies.
- IFTTT also is used in home automation, real time patient health monitoring system for instance switching on a light when detecting motion in a room (with associated compliant devices).
- Here it is used for the message of panic and real time data in google sheets.

3.3 HARDWARE REQUIREMENTS

1) ARDUINO UNO: - Arduino Uno is a microcontroller board based on the ATmega328P (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator (CSTCE16M0V53-R0), a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.. You can tinker with your Uno without worrying too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again.

"Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform; for an extensive list of current, past or outdated boards see the Arduino index of boards.

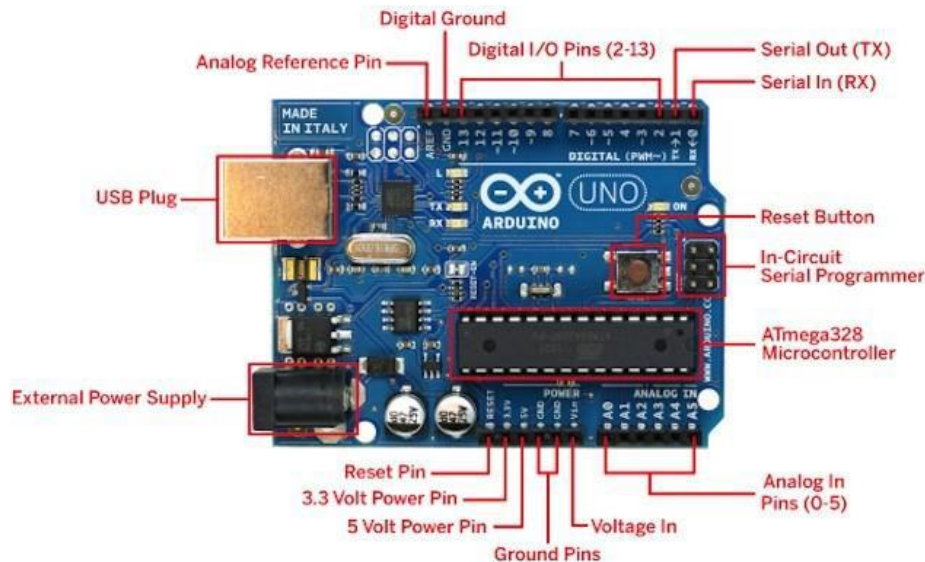


Fig 3.2 Arduino Uno

2) ESP2866: -The **ESP8266** is a very user-friendly and low-cost device to provide internet connectivity to your projects. The module can work both as an Access point (can create hotspot) and as a station (can connect to Wi-Fi), hence it can easily fetch data and upload it to the internet making the Internet of Things as easy as possible. It can also fetch data from the internet using API's hence your project could access any information that is available on the internet, thus making

it smarter. Another exciting feature of this module is that it can be programmed using the Arduino IDE which makes it a lot more user friendly.

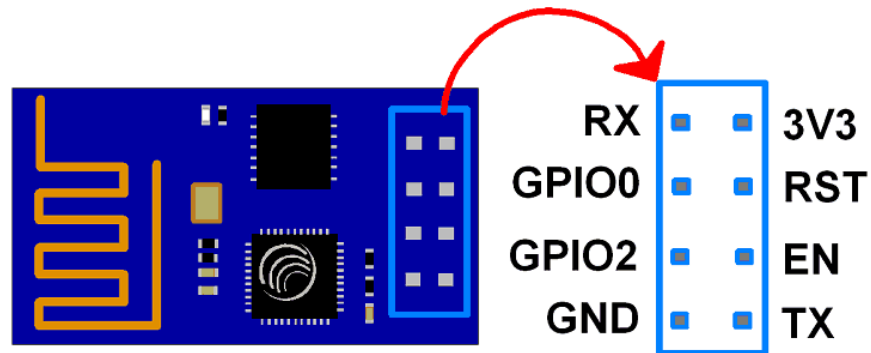


Fig 3.3 ESP8266 Module

The ESP8266 module works with 3.3V only, anything more than 3.7V would kill the module hence be cautions with your circuits. Here is its pins description.

Pin 1: Ground: Connected to the ground of the circuit

Pin 2: Tx/GPIO – 1: Connected to Rx pin of programmer/uC to upload program

Pin 3: GPIO – 2: General purpose Input/output pin

Pin 4 : CH_EN: Chip Enable/Active high

Pin 5: Flash/GPIO – 0: General purpose Input/output pin

Pin 6 : Reset: Resets the module

Pin 7: RX/GPIO – 3: General purpose Input/output pin

Pin 8: Vcc: Connect to +3.3V only.

3) NODE MCU: -NodeMCU is an open-source firmware and development kit that helps you to prototype or build IoT product. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module. The firmware uses the Lua scripting language. It is based on the eLua project and built on the Espressif Non-OS SDK for ESP8266.

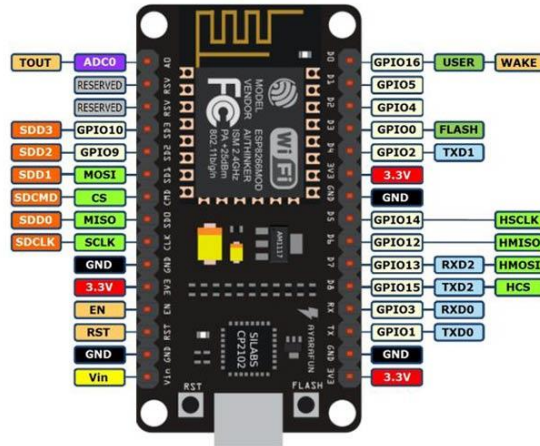


Fig 3.4 Node MCU

NodeMCU was created shortly after the ESP8266 came out. On December 30, 2013, Espressif Systems began production of the ESP8266. The ESP8266 is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core, widely used in IoT applications (see related projects). NodeMCU started on 13 Oct 2014, when Hong committed the first file of nodemcu-firmware to GitHub. Two months later, the project expanded to include an open-hardware platform when developer Huang R committed the gerber file of an ESP8266 board, named devkit v0.9.^[12] Later that month, Tuan PM ported MQTT client library from Contiki to the ESP8266 SoC platform,^[13] and committed to NodeMCU project, then NodeMCU was able to support the MQTT IoT protocol, using Lua to access the MQTT broker. Another important update was made on 30 Jan 2015, when Devsaurus ported the u8glib to NodeMCU project, enabling NodeMCU to easily drive LCD, Screen, OLED, even VGA displays.

In summer 2015 the creators abandoned the firmware project and a group of independent contributors took over. By summer 2016 the NodeMCU included more than 40 different modules. Due to resource constraints users need to select the modules relevant for their project and build a firmware tailored to their needs.

4) LCD 16x2 Display Module: - Liquid Crystal Display (LCD) is widely used in various electronics' applications. It is commonly used in various systems to show different status and parameters. LCD16x2 has 2 lines with 16 characters in each line. Each character is made up of 5x8 (column x row) pixel matrix.



Fig 3.5 16*2 LCD Display

5) Pulse Sensor: - The Pulse Sensor is a plug-and-play heart-rate sensor for Arduino. It can be used by students, artists, athletes, makers, and game & mobile developers who want to easily incorporate live heart-rate data into their projects. The essence is an integrated optical amplifying circuit and noise eliminating circuit sensor. Clip the Pulse Sensor to your earlobe or fingertip and plug it into your Arduino, you can be ready to read heart rate. Also, it has an Arduino demo code that makes it easy to use.

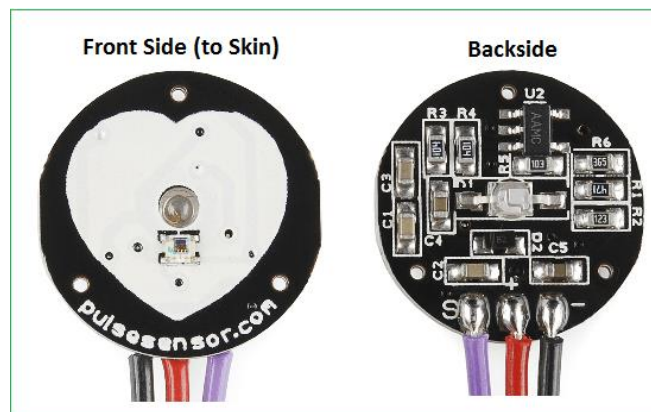


Fig 3.6 Pulse Sensor

The pulse sensor has three pins: VCC, GND & Analog Pin.

