

**Department of Computer Science**

**American International University-Bangladesh**

**Advanced Operating System**

**Fall 2020-2021**

**IOT BASED PATIENT MONITORING SYSTEM**

**Group Member Information:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Name | Id | Course Title | Section |
| 1 | MD.SHIHAB UDDIN | 17-34716-2 | Advanced |  |
| 2 | MOSTAHID IBNA ALAM | 18-36673-1 | Operating System | A |
| 3 | RUBAB ZAMAN ANIKA | 18-38860-3 |  |  |

**Submitted to**

**ARIFUR RAHMAN**

Department of Computer Science

Faculty of Science and Technology

**Date of Submission**

19-12-2020

**Project Idea:-**

Based on expanding healthcare costs, less ICU, staff and doctors in the medical field they have to fight against the Covid-19, we thought we should make a patient monitoring system, which can provide medical healthcare service for Covid-19 patients at less cost. In our country, more than 400k plus patients are affected by Covid-19 and still counting. And there is a huge chances that the patient number can increase rapidly in the upcoming few months in winter as we are going towards a second wave of Coronavirus. And still we don’t have enough beds and ICU for the patients, which is very alarming situation. That is why we wanted to make a patient monitoring system, which can provide good service for Covid-19 patients with less cost.

**Component List:-**

1. MLX90614 Temperature Sensor

2. AD8232 ECG Measurement Module Kit

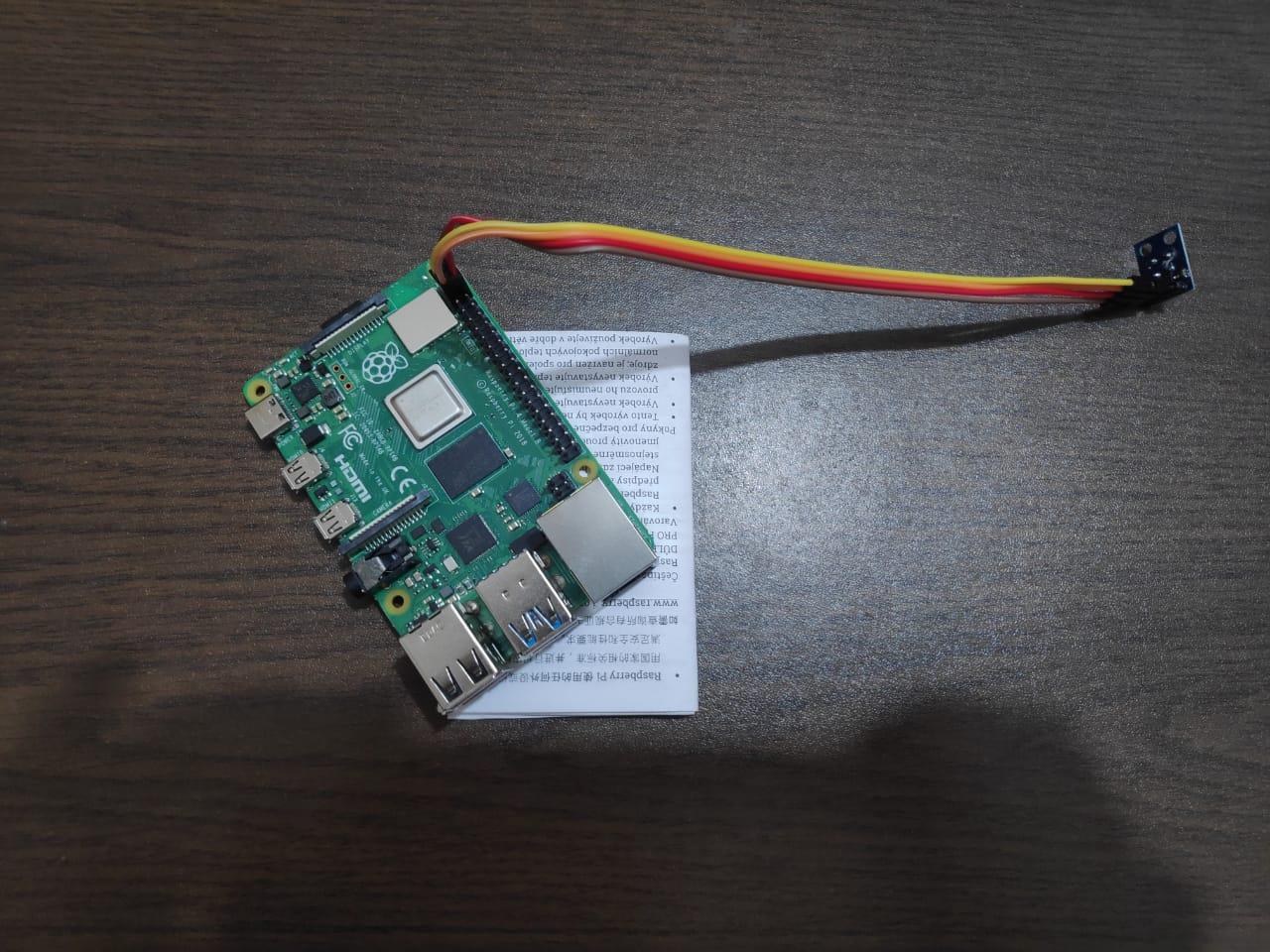
3. Raspberry pi 4b

4. Arduino Uno

5.16\*2 LCD display

**Project Implementation Description and Code:**

**Step-1:** First, we connected the MLX90614 temperature sensor in Raspberry pi.



importing Library:

Firstly installed library and drivers from melexis github page.

Sample code for checking temperature sensor:

from smbus2 import SMBus

from mlx90614 import MLX90614

bus = SMBus(1)

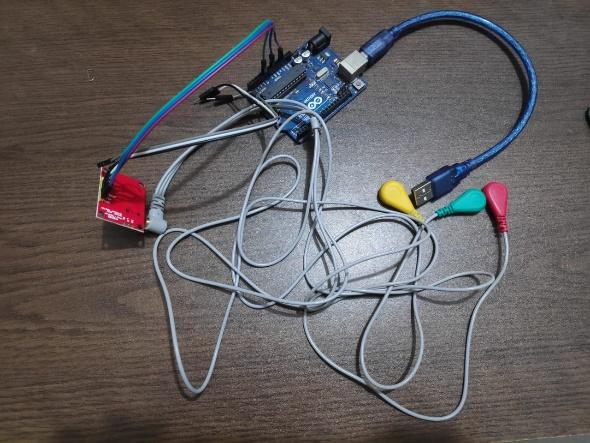
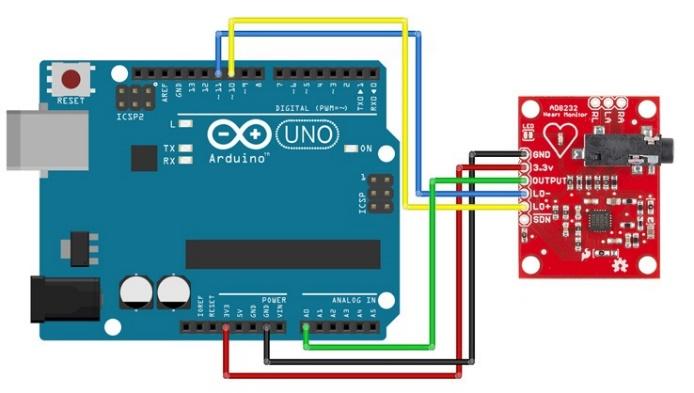
sensor = MLX90614(bus, address=0x5A)

print ("Ambient Temperature :", sensor.get\_ambient())

print ("Object Temperature :", sensor.get\_object\_1())

bus.close()

And we checked if it was working properly or not. When we found out that, it was working properly, we connected AD8232 ECG measurement module kit in Arduino



Here is the code when we checked it in the arduino IDE:

void setup() {

// initialize the serial communication:

Serial.begin(9600);

pinMode(10, INPUT); // Setup for leads off detection LO +

pinMode(11, INPUT); // Setup for leads off detection LO -

}

void loop() {

if((digitalRead(10) == 1)||(digitalRead(11) == 1)){

Serial.println('!');

}

else{

// send the value of analog input 0:

Serial.println(analogRead(A0));

