CARDI WEAR - A WEARABLE ECG DEVICE FOR REMOTE MONITORING

By

Muhammad Haseeb Rafi

Menahil Naeem

Muhammad Hashim

Usama Waheed

Supervisor Name: Dr. Mir Yasir Umair

Submitted to the Faculty of Department of Electrical Engineering, Military College of Signals (MCS), National University of Sciences and Technology, Islamabad in partial fulfillment for the requirements of a B.E Degree in Electrical Telecom Engineering

JUNE 2019

CERTIFICATE OF CORRECTNESS AND APPROVAL

It is hereby certified that information contained in this thesis titled “Cardi Wear - A Wearable ECG Device For Remote Monitoring” carried out by 1) Muhammad Haseeb Rafi 2) Menahil Naeem 3) Muhammad Hashim 4) Usama Waheed under the vigilant supervision of Assistant Professor Dr. Mir Yasir Umair in partial fulfillment for Degree of B.E in Electrical Telecom Engineering, in Military College of Signals (MCS), National University of Sciences and Technology is correct and approved. The Plagiarism is 16%.

Approved By

____

Asst. Prof. Dr. Mir Yasir Umair

Project Supervisor

Department of Electrical Engineering

Military College Of Signals, NUST

ABSTRACT

CARDI WEAR - A WEARABLE ECG DEVICE FOR REMOTE MONITORING

According to the World Health Organization, Cardiovascular Diseases (CVDs) are the primary cause of death globally: more people die annually from CVDs than from any other reason. [1] Pakistan's population has one of the highest rated risks of Coronary Heart Disease (CHD) in the world. In Pakistan, around 30% to 40% of all deaths are due to cardiovascular diseases (CVD). The (CHD) related deaths in Pakistan have reached about 200,000 per year that is 410/100,000 of the population including children. [2]

By observing the statistics, it seems like an essential deduction that persistent monitoring should be accessible to the people with heart diseases at all times. ECG instruments used in hospitals are bulky and patients often experience allergic reaction as a result of using gel electrodes.

We intend to innovate, design and create an indigenous low cost electrocardiogram (ECG) device which make use of gel free electrodes (stainless steel dry electrodes) and can be easily used by someone with none to low technical expertise for round-the-clock ECG self-monitoring and feedback.

Our project consists of hardware and software components for a wireless (ECG) device. In addition, an Application has also been developed based on Android technology which provides an interface for the user in order to display as well as record the user's results. Real-time ECG results can also be backed up on Cloud for storage and remote monitoring by the Physician.

The Scope of this project is the incorporation of Bluetooth wireless communication in medical applications for home healthcare thereby enabling patients to experience the freedom of mobility.

In the name of Allah, most gracious and most merciful

DEDICATION

This composition is endearingly dedicated to our respectable parents, teachers and all well-wishers. It could not have been possible without their generous collaboration and support.

ACKNOWLEDGEMENTS

All our gratitude is extended to our Creator Allah Subhanahu Wa Ta'ala, whose blessings empowered us to tread this journey. We dedicate all our work to the Him and thank Him for the guidance, strength and power of mind as well as bestowing us with the required skills.

We would like to thank Asst. Prof. Dr. Mir Yasir who supervised the project in a very motivating and cooperative manner. As a supervisor, his support and guidance had always been a valuable resource for our project and due to his instructions we were able to convert the idea of this project into reality.

Last but not the least special acknowledgement to all the members of this group who tolerated each other throughout the course of this project.

TABLE OF CONTENTS

List of Figures

List of Tables

CHAPTER: 01

Introduction

1.1. Introduction

This section provides a brief overview of the project. It also includes problem statement, objectives and scope and the motivation behind the project.

1.1.1. ECG

An ECG (Electrocardiogram) machine is a diagnostic tool or is appraised as one of the simplest and oldest cardiac examination methods available, which can provide a fair amount of useful information regarding the heart's activity and is considered as an essential part of the evaluation of cardiac patients. An ECG is a simple representation of electrical activity of the heart muscles as it changes with time and it is usually printed on paper for easier analysis. Like other muscles, cardiac muscle contracts in response to electrical depolarisation of the muscle cells. It is the sum of this electrical activity, when amplified and recorded for just a few seconds that we know as an ECG. [3]

1.1.2. Working of ECG

An ECG is a graphical representation of the electrical activity throughout the heart when it beats. With each beat, an electrical impulse (this impulse is the result of neurons firing) traverses throughout the heart. This electrical activity or wave causes the muscle to contract and pump blood from the heart. A regular heartbeat on ECG will show the timing of the top and lower chambers. Both right and left atria or upper chambers make the first wave called a "P wave" - following a flat line when the electrical impulse goes to the bottom chambers. The right and left bottom chambers or ventricles make the next wave called a "QRS complex". The final wave or "T wave" represents electrical recovery or return to a resting state for the ventricles. [4]

An ECG reveals two principal forms of data. Firstly, by timing the intervals on the ECG, a physician can ascertain the time it takes the electrical wave to traverse through the heart. This time period thereby indicates whether the electrical activity is normal or slow, fast or irregular. Secondly, by gauging the amount of electrical activity pulsing through the heart muscles, a cardiologist may be able to discover if parts of the heart are too large or are too overworked.

1.2. Project Overview

We aim to design and fabricate a low cost electrocardiogram (ECG) device which can be easily used by someone with low technical proficiency for continuous ECG self-monitoring and feedback.

The aforementioned product is a compact portable device which is ergonomic. The patient's mobility is not limited in any way as the device is extremely light-weight and user friendly. The project comprises of software and hardware components for a wireless (ECG) device, and a corresponding Application has also been developed based on Android technology which displays and records the user's heart signal.

1.3. Problem Statement

Cardiovascular disease is the most common cause of natural death the world over, accounting for 29.2 percent of total deaths. [5] The charted prevalence of coronary heart disease (CHD) in adult surveys has risen four-folds in urban areas and doubled in rural areas over the past 30 years. Studies indicate that approximately 40% men and 25% of women strongly agreed that heart attack is a major problem. [6] The ECG machine is a diagnostic instrument which records the electrical activity of the heart from multiple angles to identify and locate pathology, rendering valuable information of cardiac disorders. ECG instruments used in hospitals are bulky and make use of gel electrodes which causes discomfort to some individuals. It works on line voltage which makes the ECG waveform distorted and an isolation amplifier is needed for patient safety making it expensive; thus, it cannot be used in underdeveloped areas.

We hereby offer a solution by constructing a low cost, compact and handy alternative to these conventional ECG machines available in clinics. By using gel free electrodes we eliminate the complications of allergic reactions to certain people.

1.4. Objectives

The objectives of Cardi wear are:

- Making a low cost, compact as well as an equally effective wearable alternative to conventional ECG machines.
- Making ECG results accessible by providing a mobile display of ones heartbeat.
- Using gel free electrodes for added comfort and ergonomics.
- Real-time ECG results will be backed up to Cloud for storage and monitoring.

1.5. Scope of Project

- The framework for this project is the integration of wireless communication in medical applications for home healthcare.
- Patients would be no longer bound to a specific healthcare location where they are monitored by medical instruments.
- Wireless communication will not only provide them with safe and accurate monitoring, but also the freedom of movement.

1.6. Motivation and Need

Main motivation behind taking this project is to make a product that can help those patients with cardiac disorders that need constant monitoring like ones having abnormal heart rhythms or arrhythmias, Congenital Heart Disease patients and many other progressive diseases related to heart. As they need constant monitoring it becomes troublesome for them to frequently visit healthcare center. So an alternative is needed in order to resolve this problem. For that we propose a solution which gives similar results and is also reasonably priced.

CHAPTER: 02

Literature Review

2.1. Currently Existing Alternatives

This Section deals with the background study as well as the literature review carried out for design and development of Cardi Wear. It also mentions the similar projects done at MCS and ends with the organization of the entire document.

2.1.1. ECG Machines

ECG Machines are used to record and analyze the electrical activity of the heart from various angles to identify and detect pathology. The overall design consists of electrodes which are positioned on multiple locations over limbs and chest to record the electrical activity. It records the electrical activity produced by heart muscle depolarisations, which propagate in pulsating electrical waves towards the epidermal layer. Although the strength of electrical signal is not too prominent, it can be detected reliably with ECG electrodes attached to the skin (in microvolts, or µV).

The full ECG setup contains at least four electrodes which are placed on the chest region or at the four lower and upper body extremities according to standard nomenclature (right arm, left arm, right leg, and left leg). Undeniably, variants of this setup exist like in a 12-lead ECG, there are 12 leads calculated using 10 electrodes in order to allow more flexible and less invasive recordings. ECG electrodes are usually wet sensors, requiring the use of a conductive gel to increase conduction between skin and electrodes.

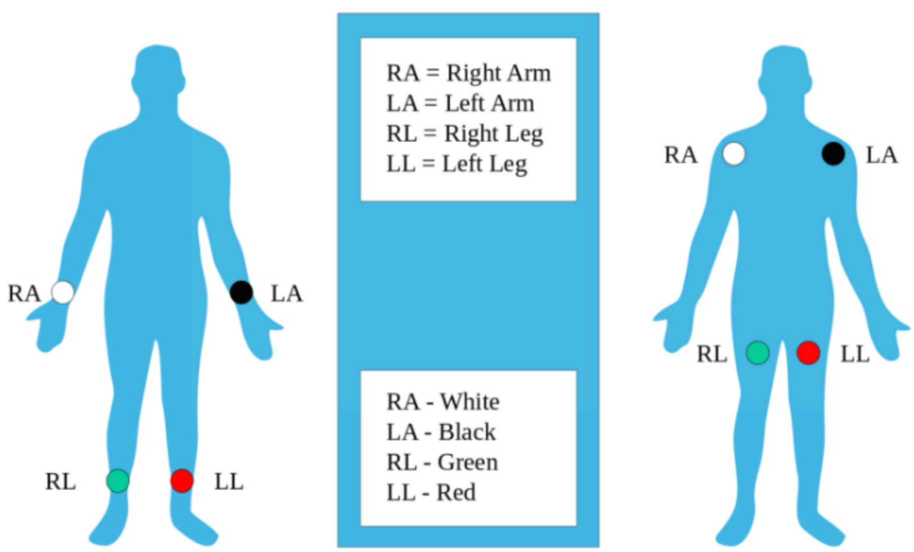

Figure 1 - Limb (Extremity) Electrodes and Placement

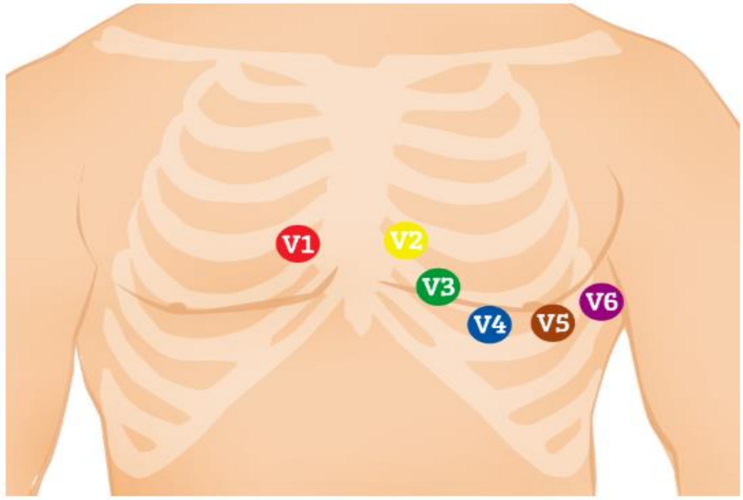

Figure 2 - Chest (Precordial) Electrodes and Placement

» V1 - Fourth intercostal space on the right sternum

» V2 - Fourth intercostal space at the left sternum

» V3 - Midway between placement of V2 and V4

» V4 - Fifth intercostal space at the midclavicular line

» V5 - Anterior axillary line on the same horizontal level as V4

» V6 - Mid-axillary line on the same horizontal level as V4 and V5

A typical ECG tracing of a normal heartbeat (or cardiac cycle) consists of a P wave, a QRS complex and a T wave. A small U wave is typically observable in 50 to 75% of ECGs. The baseline voltage of the electrocardiogram is known as the isoelectric line. Normally the isoelectric line is measured as the portion of the mapping following the T wave and former to the next P wave

