VISVESVARAYA TEHNOLOGICAL UNIVERSITY

"Jnana Sangama", Belagavi - 590018

A PROJECT REPORT ON

"CAMISA : An AI Solution for COVID-19"

Submitted in the partial fulfillment of the requirements for the award of the Degree of

BACHELOR OF ENGINEERING

in

ELECTRONICS AND COMMUNICATION ENGINEERING

Submitted By

PAULSON PREMSINGH S	**1MJ17EC090**
SRINIDHI K	**1MJ17EC133**
DIKSHITHA R	**1MJ17EC035**
ANIL KUMAR R	**1MJ17EC012**

Under the Guidance of

Mr. BHANUTEJA G

Asst. Professor, Dept. of ECE,

MVJ COLLEGE OF ENGINEERING Since 1982

An Autonomous Institute

External guide

Mr. KAUSTUBH ANAND K

Micro Electronics Design facility,

USRC, ISRO, Bengaluru

Sponsored by

KSCST

KARNATAKA STATE COUNCIL FOR SCIENCE AND TECHNOLOGY

Indian Institute of Science Campus,

Bengaluru, Karnataka 560012

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

MVJ COLLEGE OF ENGINEERING

Affiliated to VTU Belagavi, Approved by AICTE,

Recognized by UGC with 2(f) and 12(B) status, Accredited By NBA and NAAC

Near ITPB, Whitefield, Bengaluru – 560067

Academic year: 2020-21

COLLEGE OF ENGINEERING
Since 1982
Engineering A Better Tomorrow

An Autonomous Institute
Affiliated to VTU Belagavi, Approved by AICTE,
Recognized by UGC with 2(f) and 12(B) status, Accredited By NBA and NAAC
Near ITPB, Whitefield, Bengaluru – 560067

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

*This is to certify that the project work entitled "CAMISA:AN AI SOLUTION FOR COVID-19" is a bonafide work carried out by **PAULSON PREMSINGH S (1MJ17EC090), SRINIDHI K (1MJ17EC133), DIKSHITHA R (1MJ17EC035)** and **ANIL KUMAR R (1MJ17EC012)**, in partial fulfillment for the award of degree of **Bachelor of Engineering** in **Electronics & Communication Engineering** of the **Visvesvaraya Technological University, Belagavi** during the year 2020-21. It is certified that all the corrections/suggestions indicated for Internal Assessment have been incorporated in the Report. The project report has been approved as it satisfies the academic requirements.*

Signature of Guide	Signature of HOD	Signature of Principal
Mr. Bhanuteja G	**Dr. I Hameem Shanavas**	**Dr. P.Mahabaleshwarappa**
Assistant Professor,	Professor & Head,	Principal,
Dept. of ECE,	Dept. of ECE, MVJCE,	MVJCE, Bengaluru
MVJCE, Bengaluru	Bengaluru	

External Viva

Name of the Examiners	Signature with Date
1. _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _	1. _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
2. _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _	2. _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

COLLEGE OF ENGINEERING
Since 1982
Engineering A Better Tomorrow

An Autonomous Institute
Affiliated to VTU Belagavi, Approved by AICTE,
Recognized by UGC with 2(f) and 12(B) status, Accredited By NBA and NAAC
Near ITPB, Whitefield, Bengaluru – 560067

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

DECLARATION

We, hereby declare that this dissertation work entitled ***"CAMISA : An AI Solution for COVID 19"*** *was carried out by us as a team under the guidance and supervision of* ***Mr. BHANUTEJA G,*** *Assistant professor, Department of ECE, MVJ College of Engineering, Bengaluru. This project work is submitted to* ***Visvesvaraya Technological University, Belagavi****, in partial fulfillment of the requirement for the award of* ***Bachelor of Engineering*** *in* ***Electronics and Communication Engineering*** *during the academic year* ***2020-2021****. Further, the matter embodied in the dissertation has not been submitted previously by anybody for the award of any degree or diploma to any other university.*

PAULSON PREMSINGH S	**1MJ17EC090**
SRINIDHI K	**1MJ17EC133**
DIKSHITHA R	**1MJ17EC035**
ANIL KUMAR R	**1MJ17EC012**

ABSTRACT

IOT has emerged in various fields one such field is the health care monitoring system. The main purpose of our project is to give the luxury to explore improved services for patients suffering from COVID 19. It can be used to promote basic nursing care in the hospital environment as well as in your own residence by improving the quality of care and patient safety. So, remote monitoring and guidance awareness by sharing information in an authenticated manner is our main objective. The reason behind this project is to design a system for monitoring the patient's body at any time using internet connectivity. The project comprises two hardware units namely a shirt and a mask. The Shirt and Mask are both designed with a number of hardware sensors and boards which will help in the determination of various parameters such as temperature heart rate, breathing pattern and location of the individual. This whole framework is incorporated with a Man-made reasoning stage neural network model where it gathers and stores information from the individual. The computer-based intelligence framework goes about as a product stage wherewith utilization of neural organizations will play out specific undertakings and gives a safe inclination to the person. This whole framework will screen the referenced boundaries just as area and send it to the well-wisher and medical services for nonstop attention to the well being status of the patient.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned our effort with success.

We wish to express our sincere gratitude to the **Management of MVJ College of Engineering** for providing us with the state-of-the art infrastructure and facilities.

We wish to express our sincere gratitude to the **Karnataka State Council for Science & Technology**, for motivating us through their financial support.

We express our sincere gratitude to our Principal **Dr. P Mahabaleshwarappa**, MVJ College of Engineering for his continuous support and guidance.

We wish to place on record our grateful thanks to **Dr. I. Hameem Shanavas**, Head of the Department, Electronics and Communication Engineering, MVJ College of Engineering, for providing encouragement and guidance.

We consider it a privilege and honour to express our sincere gratitude to our guide **Mr.Bhanuteja G**, Assistant Professor, Department of Electronics and Communication Engineering for his valuable guidance throughout the tenure of this project work, whose support and encouragement made this work possible.

We express our sincere gratitude to **Mr.Kaustubh Anand Kandi**, Micro Electronics Design Facility, URSC, ISRO for his valuable guidance throughout the tenure of this project work.

We wish to thank the faculty of Electronics and Communication Engineering Department whose suggestions have enabled me to surpass many of the seemingly impossible hurdles.

PAULSON PREMSINGH S
SRINIDHI K
DIKSHITHA R
ANIL KUMAR R

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Coronavirus has caused much havoc around the world. As on May 20, 2021, there have been 165,565,058 confirmed cases, resulting in 3,434,004 deaths and disturbing life all around the world. The losses are exacerbating day by day. With very little research done globally on the pandemic since SARS COV-2, there is no cure and less vaccination until recent days. Controlling the growth by early testing individuals and quarantining the corona virus- affected individuals is the only implicit solution against the infectious COVID-19. However the ease to proceed with this strategy in order to control the deadly virus is not viable. There is a high risk of spread of the virus in crowded places amongst people with low immunity is still a major concern. The need for testing this virus in an early stage is still a key differentiator in most of the countries in order to reduce the pandemic curve. In many countries the vaccination drives are happening at a slow pace, this exposes the majority of the population to COVID-19. The availability of beds has also narrowed down in hospitals, and the health status of COVID-19 infected patients are not remotely monitored as expected. The need for patient monitoring which is both easy to use and also produces accurate results, is a solution that can dissolve the problem. We try to provide a solution that solves this issue which is very inexpensive for measuring the health status of a patient. We aim to produce an AI-based platform for monitoring the patient who has been infected by the coronavirus. The integration of the AI model on wearable devices solves the problem to some extent.In this regard we invite your staff and students to submit original unpublished full papers in the field of Engineering and technology by electronic submission.

As coronavirus was recognised only in the year 2019, the information available on the symptoms was very less. Considering the above situation, the previous work on this problem is highlighted where algorithms used for detecting COVID-19 such as LSTM (Long Short Term Memory) and Fuzzy rule-based predictions, are having smaller datasets. In order to overcome this, we provide a solution that evaluates the symptoms experienced by an individual, by using a relatively larger dataset. We perform algorithms like back propagation which is a simple technique and is fast in execution. This is an important aspect in resolving the COVID-19

pandemic by integrating the neural network model with the hardware that consists of a shirt along with a nebulizer. When the patient uses our proposed system during the incubation period, the data obtained from the sensors is sent to the user-friendly application for real-time monitoring. This also gives alert notifications whenever the patient's health is at risk to hospitals and guardians.

One of the main objectives of this project is to supply a new IoT based platform for observing a patient who has been affected by the COVID-19 virus. There are many medical devices that are commercially available such as respiratory monitors which are expensive and require a stable power supply to operate. We identified a critical need for respiration monitors that were inexpensive and easy to use, and thus implemented a solution that is adaptable to different environments. When examining different methods of measuring respiration, we came across several commercial respiratory monitors using different types of sensors. Some use a chest force sensor to detect force produced by chest movement, while impedance pneumograph monitors use skin electrodes to calculate transthoracic impedance. The last common method is pulse oximeter which measures O2 saturation in the blood. We aimed for our thermistor-based monitor to be comparatively cheaper, easier to use, as well as reliable. With the help of artificial intelligence, we were able to overcome the problem up to a certain extent.

AI works in a proficient way to mimic like human intelligence. It may also play a vital role in understanding and suggesting the development of a vaccine for COVID-19. This result-driven technology is used for proper screening, analyzing, prediction and tracking of current patients and likely future patients. We present a model which can work as a part of an online system as a real-time predictor to help in estimation of COVID-19 spread. This prediction model is developed using Artificial Neural Networks (ANN) to estimate the future situation by the use of geo-location, temperature , breathing pattern ,blood pressure and so on. The results of our model are confirmed by comparing them with real data and, during our research the model was accurate enough to predict the health status of the patient and intimidate the health service and well-wisher.

Our project consists of a wearable mask and shirt which has sensors and mother boards embedded in them. Where the mask collects information such as breathing and temperature changes and the shirt takes care of monitoring the blood pressure and other factors . This entire system is connected to an open source where we have fed data bases of several parameters and

different countries as well. Open source uses artificial intelligence with the help of neural networks and deep learning where the information from the data set and the device is inter related in such a way that if any abnormalities in clinical factors will result in indication total health care.

CHAPTER 2

LITERATURE SURVEY

[1] Maneesh Gupta, Hana Qudsi, "Low-Cost, Thermistor Based Respiratory Monitor"

Respiration rate could also be a customary physiological mensuration taken for monitoring a patient. Existing respiration monitors used in medical centers in area unit are unsuitable for low-resource environments. We have developed a inexpensive metastasis monitor to traumatize this issue. This device calculates a patient's respiration rate by investigation, changes in temperature through a semiconductor once the patient breathes into a mask is observed options of the device embrace associate alarm through a piezoelectric speaker that sounds once the patient stops breathing, and a low-battery indicator signal for once the battery powering the device dips below a threshold voltage. the look and implementation concerned many completely different hardware elements, as well as software package functions. Our initial model of the project accomplished our main objectives, and experimental measurements indicate that the device measures a patient's respiration rate with relative accuracy.

[2] "Thermistor based breathing sensor for circadian rhythm evaluation", Emil Jovanov, Dejan Raskovic, Rick Hormigo

One of the foremost often used ways to sense respiratory pattern is to observe flow of air using a nasal semiconductor device or a thermocouple junction device. Prolonged, minimally intrusive measuring of the respiratory pattern is very vital for polysomnography, sleeping disorders, stress observation, training program techniques, and time analysis. though most applications need solely respiratory pattern, some applications and diagnostic procedures need observation of the rhythm of modification of dominant anterior naris. throughout this paper we tend to gift our style of a differential thermistor based respiratory device for prolonged observation throughout the standard activity.

The system is supposed using a occasional power microcontroller.Lone-Star State Instruments MSP430F149 with associate degree on-chip A/D convertor for knowledge acquisition and signal process. we tend to use wireless RF link to a laptop for semipermanent knowledge acquisition and storage. Precise measuring needs decreasing zero and sensitivity errors of the measuring. we tend to discuss signal process ways, standardization and parameters accustomed characterizes respiratory patterns necessary for unit of time respiratory rhythm analysis.

[3] IoT based Patient Health Care for COVID 19 Centre Shreerang J. More, Pranav S. Patil, Jitendra M. More, Prayag S. Patil, Satish S. Marathe

COVID 19 centre watching, and management system has been planned and integration of different detector network with internet of Things. The sensors enforced will communicate with knowledge assortment and processing unit. the data assortment done by that unit will directly transferred to cloud victimisation net property at COVID nineteen centre. thus work aimed to propose COVID nineteen centre management with IoT primarily based approach to handle medical services and patient watching and treatment work flow. In the experimented model, Node MCU ESP8266 controller and temperature detector (DHT11) area unit integrated.

A system has capability to seem at and management COVID nineteen centre services and patient watching via remote association. it's evaluated with three temperature sensors connected to live temperature of patients. Mobile mostly primarily based blynk has been used for the cloud based implementation.Detector sends knowledge over blynk server thus can be seen anyplace victimisation sensible phone application. to boot, when patient get fever over regular price, associate alert was sent to authority in associate passing pace. when results, it's indicated that the developed system has effective potential to figure in pandemic situation and has technological, advantages of implemented analysis ways area unit helpful in digital health management in pandemic state of affairs. Even hospitals, COVID centers, treatment unit (ICU) is operated effectively and patient designation application supported on-line information has wide scope at intervals the realm of net of things and patient health management.

[4] Respiratory Monitor with Corrective Measure System Using Thermistor, At Mega 328 and GSM

Thermistor primarily based metabolism monitor is an integrated device that calculates the respiration rate of someone and takes stepwise measures in extreme conditions. It calculates the breathing rate by using a semiconductor device that shows variations in its output once the person breathes in or out by police investigation the corresponding temperature changes. options of this device embody LCD, buzzer, stepper motor and GSM module. LCD digital display alphanumeric display is employed to ceaselessly display the respiration rate of the person. Buzzer is utilized to sound once the person has began to breathe unpredictably and stepper motor is employed to manage the element offer.

GSM module is employed because of the last step among that a message is shipped to a doctor for immediate help. The implementation of the project includes varied hardware elements moreover as software package.

[5] Remote Monitoring of Children with Chronic Illness Using Wearable Vest R Jansi, R Amutha, S Radha

The design and development of a textile-based wearable smart vest that may be used to monitor children who are suffering from chronic illness. This system provides essential information for real-time monitoring of such children so immediate actions will be taken just in case of an emergency. Parents/Caretakers are continuously notified about the health conditions and the activities performed by their children using a smart android application. Three different sensors were fabricated on the vest that has Lilypad accelerometer, pulse oximeter, and Lilypad temperature sensors.

The activities performed by children with chronic illness need to be monitored continuously because certain activities like run if performed for an extended duration might lead to serious fatigue conditions, the Lilypad accelerometer is employed to spot the actions performed by the children. A new sparse representation-based classification algorithm was used to classify activities like walk, run, jump, hop, sit and stand. The proposed classifier achieved a high recognition rate of about 97.49%.

Pulse oximeter provided the values of oxygen saturation level (in %) and rate (in bpm). Lilypad temperature sensor was wont to monitor the temperature level of the child. The identified activity together with the values of oxygen saturation level, vital sign and temperature are indicated to the parents/caretakers employing a customized mobile application. This aids in taking immediate action just in case of abnormal conditions.

[6] AI4COVID-19: AI enabled preliminary diagnosis for COVID-19 from cough samples via an app Ali Imran, Iryna Posokhova, Haneya N. Qureshi, Usama Masood, Muhammad Sajid Riaz, Kamran Ali, Charles N. John, MD Iftikhar Hussain, Muhammad Nabeel

Pro : Results show AI4COVID-19 can distinguish among COVID-19 coughs and several types of non-COVID-19 coughs. The accuracy is promising enough to encourage a large-scale collection of labeled cough data to gauge the generalization capability of AI4COVID-19.

Con : Building on the insights from medical domain knowledge and cough data analysis.

Result : Results show AI4COVID-19 can distinguish among COVID-19 coughs and several types of non-COVID-19 coughs. The accuracy is promising enough to encourage a large-scale collection of labeled cough data to gauge the generalization capability of AI4COVID-19. AI4COVID-19 is not a clinical grade testing tool. Instead, it offers a screening tool deployable anytime, anywhere, by anyone. It can also be a clinical decision assistance tool used to channel clinical-testing and treatment to those who need it the most, thereby saving more lives.

[7] Indoor Positioning System Using Artificial Neural Network With Swarm Intelligence Chia-Hsin Cheng, (Member, IEEE), Tao-Ping Wang, Yung-Fa Huang, (Member, IEEE)

The worldwide situating framework is the most well known outside situating framework; nonetheless, it is not important to indoor conditions. In this examination, we made a half and half calculation, which applies a counterfeit neural organization to determine the issues of indoor

situating and a multitude knowledge calculation to perform the tedious undertaking of changing boundaries.

Test results exhibit that the situating exactness of the proposed half and half calculation is comparable to the thorough technique with search times that are far more limited.

The prior work done with respect to wearable devices for real time monitoring of health conditions has seen emergence of various new technology.Our prime focus has been to solve the cons of the previous work and give an appropriate solution for the above mentioned problem.

CHAPTER 3

HARDWARE USED

The hardware of this particular project consists of various sensors which are embedded inside the shirt and a mask,the shirt is embedded with sensors like Pulse oximeter sensor,GPS Module,DS18B20 temperature sensor in order to measure body temperature ,the data from these particular sensors are gathered and sent to Lilypad Arduino microcontroller.Further for the mask in order to find the breathing rate we use STM 32 microcontroller along with the thermistors which are placed inisde and outside the nebulizer.The thermistors relative voltage value helps us in determination of breathing rate.

3.1 LILYPAD ARDUINO

The LilyPad Arduino Main Board is based on the ATmega168V (the low-power version of the ATmega168) or the ATmega328V. The LilyPad Arduino was designed and developed by Leah Buechley and SparkFun Electronics..Unlike other types of Lilypad boards, this board has all ATMega328 pins open for connection purposes for wearable projects. This board comes in handy if a large number of input-output pins are required. The main reason for choosing this over other Arduino is because of its compact size and the advantage of sewing it to the fabric and similarly mounted power supplies, sensors, and actuators with conductive thread.