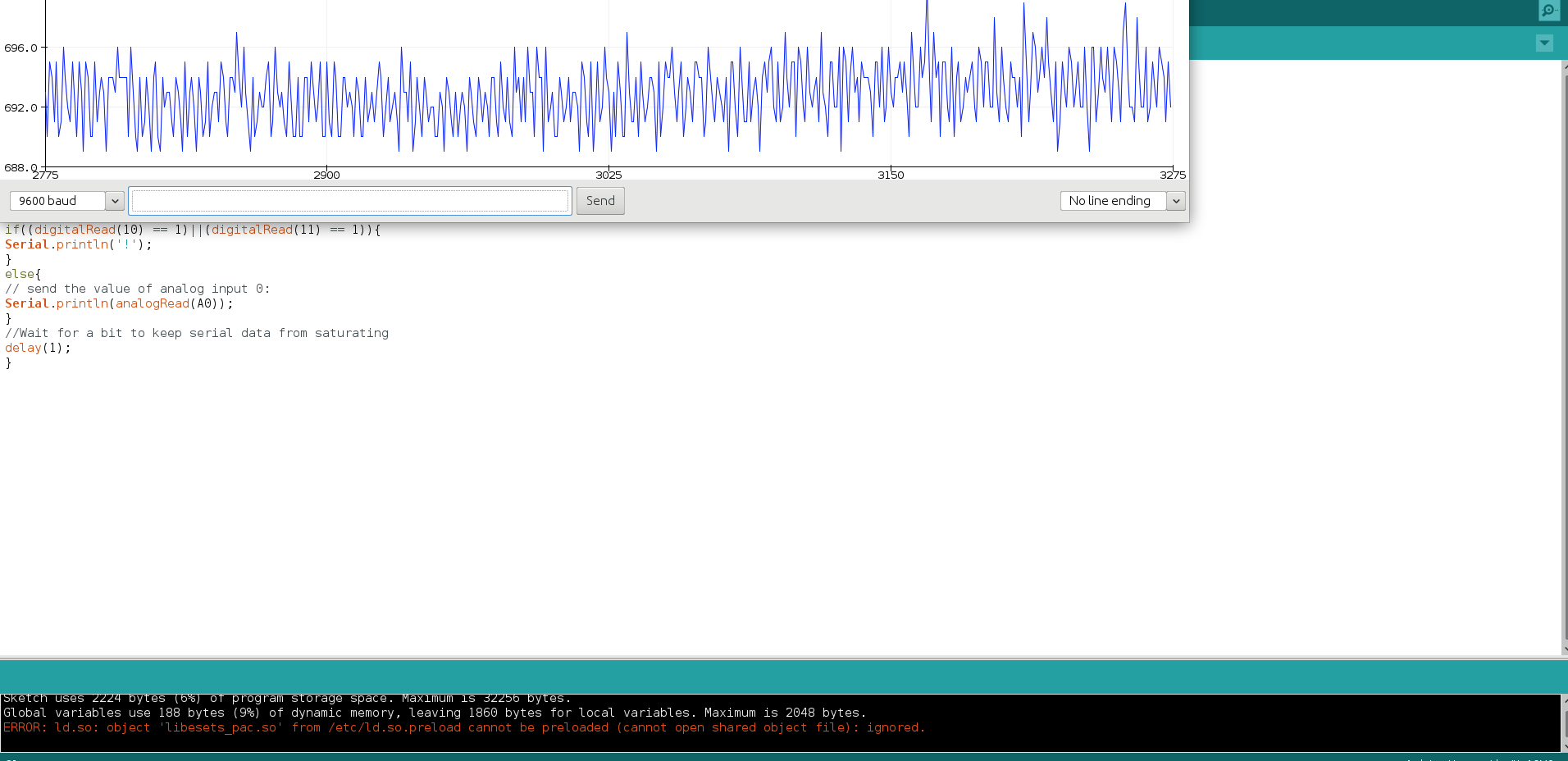
}

//Wait for a bit to keep serial data from saturating

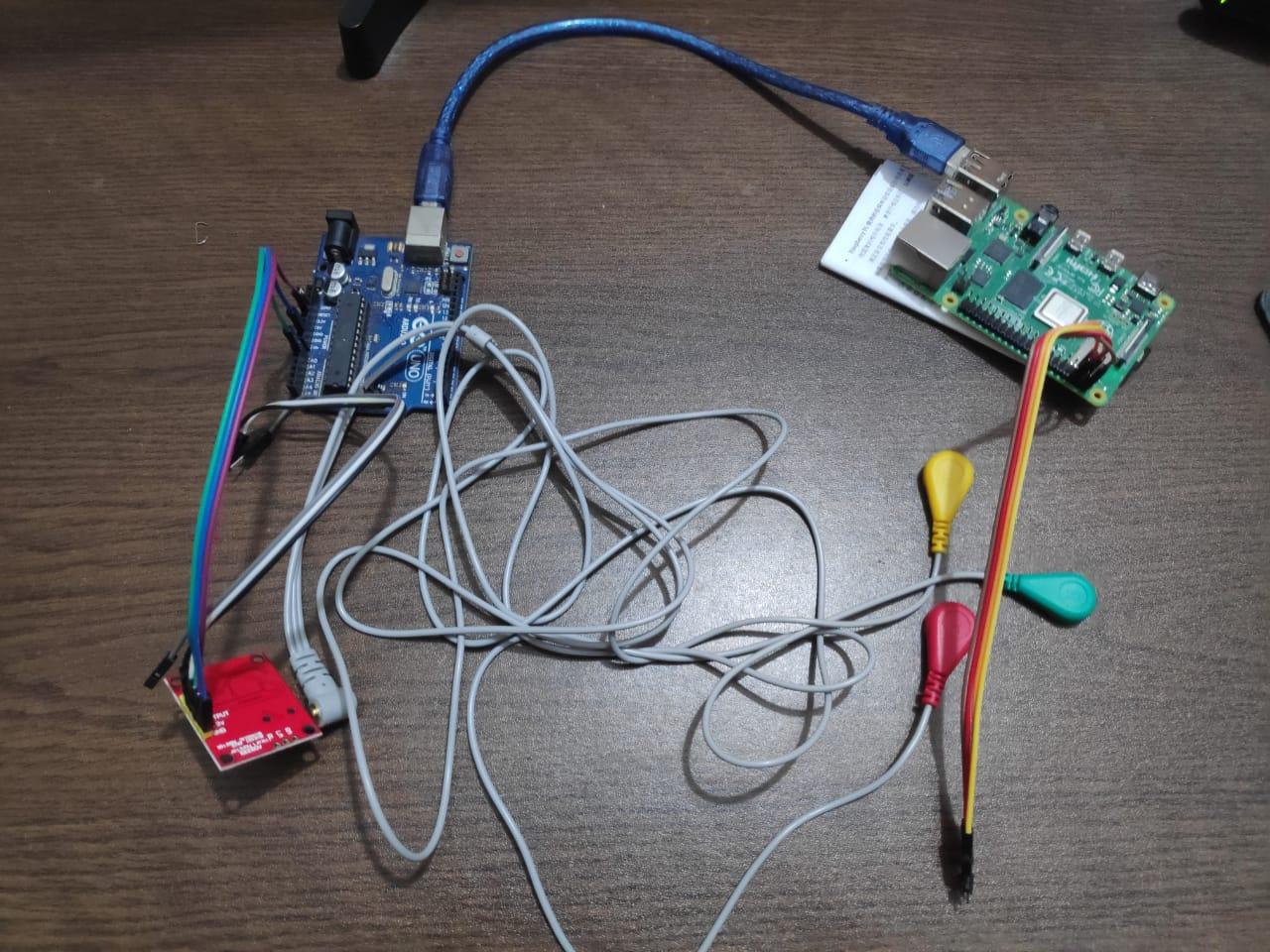
delay(1);

}

**ECG Output in Arduino IDE:**

****

Then we connected it with raspberry pi



to see if it was taking heart rate properly and was it working accurately.

To make things alright we firstly installed all the libraries like matplotlib,numpy.

Code for Raspberry pi to plot

import numpy as np

import matplotlib as mpl

import matplotlib.pyplot as plt

import matplotlib.animation as animation

import serial

import platform

print("Python version: " + platform.python\_version())

print("matplotlib version: " + mpl.\_\_version\_\_)

fig, ax = plt.subplots()

line, = ax.plot(np.random.rand(10))

ax.set\_ylim(0, 1030)

xdata, ydata = [0]\*100, [0]\*100

SerialIn = serial.Serial("/dev/ttyACM0",9600)

def update(data):

line.set\_ydata(data)

return line,

def run(data):

global xdata, ydata

x,y = data

if (x == 0):

xdata = [0]\*100

ydata = [0]\*100

del xdata[0]

del ydata[0]

xdata.append(x)

ydata.append(y)

line.set\_data(xdata, ydata)

