SMART PATIENT MONITORING OVER A CUSTOM M2M FRAMEWORK

By

NC Mohsin Sadiq

NC Umer Majeed

NC Saim Jehan

Submitted to the Faculty of Electrical Engineering, Military College of Signals,

National University of Sciences and Technology, Islamabad in partial fulfillment

For the requirements of a B.E. Degree in Electrical (Telecom) Engineering

JUNE 2015

ABSTRACT

Machine to Machine (M2M) technologies allow key information to be exchanged without human intervention, making it possible to reduce costs and improve efficiency and service to customers. M2M is widely deployed in health-care domain. The project is aimed at development and integration of hardware and software components of remote patient monitoring system application. Project is divided into four modules i.e. sensing and transmission of vital parameters, an android application for user (patient or athlete); storage, analysis and sharing of data at server, a website to readily access EMRs. Using this system the doctor can continuously monitor a patient's vital signs in real time. Thus providing developing countries like Pakistan, with a better state of the art health care system.

CERTIFICATE OF CORRECTNESS AND APPROVAL

It is certified that the work contained in this thesis title "Smart Patient Monitoring over a Custom M2M Framework", was carried out by Umer Majeed, Mohsin Sadiq and Saim Jehan under the supervision of Lt. Col. Dr. Adnan Ahmed Khan for partial fulfillment of Degree of Bachelor of Telecommunication Engineering, is correct and approved.

Approved by

_____

(Dr. Adnan Ahmed Khan)

Project Directing Staff (DS)

Military College of Signals (MCS)

Dated: _____ June 2015

DECLARATION

No portion of the work presented in this dissertation has been submitted in

Support of another award or qualification either at this institution or elsewhere

In the name of Allah, the Most Merciful, the Most Beneficent

Dedicated To our parents, without their unflinching support and cooperation,

The work of this magnitude would not have been possible.

ACKNOWLEDGEMENTS

There is no success without the will of ALLAH Almighty. We are grateful to ALLAH, who has given us guidance, strength and enabled us to accomplish this task. Whatever we have achieved, we owe it to Him, in totality. We are also grateful to our parents and family and well-wishers for their admirable support and their critical reviews. We would thank our supervisor Lt. Col. Dr. Adnan Ahmed Khan for his thorough guidance and continuous support throughout the year. We would specially like to thank Mr. Usman Siddiqi, Engr. Hassan Sadiq, Engr. Sadia Majeed, Mr. Hasaan and Mr. Mughees Sarwar for their commendable guidance through each and every phase of our project. We would also like to thank our seniors Engr. Moaaz Khan, Engr. Ramsha Ambreen, Engr. Hamza Malik and Engr. Waleed Akram for always helping us in our difficult times when we were critically stuck. The team would like to pay gratitude to the team leader for bearing us the whole year and for billion repeated document edits and lastly NC Nasir Husain for always being there for helping us whenever needed.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

EMRs	Electronic Medical Records
M2M	Machine to Machine
RPM	Remote Patient Monitoring
PHP	Hypertext Preprocessor
CSS	Cascading Style Sheets
HTML	Hyper Text Markup Language
MCU	Microcontroller Unit
SPM	Smart Patient Monitoring
WSN	Wireless Sensor Network

CHAPTER 1 INTRODUCTION

1.1 PROJECT OVERVIEW

Machine to machine solutions subjected to industrial instrumentations comprises of a device (most likely to be a sensor/meter) to abduct an event (such pressure and speed), that is relayed through a network (wireless, wired or hybrid) to an application (software program) that translates the captured event into meaningful information (for example, items need to be restocked). Such communication was originally accomplished by having a remote network of machines which relay information back to a central hub for analysis, which would then be rerouted into a system like a personal computer.

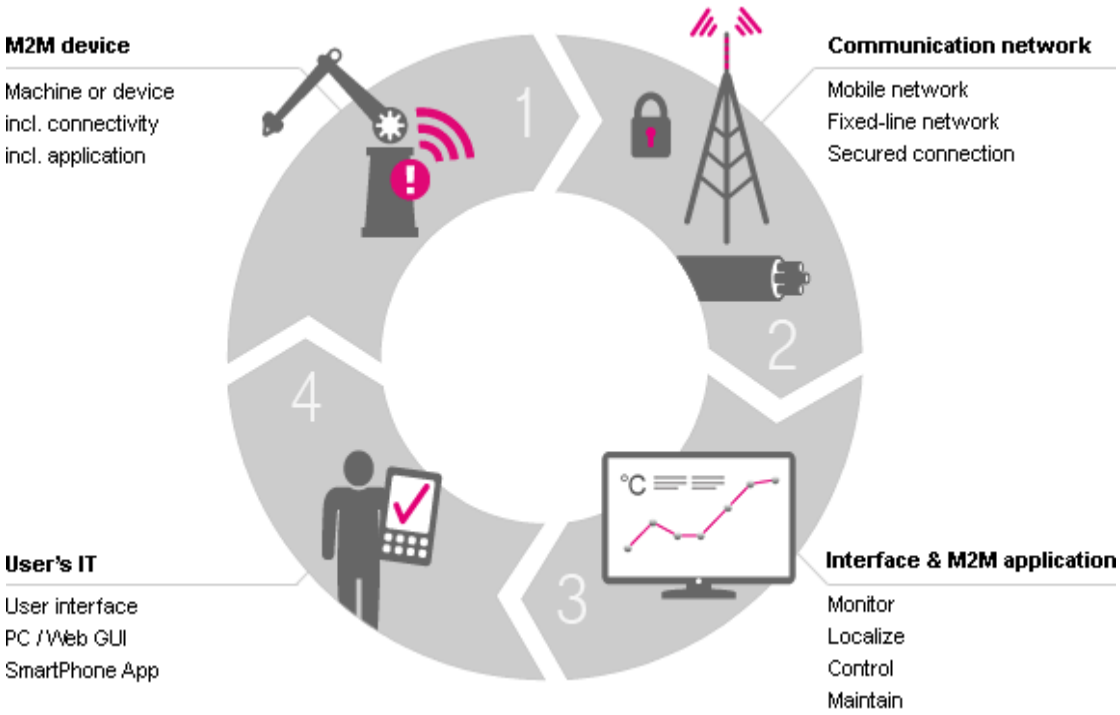

FIGURE 1- MACINE TO MACHINE FRAMEWORK

Smart Patient Monitoring (SPM) is: "a technology that enables remote patient monitoring (e.g. in the home), by stepping outside the boundaries of the previously used

conventional techniques incorporating a typical clinical setup. Thereby increasing care access and reducing healthcare costs."

SPM is an incredible way of using the patient's very own Smartphone for the purpose of transmitting of his/her own necessary body parameters to a distant located doctor. The doctor in turn can continuously monitor the patient's vitals at his end, without being physically present with the patient at all times.

1.2 BACKGROUND

Certain demographic changes have lead to an increase in the amount of people suffering from lifestyle induced chronic diseases like Chronic Obstructive Pulmonary Disease and diabetes, which cause direct costs as well as a high amount of indirect costs to the citizens. A developing countries like Pakistan face a remarkable cost eruption in the health care sector. A Paramount cause of these problems is the individuals' lifestyle paired with Low personal fitness and low activity levels. These challenges can be overcome by reinforcement of primary and secondary prevention and the empowerment of the individuals to take responsibility for their own well-being.

This can be done by increasing the persons' self-awareness, via remotely monitoring their health and fitness vitals and integrating them to a social network for immediate assistance. Additionally a lack of nursing workforce has been predicted. We are seen as a chance for individuals to keep track of their health, thereby taking more responsibility of their nutritional lifestyle and improving care efficiency by providing high quality data to formal and informal careers and health professionals. In order to implement these e-Care solutions at the highest levels, standardization is of need for easy and quick integration of personal health devices (i.e. pulse rate monitor) with

multiple gateway devices (i.e. smart-phones) and electronic medical record (EMR) systems.

1.3 PROBLEM STATEMENT

It is challenging era for Healthcare sector to provide to the masses a platform which ensures on time and quality service to health conscious people and patients. Development of such a system is the need of time which ensures timely diagnosis, digital storage of medical records and an interactive User interface for report generation for the patients as well as caretakers.

1.4 PROJECT DESCRIPTION AND SALIENT FEATURES

The M2M market is massive, with its rapid growth being steered by technology advancements, low-cost communications services and declining hardware costs across a diverse array of vertical industries. M2M technologies allows the development of patient tracking systems that keeps a constant check on the patient's location. Monitoring can then automatically alert a doctor or healthcare facility in case of an anomaly.

The project aims to develop a system which can be comfortably attached to the patient's body for the sensing of their general fitness vitals i.e. the heartbeat, body temperature and the calories burnt. After performing the required real time processing, the readings are wirelessly sent and stored to a distant located webserver as EMRs. A warning SMS to the doctor is also sent, if the system detects an abnormality.

FIGURE 2- REMOTE PATIENT MONITORING

The sensors used for detection offers the typical features of being the ideal wearable sensor i.e. being non-invasive, light in weight, low-powered, unobtrusive, motion tolerant and trivial to place.

1.4.1 HARDWARE

Complete modules of the remote unit will be made that will send the respective sensor's data to the patient's Smartphone, which in turn will be passed over a secure network to our remote server, visible only to the concerned clients and their doctors.

1.4.2 SOFTWARE

A Software setup for data management will be developed alongside separate client and doctor interfaces. This allows the user to gain access to the states of the currently operating units and settings under observations.

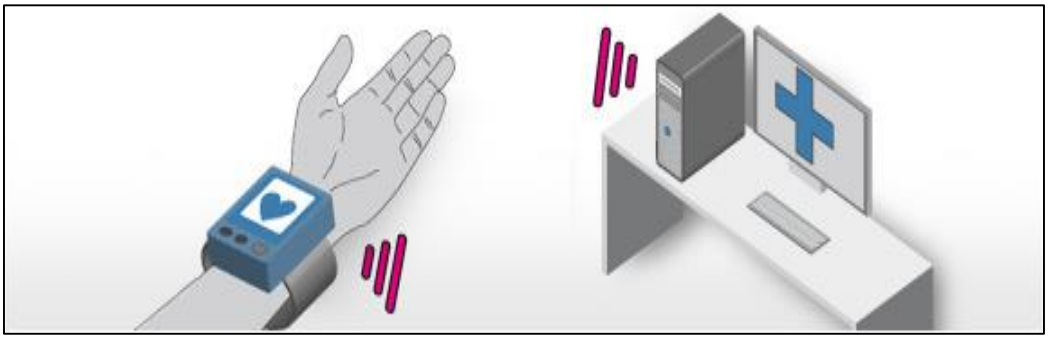

FIGURE 3 - SYSTEM MODEL DIAGRAM

1.4.3 APPROACH

The following methodology scheme will be used to implement the proposed system specifications and to target the problems mentioned in Problem Statement:

FIGURE 4 - APPROACH FOR IMPLEMENTATION OF PROJECT

The development of different modules of project will be carried out in parallel, however their interfacing will be done in above given manner.

