Smart HealthCare Patient Monitoring System

Project Report (CSE3009 –Internet of Things)

submitted by

Pranav Bhasin
15BCE0097
Karan Verma
15BCE0098
Naren Adithya
15BCE0124

4TH SEMESTER (CAL)

Computer Science

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

May & 2017

Abstract

The Healthcare industry remains among the fastest to adopt the Internet of Things. The reason for this trend is that integrating IoT features into medical devices greatly improves the quality and effectiveness of service, bringing especially high value for the elderly, patients with chronic conditions, and those requiring constant supervision. According to some estimates, spending on the Healthcare IoT solutions will reach a staggering $1 trillion by 2025 and, hopefully, will set the stage for highly personalized, accessible, and on-time Healthcare services for everyone. Networked sensors, either worn on the body or embedded in our living environments, make possible the gathering of rich information indicative of our physical and mental health. Captured on a continual basis, aggregated, and effectively mined, such information can bring about a positive transformative change in the health care landscape. The IoT is used by clinical care to monitor physiological statuses of patients through sensors by collecting and analyzing their information and then sending analyzed patient’s data remotely to processing centers to make suitable actions. Not only for patients, it also useful for normal people to check the health status by using wearable devices with sensors. This project concentrates to make a healthcare application of iot using DY-039, DB1820 sensors and a python script as the middle-ware.

Literature Survey

Arduino/Genuino 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. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. You can tinker with your UNO without worrying too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again.

KY-039 SENSOR
The sensor is made to shine an infrared led through your finger. The infrared sensor on the other side can then pick up slight changes in the light transmittance through your finger when blood is pumped in. When the sensor senses light, it becomes more resistant, causing the voltage reading in the serial monitor to go down.

DS18B20 SENSOR
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 (and ground) for communication with a central microprocessor. In addition, the DS18B20 can derive power directly from the data line ("parasite power"), eliminating the need for an external power supply. Each DS18B20 has a unique 64-bit serial code, which allows multiple DS18B20s to function on the same 1-Wire bus. Thus, it is simple to use one microprocessor to control many DS18B20s distributed over a large area. Applications that can benefit from this feature include HVAC environmental controls, temperature monitoring systems inside buildings, equipment, or machinery, and process monitoring and control systems.

Python As Middle-Ware
Middleware components provide a way to execute logic before the framework routes each request, after each request is routed but before the target responder is called, or just before the response is returned for each request. Components are registered with the *middleware* kwarg when instantiating Falcon’s API class.

Python Paste

Python Paste, often simply called **paste**, is a set of utilities for web development in Python. Paste has been described as "a framework for web frameworks".

WSGI MIDDLE-WARE

The WSGI standard is an interface that allows applications to use Python code to handle HTTP requests. A **WSGI application** is passed a Python representation of an HTTP request by an application, and returns content which will normally eventually be rendered by a web browser. A common use for this is when a web server serves content created by Python code.

There are, however, other uses: **WSGI middleware** is Python code that receives a WSGI request and then performs logic based upon this request, before passing the request on to a WSGI application or more WSGI middleware. WSGI middleware appears to an application as a server, and to the server as an application. This is analogous to the function of pipes on UNIX systems. Functionality provided by WSGI middleware may include authentication, logging, URL redirection, creation of sessions, and compression.

Wubby

Wubby (pronounced Wha-bee) is a software platform that simplifies the development of IoT devices by providing a programming environment that supports Python code execution directly in the devices microcontroller. This introduces several advantages:

a) It allows a broader set of developers to contribute, giving them the opportunity to design and develop new everyday objects based on a popular programming language like Python.

b) It speeds up the development process

c) It reduces development costs

d) It results in smarter, interoperable everyday objects Wubby separates hardware from software, abstracting the hardware complexity, while at the same time allowing developers to contribute by writing simple python scripts, rather than having to deploy the whole device image.

