ISSN 2320 - 2602
Volume 7 No.11, November 2018
International Journal of Advances in Computer Science and Technology
Available Online at http://www.warse.org/IJACST/static/pdf/file/ijacst017112018.pdf
https://doi.org/10.30534/ijacst/2018/017112018

Temperature and Heart Attack Detection using IOT(Arduino and ThingSpeak)

Prof(Dr).Jayant Shekhar[1], Mr.Desalegn Abebaw[2], Dr. Mesfin Abebe Haile[3]
Md.Ahmed Mehamed[4], Mr.Yohannis Kifle[5]
[1] Professor,Adama Science and Technology University,Adama
[2]CSE-Programe Chair,Adama Science and Technology University,Adama,
[3],CSE-PG-Coordinator,Adama Science and Technology University,Adama
[4]ECE-Lecturer,Adama Science and Technology University,Adama,
[5]CSE-Lecturer,Adama Science and Technology University,Adama,
[1] jayantshekhar@hotmail.com
[2]ledessu@gmail.com
[3] mesfinabha@gmail.com
[4]ahme.oumer@gmail.com
[5] joey.kifle@gmail.com

Abstract: Medical Electronics is also going to advance with the application of Internet of Things. Internet of Things is the fastest growing technology. IoT is about to find application everywhere and in everything. In this paper we proposed, a simple patient health monitoring device as an IoT application. This IoT device could read pulse rate and measure surrounding temperature. It continuously monitors the pulse rate and surrounding temperature and updates them to an IoT platform. The IoT platform used in this paper is ThingSpeak.

It provides full control and monitoring of the pulse rate and surrounding temperature and updates them to an IoT platform.

Keywords: *Arduino ,ESP8266, Pulse Rate, Temperature, Healthcare and Internet of Things.*

1.INTRODUCTION

Many types of Pulse Rate and Temperature monitors exist, including both wrist monitors and the more common arm monitors. The arm band monitors are unusual because typically are larger, require a separate module attached to the arm band. Arm band monitors are less convenient than the wrist monitors. The wrist monitors are self- contained, they have no separate module or any other attachments. Wireless Pulse Rate and Temperature Monitors take readings continuously to collect many readings from patients to see how their Pulse Rate and Temperature varies throughout the day. It's well known that people's Pulse Rate and Temperature varies a lot during the day but it's debatable what to do about it. If someone's Pulse Rate and Temperature is normal when taken at the home in the morning but then reads high when taken later during the same day in a doctor's office. What can be the important point in this case? A Wireless Pulse Rate and Temperature Monitor provides many readings throughout the day and would give a better and a true picture of patients' Pulse Rate and Temperature. Such a device could also be a research tool to help better understand the effects of Pulse Rate and Temperature on the body. This paper discusses the aspects of collecting of the Pulse Rate and Temperature datas and then gathered data can be wirelessly sent using any of the connection options like Bluetooth, 802.15.4, ZigBee, Wi-Fi, 3G and GPRS depending on the application. In another way data can be sent to the Cloud in order to perform permanent storage or monitoring in real time by sending the data directly to a laptop or Smartphone. iPhone and Android applications can be used in order to easily see the patient's information. This system is expected to monitor patient's Pulse Rate and Temperature for doctor to monitor the patient's condition sitting in his own office without being physically present near to the patient's bed. More convenient and accurate results will be analyzed by the doctor for diagnosing a medical problem of the patient which resulted as abnormal Pulse Rate and Temperature readings.

2.WHY IOT

We are living in Internet age where every physical object may be connected to each other for sharing information purpose. Due to enhanced wireless technologies like 6LoWPAN, Wi-Fi, Bluetooth & ZigBee, many things or objects around us have the ability to exchange information automatically. This network of things or objects that are connected to each other via Internet, local area network or Wireless Sensor Networks is called Internet of Things(IoT). IoT is made of two words one is Internet and second is The Things. Internet is network of networks that are connected world widely via some standard protocols.

Things refers to any Physical Object that may be involved in connectivity. IoT uses many technologies like Radio Frequency Identification(RFID) tag, Sensors, Actuators and Smart phone and cloud computing support etc. By using IoT, we can connect anything, can access any service and useful information of any object from anywhere and anytime [1]. In the concept of IoT every object is connected with each other through a unique identifier so that it can transfer data over the network without a human to the human interaction [1, 2]. IoT has referred as a network of everyday objects having ubiquitous computing. The ubiquity of the objects has increased by integrating every object with embedded system for interaction [14]. It connects human and devices through a highly distributed network. IoT is basically the world wide interconnection of devices. The aim of IoT is to connect every person and every object through the internet. In IoT ,every object is assigned a unique identifier, so that every object is accessible through the internet [15][16].

