1 INTRODUCTION:

A body area network (BAN), also referred to as a wireless body area network (WBAN) or a body sensor network (BSN), is a wireless network of wearable computing devices. BAN devices may be embedded inside the body, implants, may be surface-mounted on the body in a fixed position Wearable technology or may be accompanied devices which humans can carry in different positions, in clothes pockets, by hand or in various bags. Whilst there is a trend towards the miniaturization of devices, in particular, networks consisting of several miniaturized body sensor units (BSUs) together with a single body central unit (BCU). Larger decimeter (tab and pad) sized smart devices, accompanied devices, still play an important role in terms of acting as a data hub, data gateway and providing a user interface to view and manage BAN applications, in-situ. The development of WBAN technology started around 1995 around the idea of using wireless personal area network (WPAN) technologies to implement communications on, near, and around the human body. About six years later, the term "BAN" came to refer systems where communication is entirely within, on, and in the immediate proximity of a human body. A WBAN system can use WPAN wireless technologies as gateways to reach longer ranges. Through gateway devices, it is possible to connect the wearable devices on the human body to the internet. This way, medical professionals can access patient data online using the internet independent of the patient location. The rapid growth in physiological sensors, low-power integrated circuits, and wireless communication has enabled a new generation of wireless sensor networks, now used for purposes such as monitoring traffic, crops, infrastructure, and health. **.** The doctor can describe about the patient in the describe tab the described data will be viewed in the list view, so the doctor can use it for future references. The body area network field is an interdisciplinary area which could allow inexpensive and continuous health monitoring with real-time updates of medical records through the Internet.

1.1 Overview:

In medical field, there was less development in medical equipments and healthcare systems. Due to the lack of technologies and facilities, the patient's lives were put to danger in emergency conditions. Medical facilities were providing with full of wires and very complex to handle it for patient. The Medical field is very developing nowadays and Wireless body area network (WBAN) is one of the emerging technologies in the field of healthcare system which is able to change the whole process of the medical systems and its way of delivery. Many medical devices such as Electrocardiography (ECG), temperature and pulse sensors, all have been also moved to WBAN technologies. WBAN technology can able to reduce the problem of wires in the healthcare system and the patient's comfort has been increased and also provides ability for healthcare to monitor the patient remotely via cloud environment. The efficiency of using Bluetooth will be reduced by using Bluetooth Low Energy where the energy consumed is very low. By using the concept of Cognitive IoT, the data generated by the connected devices will be integrated with the actions those devices can perform. Cognitive IoT can perform understanding, reasoning and learning on the generated data.

1.2 Need for the project:

Wireless Body Area Networks (WBANs) supporting healthcare applications are in early development stage but offer valuable contributions at monitoring, diagnostic, or therapeutic levels. They cover real-time medical information gathering obtained from different sensors with secure data communication and low power consumption. As a consequence of the increasing interest in the application of this type of networks. Our system uses technologies and embedded systems to monitor and sends patient details to respectively health centers or doctors. It is also possible that a WBAN will connect itself to the internet to transmit data in a non-invasive manner.

1.3 Objective of the Project

The main objective of the project is to monitor the patient physical parameter and display it in the doctor smart phones. The parameter receive from the sensor will be view in the graphical format and the current reading will be displayed in the main page of the application. The arduino will get all the physical parameter from the sensor and transmit it to the mobile application through Bluetooth module. With the help of graphical view doctor can easily compare the current and old records. The mobile application has the additional features like prescription and description. The doctor can directly add the medicines details in the application itself after checking the patient health parameter. A WBAN node being an autonomous device can search and find a suitable communication network to transmit data to a remote database server for storage.

1.4 Scope of the project:

The Wireless Health Monitoring System uses Body Area Network technology and transmits patient detail digitally and quickly. The technology uses sensors and android application which makes data user friendly. The system uses machine learning and process the patient data and sends to doctor which makes time efficient and faster response to patients. As all the medical industries are trying to implement wireless body area network in medical field, the system has better scope. Body Area Networks (BANs) are a new generation of highly localized wireless networks that are expected to support a wide variety of medical applications, from monitoring the physiological data of patients. A WBAN system can use WPAN wireless technologies as gateways to reach longer ranges. Through gateway devices, it is possible to connect the wearable devices on the human body to the internet. This emerging technological area is to mature into a thriving industry and it will need a better definition of its scope and potential, and a clearer vision for its future.

2 LITERATURE SURVEY:

2.1 A Wireless Health Monitoring System based on Android Operating System, By Dongdong Loua,b, Huan Jina,b, Xingzu Guo a,b, Xianxiang Chena , Yundong Xuana,b, Zhan Zhaoa, Zhihong Xua,b, Zhen Fanga

This paper describes a mobile health monitoring system which consists of a portable multifunctional physiological parameters detecting 3AHcare node and a mobile program for real-time data telemetry based on the smart phone with the Android operating system. The 3AHcare node is a health monitoring device with embedding Bluetooth module in it and capable of measuring a subject's ECG, blood pressure, blood oxygenation, respiration, temperature and motion – almost equivalent to the feature set of a hospital bedside patient monitor. In the Android Application, we receive physiological parameters such as ECG via the socket connection between the Android device and the detecting module; we process the received data with special algorithms to get steady waves; we store data locally on microSD flash; we display data by waveform and digit; we alarm by threshold algorithm and we realize remote data transmission via TCP/IP protocol. We have evaluated the performance of the monitoring system in capturing, recording, transmitting and displaying ambulatory data and found the system easy to use and with high precision.

The android monitoring program of the monitoring system is developed on Eclipse 3.5.2 development environment by Java language, and is debugged and run on a smart phone with the Android operating system. The design and implementation of the Android monitoring application include two aspects: design and implementation of the android APP user interface and android APP functional modules. Two application user interface needs to be designed in this android application, one is used for the control and operation of the application, the other one is used for displaying analog and digital forms of physiological parameters. A series of operations on the physiological data are needed to be completed in the design and implement of the android APP function module. In accordance with the sequence of operation of the physiological parameters, these behaviors of data processing include: collection, pretreatment (separation, format conversion, calculation) , waveform and digital displaying, store, data uploading and downloading, as well as real-time alarm. Android smart phone receive and save the physiological data sent by the 3AHcare node via its local Bluetooth, then display in real-time. At the same time, the user can perform other functions through monitoring program interface. It is designed mainly to contain five layers: Network layer, Abstraction sensing layer, Feature layer, Inference layer and Application layer.

2.2 Real Time Wireless Health Monitoring Application Using Mobile Devices, By Amna Abdullah, Asma Ismael, Aisha Rashid, Ali Abou-ElNour, and Mohammed Tarique

In the last decade the healthcare monitoring systems have drawn considerable attentions of the researchers. The prime goal was to develop a reliable patient monitoring system so that the healthcare professionals can monitor their patients, who are either hospitalized or executing their normal daily life activities. In this work we present a mobile device based wireless healthcare monitoring system that can provide real time online information about physiological conditions of a patient. Our proposed system is designed to measure and monitor important physiological data of a patient in order to accurately describe the status of her/his health and fitness. In addition the proposed system is able to send alarming message about the patient’s critical health data by text messages or by email reports. By using the information contained in the text or e-mail message the healthcare professional can provide necessary medical advising. The system mainly consists of sensors, the data acquisition unit, microcontroller (i.e., Arduino), and software (i.e., LabVIEW). The patient’s temperature, heart beat rate, muscles, blood pressure, blood glucose level, and ECG data are monitored, displayed, and stored by our system. To ensure reliability and accuracy the proposed system has been field tested. The test results show that our system is able to measure the patient's physiological data with a very high accuracy

A modernized healthcare system should provide better healthcare services to people at any time and from anywhere in an economic and patient friendly manner. Currently, the healthcare system is undergoing a cultural shift from a traditional approach to a modernized patient centered approach. In the traditional approach the healthcare professionals play the major role. They need to visit the patients for necessary diagnosis and advising. There are two basic problems associated with this approach. Firstly, the healthcare professionals must be on site of the patient all the time International Journal of Computer Networks & Communications (IJCNC) Vol.7, No.3, May 2015 14 and secondly, the patient remains admitted in a hospital, wired to bedside biomedical instruments, for a period of time. In order to solve these two problems the patient oriented approach has been conceived. In this approach the patients are equipped with knowledge and information to play a more active role in disease diagnosis, and prevention. The key element of this second approach is a reliable and readily available patient monitoring system (PMS).

By encapsulating the advantages of modern bioinstrumentation, computers, and telecommunication technologies a modern PMS should acquire, record, display, and transmit the physiological data from the patient body to a remote location at any time. For more efficient, timely, and emergency medical care the PMS must also be incorporated with an alarm system. In order to alert the patient as well as the health care service providers the PMS should not only monitor and analyze the critical patient's data but it should also send alarming messages in case the monitored data go outside their normal ranges. Hence, an active database system must be associated with the PMS. Most of the proposed PMSs are centralized in a sense that all patients' data are stored in a single server

2.3 Wireless Body Area Networks: Applications and Technologies, By Imen Jemilia, Rim Negraa

The increasing use of wireless networks and the constant miniaturization of electrical invasive/non-invasive devices have empowered the development of Wireless Body Area Networks (WBANs). A WBAN provides a continuous health monitoring of a patient without any constraint on his/her normal daily life activities. Many technologies have proved their efficiency in supporting WBANs applications, such as remote monitoring, biofeedback and assisted living by responding to their specific quality of service (QoS) requirements. Due to numerous available technologies, selecting the appropriate technology for a medical application is being a challenging task. In this paper, the different medical applications are presented. The most common technologies used in WBANs are highlighted. Finally, a matching between each application and the corresponding suitable technology is studied

Ubiquitous healthcare is an emerging technology that promises increases in efficiency, accuracy and availability of medical treatment due to the recent advances in wireless communication and in electronics offering small and intelligent sensors able to be used on, around, in or implanted in the human body. In this context, Wireless Body area networks (WBANs) constitute an active field of research and development as it offers the potential of great improvement in the delivery and monitoring of healthcare 22. WBANs consist of a number of heterogeneous biological sensors. These sensors are placed in different parts of the body and can be wearable or implanted under the user skin. Each of them has specific requirements and is used for different missions. These devices are used for measuring changes in a patient vital signs and detecting emotions or human statuses, such as fear, stress, happiness, etc. They communicate with a special coordinator node, which is generally less energy constrained and has more processing capacities. It is responsible for sending biological signals of the patient to the medical doctor in order to provide real

2.4 Feasibility study:

- **Technically feasible:**

The system uses arduino board and three sensors (ECG, Pulse and Temperature) for retrieving physiological data. These sensors are easily available in medical fields. The arduino board is mostly used for data control and management and transmission in digital devices. So the arduino is a feasible one. The system also uses BLE (Bluetooth Low Energy) for transmit data from arduino to nearby devices (mobile). This Bluetooth sensor is new and advanced sensor and it uses low energy for transmission. So it has lots of future scope and it is easily available. The system also uses android application development using Android Studio tool which is available in online. The system mostly uses available devices in the market, so the overall technical features of the system are feasible.

- **Economic feasible:**

The WBAN uses the arduino board, ECG sensor, pulse rate sensor, temperature sensor and android devices. These resources are easily achievable and available at acceptable cost. Most of the persons have an android smart phone and it makes the system user friendly. The resource for the system is completely achievable at acceptable cost.

- **Socially feasible:**

The WBAN supports low cost, low complexity and highly reliable wireless communication system. So the system is worth for the available cost and is most needed in this society. It contributes for the digital field in medical industry. The Bluetooth sensor in the system uses less energy than other Bluetooth sensors, so the energy is also conserved using this system.

3 SYSTEM DESIGN:

3.1 Proposed System Architecture Design:

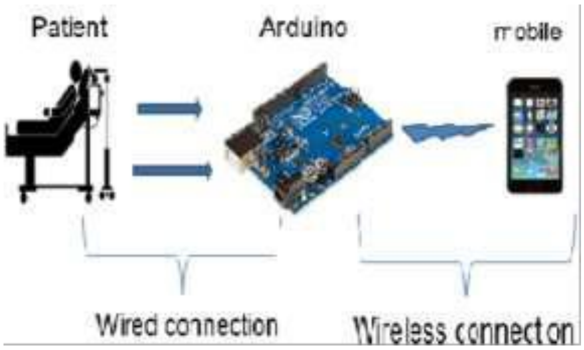

Fig 3.1.1 basic structure of WBAN

This diagram shows the basic modules used in the healthcare monitoring system. The patient date is collected using arduino board which is embedded with the sensors and sent to mobile application wirelessly.