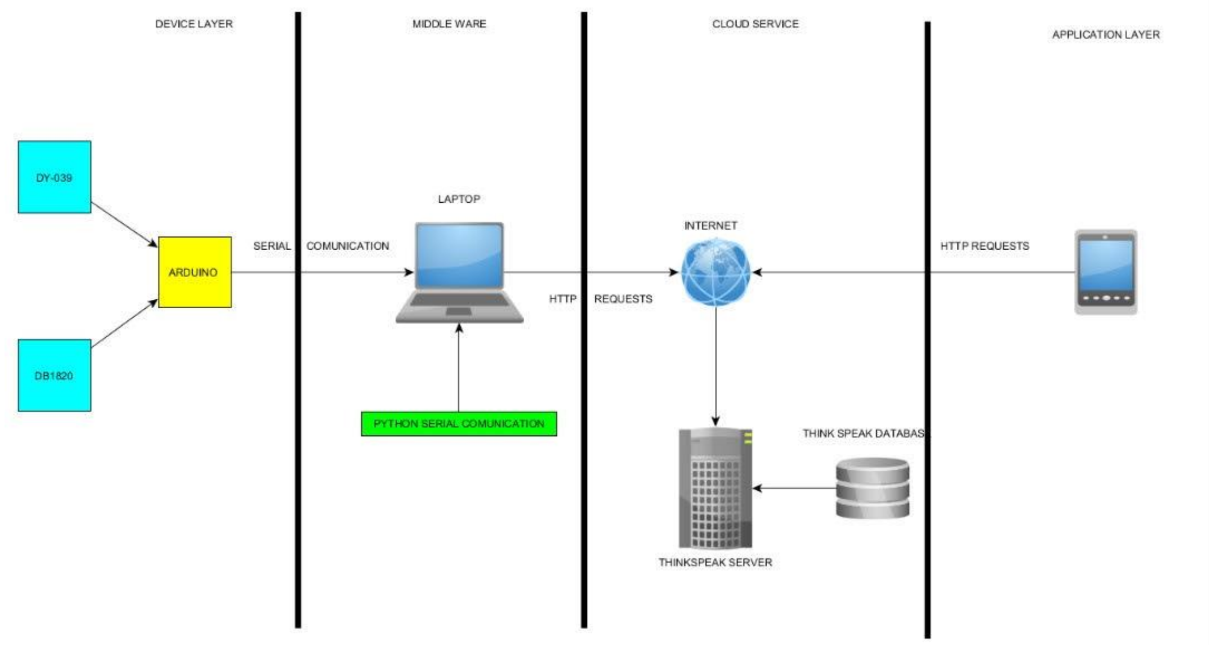
System Architecture of the existing technology:

Data Acquisition is performed by multiple wearable sensors that measure physiological biomarkers, such as ECG, skin temperature, respiratory rate, EMG muscle activity, and gait (posture). The sensors connect to the network though an intermediate data aggregator or concentrator, which is typically a smart phone located in the vicinity of the patient.

The Data Transmission components of the system are responsible for conveying recordings of the patient from the patient's house (or any remote location) to the data center of the Healthcare Organization (HCO) with assured security and privacy, ideally in near real-time. Typically, the sensory acquisition platform is equipped with a short range radio. Components of a remote patient monitoring system that is based on an IoT-Cloud architecture. as Zigbee or low-power Bluetooth, which it uses to transfer sensor data to the concentrator. Aggregated data is further relayed to a HCO for long term storage using Internet connectivity on the concentrator, typically via a smartphone's WiFi or cellular data connection. Sensors in the data acquisition part form an Internet of Things (IoT)-based architecture as each individual sensor's data can be accessed through the Internet via the concentrator. Often a storage/processing device in vicinity of a mobile client, sometimes referred to as a cloudlet, is used to augment its storage/processing capability whenever the local mobile resources do not fulfill the application's requirements. The cloudlet can be a local processing unit (such as a desktop computer) which is directly accessible by the concentrator through WiFi network. In addition to providing temporary storage prior to communication of data to the cloud, the cloudlet can also be used for running time critical tasks on the patient's aggregated data. Moreover, the cloudlet can be used to transmit the aggregated data to the cloud in case of limitations on the mobile device such as temporary lack of connectivity or energy.

