IOT Based Smart Health Monitoring

Ruthvik Aravind
SENSE
Vellore Institute of Technology
Vellore, Tamil Nadu
ruthvik.aravind2020@vitstudent.ac.in

Roopreet Saluja
SENSE
Vellore Institute of Technology
Vellore, Tamil Nadu
roopreet.saluja2020@vitstudent.ac.in

Rohan Menon
SENSE
Vellore Institute of Technology
Vellore, Tamil Nadu
rohan.menon2020@vitstudent.ac.in

Ankit Parashar
SENSE
Vellore Institute of Technology
Vellore, Tamil Nadu
ankit.parashar2020@vitstudent.ac.in

Dr. Konguvel E
SENSE
Vellore Institute of Technology
Vellore, Tamil Nadu
konguvel.e@vit.ac.in

***Abstract*—We developed a health monitoring system based on the Internet of Things that monitors the patient's body temperature and heart rate and sends email or SMS notifications when those readings surpass threshold levels. In addition, we monitored the patient's drip levels and sent an automated notification if they became too high. The internet allows patients' health to be tracked from anywhere in the world, and measures of pulse rate and body temperature are collected using ThingSpeak and Google sheets. There is also a panic button, which the patient can press in an emergency to send an email or SMS to their loved ones. The primary contribution of this work is the transmission of an electrocardiogram (ECG signal) to a specific smart mobile phone for monitoring by a doctor. This aids in the diagnosis of heart disorders before the worst case scenario occurs. Eventually, the project's outcomes are displayed on both a smartphone and a personal computer (PC).**

Keywords—Health Monitoring, Thingspeak, temperature, driplevel, heartrate , NodeMCU, IOT

I. INTRODUCTION

Health is defined as a complete condition of physical, mental, and social well-being, not only the absence of illness. People's desire for a better life is fundamentally linked to their health. Sadly, the global health crisis has produced a quandary due to reasons such as insufficient health services, vast gaps between rural and urban areas, and physician and nurse unavailability during difficult times. The internet has grown enormously in recent years, and it has now been broadened to link goods throughout the internet. All gadgets are linked to one another using various smart technologies to form a global universal network known as the Internet of Things (IoT). Before, doctors found it difficult to manage patients in metropolitan areas. It was nearly impossible to obtain the necessary medical facilities, especially in cases of acute condition. It took a long time for individuals in the remote area to get to the hospital. Even after arriving at the hospital, there is a slim likelihood of receiving express care. Sometimes the Doctor had routinely observed the patient, and the number of doctors available to treat that specific patient at that time was relatively limited.

The Internet of Things refers to the interconnection of devices, apps, sensors, and network connectivity that improves the ability of various entities to collect and exchange data. The differentiating feature of the Internet of Things in the healthcare system is the continuous monitoring of a patient by examining numerous metrics and inferring a favourable outcome from the history of such continual monitoring. Several such devices equipped with medical sensors can now be found in ICUs. Despite 24 hours continuous monitoring, there may be times when the doctor is not alerted in time when there is an emergency. There may also be difficulties in exchanging data and information with professional doctors as well as concerned family members and relatives. The technology to improve these characteristics is currently available, but it is neither accessible or inexpensive to the majority of people in poor nations such as India. As a result, these solutions to these challenges may be as easy as an extension to current devices that lack these capabilities.

Continuous patient screening at medical facilities necessitates the purchase of aid. Their heartbeats must be constantly monitored. It will be home when they trade. As a result, there is a potential that the problem will reoccur. Details concerning the patient's health (such as high temperature, heart frequency, and position) will be measured and communicated via a web server on occasion. A proposed project that uses temperature and pulse rate sensors to detect temperature and pulse rate, which are critical patient parameters that must be maintained so that the doctor can monitor the patient's temperature and heartbeat and take appropriate action if there are any anomalies. The proposed method involves attaching a node to the patient's body that measures the patient's temperature and heartbeat and provides that data to the microcontroller.

The major goal of this system is to update the data online and provide a warning to doctors if there is any abnormality, as well as to anticipate if the patient has any ailment.