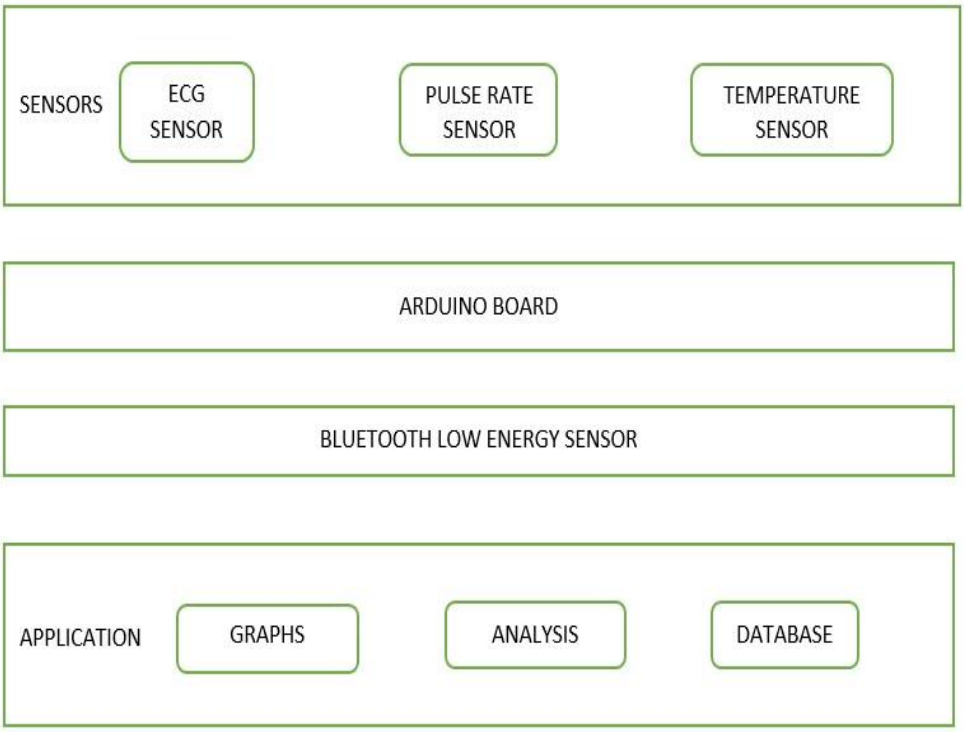

Fig 3.1.2 components of WBAN

3.1.1 Devices and technology used in our System:

There are various devices in WBAN architecture.

[1] Sensor nodes: It is a node in a sensor network that is capable of performing some processing, gathering sensory information and communicating with other connected nodes in the network.

[2] Arduino board: Arduino is a computer hardware and software company, project, and user community that designs and manufactures microcontroller kits for building digital devices and interactive objects that can sense and control objects in the physical world.

[3]Database: In this unit database is maintained andfurther sent to a specialist for consultancy or proper medical guidance.

[4] User Application: The data received from via BLE will be analysed using the android mobile application. This module has two processes namely, Graphical view, Analysis.

3.2 Data flow diagram for proposed system:

The data flow diagram starts with the patients detail register. The patient details are collected and processed in the system, so patients has to register the details before using the system. Then patient physiological data is transferred through sensor and received through Bluetooth to paired device. The data will get continuous updation upon patient current status. Then the data is processed and in treatment and diagnosis, medicine is given to the respective patient. These details are stored in the database. If the patient data is received normal within the threshold time, the alert will not be sent.

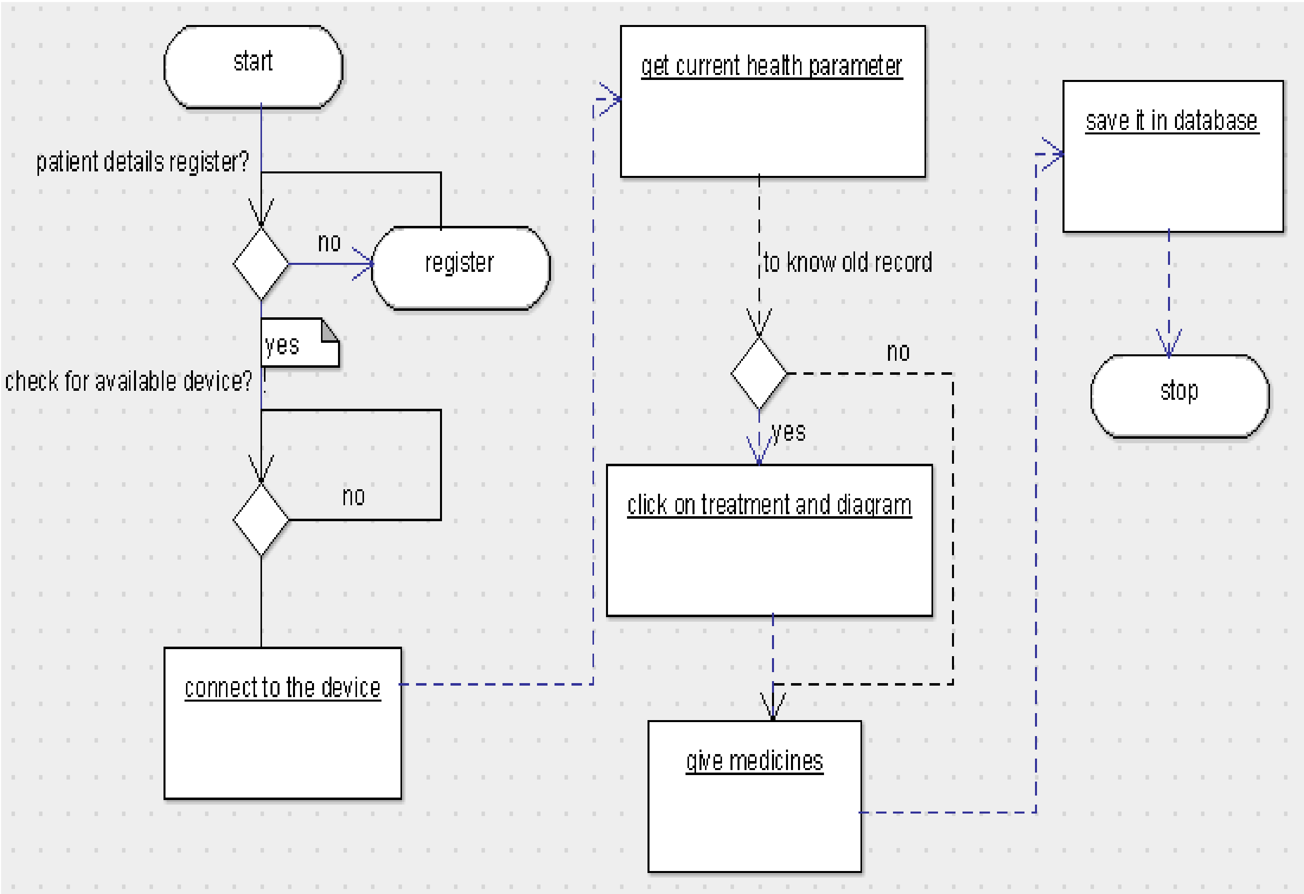

Fig 3.2.1 proposed data flow diagram

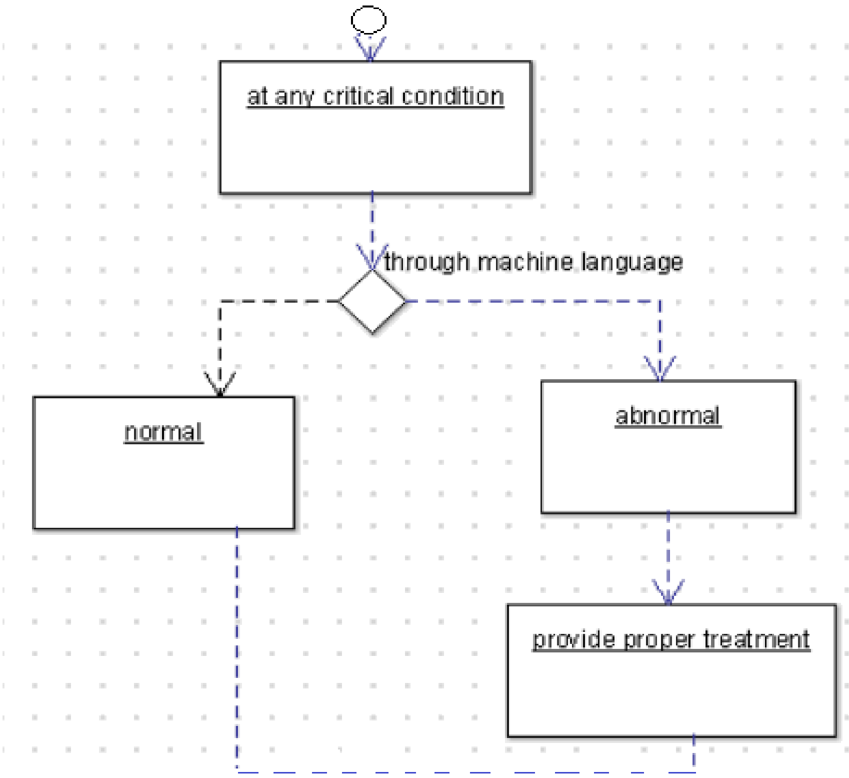

Fig 3.2.2 critical condition data flow diagram

This module shows how the system works during the critical condition of patient. If any abnormal data is received in the database, it waits for certain threshold time. If abnormal data is received continuously within the time, the system sends the alert message to the doctor. Then proper treatment is provided. If the patient data is received normal within the threshold time, the alert will not be sent.

3.2.1 Transmission Of Data:

Bluetooth low energy (BLE) is the power-version of Bluetooth that was built for the Internet of Things (IOT). The power-efficiency of Bluetooth with low energy functionality makes it perfect for devices that run for long periods on power sources, such as coin cell batteries or energy-harvesting devices. Native support for Bluetooth technology on every major operating system enables development for a broad range of connected devices, from home appliances and security systems to fitness monitors and proximity sensors. This module acts as an interface between Arduino and The Android smart phone. The transmission range if BLE will exceed 330ft.

3.3 Module Design:

3.3.1 SENSOR MODULE DESCRIPTION:

In this module, the patient physiological data is collected through sensors and data is stored in the respective column in the database i.e. if ECG data is received, the data is stored in the ECG column using the sensor identifier. These data are then used for the future reference. These sensors are connected in the arduino board which gets the data from the sensor and stores in the database. Likewise it will receive data from the sensor through arduino and store in an specific column. . If temperature data is received, the data is stored in the ECG column using the sensor identifier. . If pulse rate data is received, the data is stored in the ECG column using the sensor identifier.

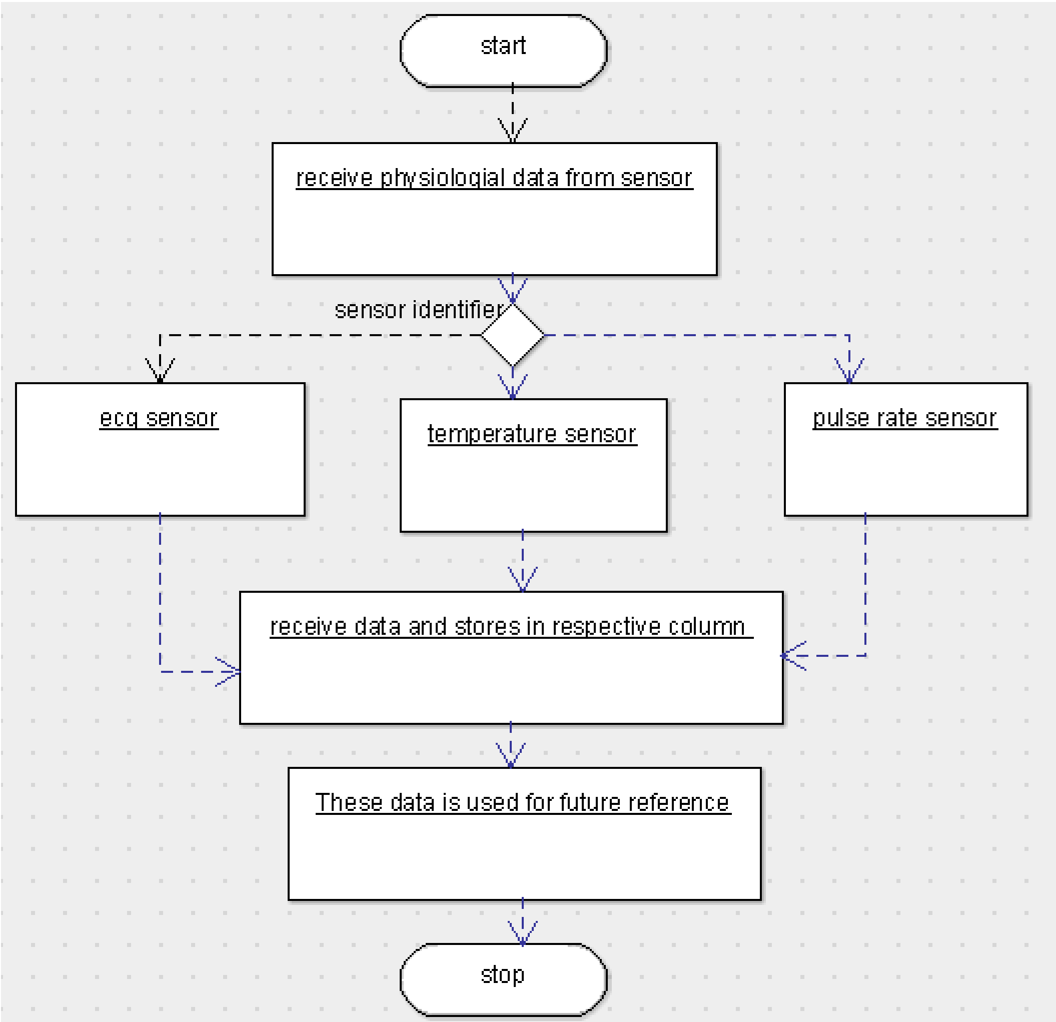

Fig 3.3.1: sensor module

PHYSIOLOGICAL DATA AND THE SENSORS:

This module explains about the working of wireless body area network. There are several sensors attached to the body of the patient which sense the physiological parameter. The sensors used in our project were ECG sensor, temperature sensor and pulse rate sensor. The sensor and Arduino are connected through wired connection. The electrocardiogram (ECG) is a diagnostic tool that is routinely used to assess the electrical and muscular functions of the heart. A temperature sensor is a device, typically, a thermocouple or RTD, that provides for temperature measurement through an electrical signal. A heart rate monitor is a personal monitoring device that allows one to measure one's heart rate in real time or record the heart rate for later study.