Cloud Processing has three distinct components: storage, analytics, and visualization. The system is designed for long term storage of patient's biomedical information as well assisting health professionals with diagnostic information. Cloud based medical data storage and the upfront challenges have been extensively addressed in the literature. Analytics that use the sensor data along with e-Health records that are becoming prevalent can help with diagnoses and prognoses for a number of health conditions and diseases. Additionally, Visualization is a key requirement for any such system because it is impractical to ask physicians to pore over the voluminous data or analyses from wearable sensors. Visualization methods that make the data and analyses accessible to them in a readily digestible format are essential if the wearable sensors are to impact clinical practice.

System Architecture



Device layer
This contain Arduino, Ky-039, db1820. This is layer in which Data acquisition take place .As this layer plays important part in reliability of performance of the whole system a lot of concern as to be made sure data collected here is authenticated. Data presentation is important function of this layer, this layer restricts the decimal point of data to 1. as we see this device only has two sensor Ky-039 and db1820
Ky-039: this sensor is used for monitoring the heart pulse .this basically identifies the time interval change in high pules which it detects with change in infrared rays emitted by it
Ds18b20: this is high resection sensor it has ability to detect temperature at precision of 6 digits. This has been used monitor the temperature of the person

Middleware
This component of IOT enabled device which make communication possible between the device layer and cloud .we have used python serial communication monitor as middle ware. Python package of serial is used. Python codes monitor the serial communication out from the Arduino of device layer and obtain the data sent by it and send the data to cloud service as HTTP Request. The message are send with TCP/IP protocol communication via HTTP Request. This method is opted as it is very reliable

Cloud Services
We opted to use the open source service provided by The MathWorks,Inc which is Thingspeak.it is an Application programming interface .it is well

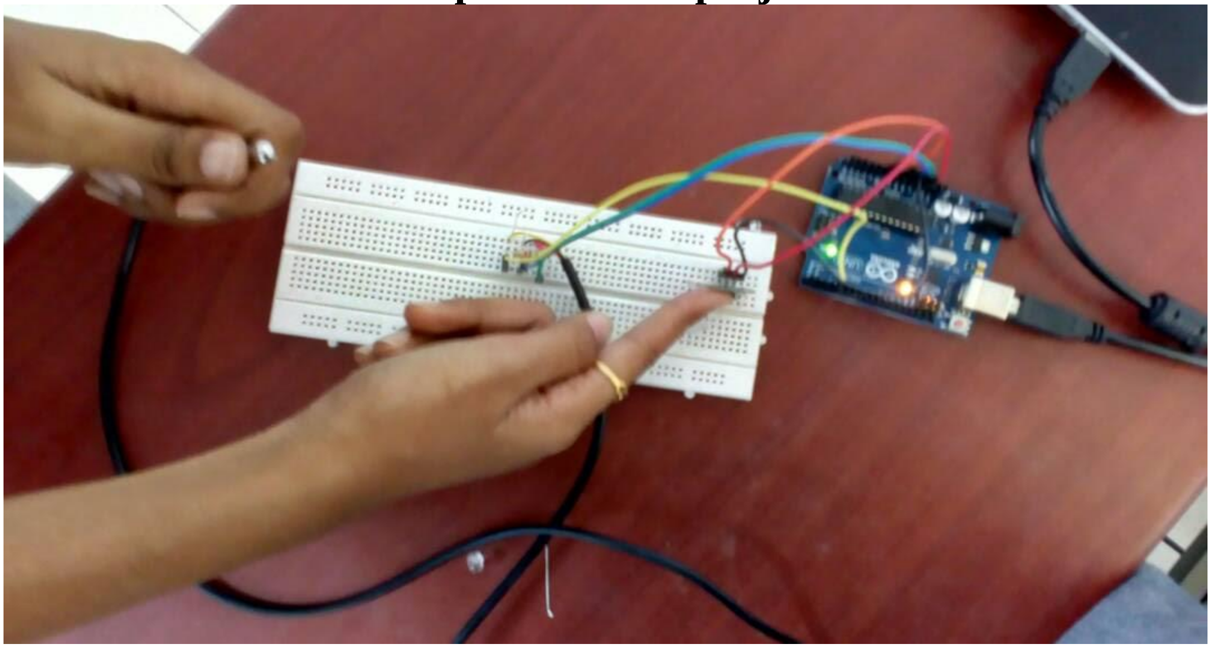
secure and provide various platform like visualization and analysis of data acquired.it provides security measure by providing unique and secure API key for reading and writing

