

**IOT based Patient recognition system**

# **Project Report**

*on*

## **IOT based Patient recognition system**

*In partial fulfillment of requirements for the degree*

*of*

**BACHELOR OF TECHNOLOGY  
IN  
INFORMATION TECHNOLOGY**

***Submitted by:***

Saloni Agrawal [18100BTIT03119]

Samarth Maheshwari [18100BTIT03121]

Tarun Basediya [18100BTIT03137]

Yasha Mishra [18100BTIT03156]

***Under the guidance of***

PROF. Rani Singh



**SHRI VAISHNAV VIDYAPEETH VISHWAVIDYALAYA, INDORE  
SHRI VAISHNAV INSTITUTE OF INFORMATIN TECHNOLOGY  
DEPARTMENT OF INFORMATION TECHNOLOGY**

# **IOT based Patient recognition system**

## **ABSTRACT**

With an improvement in technology and miniaturization of sensors, there have been attempts to utilize the new technology in various areas to improve the quality of human life. One main area of research that has seen an adoption of the technology is the healthcare sector.

As a result, this project is an attempt to solve a healthcare problem currently society is facing. The main objective of the project was to design a remote healthcare system. It's comprised of three main parts. The first part being, detection of patient's vitals using sensors, second for sending data to cloud storage and the last part was providing the detected data for remote viewing. Remote viewing of the data enables a doctor or guardian to monitor a patient's health progress away from hospital premises. The Internet of Things (IOT) concepts have been widely used to interconnect the available medical resources and offer smart, reliable, and effective healthcare service to the patients. Health monitoring for active and assisted living is one of the paradigms that can use the IOT advantages to improve the patient's lifestyle. In this project, I have presented an IOT architecture customized for healthcare applications. The aim of the project was to come up with a Remote Health Monitoring System that can be made with locally available sensors with a view to making it affordable if it were to be mass produced.

Hence the proposed architecture collects the sensor data through Arduino Microcontroller and relays it to the cloud where it is processed and analyzed for remote viewing.

## **TABLE OF CONTENT**

### **CHAPTER 1 – INTRODUCTION**

- 1.1 Introduction
- 1.2 Problem Statement
- 1.3 Need for the proper System
- 1.4 Objective
- 1.5 Modules of the system
- 1.6 Scope

### **CHAPTER 2 - LITERATURE SURVEY**

- 2.1 Existing System
- 2.2 Proposed System
- 2.3 Feasibility Study
  - 2.3.1 Technical Feasibility
  - 2.3.2 Economic Feasibility
  - 2.3.3Operational Feasibility

### **CHAPTER 3 – REQUIREMENTS ANALYSIS**

- 3.1 Method used for Requirement analysis
- 3.2 Data Requirements
- 3.3 Functional Requirements
- 3.4 Non Functional Requirements
- 3.5 System Specification
  - 3.5.1 Hardware specification
  - 3.5.2 Software specification

### **Chapter 4- DESIGN**

- 4. 1 Data flow diagram
  - 4.1.1Level-0 DFD
  - 4.1.2Level-1DFD

## **IOT based Patient recognition system**

### **4.2 Use case diagram**

## **Chapter 5- SYSTEM MODELING**

### **5.1 Activity diagram**

#### **5.1.1 Patient activity diagram**

#### **5.1.2 Doctor activity diagram**

### **5.2 Class diagram**

### **5.3Sequence diagram**

### **5.4 Component diagram**

### **5.5 Source code**

### **5.6 Implementation images**

## **Chapter 6- CONCLUSION &FUTURE SCOPE**

### **6.1Limitation of project**

### **6.2Future enhancement**

## **Chapter 7-REFERENCES**

### **7.1References**

## **Chapter 1 - Project Description**

### **1.1 Introduction**

What is a Remote Health Monitoring System?

A Remote health monitoring system is an extension of a hospital medical system where a patient's vital body state can be monitored remotely. Traditionally the detection systems were only found in hospitals and were characterized by huge and complex circuitry which required high power consumption. Continuous advances in the semiconductor technology industry have led to sensors and microcontrollers that are smaller in size, faster in operation, low in power consumption and affordable in cost.

This has further seen development in the remote monitoring of vital life signs of patients especially the elderly. The remote health monitoring system can be applied in the following scenarios:

1. A patient is known to have a medical condition with unstable regulatory body system. This is in cases where a new drug is being introduced to a patient.
2. Critical body organ situation.
3. The situation leading to the development of a risky life-threatening condition. This is for people at an advanced age and maybe having failing health conditions.

In recent times, several systems have come up to address the issue of remote health monitoring. The systems have a wireless detection system that sends the sensor information wirelessly to a remote server. Some even adopted a service model that requires one to pay a subscription fee

### **1.2 Problem Statement**

#### **REMOTE HEALTH MONITORING**

Remote health monitoring can provide useful physiological information in the home. This monitoring is useful for elderly or chronically ill patients who would like to avoid a long hospital stay. Wireless sensors are used to collect and transmit signals of interest and a processor is programmed to receive and automatically analyze the sensor signals. In this project, you are to choose appropriate sensors according to what you would like to detect and design algorithms to realize your detection. Examples are the detection of a fall, monitoring cardiac signals. Using a single parameter monitoring system an approach to a remote health monitoring system was designed that extends healthcare from the traditional clinic or hospital setting to the patient's home. The system was to collect a heartbeat detection system data, fall detection system data, temperature data and few other parameters. The data from the single parameter monitoring systems was then availed for remote detection.

## **IOT based Patient recognition system**

During design the following characteristics of the future medical applications adhered: a) Integration with current trends in medical practices and technology, b) Real-time, long-term, remote monitoring, miniature, wearable sensors and long battery life of a designed device.

### **1.3 Need for the Proper system**

Design a Remote Patient Health Monitoring System (RPHMS) which has heartbeat detection system, pressure detection system, temperature detection system. A doctor or health specialist can use the system to monitor remotely of all vital health parameters of the patient or person of interest.

An attempt at designing a remote healthcare system made with locally available components.

i)The pressure detector, temperature detector- modules comprise of an accelerometer, wireless transmitter and microcontroller. The data collected was transmitted wirelessly to a receiver module.

ii)This consists of a non-invasive infrared finger detector, Liquid Crystal Display (LCD), a designed circuit for cardiac signal detection and microcontroller. The detected analog signal was then digitized to give a digital value that was read on the LCD.

iii)A simple cloud server where hosted with a database for all the vital data to be accessed remotely whenever required.

### **1.4 Objective**

Here the main objective is to design a Remote Patient Health Monitoring System to diagnose the health condition of the patients. Giving care and health assistance to the bedridden patients at critical stages with advanced medical facilities have become one of the major problems in the modern hectic world. In hospitals where many patients whose physical conditions must be monitored frequently as a part of a diagnostic procedure, the need for a cost-effective and fast responding alert mechanism is inevitable. Proper implementation of such systems can provide timely warnings to the medical staffs and doctors and their service can be activated in case of medical emergencies. Present-day systems use sensors that are hardwired to a PC next to the bed.

The use of sensors detects the conditions of the patient and the data is collected and transferred using a microcontroller. In addition to this, use of multiple microcontroller based intelligent system provides high-level applicability in hospitals where many patients must be frequently monitored. For this, here we use the idea of network technology with wireless applicability, providing each patient a unique ID by which the doctor can easily identify the patient and his/her status of health parameters. Using the proposed system, data can be sent wirelessly to the Patient Monitoring System, allowing continuous monitoring of the patient. Contributing accuracy in measurements and providing security in proper alert mechanism give this system a higher level of customer satisfaction and low-cost implementation in hospitals. Physiological monitoring hardware can be easily implemented using simple interfaces of the sensors with a Microcontroller and can effectively be used for healthcare monitoring. This will allow

## **IOT based Patient recognition system**

development of such low-cost devices based on natural human-computer interfaces. The system we proposed here is efficient in monitoring the different physical parameters of many number bedridden patients and then in alerting the concerned medical authorities if these parameters bounce above its predefined critical values. Thus, remote monitoring and control refer to a field of industrial automation that is entering a new era with the development of wireless sensing devices.

### **1.5 Modules of the system**

- 1) Arduino Micro Controller
- 2) Displaying result
- 3) Storing result
- 4) Sharing result via email

### **1.6 Scope**

Many physicians have supported IOT as an effective patient engagement technology because of this widespread adoption. There have been positive impact on reducing patient readmissions, supporting medication adherence and delivering home health support and monitoring, because of IOT devices.

IOT has a long way to go not only in healthcare, but on all the paths it has tread so far and will be traversing in future. Throughout the healthcare industry, its use is not as widespread as it has potential to achieve, but it will come. As the sizes and prices of the devices go down, rapid scaling of these technologies will take place. IOT in healthcare is set to radically change the way healthcare sees the management of inventory, the optimization of workflow and the integration of devices. The digital transformation of healthcare industry, among others, will most certainly be brought about by what is now being called the Internet of Healthcare Things (IOHT). It would be easy for someone to analyze his /her multiple health parameters simply on a single device through internet. It would be reliable because it will be accurate.

# **Chapter 2 – Literature Survey**

## **2.1 Existing System**

In the existing system, we use active network technology to network various sensors to a single PMS. Patients' various critical parameters are continuously monitored via single PMS and reported to the Doctors or Nurses in attendance for timely response in case of critical situations. The sensors are attached to the body of the patients without causing any discomfort to them. In this PMS we monitor the important physical parameters like body temperature, pulse, heart beat rate and blood pressure using the sensors which are readily available. Thus, the analog values that are sensed by the different sensors are then given to a microcontroller attached to it. The microcontroller processes these analog signal values of health parameters separately and converts it to digital values using ADC converter. Now, the digitalized values from more than one microcontroller are sent to the Central PMS. Each of the sensors attached microcontroller with a transceiver will act as a module which has its own unique ID. Each module transmits the data wirelessly to the gateway attached to the PC of the Central PMS. The gateway is attached to the PC i.e. Central PMS which is situated in the medical center, is capable for selecting different patient IDs and allowing the gateway to receive different physical parameter values the patient specified by the ID. The software designed using Graphical User Interface (GUI) can operate on different physical parameters of each patient, consecutively with a specified time interval for each patient.

## **2.2 Proposed System**

The main objective is to design a Patient Monitoring System with two-way communication i.e. not only the patient's data will be sent to the doctor through SMS and email on emergencies, but also the doctor can send required suggestions to the patient or guardians through SMS or Call or Emails. And Patient or guardian can able to track patient's location at any point in time through Google Maps which would enable to send medical services in case of an emergency for non-bed ridden patients. The use of sensors detects the conditions of the patient and the data is collected and transferred using a microcontroller. In addition to this, use of multiple micro-controller based intelligent system provides high-level applicability in hospitals where many patients must be frequently monitored.

## **2.3 Feasibility Study**

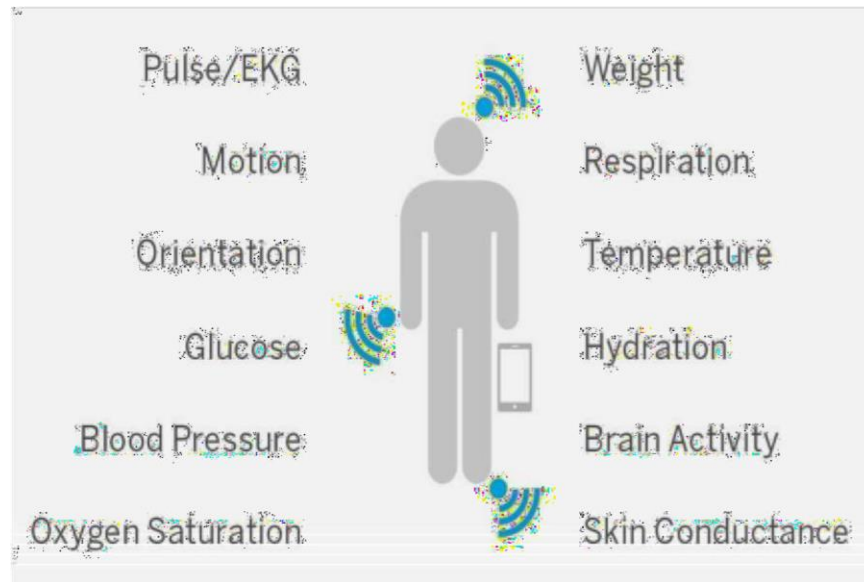
### **2.3.1 Technical Feasibility :**

1. Bio-health Engineering and Mobile Device Technology Advancements With the rise of mobile technology, Wearable Wireless Sensor Networks (WSNs) is an emerging technology which is attached and implemented into human bodies to monitor and collect information related to their physiological parameters, movements and surrounding environment.



## IOT based Patient recognition system

Low power IC and chipset will increase the significant battery life of wearables and this will give a lot more mileage out of wearables to users. Systems must leverage mobile platforms like Android, IOS, Windows and should have smartphone attachments.



**Fig:1**

### 1. Wireless Connectivity (Wi-Fi, BLE, Streaming)

For effective and efficient deployment of remote patient monitoring, wireless connectivity options like Wi-Fi, BLE or infrared are very important factors to consider. As the size of wearable matters, the size of wireless connectivity modules need to be compact as well as efficient enough to deliver connectivity speed and range as per system requirements.

By considering wearables attached to the body, it is not advisable that the device operates at a higher frequency that might harm the human body. Hence it is mandatory for connectivity modules to have the frequency between 10GHz-20GHz and a maximum range of 2 meters.

### 2. Data Security and Device Management Solution

Highly personalized devices will be able to identify if a wrong user is accessing data or devices. Biometrics, either it is a retina scan; heart rate, or finger scan-will help to manage the users accessing data on wearable devices.

## IOT based Patient recognition system

The other most important thing about data security is **device management**. Devices are supposed to manage from both ends, including providers and end users. Providers should provide maintenance on devices, mainly from a device risk point of view.