II. PROBLEM STATEMENT

In today's social insurance system, patients who remain at home during post-operative days are checked either by an overseer or a medical carer. This technology may not provide continuous monitoring because anything can change in a health parameter in a matter of seconds and if a guardian/attendant is not present, it causes more significant injury. Thus, with this innovation generated period where web administrations the world considers adding to another keen health awareness framework where time to time constant checking of the patient is completed.

III. LITERATURE SURVEY

1) Challenges and opportunities in IoT healthcare systems: A Systematic Review: Sureshkumar Selvaraj1 · Suresh Sundaravaradhan-This paper investigates the architecture utilised in IoT, particularly cloud integrated systems. Factors such as accuracy and power consumption are major concerns in the IoT, hence research projects aimed at increasing the performance of IoT-based healthcare systems are reviewed

2) Development of Smart Healthcare Monitoring System in IoT Environment Md. Milon Islam·Ashikur Rahaman1·Md. Rashedul Islam- This research presents a smart healthcare system in an IoT context that can monitor a patient's basic health indicators as well as the ambient condition in real-time. To collect data from the hospital environment, four sensors named heart beat sensor, body temperature sensor, room temperature sensor, CO sensor, and CO2 sensor are used in this system.

3) An edge AI-enabled IoT healthcare monitoring system for smart cities: Vipin Kumar Rathia, Nikhil Kumar Rajput, Shubham Mishra, Bhavya Ahuja Grover, Prayag Tiwari, Amit Kumar Jaiswal, M. Shamim Hossain- This article describes an AI-enabled IoT and edge computing-based healthcare system with low patient latency. The system collects health-related data, processes and analyses it at edge nodes, then stores and shares it permanently in edge data centres.

4)An IoT Framework for Healthcare Monitoring Systems: Dharmendra Singh Rajput Dept. of Software & System Engg. School of Information Technology & Engineering VIT University, Vellore (TN), India Rakesh Gour School of Information Technology & Engineering VIT University, Vellore (TN), India- The purpose of this article is to offer an architectural framework for documenting the entire monitoring life cycle and to identify critical service components. It serves as the cornerstone for providing robust, efficient, and secure health monitoring.

5) Secured Smart Healthcare Monitoring System Based on Iot Bhoomika. B.K MTech in VLSI and Embedded systems PES college of Engineering, Mandya Email Id: bkbhoomika@gmail.com Dr. K N Muralidhara Professor and Head of the Department Electronics and communication PES College of Engineering- The security issue is addressed by sending data over the password-protected Wi-Fi module ESP8266, which is encrypted using standard AES128 and can be accessed by users/doctors by logging into the html webpage. During an emergency, a GSM module connected to the controller sends an alert message to the doctor.

6) M. Hamim, S. Paul, S. I. Hoque, M. N. Rahman and I. Baqee, "IoT Based Remote Health Monitoring System for Patients and Elderly People," 2019 International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST), Dhaka, Bangladesh, 2019, pp. 533-538- This paper describes a ubiquitous and cost-effective system built with sensors and the Internet of Things, as well as a user-friendly app-based method for collecting real-time data. To select the best model, a survey of fever symptoms is done among persons over the age of 50, followed by machine learning analysis.

7) . Pathinarupothi, Rahul Krishnan, and Ekanath Rangan. “Effective prognosis using wireless multi-sensors for remote healthcare service.” eHealth 360°. Springer, Cham, 2017- The proposed answer in the study is to develop a method to connect via the internet and provide top-notch medical support to patients even in the most remote regions where facilities do not exist.

The main purpose of this system is to update the data online, warn doctors of any abnormalities, and evaluate whether the patient is sick.

8) Prajoona Valsalan, Tariq Ahmed Barham Baomar, Ali Hussain Omar Baabood,” IOT BASED HEALTH MONITORING SYSTEM” 2019 by Advance Scientific Research-This study presents a portable physiological testing framework, which may continuously monitor the patient's heart rate, the room's temperature, and other essential factors.

9) V. Tripathi and F. Shakeel, "Monitoring Health Care System Using Internet of Things - An Immaculate Pairing," 2017 International Conference on Next Generation Computing and Information Systems (ICNGCIS), Jammu, 2017, pp. 153-158-This technology collects precise physiological data using sensors, gateways, and the cloud to analyse and store the data, and then wirelessly transfers the processed data for further inspection and analysis. It replaces the routine of having a medical practitioner examine the patient's vital signs on a regular basis with a continuous automated flow of information.

