BACHELOR OF SCIENCE IN ELECTRONIC AND TELECOMMUNICATIONS ENGINEERING

IoT Based Covid-19 Patient Detection and Remote Monitoring System

Submitted By

Saad Ibne Omar

Matric ID: **T171030**

Imran Akber

Matric ID: **T171044**

Supervised By

Mohammad Woli Ullah

Lecturer

Department of ETE

International Islamic University Chittagong

Department of Electronic and Telecommunications Engineering

International Islamic University Chittagong

Kumira, Sitakunda, Chattagram

October, 2021

DECLARATION OF CANDIDATE

It is hereby declared that the work presented in this project or any part of this project has not been submitted elsewhere for the award of any degree or diploma, does not contain any unlawful statement.

(Signature of Candidate)

Name of the candidate: Saad Ibne Omar

Student Id: T171030

Academic Year:2021

Date:

(Signature of Candidate)

Name of the candidate: Imran Akber

Student Id: T171044

Academic Year:2021

Date:

DEDICATION

This effort honors our adored Parents and Teachers.

Saad Ibne Omar

Imran Akber

ACKNOWLEDGEMENT

In the name of Allah, the most Beneficent and most Merciful

We are grateful to Allah for the opportunities, hardships, and strength that have been bestowed upon us in order to complete the thesis. We got a wealth of experience over this process, both intellectually and personally. To begin, we would like to express our heartfelt gratitude to our supervisor **Mohammad Woli Ullah** for his direction, understanding, patience, and most significantly, for providing positive support and a warm spirit that enabled us to complete this thesis. Having him as our supervisor has been a tremendous joy and honor. We would like to give thanks to all of our dear friends, particularly **Mobtasim Fuad**, **Fahim Iqbal Sakib**, and **Lokman Hossain**, who stood with us and supported us through thick and thin. May Allah bestow success and honor upon the aforementioned individuals throughout their lives.

Abstract

We need a health care system that can handle a huge number of patients in a crisis like COVID-19 that is adaptable, accessible, and quick. The Internet of Things (IoT) has allowed us to create a system that allows doctors to remotely monitor COVID patients while also practicing social distancing. A remote patient monitoring system is designed and implemented in this work. The patient can be monitored using a wearable device that includes a pulse sensor, a body temperature sensor, and a spo2 sensor. The sensors' outputs will be sent to a website data base, where authorized doctors can access them at any time and from any location. The system will also analyze the current state of a patient's health, and doctors will be made aware of it. Patients who are in urgent condition and require rapid medical attention will be identified by the doctors using this system. Our web platform allows doctors to see a patient's complete medical history at any time and provide any advice that is required. Additionally, the pressure on current health care facilities will be relieved.

Table of Contents

DECLARATION OF CANDIDATE I

DEDICATION II

ACKNOWLEDGEMENT III

Abstract IV

Table of Figures VIII

List of Tables IX

Chapter 1 1

Introduction 1

1.1 Introduction 1

1.2 Objectives 2

1.3 Methodology 2

1.4 Motivation 3

1.5 Outline of the Report 3

Chapter 2 5

Literature Review 5

2.1 Introduction 5

2.2 Background Review 5

Chapter 3 9

Equipment Descriptions and Circuit Diagram 9

3.1 Arduino UNO: 9

3.2 Features of the Arduino Uno Board: 9

3.3 Programming: 11

3.4 Input and Output 11

3.5 Communication 12

3.6 Automatic (Software) Reset Uno 13

3.7 Arduino UNO Pin Configuration:..... 14
3.8 Arduino Uno Pinout Configuration 16
3.9 Arduino Uno Technical Specifications..... 17
3.10 ESP8266 NodeMCU:..... 18
NodeMCU Development Board Pinout Configuration 19
NodeMCU ESP8266 Specifications & Features 20
32-bit Tensilica Processor..... 21
3.15 Compactness 21
3.17 Buzzer: 21
3.18 Buzzer Pin Configuration 22
3.19 Features and Specifications of the Buzzer..... 22
3.20 LM35 Temperature Sensor: 23
3.21 Pin Configuration:..... 24
3.22 LM35 Regulator Features: 24
3.23 MAX30100 Pulse Oximeter 24
3.24 Features..... 25
3.25 MAX30100 Pulse Oximeter and Heart Rate Sensor Operation 25
3.26 Max30100 Pulse Oximeter Sensor Pinout: 26
3.27 Battery..... 26
3.28 Technical Specification: 26
3.29 HTML: 27
3.30 Google Sheet:..... 27
3.31 WordPress..... 28
3.32 WordPress Plugins:..... 28
3.33 Elementor Plugin: 29
3.34 Ultimate Member Plugin: 29

Chapter 4 30
System Discussion 30
4.1 System Block Diagram: 30
4.2 System Framework: 30
4.3 Circuit Diagram: 31
4.4 Methodology: 32
4.5 List of Components: 33
4.6 System Flow Chart: 34
Chapter 5 35
Results and Discussion 35
5.1 Introduction 35
5.2 Result 35
5.3 Design of The Website 37
5.4 Discussion: 40
5.5 A comparison of our proposed system with current systems is in order. 41
5.6 Advantages and uses: 41
Chapter 6 42
Conclusion and Future work 42
6.1 Conclusion 42
6.2 Future Work 42
References 43
APPENDIX 47

Table of Figures

Figure 1 Components of a remote patient monitoring system based on an IoT-Cloud architecture [6] 6
Figure 2 Block diagram of the proposed system of “Internet of things (IoT) Based Smart Health Care System” [7] 7
Figure 3 Arduino UNO [24] 9
Figure 4 Arduino UNO Pin Configuration [25] 14
Figure 5 NodeMCU [31] 18
Figure 6 Buzzer Pi [28] 22
Figure 7 LM35 Temperature Sensor [29] 23
Figure 8 GY-MAX30100 [30] 25
Figure 9 Battery [33] 26
Figure 10 Block diagram of our system 30
Figure 11 System Framework of our system 31
Figure 12 Proteus Simulink 31
Figure 13 Flow Chart of IoT Base Covid19 Patient Monitoring System. 34
Figure 14 Hardware 35
Figure 15 Output in Serial Monitor 36
Figure 16 Output in Serial Monitor 36
Figure 17 Output when Temperature is Abnormal 37
Figure 18 Website Login System 37
Figure 19 Registration Page 38
Figure 20 Home Page 38
Figure 21 Patient List Page 39
Figure 22 Patient Records Page 39
Figure 23 Messaging system 40

List of Tables

Table 1 Arduino Pinout Configuration 16
Table 2 Technical Specification 17
Table 3 NodeMCU Pinout Configuration [31] 19
Table 4 Buzzer Pin Configuration [11] 22
Table 5 LM35 Pin Configuration [11] 24
Table 6 comparison of our proposed system with current systems 41

Chapter 1

Introduction

1.1 Introduction

There is now a worldwide common practice of social distance and isolation following the breakout of the novel coronavirus disease pandemic (COVID-19). As a result of these actions, we now have less access to medical treatment.

Many countries' health care systems are already overwhelmed due to the increasing number of daily cases of COVID-19. Using IoT in the development of patient monitoring and detection systems will be our greatest approach for containing this problem.

It's impossible to estimate how quickly the "Internet of Things" will grow. The rapid decline in price of standard IoT components is making it possible for the general public to create new designs and products right at home, too. In the Internet of Things (IoT), people have access to a huge amount of data, and objects equipped with sensors and actuators are able to connect and exchange information amongst one another. Sensors, software, and actuators integrated in devices and systems communicate with one another and exchange data through the internet. It is possible to use the Internet of Things to keep a watchful eye on patients' health, as well as to create smarter homes and cities.

Simply said, IoT is the control of any device over the internet and the facilitation of data and outcomes via the internet. When it comes to communication, internet of things (IoT) gadgets have overcome the limitations of distance due to their use of the internet.

It lowers human involvement, improves efficiency, and provides economic advantage and accuracy in addition to these other benefits.

In the Internet of Things, medical devices transfer patient parameters via a gateway to a central location where they are stored and evaluated. So IoT can be used to keep track of any unforeseen events that occur in the lives of patients.

In our research Patients' heart rate, body temperature and blood oxygen level are being monitored with the help of a specific sensor.

Using our website, doctors will have 24/7 access to this sensor data. As a result, the doctor has an easy time deciding whether or not the patient has to go to the hospital for specialized treatment. This will alleviate some of the strain on hospitals that are already stretched thin by the influx of new patients, and by sifting through the available data, it may be feasible to identify any further cases of covid-19. Early identification of Covid-19 helps save the lives of patients and prevents the future spread of the virus in the majority of cases of covid-19 patients. That's where an IoT-based patient monitoring solution comes in.

87% of healthcare firms will be using Internet of Things (IoT) technology, according to Aruba Network research, and 76% expect it to have a significant impact on the healthcare industry. [1] [2]

Health IoT and utilizing health-related data from IoT devices are leading industries in several places, such as the United States. IoT is already being used for sensitive data by 30% of healthcare firms. [1] [2]

1.2 Objectives

- To build a continuous and real time remote monitoring system with emergency alert which will be accessible from any place through internet.
- To create a database for preserving detailed medical history of the patient.
- Doctors can easily access the patient medical record and give necessary advice remotely.
- Doctors will be able to monitor large amount of patient in short amount of time by remote monitoring.

1.3 Methodology

The procedure for applying the Methodology is as follows:

- Problem Selection

- Background Study
- Literature Review
- Deciding the Problem Statement
- Identifying Research Objectives
- Designing the Research
- Learning required analytical and simulation tools
- Designing circuit diagram
- Designing the website.
- Assemble the circuit as per circuit diagram and test it.
- Collect the sensor data by Arduino and send it to the website using Wi-Fi module.
- Write a report based on the work done.

1.4 Motivation

- In pandemic situation like COVID-19 when social distancing must be maintained, monitoring of patient becomes difficult for the doctors.
- Our system ensures continuous monitorization of the patients.
- Our system ensures remote monitorization.
- Our system keeps track of critically ill patients in real time.
- Our system provides doctors detailed medical history of the patients.
- Because of our system, healthcare facilities will have less load.

1.5 Outline of the Report

In this study, we explained and discussed our project in six chapters. The following are the main points of our project:

- CHAPTER 1: This section contains the introduction. The system's introduction has been addressed in this chapter.
- CHAPTER 2: Includes a review of the literature. This chapter goes on some of the earlier work that has been done on this system.
- CHAPTER 3: The process of development of the whole system, as well as all necessary circuit diagrams and descriptions of all necessary equipment, are included in the equipment descriptions.
- CHAPTER 4: There includes a methodology section that discusses the approach of entire system creation as well as all relevant circuit diagrams, block diagrams, and flow charts.
- CHAPTER 5: Contains a result and discussion section in which the outcome of our system is discussed.
- CHAPTER 6: Includes a conclusion that discusses the conclusion and future work, as well as references and appendices.

Chapter 2

Literature Review

2.1 Introduction

A combined approach is sought in this research in order to build a suitable healthcare system that is both beneficial to patients and beneficial to doctors. The primary goal of this project was to produce a easily accessible, small product that was also simple to use and transport. There are numerous medical monitoring systems on the market, each with its own set of limitations, such as inefficient use of power, bulkiness, wired setup, and reaction time. To get around these problems, a wireless wearable system is a lifesaver. Literature has centered on a few key and related works.

2.2 Background Review

The following are a few notable former activities that are connected to this one:

In this paper [3] author shows how to detect of possible Covid-19 cases by using IoT based sensor data. The system will send the data to the doctors via internet using Wi-fi who will examine the data and decide whether the patient should go through PCR testing or not.

In case of this paper [4] author shows system for pandemic situation like COVID-19 when people have to maintain social distancing and stay at home. IoT based devices will help to control the home remotely and sends patients information remotely via mobile application to doctor who can prescribe the patient via application also.

In this paper [5] It's a low power consumption IoT based system which is accessible from everywhere by using internet. And it also helps minimize the time wastage in a critical health situation.

The potential and difficulties for IoT in fulfilling this vision of the future of health care were highlighted in [6]. He hypothesized that wearable sensors may easily combine numerous physiological measures, allowing for the storage of data with a considerably higher sampling rate over much longer time scales.

The author of that study emphasized in the abstract on how networked sensors, whether worn on the body or integrated in our living surroundings, allow for the collection of rich data on our physical and mental health. Such data, if collected on a regular basis, pooled, and successfully mined, has the potential to revolutionize the health-care environment for the better. The availability of data at previously unimagined scales and temporal longitudes, combined with a new generation of intelligent processing algorithms, has the potential to: (a) aid in the evolution of medical practice away from the current reactive paradigm of diagnosis and treatment to a proactive framework for disease prognosis at an early stage, as well as prevention, cure, and ovulation prevention. They discuss the benefits and difficulties of IoT in fulfilling this vision of the future of health care in this article.