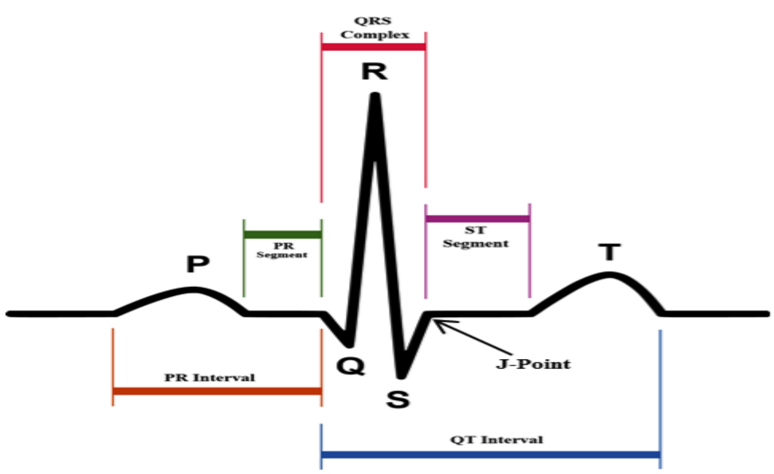

Figure 3 - Waves and Intervals [7]

❖ P Wave

The P wave represents the depolarization of the left and right atrium and also relates to atrial contraction. In reality, the atria contract a split second after the P wave initiates. Since it is fairly minute, atrial repolarization is usually not discernable on ECG. Commonly, the P wave will be smooth and rounded.

❖ QRS Complex

As the name suggests, the QRS complex includes the Q wave, R wave, and S wave. These three waves occur in rapid succession. The QRS complex represents the electrical impulse as it spreads through the ventricles and indicates ventricular depolarization. As with the P wave, the QRS complex starts just before ventricular contraction. It is important to recognize that not every QRS complex will contain Q, R, and S waves. The convention is that the Q wave is always negative and that the R wave is the first positive wave of the complex. If the QRS complex only includes an upward (positive) deflection, then it is an R wave. The S wave is the first negative deflection after an R wave. The J-point is the point where the QRS complex and the ST segment meet. It can also be thought of as the start of the ST segment. The J-point (also known as Junction) is important because it can be used to diagnose an ST segment elevation myocarial infarction.

❖ T Wave

A T wave follows the QRS complex and indicates ventricular repolarization. Unlike a P wave, a normal T wave is slightly asymmetric; the peak of the wave is a little closer to its end than to its beginning. When a T wave occurs in the opposite direction of the QRS complex, it generally reflects some sort of cardiac pathology. [7]

2.1.2. Holter Monitor

Another somewhat similar product found in the market is Holter monitor which basically is a battery-operated portable device that measures and records your heart's activity (ECG) continuously for 24 to 48 hours or longer depending on the requirements and the type of monitoring device used. The device is the size of a small camera. It has wires attached with silver dollar-sized electrodes that stick to your skin. Because the electrodes are attached with tape or adhesives, they may cause mild skin irritation. Holter monitor do provide a broader picture of heart activity necessary for the analysis of cardiac conditions; there are still quite a few disadvantages. Holter monitors are typically found to be incommodious and uncomfortable, requiring skin preparation, patches, and wires. These Holter monitors are basically intended to be used over longer periods or to test for off-site surroundings such as daily routine or specific triggers.

Figure 4 - Holter Monitor

2.2. Literature Review

- Tai Le proposed the development of a low cost mobile ECG monitoring device, which early warms abnormalities throughout long-term ECG observation. For user convenience, two active dry electrodes are used as touch sensors. A buffer circuit is applied to remove the over-sensitivity effect of the electrodes. Test results demonstrated that ECG quality when utilizing dynamic dry electrodes is identical to that of wet Ag/AgCl electrodes application. [8]

- A.S. Joutsen recommends in this paper, we study different size stainless steel dry electrodes in ECG and heart rate monitoring and compare those with commercial disposable Ag/AgCl electrodes. The results show that stainless steel dry electrodes performed well throughout the tested activities if the circular electrode diameter was 20 mm or larger. [9]

- Researchers from the Department of embedded systems at the School of Engineering, Jönköping University mention that the challenge of a wireless communication unit is to send as little information as possible to make the communication faster, without loss of information in the ECG-signal. [10]

- Yishan Wang insists that in order to make the sensor more wearable and comfortable for patients, a new lead system is studied and discussed to find the new convenience placements for electrodes. Mason-Likar limb electrode placement is considered as standard lead system. This paper provides powerful evidence that the traditional electrode placements can be replaced by new placements which make the ECG system more convenience and compact. [11]

2.3. Similar Projects done at MCS

A somewhat similar project was done in MCS by students of TE-50 with the title "Wireless HCI system for disabled persons" in which results were taken using different sensors and the signals were sent to the microcontroller for processing and do required operations. Wireless technology included Bluetooth module (HC-05) that was used to send signals to computer.

2.4. Organization of Document

Chapter 1 Presents essential details and overview of the Project.

Chapter 2 Deals with the literature review carried out for the Project and mentions similar projects done in MCS.

Chapter 3 Discusses the hardware and software requirements of the Project.

Chapter 4 Analysis and Evaluation of Results.

Chapter 5 Conclusion.

At the end of the document, the financial estimates, project timeline and appendix giving additional relevant details about Cardi Wear are specified.

CHAPTER: 03

Design and Development

This Section provides a complete technical picture of Cardi Wear by describing its technical specifications, hardware and software requirements and its complete design.

3.1. Hardware Requirements

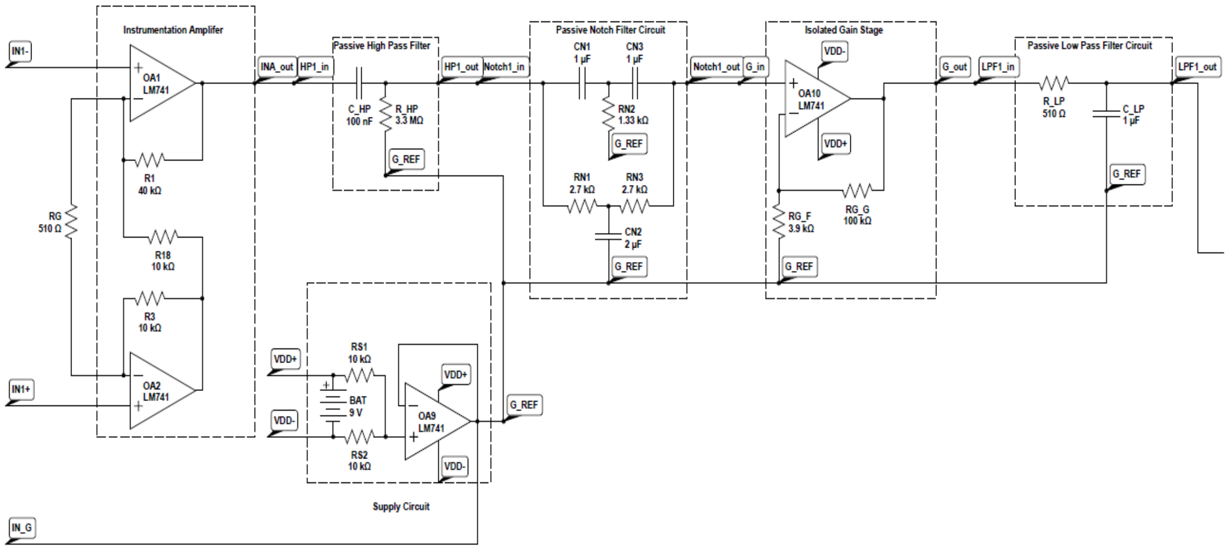

Figure 5 - Complete Hardware Circuit [12]

A sequence of amplifiers and filters are used to construct the hardware circuit. The hardware component of the project has 8 main components: electrodes, the leads, an initial amplification stage, high-pass filter, notch filter, secondary amplification stage, low-pass filter and the analog to digital converter (ADC).

3.1.1. Electrodes

Our ultimate design comprises of the strategy of using a three electrode system which is the minimum number of electrodes that can be employed to recuperate a signal from the heart. This arrangement was chosen for two main reasons. First, reducing the number of leads to a minimum makes the device less intrusive and more appropriate for sustained use. Secondly, choosing a 3 electrode system makes the hardware and software design considerably simpler and economical.

Since only one signal is measured, only one set of amplifiers, filters and analog to digital converters are required. As well, monitoring only one signal allows for less bandwidth and lower power consumption when transmitting the signal by Bluetooth.
Two of the electrodes are attached across the heart to detect a signal, while the third is positioned on the body near the hip to form a ground reference.

3.1.2. Leads

A typical ECG lead is isolated to block noise from interfering with the heart signal. To reduce costs and lower complexity, the leads used in the project are unshielded wires soldered to the electrodes. In order to minimize this interference, a series of filters were employed to clean-up some of the signal.

3.1.3. Isolation Amplifier

ICs utilized in the circuit require a split supply of approx. ±5V to run. However, to preserve agility, it was anticipated to power the circuit with a 9V battery. By using an Operational amplifier and a voltage divider to set a virtual ground the split supply was designed. This can be seen in the figure below. Note that because this virtual ground is connected to the user's body through the third lead, the ground of our circuit is in reality set to the base voltage of the user's body.

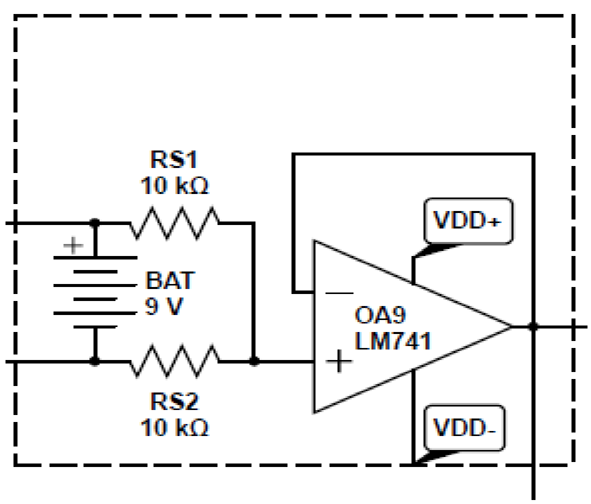

Figure 6 – Isolation and Supply Circuit [12]

3.1.4. Amplifier

To retrieve the differential signal from the user's heart, the circuit design uses a single instrumentation amplifier with a high gain.

The principal reason behind an instrumentation amplifier was floating reference of the heart signal.

The amplifier is adjusted to have some initial gain of 160 to escalate the strength of the signal before filtering. However, this signal comprises of large amount of noise that is also amplified. The major sources of noise are low frequency noise from the body, high frequency thermal and switching noise, and 60Hz power line interference due to the long leads.

