

COVIGUARD – INTELLIGENT SHIRT USING ARDUINO FOR PROTECTION AGAINST COVID

Submitted in partial fulfillment of the requirements for the award of
Bachelor of Technology in B.Tech Biomedical Engineering

by

AISHVARIYA SHIVANI.S (39240002)
AISHWARYA MOHAN KUMAR (39240004)



DEPARTMENT OF BIOMEDICAL ENGINEERING
SCHOOL OF BIO AND CHEMICAL ENGINEERING

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade “A” by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119

APRIL – 2022



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited "A" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF BIOMEDICAL ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Interdisciplinary Project Report is the bonafide work of **AISHVARIYA SHIVANI.S (39240002)** and **AISHWARYA MOHAN KUMAR (39240004)** who carried out the interdisciplinary project entitled **"COVIGUARD – INTELLIGENT SHIRT USING ARDUINO FOR PROTECTION AGAINST COVID"** under our supervision from **January to April 2022**

DR.S.KRISHNAKUMAR, M.Sc. Ph.D.,

Internal Guide

YADAVA PRASATH.G

External Guide

Dr.T.SUDHAKAR, M.Sc., Ph.D.,

Head of the Department

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

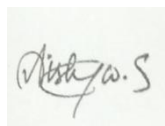
We, **AISHVARIYA SHIVANI.S (39240002)** and **AISHWARYA MOHAN KUMAR (39240004)**, hereby declare that the Interdisciplinary Project Report entitled “**COVIGUARD – INTELLIGENT SHIRT USING ARDUINO FOR PROTECTION AGAINST COVID**” done by us under the guidance of **Dr.S.KRISHNAKUMAR M.Sc., Ph.D., (Internal)** and **YADAVA PRASATH.G, CEO of SIERO (External)**, is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology degree in B.Tech Biomedical Engineering.

1. **Aishwarya Mohan Kumar**



DATE: 30th April 2022

2. **Aishvariya Shivani.S**



PLACE: CHENNAI

SIGNATURE OF THE CANDIDATES

ACKNOWLEDGEMENT

We are pleased to acknowledge our sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this interdisciplinary project and for completing it successfully. We are grateful to them.

We convey our thanks to **Dr.T.Sudhakar, M.Sc., Ph.D,** and **Dr.J.Premkumar, M.Sc., Ph.D., Head of the Department, Dept. of Biomedical Engineering** for providing us necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our Project Guide **Dr.S.Krishnakumar, M.Sc., Ph.D., Associate Professor, Dept. of Biomedical Engineering** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of our project work.

We wish to express our thanks to all Teaching and Non-teaching staff members of the Department of Biomedical Engineering who were helpful in many ways for the completion of this interdisciplinary project.

ABSTRACT

In the current pandemic situation, it is important for us to follow certain precautionary measures in order to safeguard us from the deadly virus. We have been able to contain the virus to a certain extent through social distancing, by sanitizing ourselves and sterilizing the daily-use items. Monitoring the vitals like body temperature, oxygen saturation and pulse rate have proven to be effective in diagnosing the fatal disease. In this proposed method, we have come up with a solution to help the user to keep a check on the important parameters mentioned above by incorporating various sensors like MLX90614 non-contact IR temperature sensor, SpO2 sensor, pulse rate sensor and ultrasonic sensor in a shirt- CoviGuard. The vitals are displayed on an IOT application called ThingSpeak. A buzzer is used to indicate if the user doesn't maintain the specified distance of 0.5 meters.

Keywords- Smart shirt, social distancing, SpO2, temperature, pulse rate, ThingSpeak

CONTENTS

Acknowledgement	iv
Abstract	v
CHAPTER 1 – INTRODUCTION	1
1.1. Basic Introduction	1
1.2. Body Temperature	1
1.3. Heart Rate	1
1.4. Oxygen Saturation	2
1.5. Internet of Things	2
CHAPTER 2 – LITERATURE REVIEW	3
2.1 Introduction	3
2.2 Literature survey	3
CHAPTER 3 – AIM AND SCOPE	6
3.1. Aim	6
3.2. Scope	6
CHAPTER 4 – MATERIALS AND METHODS	7
4.1. Hardware components	7
4.1.1. Power board	7
4.1.2. Arduino UNO rev3	7
4.1.3. MLX90614 - Infrared non-contact temperature sensor	8
4.1.4. MAX30100 – Heart rate and pulse oximeter sensor module	8
4.1.5. HC SR04 Ultrasonic sensor	9

4.1.6. NODE MCU 32S – ESP32	10
4.1.7. TMB12A12 Buzzer	11
4.2. Software components	12
4.2.1. Arduino IDE	12
4.2.2. ThingSpeak	12
4.3. Block diagram	13
4.4. Working	13
CHAPTER 5 – RESULTS AND DISCUSSION	16
5.1. Body temperature	16
5.2. Heart rate	17
5.3. Oxygen saturation	18
CHAPTER 6 – SUMMARY AND CONCLUSION	19
6.1. Summary	19
6.2. Conclusion	19
Reference	20
Appendix	21

List of figures

Figure	Title	Page
4.1	Power Board	7
4.2	Arduino UNO REV3	8
4.3	MLX90614 Non contact IR temperature sensor	8
4.4	MAX30100 pulse oximeter and heart rate sensor module	9
4.5	HC SR04 ultrasonic sensor	10
4.6	Node MCU 32S – ESP32	11
4.7	TMB12A12 buzzer	11
4.8	Block diagram	13
4.9	Prototype model	14
4.10	Ultrasonic sensors for social distancing	14
4.11	MAX30100 module connected to Node MCU	15
4.12	Subject wearing the prototype	15
5.1	Output of the Body temperature	16
5.2	Output of the Heart rate	17
5.3	Output of the Oxygen saturation	18

CHAPTER – 1

INTRODUCTION

1.1. BASIC INTRODUCTION

In the present scenario, it is critical that everyone must take certain preventive measures to protect themselves from the deadly virus. We have been able to contain the virus to a certain extent through social distancing, by sanitizing ourselves and sterilizing the daily-use items. Monitoring the vitals like body temperature, oxygen saturation and heart rate has proven to be effective in diagnosing the fatal disease.