1.5 DESCRIPTION OF APPLICATIONS

M2M and WSN are the kind of technologies that allows the integration of several other advanced technologies, which includes embedded systems, sensors, wireless communication technologies and modern networks.

Being healthy is not just an option people have. It demands taking intensive care of one's life and health. People lose these valuables due to lack of resources, improper monitoring and non-availability of sufficient medical practitioners. The RPM comes to the rescue in the remote areas where availability of experienced doctors is deficit.

The following are some potential medical applications

- Continuous(real time) patients monitoring
- Collecting and storing clinical data in the designed databases for future referencing.
- E-Health facilities to rural areas

A detailed insight of the above mentioned applications is as follows:

1.5.1 HOSPITALS:

At present, patient monitoring at hospitals is being carried out using the traditional way of handling a single patient at a time by a member of the nursing staff. This can be a hectic task for the conducting staff, especially if the number of patients exceed a certain limit. The proposed solution our project presents will certainly loosen things up to a great extent.

Separate sensor kits can be designed at industry level, and can be allocated to each beds at a hospital, which in turn fetches the real time body vitals of the patients, and transmit them to a centralized server placed in the ward. This reduces both the monitoring time

human intervention cost. Hence SPM can be applicable to both military and civil hospitals with the extended benefits of being an autonomous system.

1.5.2 RURAL AREAS:

The residents of rural areas lose their precious lives, mainly due to the non-availability of proper healthcare resources at proper time frames. Transportation and availability of a qualified doctor are two other undealt major issues. RPM aids the doctor by giving him/her the privilege to monitors his/her patients, disseminated in multiple villages, from their workstation located thousands of miles away. Thus proved to be fruitful to both.

1.5.3 ATHLETES

Athlete's trainings for an upcoming marathon needs to have their vital signs monitored rapidly for optimum performance. It can be a tedious task, if performed solely by a caregiver. Our proposed Solution this issues in such a way that athletes can be automatically and continuously monitored by performing the required test routines, without the need of arranging regular appointments with the doctor, hence preventing any sort of training hindrances.

1.5.4 PATIENT WELLNESS ANALYSIS:

All the sensed data is sent to a centralized server, where it is stocked and stored. A doctor can look up to a patient's profile at any time, prescribe him/her a diet plan, exercises, workouts and other necessary precautions needed based on his vital activity.

This specific feature of our project will be highly beneficial for health conscious peoples. They will blend themselves in accordance with what doctor prescribe them, and as a result, the overall disease count is significantly reduced.

1.6 SCOPE, OBJECTIVES AND SPECIFICATIONS

1.6.1 SCOPE

The project is based on technologies which are about to bring revolution in the healthcare department all over the world. The project is not only confined to remote patient monitoring but involves developing a custom M2M framework that can be used for various applications falling in the M2M domain.

1.6.2 OBJECTIVES

"Development of a prototype for a remote patient monitoring system that uses a portable device, comfortably worn by the patients, for sensing their general fitness parameters, i.e. pulse rate, body temperature and the calories burnt, and allowing the readings sensed to be transmitted wirelessly to a distant located doctor using the patient's smartphone. Storage of the vitals as Electronic medical records (EMRs) is to be carried, in a secure and readily accessible way."

1.6.2.1 LEARNING OBJECTIVE

To present the project as requirement for completion of B.E. Degree in Electrical Engineering

To apply the Concepts Studied in Digital Communication, Embedded Systems, and Micro-Controllers, Computer Networking, Programming, Mobile Communication etc.

1.6.3 SPECIFICATIONS

The planned system will be a combination of multiple sensors used for sensing temperature and the pulse rate in real time. At a predefined time interval, regular measurements will be taken and will be passed on to the processing module

(ARDUINO UNO) that will upon successful completion of its task , will pass it wirelessly to the patient's Smartphone via a connected Bluetooth module.

Development of an Android application will be carried out at the patient's Smartphone end, which will display the received data from the sensor module; and will allow him see his current readings, medical history, contact information of connected users.

The Phone will then automatically pass the data to a distant located server, via 3G/4G/Wi-Fi. The server will be storing the data in a designed database, assigning a unique ID for every new patient, and will be notifying the doctor simultaneously if he is active on a web-portal created to sign-up, see medical records of oneself or a suspect and much more.

Special warning will be issued immediately, directly to the doctor's Smartphone, in case the sensed data doesn't fall in the predefined threshold limits depending upon nature of suspect. The doctor or caregiver can advise medicines or take rapid action depending upon the nature of circumstance.

CHAPTER 2 LITERATURE REVIEW

Today's health care systems are over-burdened by the increasing number of elderly and disabled people needing medical aid. If the physical activities of these people can be monitored reliably and at a reduced cost, it will have a tremendous effect on reducing the gap between the need and the capability of the current health care systems.

Patient monitoring systems are gaining their importance as the fast-growing global elderly population increases demands for caretaking. These systems use wireless technologies to transmit vital signs for medical evaluation.

2.1 HOME BASED HEALTH MONITORING SYSTEM

SUSHAMA PAWAR [1] has stated that in intensive care unit, the heart rate and temperature of patients are continuously monitored. Patient's data (temperature, heart rate.) can be frequently measured and sent to server. Period of sending can be varied based upon condition of patient. By checking the trend of values of these parameters one can predict that patient is normal or have symptoms of illness. She proposed a low cost design using Bluetooth as mode of communication from sensors to android phone. The Bluetooth Devices communicates with each other on a secure connection through a short-range radio frequency. She prefers LM35 to measure the temperature because its voltage is linearly proportional to the degree centigrade temperature and operate over 0°C~ 150°C. She also explained working of pulse sensor and the two peaks in measured signal. Systolic peak measures the pressure that is contraction of various arteries. Diastolic peak measures the pressure that is exerted on the wall of the various arteries.

For smart living, interactive applications are increasingly important especially on interaction of people and the environment. Wireless technologies are bringing about

dramatic improvements in the quality of patient care by allowing unprecedented mobility while providing medical staff with easy and real-time access to patient data.

2.2 ANDROID BASED PATIENT MONITORING SYSTEM

Deep Modi [2] mentioned that Android mobile phone receives the data collected by hardware device and also can transmit it to remote server in time. This cost effective method simplifies and speeds up the process of information acquisition, processing and analysis. There are important considerations such as range, throughput, security, ease of implementation and cost for choosing a wireless standard. The patient monitoring involves handling of sensitive data. These data should be transmitted securely without any intrusion. She had made an Android Application to show the values of parameters of patient to doctor and database using MySQL. She stated that MySQL is an open source relational database management system. It also widely used by web application developers, together with PHP and APACHE The rapid growth of wireless technologies and personal area networks has enabled the continuous healthcare monitoring of mobile patients using compact sensors that collect and evaluate body parameters and movements.

2.3 WIRELESS BAN FOR HEALTHCARE MONITORING

Sasan Haghani [3] also worked on a related project. He explained his project that the sensors constitute a body area network (BAN) where patients' vital signs are collected and reported wirelessly to a base station. The captured data was inserted into a MySQL database where a webpage with a graphing application programming interface (API) was used to display the data. WBAN can monitor single or multiple vital signs at the same time and can include implantable or wearable bio-sensors. The data transmission in WBANs can include ZigBee, Bluetooth, and Wi-Fi. The user interface which is a

display website was designed using PHP and HTML codes. The software intended to be easy for medical personnel to use and provides enough details on patient pulse rate, temperature and location on a continuous basis. The sensor device on the patient transmits raw data to the receiver which in turn sends the data wirelessly to the MySQL database using a Wi-Fi shield. When the database gets new data from the device, it refreshes the page and displays the new data in the format that the user can understand. The data can be also saved on the server so it can be used for future references. He further stated that pulse sensor amped used was wearable on the ear or on a finger with a current consumption of 4mA at 5V.

Telemedicine is a rapidly developing application of clinic medicine where medical information is transferred through the phone or internet or other networks for the purpose of consulting and performing remote medical procedures or examinations.

2.4 PATIENT MONITORING SYSTEM USING ANDROID

Prema Sundaram [4] also explains architecture of his project. He stated that the pulse rate can be used to check overall heart health and fitness level. In medicine, one's pulse represents the tactile arterial palpation of the heartbeat by trained fingertips. Pulse (or the count of arterial pulse per minute) is equivalent to measuring the heart rate. Body temperature is a measure of the body's ability to generate and get rid of heat. The body is very good at keeping its temperature within a narrow, safe range in spite of large variations in temperatures outside the body. Normal human body temperature depends upon the place in the body at which the measurement is made, and the time of day and level of activity of the person.

He designed his android application in Eclipse. He stated that Eclipse is an open source community, whose projects are focused on building an open development platform

comprised of extensible frameworks, tools and runtimes for building, deploying and managing software across the lifecycle. The Eclipse SDK consists of the Eclipse Platform, Java development tools and the Plug-in Development Environment.

CHAPTER 3 DESIGN AND DEVELOPMENT

Based upon the background study, Cost effectiveness, feasibility and availability of the components, we have proposed a design as explained below and working on it.

A smart fitness assistant is an intelligent system that provide assistance for preventive healthcare, diagnosis and care. Smart fitness assistant is basically a system assimilating an android application with a wearable device incorporating electronic sensors that can monitor body functions and respond to environmental changes such as warning athletes of rising body temperature and pulse. The Smart fitness assistant is a combination of some cost-efficient electronics and an innovative biometric system making it easy for the patients/users for diagnosing, analyzing, keeping the record of their pulse and temperature as well as sharing their data with their doctors through social media or web.