### 3. Data collection and Analytics

With strong requirement research, providers must decide the data required for their processes and also assess how the device will transmit data, either on a continuous basis or at regular intervals. Apart from just storing and maintaining health-related data, RPM should be able to filter data with accuracy and on different time frames for better [analysis](#) and in turn, provide better care for the patients.

### 4. Standards, Platforms, and Ecosystems

The platforms include a range of available services and functionalities like sensors, smartphones, Wi-Fi gateways, BLE etc. to build secure, scalable and useful remote patient monitoring solutions. Providers need to build a platform that could bring mobile, tablet, cloud, and physiological monitoring technologies together. RPM Ecosystem should extend beyond patient monitoring. By wrapping all the things as mentioned above, a provider can build a system that can go beyond user's expectations especially in terms of health monitoring and easy accessibility.

### 2.3.2 Economical Feasibility

Economic feasibility is a kind of cost-benefit analysis of the examined project, which assesses whether it is possible to implement it. This term means the assessment and analysis of a project's potential to support the decision-making process by objectively and rationally identifying its strengths, weaknesses, opportunities and risks associated with it, the resources that will be needed to implement the project, and an assessment of its chances of success. It consists of market analysis, economic analysis, technical and strategic analysis.

Economic analysis makes it possible to make diagnoses, facilitates decision making, as well as facilitates rationalization of economic processes, both on a macro- and microeconomic scale. In economic analysis, mathematical methods are widely applied (e.g. marginal calculus and linear programming). Analysis is a way of scientific procedure, ordering and dividing the whole into components. The aim of the analysis is to examine the structure of the whole, to get to know the mechanism of connections between the components.

- System Costs:

- Development costs

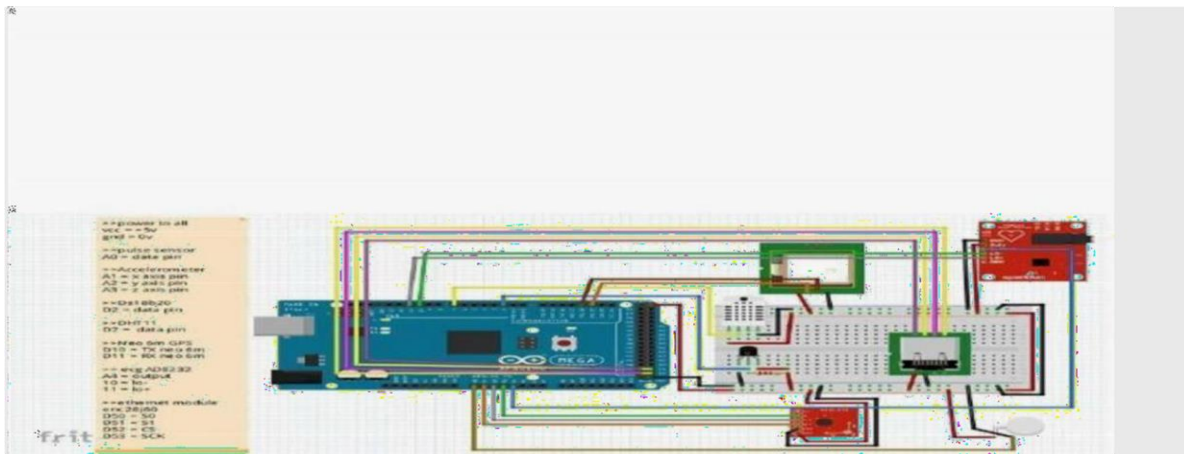
- Arduino Board – ➤ Wifi-module-

- Breadboard- ➤ Jumper wires- ➤ Pulse sensor- ➤ Temperature

## IOT based Patient recognition system

- sensor- ➤ LED-
- LCD Display-
- Production costs
  - Arduino IDE
  - Proteus Simulation
- System Benefits
  - Tangible
    - Reduced operational costs, transaction costs error

### 2.3.3 Operational Feasibility



After power on, one hand finger will touch the H- beat sensor and the LED light will start beat value that LED light will blink. It means the operation is progressing on until power off. When the U-H Beat button press the value will directly send on mobile message, android application as well as web page within 3 seconds Operational Instruction Heart rate known as pulse rate is the number of times a person's beat per minute. Normal heart rate varies from person to person but a normal range for adults is 60 to 100 beats per minute. Also normal heart rate depends on the individual age, body size, heart condition also the person is sitting or moving, medication use and even air temperature. Emotion can vary heart rate for example getting excited, scared can increase the heart rate. According to American Heart Association (AHA) well trained athlete may have a normal heart rate of 40 to 60 beats per minute. There are

4 steps to measure heart rate: 1. Wrist 2. Inside of an elbow 3. Side of the neck 4. Top of the foot How to measure accurate heart rate: Put two fingers over one of these areas and count the number of beats in 60 seconds. Also measure 20 seconds and multiply by three which is easier than first step. Resting heart rate: When a person is in resting mode, it is the best time to measure heartbeat. According (AHA) for adults and older normal heart rate is between 60 and 100 beats per minute (bpm). But below 60 (bpm) doesn't mean the person has health issue problem. Active people have lower heart rates because their muscles don't need to work as hard to maintain a steady beat. Maximum and target heart rate: A person's target heart rate zone is between 50 percent and 85 percent of his or her maximum heart rate. According to (AHA) 30 year old person would be between 50 and 85 percent of his or her max heart rate. [9] Table 2.6: Rate of heartbeat per minute.

## **Chapter 3 – Requirements Analysis**

### **3.1 Functional Requirements.**

#### **R1:- Log-in**

**Description:** Application must have a module for interface using unique credentials of a patient for creating a new channel.

**Input:** E-mail and password to create new channel on ThinkSpeak.

**Output:** User can log-in to channel to analyze Health status

#### **R2:- Display**

**Description:** Application must have a module for interface using unique credentials of a patient for Guardian/Caretaker to monitor patient's vital data.

**Input:** Input is Heart-beat and Temperature of a body sensed by the sensor.

**Output:** Resulted details will be shown

#### **R3:- Location**

**Description:** Hardware must have a GPRS module to fetch location coordinates which can be used to track location of patient.

**Input:** Maps provided with internet connection

**Output:** Patient current location

#### **R4:- Messaging**

**Description:** Hardware must have GSM module which send's SMS alert messages to doctor and guardians upon any emergencies. And application must send email alerts upon any emergencies.

**Input:** Calculated data and internet connection.

**Output:** Result data is send.

## IOT based Patient recognition system

### R5:- Storing

**Description:** Health results are stored and can be accessed on the channel created on ThinkSpeak.

**Input:** Result data.

**Output:** Data is stored in database.

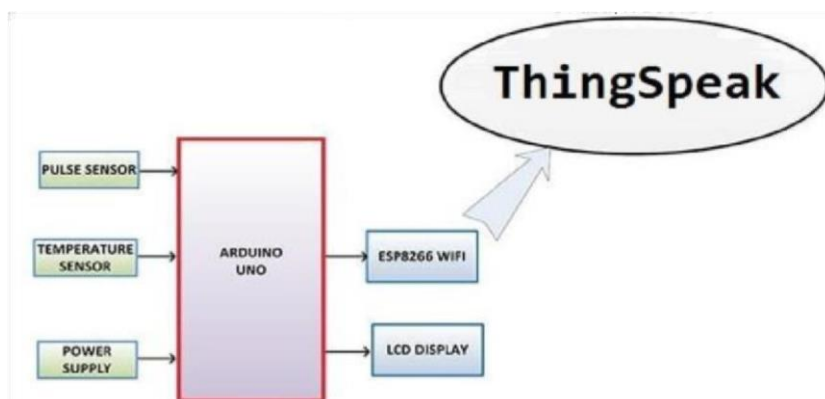
### 3.2 Non- Functional Requirements

Non-functional requirements are not directly related to the functional behavior of the system.

- Web application must be user friendly, simple and interactive.
- The user interface is designed in such way that novice users with little knowledge of web, should be able to access this application.
- Users are required to have some knowledge of using apps.

### 3.3 System Specification

#### 3.3.1 Hardware Specification



## **IOT based Patient recognition system**

- 1) Microcontroller: Arduino Uno Board
- 2)Sensors: Temperature (LM35) sensor, Pulse sensor
- 3)Processor: Pentium IV or higher
- 4)Processor speed: 1.6GHz
- 5)RAM: 512 MB
- 6)Disk Space: 250 MB or higher
- 7)Breadboard
- 8)ESP8266 WiFi module

### **3.3.2 Software Specification**

- 1)Operating System: Windows 7 or higher
- 2) Platform: IOT Cloud (Site- ThingsSpeak)
- 3) IDE: Arduino 1.8.4
- 4) Technologies used: C

## **Chapter 4 – Design**

### **4.1Data flow Diagrams.**

#### **Data Flow Diagrams**

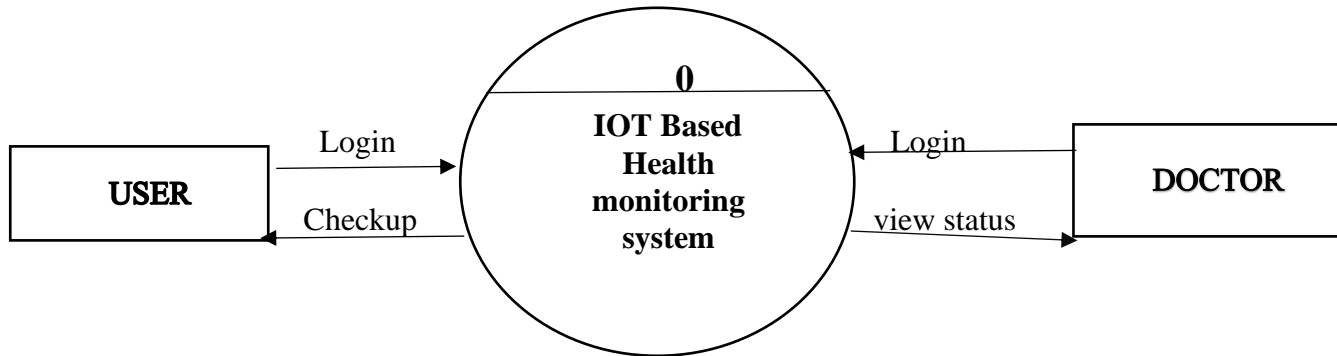
A data flow diagram (DFD) is a graphical representation of the “flow” of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated. In computers, the path of data from source document to

## IOT based Patient recognition system

data entry to processing to final reports. Data changes format and sequence (within a file) as it moves from program to program.

### 4.1.1 Level-0 DFD

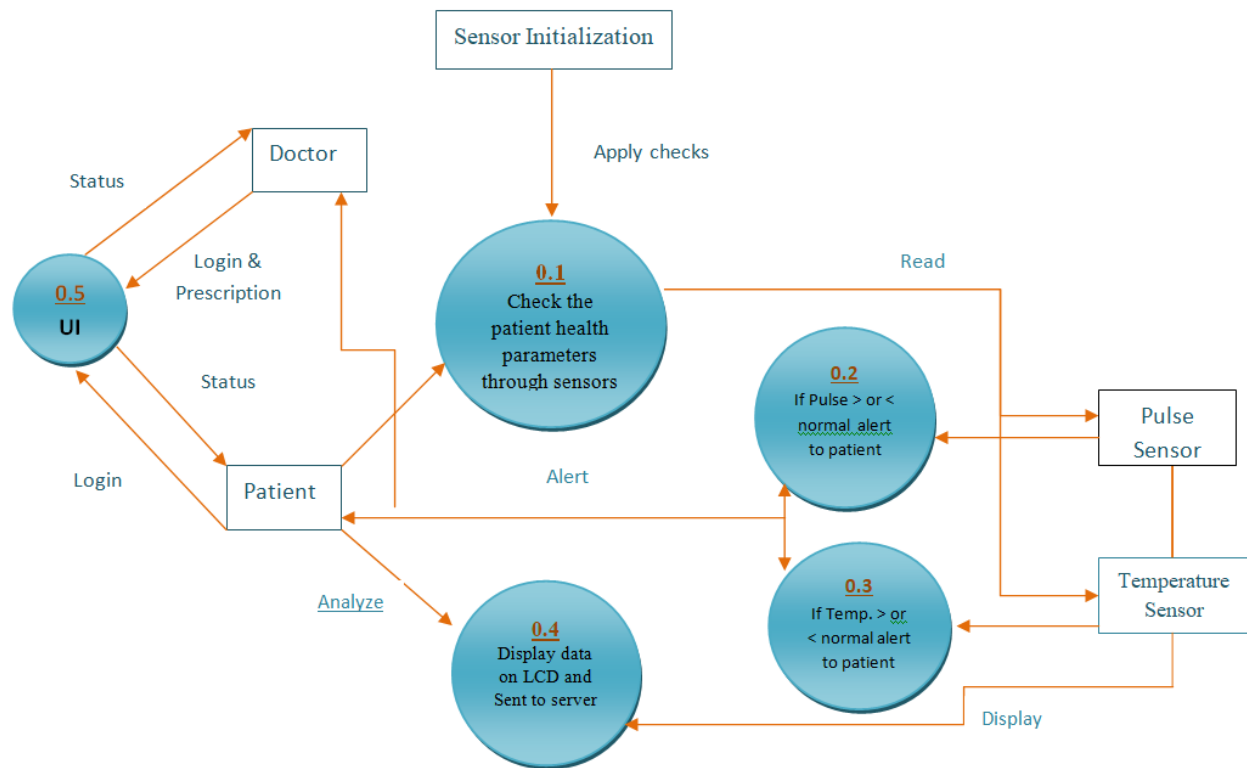
Level-0



### 4.1.2 Level-1 DFD

Level-1

## IOT based Patient recognition system



### 4.2USE CASE DIAGRAM

A use case diagram is used to represent the dynamic behavior of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. It models the tasks, services, and

