JABALPUR ENGINEERING COLLEGE,

JABALPUR

GOKALPUR, JABALPUR– 482011

Minor Project Report

On

"PATIENT HEALTH MONITORING SYSTEM"

SUBMITTED TO

JABALPUR ENGINEERING COLLEGE,

JABALPUR

Submitted in partial fulfillment of the requirement for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS & COMMUNICATION ENGINEERING

SUBMITTED BY: -	ENROLL NO: -	UNDER THE GUIDANCE OF:
Priyansh Chawla	0201EC181063	Prof. GARIMA TIWARI
Rishabh Tiwari	0201EC181065	ELECTRONIC AND COMMUNICATION
Rishav Pandey	0201EC181066	DEPARTMENT
Rohit Kumar Jaiswal	0201EC181069	

JABALPUR ENGINEERING COLLEGE, JABALPUR (M.P.)

(Department of Electronics &Communication Engineering)

CERTIFICATE

This is to certify that we, **Priyansh Chawla, Rishabh Tiwari, Rishav Pandey, Rohit Kumar Jaiswal** have made the minor project report entitled **"Patient Health Monitoring System"**, is in partial fulfillment for the degree of Bachelor of Engineering in Electronics & Communication Engineering and this project is an original work done by us. Hence, we take the full responsibility for every statement and any error that may have occurred in this report.

……………………………..

Mrs. Garima Tiwari

Supervisor

Asst. Prof. E&TC

Department

………………………………..

Dr. Prashant Jain

Head of Department

HOD E & TC

Department

JABALPUR ENGINEERING COLLEGE, JABALPUR (M.P.)

(Department of Electronics &Communication Engineering)

ACKNOWLEDGEMENT

It is our proud privilege and duty to acknowledge the kind of help and guidance received from several people in preparation of this report. It would not have been possible to prepare this report in this form without their valuable help, cooperation and guidance.

First and foremost, we wish to record our sincere gratitude to **Management of this college** and our sincere thanks to **Prof. Prashant Jain,** Head, Department of **Electronics and Communication Engineering**, JEC, for his valuable suggestions and guidance throughout the period of this report. Also for guiding us in investigations for this report and in carrying out experimental work. Our numerous discussions with his were extremely helpful. We hold his in esteem for guidance, encouragement and inspiration received from his.

Our sincere thanks to **department faculties,** Project Coordinator for having supported the work related to this project. Their contributions and technical support in preparing this report are greatly acknowledged.

Last but not the least, we wish to thank our **parents** for financing our studies in this college as well as for constantly encouraging us to learn engineering. Their personal sacrifice in providing this opportunity to learn engineering is gratefully acknowledged.

Place: Jabalpur

Name of the Students

PRIYANSH CHAWLA

RISHABH TIWARI

RISHAV PANDEY

ROHIT JAISWAL

CONTENTS

Abstract

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 patient 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**.

INTRODUCTION

Internet of Things (IoT) is the networking of physical objects that contain electronics embedded within their architecture in order to communicate and sense interactions amongst each other or with respect to the external environment. In the upcoming years, IoT-based technology will offer advanced levels of services and practically change the way people lead their daily lives. Advancements in medicine, power, gene therapies, agriculture, smart cities, and smart homes are just a very few of the categorical examples where IoT is strongly established.

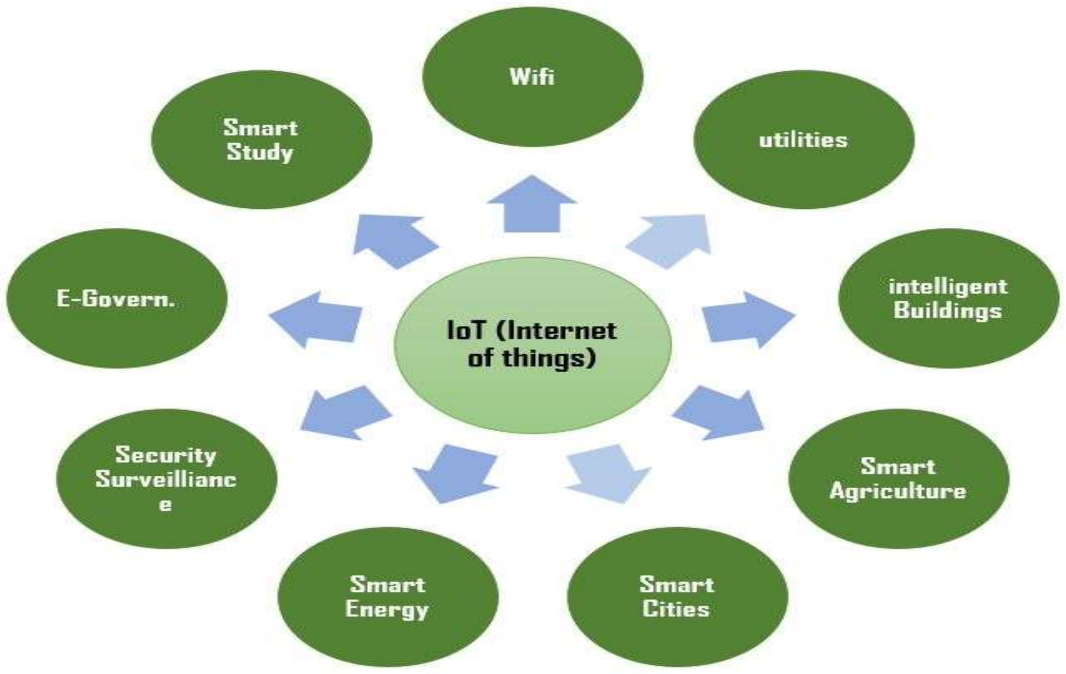
There are four main components used in IoT:

1. **Low-power embedded systems –**
   Less battery consumption, high performance are the inverse factors play a significant role during the design of electronic systems.

2. **Cloud computing –**
   Data collected through IoT devices is massive and this data has to be stored on a reliable storage server. This is where cloud computing comes into play. The data is processed and learned, giving more room for us to discover where things like electrical faults/errors are within the system.

3. **Availability of big data –**
   We know that IoT relies heavily on sensors, especially real-time. As these electronic devices spread throughout every field, their usage is going to trigger a massive flux of big data.

4. **Networking connection –**
   In order to communicate, internet connectivity is a must where each physical object is represented by an IP address. However, there are only a limited number of addresses available according to the IP naming. Due to the growing number of devices, this naming system will not be feasible anymore. Therefore, researchers are looking for another alternative naming system to represent each physical object.



Internet: Inter connectivity-For global connection

+

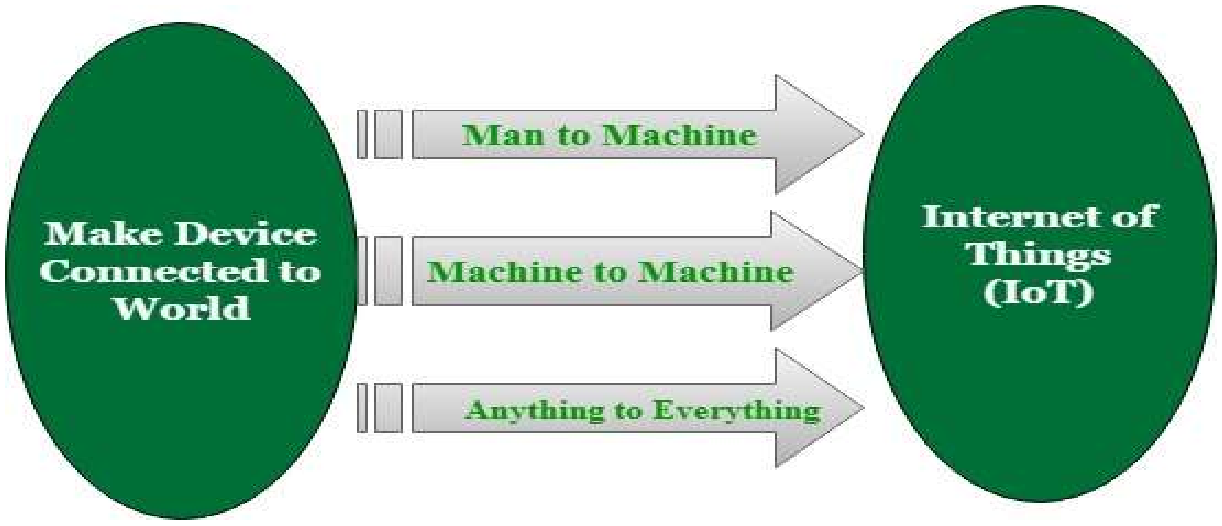
Things: Embedded system devices-sensors, actuators, RFID tags, QR code and so many.

History of IoT:

The concept of the "Internet of Things" and the term itself, first appeared in a speech by Peter T. Lewis, to the Congressional Black Caucus Foundation 15th Annual Legislative Weekend in Washington, D.C, published in September 1985.

According to Lewis, "The Internet of Things, or IoT, is the integration of people, processes and technology with connectable devices and sensors to enable remote monitoring, status, manipulation and evaluation of trends of such devices."

The term "Internet of Things" was coined independently by Kevin Ashton of Procter & Gamble, later MIT's Auto-ID Center, in 1999, though he prefers the phrase "Internet for things". At that point, he viewed radio-frequency identification (RFID) as essential to the Internet of Things, which would allow computers to manage all individual things. The main theme of the Internet of Things is to embed short-range mobile transceivers in various gadgets and daily necessities to enable new forms of communication between people and things, and between things themselves.



Characteristics of IoT:-

- Massively scalable and efficient
- IP-based addressing will no longer be suitable in the upcoming future.
- An abundance of physical objects is present that does not use IP, so IoT is made possible.
- Devices typically consume less power. When not in use, they should be automatically programmed to sleep.
- A device that is connected to another device right now may not be connected in another instant of time.
- Intermittent connectivity – IoT devices aren't always connected. In order to save bandwidth and battery consumption, devices will be powered off periodically when not in use. Otherwise, connections might turn unreliable and thus prove to be inefficient.

Application Domains:

IoT is currently found in four different popular domains:

1) Manufacturing/Industrial business - 40.2%

2) Healthcare - 30.3%

3) Security - 7.7%

4) Retail - 8.3%

Modern Applications:

- Smart home:

  IoT devices are a part of the larger concept of **home automation**, which can include lighting, heating and air conditioning, media and security systems and camera systems

- Elder care:

  One key application of a smart home is to provide assistance to physically disable and elderly individuals. Voice control can assist users with sight and mobility limitations while alert systems can be connected directly to cochlear implants worn by hearing-impaired users.

- Healthcare:

  IoT has changed people's lives, especially elderly patients, by enabling constant tracking of health conditions. This has a major impact on people living alone and their families. On any disturbance or changes in the routine activities of a person, alert mechanism sends signals to family members and concerned health providers.