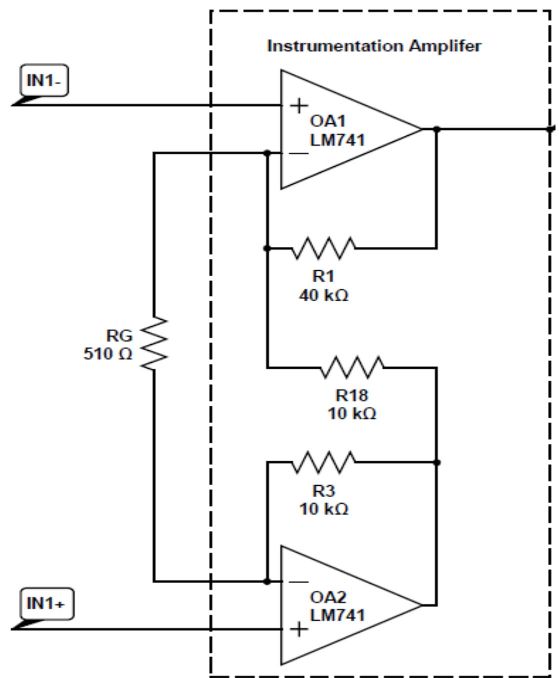

Figure 7 - Instrumentation Amplifier [12]

3.1.5. High-Pass Filter

The leading phase in the filter network is a passive high pass filter with a 0.5 Hz corner frequency. This filter eliminates maximum of the DC offset in our signal, and the corner frequency is set quite low due to our signal components existing near 1Hz. This filter is also compulsory because of a baseline wander caused by the respiratory system. Without a high-pass filter the output of the ECG would be the desired 1Hz heart signal superimposed on a ~0.2Hz sine wave.

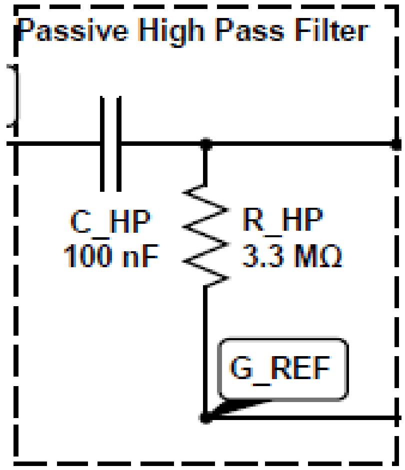

Figure 8 – High Pass Filter [12]

3.1.6. Notch Filter

The next phase is a 60Hz passive notch filter. This filter is vital due to the unshielded leads employed by the ECG. These function as antennas and collect the 60Hz noise from power lines. This interference was the most consistent part to remove from the signal due to its predictable frequency. This filter is essential due to the use of unshielded leads. Still, it was obvious that it is more cost efficient to implement a 60Hz notch filter than to use shielded leads.

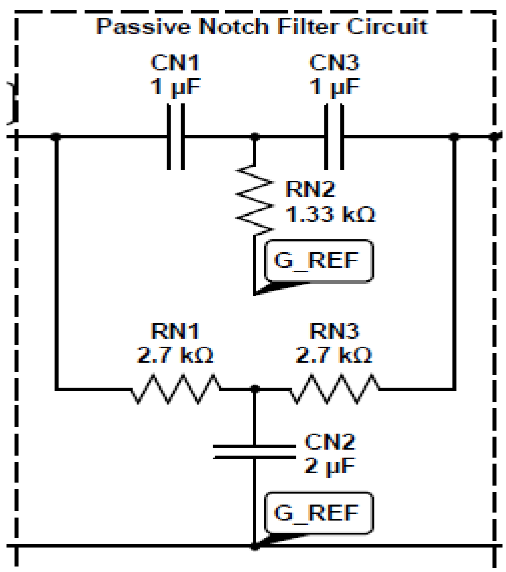

Figure 9 - Notch Filter [12]

3.1.7. Low-Pass Filter

The low pass filter is positioned with a corner frequency of 400Hz. This eradicates almost all of the unwanted high frequency noise from the waveform. In addition, this not only removes unnecessary high-frequency portion of the signal but also will reduces signal bandwidth. A passive filter network was preferred due to its significantly easier implementation as compared to an active filter.

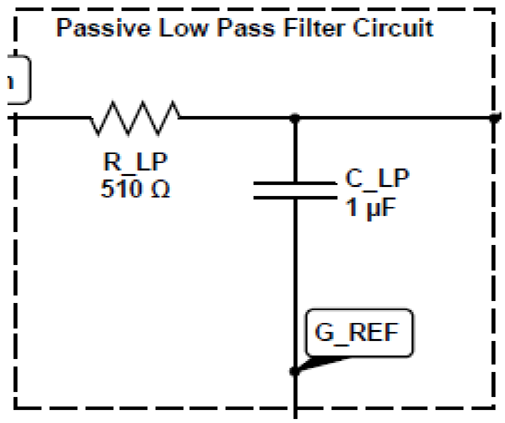

Figure 10 - Low Pass Filter [12]

3.1.8. Analog to Digital Converter

The ADC component is of extreme importance and is quite vital part of the ECG module. It takes the received ECG signal from the electrode circuitry and digitizes it for transmission to the mobile phone application. A widely available programmable microcontroller called Arduino Uno was chosen for this purpose and in order to implement this solution a standalone Bluetooth serial radio (such as a HC-05) was also acquired.

Figure 11 - Microcontroller (Arduino UNO)

3.2. Software Requirements

Choosing Bluetooth wireless communication protocol was an obvious solution and it was an easy decision to make due to various reasons. Modern Android smart phones with the capability to run the application already have a Bluetooth radio, so no additional hardware is desired in order to get the communication channel up and running.

Moreover, the ECG component would be simple to pair using this technology; particularly similar to that of a commonly employed audio headphones or speaker. This means that it would be very simple for the regular user to get the entire ECG system up and running on their mobile device. An important characteristic to point out for this project is the overall power consumption for both the ECG hardware and the mobile device.

3.2.1. Android Platform

The Android environment was chosen for ECG software for multiple reasons. One of the reasons for choosing the Android platform is that it provides the opportunity to run the app as a 'service', which permits it to run continually in the background.

The two foremost objectives for the Android application: First and foremost, the app must be able to analyze and evaluate the ECG signal attaining in real-time, and identify any heart complications that takes place. The app ought to be capable of analyzing this in a small amount of time so that a "help" message can be send out to stated individuals in situations of emergency. The other aim for the app was to store the cardiogram statistics of the user for later use. The incentive behind this originates from the fact that the response of the heart leading up to a event is often not examined, and could be useful to medical specialists - complications for this section include where to pile up the records (local, SD card, cloud service), in what format to store the data, and how to secure it and make it private for every specific user.

The Android application comprises of various Java classes and quite a few different interface screens to serve the prototype's objectives of displaying real-time ECG data and saving the data.

3.2.2. Features of the Android app

I. Login

The user has to login to use the services, if he does not have an account he will make an account using the sign up screen. Once signed up, his credentials are saved using the firebase authentication service. The main advantage of this is that when the user logs in once, the sign up firebase remembers the user unless it has been disabled or removed from the firebase using the dashboard.

Figure 12 - Login Screen

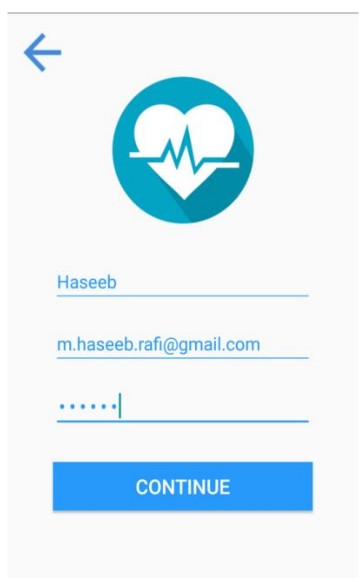

Figure 13 - User logs-in using his credentials

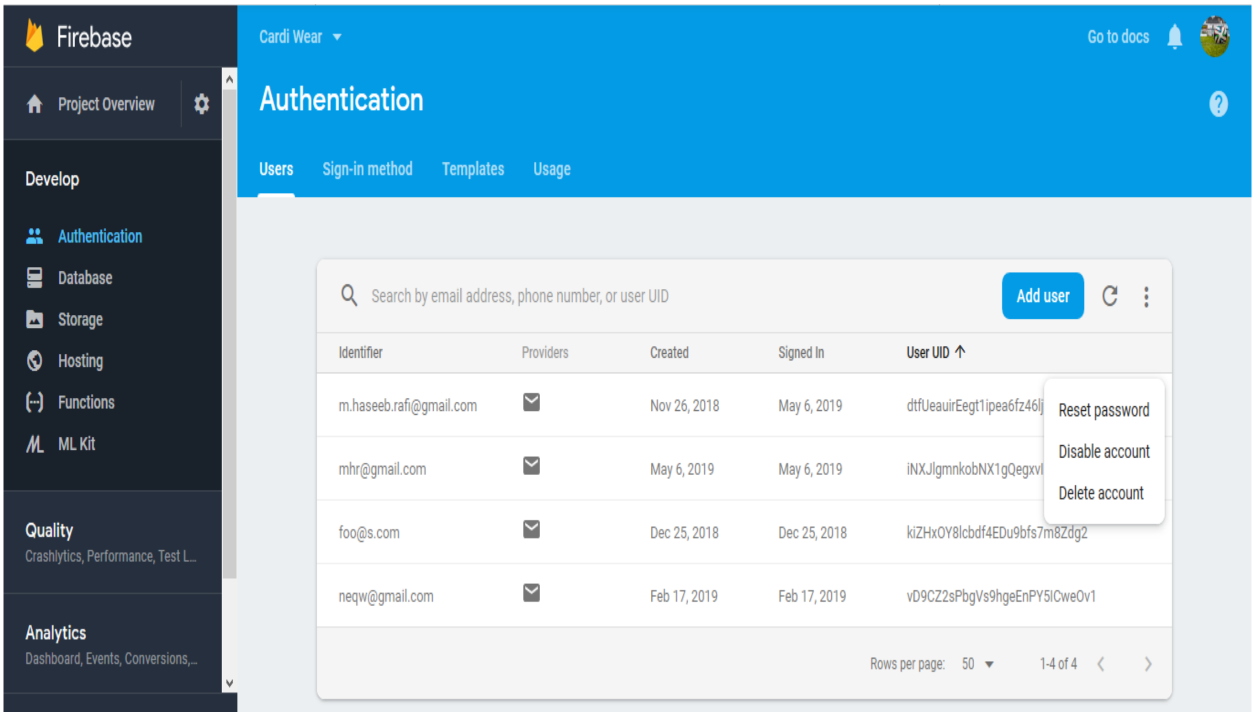

Figure 14 - Firebase Authentication

II. Main Screen:

After login, the user comes across the main screen of the Application, where he has various options. Those options includes: taking the new ECG, viewing past records and managing emergency contacts. When the new user signs up he is assigned an ID and that ID is unique to that user only.

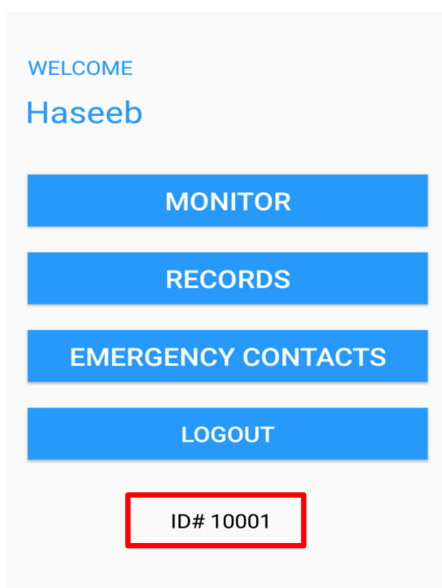

Figure 15 - Main Screen

III. Monitor:

In this screen, the user has to connect with the Bluetooth module by clicking on the button and choosing the desired module. After that he starts receiving the ECG signals which are displayed on the Application (on mobile screen). After he is done and finished with monitoring, he has to store the results by going to the records tab.

Figure 16 - Connecting to Bluetooth Module

IV. Records:

In this screen we have the previous records; it acquires data from the monitor screen of the recent ECG and displays it on the top then saves it on the firebase database.

