(IJRST) 2018, Vol. No. 8, Issue No. III, Jul-Sep **e-ISSN: 2249-0604, p-ISSN: 2454-180X**

HEALTHCARE SECTOR – LEVERAGING THE CAPABILITIES OF INTERNET OF THINGS (IOT) TO DEVELOP AN EFFICACIOUS HEALTH MONITORING AND DISEASE CONTROLLED SYSTEM

VIKAS RANA
Bahadurgarh, Haryana

ABSTRACT

*The introduction part illustrates the **Internet Of Things(IOT)** and its implementation in healthcare system. The Health Monitoring is very important criteria for healthcare in the modern era and should be integrated with IoT to improve the quality of service in health care sector. In later part, how it is implemented in the health care system has been studied.*

1. INTRODUCTION

The term “Internet of Things” was first coined in the starting of this century when work was done on **MIT Auto-ID Center** [1], to make a smart identification technology, which will help to reduce the error rate subsequently increasing efficiency and automation. Since then, the concept of IOT has evolved rapidly in various ways and now with its help huge number of small networks will remain connected to each other and can directly send data to the main network without any human interaction.

Quality of service in healthcare has always been under constant criticism in the modern era, as it is a very touchy subject. Health monitoring specially for elderly people is a topic of concern, as most people in the modern times are job holders and have so hectic life. It is difficult to manage a constant watch on the elderly person in a house and keeping a nurse or housekeeper is also a very costly issue nowadays. In this situation, remote health monitoring based on IOT can provide solution to this problem.

IOT has potential to gather and analyze information remotely with no human interaction. So, this emphasizes on it's potential to observe & stop any future hazard with exactitude and potential to aware the regarding authority just like a friend or the medico if there's any alarming state of affairs. The two fundamental reasons, that makes IOT as a vital need is that it needs no human interaction and secondly due to automation the method have less likelihood of getting errors i.e. having a lot of economical system indicating a much better quality in commission.

The current study, describe how to collect, monitor and analyze data using Custom Server and will provide insight on forecasting of the results. Additionally, it also discusses about construction of preventive reaction panel for coping with any alarming situation looming in future.

2. LITERATURE REVIEW

This chapter discusses about the concept of the project we are trying to implement, what kind of work has already been done and how this work is different and more improved then those work.

2.1 Concept

Monitoring the health of elderly people is a basic model for monitoring using different sensors. The reason for choosing elderly people for monitoring is that they are usually more vulnerable to sickness and other aging factors. So, usually it become difficult for working people to monitor the senior members of the family for whole time. Even if it is possible to take care of the elderly during their stay at home, however, it is difficult to observe their activities and condition during working hours. Thus, it was eminent to come with a solution that is to make a health monitoring system which can observe the daily basic activities of elderly people. The system will collect data of daily activities through sensors which will be placed according to the needs of the system. The retrieved data will then be compared with the provided threshold values, already provided to the system. If everything remains normal, then further analysis will not be done. But if any anomalies or abnormalities are recorded then the data are further analyzed using the appropriate predictive algorithm. The seriousness of the situation is then predicted by the system and if it is really that serious then a message will be send to the concerning relatives about the recent condition of the patient and about the prediction of the system.

The basic things that is tried to be monitored in this research are pulse rate, temperature as well as amount sleep of the patient. The sensors used in the research will work in following steps:

1. The sensors will send the data collected from the host in a regular basis after definite span of time.

2. The data thus collected will undergo a comparison with the threshold value given to the system.

3. If the data set concurs with the threshold value then the situation will be considered to be normal thus the system will not take any further action.

4. The data set thus retrieved if contains any abnormities, will then go for further analysis to predict how serious the condition of the person is.

2.2 Related Works:

An extensive research on the related system shows a very few related works where their own preliminary framework and prototype of the system were build. Some of the work like the research conducted on the **Ambient Assisted Living (AAL)** [2] actually did more of a literature survey of the state of its present condition of the monitoring system via IoT. They also tried to identify and highlight the critical issues and the quality of service as well as the user driven experiences in their work. Some, other worked on showing or highlighting the importance of IoT in the health sector and some proposals for the health monitoring architectures.

Some related findings used specific models for the health monitoring aspect. Like the abstraction of **Model Driven Tree Reference Model (MDTRM)** [3], where they explained the necessity of this model in the health field as well as identifying the complexities of the models. They also benched marked the models which came really handy for the initial phase of this research. Some other related model is **General Domain Model Architecture (GDMA)** [4], the health monitoring and sensing with cloud processing was also a helpful source behind the research, as it was useful for generating ideas to get raw data's from wearable devices which are compatible and capable of measuring many physical value which we can use in obtaining meaningful results.

Masimo Radical-7 [5], a health monitor for clinical environment helps to collect data andwirelessly transmits it for ongoing display. This provides high resolution display of information with higher graphical capabilities. It also has a touch based user interface, but as it is not cost effective and can't send an alarm message to notify any emergencies. **Free Scale Home Health Hub reference platform** [6] stores patient data to cloud *via* various sensors, where the people related to the patient can have an access. This platform too can't notify for any alarming situation to the people engaged with the patient. Some surveys of ours also lead us to projects which even discussed to monitor the health whole area through wireless network sensors [5][6]. They also tried to share their ideas by giving a model of their framework like cloud based processing [7] and big data [8].

Marcelloni and Vecchio (2008) have introduced a simple compression algorithm, particularly suited to the reduced storage and computational resources of a WSN node. They have evaluated the algorithm by compressing temperature and relative humidity data collected by a real WSN. They have obtained compression ratios of 66.99% and 67.33% for temperature and relative humidity datasets, respectively. Then, they have compared the algorithm with S-LZW, a lossless compression algorithm used in WSNs. They have shown that our algorithm can achieve higher compression ratios than S-LZW, despite a lower memory occupation and a less computational effort. Finally, we have also shown that our algorithm out performs gzip and bzip.

