1. Introduction

Monitoring heart rate is important for athletes and patients as it determines the heart's condition (just heart rate). There are many ways to measure heart rate and the most precise one is using an Electrocardiography
But the easiest way to monitor the heart rate is to use a Heartbeat Sensor. It comes in different shapes and sizes and allows an instant way to measure the heartbeat.
Heartbeat Sensors are available in wristwatches (Smart Watches), Smart Phones, chest straps, etc. The heartbeat is measured in beats per minute or bpm, which indicates the number of times the heart is contracting or expanding in a minute.

2. Principle

The principle behind the working of the Heartbeat Sensor is Photoplethysmography. According to this principle, the changes in the volume of blood in an organ are measured by the changes in the intensity of the light passing through that organ.
Usually, the source of light in a heartbeat sensor would be an IR LED and the detector would be any Photo Detector like a Photo Diode, an LDR (Light Dependent Resistor) or a Photo Transistor.
With these two i.e., a light source and a detector, we can arrange them in two ways: A Transmissive Sensor and a Reflective Sensor.
In a Transmissive Sensor, the light source and the detector are placed facing each other and the finger of the person must be placed in between the transmitter and receiver.
Reflective Sensor, on the other hand, has the light source and the detector adjacent to each other and the finger of the person must be placed in front of the sensor.

3. Components required

a. Arduino UNO
b. Pulse sensor
c. LED
d. 220-ohm resistor
e. I2C module
f. 16×2 LCD
g. Jumper wires and a breadboard
h. USB cable for uploading the code

4. Components Description

A. Arduino UNO

Arduino UNO is a microcontroller board based on the **ATmega328P**. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, 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 an AC-to-DC adapter or battery to get started. You can tinker with your UNO without worrying too much about doing something wrong, worst-case scenario you can replace the chip for a few dollars and start over again.

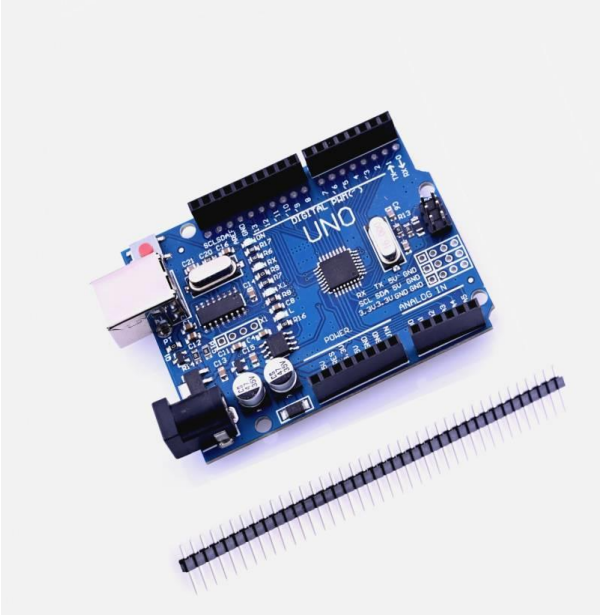

Figure 1

B. Arduino Software (IDE)

The open-source Arduino Software (IDE - Integrated Development Environment) is a cross-platform application that is written in functions from C and C++. This software can be used with any Arduino compatible board which makes it easy to write code and upload it to the board.