The IoT device proposed in this paper is built on Arduino UNO. The Arduino is one of the earliest and most popular prototyping boards. The Arduino is interfaced with ESP8266 Wi-Fi modem to connect with an internet router and access the cloud server. The Arduino is interfaced with LM-35 temperature sensor to sense the surrounding temperature and a pulse sensor to read pulse rate. The measured pulse rate and temperature are displayed on a character LCD interfaced to the Arduino and are passed to the cloud platform by transmitting data to a Wi-Fi access point. With this simple yet effective device, health status of a critically ill patient can be constantly monitored. It can be used to keep track of health of aged people who frequently have heart or blood pressure issues.

The health related data i.e. pulse rate and temperature are periodically updated and logged to the ThingSpeak platform. That data can be further utilized to keep medical history of the patient. The Freeboard.io is used as Dashboard to graphically represent the recorded data.

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. The Sketch is written, compiled and loaded using the Arduino IDE. The IoT platform used is ThingSpeak and the Freeboard.io is used to built the IoT Dashboard.

3.RELATED WORK

Recently, several IoT systems have been developed for IoT healthcare and assisted living applications. A multiple communication standard compatible IoT system for medical devices was designed by Wang etal. in[3].Xu et al.proposed a resource-based data accessing method (UDA-IoT) that is suitable for healthcare information-intensive applications [4]. Kolici et al. proposed and implemented a medical support system considering Peer-to-Peer (P2P) and IoT technologies. They used a smart box to control the situation of patients. Moreover, they performed several experiments to evaluate the implemented system for few different scenarios [5]. Sandholm et al. proposed an on-demand Web Real-Time Communication (WebRTC) and IoT device tunneling service for hospitals. The proposed system relies on intercepting key parts of the WebRTC Javascript Session Establishment Protocol (JSEP) and using local network gateways that can multiplex traffic from multiple concurrent streams efficiently without leaking any WebRTC traffic across the firewall except through a trusted port [6]. As per Linklabs By using RFID tag health care providers can track real time location, assigned physician and progress of treatment etc. Medical equipment and devices like defibrillators, ECG machines, spirometry and nebulizers etc can be tagged with sensors and tracked easily with IoT [7]. Krishnan et al. presented a real-time Internet application with distributed flow environment for medical IoT. If the patient is out of range for the Wi-Fi, or the server is unavailable, the patent's data will be stored locally and sent to the server when the patient arrives back in range of connectivity [8]. Azariadi et al. proposed an algorithm for electrocardiogram (ECG) signal analysis and arrhythmia detection on IoT-based embedded wearable medical platform, assuitable for 24-h continuous monitoring. A Galileo board is used to implement the design [9]. Mohan presented a cyber security framework for IoT Personal Medical Devices (PMDs) that enable enhanced mobility for the patient. In the meantime, it is facilitating better monitoring of the patient's condition while moving. He presents the security threats and limitations of PMD IoT that makes addressing these threats challenging. He also presents some initial solution approaches in order to address these security threats [10]. Yeh et al. presented a cloud-based fine-grained health information access control framework for lightweight IoT devices with dynamic auditing and attribute revocation. They handled the potential security challenges and the cloud reciprocity issues. The results show that the proposed scheme is promising for cloud-based Personal Health Information (PHI) platform [11]. Porambage et al. proposed a secure lightweight authentication and key establishment protocol for end-to-end communication for constrained devices in IoT-enabled ambient assisted living systems. They used proxy-based approach to assign the heavily computational operations to more powerful devices in the neighborhood of the used medical sensors. The results are promising for the real world applications [12]. Yelamarthi and Laubhan [13] have designed and implemented a portable electronic travel aid for the blind. Utilizing ultrasonic range finders mounted on the

belt, the assistive device was able find obstacles in front of the user, and provide respective navigation directions through a Bluetooth headphone. However, this device is limited in the distance and localization of obstacles with high accuracy. Addressing this limitation, in this paper, we present a cloud-based IoT system that is capable of monitoring pulse rate and temperature remotely.

4.IOT BASED HEALTHCARE SYSTEM

For overcoming the problems of health monitoring, we proposed the health monitoring equipment using an IoT device. It is built on Arduino UNO. The Arduino UNO is one of the most popular prototyping board that is commonly used even in the IoT projects. The pulse reader, LM-35 temperature sensor, character LCD and ESP8266 Wi-Fi modem are interfaced to the Arduino to make this medical IoT device.