Zaleski and John (2008) describes a distributed patient monitoring system for visually monitoring patients and patient parameters using portable processing devices in different

remote locations includes a monitoring processor. The monitoring processor is responsive to user initiated commands from multiple different portable processing devices in different remote locations and includes an input processor and a data processor. The input processor acquires vital sign parameters and associated video data representative of multiple sequences of video images of corresponding multiple different patients. The data processor processes the vital sign parameters and associated video data to provide processed first video data representing an image sequence including a composite image including a first area showing live video of a selected first patient and a second area presenting vital sign parameters of the selected first patient.

3. METHODOLOGY AND IMPLEMENTATION

This chapter discusses the hardware required for the project implementation, along with the circuit connections and pseudo codes for the implementation.

3.1 Hardware

To conduct this, sensors are needed to monitor the health condition of the elderly people. In order to do so, two different criteria's i.e. body temperature & pulse rate are chosen to monitor the health of elderly people. So, for those two criteria's following sensors were chosen.

Temperature Sensor

The model **DS18B20 [9]** was chosen for collecting the temperature data from the host**.** The DS18B20 digital thermometer provides 9-bit to 12-bit Celsius temperature measurements and has an alarm function with nonvolatile user-programmable upper and lower trigger points. The DS18B20 communicates over a 1-Wire bus that by definition requires only one data line for communication with a central microprocessor. In addition, the DS18B20 can derive power directly from the data line eliminating the need for an external power supply. This sensor is very easy to work with and provides with accurate data of temperature. The sensor provides data in Celsius temperature scale andlater converted it to Fahrenheit scale for convenience. Whenever the host touches the sensor, the sensor saturates itself with the body temperature of the host and shows that temperature.

Fig 3.1: Temperature Sensor DS18B20

Pulse Sensor

The model **SEN-11574 [10]** was selected for collecting the pulse rate data from the host. The Pulse Sensor is a plug-and-play heart-rate sensor for Arduino. It can be used by anyone who wants to easily incorporate live heart-rate data. Essence it is an integrated optical amplifying circuit and noise eliminating circuit sensor. It is very easy to use by clipping the Pulse Sensor to the hosts earlobe or fingertip and plug it into the Arduino. This sensor can give data's like Pulse Rate, Cardio Graph and Inter Beat Interval. However, forpresent work only Pulse Rate per minute was used. The data from the sensor can be retrieved from the host via the host's fingertip or the lobe of the ear.

Arduino Uno

An **Arduino Uno** [11] was used for the purpose of this project and all sensors were connectedwith the Arduino. Arduino Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins of which 6 can be used as PWM outputs, 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. The sensors are powered from the Arduino Uno so is the Wi-Fi module. The Wi-Fi module is getting the data's from the sensors through this Arduino Uno.

Fig 3.3: Arduino Uno

Wi-Fi Module

For the purpose of this project, ESP-8266 [12] Wi-Fi module was chosen. The ESP8266 Wi-Fi Module is a self-contained SOC with integrated TCP/IP protocol stack that can give any microcontroller i.e. Arduino access to Wi-Fi network. The ESP8266 is capable of either hosting an application or offloading all Wi-Fi networking functions from another application processor. Each ESP8266 module is pre-programmed with an AT command set firmware, meaning, it can be simply connected with the Arduino device and get about as much Wi-Fi-

ability as a Wi-Fi Shield offers. The ESP8266 module is an extremely cost effective board. The module helps to transmit the sensor data to Custom Server cloud storage. It remains connected with the Arduino and connects with the server via internet and sends the data to that server.

Fig 3.4: ESP-8266 (Wi-Fi Module)

3.2 Circuit Connection

3.2.1 Wi-Fi Module

The ESP-8266 has eight ports, one of them is the power port which is used to supply power to the module and this was connected with the Arduino's 3.3v. ESP-8266 is very sensitive to high power so it was not connected to the 5v power besides it works better at 3.3v. Another port of the ESP-8266 is the ground port which is connected with the Arduino's ground. ESP-8266 sends and receives data with 2 different ports these ports are called TX and RX ports. ESP-8266's TX port was connected with Arduino's digital port number 2 and ESP-8266's Rx port was connected with Arduino's digital port 3. There is a reset port on the ESP-8266 which was not used as we did not need it, there is a port called ch-pd on ESP-8266 which is connected to the Arduino's 3.3v port. There are 2 GPIO ports on the ESP-8266, the GPIO1 is connected with the Arduino's 3.3v and the other port, GPIO0 is left unused.

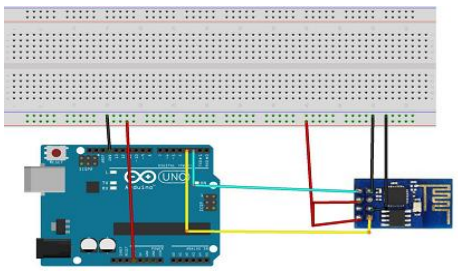


Fig 3.5: Circuit of ESP8266

3.2.2 Temperature Sensor

The DS18B20 has 3 pins to operate. One pin, VDD is for power and another pin is for ground. The DS18B20 can operate with power from 3.0v to 5.0v. The VDD port is connected with the Arduino's 3.3v through a 4.7ohm register. The ground port is connected with Arduino ground. The other port of the DS18B20 is the data port. This port is used to send temperature data to the Arduino, this port can also take power from the Arduino if it needs any extra. The data port of the DS18B20 is connected to the Arduino digital pin 5 through the same 4.7 ohm register. So,

basically one end of the 4.7ohm register is connected with the VDD port and the other end of the register is connected with the data port. This register is put with both these ports so that overpower cannot harm the DS18B20 as the data port also may take power from the Arduino.

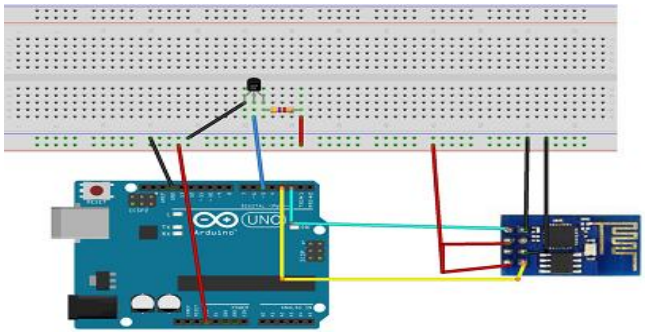


Fig 3.6: Circuit diagram of the DS18B20, temperature sensor.

