

PROJECT:

SMART HEALTH MONITORING SYSTEM FOR ELDERLY PEOPLE

TEAM NAME :RANGERS

MEMBERS:

1. BUYYANI SHRUTHI
2. KOWSHIK KRISHNA
3. MARY PRASHANTHI



ABSTRACT:

The idea of Internet of Things incorporates the usage of electronic devices that monitor data and are associated to a cloud (private or public), facilitating them to automatically precipitate certain events. The Internet of Things is slowly influencing on healthcare sector which includes both the doctor and patient fronts. The IOT also helps transform patient care at home. In particular, the Internet of Things (IOT) has been broadly applied to couple various available medical resources and provide adequate, trustworthy and smart healthcare utility to the elderly patients with a persistent illness. Sensor technology provides tracking of medical devices and improves overall efficiency. By the use of remote patient monitoring, costs can be reduced by reducing the number of hospital visits. The proposed work demonstrates the health monitoring of an elderly person based on IoT, by collecting heart rate, temperature and blood pressure values from sensors, processing the data through microcontrollers and stores the data on the cloud. On the other hand the doctor can access the patient’s heart rate and temperature values through the web page and the message is sent to the patient/caretaker.

Due to a rapidly increasing aging population and its associated challenges in health and social care, Ambient Assistive Living has become the focal point for both researchers and industry alike. The need to manage or even reduce healthcare costs while improving the quality of service is high government agendas. Although, technology has a major role to play in achieving these aspirations, any solution must be designed, implemented and validated using appropriate domain knowledge. In order to overcome these challenges, the remote real-time monitoring of a person’s health can be used to identify relapses in conditions, therefore, enabling early intervention. Thus, the development of a smart healthcare monitoring system, which is capable of observing elderly people remotely, is the focus of the research presented in this paper. The technology outlined in this paper focuses on the ability to track a person’s physiological data to detect specific disorders which can aid in Early Intervention Practices. This is achieved by accurately processing and analysing the acquired sensory data while transmitting the detection of a disorder to an appropriate career. The finding reveals that the proposed system can improve clinical decision supports while facilitating Early Intervention Practices. Our extensive simulation results indicate a superior performance of the proposed system: low latency (96% of the packets are received with less than 1 millisecond) and low packets-lost (only 2.2% of total packets are dropped). Thus, the system runs efficiently and is cost-effective in terms of data acquisition and manipulation.

INTRODUCTION

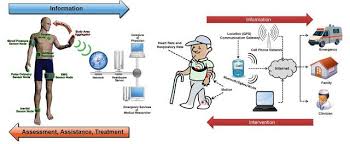
Healthcare providers can regularly notify individuals about their health through web applications unified into the cloud platform with the help of IOT devices connected to an elderly patient. With an ensured database, medical records of the patient are easily available to concerned healthcare specialists by decreasing numerous form fill ups at the consumer end. The health patterns can be analyzed and determined efficiently by using advanced analyzing tools through which time preserving solutions can be obtained. This enables low cost with excellent quality healthcare.

The proposal aims to develop a system to visualize multi-parameter analysis and reliable data transmission. The corporal parameters such as real time body temperature, pulse rate and blood pressure value are sent to the doctor through the internet. Based on these parameters of the patient, the doctor through the webpage can retrieve the health status from anywhere around the world. A Smart Health Monitor works in between this retrieval process to notify the patient in case it is an emergency. Also the relevant or suitable treatment can be carried out to the patient according to the health condition obtained from the device. This work demonstrates the health monitoring based on Internet-of-Things.

The growing elderly population, accompanied by the increasing prevalence of chronic diseases associated with ageing, will have profound implications for the health care system for decades to come. Therefore, we are proposing a system which enables continuous monitoring for elderly people health in real-time to prevent chronic diseases, thus preventing hospitalisation that burden the healthcare systems and costs. This paper presents a framework which utilises a smart-phone app and Wearable Sensors for Smart Healthcare Monitoring System (SW-SHMS) for elderly people. The system accumulates patient’s physiological data via wearable sensors (i.e., pulse, oxygen etc.) of elderly people in real-time. The data is transmitted to a data repository, where it will be stored and checked for any abnormality. Thus, any detection of disorder in a patient’s vitals will be reported to patient’s doctors and/or hospital in real-time to act on quickly and prevent a number of problems, such as, a sudden heart attack. Technologies are capable of providing patients physiological data from their locations to physicians anywhere in real-time, therefore, enabling remote remediation .For example, data such as blood oxygen saturation, heart-rate, and blood pressure can be measured via wearable devices and transmitted from patients locations to their doctors in real-time. This enables doctors and patients to communicate remotely. Providing such a system that can effectively monitor an elderly person physiological activity at regular intervals could detect diseases and other complications earlier. Especially, in the case of elderly people who are much likely to have a disorder in their physiological data. It is an utmost necessity to develop new methods and technologies in order to improve health services for the elderly community at an affordable price with ease of use while ensuring maximum comfort and independence.