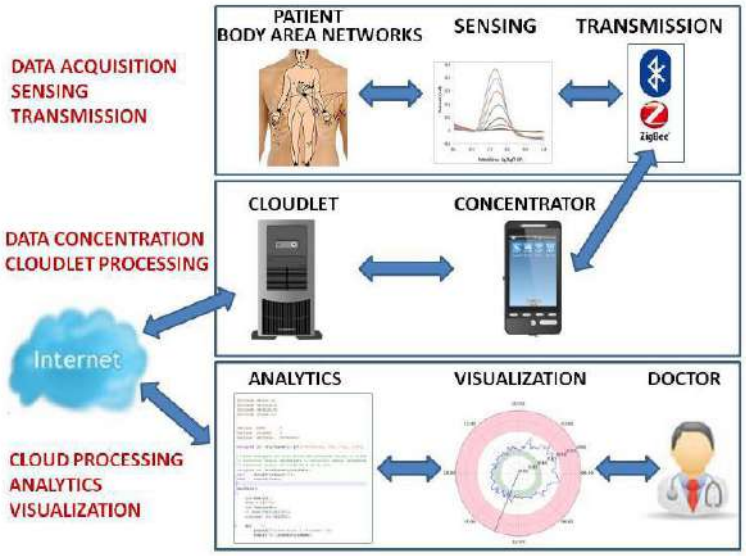


Figure 1 Components of a remote patient monitoring system based on an IoT-Cloud architecture [6]

In order to integrate remote health monitoring technology into clinical practice, they examined the current situation and expected future developments. IoT-equipped wearable sensors offer appealing choices for enabling data observation and recording in the home and business over considerably longer periods than are traditionally done at office and laboratory visits. With the right analysis and easy-to-assimilate visualizations, this wealth of data can revolutionize healthcare while also cutting costs. Prior to developing technologies that integrate seamlessly into clinical practice, several challenges in sensing, analytics, and visualization must be solved.

An Arduino Fio transmitter board, with an xbee module attached, is described in [7] as a way to connect various sensors. The patient's PC is linked to an Arduino, which wirelessly receives the measured values. All of these values are read by LabVIEW. For this reason, LabVIEW creates a URL that may be accessed from any machine.

The author's methodology is as follows: Multiple sensors are connected to the Arduino board and positioned in strategic locations on the human body. The LM35 temperature sensor output is transformed to digital values using the Arduino board's ADC pins. Pulse-rate sensors lose some sensitivity because of a finger's tendency to become slightly opaque when the heart pumps blood through the veins.

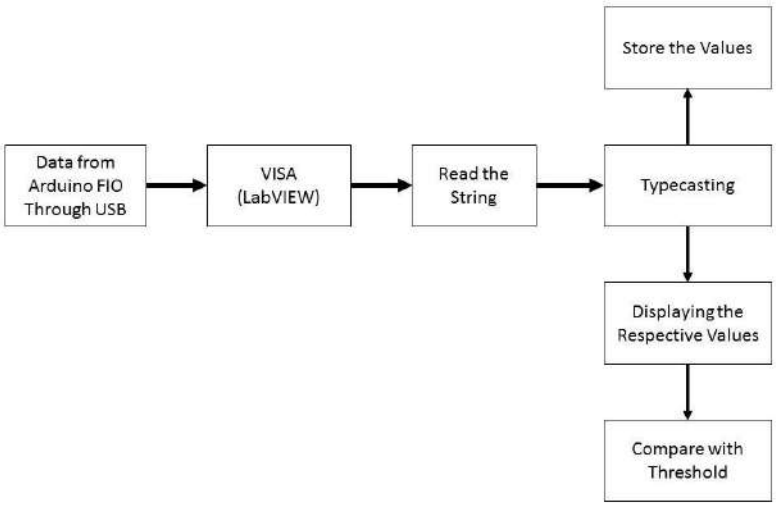


Figure 2 Block diagram of the proposed system of "Internet of things (IoT) Based Smart Health Care System" [7]

An electrical pulse is generated from variations in the detector signal caused by heartbeats. An LED blinks to show the heartbeat, which also serves as an additional pulse indicator. Standard blood pressure monitors are cumbersome, inconvenient, and require assistance to use. They only have to hit the blue button and use the Velcro straps to attach the Monitor to their wrist. They'll know their systolic and diastolic blood pressure in less than a minute. It has a large and easy-to-read display. The biomedical tool package in LabVIEW can also be used to process ECG signals.

In this paper It talked about available network architectures that can be used for data transmission and receiving in IoT. And presents policies and regulations of IoT to the people who are interested in using eHealth care system.

In this paper [6] It showed the opportunity of using vast amount of data in different time frame that is collected by using IoT. It highlighted the challenges for the system before clinical practice.

In this paper [8] Developed a system using Thingsboard that will collect heart rate data and percentage of SpO2 in blood in real time, compare it to normal value and send the data via internet to the person or authority it may concern.

Chapter 3

Equipment Descriptions and Circuit Diagram

3.1 Arduino UNO:

Arduino Uno is a Microcontroller Board 8-bit ATmega328P. The ATmega328P microcontroller has additional components, such as a crystal oscillator, serial communication, voltage control etc. The Arduino Uno features fourteen digital input/output pins, six analog input pins, a USB connector, a power bar, an ICSP header and a reset button.

Figure 3 Arduino UNO [24]

3.2 Features of the Arduino Uno Board:

It has a standard USB port. This gadget is used with USB, as it is a powerful tool.

- The board's chip communicates with your USB port and acts as a virtual serial USB hub.
- The benefit of this technology is that serial communication is an established and true standard, while USB enables easy interfacing with current computers.

- The computer brain, which is frequently referred to as the mega-processor of Arduino, is more available. Physical components include timers, interrupts internally and externally, PWM pins and various forms of sleep.
- It is an open source design, and one of the advantages of being an agnostic platform is that a huge community of people use it and solve it. It is now easy to provide project debugging.
- It runs at 16 MHz and is fast enough for many applications without the microcontroller being overheated.
- It has built-in system voltage yet can handle electricity extremely easily. Even without external power, it can only be charged through a cable. In fact, it can connect up to 12 volts of external power, which is reduced to 5 and 3.3 volts.
- There were 13 digital pins and 6 analog pins on this board. You may add a micro controller to the Arduino board using the pins. These pins are used to boost the computer's computing power.
- It is a 16 MHz clock, which is fast enough for most applications while the microcontroller is not overheated.

It has integrated voltage regulation; however, it is quite compact and has very minimal power. This can be instantly powered from a USB port without additional electricity. Up to 12 volts power supply can be connected and controlled with 5 and 3.3 volts. The board contains 13 digital and analog inputs. These lines are used to connect outside hardware to my Arduino Uno. These pins are used to expand the computer of Arduinos into the real world. Just place your electronic gadgets and sensors in the sockets which match the beginning of each pin.

This has an ICSP connector, so that the USB port may be bypassed and connected as a serial device to Arduino. This port needs to reload your chip if it gets corrupted and doesn't work with your computer anymore.

It has 32 KB of flash memory to store your code.

Finally, it has a button to reset the software of the chip.

Arduino was founded in 2005 by two Italian engineers David Cuartielles and Massimo Banzi in an attempt to keep students in mind to learn how to program and improve their skills in electronics and to apply the microcontroller Arduino uno in the real world.

By accepting data from a number of sensors, the Arduino un microcontroller can perceive the environment and can influence its environment by controlling lights, motors and other actuators. The Arduino programming language (based on Wiring) and the Arduino development environment are used for the microcontroller (based on Processing).

3.3 Programming:

- The Arduino integrated development environment (IDE) is a Java-written multiplatform application developed from IDE for the programming language processing and wiring projects.
- Users may program the Arduino Uno board with the software Arduino.
- Select from the Tools > Board menu "Arduino Uno" (according to the microcontroller on your board).
- The Arduino Uno ATmega328 comes with a boot loader, which enables you to upload fresh code to Arduino Uno without using an additional hardware programmer. The original STK500 protocol connects.
- You may also override the boot loader and program the microcontroller via the ICSP header.
- The firmware source code for the ATmega16U2 (or 8U2 for the rev1 and rev2 boards) is available.

3.4 Input and Output

See mapping between Arduino pins and ports. The mappings of Atmega8, 168 and 328 are the identical.

The 14 digital pins can be used as inputs or outputs using the pinMode() and digitalRead digital Write() utility (). It operates at 5 volts. Each pin can be 20 mA as specified and has an inner pull-up resistor of 20-50kohm (default unconnected) [14].

No, I/O pin for permanent microcontroller damage should exceed a maximum value of 40mA. Furthermore, several pins have particular functions:

- Serial numbers 0 (RX) and 1 (TX) (TX). Serial TTL data is received (RX) and transmitted (TX) using this device. These pins are connected to the equivalent pins of the ATmega8U2 USB-to-TTL Serial chip.
- External interruptions 2 and 3 These pins can be adjusted to trigger an interrupt with a low value, a rising or falling edge or a value change. See the attachInterrupt() function for more information.
- Three, five, six, nine, ten, and eleven are PWM. A 8-bit PWM signal is generated via the analogWrite() function.
- 10 (SS), 11 (M10 (SS), 11 (MOSI), 12 (MISO), 13. (SCK). To communicate with these pins, the SPI library is used.
- SPI, 12 (MISO), 13 (MISO) (SCK). To communicate with these pins, the SPI library is used.
- Number of LED: 13 Digital pin 13 is supplied with a built-in LED. The LED is enabled when the pin is high, and when the pin is low, it is disabled.
- TWI: A4 pins (SDA) and A5 pins (SCL). Support TWI connection with the Wire library.
- Each of the six analog inputs of the Uno, A0 to A5, has a 10 bit resolution (i.e. 1024 different values). They measure by default from 0 to 5 volts, however the AREF pin and analog Reference() function can be used to change the top limit of their range. There are a couple additional pins on the board.
- Reference voltage of the analog inputs, AREF. When using analog reference ().

Reset. Reset the microcontroller by lowering the wire. Usually used to add a reset button to shields that block the reset button of the board.

3.5 Communication

The Arduino Uno features many computer connectivity, Arduino board or other microcontroller communication capabilities. The ATmega328 supports UART TTL (5V) digital pins 0 (RX) and 1 serial connection (TX). An ATmega16U2 can be used

to transmit serial communication through USB and is displayed as a virtual computer interface for programs. The 16U2 firmware uses standard USB COM drivers and no further drivers are required. A.inf file is however required on Windows. The Arduino Software (IDE) is a serial monitor that makes text data to and from the device easy to transmit. The RX and TX LEDs of the board blink when the data are transmitted via USB to serial and USB chip to the computer (but not for serial communication on pins 0 and 1). A serial library software allows serial communication on all Uno digital pins. The ATmega328 also offers I2C (TWI) and SPI communication. The software Arduino (IDE) offers a library that makes it easy to use the I2C bus. Use the SPI communications library.

3.6 Automatic (Software) Reset Uno

Instead of pressing a physical reset button before uploading, the Arduino Uno board is engineered to enable the software to reset on a linked computer. One of the ATmega8U2/16U2 hardware flow control lines (DTR) is coupled to the reset lines of the ATmega328 via a 100 nanofarad condenser. The reset line will be sufficiently long to reset the chip when this line is inserted (taken low). This functionality is used by the Arduino software (IDE) to upload code by pressing the Upload button on the toolbar. This allows a shorter timeout for the bootloader, since the DTR can be synchronized simply if the upload is started. This arrangement has further implications. When Uno is connected to either Mac OS X or Linux, it resets every time a connection is made between the software (via USB). The bootloader runs on the Uno for around half a second. If defective data is to be ignored (e.g. anything other than a fresh code upload), while a connection is open, the first few bytes of data that is supplied to the board are intercepted. If a board diagram obtains one setup or another data when it starts first, ensure that the software it transmits is waiting for a second after the connection is opened and before to submission of this information. In the Uno board, a track can be broken to disable the auto-reset. The pads can be soldered to reactivate it on either side of the track. "RESET-EN" is marked. You may also disable the self-reset by adding a 5V 110 ohm resistor to the restart wire; check for details on this forum thread.-+

3.7 Arduino UNO Pin Configuration:

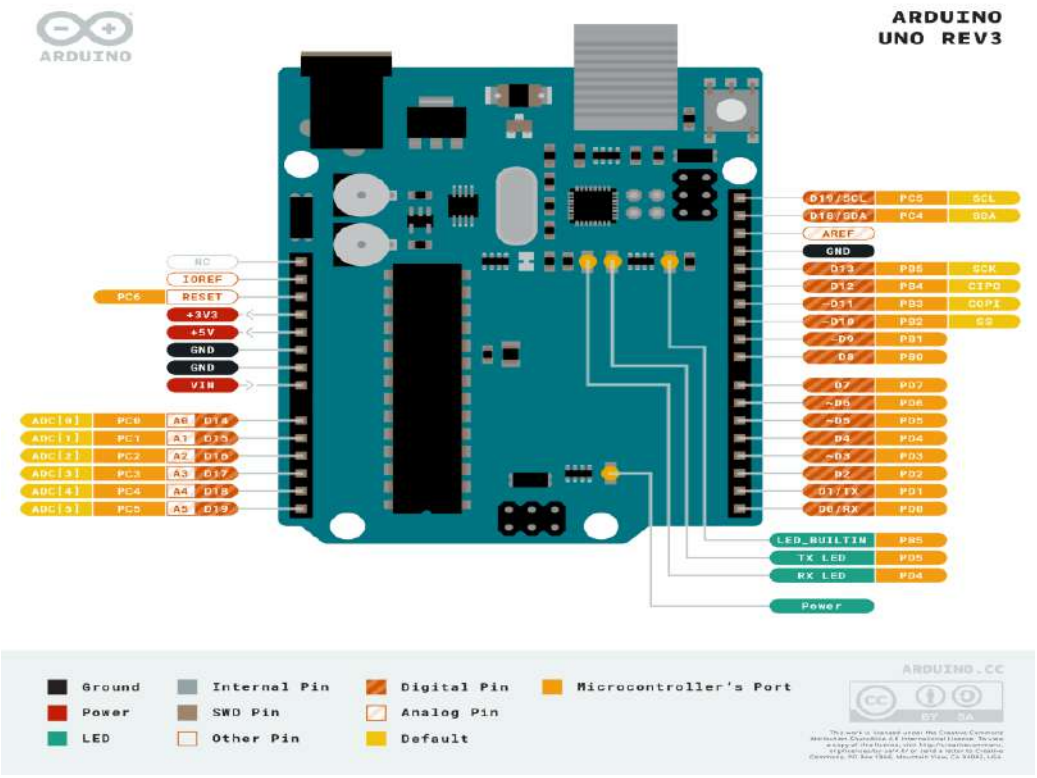


Figure 4 Arduino UNO Pin Configuration [25]

Vin: This is the Arduino board's input voltage pin used to supply input from an external power supply. [12]

5V: The Arduino board pin is utilized as a controlled power supply voltage and is used for supplying both the board and the components on board.