3.2.3 Pulse Sensor

The pulse sensor has three pins; one of them is the VCC pin to power the pulse sensor as usual. This pin is connected with Arduino's 5v power supply. There is also a ground pin in the pulse sensor which is connected with the Arduino's ground port. The other pin of the pulse sensor is the data pin. This pin sends analog data to the Arduino so we had to attach this pin with the Arduino's analog pin 0. This way the Arduino will be converting the analog data to digital and we can avoid using an extra analog to digital converter.

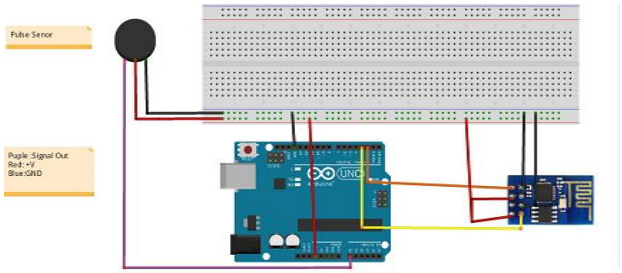


Fig 3.7: Circuit diagram of Pulse Sensor

3.3 Data Collection

After connecting all the hardware and the Arduino, data was collected from the sensors through the Arduino.

The temperature sensor is a small semi-circle hardware, to measure temperature the host just has to put their finger on the temperature sensor and the sensor will gradually adapt with the hosts temperature and take that data in the Arduino. The data that this sensor collects is in Celsius format so we have to convert it to Fahrenheit format.

The pulse sensor is a small round, disc shaped hardware, it has a green light in middle of the sensor, when the sensor receives power from the Arduino this green light will be lit. To measure the pulse of the host, they simply have to put their finger on the green light and the sensor will automatically send the pulse per minute data to the Arduino.

3.4 Data Storage

The data that were collected from the sensors through the Arduino should be stored somewhere for analysis. For that purpose Custom Server, a website that allows data storage data and analysisis used in this thesis. The ESP-8266 is used for connecting the Arduino with the internet and stores that necessary data's to Custom Server. The ESP-8266 will receive data's from the Arduino and with the help of a Wi-Fi connection, the ESP-8266 will send the acquired data's to Custom Server.
For different data's different types of channels are created in Custom Server. The temperature of the host can be continuously uploaded to Custom Server with the ESP-8266. From the temperature sensor, the temperature of the host will be saved at one channel and also the highest temperature measured will be stored too. This way we can alert the responsible person in terms of constant high temperature of the host. Same goes for pulse sensor data's. All of these are being stored in the IIS Server to analyze.

3.5 Custom Server

Custom Server is an open source IOT application and API. It helps to store and retrieve data from things using the HTTP protocol all over the internet or from the local area network (LAN). ioBridge launched Custom Server in 2010 to support IoT based application. It allows its users to use .NET which they integrated to analyze and visualize the uploaded data without purchasing the license. Custom Server was written in IIS, has a Cross-Platform operating system and stores its data in cloud.
A user can have a free signin to Custom Server and after signing, can open up unlimited channel under that account. Each channel has different fields and in those fields data's are displayed in graphical forms. User can analyze the data according to the need using the .NET. This manipulation of data is one of the most fascinating features of Custom Server.

3.6 Steps of Implementation

For completing this project and to monitor the health criteria's of elderly people fully automatically, following steps were used. There is a systematic way of doing all this things to operate the sensors, the Wi-Fi module and to manage the data's in the IIS Server. The sequential steps of this thesis is stated below,

- Individually collect data from the sensors through Arduino.

- Send the data's to IIS Server with the ESP-8266.
- Use Custom Server to sort the different data's into different channels.
- Analyze the already stored data using TIME SERIES ALGORITHM.
- Display the data's to Doctors and concerning people for relative host.
- Alarm the necessary person for abnormal health situation.

3.7 Implementation

After all the detail discussion about the hardware connections and introduction to Custom Server, in this segment following methodology is used to complete the implementation of this thesis.

3.7.1 Individual data collection

The few things that has to be ensured before implementation of the IOT based project. One of them is to ensure that it is fully automated i.e. without any human interaction. Secondly, it is better to form a nodal based architecture rather than wearable architecture. So, in our implementation each sensor performs as a node and data's are collected from these individual nodes rather than collectively.

The data collection from each node is done through simple Arduino codes. The temperature sensor can measure the temperature of the surrounding. At first, the temperature sensor gives only the room temperature. When the host directly touches the sensor, it will gradually adjust with the host's body temperature and that data is taken to the Arduino. For temperature sensor the data obtained is in Celsius scale. We have connected the pin 5 with arduino which shows the data. So, in the void loop section retrieve that data into a variable "temperature" and then we print that variable in order to see the temperature in the serial monitor [13].

In case of pulse sensor, there are 4 different tabs; the main tab we named is Pulse Sensor Amped Arduino_1.5.0, All serial handling, Interrupt and Timer Interrupt notes. In the main tab we collect the data from the analog zero or A0 pin. Whenever a heartbeat is discovered in the loop of the first tab [14], then in the Serial handling tab, beats per minute (BPM) and inter beat interval (IBI) is calculated. Then the print in the serial monitor is done for all the three different values. The visual serial plotting is then done on the second tab and we can see three different graphs for in the serial plotter [15]. The interrupt handles all the interrupt related issues which differs in different arduino board. The calculation is necessary for Arduino Uno was selected. Timer notes give the detail idea of how to interrupt work in arduino.

3.7.2. Sending data to Custom Server

In order to send data to Custom Server storage, ESP-8266 was used. The only thing needed for sending the data to Custom Server is to merge the ESP-8266 code with every sensor arduino code.