PROBLEM STATEMENT

Health and fitness is one of the most rising application areas of IOT based technology. Today Wireless sensor-based systems gather patient medical data for analytical purposes and provide care to people for whom care wasn’t previously available. Remote health monitoring and management, wellness, home care, persistent diseases and care for the elder people are some of the important areas on which various researches are being done. In today's social insurance framework for old aged patients who stay in homes during post operational days checking is done either via overseer/ medical caretaker/doctor. Accordingly, providing a decent quality of life for aged people has become a serious social challenge at this moment.



SYSTEM DEVELOPMENT:

* **Wearable devices (Patients Layer):** A wearable device and smart-phone sensors will be connected to patient’s body to collect physiological data. Such sensors are measuring vital-sign, such as, blood oxygen saturation, skin temperature and heart-rate, variety of healthcare sensors are available today. Monitoring these symptoms in patient’s body is very important, as any abnormal data could potentially end up with a disease .For instance, drop of oxygen level in the human body causes a sleep apnea, which leads to death. In addition, abnormal blood pressure causes a kidney disease or diabetes. The sensory data are transmitted to patient’s smart-phone app via Bluetooth device and ultimately to a cloud database. Moreover, sensors will operate to measure and send the data regularly without patient attention to make everything automated (i.e., IOT), thus, this enhances user’s quality of experience and makes them more comfortable.
* **Cloud (Data Layer):** The Cloud refers to the place where the system data is stored and processed. Cloud receives patient’s data from their smart-phone over internet to be sorted and then it become available for doctor’s inspections. In addition, all data analysis and processing will be held in the cloud for any disorder detection in patient’s data, thus, the abnormal changes in patient’s data will be categories based on patient status and diseases. All resulted data/info will be reported either to patients and/or doctor’s platform or emergency unit or both depend on patient status. Thus, Cloud enables collaboration and knowledge sharing through its infrastructure which allows medical professionals to host information, analytics and diagnostics about patients so that other professionals around the similar interests can immediately access the data. This reveals in faster prescriptions and real-time updates to patient’s data.
* **Monitoring platform (Hospital Layer):** This layer is a doctor’s platform to monitor patient’s records and sensory data. The doctors will be able to inspect reports provided by the system from the cloud and then able to take actions. Data synchronisation in this platform in real-time by pulling all data from the cloud as soon as it become ready to use to keep doctors up to date with patient’s status, also to help paramedic to take early action in case of emergency before the situation getting worse and prevent hospitalisation.

Hardware

* Node MCU
* Pulse sensor
* Temperature sensor

NODE MCU

## Introduction

NodeMCU is an open source [IOT](https://en.wikipedia.org/wiki/Internet_of_Things) platform. It includes [firmware](https://en.wikipedia.org/wiki/Firmware) which runs on the [ESP8266](https://en.wikipedia.org/wiki/ESP8266) [Wi-Fi](https://en.wikipedia.org/wiki/Wi-Fi) [SoC](https://en.wikipedia.org/wiki/System_on_a_chip) from [Espressif Systems](https://en.wikipedia.org/w/index.php?title=Espressif_Systems&action=edit&redlink=1), and hardware which is based on the ESP-12 module. The term "NodeMCU" by default refers to the firmware rather than the development kits. The firmware uses the [Lau](https://en.wikipedia.org/wiki/Lua_(programming_language)) scripting language. It is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson and SPIFFS