Application Layer
An Android based application ThingView Free and Iot thingSpeak Monitor Widget .Both application are a freeware, they application are tuned to alert if there is sensor signals which can lead to medical fatalities.

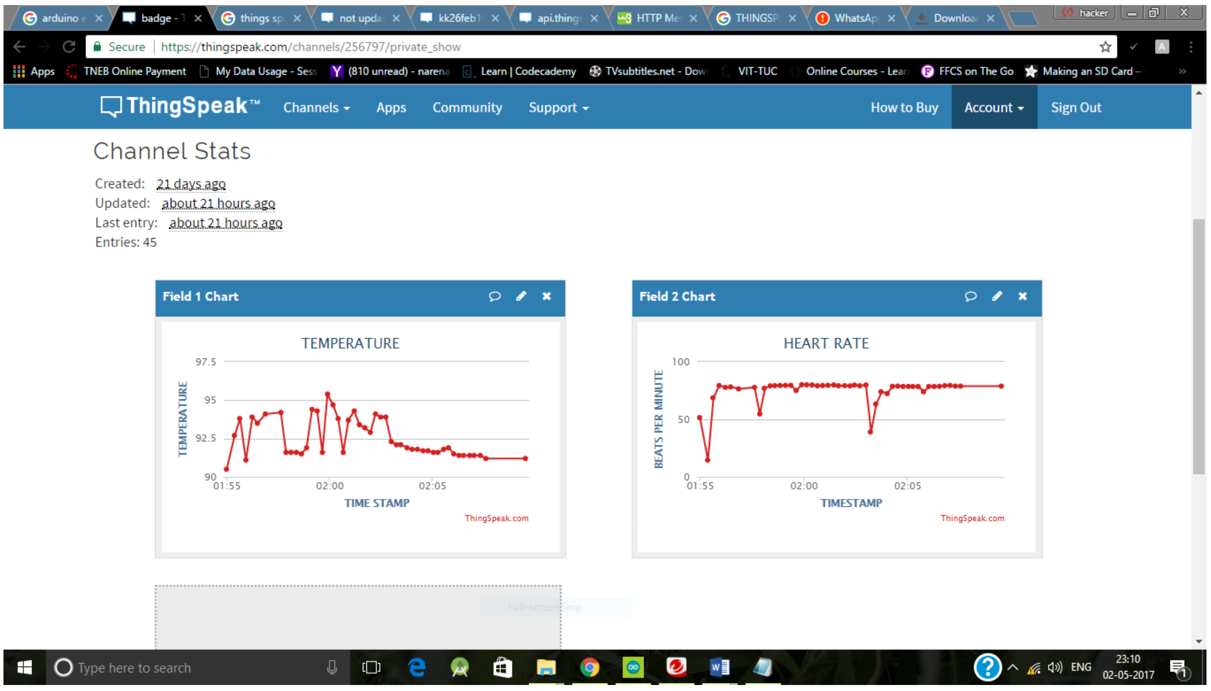
Innovative Idea in Our Project

This project actually makes remote monitoring of patients possible. As we are concentrating more on wearable tech this will be helpful for not only patients but also normal users who want to monitor their health status.
The innovation here is use of python as middle-ware rather than using a wifi module. Python being a versatile language does serial transmission with the Arduino and connects to the thinkspeak server.