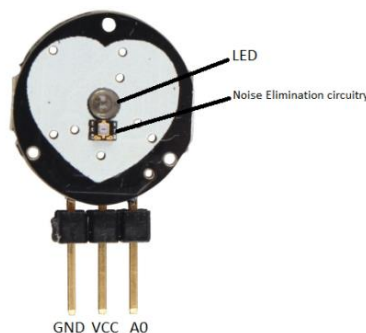


Fig 3.7 Pulse Sensor Pins

There is also a LED in the centre of this sensor module which helps in detecting the heartbeat. Below the LED, there is a noise elimination circuitry that is supposed to keep away the noise from affecting the readings.

6) LM35 Temperature Sensor: - The LM35 series are precision integrated-circuit temperature devices with an output voltage linearly proportional to the Centigrade temperature. The LM35 device has an advantage over linear temperature sensors calibrated in Kelvin, as the user is not required to subtract a large constant voltage from the output to obtain convenient Centigrade scaling. The LM35 device does not require any external calibration or trimming to provide typical accuracies of $\pm 1/4^{\circ}\text{C}$ at room temperature and $\pm 3/4^{\circ}$ Cover a full -55°C to 150°C temperature range.

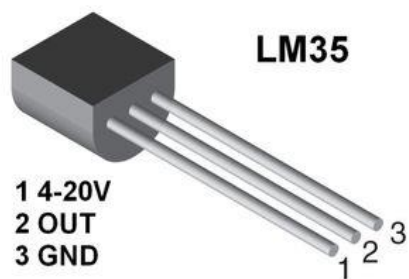


Fig 3.8 LM35 Temperature Sensor

7) Max30100 pulse Oximeter Arduino: - The MAX30100 is an integrated pulse Oximetry and heartrate monitor sensor solution. It combines two LEDs, a photodetector, optimized optics, and low-noise analog signal processing to detect pulse Oximetry and heart-rate signals.

There is another type of the Max30100 pulse Oximeter as you can see in the picture below,



Fig 3.9 Max30100 pulse Oximeter Sensor

This type of the Oximeter has the sensors and electronic components all on the same side, moreover there are many complaints about this type of the sensor. I recommend you should purchase the one I am using in this tutorial. Because, while using this sensor you don't need to remove the resistors or solder any extra wires.

SpO2 Subsystem

The SpO2 subsystem in the MAX30100 is composed of ambient light cancellation (ALC), 16-bit sigma delta ADC, and proprietary discrete time filter.

The SpO2 ADC is a continuous time oversampling sigma delta converter with up to 16-bit resolution. The ADC out-put data rate can be programmed from 50Hz to 1kHz. The MAX30100 includes a proprietary discrete time filter to reject 50Hz/60Hz interference and low-frequency residual ambient noise.

System Block Diagram

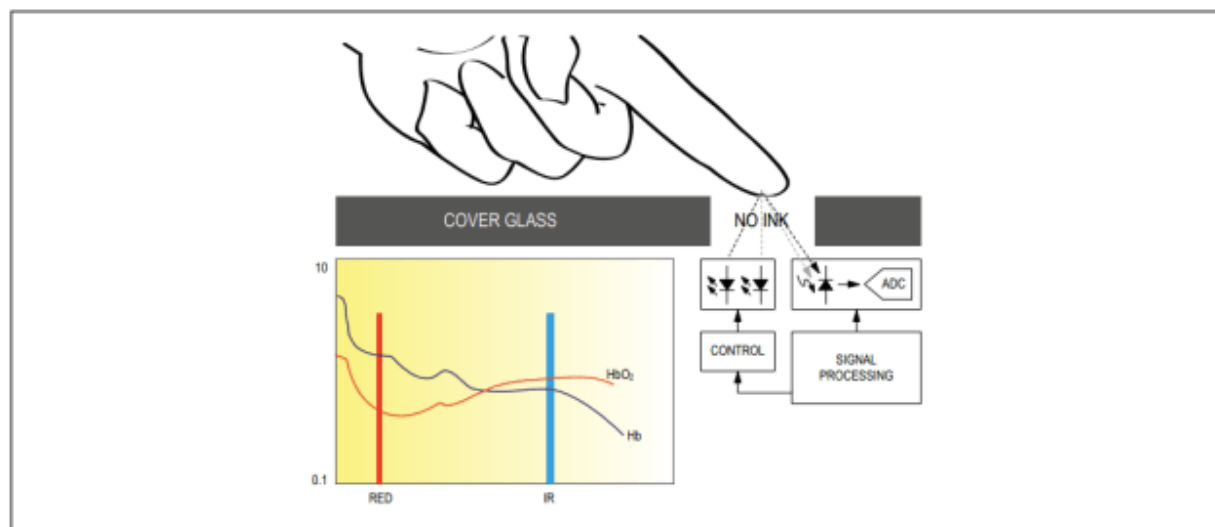


Fig 3.10 Working of Oximeter Sensor

As per the system block diagram, which is available in the datasheet, it clearly shows that there should be small distance between the sensor and finger.

8) DHT11 Humidity Sensor: -The **DHT11** is a commonly used **Temperature and humidity sensor**. The sensor comes with a dedicated NTC to measure temperature and an 8-bit microcontroller to output the values of temperature and humidity as serial data. The sensor is also factory calibrated and hence easy to interface with other microcontrollers.

The sensor can measure temperature from 0°C to 50°C and humidity from 20% to 90% with an accuracy of $\pm 1^\circ\text{C}$ and $\pm 1\%$. So, if you are looking to measure in this range then this sensor might be the right choice for you.

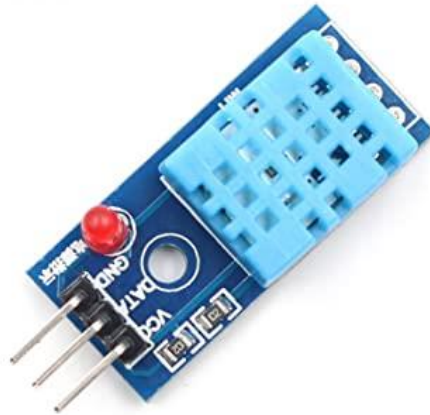


Fig 3.11 DHT11 Humidity Sensor

Applications:

- Measure temperature and humidity
- Local Weather station
- Automatic climate control
- Environment monitoring

9) AD8232 ECG Sensor: - The AD8232 Single Lead Heart Rate Monitor is used to measure the electrical activity of the heart. This electrical activity can be charted as an ECG or Electrocardiogram and output as an analog reading.

ECGs can be extremely noisy, the AD8232 Single Lead Heart Rate Monitor acts as an op amp to help obtain a clear signal from the PR and QT Intervals easily. The AD8232 is an integrated signal conditioning block for ECG and other biopotential measurement applications. It is designed to extract, amplify, and filter small biopotential signals in the presence of noisy conditions, such as those created by motion or remote electrode placement.

The AD8232 Heart Rate Monitor breaks out nine connections from the IC that you can solder pins, wires, or other connectors to. SDN, LO+, LO-, OUTPUT, 3.3V, GND provide essential pins for operating this monitor with an Arduino or other development board. Also provided on this board are RA (Right Arm), LA (Left Arm), and RL (Right Leg) pins to attach and use your own custom sensors. Additionally, there is an LED indicator light that will pulsate to the rhythm of a heartbeat.

Features:

- Operating Voltage - 3.3V
- Analog Output
- Leads-Off Detection
- 3.5mm Jack for Biomedical Pad Connection

Package Includes:

- 1 x ECG Sensor
- 1 x Professional ECG Cable
- 3 x Disposable Surface Electrode

10) Push Button: - Simple Push Button is used for Panic Button Function. This button will send an interrupt which by the use of IFTTT, sends mail to doctor and its family members.

CHAPTER 4

IMPLEMENTATION AND TESTING

4.1 Circuit Diagram & Connections:

4.1.1 Sensors Connections with Arduino

For designing IoT Based Patient Health Monitoring System using ESP8266 & Arduino, assemble the circuit as shown in the figure below.