## Working

As [Arduino.cc](https://en.wikipedia.org/wiki/Arduino) began developing new MCU boards based on non-[AVR](https://en.wikipedia.org/wiki/AVR_microcontrollers) processors like the ARM/SAM MCU and used in the Arduino Due, they needed to modify the [Arduino IDE](https://en.wikipedia.org/wiki/Arduino_IDE) so that it would be relatively easy to change the IDE to support alternate tool chains to allow Arduino C/C++ to be compiled for these new processors. They did this with the introduction of the Board Manager and the SAM Core. A "core" is the collection of software components required by the Board Manager and the Arduino IDE to compile an Arduino C/C++ source file for the target MCU's machine language. Some ESP8266 enthusiasts developed an Arduino core for the ESP8266 Wi-Fi SoC, popularly called the "ESP8266 Core for the Arduino IDE".[[16]](https://en.wikipedia.org/wiki/NodeMCU#cite_note-16) This has become a leading software development platform for the various ESP8266-based modules and development boards, including Node MCUs.

NodeMCU was created shortly after the [ESP8266](https://en.wikipedia.org/wiki/ESP8266) came out. On December 30, 2013, [Espressif Systems](https://en.wikipedia.org/w/index.php?title=Espressif_Systems&action=edit&redlink=1) began production of the ESP8266. The ESP8266 is a Wi-Fi SoC integrated with a [Tensilica](https://en.wikipedia.org/wiki/Tensilica) Xtensa LX106 core, widely used in IOT applications .NodeMCU started on 13 Oct 2014, when Hong committed the first file of nodemcu-firmware to GitHub. Two months later, the project expanded to include an open-hardware platform when developer Huang R committed the [gerber](https://en.wikipedia.org/wiki/Gerber_format) file of an ESP8266 board, named devkit v0.9. Later that month, Tuan PM ported [MQTT](https://en.wikipedia.org/wiki/MQTT) client library from [Contiki](https://en.wikipedia.org/wiki/Contiki) to the ESP8266 SoC platform, and committed to NodeMCU project, then NodeMCU was able to support the MQTT IOT protocol, using Lua to access the MQTT broker. Another important update was made on 30 Jan 2015, when Devsaurus ported the u8glib to NodeMCU project, enabling NodeMCU to easily drive LCD, Screen, OLED, even VGA displays.

In summer 2015 the creators abandoned the firmware project and a group of independent contributors took over. By summer 2016 the NodeMCU included more than 40 different modules. Due to resource constraints users need to select the modules relevant for their project and build a firmware tailored to their needs

## Pulse sensor

**Pulse Sensor is a plug-and-play heart-rate sensor for Arduino.**

It can be used by students, artists, makers, and developers who want live heart-rate data into their projects.

**The Pulse Sensor Kit includes:**

**1)**  Soft braided-wire ribbon cable.  Not kidding, that's one nice cable!

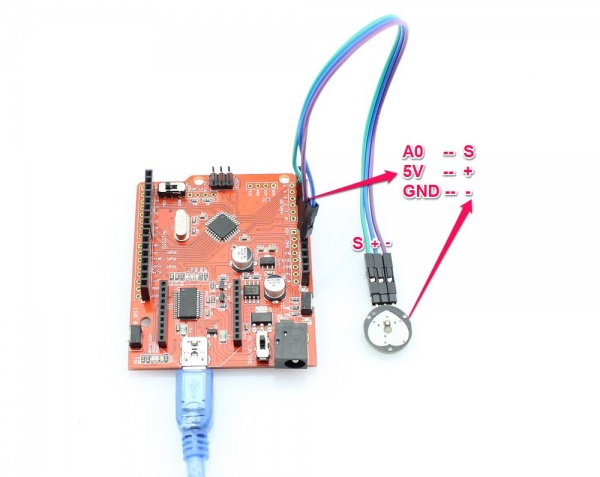
**2)**An Ear Clip that's perfectly sized to the sensor.

**3)**  A Velcro Finger Strap, just like the pro's.

**4)**Transparent Vinyl Dots, which make electrical insulation simple.

**5)**  The Pulse Sensor also has 3 sewing-holes, for you "Wearable's People"

CONNECTIONS:



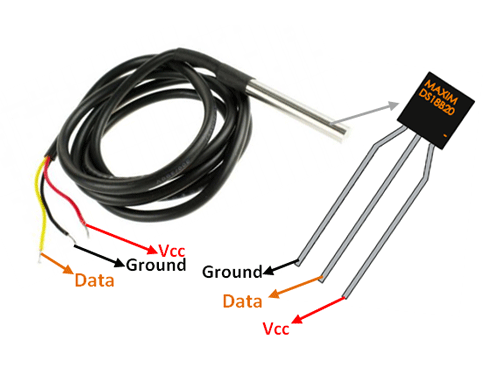
### Temperature sensor

The **LM35** is a 1-wire programmable Temperature sensor from maxim integrated. It is widely used to measure temperature in hard environments like in chemical solutions, mines or soil etc. The constriction of the sensor is rugged and also can be purchased with a waterproof option making the mounting process easy. It can measure a wide range of temperature from **-55°C to +125°** with a decent accuracy of **±5°C**. Each sensor has a unique address and requires only one pin of the MCU to transfer data so it a very good choice for measuring temperature at multiple points without compromising much of your digital pins on the microcontroller.