3.3V: This board pin is used to deliver a 3.3V power supply supplied by a board voltage regulator

GND: This board pin is used to ground the board of Arduino.

Reset: The board pin is used to reset the microcontroller It is utilized for the microcontroller resetting.

Analog pins: The A0 to A15 pins are utilized as an analog input with a range of 0 to 5V. Analog pins may be utilized as digital input or output pins on this board.

Serial pins: utilized for communication between a computer or other devices on the Arduino board.

The TXD and RXD are used to transmit & receive the serial data resp. It includes serial 0, Serial 1, serial 2, Serial 3 as follows:

1. Serial 0: It consists of Transmitter pin number 1 and receiver pin number 0
2. Serial 1: It consists of Transmitter pin number 18 and receiver pin number 19
3. serial 2: It consists of Transmitter pin number 16 and receiver pin number 17
4. Serial 3: It consists of Transmitter pin number 14 and receiver pin number 15

External Interrupt pins: this Arduino pin for the production of an External Interrupt is made by the 0,3,21,20,19,18 pin numbers.

I2C: This pin of the board is used for I2C communication.

Pin No. 20 means the Serial Data Line (SDA) and is used for data retention.

Pin number 21 is a Serial Clock Line (SCL) and is used for the synchronization of data between devices.

SPI Pins: This is the Serial Peripheral Interface pin, used to communicate with SPI via the SPI library. Include SPI pins.:

1. MISO: Pin number 50 is used as a Master In Slave Out
2. MOSI: Pin number 51 is used as a Master Out Slave In
3. SCK: Pin number 52 is used as a Serial Clock
4. SS: Pin number 53 is used as a Slave Select

LED Pin: the board features a digital pin-13 built in LED. The LED only illuminates when the digital pin is high.

AREF Pin: This is an Arduino board analog reference pin. It is used to supply an external power supply with a reference voltage.

3.8 Arduino Uno Pinout Configuration

Table 1 Arduino Pinout Configuration

Pin Category	Pin Name	Details
Power	Vin, 3.3V, 5V, GND	Vin: Arduino voltage input when an external power source is used. 5V: regulated microcontroller power supply and other board components. 3.3V: 3.3V on-board voltage regulator supply. The current maximum draw is 50mA. GND: pins on the ground.
Reset	Reset	Resets the microcontroller.
Analog Pins	A0 – A5	Used to provide analog input in the range of 0-5V
Input/output Pins	Digital Pins 0 - 13	Can be used as input or output pins.
Serial	0(Rx), 1(Tx)	Used to receive and transmit TTL serial data.
External Interrupts	2, 3	To trigger an interrupt.
PWM	3, 5, 6, 9, 11	Provides 8-bit PWM output.
SPI	10 (SS), 11 (MOSI), 12 (MISO) and 13 (SCK)	Used for SPI communication.

Inbuilt LED	13	To turn on the inbuilt LED.
TWI	A4 (SDA), A5 (SCA)	Used for TWI communication.
AREF	AREF	To provide reference voltage for input voltage.

3.9 Arduino Uno Technical Specifications

Table 2 Technical Specification

Microcontroller	ATmega328P – 8 bit AVR family microcontroller
Operating Voltage	5V
Recommended Input Voltage	7-12V
Input Voltage Limits	6-20V
Analog Input Pins	6 (A0 – A5)
Digital I/O Pins	14 (Out of which 6 provide PWM output)
DC Current on I/O Pins	40 mA
DC Current on 3.3V Pin	50 mA
Flash Memory	32 KB (0.5 KB is used for Bootloader)

SRAM	2 KB
EEPROM	1 KB
Frequency (Clock Speed)	16 MHz

3.10 ESP8266 NodeMCU:

Open source prototype boards are available for NodeMCU's open source firmware. Because of this, it's called "NodeMCU" (micro-controller unit). The firmware is referred to as "NodeMCU" rather than the development kits that go along with it. Software and prototyping board designs are both openly available. The firmware is programmed in Lua script. ESP8266 Non-OS SDK for eLua is used to build the firmware. The eLua project serves as the foundation for the firmware. SPIFFS is one of the open source projects used in this game. Due to limited resources, users must choose the modules that are most important to their project and create a firmware specifically for it. Additionally, the 32-bit ESP32 was added.

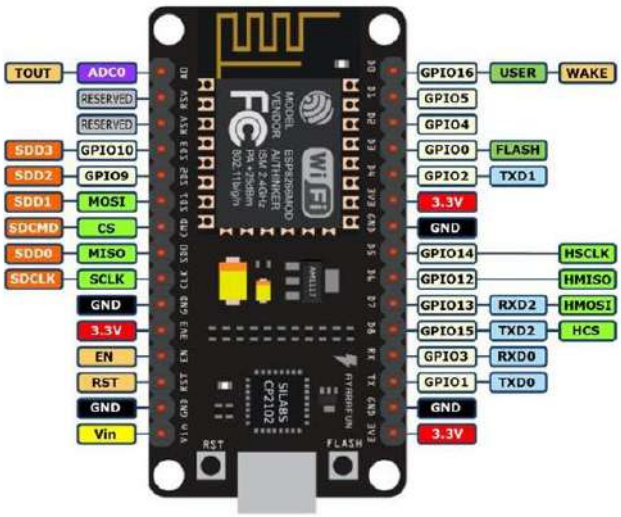


Figure 5 NodeMCU [31]

The most common type of prototyping hardware is a DIP (dual in-line package) circuit board, which combines a USB controller with a surface-mounted MCU and antenna. The DIP format makes breadboard prototyping a breeze. The initial design was based on the ESP8266 ESP-12 module, which is a Wi-Fi SoC combined with a Tensilica Xtensa LX106 core commonly used in Internet of Things applications. [31] It is possible to get NodeMCU in two different varieties: version 0.9 contains the standard-issue ESP-12 and version 1.0 adds support for the "Enhanced" version of the standard-issue ESP-12.

NodeMCU Development Board Pinout Configuration

Table 3 NodeMCU Pinout Configuration [31]

Pin Category	**Name**	**Description**
Power	Micro-USB, 3.3V, GND, Vin	Micro-USB: NodeMCU can be powered through the USB port 3.3V: Regulated 3.3V can be supplied to this pin to power the board GND: Ground pins Vin: External Power Supply
Control Pins	EN, RST	The pin and the button reset the microcontroller

Analog Pin	A0	Used to measure analog voltage in the range of 0-3.3V
GPIO Pins	GPIO1 to GPIO16	NodeMCU has 16 general purpose input-output pins on its board
SPI Pins	SD1, CMD, SD0, CLK	NodeMCU has four pins available for SPI communication.
UART Pins	TXD0, RXD0, TXD2, RXD2	NodeMCU has two UART interfaces, UART0 (RXD0 & TXD0) and UART1 (RXD1 & TXD1). UART1 is used to upload the firmware/program.
I2C Pins		NodeMCU has I2C functionality support but due to the internal functionality of these pins, you have to find which pin is I2C.

NodeMCU ESP8266 Specifications & Features

- Microcontroller: Tensilica 32-bit RISC CPU Xtensa LX106
- Input Voltage: 7-12V
- Operating Voltage: 3.3V
- Analog Input Pins (ADC): 1
- Digital I/O Pins (DIO): 16
- UARTs: 1
- SPIs: 1
- I2Cs: 1
- SRAM: 64 KB
- Flash Memory: 4 MB
- Clock Speed: 80 MHz

- USB-TTL based on CP2102 is included onboard, Enabling Plug n Play
- PCB Antenna
- Compact module that fits neatly within your IoT projects

When using the NodeMCU ESP8266 board, you may use the ESP-12E module, which contains the ESP8266 chip with the Tensilica Xtensa 32-bit LX106 RISC CPU. With a configurable clock frequency range of 80MHz to 160MHz, this microprocessor is compatible with real-time operating systems. NodeMCU can store data and applications in 128 KB of RAM and 4MB of Flash memory. It is perfect for Internet of Things (IoT) applications because of its high computing power, built-in Wi-Fi / Bluetooth, and Deep Sleep Operating characteristics. A Micro USB connector and a VIN pin can be used to power NodeMCU (External Supply Pin). It has interfaces for UART, SPI, and I2C.

32-bit Tensilica Processor

The ESP8266EX microcontroller contains a 32-bit Tensilica L106 RISC processor with extra low power consumption and a maximum clock speed of 160 MHz. The Real-Time Operational System (RTOS) and Wi-Fi stack provide around 80 percent of processing power for the creation and development of user applications [22].

3.15 Compactness

ESP8266EX is integrated with the Tensilica 32-bit CPU, standard peripheral digital interfaces, antenna switches, RF balun, power amplifier, low-noise amplifier, filter systems and modules for power management. All of them are included in our ESP8266EX, a tiny package [16].

3.17 Buzzer:

An electromechanical or piezoelectrical or mechanical type may be an auditory signaling device such as a beeper or buzzer. The basic role is to transform the signal to sound from audio. It is generally driven by DC voltage and utilized in timers, alarms, printers, alarms, computers, etc. It may generate diverse sounds, such as alarm, music, bell & siren, based on different designs.

Figure 6 Buzzer Pi [28]

The buzzer pin configuration is illustrated below. It contains two positive and negative pins. The positive terminal is shown with the sign '+' or with a longer terminal. Powered by 6Volts, the negative terminal is shown by "-" or short terminal and it is connected to the GND terminal.

3.18 Buzzer Pin Configuration

Table 4 Buzzer Pin Configuration [11]

Pin Number	Pin Name	Description
1	Positive	Identified by (+) symbol or longer terminal lead. Can be powered by 6V DC
2	Negative	Identified by short terminal lead. Typically connected to the ground of the circuit

3.19 Features and Specifications of the Buzzer

- Operating Voltage: 4-8V DC
- Rated Voltage: 6V DC
- Rated current: <30mA
- Sound Type: Continuous Beep
- Resonant Frequency: ~2300 Hz
- Friendly to breadboards and perf boards
- Small and neat sealed package

3.20 LM35 Temperature Sensor:

A temperature sensor, the LM35, generates an analog output proportional to the current temperature. For the output voltage, a temperature measurement in Celsius is easily understood. The advantage of the lm35 over the thermistor is that it does not require external calibration. The layer also protects against self-heating. Low cost (around $0.95) and higher precision make it popular among hobbyists, bricolage companies and students. Many low-end items benefit from their low cost, higher precision and the use of LM35 in their products. It is about 15+ years until its first introduction but the sensor remains unchanged and is used in all goods.

Figure 7 LM35 Temperature Sensor [29]

- The LM35 is a temperature sensor that outputs a temperature-proportional analog voltage.
- It offers Centigrade output voltage (Celsius). No external calibration circuit is required.
- The LM35 has a temperature sensitivity of 10 mV/degree Celsius. The output voltage rises as the temperature rises.
    - E.g. 250 mV means 25°C.
- A 3-terminal sensor for surrounding temperatures of -55 °C to 150 °C is employed.
- The LM35 produces a temperature output that is more exact than a thermistor.

3.21 Pin Configuration:

Table 5 LM35 Pin Configuration [11]

Pin Number	Pin Name	Description
1	Vcc	Input voltage is +5V for typical applications
2	Analog Out	There will be increase in 10mV for raise of every 1°C. Can range from -1V(-55°C) to 6V(150°C)
3	Ground	Connected to ground of circuit

3.22 LM35 Regulator Features:

- 35V and -2V are the minimum and maximum input voltages, respectively. 5V is the most common voltage.
- Can measure temperature ranging from -55°C to 150°C
- Output voltage is directly proportional (Linear) to temperature (i.e.) there will be a rise of 10mV (0.01V) for every 1°C rise in temperature.
- ±0.5°C Accuracy
- Drain current is less than 60uA
- Low cost temperature sensor
- Small, making it ideal for remote applications.
- Available in TO-92, TO-220, TO-CAN and SOIC package

3.23 MAX30100 Pulse Oximeter

The sensor functions as both a pulse oximeter and a heart rate monitor. Two LEDs, a photodetector, better optics, and low-noise analog signal processing are used to detect pulse and heart rate signals. It operates on 1.8V and 3.3V power supplies and may be

shut off by software with very minimal standby current, allowing the power supply to be connected at all times.