Sensor nodes:

These form the base of any WBAN. As on this basic stage, there are various sensors to monitor the physiological parameters i.e. BP, ECG, Pulse Rate, EEG, etc. These sensors nodes work in close vicinity to, on or in the body; picks up the bio signal and gives to further unit for processing. The sensors can be either mono or multifunctional. Nodes can be Implant node, Body surface nodes, and External node.

3.3.2 APPLICATION MODULE DESCRIPTION:

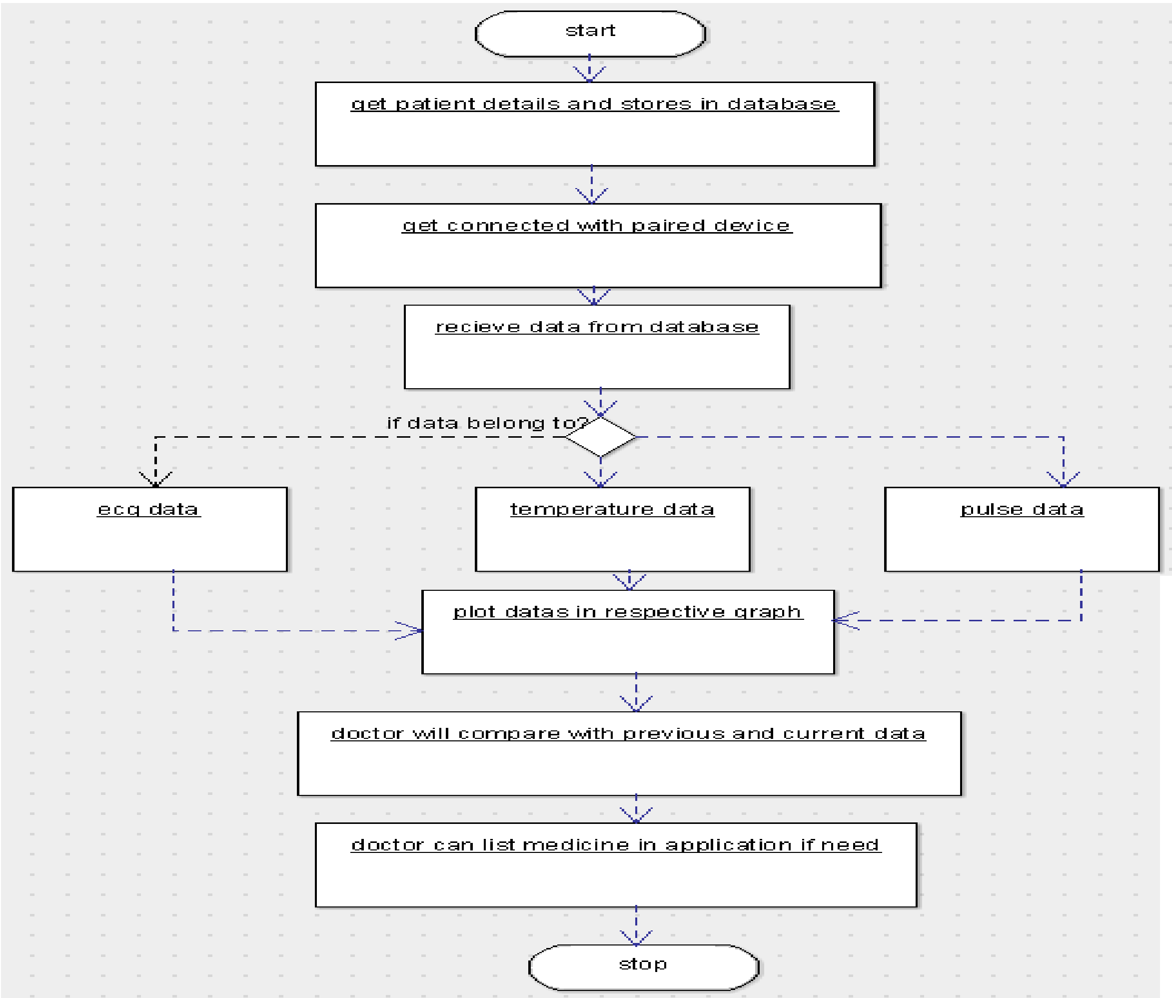

Fig 3.3.2: Application module

In application module, the data is received from the database to the android application. Before that, the device should get connected through Bluetooth with the arduino board. The application uses the current data from the arduino board and the data from the database and process the result in form of graph. ECG data is plotted in the ECG graph, temperature data is plotted in the temperature graph and pulse data is plotted in the pulse graph. Then the doctor will compare the current data with the previous data and provide medicine if necessary.

EXPERIMENTAL RESULTS:

The result will be in the form of graph i.e. for the pulse rate sensor the x axis will be the time in seconds and the y-axis will be the pulse rate acquired per 60 seconds. The received data will be automatically plotted in the graph. The graph also contains the historical datasets for future reference. If any abnormal condition occurs then alert message will be sent to the respective physician.

I. ECG DATA OUTPUT

The arduino board will receive serial input data from the respective sensor that was attached in the arduino board. The signal obtained from the body is filtered and amplified. The sensor outputs an analog signal which is then converted by the analog-to-digital converter (ADC). The serial-to-Bluetooth module transmits the digital output of the ADC to the smart phone. On the phone the sampled ECG is displayed. The ECG sensor will get the human physical parameter through the sensor and then pass it to the android mobile phone. The data received from the sensor will be plotted in graph. With the help of graph view the current and old record can be compared. When the ECG data abnormal suddenly alert message will be sent to the near by doctor and nurse around them. So that they can take necessary action to take care of the patient. The patient health condition will be recognized through Iot check whether patient health is good or poor.

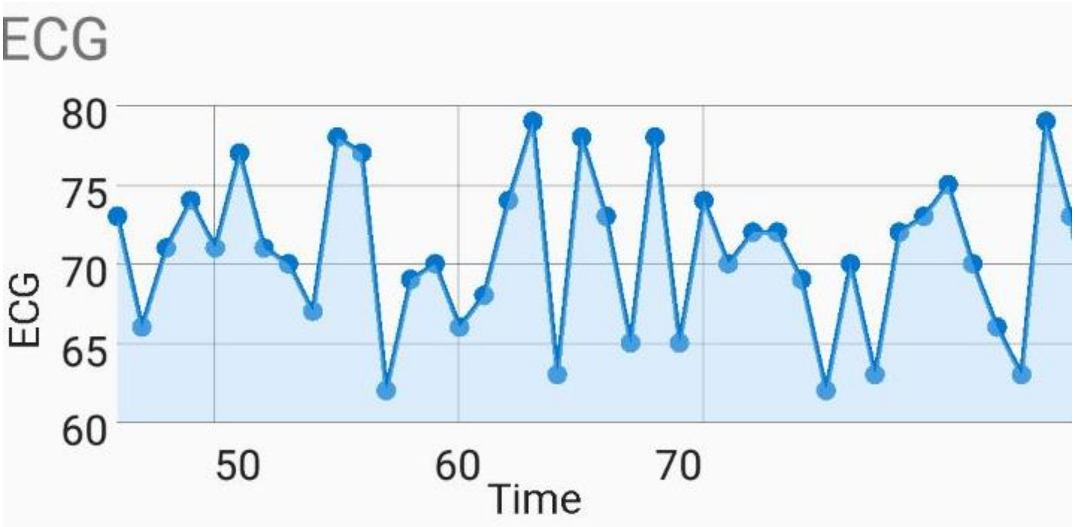

Fig 3.3.3: ECG output

II. TEMPERATURE DATA OUTPUT

The arduino board will receive serial input data from the respective sensor that was attached in the arduino board. The temperature sensor will get the human physical parameter through the sensor and then pass it to the android mobile phone. The data received from the sensor will be plotted in graph. With the help of graph view the current and old record can be compared.

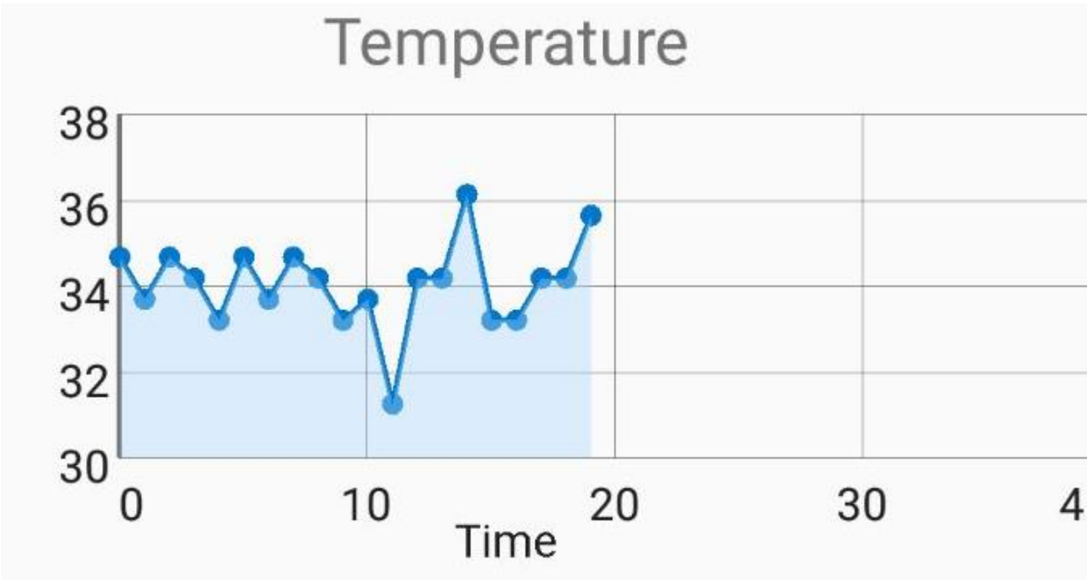

Fig 3.3.4: TEMPERATURE output

3.3.3 Use case diagram:

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use case in which the user is involved. A use case diagram can identify the different types of users of a system and the different use cases and will often be accompanied by other types of diagrams as well.

The use case diagram shows the functions of each user (patient and doctor) in the system. The patient registers the details and it is stored in the database. Then the patient physiological data is retrieved using the sensor and stores in the database and also sends through Bluetooth sensor if nearby device is available. The doctor receives the data and application process the data and shows the output in graphical form. The doctor will then provide treatment and diagnosis and give medicine using the database to the patient. These details are then stored in the database for future reference.