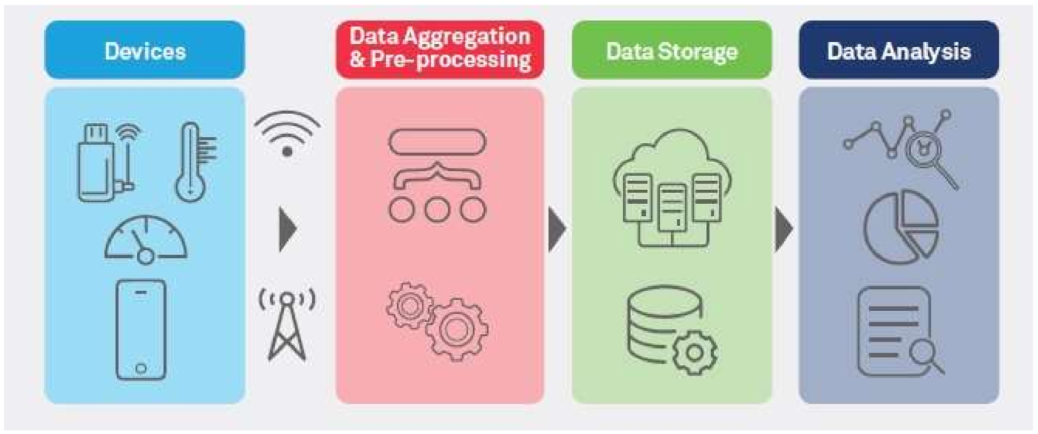
IoT for Physicians - By using wearables and other home monitoring equipment embedded with IoT, physicians can keep track of patients' health more effectively. They can track patients' adherence to treatment plans or any need for immediate medical attention. IoT enables healthcare professionals to be more watchful and connect with the patients proactively. Data collected from IoT devices can help physicians identify the best treatment process for patients and reach the expected outcomes.

IoT for Hospitals - Apart from monitoring patients' health, there are many other areas where IoT devices are very useful in hospitals. IoT devices tagged with sensors are used for tracking real time location of medical equipment like wheelchairs, defibrillators, nebulizers, oxygen pumps and other monitoring equipment. Deployment of medical staff at different locations can also be analyzed real time.
The spread of infections is a major concern for patients in hospitals. IoT-enabled hygiene monitoring devices help in preventing patients from getting infected. IoT devices also help in asset management like pharmacy inventory control, and environmental monitoring, for instance, checking refrigerator temperature, and humidity and temperature control.

IoT for Health Insurance Companies – There are numerous opportunities for health insurers with IoT-connected intelligent devices. Insurance companies can leverage data captured through health monitoring devices for their underwriting and claims operations. This data will enable them to detect fraud claims and identify prospects for underwriting. IoT devices bring transparency between insurers and customers in the underwriting, pricing, claims handling, and risk assessment processes. In the light of IoT-captured data-driven decisions in all operation processes, customers will have adequate visibility into underlying thought behind every decision made and process outcomes.
Insurers may offer incentives to their customers for using and sharing health data generated by IoT devices. They can reward customers for using IoT devices to keep track of their routine activities and adherence to treatment plans and precautionary health measures. This will help insurers to reduce claims significantly. IoT devices can also enable insurance companies to validate claims through the data captured by these devices.

Redefining Healthcare

The proliferation of healthcare-specific IoT products opens up immense opportunities. And the huge amount of data generated by these connected devices hold the potential to transform healthcare.
IoT has a four-step architecture that are basically stages in a process (See Figure 1). All four stages are connected in a manner that data is captured or processed at one stage and yields the value to the next stage. Integrated values in the process brings intuitions and deliver dynamic business prospects.
Step 1: First step consists of deployment of interconnected devices that includes sensors, actuators, monitors, detectors, camera systems etc. These devices collect the data.
Step 2: Usually, data received from sensors and other devices are in analog form, which need to be aggregated and converted to the digital form for further data processing.
Step 3: Once the data is digitized and aggregated, this is pre-processed, standardized and moved to the data center or Cloud.
Step 4: Final data is managed and analyzed at the required level. Advanced Analytics, applied to this data, brings actionable business insights for effective decision-making.
IoT is redefining healthcare by ensuring better care, improved treatment outcomes and reduced costs for patients, and better processes and workflows, improved performance and patient experience for healthcare providers.



The major advantages of IoT in healthcare include:

- Cost Reduction: IoT enables patient monitoring in real time, thus significantly cutting down unnecessary visits to doctors, hospital stays and re-admissions
- Improved Treatment: It enables physicians to make evidence-based informed decisions and brings absolute transparency
- Faster Disease Diagnosis: Continuous patient monitoring and real time data helps in diagnosing diseases at an early stage or even before the disease develops based on symptoms
- Proactive Treatment: Continuous health monitoring opens the doors for providing proactive medical treatment
- Drugs and Equipment Management: Management of drugs and medical equipment is a major challenge in a healthcare industry. Through connected devices, these are managed and utilized efficiently with reduced costs
- Error Reduction: Data generated through IoT devices not only help in effective decision making but also ensure smooth healthcare operations with reduced errors, waste and system costs

They can also be equipped with additional safety features:-

- Earthquake detection
- Radiation detection/hazardous gas detection
- Smartphone detection
- Water flow monitoring
- Traffic monitoring
- Wearables

Challenges in adoption of IoT:

1. IoT issues with security:

In cybersecurity terms, IoT devices greatly expand the "attack surface" or the amount of potential areas for cybercriminals to penetrate a secure network.

Cybercriminals don't have to crack an IoT device's plastic enclosure to access sensitive materials. They can simply finesse their way in through one of the many security vulnerabilities that are found throughout the IoT. Many IoT devices have default passwords left unchanged, unpatched software and other major security vulnerabilities.

2. Lack of regulation about IoT:

The lack of strong IoT regulations is a big part of why the IoT remains a severe security risk, and the problem is likely to get worse as the potential attack surface expands to include ever more crucial devices. When medical devices, cars and children's toys are all connected to the Internet, it's not hard to imagine many potential disaster scenarios unfolding in the absence of sufficient regulation.

3. Challenges with compatibility:

IoT is growing in many different directions, with **many different technologies competing to become the standard.** This will cause difficulties and require the deployment of extra hardware and software when connecting devices.

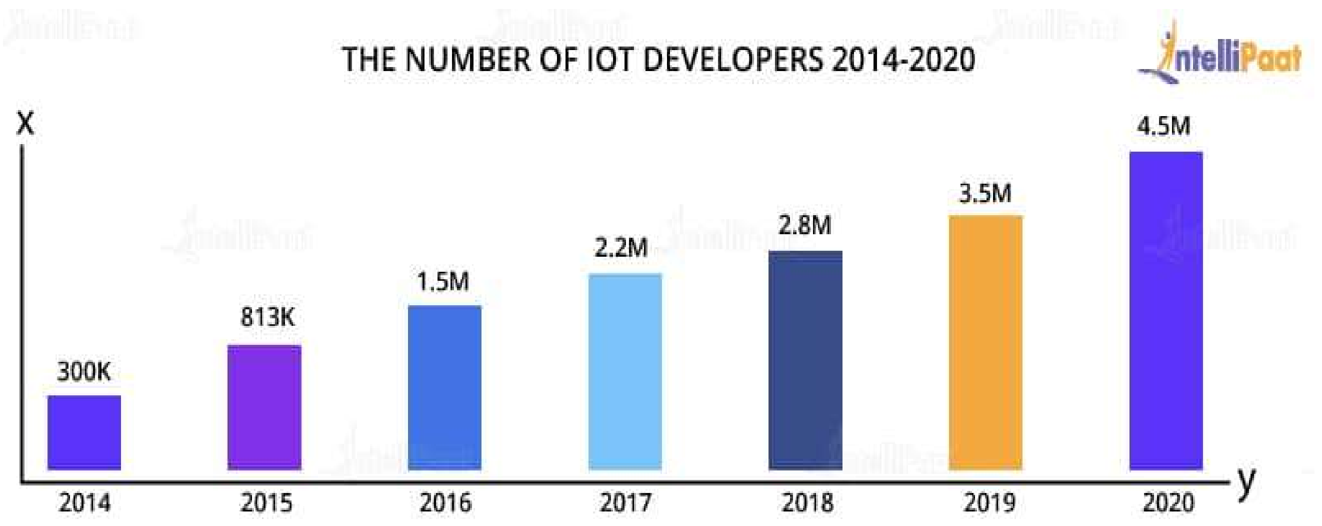
4. Limited bandwidth:

Connectivity is a bigger challenge to the IoT than you might expect. As the size of the IoT market grows exponentially, some experts are concerned that bandwidth-intensive IoT applications such as video streaming will soon struggle for space on the IoT's current server-client model.

FUTURE SCOPE OF IoT:

Internet of Things has emerged as a leading technology around the world. It has gained a lot of popularity in lesser time. Also, the advancements in Artificial Intelligence and Machine Learning have made the automation of IoT devices easy. Basically, AI and ML programs are combined with IoT devices to give them proper automation. Due to this, IoT has also expanded its area of application in various sectors. Here, in this section, we will discuss the applications and the future scope of IoT in

- Healthcare,
- Automotive, and
- Agriculture industries.



Smart Sensors

What is a Sensor?

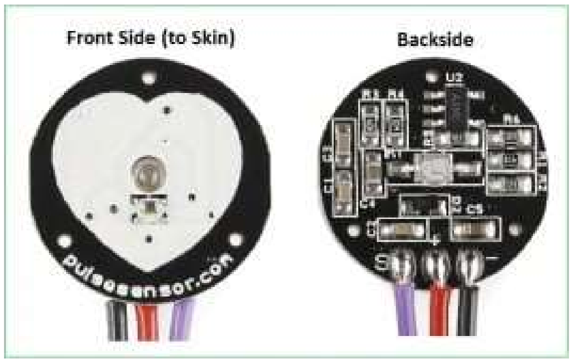
- A **sensor** is a device that detects the change in the environment and responds to some output on the other system.
- A sensor producing an output when combined with some interfacing hardware is termed to be an intelligent or smart sensor, which is a more acceptable term now.

Sensors + Interfacing Hardware = Smart Sensors

Types of Smart Sensors:-

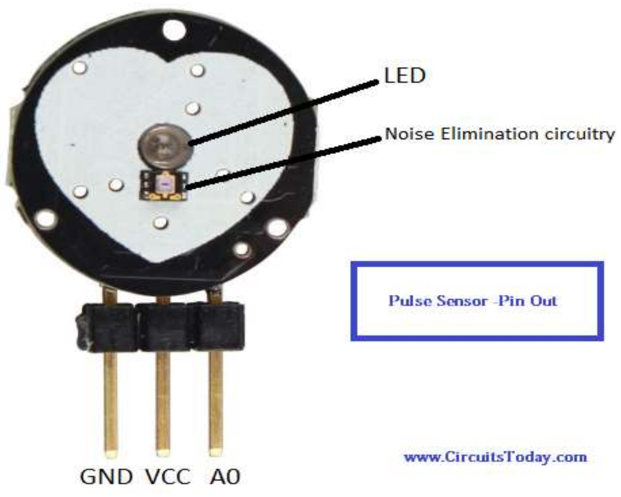
- Pulse Sensor
- LM-35 temperature sensor
- Proximity Sensor
- Pressure Sensor
- Pressure Sensor
- Gas & Smoke Sensor
- Accelerometer Sensors
- Level Sensors
- Motion Detection Sensors

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 ready to read heart rate. Also, it has an Arduino demo code that makes it easy to use.

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



Temperature Sensor:

A temperature sensor is an electronic device that measures the temperature of its environment and converts the input data into electronic data to record, monitor, or signal temperature changes.