3.1 DETAIL DESIGN

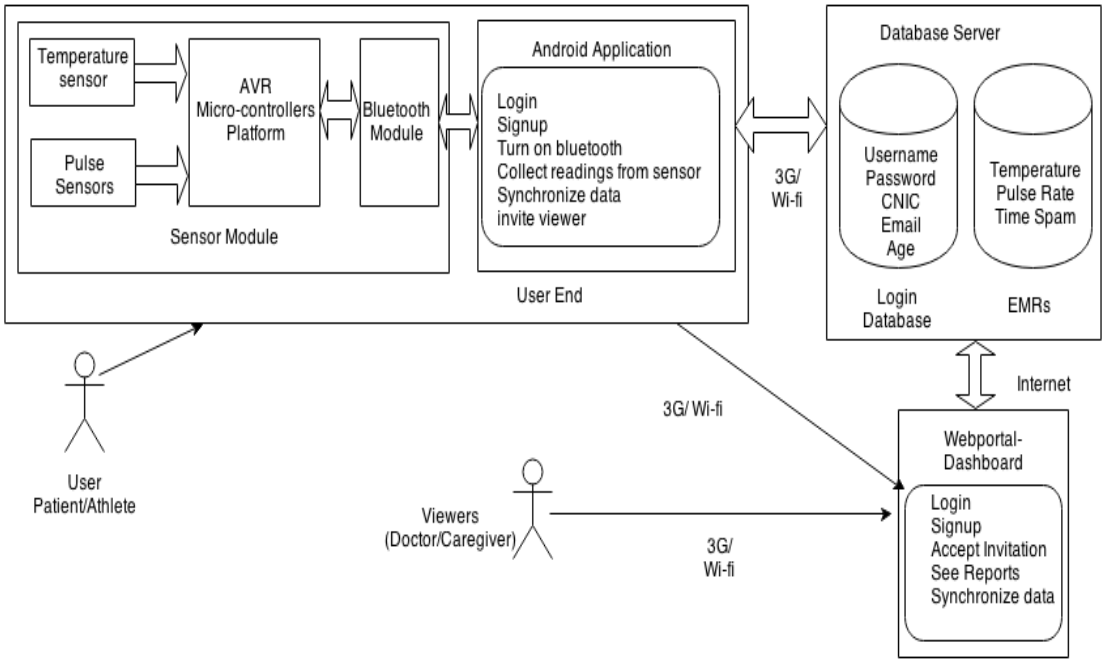

FIGURE 5 - BLOCK DIAGRAM / DETAIL DESIGN

3.2 TECHNICAL SPECIFICATIONS

Project is divided into four modules i.e.

1. Sensing and transmission of vital parameters
2. An application for user (doctor and patient) interface
3. Development of Database for storing and analysis of data at server.
4. Dashboard (Web portal)

3.2.1 SENSOR MODULE

Sensor Module consists of Sensors (Temperature and pulse rate) interfaced with a MCU and a Bluetooth shield. It is required for fetching, parsing and manipulating the readings taken from sensors and transmitting them to the Android application of the user. The user should be able to take readings from sensors on demand bases or automatically after a regular interval provided that the sensor module is powered on.

Coding the MCU was carried out in the ARDUINO IDE environment using the standard C language.

3.2.2 ANDROID APPLICATION

Android application is present to provide the patient with a user interface. It is responsible for displaying the data transmitted by sensor module. It should also able to transfer the data via Wi-Fi/ internet to database for permanent storage. The application should also able to fetch the previously stored data from the database to assist the different tasks such as report generation etc.

The Android Application Development was carried out in Android Studio using the standard languages of Java & XML.

3.2.3 DATABASE

It stores the incoming data from the Android Application as well as from the web portal. It is also responsible for providing access to stored data to the Android Application and Web portal. The Database Designing was carried out in the WAMP server using (MySQL).

3.2.4 DASHBOARD- WEBPORTAL

It provides user with a web interface over the internet to access the readings in a pleasing organized manner. Furthermore, the web portal will also allow the authorized viewers (doctors etc.) to view the data of users.

The web portal designing was carried out in Notepad++ and the Dreamweaver environments using the standard languages of PHP, HTML, CSS, and JQuery.

3.3 DESIGN SPECIFICATIONS

3.3.1 HARDWARE MODULE

Following hardware modules has been used to design the kit

3.3.1.1 PULSE SENSOR AMPED

The Pulse Sensor Amped is a plug-and-play heart-rate sensor for the Arduino platform, making it very simple and easy to be used by students, athletes and mobile developers who want to easily integrate real time pulse rate signals into their tasks. It is essentially an innovative blend of a simple optical heart rate sensor with an amplifier and a noise cancellation setup, thereby making it extremely fast and easy to get an authentic pulse reading. The sensor operates at 5V, drawing just 4mA of current, hence making it extremely suitable for motile applications.

Wrap the Pulse Sensor to your fingertip and connect it to a 5 Volt MCU and read the pulse rate.

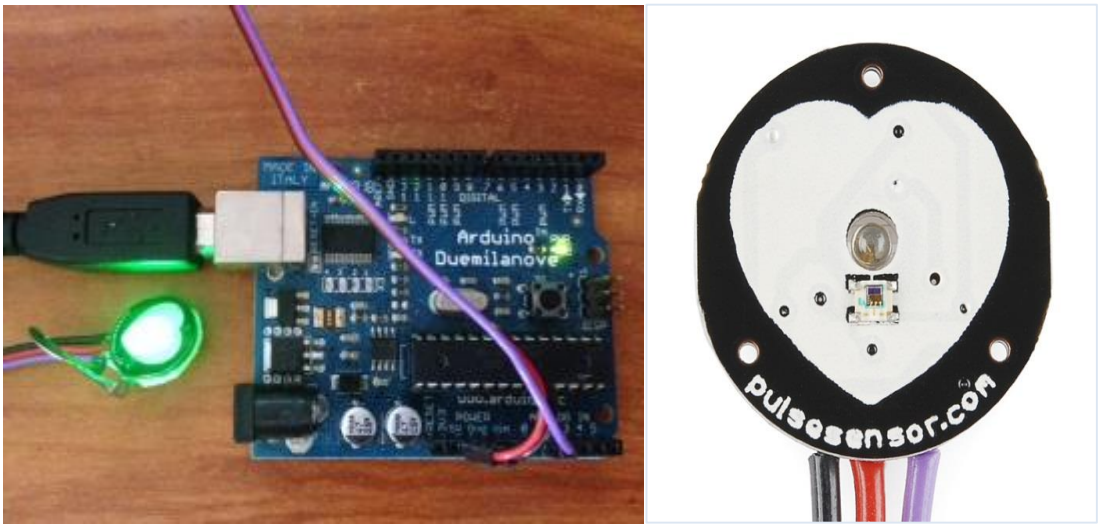

FIGURE 6 - PULSE SENSOR AMPED WITH ARDUINO

3.3.1.2 LM 35

LM35 is a precision IC temperature sensor with its output proportional to the temperature (in ºC). Temperature obtained from lm 35 is highly accurate as compared with a thermistor. The operating temperature range is between -50°C to 148°C.

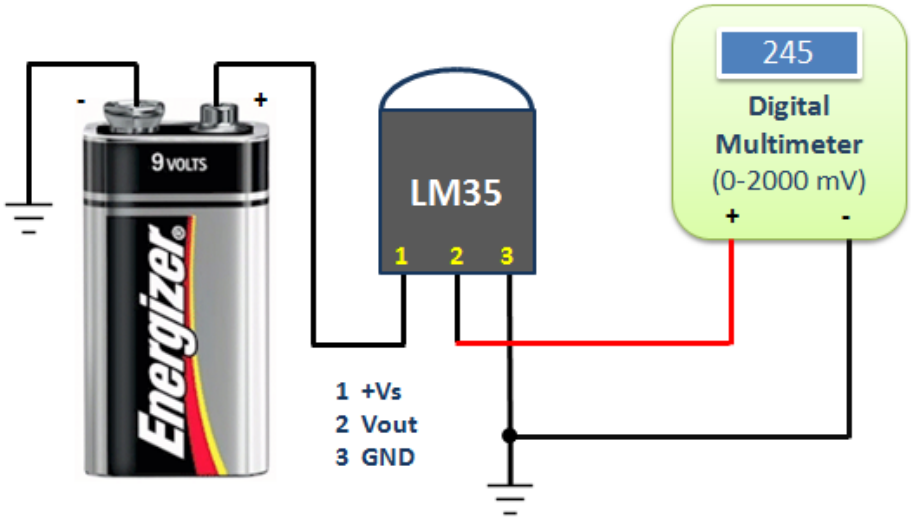

FIGURE 7 - LM 35 OUTPUTS SHOWN ON DMM WITH BATTERY CONNECTED

3.3.1.3 Bluetooth Module TTL-base HC-06

HC-06 is a wireless Bluetooth transceiver module that operates at a modulation depth of 2Mbps-3Mbps.

1. Supports Bluetooth V2.0 protocol standard
2. Operates at a voltage of 3.3V sipping 20-30mA of current
3. By default, the Baud Rate is set to be equal to 9600 bps, but this can be changed using the AT commands.
4. A built-in antenna operating at a frequency of 2.4GHz
5. 8Mbit flash storage

This device is used to create the Body area network and is helping to connect the outer sensors with the smart phone device.

FIGURE 8 - HC-06

3.3.1.4 ARDUINO UNO

The Arduino Uno is using the atmega328 MCU and comprises of 6 analog inputs, 14 digital I/O pins, a 16 MHz ceramic resonator, a USB port, a DC power jack and a reset button. Just interface it to a computer with a data cable or power it up directly with a battery to get started.

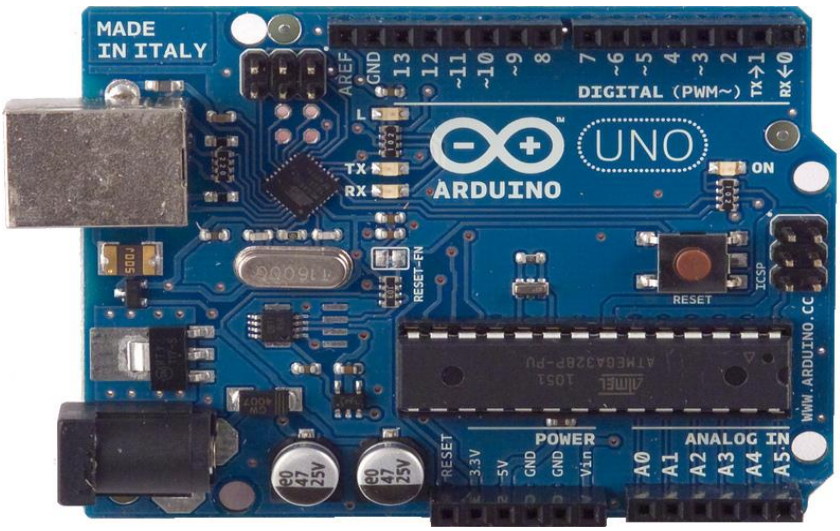

FIGURE 9 - ARDUINO UNO (ATMEGA 328)

3.3.2 ANDROID APPLICATION

The android Application will be compatible with latest version of Android as well as older version of Android (2.2).