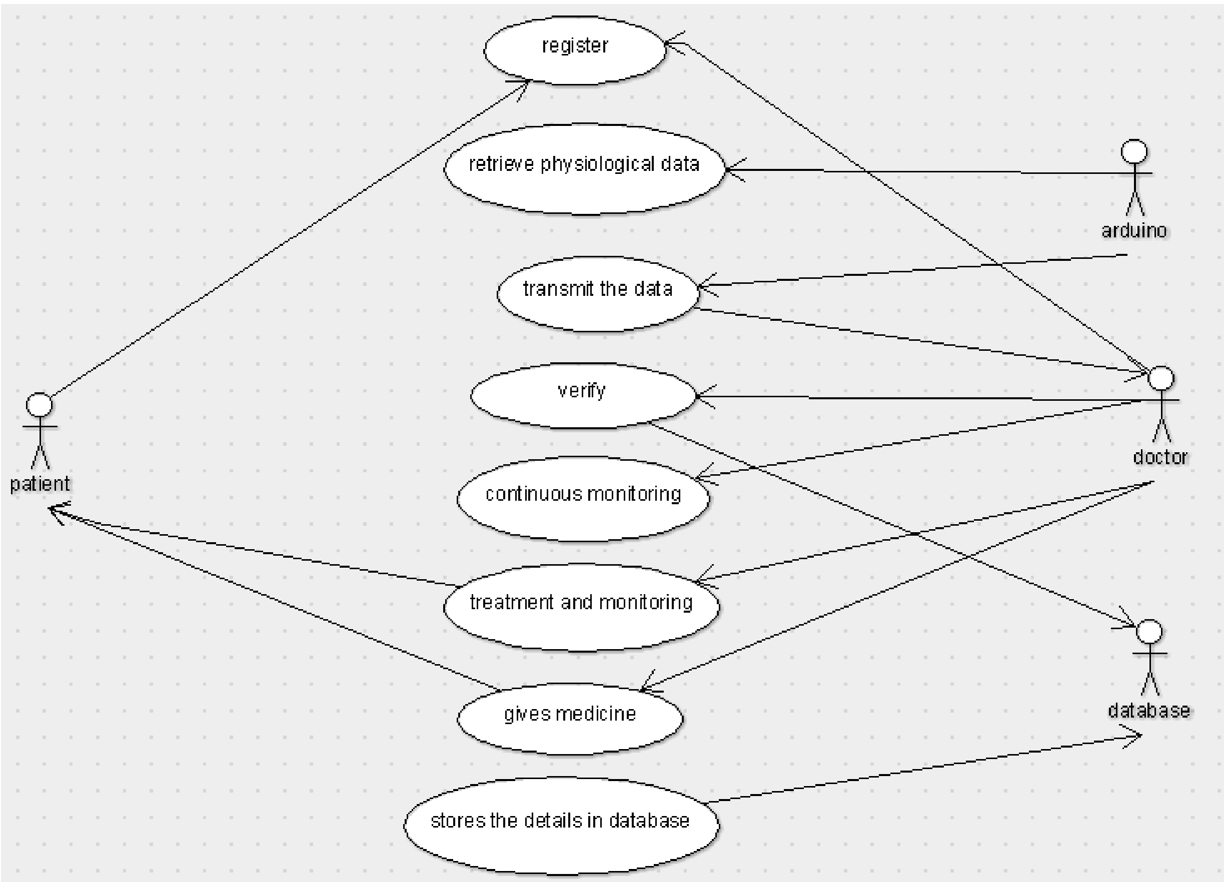

Fig: 3.3.5 use case diagram for WBAN

3.3.4 Class diagram:

A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects. The class diagram is the main building block of object-oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed.

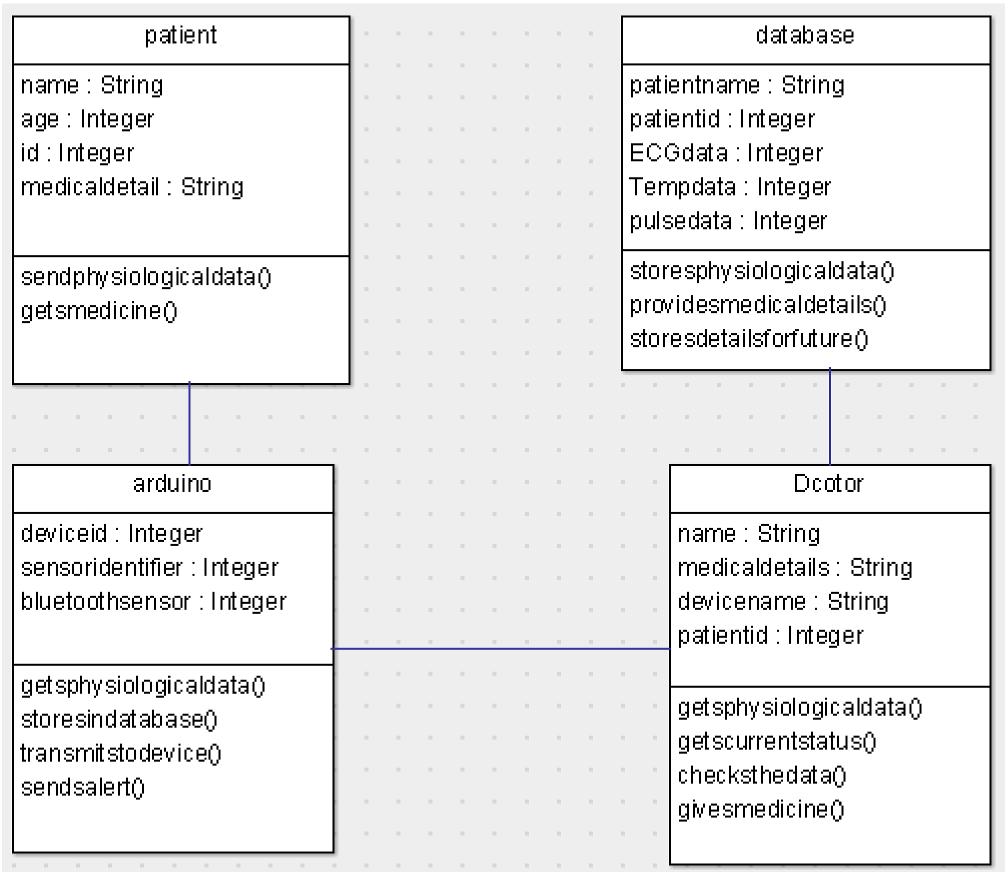

Fig 3.3.6 class diagram for WBAN.

3.3.5 Work flow of WBAN

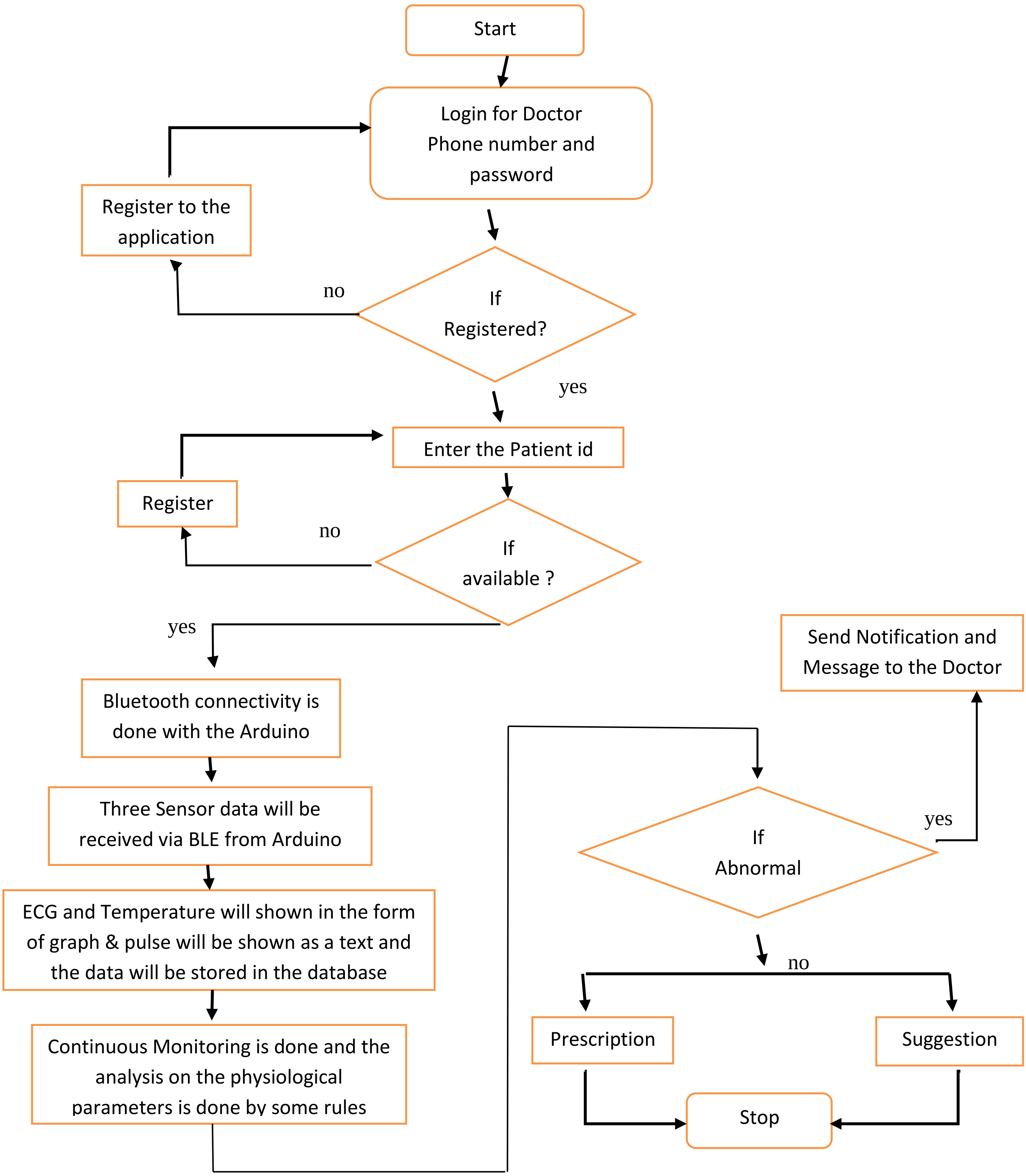

Fig 3.3.7 Work flow of WBAN

4. REQUIREMENT SPECIFICATION:

4.1. Hardware Requirement:

The hardware used for this project are

- Arduino board
- Bluetooth sensor
- ECG sensor
- Pulse rate sensor
- Temperature sensor

4.1.1 Arduino:

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing. It is designed to make electronics more accessible to artists, designers, hobbyists and interested in creating interactive objects or environments.

Arduino board designs use a variety of microprocessors and controllers. The boards are equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards and other circuits. The boards feature serial communications interfaces, including Universal Serial Bus (USB) on some models, which are also used for loading programs from personal computers. The microcontrollers are typically programmed using a dialect of features from the programming languages C and C++. In addition to using traditional compiler toolchains, the Arduino project provides an integrated development environment (IDE) based on the Processing language project.

Arduino microcontrollers are pre-programmed with a boot loader that simplifies uploading of programs to the on-chip flash memory. The default bootloader of the Arduino UNO is the optiboot bootloader. Boards are loaded with program code via a serial connection to another computer. Some serial Arduino boards contain a level shifter circuit to convert between RS-232 logic levels and transistor–transistor logic (TTL) level signals. Current Arduino boards are programmed via Universal Serial Bus (USB), implemented using USB-to-serial adapter chips such as the FTDI FT232. Some boards, such as later-model Uno boards, substitute the FTDI chip with a separate AVR chip containing USB-to-serial firmware, which is reprogrammable via its own ICSP header. Other variants, such as the Arduino Mini and the unofficial Boarduino, use a detachable USB-to-serial adapter board or cable, Bluetooth or other methods, when used with traditional microcontroller tools instead of the Arduino IDE, standard AVR in-system programming (ISP) programming is used.

The Arduino board exposes most of the microcontroller's I/O pins for use by other circuits. The Diecimila, Duemilanove, and current Uno provide 14 digital I/O pins, six of which can produce pulse-width modulated signals, and six analog inputs, which can also be used as six digital I/O pins. These pins are on the top of the board, via female 0.1-inch (2.54 mm) headers. Several plug-in application shields are also commercially available. The Arduino Nano, and Arduino-compatible Bare Bones Board and Boarduino boards may provide male header pins on the underside of the board that can plug into solderless breadboards.

Many Arduino-compatible and Arduino-derived boards exist. Some are functionally equivalent to an Arduino and can be used interchangeably. Many enhance the basic Arduino by adding output drivers, often for use in school-level education, to simplify making buggies and small robots. Others are electrically equivalent but change the form factor, sometimes retaining compatibility with shields, sometimes not. Some variants use different processors, of varying compatibility.

Features explanation:

1.Memory:

The Arduino Uno has 32 KB memory. It comes with 2 KB of SRAM and also 1 KB of EEPROM (EEPROM library is required to read or write into this).

2.Clock Speed:

The performance of this controller is based on its clock speed. The Clock speed of the Arduino is 16 Mhz so it can perform a particular task faster than the other processor or controller

3.USB Interface:

Most important feature of Arduino Uno is USB connectivity. It means if we want to operate Arduino with PC, then we can do that and data communication between PC and Arduino become easy.

4.Input Output Voltage:

The Arduino Uno can be powered via the USB connection or with an external power supply. If we are using external power then we can supply 6 to 20 volts. Arduino works on 5 volts.