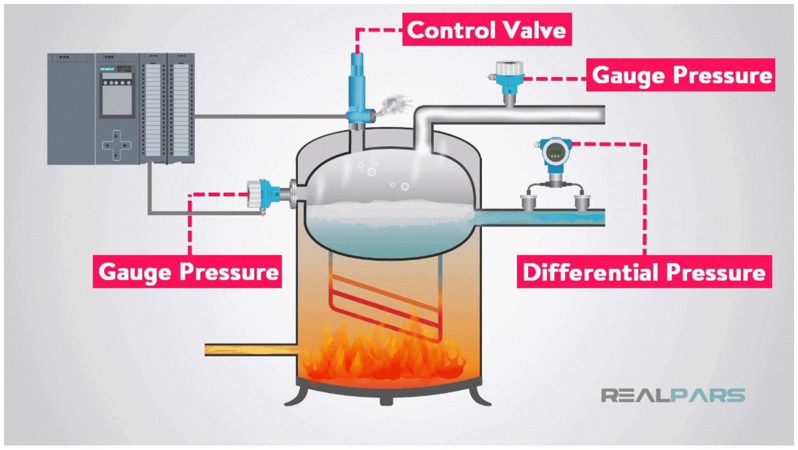
Proximity Sensor:

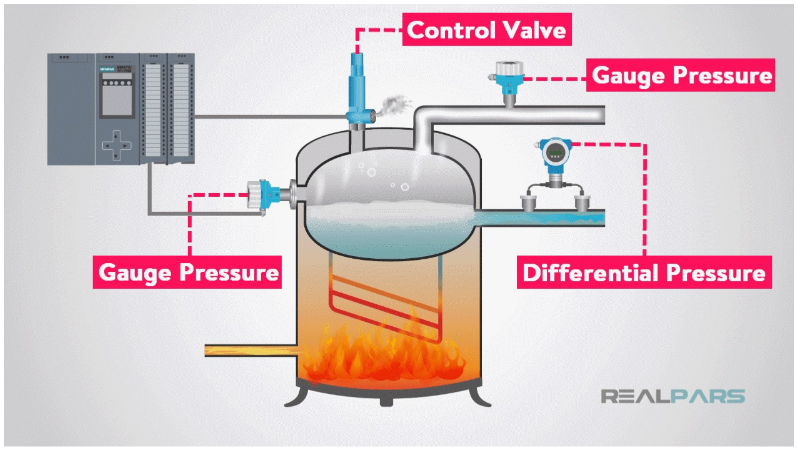
A proximity sensor is a sensor able to detect the presence of nearby objects without any physical contact.

A proximity sensor often emits an electromagnetic field or a beam of electromagnetic radiation, and looks for changes in the field or return signal. The object being sensed is often referred to as the proximity sensor's target

Pressure Sensor:

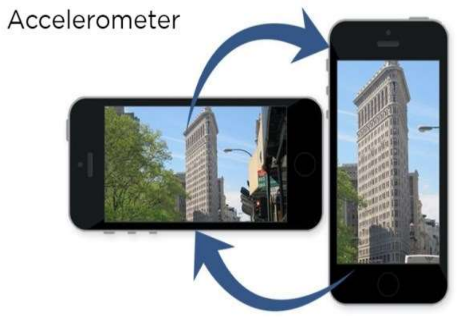
- A pressure sensor is a device that senses pressure and converts it into an electric signal.
- A pressure sensor usually acts as a transducer.





Accelerometer Sensors:

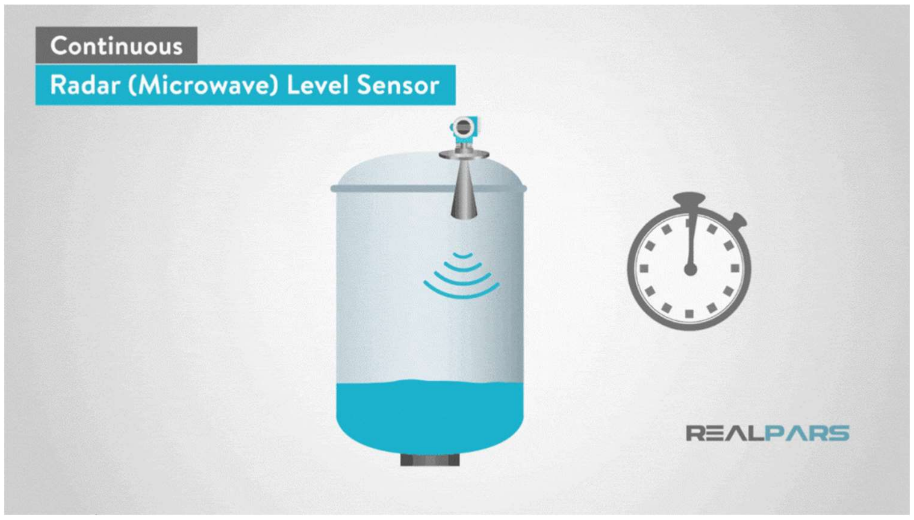
- Accelerometer is a transducer that is used to measure the physical or measurable acceleration experienced by an object due to inertial forces and converts the mechanical motion into an electrical output.



- Accelerometers in mobile phones are used to detect the orientation of the phone.

Level Sensors:

- A sensor which is used to determine the level or amount of fluids, liquids or other substances that flow in an open or closed system is called Level sensor.



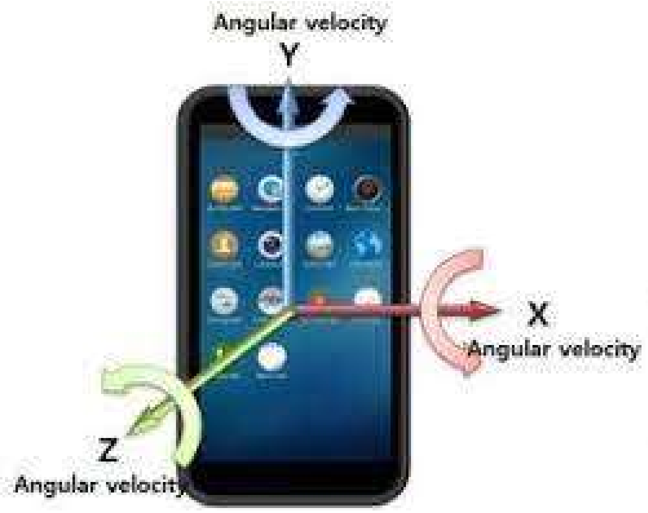
Motion Detection Sensors:

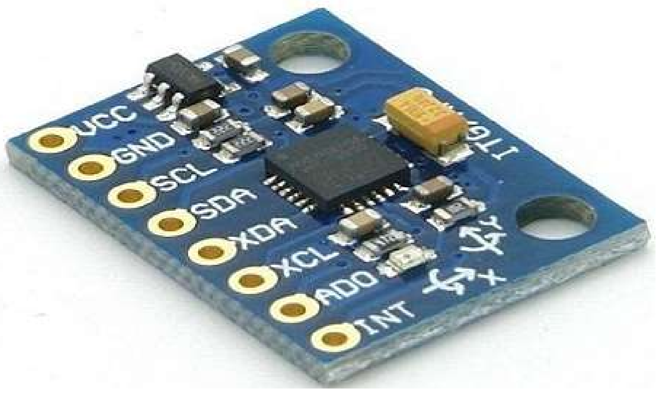
- A motion detector is an electronic device which is used to detect the physical movement(motion) in a given area and it transforms motion into an electric signal ; motion of any object or motion of human being.

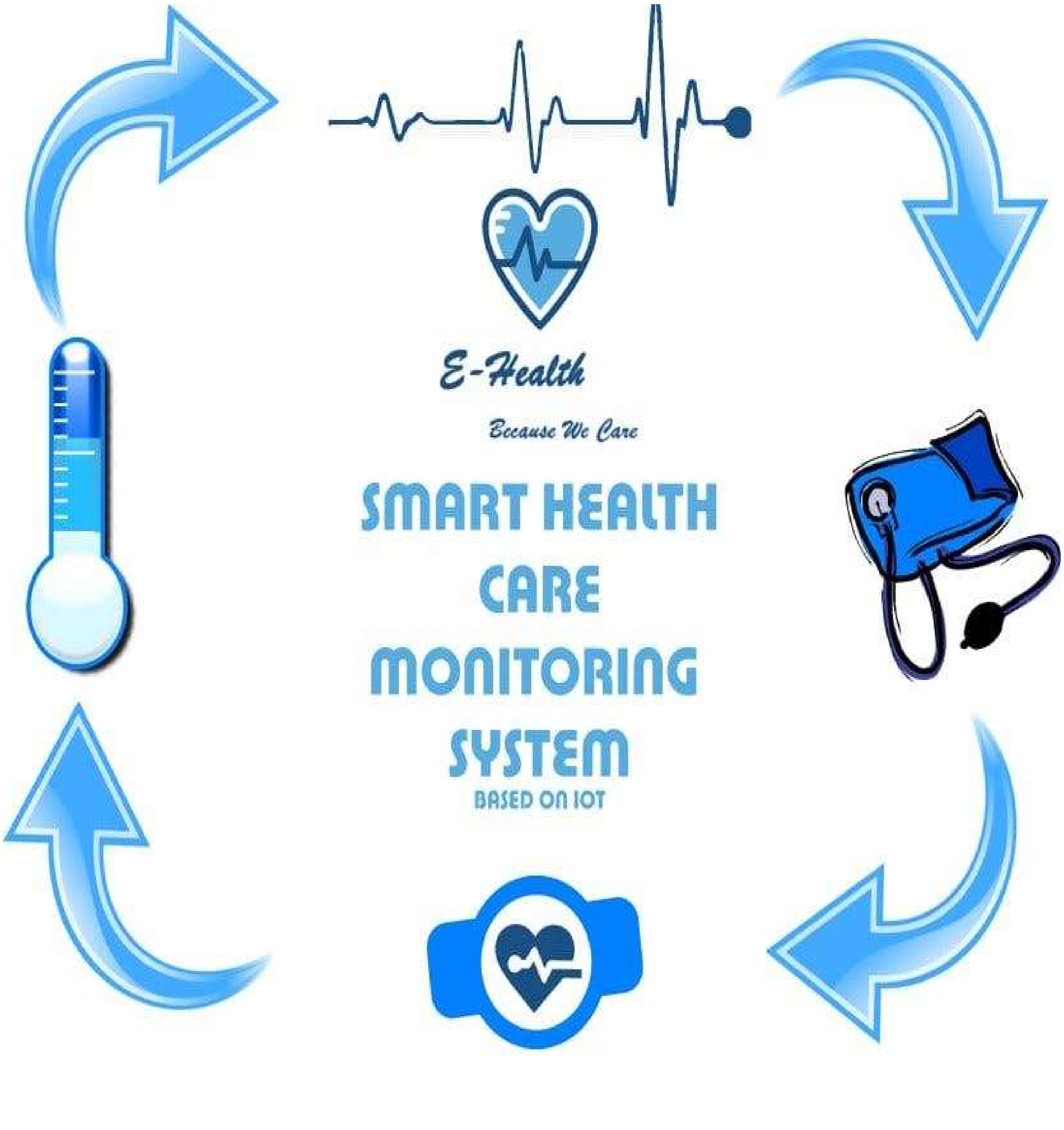
- Ex- Passive infrared (PIR), in household motion sensing devices and is designed to turn on a light only when motion is detected *and* when the surrounding environment is sufficiently dark.

