

# Chapter 1: Investigation

## 1.1 Synopsis:

### 1.1.1 Introduction:

Telemedicine will be playing a major role for analysis and determining the patient health condition. As people are more aware about their health analysis outside hospital, these technologies are becoming popular. Heart disease is the main cause for early death in our countries. The number of cardiac death is rapidly increasing day by day. However in certain conditions a fast, slow, or irregularly heart beat may lead to heart attack. Many of the cardiac deaths occur during normal routine. So the pulse sensor are required to Reduce the time and take an immediate action within the time. By the method of home care, one can analyze the basic physical parameters. The main motive of home care telemedicine is to furnish the patients with a low cost and user friendly interface system. This proposed system tests the health parameters by measuring heart beat rate with the help of biosensors continuously. The measured caregiver alerted with the help of connected sound system and data is also displayed in LCD. Honestly its hard to store **Arduino data to MySQL** directly so that in addition to **Arduino IDE** I used xamp server and My SQL. Hence the patient medical need is satisfied at the earliest.

### 1.1.2 Objectives:

The development of biomedical engineering is responsible for improving healthcare diagnosis, monitoring and therapy. The novel idea behind Health line is to provide quality health service to one and all. The idea s driven by the vision of a cable free biomedical monitoring system. On body sensors monitor the vital parameters (blood pressure, ECG and heart beat rate). Periodic health monitoring (or preventative care) allows people to discover and treat health problems early, before they have consequences. Especially for risk patients and long term applications, such a technology offers more freedom, comfort, and opportunities in clinical monitoring.

### 1.1.3 Software Requirements:

- IDE: Arduino 1.8.5
- Operating System: Windows 7 or higher
- Xampp server
- MySQL database server/phpMyAdmin)

## Heart Monitoring System

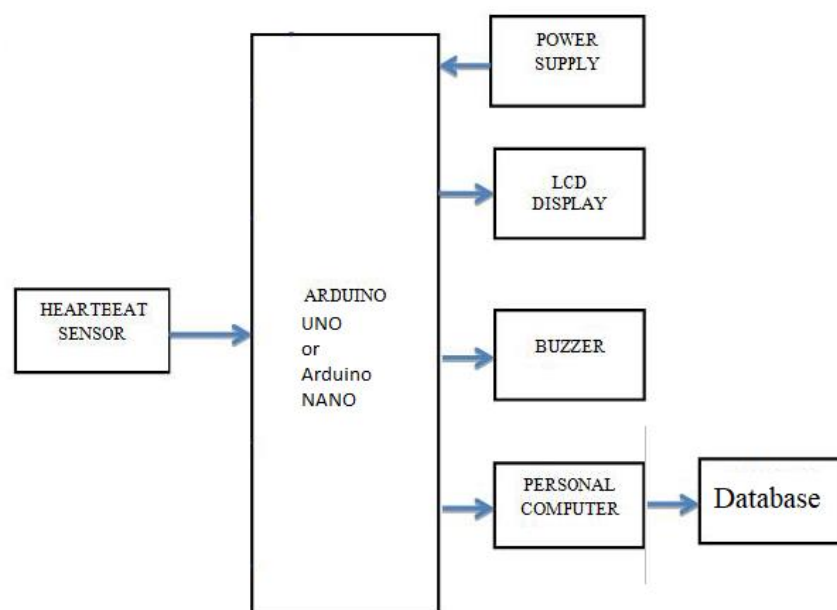
### 1.1.4 Hardware Requirements:

- Microcontroller: - Arduino Uno Board or Arduino NANO
- Sensors: -Heart Beat Sensor
- OLCD Display (128x64 (SSD1306 Driver))
- Breadboard
- Jumper Wires (Male to Female, Female to Female, Male to Male)
- Piezo Buzzer
- Arduino Ethernet Shield or Wi-Fi Module
- Ethernet cable
- Processor: Pentium IV or higher
- Processor speed: 1.6GHz
- RAM: 512 MB
- Disk Space: 250 MB or higher

### 1.1.5 Application:

- It can be used in hospitals, may be in ICU or general wards so that always a bulky ECG unit is not required.
- It can also be employed in the houses so that the doctor can come to know about the abnormalities when away.
- It also stores the data into the database and don't need to make a separate record of the output.

### 1.1.6 Block Diagram:



### **1.1.7 Proposed System:**

Sensor device implementation is done by microcontroller. Figure above shows a simplified block diagram where the block depicts the major components of the system and their interconnections. In this section, the proposed system, techniques adopted and working mechanism of sensors are explained. It consists of two main sections; the first section is to measure the patient heart rate continuously. Second section is to send all data to microcontroller for the data to pc. It is a user friendly interface and it becomes very easy for the users for getting information from the system. Physiological signs such as heart rate are continuously monitored within the system. Various types of transducer are used to sense these bioelectric signals. Pulse sensor is used to sense the heart beat per minute. All the programming and coding part are done by the microcontroller. All the sensors continuously keep track and send the signal to the microcontroller. All the acquired data are stored offline in pc. If there is any critical condition, the concerned caregivers will be given to the buzzer alert and data are displayed in LCD at a same time. In this section, the proposed system, techniques adopted and working mechanism of sensors are explained. It consists of three main sections; the first section is to measure the patient heart rate continuously. The second section is to send all data to microcontroller for the data to pc and the third section is to store the data in the database. It is a user friendly interface and it becomes very handy for the users for getting information for the system.

### **1.1.8 Existing System:**

In the existing system, we use active network technology to network various sensors to a single PMS. Patients' various critical parameters are continuously monitored via single PMS and reported to the Doctors or Nurses in attendance for timely response in case of critical situations. The sensors are attached to the body of the patients without causing any discomfort to them. In this PMS we monitor the important physical parameters like ECG, heart beat rate using the sensors which are readily available. Thus, the analog values that are sensed by the different sensors are then given to a microcontroller attached to it. The microcontroller processes these analog signal values of health parameters separately and converts it to digital values using ADC converter. Now, the digitalized values from more than one microcontroller are sent to the Central PMS. Each module transmits the data wirelessly to the gateway attached to the PC of the Central PMS. At any time, any of the doctors or nurses can log on the Central PMS and check the history of the observed critical parameters of any of the patient attached to the network.

### **1.1.9 Limitations of the system:**

The scope of the project was limited to ECG, heart pulse detection blood pressure and remote viewing of the collected data for a single patient. Here, the most important specification considered was that they should be safe to use and accurate. This is because the physiological information being detected determines the severity of a critical life-threatening situation.

# Chapter 2: Analysis

## 2.1 Project history:

- In 1872, Alexander Muirhead is reported to have attached wires to the wrist of a patient with fever to obtain an electronic record of their heartbeat.
- In 1882, John Burdon-Sanderson working with frogs, was the first to appreciate that the interval between variations in potential was not electrically quiescent and coined the term "isoelectric interval" for this period.
- In 1887, Augustus Waller invented an ECG machine consisting of a Lippmann capillary electrometer fixed to a projector. The trace from the heartbeat was projected onto a photographic plate that was itself fixed to a toy train. This allowed a heartbeat to be recorded in real time.
- In 1895, Willem Einthoven assigned the letters P, Q, R, S, and T to the deflections in the theoretical waveform he created using equations which corrected the actual waveform obtained by the capillary electrometer to compensate for the imprecision of that instrument. Using letters different from A, B, C, and D (the letters used for the capillary electrometer's waveform) facilitated comparison when the uncorrected and corrected lines were drawn on the same graph. Einthoven probably chose the initial letter P to follow the example set by Descartes in geometry. When a more precise waveform was obtained using the string galvanometer, which matched the corrected capillary electrometer waveform, he continued to use the letters P, Q, R, S, and T, and these letters are still in use today. Einthoven also described the electrocardiographic features of a number of cardiovascular disorders.
- In 1897, the string galvanometer was invented by the French engineer Clément Ader.
- In 1901, Einthoven, working in Leiden, the Netherlands, used the string galvanometer: the first practical ECG. This device was much more sensitive than both the capillary electrometer Waller used and the string galvanometer.
- In 1924, Einthoven was awarded the Nobel Prize in Medicine for his pioneering work in developing the ECG.
- By 1927, General Electric had developed a portable apparatus that could produce electrocardiograms without the use of the string galvanometer. This device instead combined amplifier tubes similar to those used in a radio with an internal lamp and a moving mirror that directed the tracing of the electric pulses onto film.
- In 1937, Taro Takemi invented a new portable electrocardiograph machine.

## Heart Monitoring System

Though the basic principles of that era are still in use today, many advances in electrocardiography have been made since 1937. Instrumentation has evolved from a cumbersome laboratory apparatus to compact electronic systems that often include computerized interpretation of the electrocardiogram.

## 2.2 Requirement Gathering:

Proposed system consists of following sensors and modules

1. Arduino Micro Controller
2. Pulse Sensor
3. LCD display
4. Buzzer
5. Connecting wires
6. Power Supply
7. Arduino Ethernet Shield or Wi-Fi Module
8. Ethernet cable
9. Breadboard
10. Arduino IDE
11. Xampp server
12. MySQL database server/phpMyAdmin)

### 2.2.1 Hardware Requirement

This project is based on both hardware and software. The hardware requirements are as follows: -

#### 1]Arduino Microcontroller: -

Arduino Uno or Nano is a microcontroller board based on the ATmega328P. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

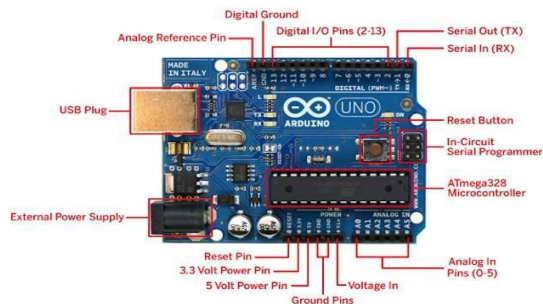


Figure: Arduino Uno Board

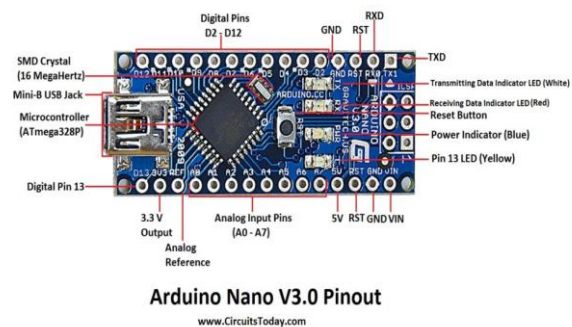


Figure: Arduino Nano Board

## Heart Monitoring System

### 2]Pulse Sensor: -

Optical heart-rate monitors are easy to understand in theory. If we've ever shined a flashlight through our finger tips and seen our heart-beat pulse (a thing most kids have done) we have a good handle on the theory of optical heart-rate pulse sensors. In an optical heart-rate pulse sensor, light is shot into a fingertip or ear lobe. The light either bounces back to a light sensor, or gets absorbed by blood cells. As we continue to shine light (into say a fingertip) and take light sensor readings, we quickly start to get a heart-beat pulse reading.



Figure: Pulse Sensor

### 3]LCD display: -

The *organic light-emitting diode* (OLED) display that we'll use in this tutorial is the SSD1306 model: a monochrome, 0.96-inch display with 128×64 pixels. The OLED display doesn't require backlight, which results in a very nice contrast in dark environments. Additionally, its pixels consume energy only when they are on, so the OLED display consumes less power when compared with other displays. The model we're using here has only four pins and communicates with the Arduino using I2C communication protocol.

**Pin wiring:-** Because the OLED display uses I2C communication protocol, wiring is very simple. You just need to connect to the Arduino Uno or Nano I2C pins as shown in the table below.

Pin	Wiring to Arduino Uno
Vin	5V
GND	GND
SCL	A5
SDA	A4



## Heart Monitoring System

To control the OLED display you need the [adafruit\\_SSD1306.h](#) and the [adafruit\\_GFX.h](#) libraries

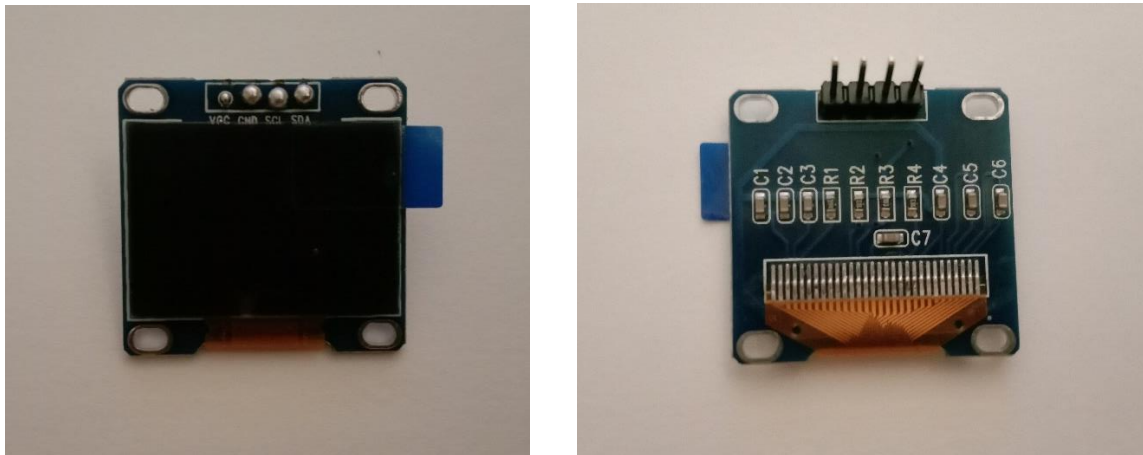


Figure: OLED Display

### 4]Buzzer: -

A buzzer or beeper is an audio signalling device, which may be mechanical, electromechanical, or piezoelectric. Typical uses of buzzers and beepers include alarm devices, timers and confirmation of user input such as a mouse click or keystroke.

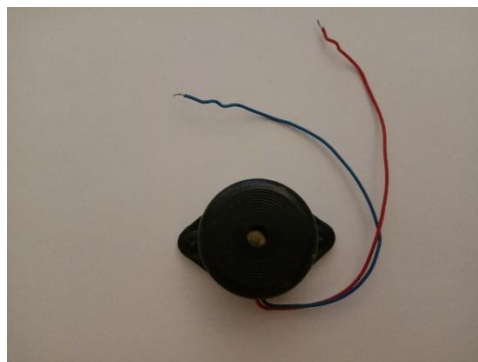


Figure: Buzzer

### 5]Arduino Ethernet shield or Wi-Fi Module: -

The Arduino Ethernet Shield allows an Arduino board to connect to the internet. It is based on the Wiznet W5100 ethernet chip (datasheet). The Wiznet W5100 provides a network (IP) stack capable of both TCP and UDP. It supports up to four simultaneous socket connections. Use the Ethernet library to write sketches which connect to the internet using the shield. The ethernet shield connects to an Arduino board using long wire-wrap headers which extend through the shield. This keeps the pin layout intact and allows another shield to be

## Heart Monitoring System

stacked on top. The latest revision of the shield adds a micro-SD card slot, which can be used to store files for serving over the network.

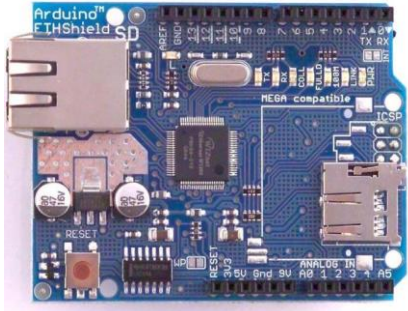


Figure: Arduino Ethernet shield

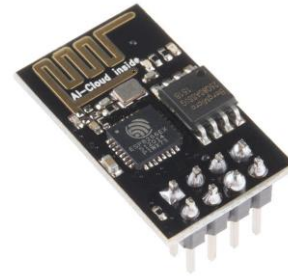


Figure: ESP8266 Wifi Module

For connecting the device to internet we used ESP8266. The ESP8266 is a low-cost Wi-Fi chip with full TCP/IP stack and MCU (Micro Controller Unit) capability produced by Shanghai-based Chinese manufacturer, Espressif Systems.

### 6]Ethernet cable: -

An [Ethernet](#) cable is one of the most common types of network cables used for wired networks. Ethernet cables connect devices within a [local area network](#), like PCs, [routers](#), and [switches](#). These physical cables are limited by their length and by their durability. If a network cable is too long or of poor quality, it won't carry a good network signal. These limits are one reason there are different types of Ethernet cables optimized to perform certain tasks in specific situations. Connect the Ethernet Shield to a network router, or to your computer, using an RJ45 cable.



Figure: Ethernet cable

### 7]Power Supply: -

A power supply is an electronic device that supplies electric energy to an electrical load. The primary function of a power supply is to convert one form of electrical energy to another and, as a result, power supplies are sometimes referred to as electric power converters. Some powersupplies are discrete, stand-alone devices, whereas others are built into larger devices

## Heart Monitoring System

along with their loads. Here, we use 5v dc power or sometimes power is given to the circuit directly from computer.



Figure: Power supply USB code and charger

### 8]Connecting Wires: -

A Wire is a single usually cylindrical, flexible strand or rod of metal. Wires are used to bear mechanical loads or electric and telecommunication signals. Wire is formed by drawing the metal through a hole in a die or draw plate.



Figure: Jumper wires

### 9]Breadboard: -

A **breadboard** is a construction base for [prototyping](#) of [electronics](#). Originally the word referred to a literal bread board, a polished piece of wood used for slicing bread[[citation needed](#)]. In the 1970s the **solderless breadboard** (a.k.a. **plugboard**, a terminal array board) became available and nowadays the term "breadboard" is commonly used to refer to these.