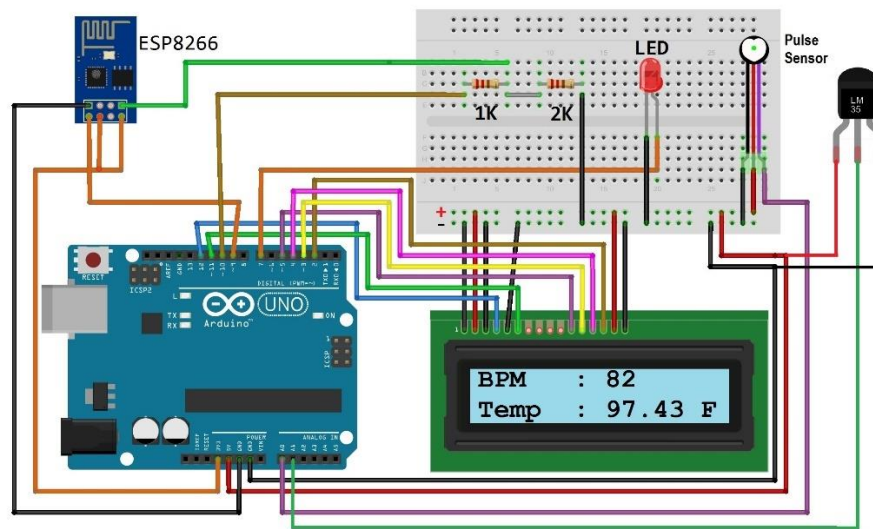


Fig 4.1 Sensors Connection with Arduino

1. Connect Pulse Sensor output pin to A0 of Arduino and other two pins to VCC & GND.
2. Connect LM35 Temperature Sensor output pin to A1 of Arduino and other two pins to VCC & GND.
3. Connect the LED to Digital Pin 7 of Arduino via a 220-ohm resistor.
4. Connect Pin 1,3,5,16 of LCD to GND.
5. Connect Pin 2,15 of LCD to VCC.
6. Connect Pin 4,6,11,12,13,14 of LCD to Digital Pin12,11,5,4,3,2 of Arduino.
7. The RX pin of ESP8266 works on 3.3V and it will not communicate with the Arduino when we will connect it directly to the Arduino. So, we will have to make a voltage divider

for it which will convert the 5V into 3.3V. This can be done by connecting the 2.2K & 1K resistor. Thus, the RX pin of the ESP8266 is connected to pin 10 of Arduino through the resistors.

8. Connect the TX pin of the ESP8266 to pin 9 of the Arduino.
9. Connect Push Button to Digital Pin of the Arduino.

4.1.2 Sensors Connections with NodeMCU

1. To interface **AD8232 ECG Sensor with ESP32 IOT Chip**, follow the circuit diagram above. Supply the AD8232 with 3.3V from ESP32 and connect GND to GND. The output pin of AD8232 is an analog signal and is connected to VP pin of ESP32. Similarly, LO+ and LO- of AD8232 is connected to D2 & D3 of ESP32.

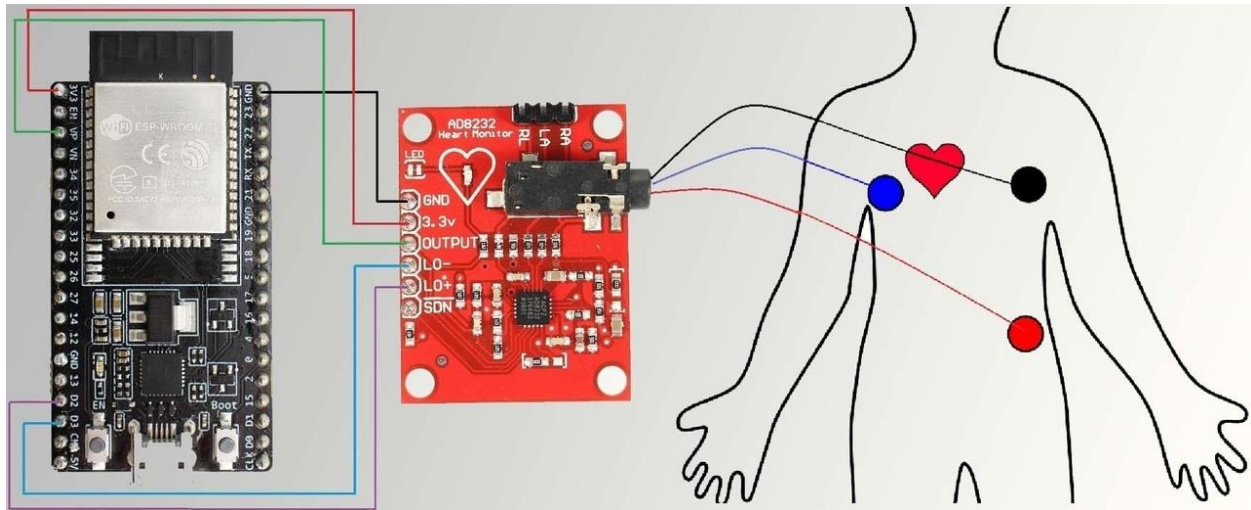


Fig 4.2 AD8232 ECG Sensor Connection with NodeMCU

2. The complete **circuit diagram for MAX30100 with ESP32** is given below. The pin 21 and 22 of the ESP32 devkit C is connected with pulse oximeter sensor MAX30100 with the SDA and SCL pins. The Oximeter is also powered by the 5V pin on the ESP32 development board.

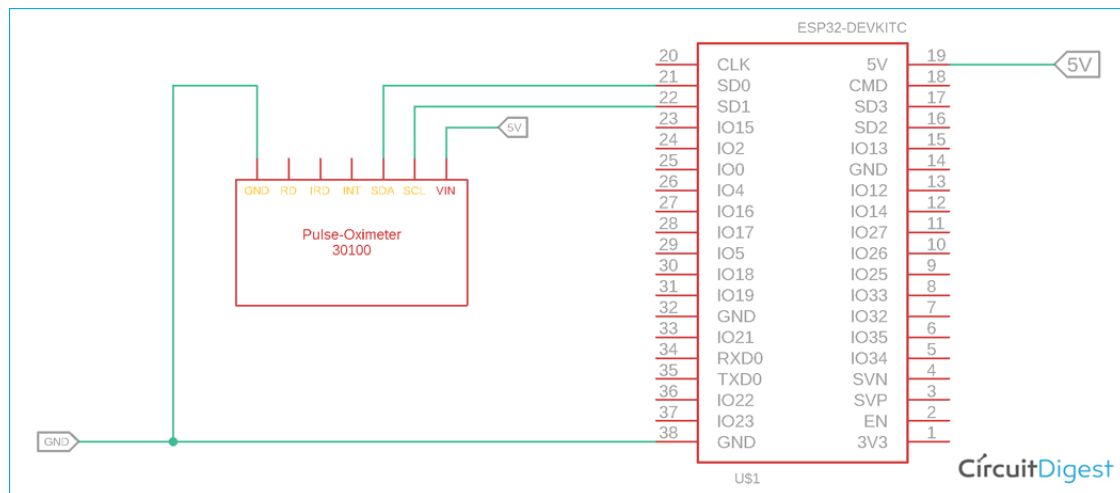


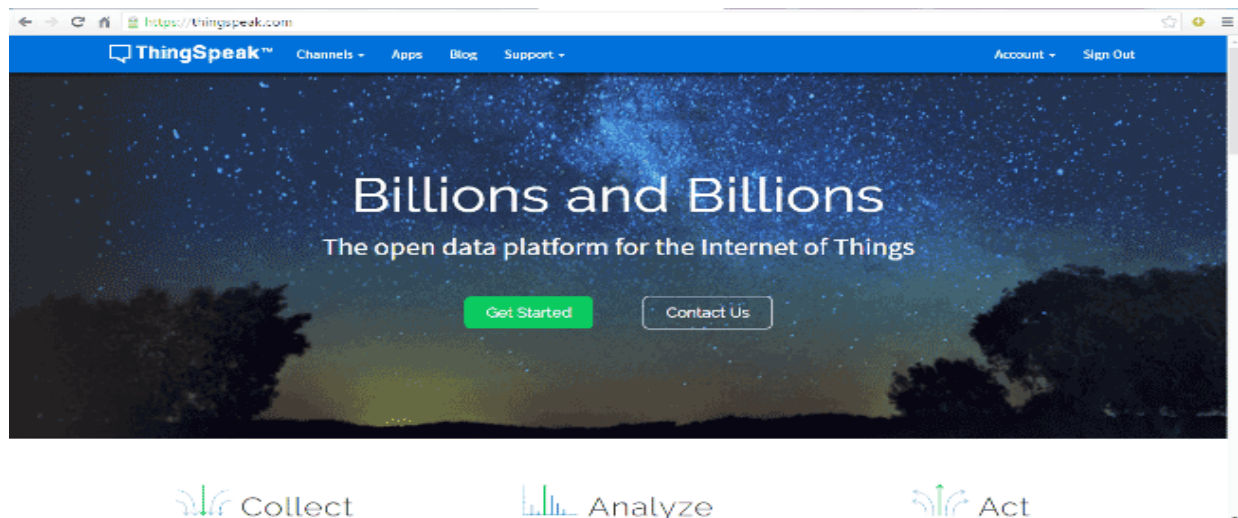
Fig 4.3 Pulse Oximeter Sensor Connection with NodeMCU