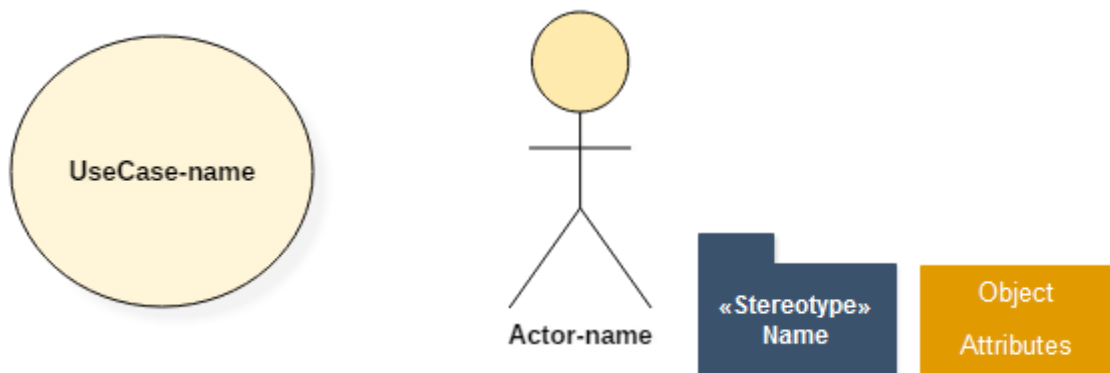


## IOT based Patient recognition system

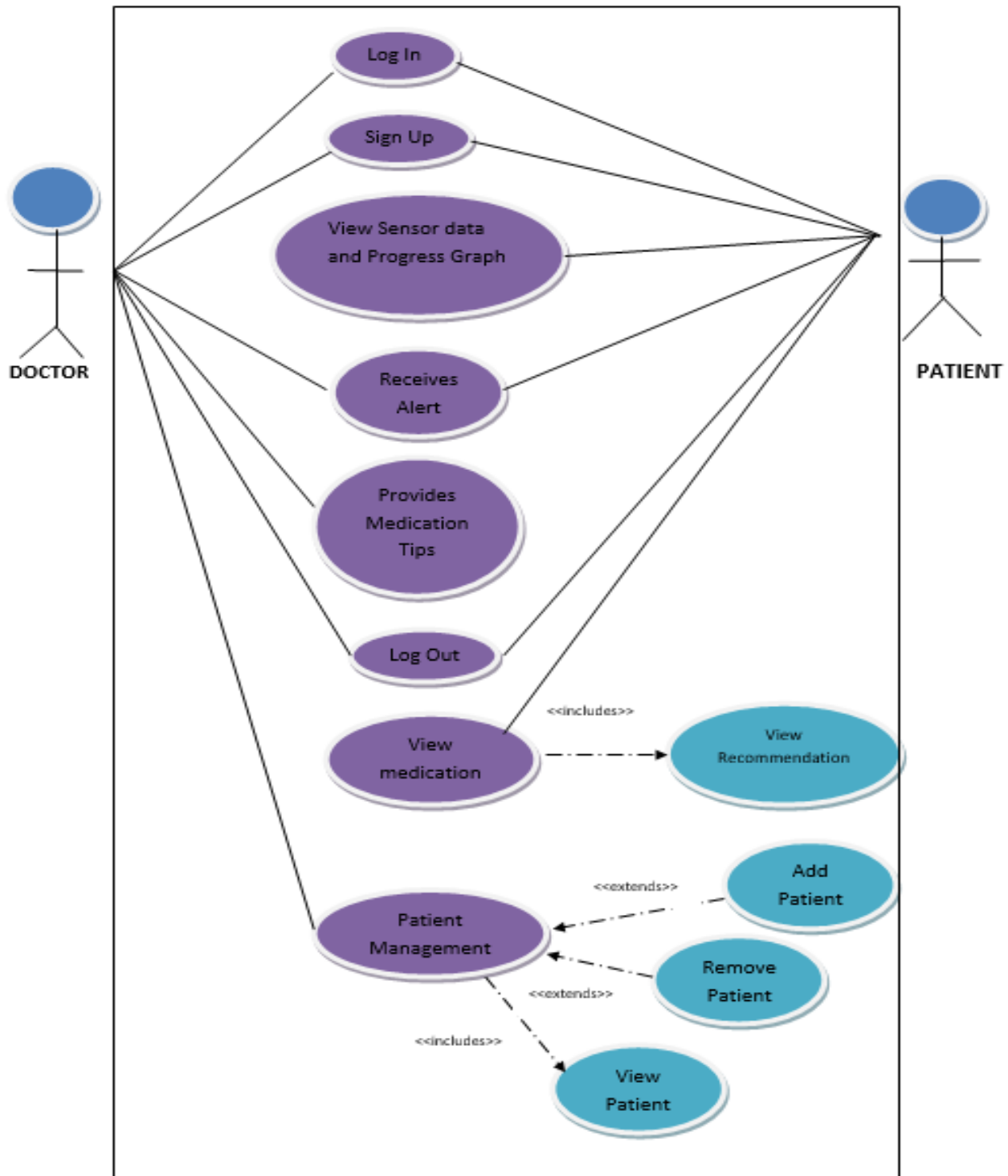
functions required by a system/subsystem of an application. It depicts the high-level functionality of a system and also tells how the user handles a system.

The **purpose** of a **use case diagram** is to capture the dynamic aspect of a system. They provide a simplified graphical representation of what the system should do in a **use case**. Further **diagrams** and documentation are needed for a complete functional and technical outlook on the system.

- **Use cases:** Horizontally shaped ovals that represent the different uses that a user might have.
- **Actors:** Stick figures that represent the people actually employing the use cases.
- **Associations:** A line between actors and use cases. In complex diagrams, it is important to know which actors are associated with which use cases.
- **System boundary boxes:** A box that sets a system scope to use cases. All use cases outside the box would be considered outside the scope of that system. For example, Psycho Killer is outside the scope of occupations in the chainsaw example found below.
- **Packages:** A UML shape that allows you to put different elements into groups. Just as with component diagrams, these groupings are represented as file folders.
- **System Domain Box :** It indicates the boundary of the system you want to highlight or focus on.