There are some common features of Custom Server that need to be ensured during merging of the code. First, ESP -8266 has a different set of language of its own. So, in order to send those data's we need to follow the language properly; secondly, every code will need the SSID and password to connect to the network. It will need the IP for the Custom Server website. Thirdly, it will need the channel ID and field number of that ID and lastly, whenever we reading or writing data on the field each channel has its own unique data reading and writing number. These are the mandatory things to be integrated with each sensors arduino code. [16]

Basic Codes of Esp-8266

AT+CWMODE: Setting WiFi mode.

AT+CWJAP: Connect to local router.

AT+CIFSR: Query for device IP.

AT+CIPSTART: ESP-8266 connecting to the required server as a client.

AT+CIPSEND: Sending data.

AT+CIPCLOSE: close of data transmission.

a) Modified pseudo code for temperature sensor

Initialize pins
Initialize SSID and Password
Initialize IP for connecting desired website.
Initialize the API key

Setup Method

Initialize Serial Baud
If
Esp8266 connects to wifi
Returns ok

Loop method

Sensor port request for Temperature data
Use update temperature method.

Connect WiFi method

```
Request connection from Esp8266 to WiFi network
If connected
    Print ok
   Return true
Else
     Return false
```

Update Temperature method

```
Check connection.
If connected not found
     Return error
Use IP and Intialize port
Start sending data
Resend .
```

b) Modified pseudo code for pulse sensor

```
Initialize pins
Initialize SSID and Password
Initialize IP for connecting desired website.
Initialize the API key
Declare Variable
```

Setup Method

```
Initialize Serial Baud
If
  Esp8266 connects to wifi
Returns ok
Use interrupt Setup method
```

Loop method

```
Use update beat method.
Print done
```

(IJRST) 2018, Vol. No. 8, Issue No. III, Jul-Sep **e-ISSN: 2249-0604, p-ISSN: 2454-180X**

Connect WiFi method

Request connection from Esp8266 to WiFi network
If connected
Print ok
Return true
Else
Return false

Update beat method

Check connection.
If connected not found
Return error
Use IP and Initialize port
Start sending data
Resend.

Interrupt setup method

Enable interrupts
Read pulse signal
Calculate IBI
Compare IBI with signal
Calculate BPM
Reset variables

.Net duration pseudo code

Declare channel ID
Initialize arrays
For all values of the array Do
Calculate duration between two intervals
Return duration

.Net min,max,average calculation pseudo code

Declare channel ID
Initialize arrays
For all values of the array Do
Calculate maximum, minimum and average

(IJRST) 2018, Vol. No. 8, Issue No. III, Jul-Sep **e-ISSN: 2249-0604, p-ISSN: 2454-180X**

Return values.

3.7.2. Custom Database

Database contains patient data (Temperature and Pulse Rate) and also training dataset for Temperature & Pulse Rate.

Training Dataset

Temperature and Pulse Rate are first trained using Training Dataset and then prediction is mad according to data coming from each channel.

	Entry_ID	Created_At	Value
1	2	2018-05-07 11:06:28 UTC	67
2	3	2018-05-07 11:06:48 UTC	69
3	4	2018-05-07 11:07:07 UTC	120
4	5	2018-05-07 11:07:23 UTC	130
5	6	2018-05-07 11:07:39 UTC	74
6	7	2018-05-07 11:08:09 UTC	0
7	8	2018-05-07 11:08:25 UTC	71
8	9	2018-05-07 11:08:41 UTC	71
9	10	2018-05-07 11:09:00 UTC	75
10	11	2018-05-07 11:09:19 UTC	90
11	12	2018-05-07 11:09:36 UTC	73
12	13	2018-05-07 11:09:54 UTC	68
13	14	2018-05-07 11:10:11 UTC	98
14	15	2018-05-07 11:10:29 UTC	66
15	16	2018-05-07 11:32:16 UTC	0
16	17	2018-05-07 11:32:39 UTC	140
17	18	2018-05-07 11:32:55 UTC	0
18	19	2018-05-07 11:33:12 UTC	70
19	20	2018-05-07 11:33:29 UTC	70
20	21	2018-05-07 11:34:01 UTC	130
21	22	2018-05-07 11:34:19 UTC	74
22	23	2018-05-07 11:34:34 UTC	69
23	24	2018-05-07 11:34:54 UTC	66

Fig 3.8: Trainig DataSet of Pulse from Custom Server

	Entry_ID	Created_At	Value
1	101	2018-05-07 11:06:28 UTC	95.342
2	102	2018-05-07 11:06:48 UTC	95.126
3	103	2018-05-07 11:07:07 UTC	95.27
4	104	2018-05-07 11:07:23 UTC	95.234
5	105	2018-05-07 11:07:39 UTC	95.09
6	106	2018-05-07 11:08:09 UTC	95.09
7	107	2018-05-07 11:08:25 UTC	95.09
8	108	2018-05-07 11:08:41 UTC	95.234
9	109	2018-05-07 11:09:00 UTC	95.27
10	110	2018-05-07 11:09:19 UTC	95.234
11	111	2018-05-07 11:09:36 UTC	95.378
12	112	2018-05-07 11:09:54 UTC	95.162
13	113	2018-05-07 11:10:11 UTC	95.234
14	114	2018-05-07 11:10:29 UTC	95.018
15	115	2018-05-07 11:32:16 UTC	94.982
16	116	2018-05-07 11:32:39 UTC	95.162
17	117	2018-05-07 11:32:55 UTC	95.09
18	118	2018-05-07 11:33:12 UTC	94.298
19	119	2018-05-07 11:33:29 UTC	94.982
20	120	2018-05-07 11:34:01 UTC	95.126
21	121	2018-05-07 11:34:19 UTC	94.874
22	122	2018-05-07 11:34:34 UTC	95.018
23	123	2018-05-07 11:34:54 UTC	94.982

Fig 3.9: Trainig DataSet of Temperature from Custom Server

Patient Heartbeat Database

In database Patient Name and Pulse rate are stored with time and date when it was recorded.