10) Amandeep Kaur, Ashish Jasuja, “Health Monitoring Based on IoT using RASPBERRY PI” International Conference on Computing, Communication and Automation (ICCCA2017), ISBN:978-1-5090-6471-7/17/ ©2017 IEE-The Smart Health Monitoring Device is a Raspberry Pi 3b+-based system that collects medical data via sensors. With a Wi-Fi module, this data are delivered to the mail over the internet. The user must actively clothe himself or herself in order to access the system's electrically interfaced oxygen level, temperature, drip level, tilt sensor, body movement, and cardiac monitor modules.

IV. Methodology

First, we include all the libraries. We are using software serial to communicate with esp8266.

2. Make instance for timer, SoftwareSerial and pulse sensor to use in our code.

3. Set-up low-level interrupts for most accurate BPM match and enable DEBUG to show ongoing commands on serial monitor.

4. Set your WiFi name , password and IP of thingspeak.com

5. Declare String to update information on ThingSpeak channel. You will need API key for this, which can be found from your ThingSpeak channel-> API key . Copy Write API key and paste here.

6. In setup function, set the baud rate for serial communication between Arduino serial monitor and esp8266. Start the ESP communication by giving AT command to it and connect it by calling connectWiFi(); function. After that we will initialize Timers by calling t.every(time_interval, do_this); which will take the readings of the sensors and update on the channel after every time_interval you defined.

7. We have to make functions for connectWiFi(), panic_button(), update_info() and getReadings().

8. Make function for connectWiFi() which will return True or False depending upon Wi-Fi connected or not. AT+CWMODE=1 will make ESP8266 work in station mode. AT+CWJAP=\, command, used in this function, is to connect to your Access Point (your Wi-Fi router).

9. Make getReadings(); function to take pulse sensor and LM35 readings and convert them to string using dtostrf(); function.

10. Define char array for BPM and temp and convert float value of these sensors to String using dtostrf().

11.Make function for updating sensor information on the ThingSpeak channel.

12."AT+CIPSTART=\"TCP\",\"", AT Command will establish TCP command over port 80

13.Attach the readings with the GET URL using "&field1="; for pulse readings and "&field2="; for temperature readings. Send this information using "AT+CIPSEND=" command.

14.Similarly, make function for panic_button. When button goes to HIGH, esp8266 send the information to the server using AT+CIPSTART and AT+CIPSEND commands.

15.Attach this information to "&field3=".

16.In loop function, call panic_button() and timers using t.update() function

Flowchart:

A. Software

ThingSpeak is an excellent tool for IoT-based projects. We can monitor our data and control our system over the Internet by using the Channels and webpages supplied by ThingSpeak.
ThingSpeak "collects" data from sensors, "analyses and visualises" the data, and "acts" via initiating a reaction. ThingSpeak has already been used in weather station projects utilising Raspberry Pi and Arduino; check them out to learn more about ThingSpeak. This section explains simply how to use ThingSpeak for this IoT Patient Monitoring Project.
We will utilise ThingSpeak to monitor the patient's heart rate and temperature over the internet.
We will also use the IFTTT platform to connect ThingSpeak to email/message services, allowing us to send alert messages anytime the patient is in a critical situation.

- First of all, user needs to Create a Account on ThingSpeak.com, then Sign In and click on Get Started.
- Now go to the 'Channels' menu and click on New Channel option on the same page for further process.
- You will now see a form for creating the channel; fill in the Name and Description fields as desired. Then, in Field 1, Field 2, and Field 3 labels, type 'Pulse Rate,' 'Temperature,' and 'Panic,' and check the checkboxes for the Fields. Additionally, check the box next to the 'Make Public' option in the form, and then Save the Channel. Your new channel has now been established.
- Three charts are displayed below. Take note of the Write API key; we will utilise it in our code.
- Now, we'll utilise the server's ThingHTTP app to trigger the IFTTT applet for data input into Google Sheets and sending email/sms.
- ThingHTTP allows communication between devices, websites, and online services without requiring the protocol to be implemented at the device level.
- ThingHTTP allows you to describe actions that you want to be triggered by other ThingSpeak apps like React.
- To make New ThingHTTP, we will need URL for triggering which we will get from IFTTT.