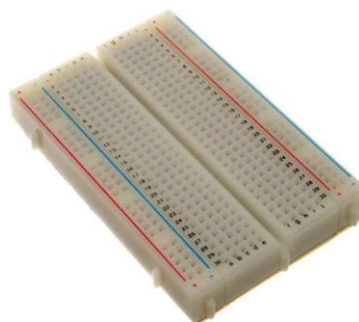


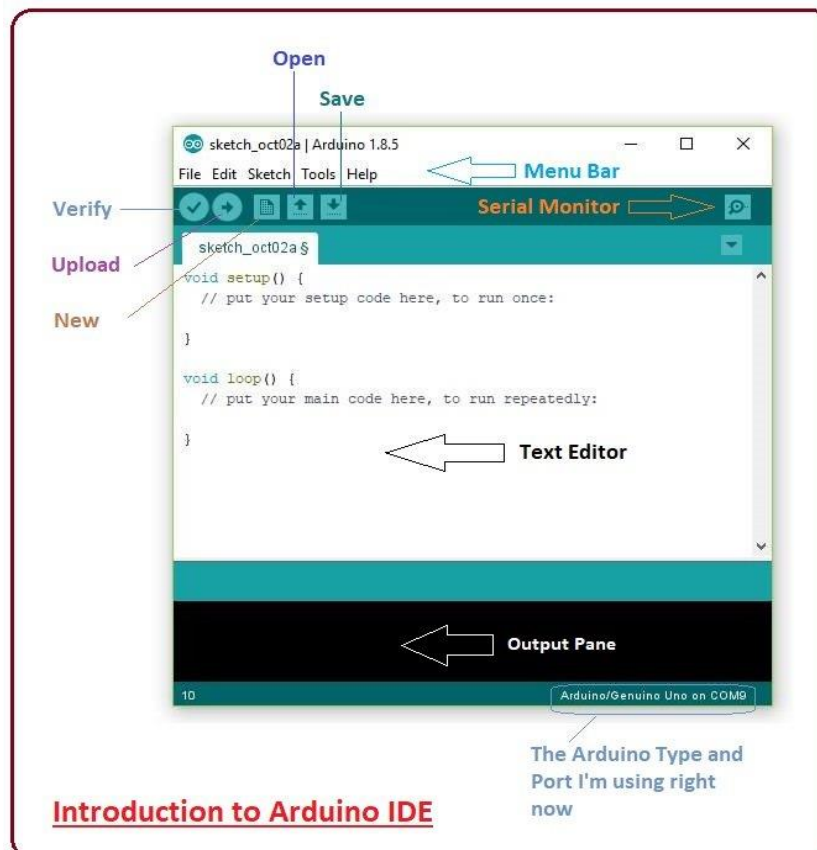
Figure: Breadboard

### 2.2.2 Software Requirement

As explained earlier our project requires two-part hardware and software. Hardware parts are explained above and software requires as follows: -

#### 1]Arduino IDE: -

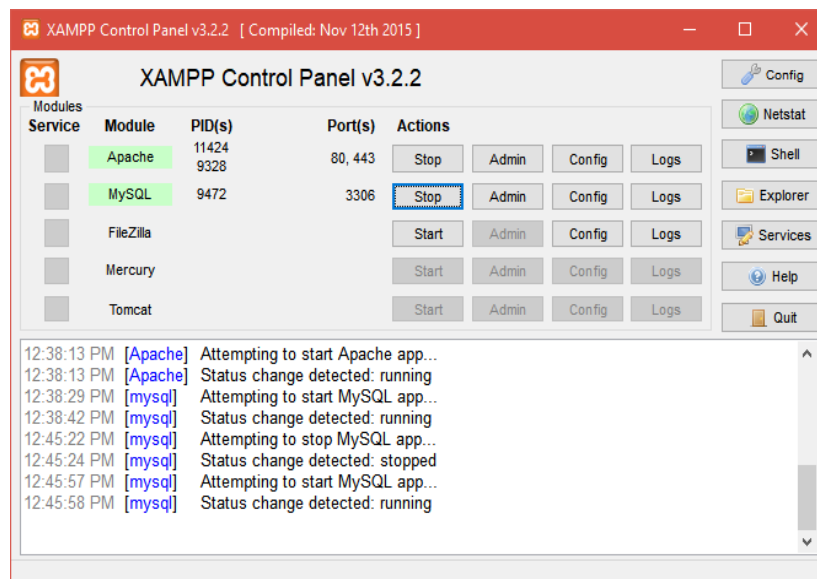
The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software. This software can be used with any Arduino board. The Arduino development environment contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions, and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them.



#### 2]Xampp server: -

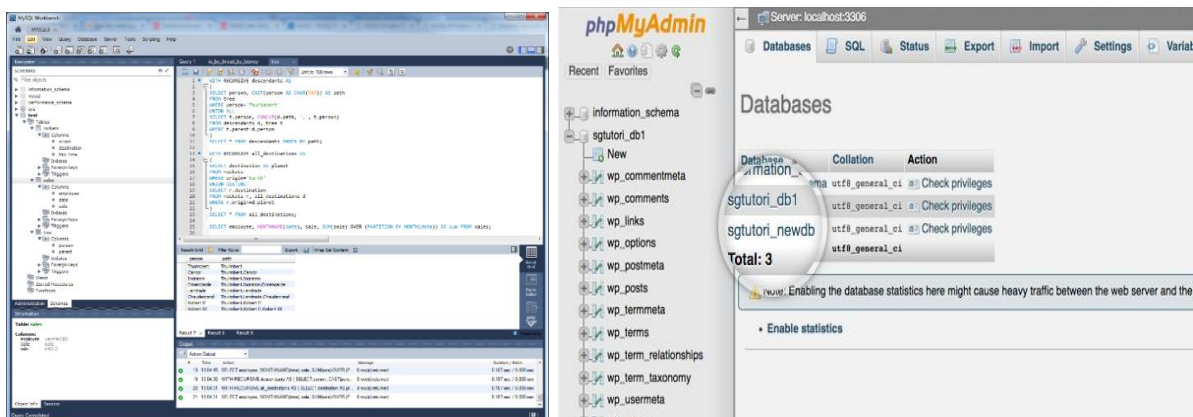
XAMPP is a free and open-source cross-platform web server solution stack package developed by Apache Friends, consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages. Since most actual web server deployments use the same components as XAMPP, it makes transitioning from a local test server to a live server possible.

## Heart Monitoring System



### 3]MySQL database server(phpMyAdmin):-

MySQL is free and open-source software under the terms of the GNU General Public License, and is also available under a variety of proprietary licenses. MySQL was owned and sponsored by the Swedish company MySQL AB, which was bought by Sun Microsystems (now Oracle Corporation). In 2010, when Oracle acquired Sun, Widenius forked the open-source MySQL project to create MariaDB. MySQL is a component of the LAMP web application software stack (and others), which is an acronym for Linux, Apache, MySQL, Perl/PHP/Python. phpMyAdmin is a free and open source administration tool for MySQL and MariaDB. As a portable web application written primarily in PHP, it has become one of the most popular MySQL administration tools, especially for web hosting services.



### **2.3 Objective & Scope of project:**

#### **2.3.1 Objective:**

The development of biomedical engineering is responsible for improving healthcare diagnosis, monitoring and therapy. The novel idea behind Health line is to provide quality health service to one and all. The idea is driven by the vision of a cable free biomedical monitoring system. On body sensors monitor the vital parameters (blood pressure, ECG and heart beat rate). Periodic health monitoring (or preventative care) allows people to discover and treat health problems early, before they have consequences. Especially for risk patients and long term applications, such a technology offers more freedom, comfort, and opportunities in clinical monitoring. Some of the objectives are:-

- To overcome the disadvantages of the existing system.
- The specialist staying at a distance can monitor the patients conditions so that he can save the life of the patient.
- This system is to be available at reasonable prices.
- Embedded technology is to be use so that we can monitor the patient condition easily.

#### **2.3.2 Scope of project:**

Long waiting time for hospitalization or ambulatory patient monitoring/treatment, are other well-known issues for both the healthcare institutions and the patients. This project provides healthcare authorities to maximize the quality and breadth of healthcare services by controlling costs. As the population increases and demand for services increases, the ability to maintain the quality and availability of care, while effectively managing financial and human resources, is achieved by this project. The use of modern communication technology in this context is the sole decisive factor that makes such communication system successful. Some of them are:-

- EEG, ECG and other health parameters can also be monitored
- Continuous monitoring and future diagnosis can be performed via the same system (TELEMEDICINE).
- More than a single patient at different places can be monitored using single system.

### **2.4 Problems with Existing System:**

- The system are not atomized.
- The patient can't able to understand the analog signals.
- Complex system and difficult to operate.
- To expensive.
- Not Handy.
- If power supply fails circuit won't work.

### **2.5 Advantages of Proposed System:**

- The staying of specialist is eliminated.
- Best to be used in rural areas.
- Reduce hospitalization fees.
- It is a multipurpose so that overall condition are easily measured.
- Easy to operate.
- Compare with compact sensor it gives better performance.
- Sensors are very cheap.



### 2.6 Feasibility Study:

Feasibility analysis is used to aid the decision of whether or not to proceed with the proposed system. Our system is independent and easy to use. This is why it can be used at home or in hospitals. In our analysis the methodology we use is more feasible than the existing methods of measuring the patient body parameters

This project has been tested in the following areas of feasibility:

#### 2.6.1. Economic feasibility

#### 2.6.2. Operation feasibility

#### 2.6.3. Technical feasibility

#### 2.6.4. Schedules feasibility

##### ❖ ECONOMICAL FEASIBILITY

The existing methods are not so cheaper because it has many disadvantages like all systems are being designed for measuring a specific parameter only. The systems which are existing today are also so costly. And these systems must be stored in a certain temperature for a perfect working. So, for the maintaining of these system air conditioners will be used this consumes much electricity and we have to pay for electricity board a lot. A system will be economically feasible if it is cost efficient along with its accuracy. Our system is cost efficient because of the tools that we used to build it. The device cost is very low which is efficient for a developing country.

##### ❖ OPERATIONAL FEASIBILITY

When compared with the existing methods the proposed system is not as complex as the existing methods because no manual operations are carried out. All the equipment will be controlled from a pc by software. No manual attention is needed. This involves questions such as whether the technology needed for the system exists, how difficult it will be to build, and whether the firm has enough experience using that technology. The assessment is based on an outline design of system requirements in terms of Input, Processes, Output, Fields, Programs, and Procedures. This can be qualified in terms of volumes of data, trends, frequency of updating in order to give an introduction to the technical system.

##### ❖ TECHNICAL FEASIBILITY

The existing methods must have a trained person to operate that system any one cannot operate the easily. If problem comes to user end it is not easy to solve. In our system it is very easy to operate and ordinary person who know to operate a pc can operate the software very easily for

monitoring purpose. For the device, we used Pulse sensor, LCD, Arduino uno , etc. These are the components that we used to build the device. We used these components because they gave almost accurate output.

### ❖ SCHEDULES FEASIBILITY

In this type of feasibility, the skills required for properly applying the new technology with training in minimum time and the time duration can be checked out to implement or overrun the new project within minimum time. Schedule feasibility is determined as the probability of a project to be carried out within its scheduled time limits. In many cases, a project will be unsuccessful if it takes more than it was estimated.

### 2.7 Cost benefit Analysis (cost estimation):

There are different types of health equipment in the market but there is no such device as ours. In developed countries there are few heart monitoring devices which are very expensive. Our purpose is to make a device which is suitable for developing countries. To build the device we used Arduino UNO, Pulse sensor, LCD, Breadboard, Connecting wires, Buzzer. Total cost will be around 1,500/- to 2,000/-.

Components	Price
Arduino Uno or Nano	300 - 430/-
OLCD display	250 - 419/-
Pulse Sensor	259/-
Breadboard	125/-
Connecting wires	90/-
Piezo buzzer	15/-
Arduino Ethernet Shield & Ethernet cable	489/-
<b>Total</b>	1827/-

This is not just a device. There are also application which is for free (Arduino IDE, Xampp server &MySQL database server/phpMyAdmin). The patient can be monitored by doctor constantly.

### 2.8 Requirement Specification:

Requirement specification is the first and important phase of the software developing activity in developing any kind of project effectively. I started to list out all the functionalities that my application should provide. There have been some minor changes with respect to the functionalities over the course of development. The following are the requirements that have been implemented in this project.

#### 2.8.1 Functional Requirements

- Application must have a module for storing the data in the database.
- Tracking data: Application must have track option with which doctor or guardian can track the patient's data using LCD.
- Sender data: Data must be send to the database using hardware components

#### 2.8.2 Non-Functional Requirements

Non-functional requirements are not directly related to the functional behavior of the system.

- Application must be user friendly, simple and interactive.
- The user interface is designed in such way that novice users with little knowledge of web, should be able to access this application.
- Users are required to have some knowledge regarding hardware components.

#### 2.8.3 Software Specifications

- Operating System: Windows 7 or higher
- Platform: IoT Cloud
- IDE: Arduino 1.8.9
- Database: MySQL database server/phpMyAdmin).
- Technologies used: C, SQL, Xampp

#### 2.8.4 Hardware Specifications

- Microcontroller: Arduino Uno or Nano Board
- Sensors: Heart Beat Sensor
- OLCD Display (128x64 (SSD1306 Driver))
- Arduino Ethernet Shield
- Ethernet cable
- Breadboard
- Jumper Wires (Male to Female, Female to Female, Male to Male)
- Piezo Buzzer
- Processor: Pentium IV or higher

## Heart Monitoring System

- Processor speed: 1.6GHz
- RAM: 512 MB
- Disk Space: 250 MB or higher

### 2.9 Tools & Technology:

- **Front-End:** C programming

C is a general-purpose, procedural computer programming language supporting structured programming, lexical variable scope, and recursion, while a static type system prevents unintended operations. By design, C provides constructs that map efficiently to typical machine instructions, and has found lasting use in applications previously coded in assembly language. Such applications include operating systems, as well as various application software for computers ranging from supercomputers to embedded systems.

- **Back-End:** MySQL database server/phpMyAdmin)

We have used MySQL database management system for our project. The reason to use of MySQL database is because it is open source, widely used and most popular SQL database management system which is distributed, developed and supported by Oracle Corporation. Again, another reason to choose MySQL is it supports relational database. Therefore it is very flexible to use since we can put information in different table rather than to put all information in

one table. phpMyAdmin is a free and open source administration tool for MySQL and MariaDB. As a portable web application written primarily in PHP, it has become one of the most popular MySQL administration tools, especially for web hosting services.

- **Arduino IDE**

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. The Arduino development environment contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions, and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them

## Chapter 3: Design Phase

### 3.1 Detailed life cycle of the project (Logical Design):

#### 3.1.1 The Spiral Model

- ☐ The spiral model is a risk driven process model generator for software project.
- ☐ This provides support for risk handling.
- ☐ It is an iterative model.
- ☐ It allows incremental releases of product or incremental refinement through each iteration around the spiral.

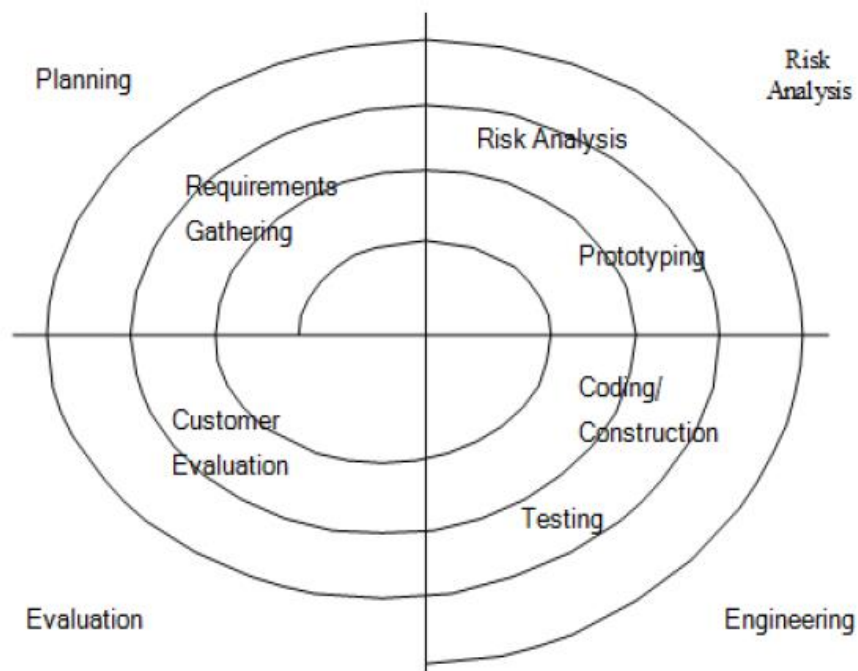


Fig. Spiral Model

#### 3.2.1 Phases of spiral model are:

1. Identification
2. Design
3. Construct or Build
4. Evaluation and risk analysis

## 1. Identification:

- This phase consists of gathering the requirements for this project.
- The requirements are the hardware components, internet and the thing is the Pulse sensor.
- Other requirements for an IOT design is attractive design of the model to sell it more easily.

## 2. Design:

To design the project there are following things to be taken under consideration:

### I. Small size and portable in nature:

- The product must acquire small space and should be easy to carry around.

### II. Attractive design:

- The design must be unique and attractive to attract the customer towards it.

## 3. Construction:

- The arrangements of components altogether starts in this phase
- The plot of burning the code on the microcontroller is in this part.
- For construction purpose various Arduino APIs are to be taken under consideration
- The arrangement of components is according to the design of the product.
- The product is tested on various basis.

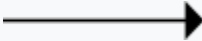



## 4. Evaluation and risk analysis:


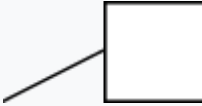



- Risk analysis includes identifying, estimating and monitoring technical feasibility and management risk such as schedule slippage and cost overrun.
- Risks are the component cost going out of budget etc.
- In evaluation, we will test all the aspects of systems and if there is any scope for development then we will do it.



### 3.2 Flow Chart:

#### Basic Flow chart Symbols and Notations: -

ANSI/ISO Shape	Name	Description
	Flowline (Arrowhead)	Shows the process's order of operation. A line coming from one symbol and pointing at another. Arrowheads are added if the flow is not the standard top-to-bottom, left-to right.
	Terminal	Indicates the beginning and ending of a program or sub-process. Represented as a <u>stadium</u> , oval or rounded (fillet) rectangle. They usually contain the word "Start" or "End", or another phrase signaling the start or end of a process, such as "submit inquiry" or "receive product".
	Process	Represents a set of operations that changes value, form, or location of data. Represented as a <u>rectangle</u> .
	Decision	Shows a conditional operation that determines which one of the two paths the program will take. The operation is commonly a yes/no question or true/false test. Represented as a diamond (rhombus).

	Input/Output	Indicates the process of inputting and outputting data, <sup>[15]</sup> as in entering data or displaying results. Represented as a <u>parallelogram</u> .
	Annotation(Comment)	Indicating additional information about a step the program. Represented as an open rectangle with a dashed or solid line connecting it to the corresponding symbol in the flowchart.
	Predefined Process	Shows named process which is defined elsewhere. Represented as a rectangle with double-struck vertical edges.
	On-page Connector	Pairs of labeled connectors replace long or confusing lines on a flowchart page. Represented by a small circle with a letter inside.
	Off-page Connector	A labeled connector for use when the target is on another page. Represented as a <u>home plate-shaped pentagon</u> .

## Heart Monitoring System

The flow diagram of the project is shown in figure, the sensors value are read and displayed on the LCD monitor and stored in the cloud for future use.