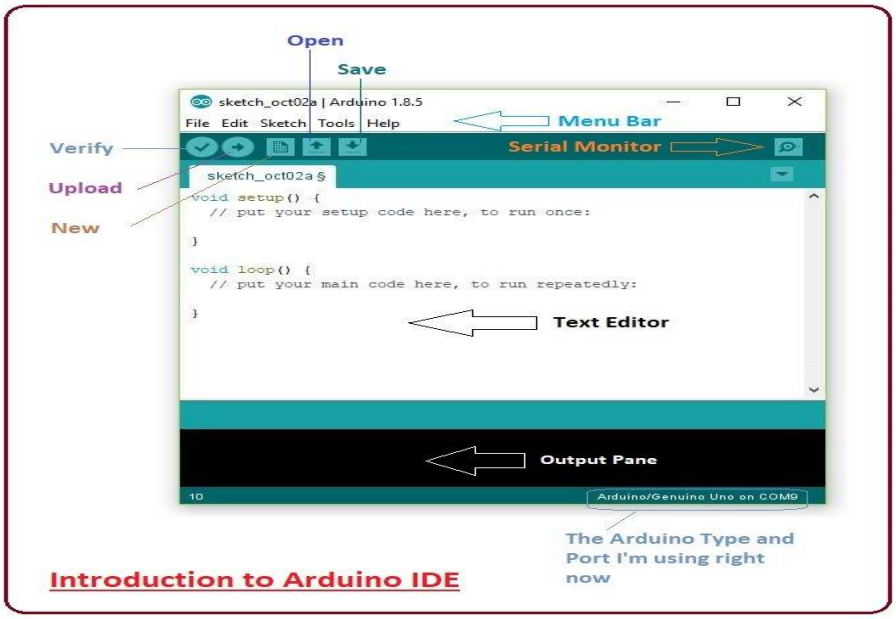

Figure 2

C. Pulse sensor

An alternate name for this sensor is a heartbeat sensor or heart rate sensor. The working of this sensor can be done by connecting it from the fingertip or human ear to an Arduino board. So that heart rate can be easily calculated.

The main specifications of this sensor mainly include the following.

This is a heartbeat-detecting and biometric pulse rate sensor

- Its diameter is 0.625
- Its thickness is 0.125
- The operating voltage ranges from +5V otherwise +3.3V
- This is a plug and play type sensor
- The current utilization is 4mA
- Includes circuits like Amplification and noise cancellation
- This pulse sensor is not approved by the FDA or medical. So it is used in student-level projects, not for commercial purposes in health issues applications.

Figure 3

- Pin-1 (GND): Black Color Wire – It is connected to the GND terminal of the system.
- Pin-2 (VCC): Red Color Wire – It is connected to the supply voltage (+5V otherwise +3.3V) of the system.
- Pin-3 (Signal): Purple Color Wire – It is connected to the pulsating o/p signal.

D. I2C Module

This is a RoHS-compliant I2C Serial LCD Daughter board that can be connected to a standard 16×2 or 20×4 Character Display Module that supports 4-bit mode. All Character Modules sold on our site support 4-bit mode, and nearly all commercially available 16×2 and 20×4 line character modules support it too.

This board has a PCF8574 I2C chip that converts I2C serial data to parallel data for the LCD. There are many examples on the internet for using this board with Arduino. Search "Arduino LCD PCF8574". The I2C address is 0x3F by default, but this can be changed via 3 solder jumpers provided on the board. This allows up to 3 LCDs to be controlled via a single I2C bus (giving each one its address)

Specification & Features:

- Operating Voltage: 5V
- Backlight and Contrast is adjusted by potentiometer
- Serial I2C control of LCD using PCF8574
- Come with 2 IIC interface, which can be connected by Dupont Line or IIC dedicated cable
- Compatible for 16x2 LCD
- This is another great IIC/I2C/TWI/SPI Serial Interface
- With this I2C interface module, you will be able to realize data display via only 2 wires.

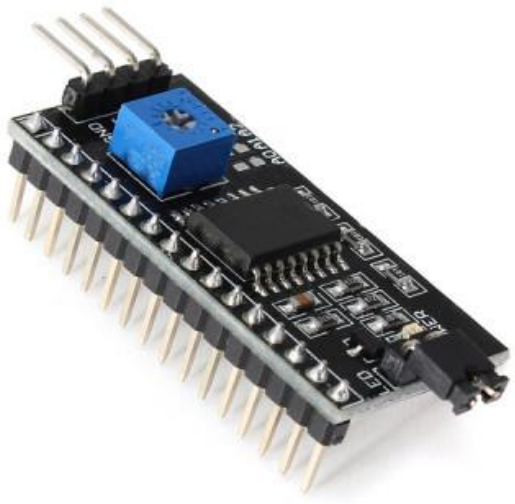

Figure 4

E. Liquid Crystal Display 16x2

It is a kind of electronic display module. These displays are mainly preferred for multi-segment light-emitting diodes and seven segments. The main benefits of using this module are inexpensive; simple, have animations, and there are no limitations for displaying custom characters, special and even animations, etc.