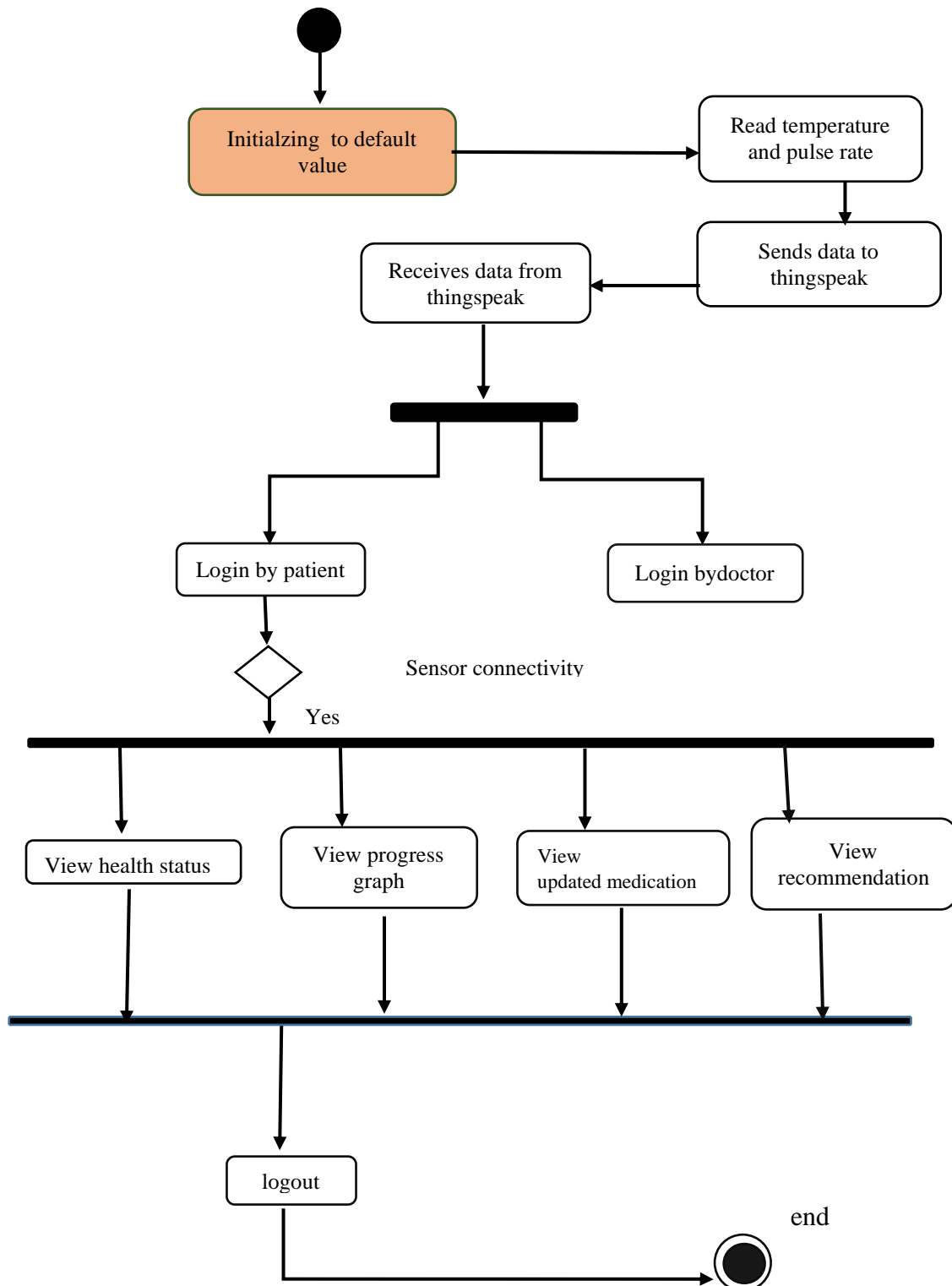
## IOT based Patient recognition system



Use case diagram

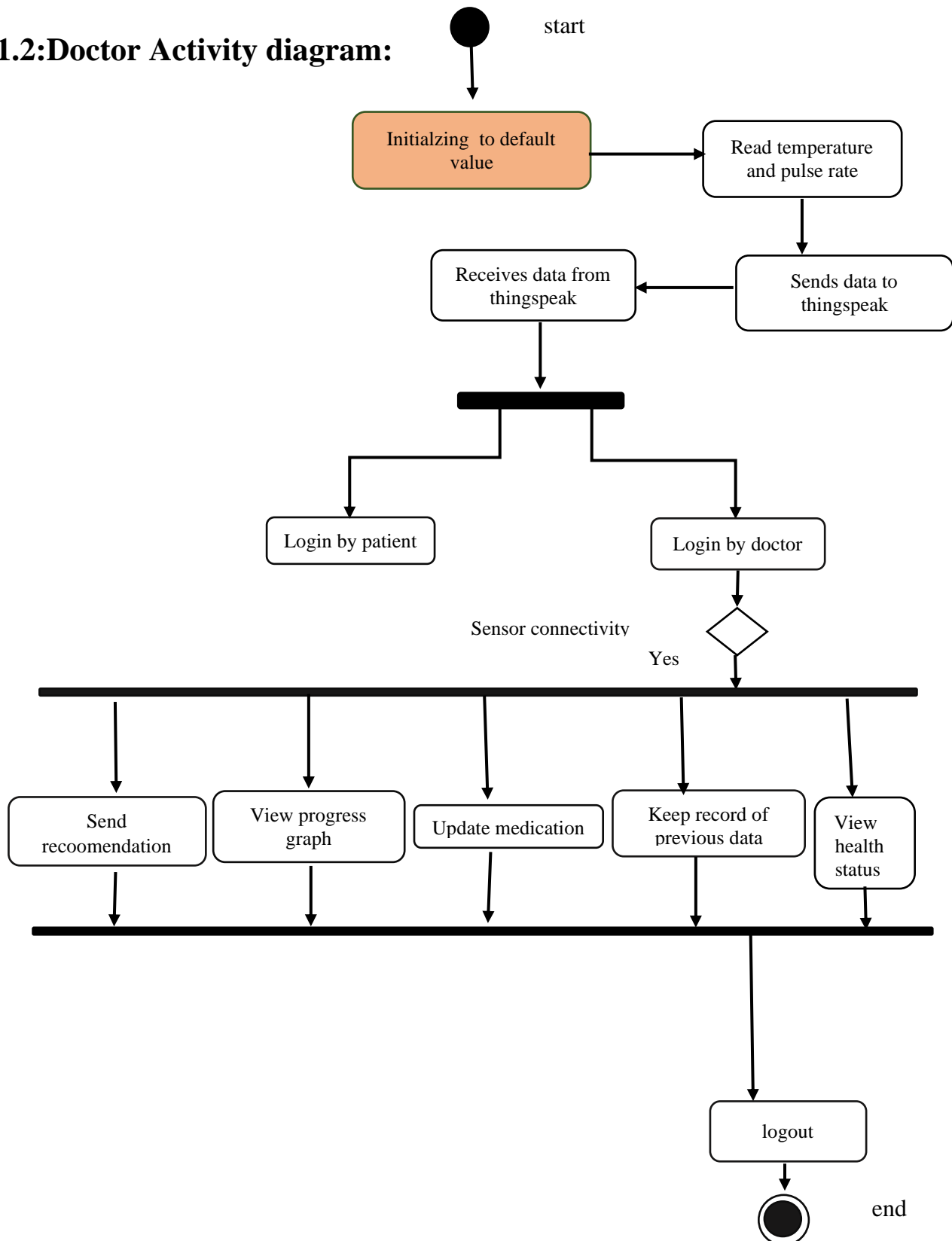
## Chapter 5 – System modeling

### 5.1 ACTIVITY DIAGRAM :5.1.1:patient activity diagram:



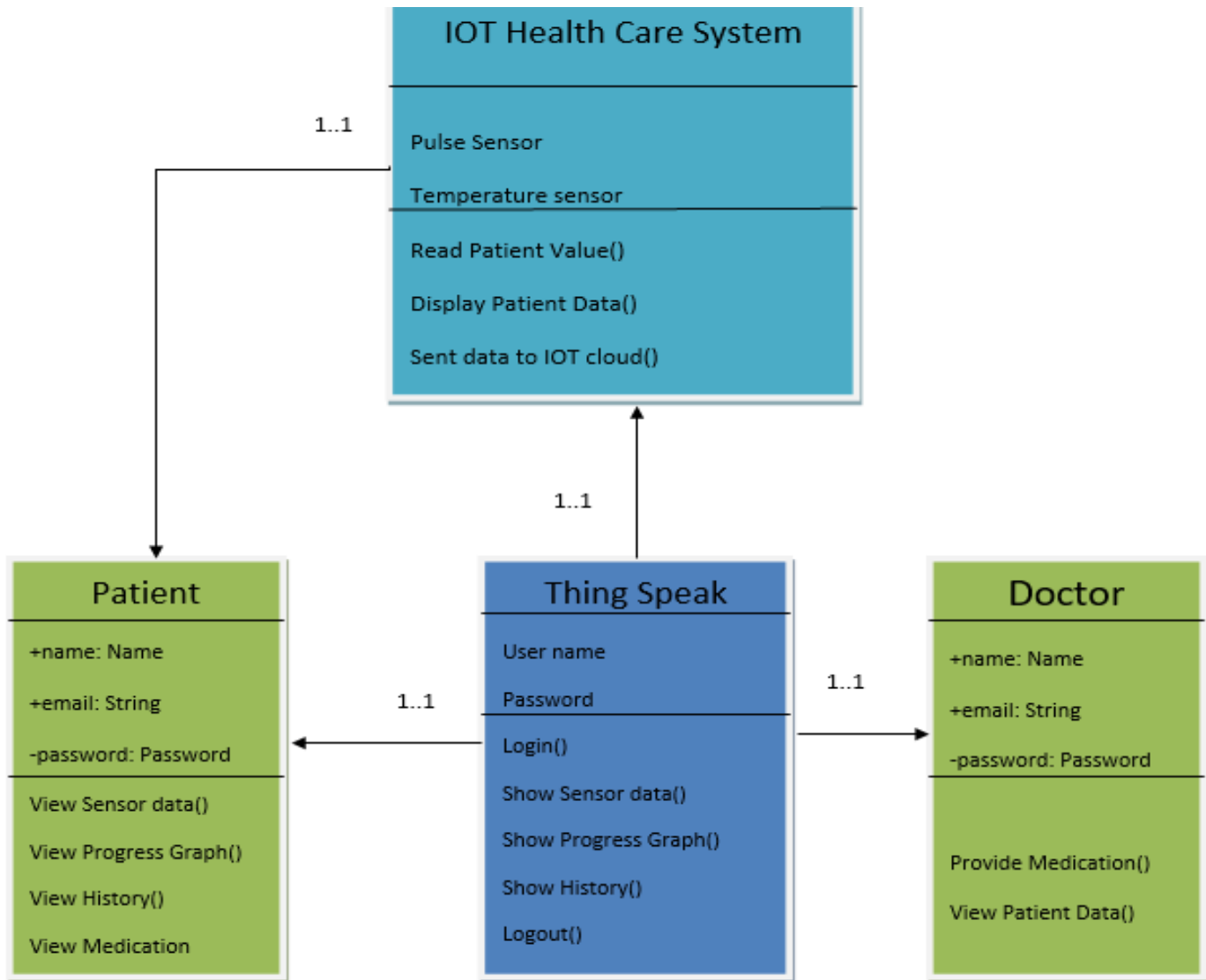
## IOT based Patient recognition system

### 5.1.2: Doctor Activity diagram:



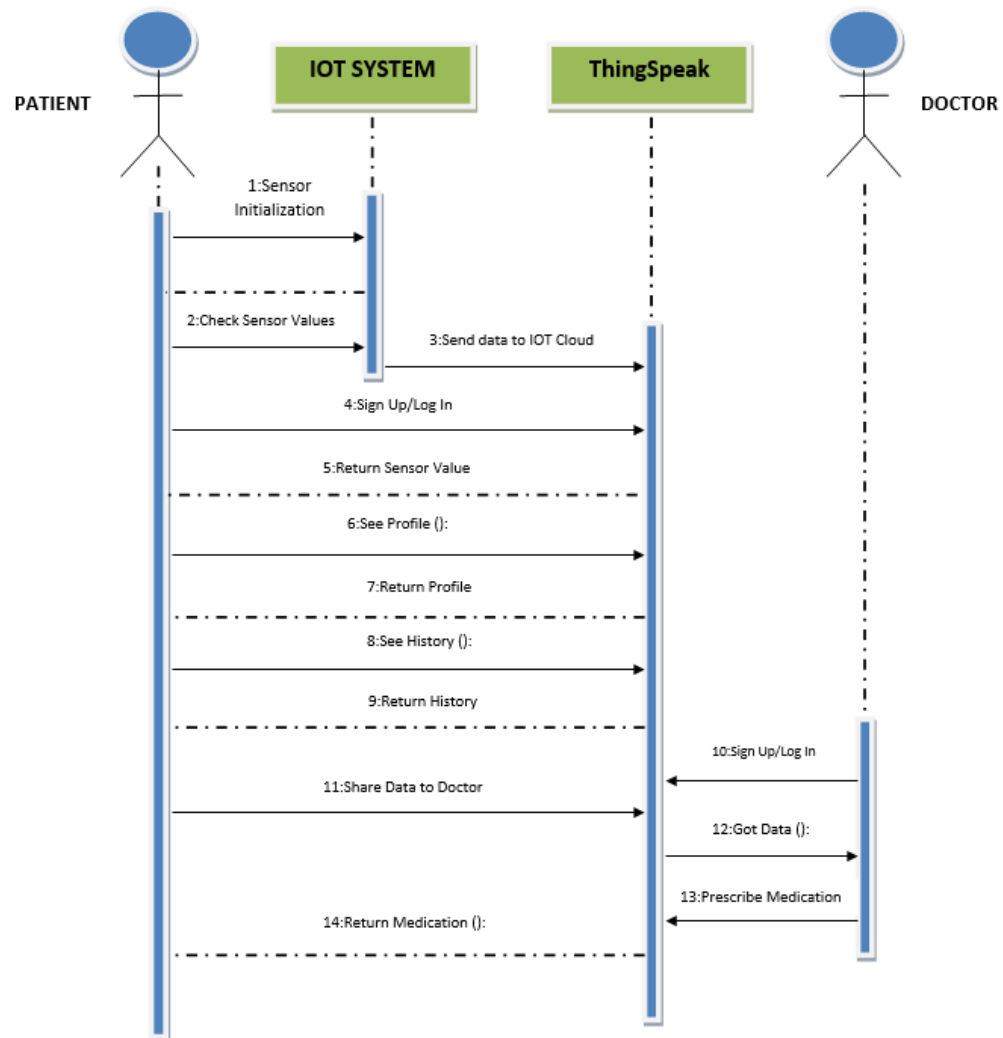
## IOT based Patient recognition system

### 5.2 CLASS DIAGRAM



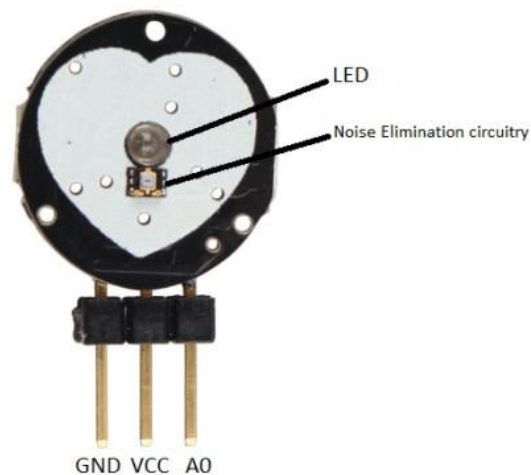
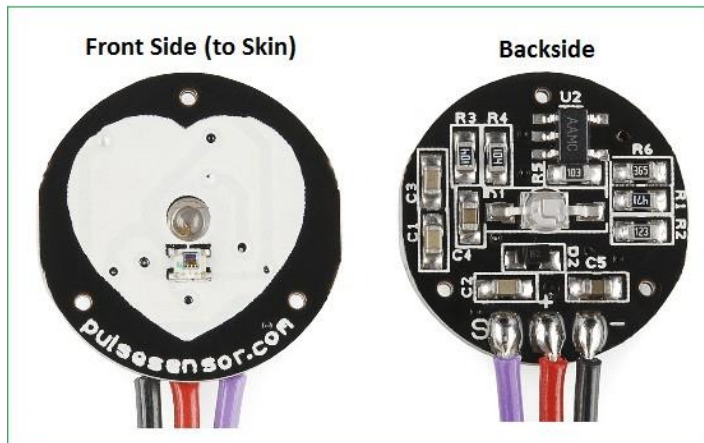
## IOT based Patient recognition system

### 5.3 SEQUENCE DIAGRAM:



### 5.4 COMPONENTS DIAGRAM

#### Pulse Sensor:



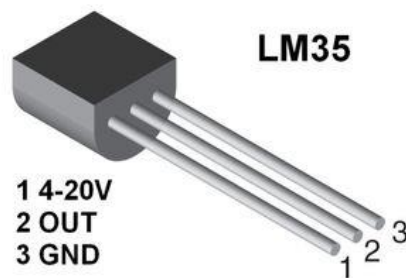
The **Pulse Sensor** is a plug-and-play **heart-rate sensor for Arduino**. It can be used by students, artists, athletes, makers, and game & mobile developers who want to easily incorporate live heart-rate data into their projects. The essence is an integrated optical amplifying circuit and noise eliminating 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 center 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.

## IOT based Patient recognition system

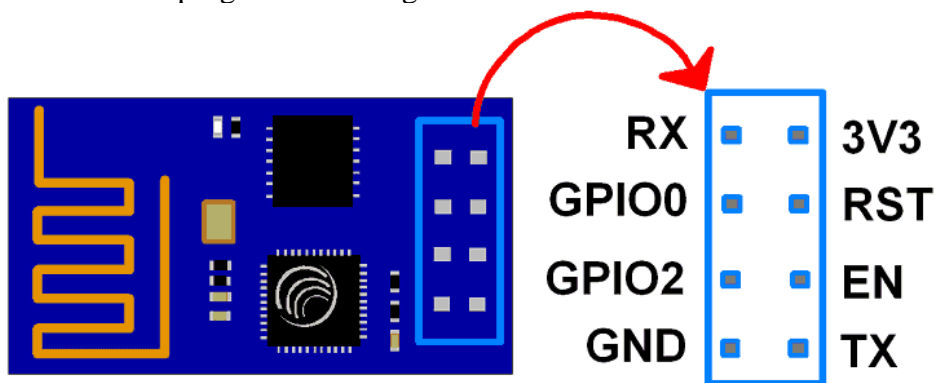
### LM35 Temperature Sensor:

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  $\pm 3/4^{\circ}\text{C}$  over a full  $-55^{\circ}\text{C}$  to  $150^{\circ}\text{C}$  temperature range.



### 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



## IOT based Patient recognition system

**Pin 4 : CH\_EN:** Chip Enable/Active high

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

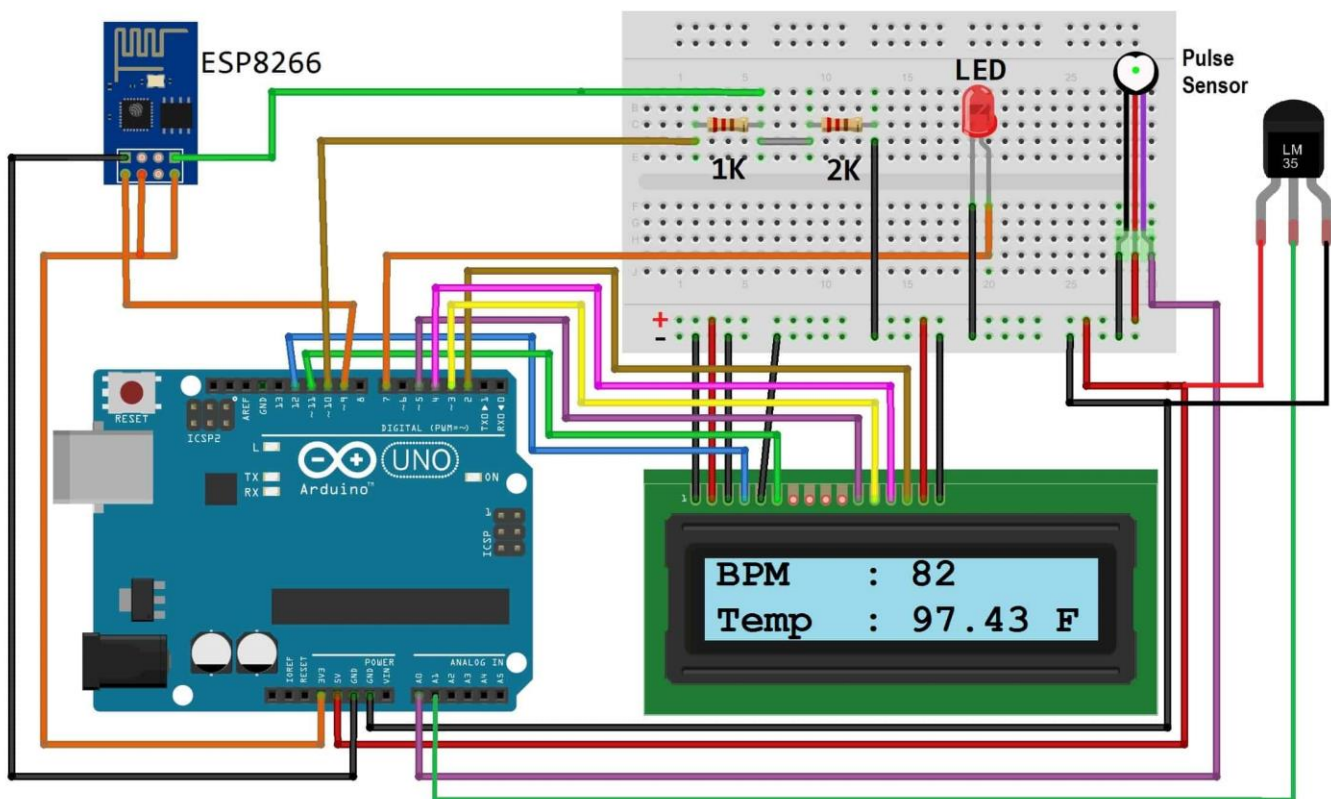
**Pin 6 : Reset:** Resets the module

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

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

### 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.

## IOT based Patient recognition system

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.