Name	Heartbeat	Id	Created_At
Divya	72	1	2018-05-07 11:0...
Divya	74	2	2018-05-07 11:2...
Divya	75	3	2018-05-07 11:3...
Divya	76	4	2018-05-07 12:0...
Divya	85	5	2018-05-07 14:2...
Karan	85	6	2018-05-16 07:1...
Karan	75	7	2018-05-16 05:0...
Karan	72	8	2018-05-16 05:4...
Karan	79	9	2018-05-16 06:0...
Karan	65	10	2018-05-16 06:3...
Rita	85	1001	2018-05-23 07:1...

Fig 3.10: Patient Heartbeat Database from Custom Server

Patient Temperature Database

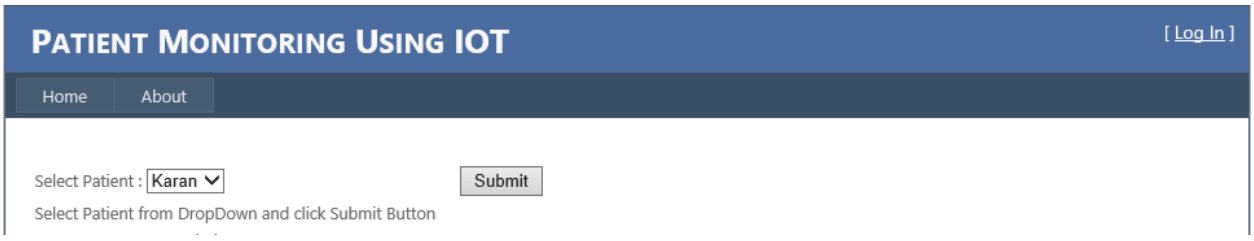
In database Patient Name and Temperature are stored with time and date when it was recorded.

Name	Temperature	Id	Created_At
Karan	96	1	2018-05-07 11:06:28 UTC
Karan	95	2	2018-05-07 11:26:55 UTC
Karan	102	3	2018-05-07 11:34:01 UTC
Karan	95	4	2018-05-07 12:00:12 UTC
Karan	94	5	2018-05-07 14:23:22 UTC
Divya	96	6	2018-05-16 07:12:32 UTC
Divya	97	7	2018-05-16 05:07:05 UTC
Divya	99	8	2018-05-16 05:43:08 UTC
Divya	96	9	2018-05-16 06:01:26 UTC
Divya	95	10	2018-05-16 06:30:26 UTC
Rita	96.7	1001	2018-05-23 07:16:43 UTC

Fig 3.11: Patient Temperature Database from Custom Server

Data Retrieval

The raw data from IIS Server canbe retrieve using an option provided in it called the Export/Import Data. The updated data of the entire field in the corresponding channel can be downloaded. The data which was retrieved are provided in RDBMS.



Temperature Data and Chart

ID	Name	Temperature	Created_At
1	Karan	96	2018-05-07 11:06:28 UTC
2	Karan	95	2018-05-07 11:26:55 UTC
3	Karan	102	2018-05-07 11:34:01 UTC
4	Karan	95	2018-05-07 12:00:12 UTC
5	Karan	94	2018-05-07 14:23:22 UTC

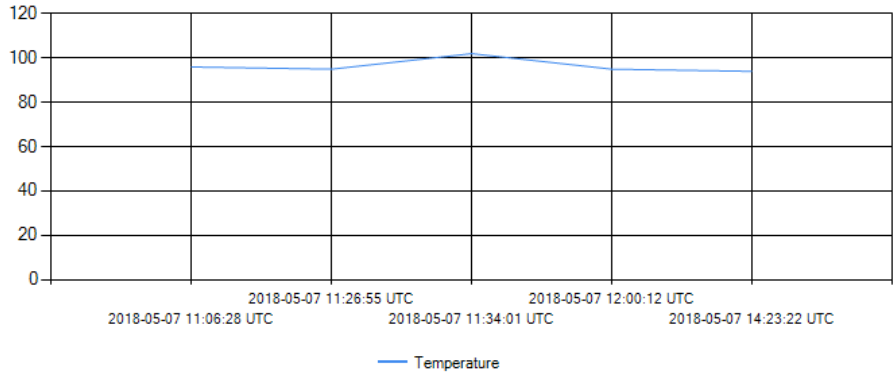


Fig 3.12: Retrieving Temperature data from Custom Server

Heart Beat Data and Chart

ID	Name	Temperature	Created_At
1	Karan	96	2018-05-07 11:06:28 UTC
2	Karan	95	2018-05-07 11:26:55 UTC
3	Karan	102	2018-05-07 11:34:01 UTC
4	Karan	95	2018-05-07 12:00:12 UTC
5	Karan	94	2018-05-07 14:23:22 UTC