1.2. BODY TEMPERATURE

The normal human adult body temperature is found to be 37°C (98.6°F) but it may vary according to different individuals. Body temperature is one among the four key vital signs and also being a symptom that can be measured for diagnosing major illnesses like respiratory infections and cancers. High body temperature indicates fever and low body temperature indicates viral infection.

1.3. HEART RATE

The heart or pulse rate refers to the number of times heart beats per minute. Normal range of heart rate for an adult is between 60 to 100 beats per minute (bpm). High pulse rate is considerably more than 100bpm. During high pulse rate the muscles are weakened by virus and this forces the heart to beat more often to pump enough blood to rest of the body. High pulse rate is caused due to underlying conditions such as fear, anxiety, anger and stress. High pulse rate shows symptoms such as fainting, fatigue, shortness of breath and fluttering in the chest. It is also called as tachycardia. Slow heart rate is considerably less than 60bpm. During slow heart rate the heart beats slowly. Slow heart rate is followed by symptoms like confusion, dizziness and can even

cause cardiac arrest. Slow heart rate is also known as bradycardia.

1.4. OXYGEN SATURATION

Oxygen saturation or "Pulse ox" indicates the amount of oxygen traveling through the body within the red blood cells. Normal oxygen saturation is usually around 95% to 100% for most healthy adults. An abnormal level of oxygen saturation indicates sickness and diseases like respiratory infections, asthma and heart diseases.

1.5. INTERNET OF THINGS

The internet of things is defined as a system of interrelated, internet-connected objects which makes it possible to collect and transfer data over a wireless network without giving room for any errors because it does not require any human intervention. This method is proven efficient compared to other data transfer methods.

Remote monitoring in the healthcare sector is now possible thanks to Internet of Things (IoT)-enabled devices, which have the ability to keep patients safe and secure while also inspiring physicians to provide superior treatment. The Internet of Things (IoT) is undeniably changing the healthcare industry by redefining the space of devices and human involvement in the delivery of healthcare solutions.

CHAPTER – 2

LITERATURE REVIEW

2.1. INTRODUCTION

The following shows the survey done for CoviGuard. Some of the recent research works done related to our project has been discussed as follows.

2.2. LITERATURE SURVEY

Nakul Sharma et al., (2020) has constructed a smart jacket that has three ultrasonic sensors, buzzer, LCD and Arduino UNO. It is used to maintain social distancing by giving off an alarm. The jacket functions with the help of an Arduino board system which comes with an inbuilt LED. It is then connected to a buzzer to alert the user immediately. The distance can be maintained in the left, right and back of the user using an alarm system. At the same time the distance values are indicated on an LCD monitor.

Irmandy Wicaksono et al., (2020) have developed a tailored, thin, soft electronic textile conformable suit (E-TeCS) to perform large-scale, multimodal physiological (temperature, heart rate, and respiration) sensing in vivo. This platform can be customized for various forms, sizes and functions using standard, accessible and high-throughput textile manufacturing and garment patterning techniques. Similar to a compression shirt, the soft and stretchable nature of the tailored E-TeCS allows intimate contact between electronics and the skin with a pressure value of around ~25 mmHg, allowing for physical comfort and improved precision of sensor readings on skin.

Yuxuan Liu et al., (2021) in his review paper stated that a variety of symptoms of COVID-19 can be monitored by different types of soft wearable sensors, such as body temperature (temperature sensor), respiration (temperature and humidity sensors),

oxygen saturation (pulse oximeter), ECG and HR (ECG electrode), and cough (pressure and strain sensors and stethoscope). This wearable sensor provides potential opportunities to detect and monitor these parameters rapidly and remotely, which offers the following advantages: (a) providing early warning of COVID-19 infection; (b) closely monitoring the clinical situation and preventing sudden worsening of the situation; (c) maintaining distance between medical staff and patients during screening and monitoring; (d) reducing the load of the medical system with at-home monitoring; (e) enabling data-level analysis when integrated with artificial intelligence.

Naqiyuddin et al., (2020) has developed a system that comprises of an ultrasonic sensor (HC-SR04), a microcontroller, a buzzer and an LCD Module. The function of the microcontroller is to read the value from the sensor, calculate the distance between the subject and a person behind him and remind the person if the distance is less than a meter. The buzzer was included to produce a sound whenever a person behind is closer than 1 meter. The function of the LCD module is to display the distance between the subject and the person behind him to automatically detect social distancing, an ARM C language program was developed using STM32 Cube Mx firmware. The distance measured was displayed on a computer screen using software called RealTerm Serial Capture Program. When there is a person within the distance of less than 1 meter, the buzzer produces a warning sound while the LCD displays the distance and also a warning message "WARNING! STEP BACK". When the obstacle is over a meter from the wearer, the LCD displays the distance and message, "SAFE DISTANCE", to notify the person is in a secure distance.