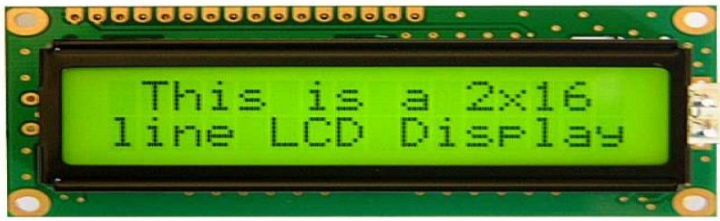

Figure 5

F. Breadboard

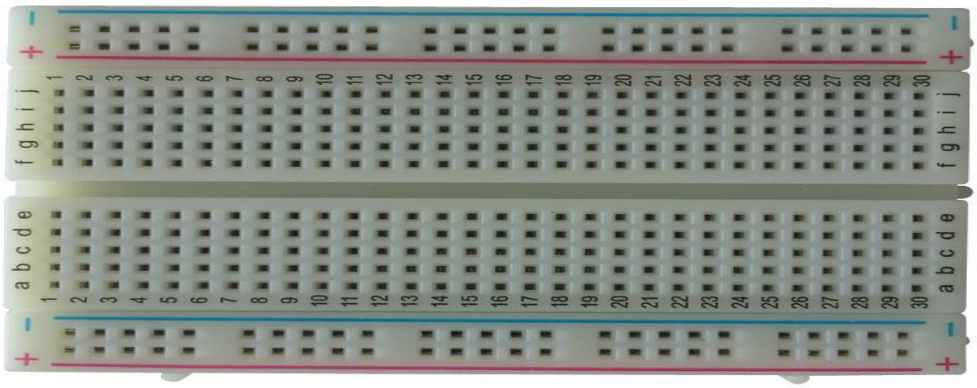

Figure 6

A breadboard, solderless breadboard, or protoboard is a construction base used to build semi-permanent prototypes of electronic circuits. Unlike a perfboard or stripboard, breadboards do not require soldering or destruction of tracks and are hence reusable. For this reason, breadboards are also popular with students and in technological education.

A variety of electronic systems may be prototyped by using breadboards, from small analog and digital circuits to complete central processing units (CPUs).

Compared to more permanent circuit connection methods, modern breadboards have high parasitic capacitance, relatively high resistance, and less reliable connections, which are subject to jostle and physical degradation. Signaling is limited to about 10 MHz, and not everything works properly even well below that frequency.

G. Resistors

A resistor is a passive two-terminal electrical component that implements electrical resistance as a circuit element. In electronic circuits, resistors are used to reduce current flow, adjust signal levels, to divide voltages, bias active elements, and terminate transmission lines, among other uses. High-power resistors that can dissipate many watts of electrical power as heat may be used as part of motor controls, in power distribution systems, or as test loads for generators. Resistors are common elements of electrical networks and electronic circuits and are ubiquitous in electronic equipment. Practical resistors as discrete components can be composed of various compounds and forms. Resistors are also implemented within integrated circuits.

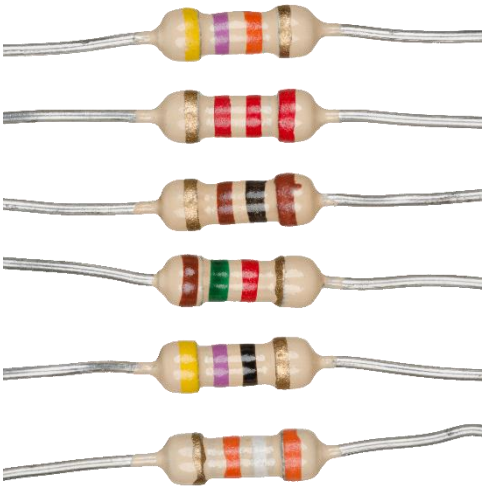

Figure 7

H. Connecting wires

A wire is a flexible strand of metal. The wire is commonly formed by drawing the metal through a hole in a die or draw plate. Wire gauges come in various standard sizes, as expressed in terms of a gauge number.

Figure 8

I. Light Emitting Diode (LED)

A LED is a semiconductor light source that emits light when current flows through it. It is a special type of PN junction diode. Light is produced when the particles that carry the current combine together within the semiconductor material.