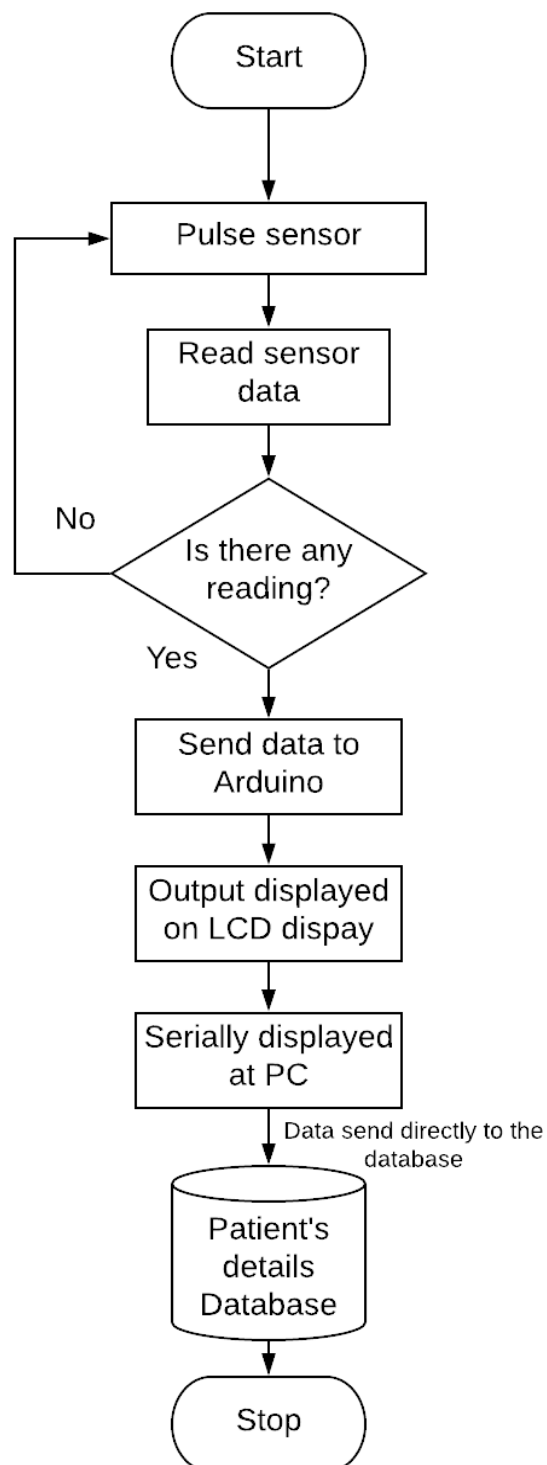


Figure 1: -Flow Diagram for HMS

### 3.3 Class Diagram:

The UML Class diagram is a graphical notation used to construct and visualize object oriented systems. 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:

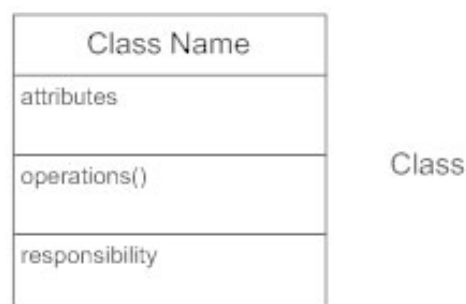
- classes,
- their attributes,
- operations (or methods),
- and the relationships among objects.

#### Basic Class Diagram Symbols and Notations: -

##### Classes: -

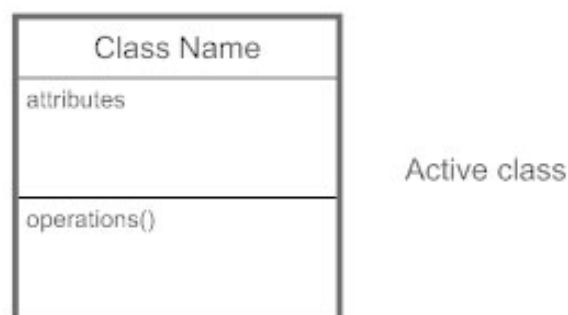
Classes represent an abstraction of entities with common characteristics. Associations represent the relationships between classes.

Illustrate classes with rectangles divided into compartments. Place the name of the class in the first partition (centered, bolded, and capitalized), list the attributes in the second partition (left-aligned, not bolded, and lowercase), and write operations into the third. (left-aligned, not bolded, and lowercase), and write operations into the third.



##### Active Classes: -

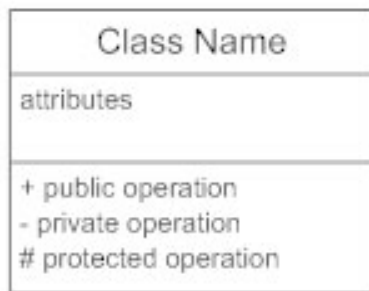
Active classes initiate and control the flow of activity, while passive classes store data and serve other classes. Illustrate active classes with a thicker border.



## Heart Monitoring System

### Visibility: -

Use visibility markers to signify who can access the information contained within a class. Private visibility, denoted with a - sign, hides information from anything outside the class partition. Public visibility, denoted with a + sign, allows all other classes to view the marked information. Protected visibility, denoted with a # sign, allows child classes to access information they inherited from a parent class.

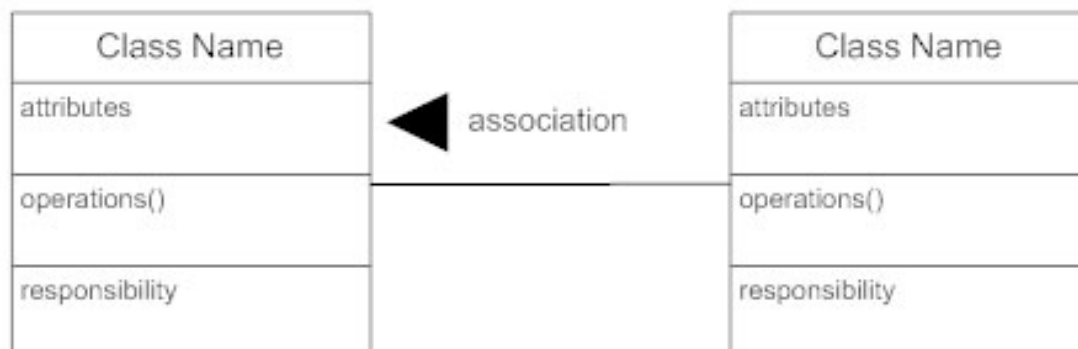


Visibility

Marker	Visibility
+	public
-	private
#	protected
~	package

### Associations: -

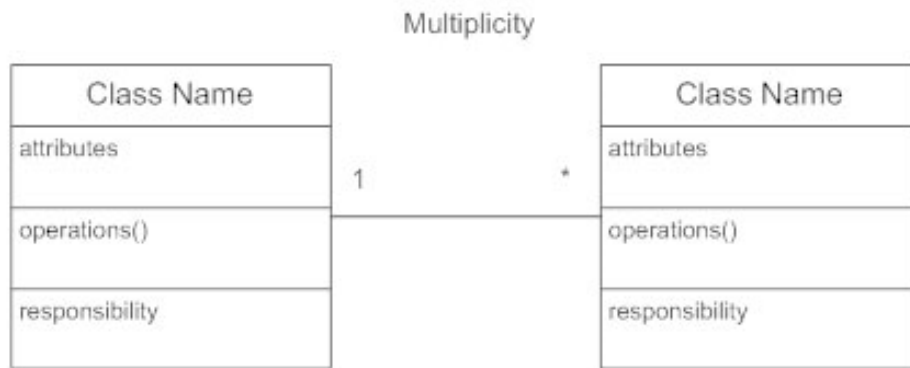
Associations represent static relationships between classes. Place association names above, on, or below the association line. Use a filled arrow to indicate the direction of the relationship. Place roles near the end of an association. Roles represent the way the two classes see each other.



### Multiplicity (Cardinality): -

Place multiplicity notations near the ends of an association. These symbols indicate the number of instances of one class linked to one instance of the other class. For example, one company will have one or more employees, but each employee works for just one company.

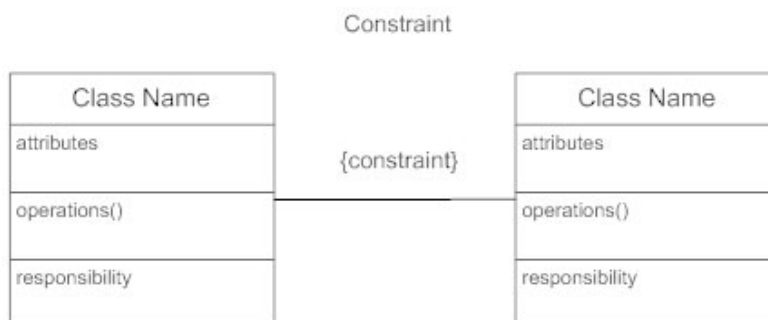
## Heart Monitoring System



Indicator	Meaning
0..1	Zero or one
1	One only
0..*	0 or more
1..*   *	1 or more
$\underline{n}$	Only $\underline{n}$ (where $\underline{n} > 1$ )
0.. $\underline{n}$	Zero to $\underline{n}$ (where $\underline{n} > 1$ )
1.. $\underline{n}$	One to $\underline{n}$ (where $\underline{n} > 1$ )

### Constraint: -

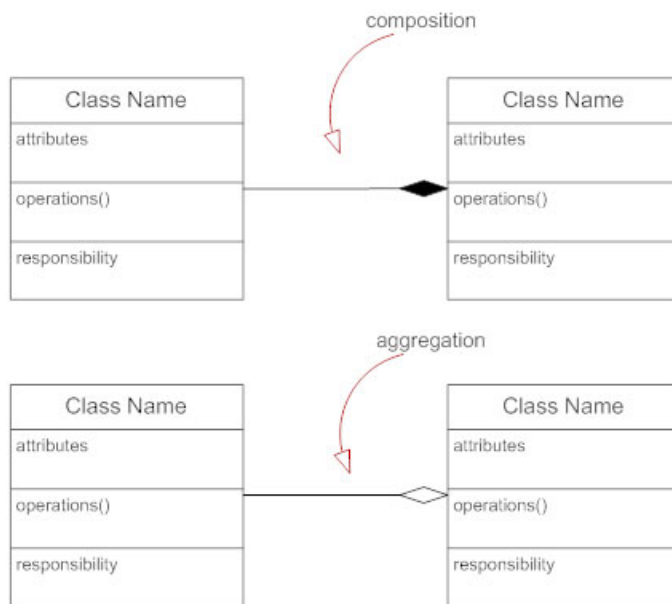
Place constraints inside curly braces { }.



### Composition and Aggregation: -

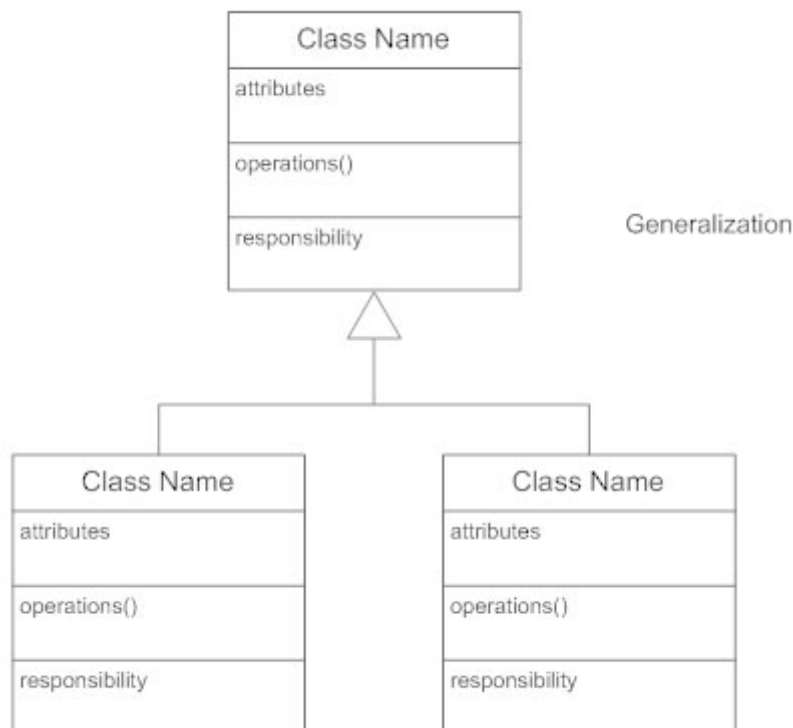
Composition is a special type of aggregation that denotes a strong ownership between Class A, the whole, and Class B, its part. Illustrate composition with a filled diamond. Use a hollow diamond to represent a simple aggregation relationship, in which the "whole" class plays a more important role than the "part" class, but the two classes are not dependent on each other. The diamond ends in both composition and aggregation relationships point toward the "whole" class (i.e., the aggregation).

Composition and Aggregation



### Generalization: -

Generalization is another name for inheritance or an "is a" relationship. It refers to a relationship between two classes where one class is a specialized version of another. For example, Honda is a type of car. So the class Honda would have a generalization relationship with the class car.



## Heart Monitoring System

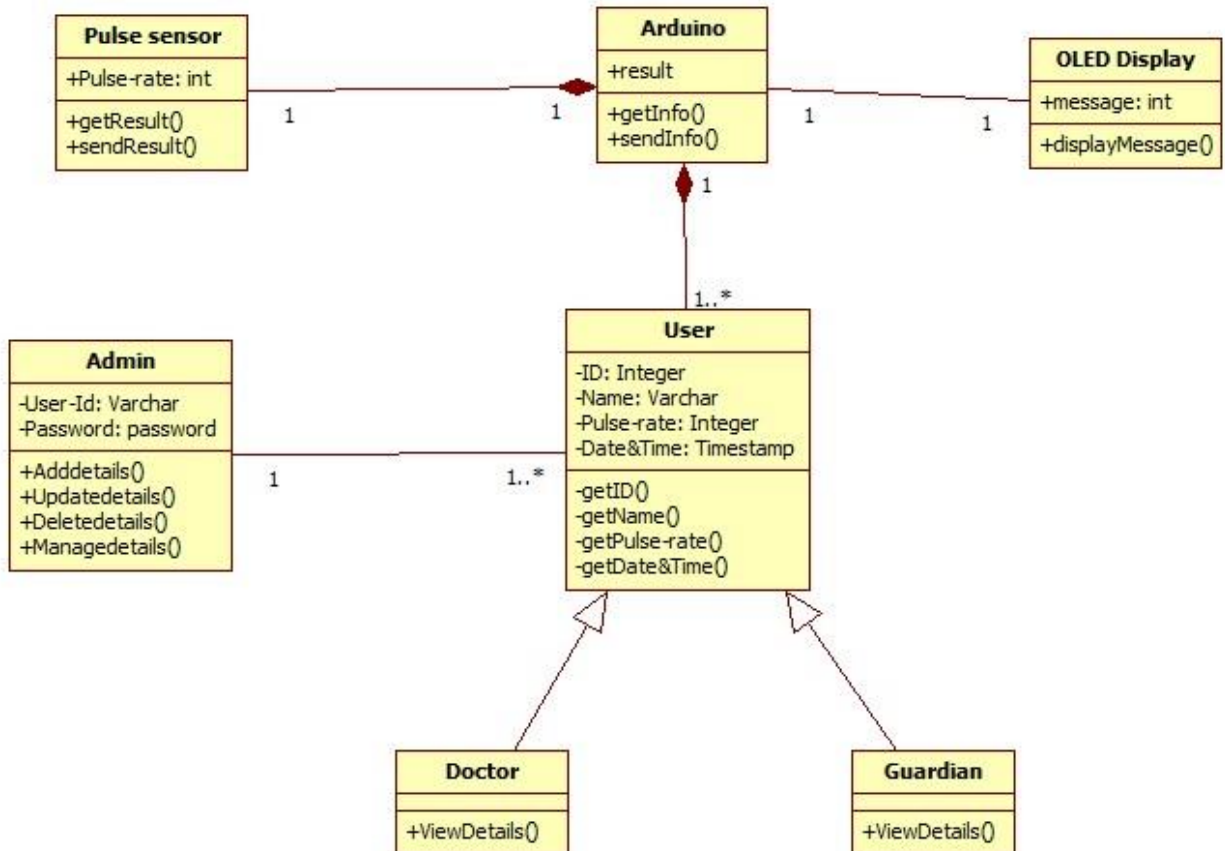


Figure 2: -Class diagram for HMS

The following diagram is an example of a heart monitoring system's database. It describes a particular aspect of the entire application.

- First of all, Arduino and pulse sensor have a one-to-one relationship between each other.
- Arduino and LCD display also have one to one relationship between each other.
- Admin and User are identified as the two elements of the system. They have a one-to-many relationship because an Admin can have multiple User but User have only one admin.
- User class is an abstract class and it has two concrete classes (Generalization relationship) Doctor and Guardian.

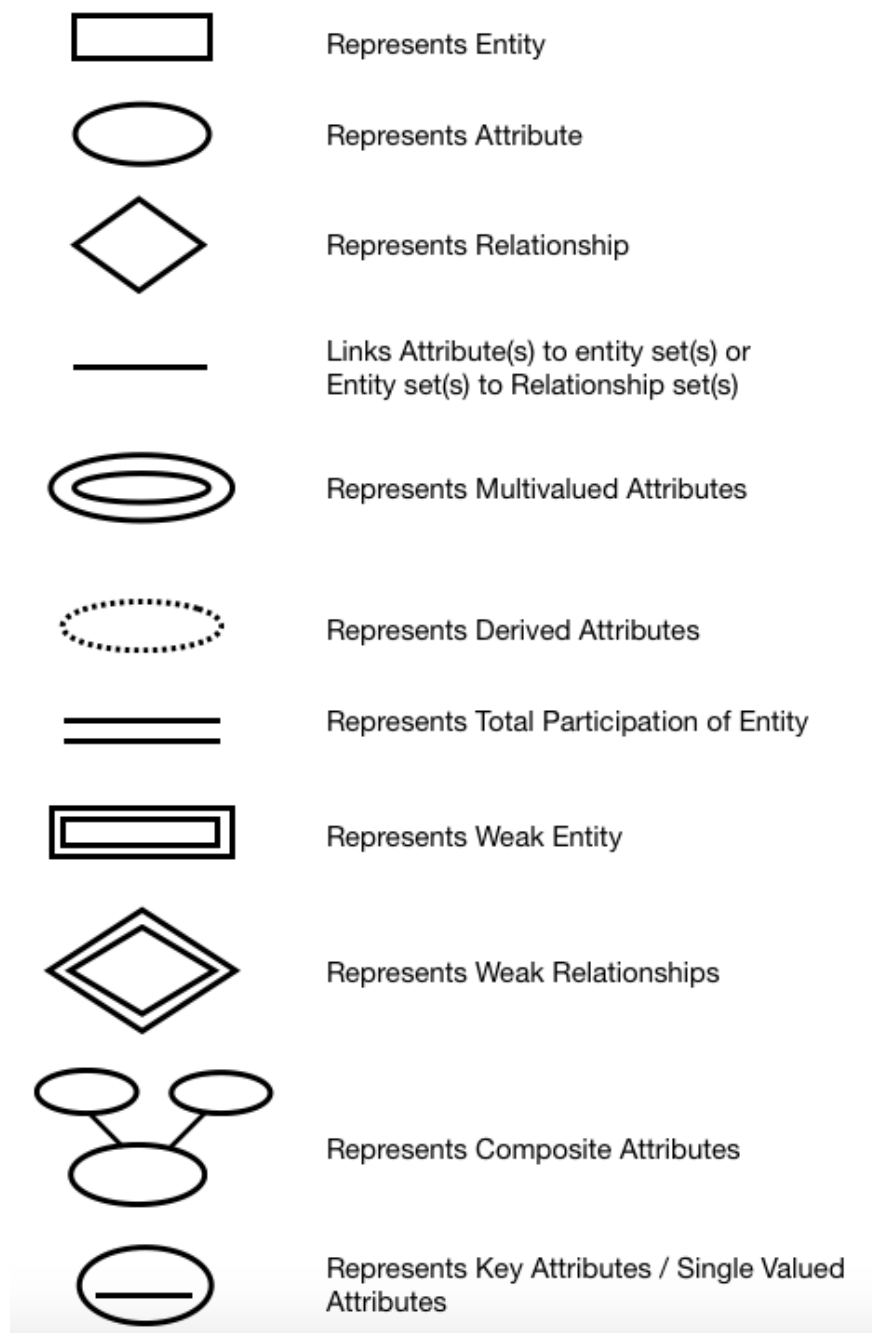


### 3.4 E-R Diagram:

Entity Relationship Diagram, also known as ERD, ER Diagram or ER model, is a type of structural diagram for use in database design. An ERD contains different symbols and connectors that visualize two important information: **The major entities within the system scope**, and the **inter-relationships among these entities**.