5.Input Output Pins:

Each of the 14 digital pins on the Uno can be used as an input or output. 6 pins out of 14 can be used as PWM output. 6 pins can be used as analog pins.

6.Communication:

Arduino board supports I2C and SPI communication. The Arduino software includes wire library for I2C and SPI library for the SPI communication. These are the main features of Arduino Uno. Another important thing is no requirement of a physical press of reset button because it can be reset by software running on a connected computer. At last I can say that Arduino Uno is the most suitable controller in automation industries because of its simple yet effective features.

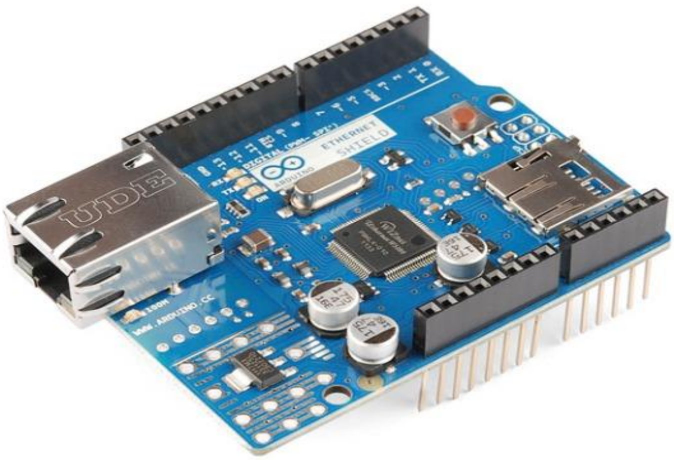

Fig 4.1.1 Arduino board

4.1.2 Bluetooth sensor:

Bluetooth low energy (BLE) is the power- version of Bluetooth that was built for the Internet of Things (IOT). The power-efficiency of Bluetooth with low energy functionality makes it perfect for devices that run for long periods on power sources, such as coin cell batteries or energy-harvesting devices. Native support for Bluetooth technology on every major operating system enables development for a broad range of connected devices, from home appliances and security systems to fitness monitors and proximity sensors. It is a wireless personal area network technology designed and marketed by the Bluetooth Special Interest Group aimed at novel applications in the healthcare, fitness, beacons, security, and home entertainment industries. Compared to Classic Bluetooth, Bluetooth Smart is intended to provide considerably reduced power consumption and cost while maintaining a similar communication range.

Bluetooth makes it easy for developers and OEMs to create innovative new products that communicate with the billions of Bluetooth enabled devices already in the market. Bluetooth with low energy is inexpensive and developer-friendly, with a flexible development architecture that means you’re only limited by your imagination. And as a Bluetooth SIG member, you can tap into a network of tens of thousands of application developers, device makers and service providers.

Key features of Bluetooth with low energy include:

The Industry-standard wireless protocol that allows for interoperability across platforms. Ultra-low peak, average and idle mode power consumption. Standardized application development architecture eases development and deployment time and cost. Allows for some of the government-grade security with 128-bit AES data encryption.

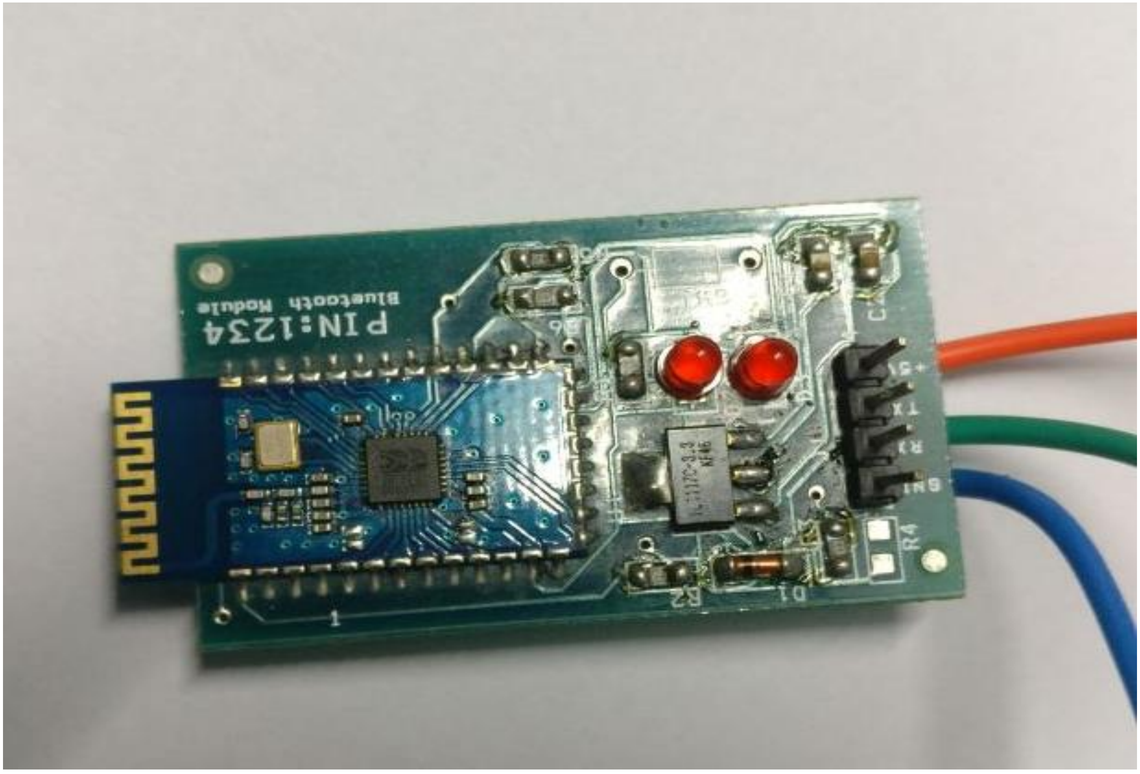

Fig 4.1.2 Bluetooth sensor

4.1.3 Pulse rate sensor:

The Pulse Sensor is a plug-and-play heart-rate sensor for Arduino. It essentially combines a simple optical heart rate sensor with amplification and noise cancellation circuitry making it fast and easy to get reliable pulse readings. Also, it sips power with just 4mA current draw at 5V so it’s great for mobile applications. Simply clip the Pulse Sensor to your earlobe or finger tip and plug it into your 3 or 5

Volt Arduino and you’re ready to read heart rate! The 24" cable on the Pulse Sensor is terminated with standard male headers so there’s no soldering required.

Characteristics of pulse rate sensor:

- **Rate**

Normal pulse rates at rest, in beats per minute (BPM):

newborn (0–3 months old)	infants (3 – 6 months)	infants (6 – 12 months)	children (1 – 10 years)	children over 10 years & adults, including seniors	well-trained adult athletes
100-150	90–120	80-120	70–130	60–100	40–60

The pulse rate can be used to check overall heart health and fitness level. Generally lower is better, but bradycardias can be dangerous. Symptoms of a dangerously slow heartbeat include weakness, loss of energy and fainting.

- **Rhythm**

A normal pulse is regular in rhythm and force. An irregular pulse may be due to sinus arrhythmia, ectopic beats, atrial fibrillation, paroxysmal atrial tachycardia, atrial flutter, partial heart block etc. Intermittent dropping out of beats at pulse is called "intermittent pulse". Examples of regular intermittent (regularly irregular) pulse include pulsus bigeminus, second-degree atrioventricular block. An example of irregular intermittent (irregularly irregular) pulse is atrial fibrillation.

- **Volume**

The degree of expansion displayed by artery during diastolic and systolic state is called volume. It is also known as amplitude, expansion or size of pulse.

- **Hypokinetic pulse**

A weak pulse signifies narrow pulse pressure. It may be due to low cardiac output (as seen in shock, congestive cardiac failure), hypovolemia, valvular heart disease (such as aortic outflow tract obstruction, mitral stenosis, aortic arch syndrome) etc.

- **Hyperkinetic pulse**

A bounding pulse signifies high pulse pressure. It may be due to low peripheral resistance (as seen in fever, anemia, thyrotoxicosis, hyperkinetic heart syndrome (de), A-V fistula, Paget's disease, beriberi, liver cirrhosis), increased cardiac output, increased stroke volume (as seen in anxiety, exercise, complete heart block, aortic regurgitation), decreased distensibility of arterial system (as seen in atherosclerosis, hypertension and coarctation of aorta).

The strength of the pulse can also be reported:

- 0 = Absent
- 1 = Barely palpable
- 2 = Easily palpable
- 3 = Full
- 4 = Aneurysmal or bounding pulse

- **Force**

It is also known as compressibility of pulse. It is a rough measure of systolic blood pressure.

- **Tension**

It corresponds to diastolic blood pressure. A low tension pulse (pulsus mollis), the vessel is soft or impalpable between beats. In high tension pulse (pulsus durus), vessels feel rigid even between pulse beats.

- **Form**

A form or contour of a pulse is palpatiory estimation of arteriogram. A quickly rising and quickly falling pulse (pulsus celer) is seen in aortic regurgitation. A slow rising and slowly falling pulse (pulsus tardus) is seen in aortic stenosis.

- **Equality**

Comparing pulses and different places gives valuable clinical information. A discrepant or unequal pulse between left and right radial artery is observed in anomalous or aberrant course of artery, coarctation of aorta, aortitis, dissecting aneurysm, peripheral embolism etc. An unequal pulse between upper and lower extremities is seen in coarctation to aorta, aortitis, block at bifurcation of aorta, dissection of aorta, iatrogenic trauma and arteriosclerotic obstruction.

- **Condition of arterial wall**:

A normal artery is not palpable after flattening by digital pressure. A thick radial artery which is palpable 7.5–10 cm up the forearm is suggestive of arteriosclerosis.

- **Radio-femoral delay**:

In coarctation of aorta, femoral pulse may be significantly delayed as compared to radial pulse (unless there is coexisting aortic regurgitation). The delay can also be observed in supravalvar aortic stenosis.

Heart beat sensor works on a very basic principle of optoelectronics. All it takes to measure you heart rate is a pair of LED and LDR and a microcontroller. IR led emits infrared radiation and surface reflects the infrared light. Depending on reflectivity of the surface amount of light reflected varies this reflected light is made incident on reverse biased IR sensor which results in reverse leakage current. Amount of electron-hole pairs generated depends on intensity of incident IR radiation. More intense radiation results in more reverse leakage current. This current can be passed through a resistor to get proportional voltage. Thus as intensity of incident rays varies, voltage across resistor will vary accordingly.

APPLICATIONS

- Navigation systems.
- Environmental control systems.
- Augmentative communication devices.
- Emergency response systems.

Resting heart rate chart for men							
Age in Years	Heartbeats Per Minute						
	Athlete	Excellent	Good	Above Average	Average	Below Average	Poor
18-25	49-55	56-61	62-65	66-69	70-73	74-81	82+
26-35	49-54	55-61	62-65	66-70	71-74	75-81	82+
36-45	50-56	57-62	63-66	67-70	71-75	76-82	83+
46-55	50-57	58-63	64-67	68-71	72-76	77-83	84+
56-65	51-56	57-61	62-67	68-71	72-75	76-81	82+
65+	50-55	56-61	62-65	66-69	70-73	74-79	80+

Table 4.1.1 Heart rate for men

Resting heart rate chart for women							
Age in Years	**Heartbeats Per Minute**						
	Athlete	**Excellent**	**Good**	**Above Average**	**Average**	**Below Average**	**Poor**
18-25	54-60	61-65	66-69	70-73	74-78	79-84	85+
26-35	54-59	60-64	65-68	69-72	73-76	77-82	83+
36-45	54-59	60-64	65-69	70-73	74-78	79-84	85+
46-55	54-60	61-65	66-69	70-73	74-77	78-83	84+
56-65	54-59	60-64	65-68	69-73	74-77	78-83	84+
65+	54-59	60-64	65-68	69-72	73-76	77-84	84+