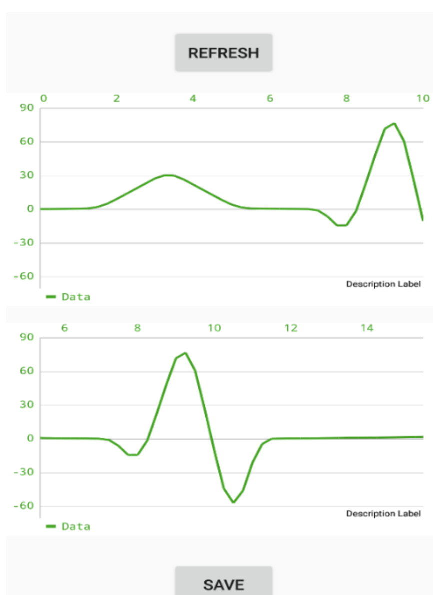

Figure 17 - Storing the Records on Database

V. Emergency contacts:

In this screen the user can save his emergency contacts and can send alert messages in case of emergency. The emergency contact numbers are also stored on the database.

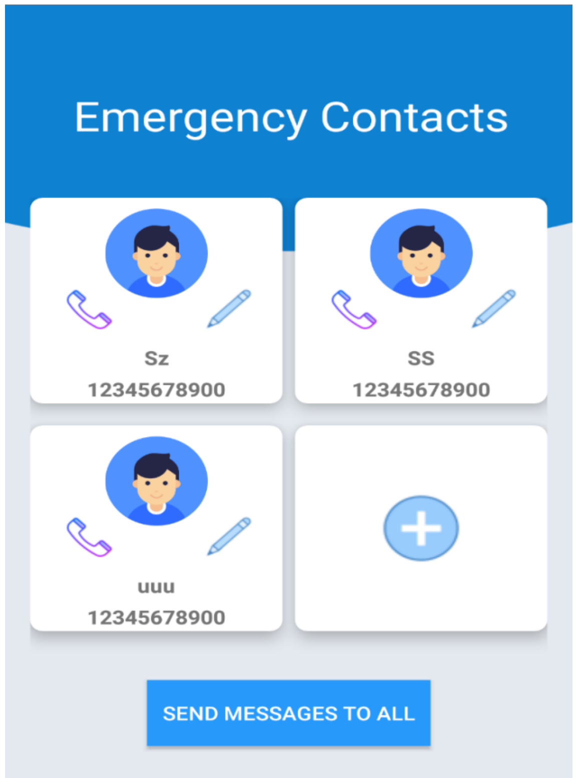

Figure 18 - Emergency Contacts

3.2.3. Web Portal

I. Login:

The user has to log in to view his records.

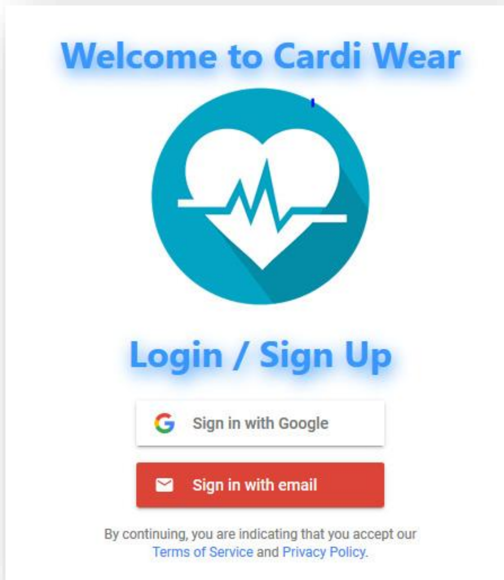

Figure 19 - Login Screen

II. Search:

Every user has already been assigned his own unique ID when he signed up. In order to access all his records he needs to enter the unique ID in the search tab after which past results will be displayed on the screen.

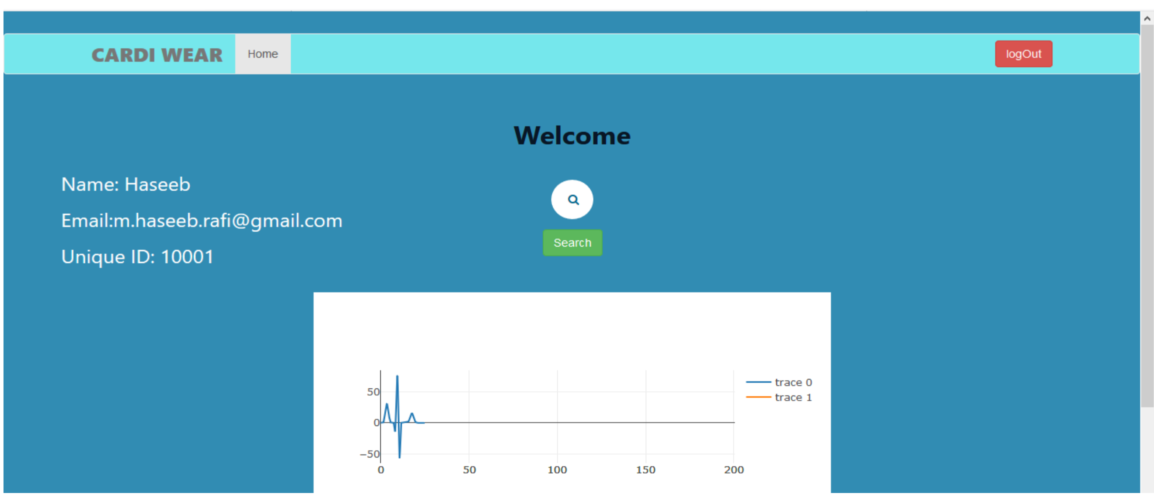

Figure 20 - Accessing Past Records

CHAPTER: 04

Project Analysis and Evaluation

4.1. Introduction

In this section we'll discuss the results we acquired after testing the model we made. The results we attained from our prototype were Correlated and evaluated with those obtained from conventional ECG machines available in the hospitals.

4.2. Circuit Design and Implementation of PCB

The following image shows the schematics for the ECG acquisition circuit designed on Proteus software. It has 8 main components: electrodes, the leads, an initial amplification stage, high-pass filter, notch filter, secondary amplification stage, low-pass filter.

The first portion of the circuit constitutes the INA126 precision instrumentation amplifiers for accurate, low noise differential signal acquisition. This two-op-amp design provides excellent performance with very low quiescent current (175mA/chan.). This, combined with wide operating voltage range of ±1.35V to ±18V, makes them ideal for portable instrumentation and data acquisition systems.

Then we have different filers that will clean the signal that INA126 fetches from the dry stainless steel electrodes. The LM741 amplifier offer many features which make their application nearly foolproof: overload protection on the input and output, no latch-up when the common mode range is exceeded, as well as freedom from oscillations.

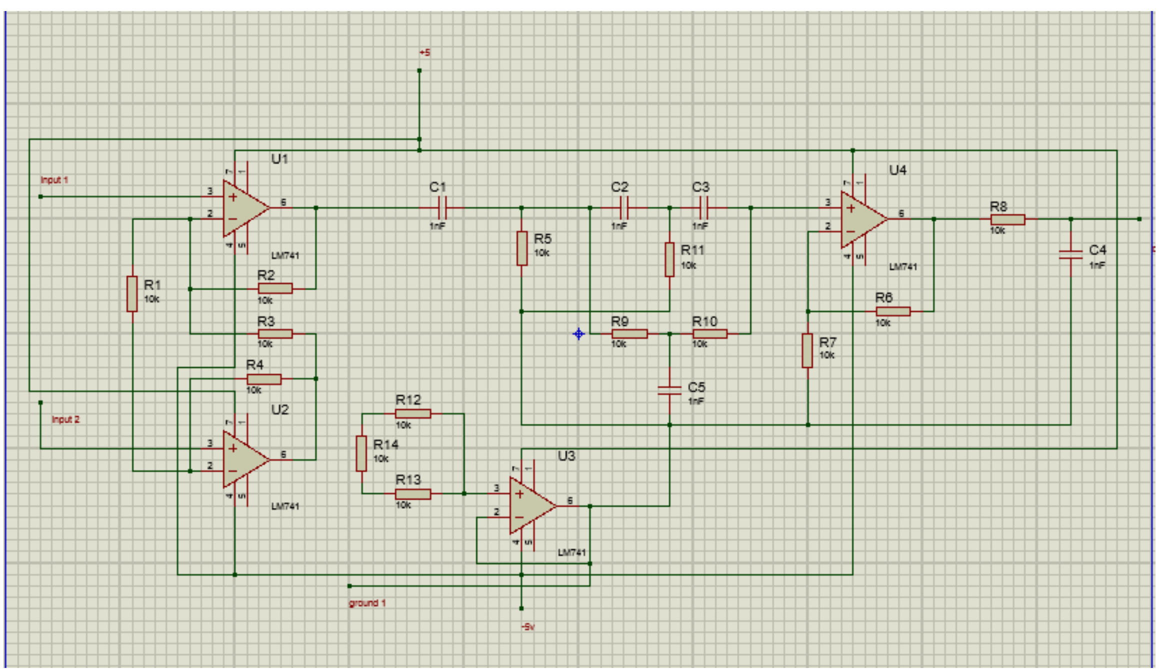

Figure 21 - Schematic Diagram on Proteus

The figures below show the PCB and 3D-design of the schematic using ARES PCB Design Software.