A temperature sensor is a device, usually an RTD (resistance temperature detector) or a thermocouple, that collects the data about temperature from a particular source and converts the data into understandable form for a device or an observer. Temperature sensors are used in many applications like HV and AC system environmental controls, food processing units, medical devices, chemical handling and automotive under the hood monitoring and controlling systems, etc.

The most common type of temperature sensor is a thermometer, which is used to measure temperature of solids, liquids and gases. It is also a common type of temperature sensor mostly used for non-scientific purposes because it is not so accurate.

CONNECTIONS:



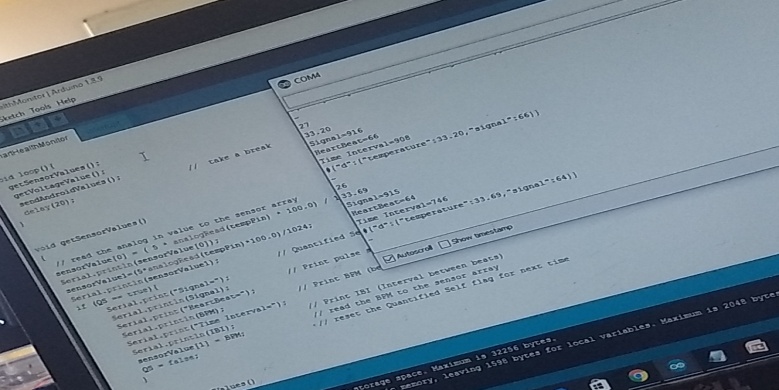
### SOFTWARE

* MIT APP
* Arduino IDE
* IBM Watson Cloud platform

## ARDUINO IDE

The Arduino integrated development environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java. It is used to write and upload programs to Arduino compatible boards, but also, with the help of 3rd party cores, other vendor development boards.

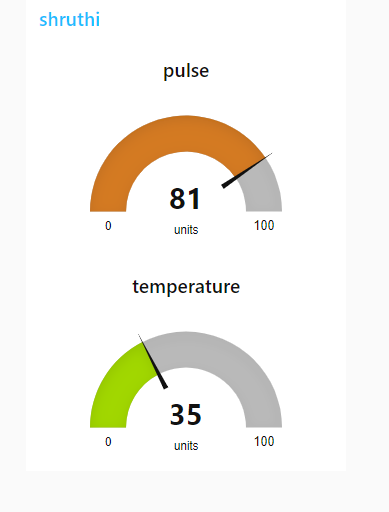
The source code for the IDE is released under the GNU General Public License, version 2. The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub main() into an executable cyclic executive program with the GNU tool chain, also included with the IDE distribution. The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware



## IBM Cloud Platform

BM cloud computing is a set of cloud computing services for business offered by the information technology company IBM. IBM cloud includes infrastructure as a service (IaaS), software as a service (SaaS) and platform as a service (PaaS) offered through public, private and hybrid cloud delivery models, in addition to the components that make up those clouds.

IBM offers three hardware platforms for cloud computing. These platforms offer built-in support for virtualization. For virtualization IBM offers IBM Websphere application infrastructure solutions that support programming models and open standards for virtualization.



The management layer of the IBM cloud framework includes IBM Tivoli middleware. Management tools provide capabilities to regulate images with automated provisioning and de-provisioning, monitor operations and meter usage while tracking costs and allocating billing. The last layer of the framework provides integrated workload tools. Workloads for cloud computing are services or instances of code that can be executed to meet specific business needs. IBM offers tools for cloud based collaboration, development and test, application development, analytics, business-to-business integration, and security.