Different notations used for ER Diagram in DBMS –

Below picture clearly depicts about different notations we need while using ER diagram in DBMS. They are –



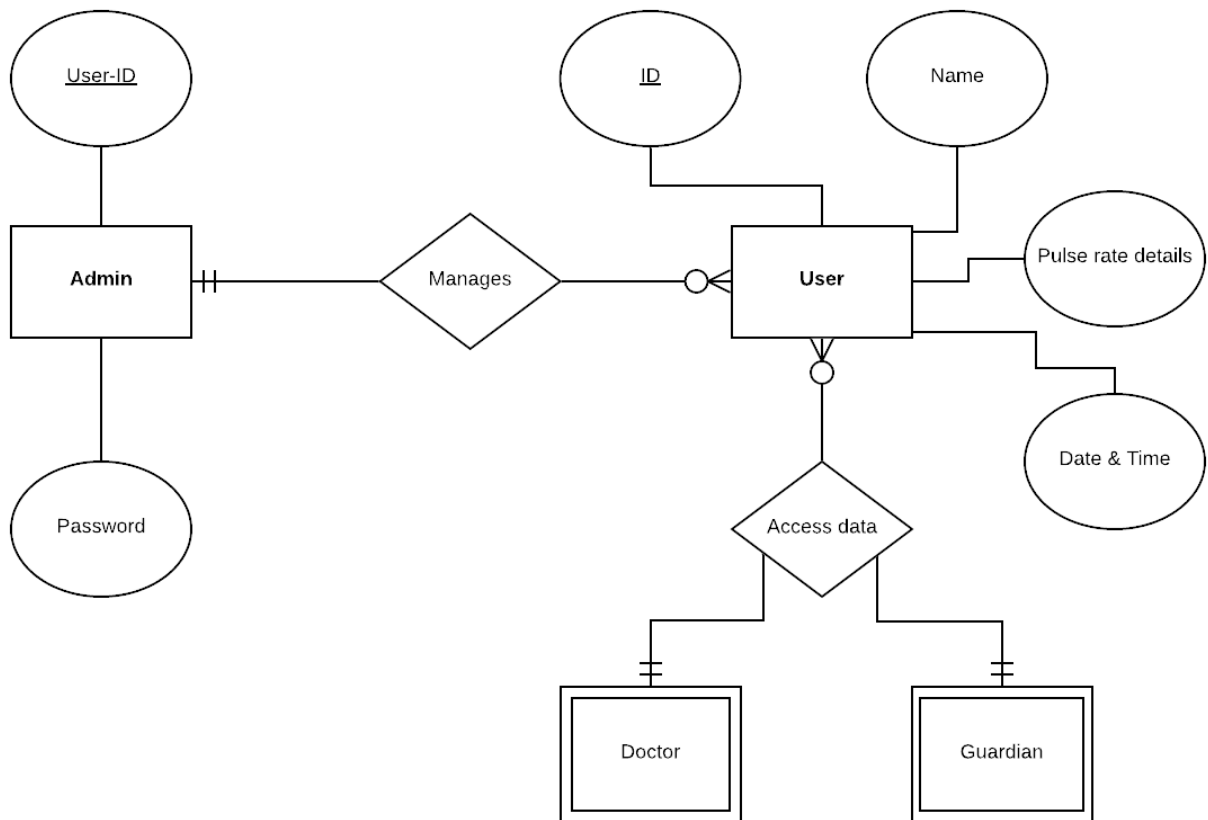


Figure 3: -E-R Diagram for HMS

### 3.5 DFD:

A **data-flow diagram** (DFD) is a way of representing a flow of a data of a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow, there are no decision rules and no loops. Specific operations based on the data can be represented by a flowchart.

#### ➤ 0 Level DFD – Heart Monitoring System

The 0 level dfd known as **context level** data flow diagram. The context level data flow diagram (dfd) is describe the whole system. The (o) level dfd describe the all user modules who run the system.

Below context level data flow diagram of heart monitoring system project shows the one Admin user can operate the system. Admin do all activities after login to system.

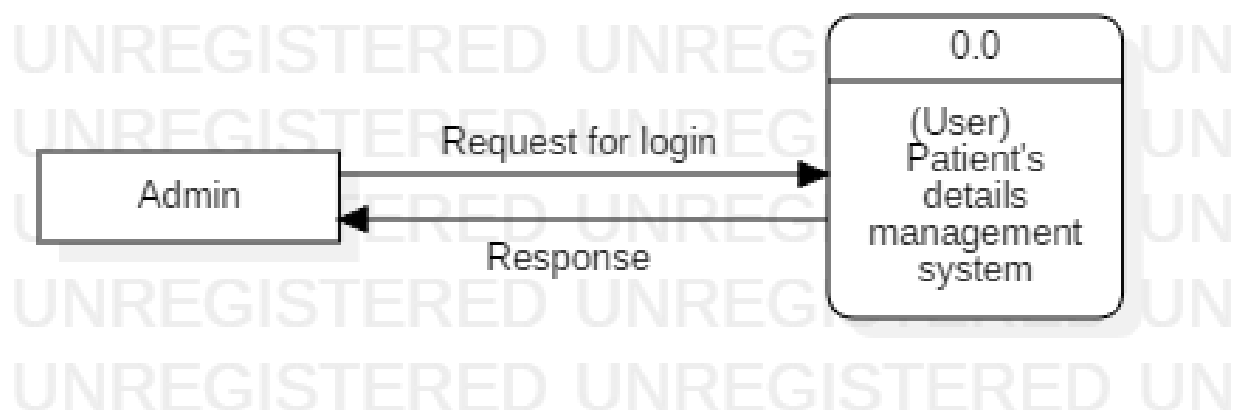


Figure 4: -Context Diagram for HMS

#### ➤ 1st level DFD – Heart Monitoring System

The Admin side DFD describe the functionality of Admin. Admin is a responsible person who run the project. After login to system admin can first Add Detail and then admin can manage Patient's reports and edit patient's details.

## Heart Monitoring System

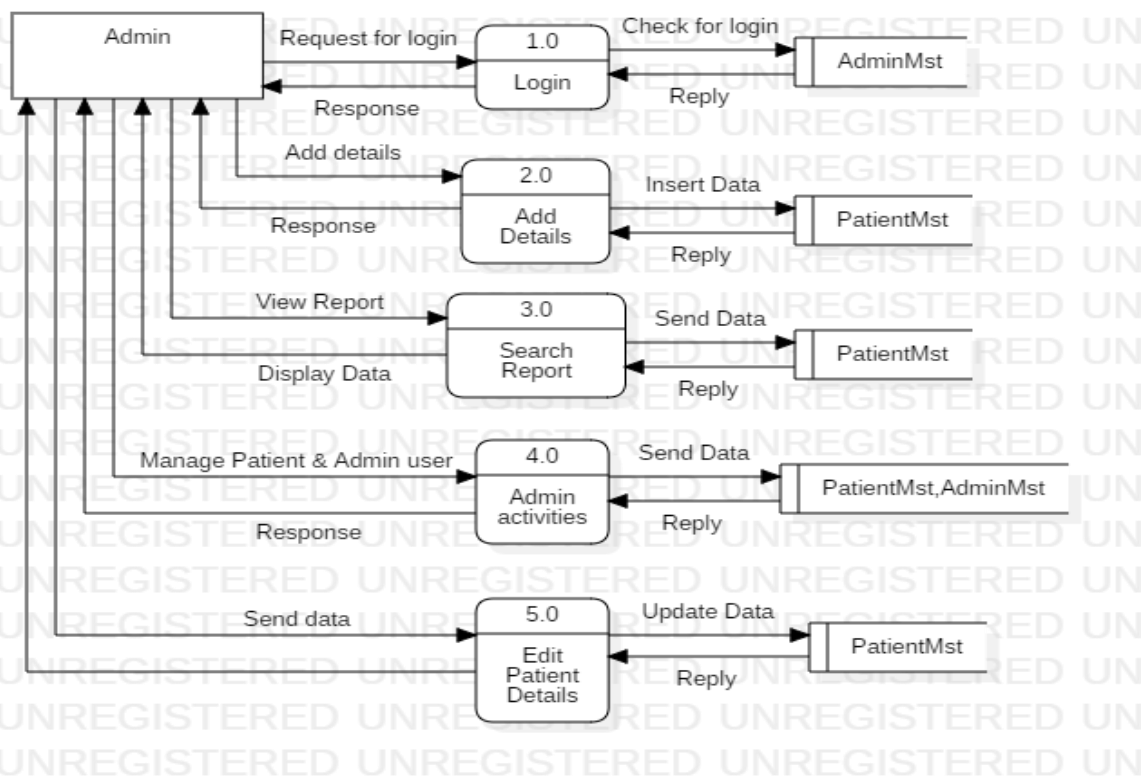


Figure 5: -1st level DFD diagram for HMS

### ➤ 2nd level DFD – Heart Monitoring System

2-level DFD goes one step deeper into parts of 1-level DFD. It can be used to plan or record the specific/necessary detail about the system's functioning.

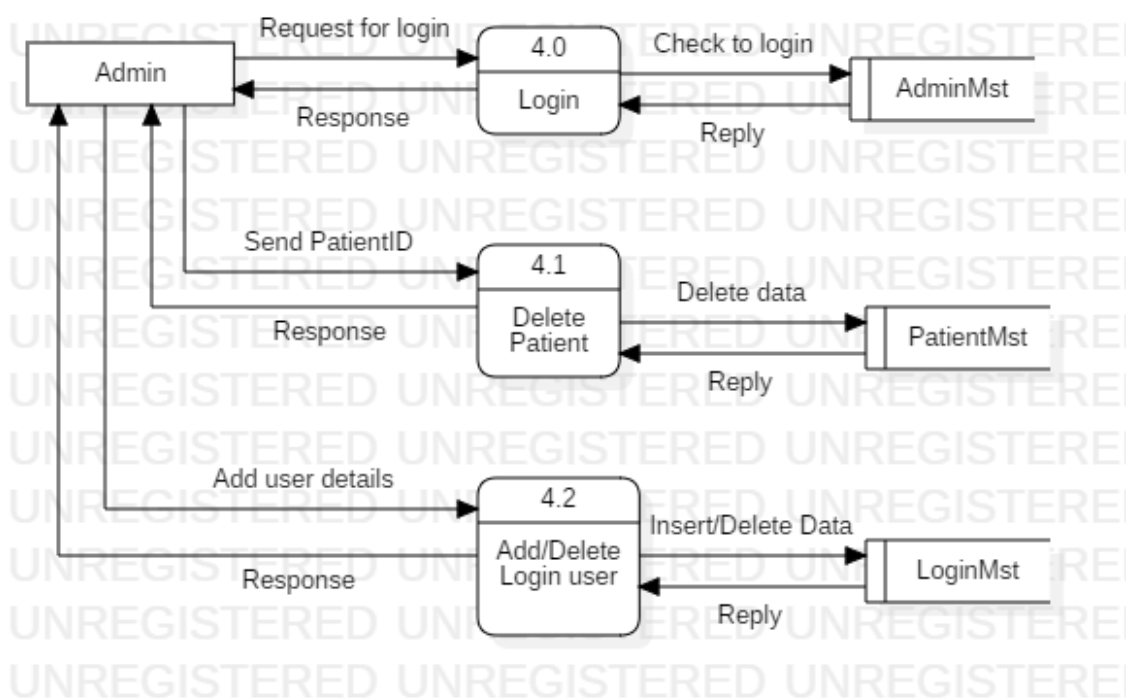



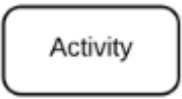

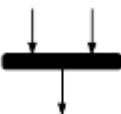

Figure 6: -2nd level DFD diagram for HMS









### 3.6 Activity Diagram:

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. It is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc.

These activity diagram shapes and symbols are some of the most common types you'll find in UML diagrams.

Symbol	Name	Description
	Start symbol	Represents the beginning of a process or workflow in an activity diagram. It can be used by itself or with a note symbol that explains the starting point.
	Activity symbol	Indicates the activities that make up a modeled process. These symbols, which include short descriptions within the shape, are the main building blocks of an activity diagram.
	Connector symbol	Shows the directional flow, or control flow, of the activity. An incoming arrow starts a step of an activity; once the step is completed, the flow continues with the outgoing arrow.
	Joint symbol/ Synchronization bar	Combines two concurrent activities and re-introduces them to a flow where only one activity occurs at a time. Represented with a thick vertical or horizontal line.
	Fork symbol	Splits a single activity flow into two concurrent activities. Symbolized with multiple arrowed lines from a join.

	Decision symbol	Represents a decision and always has at least two paths branching out with condition text to allow users to view options. This symbol represents the branching or merging of various flows with the symbol acting as a frame or container.
	Note symbol	Allows the diagram creators or collaborators to communicate additional messages that don't fit within the diagram itself. Leave notes for added clarity and specification.
	Send signal symbol	Indicates that a signal is being sent to a receiving activity.
	Receive signal symbol	Demonstrates the acceptance of an event. After the event is received, the flow that comes from this action is completed.
	Shallow history pseudostate symbol	Represents a transition that invokes the last active state.
	Option loop symbol	Allows the creator to model a repetitive sequence within the option loop symbol.
	Flow final symbol	Represents the end of a specific process flow. This symbol shouldn't represent the end of all flows in an activity; in that instance, you would use the end symbol. The flow final symbol should be placed at the end of a process in a single activity flow.
[Condition]	Condition text	Placed next to a decision marker to let you know under what condition an activity flow should split off in that direction.
	End symbol	Marks the end state of an activity and represents the completion of all flows of a process.

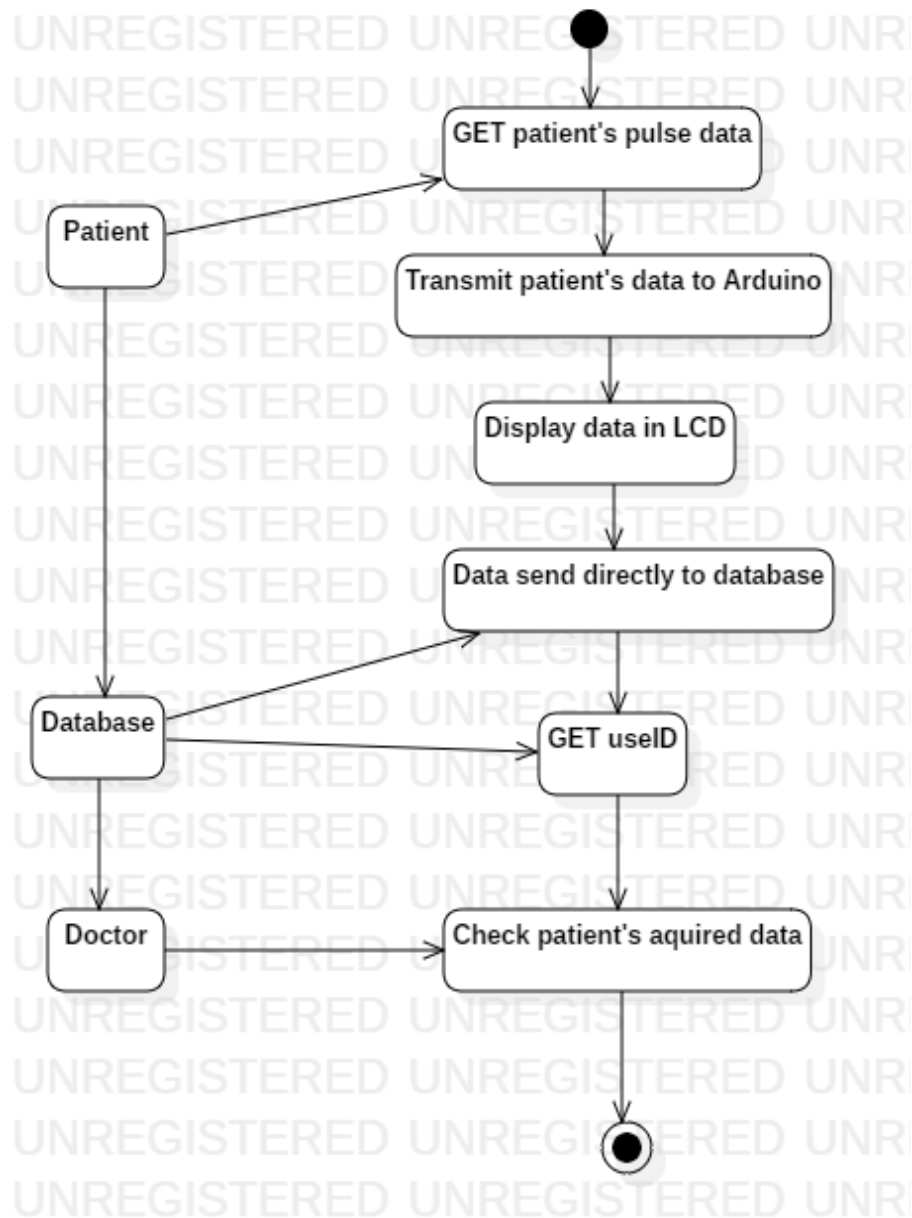


Figure 7: -Activity diagram for HMS

### 3.7 Use-case diagram:

#### System

Draw your system's boundaries using a rectangle that contains use cases. Place actors outside the system's boundaries.



#### Use Case

Draw use cases using ovals. Label the ovals with verbs that represent the system's functions.



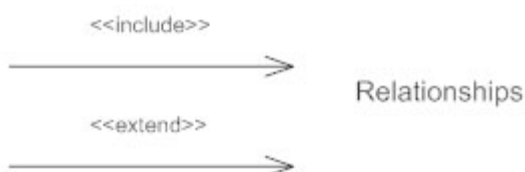
#### Actors

Actors are the users of a system. When one system is the actor of another system, label the actor system with the actor stereotype.



#### Relationships

Illustrate relationships between an actor and a use case with a simple line. For relationships among use cases, use arrows labeled either "uses" or "extends." A "uses" relationship indicates that one use case is needed by another in order to perform a task. An "extends" relationship indicates alternative options under a certain use case.