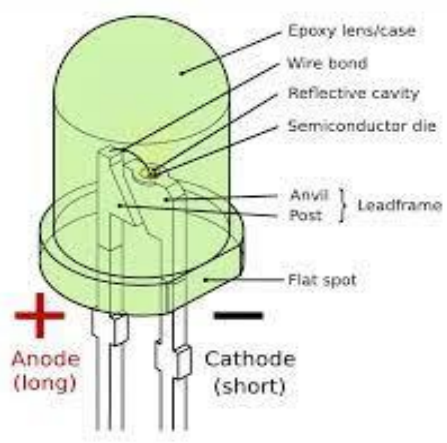

Figure 9

5. Circuit for Heart Beat Sensor BPM Monitor

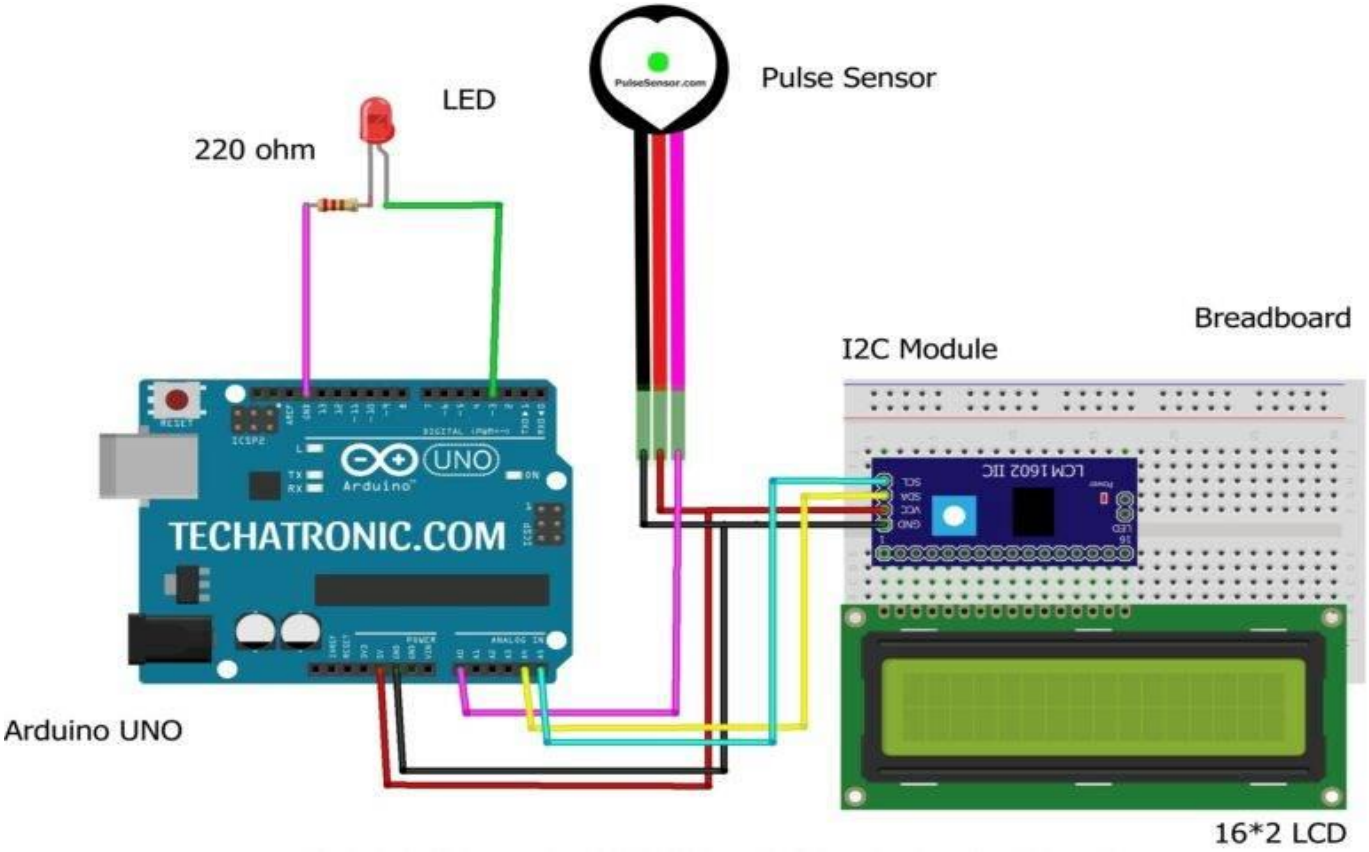

Figure 10

- A pulse sensor and connect its VCC pin with the 5-volt pin of the Arduino.
- Join the GND pin of the pulse sensor with the GND pin of the Arduino.
- Attach the OUT/signal pin of the heartbeat sensor to the Analog-0 pin of the Arduino.
- Now take an LED and connect its positive leg with the digital-3 pin of the Arduino.
- Join the negative leg of the LED with the GND pin of the Arduino via a 220-ohm resistor.
- Then Connect the I2C module with the 16×2 LCD module.
- You can also check the interfacing of the I2C module with Arduino.