The android application will provide the following Interfaces

1. To Sign up
2. To Login
3. To Turn on Bluetooth
4. To collect readings from sensors
5. To send invitation to the viewer (Doctor and care givers)
6. To See the readings collected
7. To synchronize data to the database.

3.3.3 DATABASE

There are two databases:

1. Database-one for the Login info of the user.
2. Database-two for storing and keeping record of all the readings of the sensors attached to the application as well as for storing the medical reports and miscellaneous history

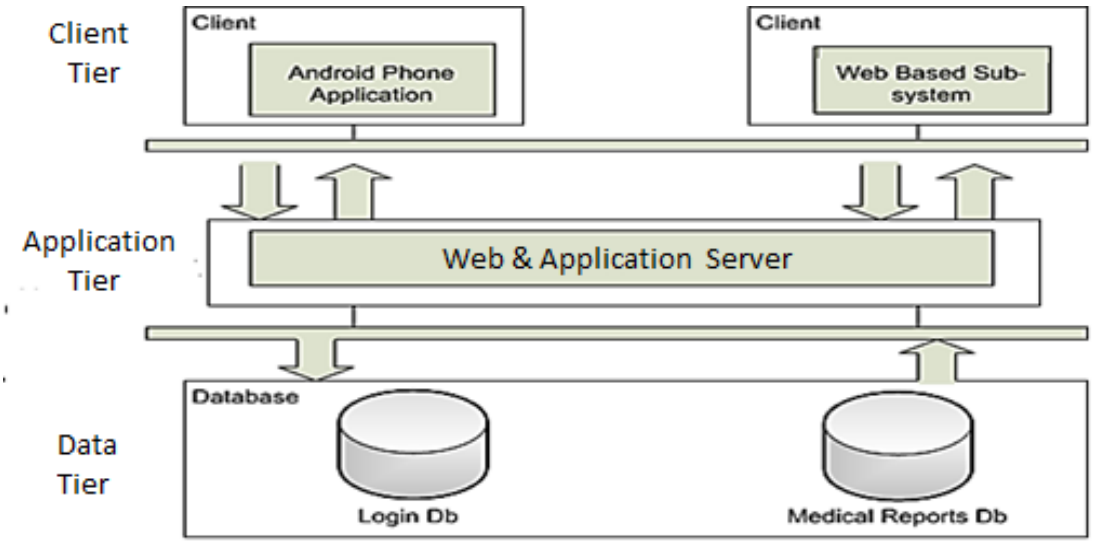

FIGURE 10 - ARCHITECTURE OF DATABASE AND ACCESSING IT

3.3.4 DASHBOARD-WEBPORTAL

The web portal will provide the user and viewers (Doctors) an interface to deal with the designed system through a web browser. The web portal will provide following Interfaces

1. To Sign up and to login
2. To socialize and network
3. To synchronize data to the database
4. To view the Reports

CHAPTER 4 PROJECT ANALYSIS AND EVALUATION

The Project is sub categorized into the following 4 sub modules.

i. Hardware Kit
ii. Android Application Development
iii. Database Designing for EMR Storage
iv. Dashboard/Webpage Designing

4.1 TESTING

To ensure quality of the product, testing is conducted. Accuracy and efficiency of tasks performed by our system had to be tested to analyze the system and verify and validate it. Both Hardware and Software testing techniques and results obtained are discussed in the upcoming sections

4.1.1 HARDWARE KIT

The Designed kit consists of Sensors (Temperature and pulse rate) interfaced with a Microcontroller (Arduino ATmege328) and a Bluetooth shield (HC-06). The hardware kit is responsible for fetching, parsing and manipulating the readings taken from the biometric sensors and transmitting them to the Android application of the user via Bluetooth communication. The user readings are taken from the sensors automatically after a regular interval provided that the sensor module is powered on.

For individual testing of the hardware kit, a dummy android application from GOOGLE Play store was used, to validate the successful flow of the Bluetooth communication from the hardware kit to the Smartphone.