Snapshots of the project



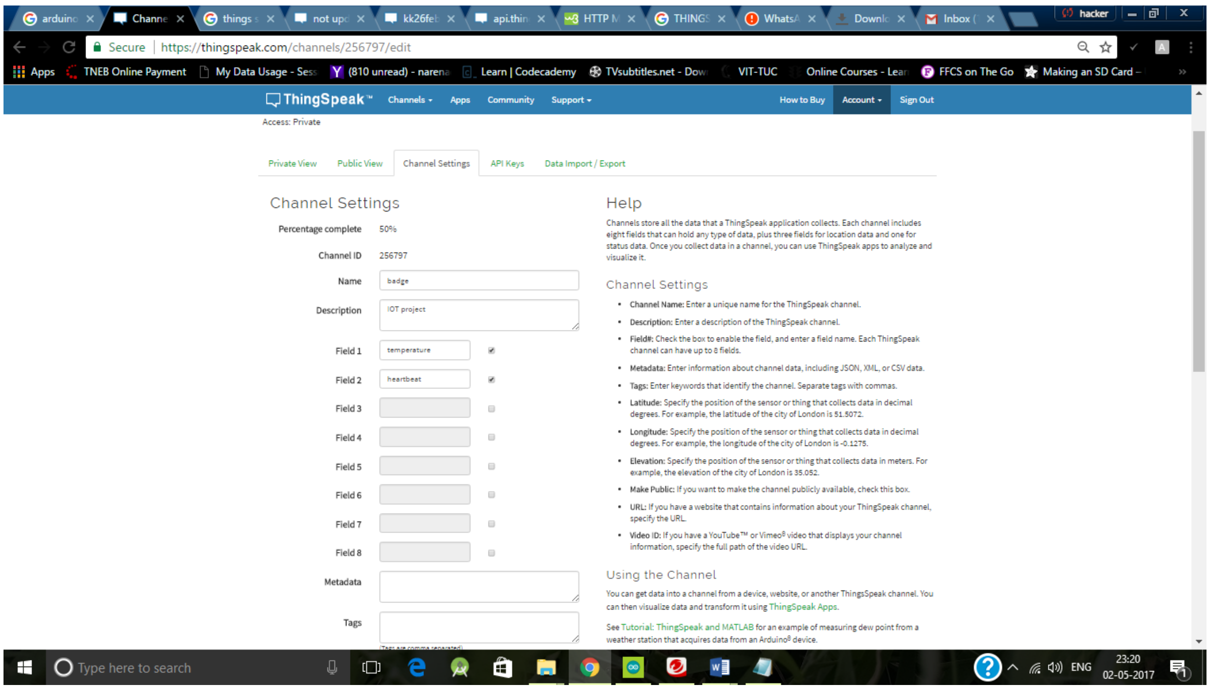
THINKSPEAK DATA GRAPH



Channel Stats

Created: 21 days ago
Updated: about 21 hours ago
Last entry: about 21 hours ago
Entries: 45

Channel settings



https://thingspeak.com/channels/256797/edit

Access: Private

Private View | Public View | Channel Settings | API Keys | Data Import / Export

Channel Settings

Percentage complete	50%
Channel ID	256797
Name	badge
Description	IOT project
Field 1	temperature ✓
Field 2	heartbeat ✓
Field 3	
Field 4	
Field 5	
Field 6	
Field 7	
Field 8	
Metadata	
Tags	

Help

Channels store all the data that a ThingSpeak application collects. Each channel includes eight fields that can hold any type of data, plus three fields for location data and one for status data. Once you collect data in a channel, you can use ThingSpeak apps to analyze and visualize it.