Raghav et al., (2020) developed a prototype that consists of PIR motion that detects humans and animal bodies and not arbitrary objects. The detection range of PIR sensors is comparatively larger, which is about 120° , hence, by using three sensors and in the right angle, it enables the device to help cover 360° . If any person comes within the range of radius of 1.5m from the user, the Buzzer and LED will turn ON. The scan response time of the PIR sensors is 1s. The loop will exit after 1 second. The Suraksha device is capable of alerting the user of their surroundings at anytime and

anywhere. The device is more preferable as the user need not worry about registering/creating an account or even installing an app in their mobile phones to be aware of their surroundings at all times.

CHAPTER – 3

AIM AND SCOPE

3.1. AIM

The aim of our project is to help people protect themselves from the vicious corona virus by aiding them to follow some precautionary methods. In this proposed method, we have come up with a solution to help the user to keep in check on the important parameters by incorporating various sensors like IR temperature sensor, SpO2 sensor, pulse rate sensor and ultrasonic sensor in a shirt- CoviGuard. The vitals are displayed on an IoT application called ThingSpeak for analysis and monitoring of the vitals. A buzzer is used to indicate if the user doesn't maintain the specified distance of 0.5 meters.

3.2. SCOPE

In future with necessary financial aid and technical assistance further development of diagnosis can be provided. For example, respiration rate and glucose concentration can be detected and monitored for sick patients or can be used for drug monitoring. This real-time tracking data can be displayed in smart watches and mobiles. The data of these vital signs can also be used for analysis of the treatments or therapies. The diagnosed data can also be sent to the doctors and healthcare experts for further diagnosis. Overall, this intelligent shirt can be conformable and personalized according to the patient or end user with fabricating different soft-wearable sensors.

CHAPTER – 4

MATERIALS AND METHODS

4.1. HARDWARE COMPONENTS

4.1.1 POWER BOARD

A power board is used to provide electric power to the system and typically can be a battery, USB cable, A adapter, or a regulated power source device. It can be used with any power adapter that gives 7 to 20 volts and will regulate the voltage down to a very clean 5 volts. The adapter must be DC (Direct current).

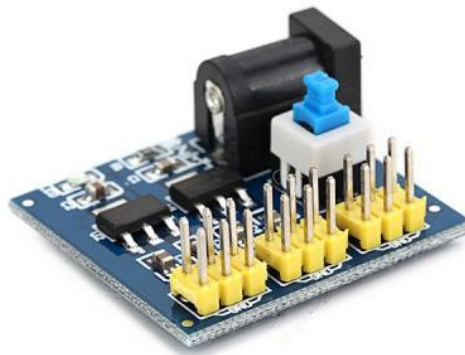


Fig 4.1.Power Board

4.1.2. ARDUINO UNO REV3

Arduino UNO is a low-cost, flexible, and easy-to-use programmable open-source microcontroller board that can be integrated into a variety of electronic projects. This board can be interfaced with other Arduino boards, Arduino shields, Raspberry Pi boards and can control relays, LEDs, servos, and motors as an output. Arduino UNO features AVR microcontroller Atmega328, 6 analogue input pins, and 14 digital I/O pins out of which 6 are used as PWM output.



Fig 4.2.Arduino UNO REV3 SMD

4.1.3. MLX90614 - INFRARED NON-CONTACT TEMPERATURE SENSOR

The MLX90614 is an Infra Red thermometer for non contact temperature measurements. Both the IR sensitive thermopile detector chip and the signal conditioning ASIC are integrated in the same TO-39 can. Integrated into the MLX90614 are a low noise amplifier, 17-bit ADC and powerful DSP unit thus achieving high accuracy and resolution. The sensor is factory calibrated with a digital SMBus output giving full access to the measured temperature in the complete temperature range(s) with a resolution of 0.02°C. The user can configure the digital output to be PWM. As a standard, the 10-bit PWM is configured to continuously transmit the measured temperature in range of -20 to 120 °C, with an output resolution of 0.14 °C.



Fig 4.3.MLX90614 Non contact IR temperature sensor

4.1.4. MAX30100 – HEART RATE AND PULSE OXIMETER SENSOR MODULE

The MAX30100 pulse oximeter and heart rate sensor is an I2C-based low-power plug-and-play biometric sensor. The module features the MAX30100 – a modern, integrated pulse oximeter and heart rate sensor IC, from Analog Devices. It combines two LEDs, a photodetector, optimized optics, and low-noise analog signal processing to detect pulse oximetry (SpO₂) and heart rate (HR) signals. One of the most important features of the MAX30100 is its low power consumption: the MAX30100 consumes less than 600µA during measurement. Also it is possible to put the MAX30100 in standby mode, where it consumes only 0.7µA. This low power consumption allows implementation in battery powered devices such as handsets, wearables or smart watches. This is a reasonably precise temperature sensor that measures the temperature in the range of -40°C to +85°C with an accuracy of ±1°C.