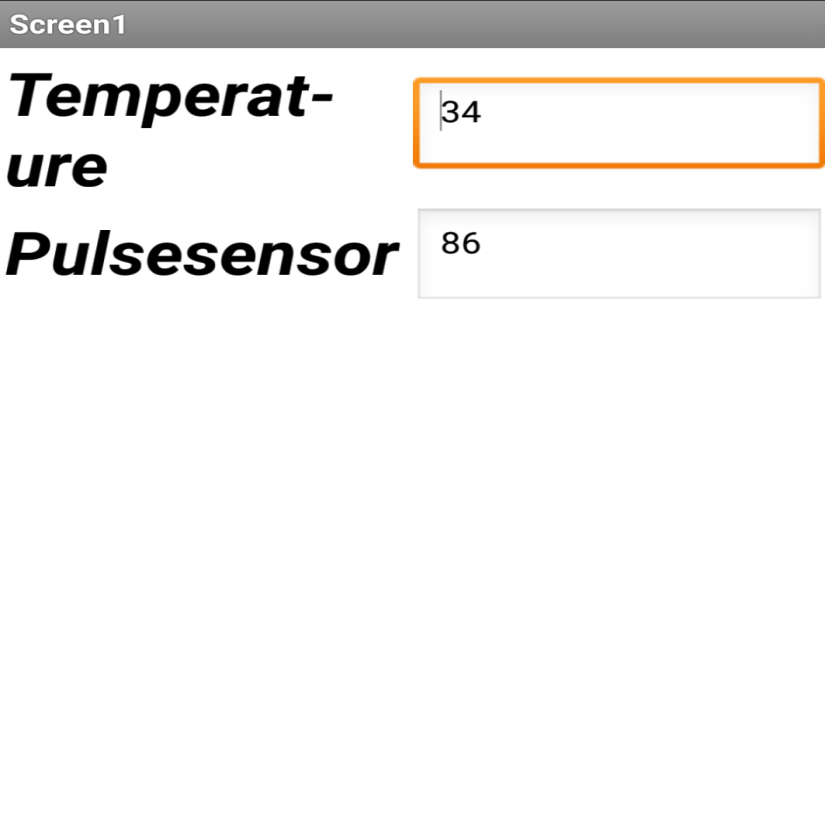
### MIT APP INVENTOR

App Inventor for Android is an open-source web application originally provided by Google, and now maintained by the Massachusetts Institute of Technology (MIT), which allows newcomers to computer programming to create software applications for the Android operating system (OS).

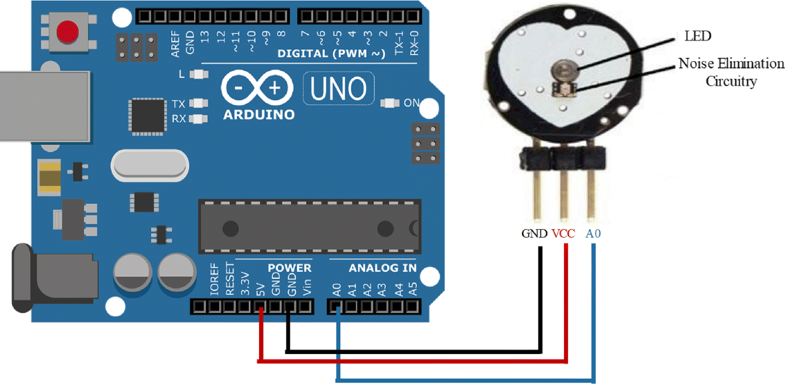
It uses a graphical interface very similar to Scratch and the StarLogo TNG user interface, which allows users to drag-and-drop visual objects to create an application that can run on Android devices. In creating App Inventor, Google drew upon significant prior research in educational computing, as well as work done within Google on online development environments.

App Inventor and the projects on which it is based are informed by constructionist learning theories, which emphasizes that programming can be a vehicle for engaging powerful ideas through active learning. As such, it is part of an ongoing movement in computers and education that began with the work of Seymour Papert and the MIT Logo Group in the 1960s and has also manifested itself with Mitchell Resnick's work on Lego Mindstorms and StarLogo.

App Inventor also supports the use of cloud data via an experimental FirebaseDB component.