B. SETTING UP IFTTT TO TRIGGER MAIL/SMS BASED ON THINGSPEAK VALUES:

- Step 1: Go to IFTTT and look for Webhooks, then click on it.
- Step 2: Choose Documentation.
- Step 3: In the event box, type "Patient Info" and copy the Link. This URL will be used in ThingHTTP.
- Step 4: On the My Applete menu, select Create Applete.
- Step 5: Click "+this," then search for and choose Webhooks. Choose "Receive a web request" as the trigger.

- Step 6: Enter the Event Name that you entered in the event box in the webhooks URL. Choose Create Trigger.
- Step 7: Click "+that," then search for and select Google Sheets.
- Choose Add row to spreadsheet.
- Step 8: Give your document any name you like. Date and time, event name, BPM number, and body temperature will all be written in the formatted row box.
- Step 9: Evaluate your applet and press the Finish button.

Similarly, when the Panic event occurs, we must create an applet that sends an email. Therefore, once again, click "+this" and select Webhooks, then put "Panic" in the event name field. Look for Gmail in "+that" and click on it. Now, select Send an email. Enter the email addresses for which you want to receive emails when the patient is in distress. Enter the email body content and click the Create Action button. Finish by reviewing it.

C. ThingHTTP for connecting ThingSpeak with IFTTT

Choose New ThingHTTP. Enter any name you like and paste the URL you copied from the webhooks manual. Fill in the remaining fields as indicated below. In Body, we must enter the data that we wish to send to the IFTTT applet. We are providing the patient's pulse and temperature. "value1": "%%channel channelID field fieldNumber%%","value2": "%%channel channelID field fieldNumber%%" After entering these details, click Save ThingHTTP. Similarly, we must create ThingHTTP for "Panic." Repeat the process. In the URL, replace Patient Info with Panic. The body stays empty, and the rest of the information is the same as in the previous ThingHTTP. Keep it. Now we must make React trigger the URL.

- Give your React a name. The condition type is Numeric, and the test frequency is Data Insertion. Choose the Condition for which you want the URL to be triggered. From the If Channel drop down option, choose your channel. Choose field 1 (Pulse rate) and set the condition to greater than any number. I used 60 words. Select ThingHTTP from the Action drop down option and click on it. Choose "Run action whenever condition is met" and then click Save React. Similarly, make the Panic Respond as illustrated. Choose "Run action whenever condition is met" and then click Save React. We are finished with web-based work. Today we'll look at Arduino code.

D. Hardware

Pulse Sensor is a well-designed Arduino plug-and-play heart-rate sensor. The sensor attaches to a fingertip or earlobe and connects directly to Arduino. It also comes with an open-source monitoring app that displays your pulse in real time. The LM35 is a linear analogue temperature sensor. Its production varies in direct proportion to the temperature (in degree Celsius). The temperature range of operation is -55°C to 150°C. The output voltage varies by 10mV for every oC change in temperature. It can be powered by a 5V or 3.3V supply and has a standby current of less than 60uA. The ESP8266 is commonly referred to as a WiFi module, but it is essentially a microcontroller. The ESP8266 is the name of a microcontroller produced by Espressif Systems, a Shanghai-based firm. Because this microcontroller can execute WIFI-related tasks, it is commonly utilised as a WIFI module.

- Signal pin of pulse sensor -> A0 of arduino
- Vcc pin of pulse sensor -> 5V of arduino
- GND pin of pulse sensor -> GND of arduino
- Vout of LM35 -> A1 of Arduino
- Tx of ESP8266 -> pin 10 of arduino
- Rx of ESP8266 -> pin 11 of arduino
- CH_PD and Vcc of ESP8266 -> 3.3 V of arduino
- GND of ESP8266 -> GND of arduino
- Push button -> digital pin 8 of arduino

Figure 1. ESP8266

V. RESULTS

After following the preceding steps, the following results were obtained on the application:

1) Reading of the drip level:

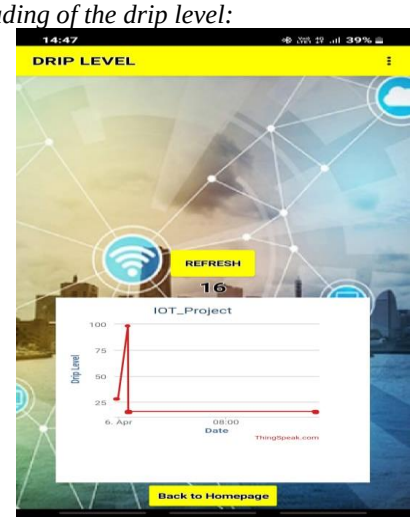

Figure 2: Drip Level

2) Pulse Reading:

Figure 3: Pulse Reading

3) Pulse Reading:

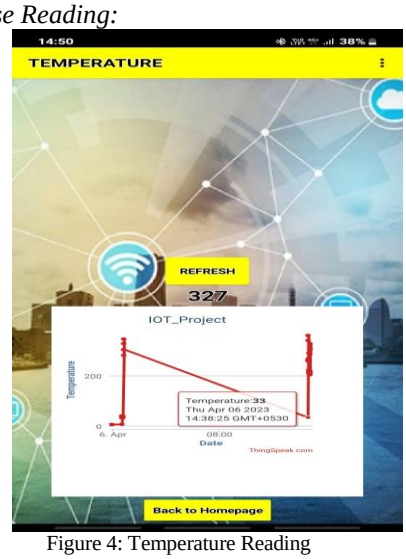

Figure 4: Temperature Reading

4) Integration:

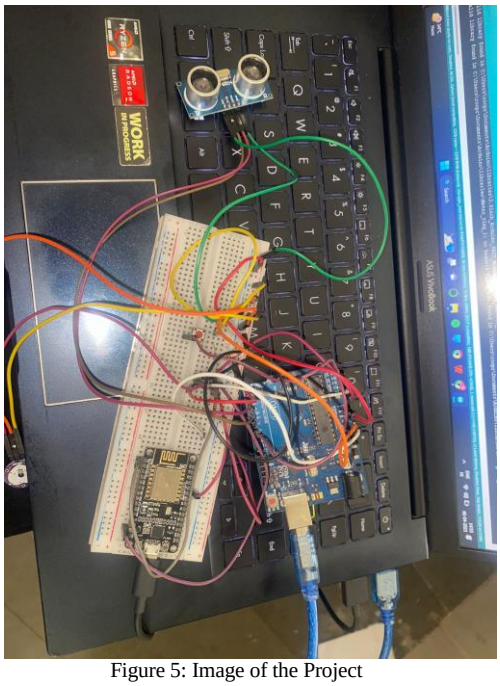

Figure 5: Image of the Project

VI. Conclusion and Future Work

In this study, we proposed and demonstrated a prototype for an automatic system that assures continuous monitoring of multiple health indices and prediction of any form of disease or disorder, sparing the patient the agony of repeated hospital visits. The proposed technology can be installed in hospitals, and a large amount of data can be collected and saved in an online database. Even the results can be made mobile-accessible via an application.

The system can be developed further by incorporating artificial intelligence system components to help doctors and patients. Data mining can be used to analyse the data, which consists of the medical history of many patients' parameters and related outcomes, in search of consistent patterns and systematic linkages in the disease. For example, if a patient's health metrics change in the same way as a previous patient in the database, the repercussions might be anticipated. If similar patterns are discovered again, doctors and medical researchers will have an easier time finding a solution to the problem.

VII. References

[1] S. Tyagi, A. Agarwal, and P. Maheshwari. A conceptual framework for iot-based healthcare system using cloud computing. In 2016 6th International Conference - Cloud System and Big Data Engineering (Confluence), pages 503–507, Jan 2016.

[2] N. V. Lopes, F. Pinto, P. Furtado, and J. Silva. Iot architecture proposal for disabled people. In 2014 IEEE 10th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob), pages 152–158, Oct 2014.

[3] T. S. Barger, D. E. Brown, and M. Alwan. Health status monitoring through analysis of behavioral patterns. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans, 5(1):22–27, Jan 2005. ISSN 1083-4427.

[4] Luca Catarinucci, Danilo de Donno, Luca Mainetti, Luca Palano, Luigi Patrono, Maria Laura Stefanizzi, and Luciano Tarricone “An IoT-Aware Architecture for Smart Healthcare Systems” IEEE INTERNET OF THINGS JOURNAL, VOL. 2, NO. 6, DECEMBER 2015.