Fig 4.4. MAX30100 pulse oximeter and heart rate sensor module

4.1.5. HC SR04 ULTRASONIC SENSOR

The HC-SR04 Ultrasonic Distance Sensor is a sensor used for detecting the distance to an object using sonar. The HC-SR04 uses non-contact ultrasound sonar to measure the distance to an object, and consists of two ultrasonic transmitters, a receiver, and a control circuit. The transmitters emit a high frequency ultrasonic sound, which bounce

off any nearby solid objects, and the receiver listens for any return echo. That echo is then processed by the control circuit to calculate the time difference between the signal being transmitted and received. This time can subsequently be used, along with some clever math, to calculate the distance between the sensor and the reflecting object. The HC-SR04 sensor works best between 2cm – 400 cm (1" - 13ft) within a 30 degree cone, and is accurate to the nearest 0.3cm.



Fig 4.5.HC SR04 ultrasonic sensor

4.1.6. NODE MCU 32S – ESP32

NodeMCU-32S is a Lua WiFi IOT development board based on ESP-32S module. Some of its features are:

- Onboard ESP32-S module
- Onboard CH340, USB to UART converter
- USB port for power input, firmware programming, or UART debugging
- 2x19pin extension headers, breakout all the I/O pins of the module
- 2x keys, used as reset or user-defined



Fig 4.6. Node MCU 32S – ESP32

4.1.7. TMB12A12 BUZZER

An audio signaling device like a beeper or buzzer may be electromechanical or piezoelectric or mechanical type. The main function of this is to convert the signal from audio to sound. Generally, it is powered through DC voltage and used in timers, alarm devices, printers, alarms, computers, etc. Based on the various designs, it can generate different sounds like alarm, music, bell & siren. This type of buzzer is made with a magnet, solenoid coil, oscillator, housing, vibration diaphragm, and magnet. Once the power supply is given, the oscillator which produces the audio signal current will supply throughout the solenoid coil to generate a magnetic field. Sometimes, the vibration diaphragm will vibrate & generates sound under the magnet & solenoid coil interaction. The frequency range of this buzzer ranges from 2 kHz to 4 kHz.



Fig 4.7. TMB12A12 buzzer

4.2. SOFTWARE COMPONENTS

4.2.1. ARDUINO IDE

The Arduino IDE is an open-source software, which is used to write and upload code to the Arduino boards. The IDE application is suitable for different operating systems such as Windows, Mac OS X, and Linux. It supports the programming languages C and C++. IDE stands for Integrated Development Environment. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub `main()` into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program *avrdude* to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware.

4.2.2. THINGSPEAK

ThingSpeak is a cloud-based IoT analytics application that allows you to compile, visualize and analyse live data streams. ThingSpeak provides instant visualizations of information posted by your devices to ThingSpeak. With the power to execute MATLAB® code in ThingSpeak you'll perform online analysis and processing of the information because it comes in. ThingSpeak is usually used for prototyping and proof of concept IoT systems that need analytics. ThingSpeak is a cloud-based tool for aggregating, visualising and analysing live data streams. A number of the key capabilities of ThingSpeak include the flexibility to:

- Configure common IoT protocols to send data to ThingSpeak with ease.
- Visualize your sensor data in real-time.
- Aggregate data on-demand from third-party sources.
- Use the ability of MATLAB to create sense of your IoT data.

- Run your IoT analytics automatically supported schedules or events.
- Prototype and build IoT systems without fitting servers or developing web software.
- Use third-party applications like Twilio® or Twitter® to automatically action your data and connect.

4.3. BLOCK DIAGRAM

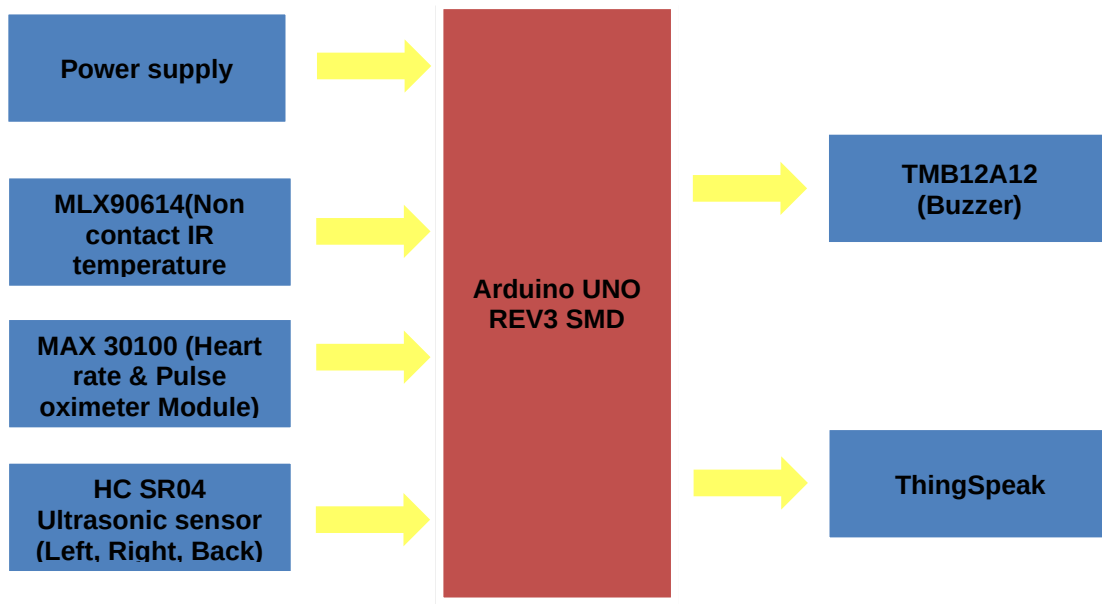


Fig 4.8. Block diagram of CoviGuard coat

4.4. WORKING

The proposed system consists of an Arduino UNO microcontroller that has a built-in Wi-Fi module and other vital detecting sensors like MAX30100 (Heart rate and SpO₂ module), MLX90614 (Non contact IR temperature sensor) to measure heart rate, oxygen saturation and temperature respectively. The readings are collected by placing the subject's finger above the sensors. The system then transmits the data collected to the ThingSpeak cloud database via Node MCU. The database credentials are fed into the firmware of the microcontroller for the data transfer to take place. ThingSpeak helps the user to visualize, analyse and monitor his/her vital signs and hence can stay