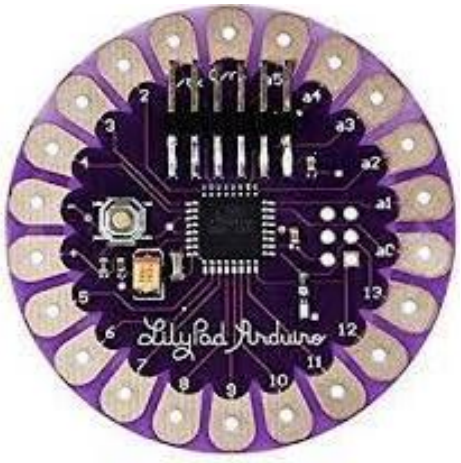

Fig. 3.1.1 Lilypad Arduino

The following are the technical specification of Lilypad Arduino:

3.1.1 Power

The LilyPad Arduino can be powered via the USB connection or with an external power supply. If an external power supply is used, it should provide between 2.7 and 5.5 volts. This can come either from an AC-to-DC adapter (wall-wart) or battery.

3.1.2 Programming

The LilyPad Arduino can be programmed with the Arduino Software (IDE). Select "LilyPad Arduino" from the Tools > Board menu (according to the microcontroller on your board). The ATmega168V or ATmega328V on the LilyPad Arduino comes preburned with bootloader that allows you to upload new code to it without the use of an external hardware programmer. The microcontroller can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header. While the holes are too small to insert pins into, you can insert male header pins into the ISP connector on your programmer and press them against the ICSP header on the board (from above). See these instructions for more information.

3.1.3 Physical Characteristics

The LilyPad Arduino can be powered via the USB connection or with an external power supply. If an external power supply is used, it should provide between 2.7 and 5.5 volts. This can come either from an AC-to-DC adapter (wall-wart) or battery.

3.2 PULSE OXIMETER - MAX30100

The MAX30100 is an integrated pulse oximetry and heart-rate monitor sensor solution. It combines two LEDs, a photodetector, optimized optics, and low-noise analog signal processing to detect pulse oximetry and heart-rate signals. The MAX30100 operates from 1.8V and 3.3V power supplies and can be powered down through software with negligible standby current, permitting the power supply to remain connected at all times. The MAX30100 is an integrated pulse oximetry and heart-rate monitor sensor solution. It combines two LEDs, a photodetector,

optimized optics, and low-noise analog signal processing to detect pulse oximetry and heart-rate signals.

The MAX30100 is fully configurable through software registers, and the digital output data is stored in a 16-deep FIFO within the device. The FIFO allows the MAX30100 to be connected to a microcontroller or microprocessor on a shared bus, where the data is not being read continuously from the device's registers.

Fig. 3.2.1 MAX30100

3.3 GPS MODULE - UBLOX NEO-6M

The NEO-6 module series is a family of stand-alone GPS receivers featuring the high performance ublox 6 positioning engine. These flexible and cost effective receivers offer numerous connectivity options in a miniature 16 x 12.2 x 2.4 mm package. Their compact architecture and power and memory options make NEO-6 modules ideal for battery operated mobile devices with very strict cost and space constraints.

GPS modules contain tiny processors and antennas that directly receive data sent by satellites through dedicated RF frequencies. From there, it'll receive a timestamp from each visible satellite, along with other pieces of data. The chosen GPS module is accurate , compact , and has a good adaptive rate where it recalculates and reports the position within fractions of seconds.

Fig. 3.3.1 Ublox Neo 6M

NEO-6 modules provide a USB version 2.0 FS (Full Speed, 12Mbit/s) interface as an alternative to the UART. The pull-up resistor on USB_DP is integrated to signal a full-speed device to the host. The VDDUSB pin supplies the USB interface. u-blox provides a Microsoft® certified USB driver for Windows XP, Windows Vista and Windows 7 operating systems.U-blox receivers support different power modes. These modes represent strategies of how to control the acquisition and tracking engines in order to achieve either the best possible performance or good performance with reduced power consumption.

3.4 DS18B20 TEMPERATURE SENSOR

The DS18B20 is a small temperature sensor with a built in 12bit ADC. It can be easily connected to an Arduino digital input. The sensor communicates over a one-wire bus and requires little in the way of additional components. The sensors have a quoted accuracy of +/-0.5 deg C in the range -10 deg C to +85 deg C. The core functionality of the DS18B20 is its direct-to- digital temperature sensor. The resolution of the temperature sensor is user-configurable to 9, 10, 11, or 12 bits, corresponding to increments of 0.5°C, 0.25°C, 0.125°C, and 0.0625°C, respectively. The default resolution at power-up is 12-bit.

The DS18B20 sensor has a smaller mean error of ± 0.85 %, so that obtained a more accurate sensor and it can be used for testing of Grashof Portable Incubator made by University of Indonesia. Temperature is one of the main parameters in a baby incubator.The resolution of this sensor ranges from 9-bits to 12-bits. But the default resolution which is used to power-up is 12-bit. This sensor gets power within a low-power inactive condition. The temperature measurement, as well as the conversion of A-to-D, can be done with a convert-T

command. The resulting temperature information can be stored within the 2-byte register in the sensor, and after that, this sensor returns to its inactive state.

Fig 3.4.1 DS18B20 Temperature Sensor

3.5 STM32F103C8T6

The ultra-low-power STM32L151xE and STM32L152xE devices incorporate the connectivity power of the universal serial bus (USB) with the high-performance ARM® Cortex®-M3 32-bit RISC core operating at a frequency of 32 MHz (33.3 DMIPS), a memory protection unit (MPU), high-speed embedded memories (Flash memory up to 512 Kbytes and RAM up to 80 Kbytes), and an extensive range of enhanced I/Os and peripherals connected to two APB buses.

Fig 3.5.1 STM32F103C8T6

STM32 is a family of 32-bit microcontroller integrated circuits by STMicroelectronics. Internally, each microcontroller consists of the processor core, static RAM, flash memory, debugging interface, and various peripherals.

3.6 THERMISTOR

NTC thermistor probes are generally used as resistance thermometers. They are extremely versatile and accurate, which makes them ideal for a wide variety of applications that measure temperature. The main reason for using this is because it is small, quick response, highly sensitive and there are even options for customisation.

Fig. 3.6.1 NTC 10k Thermistor

3.7 NODEMCU

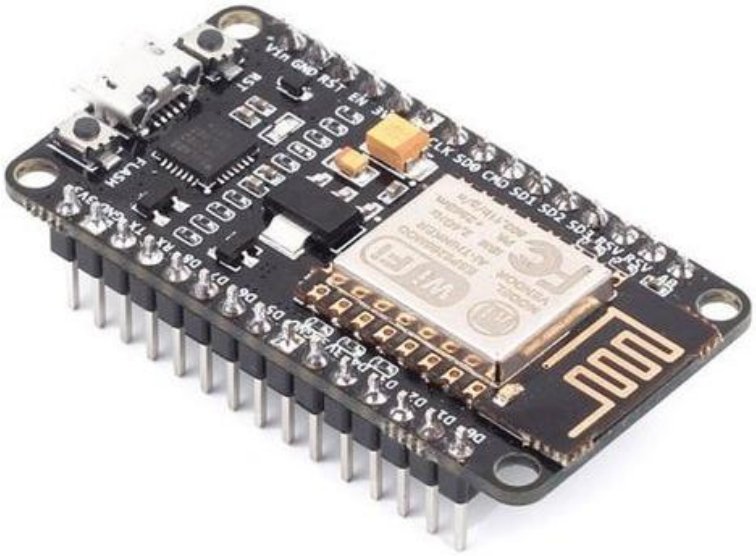

Fig. 3.7.1 NodeMCU

NodeMCU is an open source firmware for which open source prototyping board designs are available. The name "NodeMCU" combines "node" and "MCU" (micro-controller unit). The

term "NodeMCU" strictly speaking refers to the firmware rather than the associated development kits. The firmware uses the Lua scripting language. The firmware is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson[9] and SPIFFS.[10] Due to resource constraints, users need to select the modules relevant for their project and build a firmware tailored to their needs. Support for the 32-bit ESP32 has also been implemented.

The prototyping hardware typically used is a circuit board functioning as a dual in-line package (DIP) which integrates a USB controller with a smaller surface-mounted board containing the MCU and antenna. The reason for Node MCU is because via the Wifi the data is transmitted with ease and the final output is visible on the user friendly application where the breathing rate,heart rate and blood oxygen level is clearly visible in the health monitoring section of the application.

CHAPTER 4

SOFTWARE USED

The software used for programming the components of the shirt and mask is Arduino IDE. The ease with which an Arduino can obtain sensor values is one of the features that makes it very useful. ThingSpeak is used to aggregate, visualize, and analyze live data streams, which are sent to the app over the Internet. PlatformIO is another software used, which is an open-source ecosystem for IoT development.

4.1 ARDUINO IDE

Arduino IDE is an open-source software program that allows users to write and upload code within a real-time work environment. As this code will thereafter be stored within the cloud, it is often utilised by those who have been searching for an extra level of redundancy. The system is fully compatible with any Arduino software board.

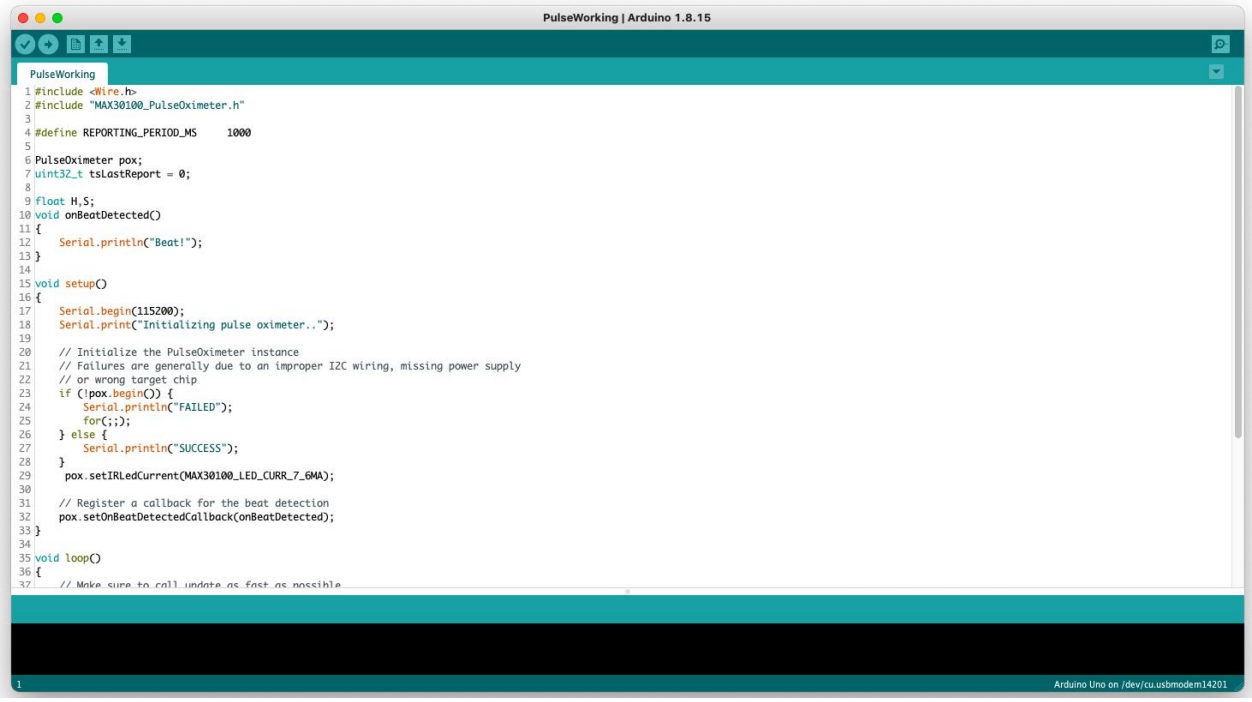

Fig. 4.1.1 Example of Code in Arduino IDE

Arduino IDE can be implemented within Windows, Mac and Linux operating systems. The majority of its components are written in JavaScript for easy editing and compiling. While its primary intention is based around writing codes, there are several other features worth noting. It has been equipped with a means to easily share any details with other project stakeholders. Users can modify internal layouts and schematics when required. There are in-depth help guides which will prove useful during the initial installation process. Tutorials are likewise available for those who might not have a substantial amount of experience with the Arduino framework.

The IDE environment is mainly distributed into three sections,

1. Menu Bar
2. Text Editor
3. Output Pane

The Arduino Integrated Development Environment (IDE) is a cross-platform application (for Windows, macOS, Linux) that is written in functions from C and C++. It is used to write and upload programs to Arduino compatible boards, but also, with the help of third-party cores, other vendor development boards.

The source code for the IDE is released under the GNU General Public License, version2. The Arduino IDE supports the languages C and C++ using special rules of code structuring. 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. By default, avrdude is used as the uploading tool to flash the user code onto official Arduino boards. Arduino IDE is a derivative of the Processing IDE, however as of version 2.0, the Processing IDE will be replaced with the Visual Studio Code-based Eclipse Theia IDE framework.

With the rising popularity of Arduino as a software platform, other vendors started to implement custom open source compilers and tools (cores) that can build and upload sketches to other microcontrollers that are not supported by Arduino's official line of microcontrollers. The bar appearing on the top is called Menu Bar that comes with five different options as follow.

- File – You can open a new window for writing the code or open an existing one. As you go to the preference section and check the compilation section, the Output Pane will show the code compilation as you click the upload button. At the end of compilation, it will show you the hex file it has generated for the recent sketch that will send to the Arduino Board for the specific task you aim to achieve.
- Edit – Used for copying and pasting the code with further modification for font.
- Sketch – For compiling and programming
- Tools – Mainly used for testing projects. The Programmer section in this panel is used for burning a bootloader to the new microcontroller.
- Help – In case you are feeling skeptical about software, complete help is available from getting started to troubleshooting.

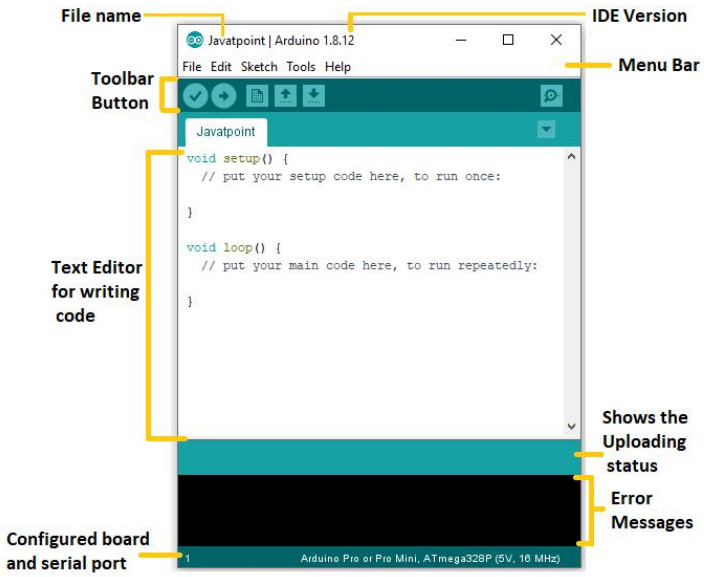

Fig. 4.1.2 Description of Arduino IDE

The button appearing on the top right corner is a Serial Monitor – A separate pop-up window that acts as an independent terminal and plays a vital role for sending and receiving the

Serial data. You can also go to the Tools panel and select Serial Monitor or pressing Ctrl+Shift+M all at once will open it instantly. The Serial Monitor will help to debug the written Sketches where you can get a hold of how your program is operating. Your Arduino Module should be connected to your computer by USB cable to activate the Serial Monitor. You need to select the baud rate of the Arduino Board you are using right now. For any Arduino Uno Baud Rate is 9600, as the basic code for blink of LED will show the output and click the Serial Monitor, the output will be visible.

4.2 C-LANGUAGE

C Language is developed by Dennis Ritchie for creating system applications that directly interact with the hardware devices such as drivers, kernels, etc. C programming is considered as the base for other programming languages, that is why it is known as mother language. It is language is considered as the mother language of all the modern programming languages because most of the compilers, JVMs, Kernels, etc. are written in C language, and most of the programming languages follow C syntax, for example, C++, Java, C#, etc.

It provides the core concepts like the array, strings, functions, file handling, etc. that are being used in many languages like C++, Java, C#, etc. A system programming language is used to create system software. C language is a system programming language because it can be used to do low-level programming (for example driver and kernel). It is generally used to create hardware devices, OS, drivers, kernels, etc. For example, Linux kernel is written in C. A Procedure is known as a function, method, routine, subroutine, etc.

A procedural language specifies a series of steps for the program to solve the problem. A structured programming language is a subset of the procedural language. Structure means to break a program into parts or blocks so that it may be easy to understand. In the C language, we break the program into parts using functions. It makes the program easier to understand and modify. C is considered as a middle-level language because it supports the feature of both low-level and high-level languages. C language program is converted into assembly code, it supports pointer arithmetic (low-level), but it is machine independent (a feature of high-level). A Low-level language is specific to one machine, i.e., machine dependent. It is machine dependent, fast to run. But it is not easy to understand. A High-Level language is not specific to one machine,

i.e., machine independent. It is easy to understand.Arduino is the hardware platform used to teach the C programming language as Arduino boards are available worldwide and contain the popular AVR microcontrollers from Atmel. Atmel Studio is used as the development environment for writing C programs for AVR microcontrollers.

4.3 PYTHON LANGUAGE

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. Python has a simple syntax similar to the English language. Python has syntax that allows developers to write programs with fewer lines than some other programming languages. Python runs on an interpreter system, meaning that code can be executed as soon as it is written.