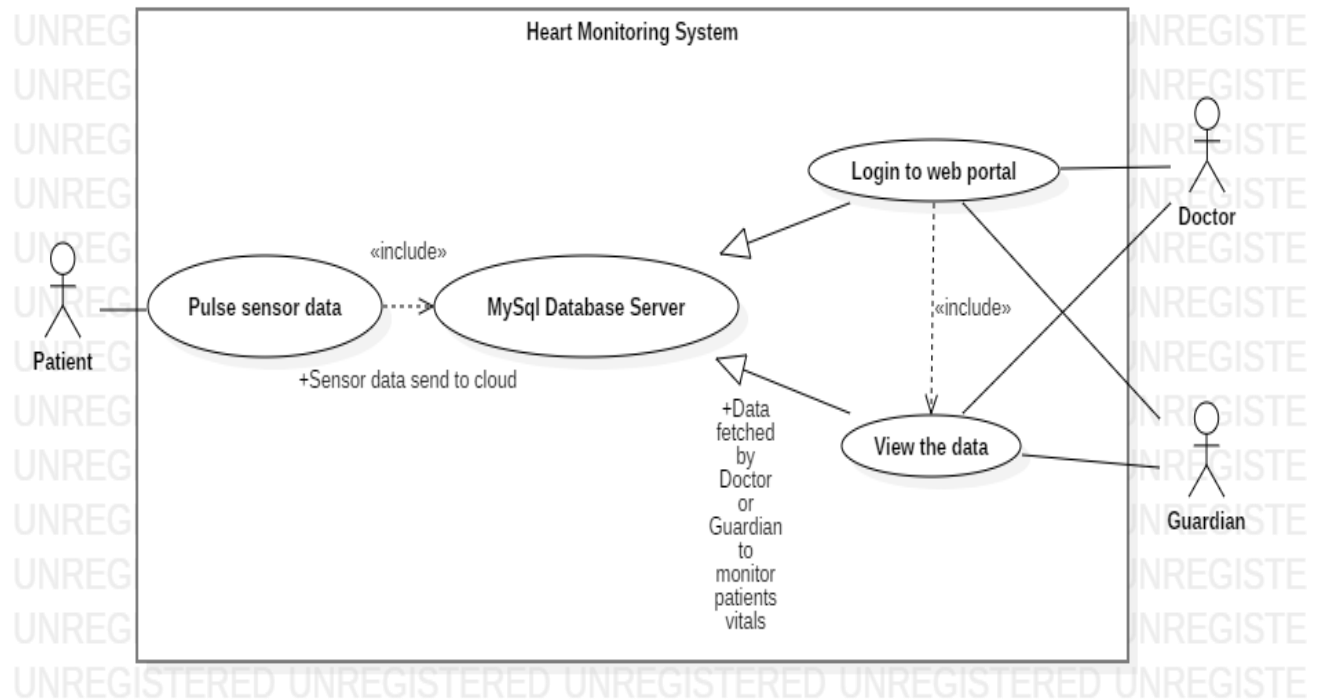


Figure 8: -Use Case Diagram for HMS

### Actors:

The Actors of the system are Patient, Guardian and Doctor

### Use cases:

I have identified a set of use cases based on the functionalities and goals of the application.

- **Login**- This use case denotes a set of actions required for Subject to login into the application (default username & password is root & toor).
- **View data**-This use case denotes a set of actions required by Guardian or Doctor to view the data.

### 3.8 Database Relationship diagram:

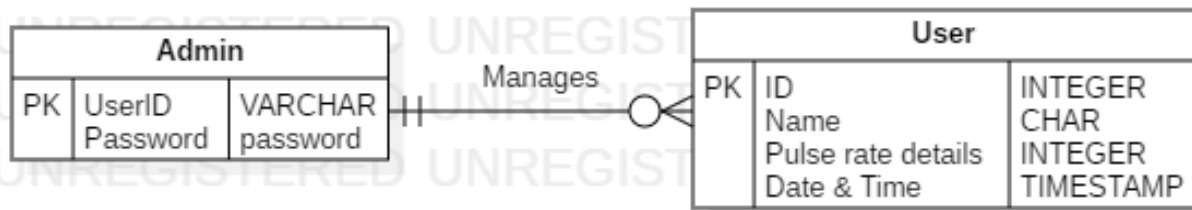


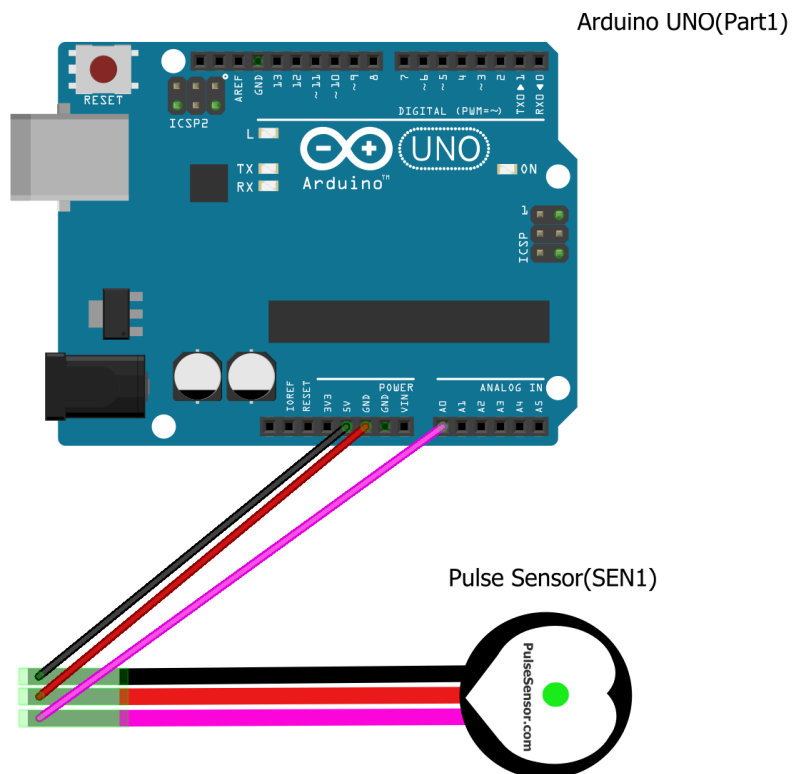
Figure 9: -Database Relationship diagram for HMS

### 3.9 Event Table:

Sr. No	Event	Trigger	Initiator	Output	Recipients
1.	Read pulse sensor data	Input from Pulse sensor	User	Notification on LCD	User
2.	Display pulse sensor data	Output in OLED display	User	Notification on LCD	User
3.	Admin login	Entering credentials on login screen on phpmyadmin	Admin	Login to system	Admin
4.	Keeps Track of details	Input on database	Admin	Shows the data on database	Admin

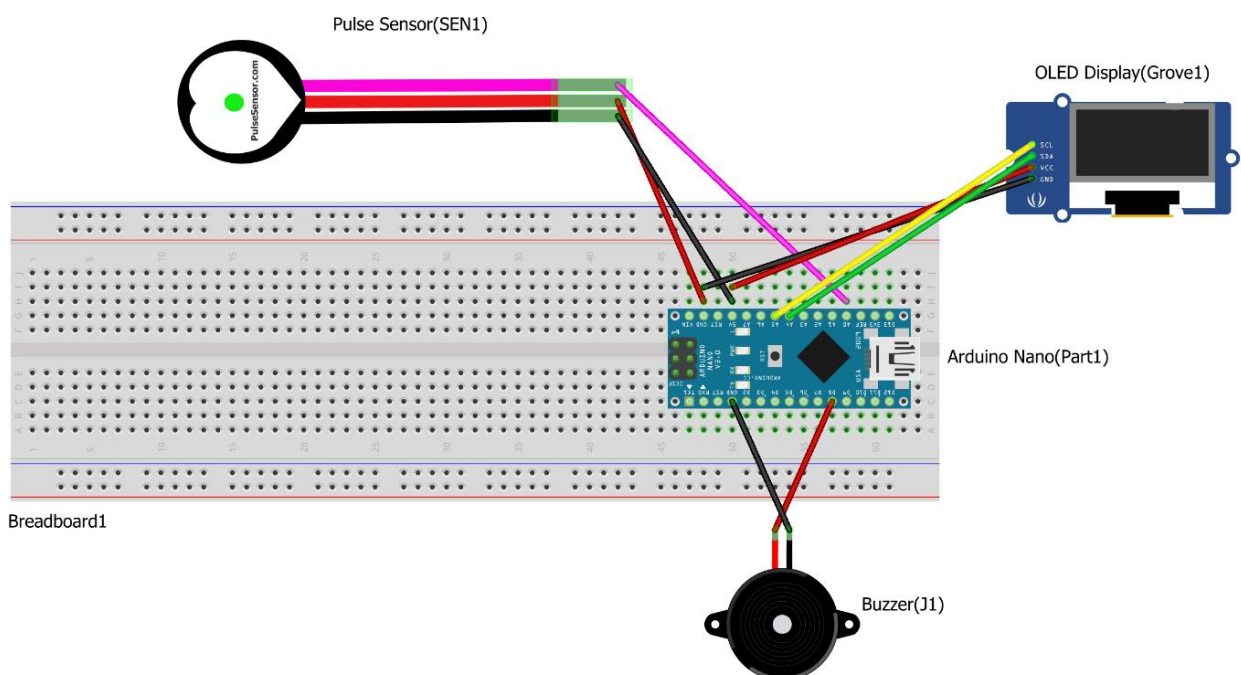
## Chapter 4: GUI Design

### 1] Simple GUI design with Pulse sensor and Arduino: -



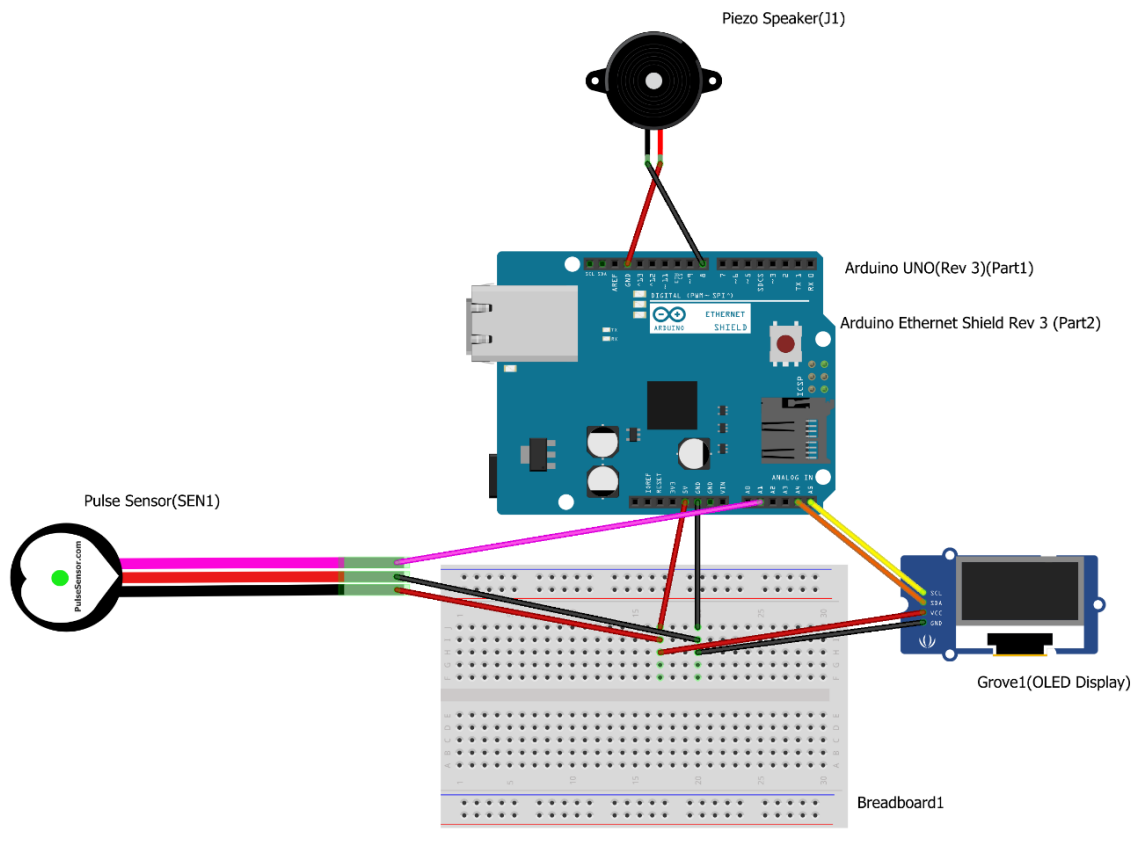
fritzing

### 2] GUI Design with Arduino Nano: -



fritzing

### 3]GUI Design for Arduino with Ethernet shield:

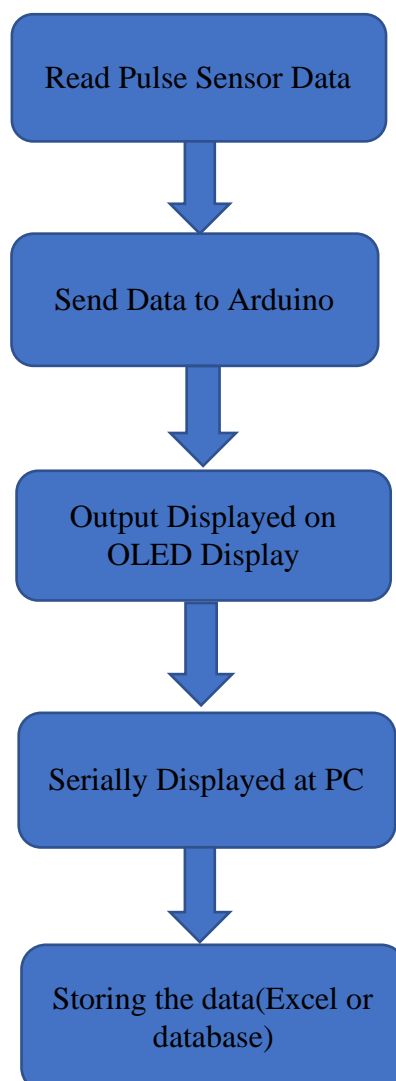


## Chapter 5: Implementation and Testing

### 5.1 Implementation: -

This project has been developed with Arduino microcontroller connected with sensors which are attached to the patient. If there is any critical condition, the concerned caregivers will be given to the buzzer alert and data are displayed in LCD at a same time. All the sensors and data sent from microcontroller to different storage device. A doctor or guardian can view or monitor patient's data at any point in time.

At any point of time either a doctor or guardian can log into the patients data which would help medical services to send appropriate help in case of emergencies.



## Heart Monitoring System

### How they work: -

As can be seen there are only three wires to these devices. One is power, the other ground and the final one is the analogue signal. So, these are not digital devices like the temperature sensor or the many other sensors available. This device outputs a voltage from 0V to VDD (supply voltage) depending on what they “sense”. They work by emitting an Infra-Red signal from an IR-Diode onto your skin. Just underneath your skin are capillaries carrying blood. Every time your heart pumps there is an small increase in blood flow/pressure. This swells the capillaries slightly, this slightly more filled capillary reflects more infra-red than at times when the heart is not giving your blood a “push”. An Infra-detector on the device senses the different reflected IR levels. Some simple comparator circuitry converts this into a voltage signal which we can read with the Arduino’s analogue inputs.

### Connecting it up and initial tests: -

Any micro-controller with an analogue input should work as the device operates from 3-6V range (maybe even higher). Just connect the signal line (labelled “S” on my device) to an analogue input pin. The value you read from the pin will be in the range of zero to whatever the top value is on your particular micro-controller. For the normal default Arduino the range is from 0-1023 (1024 values). Here is the connection diagram showing just the Heart Beat connections.

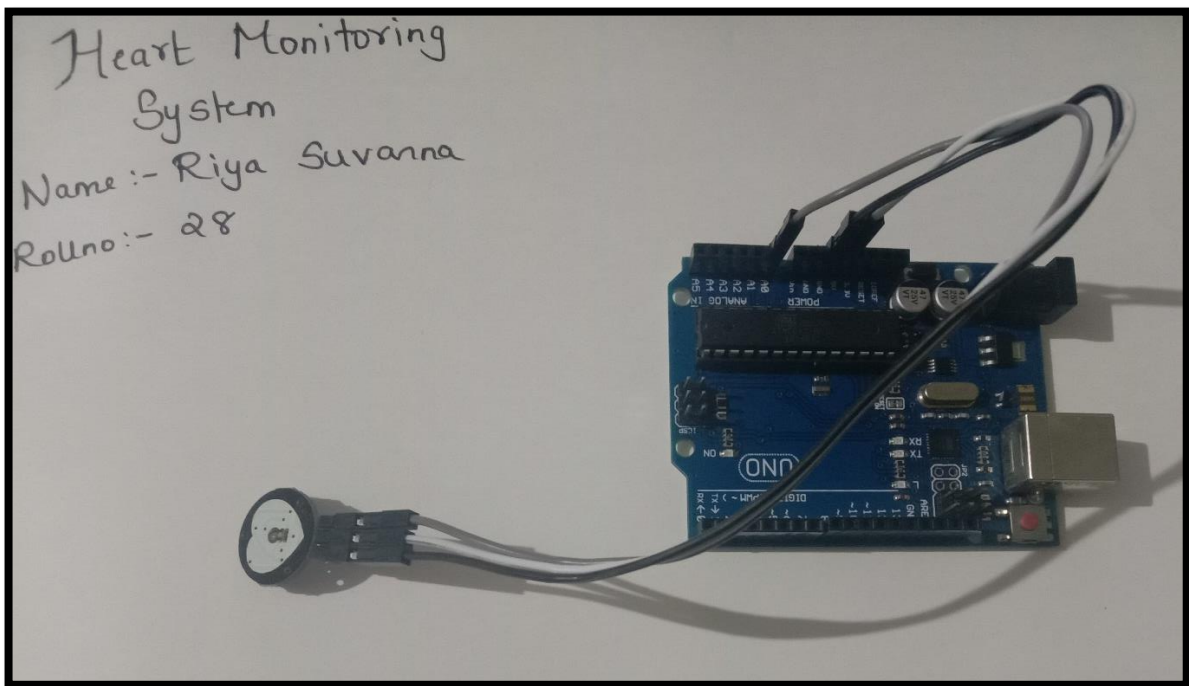


Figure: Pulse Sensor Connected with Arduino

## Heart Monitoring System

This shows :

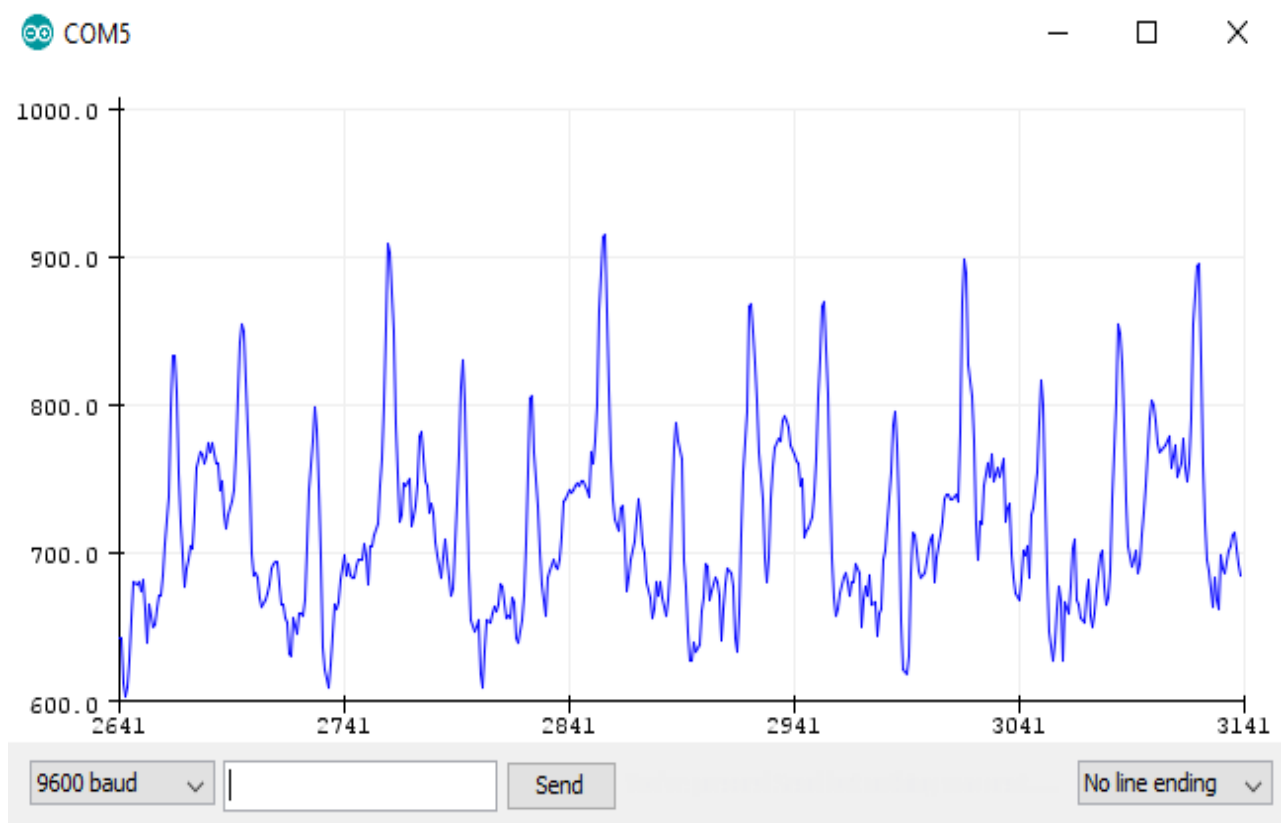
Pulse Sensor “+” to +5v on Arduino

Pulse Sensor “-” to Gnd on Arduino

Pulse Sensor “S” to A0 on Arduino

To show your heart beat graphically you can use the Arduino IDE in-built serial monitor plotter. Load up the code below into your IDE, then upload to your Arduino. Then select “Tools->Serial Plotter”

Here’s my screen shot showing you what to expect. The Arduino Serial Plotter plots a value sent to it on a graph against time.