Figure 8 GY-MAX30100 [30]

3.24 Features

1. It's quite energy efficient (operates from 1.8V and 3.3V)

2. Ultra-Low Shutdown Current (0.7µA, type)

3. Fast Data Output Capability

4. Interface Type: I2C

3.25 MAX30100 Pulse Oximeter and Heart Rate Sensor Operation

The device contains two LEDs, one of which generates red light and the other infrared light. To estimate pulse rate, just infrared light is required. Both red and infrared light are used to measure the amount of oxygen in the blood.

When the heart beats, the amount of oxygenated blood in the blood increases because there is more blood. When the heart relaxes, the volume of oxygenated blood decreases. The period between the increase and fall of oxygenated blood is used to compute the pulse rate.

Blood that is deoxygenated absorbs red light but transmits more infrared light, and blood that is oxygenated absorbs red light but transmits more infrared light. The MAX30100's primary function is to read the absorption levels of both light sources and store them in a buffer that can be read using the I2C communication protocol. [30]

3.26 Max30100 Pulse Oximeter Sensor Pinout:

The VIN, GND, SCL, SDA, and INT male headers of the GY-Max30100 Pulse Oximeter are all well labeled. This is an i2c-compatible sensor that communicates with the Arduino board over the i2c bus.

3.27 Battery

The 9-volt battery, often known as the battery 9-volt, was a common battery size for early transistor radios. It's shaped like a rectangular prism and has a polarized snap connector on top. Walkie talkies, smoke alarms, and clocks all employ this type. Primary carbon zinc and alkaline chemistry, primary lithium iron disulfide, nickel cadmium nickel, nickel metal hydride, and lithium ion in rechargeable form all use the 9-volt battery architecture. This format, previously widespread, was not produced in many years because of its mercury level.

Figure 9 Battery [33]

3.28 Technical Specification:

The most common type of nine-volt battery is simply known as 9-volt, whereas nine-volt batteries of other sizes are less common. PP3 (size and voltage, any technological features), 6LR61 (alkaline battery IEC code), and 006P are the most common size codes in Japan. The PP3 battery is 48.5 mm x 26.5 mm x 17.5 mm, or 1.91 in x 1.04 = 0.69 in. At the same end and their centers, both terminals are 12 inches (12.7 mm) apart. A 9-volt alkaline or carbon zinc battery has six cylindrical or flat cells connected in series. Internally, some versions use welded tabs to fit cells, while others use foil strips against the cell ends. 6 to 8 1.2-volt cells are found in rechargeable 9V nominal nickel-cadmium (NiCd) and nickel-metal hydride (NiMH) batteries. In the lithium ion variant, two cells (3.7–4.2 V nominal each) are often utilized. Lithium polymers and NiMH variants with reduced self-discharge are also available. This size of mercury

battery had previously been manufactured. Their capacity was higher than regular carbon zinc kinds, and their nominal voltage was 8.4 volts, with a very stable voltage. Mercury is no longer a pollutant for the environment when used in photography, measurement, or long-term applications.

3.29 HTML:

For Web sites and online applications, the most prevalent programming language is Hypertext Markup Language (HTML). A hypertext is a text used to reference other parts of text, whereas a markup language is a number of markers indicating the style and structure of a page on the web servers [21].

HTML is not a programming language, as dynamic functionality cannot be created. With HTML, instead, online users can utilize elements, tags and attributes to design and arrange sections, paragraphs and connections.

HTML is commonly used in the following ways:

- Development of the Web. Developers are using HTML code to develop how a browser displays aspects of the web page, like text, hyperlinks and media files.

- Internet browsing. Users can quickly navigate and insert links between associated pages and websites because HTML is widely utilized for the inclusion of hyperlinks.

- Documentation on the Web. HTML enables documents like Microsoft Word to be organized and formatted.

It should also be noted that HTML is now an established web standard. The World Wide Web Consortium (W3C), together with frequent revisions, maintains and develops HTML specifications.

3.30 Google Sheet:

Google Sheets is a free web-based table application that the Google Drive service provides. It is also available on Chrome OS as a desktop app, and on Android, Windows, iOS, and BlackBerry as a mobile app. In Google Drive, other products such as Google Docs, Slides and Formats are also available. Users may modify, organize and analyze various sorts of information with Google Sheets. It permits collaboration,

and several users can edit and format files in real time, and revision history can trace any changes to the table.

3.31 WordPress

WordPress (WP, WordPress.org) is a free and open-source content management system (CMS) that is written in PHP [4] and runs on a MySQL or MariaDB database. Among the features are a plugin architecture and a template system known as Themes in WordPress. Although WordPress began as a platform for posting blogs, it has subsequently expanded to incorporate more conventional mailing lists and forums, as well as media galleries, membership sites, learning management systems (LMS), and online commerce. WordPress was used by 41.4 percent of the top ten million websites as of May 2021. One of the most popular content management systems is WordPress. [6] WordPress has also been used in a wide range of other applications, such as pervasive display systems (PDS). [7] On May 27, 2003, its inventors, American developer Matt Mullenweg [1] and English developer Mike Little [8][9], launched WordPress as a fork of b2/cafelog. The GNU General Public License, version 2 is used to distribute the software (or later). [10] When using WordPress, you'll need to have it installed on a web server, either through a service such as WordPress.com or on your own computer using the WordPress.org software package. [11] A local PC can be utilized for single-user testing and learning.

3.32 WordPress Plugins:

WordPress's plugin architecture allows users to extend the features and functionality of a website or blog. With 58,463 plugins accessible as of May 2021[17], WordPress.org is able to provide users with a wide range of customization options for their sites. There are almost 1,500 free plugins on WordPress.org, however not all of them are included in the WordPress.org repository. Content management systems, search engine optimization (SEO), client portals for exposing private information, and content display elements like widgets and navigation bars are all examples of customizations. Due to the fact that not all plugins are updated on a regular basis, they may not operate correctly or at all. There are a variety of plugins available, most of which can be installed directly through WordPress, either manually via FTP or using

the plugin dashboard. The majority of plugins offered by third parties are in the form of paid subscriptions.

3.33 Elementor Plugin:

Elementor is a WordPress page builder that lets you drag and drop elements on the page as you like. You may use a visual editor to build stunning pages with the aid of this plugin. It's made to let you easily create dynamic websites. Customize your website to match your company's image with animations, various fonts, and eye-catching backgrounds.

3.34 Ultimate Member Plugin:

The most popular WordPress user profile and membership plugin is Ultimate Member. With the aid of this plugin, users can effortlessly sign up and become members of your website. The plugin allows you to create advanced online communities and membership sites by allowing you to add gorgeous user profiles to your site. Ultimate Member is a lightweight and highly extendible membership management system that allows you to develop nearly any type of site where users may join and become members with ease [15]. The following are some of the features of the plugin:

- Front-end user profiles
- Front-end user registration
- Front-end user login
- Custom form fields
- Conditional logic for form fields
- Drag and drop form builder
- User account page
- Custom user roles
- Member directories
- User emails
- Content restriction
- Conditional nav menus
- Show author posts & comments on user profiles
- Developer friendly with dozens of actions and filters

Chapter 4

System Discussion

4.1 System Block Diagram:

Here is a Block Diagram of our IOT based Covid-19 Patient detection and monitoring System

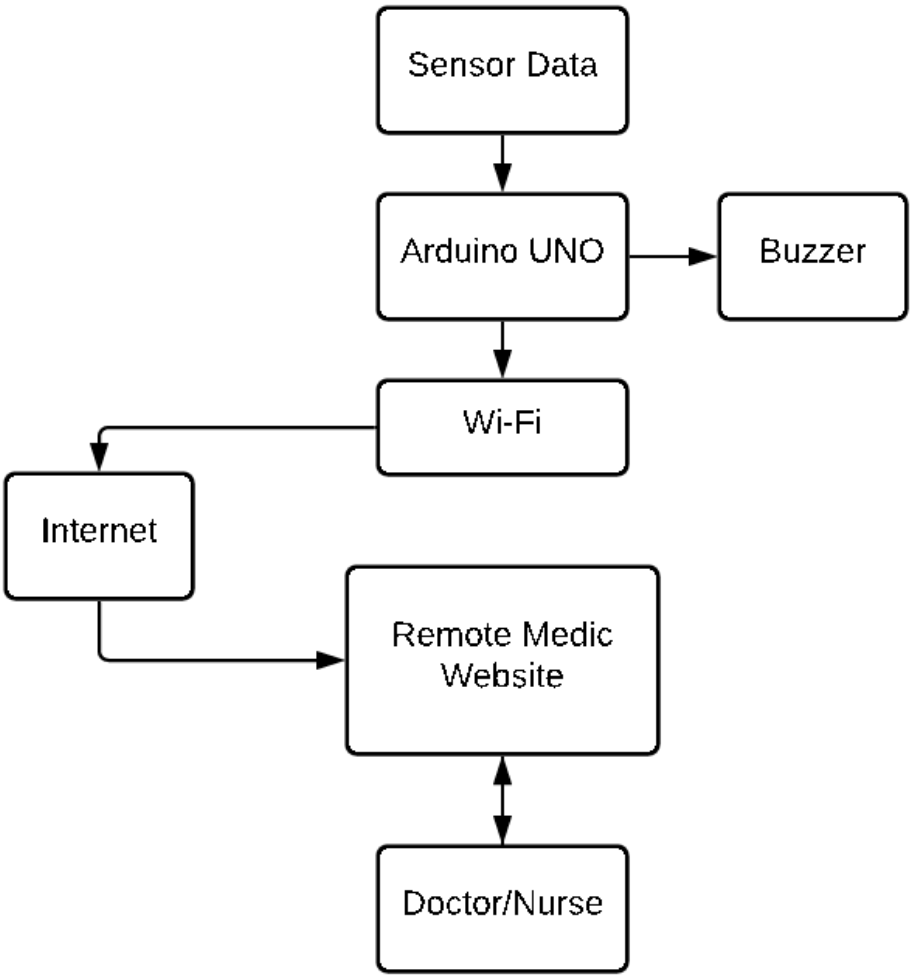


Figure 10 Block diagram of our system

4.2 System Framework:

The following diagram depicts the entire system architecture.

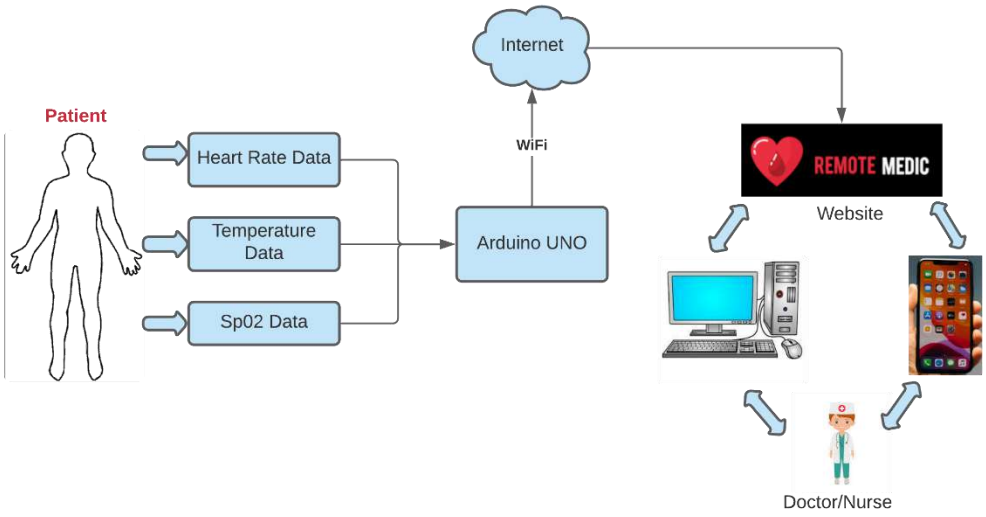


Figure 11 System Framework of our system

4.3 Circuit Diagram:

The Circuit Diagram of COVID19 Patient Monitoring System:

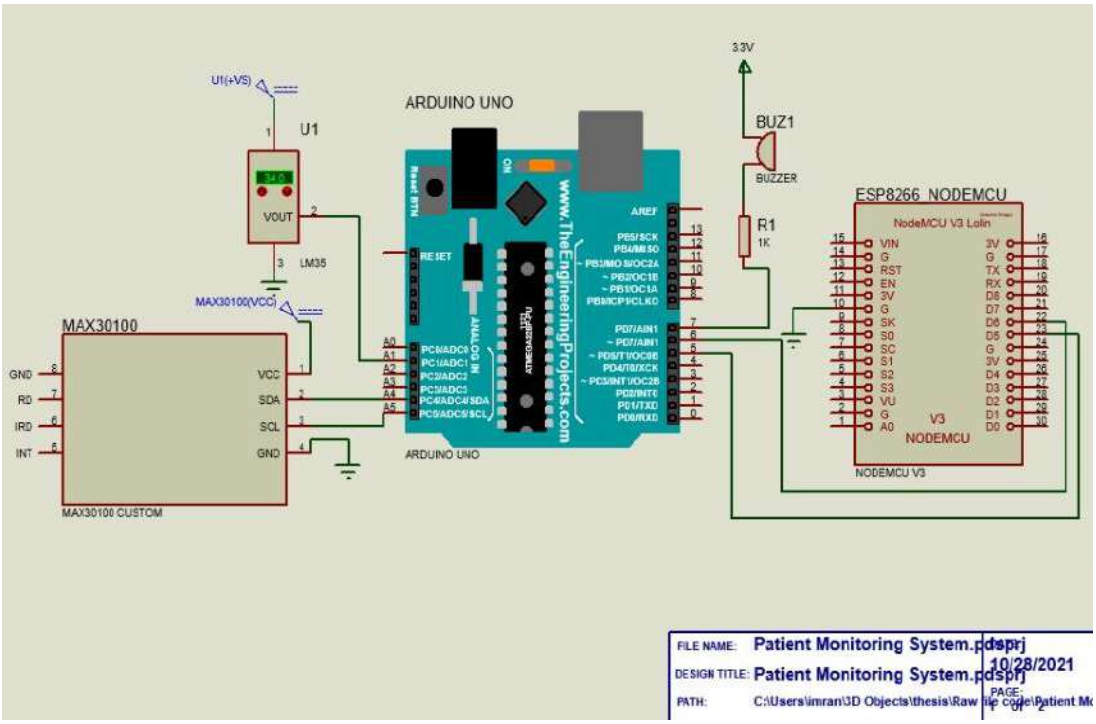


Figure 12 Proteus Simulink

4.4 Methodology:

The suggested system is divided into two parts: the patient side, which is comprised of wearable sensors, and the web server side, which is composed of web pages.