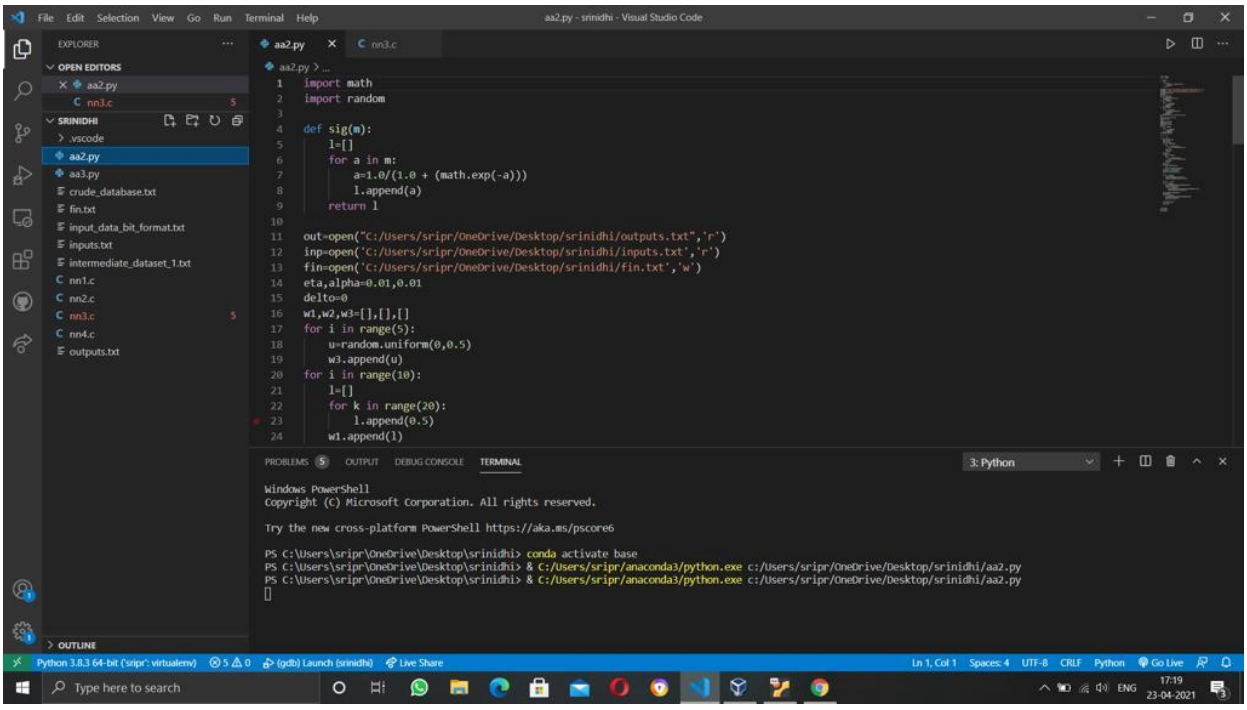

Fig. 4.3.1 Python Script for AI Model

Machine learning (ML) and deep learning (DL) are also approaches to solving problems. The difference between these techniques and a Python script is that ML and DL use training

data instead of hard-coded rules, but all of them can be used to solve problems using AI.The goal of supervised learning tasks is to make predictions for new, unseen data. To do that, assume that this unseen data follows a probability distribution similar to the distribution of the training dataset. If in the future this distribution changes, then you need to train the model again using the new training dataset.

4.4 THINGSPEAK IOT PLATFORM

ThingSpeak is IoT Cloud platform where you can send sensor data to the cloud. We can also analyze and visualize your data with MATLAB or other software, including making your own applications.

The ThingSpeak service is operated by MathWorks. In order to sign up for ThingSpeak, you must create a new MathWorks Account or log in to your existing MathWorks Account. ThingSpeak is free for small non-commercial projects.

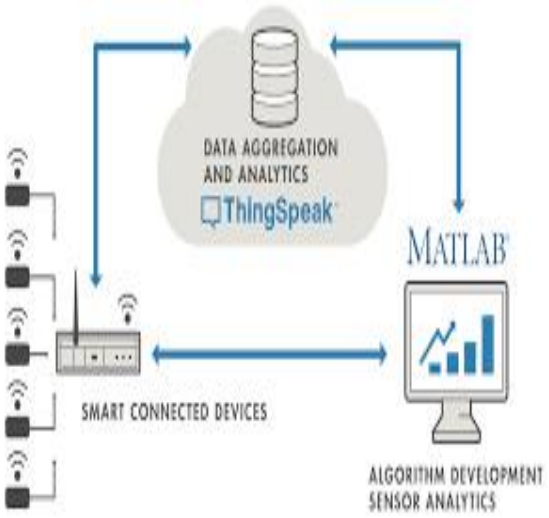

Fig. 4.4.1 Thingspeak Flowchart

ThingSpeak includes a Web Service (REST API) that lets you collect and store sensor data in the cloud and develop Internet of Things applications. It works with Arduino, Raspberry Pi and MATLAB (premade libraries and APIs exists) But it should work with all kind of Programming Languages, since it uses a REST API and HTTP.

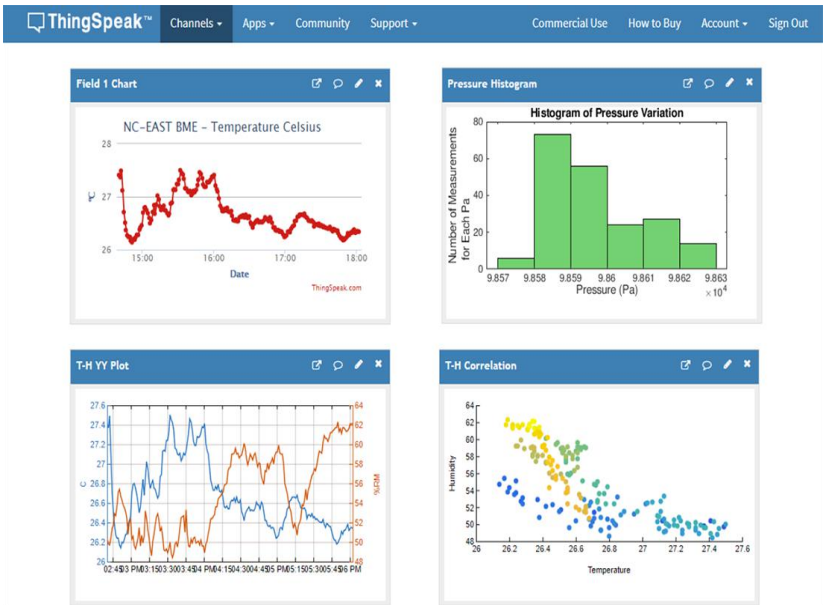

Fig. 4.4.2 Thingspeak Visualizations

ThingSpeak is an IoT analytics platform service that allows you to aggregate, visualize, and analyze live data streams in the cloud. You can send data to ThingSpeak from your devices, create instant visualizations of live data, and send alerts using web services like Twitter and Twilio. With MATLAB analytics inside ThingSpeak, we can write and execute MATLAB code to perform preprocessing, visualizations, and analyses. ThingSpeak enables engineers and scientists to prototype and build IoT systems without setting up servers or developing web software.

4.5 PLATFORM IO IDE

The Platform IO ecosystem has a decentralized architecture, allowing development for a range of development platforms. A development platform (or just “platform” for short) is usually a particular microcontroller or processor architecture that Platform IO projects can be compiled to run on. Each of the three supported host systems Mac OS X, Linux and Windows support compiling for all platforms. Some platforms are also supported under ARM Linux hosts such as Raspberry Pi.

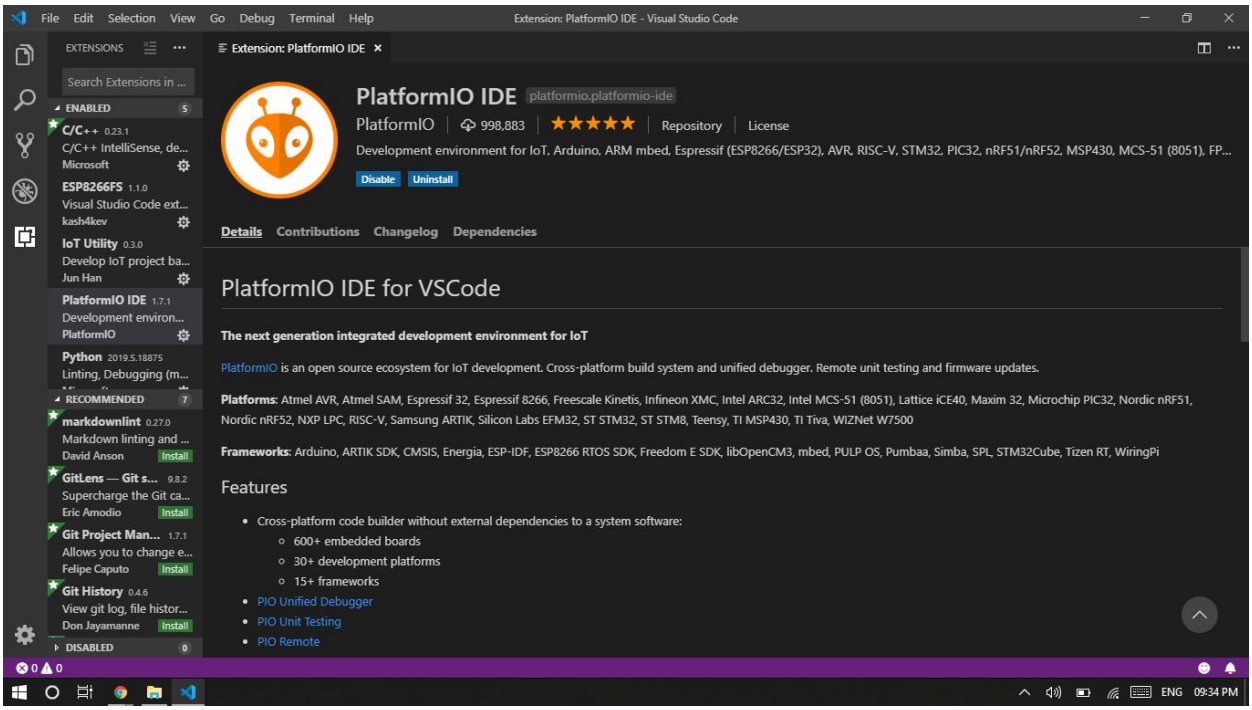

Fig. 4.5.1 Platform IO IDE

For each development platform, Platform IO defines:

1) The Platform IO Build System build scripts for the supported frameworks and SDKs.
2) Pre-configured presets for embedded circuit boards.
3) Pre-compiled tool chains and related tools for the architechture.

CHAPTER 5

PROPOSED SYSTEM

The proposed system consists of a shirt and a mask. Shirt comprises of a temperature sensor DS18B20 sensor which measures the temperature of patient. The MAX30100 pulse oximeter sensor will retrieve the pulse beat data along and the SpO2 level i.e, oxygen level. We aim to interface the sensors and provide the necessary output. The NTC Thermistor sensor will retrieve the temperature data. Then this data will be sent to the Arduino Lilypad from where the entire data will be sent to the app using NodeMCU. Now with respect to the mask, it plays an important role in measuring the breathing rate and the pattern of breathing as difficulty in breathing is one of the most common signs of COVID-19. The STM32 microcontroller calculates the breathing rate with a thermistor and is fixed with a few driver circuits inside the mask and the information is being sent to the software application. The hospital staff and the guardian are alerted and could track the location and make sure the patient reaches the hospital if all the 4 parameters mentioned would cross the threshold value.

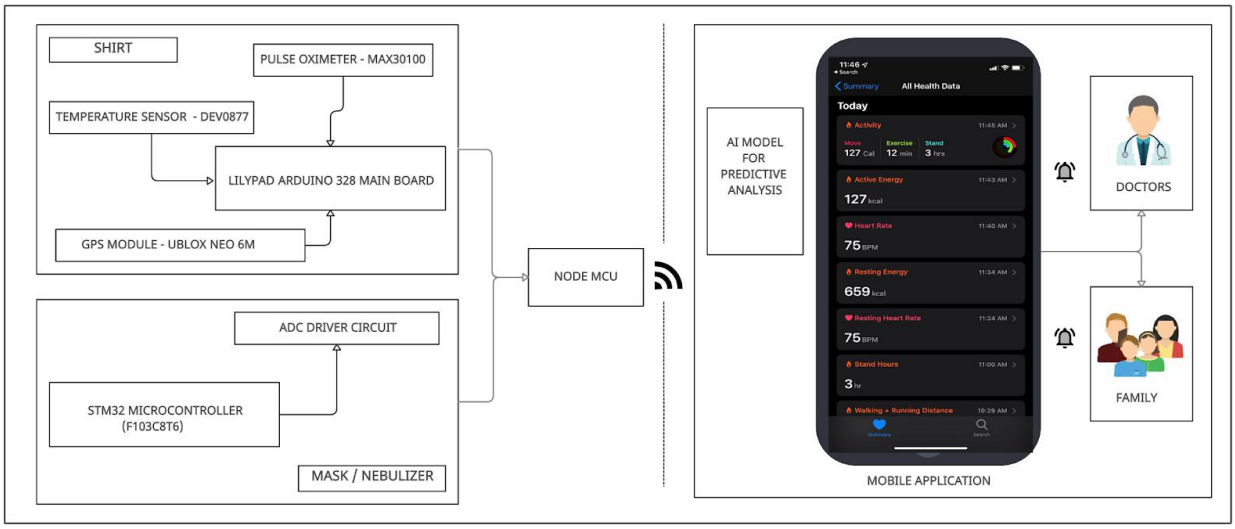

Fig. 5.1 Proposed System

The above figure illustrates the block diagram of project. The hardware part of the project has two parts to it one is the shirt and other is the mask. From the shirt and mask the sensor data such as temperature, heart rate, oxygen level and breathing pattern is collected if the individual feels normal the whole process runs in a loop until there is any change in threshold level of the

clinical aspects. When abnormality is observed information such as location and other clinical aspects to the health care services and well wishers for them to act accordingly.

With the ongoing swath of the COVID-19 pandemic situation, health care monitoring has become an essential part of human lives. A few health monitoring systems including wearable devices like watches, waist belt, shirt, mask, etc. are effective when used for remote applications. One such confirmation for real-time health monitoring is the use of a shirt that is embedded with sensors. The block diagram of shirt and mask is shown in Fig. 5.2.

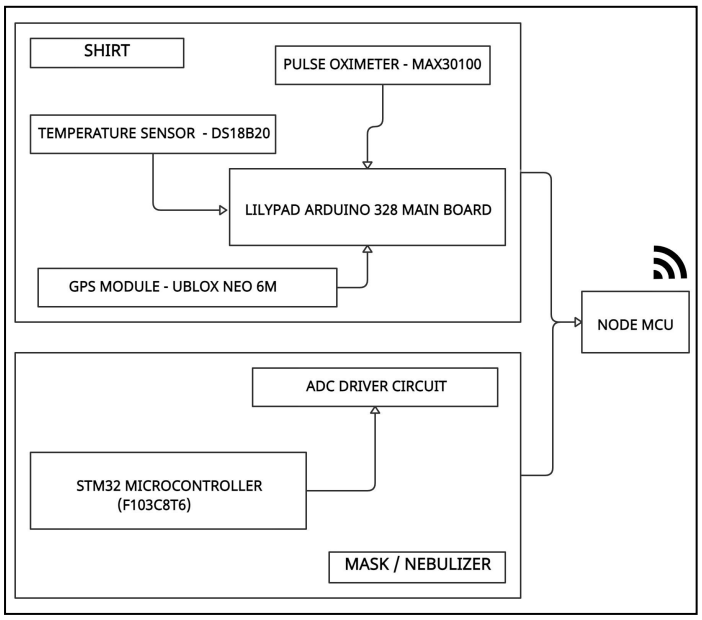

Fig. 5.2 Block diagram Of the Shirt and Mask

This gives us accurate results of patients when they are in their quarantine period and get alerts when the sensor values reach a threshold level indicating at-risk condition.The shirt being one of the main component in this project, when the patient is is showing symptoms of COVID ,we aim to reduce the risk situation by alarming about the health conditions of the patient and the parameters measured for a patient includes heart rate level,next is the blood oxygen saturation level, the shirt is also embedded with GPS module which is Ublox Neo 6M the sensor shares real time location of the patient and minimizes the risk,when these parameters cross threshold value ,the guardian is notified. The Lilypad Arduino gathers the data from these sensors and processes the data and finally the output is visualized on the user friendly application

where the nearby hospitals location is seen on the application. The below Table 5.1 mentions the threshold values of the sensors embedded inside the shirt.

Table 5.1 Threshold Values for Shirt

S. No.	Ideal versus Risk Values for Alert Systems		
	Parameters	***Ideal***	***At-Risk***
1.	SpO2 Level (%)	94 -100	< 91
2.	Heart Rate (bpm)	70-100	>110
3.	Temperature (in °F)	97 - 99	>100

The above table mentions minimum value the patient must have in order to stay safe when the face COVID-19 symptoms. As Blood Oxygen Saturation level is one of most considered parameter for COVID-19 when the saturation level drops below 90% the patient is at risk and similarly with respect to the heart rate goes above 110bpm the patient is termed “at-risk” situation.

The sensors embedded with the shirt are combined and the data collected from these sensors are transferred via the WiFi module and visualized, with this another parameter is included with the help of a mask which is the breathing rate.As breathing rate is one of the common criteria in order to measure COVID-19.

The use of face masks and shields has reduced the risk of virus transmission. Adding onto this primary solution, our focus is to develop a smart nebulizer, a which can be worn by the patient during their quarantine period. There are quite a few different types of airborne threats pathogens being only one of them but that happens to be something that now the whole world is very tuned into and has become kind of normal. So the last year has been a really interesting ride.This mask is helpful in order to continuously monitor their breathing pattern and temperature using the NTC thermistor that is attached inside the nebulizer.

From the very beginning, the need for a very robust approach to protect from all three airborne threat types. Those are pathogens, like we're dealing with now and the current pandemic. There are ecological airborne risks those would be things like wildfires, or dust storms, things of

that nature. And then there's anthropogenic, which is man-made or human-made air pollution like carbon combustion from factory emissions and things like that. So, from the very outset, we were really looking at covering all of those. From a protection standpoint, the mask plays an important role. Considering the above mentioned situation the need to measuring breathing rate has become and important aspect. The mask consists of STM32 Microcontroller along with 2 NTC 10k Thermistors where one thermistor is placed inside the nasal area and other thermistor is placed outside the nasal area where the relative voltages between two thermistors are compared and breathing rate is obtained.

The below table represents the threshold values of breathing rate in a patient when the breathing rate drops below 20 breaths per min the patient is immediately rushed to the hospital.

Table 5.2 Threshold Values for Mask

S. No.	Ideal versus Risk Values for Alert Systems		
	Parameters	***Ideal***	***At-Risk***
1.	Breathing Rate (respiration per min)	12 - 20	< 10

The sensors embedded inside the mask mainly include 2 thermistors. The first that can be tracked, is breathing health and so we can give patients a real-time understanding of their breath count, and the breath cycles, so the rhythm in which they're breathing, and the volume of air that they're moving. And measuring breathing been able to be done before in the patient context, and so it allows us to start to understand the role breathing plays in our everyday lives as well as our active lives. We try to provide a kind of a local air quality indexing. In Asia, checking your air quality is akin to checking the weather for the day. Understanding the ambient air quality around you really does start to figure into your everyday understanding of what's healthy and what's not. Using that metric, we're able to tell you the delta between the air that you're breathing in, and your ambient air. The breathing rate from the thermistors is visualized on the real time application in breaths per minute and alarmed if the situation goes out of hand.

The parameters from both shirt and mask are combined together and transferred via the internet and for better care of the patient the guardian is notified if the patients are facing

difficulty in breathing and within no time the patient can get immediate care and rushed to the hospital whose location is visible on the user friendly application.

As COVID 19 symptoms determine an important role in decision making whether or not the person is suffering from COVID-19, the ANN predictive model helps in overcoming this task.