Gyroscope Sensors:

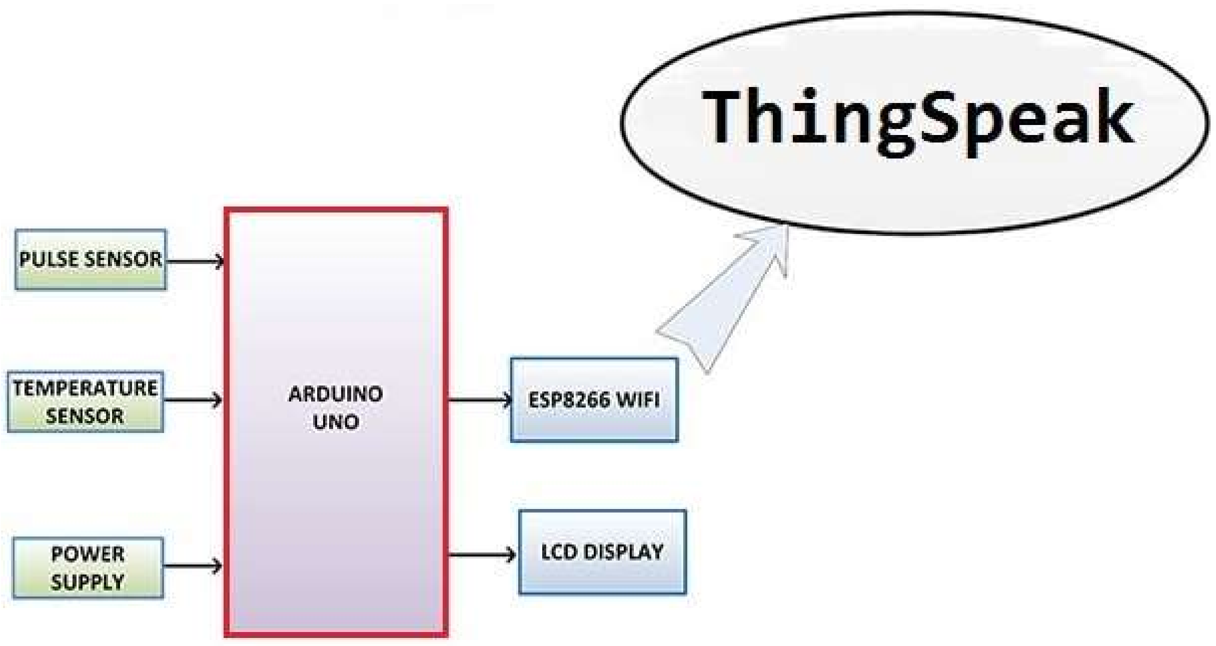
- A sensor or device which is used to measure the angular rate or angular velocity is known as Gyro sensors. The most important application is monitoring the orientation of an object.






E-Health
Because We Care
SMART HEALTH
CARE
MONITORING
SYSTEM
BASED ON IOT

Block Diagram:

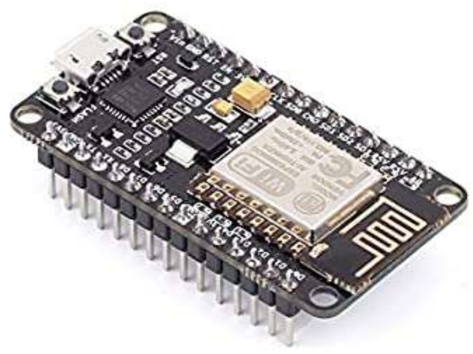


This is a simple block diagram that explains the **IoT Based Patient Health Monitoring System using ESP8266 & Arduino**. Pulse Sensor and LM35 Temperature Sensors measure BPM & Environmental Temperature respectively. The Arduino processes the code and displays it to 16*2 LCD Display. **ESP8266 Wi-Fi 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.

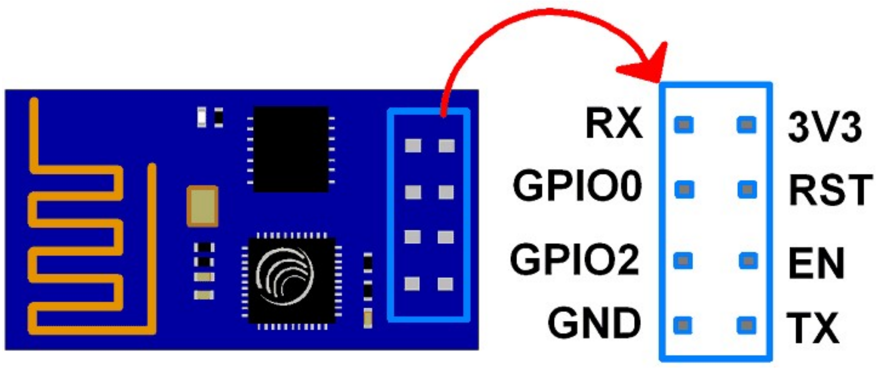
Components Used:

1. Arduino UNO Rev3 microcontroller board
2. ESP8266-01 WIFI Module
3. LCD Display
4. Pulse Sensor
5. LM35 Analog Temperature Sensor
6. Breadboard
7. Potentiometer
8. Resistors of 2k and 1k ohms
9. LED
10. Jumper Wires

ESP8266:



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.



The ESP8266 module works with 3.3V only, anything more than 3.7V would kill the module hence be cautious 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

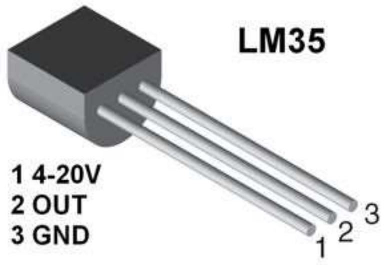
Pin 5: Flash/GPIO – 0: General purpose Input/output pin

Pin 6 : Reset: Resets the module

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

Pin 8: Vcc: Connect to +3.3V only

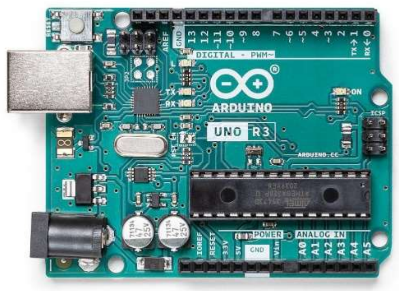
LM35 Temperature Sensor:



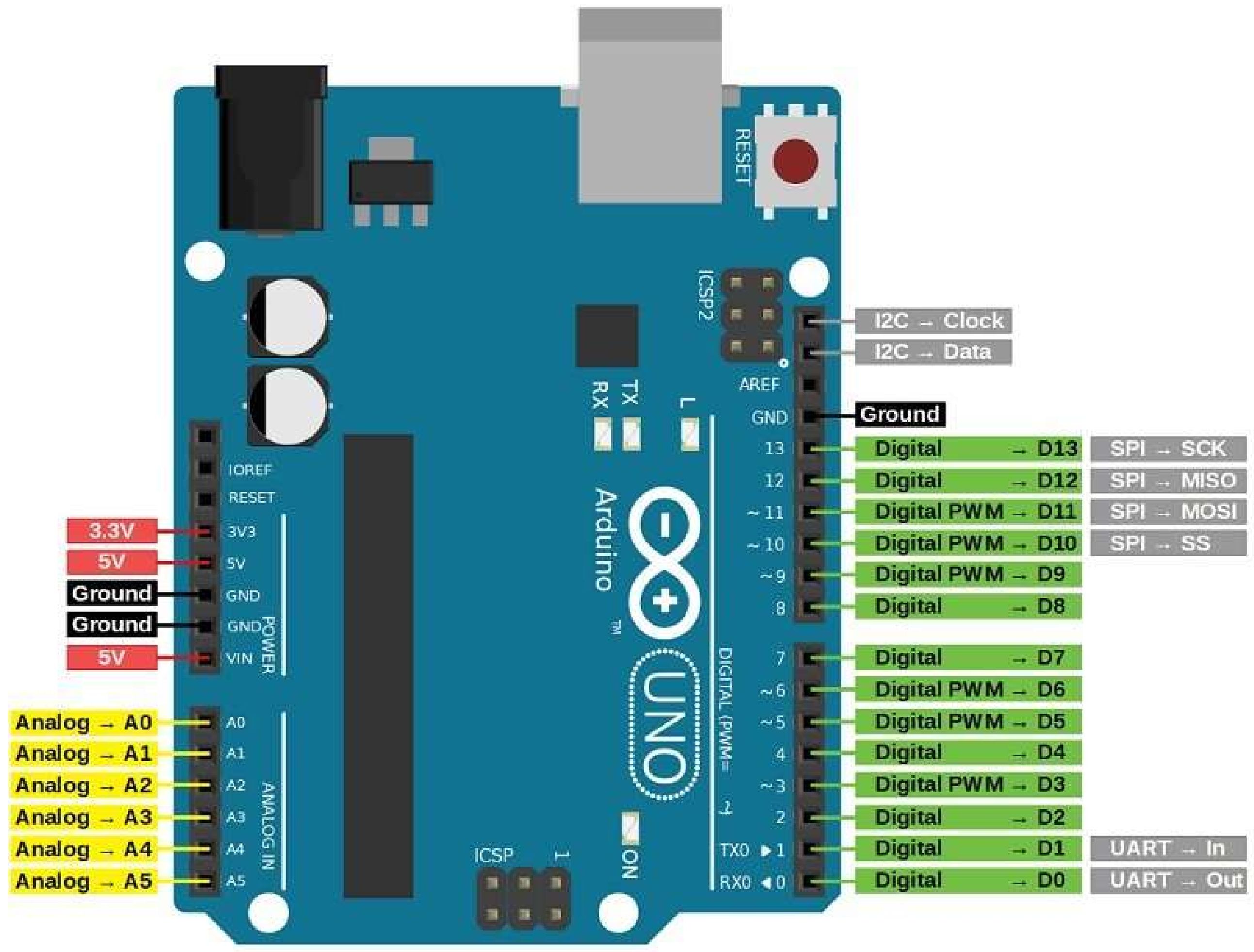
The **LM35** series are precision integrated-circuit temperature devices with an output voltage linearly-proportional to the Centigrade temperature. The LM35 device has an advantage over linear temperature sensors calibrated in Kelvin, as the user is not required to subtract a large constant voltage from the output to obtain convenient Centigrade scaling. The LM35 device does not require any external calibration or trimming to provide typical accuracies of ±¼°C at room temperature and ±¾°C over a full −55°C to 150°C temperature range.

Arduino UNO Rev3 microcontroller board:

Arduino is an **open-source hardware** and software company, project and user community that designs and manufactures single-board microcontrollers and microcontroller kits for building digital devices.