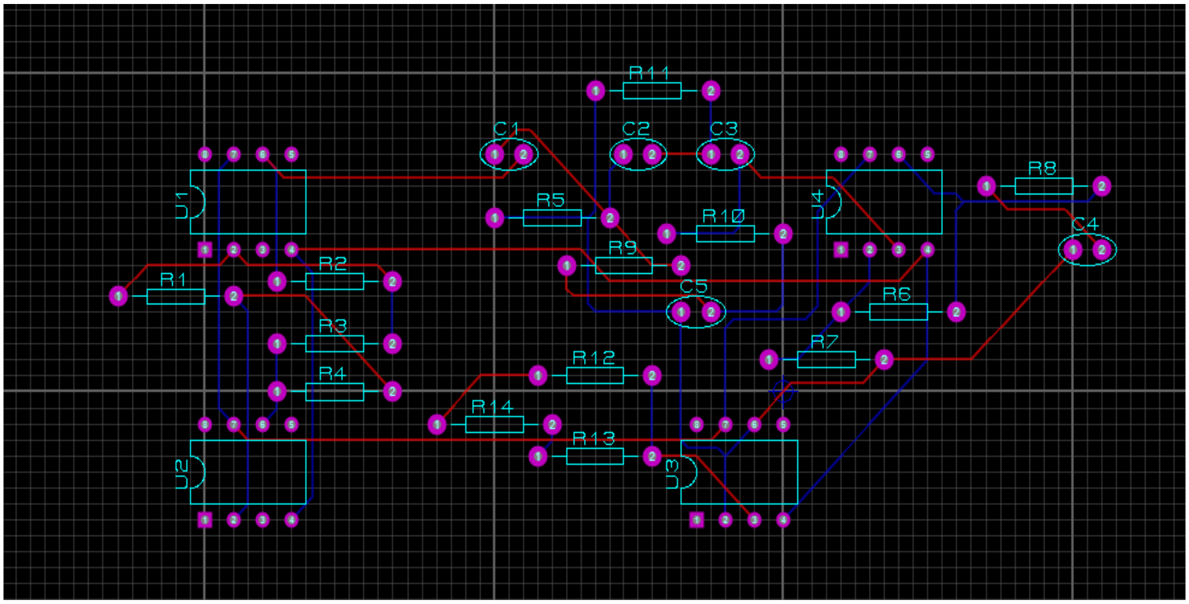

Figure 22 - ARES PCB Diagram

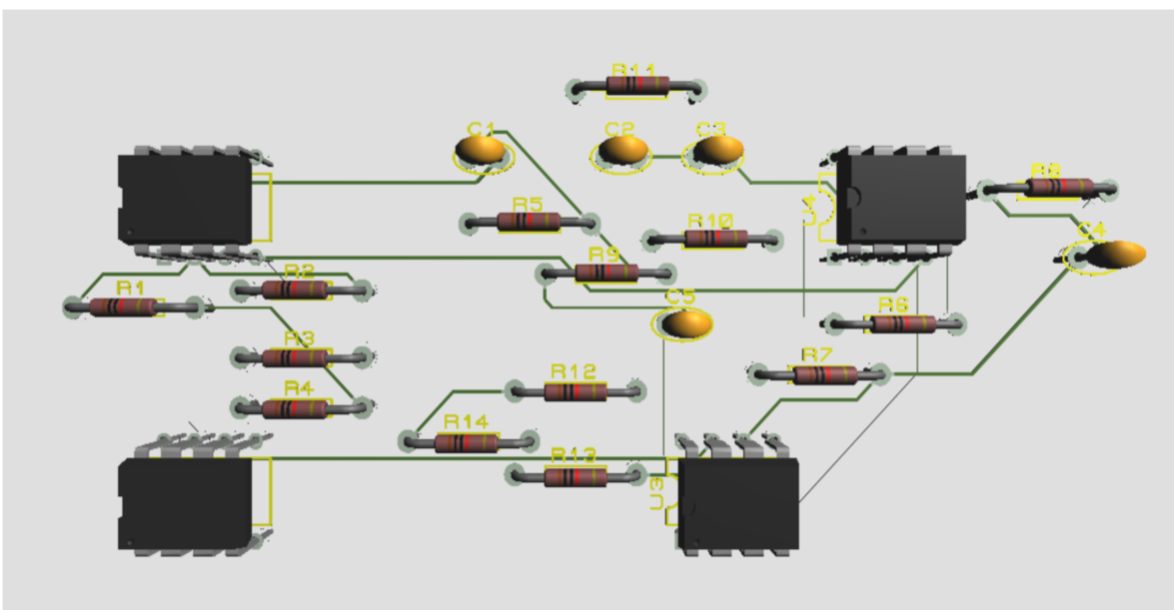

Figure 23 - 3D View of AERS Diagram

4.3. Breadboard Implementation and Real-Time ECG Acquisition

One of the most important steps to successfully implement an electronic circuit is to first implement it on a very basic scale. For this purpose we have created the circuit on the breadboard for testing purposes and received very convincing results on the application interfaced with the hardware section of the circuit. This is shown in the following images. This image corresponds to the Arduino interfaced with the hardware section. The sensors (Dry Electrodes) are acquiring the input signals from the person and the further processes are being carried out by the hardware section.

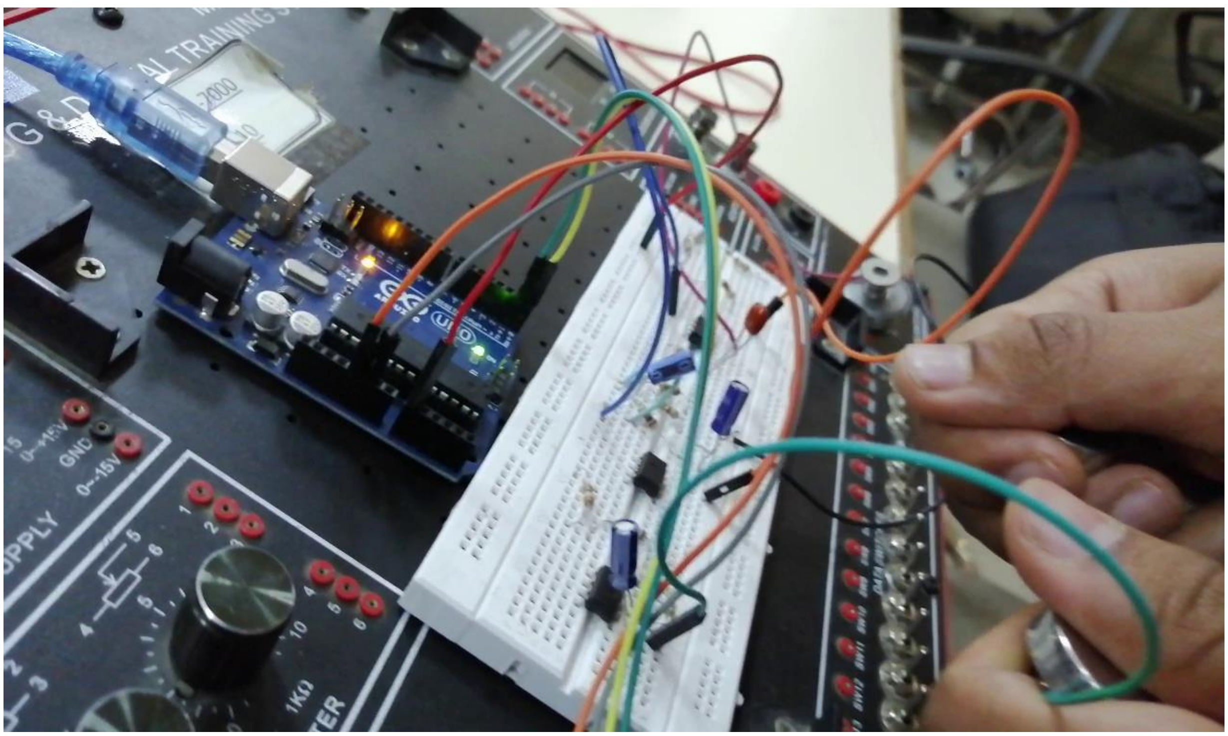

Figure 24 - Breadboard Implementation

The following results are being obtained on the mobile application corresponding to the successful acquisition of the required ECG Signal.

Figure 25 - Application Results

4.4. Raw ECG Results on an Oscilloscope

The following image shows the results from the ECG circuit without amplification and filtering processes. As it is visible from the image that there are certain noises and unwanted edges that needs to be eradicated in order to get a satisfactory result. This is basically the direct results from the sensors with a hardware filtration that is not enough to get precise and accurate results.

Figure 26 - ECG Results on oscilloscope (UNFILTERED)

4.5. ECG Results after Amplification and Filtration

The following graph shows the results after amplification and filtration processes have been performed. QRS complex in this image is more visible and amplified and presents a better understanding of ECG waveform of a patient as compared to the raw results. There exists a very small amount of noises that are contaminating the waveform which requires digital signal processing to be completely diminished. The filtering and amplification is performed in such a way that the raw results are fed to the Arduino which in turn runs the filtration and amplification codes to make the signal noise-free. The results are then displayed on the oscilloscope from the Arduino.

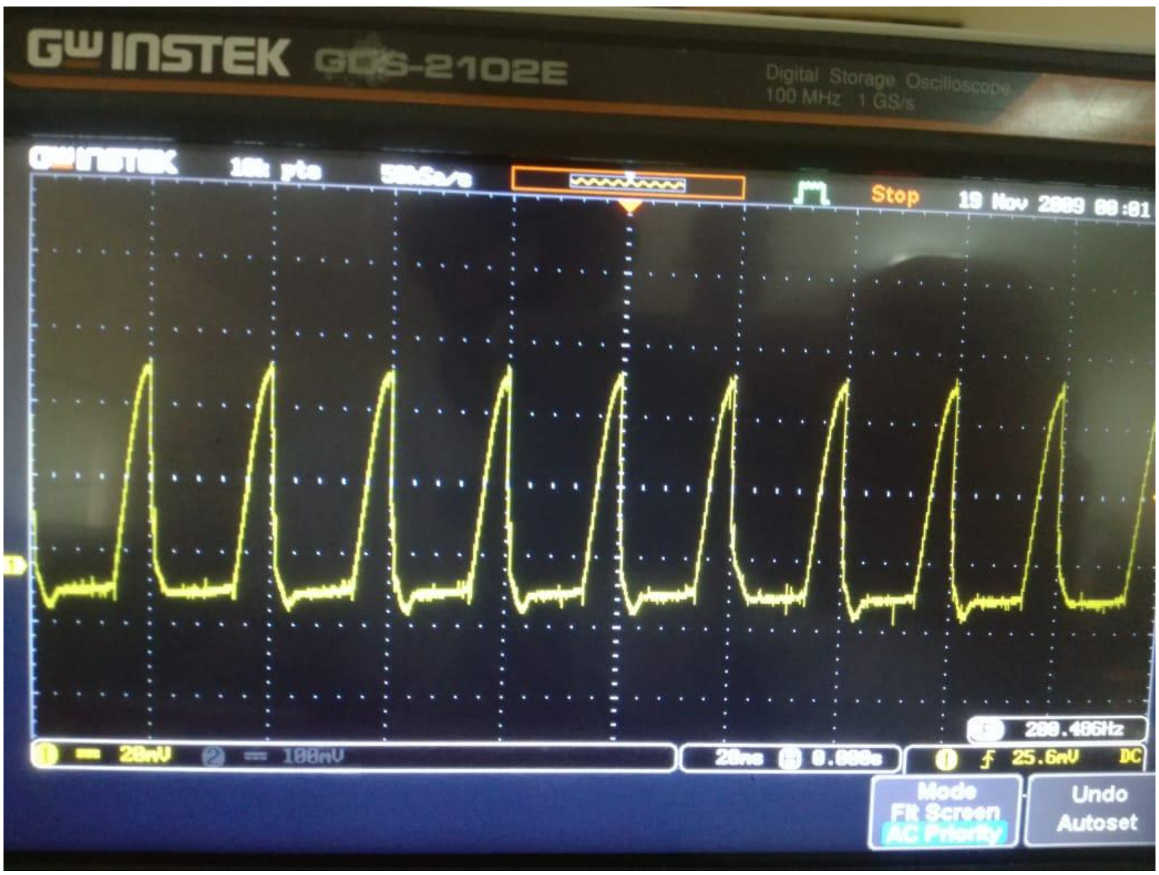

Figure 27 - ECG Results on oscilloscope (FILTERED)

4.6. ECG Database on Web-Portal

The results that have been obtained on Mobile Application are uploaded on Google Firebase. Firebase is essentially a real time database. The Firebase Real-time Database is a cloud-hosted on SQL database that lets you store and sync between your users in real-time. In order to get the ECG results for a certain patient, you just have search for the unique ID given to that patient and that will just display the results on the screen. This can be seen as follow:

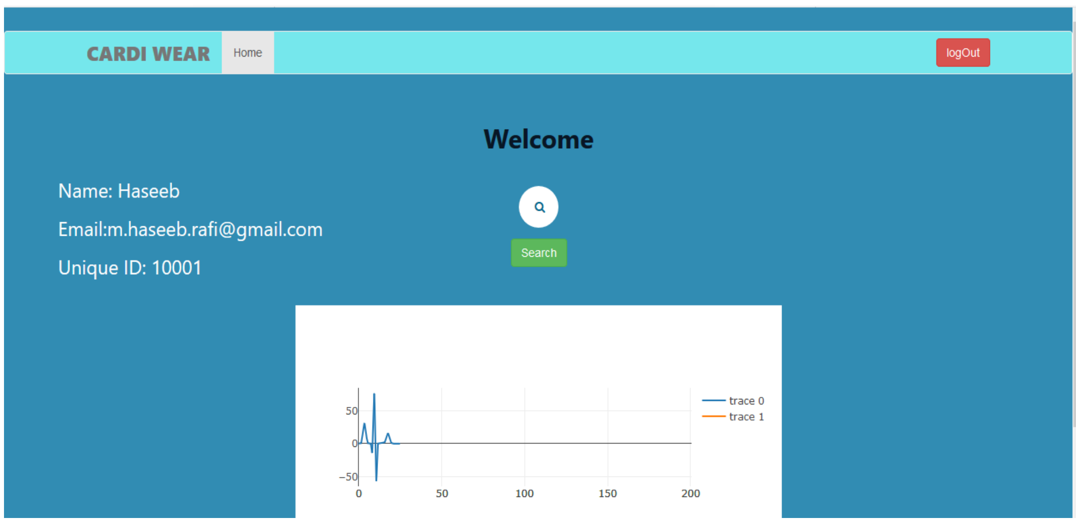

Figure 28 - Results on Webportal

4.7. Correlation between the ECG Results

The following plot is obtained by plotting the two results simultaneously on MATLAB and the correlation is obtained by using xcorr function which is built-in function of MATLAB. As it observed, the correlation between the two signals is 87.2% which is sufficiently good for an ECG signal acquired through normal dry (stainless steel) electrodes as compared to medical type electrodes.