The predictive model that consists of datasets from around the world where the symptoms vary as mutations of corona increase. The block diagram of ANN model and app is shown in Fig. 5.3. This ANN model which gives way for predictive analysis.

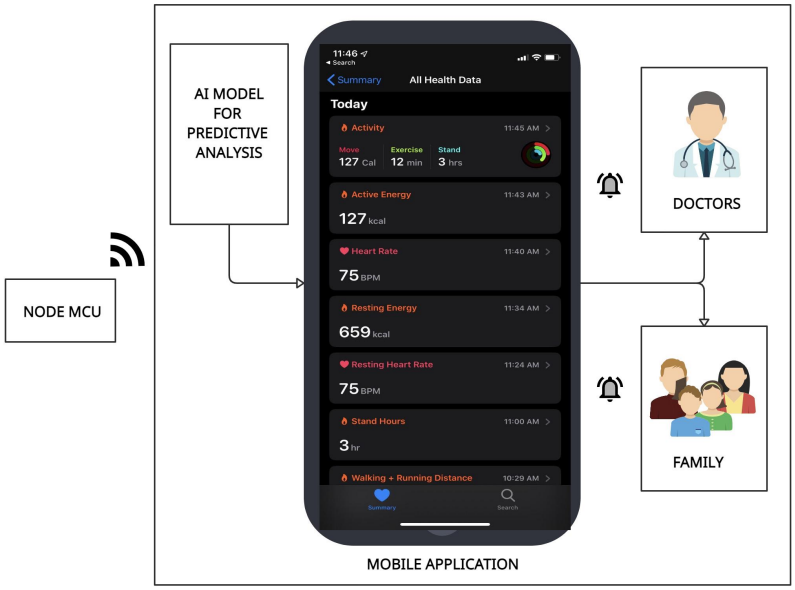

Fig 5.3 Block Diagram of ANN with APP

Artificial neural networks (ANNs), usually called neural networks (NNs), are computing systems inspired by the biological neural networks that constitute human brains.An ANN is based on a collection of connected units or nodes called artificial neurons, which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit a signal to other neurons. An artificial neuron that receives a signal then processes it and can signal neurons connected to it. The "signal" at a connection is a real number, and the output of each neuron is computed by some non-linear function of the sum of its inputs. The connections are called edges. Neurons and edges typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Neurons may have a threshold such that a signal is sent only if the aggregate signal crosses that threshold. Typically, neurons are aggregated into layers. Different layers may perform different

transformations on their inputs. Signals travel from the first layer (the input layer), to the last layer (the output layer), possibly after traversing the layers multiple times as shown in Fig. 5.3.

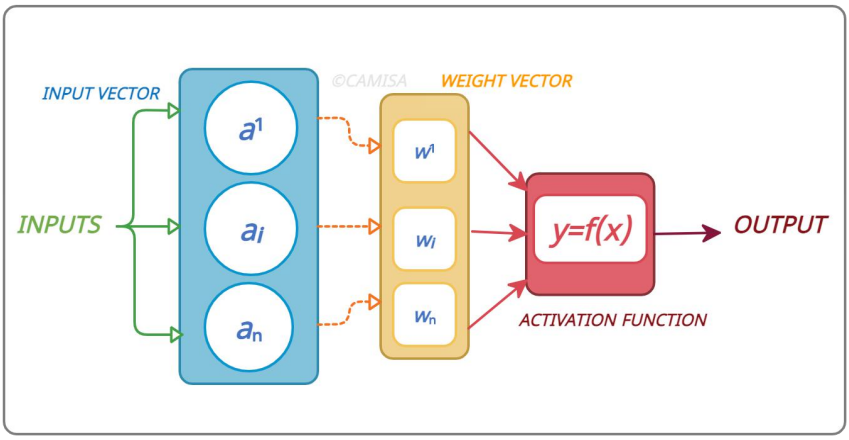

Fig 5.4 Neural Network Architecture

An ANN has hundreds, or thousands of artificial neurons called processing units, which are interconnected by nodes. These processing units are made up of input and output units. The input units receive various forms and structures of information based on an internal weighting system, and the neural network attempts to learn about the information presented to produce one output report. Just like humans need rules and guidelines to come up with a result or output, ANNs also use a set of learning rules called backpropagation, an abbreviation for backward propagation of error, to perfect their output results. An ANN initially goes through a training phase where it learns to recognize patterns in data, whether visually, aurally, or textually. During this supervised phase, the network compares its actual output produced with what it was meant to produce the desired output. The difference between both outcomes is adjusted using backpropagation. This means that the network works backward, going from the output unit to the input units to adjust the weight of its connections between the units until the difference between the actual and desired outcome produces the lowest possible error. During the training and supervisory stage, the ANN is taught what to look for and what its output should be, using yes/no question types with binary numbers.

A neural network is a system that learns how to make predictions by following these steps:

1. Taking the input data.
2. Making a prediction.

3. Comparing the prediction to the desired output.
4. Adjusting its internal state to predict correctly the next time.

Initially, a mathematical model of all the neurons and the interconnections between them are created. Then the inputs are provided to the network, and the neurons convolute with the inputs systematically, to generate an output, which is fed into the next level of neurons, as shown in Fig. 5.3. The process repeats itself until the output is reached, in the predefined hidden layers.

The ANN model can also be used to estimate the confirmation of the coronavirus through the available datasets. In order to limit the propagation speed and the propagation power of the coronavirus, it is important to predict and identify the infected patients and isolate them. The flowchart of neural network is explained in Fig. 5.1.1

Below are the steps carried out for the Predictive AI model using C language and Python.

5.1 Collection of data from various sources

Since the virus exhibits different symptoms in different countries due to the mutations, the dataset is obtained from a standardized database. The network is tuned by training it on this dataset built on health conditions of 300,000 people from different geographic locations.

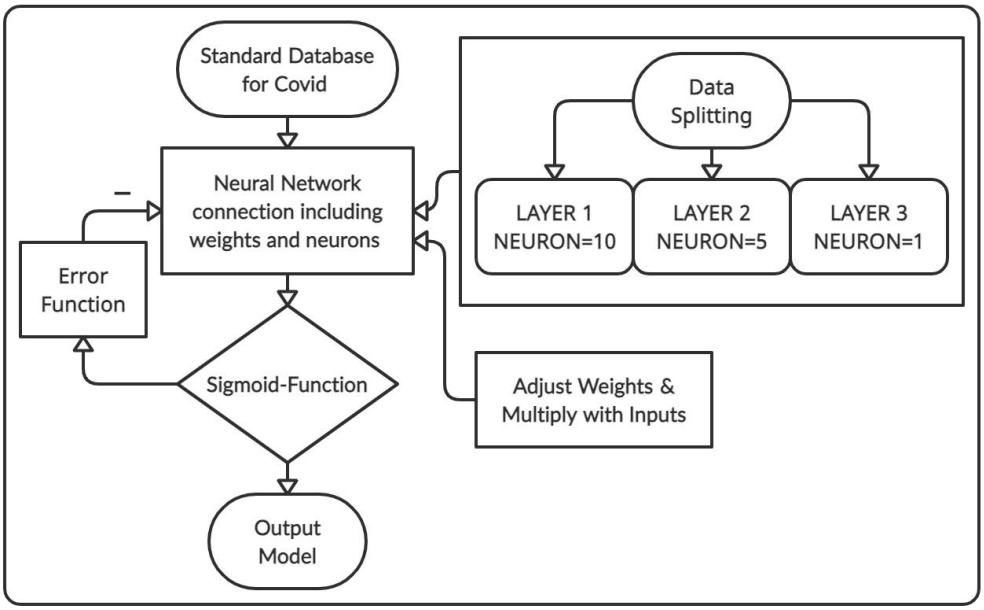

Fig 5.1.1 Flowchart of ANN

5.2 Database formatting and creation of labelled data

A data set is a collection of numbers or values that relate to a particular subject. The data obtained is formatted for inputs with 20 different parameters as shown in Table. 3, where each bit in the every input corresponds to a symptom experienced by an individual like cold, fever, cough, breathlessness, nausea, headache, conjestion, diarrhoea, and so on. Thereby giving us maximum of 20 different parameters to predict the condition of a single user show in result section. The parameters are set giving the input layer size over 200,000 inputs. We labelled the data as per the requirement and get the final raw data for further tuning.

Table 5.2.1 Parameters considered for 20 inputs

S. No.	Input Parameters
1.	Fever
2.	Cold
3.	Dry Cough
4.	Difficulty in Breathing
5.	Sore Throat
6.	Pains
7.	Nasal Congestion
8.	Runny Nose
9.	Diarrhea
10.	Body Ache
11.	Tiredness
12.	Lose in Smell and Taste
13.	Pink Eyes
14.	Other Gastrointestinal Symptoms
15.	Hearing Impairment
16.	Asthma
17.	Diabetes

S. No.	Input Parameters
18.	Hypertension
19.	Chronic Illness
20.	Heart Disease

A validation dataset is a dataset of examples used to tune the hyperparameters (i.e. the architecture) of a classifier. It is sometimes also called the development set or the "dev set". An example of a hyperparameter for artificial neural networks includes the number of hidden units in each layer.

5.3 Database formatting to make it compatible with AI

Each data format represents how the input data is represented in memory. Interchanging between various data formats and chosing the correct format is a major optimization technique. For example, TensorFlow is built around NHWC format while MKLDNN is built around NCHW data format.The raw input data is made compatible for AI by preprocessing. The raw data is processed and fed to the 3 layer feed forward network, where forward propagation is performed below as in Fig. 5.3.1

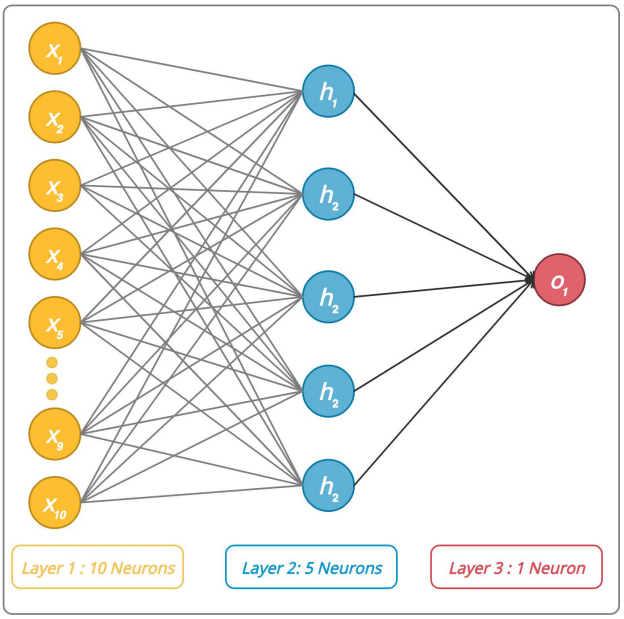

Fig. 5.3.1 1-Layer Feed Forward Network

5.4 Theoretical Modeling of ANN

Vectors, layers, and linear regression are some of the building blocks of neural. Networks. The data is stored as vectors, and. Each layer transforms the data that is acquired from the previous layer. Each layer can be thought of as a feature engineering step because each layer extracts some representation of the data that came previously. A major factor concerning Neural Network layers is that the same computations can extract information from any kind of data. This means that it does not matter if you're using image data or text data. The process to extract meaningful information and train the deep learning model is the same for both scenarios. In the project carried forward we have tried to implement a model that predicts the output value with the 3 layer feed forward network.In order to train the model, it is appropriate to write the weights between unit in layer 1 and unit in layer 2 as an array weight I[j]. Thus, to get the output of unit in layer 2.

For performing the theoretical analysis, the arrays are set up by assigning random weights. This is a 2D array matrix where one column represents a single node, thus obtaining a 10x20 matrix. This final matrix obtained is used to train the model. Equations (1) is the input vector fed into the network. Equations (2) and (4) calculate the weighted sum of inputs of the respective layers which is subsequently used to measure the activation function given by (3) and (5).

$$x_i = a_i^{(1)}, i \in 1,2,3 \quad (1)$$

$$z^{(2)} = W^{(1)}x + b^{(1)} \quad (2)$$

$$a^{(2)} = f(z^{(2)}) \quad (3)$$

$$z^{(3)} = W^{(2)}a^{(2)} + b^{(2)} \quad (4)$$

$$a^{(3)} = f(z^{(3)}) \quad (5)$$

5.4.1 Forward Propagation

A feed forward neural network is a biologically inspired classification algorithm. It consists of several simple neuron-like processing units, organized in layers. Every unit in a layer relates to all the units in the previous layer. These connections are not all equal: each connection may have a different strength or weight. The weights on these connections encode the knowledge of a network. Often the units in a neural network are also called nodes.

Data enters at the inputs and passes through the network, layer by layer, until it arrives at the outputs. During normal operation, that is when it acts as a classifier, there is no feedback between layers. Therefore, they are called feed forward neural networks.

At each neuron in a hidden or output layer, the processing happens in two steps:

- Pre-activation: it is a weighted sum of inputs i.e., the linear transformation of Weights w.r.t to inputs available. Based on this aggregated sum and activation function the Neuron decides whether to pass this information further or not.

- Activation: the calculated weighted sum of inputs is passed to the activation function. An activation function is a mathematical function which adds non-linearity to the Network. There are four commonly used and popular activation functions sigmoid, Hyperbolic tangent. A sigmoid function is a mathematical function having a characteristic "S"-shaped curve or sigmoid curve. A common example of a sigmoid function is the logistic function shown in the first figure and defined by the equation (6).

$$\sigma(x) = \frac{1}{1+e^{-x}} \quad (6)$$

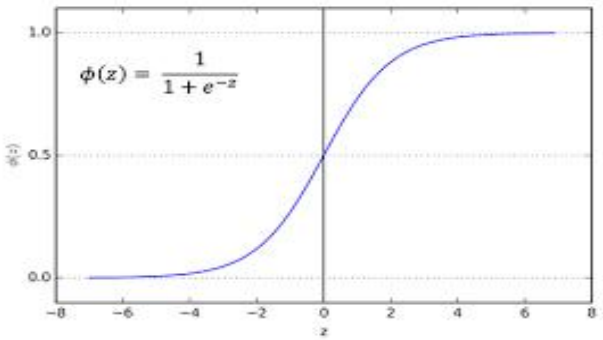

Fig 5.4.1.1 Sigmoid Function Graph

Later on updating the associated weights and comparing the output after multiple iterations, on how well the neural network performed as to the given training sample. This is referred to as the cost function, given by (7) and illustrated in Fig. 5.4.1.2

$$\textit{Cost Function (J)} \quad = \frac{1}{n}\Sigma_{i=0}^{n}(y^{(i)} - (mx^{(i)} + b))^2 \qquad (7)$$

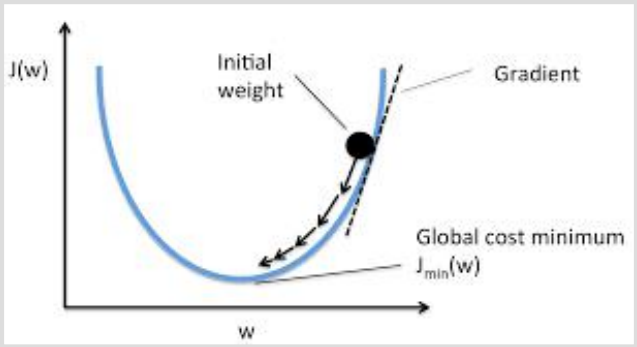

Fig 5.4.1.2 Cost Function

5.4.2 Back Propagation

A neural network propagates the signal of the input data forward through its parameters towards the moment of decision, and then back propagation information about the error, in reverse through the network, so that it can alter the parameters.The back propagation algorithm works by computing the gradient of the loss function with respect to each weight by the chain rule, computing the gradient one layer at a time, iterating backward from the last layer to avoid redundant calculations of intermediate terms in the chain rule; this is an example of dynamic .

Back propagation (backward propagation) is an important mathematical tool for improving the accuracy of predictions in data mining and machine learning. Artificial neural networks use back propagation as a learning algorithm to compute a gradient descent with respect to weights. This happens step by step:

1. The network makes a guess about data, using its parameters

2. The network's is measured with a loss function

3. The error is back propagated to adjust the wrong-headed parameters

We can compare a neural network to a large piece of artillery that is attempting to strike a distant object with a shell. When the neural network makes a guess about an instance of data, it fires, a cloud of dust rises on the horizon, and the gunner tries to make out where the shell struck, and how far it was from the target. That distance from the target is the measure of error. The measure of error is then applied to the angle and direction of the gun (parameters).

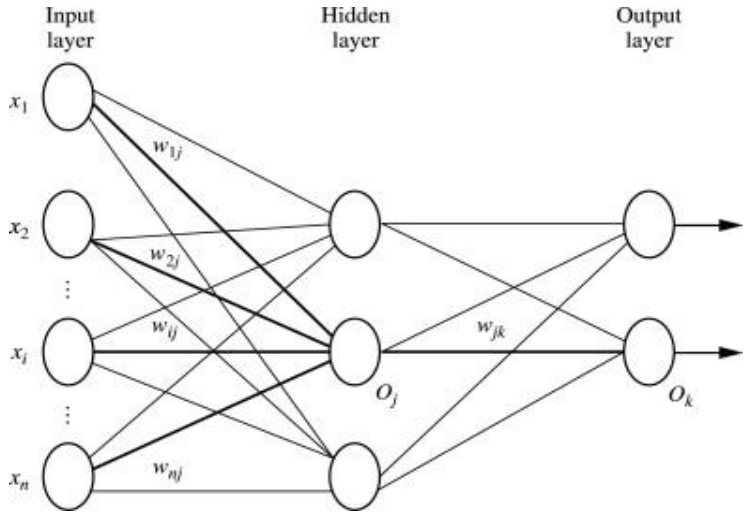

Fig 5.4.2.1 Back Propagation of 3 Layer Network

The back propagation algorithm checks for the least value of the error function in the weights vector. On back propagating the error function to the hidden layer, the algorithm tunes the network in accordance with the error rate. This calculates the gradient of the obtained loss function with respect to weights and thus minimizes the error.

5.4.3 Gradient Descent

The gradient descent for a given algorithm is an optimised algorithm used to find the parameters that reduce the cost function as shown below in equation (8).

$$Gradient\ Descent = \frac{\partial J}{\partial w} \quad (8)$$

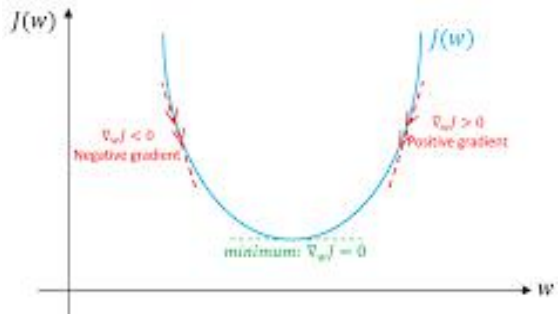

Fig 5.4.3.1 Gradient Curve

Where E is the error and w are the weight. If the loss increases with an increase in weight so Gradient will be positive, basically at the point C. If Loss decreases with an increase in weight so gradient will be negative. So, always the negative of the Gradient shows the directions along which the weights should be moved in.