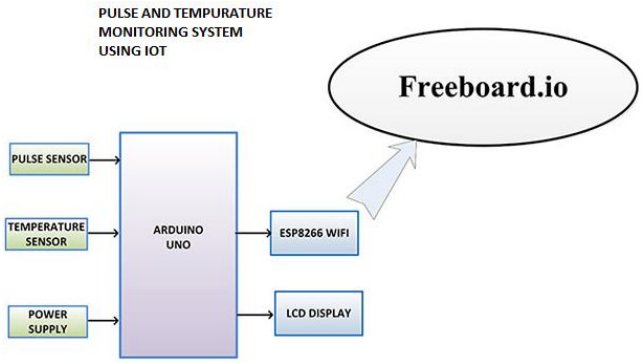

Figure 1: System overview

This IoT device could read pulse rate and measure surrounding temperature. It continuously monitors the pulse rate and surrounding temperature and updates them to an IoT platform. The IoT platform used in this paper is ThingSpeak..

5.ARCHITECTURE OF THE SYSTEM

The Arduino based IoT device has the following circuit connections –
Arduino UNO - The Arduino UNO is ATmega328 based microcontroller board. It is one of the most popular prototyping boards. The board comes with built-in arduino boot loader. It has 14 GPIO pins, 6 PWM pins, 6 Analog inputs and on board UART, SPI and TWI interfaces, an on-board resonator, a reset button, and holes for mounting pin headers. While programming the board, it can be connected to the PC using USB port and the board can runs on USB power. The Arduino UNO has 32 Kb Flash memory, 1 Kb EEPROM and 2 Kb SRAM. The board can be connected to different Arduino Shields for connectivity with Ethernet, Bluetooth, Wi-Fi, Zigbee or Cellular network and it can be connected to most of the IoT platforms. The ATmega328 controller has the following pin configuration –

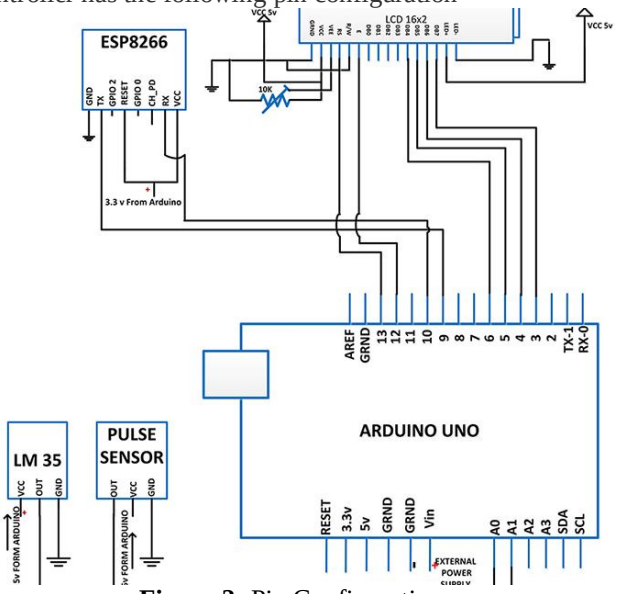

Figure 2: Pin Configuration

In this proposed device, the two Analog Input pins of the board are used to interface the Pulse sensor and LM-35 temperature sensor, two GPIO are used to interface ESP8266 module where pins are configured UART transmitter and receiver pins using software serial and 6 GPIO pins are used to interface the 16X2 character LCD.

LM-35 Temperature Sensor –

LM-35 is a precision IC temperature sensor with its output proportional to the temperature (in oF). The sensor circuitry is sealed and therefore it is not subjected to oxidation and other processes. With LM-35, temperature can be measured more accurately than with a Thermistor. It also possess low self heating and does not cause more than 0.1 oC temperature rise in still air. The LM-35 has three pins - VCC (Pin 1), Out (Pin 2) and Ground (Pin 3). The VCC and Ground pin are connected to VCC and ground respectively. The LM-35 can be supplied a voltage between 4V and 20V, so a 5V supply is used on it same which is powering the Arduino board. The out pin of the LM-35 is connected to A1 pin of the Arduino since the output from the LM-35 is analog in nature. The value read is converted to Fahrenheit using the standard formulas.