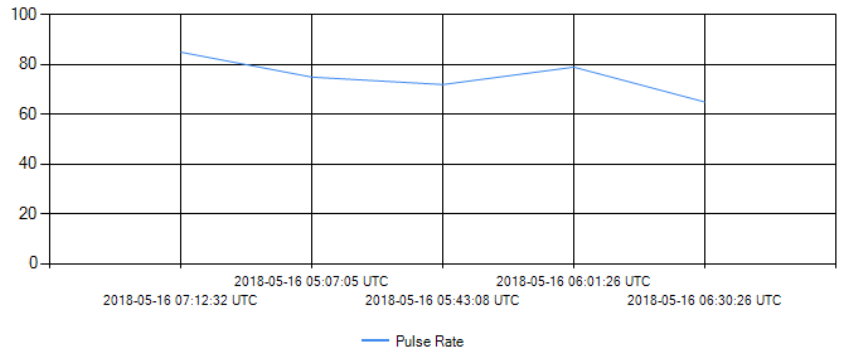


Fig 3.13: Retrieving Pulse Rate data from Custom Server

4. RESULT AND DISCUSSION

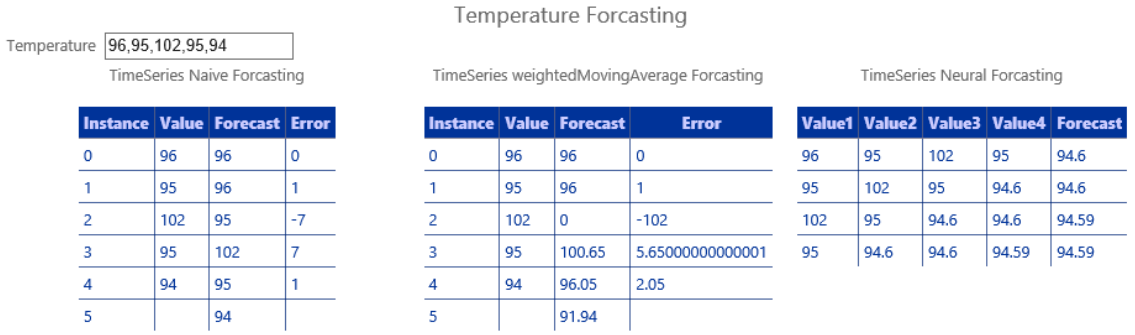
In this segment after implementing the code we will discuss how it will display in the website account of a user and also it will forecast the upcoming temperature and heart beat of a host using time series and naïve algorithm. The data's obtained after transmitting is saved

in the cloud and from there with the help the features provided from Custom Server we are able to display data and send alarming message and prediction.

4.1 Result Display

The Custom Server takes 15 second for each data entry. The data entered in the storage is then graphically portrayed in the display. The data entered in the storage is channel and field specific. That means it will go the specific field of that channel that is given by the user. For, convenience of the doctor of the elderly, a separate channel has beenopened for them. The patient who is under observation of that doctor is then placed on different fields. According to fields, forecasting was done and intimate the concern person in case of alarming situation.

For multiple hosts, multiple doctors were selected in this case and the display is absolutely privacy protected. That is, the user can keep the channel open for public viewing or it can be made private viewing for convenience. After selecting the specified channel the user can see the following field of the host's daily updated graphs for monitoring.

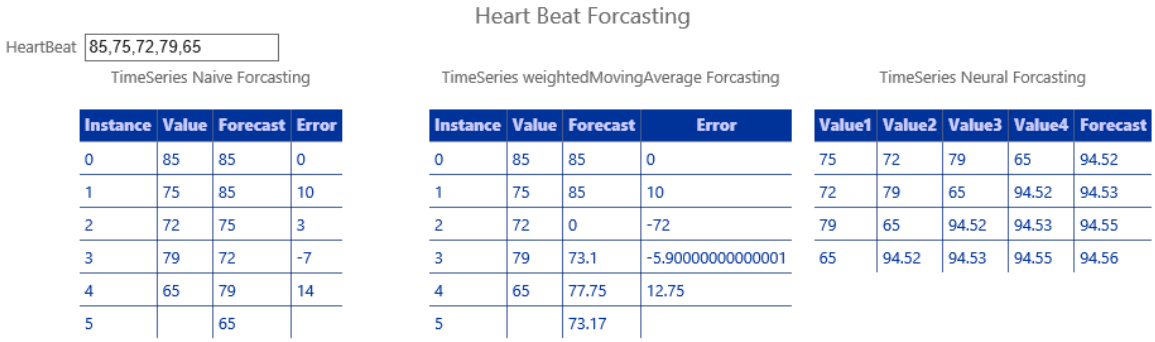


Instance	Value	Forecast	Error
0	96	96	0
1	95	96	1
2	102	95	-7
3	95	102	7
4	94	95	1
5		94	

Instance	Value	Forecast	Error
0	96	96	0
1	95	96	1
2	102	0	-102
3	95	100.65	5.65000000000001
4	94	96.05	2.05
5		91.94	

Value1	Value2	Value3	Value4	Forecast
96	95	102	95	94.6
95	102	95	94.6	94.6
102	95	94.6	94.6	94.59
95	94.6	94.6	94.59	94.59

Fig 4.1 Temperature Forecasting Result



Instance	Value	Forecast	Error
0	85	85	0
1	75	85	10
2	72	75	3
3	79	72	-7
4	65	79	14
5		65	

Instance	Value	Forecast	Error
0	85	85	0
1	75	85	10
2	72	0	-72
3	79	73.1	-5.90000000000001
4	65	77.75	12.75
5		73.17	

Value1	Value2	Value3	Value4	Forecast
75	72	79	65	94.52
72	79	65	94.52	94.53
79	65	94.52	94.53	94.55
65	94.52	94.53	94.55	94.56

Fig 4.2 Pulse Rate Forecasting Result

4.1.1 Temperature graph

The temperature of a random is shown below. Here the peak value which was taken at an approximate time 2:40-2:50 PM GMT+6. The value shown here is seen to be slowly getting saturated with the body temperature and showing the peak value at about 98 degree Fahrenheit.

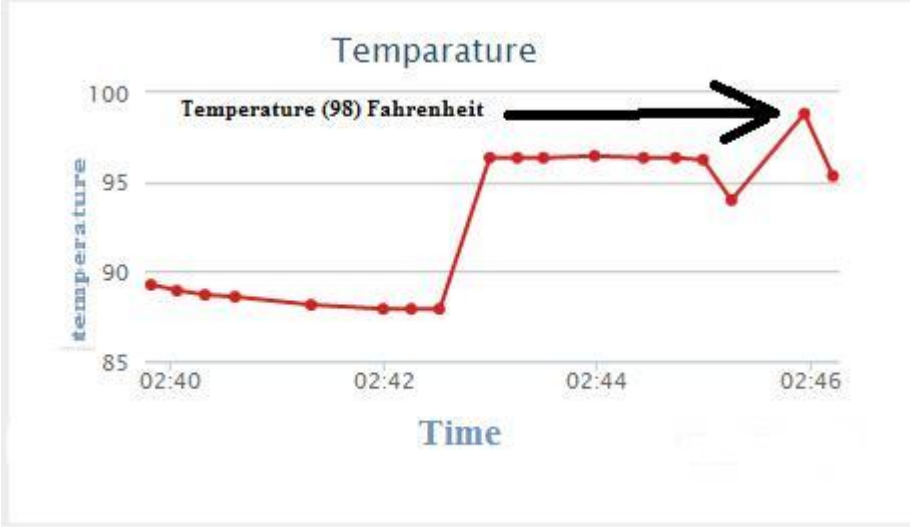


Fig 4.3: Temperature graph of a host

The temperature data received is in degree Celsius scale, which is converted into degree Fahrenheit scale using .Net. So, our retrieved data for temperature remains in Celsius scale.
The following figure shows tan increase of temperature with a human touch and again goes back to room temperature when touch is removed along with the time of entry.