A gradient is a slope whose angle can be measured. Like all slopes, it can be expressed as a relationship between two variables: “y over x”, or rise over run. In this case, the y is the error Produced by the neural network, and x is the parameter of the neural network. The parameter has a relationship to the error, and by changing the parameter, we can increase or decrease The error. So, the gradient tells us the change we can expect in y with regard to x. To obtain this information, we must use differential calculus, which enables us to measure.

Instantaneous rates of change, which in this case is the tangent of a changing slope expressing the relationship of the parameter to the neural network’s error. As the parameter changes, the Error changes, and we want to move both variables in the direction of less error. A neural network has many parameters, so we aim to measure the partial derivatives of each parameter’s contribution to the total change in error. What’s more, neural networks have parameters that process the input data sequentially, one After another. Therefore, back propagation establishes the relationship between the neural Network’s error and the parameters of the net’s last layer; then it establishes the relationship Between the parameters of the neural net’s last layer those the parameters of the second-to-Last layer, and so forth, in an application of the chain rule of calculus.Back propagation in artificial neural networks has echoes in the functioning of biological neurons, which respond to rewards such as dopamine to reinforce How they fire; i.e., how they interpret the world. Dopaminergic behavior tends to strengthen The ties

between the neurons involved, and helps those ties last longer. After the derivative of the error with respect to the weight is obtained,it is referred to as Gradient.

5.5 Parameter tuning

Using the obtained values, we perform parameter tuning in order to find the error functions. It is observed that the error functions are in a decreasing trend as seen in Fig.5.5.1. The obtained outputs of the error function are compared as the values of various parameters such as learning rates, i.e, alpha are varied (9).

$$dx = alpha * |\frac{\partial J}{\partial w}| \qquad (9)$$

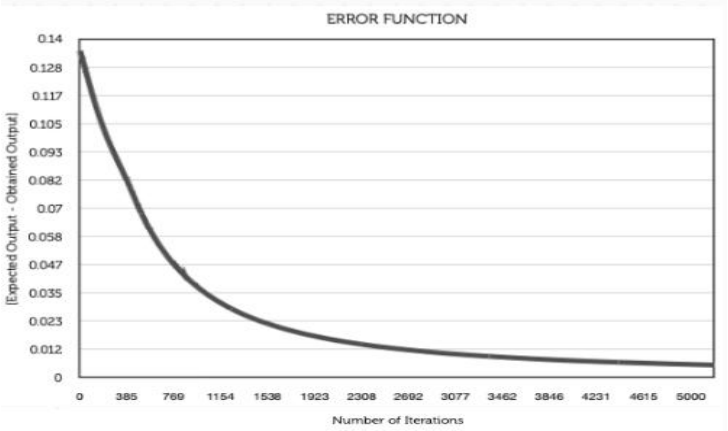

Fig 5.5.1 Error Plot of the trained model

5.6 Training the model and validation

The model and algorithms are trained on 60% of the dataset where the dataset is split. The remaining 40% of the dataset is used for validation, to prove the predictability of the algorithm.

Multiple iterations and parameter tuning to obtain the least error. This is the final step of training the model. The output thus obtained is an error function that is obtained after every sample is trained. The Fig. 5.6.1 illustrates the output file generated, showing 1-bit binary output for every combination of input parameters. Zeroes and ones represent absence and presence of COVID.

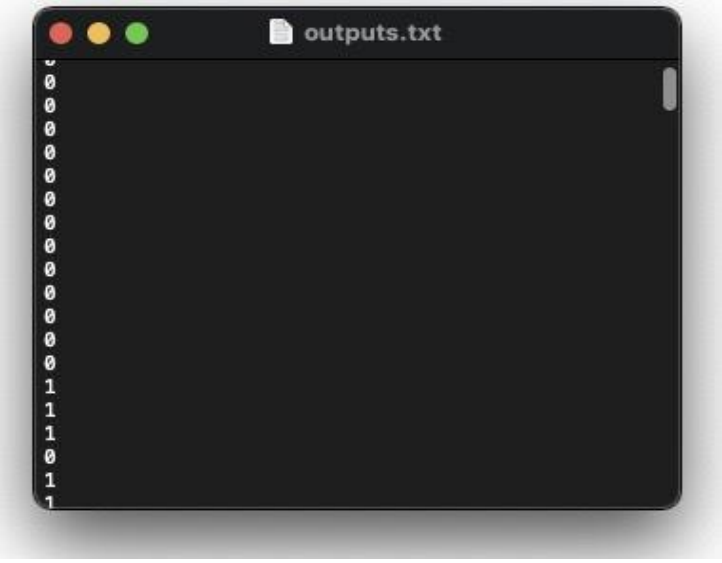

Fig 5.6.1 Output file generated showing the presence of COVID-19

Neural network gives us a total of (20x10) + (5x1) = 205 trainable parameters in the model. This is almost similar to a 205th-degree equation since it is highly impossible to fit in such a high degree equation using mathematical methods or any conventional methods. Neural networks giving a very efficient solution to this problem is a good tool to model such diverse datasets.

To conclude this model we integrate the output obtained from the neural network, which gives us information about whether a person has COVID-19 or not based on the symptoms.

The predictive ANN model which is carried out by these 7 steps is further deployed on user friendly application where the 2 main components of the applications are

1. Self Assessment based Questionnaire

2. Real Time health Monitoring

The model deployed on the self assessment questionnaire asks the patient about their symptoms and generates 1-bit binary output.The data from from sensors and the neural network model is visible on the application.The application shares real time data and provides immediate care to the patient.

Chronic diseases impose heavy burden and costs on the health industry in many countries. Suitable health procedures, management, and prevention of disease by continuous monitoring through modern technologies can lead to a decrease in health costs and improve people empowerment. Applying remote medical diagnosis and monitoring system based on mobile health systems can help significantly reduce health care costs and correct performance management particularly in chronic disease management. In this chapter, Health opportunities in patient monitoring with the introduction of various systems specifically in chronic disease are expressed. Also Health challenges in patient monitoring in general and specific aspects are identified.

Some of the general challenges include threats to confidentiality and privacy, and lack of information communication technology (ICT), and mobile infrastructure. In specific aspect, some difficulties include lack of system interoperability with electronic health records and other IT tools, decrease in face-to-face communication between doctor and patient, ill-functioning of system that leads to medical errors and negative effects on care outcomes, patients, and personnel, and factors related to the telecommunication industry include reliability and sudden interruptions of telecommunication networks.The health monitoring app combines the output from sensors and modules, then visualizes the data received. After the predictive model is trained, it is deployed on the application. Although Health technology has a key role in health care systems, yet its uptake has faced with general and specific challenges. Some problems in general dimension include organizational challenges like organizational culture, support of high-level management, technological barriers such as lack of ICT and mobile infrastructure human challenges.

The app consists of two main section:

5.7 Self Assessment

It has a questionnaire filled using toggle switches that record users' responses, which correspond to the COVID-19 symptoms experienced by the user. Based on which the app predicts an output of whether the patient is COVID-19 positive or not.

5.8 Real-Time Health Monitoring

The health parameters obtained from each of the sensors are visualized on the app using the Thingspeak IoT Platform. The app alerts the guardians and shares the real-time location of the patient when the patient is at risk.

Sensor technology and AI in the form of wearables gives a better scope in order to minimize the risk of COVID 19 and monitor the affected ones. With the help of AI, the predictive analysis is done which helps an individual confirm the presence of the virus. This framework has altogether brought the need of wearables as one of primary solution in order to reduce the risk.

CHAPTER 6

RESULTS

Sensor technology and AI in the form of wearables gives a better scope in order to minimize the risk of COVID 19 and monitor the affected ones. The hardware of our system comprises of a shirt and a nebulizer mask which is used by a patient in quarantine period.

Shirt comprises of a temperature sensor DS18B20 sensor which measures the temperature of patient. The MAX30100 pulse oximeter sensor will retrieve the heart rate along and the SpO2 level. The GPS Module is also used to get location of patient. The prototype of shirt circuit is shown in Fig. 6.1.

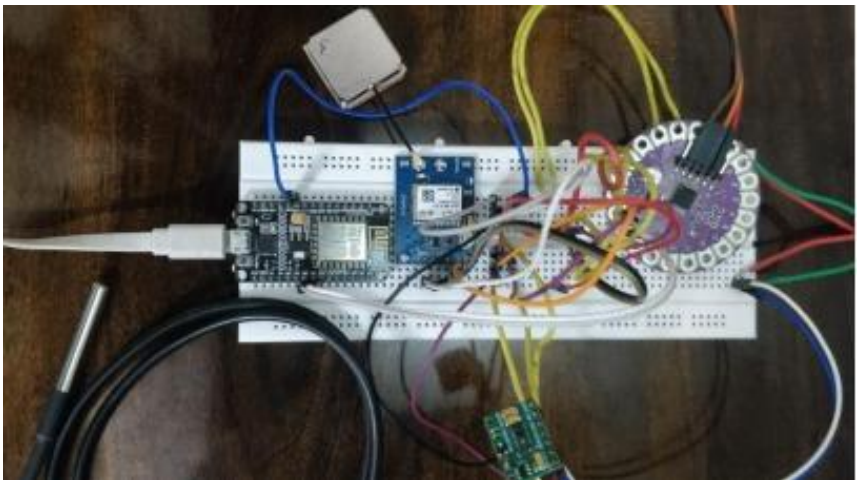

Fig. 6.1 Prototype Circuit of Shirt

We aim to interface the sensors and provide the necessary output. Now with respect to the mask, it plays an important role in measuring the breathing rate and the pattern of breathing as difficulty in breathing is one of the most common signs of COVID-19. The STM32 microcontroller calculates the breathing rate with a thermistor and is fixed with a few driver circuits inside the mask. The NTC Thermistor sensor will retrieve the temperature data and this is used in a algorithm to calculate the breathing rate.

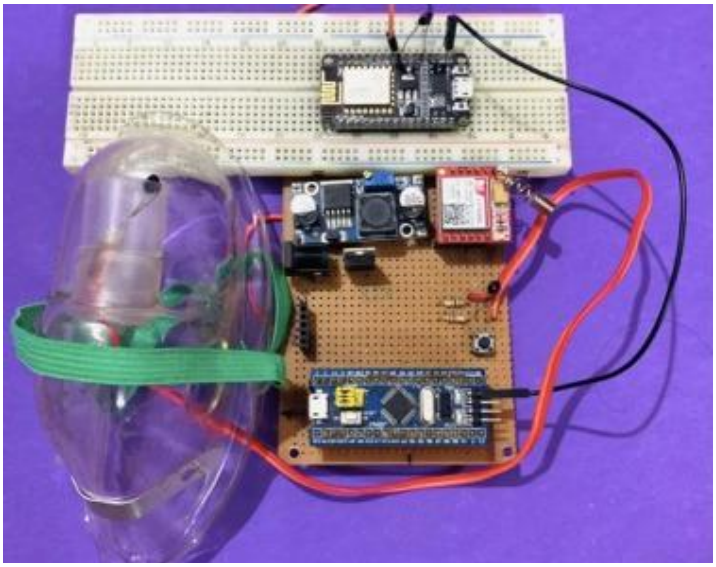

Fig. 6.2 Prototype Circuit of Mask

Then this data will be sent to the Arduino Lilypad from where the entire data will be sent to the app using NodeMCU. and the information is being sent to the software application. The hospital staff and the guardian are alerted and could track the location and make sure the patient reaches the hospital if all the 4 parameters mentioned would cross the threshold value.

Fig. 6.3 Hardware integration on Shirt

The sensors are embedded on the shirt, where the pulse oximeter is placed towards the end of left sleeve. It is placed in this position because the readings from pulse oximeter taken from the fingertips are easy and are used to get accurate values. Moving up the sleeve, the temperature sensor is placed next to the arm. This gets the reading when it is in contact with human arm. On top of the left shoulder the GPS module and the antenna is placed. It is placed in

upward direction to get accurate values. The Lilypad Arduino is sewed on the shirt. This is shown in Fig. 6.3.

The STM32 micro-controller is placed outside the mask. The 2 thermistors employed to measure the breathing rate is kept in a configuration such that one thermistor is placed outside the mask and the other thermistor is placed inside the mask. This setup is depicted in Fig. 6.4. Both the STM32 and Lilypad are then connected to the NodeMCU.

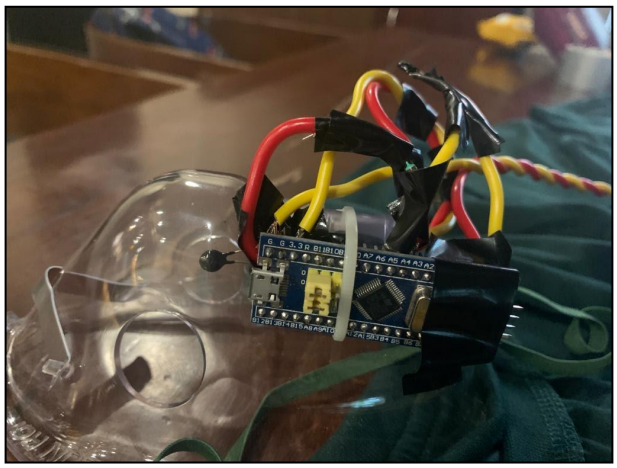

Fig. 6.4 Hardware integration on Mask

This solution an ease in need for real time monitoring as the accuracy of the system is proven. With the help of AI, the predictive analysis is done which helps an individual confirm the presence of the virus. This framework has altogether brought the need of wearables as one of primary solution in order to reduce the risk.

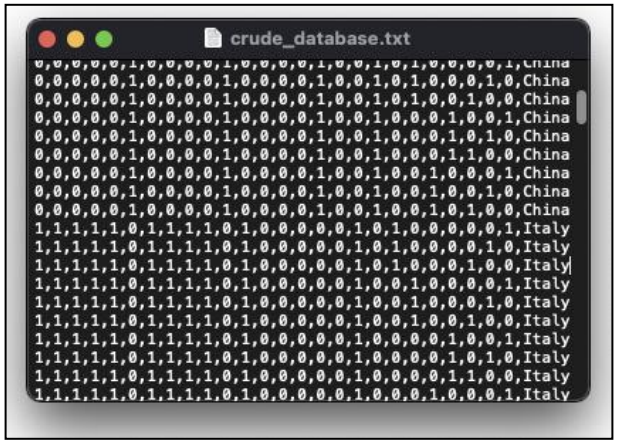

Fig. 6.5 Crude Dataset File

Since the virus exhibits different symptoms in different countries due to the mutations, the dataset is obtained from a standardized database. The crude dataset is shown in Fig. 6.5. The network is tuned by training it on this dataset built on health conditions of 300,000 people from different geographic locations.

The data obtained is formatted for inputs with 20 different parameters as shown in Fig. 6.6, where each bit in the every input corresponds to a symptom experienced by an individual like cold, fever, cough, breathlessness, nausea, headache, conjestion, diarrhoea, and so on.

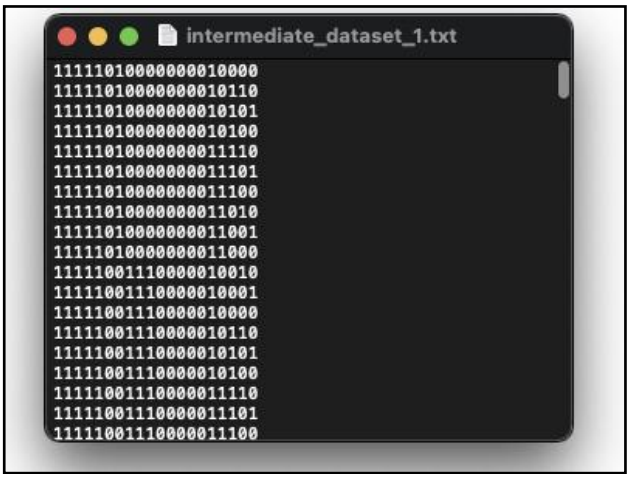

Fig. 6.6 20-bit Input

The data obtained is formatted for inputs with 20 different parameters as shown in Table. 3, where each bit in the every input corresponds to a symptom experienced by an individual like cold, fever, cough, breathlessness, nausea, headache, conjestion, diarrhoea, and so on.

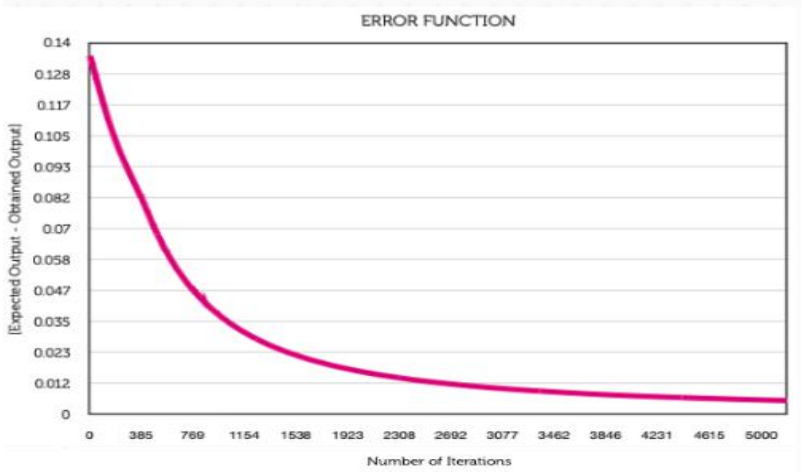

Fig. 6.7 Error Plot of the trained model

Multiple iterations and parameter tuning to obtain the least error. It is observed that the error functions in training the predictive model are in a decreasing trend as seen in Fig. 6.7. This is the final step of training the model. The output thus obtained is an error function that is obtained after every sample is trained.