alarmed if it crosses the threshold (normal) value. The intelligent shirt also contains three ultrasonic sensors (each on the left, right, and back) and a buzzer that helps the user to maintain proper social distance from others. If the ultrasonic sensor detects a person in the surrounding within the specified distance then it would alert the user through the buzzing sound.



Fig 4.9. Prototype model

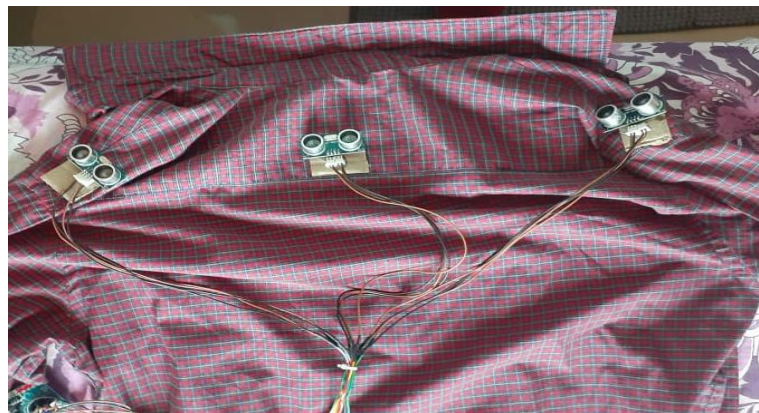


Fig 4.10. Ultrasonic sensors for social distancing

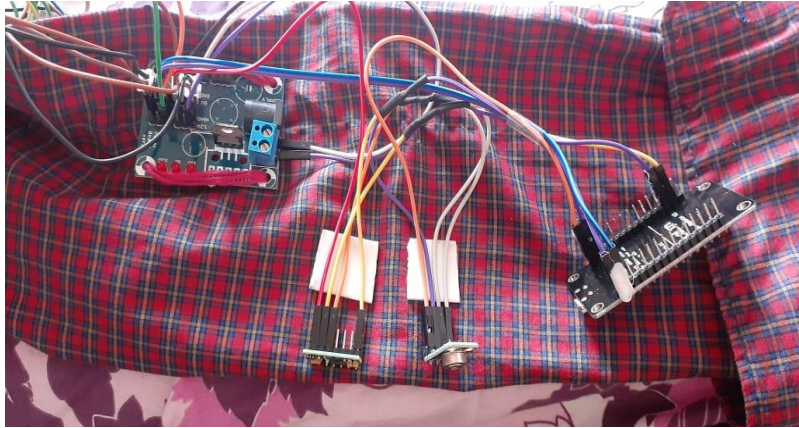


Fig 4.11. MAX30100 module connected to Node MCU



Fig 4.12. Subject wearing the prototype

CHAPTER – 5

RESULTS AND DISCUSSION

5.1. BODY TEMPERATURE

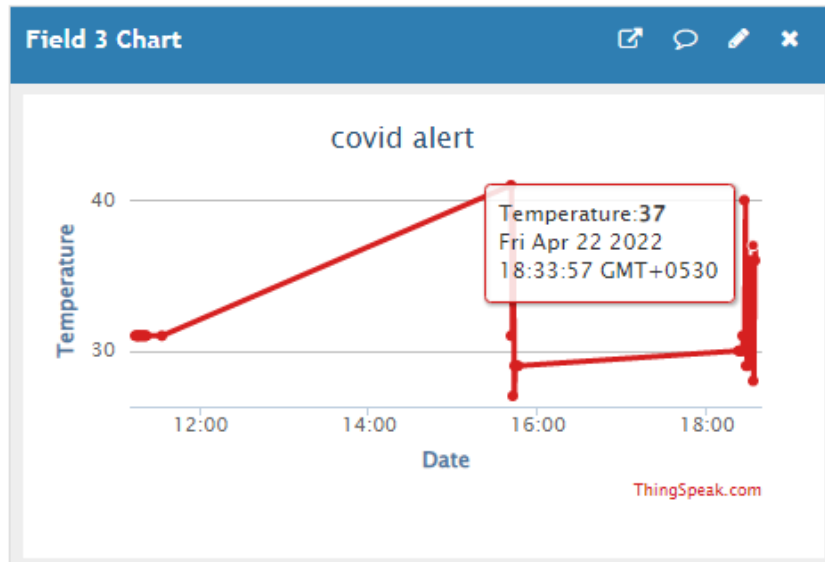


Figure 5.1 Output of Body temperature

The above image is the output from the IR temperature sensor displayed as a graph via ThingSpeak providing the details such as day, date, and time to tell us when the temperature data was acquired. A healthy human adult body temperature is 37° Celsius (98.6°F). The value obtained and displayed is 37 degree Celsius.