Patient side:

The Arduino Uno serves as the primary processing component in this wearable gadget. Patient heart rate data is collected using a piezoelectric sensor. When you put mechanical stress on a crystal, the piezoelectric effect, also known as the piezoelectricity, appears as an electrical potential (or voltage) (by squeezing it). The piezoelectric sensor was utilized in medicine to create an electrical pulse from sound energy in the body. The wrist-worn piezoelectric sensor measures heart rate by detecting the wearer's pulse. Afterwards, the information is delivered to the microcontroller, which records and processes it before sending it to the internet through the NodeMCU Wi-Fi module.

The body temperature is measured using an LM35 temperature sensor. It communicates with the microcontroller by sending data there. The information is stored and processed by the microcontroller before being sent to the website database.

Max30100 sensor measures patient's heart rate and oxygen saturation. There are two LEDs on the device, one red and one infrared. So far, everything has gone well. Blood oxygen levels are measured using red and infrared light. More blood is pumped by the heart, which equals more oxygenated blood. Less oxygenated blood is produced as the heart relaxes. The time between oxygenated blood increases and declines controls the pulse rate. Infrared light is absorbed by blood and passed through it, whereas red light is absorbed by deoxygenated blood and passed through it. The MAX30100 reads both light sources' absorption levels and saves them in a buffer accessible via the I2C interface.

A buzzer and programming are used to create a buzzer system. In order to detect when one of the sensor data is outside of the expected range, the Arduino has been coded to sound the buzzer.

The microcontroller is connected to the internet through an ESP8266 NodeMCU module. Microchip with full TCP/IP stack and microprocessor, the NodeMCU is an inexpensive Wi-Fi chip. Simple TCP/IP connections can be made using Hayes-style commands on a Wi-Fi network with this module. ESP8266 NodeMCU module transmits the microcontroller's stored processed data to our website.

Web Server:

We created a website for displaying the data on the internet using the WordPress platform. PHP is the programming language used by WordPress, and MySQL is the database. The MySQL database is where we keep all of our data. Doctors must first create an account on our website in order to access the website. The admin panel will verify all submitted accounts. Our website will only allow doctors who are already registered to create an account. This limitation is in place to safeguard the patient's personal information. In addition, patients must create an account on our website in order to upload their other reports.

On the other hand, the doctor will have access to the patient list and medical record after opening an account. after which it is possible to analyze the information and provide helpful suggestions to the patients.

4.5 List of Components:

1. Microcontroller (Arduino UNO)
2. NodeMCU
3. Temperature Sensor (LM35)
4. GY-MAX30100
5. Buzzer
6. Resistor(10k,1k,4.5k)
7. LED
8. Connecting Wires
9. Battery
10. Board
11. capacitor

4.6 System Flow Chart:

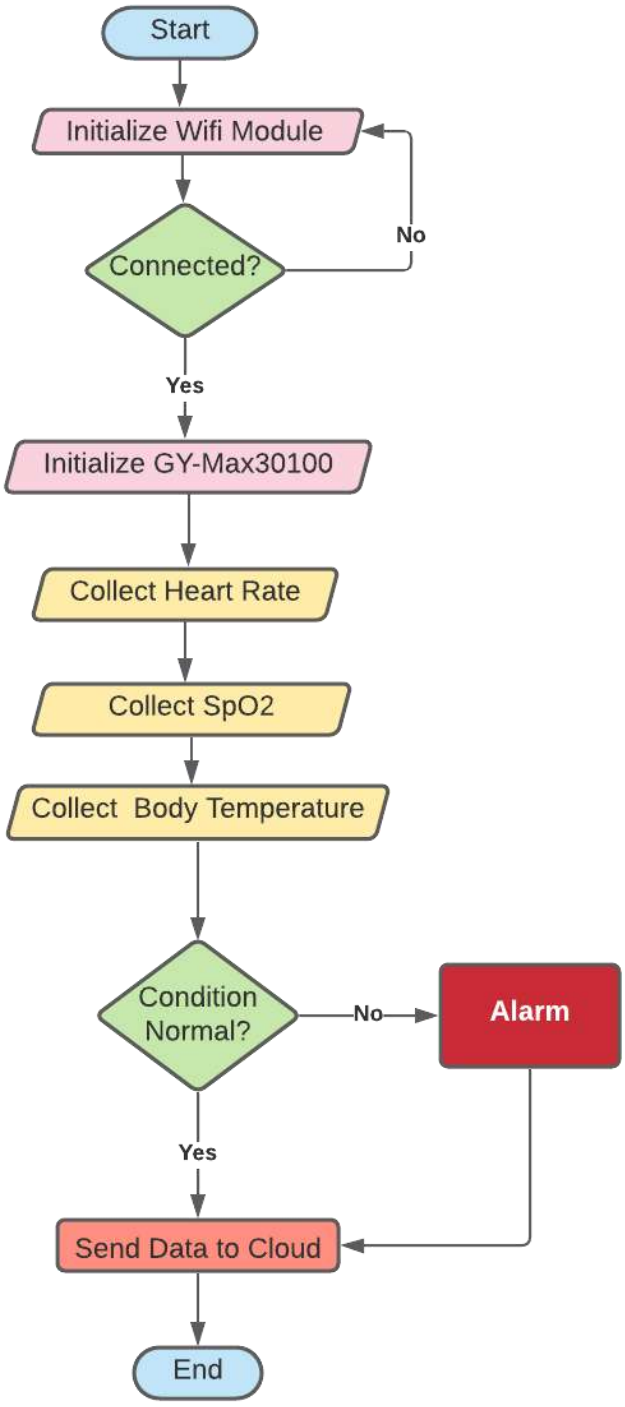


Figure 13 Flow Chart of IoT Base Covid19 Patient Monitoring System.

Chapter 5

Results and Discussion

5.1 Introduction

The whole study result has been discussed in this chapter. The hardware, results and the designed web server can be seen in pictures. The entire working procedure is depicted in pictures in a chronological order.

5.2 Result

Our designed hardware is shown below.

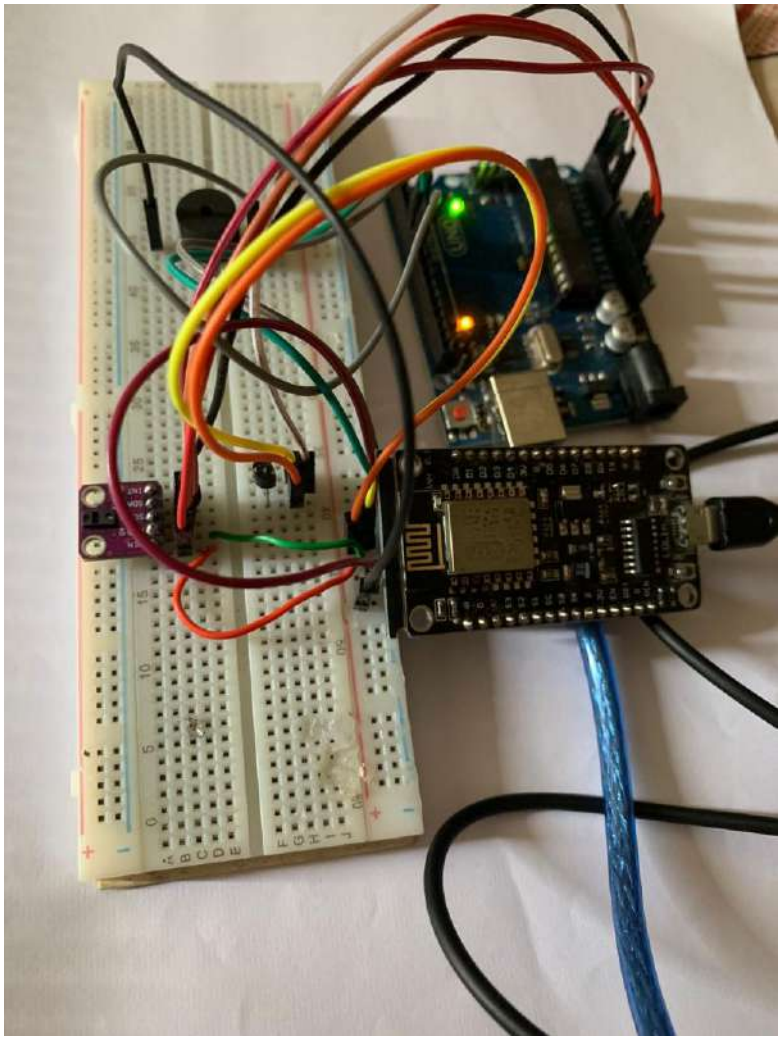


Figure 14 Hardware

Output result in serial monitor of Arduino IDE is given below

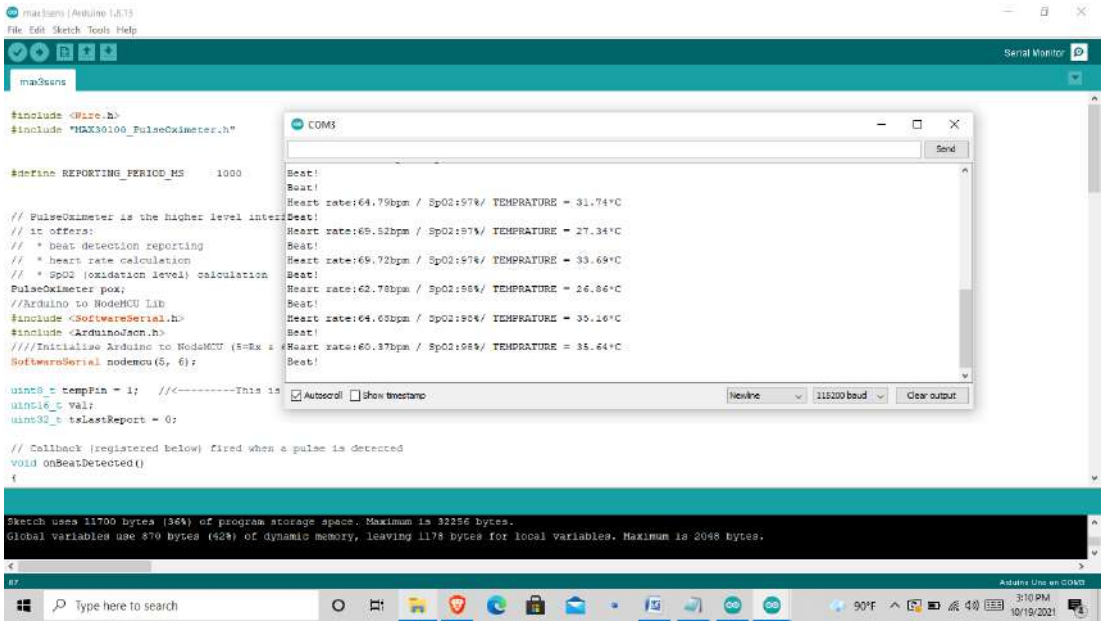


Figure 15 Output in Serial Monitor

As you can see in the screenshot, the output displays the user's body temperature, heart rate, and level of blood oxygen.