- Join the VCC pin of the I2C module with the 5-volt pin of the Arduino and the GND pin of the Arduino with the GND pin of the I2C module.
- Connect the SDA and SCK pins of the I2C module with the analog-4 and analog-5 pins of the Arduino as shown in the diagram.

6. Working Model

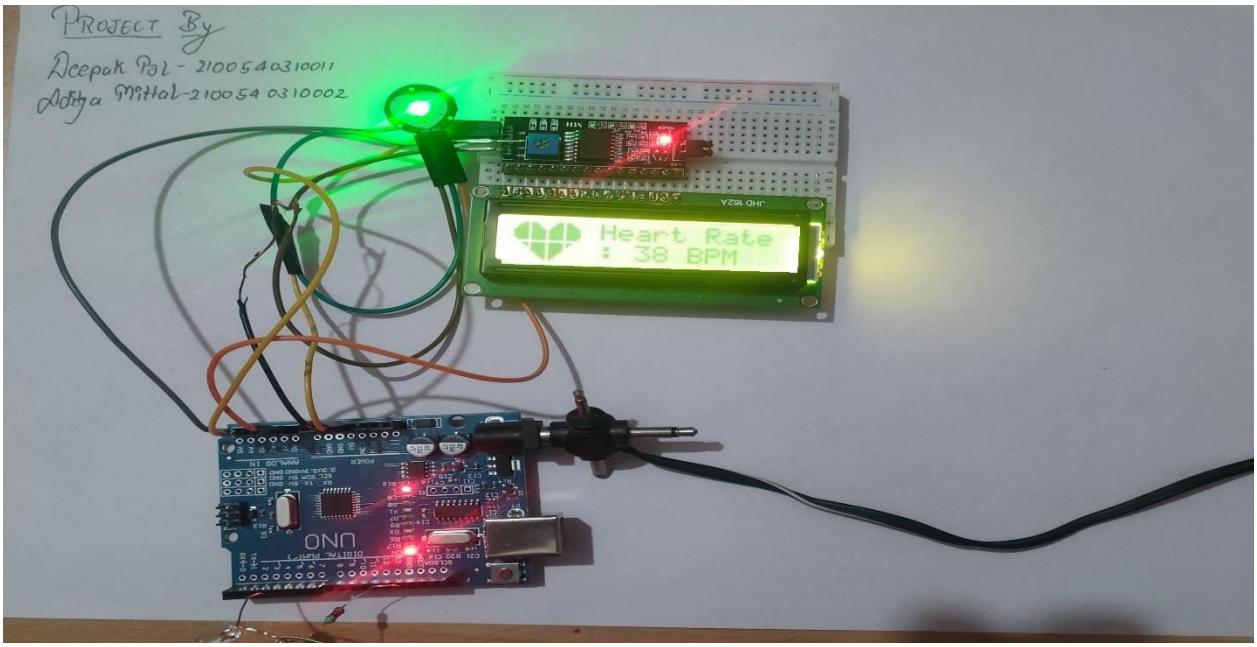

Figure 11

Connection description

Arduino UNO	**Pulse Sensor/heartbeat sensor**
A0 Pin	OUT Pin
VCC	VCC
GND	GND
Arduino UNO	**I2C Module**
A4 Pin (SDA)	SDA Pin
A5 Pin (SCL)	SCL Pin
VCC	VCC
GND	GND

16*2 LCD Display	**I2C Module**	
16 Pin connect	16 Pin connect	
Arduino UNO	LED	220-ohm Resistor
3 Pin	Anode Terminal	
GND		Terminal 1
	Cathode Terminal	Terminal 2