[5] Mirjana Maksimović, Vladimir Vujović and Branko Perišić “ A Custom Internet of Things Healthcare System” 10th Iberian Conference on Information Systems and Technologies - CISTI' 2015, At Águeda, Portugal, Volume: 1.

[6] Alok Kulkarni, Sampda Sathe, “Healthcare applications Of Internet of the Things: A Review,” (IJCSIT) International Journal of Computer Science and Information Technologies, Vol.5, 2014, PP6229-6232.

[7] S. Raja Gopal, Patan.SA, "Design and Analysis of Heterogeneous Hybrid topology for VLAN configuration.", International journal of emerging trends in engineering research, vol:7, no:11, pn:487-491, November, 2019

[8] Ufoaroh S.U , Oranugo C.O, Uchechukwu M.E , HEARTBEAT MONITORING AND ALERT SYSTEM USING GSM TECHNOLOGY , International Journal of Engineering Research and General Science Volume 3, Issue 4, July-August, 2015 ISSN 2091-2730

[9] Aieshwarya. B. Chavan Patil "An IoT Based Health Care and Patient Monitoring System to Predict Medical Treatment using Data Mining Techniques: Survey", International Journal of Advanced Research in Computer and Communication Engineering(IJARCCE), ISO 3297:2007 Certified Vol. 6, Issue 3, March 2017

[10] BN. Karthik, L. Durga Parameswari, R. Harshini, A.Akshaya, "Survey on IOT & Arduino Based Patient Health Monitoring System", International Journal of Scientific Research in Computer Science, Engineering and Information Technology © 2018 IJSRcSEIT | Volume 3 | Issue 1 | ISSN : 2456-3307

[11] Zhibo Pang, Qiang Chen, Junzhe Tian, Lirong Zheng, and Elena Dubrova, “Ecosystem Analysis in the Design of Open Platform based In-Home Healthcare Terminals towards the Internet-of-Things” Corporate Research, ABB AB, Vasteras, Sweden lCT School, Royal Institute of Technology (KTH), Stockholm, Sweden.International Conference on Advanced Communication Technology, ICACT, ISSN 1738-9445, 529-534 p.

[12] Marco Bazzani, Davide Conzon, Andrea Scalera, Maurizio A. Spirito, and Claudia Irene Trainito, “Enabling the IoT paradigm in e-health solutions through the VIRTUS middleware” 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications.