4.2 Configuring ThingSpeak to record Patient Data online

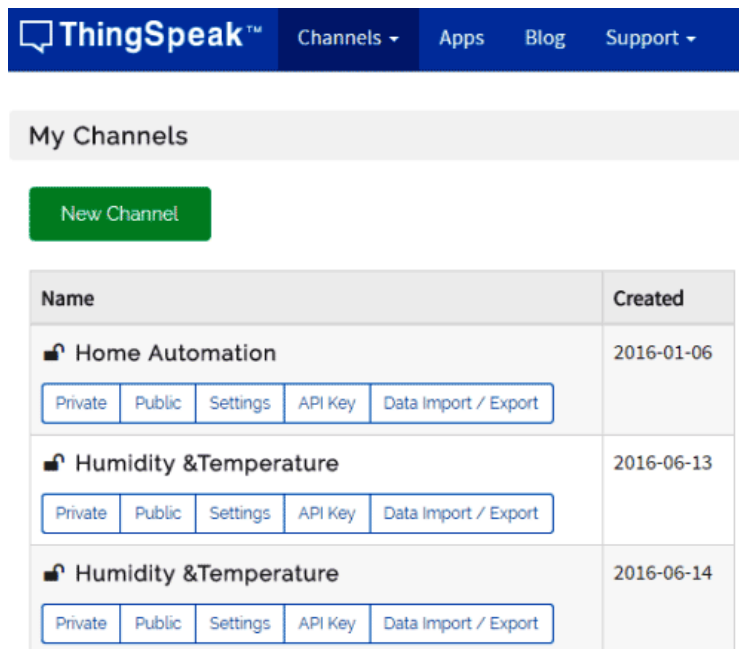
ThingSpeak provides very good tool for IoT based projects. By using ThingSpeak site, we can monitor our data and control our system over the Internet, using the Channels and webpages provided by ThingSpeak. ThingSpeak ‘**Collects**’ the data from the sensors, ‘**Analyze and Visualize**’ the data and ‘**Acts**’ by triggering a reaction. We have previously used ThingSpeak in Weather station project using Raspberry Pi and using Arduino, check them to learn more about ThingSpeak. Here we are briefly explaining to use ThingSpeak for this **IoT Patient Monitoring Project**.

We will use **ThingSpeak** to monitor patient heartbeat and temperature online using internet. We will also use **IFTTT** platform to connect ThingSpeak to email/message service so that alert message can be sent whenever the patient is in critical state.

Step 1:- First of all, user needs to Create a Account on ThingSpeak.com, then **Sign In** and **click on Get Started**.



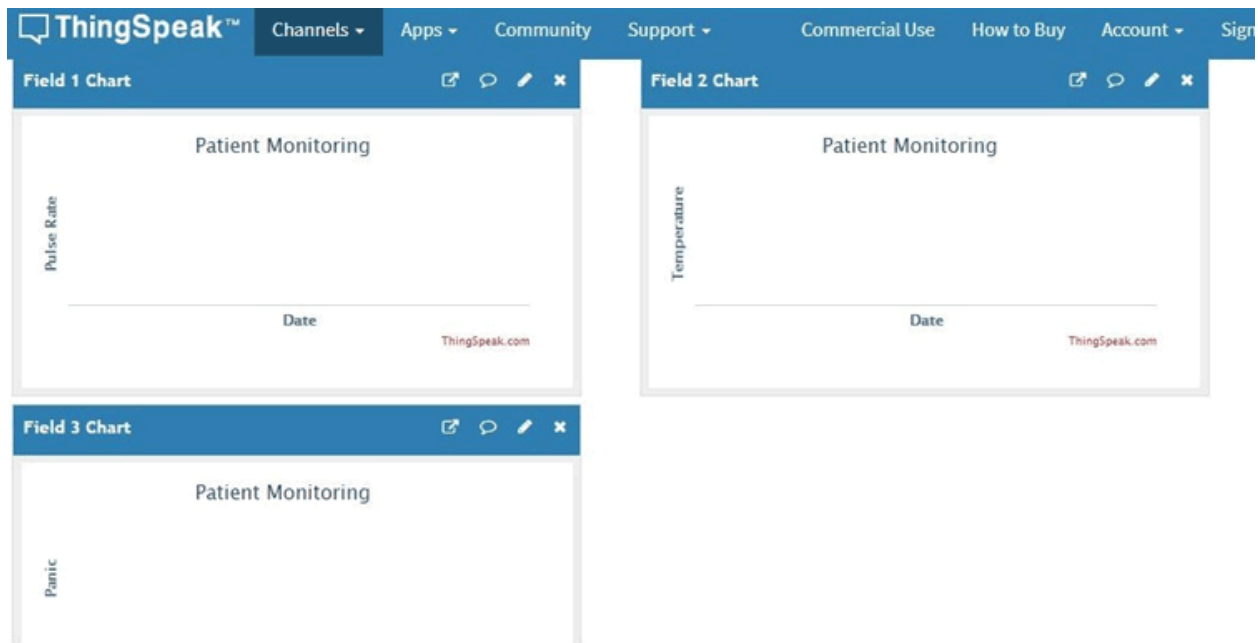
Step 2:- Now go to the ‘Channels’ menu and click on **New Channel** option on the same page for further process.



Step 3:- Now you will see a form for **creating the channel**, fill in the Name and Description as per your choice. Then fill ‘Pulse Rate’, ‘Temperature’ and ‘Panic’ in Field 1, Field 2 and Field 3 labels, tick the checkboxes for the Fields. Also tick the check box for ‘Make Public’ option below in the form and finally Save the Channel. Now your new channel has been created.

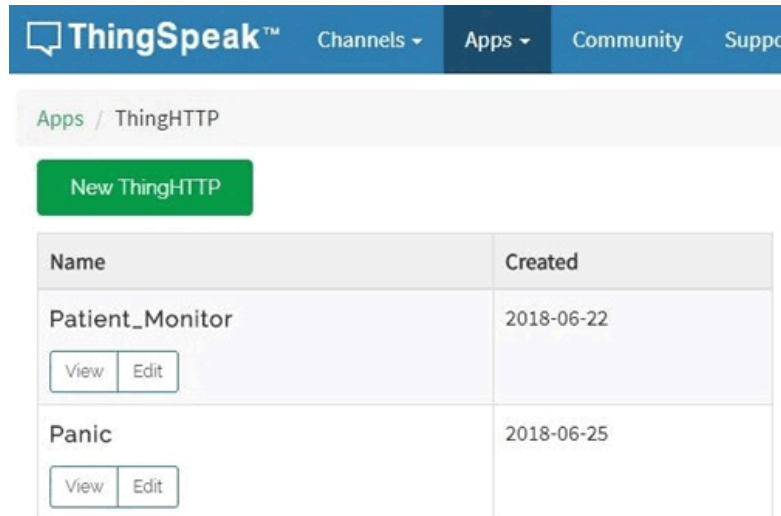
Name	<input type="text" value="Patient Monitoring"/>	
Description	<input type="text"/>	
Field 1	<input type="text" value="Pulse Rate"/>	<input checked="" type="checkbox"/>
Field 2	<input type="text" value="Temperature"/>	<input checked="" type="checkbox"/>
Field 3	<input type="text" value="Panic"/>	<input checked="" type="checkbox"/>
Field 4	<input type="text"/>	<input type="checkbox"/>
Field 5	<input type="text"/>	<input type="checkbox"/>
Field 6	<input type="text"/>	<input type="checkbox"/>
Field 7	<input type="text"/>	<input type="checkbox"/>

Step 4:- You will see three charts as shown below. Note the **Write API key**, we will use this key in our code.



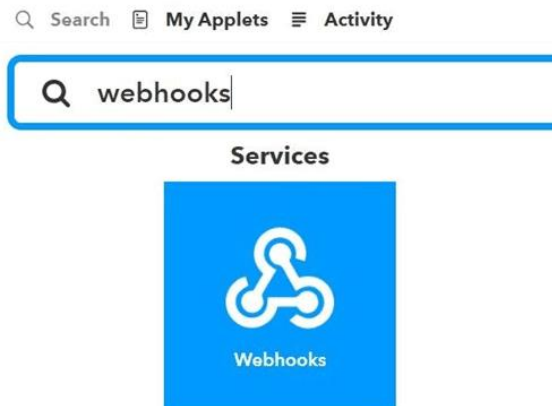
Step 5:- Now, we will use **ThingHTTP** app of the server to trigger the IFTTT applet for data entry to Google sheets and send email/sms. ThingHTTP enables communication among devices, websites, and web services without having to implement the protocol on the device level. You can specify actions in ThingHTTP, which you want to trigger using other ThingSpeak apps such as **React**.