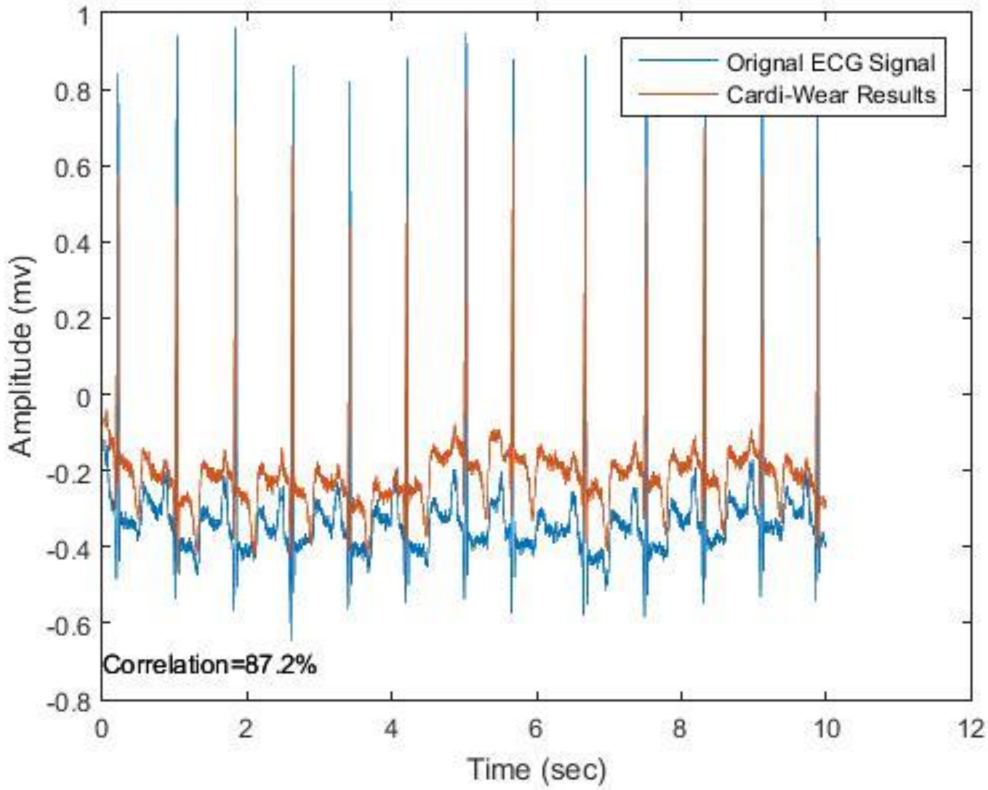

Figure 29 - Comparison of Results

CHAPTER: 05

Conclusion

5.1. Introduction:

Conclusions are the results which are attained after profound study and it tells us about the applicability of the device that how this device is beneficial and in which fields? Also discussed in the chapter are mentioned some future work that can be taken up by any interested individual or industry.

5.2. Potential Problems:

Though there were no extreme problems that we had to face either technically or socially, some of the minor issues that we came across were as follows:

- Too narrow audience
- Lack of audience interest
- Lack of funds
- Prototype not user friendly lack of resources

5.3. Key Barriers:

Some of the barriers and difficulties we encountered while designing this device are enumerated as mentioned below:

- Conversion from Prototype to marketable product.
- An accessory to wear.

5.4. Future Work:

Generally, there can be some improvements made in the project from both Android and Hardware perspectives:

5.4.1. Android Application

I. Multiple mobile devices attached to one ECG

An extension made to the prototype can include connecting multiple smartphones with a single ECG device so as that a number of individuals can monitor a single patient’s result at a time. To have multiple applications reading from the same device would require the server functionality to be implemented on the firmware of the ECG.

II. Detecting the nature of abnormalities

In addition to monitoring the records, analyzing the results and detecting the sort of diseases and abnormalities a user has is also a feature that can be added into out model. This generally involves machine learning and a data set will also have to be created for this purpose.

III. Automatically getting in touch with the contacts in case of emergency

Furthermore, adding the feature of automatically sending a message to an emergency contact can also be included in android application. To accomplish this, a processor-heavy signal analysis is required. Recommendation is to use Android's Native Development Kit.

5.4.2. Hardware Improvements

I. Migration to Programmable Microcontroller

Replacing the Arduino component with a programmable microcontroller is another up gradation that can be done. As the firmware component's only task is to digitize and transmit the analog ECG signal, this can be easily performed on an inexpensive microcontroller.

II. Leads

Using medical based stainless steel leads would create an immediate difference in the amount of noise the circuit would experience. While the existing circuit accurately removes most of the noise, better leads would reduce the amount of filtering that is necessary, and improve the accuracy of the final waveform.

III. Power

A better battery system would need to be developed before the hardware portion of this project could be brought to market. The existing two 5V batteries are bulky, considering the size of the application. With a smaller microcontroller, a battery would be sufficient to power the entire circuit, and a battery would be developed to keep a Bluetooth system active. Ideally, this battery would be rechargeable, reducing waste.

5.5. Conclusion:

An economical wearable device has been assembled that works as a replacement for the typical ECG device that used wet electrodes. It provides a clear outcome to the physician of the ECG curve of his patients who suffers from various heart problems.

The idea of Cardi-Wear is employed through the concept of “Dry Electrodes”. It allows users to get a hold of their ECG. An application has been developed based on Android Technology. This app is interfaced with the device via Bluetooth for the user. When the user turns on the Bluetooth and connects his smartphone with the device, ECG results are transferred through the Bluetooth module to the Android application and displayed on the mobile screen. We tried our best that the device can easily fit around the person.

A simple solution has presented and a low-priced prototype has been built in order to prove its viability. Dry Electrode ECG seems to be a good solution for the abovementioned situations, and its implementation in both public and private spaces should be a subject of future study.

CHAPTER: 06

Bibliography

[1] World Health Organization.int, ‘Cardiovascular diseases (CVDs)’, (2019).
Available: https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds)#
[Accessed: April’2019]

[2] Shifa.com.pk, ‘Chronic diseases in Pakistan’, (2019).
Available: https://www.shifa.com.pk/chronic-disease-pakistan/
[Accessed: April’2019]

[3] Southsudanmedicaljournal.com, ‘Basic principles of the ECG’, (2019).
Available: http://www.southsudanmedicaljournal.com/archive/may-2010/how-to-read-an-electrocardiogram-ecg.-part-one-basic-principles-of-the-ecg.-the-normal-ecg.html
[Accessed: May’2019]

[4] Heart.org, ‘Electrocardiogram (ECG or EKG)’, (2019).
Available: https://www.heart.org/en/health-topics/heart-attack/diagnosing-a-heart-attack/electrocardiogram-ecg-or-ekg
[Accessed: April’2019]

[5] National Center for Biotechnology Information, ‘Heart Disease and Stroke Statistics’, (2019).
Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5408160/
[Accessed: May’2019]

[6] Aziz, Kalim Uddin, et al. "Prevalence and awareness of cardiovascular disease including life styles in a lower middle class urban community in an Asian country." Pakistan Heart Journal (41.3-4) (2012)

[7] aclsmedicaltraining.com, ‘Basics of ECG’, (2019).
Available: https://www.aclsmedicaltraining.com/basics-of-ecg/
[Accessed: April’2019]

[8] Le, Tai, et al. "A low cost mobile ECG monitoring device using two active dry electrodes." 2016 IEEE Sixth International Conference on Communications and Electronics (ICCE). IEEE, 2016.

[9] Joutsen, Atte S., et al. "Dry electrode sizes in recording ECG and heart rate in wearable applications." EMBEC & NBC 2017. Springer, Singapore, 2017. 735-738.

[10] Pons, Emiliano Elías Mediavilla. "Wireless Electrocardiogram." (2009).

[11] Wang, Yishan, Ralf Wunderlich, and Stefan Heinen. "Design and evaluation of a novel wireless reconstructed 3-lead ECG monitoring system." Biomedical circuits and systems conference (biocas). IEEE, 2013.

[12] Atkinson, Michael, et al. "Android Bluetooth Electrocardiogram." University of Victoria (2013).

CHAPTER: 07

Appendices

APPENDIX - A

Main Activity.XML:

```
<RelativeLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/electrocardio"
        android:text="@string/electrocardio"

        android:textSize="@dimen/abc_text_size_large_material"
        android:textColor="#FFFFFF"
        android:textStyle="bold"

        android:background="#0174DF"

        android:layout_width="fill_parent"
        android:layout_height="wrap_content"

        android:paddingTop="14dp"
        android:paddingLeft="14dp"
        android:paddingRight="14dp"
        android:paddingBottom="14dp" />

    <LinearLayout
        android:id="@+id/graphToggleButtons"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:orientation="horizontal"
        android:weightSum="10">

        <ToggleButton
            android:id="@+id/tbScroll"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:checked="true"
            android:textOff="Auto Scroll"
            android:textOn="Auto Scroll"
            android:layout_weight="2.3"
            />

        <ToggleButton
```

```
        android:id="@+id/tbLock"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:checked="true"
        android:textOff="Lock X-Axis"
        android:textOn="Lock X-Axis"
        android:layout_weight="2.3"
        />

    <Button
        android:id="@+id/bXplus"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="+"
        android:layout_weight="2.7"/>
    <Button

        android:id="@+id/bXminus"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:text="-"
        android:layout_weight="2.7"/>

</LinearLayout>

<LinearLayout
    android:id="@+id/btButtons"

    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:layout_above="@id/graphToggleButtons"
    android:orientation="horizontal">

    <ToggleButton
        android:id="@+id/streamToggle"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:checked="false"
        android:textOff="Start Stream"
        android:textOn="Stop Stream"
        android:layout_weight="1"/>

    <Button
        android:id="@+id/btConnect"
        android:text="@string/btConnect"
```

```
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_weight="1" />

        <Button
            android:id="@+id/btDisconnect"
            android:text="@string/btDisconnect"

            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_weight="1" />
    </LinearLayout>

    <LinearLayout
        android:id="@+id/hrGraph"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/electrocardio"
        android:layout_above="@id/btButtons"
        android:orientation="horizontal"
        android:keepScreenOn="true"/>

</RelativeLayout>
```

APPENDIX - B

Main Screen.XML:

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainScreen">

    <TextView
        android:id="@+id/nonedit_text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="40dp"
        android:backgroundTint="#ffffff"
        android:ems="5"
        android:inputType="textPersonName"
        android:text="WELCOME"
        android:textColor="#2699fb"
        android:textSize="18dp"
        app:layout_constraintStart_toStartOf="@+id/guideline7"
        app:layout_constraintTop_toTopOf="parent" />;
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/profilenamehere_text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="16dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="10dp"
        android:backgroundTint="#ffffff"
        android:ems="8"
        android:inputType="textPersonName"
        android:text="Profile Name"
        android:textColor="#2699fb"
        android:textSize="28dp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/nonedit_text" />

    <Button
        android:id="@+id/monitor_button"
        android:layout_width="0dp"
```