5.2. HEART RATE

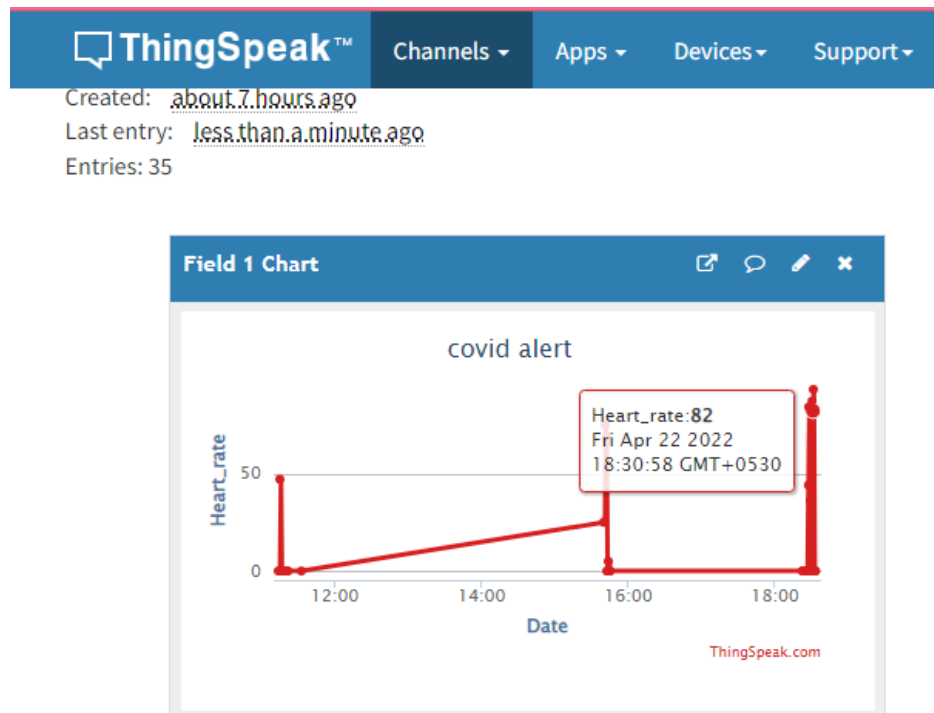


Figure 5.2 Output of Heart rate

The above figure is the output from the heart rate sensor MODULE MAX30100 and displayed as a graph via ThingSpeak providing the details such as day, date, and time to tell us when the heart rate data was acquired. The normal value for an adult human ranges from 60-100 beats per minute. The value obtained and displayed is 82 beats per minute.

5.3. OXYGEN SATURATION

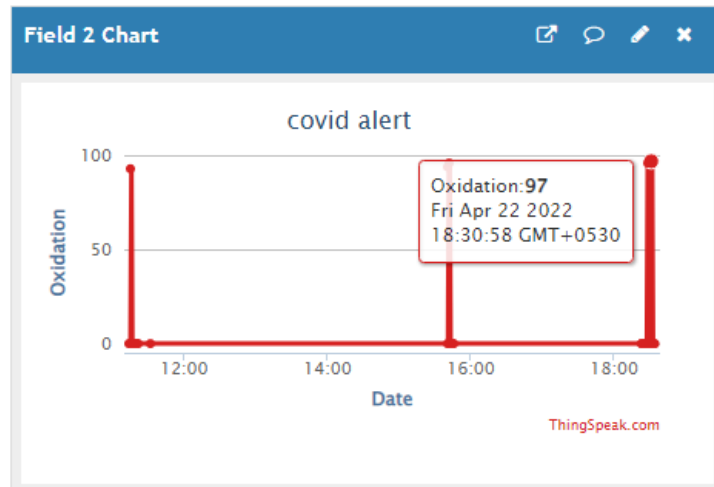


Figure 5.3 Output of Oxygen Saturation

The above figure is the output from the pulse oximeter MODULE MAX30100 and displayed as a graph via ThingSpeak providing the details such as day, date, and time to tell us when the oxygen saturation data was acquired. Normal oxygen saturation for an adult human being is usually between 95% and 100%. The value obtained and displayed is 97 percentage.

CHAPTER – 6

SUMMARY AND CONCLUSION

6.1 SUMMARY

The purpose of this smart shirt is to periodically monitor vital signs such as body temperature, heart rate, and oxygen saturation. This enables quicker medical assistance and the mitigation of serious disease risks. By sending this critical information to healthcare professionals, they will be able to rapidly diagnose the disease and administer the appropriate treatment.

With the help of a buzzer, this shirt also encourages in maintaining social distance. We have learnt the need of improving and upgrading our healthcare infrastructure, especially in light of recent pandemics. Within the health infrastructure budget, which has remained between 1.2 and 1.6 percent of total GDP for the past five years, Coviguard would serve a better benefit in terms of getting a quality and functional health monitoring system within the allocated budget.

Because it uses IOT as a platform to relay the data, it saves a lot of time, and the data gathered is also preserved in the cloud for future use. The possibilities for the intelligent shirt and the Internet of Things are endless. Additional study has turned up a slew of examples, more improved statistics and diagnostic systems that might be supplied as a result of numerous improvements.

6.2 CONCLUSION

The intelligent shirt, Coviguard, allows the user to visualize and monitor his or her vital signs, such as heart rate, oxygen saturation, and temperature, on a desktop or mobile phone, which are useful in detecting the disease called corona virus. The shirt also employs a microcontroller, ultrasonic sensors, and a buzzer to alert the user if the necessary social distance of 0.5 metres is not maintained.