To make New ThingHTTP, we will need URL for triggering which we will get from IFTTT.

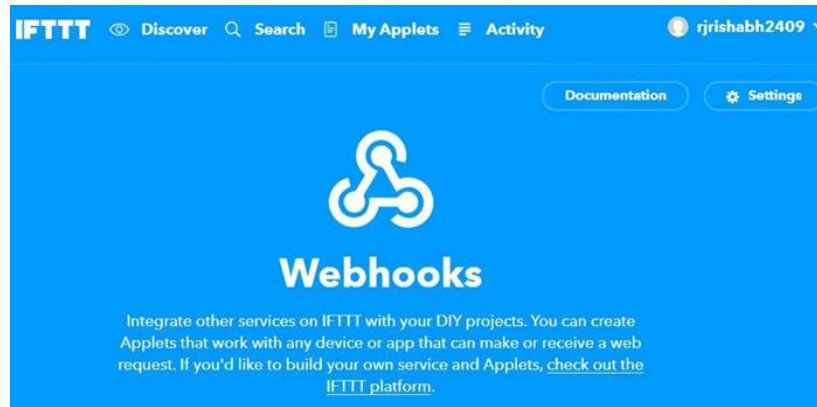


Configuring IFTTT for triggering Mail/SMS based on ThingSpeak Values

Step 1:- Login to IFTTT and search for **Webhooks** and click on it.



Step 2:- Click on **Documentation**.

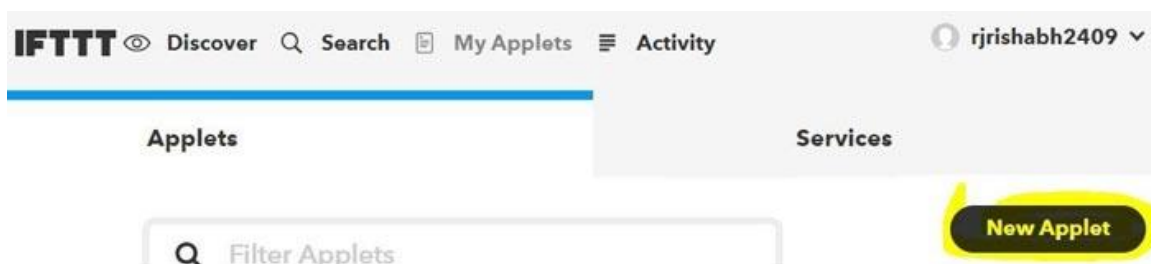


Step 3:- Type “Patient_Info” in the event box and copy the URL. We will use this URL in ThingHTTP.

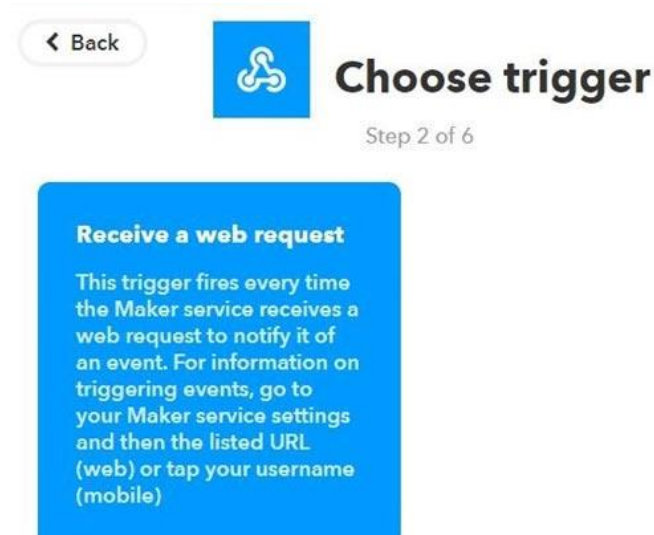


Now let us make Applet to link ThingHTTP to Google sheet and to send email/SMS. After that we will jump to complete our ThingHTTP.

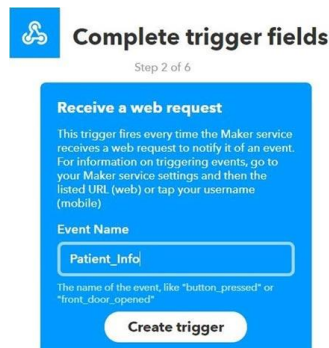
Step 4:- Click on *New Applet* in *My Applet* option.



Step 5:- Click on “+this” and search for Webhooks and click on it. Choose trigger as “Receive a web request”.



Step 6:- Type the Event Name which is same as you write in the event box in webhooks URL. Click on Create Trigger.



Step 7:- Click on “+that” and search for Google Sheets and click on it.

Click on **Add row to spreadsheet**.



Step 8:- Give any name to your sheet. In formatted row box, you have date and time, event name, BPM value and body temperature which will be written as shown.



Add row to spreadsheet

This action will add a single row to the bottom of the first worksheet of a spreadsheet you specify. Note: a new spreadsheet is created after 2000 rows.

Spreadsheet name

IFTTT_Maker_Webhooks_Events

Will create a new spreadsheet if one with this title doesn't exist **Add ingredient**

Formatted row

{{OccurredAt}} ||| {{EventName}} |||
{{Value1}} |||{{Value2}}

Use "|||" to separate cells **Add ingredient**

Step 9:- Review your applet and click on finish.



Review and finish

Step 6 of 6



If maker Event "Patient_Info",
then Add row to Rishabh
Jain's Google Drive
spreadsheet

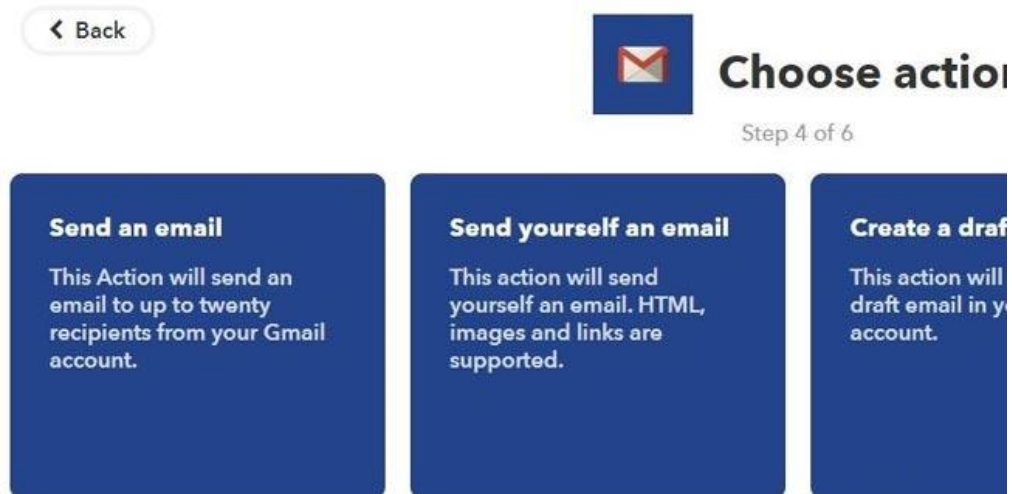
86/140

by rjishabh2409

works with 

In the same way, **we have to make applet for sending email when Panic event is occurred.**

So again click on “+this” and select *Webhooks*, then in event name enter “Panic”. In “+that” search for Gmail and click on it. Now, click on Send an email.



Type the email addresses on which you wish to receive email when there is a panic to the patient.

 **Complete action fields**

Step 5 of 6

Send an email

This Action will send an email to up to twenty recipients from your Gmail account.

To address

Accepts up to twenty email addresses, each separated with a space or comma **Add ingredient**

CC address

Accepts up to twenty email addresses, each separated **Add ingredient**

Type the body content you wish to send in the email and click on create action. Review it and finish.

Subject

The event named " **EventName** " occurred on the Maker Webhooks service

Add ingredient

Body

Patient is in problem!!!
When: **OccurredAt**

Some HTML ok

Add ingredient

Attachment URL

URL to include as an attachment

Add ingredient

Create action

We have made our applets to perform the tasks. Now, come back to *Thingspeak*->*Apps*->*ThingHTTP*.

ThingHTTP for connecting ThingSpeak with IFTTT

Step 1:- Click on *New ThingHTTP*. Give any name and Paste the URL that you copied from the webhooks documentation. Fill Remaining information as shown below.