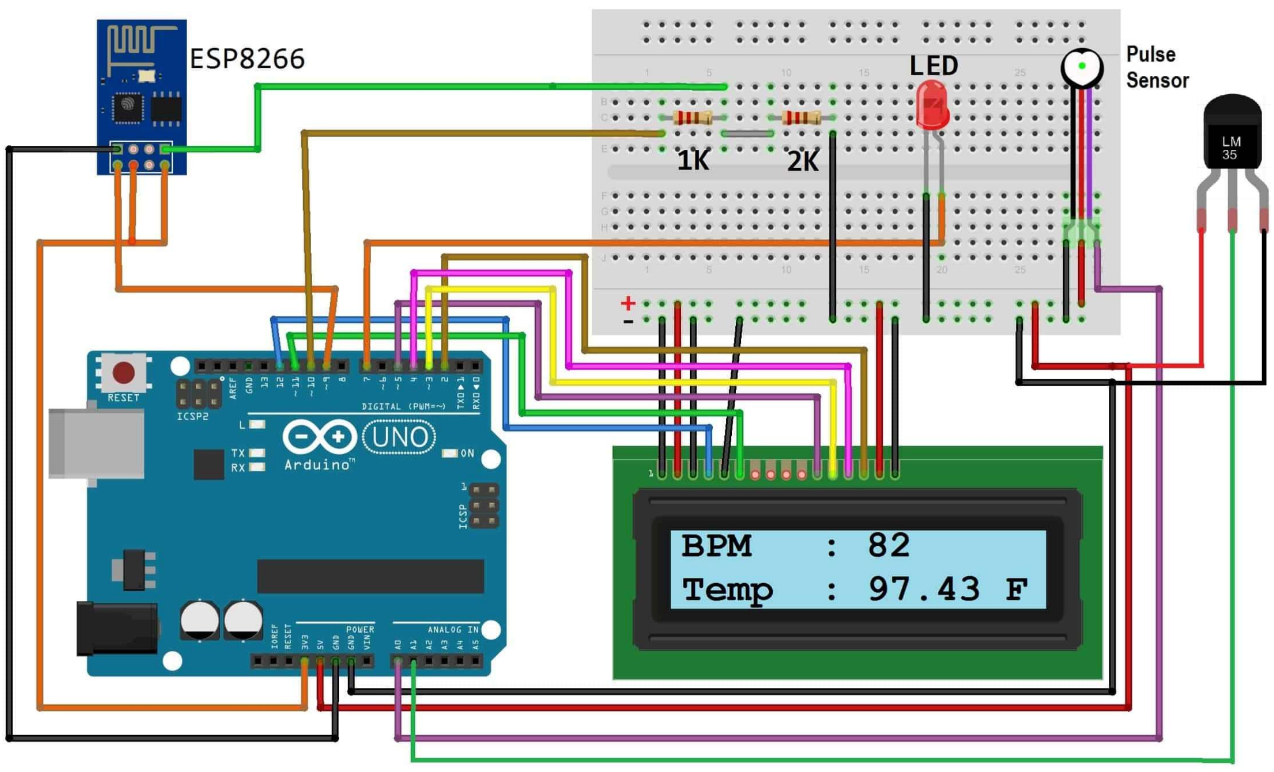
The **Arduino UNO** is the best board to get started with electronics and coding. If this is your first experience tinkering with the platform, the UNO is the most robust board you can start playing with. The UNO is the most used and documented board of the whole Arduino family.



Detailed Working:

Circuit Diagram & Connections:

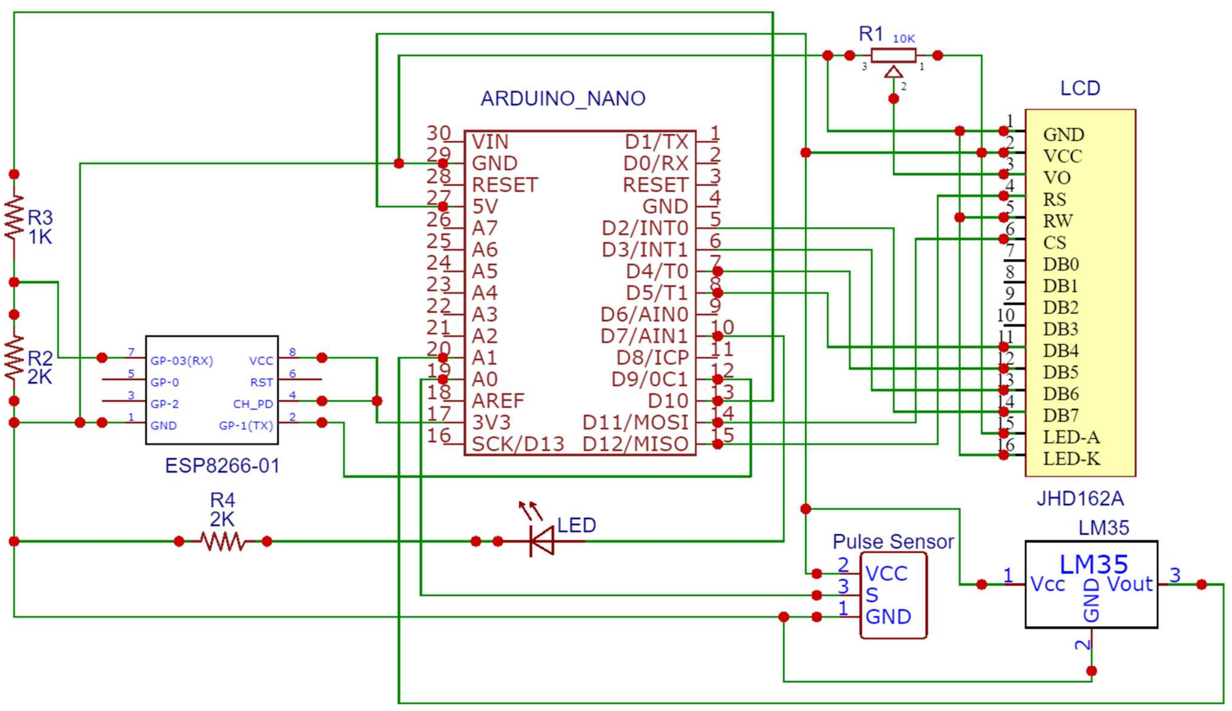
For designing IoT Based Patient Health Monitoring System using ESP8266 & Arduino, assemble the circuit as shown in the figure below.



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.

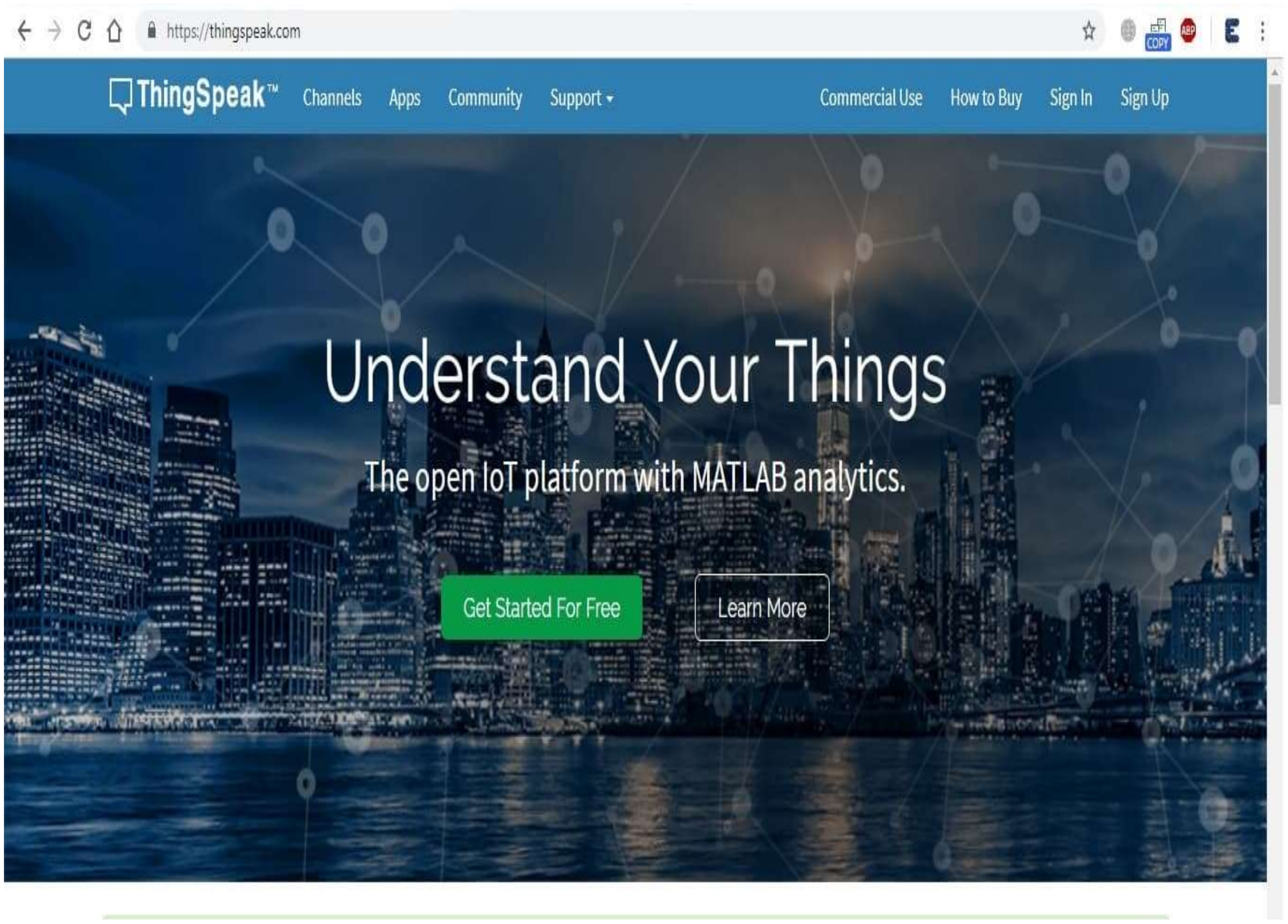
Schematic designed using EasyEDA software:



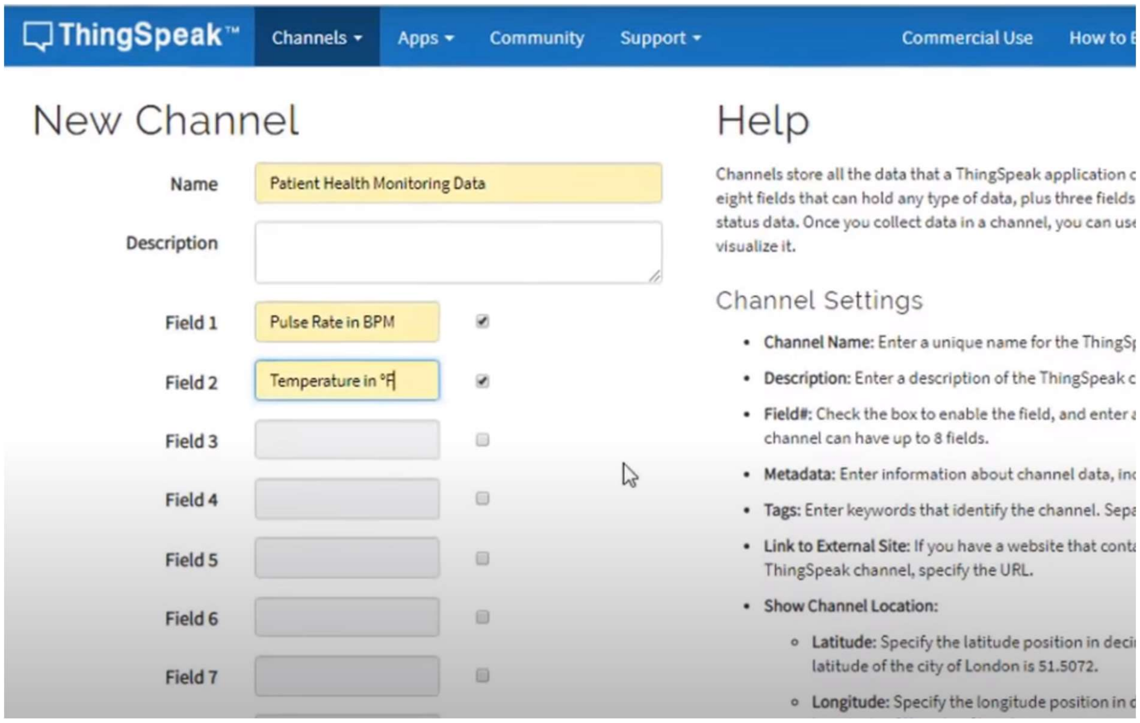
Setting the ThingSpeak:

ThingSpeak

ThingSpeak provides a very good tool for IoT based projects. By using the ThingSpeak site, we can monitor our data and control our system over the Internet, using the Channels and web pages provided by ThingSpeak. So first you need to sign up for ThingSpeak. So, visit **https://thingspeak.com** and create an account.