### SOFTWARE: ARDUINO SOFTWARE (IDE)



The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java. It is an official Arduino software, making code compilation too easy that even a common person with no prior technical knowledge can get their feet wet with the learning process. This IDE comes with inbuilt functions and commands that play a vital role for debugging, editing and compiling the code in the environment.

### Setting the 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

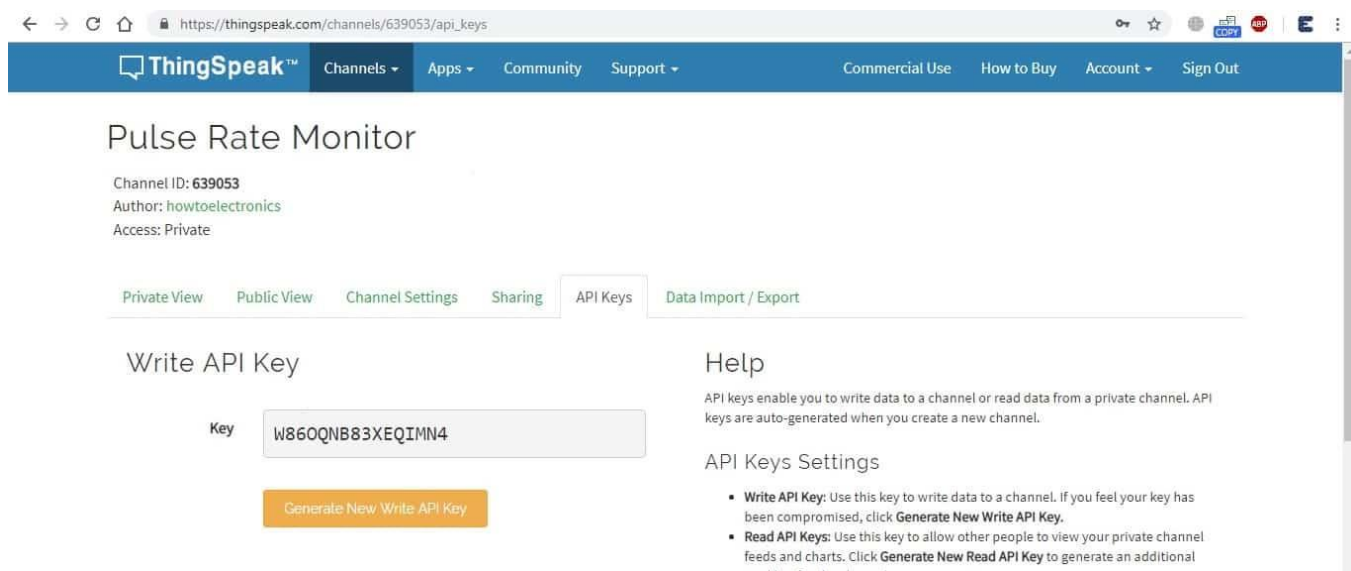
## IOT based Patient recognition system

an account.



Then create a new channel and set up what you want. The tutorial in the video below. Follow the video for more clarification.

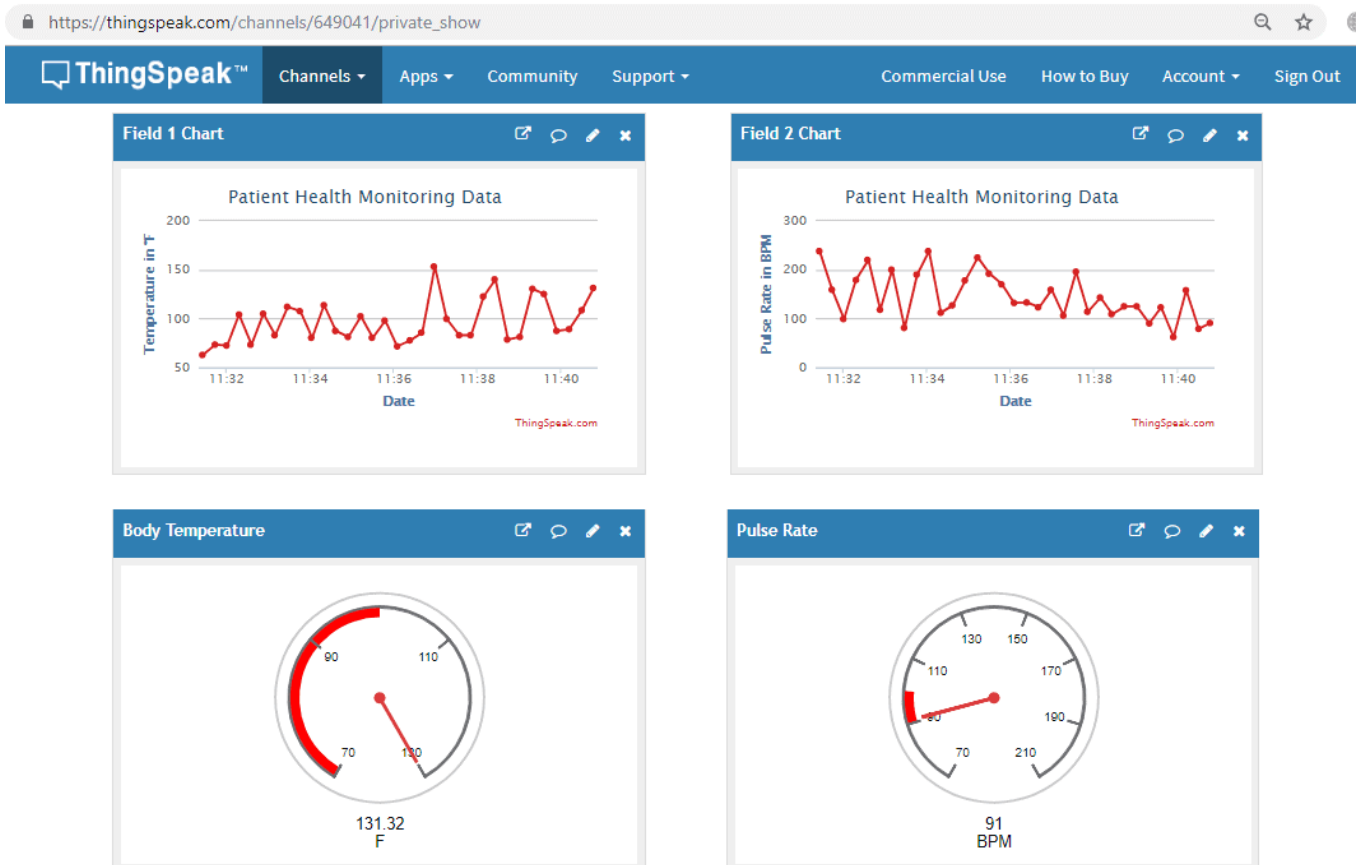
Then create the **API keys**. This key is required for programming modifications and setting your data.



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.

# IOT based Patient recognition system



## IOT based Patient recognition system

### 5.5 SOURCE CODE

```
1  #include <LiquidCrystal.h>
2  LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
3  #include <SoftwareSerial.h>
4  float pulse = 0;
5  float temp = 0;
6  SoftwareSerial ser(9,10);
7  String apiKey = "00707TGA1BLUNN12";
8
9  // Variables
10 int pulsePin = A0; // Pulse Sensor purple wire connected to analog pin 0
11 int blinkPin = 7 ; // pin to blink led at each beat
12 int fadePin = 13; // pin to do fancy classy fading blink at each beat
13 int fadeRate = 0; // used to fade LED on with PWM on fadePin
14
15 // Volatile Variables, used in the interrupt service routine!
16
17 volatile int BPM; // int that holds raw Analog in 0. updated every 2mS
18 volatile int Signal; // holds the incoming raw data
19 volatile int IBI = 600; // int that holds the time interval between beats! Must be seeded!
20 volatile boolean Pulse = false; // "True" when User's live heartbeat is detected. "False" when not a "live beat".
21 volatile boolean QS = false; // becomes true when Arduino finds a beat.
22
23 // Regards Serial OutPut -- Set This Up to your needs
24 static boolean serialVisual = true; // Set to 'false' by Default. Re-set to 'true' to see Arduino Serial Monitor ASCII Visual Pulse
25 volatile int rate[10]; // array to hold last ten IBI values
26 volatile unsigned long sampleCounter = 0; // used to determine pulse timing
27 volatile unsigned long lastBeatTime = 0; // used to find IBI
28 volatile int P = 512; // used to find peak in pulse wave, seeded
29 volatile int T = 512; // used to find trough in pulse wave, seeded
30 volatile int thresh = 525; // used to find instant moment of heart beat, seeded
31 volatile int amp = 100; // used to hold amplitude of pulse waveform, seeded
32 volatile boolean firstBeat = true; // used to seed rate array so we startup with reasonable BPM
33 volatile boolean secondBeat = false; // used to seed rate array so we startup with reasonable BPM
34
35 void setup()
36 {
37   lcd.begin(16, 2);
38   pinMode(blinkPin, OUTPUT); // pin that will blink to your heartbeat!
39   pinMode(fadePin, OUTPUT); // pin that will fade to your heartbeat!
40   Serial.begin(115200); // we agree to talk fast!
41   interruptSetup(); // sets up to read Pulse Sensor signal every 2mS
42
43   // IF YOU ARE POWERING The Pulse Sensor AT VOLTAGE LESS THAN THE BOARD VOLTAGE,
44
45   // UN-COMMENT THE NEXT LINE AND APPLY THAT VOLTAGE TO THE A-REF PIN
46
47   // analogReference(EXTERNAL);
48
49   lcd.clear();
50   lcd.setCursor(0,0);
51   lcd.print(" Patient Health");
52   lcd.setCursor(0,1);
53   lcd.print(" Monitoring ");
54   delay(4000);
55   lcd.clear();
56   lcd.setCursor(0,0);
57   lcd.print("Initializing....");
```

## IOT based Patient recognition system

```
58 delay(5000);
59 lcd.clear();
60 lcd.setCursor(0,0);
61 lcd.print("Getting Data....");
62 ser.begin(9600);
63 ser.println("AT");
64 delay(1000);
65 ser.println("AT+GMR");
66 delay(1000);
67 ser.println("AT+CWMODE=3");
68 delay(1000);
69 ser.println("AT+RST");
70 delay(5000);
71 ser.println("AT+CIPMUX=1");
72 delay(1000);
73
74 String cmd="AT+CWJAP=\"Alexahome\", \"98765432\"";
75 ser.println(cmd);
76 delay(1000);
77 ser.println("AT+CIFSR");
78 delay(1000);
79 }
80
81 // Where the Magic Happens
82 void loop()
83 {
84   serialOutput();
85   if (QS == true) // A Heartbeat Was Found
86   {
87
88     // BPM and IBI have been Determined
89     // Quantified Self "QS" true when arduino finds a heartbeat
90     fadeRate = 255; // Makes the LED Fade Effect Happen, Set 'fadeRate' Variable to 255 to fade LED with pulse
91     serialOutputWhenBeatHappens(); // A Beat Happened, Output that to serial.
92     QS = false; // reset the Quantified Self flag for next time
93   }
94   ledFadeToBeat(); // Makes the LED Fade Effect Happen
95   delay(20); // take a break
96   read_temp();
97   esp_8266();
98 }
99 void ledFadeToBeat()
100 {
101   fadeRate -= 15; // set LED fade value
102   fadeRate = constrain(fadeRate,0,255); // keep LED fade value from going into negative numbers!
103   analogWrite(fadePin,fadeRate); // fade LED
104 }
105 void interruptSetup()
106 {
107   // Initializes Timer2 to throw an interrupt every 2mS.
108   TCCR2A = 0x02; // DISABLE PWM ON DIGITAL PINS 3 AND 11, AND GO INTO CTC MODE
109   TCCR2B = 0x06; // DON'T FORCE COMPARE, 256 PRESCALER
110   OCR2A = 0x7C; // SET THE TOP OF THE COUNT TO 124 FOR 500Hz SAMPLE RATE
111   TIMSK2 = 0x02; // ENABLE INTERRUPT ON MATCH BETWEEN TIMER2 AND OCR2A
112   sei(); // MAKE SURE GLOBAL INTERRUPTS ARE ENABLED
113 }
114 void serialOutput()
115 { // Decide How To Output Serial.
116   if (serialVisual == true)
117   {
```