IMPLEMENTATION:

The development of the proposed system starts with a database that is able to handle system services and storing patient data. The process of storing and retrieving data within the system has two-part processes. First process involves data encryption/decryption according to the appropriate standard of patient’s data security. Second process involves the method of data representation changes (i.e., for storing data). For example, analogue data outputs from sensors are converted to some readable row-data format to be stored in the data repositories. Thus, stored data can be retrieved by another process at a later stage and the same method of data representation will be applied to provide a proper data visualisation. Moreover, the development of system platforms starts with implantation of patent’s platform of the SW-SHMS. Patient’s platform is consists of a mobile application that able to communicate with the sensors to accumulate patient’s vital sign. Android studio version 2.5 has been used to implement the mobile application.



A pulse sensor has been used to measure patient’s heart-rate and transfer the data through Bluetooth device to the mobile app. The pulse sensor used to measure heart-rate through an optical heart-rate chip. The output data will be analogy data format. Therefore, Arduino UNO has been used to receive the pulse sensor readings. The reading of the sensor sends to patient mobile app through the HC-06 Bluetooth device which is connected to the Arduino UNO. Figure [5](https://link.springer.com/article/10.1007/s11042-018-7134-7#Fig5) shows the connection between the pulse sensor and the Arduino UNO. The pulse sensor has three points of connections as shown in Fig. [5](https://link.springer.com/article/10.1007/s11042-018-7134-7#Fig5). The first one, is the red coloured wire which is the power wire, this wire accept voltage value between + 3V to + 5V. The black wire represents the ground connection (GND). The blue wire is responsible for transferring the signal from the sensor to the Arduino UNO. The open-source Arduino Software has been used to program the UNO in order to receive and send sensor readings.

The following step involves sending the sensory data to the SW-SHMS mobile app. After sensors’ readings have been collected from the Pulse sensor in the Arduino UNO, it will be sent to the android app through a Bluetooth device. Therefore, the UNO has been connected with a Bluetooth device, which is the HC-06 version. The physical connection between the Bluetooth HC-06 device and the Arduino UNO board is shown in Fig. [6](https://link.springer.com/article/10.1007/s11042-018-7134-7#Fig6). Arduino Bluetooth version HC-06 has four connection points, these are VCC, GND, TXD and RXD. The four connection points need to be connected to the UNO board in order to complete the operation of sending Pulse readings to the app through the HC-06. Moreover, the Bluetooth device connected to the Arduino board through the following wires: 1) Connecting Bluetooth VCC pin to 5V pin on the Arduino board. 2) Connecting Bluetooth GND pin to the GND pin on the Arduino board. 3) Connecting Bluetooth TXD pin to RXD pin on the Arduino board. 4) Connecting Bluetooth RXD pin to TXD pin on the Arduino board.

CODE FOR ARDUINO:

#include <SoftwareSerial.h>

SoftwareSerial mySerial(9, 8); // VARIABLES

int pulsePin = A1; // Pulse Sensor purple wire connected to analog pin 0

int tempPin = A0; // Temperature Sensor connected to analog pin 1

int blinkPin = 13; // pin to blink led at each beat

float sensorValue1;

int sensorValue[2]={0,0};

int voltageValue[2] = {0, 0};

char inbyte = 0; // these variables are volatile because they are used during the interrupt service routine!

volatile int BPM; // used to hold the pulse rate

volatile int Signal; // holds the incoming raw data

volatile int IBI = 600; // holds the time between beats, must be seeded!

volatile boolean Pulse = false; // true when pulse wave is high, false when it's low

volatile boolean QS = false; // becomes true when Arduino finds a beat.

Void setup() {

pinMode(blinkPin, OUTPUT); // pin that will blink to your heartbeat!

Serial.begin(9600); // we agree to talk fast!

mySerial.begin(9600); //

interruptSetup(); // sets up to read Pulse Sensor signal every 2mS

// UN-COMMENT THE NEXT LINE IF YOU ARE POWERING The Pulse Sensor AT LOW VOLTAGE,

// AND APPLY THAT VOLTAGE TO THE A-REF PIN

//analogReference(EXTERNAL);

}

void loop(){

getSensorValues();

getVoltageValue();

sendAndroidValues();

delay(20); // take a break

}

void getSensorValues()

{

// read the analog in value to the sensor array

sensorValue[0] = (5 \* analogRead(tempPin) \* 100.0) / 1024.0;

Serial.println(sensorValue[0]);

sensorValue1=(5\*analogRead(tempPin)\*100.0)/1024;

Serial.println(sensorValue1);

if (QS == true){ // Quantified Self flag is true when arduino finds a heartbeat

Serial.print("Signal=");