Table 4.1.2 Heart rate for women

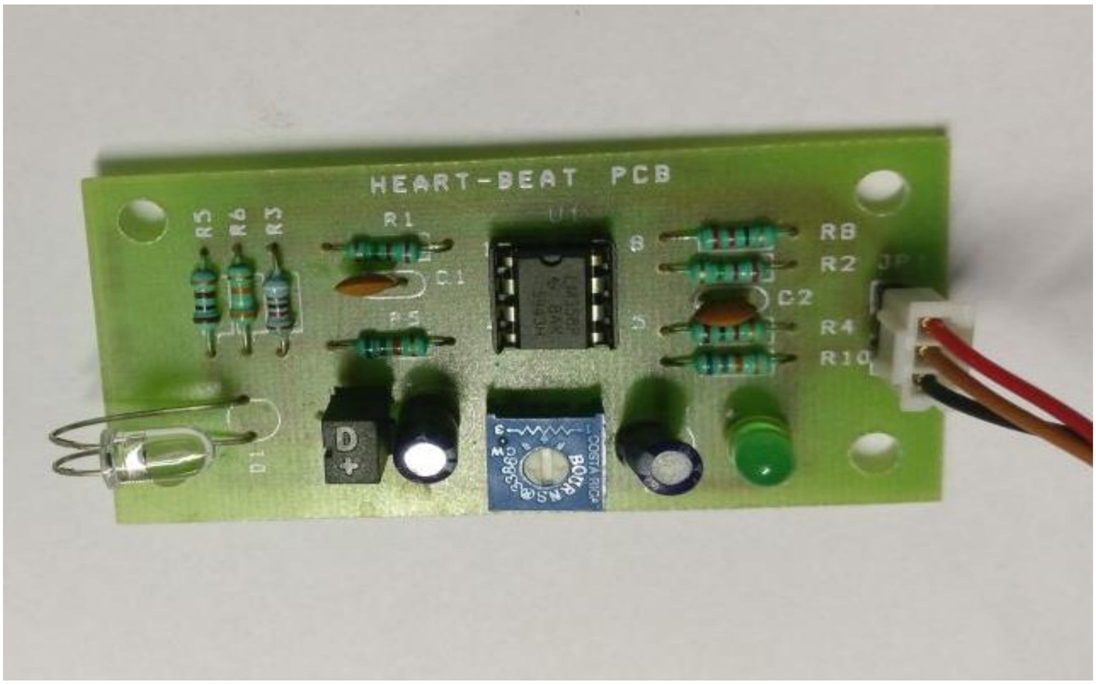

Fig 4.1.3 Pulse Rate sensor

4.1.4 ECG sensor:

The Electrocardiogram (ECG) allows you to assess the electrical and muscular functions of the heart. Heartbeats are triggered by bioelectrical signals of very low amplitude generated by a special set of cells in the heart (the SA node). Electrocardiography (ECG) enables the translation of these electrical signals into numerical values, enabling them to be used in a wide array of applications. Our sensor allow data acquisition not only at the chest ("on-the-person"), but also at the hand palms ("off-the-person"), and works both with pre-gelled and most types of dry electrodes. The bipolar configuration is ideal for low noise. The ECG sensor is attached to the patient using disposable electrodes on the left and right side of the chest. The signal obtained from the body is filtered and amplified. The sensor outputs an analog signal which is then converted by the analog-to-digital converter (ADC). The serial-to-Bluetooth module transmits the digital output of the ADC to the cellphone. On the phone the sampled ECG is displayed.

Fig 4.1.4 ECG sensor

Abnormal state in ECG:

Many people may have heart rhythm disturbances and never be aware of them. Premature atrial contractions (PACs) and premature ventricular contractions (PVCs) are variations of normal and most often, people are unaware that an extra beat has occurred. However, some patients are keenly aware of any extra heartbeat, even if it is a normal variant and requires no treatment. That said, the initial symptom of dysrhythmia is often palpitations, a sensation that the heart is beating too quickly, too slowly, beating irregularly, or skipping a beat. The palpitations may be intermittent or may require medical intervention to resolve. Because of the heart rhythm abnormality, other symptoms may occur because of decreased cardiac output (the amount of blood that the heart pushes out to meet the body's demand for oxygen and energy). The patient may complain of lightheadedness, weakness, nausea and vomiting, chest pain, and shortness of breath. In critical situations, the patient may fall to the ground or lose consciousness. This may be due to life-threatening dysrhythmias like ventricular fibrillation or ventricular tachycardia. It may be due to heart rates so fast that there isn't enough blood pressure generated to supply the brain with what it needs. The same result can also occur if the heart beats too slowly and insufficient blood pressure is generated.

How abnormal state is treated:

Treatment response to an abnormal EKG will typically depend upon the underlying cause. For example, some people experience a very slow heartbeat where the heart is not conducting electrical signals in the correct order. This person may require a pacemaker, which helps to restore the heart to a more normal rhythm. Other people may require medications taken regularly to maintain a more normal heart rhythm. People who are experiencing a heart attack may require cardiac catheterization or surgery to allow for blood flow to return to the heart. People with electrolyte imbalances may require correction with medications or fluids. For example, a dehydrated person may have imbalanced electrolytes that are causing an abnormal EKG. This person may require fluids, electrolyte-containing beverages, or medicines to restore electrolytes.

FEATURES

- Bipolar Differential Measurement.
- LED Indicator
- Input voltage: 5v
- Output: Analog
- Output voltage : 5v
- 3.5mm Jack for Biomedical Pad Connection

APPLICATIONS

- Heart rate Variability.
- Human - Computer Interaction.
- Biometrics.
- Bio-Feedback.
- Biomedical Devices Prototyping.

4.1.5 Temperature sensor:

A temperature sensor is a device, typically, a thermocouple or RTD, that provides for temperature measurement through an electrical signal. A thermocouple (T/C) is made from two dissimilar metals that generate electrical voltage in direct proportion to changes in temperature.

An RTD (Resistance Temperature Detector) is a variable resistor that will change its electrical resistance in direct proportion to changes in temperature in a precise, repeatable and nearly linear manner.

The LM35 series are precision integrated-circuit temperature devices with an output voltage linearly-proportional to the Centigrade temperature. The LM35 device has an advantage over linear temperature sensors calibrated in Kelvin, as the user is not required to subtract a large constant voltage from the output to obtain convenient Centigrade scaling.

The LM35 device does not require any external calibration or trimming to provide typical accuracies of ±¼°C at room temperature and ±¾°C over a full −55°C to 150°C temperature range.

The low-output impedance, linear output, and precise inherent calibration of the LM35 device makes interfacing to readout or control circuitry especially easy to use the temperature sensor.

Body Temperature Chart		
Classed as:	**Celsius**	**Fahrenheit**
Hypothermia	<35.0°C	95.0°F
Normal	**36.5 - 37.5°C**	**97.7 - 99.5°F**
Fever / Hyperthermia	>37.5 or 38.3°C	99.5 or 100.9°F
Hyperpyrexia	>40.0 or 41.5°C	104.0 or 106.7°F

Table 4.1.3 temperature range

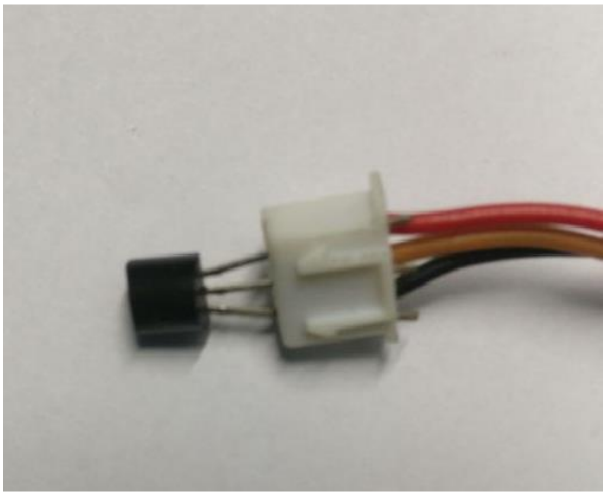

Fig 4.1.5 Temperature sensor

APPLICATIONS

- Power Supplies
- Battery Management
- HVAC
- Appliances

4.2. Software Requirement:

The software used for this project are

- Android studio
- Arduino Uno
- Php mysql

4.2.1 Android OS version 4.0 and above:

Android is a mobile operating system developed by Google, based on the Linux kernel and designed primarily for touchscreen mobile devices such as smartphones and tablets. Android's user interface is mainly based on direct manipulation, using touch gestures that loosely correspond to real-world actions, such as swiping, tapping and pinching, to manipulate on-screen objects, along with a virtual keyboard for text input.

Features:

Applications ("apps"), which extend the functionality of devices, are written using the Android software development kit (SDK) and, often, the Java programming language.Java may be combined with C/C++, together with a choice of non-default runtimes that allow better C++ support. The Go programming language is also supported, although with a limited set of application programming interfaces (API).

The SDK includes a comprehensive set of development tools, including a debugger, software libraries, a handset emulator based on QEMU, documentation, sample code, and tutorials. Initially, Google's supported integrated development environment (IDE) was Eclipse using the Android Development Tools (ADT) plugin; Google released Android Studio, as its primary IDE for Android application development. Other development tools are available, including a native development kit(NDK) for applications or extensions in C or C.

4.2.2 Arduino Uno

Arduino/Genuino Uno is a microcontroller board based on the ATmega328P (datasheet). 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.

"Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform; for an extensive list of current, past or outdated boards see the Arduino index of boards.

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software. This software can be used with any Arduino board.

Arduino Uno is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. 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. Arduino Uno has a number of facilities for communicating with a computer, another Arduino board, or other microcontrollers.

A minimal Arduino C/C++ sketch, as seen by the Arduino IDE programmer, consist of only two functions:

- setup: This function is called once when a sketch starts after power-up or reset. It is used to initialize variables, input and output pin modes, and other libraries needed in the sketch.
- loop: After setup has been called, function loop is executed repeatedly in the main program. It controls the board until the board is powered off or is reset.

Features:

- Operating Voltage: 5V
- Input Voltage (recommended): 7-12V
- Input Voltage (limits): 6-20V
- Digital I/O Pins: 14 (of which 6 provide PWM output)
- Analog Input Pins: 6
- DC Current per I/O Pin: 40 mA
- DC Current for 3.3V Pin: 50 mA
- Flash Memory: 32 KB of which 0.5 KB used by bootloader
- SRAM: 2 KB (ATmega328)
- EEPROM: 1 KB (ATmega328)
- Clock Speed: 16 MHz

APPLICATIONS

- Real time biometrics
- Robotic applications
- Academic applications

4.2.3 Php mysql

PHP is the most popular scripting language for web development. It is free, open source and server-side (the code is executed on the server). MySQL is a Relational Database Management System (RDBMS) that uses Structured Query Language (SQL).

It is also free and open source. The combination of PHP and MySQL gives unmet options to create just about any kind of website - from small contact form to large corporate portal. The environment is written in Java and based on Processing and other open-source software.

This PHP/MySQL tutorial is part of the rich collection of web hosting tutorials which Site Ground experts have prepared for you. Find out how to start your website, how to promote it, how to deal with c Panel, webmail and FTP.

Features :

- Cross-platform support
- Stored procedures, using a procedural language that closely adheres to SQL/PSM
- Triggers
- Embedded database library
- Cursors
- Updatable views

4.2.4 JSON

JSON (JavaScript Object Notation) is a minimal, readable format for structuring data. It is used primarily to transmit data between a server and web application, as an alternative to XML. JavaScript Object Notation (JSON) is a lightweight, text-based, language-independent data interchange format. It was derived from the ECMAScript Programming Language Standard. JSON defines a small set of formatting rules for the portable representation of structured data. It was derived from JavaScript, but as of 2017 many programming languages include code to generate and parse JSON-format data.

Features:

- If you want the JSON representation in the output file to be more human readable, you can enable formatting.
- The JSON will be formatted with spaces, tabs, and carriage returns to improve its readability.
- A formatted JSON may be beneficial for application development and testing. However, it is not recommended for actual production applications since all of the whitespace is unnecessary and ignored by JSON parsers.
- The environment is written in Java and based on Processing and other open-source software.
- Additionally, formatted JSON can be significantly larger than its JSON equivalent and the file size will be greater than its corresponding JSON representation. This can affect application performances.
- It is also free and open source
- The conversion does not support joins, relates, and attachments of the features.
- To convert a subset of features in a feature class or layer, use the Select Layer By Attribute or Select Layer By Location tools and select the subset of features to be converted before using the Features To JSON conversion tool.

5 IMPLEMENTATION

5.1 Sample Codes

5.1.1 Description

This description tab is mainly used for doctors. They can mention some necessary details about the patient in the application itself. They can add remainder for the patient to take x-ray reports. The patient current scenario can also noted in the description tab. These data can be used for future references in case of maintain the report.