## IOT based Patient recognition system

```
118 arduinoSerialMonitorVisual('-', Signal); // goes to function that makes Serial Monitor Visualizer
119 }
120 else
121 {
122 sendDataToSerial('S', Signal); // goes to sendDataToSerial function
123 }
124 }
125 void serialOutputWhenBeatHappens()
126 {
127 if (serialVisual == true) // Code to Make the Serial Monitor Visualizer Work
128 {
129 Serial.print("*** Heart-Beat Happened *** "); //ASCII Art Madness
130 Serial.print("BPM: ");
131 Serial.println(BPM);
132 }
133 else
134 {
135 sendDataToSerial('B',BPM); // send heart rate with a 'B' prefix
136 sendDataToSerial('Q',IBI); // send time between beats with a 'Q' prefix
137 }
138 }
139 void arduinoSerialMonitorVisual(char symbol, int data )
140 {
141 const int sensorMin = 0; // sensor minimum, discovered through experiment
142 const int sensorMax = 1024; // sensor maximum, discovered through experiment
143 int sensorReading = data; // map the sensor range to a range of 12 options:
144 int range = map(sensorReading, sensorMin, sensorMax, 0, 11);
145 // do something different depending on the
146 // range value:
147 switch (range)
148 {
149 case 0:
150 Serial.println(""); //ASCII Art Madness
151 break;
152 case 1:
153 Serial.println("---");
154 break;
155 case 2:
156 Serial.println("-----");
157 break;
158 case 3:
159 Serial.println("-----");
160 break;
161 case 4:
162 Serial.println("-----");
163 break;
164 case 5:
165 Serial.println("-----|-");
166 break;
167 case 6:
168 Serial.println("-----|---");
169 break;
170 case 7:
171 Serial.println("-----|-----");
172 break;
173 case 8:
174 Serial.println("-----|-----");
175 break;
176 case 9:
177 Serial.println("-----|-----");
```

## IOT based Patient recognition system

```
178 break;
179 case 10:
180 Serial.println("-----|-----");
181 break;
182 case 11:
183 Serial.println("-----|-----");
184 break;
185 }
186 }
187
188 void sendDataToSerial(char symbol, int data )
189 {
190 Serial.print(symbol);
191 Serial.println(data);
192 }
193 ISR(TIMER2_COMPA_vect) //triggered when Timer2 counts to 124
194 {
195 cli(); // disable interrupts while we do this
196 Signal = analogRead(pulsePin); // read the Pulse Sensor
197 sampleCounter += 2; // keep track of the time in mS with this variable
198 int N = sampleCounter - lastBeatTime; // monitor the time since the last beat to avoid noise
199 // find the peak and trough of the pulse wave
200
201 if(Signal < thresh && N > (IBI/5)*3) // avoid dichrotic noise by waiting 3/5 of last IBI
202 {
203 if (Signal < T) // T is the trough
204 {
205 T = Signal; // keep track of lowest point in pulse wave
206 }
207 }
208 if(Signal > thresh && Signal > P)
209 { // thresh condition helps avoid noise
210 P = Signal; // P is the peak
211 } // keep track of highest point in pulse wave
212 // NOW IT'S TIME TO LOOK FOR THE HEART BEAT
213 // signal surges up in value every time there is a pulse
214 if (N > 250)
215 { // avoid high frequency noise
216 if ( (Signal > thresh) && (Pulse == false) && (N > (IBI/5)*3) )
217 {
218 Pulse = true; // set the Pulse flag when we think there is a pulse
219 digitalWrite(blinkPin,HIGH); // turn on pin 13 LED
220 IBI = sampleCounter - lastBeatTime; // measure time between beats in mS
221 lastBeatTime = sampleCounter; // keep track of time for next pulse
222
223 if(secondBeat)
224 { // if this is the second beat, if secondBeat == TRUE
225 secondBeat = false; // clear secondBeat flag
226 for(int i=0; i<=9; i++) // seed the running total to get a realistic BPM at startup
227 {
228 rate[i] = IBI;
229 }
230 }
231 if(firstBeat) // if it's the first time we found a beat, if firstBeat == TRUE
232 {
233 firstBeat = false; // clear firstBeat flag
234 secondBeat = true; // set the second beat flag
235 sei(); // enable interrupts again
236 return; // IBI value is unreliable so discard it
237 }
```



## IOT based Patient recognition system

```
238 // keep a running total of the last 10 IBI values
239 word runningTotal = 0; // clear the runningTotal variable
240 for(int i=0; i<=8; i++)
241 { // shift data in the rate array
242   rate[i] = rate[i+1]; // and drop the oldest IBI value
243   runningTotal += rate[i]; // add up the 9 oldest IBI values
244 }
245 rate[9] = IBI; // add the latest IBI to the rate array
246 runningTotal += rate[9]; // add the latest IBI to runningTotal
247 runningTotal /= 10; // average the last 10 IBI values
248 BPM = 60000/runningTotal; // how many beats can fit into a minute? that's BPM!
249 QS = true; // set Quantified Self flag
250 // QS FLAG IS NOT CLEARED INSIDE THIS ISR
251 pulse = BPM;
252 }
253 }
254 if (Signal < thresh && Pulse == true)
255 { // when the values are going down, the beat is over
256   digitalWrite(blinkPin,LOW); // turn off pin 13 LED
257   Pulse = false; // reset the Pulse flag so we can do it again
258   amp = P - T; // get amplitude of the pulse wave
259   thresh = amp/2 + T; // set thresh at 50% of the amplitude
260   P = thresh; // reset these for next time
261   T = thresh;
262 }
263 if (N > 2500)
264 { // if 2.5 seconds go by without a beat
265   thresh = 512; // set thresh default
266   P = 512; // set P default
267   T = 512; // set T default
268   lastBeatTime = sampleCounter; // bring the lastBeatTime up to date
269   firstBeat = true; // set these to avoid noise
270   secondBeat = false; // when we get the heartbeat back
271 }
272 sei(); // enable interrupts when youre done!
273 } // end isr
274 void esp_8266()
275 {
276   // TCP connection AT+CIPSTART=4,"TCP","184.106.153.149",80
277   String cmd = "AT+CIPSTART=4,\"TCP\", \"\"";
278   cmd += "184.106.153.149"; // api.thingspeak.com
279   cmd += "\",80";
280   ser.println(cmd);
281   Serial.println(cmd);
282   if(ser.find("Error"))
283   {
284     Serial.println("AT+CIPSTART error");
285     return;
286   }
287   String getStr = "GET /update?api_key=";
288   getStr += apiKey;
289   getStr += "&field1=";
290   getStr += String(temp);
291   getStr += "&field2=";
292   getStr += String(pulse);
293   getStr += "\r\n\r\n";
294   // send data length
295   cmd = "AT+CIPSEND=4, ";
296   cmd += String(getStr.length());
297   ser.println(cmd);
```

# IOT based Patient recognition system

```
298 Serial.println(cmd);
299 delay(1000);
300 ser.print(getStr);
301 Serial.println(getStr); //thingspeak needs 15 sec delay between updates
302 delay(3000);
303 }
304 void read_temp()
305 {
306 int temp_val = analogRead(A1);
307 float mv = (temp_val/1024.0)*5000;
308 float cel = mv/10;
309 temp = (cel*9)/5 + 32;
310 Serial.print("Temperature:");
311 Serial.println(temp);
312 lcd.clear();
313 lcd.setCursor(0,0);
314 lcd.print("BPM :");
315 lcd.setCursor(7,0);
316 lcd.print(BPM);
317 lcd.setCursor(0,1);
318 lcd.print("Temp.");
319 lcd.setCursor(7,1);
320 lcd.print(temp);
321 lcd.setCursor(13,1);
322 lcd.print("F");
323 }
```

## Screenshots:



# IOT based Patient recognition system

HEALTH\_MONITORING | Arduino 1.8.13

File Edit Sketch Tools Help

```
HEALTH_MONITORING
//-----
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);
}
```

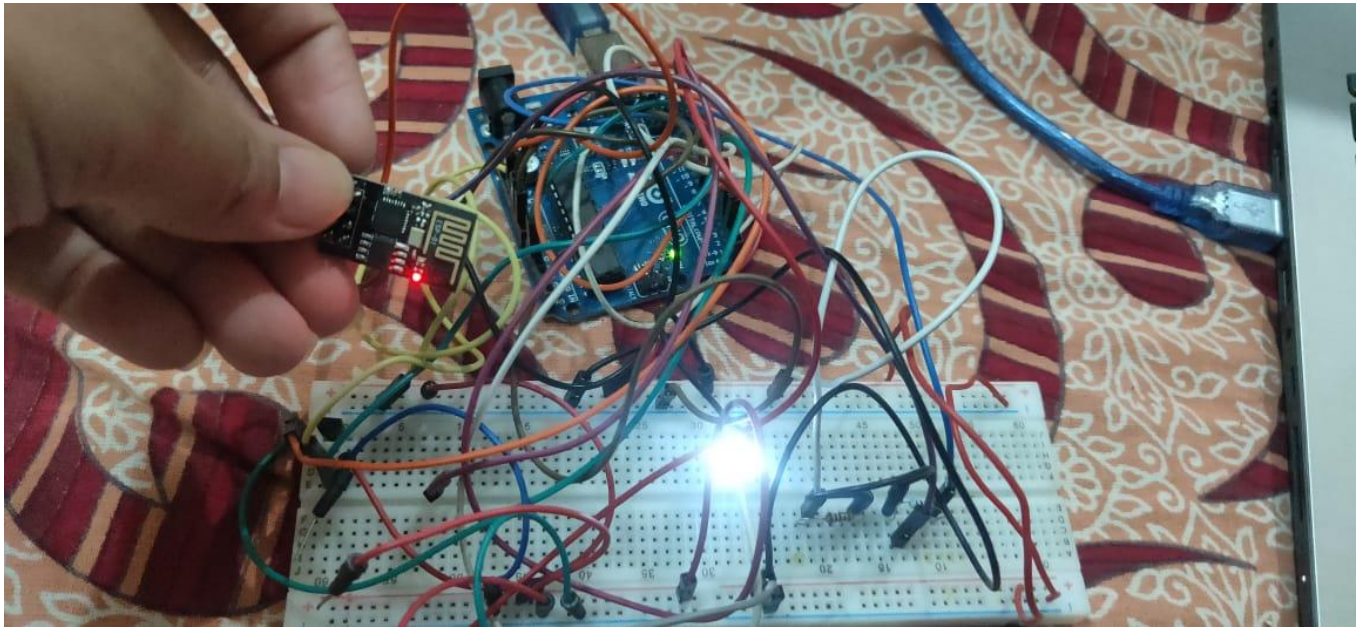
HEALTH\_MONITORING | Arduino 1.8.13

File Edit Sketch Tools Help

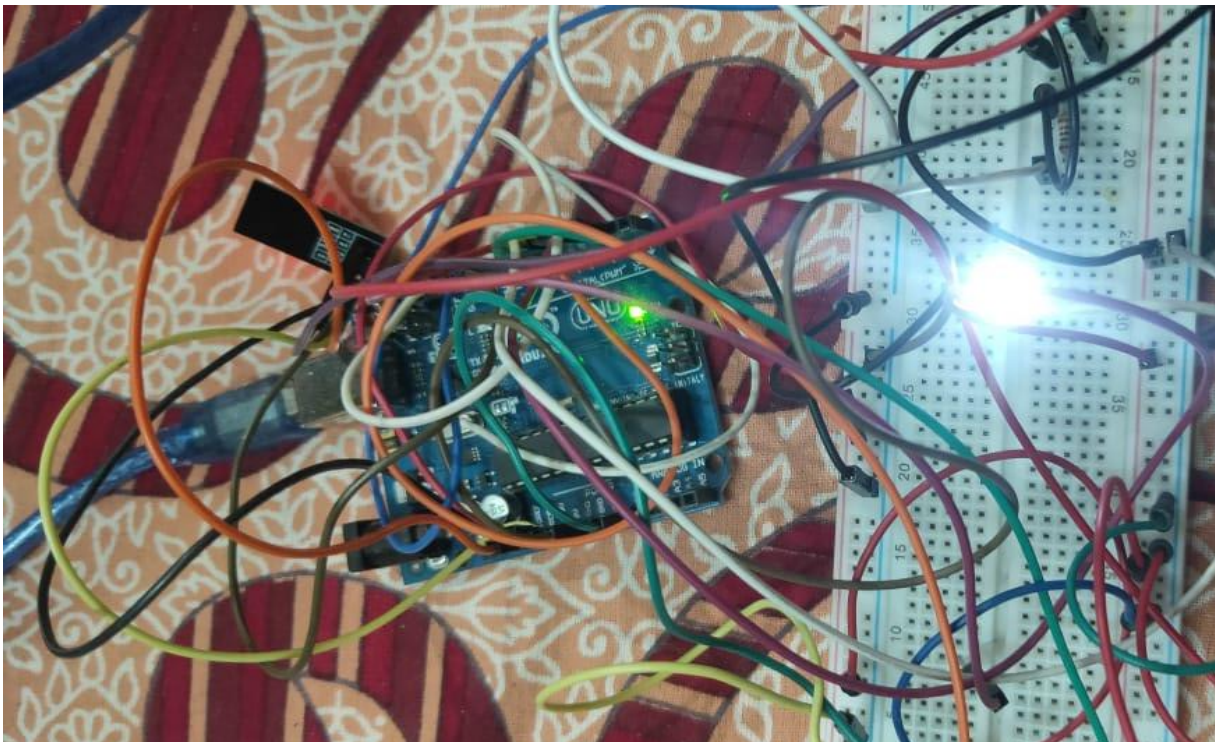
```
HEALTH_MONITORING
Serial.println("-----");
break;
case 5:
Serial.println("-----|");
break;
case 6:
Serial.println("-----|---");
break;
case 7:
Serial.println("-----|-----");
break;
case 8:
Serial.println("-----|-----");
break;
case 9:
Serial.println("-----|-----");
break;
case 10:
Serial.println("-----|-----");
break;
case 11:
Serial.println("-----|-----");
break;
}
```