Pulse Sensor –

The Pulse Sensor Amped is a plug-and-play heart-rate sensor for microcontrollers like PIC, AVR, Arduino etc. It can be used to easily incorporate live heart-rate data into a

project. It essentially combines a simple optical heart rate sensor with amplification and noise cancellation circuitry making it fast and easy to get reliable pulse readings. It simply needs to be clipped to earlobe or finger tip and plug into 3.3 V or 5 V supply from Arduino or battery. The pulse sensor module has three terminals - VCC, Ground and Out. The output pin of the pulse sensor module is connected to analog pin A0 of the Arduino. The VCC is connected to 5V DC output of Arduino and Ground is connected to the common ground.

ESP8266 Wi-Fi Modem –

The ESP8266 Wi-Fi Module is used to connect the Arduino board with a Wi-Fi router, so that it can access the cloud. It is a self contained SOC with integrated TCP/IP protocol stack that can access to a Wi-Fi network. The ESP8266 is capable of either hosting an application or off loading all Wi-Fi networking functions from another application processor. Each ESP8266 module comes pre-programmed with an AT command set firmware. The module comes available in two models - ESP-01 and ESP-12. ESP-12 has 16 pins available for interfacing while ESP-01 has only 8 pins available for use. The ESP-12 has the following pin configuration –

Pin Number	Pin Name	Pin Function
1	RESET	Active Low External Reset Signal
2	ADC(TOUT)	ADC Pin Analog Input
3	CH_PD	Active High Chip Enable
4	GPIO16	General purpose IO
5	GPIO14	General purpose IO
6	GPIO12	General purpose IO
7	GPIO13	General purpose IO
8	VCC	Power Supply
9	Ground	Ground
10	GPIO15	General purpose IO, should be connected to ground for booting from internal flash
11	GPIO1	General purpose IO, Serial Tx1
12	GPIO0	General purpose IO, Launch Serial Programming Mode if Low while Reset or Power ON
13	GPIO4	General purpose IO
14	GPIO5	General purpose IO
15	GPIO3	General purpose IO, Serial Rx
16	GPIO1	General purpose IO, Serial Tx

Figure 3: ESP-12 pin configuration

In this paper we are using the ESP-01 model. The ESP-01 model has the following pin configuration –

Pin Number	Pin Name	Pin Function
1	Ground	Ground
2	GPIO1	General purpose IO, Serial Tx1
3	GPIO2	General purpose IO
4	CH_PD	Active High Chip Enable
5	GPIO0	General purpose IO, Launch Serial Programming Mode if Low while Reset or Power ON
6	RESET	Active Low External Reset Signal
7	GPIO3	General purpose IO, Serial Rx
8	VCC	Power Supply

Figure 4: ESP-01 pin configuration

The RESET and VCC pins of the module are connected to the 3.3 V DC from Arduino while Ground pin is connected to the common ground. The Tx and Rx pins of the module are connected to the 9 and 10 pins of the Arduino UNO.

16X2 LCD - A character LCD is used to display the pulse rate and surrounding temperature. The 16X2 LCD display is connected to the Arduino board by connecting its data pins to pins 3 to 6 of the Arduino board. The RS and E pins of the LCD are connected to pins 13 and 12 of the Arduino board respectively. The RW pin of the LCD is grounded. The VCC pin of the LCD module is connected to 5V DC from the Arduino. For adjusting brightness of the LCD module, a variable resistor is connected at VEE pin and the other two terminals of the variable resistor are connected between VCC and ground.

LCD	Arduino UNO
RS	13
RW	GRND
E	12
D7, D6, D5, D4	3, 4, 5, 6 respectively

Figure 5: LCD pin configuration

The standard open-source library for interfacing LCD with Arduino board is used in the project. The library works as expected and needs no changes or modifications.

Power Supply - All the components in the circuit require 5V DC. The circuit is initially powered by a 12V battery. The power from the battery is regulated to 5V DC using 7805 voltage regulator IC. The pin 1 of the voltage regulator IC is connected to the anode of the battery and pin 2 of it is connected to the ground. The voltage output is drawn from pin 3 of the IC. An LED along with a 10K Ω pull-up resistor is also connected between common ground and output pin to get a visual hint of supply continuity. The character LCD, pulse sensor and LM-35 temperature sensor are provided 5V DC from the 5V DC power output of the Arduino while the ESP module is provided 3.3 V DC from 3.3 V DC power output of the Arduino.