REFERENCES

1. Irmandy Wicaksono, Carson I. Tucker, Tao Sun, Cesar A. Guerrero, Clare Liu, Wesley M. Woo, Eric J. Pence & Canan Dagdeviren, 'A tailored, electronic textile conformable suit for large-scale spatiotemporal physiological sensing in vivo', *npj Flexible Electronics* volume 4, Article number: 5 April(2020)\
2. Nakul Sharma, Parth A Shah, Milind Chitale, Nikita Bavishi Shah, 'Smart Jacket for Social distancing', *International Journal of Scientific Research and Engineering Development(IRJET)* ISSN : 2581-7175 Volume 3 - Issue 5, Sep - Oct 2020
3. Naqiyuddin F. A. A., W. Mansor, N. M. Sallehuddin, M. N. S. Mohd Johari, M. A. S. Shazlan and A. N. Bakar, 'Wearable Social Distancing Detection System,' 2020 IEEE International RF and Microwave Conference (RFM), 2020, pp. 1-4, doi: 10.1109/RFM50841.2020.9344786.
4. Raghav S. *et al.*, 'Suraksha: Low Cost Device to Maintain Social Distancing during CoVID-19,' 2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA), 2020, pp. 1476-1480, doi: 10.1109/ICECA49313.2020.9297503.
5. Yuxuan Liu, Darpan Shukla, Holly Newman and Yong Zhu, Soft wearable sensors for monitoring symptoms of COVID-19 and other respiratory diseases: a review, *Progress in Biomedical Engineering*, Volume 4, Number 1, October 2021

APPENDIX

1)

```
// defines arduino pins numbers
const int echoPin_1 = 13;
const int trigPin_1 = 12;
const int echoPin_2 = 11;
const int trigPin_2 = 10;
const int echoPin_3 = 9;
const int trigPin_3 = 8;
// defines variables
long duration_1, duration_2, duration_3;
int distance_1, distance_2, distance_3;
int alert_1, alert_2, alert_3;

void setup()
{
  pinMode(trigPin_1, OUTPUT); // Sets the trigPin_1 as an Output
  pinMode(echoPin_1, INPUT); // Sets the echoPin_1 as an Input
  pinMode(trigPin_2, OUTPUT); // Sets the trigPin_1 as an Output
  pinMode(echoPin_2, INPUT); // Sets the echoPin_1 as an Input
  pinMode(trigPin_3, OUTPUT); // Sets the trigPin_1 as an Output
  pinMode(echoPin_3, INPUT); // Sets the echoPin_1 as an Input
  pinMode(5, OUTPUT);
  digitalWrite(5, LOW);
  Serial.begin(9600); // Starts the serial communication
}

void loop() {
  delay(100);
  Serial.println("ultra");
  // Clears the trigPin_1
  digitalWrite(trigPin_1, LOW);
  delayMicroseconds(2);
  // Sets the trigPin_1 on HIGH state for 10 micro seconds
  digitalWrite(trigPin_1, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin_1, LOW);
  // Reads the echoPin_1, returns the sound wave travel time in microseconds
  duration_1 = pulseIn(echoPin_1, HIGH);
  // Calculating the distance
  distance_1 = duration_1 * 0.034 / 2;
  // Prints the distance on the Serial Monitor
  // Serial.print("Distance from the object 1 = ");
  // Serial.print(distance_1);
  // Serial.println(" cm");
  delay(10);

  // Clears the trigPin_2
  digitalWrite(trigPin_2, LOW);
  delayMicroseconds(2);
  // Sets the trigPin_2 on HIGH state for 10 micro seconds
  digitalWrite(trigPin_2, HIGH);
```

```

delayMicroseconds(10);
digitalWrite(trigPin_2, LOW);
// Reads the echoPin_2, returns the sound wave travel time in microseconds
duration_2 = pulseIn(echoPin_2, HIGH);
// Calculating the distance
distance_2 = duration_2 * 0.034 / 2;
// Prints the distance on the Serial Monitor
// Serial.print("Distance from the object 2 = ");
// Serial.print(distance_2);
// Serial.println(" cm");
delay(10);

// Clears the trigPin_3
digitalWrite(trigPin_3, LOW);
delayMicroseconds(2);
// Sets the trigPin_3 on HIGH state for 10 micro seconds
digitalWrite(trigPin_3, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin_3, LOW);
// Reads the echoPin_3, returns the sound wave travel time in microseconds
duration_3 = pulseIn(echoPin_3, HIGH);
// Calculating the distance
distance_3 = duration_3 * 0.034 / 2;
// Prints the distance on the Serial Monitor
// Serial.print("Distance from the object 3 = ");
// Serial.print(distance_3);
// Serial.println(" cm");
delay(10);
if(distance_1 < 50)
{
    alert_1 = 1;
}
else
{
    alert_1 = 0;
}
if(distance_2 < 50)
{
    alert_2 = 1;
}
else
{
    alert_2 = 0;
}
if(distance_3 < 50)
{
    alert_3 = 1;
}
else
{
    alert_3 = 0;
}
// Serial.print(alert_1);
// Serial.print(alert_2);
// Serial.println(alert_3);
if((alert_1 == 1) || (alert_2 == 1) || (alert_3 == 1))
{

```

```
//    Serial.print("buzzer HIGH");
    digitalWrite(5, HIGH);
}
else
{
//    Serial.print("buzzer LOW");
    digitalWrite(5, LOW);
}
}
```