return line,

def data\_gen():

x = 9

while True:

if (x >= 9):

x = 0

else:

x += 0.1

try:

inRaw = SerialIn.readline()

inInt = int(inRaw)

except:

inInt = 0

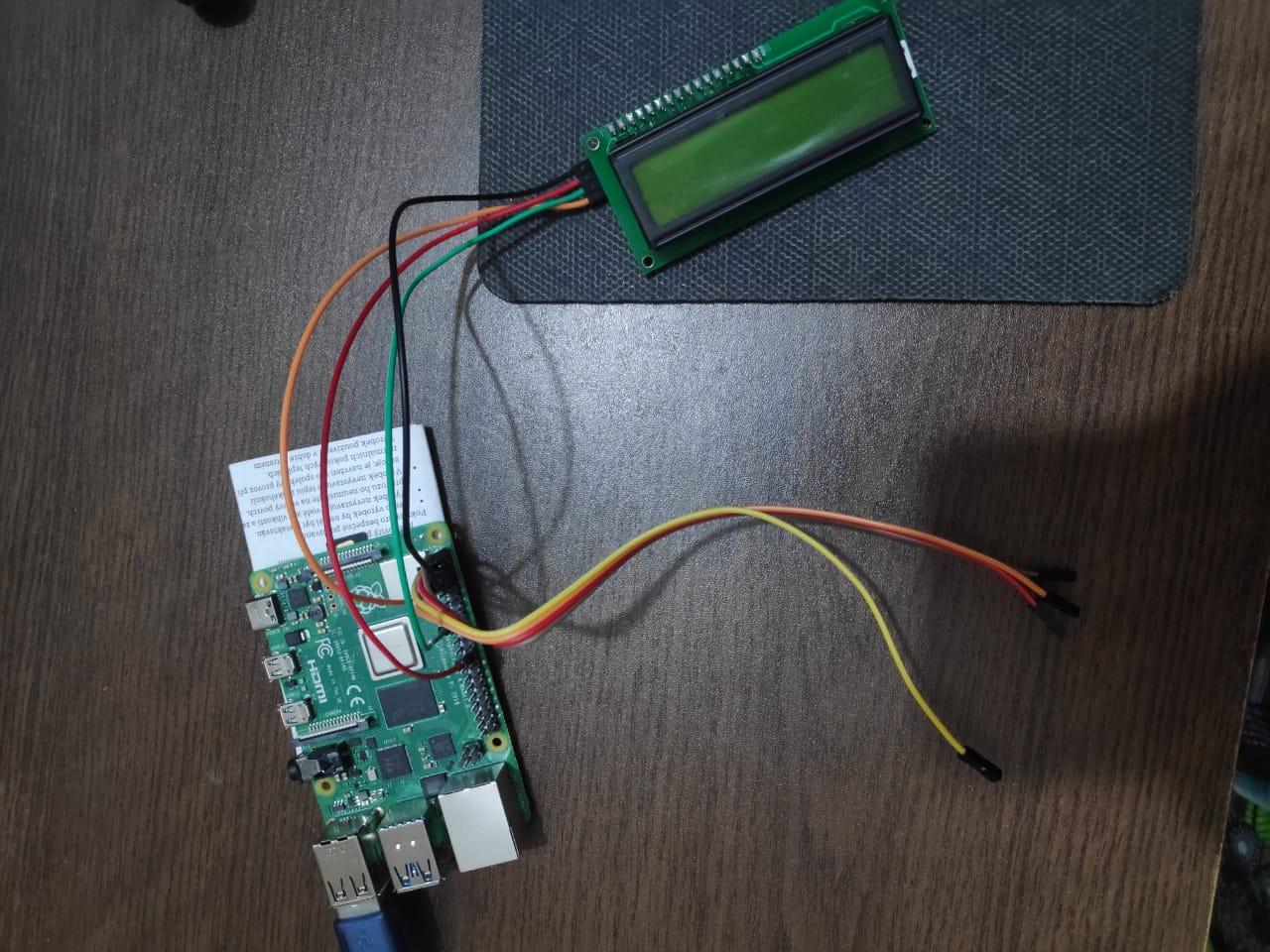
yield x, inInt

ani = animation.FuncAnimation(fig, run, data\_gen, interval=0, blit=True)

plt.show()

**Step-2:** Then we connected everything together to see and check if all of the components were working accurately.

**Step-3:** When we saw that every component was working properly, we connected the display with the device. We can see the heart rate and body temperature on the display. And all of this information can be seen by patient guardian and doctor through thingspeak or Blynk app. These datas can be accessed through a web interface if frontend development is developed more.