ThingSpeak™ Channels Apps Community Support

Name Patient_Monitor

API Key [Redacted]

URL https://maker.ifttt.com/trigger/Patient_info/with/key/hWzoxX

HTTP Auth Username

HTTP Auth Password

Method POST

Content Type application/json

HTTP Version 1.1

Host

Headers Name

In Body, we have to write the information that we want to send to the IFTTT applet. We are sending patient Pulse reading and temperature. So the format is

```
{ "value1" : "%%channel_channelID_field_fieldNumber%%","value2" : "%%channel_channelID_field_fieldNumber%%" }
```

Value

remove header

add new header

Body

["value1": "%%channel_2395_field_1%%", "value2": "%%channel_2395_field_2%%"]

Parse String

Save ThingHTTP

After filling these informations, click on *Save ThingHTTP*.

In the same manner, we have to make ThingHTTP for “Panic”. Follow the same steps.

In URL, write **Panic** in place of **Patient_Info**. Body remains empty and All other information are same as in previous ThingHTTP. Save it.

Now, we have to make *React* to trigger the URL.

React works with ThingHTTP app to perform actions when channel data meets a certain condition.

To make React, click on Apps -> React. Click on **New React**.

Step 2:- Give name to your React. Condition type as Numeric and Test Frequency as on Data Insertion.

Choose the Condition on which you want to trigger the URL. Select your channel from the *If Channel* drop down menu. Choose field 1 i.e Pulse rate and make condition of *greater than* any value. I have used 60. As shown

The screenshot shows the 'React' configuration interface in ThingSpeak. At the top, there's a navigation bar with 'Channels', 'Apps', 'Community', and 'Support'. Below it, a breadcrumb trail reads 'Apps / React / Pulse rate / Edit'. The main form has the following fields:

- React Name:** A text input containing 'Pulse rate'.
- Condition Type:** A dropdown menu set to 'Numeric'.
- Test Frequency:** A dropdown menu set to 'On Data Insertion'.
- Condition:** A section with three sub-fields:
 - If channel:** A dropdown menu set to 'Patient Monitoring (523997)'.
 - field:** A dropdown menu set to '1 (Pulse Rate)'.
 - Operator:** A dropdown menu set to 'is greater than'.
 - Value:** A text input containing '60'.

Choose **ThingHTTP** from Action drop down menu and select the **ThingHTTP**.

Select “Run action each time condition is met” and click on *Save React*.

This screenshot shows the 'Action' configuration section. It includes:

- Action:** A dropdown menu set to 'ThingHTTP'.
- then perform ThingHTTP:** A dropdown menu set to 'Patient_Monitor'.
- Options:** Two radio button options:
 - ☐ Run action only the first time the condition is met
 - ☒ Run action each time condition is met
- Save React:** A green button at the bottom.

In the same way, make React for the **Panic** as shown. Select “Run action each time condition is met” and click on Save React.

4.3 Arduino Code

The code of this project is uploaded on Google Drive.

Code Folder Link: -

https://drive.google.com/drive/folders/1Z02iutvD_EFvdWDSxYbtUzqtxECCKYJa?usp=sharing

4.4 Testing

Performance of the system can be determined based on the system/application responsiveness under all kinds of load. Performance testing in IoT framework is little different than traditional performance testing. IoT devices generates a lot of data which is saved in server and analyzed for immediate decisions. Hence IoT system must be built for high performance and scalability. And to measure these two key attributes, it is important to understand the business value for which it is build i.e., in our case patient health data. Hence it is necessary to simulate real world models, network conditions etc.

- Debug your code before burning into Arduino Uno & NodeMCU.
- Include all libraries which are using in the code.
- Always remove all connections from the board before burning the code into NodeMCU.
- Make connections of the module and sensors correctly and properly.
- Always check that your ESP8266 nodeMCU WI-FI module should connected to internet.

CHAPTER 5

RESULTS AND DISCUSSION

5.1 BLOCK DIAGRAM AND CONNECTION REPRESENTATION

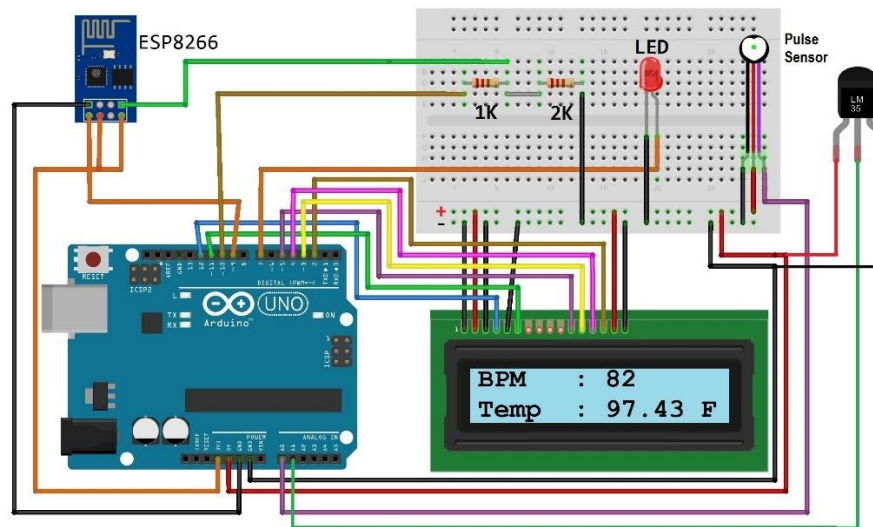
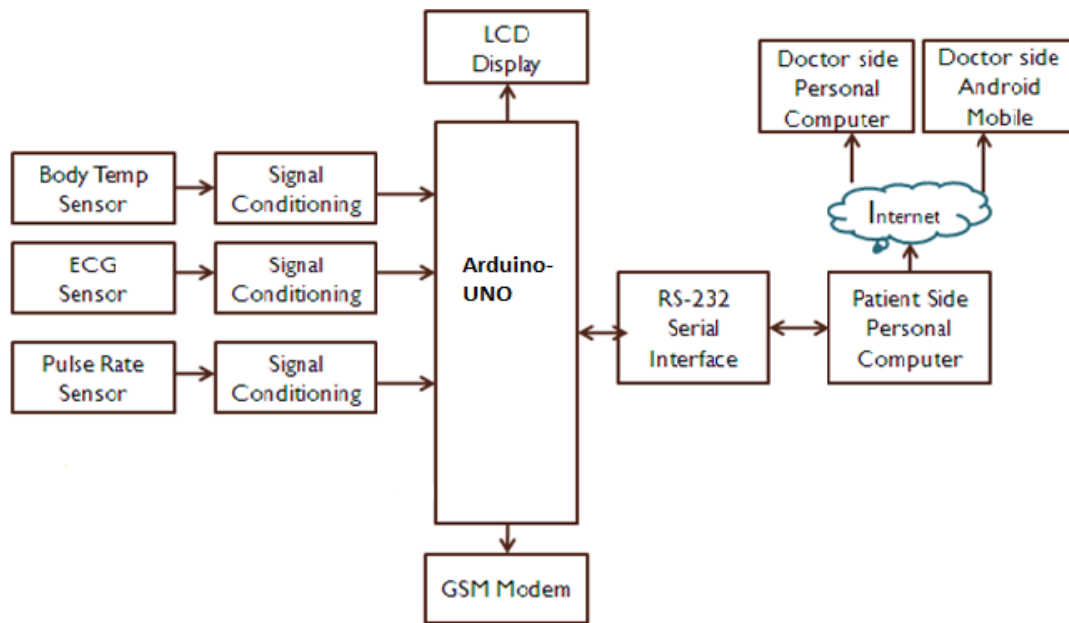


Fig 4.1 Block Diagram & Connections of the project.