7. Code

```
#define USE_ARDUINO_INTERRUPTS true //--> Set-up low-level interrupts for most acurate BPM math.

#include <PulseSensorPlayground.h> //--> Includes the PulseSensorPlayground Library.

#include <LiquidCrystal_I2C.h> //--> Includes the LiquidCrystal Library.

LiquidCrystal_I2C lcd(0x27,16,2);



const int PulseWire = 0; //--> PulseSensor PURPLE WIRE connected to ANALOG PIN 0

int LED_3 = 3; //--> LED to detect when the heart is beating. The LED is connected to PIN 3 on the Arduino UNO.

int Threshold = 550; //--> Determine which Signal to "count as a beat" and which to ignore.

                //--> Use the "Gettting Started Project" to fine-tune Threshold Value beyond default setting.

                //--> Otherwise leave the default "550" value.


byte heart1[8] = {B11111, B11111, B11111, B11111, B01111, B00111, B00011, B00001};

byte heart2[8] = {B00011, B00001, B00000, B00000, B00000, B00000, B00000, B00000};

byte heart3[8] = {B00011, B00111, B01111, B11111, B11111, B11111, B11111, B01111};

byte heart4[8] = {B11000, B11100, B11110, B11111, B11111, B11111, B11111, B11111};

byte heart5[8] = {B00011, B00111, B01111, B11111, B11111, B11111, B11111, B11111};
```

```
byte heart6[8] = {B11000, B11100, B11110, B11111, B11111, B11111, B11111, B11110};

byte heart7[8] = {B11000, B10000, B00000, B00000, B00000, B00000, B00000, B00000};

byte heart8[8] = {B11111, B11111, B11111, B11111, B11110, B11100, B11000, B10000};

//---------------------------------------


int Instructions_view = 500; //--> Variable for waiting time to display instructions on LCD.


PulseSensorPlayground pulseSensor; //--> Creates an instance of the PulseSensorPlayground object called "pulseSensor"


//--------------------------------------------------------------------------------void setup

void setup() {

  Serial.begin(9600);//--> Set's up Serial Communication at certain speed.

  lcd.begin(); //--> Initializes the interface to the LCD screen, and specifies the dimensions (width and height) of the display


  //---------------------------------------Create a custom character (glyph) for use on the LCD

  lcd.createChar(1, heart1);

  lcd.createChar(2, heart2);

  lcd.createChar(3, heart3);

  lcd.createChar(4, heart4);

  lcd.createChar(5, heart5);

  lcd.createChar(6, heart6);

  lcd.createChar(7, heart7);

  lcd.createChar(8, heart8);

  //---------------------------------------


  lcd.setCursor(0,0);
```

```
  lcd.print(" HeartBeat Rate ");
  lcd.setCursor(0,1);
  lcd.print("   Monitoring   ");

  //----------------------------------------Configure the PulseSensor object, by assigning our variables to it.
  pulseSensor.analogInput(PulseWire);
  pulseSensor.blinkOnPulse(LED_3); //--> auto-magically blink Arduino's LED with heartbeat.
  pulseSensor.setThreshold(Threshold);
  //----------------------------------------

  //----------------------------------------Double-check the "pulseSensor" object was created and "began" seeing a signal.
  if (pulseSensor.begin()) {
    Serial.println("We created a pulseSensor Object !"); //--> This prints one time at Arduino power-up,  or on Arduino reset.
  }
  //----------------------------------------

  delay(2000);
  lcd.clear();
}
//--------------------------------------------------------------------------------

//----------------------------------------------------------------------------------void loop
void loop() {
  int myBPM = pulseSensor.getBeatsPerMinute(); //--> Calls function on our pulseSensor object that returns BPM as an "int". "myBPM" hold this BPM value now.

  //----------------------------------------Condition if the Sensor does not detect the heart rate / the sensor is not touched.
```

```
 if (Instructions_view < 500) {
   Instructions_view++;
 }


 if (Instructions_view > 499) {
   lcd.setCursor(0,0);
   lcd.print("Put your finger ");
   lcd.setCursor(0,1);
   lcd.print("on the sensor   ");
   delay(1000);
   lcd.clear();
   delay(500);
 }
 //----------------------------------------


 //----------------------------------------Constantly test to see if "a beat happened".
 if (pulseSensor.sawStartOfBeat()) { //--> If test is "true", then the following conditions will be executed.
   Serial.println("♥  A HeartBeat Happened ! "); //--> Print a message "a heartbeat happened".
   Serial.print("BPM: "); //--> Print phrase "BPM: "
   Serial.println(myBPM); //--> Print the value inside of myBPM.


   //----------------------------------------Displays a "Heart" shape on the LCD.
   lcd.setCursor(1,1);
   lcd.write(byte(1));
   lcd.setCursor(0,1);
   lcd.write(byte(2));
   lcd.setCursor(0,0);
   lcd.write(byte(3));
   lcd.setCursor(1,0);
```

```
    lcd.write(byte(4));
    lcd.setCursor(2,0);
    lcd.write(byte(5));
    lcd.setCursor(3,0);
    lcd.write(byte(6));
    lcd.setCursor(3,1);
    lcd.write(byte(7));
    lcd.setCursor(2,1);
    lcd.write(byte(8));
    //---------------------------------------

    //---------------------------------------Displays the BPM value on the LCD.
    lcd.setCursor(5,0);
    lcd.print("Heart Rate");
    lcd.setCursor(5,1);
    lcd.print(": ");
    lcd.print(myBPM);
    lcd.print(" ");
    lcd.print("BPM     ");
    //---------------------------------------

    Instructions_view = 0;
  }
  //---------------------------------------

  delay(20); //--> considered best practice in a simple sketch.
}
```