Email code:

import smtplib

from email.mime.multipart import MIMEMultipart

from email.mime.text import MIMEText

toadd="emon50359@gmail.com"

email="mdshihabuddinv@gmail.com"

password=”dsaioptvad55mavhnhnvhbdsdsqei"

temp=105

msg = MIMEMultipart()

msg['From'] = email

msg['To'] = toadd

msg['Subject'] = "Test now"

body = '''Dear Mr. XYZ,

This is a sample code to send email.testing Emon

Thanks,

Python

'''

msg.attach(MIMEText(body, 'plain'))

text = msg.as\_string()

server=smtplib.SMTP\_SSL("smtp.gmail.com",465)

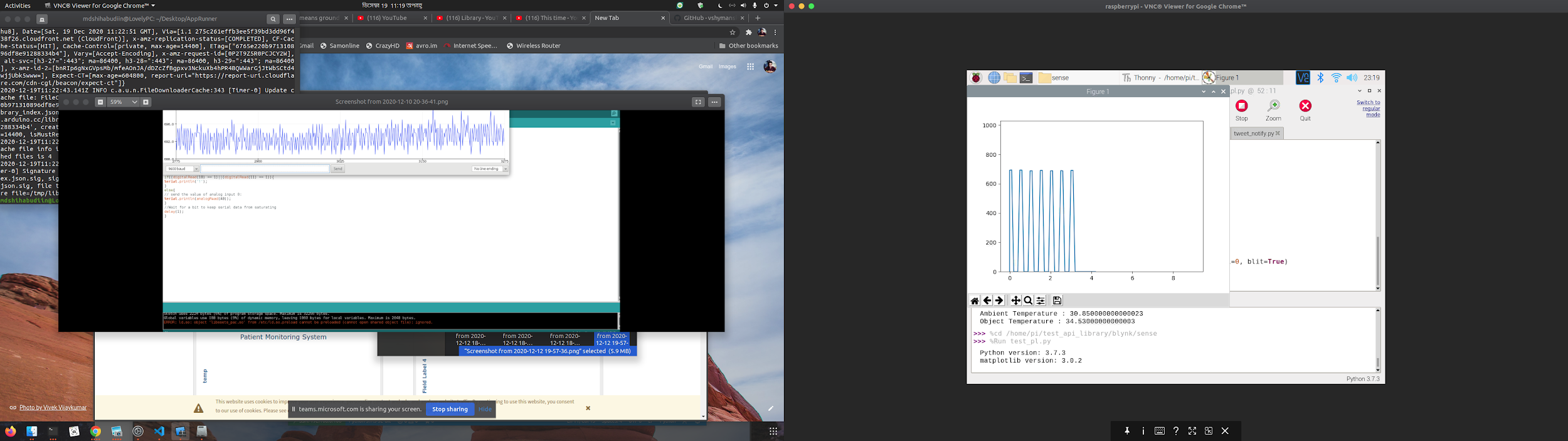
server.login(email,password)

if(temp>100):

server.sendmail(email,toadd,text)

server.quit()

**Sensor data from raspberry pi when arduino was connected:**



**Step-4:** If any of the information on the display plate is risky or good it will also show on the display through a report and both doctor and patient guardian can also see it through the internet .

we downloaded the library from following link:

https://github.com/vshymanskyy/blynk-library-python

Code for Blynk:

import BlynkLib

temp=50

BLYNK\_AUTH = 'authdhhdh'

# Initialize Blynk

blynk = BlynkLib.Blynk(BLYNK\_AUTH)

if temp>100:

blynk.notify("patient has high temp")

blynk.virtual\_write(2, '{:.2f}'.format(temp))

try:

while True:

blynk.run()

except KeyboardInterrupt:

sys.exit(0)

Sending into Thingspeak:

import sys

import RPi.GPIO as GPIO

import os

from urllib.request import urlopen

import smbus

import time

from ctypes import c\_short

import serial

ser = serial.Serial('/dev/ttyACM0',9600)

ecg=""

key="fhjhjfjf75656"

def main():