	A	C
1	created_at	Temperature
2	2017-03-01 04:36:35 UTC	96.5
3	2017-03-01 04:37:10 UTC	97.5
4	2017-03-01 04:37:43 UTC	87
5	2017-03-01 04:38:17 UTC	86
6	2017-03-01 04:38:52 UTC	86
7	2017-03-01 04:41:56 UTC	86
8	2017-03-01 04:42:30 UTC	86
9	2017-03-01 04:43:04 UTC	87
10	2017-03-01 04:43:38 UTC	93
11	2017-03-01 04:44:12 UTC	95.5
12	2017-03-01 04:44:47 UTC	96
13	2017-03-01 04:45:23 UTC	96
14	2017-03-01 04:45:55 UTC	96
15	2017-03-01 07:07:19 UTC	96
16	2017-03-01 07:07:48 UTC	96
17	2017-03-01 07:08:22 UTC	96
18	2017-03-01 07:08:58 UTC	96
19	2017-03-01 07:09:30 UTC	96
20	2017-03-01 07:10:04 UTC	96
21	2017-03-01 07:10:38 UTC	97
22	2017-03-01 07:12:25 UTC	98
23	2017-03-01 07:12:57 UTC	98.5
24	2017-03-01 07:13:31 UTC	98
25	2017-03-01 07:14:05 UTC	95.5

Fig 4.4: The retrieved temperature in Celsius scale

4.1.2 Pulse graph

The pulse graph which is taken at first gives three different values of parameters, which are already discussed above. The first figure shows the three different column, the first column being the pulse rate, the second being the IBI and the third being the pulse signal.

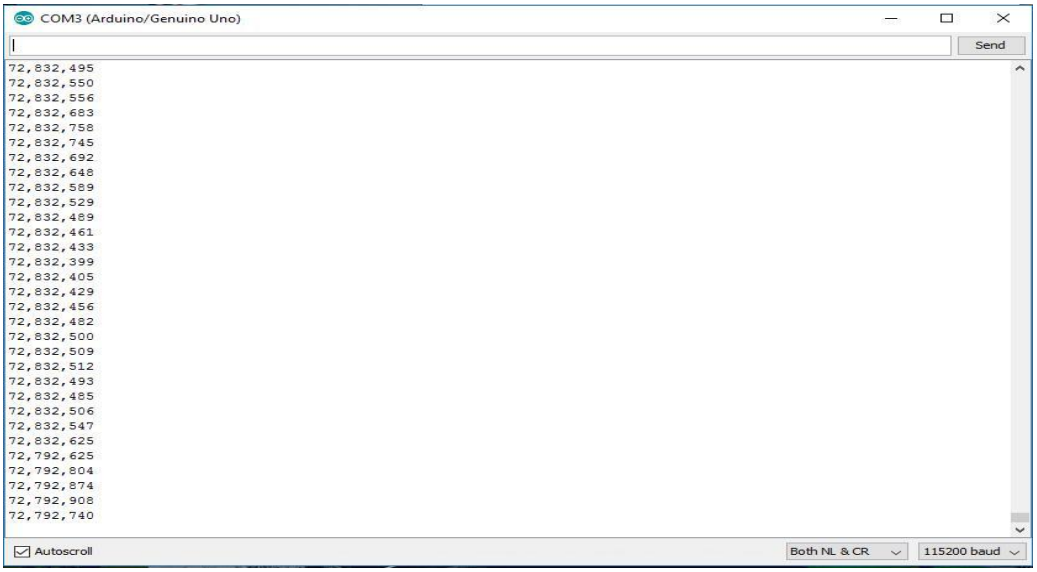


Fig 4.5: The initial values of from the serial monitor of arduino.

In the serial plotter of arduino the following graph is obtained. The blue colored indicates the pulse rate, the red color shows the Inter beat Interval (IBI) and the green graph shows the pulse signal.

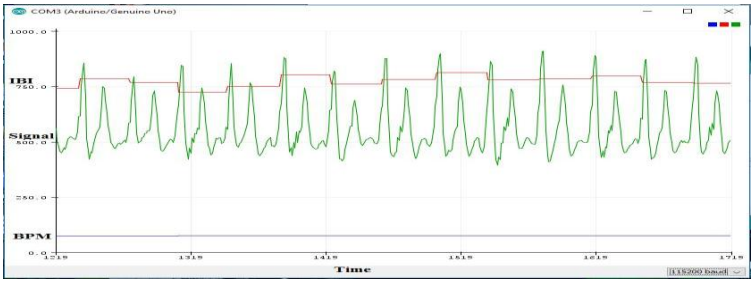


Fig 4.6: Serial plotter view of the data received from pulse sensor.

From this data set for the convenience of this project, only the pulse rate data was selected as a criterion for monitoring. After implementing the ESP-8266 code with this pulse sensor data, it is formed in the Custom Server as following graph of a host. Initially there are some values which are a little high for noise factor. But, with time normal pulse starts to appear as seen in the following graph.

Fig 4.7: The Custom Server graph of a host corresponding to the plotter graph

The corresponding raw data value that is retrieved from the Custom Server can be seen in a .CSV format. Initially, the pulse rate is unusually high for high noise margin but with time the noise margin stabilizes.