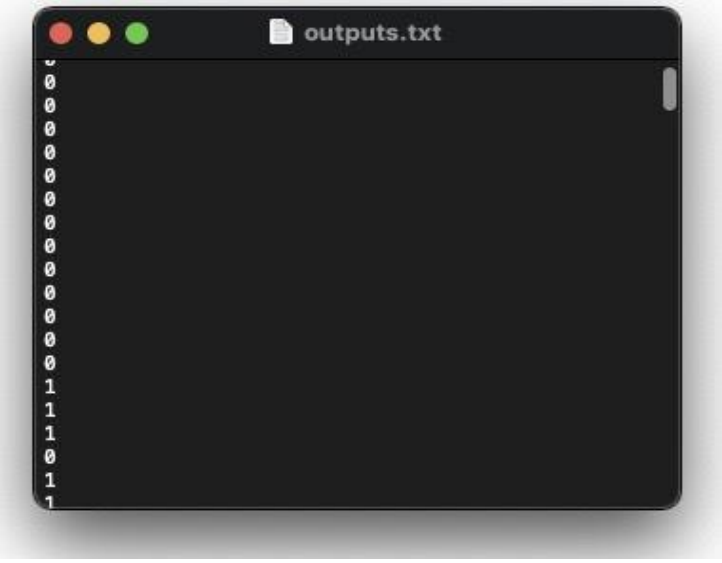

Fig 6.8 Output file generated showing the presence of COVID-19

The Fig. 6.8 illustrates the output file generated, showing 1-bit binary output for every combination of input parameters. Zeroes and ones represent absence and presence of COVID.

Neural network gives us a total of (20x10) + (5x1) = 205 trainable parameters in the model. This is almost similar to a 205th-degree equation since it is highly impossible to fit in such a high degree equation using mathematical methods or any conventional methods. Neural networks giving a very efficient solution to this problem is a good tool to model such diverse datasets.

The health monitoring app combines the output from sensors and modules, then visualizes the data received. After the predictive model is trained, it is deployed on the application. The health parameters obtained from each of the sensors are visualized on the app using the Thingspeak IoT Platform. The app alerts the guardians and shares the real-time location of the patient when the patient is at risk.

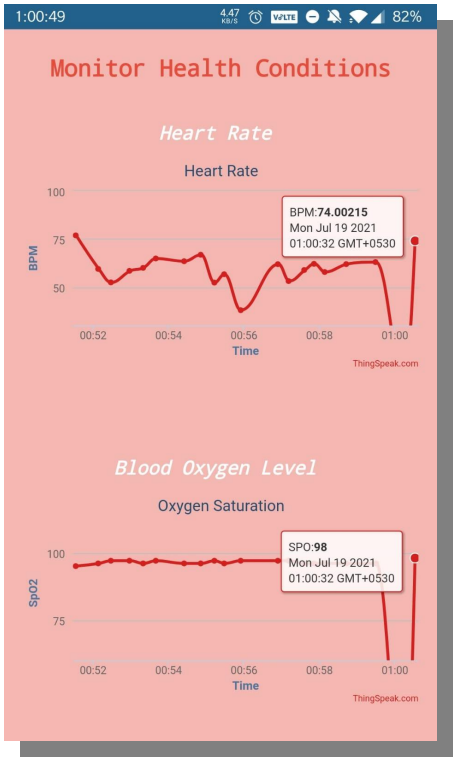

Fig 6.9 (a)

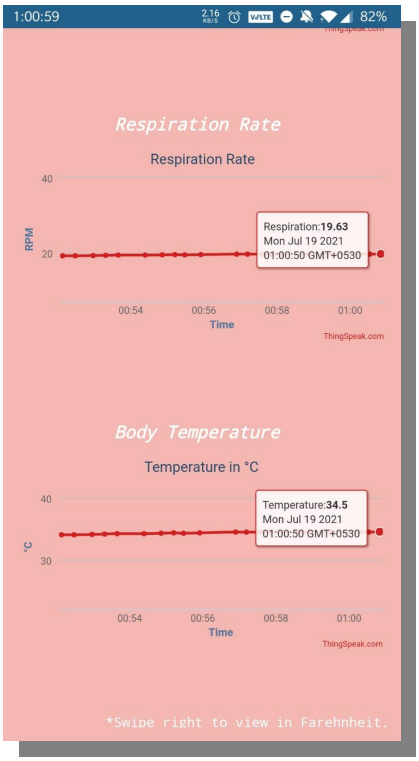

Fig 6.9 (b)

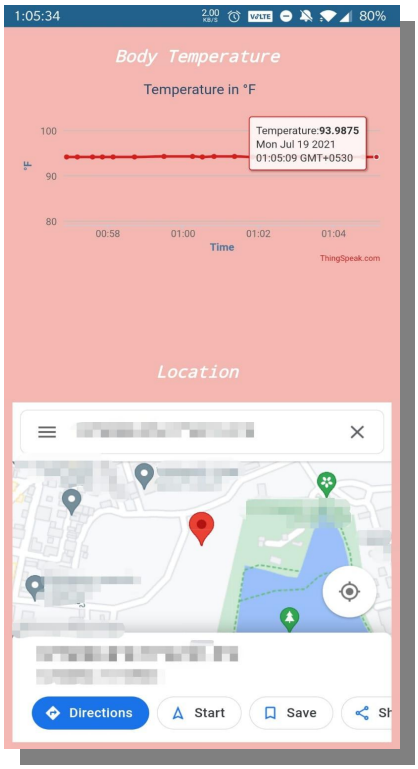

Fig 6.9 (c)

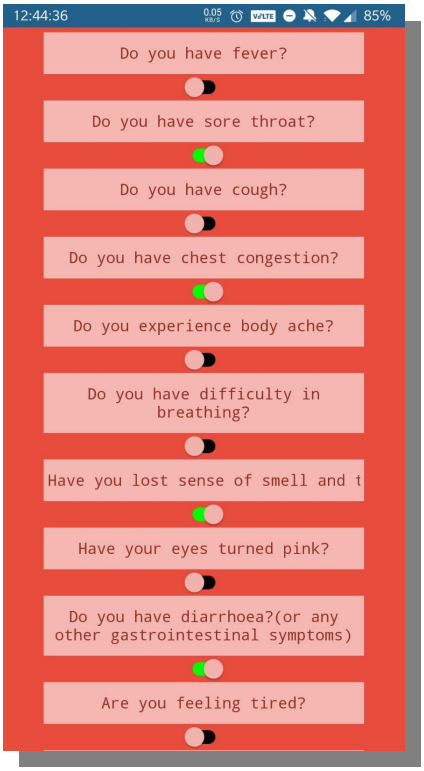

Fig 6.9 (d)

Fig 6.9 Graphical Representation (a) Heart Rate & Blood Saturation Level (b) Respiration Rate & Body Temperature in °C (c) Temperature in °F & Location of Patient (d) Self Assessment Questionnaire

The Fig 6.9 (a) shows the visualization of the heart rate and blood oxygen level, which was 74.002bpm and 98% Spo2 respectively. This data has been analyzed at 1:00:32 time frame. The Fig 6.9 (b) shows the visualization of the respiration rate and temperature in Celsius, which was 19.63respiration per minute and 34.5°C respectively. The Fig 6.9 (c) shows the visualization temperature in Fahrenheit, which was 93.9875°F. The location of the patient is also shown on the app, which is blurred out for privacy. The Fig 6.9 (d) shows the self assessment based questionnaire, which can be used to enter responses by the user, aims to predict the presence of COVID-19.

Table 6.1 Result Comparison

Parameters	***Reference [10]***	***Reference [1]***	***Reference [7]***	***Proposed Work***
Application	Heart Rate, Temperature	Breathing Rate	ANN Network	Heart Rate, Temperature, Blood Oxygen level, Breathing Rate
Main Integrant	Heart Rate Sensor, Temperature Sensor	Thermistor	Neural Network	Pulse Oximeter, Thermistor, Temperature Sensor
Working Principle	Sensor Technology	Comparison of Relative Voltages	Prediction by LSTM	Sensor Technology & Predictive Analysis using Neural Network
Micro-controller	Atmega328	TI-MSP430	-	Lilypad Arduino

The comparison of key characteristics of prior work and proposed work is shown in Table 6.1. As the parameters considered vary from each other, our work has tried to overcome the cons and provide accurate results.

CHAPTER 7

CONCLUSION AND FUTURE SCOPE

In today's world due to rising threats of COVID-19, the need to solve the problem with the help of different technologies is essential. The solution provided above satisfies the need and also can be further considered for advancements.

Artificial intelligence is a very important and promising tool for detecting early COVID-19 infections and monitoring the condition of infected ones remotely. The advent of useful algorithms greatly improves the sequence of processing and decision-making. The participation of the Internet of Things that integrates all the above technologies provides a good result. In this paper, we've introduced a low-power IoT wearable devices for the COVID-19 application called Camisa (Contactless Patient Monitoring System). We've identified the foremost components of the proposed system and explained its implementation details. Performance ratings indicate that wearable shirts and masks are an inexpensive device.

Future scope to this area will include ensuring data access and security to shield the privacy of hospitals and patients, further as enabling voice recognition for the proposed wearable device which can be more advanced.The mask is capable of analyzing the breathing pattern of an individual, after consecutive 'no breaths', an alert is notified to the person monitoring. Pulse oximeter and Temperature Module provides output signifying the Heart Rate, SpO2 and Temperature of the patient. In case of emergency, location of the patient is shared with the hospital to rush the patient for immediate intensive care. The given datasets are recognized preprocessing and data splitting process.The output is a least error count function which signifies if the patient needs to be isolated and monitored. Hence the proposed model of the shirt and mask, and the neural network provides the solution for remote monitoring and prediction of COVID-19 respectively. In order to stop the spread and reduce the risk of fatalities around the world.

REFERENCES

[1] Gupta, Maneesh, and Hana Qudsi. "Low-cost, thermistor based respiration monitor." 2013 39th Annual Northeast Bioengineering Conference. IEEE, 2013.

[2] Jovanov, Emil, Dejan Raskovic, and Rick Hormigo. "Thermistor-based breathing sensor for circadian rhythm evaluation." Biomedical sciences instrumentation 37 (2001): 493-498.

[3] Nasajpour, Mohammad, et al. "Internet of Things for current COVID-19 and future pandemics: An exploratory study." Journal of healthcare informatics research (2020): 1-40

[4] Prince, P. Grace Kanmani, et al. "Characterization of Signals of Noncontact Respiration Sensor for Emotion Detection Using Intelligent Techniques." Computational Intelligence in Healthcare: 161.

[5] Jansi, R., R. Amutha, and S. Radha. "Remote monitoring of children with chronic illness using wearable vest." Telemedicine Technologies. Academic Press, 2019. 121-137.

[6] Imran, Ali, et al. "AI4COVID-19: AI enabled preliminary diagnosis for COVID-19 from cough samples via an app." Informatics in Medicine Unlocked 20 (2020): 100378.

[7] Shahid Farah Anleela Zameer and Muhammad Muneeb “Predictions for COVID-19 with Deep Learning Models of LSTM GRU and B1-LSTM” Chaos Solitions &Fractals 140(2020)

[8] Nachiar, Cecil C., et al. "Design of Cost-effective Wearable Sensors with Integrated Health Monitoring System." 2020 Fourth International Conference on I-SMAC (IoT in Respiratory Monitoring System using thermistor 1 Gagandeep kour , 2Mohammad Rouman ,Geetha.M31,2UG Students, Assistant Professor Department of Biomedical Engineering BIHAR, BIST, Bharath UniversityChennai- 6000073.

[9] Qudsi, Hana, and Maneesh Gupta. "Low-cost, thermistor based respiration monitor." 2013 29th Southern Biomedical Engineering Conference. IEEE, 2013.

[10] Aziz, Kahtan, et al. "Smart real-time healthcare monitoring and tracking system using GSM/GPS technologies." 2016 3rd MEC International Conference on Big Data and Smart City (ICBC). IEEE

[11] Kalavakonda, Rohan Reddy, et al. "A Smart Mask for Active Defense Against Coronaviruses and Other Airborne Pathogens." IEEE Consumer Electronics Magazine (2020)

[12] Huang, Chiou-Jye, et al. "Multiple-input deep convolutional neural network model for covid-19 forecasting in china." MedRxiv (2020)

[13] Che, Zhen-Guo, Tzu-An Chiang, and Zhen-Hua Che. "Feed-forward neural networks training: a comparison between genetic algorithm and back-propagation learning algorithm." International Journal of Innovative Computing Information and Control 7.10 (2011): 5839-5850.

[14] Tsai, Chang-Hung, et al. "A hardware-efficient sigmoid function with adjustable precision for a neural network system." IEEE Transactions on Circuits and Systems II: Express Briefs 62.11 (2015): 1073-1077

[15] Erb, Randall J. "Introduction to backpropagation neural network computation." Pharmaceutical research 10.2 (1993): 165-170.

[16] Nachiar, Cecil C., et al. "Design of Cost-effective Wearable Sensors with Integrated Health Monitoring System." 2020 Fourth International Conference on I-SMAC (IoT in Respiratory Monitoring System using thermistor 1 Gagandeep kour , 2Mohammad Rouman ,Geetha.M31,2UG Students, Assistant Professor Department of Biomedical Engineering BIHAR, BIST, Bharath UniversityChennai- 6000073.

[17] Kalavakonda, Rohan Reddy, et al. "A Smart Mask for Active Defense Against Coronaviruses and Other Airborne Pathogens." IEEE Consumer Electronics Magazine (2020)

[18] [4]Prince, P. Grace Kanmani, et al. "Characterization of Signals of Noncontact Respiration Sensor for Emotion Detection Using Intelligent Techniques." Computational Intelligence in Healthcare: 161

[19] martin, james, and randall hickle. “Apparatus to deliver oxygen to a patient.” U.s. patent application no. 11/157,459.

[20] krishnan, d. shiva rama, subhash chand gupta, and tanupriya choudhury. “an iot based patient health monitoring system.” 2018 international conference on advances in computing and communication engineering (icacce). Ieee, 2018

APPENDIX -A

CAMISA : An AI Solution for COVID-19

Bhanuteja G
Electronics and Communication
MVJ College of Engineering
Bangalore, India
bhanuteja.g@mvjce.edu.in

Kaustubh Anand Kandi
Micro Electronics Design Facility
URSC, ISRO
Bangalore, India
kaustubh@ursc.gov.in

Srinidhi K
Electronics and Communication
MVJ College of Engineering
Bangalore, India
srinidhikrao05@gmail.com

Paulson Premsingh S
Electronics and Communication
MVJ College of Engineering
Bangalore, India
paulsonpremsingh7@gmail.com

Dikshitha R
Electronics and Communication
MVJ College of Engineering
Bangalore, India
dikshitharavisha@gmail.com

Anil Kumar R
Electronics and Communication
MVJ College of Engineering
Bangalore, India
anilkumarr0475@gmail.com

***Abstract*—The COVID-19 pandemic has created an unparalleled need for remote patient monitoring and has primarily impacted the world as the mortality rate has increased rapidly. As long as coronavirus exists, mutations of the virus continue to happen, which also insists on the need for remote monitoring. Healthcare sectors require the help of many new technologies such as IoT, Artificial Intelligence, Neural Networks, and sensor technology which can play an important role. The proposed system predicts the COVID-19 symptoms in a patient with the integration of sensor technology and AI. This system is effective in solving the crisis. It includes a shirt and a mask measuring the heart rate, blood oxygen level, and respiration rate. In addition to this is the predictable AI model, where the symptoms predict whether the patient is COVID-19 positive or not.**

Keywords—COVID-19, IoT, Contactless Health Monitoring, Neural Network, Sensor Technology, Lilypad Arduino, Pulse Oximeter

I. Introduction

Coronavirus has caused much havoc around the world. As on May 20, 2021, there have been 165,565,058 confirmed cases, resulting in 3,434,004 deaths and disturbing life all around the world. The losses are exacerbating day by day. With very little research done globally on the pandemic since SARS COV-2, there is no cure and less vaccination until recent days. Controlling the growth by early testing individuals and quarantining the coronavirus-affected individuals is the only implicit solution against the infectious COVID-19. However the ease to proceed with this strategy in order to control the deadly virus is not viable. There is a high risk of spread of the virus in crowded places amongst people with low immunity is still a major concern. The need for testing this virus in an early stage is still a key differentiator in most of the countries in order to reduce the pandemic curve.

In many countries the vaccination drives are happening at a slow pace, this exposes the majority of the population to COVID-19. The availability of beds has also narrowed down in hospitals, and the health status of COVID-19 infected patients are not remotely monitored as expected. The need for patient monitoring which is both easy to use and also produces accurate results, is a solution that can dissolve the problem. We try to provide a solution that solves this issue which is very inexpensive for measuring the health status of a patient. We aim to produce an AI-based platform for monitoring the patient who has been infected by the coronavirus. The integration of the AI model on wearable devices solves the problem to some extent.

This project was supported by Karnataka State Council for Science and Technology, under "Student Project Programme - 44th Series", with Reference Number 44S_BE_4342

As coronavirus was recognised only in the year 2019, the information available on the symptoms was very less. Considering the above situation, the previous work on this problem is highlighted where algorithms used for detecting COVID-19 such as LSTM in [1] (Long Short Term Memory) and Fuzzy rule-based predictions, are having smaller datasets. In order to overcome this, we provide a solution that evaluates the symptoms experienced by an individual, by using a relatively larger dataset. We perform algorithms like backpropagation which is a simple technique and is fast in execution. This is an important aspect in resolving the COVID-19 pandemic by integrating the neural network model with the hardware that consists of a shirt along with a nebulizer.

When the patient uses our proposed system during the incubation period, the data obtained from the sensors is sent to the user-friendly application for real-time monitoring. This also gives alert notifications whenever the patient's health is at risk to hospitals and guardians.

Section II gives us an overview of the previous work, Section III contains the detailed discussion of our proposed system, and Section IV gives the result and compares the solution with the previous work carried out. Finally, Section V provides the conclusion of this project.

II. Literature Survey

As per the previous research for health monitoring systems, the technology used changes day by day to be more accurate and use advanced methodologies. The IoT has been a helping hand in achieving this goal today with many patients being monitored remotely in real-time. Continuous care is given to the patients as their well-wishers are also aware of their health status.

Maneesh Gupta and Hana Qudsi [2] share an important aspect from their perspective where the thermistor is used to measure the breathing rate as COVID-19 affected patients are targeted with the damage of lungs. This device measures an individual's respiration rate by detecting changes in temperature which is mounted on the base of the face mask. However, a delay was observed as the main drawback and is being solved by advancing the hardware used.

AI 4 COVID-19 [3] by Ali Imran, throws light on the fact that AI can solve the problem of COVID-19 and focus on AI over the healthcare system to predict the contagious disease using various algorithms. The ResNet-100 Convolutional Neural Network, a deep learning technique together with a Logistic Regression classifier, is employed to spot the coronavirus pandemic rapidly. With the help of AI, we can categorize a person's health situation as having few symptoms of COVID-19 will be under continuous monitoring.

Dr.R.Poovendran, in his paper [4], has mentioned cost-effective hardware components that are use to detect heart rate, breathing rate and SpO2, and informs the guardian about the patient's situation through alert or alarm call-out system. The paper mentions that the patient uses a headset with the mask for determining the breathing rate and heart rate using a sensor. It has a user-friendly application to collect the data, design all these wearables with the integration of an application where the alarm or alert system can save patients immediately, even in a remote location.