6.EXPERIMENTATION AND RESULTS

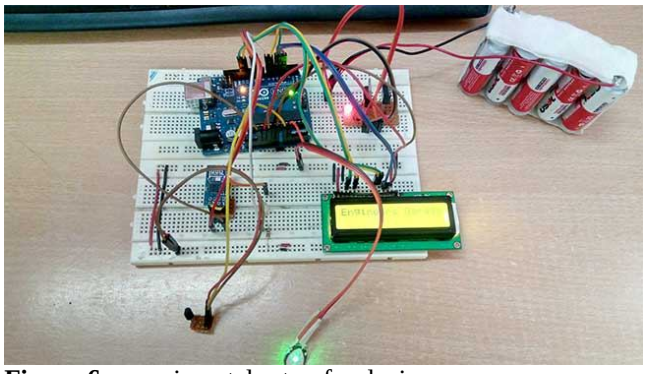

Figure 6: experimental setup for device

As shown in figure 6, This is a prototype model for IoT based pulse rate monitor. It can be designed as a wearable watch or ear plug. In a wearable design, the character LCD could be removed and the entire circuit can be shifted to small controller board or SOC.

When the circuit is powered by the battery, the Arduino starts reading the pulse rate from the pulse sensor and the ambient temperature from the LM-35 temperature sensor. The pulse sensor has an infrared LED and a photo transistor which help detect pulse at the tip of the finger or earlobe. Whenever it detects pulse, its IR LED flashes. The flash of the IR LED is detected by the phototransistor and its resistance changes when the pulse is changed. The heartbeat of a normal adult ranges from 60 to 100 per minute. For detecting beats per minute (BPM), first an interrupt is set which triggers in every 2 Milliseconds. So, the sampling rate by the Arduino to detect pulse is 500 Hz. This sampling rate is sufficient to detect any pulse rate.

So, at every 2 Milliseconds, the Arduino reads analog voltage output from the pulse sensor. The analog output from the pulse sensor is converted to a digital value using in-built ADC channel. The Arduino has 10-bit long ADC channel, so the digitized value can range from 0 to 1024. The middle value for this range is 512. Initially, the first beat is set to true and the second beat is counted when the condition that analog output from the pulse sensor is greater than the middle point i.e. 512 is satisfied. Then, onwards, every next beat is counted when the analog output from the pulse sensor is greater than the middle point i.e. 512 and 3/5 of the time between the beats recorded in previous cycle has passed. Every time, the beat is detected, a variable representing BPM is updated. This value in this variable is pushed to an array in every minute and is used to represent the actual Beats Per Minute or Heart Rate. The Arduino code also uses a function to provide an LED fading effect on every beat.

The pulse sensor can also detect body temperature. The LM-35 is used to detect the surrounding temperature here. The operating temperature range of LM-35 is from -55 °C to 150 °C. The output voltage varies by 10 mV in response to every oC rise/fall in ambient temperature, i.e., its scale factor is 0.01 V/ oC. The LM-35 IC does not require any external calibration or trimming to provide typical accuracies of ±0.25 °C at room temperature and ±0.75 °C over temperature range from −55 °C to 150 °C. Under normal conditions, the temperature measured by the sensor won't exceed or recede the operational range of the sensor. Typically in the temperature range from −55 °C to 150 °C, the voltage output of the sensor increases by 10 mV per degree Celsius. The voltage output of the sensor is given by the following formulae -

$$V_{out} = 10\ mV/°C * T$$

where,
V_{out} = Voltage output of the sensor

T = Temperature in degree Celsius

So, T (in °C) = Vout/10 mV

T (in °C) = Vout(in V)*100

If VCC is assumed to be 5 V, the analog reading is related to the sensed voltage over 10-bit range by the following formulae -

Vout = (5/1024)*Analog-Reading

So, the temperature in degree Celsius can be given by the following formulae -

T (in °C) = Vout(in V)*100

T (in °C) = (5/1024)*Analog-Reading *100

So, the temperature can be measured directly by sensing the analog voltage output from the sensor. The analogRead() function is used to read analog voltage at the controller pin.

The Arduino collects data from both the sensors and convert the values to the strings. The heartbeat is graphically represented on the character LCD along with the measured pulse rate and time between pulses as text. The temperature is also displayed on the LCD module.

The ESP8266 Wi-Fi module connected to the Arduino uploads the same data to ThingSpeak Server as it finds the Wi-Fi Access Point. For displaying and monitoring data uploaded to the ThingSpeak server, either a digital dashboard or data broker is needed. In this project, a digital dashboard called Freeboard.io is used to monitor the sensor data visually online. The Freeboard.io use JASON file to visualize ThingSpeak data. It offers three elements to build a dashboard -

1) Data Sources - The data sources get the data from external sources. These external sources can be data broker services, JavaScript applications or JSON files receiving content from an HTTP server. In this project, the data source is a JSON file that receives data from the ThingSpeak server.