Code:

```
package com.example.hivestudios.hms;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import com.android.volley.Request;
import com.android.volley.Response;
import com.android.volley.toolbox.Volley;
import java.util.HashMap;
import java.util.Map;

public class Description extends AppCompatActivity {
    EditText editText;
    Button button;
    String desc;
    String id;
    private static final String URL =
"http://192.168.43.196/hms/description.php";
```

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_description);
        editText = (EditText) findViewById(R.id.desc);
        button = (Button)findViewById(R.id.save);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                storeDesc();
            }
        });
    }

    private void storeDesc() {
        StringRequest stringRequest = new
StringRequest(Request.Method.POST, URL,
                new Response.Listener<String>() {
                    @Override
                    public void onResponse(String response) {
                        System.out.println(response);
                    }
                },
                new Response.ErrorListener() {
                    @Override
                    public void onErrorResponse(VolleyError error) {
                        Toast.makeText(Description.this, error.toString(),
Toast.LENGTH_LONG).show();
```

```
                }
            }) {
        @Override
        protected Map<String, String> getParams() {
            Map<String, String> params = new HashMap<String, String>();
            params.put("desc", desc);
            params.put("id", id);
            return params;
        }
    };
    RequestQueue requestQueue = Volley.newRequestQueue(this);
    requestQueue.add(stringRequest);
  }
}
```

5.1.2 Login page

In the case ,if the doctor is already registered with the application he/she may not required to register with the application so that they can directly access the patient details. Once the doctor loin with the application he/she can access it from the main page if he again open the application after closed it.

Code

```
package com.example.hivestudios.hms;
import android.app.ProgressDialog;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
```

```
import android.widget.EditText;
import com.android.volley.Request;
import com.android.volley.Response;
import org.json.JSONException;
import org.json.JSONObject;
import java.util.HashMap;
import java.util.Map;
public class Login_Doctor extends AppCompatActivity {
    private static final String TAG = "LoginActivity";
    private static final String URL_FOR_LOGIN = 
"http://192.168.0.3/hms/login_doc.php";
    ProgressDialog progressDialog;
    private EditText loginInputNum, loginInputPassword;
    private Button btnlogin;
    private Button btnLinkSignup;
    SessionManager sessionManager;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login__doctor);
        loginInputNum = (EditText) findViewById(R.id.login_input_num);
        loginInputPassword = (EditText) 
findViewById(R.id.login_input_password);
        btnlogin = (Button) findViewById(R.id.btn_login);
        btnLinkSignup = (Button) findViewById(R.id.btn_link_signup);
        sessionManager = new SessionManager(getApplicationContext());
        if(sessionManager.isLoggedIn()){
            Intent intent = new Intent(this, SplashScreen.class);
            startActivity(intent);
```

```
        }
        // Progress dialog
        progressDialog = new ProgressDialog(this);
        progressDialog.setCancelable(false);
        btnlogin.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                loginUser(loginInputNum.getText().toString(),
                        loginInputPassword.getText().toString());
            }
        });
        btnLinkSignup.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent i = new Intent(getApplicationContext(),
Register_doctor.class);
                startActivity(i);
            }  });
    }
    private void loginUser( final String num, final String password) {
        String cancel_req_tag = "login";
        progressDialog.setMessage("Logging you in...");
        showDialog();
        StringRequest strReq = new StringRequest(Request.Method.POST,
                URL_FOR_LOGIN, new Response.Listener<String>() {
            @Override
            public void onResponse(String response) {
                Log.d(TAG, "Register Response: " + response.toString());
                hideDialog();
```

```
            try {
                JSONObject jObj = new JSONObject(response);
                boolean error = jObj.getBoolean("error");

                if (!error) {
                    String user = jObj.getJSONObject("user").getString("name");
                    sessionManager.createLoginSession(num);
                            Intent intent = new Intent(
                            Login_Doctor.this,
                            SplashScreen.class);
                    intent.putExtra("number", num);
                    startActivity(intent);
                    finish();
                } else {
                    String errorMsg = jObj.getString("error_msg");
                    Toast.makeText(getApplicationContext(),
                            errorMsg, Toast.LENGTH_LONG).show();
                }
            } catch (JSONException e) {
                e.printStackTrace();
            }
        }
    }, new Response.ErrorListener() {

        @Override
        public void onErrorResponse(VolleyError error) {
            Log.e(TAG, "Login Error: " + error.getMessage());
            Toast.makeText(getApplicationContext(),
                    error.getMessage(), Toast.LENGTH_LONG).show();
```

```
                hideDialog();
            }
        }) {
            @Override
            protected Map<String, String> getParams() {
                // Posting params to login url
                Map<String, String> params = new HashMap<String, String>();
                params.put("num", num);
                params.put("password", password);
                return params;
            }

        };
        // Adding request to request queue

    AppSingleton.getInstance(getApplicationContext()).addToRequestQueue(strR
    eq,cancel_req_tag);
    }

    private void showDialog() {
        if (!progressDialog.isShowing())
            progressDialog.show();
    }
    private void hideDialog() {
        if (progressDialog.isShowing())
            progressDialog.dismiss();
    }
}
```

5.1.3 Arduino Code

```
int temp = A1;
int ecgs = A3;
const int heartBeat = A2;
int tempvalue;
int heartBeatValue = 0;
int HB=0;
//int p,q,r,s,t,u,v;
float ecgsensorValue;
char val;



void setup() {
Serial.begin(9600);
//Serial.begin(9600);
pinMode(A1, INPUT);
pinMode(A2, INPUT);
pinMode(A3, INPUT);

}
void loop() {
heartBeatValue = (analogRead(heartBeat));
if(heartBeatValue<255)
        {
         HB=0;
int x;
for(x=0;x<110;x++)
           {
```

```
heartBeatValue = (analogRead(heartBeat));
            //Serial.println(heartBeatValue);
            {
if(heartBeatValue<255)
            {
             HB=HB+1;
heartBeatValue = (analogRead(heartBeat));
             //Serial.println(heartBeatValue);
            }
else
            {
             HB=HB;
heartBeatValue = (analogRead(heartBeat));
             //Serial.println(heartBeatValue);
            }
delay(100);delay(100);delay(100);
          }
Serial.println("Keep Finger");
   }
Serial.println("Remove Finger");
Serial.print("HeartBeat:");
Serial.println(HB);
delay(20);
delay(100);
delay(2000);
}
for(int i=0; i<10; i++){
tempvalue = analogRead(A1);
float mv = (tempvalue/1024.0)*5000;
```