2)

```
#include <ESP8266WiFi.h>
#include <ThingSpeak.h>
#include <Wire.h>
#include "MAX30100_PulseOximeter.h"
#include <Adafruit_MLX90614.h>
#include <Ticker.h> //Ticker Library

Ticker timer;

Adafruit_MLX90614 mlx = Adafruit_MLX90614();

#define REPORTING_PERIOD_MS    1000

PulseOximeter pox;
uint32_t tsLastReport = 0;
unsigned int heart = 0, oxidation = 0, data_timer = 0, temp = 0, flag=0,
flag_timer=0;

const char* ssid = "SIEORA"; //Your Network SSID

const char* wifi_password = "sieora@123"; //Your Network Password

WiFiClient wifiClient;

unsigned long myChannelNumber = 1709000; //Your Channel Number (Without
Brackets)
    char data[18];
    int val,i;

unsigned int gas = 0, x = 0, y = 0, z = 0, t = 0, h = 0,p = 0,a = 0;
int data_received=0;
String
no_of_sensor,sensor1,sensor2,sensor3,sensor4,sensor5,sensor6,sensor7,sensor8
;
char
gas_value[10],x_value[10],y_value[10],z_value[10],t_value[10],h_value[10],p_
value[10],a_value[10];
int send_data=0;

int LDRpin = A0; //LDR Pin Connected at A0 Pin

const char * myWriteAPIKey = "1JY6GD XD U5C3WF40"; //Your Write API Key
    const int analogInPin = A0; // Analog input pin that the potentiometer is
attached to
```

```

char str[100];
int sensorValue = 0;          // value read from the pot
int outputValue = 0;
char value=0; // value output to the PWM (analog out)
int number1 = 0;
int number2 = random(0,100);
int number3 = random(0,100);
int number4 = random(0,100);
String myStatus = "";

void onBeatDetected()
{
    //    Serial.println("Beat!");
}
void setup()
{

    Serial.begin(9600);
    Serial.print("Connecting to ");
    Serial.println(ssid);
    // Connect to the WiFi
    WiFi.begin(ssid, wifi_password);
    // Wait until the connection has been confirmed before continuing
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }
    // Debugging - Output the IP Address of the ESP8266
    Serial.println("WiFi connected");
    Serial.print("IP address: ");
    Serial.println(WiFi.localIP());

    ThingSpeak.begin(wifiClient);

    //    //Initialize Ticker every 0.5s
    timer.attach_ms(1000, changeState); //Use <strong>attach_ms</strong> if
you need time in ms
    //    timer.attach_ms(1000, thingspeak_data); //Use
<strong>attach_ms</strong> if you need time in ms

    // put your setup code here, to run once:
    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");
    }
    pox.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);

```

```

    // Register a callback for the beat detection
    pox.setOnBeatDetectedCallback(onBeatDetected);
    Serial.println("Adafruit MLX90614 test");

    mlx.begin();
}

void loop()
{
    // delay(100);
    // Make sure to call update as fast as possible
    pox.update();
    if (millis() - tsLastReport > REPORTING_PERIOD_MS) {
        // Serial.print("Heart rate:");
        // Serial.print(pox.getHeartRate());

        heart = pox.getHeartRate();
        // Serial.print(heart);

        // Serial.print("bpm / SpO2:");
        // Serial.print(pox.getSpO2());
        oxidation = pox.getSpO2();

        // Serial.println(oxidation);
        // Serial.println("%");

        tsLastReport = millis();
    }
    temp = mlx.readObjectTempC();

    //flag_timer++;
    //if(flag_timer >= 15000)
    //{
    //    flag_timer = 0;
    //    Serial.println(" * ");
    //    thingspeak_data();
    //}
    //Serial.println(temp);
}

void changeState()
{
    data_timer++;
    if (data_timer > 40)
    {
        data_timer = 0;
        Serial.print('2');
        Serial.print(',');
        sensor1 = String(heart);
        Serial.print(sensor1);
        Serial.print(',');
        sensor2 = String(oxidation);
        Serial.print(sensor2);
        Serial.print(',');
        sensor3 = String(temp);
    }
}

```

```

        Serial.print(sensor3);
        Serial.print(',');
        ThingSpeak.setField(1, sensor1);
        ThingSpeak.setField(2, sensor2);
        ThingSpeak.setField(3, sensor3);
        //      delay(100);
//      delay(100);
//      delay(1000);
        send_data = 1;
//      delay(100);
//      thingspeak_data();
if(send_data == 1)
{
    unsigned long myChannelNumber = 1709000; //Your Channel Number (Without
Brackets)
    const char * myWriteAPIKey = "1JY6GDxDU5C3WF40"; //Your Write API Key

    //data=data-3;

    ThingSpeak.writeFields(myChannelNumber, myWriteAPIKey);
    Serial.print("data sent");
    send_data = 0;

}

}

}

void thingspeak_data()
{
    if(send_data == 1)
    {
        unsigned long myChannelNumber = 1709000; //Your Channel Number (Without
Brackets)
        const char * myWriteAPIKey = "1JY6GDxDU5C3WF40"; //Your Write API Key

        //data=data-3;
        ThingSpeak.setField(1, sensor1);
        ThingSpeak.setField(2, sensor2);
        ThingSpeak.setField(3, sensor3);
        ThingSpeak.writeFields(myChannelNumber, myWriteAPIKey);
        Serial.print("data sent");
        send_data = 0;

    }

}

```