Vishal Varun [6] in his paper mentions that measuring respiratory rate as one of the most tedious tasks. It requires utmost accuracy in order to prove that the patient is suffering from breathlessness or not. The calculation of barometric pressure and the usage of thermistor provides an accuracy of 93% in this research. However the author faces issues with the data communication from the mask to the device like mobile/tablet etc. The usage of signal processing algorithms is seen here monitoring lung functions.

Muhammad E. H. Chowdhury [7] has mentioned solving the problem of COVID-19 through the datasets of SARS using CNN model and chest X-rays through image. Since the dataset collected was only about 319 CheXNet dataset, it wasn't accurate enough to detect pneumonia in humans. The pre-dataset being trained is not sufficient to provide high accuracy rate as the image dataset available from the period of pneumonia is ineffective in detecting coronavirus.

III. Proposed Architecture

A. Hardware

For the hardware which is the combination of a shirt and a mask the following components are considered and are connected to give us the results of pulse, blood oxygen level, breathing rate, real time location and temperature. The proposed block diagram is illustrated in Fig.1.

- The microcontroller used is Lilypad Arduino, which is based on the ATmega328V. It's designed for e-textiles, thereby it can be sewn on fabric with other components using a conductive thread. The LilyPad Arduino can be programmed with the Arduino IDE.
- The NodeMCU is used to collect the sensory outputs from the microcontrollers and send it to the App over the Internet. It runs on ESP8266 firmware having 128KB RAM and 4MB of flash memory to store the data.

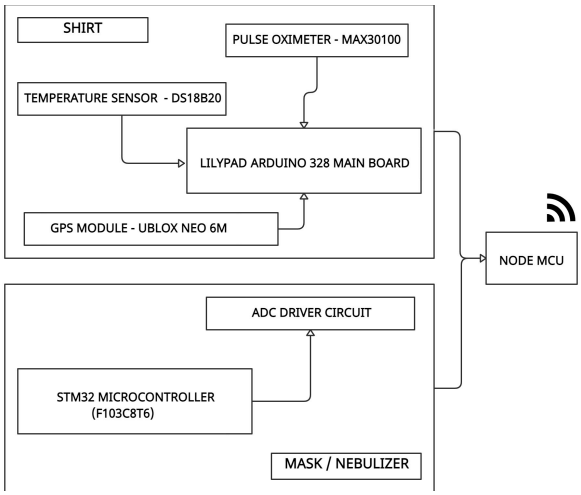

Fig. 1. Block diagram of Shirt and Mask

- Here we use the MAX30100 Pulse Oximeter sensing device. It's a measuring device that obtains its readings from received intensity of light which is later converted to electrical signals. The LEDs are used to get the data by placing the sensor on the fingertip. The sensor is embedded on the shirt circuit, connected by an elongated cable that reaches the tip of the finger measuring accurate values.
- The STM32 Blue Pill is an ARM Cortex M3 Microcontroller. The STM board operates on a 3.3V power supply and uses a 32-bit processor. With the thermistor attached, a code was built for measuring the respiration rate.
- The Ublox NEO6M GPS module used in this project determines its location and we obtain the output, that is the latitude and longitude of its position. It exhibits high sensitivity when used indoors. The battery in the GPS module is used for power backup, and the EEPROM present in the GPS module stores the configured settings providing the needful results.
- There is a patch antenna present in the module which has a sensitivity of -161dBm. A U.FL cable is used to connect the antenna and the module. This allows great flexibility when the GPS module is mounted onto our shirt.
- The temperature sensor that is embedded in the shirt is the DS18B20 and has a 1-Wire bus used for communication with the microcontroller. It operates over the temperature range of -55°C to +125°C and accuracy is ±0.5°C for the range from -10°C to +85°C.
- For heart rate and SpO2 (Blood Oxygen level) which is the major parameter for COVID-19, if the SpO2 level drops below 90%, it will create an alert system through WiFi module or by sending an SMS to the patient's guardian and hospital.

B. Software

The software used for programming the components of the shirt and mask is Arduino IDE. The ease with which an Arduino can obtain sensor values is one of the features that makes it very useful. ThingSpeak is used to aggregate, visualize, and analyze live data streams, which are sent to the app over the Internet. PlatformIO is another software used, which is an open-source ecosystem for IoT development.

C. Camisa - Shirt

With the ongoing swath of the COVID-19 pandemic situation, health care monitoring has become an essential part of human lives. A few health monitoring systems including wearable devices like watches, waist belt, shirt, mask, etc. are effective when used for remote applications. One such confirmation for real-time health monitoring [5] is the use of a shirt that is embedded with sensors. The circuit of shirt is shown in Fig. 2. This gives us accurate results of patients when they are in their quarantine period and get alerts when the sensor values reach a threshold level indicating at-risk condition.

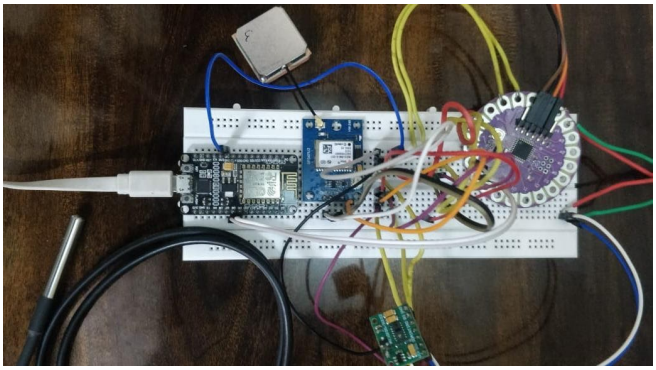

Fig. 2. Shirt Circuit which can be sewn on textile.

The need to measure blood oxygen saturation decreasing during the incubation period gives way to the pulse oximeter. Based on the light signals which are reflected from the blood cells, it converts them to electrical signals. The output is mathematically processed to measure oxygen saturation level and heart rate. The patient's temperature is recorded by the temperature sensor.

If the threshold is exceeded, the real-time location of the patient is shared with the hospital to rush the patient for immediate intensive care. The threshold values of sensors in the shirt is given in Table I.

TABLE I. THRESHOLD VALUES FOR SHIRT

S. No.	Ideal versus Risk Values for Alert Systems		
	Parameters	*Ideal*	*At-Risk*
1.	SpO2 Level (%)	94 -100	< 91
2.	Heart Rate (bpm)	70-100	>110 [a]
3.	Temperature (in °F)	97 - 99	>100

a. Irregularity is experienced, varies among individuals

D. Camisa - Mask

The use of face masks and shields has reduced the risk of virus transmission. Adding onto this primary solution, our focus is to develop a smart nebulizer, as illustrated in Fig. 3, which can be worn by the patient during their quarantine period. This is helpful in order to continuously monitor their breathing pattern and temperature using the NTC thermistor that is attached inside the nebulizer [6].

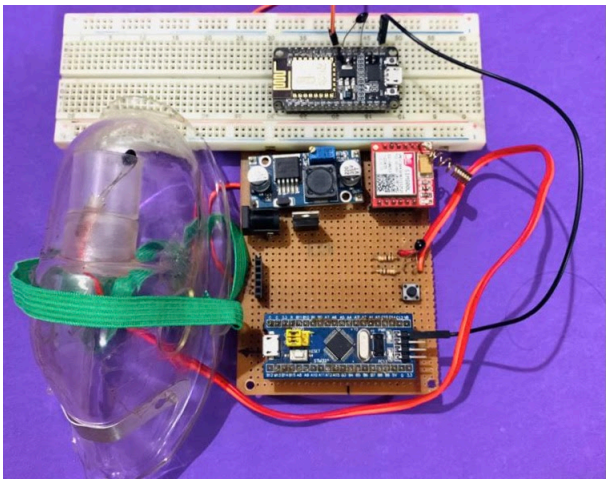

Fig. 3. Circuit for Mask

The STM32 is an ARM Cortex M3 Microcontroller that is programmed with a thermistor to collect the breathing pattern of an individual. We try to analyze the number of 'no breaths' conditions, using the parameter shown in Table II. After consecutive 'no breaths', an alert is notified to the person monitoring.

TABLE II. THRESHOLD VALUES FOR MASK

S. No.	Ideal versus Risk Values for Alert Systems		
	Parameters	*Specified Value*	*At Risk*
1.	Breathing Rate (respirations per min)	12 - 20	> 20 [a]

a. Varies between individuals

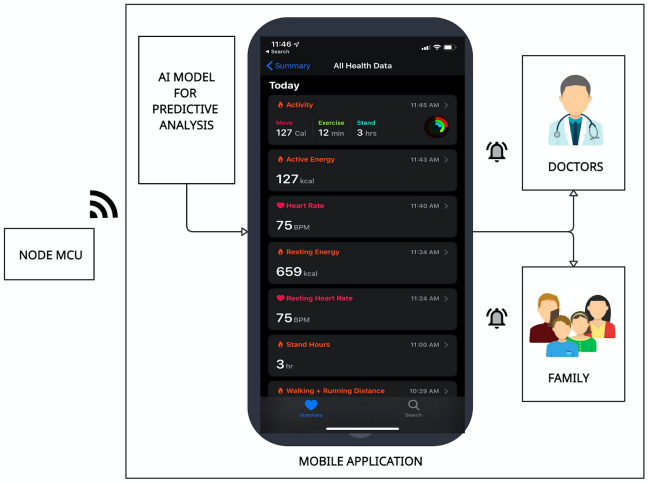

Fig. 4. Transfer of data to the application

After the data is collected from the shirt and the mask, the data is visualized on the application for remote monitoring. The transfer of data to the application is illustrated in Fig. 4.

E. Neural Network Implementation

An ANN model is used to design and compute mathematical algorithms resembling how the brain reacts to sensory inputs. The brain consists of neurons that are interconnected to form a huge network. Neural Network, in general, give us various advantages in numerous fields, one such advantage is the healthcare system, which uses the neural network in prediction and spread of disease. A neural network is made up of millions of artificial neurons and is interconnected by nodes, therefore called a processing unit.

Initially, a mathematical model of all the neurons and the interconnections between them are created. Then the inputs are provided to the network, and the neurons convolute with the inputs systematically, to generate an output, which is fed into the next level of neurons, as shown in Fig. 5. The process repeats itself until the output is reached, in the predefined hidden layers.

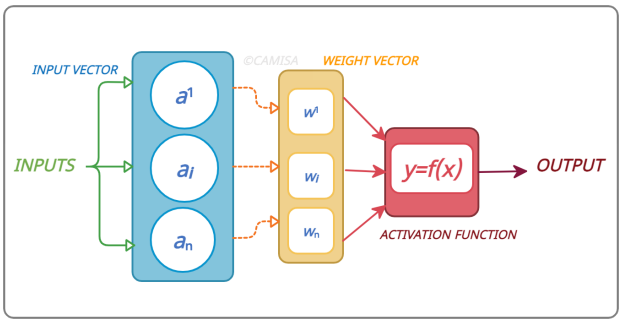

Fig. 5. Neural Network Architecture

The ANN model can also be used to estimate the confirmation of the coronavirus through the available datasets [7]. In order to limit the propagation speed and the propagation power of the coronavirus, it is important to predict and identify the infected patients and isolate them. The flowchart of neural network is explained in Fig. 6. Below are the steps carried out for the Predictive AI model using C language and Python.

a) Collection of data from various sources: Since the virus exhibits different symptoms in different countries due to the mutations, the dataset is obtained from a standardized database. The network is tuned by training it

on this dataset built on health conditions of 300,000 people from different geographic locations.

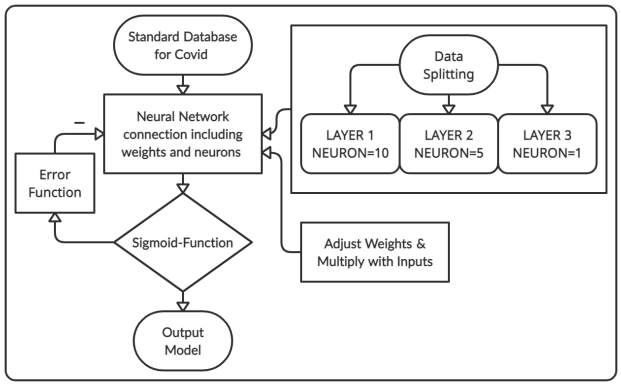

Fig. 6. Flowchart for Neural Network

b) Database formatting and creation of labelled data: The data obtained is formatted for inputs with 20 different parameters as shown in Fig. 7, where each bit in the every input corresponds to a symptom experienced by an individual like cold, fever, cough, breathlessness, diarrhoea, and so on. Thereby giving us maximum of 20 different parameters to predict the condition of a single user. The parameters are set giving the input layer size over 200,000 inputs. We labelled the data as per the requirement and get the final raw data for further tuning.

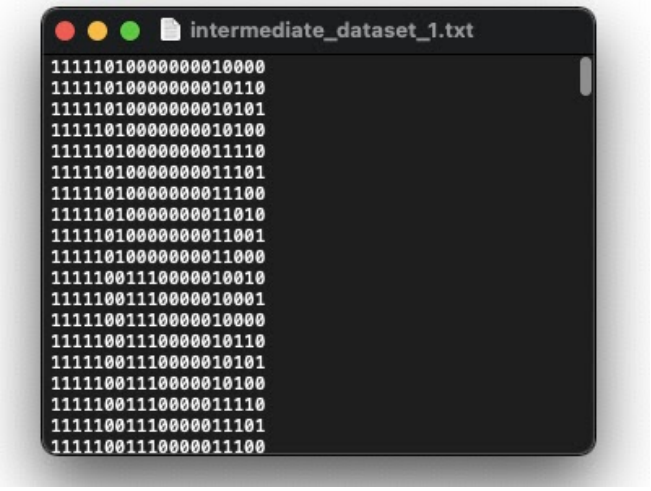

Fig. 7. Formatted Input Dataset File

c) Database formatting to make it compatible with AI: The raw input data is made compatible for AI by preprocessing. The raw data is processed and fed to the 3 layer feed forward network, where forward propagation is performed below as in Fig. 8.

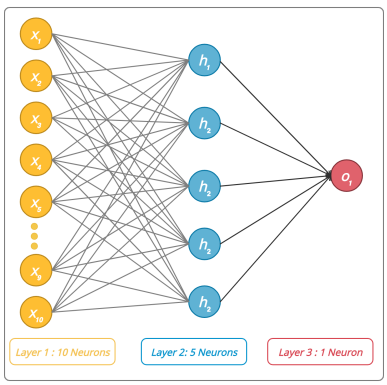

Fig. 8. A 3 Layer Feed Forward Network

d) Theoretical modelling of ANN: For performing the theoretical analysis, the arrays are set up by assigning random weights. This is a 2D array matrix where one column represents a single node, thus obtaining a 10x20 matrix. This final matrix obtained is used to train the model. Equations (1) is the input vector fed into the network. Equations (2) and (4) calculate the weighted sum of inputs of the respective layers which is subsequently used to measure the activation function given by (3) and (5).

$$x_i = a_i^{(1)},\ i \in 1,2,3 \qquad (1)$$

$$z^{(2)} = W^{(1)}x + b^{(1)} \qquad (2)$$

$$a^{(2)} = f(z^{(2)}) \qquad (3)$$

$$z^{(3)} = W^{(2)}a^{(2)} + b^{(2)} \qquad (4)$$

$$a^{(3)} = f(z^{(3)}) \qquad (5)$$

Initially, a loop is started to run through the data used for training in each layer. In each iteration the order in which the training data runs through is randomized, to ensure that local minima do not have a union.

The data is fed through the network in order to calculate the activation function of the hidden layers and output layer's nodes (in our case, sigmoid function as in (6))[9].

$$\sigma(x) = \frac{1}{1+e^{-x}} \qquad (6)$$

Later on updating the associated weights and comparing the output after multiple iterations, on how well the neural network performed as to the given training sample. This is referred to as the cost function, given by (7) and illustrated in Fig. 9.

$$Cost\ Function\ (J) = \frac{1}{n}\Sigma_{i=0}^{n}(y^{(i)} - (m\,x^{(i)} + b))^2 \qquad (7)$$

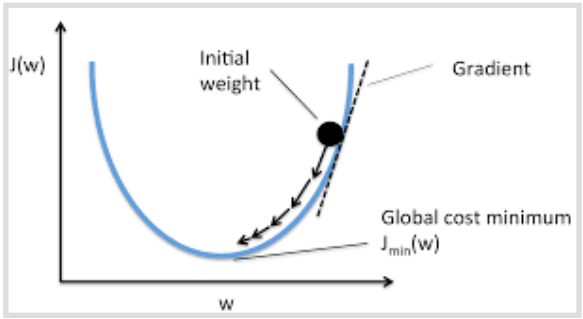

Fig. 9. Cost Function

The backpropagation algorithm checks for the least value of the error function in the weights vector. On backpropagating the error function to the hidden layer, the algorithm tunes the network in accordance with the error rate [10]. This calculates the gradient of the obtained loss function with respect to weights and thus minimizes the error.

$$Gradient\ Descent = \frac{\partial J}{\partial w} \qquad (8)$$

The gradient descent for a given algorithm is an optimised algorithm used to find the parameters that reduce the cost function as shown above in (8).

e) Parameter tuning: Using the obtained values, we perform parameter tuning in order to find the error functions. It is observed that the error functions are in a decreasing trend as seen in Fig.10. The obtained outputs of the error function are compared as the values of various parameters such as learning rates, i.e, alpha are varied (9).

$$dx = alpha * |\frac{\partial J}{\partial w}| \qquad (9)$$

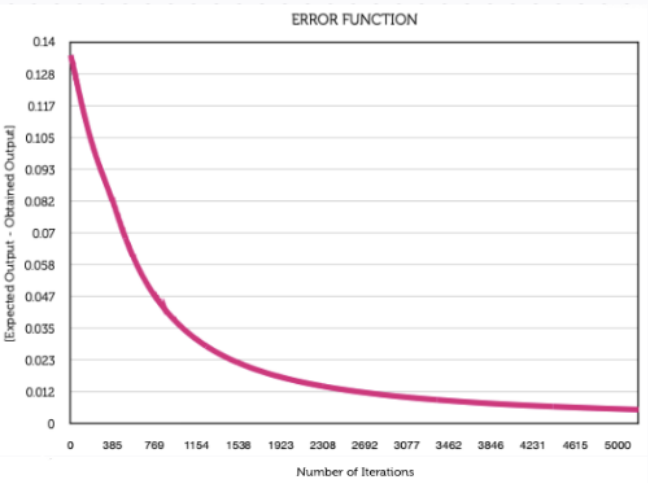

Fig. 10. Error Plot of the trained model

f) Training the model and validation: The model and algorithms are trained on 60% of the dataset where the dataset is split. The remaining 40% of the dataset is used for validation, to prove the predictability of the algorithm.

g) Multiple iterations and parameter tuning to obtain the least error: This is the final step of training the model. The output thus obtained is an error function that is obtained after every sample is trained. The Fig. 11. illustrates the output file generated, showing 1-bit binary output for every combination of input parameters. Zeroes and ones represent absence and presence of COVID.