2) Widgets - The Widgets help to display data in textual or graphical form. There are many widgets available in Freeboard.io like text, graph, gauge etc.

3) Panes - These are used to organize widgets.
Freeboard.io requires sign up and after sign widgets can be created.

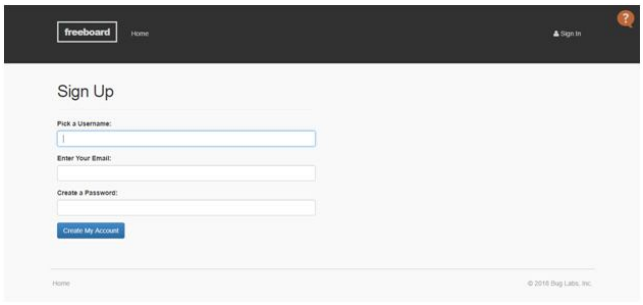

Figure 7: Freeboard.io Sign Up

There are two gauge type widgets created to monitor temperature and heartbeat.

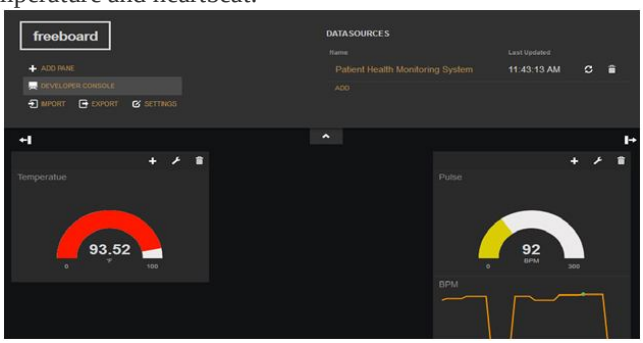

Figure 8: Freeboard.io Widgets

The dashboard on Freeboard.io can be created as follow -

1. Go to freeboard.io website and sign up with a new account.

2. Enter the name and click on the create button, after entering into the new window, click on the create data source and select type as JASON.

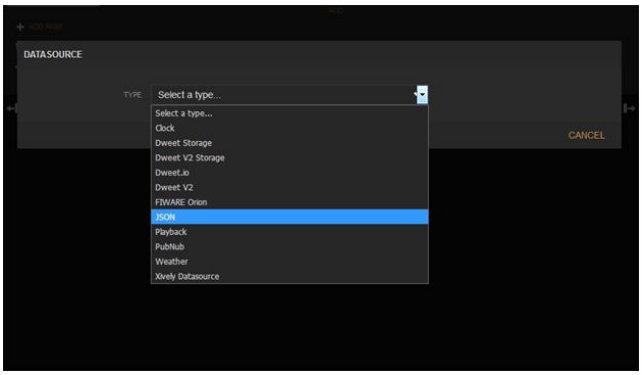

Figure 9: *Freeboard.io Data Source Selection*

3. After that fill the fields as shown in below image. In the URL tab shown in the image, change the number 392797 with the respective channel id.