5.2 PROJECT SNAPSHOTS

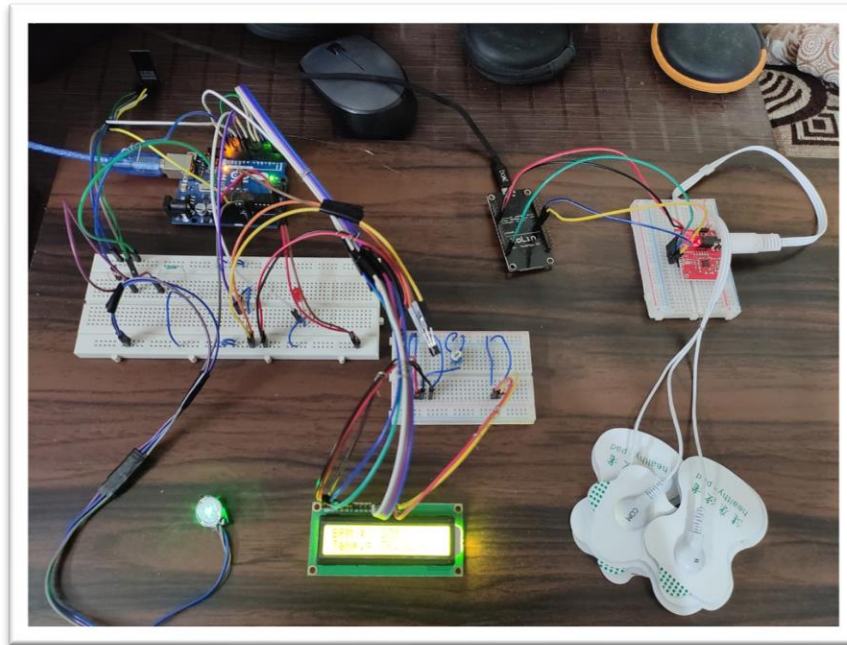
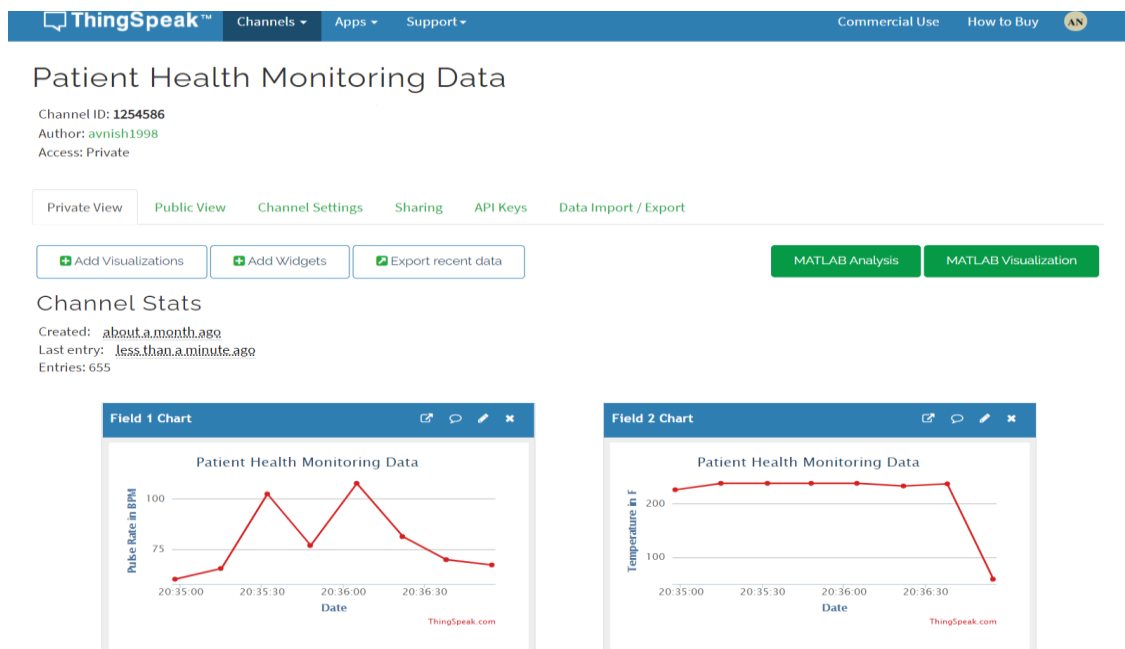


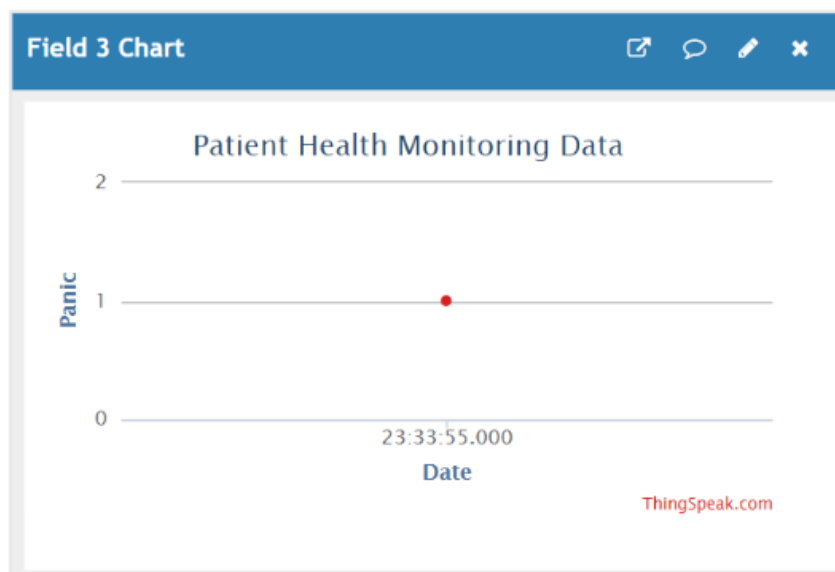
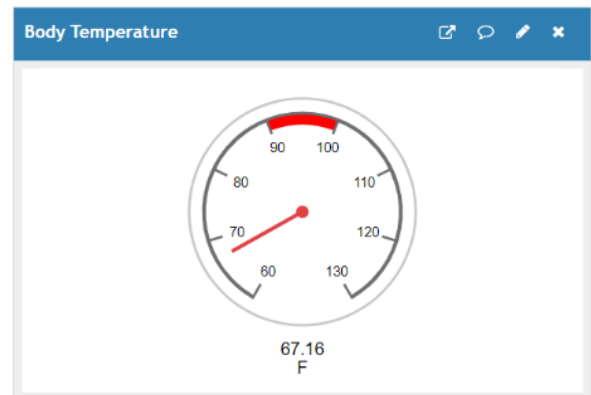
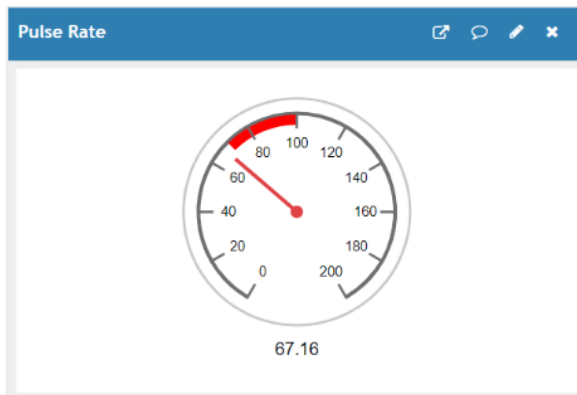
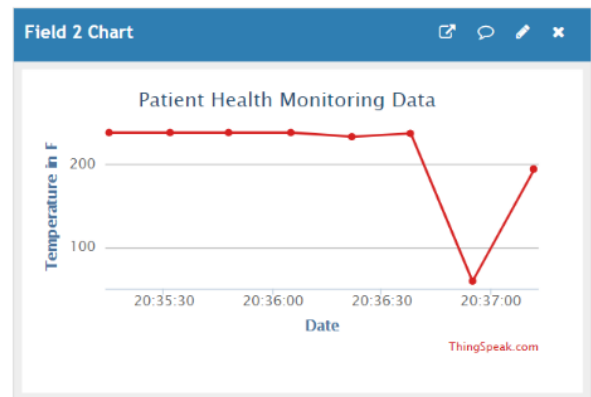
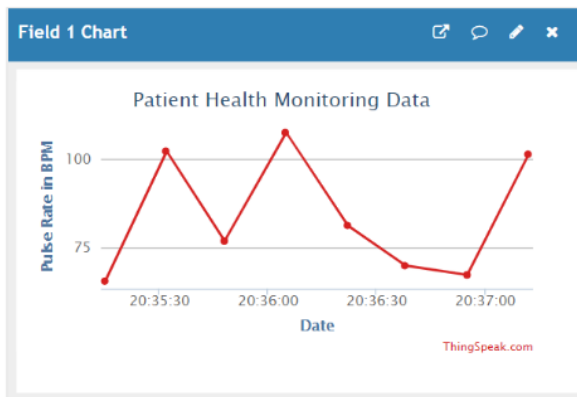
Fig 4.2 a) Completed Patient Health Monitoring Project.