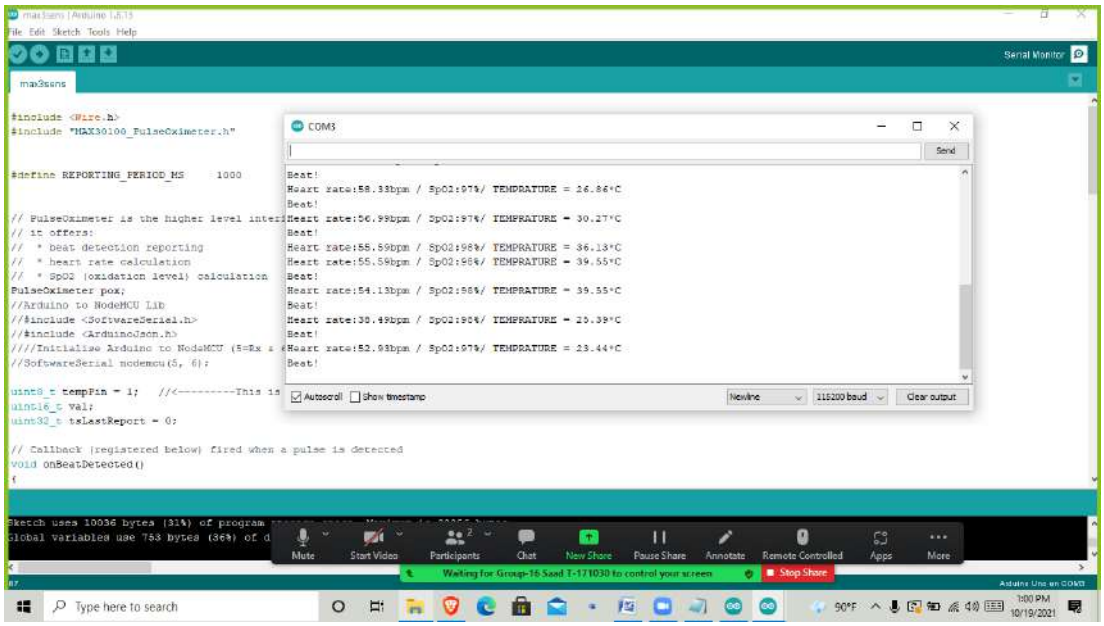
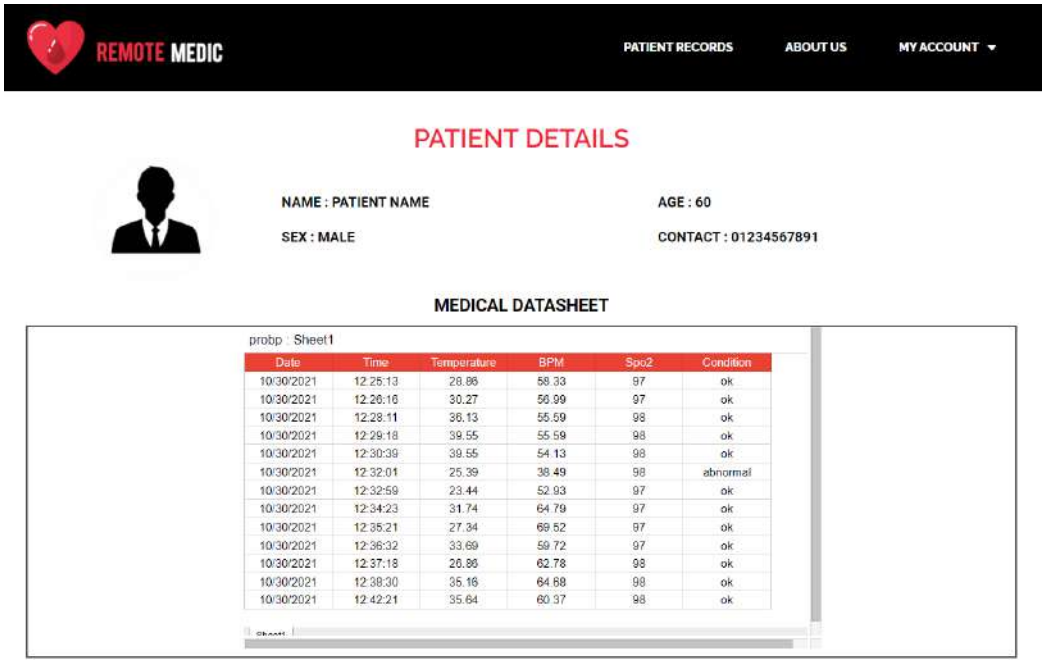


Figure 16 Output in Serial Monitor

As you can see in the screenshot, the output displays the user's body temperature, heart rate, and level of blood oxygen. The fact that the patient's blood oxygen level is low is also telling.



Date	Time	Temperature	BPM	Spo2	Condition
10/30/2021	12:25:13	28.86	58.33	97	ok
10/30/2021	12:26:16	30.27	56.99	97	ok
10/30/2021	12:28:11	36.13	55.59	98	ok
10/30/2021	12:29:18	39.55	55.59	98	ok
10/30/2021	12:30:39	39.55	54.13	98	ok
10/30/2021	12:32:01	25.39	38.49	98	abnormal
10/30/2021	12:32:59	23.44	52.93	97	ok
10/30/2021	12:34:23	31.74	64.79	97	ok
10/30/2021	12:35:21	27.34	69.52	97	ok
10/30/2021	12:36:32	33.69	59.72	97	ok
10/30/2021	12:37:18	26.86	62.78	98	ok
10/30/2021	12:38:30	35.16	64.68	98	ok
10/30/2021	12:42:21	35.64	60.37	98	ok

Figure 17 Output when Temperature is Abnormal

This is the result we'll see on our newly built website. Arduino will send the patient's heart rate, body temperature, and blood oxygen level to our website via NodeMCU.

5.3 Design of The Website

The picture of website login system is shown below

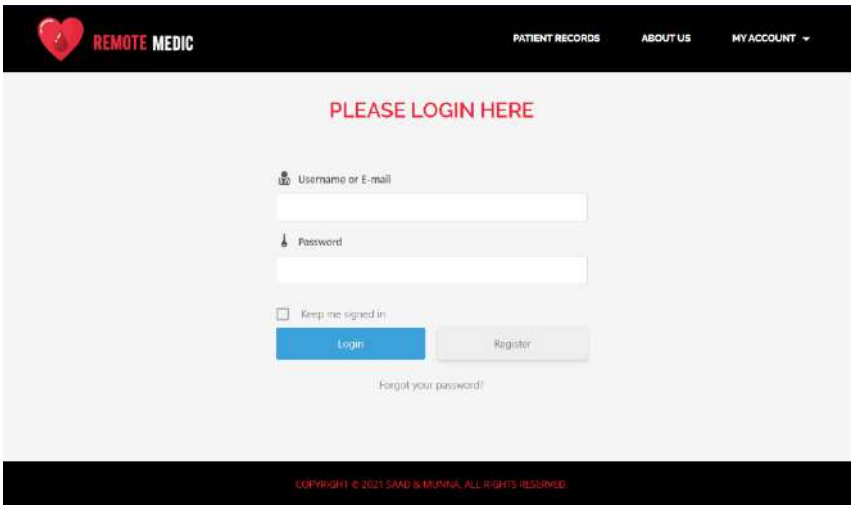


Figure 18 Website Login System

This is the registration page. Only registered doctors who have completed this form will be accepted and be allowed to use the website.

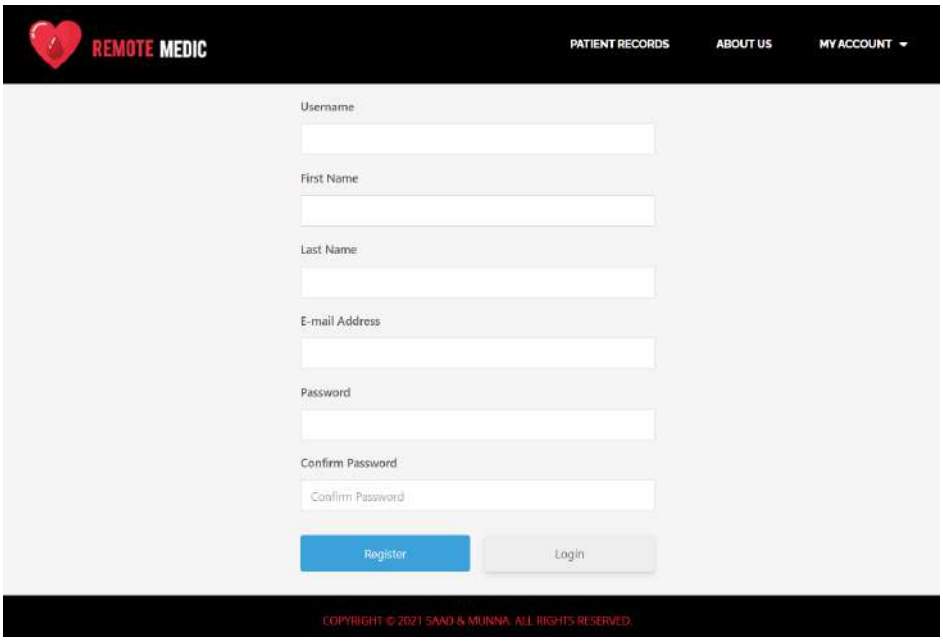


Figure 19 Registration Page

This is a home page which doctors will be able to see after login. From here he will go to patient list to see the patient list.

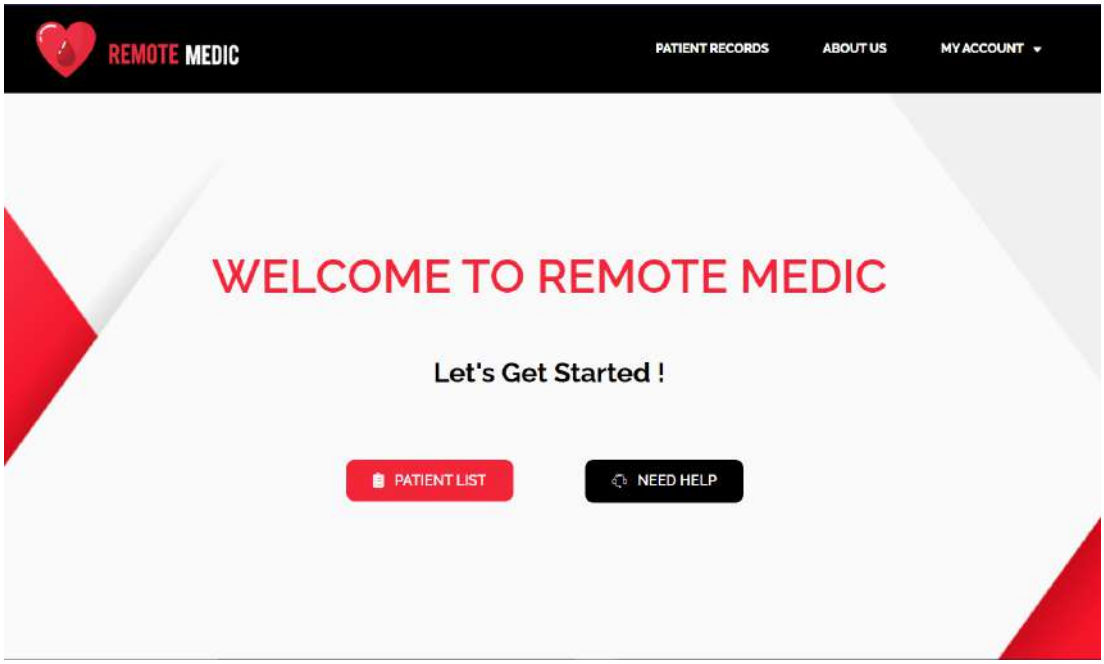


Figure 20 Home Page

In this page doctor can see the patient list. By selecting individual patient, he can see the medical record of that patient.