4.1.1.1 OUTPUTS

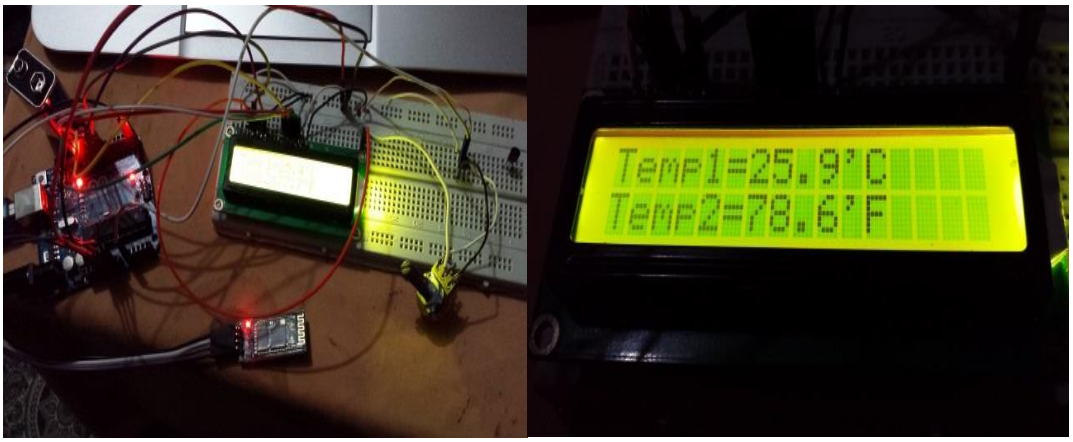

FIGURE 11 - RUNTIME SIMULATION OF THE INITIAL PROTOTYPE DESIGN

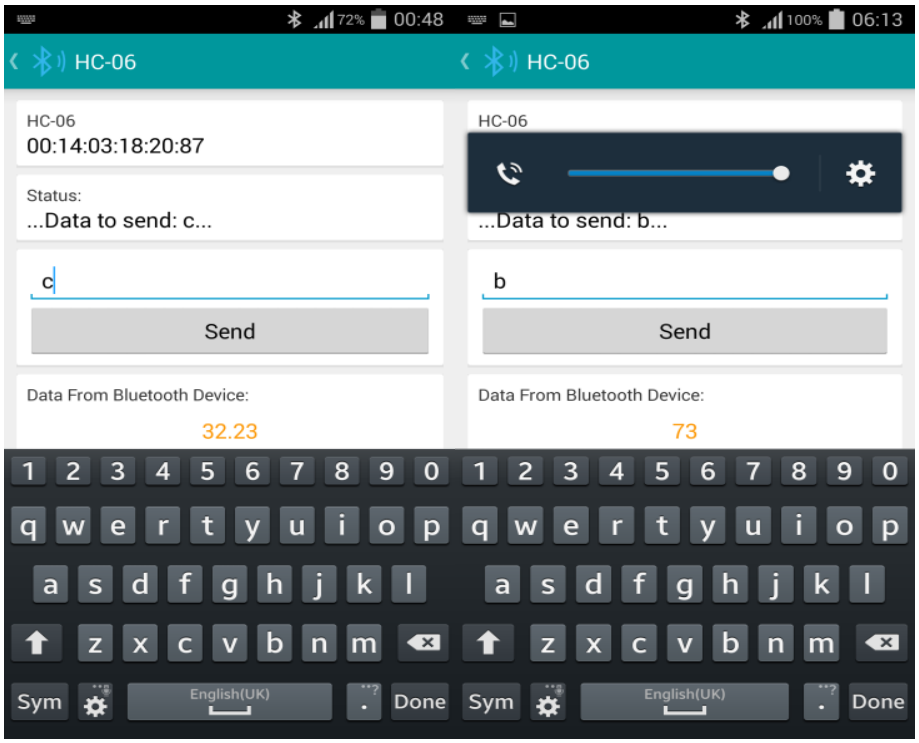

FIGURE 12 - OUTPUTS VERIFIED OF THE DUMMY ANDROID APPLICATION

4.1.1.2 PROTEUS SIMULATION AND PCB LAYOUT

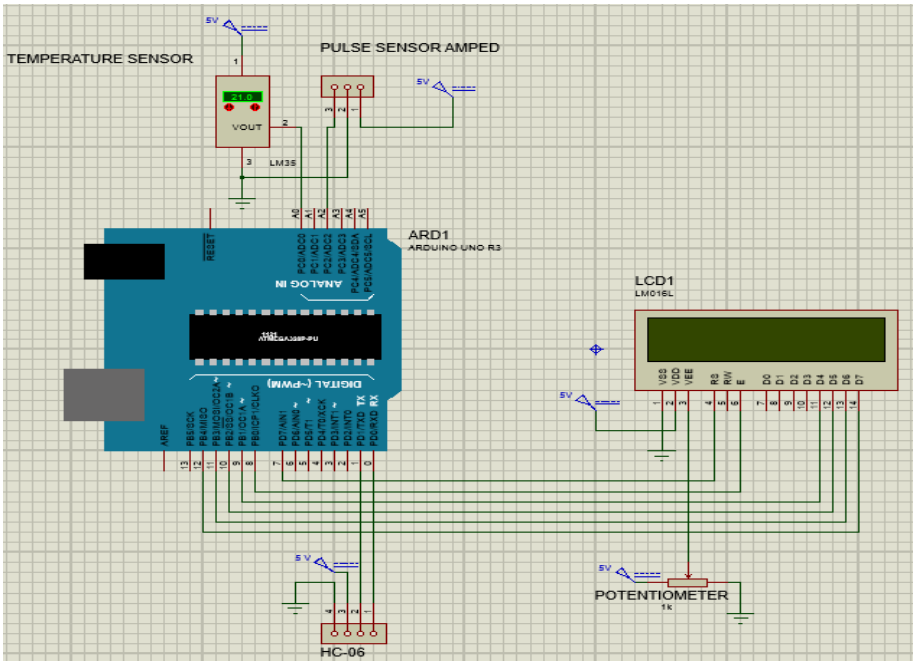

FIGURE 13 - ISIS SCHEMATIC OF THE HARDWARE KIT

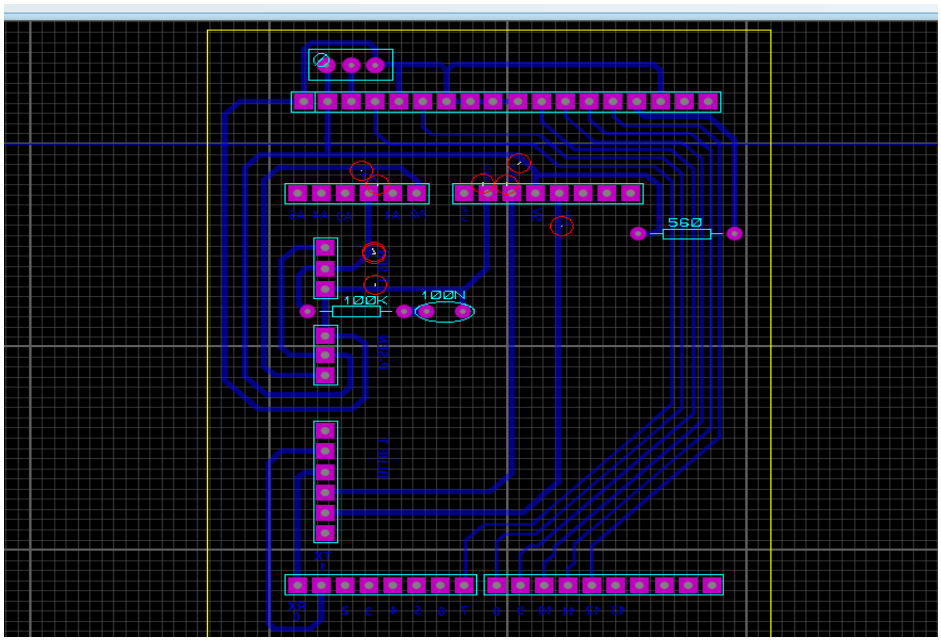

FIGURE 14 - PCB LAYOUT OF THE HARDWARE KIT

4.1.2 ANDROID APPLICATION DEVELOPMENT

An Android application has been designed to aid the communication flow of vital sensor parameters from the sensor kit to a distant located web server via Wi-Fi/3G/4G. Android Studio and Eclipse were used as a standard coding platform for developing the application, using an API level ranging between 8(Gingerbread) to 22 (Lollipop), thus providing massive compatibility to a great range of android OS supporting devices.

4.1.2.1 OUTPUTS

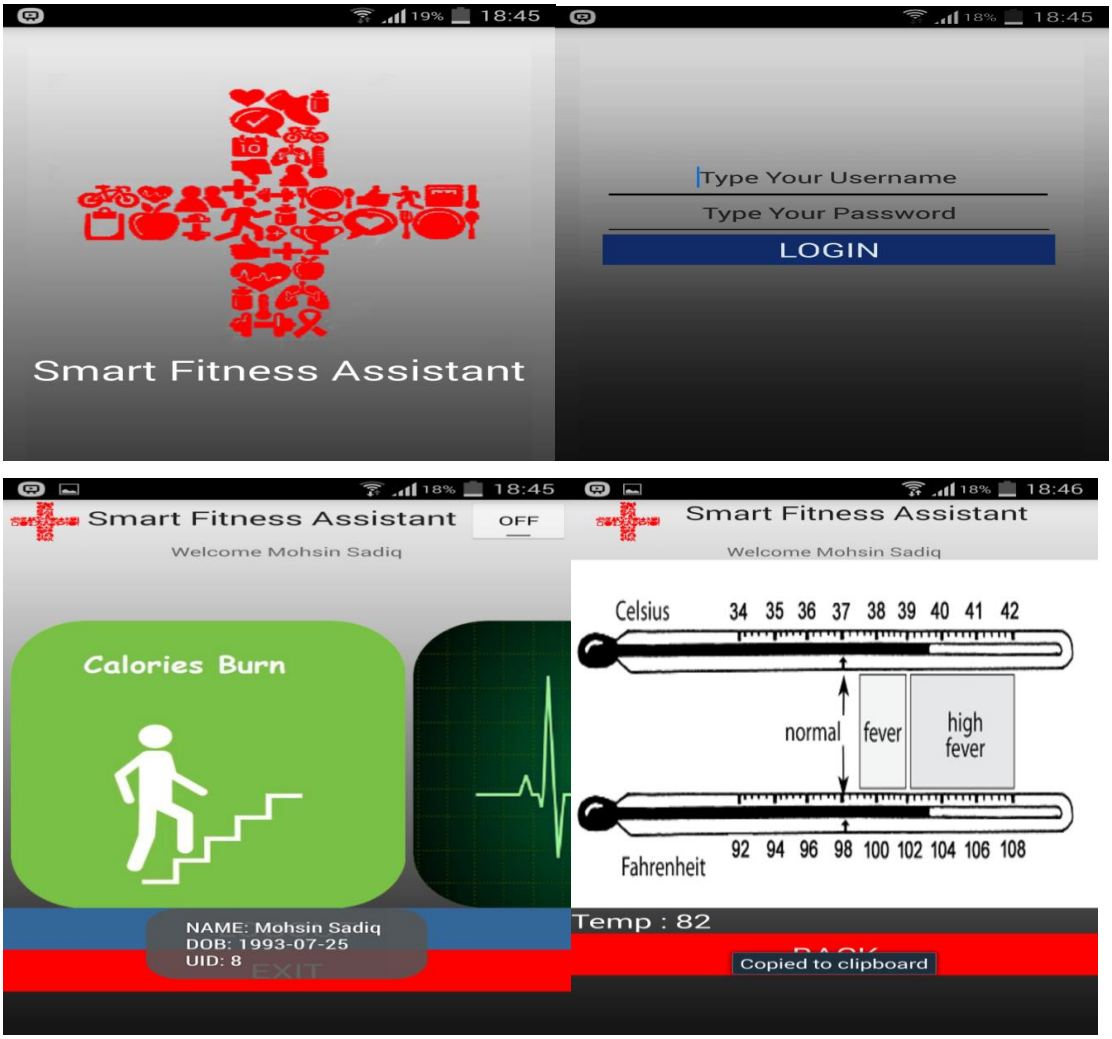

FIGURE 15 - SNAPSHOTS OF THE ANDROID APPLICATION

4.1.3 DATABASE

For storing the incoming data values from the Android Application through Wifi/3G/4G ,as well as from the web portal itself, a single database has been developed with 4 entry tables responsible for handling the following :

1. Signup and login Credentials
2. Electronic Medical Records(EMR) storage
3. Viewership for social connections
4. Feedback

The Database has been designed using a MySQL server using the standard SQL language.

4.1.1.1 OUTPUT

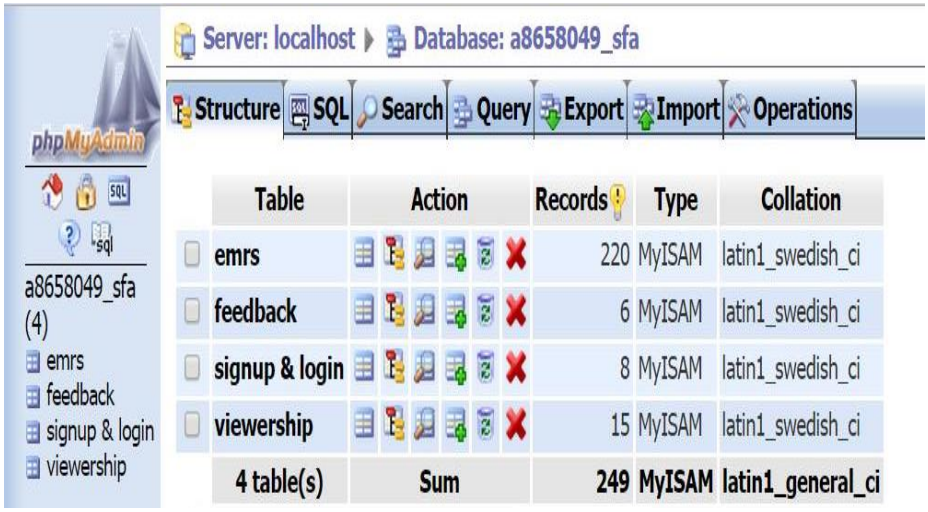

FIGURE 16 - INSIGHT TO THE DATABASE

4.1.3 DASHBOARD/WEBPAGE DESIGNING

Dashboard provides the user (Patient, Doctor, Caretaker etc.) with a user interface over the web, so that he/she can access the readings in a pleasing and organized manner. The web portal is designed using the standard languages of HTML5, CSS3, PHP5, JavaScript, JQuery and Ajax in Notepad++ and PHPSTORM.