Create a New Channel with 2 field for each sensor, one for Pulse Rate in BPM and other one for Temperature in Fahrenheit.



New Channel

Name	Patient Health Monitoring Data	
Description		
Field 1	Pulse Rate in BPM	✓
Field 2	Temperature in °F	✓
Field 3		☐
Field 4		☐
Field 5		☐
Field 6		☐
Field 7		☐

Help

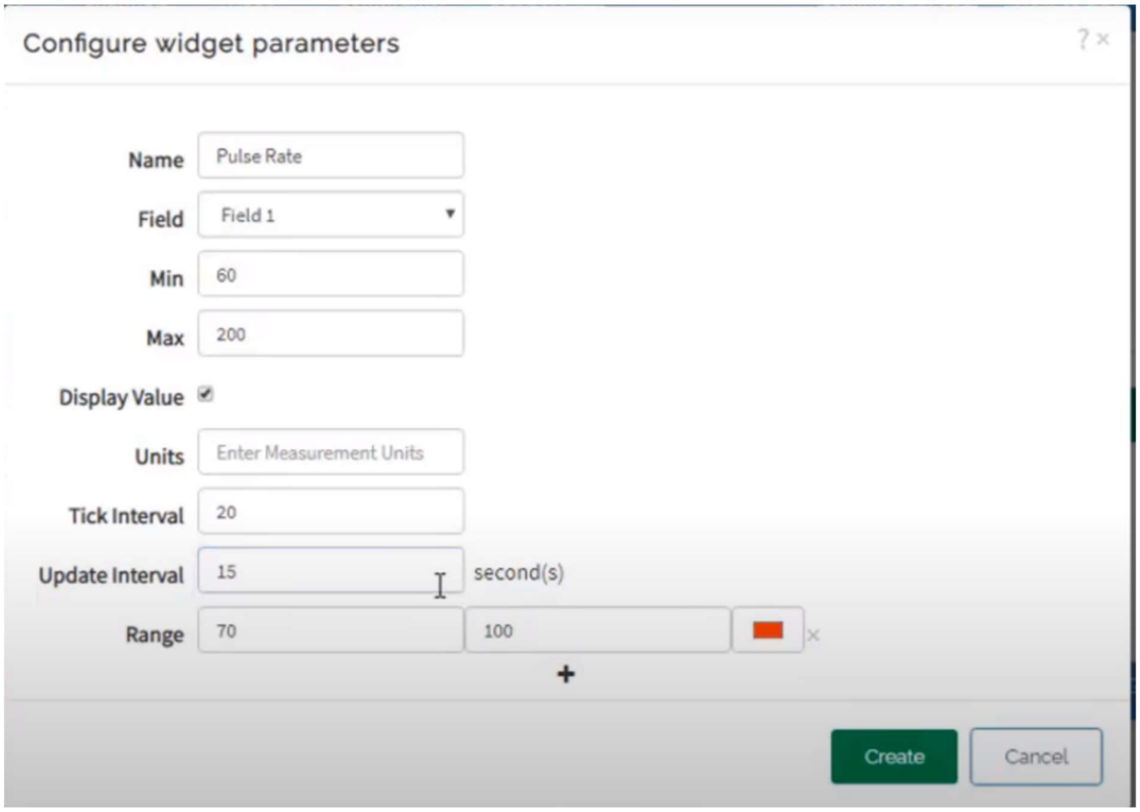
Channels store all the data that a ThingSpeak application c
eight fields that can hold any type of data, plus three fields
status data. Once you collect data in a channel, you can use
visualize it.

Channel Settings

- **Channel Name:** Enter a unique name for the ThingSp
- **Description:** Enter a description of the ThingSpeak c
- **Field#:** Check the box to enable the field, and enter a
  channel can have up to 8 fields.
- **Metadata:** Enter information about channel data, inc
- **Tags:** Enter keywords that identify the channel. Sepa
- **Link to External Site:** If you have a website that conta
  ThingSpeak channel, specify the URL.
- **Show Channel Location:**
  - **Latitude:** Specify the latitude position in deci
    latitude of the city of London is 51.5072.
  - **Longitude:** Specify the longitude position in d

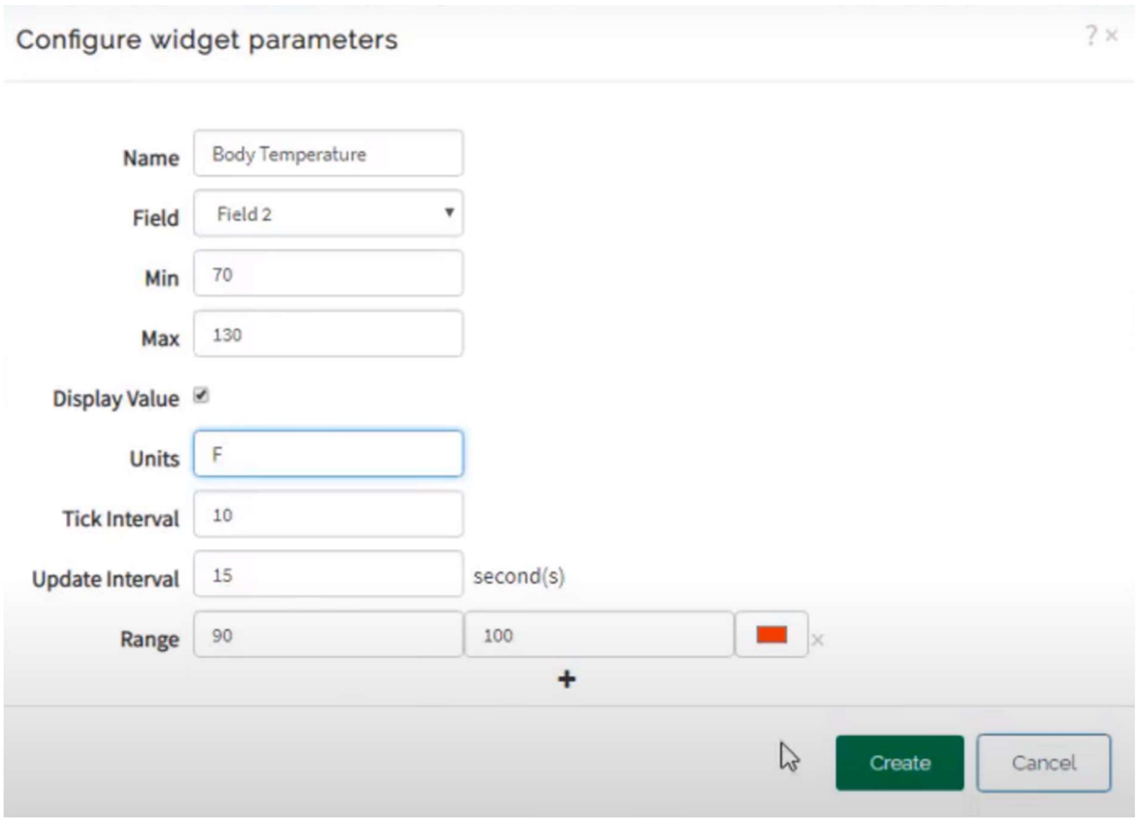
Then create the **API keys**. This key is required for programming modification and setting your data.

Configure widget parameters :

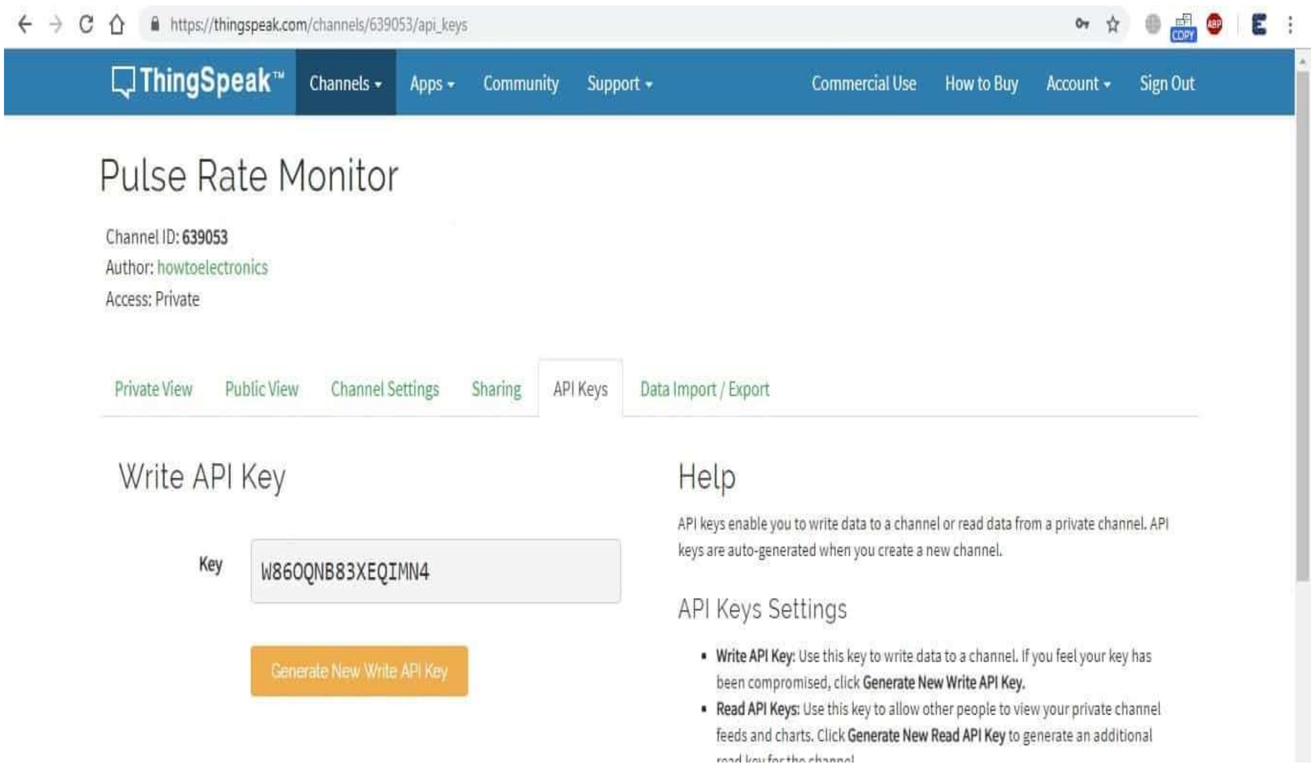


Configure widget parameters

Name	Pulse Rate	
Field	Field 1	
Min	60	
Max	200	
Display Value	✓	
Units	Enter Measurement Units	
Tick Interval	20	
Update Interval	15	second(s)
Range	70	100 🟥


Configure widget parameters

Name	Body Temperature
Field	Field 2
Min	70
Max	130
Display Value	✓
Units	F
Tick Interval	10
Update Interval	15 second(s)
Range	90 / 100 / 🟥


https://thingspeak.com/channels/639053/api_keys

Pulse Rate Monitor

Channel ID: **639053**
Author: howtoelectronics
Access: Private

Write API Key

Key W86OQNB83XEQIMN4

Help

API keys enable you to write data to a channel or read data from a private channel. API keys are auto-generated when you create a new channel.