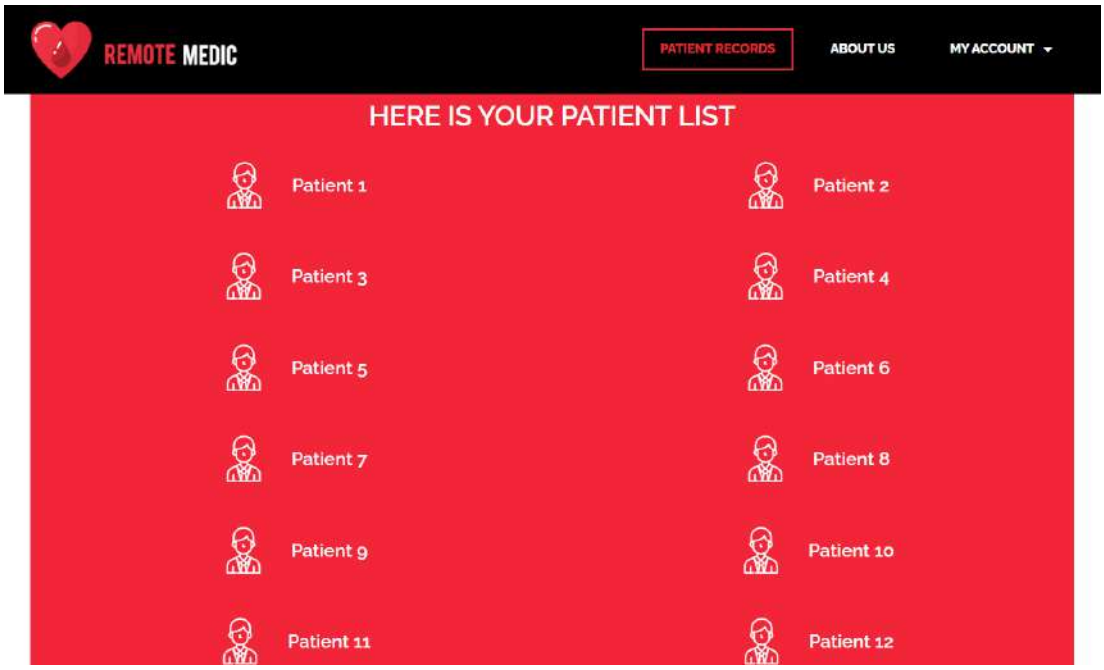
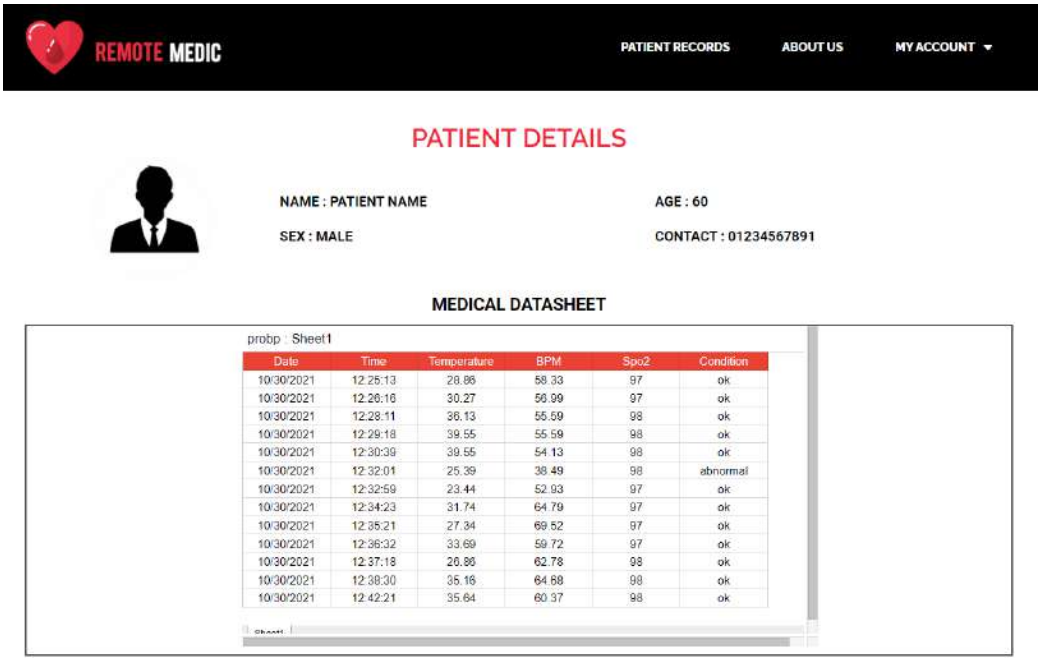


Figure 21 Patient List Page

Here doctor can see the medical record of the patient and necessary information to contact the patient.



Date	Time	Temperature	BPM	Spo2	Condition
10/30/2021	12:25:13	28.86	58.33	97	ok
10/30/2021	12:26:16	30.27	56.99	97	ok
10/30/2021	12:28:11	36.13	55.59	98	ok
10/30/2021	12:29:18	39.55	55.59	98	ok
10/30/2021	12:30:39	39.55	54.13	98	ok
10/30/2021	12:32:01	25.39	38.49	98	abnormal
10/30/2021	12:32:59	23.44	52.93	97	ok
10/30/2021	12:34:23	31.74	64.79	97	ok
10/30/2021	12:35:21	27.34	69.52	97	ok
10/30/2021	12:36:32	33.69	59.72	97	ok
10/30/2021	12:37:18	26.86	62.78	98	ok
10/30/2021	12:38:30	35.16	64.88	98	ok
10/30/2021	12:42:21	35.64	60.37	98	ok

Figure 22 Patient Records Page

This is the system that we use to stablish communication between patient and doctors. Doctors and patients can easily communicate with each other using this platform. Patients can describe their current circumstances, and doctors can provide advice quickly. Patients can also use this to upload medical reports, which will aid doctors in better understanding the patient's condition.

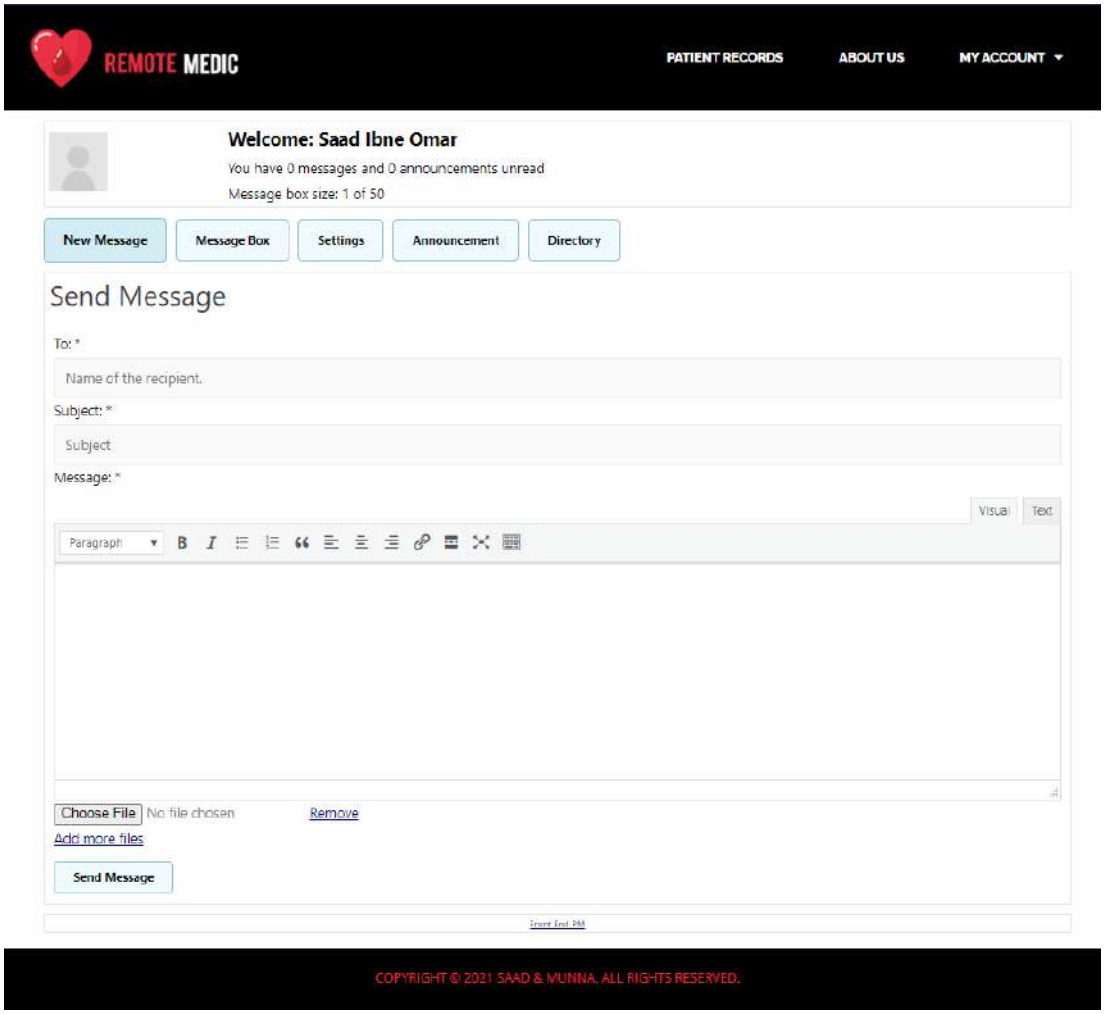


Figure 23 Messaging system

5.4 Discussion:

In our study, we attempted to create a device that could be taken anywhere. The accessibility of patient medical data was our primary concern in this case. By utilizing our system, individuals can quickly and easily connect to health-care services without having to visit a hospital. In addition, doctors can access their patient's medical record

from any location in the world by utilizing our technology. Communication between the doctor and the patient will be secure, effective, and quick.

5.5 A comparison of our proposed system with current systems is in order.

Table 6 comparison of our proposed system with current systems

SL no	Previous Systems	Our Systems
1	One patient monitoring system is supported.	Multiple patient monitoring system is supported.
2	Less secured webserver	Webserver is fully secured with registration system
3	It is used in general purpose	It can be used for detecting possible Covid-19 case.

5.6 Advantages and uses:

- The device is wearable.
- The device is portable.
- Low costing device.
- Easily accessible.
- Patient records are secured and privacy is ensured.

Chapter 6

Conclusion and Future work

6.1 Conclusion

Our system is easy to use, secure, and simple to develop, all at a low cost. With this method, the shortage of doctors during a pandemic will be completely eliminated. The patient's health risks are reduced thanks to the emergency alert system. With this information, clinicians have the ability to make decisions by looking at prior medical experience. This method will make things run more smoothly and waste less time when it comes to a life-or-death medical issue. With an internet connection, the system can be accessed from anywhere in the world.

6.2 Future Work

The minds of innovative people can improve our system. Blood pressure measuring system can be added to our system in future which will make the device more useful and unique. By carefully selecting sensors and other components, the system can be made lightweight. The subscription system on our website can also be improved. For our system, an app might be created that displays the patient's medical data within the app. We can also include a messaging feature in the app for doctor-patient communication. The app can be upgraded to include an alarm system. If a patient's health changes, the app will notify doctors, and the alarm system can be used to remind individuals to take their meds on time. This is how you can make a system more user-friendly.

References

[1] A. Zanella, N. Bui,, A. Castellani, L. Vangelista and M. Zorzi, "Internet of Things for Smart Cities," *IEEE Internet of Things Journal,* vol. 1, pp. 22-32, 2014.

[2] "Aruba," [Online]. Available: https://www.arubanetworks.com/solutions/internet-of-things/.

[3] R. K. N.V. , A. B. M, E. S. J. P. K. A and A. P. , "Detection and monitoring of the asymptotic COVID-19 patients using IoT devices and sensors," *International Journal of Pervasive Computing and Communications,* 2020.

[4] Mwaffaq Otoom a, Nesreen Otoum b, Alzubaidi MA, Etoom Y and Banihani R, "An IoT-based framework for early identification and monitoring of COVID-19 cases," *Biomedical signal processing and control,* p. 62, 2020.

[5] A. Zia Uddin , M. G. Mortuza, Mohammed Jashim Uddin and Md Humayun Kabir, "Internet of Things Based Patient Health Monitoring System Using Wearable Biomedical Device," *International Conference on Innovation in Engineering and Technology (ICIET),* 2018.

[6] Moeen Hassanalieragh, Alex Page, Tolga Soyata, Gaurav Sharma, Mehmet Aktas, Gonzalo Mateos and Burak Kantarci, "Health monitoring and management using Internet-of-Things (IoT) sensing with cloud-based processing: Opportunities and challenges," *IEEE International Conference on Services Computing,* pp. 285-292, 2015.

[7] Vikas Vippalapalli and Snigdha Ananthula, "Internet of things (IoT) based smart health care," *2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES),* pp. 1229-1233, 2016.

[8] T M Cadarina and R Priambodo, "Monitoring Heartrate and SpO2 using Thingsboard IoT platform for mother and child preventive healthcare," *IOP conference series: materials science and engineering,* 2018.

[9] SM Riazul Islam, Daehan Kwak, MD Humaun Kabir, Mahmud Hossain and Kyung-Sup Kwak, "The internet of things for health care: a comprehensive survey," *IEEE access 3 (2015):,* 2015.

[10] Ashwin K, Withchurch and Joice TM, "Measure SpO2, Heart Rate and BP Trends (BPT) Using Arduino," 2020.

[11]"LM35 Temperature Sensor", *Components101*, 2021. [Online]. Available: https://components101.com/sensors/lm35-temperature-sensor#:~:text=LM35%20is%20a%20precession%20Integrated,C%20to%20150%C2%B0C.&text=Ther?cv=1. [Accessed: 02- Oct- 2021].

[12]"Arduino Pin Configuration: A Detailed Guide (2021) | Robu.in", *Robu.in | Indian Online Store | RC Hobby | Robotics*, 2021. [Online]. Available: https://robu.in/arduino-pin-configuration/?cv=1. [Accessed: 02- Oct- 2021].

[13]"Piezoelectric sensor - Wikipedia", *En.wikipedia.org*, 2021. [Online]. Available: https://en.wikipedia.org/wiki/Piezoelectric_sensor#:~:text=A%20piezoelectric%20sens?cv=1. [Accessed: 02- Oct- 2021].

[14]"Arduino Uno Rev3", *Arduino Online Shop*, 2021. [Online]. Available: https://store-usa.arduino.cc/products/arduino-uno-rev3/?cv=1. [Accessed: 02- Oct- 2021].

[15]"Ultimate Member – User Profile & Membership Plugin 1.3.88 - Wordpress plugin", *Find-wordpress-plugins.com*, 2021. [Online]. Available: http://www.find-wordpress-plugins.com/ultimate-member-wordpress-plugin.html?cv=1. [Accessed: 02- Oct- 2021].