URL='https://api.thingspeak.com/update?api\_key=%s' %key

print("wait")

while True:

ecg=ser.readline())

finalURL=URL+"&field1="str(ecg)

print(finalURL)

s=urlopen(finalURL)

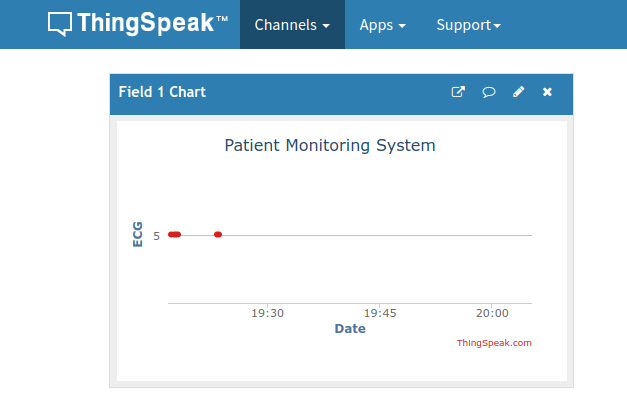
print(ecg)

s.close()

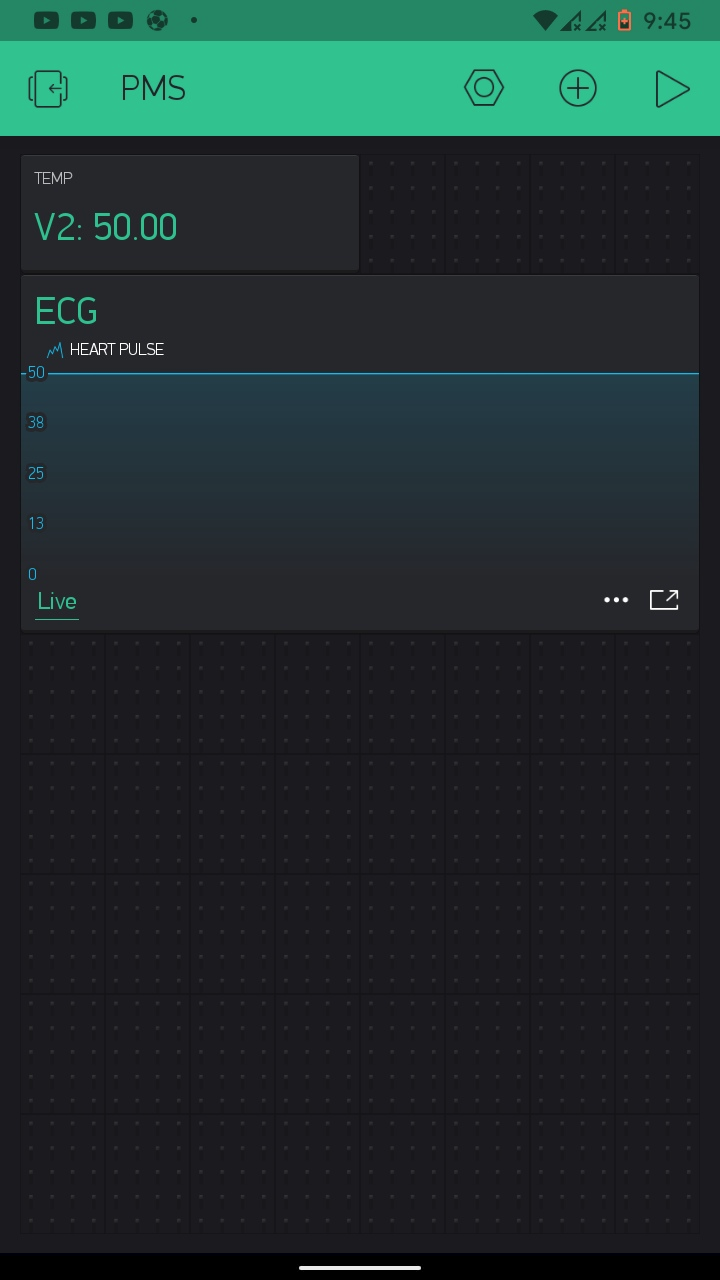
time.sleep(1)

if \_\_name\_\_=="\_\_main\_\_":