```
float cel = mv/10;
        //float farh = (cel*9)/5 + 32;
Serial.print("B=");
Serial.println(cel);
delay(1000);
    }
for(int i=0;i<=60;i++){
ecgsensorValue = analogRead(A1);
Serial.print("A=");
Serial.println(ecgsensorValue);
delay(1000);
        }
}
```

5.2 Sample Screen shots

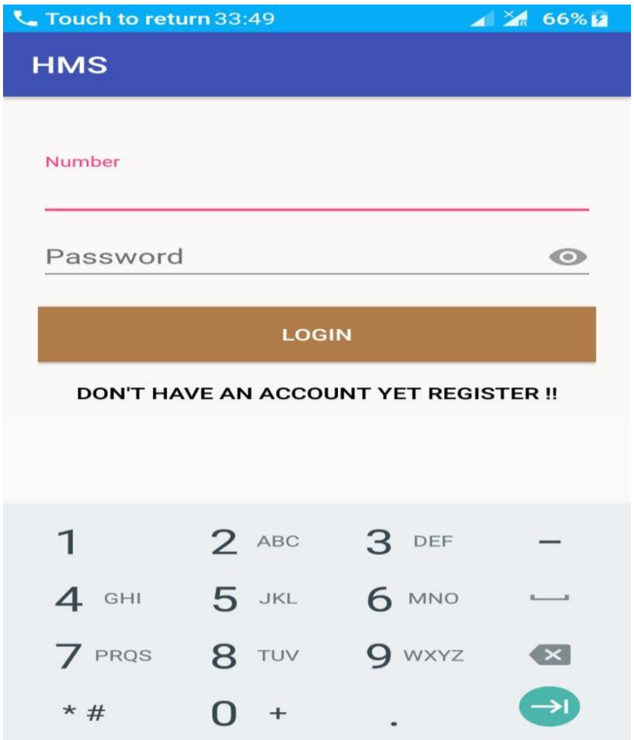

Fig 5.2.1 Screenshot for doctor login

Fig 5.2.2 Screenshot for patient id

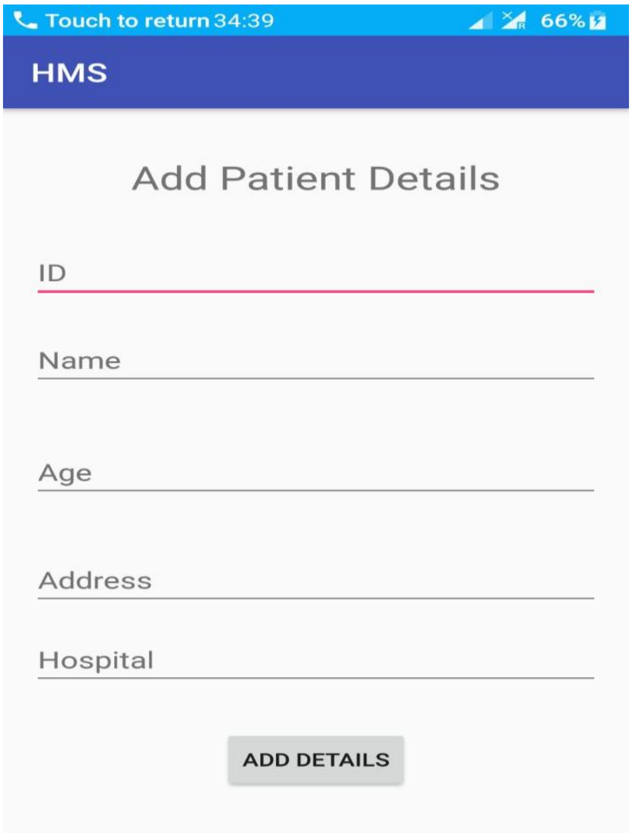

Fig 5.2.3 Screenshot for adding new patient details

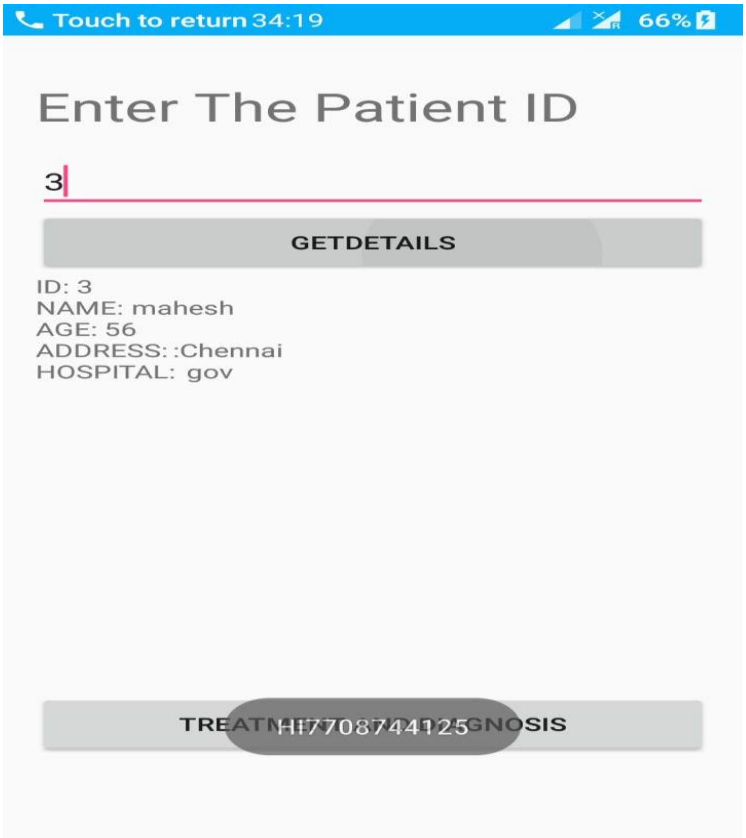

Fig 5.2.4 Screenshot of already registered patient

Fig 5.2.5 Screenshot for connecting with devices

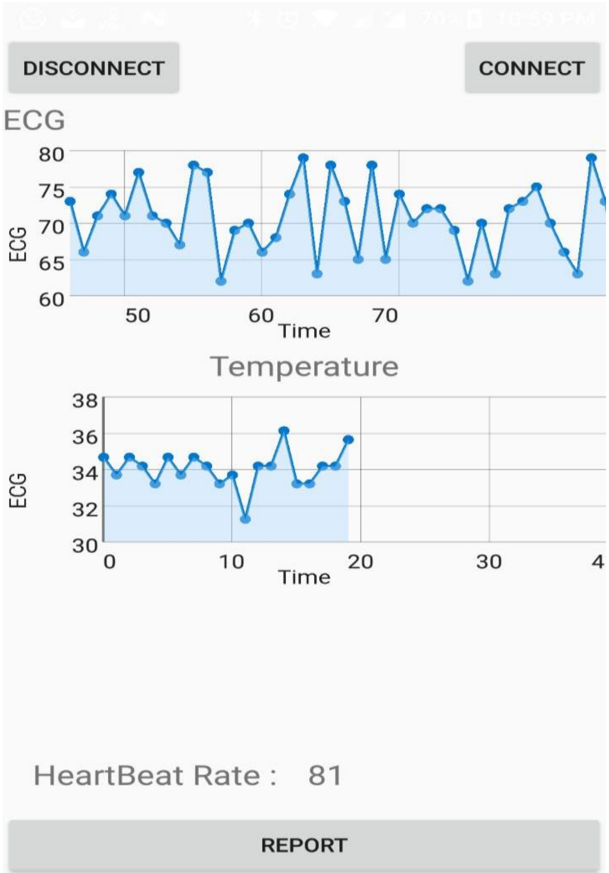

Fig 5.2.6 Screenshot of patient health parameter

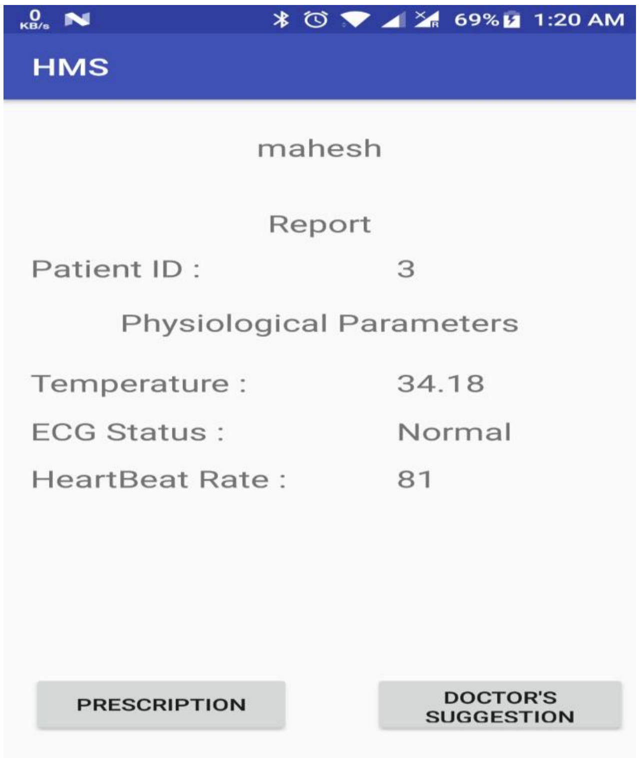

Fig 5.2.7 Screenshot for patient report

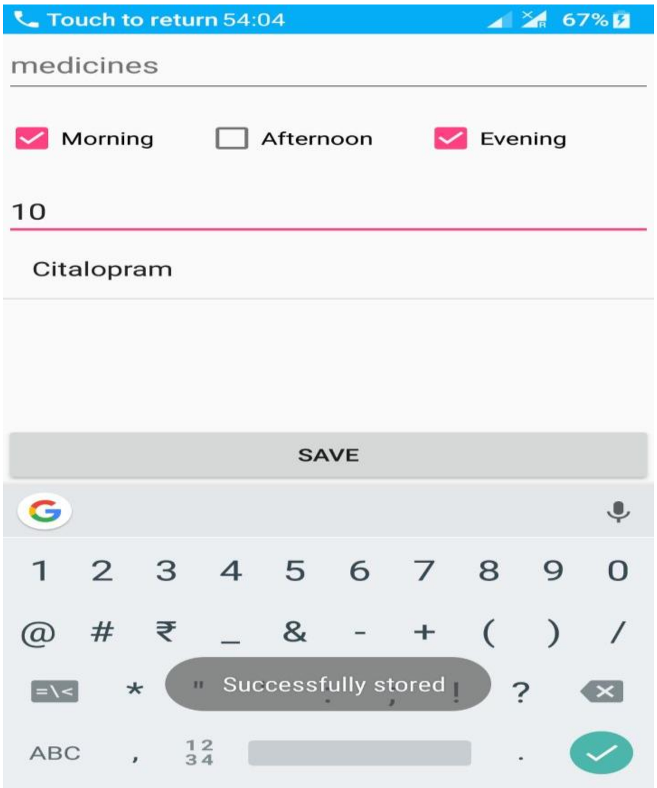

Fig 5.2.8 Screenshot for prescription

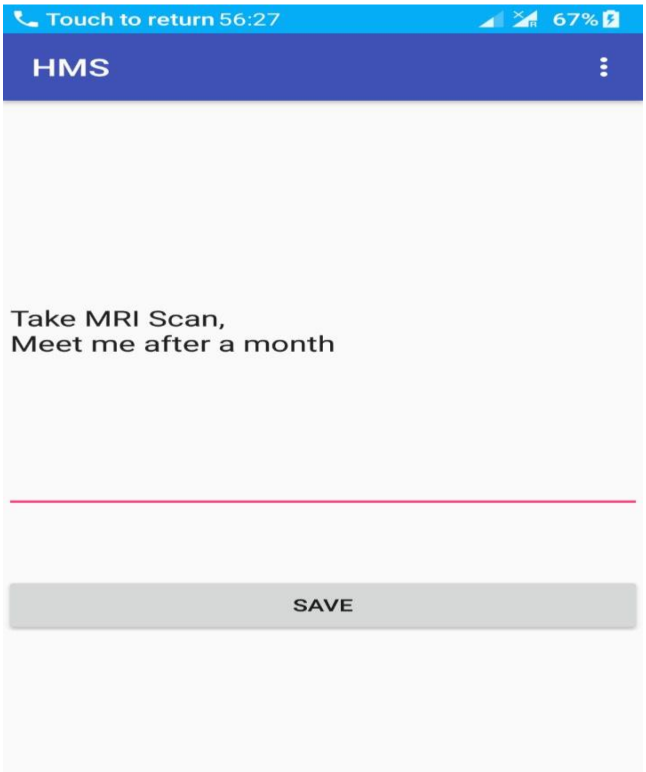

Fig 5.2.9 Screenshot for description

6 TESTING AND MAINTENANCE

Generate test cases:

Modules name	Input	condition	Expected output	Obtained output	Status
ECG	sensor will measure the ecg data from human body through ecg module.	Normal ecg value for the patient should be from 60 to 100. If not patient will go to abnormal state.	The patient ecg value measured through sensor should be in normal state.	The patient ecg value is normal as expected.	Pass
Pulse rate	When the patient kept his finger near the led light ,the sensor will start reading the pulse count .	For the age 18-25,the pulse count should be around 61-65 then only we can tell that patient health is excellent.	The pulse count should be excellent to the patient whose age is 18	The pulse count is excellent to the patient .	Pass
Temperature	Patient touches the sensor, through the body temperature it start taking the reading.	When the temperature is between 36.5-37.5°C ,then the patient is said to be normal.	The patient body temperature should be in normal condition.	Temperature is normal	Pass

Table 6.1 Test cases

6.1 Black box testing:

Black-box testing is a method of software testing that examines the functionality of an application without peering into its internal structures or workings. This method of test can be applied virtually to every level of software testing: unit, integration, system and acceptance. Specific knowledge of the application's code/internal structure and programming knowledge in general is not required. The tester is aware of what the software is supposed to do but is not aware of how it does it. For instance, the tester is aware that a particular input returns a certain, invariable output but is not aware of how the software produces the output in the first place. It is also known as Behavioral Testing, is a software testing method in which the internal structure/ design/ implementation of the item being tested is not known to the tester. These tests can be functional or non-functional, though usually functional.

6.2White box Testing:

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality (i.e. black box testing). In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs. This is analogous to testing nodes in a circuit. White-box testing can be applied at the unit, integration and system levels of the software testing process. Although traditional testers tended to think of white-box testing as being done at the unit level, it is used for integration and system testing more frequently today. It can test paths within a unit, paths between units during integration, and between subsystems during a system–level test. Though this method of test design can uncover many errors or problems, it has the potential to miss unimplemented parts of the specification or missing requirements.

6.3 Unit testing:

A unit is the smallest testable part of software. It usually has one or a few inputs and usually a single output. In procedural programming a unit may be an individual program, function, procedure, etc. In object-oriented programming, the smallest unit is a method, which may belong to a base/ super class, abstract class or derived/ child class. (Some treat a module of an application as a unit. This is to be discouraged as there will probably be many individual units within that module.

6.4 Integration testing:

Integration testing is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing. The purpose of integration testing is to verify functional, performance, and reliability requirements placed on major design items.

6.5 System Testing:

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black-box testing, and as such, should require no knowledge of the inner design of the code or logic.

As a rule, system testing takes, as its input, all of the "integrated" software components that have passed integration testing and also the software system itself integrated with any applicable hardware system(s). The purpose of integration testing is to detect any inconsistencies between the software units that are integrated together. System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole.

6.6 Acceptance Testing:

Acceptance Testing is a level of the software testing where a system is tested for acceptability. The purpose of this test is to evaluate the system's compliance with the business requirements and assess whether it is acceptable for delivery. It is also stated that the Formal testing with respect to user needs, requirements, and business processes conducted to determine whether or not a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system**.** Acceptance Testing is a level of the software testing where a system is tested for acceptability. Definition by ISTQB. The **Factory Acceptance Test** (FAT) is a major project milestone in a laboratory automation project where the equipment and/or system integrator demonstrates that the system design and manufacturing meets the contract or Purchase Order (P.O.).

Acceptance testing, a testing technique performed to determine whether or not the software system has met the requirement specifications. The main purpose of this test is to evaluate the system's compliance with the business requirements and verify if it is has met the required criteria for delivery to end users.

There are various forms of acceptance testing:

- User acceptance Testing
- Business acceptance Testing
- Alpha Testing
- Beta Testing

The acceptance test activities are designed to reach at one of the conclusions:

1. Accept the system as delivered
2. Accept the system after the requested modifications have been made
3. Do not accept the system

7 CONCLUSION AND FUTURE ENHANCEMENT:

A smart phone based health monitoring system has been presented in this work. WBAN promise inexpensive, unobtrusive, and unsupervised ambulatory monitoring during normal daily activities for prolonged periods of time. To make this technology ubiquitous and affordable, a number of challenging issues should be resolved, such as system design, configuration and customization, seamless integration, standardization, further utilization of common off-the-shelf components, security and privacy, and social issues. By using the system the healthcare professionals can monitor, diagnose, and advice their patients all the time. The physiological data are stored and published online. Hence, the healthcare professional can monitor their patients from a remote location at any time. Our system is simple. The arduino board will receive serial input data from the respective sensor that was attached in the arduino board. The data will be viewed in the graphical format. Each data will be compared with our rules and come to conclude that patient is normal or abnormal. The description of the patient will in stored in the database so that doctor can be able to have an idea about the patient when he\she came for next visit to consult the doctor. The prescription tab is used to give medicines list to the patient and stored in database for future references. The heart rate will be viewed in text view because the sensor will detect our heart rate when we keep the finger near the sensor and after 60 second a toast message will be displayed that remove finger and then output will be printed in the text view. It is just few wires connected to a small kit with a smart phone. The system is very power efficient. Only the smart phone or the tablet needs to be charged enough to do the test. It is easy to use, fast, accurate, high efficiency, and safe (without any danger of electric shocks). As a future enhancement, we are going to implement this same system using Beacon sensor which may have larger functionalities than this system. If the Beacon is used then the data transmission range will be little higher.

REFERENCE

[1] "Wireless Body Area Networks: Applications and Technologies" By Abdelfettah Belghitha HANAlab, Imen Jemilia , Rim Negraa , University of Manouba, Tunisia - College of Computer and Information Sciences, King Saud University, Saudi Arabia

[2] "Implementation of a Wireless Body Area Network for Healthcare Monitoring" By Aime V. Mbakop, Ashenafi Lambebo , Lalindra Jayatilleke and Sasan Haghani - University of the District of Columbia 4200 Connecticut Ave, Washington, DC,

[3] "Real time wireless health monitoring application using mobile devices" By Aisha Rashid, Amna Abdullah, Ali Abou-ElNour, Asma Ismael, and Mohammed Tarique - Department of Electrical Engineering, Ajman University of Science and Technology, P.O. Box 2202, Fujairah, United Arab Emirates

[4] "The Health Care Monitoring Using Android Mobile Phone" By Balaji Devkatta , Pankaja Jadhav , Vidya Dhumal - Department of Electronics Engineering, Savitribai Phule University of Pune, PDEA's College Of Engineering, Pune, India

[5] "A Wireless Health Monitoring System based on Android Operating System" By Dongdong Loua,b, Huan Jina,b, Xianxiang Chena ,Xingzu Guo a,b,, Yundong Xuana,b, Zhan Zhaoa , Zhen Fanga, Zhihong Xua,b, - a Institute of Electronics, Chinese Academy of Sciences, BeiJing,100190,China b University of Chinese Academy of Sciences, BeiJing,100049,China

[6] "A Survey on Wireless Body Area Network" Emmanuel Davies, Kola Sanjay, J.Mohana - Department of Electronics and Communication Engineering, Saveetha School of Engineering, Saveetha University, Chennai, India

[7] "Android Based Heart Monitoring and Reporting System" By Mayuri Navale , Rameshwari Chavan , Renuka Dube ,Sandhya Damare, Prof. Suhas Patil – Computer Engineering, KJCOEMR, Pune, India