## IOT based Patient recognition system

### 5.6 Implementation images:



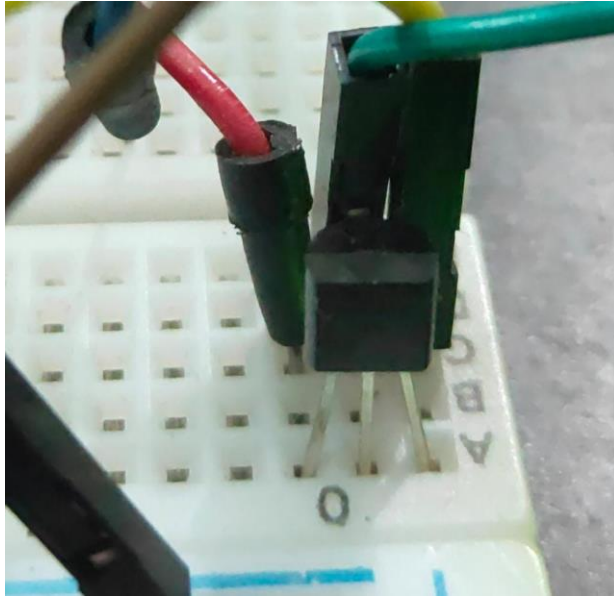
**Wifi module**



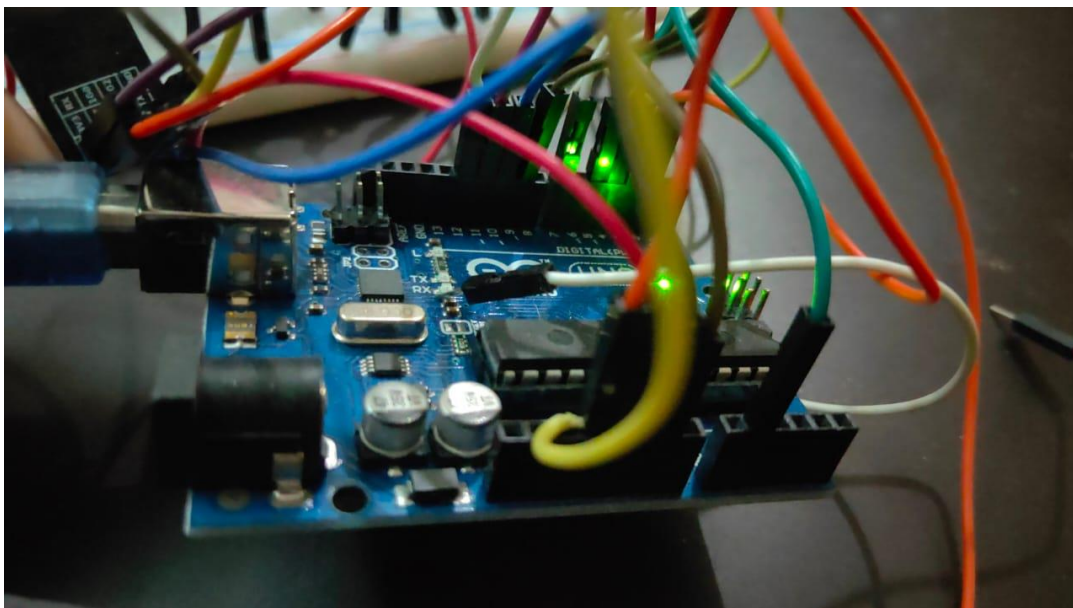


## IOT based Patient recognition system

### Aurdino and bread board

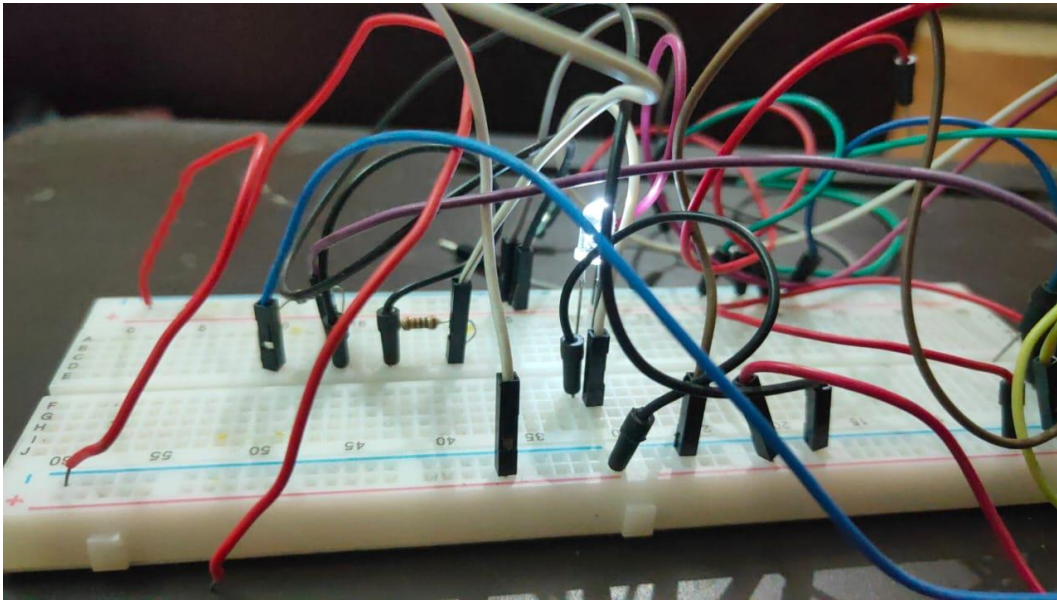


Temperature sensor

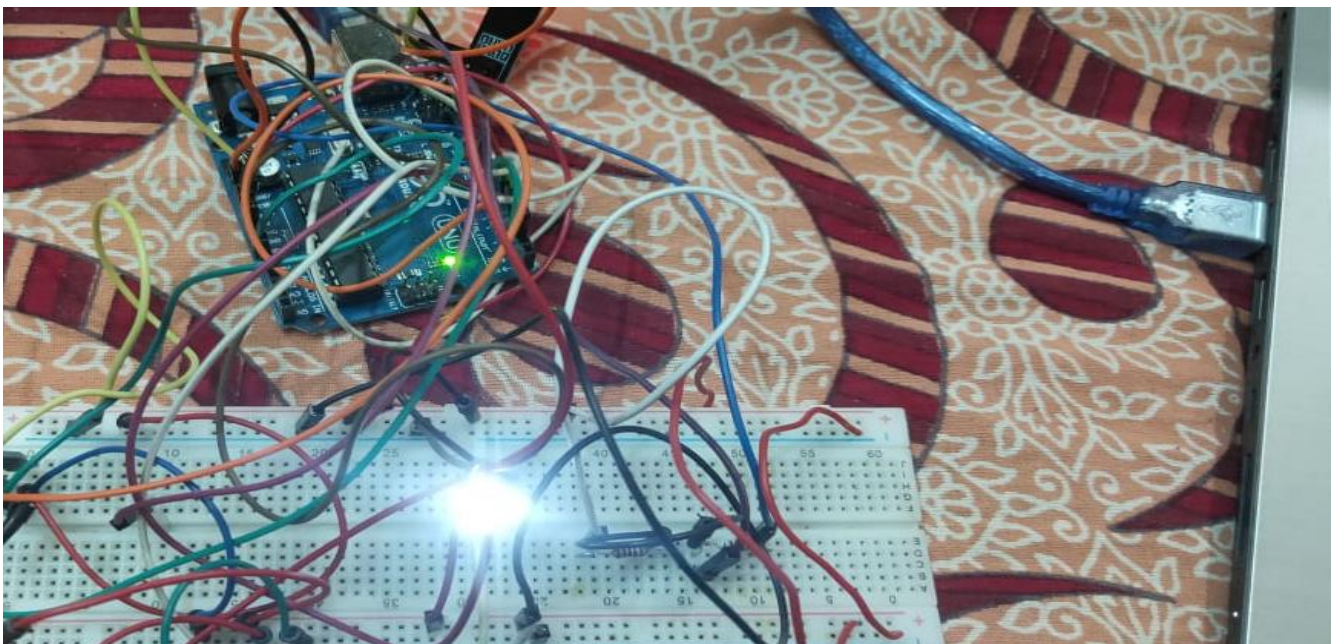


Aurdino

## IOT based Patient recognition system



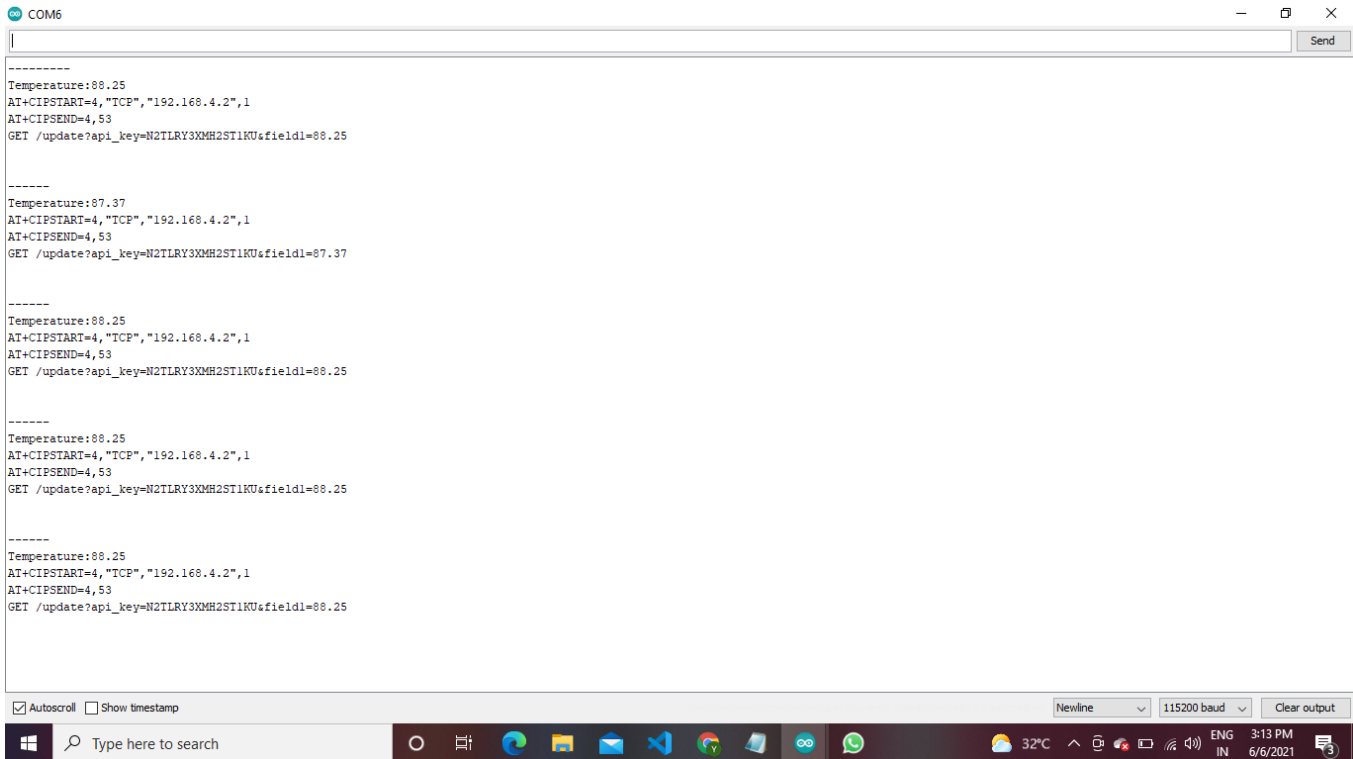
**Breadboard and led connection**



**Whole circuit connection**

# IOT based Patient recognition system

. Data being printed out on serial monitor and sent further to “ThingSpeak” server.



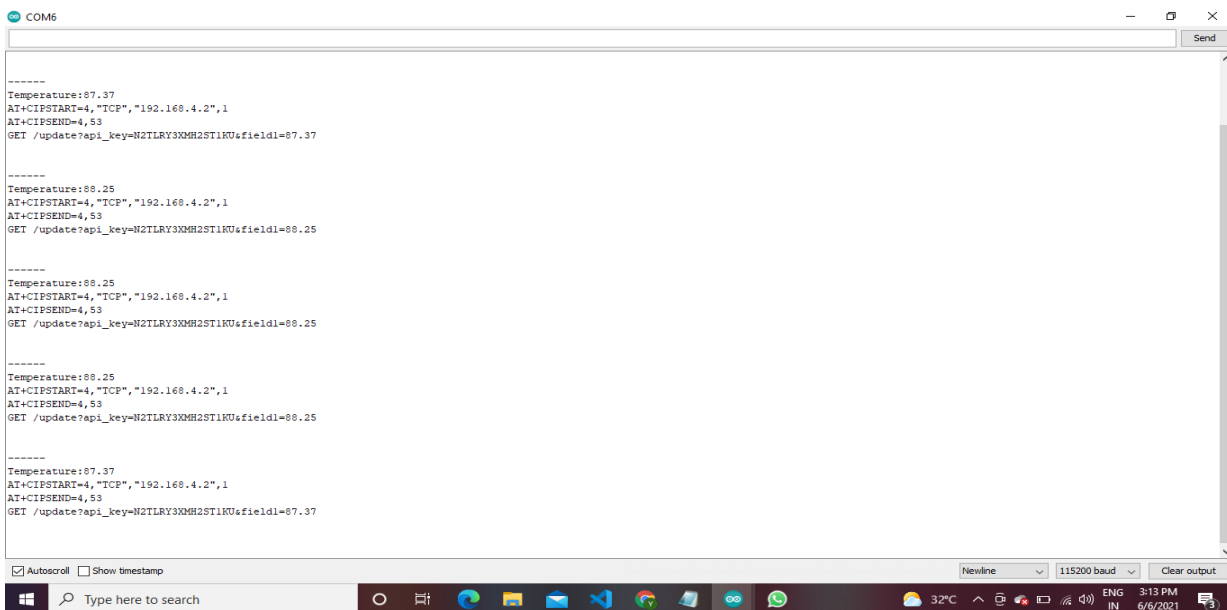
COM6

```
-----  
Temperature:88.25  
AT+CIPSTART=4,"TCP","192.168.4.2",1  
AT+CIPSEND=4,53  
GET /update?api_key=N2TLRY3XMH2ST1KU&field1=88.25  
  
-----  
Temperature:87.37  
AT+CIPSTART=4,"TCP","192.168.4.2",1  
AT+CIPSEND=4,53  
GET /update?api_key=N2TLRY3XMH2ST1KU&field1=87.37  
  
-----  
Temperature:88.25  
AT+CIPSTART=4,"TCP","192.168.4.2",1  
AT+CIPSEND=4,53  
GET /update?api_key=N2TLRY3XMH2ST1KU&field1=88.25  
  
-----  
Temperature:88.25  
AT+CIPSTART=4,"TCP","192.168.4.2",1  
AT+CIPSEND=4,53  
GET /update?api_key=N2TLRY3XMH2ST1KU&field1=88.25  
  
-----  
Temperature:88.25  
AT+CIPSTART=4,"TCP","192.168.4.2",1  
AT+CIPSEND=4,53  
GET /update?api_key=N2TLRY3XMH2ST1KU&field1=88.25
```