```
    android:layout_height="wrap_content"
    android:layout_marginTop="40dp"
    android:background="#2699fb"
    android:text="MONITOR"
    android:textColor="#ffffff"
    android:textSize="23dp"
    app:layout_constraintEnd_toStartOf="@+id/guideline8"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="@+id/guideline7"
    app:layout_constraintTop_toBottomOf="@+id/profilenamehere_text" />

<Button
    android:id="@+id/data_button"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="24dp"
    android:background="#2699fb"
    android:text="RECORDS"
    android:textColor="#ffffff"
    android:textSize="23dp"
    app:layout_constraintEnd_toStartOf="@+id/guideline8"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="@+id/guideline7"
    app:layout_constraintTop_toBottomOf="@+id/monitor_button" />


<Button
    android:id="@+id/emergency_button"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="24dp"
    android:background="#2699fb"
    android:text="EMERGENCY CONTACTS"
    android:textColor="#ffffff"
    android:textSize="23dp"
    app:layout_constraintEnd_toStartOf="@+id/guideline8"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="@+id/guideline7"
    app:layout_constraintTop_toBottomOf="@+id/data_button"
    tools:ignore="MissingConstraints" />

<Button
    android:id="@+id/logout_button"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="24dp"
```

```
        android:background="#2699fb"
        android:text="LOGOUT"
        android:textColor="#ffffff"
        android:textSize="20dp"
        app:layout_constraintEnd_toStartOf="@+id/guideline8"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="@+id/guideline7"
        app:layout_constraintTop_toBottomOf="@+id/emergency_button" />

    <android.support.constraint.Guideline
        android:id="@+id/guideline7"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        app:layout_constraintGuide_percent="0.05" />

    <android.support.constraint.Guideline
        android:id="@+id/guideline8"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        app:layout_constraintGuide_percent="0.95" />

    <TextView
        android:id="@+id/uniqueID"
        android:layout_width="wrap_content"
        android:layout_height="21dp"
        android:layout_marginEnd="16dp"
        android:layout_marginLeft="28dp"
        android:layout_marginRight="16dp"
        android:layout_marginStart="28dp"
        android:layout_marginTop="44dp"
        android:ems="5"
        android:textColor="#000000"
        android:textSize="18dp"
        app:layout_constraintEnd_toStartOf="@+id/guideline8"
        app:layout_constraintStart_toStartOf="@+id/guideline7"
        app:layout_constraintTop_toBottomOf="@+id/logout_button" />


</android.support.constraint.ConstraintLayout>
```

APPENDIX - C

Sign In.XML:

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".sign">

    <ImageView
        android:id="@+id/back_img"
        android:layout_width="0dp"
        android:layout_height="45dp"
        android:layout_marginLeft="16dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="16dp"
        android:src="@mipmap/back"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <android.support.constraint.Guideline
        android:id="@+id/guideline5"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        app:layout_constraintGuide_percent="0.13" />

    <android.support.constraint.Guideline
        android:id="@+id/guideline6"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical"
        app:layout_constraintGuide_percent="0.87" />

    <ImageView
        android:id="@+id/main_logo"
        android:layout_width="150dp"
        android:layout_height="150dp"
        android:layout_marginEnd="8dp"
        android:layout_marginLeft="8dp"
        android:layout_marginRight="8dp"
        android:layout_marginStart="8dp"
        android:layout_marginTop="60dp"
        android:contentDescription=""
```

```
    android:src="@mipmap/logo"
    app:layout_constraintEnd_toStartOf="@+id/guideline6"
    app:layout_constraintStart_toStartOf="@+id/guideline5"
    app:layout_constraintTop_toTopOf="parent" />


<EditText
    android:id="@+id/profname_text"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="32dp"
    android:backgroundTint="#2699fb"
    android:ems="10"
    android:hint="Profile Name"
    android:inputType="textPersonName"
    android:textColor="#2699fb"
    app:layout_constraintEnd_toStartOf="@+id/guideline6"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="@+id/guideline5"
    app:layout_constraintTop_toBottomOf="@+id/main_logo"
    tools:ignore="MissingConstraints" />

<EditText
    android:id="@+id/editText"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="16dp"
    android:backgroundTint="#2699fb"
    android:ems="10"
    android:hint="Email Address"
    android:inputType="textPersonName"
    android:textColor="#2699fb"
    app:layout_constraintEnd_toStartOf="@+id/guideline6"
    app:layout_constraintStart_toStartOf="@+id/guideline5"
    app:layout_constraintTop_toBottomOf="@+id/profname_text" />

<EditText
    android:id="@+id/password_text"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="16dp"
    android:backgroundTint="#2699fb"
    android:ems="10"
    android:hint="Password"
    android:inputType="textPassword"
    android:textColor="#2699fb"
```

```
        app:layout_constraintEnd_toStartOf="@+id/guideline6"
        app:layout_constraintStart_toStartOf="@+id/guideline5"
        app:layout_constraintTop_toBottomOf="@+id/editText" />

    <Button
        android:id="@+id/continue_button"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginTop="32dp"
        android:background="#2699fb"
        android:text="CONTINUE"
        android:textColor="#ffffff"
        android:textSize="20dp"
        app:layout_constraintEnd_toStartOf="@+id/guideline6"
        app:layout_constraintStart_toStartOf="@+id/guideline5"
        app:layout_constraintTop_toBottomOf="@+id/password_text" />



</android.support.constraint.ConstraintLayout>
```

APPENDIX - D

Main Activity.JAVA:

```
package com.sun.michael.electrocardio2;
import android.app.Activity;
import android.bluetooth.BluetoothSocket;
import android.content.Intent;
import android.content.pm.ActivityInfo;
import android.graphics.Color;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.util.Log;
import android.view.View;
import android.view.Window;
import android.view.WindowManager;
import android.widget.Button;
import android.widget.LinearLayout;
import android.widget.Toast;
import android.widget.ToggleButton;

import com.jjoe64.graphview.GraphView;
import com.jjoe64.graphview.GraphView.GraphViewData;
import com.jjoe64.graphview.GraphView.LegendAlign;
import com.jjoe64.graphview.GraphViewSeries;
import com.jjoe64.graphview.GraphViewSeries.GraphViewStyle;
import com.jjoe64.graphview.LineGraphView;

public class MainActivity extends Activity implements View.OnClickListener{

   // Create variables
   static boolean Lock;
   static boolean AutoScrollX;
   static boolean Stream;

   Button xMinus;
   Button xPlus;

   ToggleButton lockToggle;
   ToggleButton scrollToggle;
   ToggleButton streamToggle;

   static LinearLayout GraphView;
   static GraphView graphView;
   static GraphViewSeries Series;

   private static double graphLastXValue = 0;
   private static int xView = 10;
```

```
    Button bluetoothConnect, bluetoothDisconnect;

    /** Handler for managing the threads. */
    Handler myHandler = new Handler() {
        @Override
        public void handleMessage(Message msg) {
            // TODO Auto-generated method stub
            super.handleMessage(msg);
            switch (msg.what) {
                case Bluetooth.SUCCESS_CONNECT:
                    Bluetooth.connectedThread = new
Bluetooth.ConnectedThread((BluetoothSocket)msg.obj);
                    Toast.makeText(getApplicationContext(),"Connected! Press Back to
Return",Toast.LENGTH_SHORT).show();
                    //String s = "successfully connected";
                    Bluetooth.connectedThread.start();
                    break;

                case Bluetooth.MESSAGE_READ:

                    byte[] readBuf = (byte[])msg.obj;
                    String strIncom = new String(readBuf,0,5);          //Create string from
bytes array

                    Log.d("strIncom", strIncom);
                    if (strIncom.indexOf('.') == 2 && strIncom.indexOf('s') == 0){
                        strIncom = strIncom.replace("s","");
                        if (isFloatNumber(strIncom)){
                            Series.appendData(new GraphViewData(graphLastXValue,
                                    Double.parseDouble(strIncom)),AutoScrollX);

                            //X-axis control
                            if (graphLastXValue >= xView && Lock){
                                Series.resetData(new GraphViewData[]{});
                                graphLastXValue = 0;
                            } else
                                graphLastXValue += 0.1;

                            if(Lock)
                                graphView.setViewPort(0,xView);
                            else
                                graphView.setViewPort(graphLastXValue-xView,xView);

                            //Refresh
                            GraphView.removeView(graphView);
```

```
                    GraphView.addView(graphView);
                    graphView.redrawAll();
                    //graphView.invalidate();
                }
            }
            break;
        }
    }

    public boolean isFloatNumber(String num){
        try{
            Double.parseDouble(num);
        } catch(NumberFormatException nfe) {
            return false;
        }
        return true;
    }
};

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

//this.setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_LANDSCAPE);
    requestWindowFeature(Window.FEATURE_NO_TITLE);
    this.getWindow().setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
            WindowManager.LayoutParams.FLAG_FULLSCREEN);
    setContentView(R.layout.activity_main);
    initializeGraph();
    initializeButtons();
}

/** Stop streaming if Back button pressed.*/
@Override
public void onBackPressed() {
    // TODO Auto-generated method stub
    if (Bluetooth.connectedThread != null) {
        Bluetooth.connectedThread.write("Q");
    }
    super.onBackPressed();
}

void initializeGraph(){

    Bluetooth.getHandler(myHandler);
```

```
    GraphView = (LinearLayout)findViewById(R.id.hrGraph);
    GraphView.setBackgroundColor(Color.BLACK);
    Series = new GraphViewSeries("Signal",new GraphViewStyle(Color.GREEN,2),
          new GraphViewData[]{new GraphViewData(0,0)});
    graphView = new LineGraphView(this,"Heart Rate");

    graphView.setViewPort(0,xView);
    graphView.setScrollable(true);
    graphView.setScalable(true);
    graphView.setShowLegend(true);
    graphView.setLegendAlign(LegendAlign.BOTTOM);
    graphView.setManualYAxis(true);
    graphView.setManualYAxisBounds(5,0);
    graphView.addSeries(Series);
    GraphView.addView(graphView);
}

void initializeButtons(){
    bluetoothConnect = (Button)findViewById(R.id.btConnect);
    bluetoothConnect.setOnClickListener(this);

    bluetoothDisconnect = (Button)findViewById(R.id.btDisconnect);
    bluetoothDisconnect.setOnClickListener(this);

    xMinus = (Button)findViewById(R.id.bXminus);
    xMinus.setOnClickListener(this);

    xPlus = (Button)findViewById(R.id.bXplus);
    xPlus.setOnClickListener(this);

    lockToggle = (ToggleButton)findViewById(R.id.tbLock);
    lockToggle.setOnClickListener(this);

    scrollToggle = (ToggleButton)findViewById(R.id.tbScroll);
    scrollToggle.setOnClickListener(this);

    streamToggle = (ToggleButton)findViewById(R.id.streamToggle);
    streamToggle.setOnClickListener(this);

    Lock = true;
    AutoScrollX = true;
    Stream = true;
}

/** OnClickListener method for the buttons.*/
@Override
```

```
    public void onClick(View v){

        // TODO Auto-generated method stub
        switch(v.getId()){
        case R.id.btConnect:
            startActivity(new Intent("android.intent.action.BT1"));
            break;
        case R.id.btDisconnect:
            Bluetooth.disconnect();
            Toast.makeText(getApplicationContext(), "Disconnected!",
Toast.LENGTH_SHORT).show();
            break;
        case R.id.bXminus:
            if (xView > 1) xView--;
            break;
        case R.id.bXplus:
            if (xView < 30) xView++;
            break;
        case R.id.tbLock:
            if (lockToggle.isChecked()){
                Lock = true;
            }else{
                Lock = false;
            }
            break;
        case R.id.tbScroll:
            if (scrollToggle.isChecked()){
                AutoScrollX = true;
            }else{
                AutoScrollX = false;
            }
            break;
        case R.id.streamToggle:
            if (streamToggle.isChecked()){
                if(Bluetooth.connectedThread != null)
                    Bluetooth.connectedThread.write("E");
            } else {
                if (Bluetooth.connectedThread != null)
                    Bluetooth.connectedThread.write("Q");
            }
            break;
        } }
}
```