Channel Settings

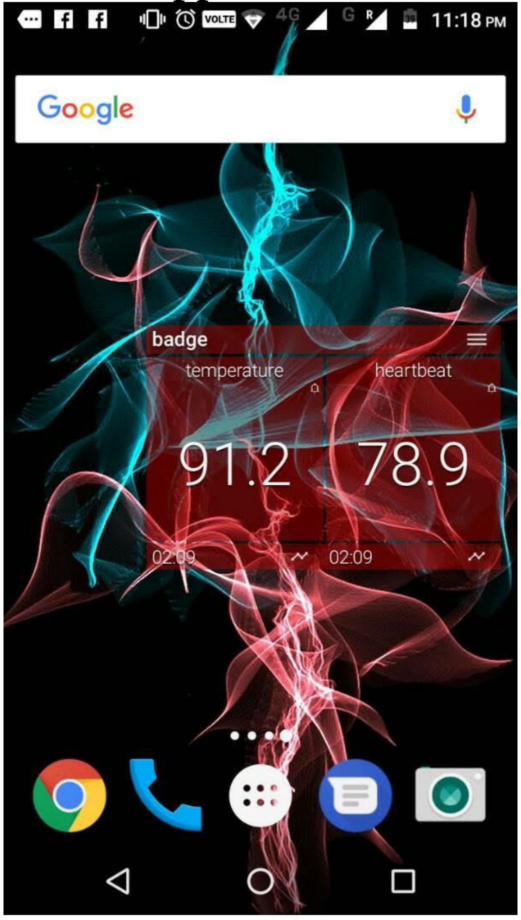
- **Channel Name:** Enter a unique name for the ThingSpeak channel.
- **Description:** Enter a description of the ThingSpeak channel.
- **Field#:** Check the box to enable the field, and enter a field name. Each ThingSpeak channel can have up to 8 fields.
- **Metadata:** Enter information about channel data, including JSON, XML, or CSV data.
- **Tags:** Enter keywords that identify the channel. Separate tags with commas.
- **Latitude:** Specify the position of the sensor or thing that collects data in decimal degrees. For example, the latitude of the city of London is 51.5072.
- **Longitude:** Specify the position of the sensor or thing that collects data in decimal degrees. For example, the longitude of the city of London is -0.1275.
- **Elevation:** Specify the position of the sensor or thing that collects data in meters. For example, the elevation of the city of London is 35.052.
- **Make Public:** If you want to make the channel publicly available, check this box.
- **URL:** If you have a website that contains information about your ThingSpeak channel, specify the URL.
- **Video ID:** If you have a YouTube™ or Vimeo® video that displays your channel information, specify the full path of the video URL.

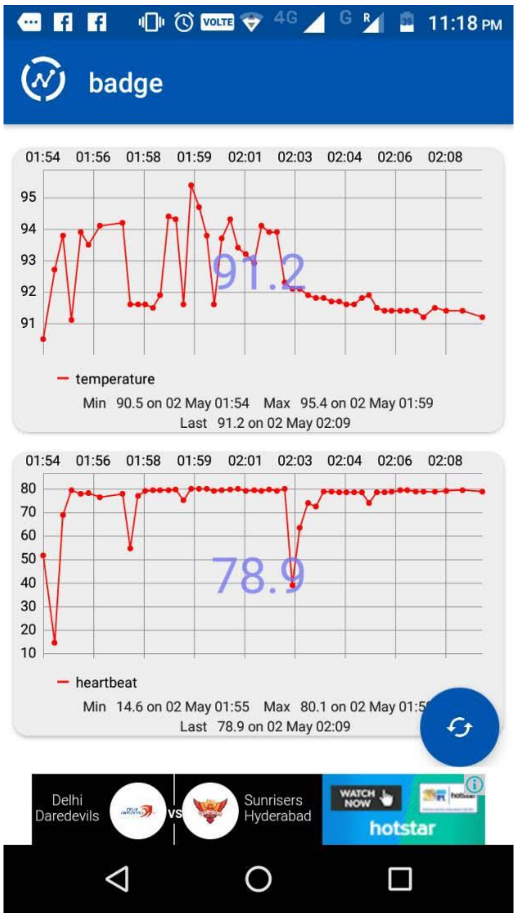
Using the Channel

You can get data into a channel from a device, website, or another ThingsSpeak channel. You can then visualize data and transform it using ThingSpeak Apps.

See Tutorial: ThingSpeak and MATLAB for an example of measuring dew point from a weather station that acquires data from an Arduino® device.