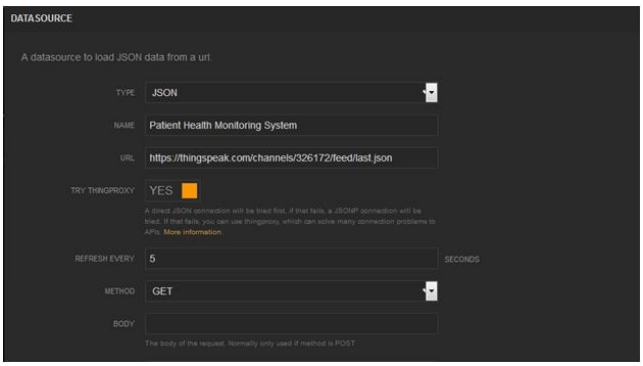

Figure 10: ***Setting Data Source on Freeboard.io to ThingSpeak Platform***

4. After creating the data source, click on add pane and select as Gauge.

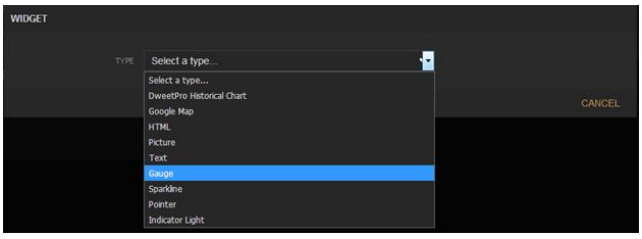

Figure 11: *Freeboard.io Widget Creation*

5. At the pane, after selecting the Gauge select the following as shown below and create the widgets.

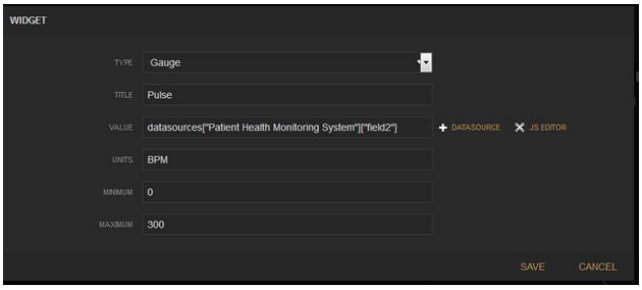

Figure 12: ***Freeboard.io Gauge Widget Creation***

This way the sensor data can be uploaded on ThingSpeak server and viewed online at Freeboard.io dashboard. The dashboard can be accessed from any device like Smart Phone, Laptop or PC having an Internet connectivity.

The Arduino board needs to connect with a Wi-Fi access point in order to connect with the internet. The name and password of the Wi-Fi access point are hard-coded in the Arduino sketch. The initialization of the Wi-Fi connection is done within the setup() function of the Arduino Sketch which runs once the board is powered on.

The setup of Wi-Fi connection is run by passing AT commands to the ESP8266 Wi-Fi modem. The modem is connected to GPIO pins of the Arduino which are configured as UART transmitter and receiver pins by using software serial library. The Wi-Fi is initialized by passing the following AT commands to ESP module -

AT : This command is passed to check if modem is functioning properly.

AT+GMR: This command is passed to print the firmware version.

AT+CWMODE=3 : This command is passed to set the Wi-Fi mode to both AP as well as Station mode.

AT+RST: This command is passed to reset the modem.

After reset, the modem check IP addresses of the available access points. The ESP modem can connect with the access point whose SSID and Password are hard-coded in the Arduino Sketch. The following AT command is passed to connect with the Access Point -
AT+CWJAP
Once the modem is connected to an access point, it obtains IP address by executing the following command -

AT+CIFSR: This command is used to obtain IP address of ESP module as an client.

The IP address is stored in a string and acknowledged to the Arduino board. Once the sensor data is collected, the following AT commands are passed to the ESP module for sending it to the cloud-

AT+CIPSTART=4,"TCP","184.106.153.149",80: This command is passed to start a TCP connection with the given IP address at the specified port (80).

AT+CIPSEND=4, String (getStr.length()): This command is passed to send data at the previously mentioned IP address with the number of transmit connections set to 4 and length of data (which can be maximum 2048 bytes) specified.

So, the Arduino connects with the server via ESP modem and the sent data can be graphically observed on Freeboard.io.

7.VALIDITY OF PROPOSED SOLUTION

Our proposed device is constructed using Arduino Uno board, ESP8266 and few sensors connected as shown in the figure-6. Once the circuit is connected in the given fashion and power is ON the sensor turns on and starts collecting data. Once the data is collected from individual sensors the data will be displayed on the LCD as shown in figure13.

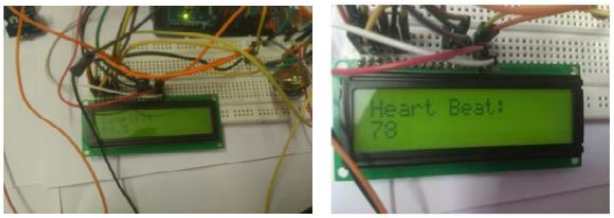

Figure 13 : Values of Temperature / Heartbeat Sensor

This health related data i.e. pulse rate and temperature are periodically updated and logged to the ThingSpeak platform through ESP8266. That data can be further utilized to keep medical history of the patient. Providing accurate, up to date and complete information about patient at the point of care. Enabling quick access to patient record for more coordinated efficient care.

8. CONCLUSION

In this proposed paper, we have used two diverse sensors to observe the patient's health. According to the patients requirement doctor will set the periodicity of the check up in the machine. According to that the patient can go through the check-up and the result of his health condition will be sent to the doctor immediately through ThingSpeak.

Advantages

Providing accurate, up to date and complete information about patient at the point of care. Enabling quick access to patient record for more coordinated efficient care. Securely sharing electronic information with patience and other

clinicians. Helping providers more effectively to diagnose patients, reduce medical errors and provide safer care. Improving patient and provider interaction and communication, as well as health care convenience. Enabling safer, more reliable prescribing. Helping promote legible, complete documentation and accurate, streamlined coding and billing. Enhancing privacy and security of the patient data.