### Adding the OLED 128×64 (SSD1306 Driver) display

The in built plotter is great for a quick play but it doesn’t show you your heart rate in BPM (Beats Per Minute) at the same time. So for this project we’re adding in our own OLED display. These are readily available and the set up has been discussed in [this article](#) before, but we’ll go through it again here. Firstly ensure you’ve bought a OLED 128×63 I²C display (SSD1306 driver) display. It should have four connections, 5v, Gnd, SDA and SCK. Connect up as shown above.



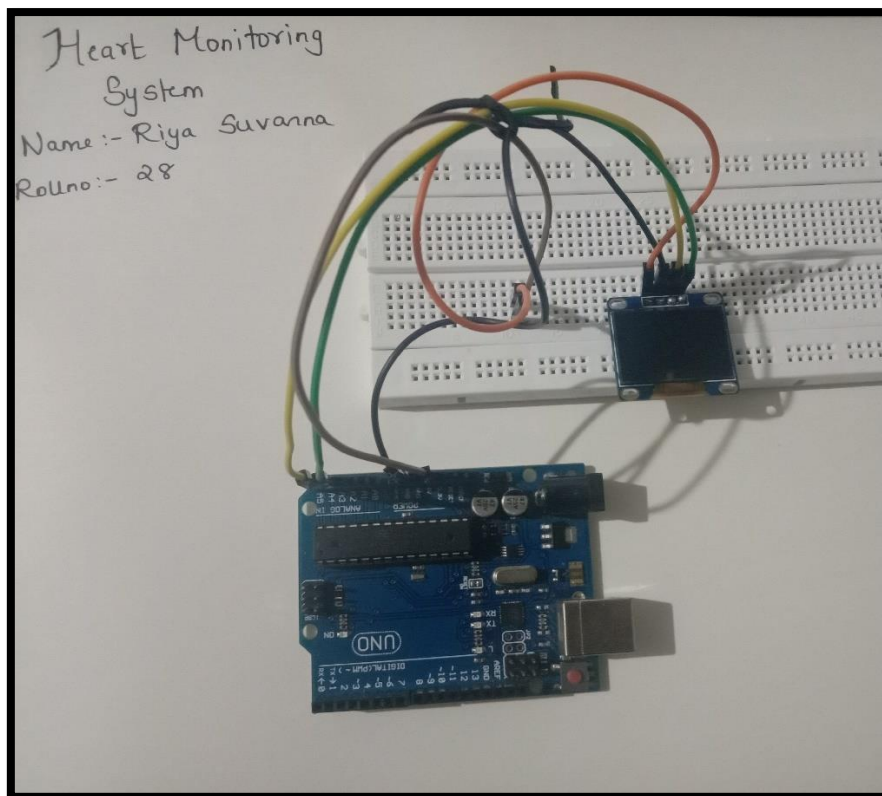


Figure: OLED display with Arduino

### Installing the OLED display libraries

The first is the driver for 1306 OLED display driver chip. Go to the Arduino IDE library manager, “Sketch->Include Library->Manage Libraries”. Type “Adafruit SSD1306” into the search field. The first result should be the one you require titled “Adafruit SSD1306 by Adafruit”. Install this. The second library is the “Adafruit GFX library”, type in “adafruit graphics” and install the graphics library. The final code is below, copy and paste into your Arduino IDE. Upload and you should see a trace going across the screen. Gently rest your finger on the sensor and your trace should appear (if not see below after code).

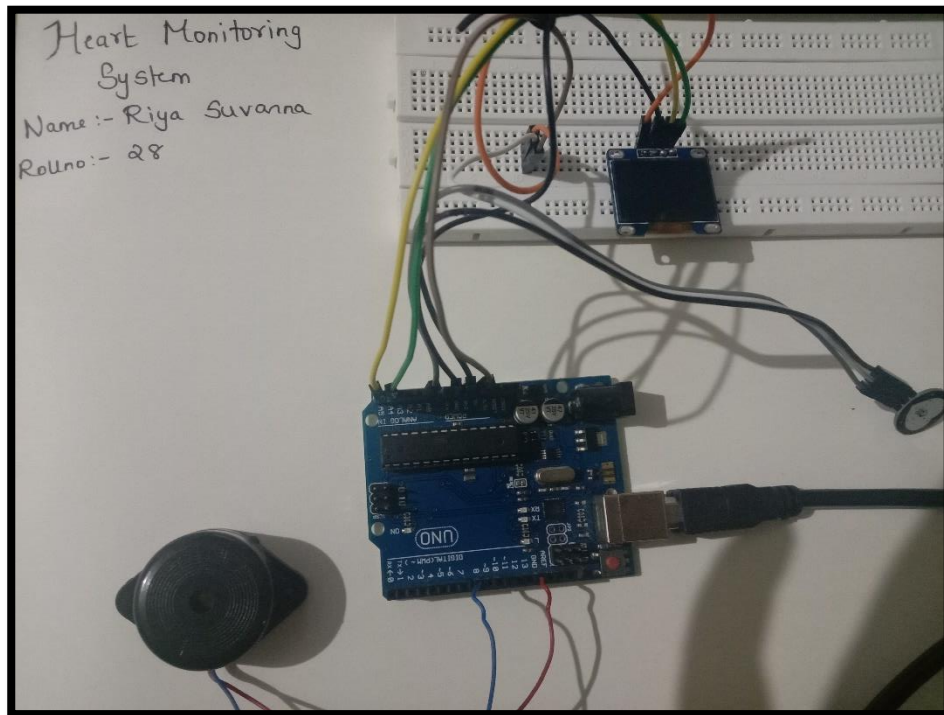
### Use and limitations

As mentioned these are cheap but remarkable sensors but don't expect to strap one to your finger and go jogging. They are very sensitive to movement and moving your hand about will cause massive swings of values – even if taped on. Also, tape them on too tightly and it reduces sensitivity dramatically. They can only be used if your hand is still, having said that you could still measure your Heart Beat at resting and just after exercise to compare. If you have any tips for monitoring with these devices whilst exercising please leave a comment.

### Adding a Piezo Buzzer

## Heart Monitoring System

To make this a little more “realistic” you could add a sound output. So the addition of a Piezo buzzer has been added. Simply connect it to pin D8 and GND, load up the code below and you’ll be good to go. This code beeps every heart-beat. it does not sound a “flat-line” single tone if limited activity. This is because it was not always very reliable due to the nature of the sensor which detects spurious pulses. In the real world a heart-beat monitor has several sensors monitoring actual heart activity. The photo shows my speaker addition.



### 5.2 Coding: -

#### 1)Code for plotting a graph in serial plotter

```
1 void setup() {
2   Serial.begin(9600);
3 }
4
5
6 void loop()
7 {
8   Serial.println(analogRead(0));
9   delay(20);
10 }
```

#### 2)Code for OLED Display

```
1 #include <Adafruit_SSD1306.h>
2
3 #define OLED_Address 0x3C    // Try 0x3D if not working
4 Adafruit_SSD1306 oled(128, 64); // create our screen object setting
5 resolution to 128x64
6
7 int x=0;
8 int lastx=0;
9 int lasty=0;
10 int LastTime=0;
11 int ThisTime;
12 bool BPMTiming=false;
13 bool BeatComplete=false;
14 int BPM=0;
15 #define UpperThreshold 560
16 #define LowerThreshold 530
17
18 void setup() {
19   oled.begin(SSD1306_SWITCHCAPVCC, OLED_Address);
20   oled.clearDisplay();
21   oled.setTextSize(2);
22 }
23
24
25 void loop()
```

```
26 {
27 if(x>127)
28 {
29 oled.clearDisplay();
30 x=0;
31 lastx=x;
32 }
33
34 ThisTime=millis();
35 int value=analogRead(0);
36 oled.setTextColor(WHITE);
37 int y=60-(value/16);
38 oled.writeLine(lastx,lasty,x,y,WHITE);
39 lasty=y;
40 lastx=x;
41 // calc bpm
42
43 if(value>UpperThreshold)
44 {
45 if(BeatComplete)
46 {
47 BPM=ThisTime-LastTime;
48 BPM=int(60/(float(BPM)/1000));
49 BPMTiming=false;
50 BeatComplete=false;
51 tone(8,1000,250);
52 }
53 if(BPMTiming==false)
54 {
55 LastTime=millis();
56 BPMTiming=true;
57 }
58 }
59 if((value<LowerThreshold)&(BPMTiming))
60 BeatComplete=true;
61
62 // display bpm
63 oled.writeFillRect(0,50,128,16,BLACK);
64 oled.setCursor(0,50);
65 oled.print(BPM);
66 oled.print(" BPM");
67 oled.display();
68 x++;
```

```
}
```

### 3)Code for buzzer

```
1 #include <Adafruit_SSD1306.h>
2
3 #define OLED_Address 0x3C
4 Adafruit_SSD1306 oled(1);
5
6 int x=0;
7 int lastx=0;
8 int lasty=0;
9 int LastTime=0;
10 int ThisTime;
11 bool BPMTiming=false;
12 bool BeatComplete=false;
13 int BPM=0;
14 #define UpperThreshold 560
15 #define LowerThreshold 500
16
17 void setup() {
18   oled.begin(SSD1306_SWITCHCAPVCC, OLED_Address);
19   oled.clearDisplay();
20   oled.setTextSize(2);
21 }
22
23
24 void loop()
25 {
26   if(x>127)
27   {
28     oled.clearDisplay();
29     x=0;
30     lastx=x;
31   }
32
33   ThisTime=millis();
34   int value=analogRead(0);
35   oled.setTextColor(WHITE);
36   int y=60-(value/16);
37   oled.writeLine(lastx,lasty,x,y,WHITE);
38   lasty=y;
```

```
39 lastx=x;
40 // calc bpm
41
42 if(value>UpperThreshold)
43 {
44     if(BeatComplete)
45     {
46         BPM=ThisTime-LastTime;
47         BPM=int(60/(float(BPM)/1000));
48         BPMTiming=false;
49         BeatComplete=false;
50         tone(8,1000,250);
51     }
52     if(BPMTiming==false)
53     {
54         LastTime=millis();
55         BPMTiming=true;
56     }
57 }
58 if((value<LowerThreshold)&(BPMTiming))
59     BeatComplete=true;
60
61 // display bpm
62 oled.writeFillRect(0,50,128,16,BLACK);
63 oled.setCursor(0,50);
64 oled.print(BPM);
65 oled.print(" BPM");
66 oled.display();
67 x++;
68 }
```

### 5.3 Testing: -

Software testing is the process of evaluation a software item to detect differences between given input and expected output. Also to assess the feature of A software item. Testing assesses the quality of the product. Software testing is a process that should be done during the development process. In other words software testing is a verification and validation process.

#### *Verification*

Verification is the process to make sure the product satisfies the conditions imposed at the start of the development phase. In other words, to make sure the product behaves the way we want it to.

#### *Validation*

Validation is the process to make sure the product satisfies the specified requirements at the end of the development phase. In other words, to make sure the product is built as per customer requirements.

#### *Basics of software testing*

There are two basics of software testing: blackbox testing and whitebox testing.

#### **Blackbox Testing**

Black box testing is a testing technique that ignores the internal mechanism of the system and focuses on the output generated against any input and execution of the system. It is also called functional testing.

#### **Whitebox Testing**

White box testing is a testing technique that takes into account the internal mechanism of a system. It is also called structural testing and glass box testing.

Black box testing is often used for validation and white box testing is often used for verification.

**Testing methods that are applied on the model are:**

- 1. Unit testing**
- 2. Integration Testing**
- 3. System testing**
- 4. Performance testing**
- 5. Security testing**

### **1]Unit Testing:**

- ❖ The unit test works on single component of the system.
- ❖ I checked whether the individual component of the system properly or not.
- ❖ In a hardware project, it is important of having each individual component working properly.
- ❖ In Heart Monitoring System, I tested individual component like Arduino UNO or NANO mcu board , Pulse sensor, OLED display ,Buzzer.
- ❖ The components used in the system were checked by running small scripts on them like blinking LED's with help of the Microcontroller board to ensure that it is working or not.

### **2] Integration Testing:**

- ❖ In integration testing, I tested whether the components complement each other well or not.
- ❖ I tested here the flow of data from one component to other eg. Pulse Sensor to Arduino UNO or Nano mcu and Arduino UNO or Nano mcu to OLED Display etc.
- ❖ This test is very important in combine work of the system.
- ❖ I also had to make sure that each of the component is working finely with respect to their required voltages.

### **3] System Testing:**

- ❖ In system testing I tested weather the components connected together acts as a system well or not.
- ❖ The system is tested on terms like if we achieve the functionality that I want or not.



## | Heart Monitoring System

- ❖ If the system is not working, as we want then I had to fix the problems that system is facing while working.
- ❖ System testing will gives us many positive and negative points about system we have built.
- ❖ I got many corrections to do in the system after system testing, so it improved quality and accuracy of the proposed system.

### **4] Performance Testing:**

- ❖ System's performance is also a big issue in system making.
- ❖ System performance must be optimized.
- ❖ I tested the performance by testing the system with respect to time.
- ❖ In addition, I had to test the accuracy of the system.

### **5] Security Testing:**

- ❖ The system data will be on internet so it will be a risk of security.
- ❖ Security testing can be done with testing weather the admin credentials are hidden and secrete are not.
- ❖ In addition, we can test whether any other person other than admin and the doctor can login in system without proper authentications or not.
- ❖ In addition, the end users of the system will not have any direct access to the database, hence they cannot modify the patients data.
- ❖ Also the third party intruders can't interfere in database as it requires admin's Google account credentials to access the database.

### Test Case table: -

Test Case ID	Test Scenario	Operator actions	Expected results	Actual results	Pass/Fail
T01	Check for uploading the code into Arduino	Input from Arduino IDE	Done uploading msg should be displayed	Done uploading	Fail/Pass
T02	Check pulse sensor reading	Input from the Pulse sensor	Notification on LCD	Message shown in OLED display	Pass
T03	Check pulse sensor reading in the display	Take input from pulse sensor	Notification on LCD	Message shown in OLED display	Pass
T04	Check Serial Plotter for output in PC	Pulse sensor reading	Data displayed in serial plotter	Data displayed in serial plotter	Pass
T05	Check whether buzzer starts	The pulse sensor and the buzzer is	The buzzer starts when activity found	The buzzer starts when pulse is found	Pass

	properly	connected to the Arduino			
T06	Check admin's login with valid ID and password on web browser	Enter valid credentials on login screen	Admin must be logged in	Admin logged in	Pass
T07	Check and keep a track on the details	Input on the database	Shows the pulse data in the database	Display data in the database	Pass

### 5.4 Cost benefit Analysis:

#### Effort Estimation Using COCOMO Model:-

- ❖ For any given set of the requirement it is essential to know how much it will cost to develop the application software to satisfy the given requirements and how much time it will take for the development.
- ❖ These estimates are needed before development is initiated.
- ❖ The primary reason for cost and schedule estimation is to enable client or developer to perform a cost benefits analysis and project monitoring and control.
- ❖ The project has been developed considering the COCOMO (Constructive Cost Model) which computer software development effort as a function of program size and set of “Cost Drivers” that includes product, hardware, personnel and project attribute.
- ❖ Cost effort is calculated in terms of person-month.

#### Estimated Lines of code

- Minimum lines of codes,  $s_i = 200$
- Maximum lines of codes,  $s_n = 1000$
- Most likely lines of codes,  $s_m = 700$
- Estimated lines of code  $= s_i + s_n + 0.67s_m = 1669$
- KLOC  $= \text{Estimated lines of code} / 1000 = 1669 / 1000 = 1.669$

#### Estimating B (disproportionate effort)

Rating (1-5)	
Development Flexibility	3
Architecture/ Risk Resolution	3
Man Power	2
Process Maturity	3
Total	11

$$B = (\text{sum}/100) + 1.01$$

$$= 0.11 + 1.01$$

$$= 1.12$$

Determine a set of 15 multiply factors from different attributes of the product which are

- **Computer Attributes:**  
Execution Time Constraints, Main Storage, Constraints, Virtual machine, Volatility, Computer Turnaround Time.
- **Product Attributes:**  
Required Reliability, Product Complexity.
- **Personnel Attributes:**  
Analyst Capability, Application Experience, Programmer Capability, Virtual Machine Experience, Programming Language Experience.
- **Project Attributes:**  
Modern Programming Practices, Use Software Tools, Required Development Schedule.

Estimating M (multiplier reflecting product, process & people attributes)

Cost Drivers	Value	Rating
<b>Product Attribute</b>		
Requires attributes	High	1.00
Complexity of the product	High	0.95
<b>Hardware Attributes</b>		
Run time performance constraint	High	1.00
Memory constraint	Normal	1.00
Volatility of virtual machine environment	High	1.00
<b>Personnel Attribute</b>		
Analyst Capability	Very High	1.15
Software engineer capability	Very High	1.15
Application experience	High	1.00

## Heart Monitoring System

Programming language experience	Normal	0.95
<b>Project Attribute</b>		
Use of software tools	Normal	0.95
Application of SE method	High	0.95
Required development schedule	Normal	1.00

$$M = 1.00 * 0.95 * 1.00 * 1.00 * 1.00 * 1.15 * 1.15 * 1.00 * 0.95 * 0.91 * 0.91 * 1.00 = 0.98$$

### Estimating Efforts

$$\text{Efforts} = A * \text{Size}^B * M$$

$$= 2.4 * 1.669^{1.12} * 0.98$$

$$= 4.17 \text{ PM}$$

### Development Time

$$\text{TDEV} = 3 * (\text{PM})^{(0.33 + 0.2 * (B - 1.01))}$$

$$= 4.9 \text{ Months}$$

### Actual Cost

- Electricity Cost: Rs.1000/-
- Material Cost: Rs.500/-
- Software Cost: Rs.1000/-

$$\text{Actual cost} = \text{Effort} * \text{Total cost}$$

$$= 4.17 * 2500$$

$$= \text{Rs.10,425/- A}$$

## Chapter 6: Result and Discussion

### 6.1 Results: -

Once the circuit has been built, codes have been uploaded into the Nano Arduino then the project is ready for testing. When we carried out implementations and testing the following results were obtained.