Serial.println(Signal); // Print pulse sensor raw data

Serial.print("HeartBeat=");

Serial.println(BPM); // Print BPM (beats per Minute)

Serial.print("Time Interval=");

Serial.println(IBI); // Print IBI (Interval between beats)

SensorValue[1] = BPM; // read the BPM to the sensor array

QS = false; // reset the Quantified Self flag for next time

}

}

Void sendAndroidValues()

{

//puts # before the values so our app knows what to do with the data

mySerial.print('#');

Serial.print('#');

//for loop cycles through 2 sensors and sends values via serial

String payload ="{\"d\":{\"temperature\":";

payload +=sensorValue1;

payload+=",""\"signal\":";

payload+=BPM;

payload+="}}";

Serial.println (payload);

mySerial.println (payload);

mySerial.println ('~'); //used as an end of transmission character - used in app for string length

Serial.println('~');

delay(2000); //added a delay to eliminate missed transmissions

}

Void getVoltageValue()

{

if (BPM>=60 && BPM<=100){

for (int x = 0; x < 2; x++)

{

voltageValue[x] = sensorValue[x];

}

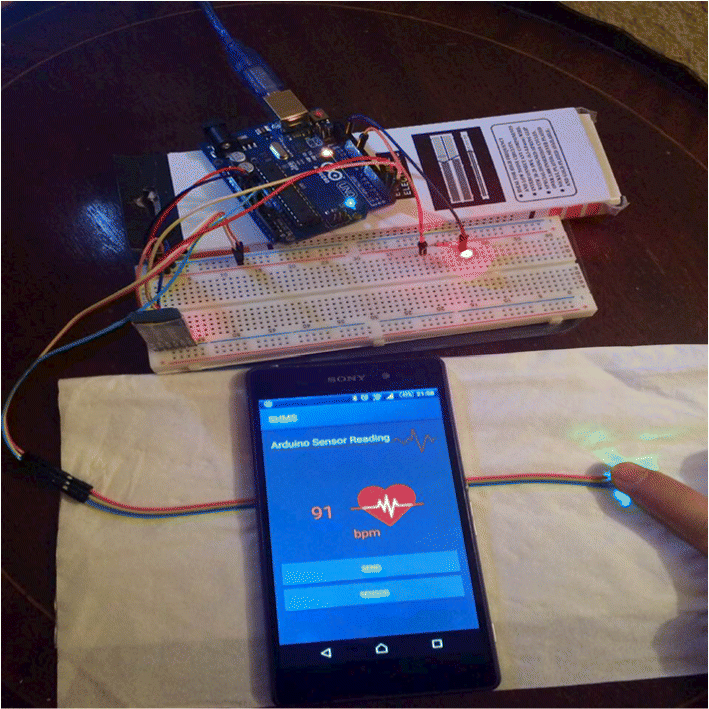
}

else {

voltageValue[1] = 0;