4.1.1.2 OUTPUTS

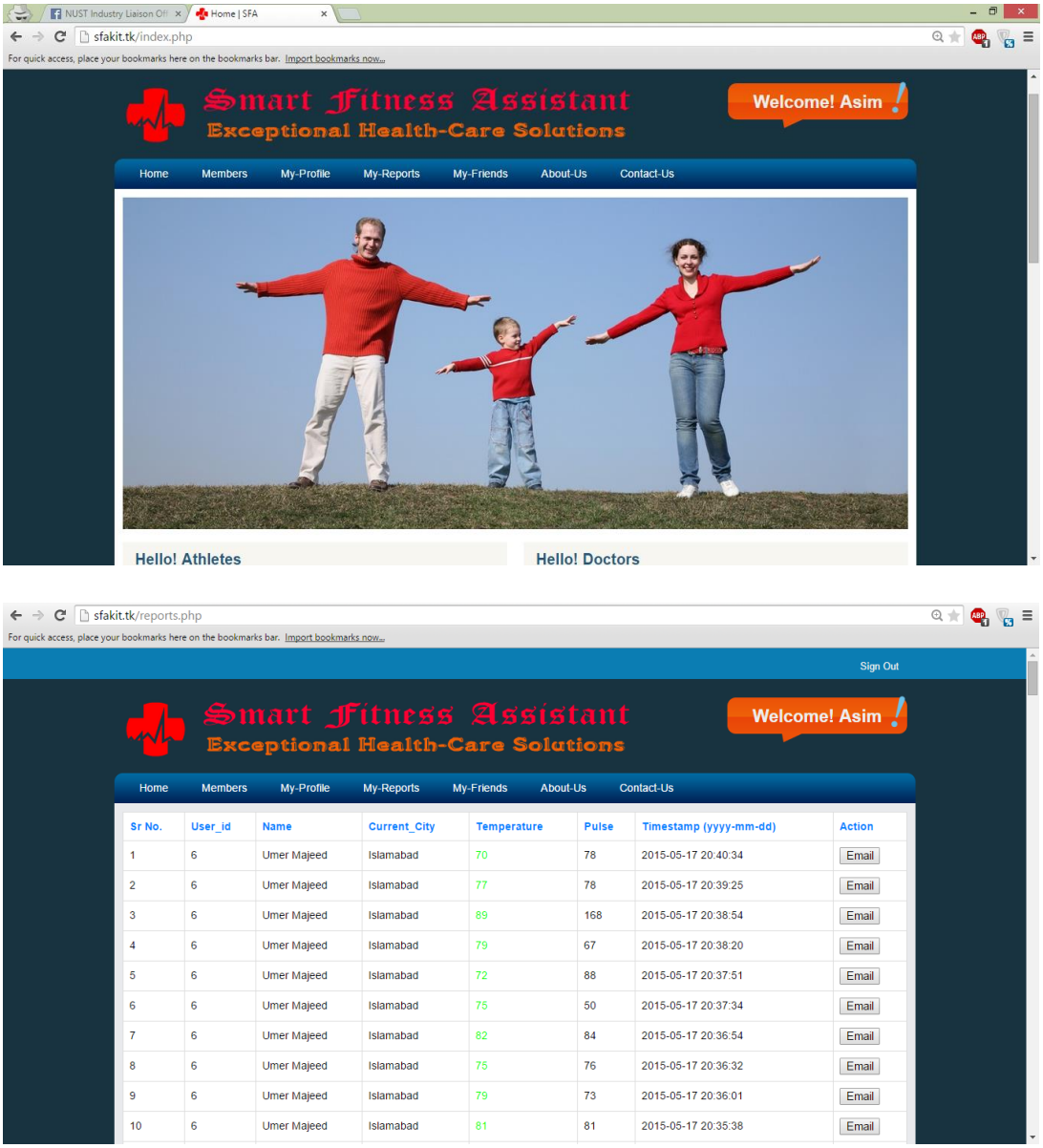

FIGURE 17 - A DOCTOR IS CHECKING EMRS OF CONNECTED USERS

4.2 COMBINED TESTING AND RESULTS VERIFICATION

The four individual modules were synchronized to work as a single unit, and their results were verified successfully.

FIGURE 18 - COMPLETE PROJECT SIMULATION AND VERIFICATION

CHAPTER 5 CONCLUSION

Healthcare industries around the world face the same challenge-decrease expenditure while caring for more and more patients.M2M can help. Telemedicine is able to decrease the cost of expensive doctor visits by enable remote communication between physicians and patients. Remote monitoring can reduce or eliminate hospital stays, freeing up scarce hospital beds for more serious cases, and tracking leads to increased security, and peace of mind.

Being healthy and in shape in not just a trend anymore. It's a lifestyle. M2M solutions can not only monitor vital signs during exercise, it can also make real time transmission of the data to servers or the internet. Users can then quickly view and gain valuable insights about their health and fitness programs, and even share their workouts on social networks. How? Through connected weight scales, heart rate monitors and other devices. Thus the end consumer has an improved fitness and quality of life.

5.1 OBJECTIVE ACHIEVED

Our Main Objective was to successfully develop of a prototype for a remote patient monitoring system that uses a portable device, worn comfortably by the patients/athletes, for sensing their general fitness vitals i.e. pulse rate, calories burnt and the body temperature, and allowing the readings sensed to be transmitted wirelessly to a distant located doctor using the patient's smartphone, as well as to store the Electronic medical records (EMRs) securely in a readily accessible way. The Product is fully functional and is successfully achieving all the proposed objectives correctly, accurately and cost effectively.

5.2 LIMITATIONS

At present, the system is limited only to measuring the general fitness vitals on the move, that aren't affected by physical body movements. Although the end product is way above the similar products that are already present in the market at a very high cost, but certain improvements are still of need, thus enhancing the scope of the project to the critical patients too.

5.3 FUTURE WORK

Machine to Machine technologies and Remote Patient Monitoring have a vast domain. The concept projected can be greatly improvised by the addition of the some more features. A few suggestions for the future are as follows:

1. Adding more sensors to measure i.e. Blood Pressure, ECG, Diabetes, Sugar Level and Air flow, thus, taking the concept of mobile e-health a step forward.
2. Addition of features in the Android Application i.e. WhatsApp and Signup interface, making the application more user-friendly, and highly interactive.
3. Designing the website on any CMS such as WordPress, enabling the admin to maintain the website with more ease.
4. Offering text chat and video calling feature in Website.
5. An interactive interface for the Admin, for better administration.
6. Addition of a Notification System in the Dashboard.

APPENDIX A

//HARDWARE CODE FOR ARDUINO UNO (ATMEGA 328)

```
#include <LiquidCrystal.h>.
LiquidCrystal lcd(7, 8, 9, 10, 11, 12);

// VARIABLES
int pulsePin = 0;                 // Pulse Sensor purple wire connected to analog pin0
int blinkPin = 13;                // pin to blink led at each beat
int fadePin = 5;                  // pin to do fancy classy fading blink at each beat
int fadeRate = 0;
int beat=0;
float tempC;              // Variable to store temp. in Celsius
int tempF;                // Variable to store temp. in Fahrenheit
int tempPin = 2;          // The temp. Sensor is connected to analog pin (1)0
int TP;
char BluetoothData='0';
volatile int BPM;
volatile int BP;                      // used to hold the pulse rate
volatile int Signal;                  // holds the incoming raw data
volatile int IBI = 600;
volatile boolean Pulse = false;
volatile boolean QS = false;



void setup(){
```

```
  pinMode(blinkPin,OUTPUT);
 pinMode(fadePin,OUTPUT);                  // pin that will fade to your
heartbeat!


  Serial.begin(9600);       !


// set up the LCD's number of columns and rows:
lcd.begin(16, 2);                          // LCD size 16*2
lcd.print("PULSE:");                       // Print BMP on LCD
lcd.setCursor(0, 1);                       // The second row to print
lcd.print("TEMP :");                       // Print Temp. in Fahrenheit
on LCD



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


Void loop ()
{


  tempC = analogRead(tempPin);              // read the value from the
sensor
  tempC = (5.0 * tempC * 100.0)/1024.0;     // convert the analog data to
temperature
  tempF = ((tempC*9)/5) + 32;               // convert celcius to
farenheit



// print result to lcd display
lcd.setCursor(6, 1);
```

```
//Serial.print(tempC,1);
lcd.print(tempC,1);

//lcd.setCursor(6, 0);
//Serial.print(tempC,1);
//lcd.print(tempF,1);


 delay(300);

  if(Serial.available()>0)
{
 BluetoothData=Serial.read()  ;
   if(BluetoothData=='c')
    {
        Serial.println(tempC);
    }
   if(BluetoothData=='f')
    {
    Serial.print(tempF);
    Serial.print(" ");
    Serial.println(BPM);

   }

    if(BluetoothData=='b')
    {
    //Serial.println("TEMP 2");

    Serial.println(BPM);
```

```
   }
}
delay(1000);


sendDataToProcessing('S', Signal);     data
if (QS == true)
  {
      fadeRate = 255;
      sendDataToProcessing('B',BPM);        // send heart rate with a 'B'
prefix
      sendDataToProcessing('Q',IBI);        // send time between beats
with a 'Q' prefix
       QS = false;
     }


  ledFadeToBeat();
 delay(20);


}



void ledFadeToBeat()
{
    fadeRate -= 15;
    fadeRate = constrain(fadeRate,0,255);
    analogWrite(fadePin,fadeRate);
}


void sendDataToProcessing(char symbol, int data )
{
    Serial.print(symbol);
```

```
   Serial.println(data);
  }
```