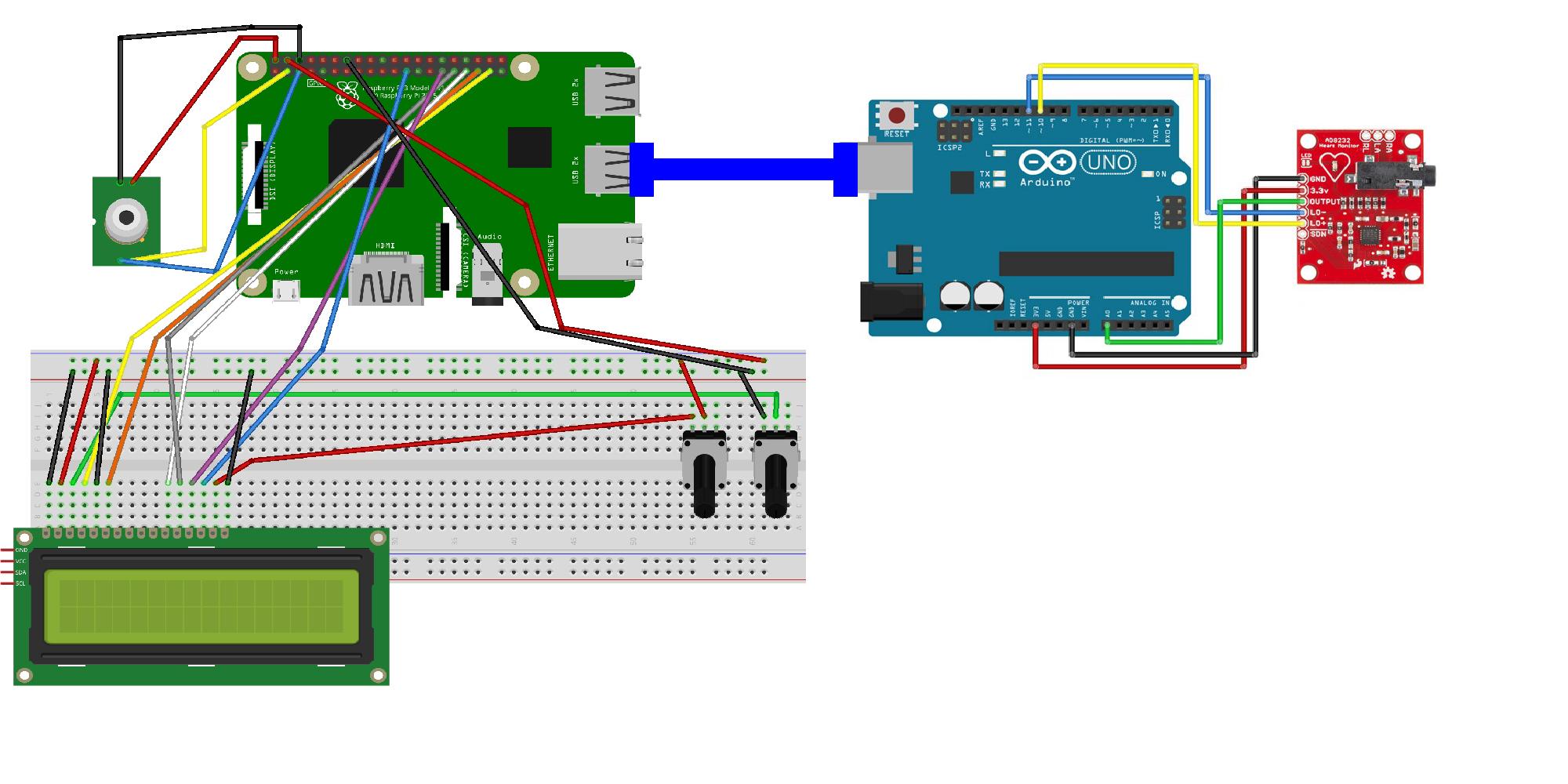
main()



View in Blynk:



**Circuit Diagram:-**

****

**References:-**

1. <https://github.com/vshymanskyy/blynk-library-python> for accessing blynk IOT platform
2. <https://www.melexis.com/en/product/MLX90614/Digital-Plug-Play-Infrared-Thermometer-TO-Can> for datasheet and driver download
3. <https://www.arduino.cc/en/software> for installing arduino softwares and necessary drivers
4. <https://thingspeak.com/> for sensor data analysis
5. <https://fritzing.org/> fritzing software for diagram design