}

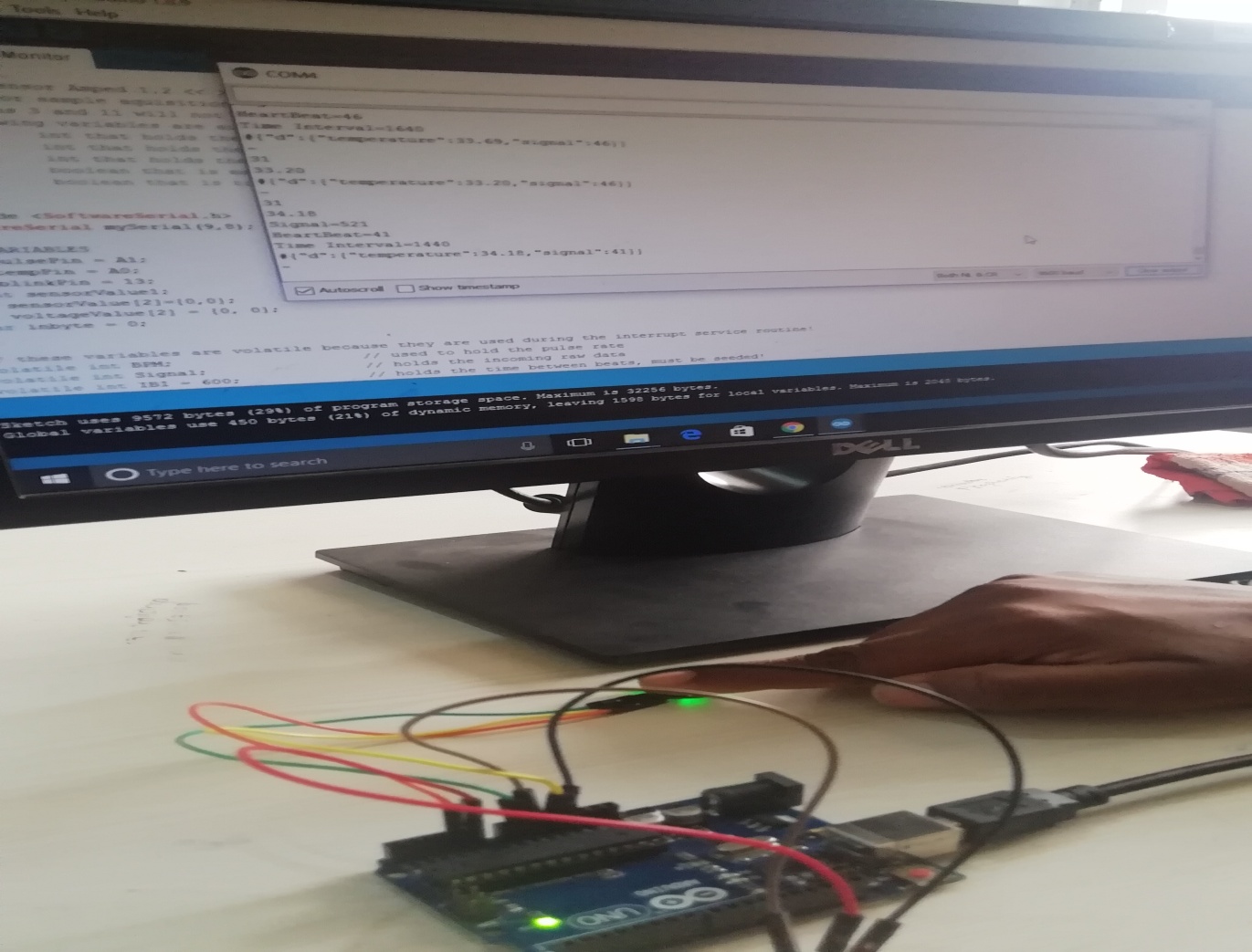
}



CONCLUSION:

The smart healthcare monitoring system (i.e., SW-SHMS) has been proposed in this paper to handle the challenges of providing home based healthcare monitoring and avoiding hospitalisation. The literature shows that there is a great demand of producing an effective healthcare solution that monitor elderly people in their home and in real-time. SW-SHMS can highly contribute to provide comfortable and safe environment for elder and disable people, thus, enable them to live independently without the fear of any emergency or critical healthcare situation through continuous monitoring of their health by SW-SHMS. Briefly, SW-SHMS accumulate patient’s physiological data via wearable sensors and transmit it to Cloud for data analysing and processing. Thus, any detection of disorder in patient’s data will be reported to patient’s doctors via hospital platform. SW-SHMS has a fixable architecture that can scale and expand easily, thus, providing a reliable and cost effective system to monitor patients remotely. In addition, the results shows that the system could efficiently contribute to improve healthcare services by using the prosed impeccable SW-SHMS system which able to monitor patients symptoms remotely and in real-time.

Future developments and enhancements for the SW-SHMS are wide, for instance, the system can be extended to apply artificial intelligence and/or machine learning concepts to help predicting life threatening diseases early. Moreover, the system holds a fair amount of medical data which be invested to develop a recommendations system that able to give a recommendations about the health and diets for better life style. In addition, the network topology can expand to have Fog computing node which will help processing the data at the edge of the network instead of travelling for long distance to Cloud, also, Fog computing cloud reduce data traffic on the cloud and help to make better healthcare decisions due to its location within the local area network. Thus, Fog computing will be employed to reduce the packets loses by upgrading the network infrastructure to allow distributing several Fog nodes at the edge of the network. This will help in empowering the system in data acquisition and manipulation via Fog-Cloud collaboration model that act faster on the data packets before it gets lost.





TEAM:RANGERS

MEMBERS:

1. BUYYANI SHRUTHI
2. KOWSHIK KRISHNA
3. MARY PRASHANTHI