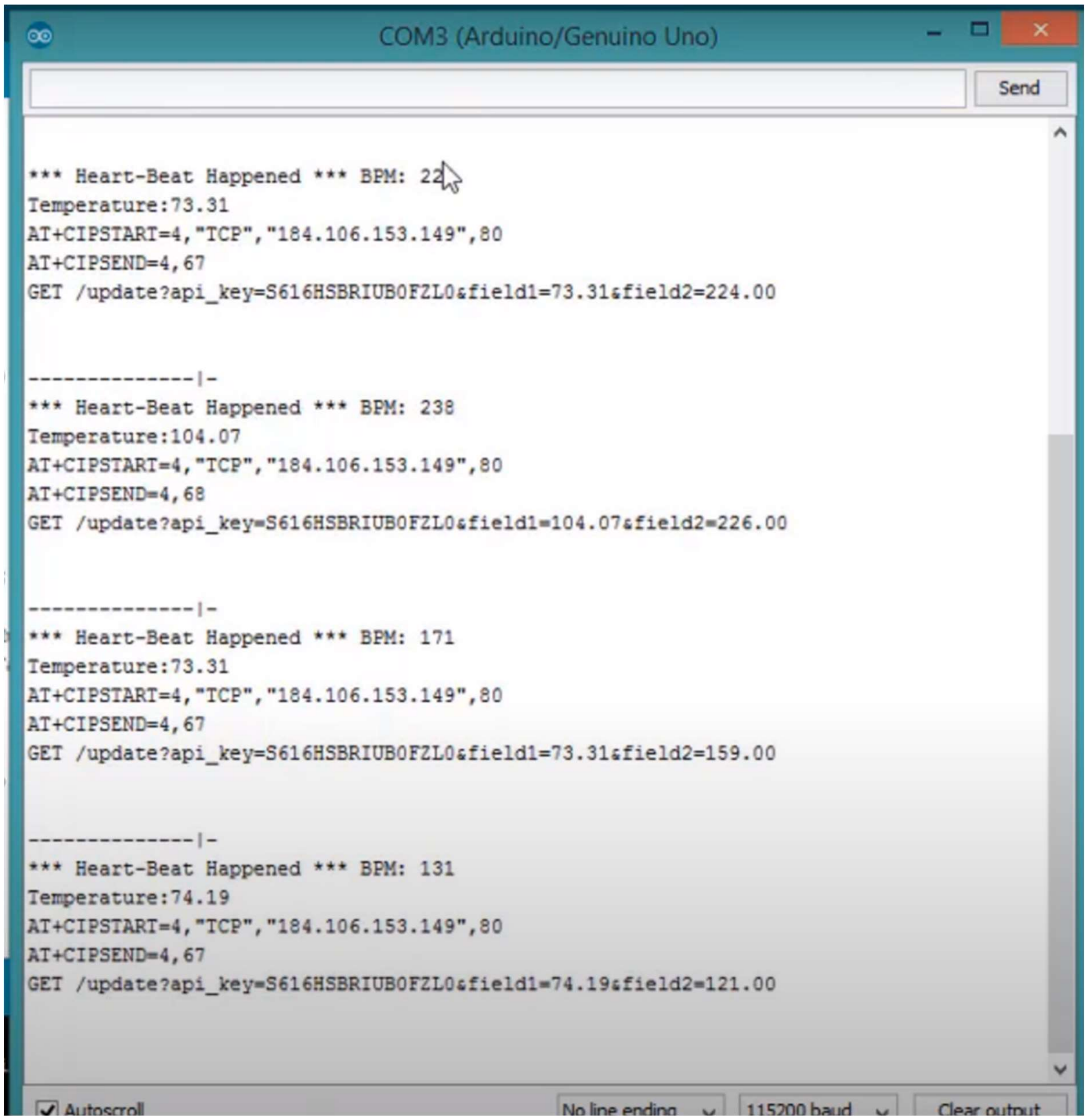
API Keys Settings

- **Write API Key:** Use this key to write data to a channel. If you feel your key has been compromised, click **Generate New Write API Key.**
- **Read API Keys:** Use this key to allow other people to view your private channel feeds and charts. Click **Generate New Read API Key** to generate an additional read key for the channel.

Then upload the code to the Arduino UNO by assembling the circuit shown above. Open the serial monitor and it will automatically connect to Wi-Fi and set up everything.

Now click on channels so that you can see the online data streaming, i.e IoT Based Patient Health Monitoring System using ESP8266 & Arduino as shown in the figure here.

Real time data can be seen from Arduino Monitor taken from sensors:

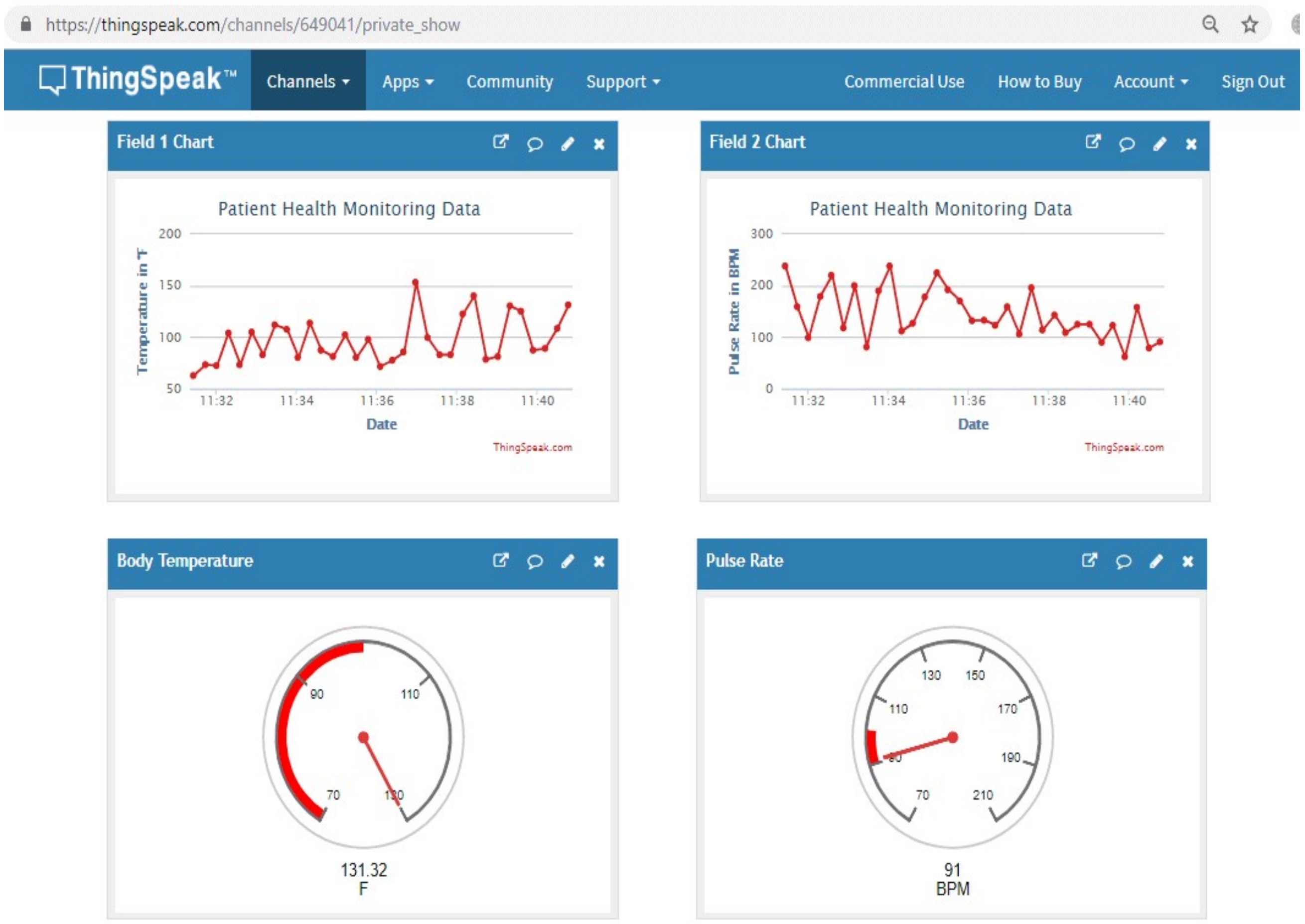


```
*** Heart-Beat Happened *** BPM: 22
Temperature:73.31
AT+CIPSTART=4,"TCP","184.106.153.149",80
AT+CIPSEND=4,67
GET /update?api_key=S616HSBRIUB0FZL0&field1=73.31&field2=224.00


--------------|-
*** Heart-Beat Happened *** BPM: 238
Temperature:104.07
AT+CIPSTART=4,"TCP","184.106.153.149",80
AT+CIPSEND=4,68
GET /update?api_key=S616HSBRIUB0FZL0&field1=104.07&field2=226.00


--------------|-
*** Heart-Beat Happened *** BPM: 171
Temperature:73.31
AT+CIPSTART=4,"TCP","184.106.153.149",80
AT+CIPSEND=4,67
GET /update?api_key=S616HSBRIUB0FZL0&field1=73.31&field2=159.00


--------------|-
*** Heart-Beat Happened *** BPM: 131
Temperature:74.19
AT+CIPSTART=4,"TCP","184.106.153.149",80
AT+CIPSEND=4,67
GET /update?api_key=S616HSBRIUB0FZL0&field1=74.19&field2=121.00
```


https://thingspeak.com/channels/649041/private_show
ThingSpeak™
Channels
Apps
Community
Support
Commercial Use
How to Buy
Account
Sign Out
Field 1 Chart
Patient Health Monitoring Data
Temperature in °F
Date
ThingSpeak.com
Field 2 Chart
Patient Health Monitoring Data
Pulse Rate in BPM
Date
ThingSpeak.com
Body Temperature
131.32
F
Pulse Rate
91
BPM

Source Code/Program:

The source code for the project IoT Based Patient Health Monitoring System using ESP8266 & Arduino is given below. API has to be updated according to thingSpeak platform provided one. We use Arduino IDE for coding downloaded from https://www.arduino.cc/en/software.