[16] M. Thakur, "ESP8266 Pin Diagram | Circuits4you.com", *Circuits4you.com*, 2021. [Online]. Available: https://circuits4you.com/2016/12/14/esp8266-pin-diagram/amp/?cv=1. [Accessed: 02- Oct- 2021].

[17]2021. [Online]. Available: https://www.researchgate.net/publication/280924370_Health_Monitoring_and_Management_Using_Internet-of-Things_IoT_Sensing_with_Cloud-Based_Processing_Opportunities_and_Challenges?cv=1. [Accessed: 02- Oct- 2021].

[18]"Arduino / ESP8266 WiFi Modülü", *Dersmax.net*, 2021. [Online]. Available: https://dersmax.net/icerik/122/5739/esp8266_wifi_modulu.html?cv=1. [Accessed: 02- Oct- 2021].

[19]"Pulse sensor showing BPM 150+ · Issue #60 · WorldFamousElectronics/PulseSensor_Amped_Arduino", *GitHub*, 2021. [Online]. Available: https://github.com/WorldFamousElectronics/PulseSensor_Amped_Arduino/issues/60?cv=1. [Accessed: 02- Oct- 2021].

[20]"Internet of Things Based Patient Health Monitoring System Using Wearable Biomedical Device", *Ieeexplore.ieee.org*, 2021. [Online]. Available: https://ieeexplore.ieee.org/document/8660846?cv=1. [Accessed: 02- Oct- 2021].

[21]"What Is HTML? Hypertext Markup Language Basics for Beginners", *Hostinger Tutorials*, 2021. [Online]. Available: https://www.hostinger.com/tutorials/what-is-html?cv=1. [Accessed: 02- Oct- 2021].

[22]"ESP8266 ESP-12E CH340G NodeMCU V3 Development Board - ESP8266 Shop", *ESP8266 Shop*, 2021. [Online]. Available: https://esp8266-shop.com/product/nodemcu-esp8266-esp-12e/?cv=1. [Accessed: 02- Oct- 2021].

[23] *Irjet.net*, 2021. [Online]. Available: https://www.irjet.net/archives/V6/i4/IRJET-V6I41067.pdf?cv=1. [Accessed: 02- Oct- 2021].

[24] *Docs.arduino.cc*, 2021. [Online]. Available: https://docs.arduino.cc/hardware/uno-rev3. [Accessed: 02- Oct- 2021].

[25] *Docs.arduino.cc*, 2021. [Online]. Available: https://docs.arduino.cc/hardware/uno-rev3. [Accessed: 02- Oct- 2021].

[26]"Esp8266 Wifi Module", *indiamart.com*, 2021. [Online]. Available: https://www.indiamart.com/proddetail/esp8266-wifi-module-18795392230.html. [Accessed: 02- Oct- 2021].

[27]"ESP8266 WiFi Module | Sensors & Modules", *Electronicwings.com*, 2021. [Online]. Available: https://www.electronicwings.com/sensors-modules/esp8266-wifi-module. [Accessed: 02- Oct- 2021].

[28]"Buzzer : Working, Types, Circuit, Advantages & Disadvantages", *ElProCus - Electronic Projects for Engineering Students*, 2021. [Online]. Available:

https://www.elprocus.com/buzzer-working-applications/. [Accessed: 02- Oct- 2021].

[29]"LM35DZ TEMPERATURE SENSOR - Ktechnics Systems", *Ktechnics Systems*, 2021. [Online]. Available: https://www.ktechnics.com/product/lm35dz-precision-temperature-sensor/. [Accessed: 02- Oct- 2021].

[30]"max30100 notes", *tlfong01.blog*, 2021. [Online]. Available: https://tlfong01.blog/2020/05/05/max30100-notes-5/. [Accessed: 27- Oct- 2021]

[31]"NodeMCU ESP8266", *Components101*, 2021. [Online]. Available: https://components101.com/development-boards/nodemcu-esp8266-pinout-features-and-datasheet. [Accessed: 27- Oct- 2021]

[32]"Light Dependent Resistors, Power Type : AC by Cosmic Devices from Delhi Delhi | ID - 1906180", *Exportersindia.com*, 2021. [Online]. Available: https://www.exportersindia.com/product-detail/light-dependent-resistors-1906180.htm. [Accessed: 02- Oct- 2021].

[33]"Type: Battery, Alarm, DAITEM", *Batteries4pro.com*, 2021. [Online]. Available: https://www.batteries4pro.com/en/s/14/battery-alarm-daitem. [Accessed: 02- Oct- 2021].

APPENDIX

ARDUINO SCRIPT

```
#include <Wire.h>

#include "MAX30100_PulseOximeter.h"

#define REPORTING_PERIOD_MS     1000

// PulseOximeter is the higher level interface to the sensor

// it offers:

//  * beat detection reporting

//  * heart rate calculation

//  * SpO2 (oxidation level) calculation

PulseOximeter pox;

//Arduino to NodeMCU Lib

#include <SoftwareSerial.h>

#include <ArduinoJson.h>

////Initialise Arduino to NodeMCU (5=Rx & 6=Tx)

SoftwareSerial nodemcu(5, 6);


uint8_t tempPin = 1;    //<---------This is the PIN the LM53 is connected! CHANGE accordingly

uint16_t val;

uint32_t tsLastReport = 0;

int buzzer_pin=7;
```

```
// Callback (registered below) fired when a pulse is detected
void onBeatDetected()
{
  Serial.println("Beat!");
}

void setup()
{
  Serial.begin(115200);
  nodemcu.begin(9600);
  Serial.print("Initializing pulse oximeter..");

  // Initialize the PulseOximeter instance
  // Failures are generally due to an improper I2C wiring, missing power supply
  // or wrong target chip
  if (!pox.begin()) {
    Serial.println("FAILED");
    for (;;);
  } else {
    Serial.println("SUCCESS");
  }
```

```
  // The default current for the IR LED is 50mA and it could be changed
  //   by uncommenting the following line. Check MAX30100_Registers.h for all the
  //   available options.
//   pox.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);

  // Register a callback for the beat detection
  pox.setOnBeatDetectedCallback(onBeatDetected);
  pinMode (buzzer_pin,OUTPUT );

}

void loop()
{
  val = analogRead(tempPin);
  float mv = ( val / 1024.0) * 5000;
  float cel = mv / 10;
  String cond;

  // Make sure to call update as fast as possible
  pox.update();
```

```
  // Asynchronously dump heart rate and oxidation levels to the serial
  // For both, a value of 0 means "invalid"
  if (millis() - tsLastReport > REPORTING_PERIOD_MS) {
    Serial.print("Heart rate:");
    Serial.print( pox.getHeartRate());
    Serial.print("bpm / SpO2:");
    Serial.print( pox.getSpO2());
    Serial.print("%");
    Serial.print(" TEMPRATURE = ");
    Serial.print(cel);
    Serial.println("*C");
    tsLastReport = millis();
//
//    /* uncomment this to get temperature in farenhite
//      float farh = (cel*9)/5 + 32;
//      Serial.print("TEMPRATURE = ");
//      Serial.print(farh);
//      Serial.print("*F");
//      Serial.println();
//    */
//
```

```
//
  }

  if ( pox.getHeartRate()>110 ||  pox.getSpO2()<90 && pox.getSpO2()>10 || cel > 50)
  {
   digitalWrite(buzzer_pin,HIGH);
   cond = "Ok";
   }
   else{
      digitalWrite(buzzer_pin,LOW);
      cond = "Abnormal";
     }

  StaticJsonBuffer<400> jsonBuffer;
  JsonObject& data = jsonBuffer.createObject();



   //Assign collected data to JSON Object
  data["bpm"] = pox.getHeartRate();
  data["satu"] = pox.getSpO2();
  data["temperature"] = cel;
  data["condition"] = cond;
```

```
//Send data to NodeMCU


data.printTo(nodemcu);

jsonBuffer.clear();


}
```

NodeMCU Script

```
//----------------------------------------Include the NodeMCU ESP8266 Library

#include <ESP8266WiFi.h>

#include <WiFiClientSecure.h>


//Include Lib for Arduino to Nodemcu

#include <SoftwareSerial.h>

#include <ArduinoJson.h>


//D6 = Rx & D5 = Tx

SoftwareSerial nodemcu(D6, D5);


#define ON_Board_LED 2  //--> Defining an On Board LED, used for indicators when the process of connecting to a wifi router


//----------------------------------------SSID and Password of your WiFi router.

const char* ssid = "REVOLT21"; //--> Your wifi name or SSID.

const char* password = "sir@sir@sir"; //--> Your wifi password.

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


//----------------------------------------Host & httpsPort

const char* host = "script.google.com";
```

```
const int httpsPort = 443;

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


WiFiClientSecure client; //--> Create a WiFiClientSecure object.


String GAS_ID = "AKfycbztfusPq-
0fkXCTYojjLpRS2f2Rwiq05pDEh4k3izgUr0GXUyrgMGAX9NShNWq-
B0XK4Q"; //--> spreadsheet script ID


//============================================================
================== void setup

void setup() {

  // put your setup code here, to run once:

  Serial.begin(115200);

  delay(500);


  nodemcu.begin(9600);

  while (!Serial) continue;




  WiFi.begin(ssid, password); //--> Connect to your WiFi router

  Serial.println("");
```

```
  pinMode(ON_Board_LED,OUTPUT); //--> On Board LED port Direction output
  digitalWrite(ON_Board_LED, HIGH); //--> Turn off Led On Board


  //------------------------------------------Wait for connection
  Serial.print("Connecting");
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    //----------------------------------------Make the On Board Flashing LED on the process of connecting to the wifi router.
    digitalWrite(ON_Board_LED, LOW);
    delay(250);
    digitalWrite(ON_Board_LED, HIGH);
    delay(250);
    //----------------------------------------
  }
  //----------------------------------------
  digitalWrite(ON_Board_LED, HIGH); //--> Turn off the On Board LED when it is connected to the wifi router.
  //----------------------------------------If successfully connected to the wifi router, the IP Address that will be visited is displayed in the serial monitor
  Serial.println("");
  Serial.print("Successfully connected to : ");
```

```
  Serial.println(ssid);
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());
  Serial.println();



  client.setInsecure();
}



//============================================================
=================== void loop
void loop() {

  StaticJsonBuffer<500> jsonBuffer;
  JsonObject& data = jsonBuffer.parseObject(nodemcu);

  if (data == JsonObject::invalid()) {
    //Serial.println("Invalid Json Object");
    jsonBuffer.clear();
    return;
  }
```

```
Serial.println("JSON Object Recieved");

Serial.print("Recieved BPM:  ");

int pbp = data["bpm"];

Serial.println(pbp);

Serial.print("Recieved Temperature:  ");

float temp = data["temperature"];

Serial.println(temp);

Serial.print("Recieved SpO2:  ");

int spo = data["satu"];

Serial.println(spo);

Serial.print("Patient Condition:  ");

int con = data["condition"];

Serial.println(con);

Serial.println("----------------------------------------");




String Temp = "Temperature : " + String(temp) + " °C";

String Pbpm = "Heartbeat : " + String(pbp) + " /s";

String Spot = "Saturation : " + String(spo) + " %";

String Cond = "Condition : " + String(con) + " !";
```

```
  Serial.println(Temp);
  Serial.println(Pbpm);
  Serial.println(Spot);
  Serial.println(Cond);

  sendData(temp, pbp, spo, con); //--> Calls the sendData Subroutine
}
//============================================================
==================
//============================================================
================== void sendData
// Subroutine for sending data to Google Sheets
void sendData(float temp, int pbp, int spo, int con) {
  Serial.println("==========");
  Serial.print("connecting to ");
  Serial.println(host);

  //----------------------------------------Connect to Google host
  if (!client.connect(host, httpsPort)) {
    Serial.println("connection failed");
    return;
  }
  //----------------------------------------
```

```
  //---------------------------------------Processing data and sending data
  String string_temperature =  String(temp);
  // String string_temperature =  String(tem, DEC);
  String string_heartbeat =  String(pbp);
  String string_saturation =  String(spo);
  String string_condition =  String(con);
  String url = "/macros/s/" + GAS_ID + "/exec?temperature=" + string_temperature + "&heartbeat=" + string_heartbeat + "&saturation=" + string_saturation + "&condition=" + string_condition;
  Serial.print("requesting URL: ");
  Serial.println(url);

  client.print(String("GET ") + url + " HTTP/1.1\r\n" +
        "Host: " + host + "\r\n" +
        "User-Agent: BuildFailureDetectorESP8266\r\n" +
        "Connection: close\r\n\r\n");

  Serial.println("request sent");
  //---------------------------------------

  //---------------------------------------Checking whether the data was sent successfully or not
```

```
  while (client.connected()) {
    String line = client.readStringUntil('\n');
    if (line == "\r") {
      Serial.println("headers received");
      break;
    }
  }
  String line = client.readStringUntil('\n');
  if (line.startsWith("{\"state\":\"success\"")) {
    Serial.println("esp8266/Arduino CI successfull!");
  } else {
    Serial.println("esp8266/Arduino CI has failed");
  }
  Serial.print("reply was : ");
  Serial.println(line);
  Serial.println("closing connection");
  Serial.println("==========");
  Serial.println();
  //----------------------------------------
}
```