#### 6.1.1 Heart Rate Results in Serial Plotter and OLED display:

The heart rate was obtained using two methods, the manual method and using the pulse sensor to determine the accuracy of the project's circuit. The circuit is supplied by 5V power. For accurate reading as much as possible, the finger or the wrist needs to be placed close to sensor. The output result as an ECG in the Serial Plotter was represented to a certified doctor to determine its accuracy.

The certified doctor examined the signal shown in Figure 6.1. She stated that the signal is noisy and it is hard to determine the HR from it as it is measured from at least three successive peaks which are not existed within these signal.

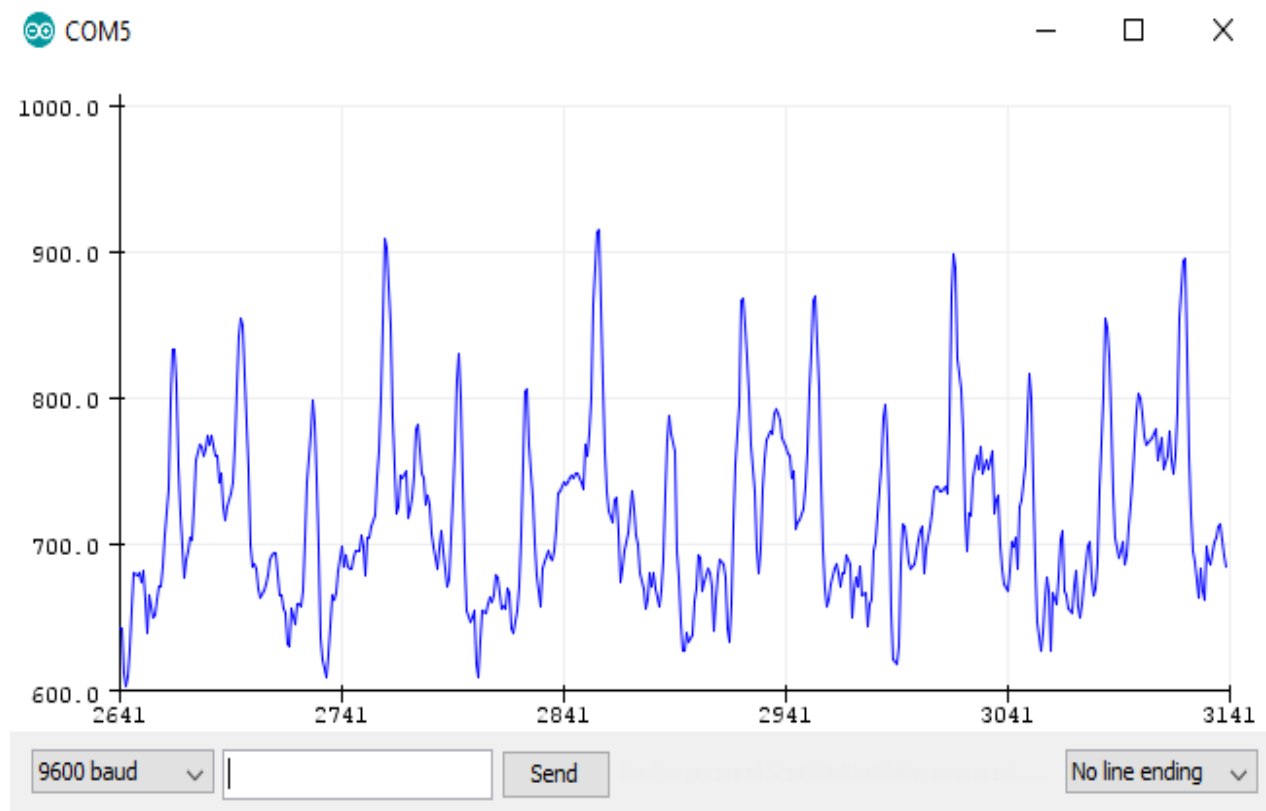
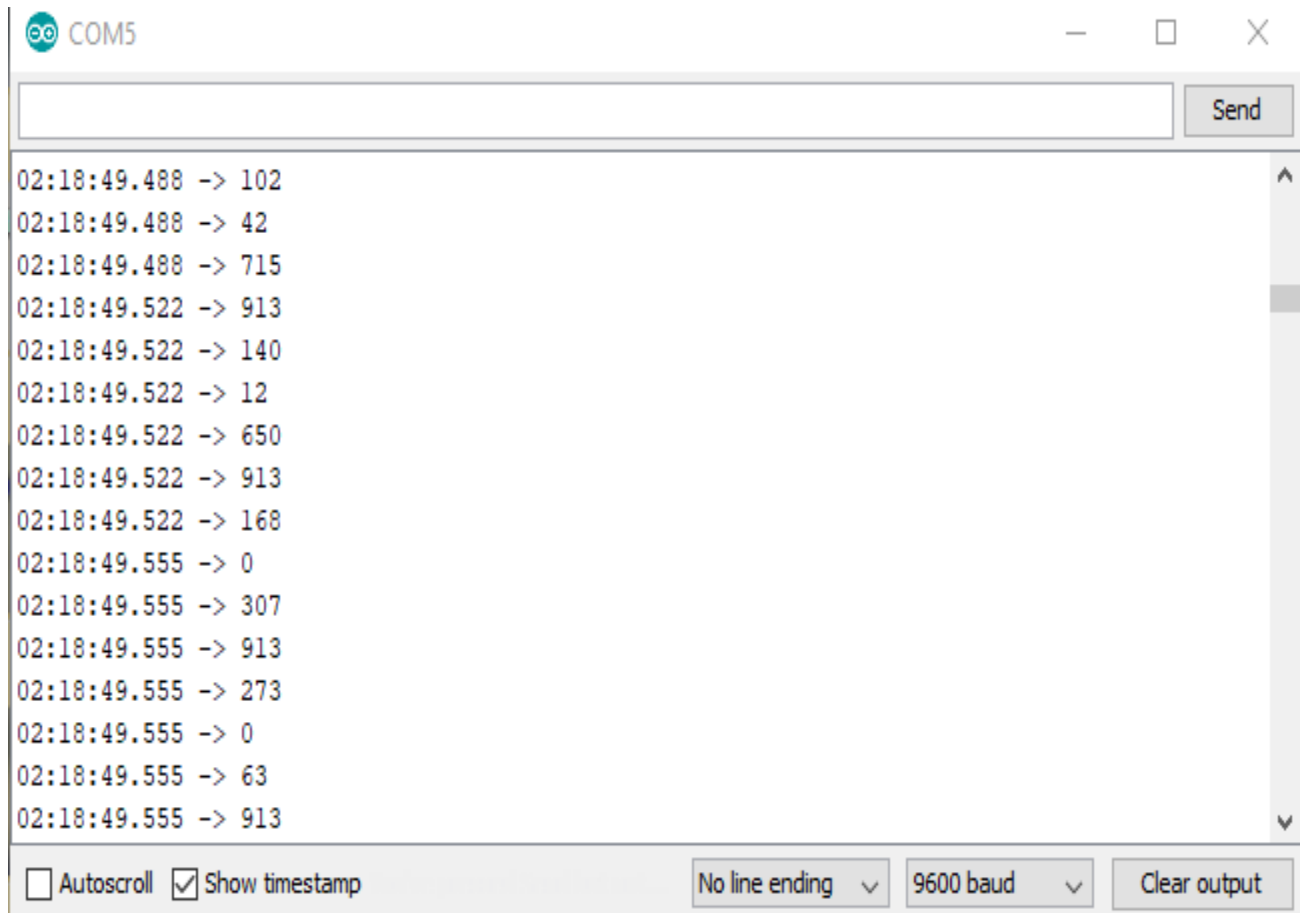


Figure 6.1 HR Wave Output in Serial Plotter

## Heart Monitoring System

After that, the same experiment was repeated but this time results were outputted via the Serial Monitor as shown in Figure 6.2 below.



**Figure 6.2 HR Values Output in Serial Monitor**

The pulsed calculated manually, to determine the accuracy by comparing the manual with Serial Monitor value.

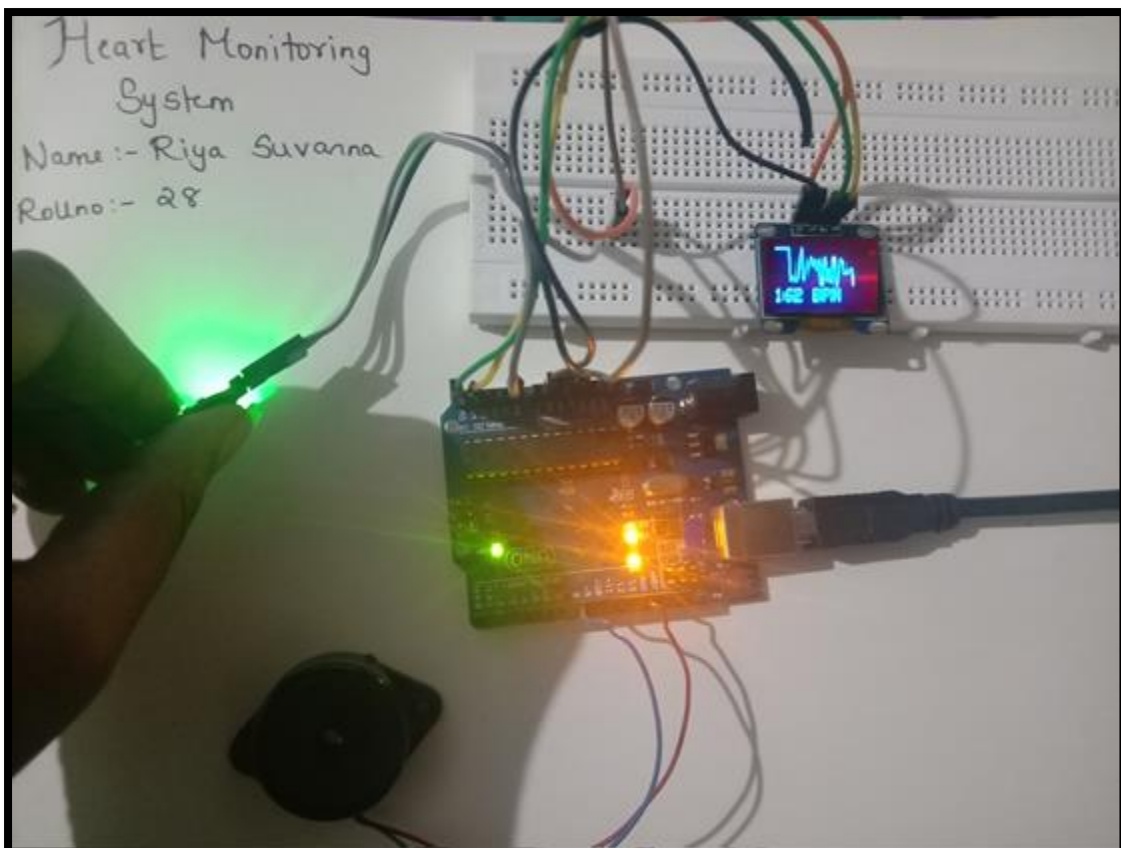
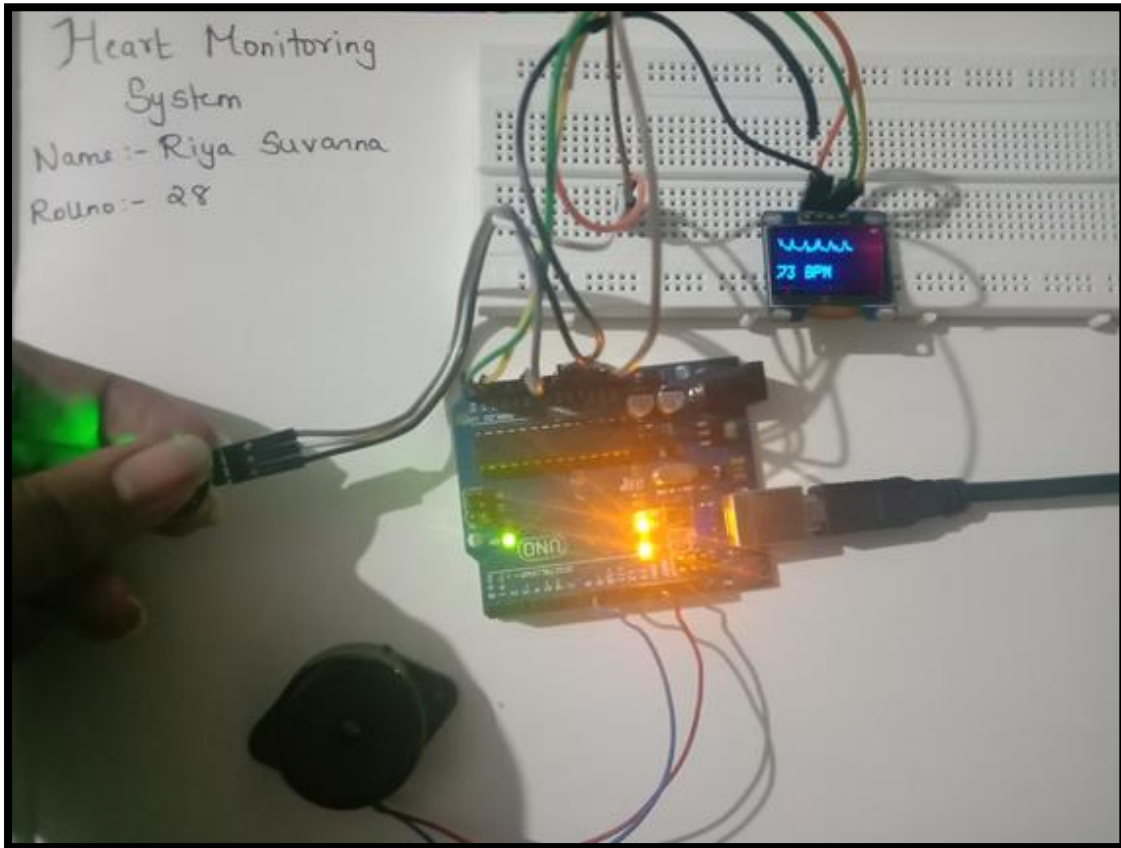
Error was calculated to be:

$e = (91 - 88) / 91 = 0.032$ , which means an accuracy of 96.8%.



## Heart Monitoring System

Output: -



### **6.2 Discussion: -**

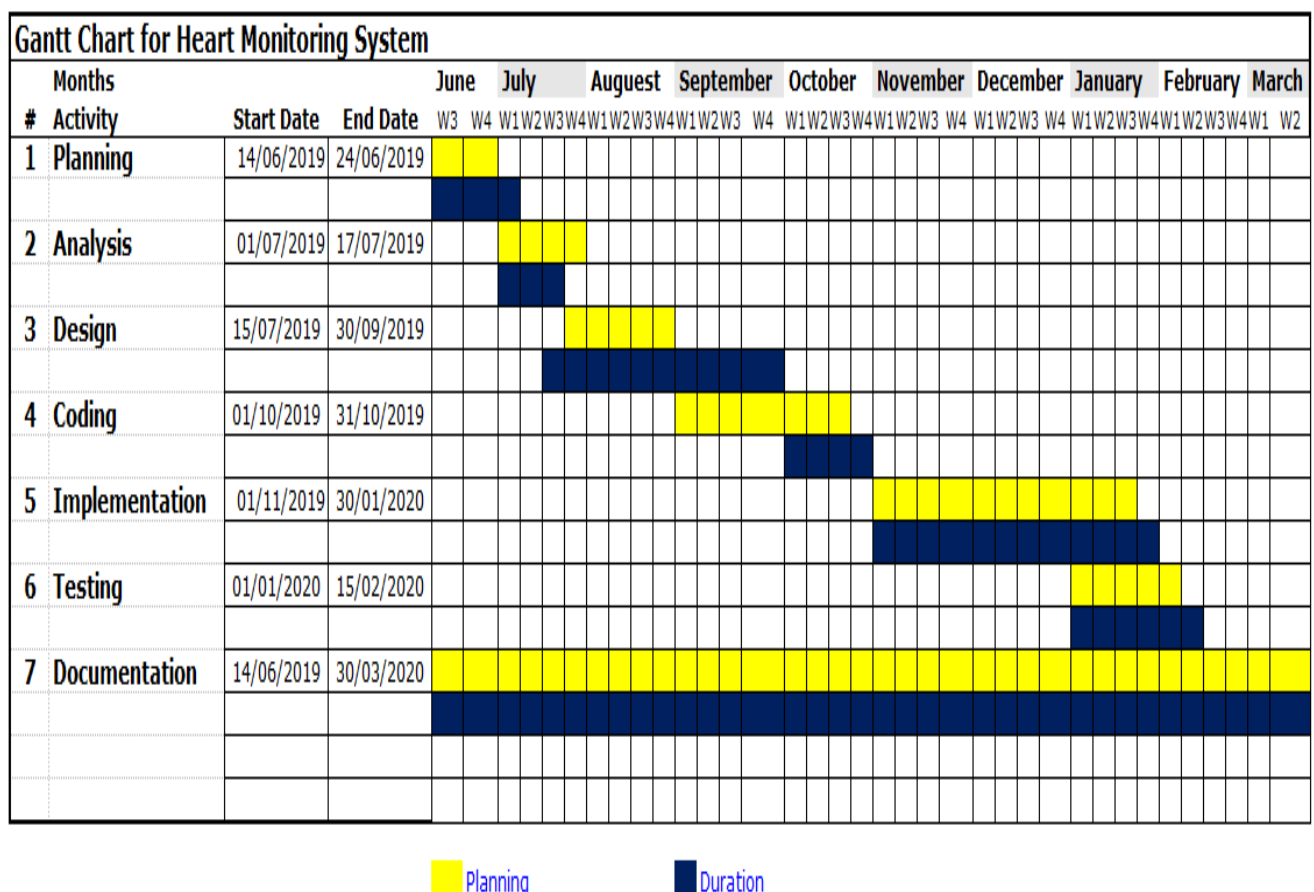
- According to the specialized doctor, the heart rate ECG is not measurable, in other words the signal is noisy and has to be filtered first in order to be able to extract heart rate value from it.
- The ambient noise maybe generated from improper holding of pulse sensor or the component is defected.
- On the other hand, the OLED display was much better which outputted data with accuracy of almost 97%.
- The module were implemented correctly and performed their supposed functions successfully which is detection the pulse as well as receiving data from Arduino and determining the data of the patient respectively.