8. Applications

- A simple project involving Arduino UNO, 16×2 LCD, and a Heartbeat Sensor Module is designed here which can calculate the heart rate of a person.
- This project can be used as an inexpensive alternative to Smart Watches and other expensive Heart Rate Monitors.

9. Merits and Demerits

Merits

- Firstly, it is simple to use and provides single-pin Data output which is kind of easier to handle as compared to other sensors like MAX3010x series sensors which use the I2C protocol.
- Moreover, for small DIY projects, this sensor is quite accurate and data extracting and processing is also simple. Also, it can be used with almost any microcontroller.
- Greenlight used for sensing the pulses is more useful than the red light as per the principle of the Photoplethysmogram while the other sensor like max & MAX30102, MAX30100 uses red light

Demerits

- The main disadvantage of this sensor is that the distance between the skin and the sensor affects a lot, also the shade of the skin varying from while to dark reduces the light entering into the blood vessels.
- The pulse rate sensor sometimes shows inappropriate reading and can sometimes change the average of the data also may arise a problem.
- As all the electronic parts are very open to the skin, they might be dangerous if not used carefully. Also, the pins for powering and delivering the data are at the end parallel to PCB not perpendicular

10. Future Scope

Using heart beat sensor, we can study the heart's function in a simple and efficient manner. This sensor senses the blood flow through the ear lobe. The earlobe projects a noise free and better measurement of blood flow in comparison to any device that is wearable at the wrist.

Further improvements can be applied to this project to enhance its performance: - Design robust system to improve measuring efficiency even in the presence of noise. In addition to propose a new method for efficient transmission of data between the MCU and the Android application. - To ensure the accuracy of heart rate monitor device, more testing can be performed to larger number of people with different ages and weights. - Replace the LM35 with specific temperature sensor of body measurement in order to make it more accurate and more functional to use. - More vital signs parameters should be added to increase the value of the project to the patients. These can include: Blood Pressure, Respiratory Rate and other parameters. - Implement pulse and other parameters measurements using the mobile phone camera along with other built-in sensors in order to obtain these parameters on demand if the patient started experiencing some symptoms or abnormalities. - The MCU should send a control signal along with the measured data when detect heart attack and the buzzer is turned ON. The control signal should enable GPS, instruct the application to send an SMS containing the measured data and the patient’s location to the medical emergency and emergency contacts of the patient in order to get an ambulance and notify his relatives. - The device should be miniaturized into a PCB making its weight lighter in order to make the device commercial for public use. - Portable battery unit for the device to provide required power by the sensors and MCU.

11. Conclusion

Upload the code to Arduino UNO and Power on the system. The Arduino asks us to place our finger on the sensor and press the switch. Place any finger (except the Thumb) in the sensor clip and push the switch (button). Based on the data from the sensor, Arduino calculates the heart rate and displays the heartbeat in bpm.

12. References

1.https://techatronic.com/heart-beat-sensor-using-arduino-bpm-monitor/#google_vignette

2. https://www.scribd.com

3. https://www.electronicsforyou.com

4. https://www.wikipedia.org

5. https://www.engineersgarage.com

7. Yuri ilyin. Technology that see through the walls. 2015.