Appendix [1]

```
#define USE_ARDUINO_INTERRUPTS true

#define DEBUG true

#define SSID "Rohan's Oneplus"     // "SSID-WiFiname"

#define PASS "RubiksCube" // "password"

#define IP "184.106.153.149"      // thingspeak.com ip


#include <SoftwareSerial.h>


#include "Timer.h"

#include <PulseSensorPlayground.h>                    // Includes the PulseSensorPlayground Library.

Timer t;

PulseSensorPlayground pulseSensor;


String msg = "GET /update?key=V0XJU37FWWRVMOFR";

SoftwareSerial esp8266(10,11);



//Variables
int trigPin = 5;    // TRIG pin

int echoPin = 4;    // ECHO pin

float duration_us, distance_cm;

const int PulseWire = A0;       // PulseSensor PURPLE WIRE connected to ANALOG PIN 0

const int LED13 = 13;          // The on-board Arduino LED, close to PIN 13.

int Threshold = 550;           //for heart rate sensor

float myTemp;

int myBPM;

String BPM;

String temp;

int error;

int panic;

int raw_myTemp;

float Voltage;

float tempC;

void setup()

{


  Serial.begin(9600);

  esp8266.begin(9600);

  pulseSensor.analogInput(PulseWire);

  pulseSensor.blinkOnPulse(LED13);        //auto-magically blink Arduino's LED with heartbeat.

  pulseSensor.setThreshold(Threshold);
  // configure the trigger pin to output mode
  pinMode(trigPin, OUTPUT);
  // configure the echo pin to input mode
  pinMode(echoPin, INPUT);


  // Double-check the "pulseSensor" object was created and "began" seeing a signal.

   if (pulseSensor.begin()) {

    Serial.println("We created a pulseSensor Object !");  //This prints one time at Arduino power-up,  or on Arduino reset.

  }

  Serial.println("AT");

  esp8266.println("AT");


  delay(3000);


  if(esp8266.find("OK"))

  {

    connectWiFi();
    Serial.println("Connected");

  }

  t.every(10000, getReadings);

   t.every(10000, updateInfo);

}


void loop()

{
  // generate 10-microsecond pulse to TRIG pin
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  // measure duration of pulse from ECHO pin
  duration_us = pulseIn(echoPin, HIGH);

  // calculate the distance
  distance_cm = 0.017 * duration_us;

  // print the value to Serial Monitor
  Serial.print("distance: ");
  Serial.print(distance_cm);
  Serial.println(" cm");

  delay(1000);


  panic_button();

start: //label

    error=0;

   t.update();

    //Resend if transmission is not completed

    if (error==1)
```

```
   {
     goto start; //go to label "start"
   }
 delay(4000);
}

void updateInfo()
{
 String cmd = "AT+CIPSTART=\"TCP\",\"";
 cmd += IP;
 cmd += "\",80";
 Serial.println(cmd);
 esp8266.println(cmd);
 delay(2000);
 if(esp8266.find("Error"))
 {
  return;
 }
 cmd = msg ;
 cmd += "&field1=";    //field 1 for BPM
 cmd += BPM;
 cmd += "&field2=";  //field 2 for temperature
 cmd += temp;
 cmd += "\r\n";
 Serial.print("AT+CIPSEND=");
 esp8266.print("AT+CIPSEND=");
 Serial.println(cmd.length());
 esp8266.println(cmd.length());
 if(esp8266.find(">"))
 {
  Serial.print(cmd);
  esp8266.print(cmd);
 }
 else
 {
  Serial.println("AT+CIPCLOSE");
  esp8266.println("AT+CIPCLOSE");
  //Resend...
  error=1;
 }
}

boolean connectWiFi()
{
 Serial.println("AT+CWMODE=1");
 esp8266.println("AT+CWMODE=1");
 delay(2000);
 String cmd="AT+CWJAP=\"";
 cmd+=SSID;
 cmd+="\",\"";
 cmd+=PASS;
 cmd+="\"";
 Serial.println(cmd);
 esp8266.println(cmd);
 delay(5000);
 if(esp8266.find("OK"))
 {
  return true;
 }
 else
 {
  return false;
 }
}

void getReadings(){
 raw_myTemp = analogRead(A1);
 Voltage = (raw_myTemp / 1023.0) * 5000; // 5000 to get millivots.
 tempC = Voltage * 0.1;
 myTemp = (tempC * 1.8) + 32; // conver to F
 Serial.println(myTemp);
 int myBPM = pulseSensor.getBeatsPerMinute();  // Calls function on our pulseSensor object that returns BPM as an "int".
                                               // "myBPM" hold this BPM value now.
if (pulseSensor.sawStartOfBeat()) {            // Constantly test to see if "a beat happened".
Serial.println(myBPM);                        // Print the value inside of myBPM.
}

 delay(20);
```

```
  char buffer1[10];
   char buffer2[10];
  BPM = dtostrf(myBPM, 4, 1, buffer1);
  temp = dtostrf(myTemp, 4, 1, buffer2);
 }

void panic_button(){
 panic = digitalRead(8);
  if(panic == HIGH){
  Serial.println(panic);
    String cmd = "AT+CIPSTART=\"TCP\",\"";
 cmd += IP;
 cmd += "\",80";
 Serial.println(cmd);
 esp8266.println(cmd);
 delay(2000);
 if(esp8266.find("Error"))
 {
  return;
 }
 cmd = msg ;
 cmd += "&field3=";
 cmd += panic;
 cmd += "\r\n";
 Serial.print("AT+CIPSEND=");
 esp8266.print("AT+CIPSEND=");
 Serial.println(cmd.length());
 esp8266.println(cmd.length());
 if(esp8266.find(">"))
 {
  Serial.print(cmd);
  esp8266.print(cmd);
 }
 else
 {
  Serial.println("AT+CIPCLOSE");
  esp8266.println("AT+CIPCLOSE");
  //Resend...
  error=1;
  }
}
}
```