## GANTT CHART

A Gantt chart, commonly used in project management, is one of the most popular and useful ways of showing activities (tasks or events) displayed against time. On the left of the chart is a list of the activities and along the top is a suitable time scale. Each activity is represented by a bar; the position and length of the bar reflects the start date, duration and end date of the activity. This allows you to see at a glance:

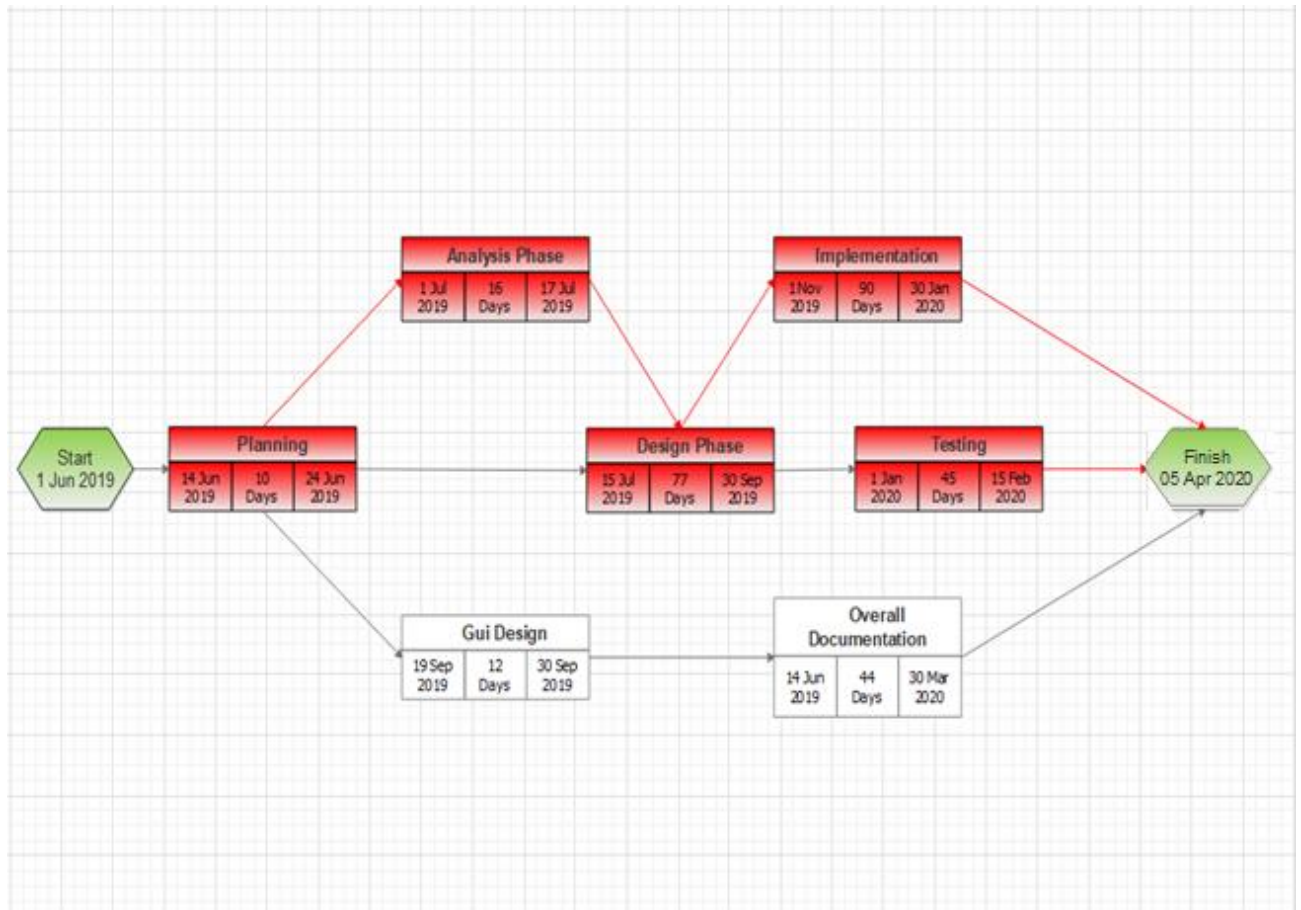
- What the various activities are
- When each activity begins and ends
- How long each activity is scheduled to last
- Where activities overlap with other activities, and by how much
- The start and end date of the whole project

The project timeline for project has been scheduled.



### PERT CHART

PERT Chart is acronym for (Program Evaluation and Review Technique). A PERT chart is a project management tool used to schedule, organize, and coordinate tasks within a project. It is basically a method to analyze the tasks involved in completing a given project, especially the time needed to complete each task, and to identify the minimum time needed to complete the total project.



## **Chapter 7: Conclusion and Future Enhancement**

### **7.1 Conclusion: -**

The progress in bio medical engineering, science and technology paved way for new inventions and technologies. As we are moving towards miniaturization, handy electronic components are in need. New products and new technology are being invented. ARDUINO was found to be more compact, user friendly and less complex, which could readily be used in order to perform several tedious and repetitive tasks. Simulation is performed using Arduino software by placing appropriate sensors like temperature and heart beat rate for sensing the health condition and the results are analyzed under normal conditions and abnormality conditions.

### 7.2 Limitation: -

While building the entire system we had to face some challenges. In our system there is one phases:

a. Hardware implementation

The challenges that we faced are discussed below:

- a) **Hardware Implementation:** The process of getting all components together was a really challenging part. While taking the values through the sensor and get it saved in Arduino, we faced some problems. For pulse sensor we used SEN 11574. This sensor is not so expensive and the output is reliable. But at first it was not functioning properly due to the connectivity. I also faced the most common hardware problem which is that the Arduino was not operating. Sometimes it showed the error that there was low capacity even though there was not. After getting the outputs into an Arduino we had to proceed to the next step which was storing the data. Even when we connected it to the Arduino successfully it sometimes could not get connected to the data and for this reason even though the outputs were shown in the serial monitor but they were not easily stored. It took a very long time to make it work properly. There was another difficulty which is data upload delay. We succeed to reduce it.

### 7.3 Future Enhancement: -

#### **a. Advanced Medical Equipment Including:**

Our system is just a platform for the developing countries. We have done it with an efficient cost. Other medical equipment such as Continuous Glucose Monitoring, Shield EKG-EMG (ELECTROCARDIOGRAPHY ELECTROMYOGRAPHY SHIELD) etc for better service.

#### **b. Upgraded Version of the application:**

We want to add some features to this application in the future such as making it dynamic so that the doctor can customize the time interval. Currently the data is sent according to the device is configured.

#### **c. Include Blood Pressure Data:**

We already have hacked the blood pressure machine. We want to add it to our device in future. Blood pressure values are really important to determine a person's health condition. So, we want to include it in the future.

#### **d. Multiple Patients can use one device:**

Our current device is for one particular patient only. But it can be used for multiple patients too. We hope to work on it.

#### **e. Push Notification System for the application:**

We have an automated emergency email sending system in our application. When the patient's data is abnormal, an emergency email will be sent to the emergency email address. We also plan to add a push notification system in the future.

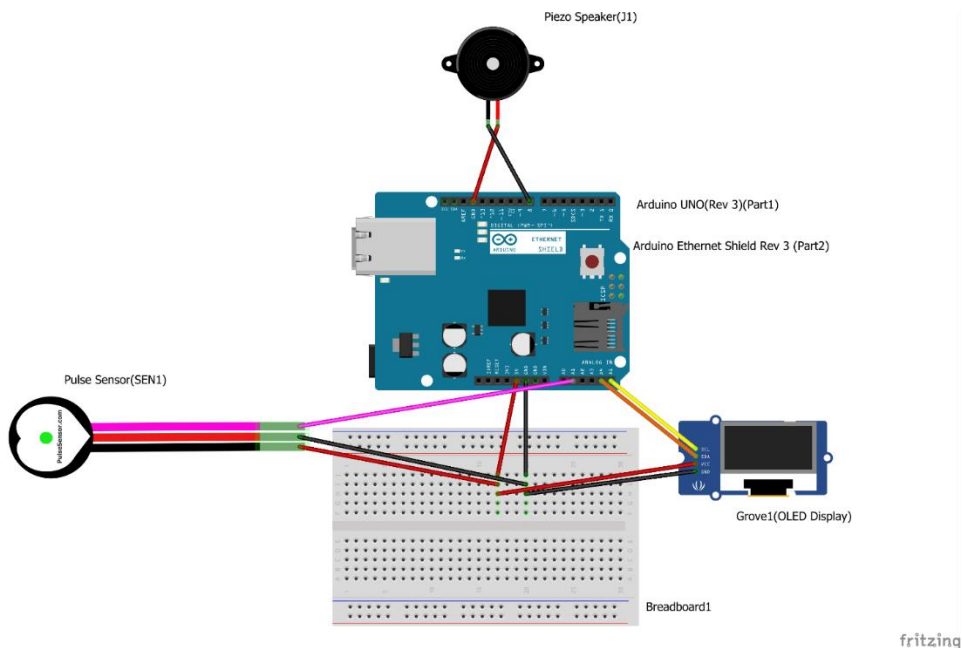
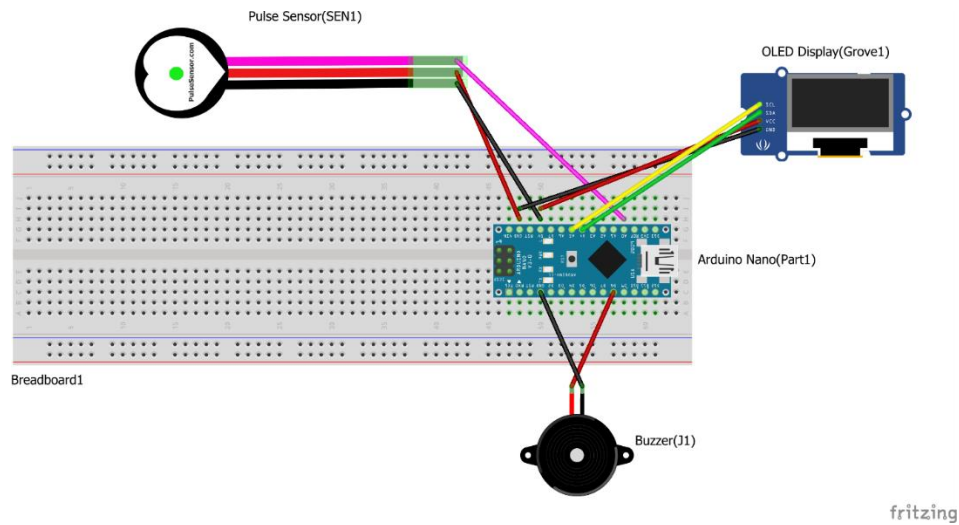
#### **f. Include GPS system:**

We also will include GPS system to know the current location of the patient.

### 7.4 Actual system vs proposed system: -

#### Proposed System: -

In the proposed system from last semester, I had planned to use an Arduino as the microcontroller board. The system was going to have an user interface on the system itself made with the help of various Pulse sensors, Buzzer and a OLED display. The microcontroller would have been working with the wifi shield to send the data to cloud. I was planning to have a GUI on the server side also for the various logins for administrator and faculty of the college. The data would have been stored using RDBMS with SQL.

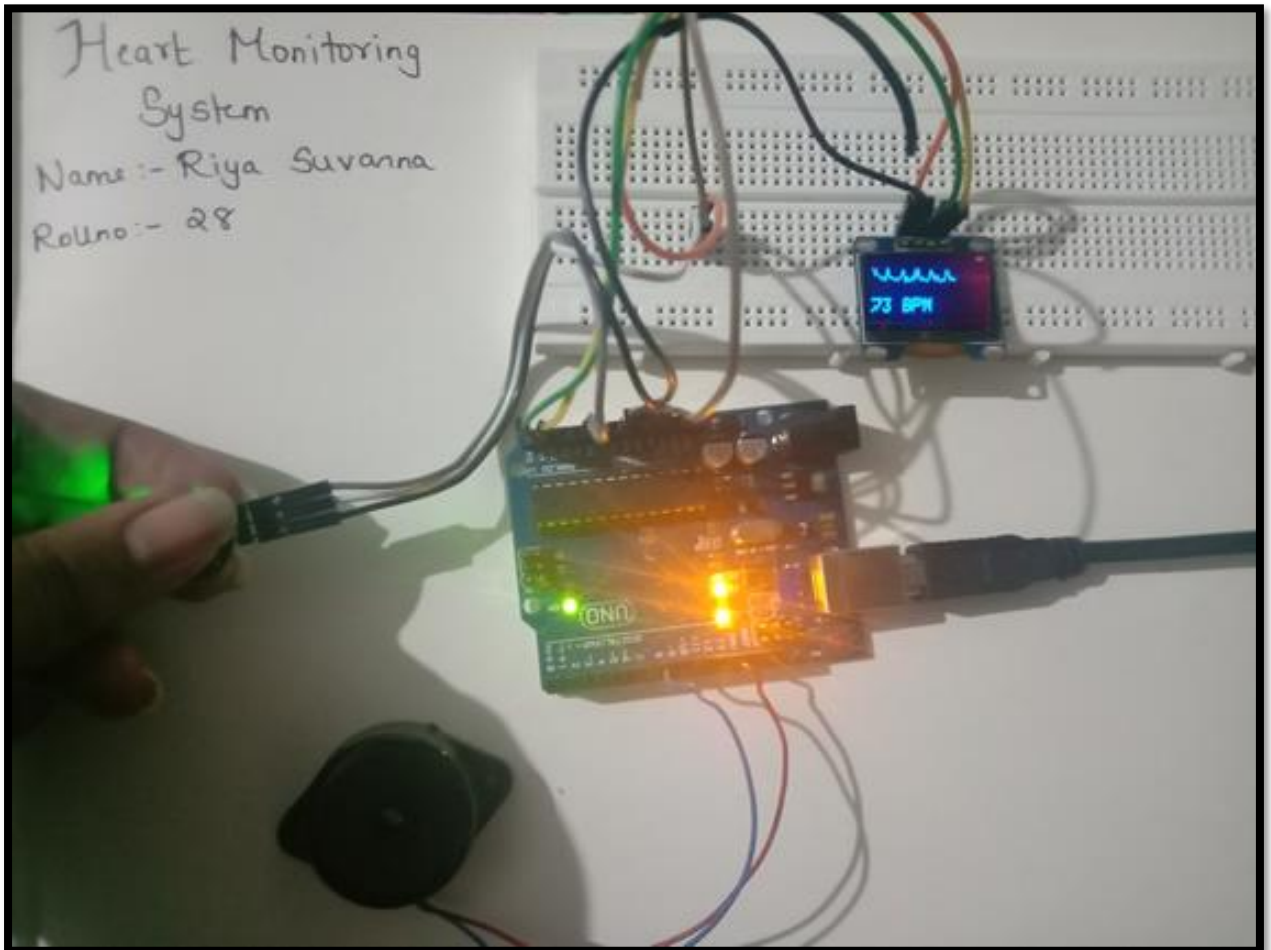




## Heart Monitoring System

### Actual system: -

There are many changes made in the system for reducing the build cost and removing hardware to make the system more compact and easy to transport. The system consist of the Arduino Uno or Nano mcu, Pulse sensor, OLED display, buzzer. The UI provided on system is with the help of LED and a Oled display which ic flexible and very less expensive. And storing the data directly.



## Chapter 8: References

### Books:

1. The Internet of things by Samuel Greengard.
2. Getting started with Internet of Things by Cuno Pfister.
3. Learning Internet of Things by Peter Waher.
4. Designing the Internet of Things by Adrian McEwen.

### Websites:

Sr No	Site link	Viewed date and time
1	“Arduino Architecture” <a href="https://www.engineersgarage.com/what-is-gsm-gprs-module">https://www.engineersgarage.com/what-is-gsm-gprs-module</a>	[June. 1, 2019] 9 PM
2	“Systems design” <a href="https://en.wikipedia.org/wiki/Systems_design">https://en.wikipedia.org/wiki/Systems_design</a>	[Aug.15,2019] 8.30 PM
3	“UML - Standard Diagrams” <a href="https://www.tutorialspoint.com/uml/uml_standard_diagrams.htm">https://www.tutorialspoint.com/uml/uml_standard_diagrams.htm</a>	[Aug.18,2019] 11.30 AM
4	“The Internet of Things in healthcare: an overview” <a href="https://scholar.google.com/citations?user=Y4opLB8AAAAJ&amp;hl=en">https://scholar.google.com/citations?user=Y4opLB8AAAAJ&amp;hl=en</a>	[Sept. 7, 2019] 7.00 PM
5	“Envisioning inclusive futures: technology-based assistive sensory and action substitution” <a href="https://www.infona.pl/resource/bwmeta1.element.elsevier-3d45bfdd-fe55-359f-84e4-674a21cae024">https://www.infona.pl/resource/bwmeta1.element.elsevier-3d45bfdd-fe55-359f-84e4-674a21cae024</a>	[Sept 4, 2019] 9.00 PM
6	“A multiple communication standards compatible IoT system for medical usage” <a href="http://ieeexplore.ieee.org/document/6577775/?reload=true">http://ieeexplore.ieee.org/document/6577775/?reload=true</a>	[Sept 5, 2019] 6.00 PM
7	“Ubiquitous data accessing method in IoT-based information system for emergency medical services” <a href="https://www.deepdyve.com/lp/institute-of-electrical-and-electronics-engineers/ubiquitous-data-accessing-method-in-iot-based-information-">https://www.deepdyve.com/lp/institute-of-electrical-and-electronics-engineers/ubiquitous-data-accessing-method-in-iot-based-information-</a>	[Oct 6, 2019] 3.00 PM

	system-for-YCZzyY5W9g	
<b>8</b>	<p>“Implementation of a medical support system considering P2P and IoT technologies”</p> <p><a href="https://www.computer.org/csdl/proceedings/cisis/2014/4325/00/4325a101-abs.html">https://www.computer.org/csdl/proceedings/cisis/2014/4325/00/4325a101-abs.html</a></p>	<p>[Oct 7, 2019]</p> <p>10.00 PM</p>
<b>9</b>	<p>“Acquisition and management of biomedical data using Internet of Things concepts”</p> <p><a href="http://ieeexplore.ieee.org/document/7050625/">http://ieeexplore.ieee.org/document/7050625/</a></p>	<p>[Oct 10,2019]</p> <p>9.00 PM</p>
<b>10</b>	<p>“Real time internet application with distributed flow environment for medical IoT”</p> <p><a href="https://csdl.computer.org/csdl/proceedings/icgciot/2015/7910/00/07380578-abs.html">https://csdl.computer.org/csdl/proceedings/icgciot/2015/7910/00/07380578-abs.html</a></p>	<p>[Nov 11,2019]</p> <p>9.45 PM</p>
<b>11</b>	<p>“Secure end-to-end communication for constrained devices in IoT-enabled ambient assisted living systems”</p> <p><a href="https://www.computer.org/csdl/proceedings/wf-iot/2015/0366/00/07389141-abs.html">https://www.computer.org/csdl/proceedings/wf-iot/2015/0366/00/07389141-abs.html</a></p>	<p>[Sept 11,2019]</p> <p>8.00 PM</p>
<b>12</b>	<p>“Software Testing”</p> <p><a href="https://en.wikipedia.org/wiki/Software_testing">https://en.wikipedia.org/wiki/Software_testing</a></p>	<p>[Dec 20, 2019]</p> <p>5.00 PM</p>
<b>13</b>	<p>“Sensors”</p> <p><a href="https://tkkrlab.nl/wiki/Arduino_37_sensors">https://tkkrlab.nl/wiki/Arduino_37_sensors</a></p>	<p>[Feb 20, 2019]</p> <p>6.00 PM</p>