	A	D
63	2017-03-05 01:48:11 UTC	98
64	2017-03-05 01:48:45 UTC	105
65	2017-03-05 01:49:03 UTC	104
66	2017-03-05 01:49:29 UTC	98
67	2017-03-05 01:50:13 UTC	94
68	2017-03-05 01:50:37 UTC	102
69	2017-03-05 01:50:55 UTC	94
70	2017-03-05 01:51:49 UTC	97
71	2017-03-05 01:52:33 UTC	99
72	2017-03-05 01:52:57 UTC	92
73	2017-03-05 01:53:31 UTC	102
74	2017-03-05 01:53:59 UTC	100
75	2017-03-05 01:54:47 UTC	100
76	2017-03-05 01:55:15 UTC	99
77	2017-03-05 01:55:39 UTC	106
78	2017-03-05 01:55:57 UTC	100
79	2017-03-05 01:56:45 UTC	103
80	2017-03-05 01:57:10 UTC	100
81	2017-03-05 01:57:28 UTC	103
82	2017-03-05 01:57:52 UTC	99
83	2017-03-05 01:58:26 UTC	101
84	2017-03-05 01:58:50 UTC	97
85	2017-03-05 01:59:54 UTC	95
86	2017-03-05 02:00:28 UTC	97
87	2017-03-05 02:00:52 UTC	92

Fig 4.8: The retrieved BPM data along with time duration

4.2 Reaction display

The reactions are displayed through the e-mail and social networking site Twitter. The Custom Server have a privilege of using react and ThinkHTTP option. The react option works on channel and field specific. It requires the reference data of the host, whenever the reference data is crossed a tweet or message on e-mail will be sent. Tweet sending is relatively easy with react option of Custom Server but in order to send an e-mail a third

party website Push Inbox is used for out project. ThinkHTTP helps to connect the react panel with the Push Inbox site. And then the message is then sent to the desired e-mail address through Push Inbox site.

4.3 Cost Analysis

The only cost that was needed for the project was the hardware cost, on the other hand other project like the **Masimo Radical-7** and **Free Scale Home Health Hub reference platform** needs costlier hardware with more limited features. So, it is safe to saythat our project is more cost effective, with more features.

4.4 Challenges

There are three basic challenges we face during the project implementation.

ESP-8266 WiFi Module

The wifi module works in its own specific language. So, a lot of problem was faced interpreting its language. Sometimes the wifi module itself cannot connect with the local network as a result the data sending got interrupted many times. So, a better hardware support for wifi module is expected to send data smoothly.

Pulse Sensor

The data that is retrieved from the pulse sensor can sometime give some readings with error. Due to unavailability of better pulse sensor in our country, a better sensor may be procured from outside for efficient workability. This delayed the implementation of the project in some ways as pulse sensor is an integral part of this project.

Data Analysis and React

The data analysis with .NET of Custom Server and react of the Custom Server gave us some challenges. Especially with alarm message, it was difficult to link with the Push Inbox and with twitter.

4.5 Future works

- Integrating more sensors for more specific data acquisition and analysis.
- Will be applicable in army services in active situation.
- Will be used to provide health service to rural areas in affordable price.

- Huge database will be built for doctors to diagnose people from different areas and cultures.

Our project can be considered as platform developed in the field of IoT for the health sector. In developing countries like ours, this kind of innovative and cost effective project can improve the future of technology. So, we are looking forward to implement the project in order to make an impact in the new era of technology.

CONCLUSION

Our main objective in this project was to successfully monitor the basic two criteria's namely temperature, pulse and to analyze the upcoming temperature and heartbeat of host; so, that disease would be predicted in earlier stage and monitor react during emergency situation without any human interaction. We wanted to make a mark on the field of IoT with the health sector. With the rise of IoT, an era of technology is moving towards a far superior dimension. In order to keep pace with the new technologies, this project can surely make a way for the advancement in this sector. Though our model is tested and implemented, it will be difficult to continue the project without superior quality hardware support along with a lot of new integration. The real benefit of this work can only be fully realized when it can be implemented in full scale.

REFERENCES

[1] Brock, D. L. (2001). The electronic product code (epc). Auto-ID Center White Paper MIT-AUTOID-WH-002.

[2] Ni, Q., García Hernando, A. B., & de la Cruz, I. P. (2015).[2] The elderly's independent living in smart homes: A characterization of activities and sensing infrastructure survey to facilitate services development[2] . Sensors, 15(5), 11312-11362

[3] Ray, P. P. (2014, November).[3] Home Health Hub Internet of Things[3] (H 3 IoT): an architectural framework for monitoring health of elderly people. In Science Engineering and Management Research (ICSEMR), 2014 International Conference on (pp. 1-3). IEEE.

[4] Niewolny, D. (2013). How the internet of things is revolutionizing healthcare. White paper.

[5] Patel, S., Park, H., Bonato, P., Chan, L., & Rodgers, M. (2012).[5] A review of wearable sensors and systems with application in rehabilitation[5] . Journal of neuroengineering and rehabilitation, 9(1), 21

[6] De Battista, N., Rice, J. A., Sim, S. H., Brownjohn, J. M. W., & Tan, h. p. structural health monitoring of civil infrastructure using wireless sensor networks.

[7] Hassanalieragh, M., Page, A., Soyata, T., Sharma, G., Aktas, M., Mateos, G., ... & Andreescu, S. (2015, June). Health monitoring and management using Internet-of-Things (IoT) sensing with cloud-based processing: Opportunities and challenges. In Services Computing (SCC), 2015 IEEE International Conference on (pp. 285-292). IEEE.

[8] Tyagi, S., Agarwal, A., & Maheshwari, P. (2016). A conceptual framework for IoT-based healthcare system using cloud computing. In cloud system and big data engineering (confluence), 2016 6th international conference (pp. 503-507). IEEE.

[9] https://wiki.eprolabs.com/index.php?title=DS18B20_Temperature_Sensor

[10]http://www.hobbytronics.co.uk/pulse-rate-sensor

[11]https://en.wikipedia.org/wiki/Arduino_Uno

[12]https://en.wikipedia.org/wiki/ESP8266

[13]https://create.arduino.cc/projecthub/everth-villamil-ruiz/temperature-sensor-ds18b20-3decfc

[14] https://pulsesensor.com/pages/code-and-guide

[15] https://github.com/worldfamouselectronics/pulsesensor_amped_arduino

[16] https://circuitdigest.com/microcontroller-projects/sending-arduino-data-to-webpage