APPENDIX - E

Bluetooth.JAVA:

```
package com.sun.michael.electrocardio2;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.ArrayList;
import java.util.Set;
import java.util.UUID;

import android.app.Activity;
import android.bluetooth.BluetoothAdapter;
import android.bluetooth.BluetoothDevice;
import android.bluetooth.BluetoothSocket;
import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.content.IntentFilter;
import android.os.Bundle;
import android.os.Handler;
import android.util.Log;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.Toast;

public class Bluetooth extends Activity implements OnItemClickListener{

    // Create variables
    static Handler myHandler = new Handler();

    static ConnectedThread connectedThread;
    public static final UUID MY_UUID = UUID.fromString("00001101-0000-1000-8000-
00805F9B34FB");
    protected static final int SUCCESS_CONNECT = 0;
    protected static final int MESSAGE_READ = 1;
    ArrayAdapter<String> listAdapter;
    ListView listView;
    static BluetoothAdapter bluetoothAdapter;
    Set<BluetoothDevice> deviceArray;
    ArrayList<String> pairedDevices;
    ArrayList<BluetoothDevice> devices;
    IntentFilter filter;
    BroadcastReceiver receiver;

    @Override
```

```
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_bluetooth);
        initialize();
        if (bluetoothAdapter == null){
            Toast.makeText(getApplicationContext(), "No bluetooth detected",
Toast.LENGTH_SHORT).show();
            finish();
        }else{
            if (!bluetoothAdapter.isEnabled()){
                turnOnBluetooth();
            }
            getPairedDevices();
            startDiscovery();
        }
    }

    /** Method for disconnecting from the Bluetooth module.*/
    public static void disconnect(){
        if (connectedThread != null) {
            connectedThread.cancel();
            connectedThread = null;
        }
    }

    public static void getHandler(Handler handler){//Bluetooth handler
        myHandler = handler;
    }

    /** Method for discovery nearby Bluetooth devices.*/
    private void startDiscovery() {
        // TODO Auto-generated method stub
        //bluetoothAdapter.cancelDiscovery();
        bluetoothAdapter.startDiscovery();
    }

    private void turnOnBluetooth() {
        Intent enableBluetoothIntent = new
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE);
        startActivityForResult(enableBluetoothIntent, 1);
    }

    private void getPairedDevices() {
        deviceArray = bluetoothAdapter.getBondedDevices();
        if (deviceArray.size()>0){
            for(BluetoothDevice device:deviceArray){
```

```
            pairedDevices.add(device.getName());
          }
        }
    }

    private void initialize(){
        bluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
        filter = new IntentFilter(BluetoothDevice.ACTION_FOUND);
        pairedDevices = new ArrayList<String>();
        devices = new ArrayList<BluetoothDevice>();

        listView = (ListView)findViewById(R.id.listView);
        listAdapter = new ArrayAdapter<String>(this,
android.R.layout.simple_list_item_1,0);
        listView.setAdapter(listAdapter);
        listView.setOnItemClickListener(this);

        receiver = new BroadcastReceiver(){
          @Override
          public void onReceive(Context context, Intent intent) {
            String action = intent.getAction();
            if (BluetoothDevice.ACTION_FOUND.equals(action)){
              BluetoothDevice device =
intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
              devices.add(device);
              String s = "";
              for(int a=0;a<pairedDevices.size();a++){
                if (device.getName().equals(pairedDevices.get(a))){
                  //append
                  s = "(Paired)";
                  break;
                }
              }
              listAdapter.add(device.getName()+" "+s+" "+"\n"+device.getAddress());

            }else if
(BluetoothAdapter.ACTION_DISCOVERY_STARTED.equals(action)){

            }else if
(BluetoothAdapter.ACTION_DISCOVERY_FINISHED.equals(action)){

            }else if (BluetoothAdapter.ACTION_STATE_CHANGED.equals(action)){
              if (bluetoothAdapter.getState() == bluetoothAdapter.STATE_OFF){
                turnOnBluetooth();
              }
            }
```

```
            }
        };

        registerReceiver(receiver, filter);
        IntentFilter filter = new
IntentFilter(BluetoothAdapter.ACTION_DISCOVERY_STARTED);
        registerReceiver(receiver, filter);
        filter = new IntentFilter(BluetoothAdapter.ACTION_DISCOVERY_FINISHED);
        registerReceiver(receiver, filter);
        filter = new IntentFilter(BluetoothAdapter.ACTION_STATE_CHANGED);
        registerReceiver(receiver, filter);
    }

    @Override
    protected void onPause() {
        // TODO Auto-generated method stub
        super.onPause();
        unregisterReceiver(receiver);
    }

    protected void onActivityResult(int requestCode, int resultCode, Intent data){
        super.onActivityResult(requestCode, resultCode, data);
        if (resultCode == RESULT_CANCELED){
            Toast.makeText(getApplicationContext(), "Bluetooth must be enabled to
continue", Toast.LENGTH_SHORT).show();
            finish();
        }
    }

    @Override
    public void onItemClick(AdapterView<?> arg0, View arg1, int arg2, long arg3) {
        // TODO Auto-generated method stub
        if (bluetoothAdapter.isDiscovering()){
            bluetoothAdapter.cancelDiscovery();
        }
        if (listAdapter.getItem(arg2).contains("(Paired)")){

            BluetoothDevice selectedDevice = devices.get(arg2);
            ConnectThread connect = new ConnectThread(selectedDevice);
            connect.start();
        }else {
            Toast.makeText(getApplicationContext(), "Device is not paired",
Toast.LENGTH_SHORT).show();
        }
    }
```

```
private class ConnectThread extends Thread {
    private final BluetoothSocket mmSocket;
    private final BluetoothDevice mmDevice;

    public ConnectThread(BluetoothDevice device) {
        // Use a temporary object that is later assigned to mmSocket,
        // because mmSocket is final
        BluetoothSocket tmp = null;
        mmDevice = device;

        // Get a BluetoothSocket to connect with the given BluetoothDevice
        try {
            // MY_UUID is the app's UUID string, also used by the server code
            tmp = device.createRfcommSocketToServiceRecord(MY_UUID);
        } catch (IOException e) { }
        mmSocket = tmp;
    }

    public void run() {
        // Cancel discovery because it will slow down the connection
        bluetoothAdapter.cancelDiscovery();

        try {
            // Connect the device through the socket. This will block
            // until it succeeds or throws an exception
            mmSocket.connect();
            //connectedThread = new ConnectedThread(mmSocket);
        } catch (IOException connectException) {
            // Unable to connect; close the socket and get out
            try {
                mmSocket.close();
            } catch (IOException closeException) { }
            return;
        }

        // Do work to manage the connection (in a separate thread)
        myHandler.obtainMessage(SUCCESS_CONNECT, mmSocket).sendToTarget();
    }

    /** Will cancel an in-progress connection, and close the socket */
    public void cancel() {
        try {
            mmSocket.close();
        } catch (IOException e) { }
    }
}
```

```
    static class ConnectedThread extends Thread {
        private final BluetoothSocket mmSocket;
        private final InputStream mmInStream;
        private final OutputStream mmOutStream;

        public ConnectedThread(BluetoothSocket socket) {
            mmSocket = socket;
            InputStream tmpIn = null;
            OutputStream tmpOut = null;

            // Get the input and output streams, using temp objects because
            // member streams are final
            try {
                tmpIn = socket.getInputStream();
                tmpOut = socket.getOutputStream();
            } catch (IOException e) { }

            mmInStream = tmpIn;
            mmOutStream = tmpOut;
        }
        StringBuffer sbb = new StringBuffer();
        public void run() {

            byte[] buffer;  // buffer store for the stream
            int bytes; // bytes returned from read()

            // Keep listening to the InputStream until an exception occurs
            while (true) {
                try {
                    try {
                        sleep(30);
                    } catch (InterruptedException e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                    }

                    buffer = new byte[1024];
                    // Read from the InputStream
                    bytes = mmInStream.read(buffer);
                    // Send the obtained bytes to the UI activity
                    myHandler.obtainMessage(MESSAGE_READ, bytes, -1,
buffer).sendToTarget();
                } catch (IOException e) {
                    break;
                }
```

```
            }
        }

        /* Call this from the main activity to send data to the remote device */
        public void write(String income) {

            try {
                mmOutStream.write(income.getBytes());
                for(int i=0;i<income.getBytes().length;i++)

Log.v("outStream"+Integer.toString(i),Character.toString((char)(Integer.parseInt(Byte.to
String(income.getBytes()[i])))));
                try {
                    Thread.sleep(20);
                } catch (InterruptedException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
            } catch (IOException e) { }
        }

        /* Call this from the main activity to shutdown the connection */
        public void cancel() {
            try {
                mmSocket.close();
            } catch (IOException e) { }
        }
    }
}
```

APPENDIX - F

CARDI WEAR

Extended Title: CARDI WEAR

Brief Description of The Project / Thesis with Salient Specifications: Cardiovascular disease is the leading global culprit for natural death, accounting for 29.2 percent of total death. The reported prevalence of coronary heart disease (CHD) in adult surveys has risen 4-fold in urban areas and doubled in rural areas over the past 30 years. The Electrocardiograph (ECG) is a diagnostic instrument which records the electrical activity of the heart from different angles to identify and locate pathology, providing valuable information of cardiac disorders. ECG instruments used in hospitals are bulky, work on line voltage which makes the ECG waveform distorted and an isolation amplifier is needed for patient safety making it expensive; thus, it cannot be used in underdeveloped areas.

Scope of Work: The project comprises a wearable vest using Dry Electrodes (Stainless Steel), the device will be interfaced to an App via Bluetooth to give real-time results. The recorded data will be stored on a cloud. A cardiologist can monitor the patients' results remotely in real-time and can give an E-subscription that will be displayed to the user via the App. A button is provided which may be pushed in case of any cardiac emergency alerting emergency contacts and services.

Academic Objectives :

- Electronics
- Microprocessors and system
- Programming techniques
- Structure designing

Application / End Goal Objectives:

- Making a low cost, compact and effective wearable alternative to conventional ECG machines.
- Making ECG results accessible by providing a mobile display of ones heartbeat.
- Using gel free electrodes for added comfort.
- Real-time ECG results on Cloud for storage and monitoring.

Previous Work Done on The Subject: Different variants of portable ECG have been designed called as holter monitors. But use wet electrodes and this includes the research of Linköping Studies, international journal of circuits, systems and signal processing, IJAET.

Material Resources Required: The system will work using Microprocessor and in concurrence with different sensors, Bluetooth module, Mobile Application and Cloud Interface. Required programming languages are Python, Java, HTML and MATLAB.

No of Students Required : 04

Group Members:

- NC Haseeb Rafi (Syndicate Leader)
- NC Muhammad Hashim
- NC Menahil Naeem
- NC Usama Waheed

Special Skills Required:

- App development
- Cloud storage
- Programming
- Embedded systems

APPENDIX - G

Cost Breakdown

ITEMS	COST
Arduino	$ 35
IC LM741	$ 10
Bluetooth Module	$ 15
Dry Electrodes	$ 5
Database Storage	$ 5
Total	**$ 65**

Table - 1

APPENDIX - H

Plan of Action

Ser. #	Date	Task	Completed Status
1	June 2018	• Project Synopsis and Approval	Completed
2	July-Oct 2018	• Literature Review • Hardware Implementation	Completed
3	Nov-Dec-Jan 2018	• Designing of Mobile Application • Hardware Implementation and Code Testing	Completed
5	Feb – Mar 2019	• Implementation of Mobile Application • Designing of Web Portal	Completed
6	April 2019	• Implementation of Cloud	Completed
7	May 2019	• Thesis Write-up • Finalizing	Completed

Table - 2