Applications

1. Significantly advances the individualization and thereby the patients acceptance of electronic healthcare services for treatment and prevention.
2. Increases the ability to exploit very large knowledge spaces for individuals and professionals. In the eHealth domain.
3. Develop an adaptive, sustainable platform for electronic healthcare services increasing the computer-based diagnosis standard for medical decision support services.

REFERENCES

[1] Morais, Raul, A. Valente, and C. Serôdio. "A wireless sensor network for smart irrigation and environmental monitoring: A position article." In 5th European federation for information technology in agriculture, food and environement and 3rd world congress on computers in agriculture and natural resources (EFITA/WCCA), pp.45-850. 2005.

[2] Agrawal, Sarita, and Manik Lal Das. "Internet of Things—A paradigm shift of future Internet applications." In Engineering (NUiCONE), 2011 Nirma University International Conference on, pp.1-7. IEEE, 2011.

https://doi.org/10.1109/NUiConE.2011.6153246

[3] Wang, X., Wang, J.T., Zhang, X., Song, J.: A multiple communication standards compatible IoT system for medical usage. In: IEEE Faible Tension Faible Consommation (FTFC), Paris, pp. 1–4 (2013)

https://doi.org/10.1109/FTFC.2013.6577775

[4] Xu, B., Xu, L.D., Cai, H., Xie, C., Hu, J., Bu, F.: Ubiquitous data accessing method in IoT-based information system for emergency medical services. IEEE Trans. Ind. Inf. 10(2), 1578–1586 (2014)

https://doi.org/10.1109/TII.2014.2306382

[5] Kolici, V., Spaho, E., Matsuo, K., Caballe, S., Barolli, L., Xhafa, F.: Implementation of a medical support system considering P2P and IoT technologies. In: Eighth International Conference on Complex, Intelligent and Software Intensive Systems, Birmingham, pp. 101– 106 (2014)

[6] Sandholm, T., Magnusson, B., Johnsson, B.A.: An on-demand WebRTC and IoT device tunneling service for hospitals. In: International Conference on Future Internet of Things and Cloud, Barcelona, pp. 53–60 (2014)

https://doi.org/10.1109/FiCloud.2014.19

[7] Linklabs" IoT In Health Care: What You Should Know", online [Available]: https://www.link-labs.com/blog/IoTin-healthcare.

[8] Krishnan, B., Sai, S.S., Mohanthy, S.B.: Real time internet application with distributed flow environment for medical IoT. In: International Conference on Green Computing and Internet of Things, Noida, pp. 832–837 (2015)

[9] Azariadi, D., Tsoutsouras, V., Xydis, S., Soudris, D.: ECG signal analysis and arrhythmia detection on IoT wearable medical devices. In: 5th International Conference on Modern Circuits and Systems Technologies, Thessaloniki, pp. 1–4 (2016)

https://doi.org/10.1109/MOCAST.2016.7495143

[10] Mohan, A.: Cyber security for personal medical devices Internet of Things. In: IEEE International Conference on Distributed Computing in Sensor Systems, Marina Del Rey, CA, pp. 372–374 (2014)

https://doi.org/10.1109/DCOSS.2014.49

[11] Yeh, L.Y., Chiang, P.Y., Tsai, Y.L., Huang, J.L.: Cloud-based fine-grained health information access control framework for lightweight IoT devices with dynamic auditing and attribute revocation. IEEE Trans. Cloud Comput. PP(99), 1–13 (2015)

[12] Porambage, P., Braeken, A., Gurtov, A., Ylianttila, M., Spinsante, S.: Secure end-to-end communication for constrained devices in IoT-enabled ambient assisted living systems. In: IEEE 2nd World Forum on Internet of Things, Milan, pp. 711–714 (2015)

[13] Yelamarthi, K., Laubhan, K.: Space perception and navigation assistance for the visually impaired using depth sensor and haptic feedback. Int. J. Eng. Res. Innov. 7(1), 56–62 (2015)

[14] Coetzee, Louis, and Johan Eksteen. "The Internet of Things-promise for the future? An introduction." In IST-Africa Conference Proceedings, 2011, pp. 1-9. IEEE, 2011.

[15] Ashton, Kevin. "That 'internet of things' thing." RFiD Journal, Vol. 22, no. 7 ,pp.97-114,2009.

[16] Chase, Jim. "The evolution of the internet of things." Texas Instruments ,2013.