☒ Autoscroll ☐ Show timestamp

Newline 115200 baud Clear output

Windows taskbar: Type here to search, 32°C, 3:13 PM, 6/6/2021



COM6

```
-----  
Temperature:87.37  
AT+CIPSTART=4,"TCP","192.168.4.2",1  
AT+CIPSEND=4,53  
GET /update?api_key=N2TLRY3XMH2ST1KU&field1=87.37  
  
-----  
Temperature:88.25  
AT+CIPSTART=4,"TCP","192.168.4.2",1  
AT+CIPSEND=4,53  
GET /update?api_key=N2TLRY3XMH2ST1KU&field1=88.25  
  
-----  
Temperature:88.25  
AT+CIPSTART=4,"TCP","192.168.4.2",1  
AT+CIPSEND=4,53  
GET /update?api_key=N2TLRY3XMH2ST1KU&field1=88.25  
  
-----  
Temperature:88.25  
AT+CIPSTART=4,"TCP","192.168.4.2",1  
AT+CIPSEND=4,53  
GET /update?api_key=N2TLRY3XMH2ST1KU&field1=88.25  
  
-----  
Temperature:87.37  
AT+CIPSTART=4,"TCP","192.168.4.2",1  
AT+CIPSEND=4,53  
GET /update?api_key=N2TLRY3XMH2ST1KU&field1=87.37
```

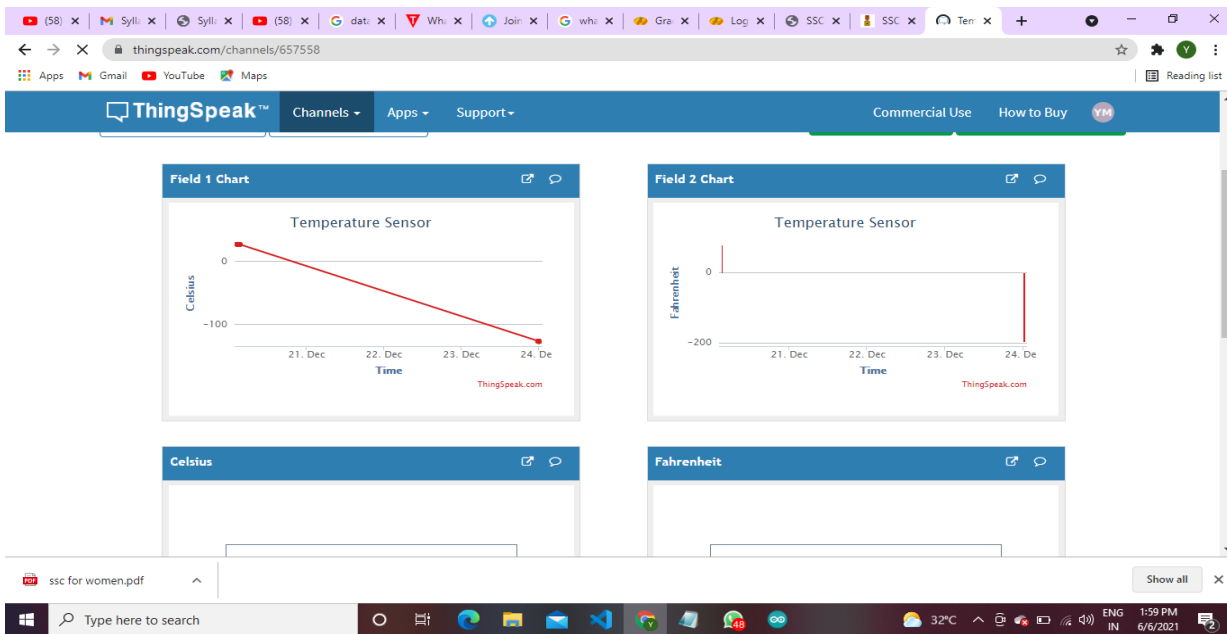
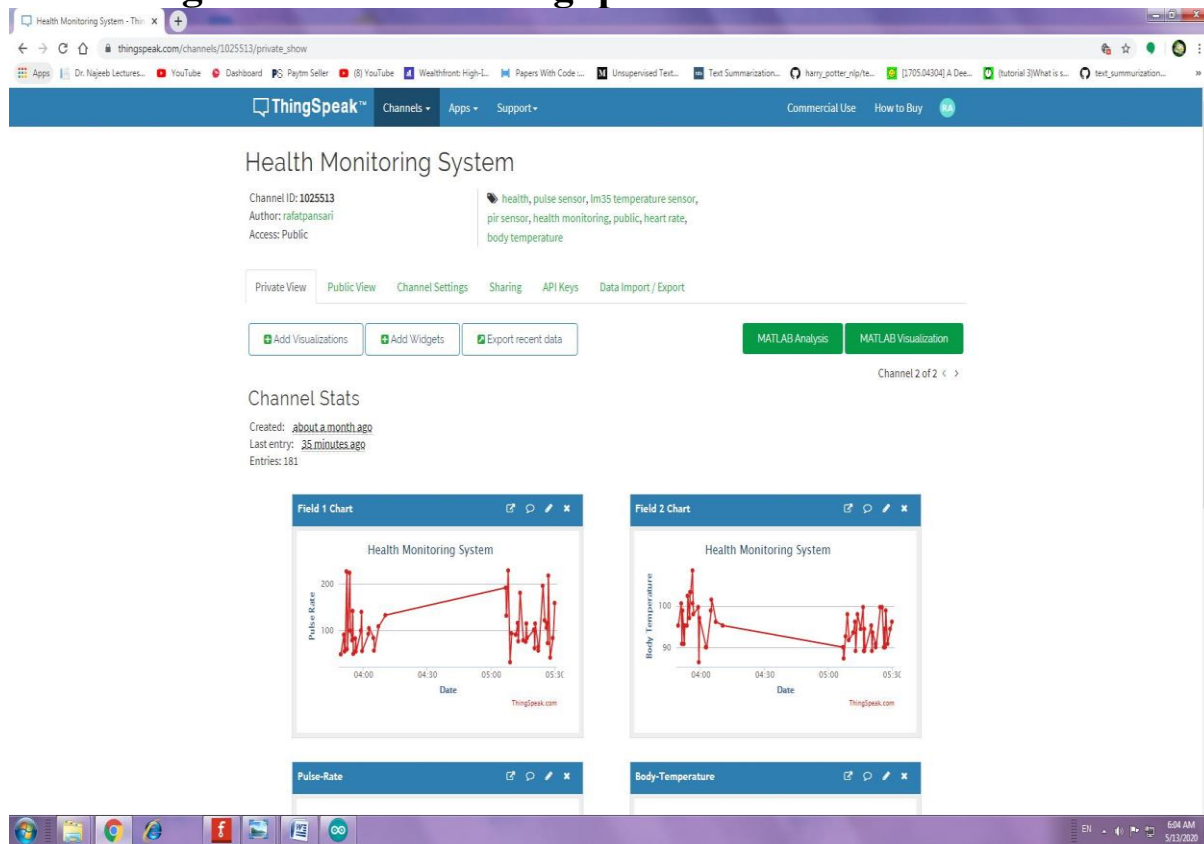
☒ Autoscroll ☐ Show timestamp

Newline 115200 baud Clear output

Windows taskbar: Type here to search, 32°C, 3:13 PM, 6/6/2021

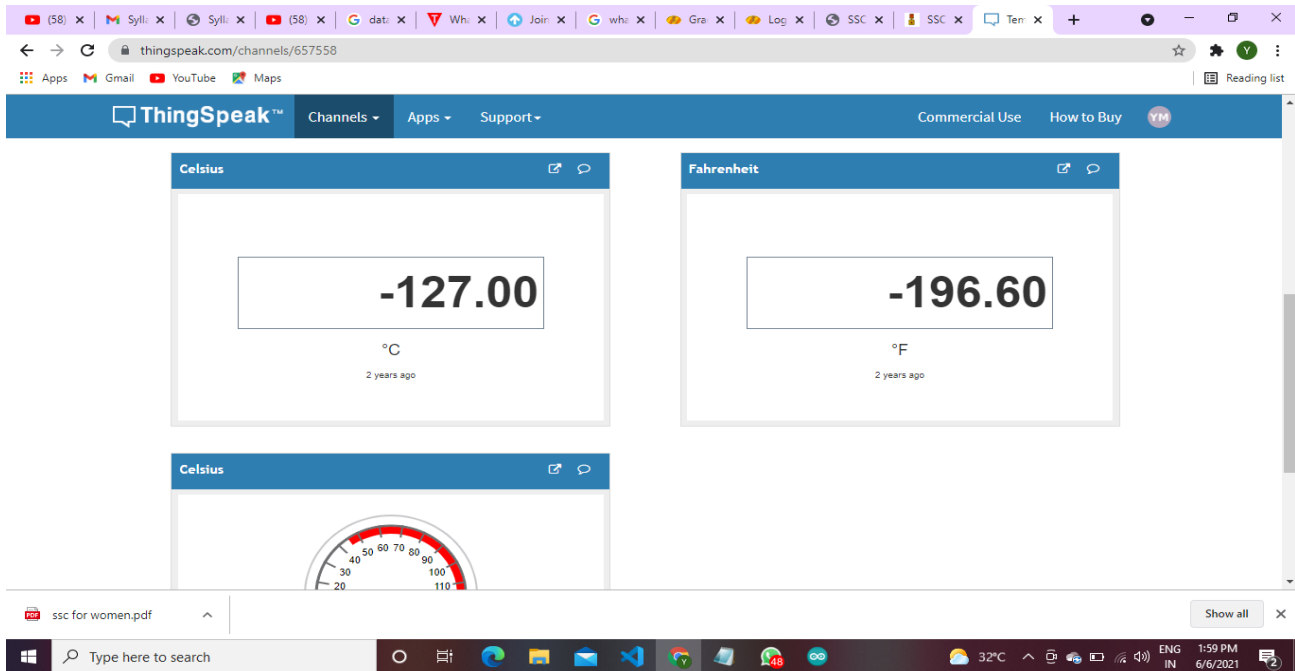
# IOT based Patient recognition system

## Data being recorded on “ThingSpeak” cloud.





# IOT based Patient recognition system



### Chapter 6 – Conclusion and future work

#### **6.1 Future scope of work:**

According to the availability of sensors or development in biomedical trend more parameter can be sensed and monitored which will drastically improve the efficiency of the wireless monitoring system in biomedical field. A graphical LCD can be used to display a graph of rate of change of health parameters over time. The whole health monitoring system which we have framed can be integrated into a small compact unit as small as a cell phone or a wrist watch. This will help the patients to easily carry this device with them wherever they go. In addition with medical application we can use our system in industrial and agricultural application by using sensors like humidity sensors, fertility check sensors, etc.

Conclusion:

In this proposed system a mobile physiological monitoring system is presented, which is able to continuously monitor the patients heart beat, blood pressure and other critical parameters in the hospital. We proposed a continuous monitoring and control mechanism to monitor the patient condition and store the patient data's in server using Wi-Fi Module based wireless communication, we also proposed remote health care data acquisition and smart storage system. The Future work of the project is very essential in order to make the design system more advanced. In the designed system the enhancement would be connecting more sensors to internet which measures various other health parameters and would be beneficial for patient monitoring i.e. connecting all the objects to internet for quick and easy access. Establishing a Wi-Fi mesh type network to increase in the communication range.

Reference

[1]. Real time wireless health monitoring application using mobile devices, International Journal of Computer Networks & Communications (IJCNC) Vol.7, No.3, May 2015, Amna Abdullah, Asma Ismael, Aisha Rashid, Ali Abou-ElNour, and Mohammed Tarique

[2]. Secured Smart Healthcare Monitoring System Based on Iot, International Journal on Recent and Innovation Trends in Computing and Communication Volume: 3 Issue: 7, Bhoomika.B.K, Dr. K N Muralidhara

[3]. Zigbee and GSM Based Patient Health Monitoring System, 2014 International Conference on Electronics and Communication System (ICECS-2014), Purnima, Puneet Singh

#### **6.2 LIMITATION:**

Privacy can be potentially undermined. As we've already mentioned, systems get hacked. Lots of attention will need to be focused on data security, which requires significant additional spendings.

Unauthorized access to centralization. There is a chance that dishonest interlopers may access centralized systems and realize some cruel intentions.

## **IOT based Patient recognition system**

Global healthcare regulations. International health administrations are already issuing guidelines that must be strictly followed by governmental medical establishments integrating the IoT in their workflow. These may restrict possible capacities to some extent.

### **6.3CONCLUSION**

In this paper, we have proposed and implemented a Smart Health Monitoring System. It is working successfully. Through this system, Doctors can screen the patient's condition without the need of being in patient's vicinity all the time. By using biomedical sensors, we have monitored patient's data viz. temperature and heart beat rate. This data is further uploaded on the ThingSpeak server. From the server, the data is converted into JSON and any abrupt changes occurring in patient's health status is notified to the concerned Doctor or Paramedical staff through an SMS which is automatically sent by the system. For future work, we can increase the functionality of system by adding more biomedical sensors and by making our system readings more accurate.

## **Chapter 7 – Bibliography and references**

### **7.1 References**

<https://www.javatpoint.com/arduino-coding-basics>

<https://youtu.be/IqxzXzJVjkg>

<https://how2electronics.com/iot-patient-health-monitoring-system-esp8266/>