//INTERRUPT ROUTINE FILE

```
volatile int rate[10];                    // array to hold last ten IBI values
volatile unsigned long sampleCounter = 0;          // used to determine pulse timing
volatile unsigned long lastBeatTime = 0;           // used to find IBI
volatile int P =512;                      // used to find peak in pulse wave, seeded
volatile int T = 512;                     // used to find trough in pulse wave, seeded
volatile int thresh = 512;                // used to find instant moment of heart beat, seeded
volatile int amp = 100;                   // used to hold amplitude of pulse waveform, seeded
volatile boolean firstBeat = true;        // used to seed rate array so we startup with reasonable BPM
volatile boolean secondBeat = false;      // used to seed rate array so we startup with reasonable BPM


void interruptSetup(){
  // Initializes Timer2 to throw an interrupt every 2mS.
  TCCR2A = 0x02;     // DISABLE PWM ON DIGITAL PINS 3 AND 11, AND GO INTO CTC MODE
  TCCR2B = 0x06;     // DON'T FORCE COMPARE, 256 PRESCALER
  OCR2A = 0X7C;      // SET THE TOP OF THE COUNT TO 124 FOR 500Hz SAMPLE RATE
  TIMSK2 = 0x02;     // ENABLE INTERRUPT ON MATCH BETWEEN TIMER2 AND OCR2A
  sei();             // MAKE SURE GLOBAL INTERRUPTS ARE ENABLED
```

```
}


// THIS IS THE TIMER 2 INTERRUPT SERVICE ROUTINE.
// Timer 2 makes sure that we take a reading every 2 miliseconds
ISR(TIMER2_COMPA_vect){                    // triggered when Timer2 counts to 124
  cli();                                   // disable interrupts while we do this
  Signal = analogRead(pulsePin);           // read the Pulse Sensor
  sampleCounter += 2;                      // keep track of the time in mS with this variable
  int N = sampleCounter - lastBeatTime;    // monitor the time since the last beat to avoid noise


    //  find the peak and trough of the pulse wave
  if(Signal < thresh && N > (IBI/5)*3){    // avoid dichrotic noise by waiting 3/5 of last IBI
    if (Signal < T){                       // T is the trough
      T = Signal;                          // keep track of lowest point in pulse wave
    }
  }


  if(Signal > thresh && Signal > P){       // thresh condition helps avoid noise
    P = Signal;                            // P is the peak
  }                                        // keep track of highest point in pulse wave


  //  NOW IT'S TIME TO LOOK FOR THE HEART BEAT
  // signal surges up in value every time there is a pulse
  if (N > 250)
```

```
 {                              // avoid high frequency noise
   if ( (Signal > thresh) && (Pulse == false) && (N > (IBI/5)*3) )


   {
     Pulse = true;                          // set the Pulse flag when we
think there is a pulse
     digitalWrite(blinkPin,HIGH);           // turn on pin 13 LED
     IBI = sampleCounter - lastBeatTime;    // measure time between
beats in mS
     lastBeatTime = sampleCounter;          // keep track of time for
next pulse


     if(secondBeat)
     {                    // if this is the second beat, if secondBeat ==
TRUE
       secondBeat = false;                  // clear secondBeat flag
       for(int i=0; i<=9; i++){             // seed the running total to get a
realisitic BPM at startup
         rate[i] = IBI;
       }
     }


     if(firstBeat)
     {                    // if it's the first time we found a beat, if
firstBeat == TRUE
       firstBeat = false;                   // clear firstBeat flag
       secondBeat = true;                   // set the second beat flag
       sei();                               // enable interrupts again
       return;                              // IBI value is unreliable so discard it
     }
     // keep a running total of the last 10 IBI values
     word runningTotal = 0;                 // clear the runningTotal
variable
```

```
    for(int i=0; i<=8; i++)
    {             // shift data in the rate array
     rate[i] = rate[i+1];              // and drop the oldest IBI value
     runningTotal += rate[i];           // add up the 9 oldest IBI values
    }


    rate[9] = IBI;                        // add the latest IBI to the rate
array
    runningTotal += rate[9];            // add the latest IBI to
runningTotal
    runningTotal /= 17;                // average the last 10 IBI values
    BPM = 60000/runningTotal;


delay(3000);
lcd.setCursor(6,0);
lcd.print(BPM,1);
     QS = true;
   } }
 if (Signal < thresh && Pulse == true)
{
   digitalWrite(blinkPin,LOW);              // turn off pin 13 LED
   Pulse = false;                       // reset the Pulse flag so we can
do it again
   amp = P - T;                         // get amplitude of the pulse wave
   thresh = amp/2 + T;                 // set thresh at 50% of the
amplitude
   P = thresh;                          // reset these for next time
   T = thresh;
 }
```

```
  if (N > 2500){                          // if 2.5 seconds go by
without a beat
    thresh = 512;                         // set thresh default
    P = 512;                              // set P default
    T = 512;                               // set T default
    lastBeatTime = sampleCounter;         // bring the lastBeatTime up
to date
    firstBeat = true;                     // set these to avoid noise
    secondBeat = false;                   // when we get the heartbeat
back
  }
  sei();                                 // enable interrupts when you’re
done!
}
```

APPENDIX B

```
//MAIN ACTIVITY ANDROID APPLICATION
package com.example.knk;

import android.app.Activity;
import android.app.PendingIntent;
import android.net.NetworkInfo;
import android.os.Bundle;
import android.os.SystemClock;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.ListView;
import android.widget.RelativeLayout;
import android.widget.TextView;
import android.widget.Toast;

import java.util.Calendar;
import java.util.LinkedList;
import java.util.List;

public class MainActivity extends Activity {
        public TextView txt, logo, user_name;
        public Button  tmp, login, logout, register, edit,
upload,invite,back,pulse,bluetooth,history,full,stress,friends,calory;
        MySQLiteHelper db = new MySQLiteHelper(this);
         ListView listView ;
         RelativeLayout rl;
```

```
        ImageView img;
    boolean debug = false;

View v;
    private static final int PERIOD=20000;
        @Override
        protected void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_main);


mgr.setRepeating(AlarmManager.ELAPSED_REALTIME_WAKEUP,
                 SystemClock.elapsedRealtime() + 60000,
                 PERIOD,
                 pi);
        }

        }
                main_view();
                img=(ImageView)findViewById(R.id.image);
                rl=(RelativeLayout) findViewById(R.id.layout);


                ////////////////////////////////


                ////////////////////////////////
                pulse=(Button)findViewById(R.id.pulse);
                invite=(Button) findViewById(R.id.invite);
                full=(Button)findViewById(R.id.full);
                friends=(Button)findViewById(R.id.friends);
```

```
        login = (Button) findViewById(R.id.login);
        logout = (Button) findViewById(R.id.logout);
        register = (Button) findViewById(R.id.register);
        edit = (Button) findViewById(R.id.edit);
        upload = (Button) findViewById(R.id.upload);
        user_name = (TextView) findViewById(R.id.name);
        history = (Button) findViewById(R.id.history);
        stress = (Button) findViewById(R.id.stress);
        toast1 = Toast.makeText(this, "Please Turn on Your
internet",
                        Toast.LENGTH_SHORT);
        toast2=Toast.makeText(this, "Bluetooth Turned ON",
Toast.LENGTH_SHORT);
        toast3=Toast.makeText(this, "Bluetooth Turned OFF",
Toast.LENGTH_SHORT);
        log();

    stress.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View arg0) {
                // TODO Auto-generated method stub
                Intent intent = new Intent(MainActivity.this,
Stress.class);
           startActivity(intent);
            }
        });

        full.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View arg0) {
```

```
                        // TODO Auto-generated method stub
                        Intent intent = new Intent(MainActivity.this,
Full.class);
                startActivity(intent);
                 }
            });

            friends.setOnClickListener(new View.OnClickListener() {

                 @Override
                 public void onClick(View arg0) {
                        // TODO Auto-generated method stub
                        Intent intent = new Intent(MainActivity.this,
Friends.class);
                startActivity(intent);
                 }
            });

            bluetooth.setOnClickListener(new View.OnClickListener()
{

                 @Override
                 public void onClick(View arg0) {
                        // TODO Auto-generated method stub
                        if(!mBluetoothAdapter.isEnabled())
                {

                   Intent enableBluetooth = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
                   startActivity(enableBluetooth);
                   toast2.show();
```

```
            }

                    if(mBluetoothAdapter.isEnabled())
                    {
                        mBluetoothAdapter.disable();
                        toast3.show();
                    }

             }
        });

        pulse.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View view) {
              Intent intent = new Intent(MainActivity.this,
Pulse.class);
              startActivity(intent);
            }
        });
        history.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View view) {
              Intent intent = new Intent(MainActivity.this,
reporting.class);
              startActivity(intent);
            }
        });
        login.setOnClickListener(new View.OnClickListener() {
```

```
                @Override
                public void onClick(View view) {

                    Intent intent = new Intent(MainActivity.this,
login.class);
                    startActivity(intent);



                }
            });
            invite.setOnClickListener(new View.OnClickListener() {

                @Override
                public void onClick(View view) {
                    if(haveNetworkConnection()){
                    Intent intent = new Intent(MainActivity.this,
invite_friends.class);
                    startActivity(intent);
                    }else{
                        toast1.show();
                    }
                }
            });
            edit.setOnClickListener(new View.OnClickListener() {

                @Override
                public void onClick(View view) {
                    if(haveNetworkConnection()){
                    Intent intent = new Intent(MainActivity.this,
                            edit_profile.class);
                    startActivity(intent);
```

```
                }
                else{
                        toast1.show();
                }
        }
});
register.setOnClickListener(new View.OnClickListener() {

        @Override
        public void onClick(View view) {

                Intent intent = new Intent(MainActivity.this,
register.class);
                if (haveNetworkConnection()) {
                        startActivity(intent);
                } else {
                        toast1.show();
                }
        }
});
upload.setOnClickListener(new View.OnClickListener() {

        @Override
        public void onClick(View view) {

                Intent intent = new Intent(MainActivity.this,
                                upload_users.class);
                if (haveNetworkConnection()) {
                        startActivity(intent);
                } else {
                        toast1.show();
```

```
                }
            }
        });
        logout.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View view) {

                Intent intent = new Intent(MainActivity.this,
logout.class);
                startActivity(intent);

            }
        });

        calory = (Button) findViewById(R.id.cal);

        calory.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View view) {
            //      Intent intent = new Intent(MainActivity.this,
calories.class);
            //      startActivity(intent);
            }
        });

        tmp = (Button) findViewById(R.id.temp);
        tmp.setOnClickListener(new View.OnClickListener() {

            @Override
```

```
                public void onClick(View view) {
                        Intent intent = new Intent(MainActivity.this,
temperature.class);
                        startActivity(intent);
                }
        });
    }


    public void main_view() {
        txt = (TextView) findViewById(R.id.textView1);
        logo = (TextView) findViewById(R.id.nurse);
        Calendar cal = Calendar.getInstance();
        int date = cal.get(Calendar.DATE);
        int month = cal.get(Calendar.MONTH);
        int year = cal.get(Calendar.YEAR);
        int day = cal.get(Calendar.DAY_OF_WEEK);
        int mnth = cal.get(Calendar.MONTH)+1;


        txt.setTextColor(Color.WHITE);
        logo.setTextColor(Color.RED);
        txt.setText("Today is " + Integer.toString(date) + "/"
                + Integer.toString(mnth) + "/" +
Integer.toString(year));
        logo.setText("EXCEPTIONAL HEALTH CARE");
        logo.setTextColor(Color.BLUE);
        //logo.append(" OWN ");
        // logo.setTextColor(Color.RED);
        logo.append("\n        SOLUTIONS ");


    }
```

```
	@Override
	public boolean onCreateOptionsMenu(Menu menu) {
		// Inflate the menu; this adds items to the action bar if it is present.
		getMenuInflater().inflate(R.menu.main, menu);
		return true;
	}

	@Override
	public boolean onOptionsItemSelected(MenuItem item) {
		// TODO Auto-generated method stub

		switch(item.getItemId())
		{
		case R.id.back:
			finish();
			break;

		}
		return super.onOptionsItemSelected(item);
	}

	@Override
	protected void onResume() {
		// TODO Auto-generated method stub
		super.onResume();
		log();

	}
```

```
public void log() {
        Book book = new Book();
        List<Book> books = new LinkedList<Book>();
        books = null;
        db.getAllBooks();
        books = db.getLogin();
        if (books.size() < 1) {
                login.setVisibility(View.VISIBLE);
                register.setVisibility(View.VISIBLE);
                logout.setVisibility(View.INVISIBLE);
                edit.setVisibility(View.INVISIBLE);
                upload.setVisibility(View.INVISIBLE);
                img.setVisibility(View.VISIBLE);
                invite.setVisibility(View.INVISIBLE);
                friends.setVisibility(View.INVISIBLE);



        } else {
                book = books.get(0);
                user_name.setText("Welcome "+book.getfirst() + " "
+ book.getlast());
                logout.setVisibility(View.VISIBLE);
                edit.setVisibility(View.VISIBLE);
                img.setVisibility(View.INVISIBLE);
                login.setVisibility(View.INVISIBLE);
                register.setVisibility(View.INVISIBLE);
                upload.setVisibility(View.VISIBLE);
                invite.setVisibility(View.VISIBLE);
                friends.setVisibility(View.VISIBLE);
        }
```

```
        }

        public boolean haveNetworkConnection() {
                boolean haveConnectedWifi = false;
                boolean haveConnectedMobile = false;

                ConnectivityManager cm = (ConnectivityManager)
getSystemService(Context.CONNECTIVITY_SERVICE);
                NetworkInfo[] netInfo = cm.getAllNetworkInfo();
                for (NetworkInfo ni : netInfo) {
                        if (ni.getTypeName().equalsIgnoreCase("WIFI"))
                                if (ni.isConnected())
                                        haveConnectedWifi = true;
                        if (ni.getTypeName().equalsIgnoreCase("MOBILE"))
                                if (ni.isConnected())
                                        haveConnectedMobile = true;
                }
                return haveConnectedWifi || haveConnectedMobile;
        }
}
```