5.3 PATIENT MONITORING THROUGH THINGSPEAK & UBIDOT GRAPH

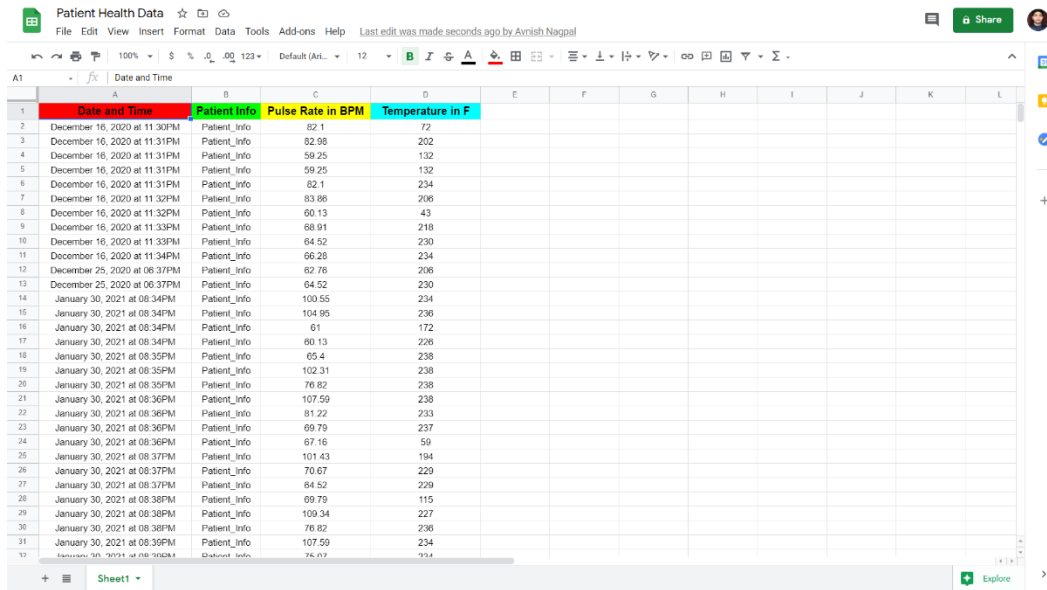
1) ThingSpeak IOT Platform showing Patient's Health Stats over the Internet.



- Below Figures gives parameters that is temperature and pulse rate is shown online on IOT platform.



2) Patient's Data is saving in Google Sheets with Time Stamps



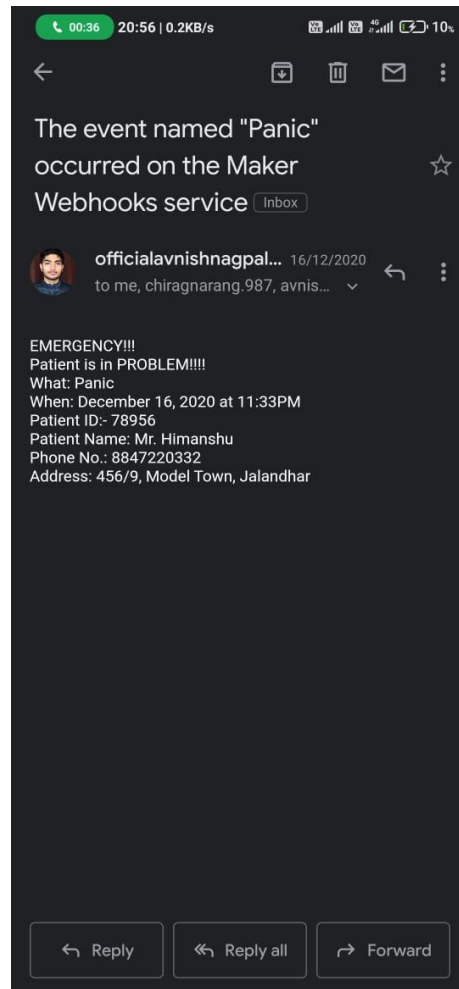
The screenshot shows a Google Sheet titled "Patient Health Data" with the following data:

	A	B	C	D	E	F	G	H	I	J	K	L
1	Date and Time	Patient_Info	Pulse Rate in BPM	Temperature in F								
2	December 16, 2020 at 11:30PM	Patient_Info	82.1	72								
3	December 16, 2020 at 11:31PM	Patient_Info	82.98	202								
4	December 16, 2020 at 11:31PM	Patient_Info	59.25	132								
5	December 16, 2020 at 11:31PM	Patient_Info	59.25	132								
6	December 16, 2020 at 11:31PM	Patient_Info	82.1	234								
7	December 16, 2020 at 11:32PM	Patient_Info	83.86	206								
8	December 16, 2020 at 11:32PM	Patient_Info	60.13	43								
9	December 16, 2020 at 11:33PM	Patient_Info	88.91	218								
10	December 16, 2020 at 11:33PM	Patient_Info	64.52	230								
11	December 16, 2020 at 11:34PM	Patient_Info	66.28	234								
12	December 25, 2020 at 06:37PM	Patient_Info	62.76	206								
13	December 25, 2020 at 06:37PM	Patient_Info	64.52	230								
14	January 30, 2021 at 08:34PM	Patient_Info	100.55	234								
15	January 30, 2021 at 08:34PM	Patient_Info	104.95	236								
16	January 30, 2021 at 08:34PM	Patient_Info	61	172								
17	January 30, 2021 at 08:34PM	Patient_Info	60.13	226								
18	January 30, 2021 at 08:35PM	Patient_Info	65.4	238								
19	January 30, 2021 at 08:35PM	Patient_Info	102.31	238								
20	January 30, 2021 at 08:35PM	Patient_Info	76.82	238								
21	January 30, 2021 at 08:36PM	Patient_Info	107.59	238								
22	January 30, 2021 at 08:36PM	Patient_Info	81.22	233								
23	January 30, 2021 at 08:36PM	Patient_Info	69.79	237								
24	January 30, 2021 at 08:36PM	Patient_Info	67.16	59								
25	January 30, 2021 at 08:37PM	Patient_Info	101.43	194								
26	January 30, 2021 at 08:37PM	Patient_Info	70.67	229								
27	January 30, 2021 at 08:37PM	Patient_Info	64.52	229								
28	January 30, 2021 at 08:38PM	Patient_Info	69.79	115								
29	January 30, 2021 at 08:38PM	Patient_Info	109.34	227								
30	January 30, 2021 at 08:38PM	Patient_Info	76.82	236								
31	January 30, 2021 at 08:39PM	Patient_Info	107.59	234								
32	January 30, 2021 at 08:39PM	Patient_Info	75.07	234								

3) Patient's ECG Graph over Internet through Ubidots.



4) Doctor & Patient's family members will get email if "Panic Button"



CHAPTER 6

SUMMARY OF THE STUDY, CONCLUSION AND RECOMMENDATIONS

6.1 SUMMARY OF THE STUDY

The remote patient monitoring system was researched, designed and presented the concept of the Internet of things. Personal physiological data from the patient is collected that simulates fall detection, heartbeat, temperature, humidity, toxic gas, air quality control, pressure. The readings are collected in a simple cloud database and can be viewed remotely by a doctor or Healthcare giver. The data can also be used in research on medical issues affecting the elderly or chronically ill. On the security of the data, the database system is protected with Advanced Encryption Standard (AES). This generates the secret key which can be used to decrypt the patients' records ensuring that only authorized personnel access the data. This safeguards the patients' records from unauthorized users and hackers who may want to intercept.

6.2 CONCLUSION

The main objective of the experiment was successfully achieved. All the individual modules like Heartbeat detection module, fall detection module etc. and remote viewing module gave out the intended results. The designed system modules can further be optimized and produced to a final single circuit. More important fact that came up during project design is that all the circuit components used in the remote health detection system are available easily.

With the development in the integrated circuit industry, Micro Electromechanical Systems (MEMs) and microcontrollers have become affordable, have increased processing speeds, miniaturized and power efficient. This has led to increased development of embedded systems that the healthcare specialists are adopting. These embedded systems have also been adopted in the Smartphone technology. And with increased internet penetration in most developing countries through mobile phones, and with use of Internet of things (IoT) will become adopted at a faster rate. The Remote Health Care system utilizes these concepts to come up with a system for better quality of life for people in society.

From an engineering perspective, the project has seen concepts acquired through the computer science and embedded study period being practically applied. The Electric circuit analysis knowledge was used during design and fabrication of the individual modules. Electromagnetic fields analysis used in the wireless transmission between microcontrollers and Software programming used during programming of the microcontrollers to come up with a final finished circuit system.

6.3 RECOMMENDATIONS ON FUTURE WORK

a) Physiological data collection

1. Home Ultrasound

2. Brain signal monitoring

b) Remote viewing of data

1. Problems associated with having data online. Tackle Distributed denial of service. DDOS, and Data privacy/security especially of medical systems.

c) IoT based Remote Patient Monitoring System can be enhanced to detect and collect data of several anomalies for monitoring purpose such as home ultrasound, Brain signal monitoring, Tumor detection etc.

d) More research on problems associated with having data online, data privacy as IoT is managed and run by multiple technologies and multiple vendors are involved in it. Security algorithms and certain precautions by the users will help avoid any security related threats in IoT network.

e) The interface can be designed to control which sensors can be used by consumers according to their needs.

f) Web UI can be enhanced to perform several activities which include controlling the hardware, real-time graphs, history, and analysis graphs to observe anomalies etc.