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

#include <SoftwareSerial.h>

float pulse = 0;

float temp = 0;

SoftwareSerial ser(9,10);

String apiKey = "OO707TGA1BLUNN12";   // Can be changed a/c to thingSpeak


// Variables

int pulsePin = A0;

int blinkPin = 7 ;

int fadePin = 13;

int fadeRate = 0;


// Volatile Variables, used in the interrupt service routine!

volatile int BPM; /

volatile int Signal; // holds the incoming raw data

volatile int IBI = 600; // int that holds the time interval between beats! Must be seeded!

volatile boolean Pulse = false; // "True" when User's live heartbeat is detected.          "False"
when nota "live beat".

volatile boolean QS = false; // becomes true when Arduoino finds a beat.
```

```
// Regards Serial OutPut -- Set This Up to your needs

static boolean serialVisual = true; //

volatile int rate[10]; // array to hold last ten IBI values

volatile unsigned long sampleCounter = 0; // used to determine pulse timing

volatile unsigned long lastBeatTime = 0; // used to find IBI

volatile int P = 512; // used to find peak in pulse wave, seeded

volatile int T = 512; // used to find trough in pulse wave, seeded

volatile int thresh = 525; // used to find instant moment of heart beat, seeded

volatile int amp = 100; // used to hold amplitude of pulse waveform, seeded

volatile boolean firstBeat = true; // used to seed rate array so we startup with reasonable BPM

volatile boolean secondBeat = false; // used to seed rate array so we startup with reasonable
BPM


void setup()

{

lcd.begin(16, 2);

pinMode(blinkPin,OUTPUT); // pin that will blink to your heartbeat!

pinMode(fadePin,OUTPUT); // pin that will fade to your heartbeat!

Serial.begin(115200); //

interruptSetup(); //

lcd.clear();

lcd.setCursor(0,0);

lcd.print(" Patient Health");

lcd.setCursor(0,1);

lcd.print(" Monitoring ");

delay(4000);
lcd.clear();

lcd.setCursor(0,0);

lcd.print("Initializing....");

delay(5000);

lcd.clear();

lcd.setCursor(0,0);

lcd.print("Getting Data....");

ser.begin(9600);

ser.println("AT");

delay(1000);

ser.println("AT+GMR");

delay(1000);

ser.println("AT+CWMODE=3");

delay(1000);

ser.println("AT+RST");

delay(5000);

ser.println("AT+CIPMUX=1");

delay(1000);


String cmd="AT+CWJAP=\"Alexahome\",\"98765432\"";

ser.println(cmd);

delay(1000);

ser.println("AT+CIFSR");

delay(1000);

}

// Main part of Program

void loop()
```

```
{

serialOutput();

if (QS == true) // A Heartbeat Was Found

{

// BPM and IBI have been Determined

// Quantified Self "QS" true when arduino finds a heartbeat

fadeRate = 255; // Makes the LED Fade Effect Happen, Set 'fadeRate' Variable to 255 to fade
LED with pulse

serialOutputWhenBeatHappens(); // A Beat Happened, Output that to serial.

QS = false; // reset the Quantified Self flag for next time

}

ledFadeToBeat(); // Makes the LED Fade Effect Happen

delay(20); // take a break

read_temp();

esp_8266();

}

void ledFadeToBeat()

{

fadeRate -= 15; // set LED fade value

fadeRate = constrain(fadeRate,0,255); // keep LED fade value from going into negative
numbers!

analogWrite(fadePin,fadeRate); // fade LED

}

void interruptSetup()

{

// Initializes Timer2 to throw an interrupt every 2mS.

TCCR2A = 0x02; //

TCCR2B = 0x06; //
```

```
OCR2A = 0X7C; //

TIMSK2 = 0x02;

sei(); // MAKE SURE GLOBAL INTERRUPTS ARE ENABLED

}

void serialOutput()

{ // Decide How To Output Serial.

if (serialVisual == true)

{

arduinoSerialMonitorVisual('-', Signal); // goes to function that makes Serial Monitor
Visualizer

}

else

{

sendDataToSerial('S', Signal); // goes to sendDataToSerial function

}

}

void serialOutputWhenBeatHappens()

{

if (serialVisual == true) // Code to Make the Serial Monitor Visualizer Work

{

Serial.print("*** Heart-Beat Happened *** "); //ASCII Art Madness

Serial.print("BPM: ");

Serial.println(BPM);

}

else

{

sendDataToSerial('B',BPM); // send heart rate with a 'B' prefix
sendDataToSerial('Q',IBI); // send time between beats with a 'Q' prefix

}

}

void arduinoSerialMonitorVisual(char symbol, int data )

{

const int sensorMin = 0; // sensor minimum, discovered through experiment

const int sensorMax = 1024; // sensor maximum, discovered through experiment

int sensorReading = data; // map the sensor range to a range of 12 options:

int range = map(sensorReading, sensorMin, sensorMax, 0, 11);

// do something different depending on the

// range value:

switch (range)

{

case 0:

Serial.println(""); /////ASCII Art Madness

break;

case 1:

Serial.println("---");

break;

case 2:

Serial.println("------");

break;

case 3:

Serial.println("---------");

break;

case 4:

Serial.println("------------");
```

```
break;

case 5:

Serial.println("--------------|-");

break;

case 6:

Serial.println("--------------|---");

break;

case 7:

Serial.println("--------------|-------");

break;

case 8:

Serial.println("--------------|----------");

break;

case 9:

Serial.println("--------------|----------------");

break;

case 10:

Serial.println("--------------|-------------------");

break;

case 11:

Serial.println("--------------|------------------------");

break;

}

}

void sendDataToSerial(char symbol, int data )

{

Serial.print(symbol);
Serial.println(data);
}
ISR(TIMER2_COMPA_vect) //triggered when Timer2 counts to 124
{
cli(); // disable interrupts while we do this
Signal = analogRead(pulsePin); // read the Pulse Sensor
sampleCounter += 2; // keep track of the time in mS with this variable
int N = sampleCounter - lastBeatTime; // monitor the time since the last beat to avoid noise
// find the peak and trough of the pulse wave

if(Signal < thresh && N > (IBI/5)*3) // avoid dichrotic noise by waiting 3/5 of last IBI
{
if (Signal < T) // T is the trough
{
T = Signal; // keep track of lowest point in pulse wave
}
}
if(Signal > thresh && Signal > P)
{ // thresh condition helps avoid noise
P = Signal; // P is the peak
} // keep track of highest point in pulse wave
// NOW IT'S TIME TO LOOK FOR THE HEART BEAT
// signal surges up in value every time there is a pulse
if (N > 250)
{ // avoid high frequency noise
if ( (Signal > thresh) && (Pulse == false) && (N > (IBI/5)*3) )
{
```

```
Pulse = true; // set the Pulse flag when we think there is a pulse
digitalWrite(blinkPin,HIGH); // turn on pin 13 LED
IBI = sampleCounter - lastBeatTime; // measure time between beats in mS
lastBeatTime = sampleCounter; // keep track of time for next pulse

if(secondBeat)
{ // if this is the second beat, if secondBeat == TRUE
secondBeat = false; // clear secondBeat flag
for(int i=0; i<=9; i++) // seed the running total to get a realisitic BPM at startup
{
rate[i] = IBI;
}
}
if(firstBeat) // if it's the first time we found a beat, if firstBeat == TRUE
{
firstBeat = false; // clear firstBeat flag
secondBeat = true; // set the second beat flag
sei(); // enable interrupts again
return; // IBI value is unreliable so discard it
}
// keep a running total of the last 10 IBI values
word runningTotal = 0; // clear the runningTotal variable
for(int i=0; i<=8; i++)
{ // shift data in the rate array
rate[i] = rate[i+1]; // and drop the oldest IBI value
runningTotal += rate[i]; // add up the 9 oldest IBI values
}
```

```
rate[9] = IBI; // add the latest IBI to the rate array

runningTotal += rate[9]; // add the latest IBI to runningTotal

runningTotal /= 10; // average the last 10 IBI values

BPM = 60000/runningTotal; // how many beats can fit into a minute? that's BPM!

QS = true; // set Quantified Self flag

// QS FLAG IS NOT CLEARED INSIDE THIS ISR

pulse = BPM;

}

}

if (Signal < thresh && Pulse == true)

{ // when the values are going down, the beat is over

digitalWrite(blinkPin,LOW); // turn off pin 13 LED

Pulse = false; // reset the Pulse flag so we can do it again

amp = P - T; // get amplitude of the pulse wave

thresh = amp/2 + T; // set thresh at 50% of the amplitude

P = thresh; // reset these for next time

T = thresh;

}

if (N > 2500)

{ // if 2.5 seconds go by without a beat

thresh = 512; // set thresh default

P = 512; // set P default

T = 512; // set T default

lastBeatTime = sampleCounter; // bring the lastBeatTime up to date

firstBeat = true; // set these to avoid noise

secondBeat = false; // when we get the heartbeat back

}
```

```
sei(); // enable interrupts when youre done!

}// end isr

void esp_8266()

{

// TCP connection AT+CIPSTART=4,"TCP","184.106.153.149",80

String cmd = "AT+CIPSTART=4,\"TCP\",\"";

cmd += "184.106.153.149"; // api.thingspeak.com

cmd += "\",80";

ser.println(cmd);

Serial.println(cmd);

if(ser.find("Error"))

{

Serial.println("AT+CIPSTART error");

return;

}

String getStr = "GET /update?api_key=";

getStr += apiKey;

getStr +="&field1=";

getStr +=String(temp);

getStr +="&field2=";

getStr +=String(pulse);

getStr += "\r\n\r\n";

// send data length

cmd = "AT+CIPSEND=4,";

cmd += String(getStr.length());

ser.println(cmd);

Serial.println(cmd);
```

```
delay(1000);

ser.print(getStr);

Serial.println(getStr); //thingspeak needs 15 sec delay between updates

delay(3000);

}

void read_temp()

{

int temp_val = analogRead(A1);

float mv = (temp_val/1024.0)*5000;

float cel = mv/10;

temp = (cel*9)/5 + 32;

Serial.print("Temperature:");

Serial.println(temp);

lcd.clear();

lcd.setCursor(0,0);

lcd.print("BPM :");

lcd.setCursor(7,0);

lcd.print(BPM);

lcd.setCursor(0,1);

lcd.print("Temp.:");

lcd.setCursor(7,1);

lcd.print(temp);

lcd.setCursor(13,1);

lcd.print("F");

}
```

CONCLUSION

Remote patient monitoring (RPM) has seen exponential growth in recent years,2 boosted further by the widespread adoption of consumer wellbeing monitoring and its subsequent convergence with long-term healthcare monitoring in home and other non-clinical environments. Initially aimed at exercise monitoring, wellness devices, including fitness tracking, sleep optimization, stress management and home monitoring of elderly and other vulnerable population groups grown in scope to support users in broader lifestyle improvements. More sophisticated systems are also proving to be effective healthcare support tools by using data aggregation from a variety of health and wellbeing devices, combined with analysis and artificial intelligence. The platform helps in managing patients' healthcare goals, such as weight loss, diabetes management or cholesterol reduction, by tracking and analysing factors such as weight, blood pressure, activity, diet and blood glucose levels. The data can access via connected medical and wellbeing devices, including medical and consumer-grade wearables. Not only a data aggregator, it sends messages to the patient throughout the day to manage key health factors and nudge behavior at home, such as prompts to take exercise or maintain a healthier diet. All of this can be monitored and subsequently tailored by the physician.

BIBLIOGRAPHY

The following sites have been used for some contents:

- ✓ Introduction to Iot https://intellipaat.com/blog/future-scope-of-iot/#4
- ✓ https://www.iot-now.com/2020/06/03/103228-5-challenges-still-facing-the-internet-of-things/
- ✓ Wikipedia https://en.wikipedia.org/wiki/Internet_of_things#History
- ✓ https://www.geeksforgeeks.org/