APPENDIC C

```
//DATABASE CODE
-- phpMyAdmin SQL Dump
-- version 2.11.4
-- http://www.phpmyadmin.net
--
-- Host: localhost
-- Generation Time: Jun 16, 2015 at 01:34 AM
-- Server version: 5.1.57
-- PHP Version: 5.2.17

SET SQL_MODE="NO_AUTO_VALUE_ON_ZERO";


--
-- Database: `a8658049_sfa`
--


-- --------------------------------------------------------


--
-- Table structure for table `emrs`
--

CREATE TABLE `emrs` (
  `EMR_ID` int(11) NOT NULL AUTO_INCREMENT,
  `User_ID` int(11) NOT NULL,
  `Temperature` int(11) NOT NULL,
  `Pulse` int(11) NOT NULL,
  `Timestamp` timestamp NULL DEFAULT NULL,
```

```
  UNIQUE KEY `EMR_ID` (`EMR_ID`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=534 ;

--
-- Dumping data for table `emrs`
--

INSERT INTO `emrs` VALUES(437, 4, 70, 78, '2015-05-20 14:16:36');
INSERT INTO `emrs` VALUES(436, 4, 77, 78, '2015-05-20 14:16:36');
INSERT INTO `emrs` VALUES(486, 6, 86, 96, '2015-05-21 11:48:51');
INSERT INTO `emrs` VALUES(505, 8, 97, 117, '2015-05-21 15:50:14');
INSERT INTO `emrs` VALUES(506, 8, 97, 91, '2015-05-21 15:51:08');
INSERT INTO `emrs` VALUES(507, 8, 97, 109, '2015-05-21 15:51:45');
INSERT INTO `emrs` VALUES(508, 8, 91, 148, '2015-05-21 15:52:17');
INSERT INTO `emrs` VALUES(509, 8, 97, 64, '2015-05-21 15:52:41');
INSERT INTO `emrs` VALUES(510, 8, 97, 62, '2015-05-21 15:53:12');
INSERT INTO `emrs` VALUES(511, 8, 97, 94, '2015-05-21 15:53:56');
INSERT INTO `emrs` VALUES(512, 8, 97, 94, '2015-05-21 15:54:24');
INSERT INTO `emrs` VALUES(527, 6, 79, 103, '2015-05-22 09:37:52');
INSERT INTO `emrs` VALUES(528, 6, 118, 120, '2015-05-22 09:38:21');
INSERT INTO `emrs` VALUES(529, 6, 75, 99, '2015-05-22 09:38:47');
INSERT INTO `emrs` VALUES(530, 6, 75, 39, '2015-05-22 09:39:17');
INSERT INTO `emrs` VALUES(531, 6, 102, 98, '2015-05-27 15:48:04');
INSERT INTO `emrs` VALUES(532, 6, 78, 121, '2015-06-06 16:34:50');
INSERT INTO `emrs` VALUES(533, 6, 79, 121, '2015-06-06 16:35:11');

-- --------------------------------------------------------

--
-- Table structure for table `feedback`
```

```
--


CREATE TABLE `feedback` (
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `Name` varchar(100) NOT NULL,
  `Email` varchar(100) NOT NULL,
  `Subject` varchar(200) NOT NULL,
  `Message` text NOT NULL,
  `TIME` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`ID`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=9 ;


--
-- Dumping data for table `feedback`
--
INSERT INTO `feedback` VALUES(4, 'Javed', 'javed2014@live.com', 'Service Lanch', 'When the service will be launched?', '2015-03-20 03:14:34');
INSERT INTO `feedback` VALUES(5, 'Admin', 'smartfitnessassistant@gmail.com', 'Reply @ Umer', 'Thanks Umer for Support', '2015-03-20 03:15:58');
INSERT INTO `feedback` VALUES(6, 'Admin', 'smartfitnessassistant@gmail.com', 'Reply @ Javed', ' The service will be launched by July 2015', '2015-03-20 03:18:05');


-- --------------------------------------------------------


--
-- Table structure for table `signup & login`
--
  `Loggedin_loggedout` enum('loggedin','loggedout') NOT NULL DEFAULT 'loggedout',
  `profile_pic_ext` varchar(20) DEFAULT NULL,
  `cover_pic_ext` varchar(20) DEFAULT NULL,
  `status_activated` enum('activated','pending') NOT NULL DEFAULT 'pending',
```

```
  PRIMARY KEY (`Username`),
  UNIQUE KEY `ID` (`user_id`),
  UNIQUE KEY `Email` (`Email`)
) ENGINE=MyISAM  DEFAULT CHARSET=latin1 AUTO_INCREMENT=11 ;

-- Dumping data for table `signup & login`


INSERT INTO `signup & login` VALUES(5, 'Batool', 'Khan', 'Female', '1953-07-08', 6799464670467, 'batool@yahoo.com', 'batool123', 'e10adc3949ba59abbe56e057f20f883e', 'Caregiver', '3365367888', 'batool67', 'H # 675\r\nSt # 109\r\nDHA II\r\nIslamabad', 'Islamabad', 'Caregiver Package', 'loggedin', NULL, 'jpg', 'activated');

INSERT INTO `signup & login` VALUES(4, 'Kiran', 'Khan', 'Female', '1974-04-08', 1234567893542, 'kiran123@live.com', 'kiran117', 'e10adc3949ba59abbe56e057f20f883e', 'Patient', '123458978522', 'kiran117', 'H # 375\r\nSt # 103\r\nModel Town Humak Zimni\r\nLahore', 'Lahore', ' HealthCare Package', 'loggedout', NULL, NULL, 'activated');

INSERT INTO `signup & login` VALUES(2, 'Qasim', 'Raza', 'Male', '1983-01-06', 6886866868666, 'qasimraza92@gmail.com', 'qasim11', 'e10adc3949ba59abbe56e057f20f883e', 'Athlete', '345699800', 'qasim11', 'Bahria town\r\nRawalpindi', 'Rawalpindi', 'Fitness Package', 'loggedout', NULL, NULL, 'activated');

INSERT INTO `signup & login` VALUES(10, 'Javed', 'Hashmi', NULL, NULL, NULL, 'cheeema@gmail.com', 'javed123', 'e10adc3949ba59abbe56e057f20f883e', 'Patient', NULL, NULL, NULL, 'Islamabad', 'HealthCare Package', 'loggedout', NULL, NULL, 'activated');


-- --------------------------------------------------------

-- Dumping data for table `viewership`

INSERT INTO `viewership` VALUES(22, 6, 5);

INSERT INTO `viewership` VALUES(23, 4, 5);
```

BIBLIOGRAPHY& REFRENCING

6.1 LITERATURE

1. Sushama Pawar, “Home Based Health Monitoring System Using Android Smartphone”, International Journal of Electrical, Electronics and Data Communication, 2014

2. Deep Modi, “Android Based Patient Monitoring System”, International Journal For Technological Research In Engineering, 2014.

3. Sasan Haghani, “Implementation of a Wireless Body Area Network for Healthcare Monitoring”, University of the District of Columbia, 4200 Connecticut Ave, Washington, DC, 20008

4. Prema Sundaram, “Patient Monitoring System Using Android Technology”, International Journal of Computer Science and Mobile Computing, IJCSMC, Vol. 2, Issue. 5, May 2013, pg.191 – 201

5. L. Y. K. Y. Heming Pang, Linying Jiang, “Research of android Smartphone surveillance system,” Proceedings of the 2010 International Conference On Computer Design And Applications, vol. 2, pp. 373 –376, 2010

6. V.Ramesh,” Efficient Patient Monitoring For Multiple Patients Using WSN”, International Conference on Advances in Mobile Network, Communication and Its Applications, 2012

7. J. W. Zheng, Z. B. Zhang, T. H. Wu, Y. Zhang, "A Wearable Mobihealth Care System Supporting Real-Time Diagnosis and Alarm", Medical & biological engineering & computing, 2007 Sep;45(9):877-85, PubMed PMID: 17619091, Epub 2007/07/10, eng

6.2 SIMILAR PROJECTS

1. Abdul Moiz Ahmed, “Medical Wireless Sensor Network For Remote Patient Monitoring”, MCS, NUST, 2011

2. AsadNaeem, “Smart M2M”, CEME,NUST, 2013

3. WaleedAkram, “Personal Health Assistant using Body Area Network”, MCS, NUST, 2014

4. ZoltánBenyó,“An open architecture patient monitoring system using standard technologies”, MIT