Mobile Applications to monitor the data





— temperature
Min 90.5 on 02 May 01:54 Max 95.4 on 02 May 01:59
Last 91.2 on 02 May 02:09

— heartbeat
Min 14.6 on 02 May 01:55 Max 80.1 on 02 May 01:5
Last 78.9 on 02 May 02:09

Sample Code Of Our Project

Arduino Code:

```
#include <OneWire.h>
#include <DallasTemperature.h>
#define ONE_WIRE_BUS 4
OneWire oneWire(ONE_WIRE_BUS);
float temp;
DallasTemperature sensors(&oneWire);
int sensorPin = A0;
double alpha = 0.75;
int period = 2000;
double change = 0.0;
double minval = 0.0;
void setup ()
{
  Serial.begin (9600);
  pinMode(A0,INPUT);
 sensors.begin();
}
void loop ()
{
    static double oldValue = 0;
    static double oldChange = 0;

    int rawValue = analogRead (A0);
    double value = alpha * oldValue + (1 - alpha) * rawValue;

    oldValue = value;
     sensors.requestTemperatures();
  temp=sensors.getTempCByIndex(0);
  temp=(temp*1.8)+32;
 Serial.print(temp);
 Serial.print(",");
 Serial.println(value);


    delayMicroseconds(100);
}
```

Code for the middleware

Python Code:

```
import serial
import time
import httplib2
ser = serial.Serial('COM3', 9600)
ser.baudrate = 9600
a = 0
b = 0
while True:
	ch = ser.readline()
	a = float(ch[0:4])
	b = float(ch[6:10])
	conn = httplib2.Http()
	conn.request("http://api.thingspeak.com/update?key=88O2KI49
L8SZ1XWA&field1={0}&field2={1}".format(a,b))
	print "sent {0} {1}".format(a,b)
	time.sleep(1)
```

CONCLUSION AND FUTURE WORK

The healthcare system developed by us can have numerous applications. The system can be used for fixed hospital wards and even for mobile ambulances. As the system only needs to be connected to a network, the design is therefore flexible and easily accessible. These types of systems can not only be implemented in hospitals but also at home places where a person needs to have immediate medical attention whenever his/her health goes unstable. Furthermore, this is a cloud based development and hence requires no physical storage to store data. This also allows to keep a track of patients' heartbeats and body temperature value with change in time, meals or surrounding atmosphere. This would give doctor a more wide perspective of treating the patient in a much effective way within less time. Clearly a better treatment needs better observation and that is what this model aims to achieve. The development of this software has been done keeping in mind the necessary real life scenarios that might occur during cases of heart attacks. The increased heart rate can be used to enable notifications via GSM modules on the doctor's handset. This is indeed a smart treatment for any patient.

In future, this project can be used for treatment of people in villages due to it’s low cost and effective results. Also, if a system that monitors blood pressure can be merged with this, that would result in a perfectly complete system that would fetch a patient’s critical values of basic yet important traits. Also, if further development is done, a medicine log for the patient can also be encapsulated in the same application. Adding even more sposciated sensors will increase the efficientciy of the product. This would make the patient somewhat self dependent and prepared for emergency situations.

References

1. A REVIEW PAPER ON SMART HEALTH CARE SYSTEM USING INTERNET OF THINGS by Zankhana Mehul Kalarthi, International Journal of Research in Engineering and Technology
2. https://www.tundraware.com/Technology/Python-Is-Middleware/
3. IoT for Healthcare by B. Sobhan Babu1 , K. Srikanth2 , T. Ramanjaneyulu3 , I. Lakshmi Narayana4, International Journal of Research in Engineering and Technology
4. Health Monitoring and Management Using Internet-of-Things (IoT) Sensing with Cloud-based Processing: Opportunities and Challenges , 2015 IEEE International Conference on Services Computing
5. https://www.researchgate.net/publication/302596849_Engineering_IoT_Healthcare_Applications_Towards_a_Semantic_Data_Driven_Sustainable_Architecture
6. https://www.kaaproject.org/healthcare/
7. Extension to Middleware for IoT Devices, with Applications in Smart Cities,2015 IEEE International Conference on Services Computing