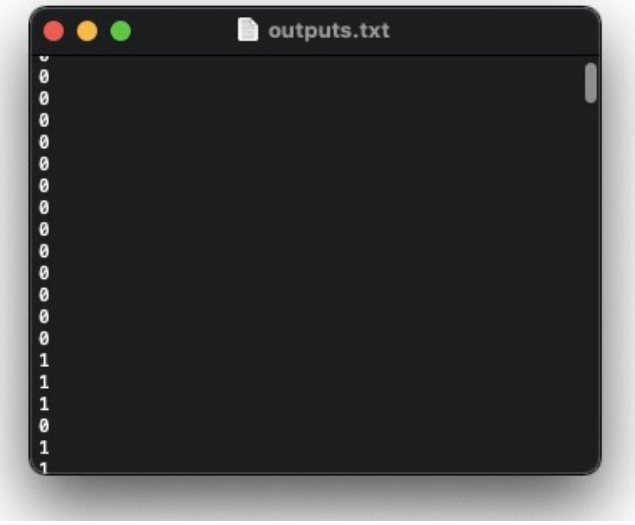

Fig. 11. Output file generated showing the presence of COVID-19

Neural network gives us a total of (20x10) + (5x1) = 205 trainable parameters in the model. This is almost similar to a 205th-degree equation since it is highly impossible to fit in such a high degree equation using mathematical methods or any conventional methods. Neural networks giving a very efficient solution to this problem is a good tool to model such diverse datasets.

To conclude this model we integrate the output obtained from the neural network, which gives us information about whether a person has COVID-19 or not based on the symptoms.

Our project shows that it is possible to acquire a quality model for the prediction of disease using AI, with inputs as symptoms experienced by an individual. The predictions prove that AI models can also solve problems in prediction of any disease. The application of AI methods should also be modelled keeping in mind the present and future spread of diseases and in an attempt to predict and prevent the impact of such infections. Our future plan widely leans towards making the model available worldwide solving the problem of COVID-19.

F. User Friendly Application

The health monitoring app combines the output from sensors and modules, then visualizes the data received. After the predictive model is trained, it is deployed on the application. The app consists of two main section:

a) Self Assessment: It has a questionnaire filled using toggle switches that record users' responses, which correspond to the COVID-19 symptoms experienced by the user. Based on which the app predicts an output of whether the patient is COVID-19 positive or not. Fig. 12 illustrates the questionnaire.

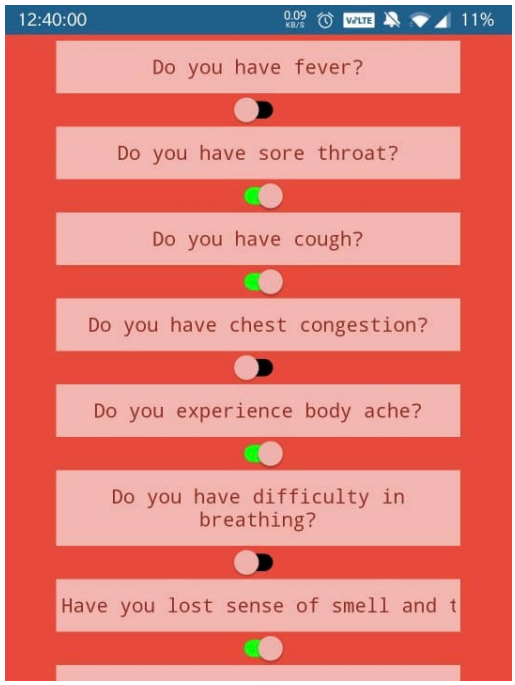

Fig.12. Self Assessment Questionnaire

b) Real-Time Health Monitoring: The health parameters obtained from each of the sensors are visualized on the app using the Thingspeak IoT Platform. The app alerts the guardians and shares the real-time location of the patient when the patient is at risk.

Fig. 13 shows heart rate and blood oxygen saturation in real time. Similarly the other parameters, breath rate, temperature and location of the patient is also visualized on the application.

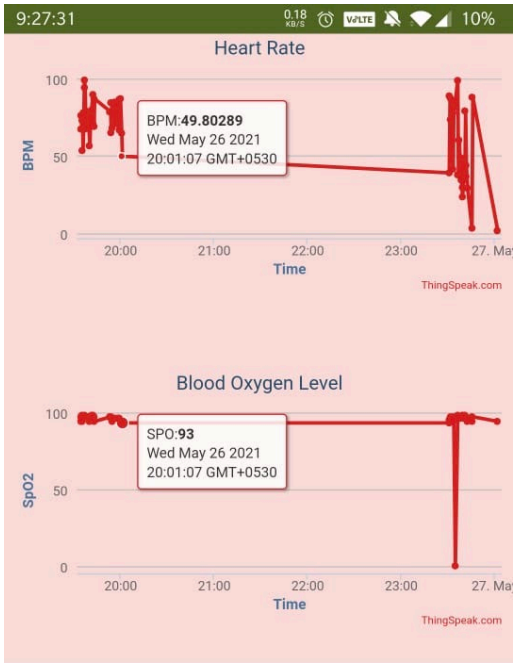

Fig. 13. Health Monitoring on the App

IV. Result Comparison

The comparison of key characteristics of prior work and proposed work is shown in Table III. As the parameters considered vary from each other, our work tries to overcome the cons and provide accurate results.

Table III. Result Comparison

Parameters	***Reference [5]***	***Reference [2]***	***Reference [3]***	***Proposed Work***
Application	Heart Rate, Temperature	Breathing Rate	ANN Network	Heart Rate, Temperature, Blood Oxygen level, Breathing Rate
Main Integrant	Heart Rate Sensor, Temperature Sensor	Thermistor	Neural Network	Pulse Oximeter, Thermistor, Temperature Sensor
Working Principle	Sensor Technology	Comparison of Relative Voltages	Prediction by LSTM	Sensor Technology & Predictive Analysis using Neural Network
Microcontroller	Atmega328	TI-MSP430	-	Lilypad Arduino

V. Conclusion

In today's world due to rising threats of COVID-19, the need to solve the problem with the help of different technologies is essential. The solution provided above satisfies the need and also can be further considered for advancements.

Artificial intelligence is a very important and promising tool for detecting early COVID-19 infections and monitoring the condition of infected ones remotely. The advent of useful algorithms greatly improves the sequence of processing and decision-making. The participation of the Internet of Things that integrates all the above technologies provides a good result. In this paper, we've introduced a low-power IoT wearable devices for the COVID-19 application called Camisa (Contactless Patient Monitoring System). We've identified the foremost components of the proposed system and explained its implementation details. Performance ratings indicate that wearable shirts and masks are an inexpensive device. Future scope to this area will include ensuring data access and security to shield the privacy of hospitals and patients, further as enabling voice recognition for the proposed wearable device which can be more advanced.

References

1. Shahid Farah Anleela Zameer and Muhammad Muneeb "Predictions for COVID-19 with Deep Learning Models of LSTM GRU and B1-LSTM" Chaos Solitions &Fractals 140(2020)
2. Gupta, Maneesh, and Hana Qudsi. "Low-cost, thermistor based respiration monitor." 2013 39th Annual Northeast Bioengineering Conference. IEEE, 2013.
3. Imran, Ali, et al. "AI 4COVID-19: AI enabled preliminary diagnosis for COVID-19 from cough samples via an app."
4. Nachiar, Cecil C., et al. "Design of Cost-effective Wearable Sensors with Integrated Health Monitoring System." 2020 Fourth International Conference on I-SMAC (IoT in Respiratory Monitoring System using thermistor 1 Gagandeep kour , 2Mohammad Rouman ,Geetha.M31,2UG Students, Assistant Professor Department of Biomedical Engineering BIHAR, BIST, Bharath UniversityChennai- 6000073.
5. Aziz, Kahtan, et al. "Smart real-time healthcare monitoring and tracking system using GSM/GPS technologies." 2016 3rd MEC International Conference on Big Data and Smart City (ICBC). IEEE
6. Kalavakonda, Rohan Reddy, et al. "A Smart Mask for Active Defense Against Coronaviruses and Other Airborne Pathogens." IEEE Consumer Electronics Magazine (2020).
7. Huang, Chiou-Jye, et al. "Multiple-input deep convolutional neural network model for covid-19 forecasting in china." MedRxiv (2020).
8. Che, Zhen-Guo, Tzu-An Chiang, and Zhen-Hua Che. "Feed-forward neural networks training: a comparison between genetic algorithm and back-propagation learning algorithm." International Journal of Innovative Computing Information and Control 7.10 (2011): 5839-5850.
9. Tsai, Chang-Hung, et al. "A hardware-efficient sigmoid function with adjustable precision for a neural network system." IEEE Transactions on Circuits and Systems II: Express Briefs 62.11 (2015): 1073-1077
10. Erb, Randall J. "Introduction to backpropagation neural network computation." Pharmaceutical research 10.2 (1993): 165-170.

APPENDIX-B

यू.आर.राव उपग्रह केंद्र
अंतरिक्ष विभाग
भारत सरकार
विमानपुर पोस्ट
बेंगलूरु - 560 017

U. R. Rao Satellite Centre
Department of Space
Government of India
Vimanapura Post
Bengaluru - 560 017

DATE : 05-07-2021

Certificate

This is to certify that Ms. Dikshitha R
of MVJ College of Engineering *has undergone*
Project training in Micro Circuit & ECAD Facilities *at this Centre*
during the period from 18/01/2021 to 30/04/2021.

COURSE	:	B.E (Electronics & Communication)
PROJECT TITLE	:	CAMISA: An AI Solution for COVID-19
PROJECT MEMBERS	:	1. Srinidhi K 2. Paulson Premsingh S 3. Anil Kumar R
PERFORMANCE	:	Excellent

(Basavaraj.S Akkimaradi)
Group Director PPEG, URSC

यू.आर.राव उपग्रह केंद्र
अंतरिक्ष विभाग
भारत सरकार
विमानपुर पोस्ट
बेंगलूरु - 560 017

U. R. Rao Satellite Centre
Department of Space
Government of India
Vimanapura Post
Bengaluru - 560 017

DATE : 05-07-2021

Certificate

This is to certify that Mr. Anil Kumar R
of MVJ College of Engineering *has undergone*
Project training in Micro Circuit & ECAD Facilities *at this Centre*
during the period from 18/01/2021 to 30/04/2021.

COURSE	:	B.E (Electronics & Communication)
PROJECT TITLE	:	CAMISA: An AI Solution for COVID-19
PROJECT MEMBERS	:	1. Dikshitha R 2. Paulson Premsingh S 3. Srinidhi K
PERFORMANCE	:	Excellent

(Basavaraj.S Akkimaradi)
Group Director PPEG, URSC

यू.आर.राव उपग्रह केंद्र
अंतरिक्ष विभाग
भारत सरकार
विमानपुर पोस्ट
बेंगलूरु - 560 017

इसरो ISRO
Indian Space Research Organisation

U. R. Rao Satellite Centre
Department of Space
Government of India
Vimanapura Post
Bengaluru - 560 017

DATE : 05-07-2021

Certificate

This is to certify that Ms. Srinidhi K
of MVJ College of Engineering *has undergone*
Project training in Micro Circuit & ECAD Facilities *at this Centre*
during the period from 18/01/2021 to 30/04/2021.

COURSE	:	B.E (Electronics & Communication)
PROJECT TITLE	:	CAMISA: An AI Solution for COVID-19
PROJECT MEMBERS	:	1. Dikshitha R 2. Paulson Premsingh S 3. Anil Kumar R
PERFORMANCE	:	Excellent

(Basavaraj.S Akkimaradi)
Group Director PPEG, URSC

यू.आर.राव उपग्रह केंद्र
अंतरिक्ष विभाग
भारत सरकार
विमानपुर पोस्ट
बेंगलूरु - 560 017

इसरो ISRO
Indian Space Research Organisation

U. R. Rao Satellite Centre
Department of Space
Government of India
Vimanapura Post
Bengaluru - 560 017

DATE : 05-07-2021

Certificate

This is to certify that Mr. Paulson Premsingh S
of MVJ College of Engineering *has undergone*
Project training in Micro Circuit & ECAD Facilities *at this Centre*
during the period from 18/01/2021 to 30/04/2021.

COURSE	:	B.E (Electronics & Communication)
PROJECT TITLE	:	CAMISA: An AI Solution for COVID-19
PROJECT MEMBERS	:	1. Dikshitha R 2. Srinidhi K 3. Anil Kumar R
PERFORMANCE	:	Excellent

(Basavaraj.S Akkimaradi)
Group Director PPEG, URSC

APPENDIX - C

MVJ COLLEGE OF ENGINEERING
Since 1982
Engineering A Better Tomorrow
An Autonomous Institute

Approved by AICTE, New Delhi
Affiliated to Visvesvaraya Technological University, Belagavi
Recognized by UGC with 2(f) and 12(B) status
Accredited by NBA and NAAC

iC di3C
compute-communicate-control

Certificate of Appreciation

This is to certify that

Anil Kumar R

is the author of the paper entitled

"CAMISA : An AI Solution for COVID-19"

presented at the Fifth International Conference on Design Innovations for 3Cs Compute-Communicate-Control organized by the Department of ECE and EEE on 11th and 12th June 2021, held at MVJ College of Engineering, Bengaluru.

Dr. M Brindha
Vice Principal

Dr. P Mahabaleshwarappa
Principal

MVJ COLLEGE OF ENGINEERING
Since 1982
Engineering A Better Tomorrow
An Autonomous Institute

Approved by AICTE, New Delhi
Affiliated to Visvesvaraya Technological University, Belagavi
Recognized by UGC with 2(f) and 12(B) status
Accredited by NBA and NAAC

iC di3C
compute-communicate-control

Certificate of Appreciation

This is to certify that

Dikshitha R

is the author of the paper entitled

"CAMISA : An AI Solution for COVID-19"

presented at the Fifth International Conference on Design Innovations for 3Cs Compute-Communicate-Control organized by the Department of ECE and EEE on 11th and 12th June 2021, held at MVJ College of Engineering, Bengaluru.

Dr. M Brindha
Vice Principal

Dr. P Mahabaleshwarappa
Principal

MVJ COLLEGE OF ENGINEERING
Since 1982
Engineering A Better Tomorrow
An Autonomous Institute

iC di3C
compute-communicate-control

Approved by AICTE, New Delhi
Affiliated to Visvesvaraya Technological University, Belagavi
Recognized by UGC with 2(f) and 12(B) status
Accredited by NBA and NAAC

Certificate of Appreciation

This is to certify that

Paulson Premsingh S

is the author of the paper entitled

"CAMISA : An AI Solution for COVID-19"

presented at the Fifth International Conference on Design Innovations for 3Cs Compute-Communicate-Control organized by the Department of ECE and EEE on 11th and 12th June 2021, held at MVJ College of Engineering, Bengaluru.

Dr. M Brindha
Vice Principal

Dr. P Mahabaleshwarappa
Principal

MVJ COLLEGE OF ENGINEERING
Since 1982
Engineering A Better Tomorrow
An Autonomous Institute

iC di3C
compute-communicate-control

Approved by AICTE, New Delhi
Affiliated to Visvesvaraya Technological University, Belagavi
Recognized by UGC with 2(f) and 12(B) status
Accredited by NBA and NAAC

Certificate of Appreciation

This is to certify that

Srinidhi K

is the author of the paper entitled

"CAMISA : An AI Solution for COVID-19"

presented at the Fifth International Conference on Design Innovations for 3Cs Compute-Communicate-Control organized by the Department of ECE and EEE on 11th and 12th June 2021, held at MVJ College of Engineering, Bengaluru.

Dr. M Brindha
Vice Principal

Dr. P Mahabaleshwarappa
Principal

APPENDIX - D

```
//Lilypad Code
#include <SoftwareSerial.h>
#include <Wire.h>
#include <TinyGPS.h>
#include "MAX30100_PulseOximeter.h"
#include <OneWire.h>
#include <DallasTemperature.h>
#include <ArduinoJson.h>

#define ONE_WIRE_BUS 5
#define REPORTING_PERIOD_MS         1000

float Bpm=0.0;
float Spo=0.0;
int i=0;
float TempC=0.0;
float TempF=0.0;
float lat,lon;
String latitude; String longitude;
int RXPin = 13;
int TXPin = 15;
uint32_t tsLastReport = 0;

SoftwareSerial gpsSerial(RXPin, TXPin);
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);
TinyGPS gps;
SoftwareSerial nodemcu(5, 6);
PulseOximeter pox;

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

void pulsox()
{
    pox.update();
    if (millis() - tsLastReport > REPORTING_PERIOD_MS)
    {
        Bpm=pox.getHeartRate();
        Spo=pox.getSpO2();
```

```
                Serial.println("");
                Serial.print("Heart rate:");
                Serial.print(Bpm);
                Serial.print("bpm | SpO2:");
                Serial.print(Spo);
                Serial.println("%");
                Serial.println("");
            tsLastReport = millis();
        }
}

void gpsget()
{
   while(gpsSerial.available())
      {
         if(gps.encode(gpsSerial.read()))
            {
               gps.f_get_position(&lat,&lon);
               latitude = String(lat,6);
               longitude = String(lon,6);
               Serial.println("Lat="+latitude);
               Serial.println("Lon="+longitude);
            }
      }
}

void setup()
{
   Serial.begin(9600);
   sensors.begin();
   nodemcu.begin(9600);
   delay(2000);
   gpsSerial.begin(9600);
   Serial.print("Initializing Pulse Oximeter");
      if (!pox.begin()) {
            Serial.println("FAILED");
            for(;;);
      } else {
            Serial.println("SUCCESS");
      }
      pox.setIRLedCurrent(MAX30100_LED_CURR_7_6MA);
      pox.setOnBeatDetectedCallback(onBeatDetected);

}
```

```
void loop()
{
    DynamicJsonBuffer jsonBuffer;
    JsonObject& data = jsonBuffer.createObject();
    sensors.requestTemperatures();
    pulsox();
    gpsget();

    TempC = sensors.getTempCByIndex(0);
    TempF = ((sensors.getTempCByIndex(0)* 9.0) / 5.0 + 32.0);
    Serial.println("");
    Serial.print("Temperature: ");
    Serial.print(TempC); //print the temperature in Celsius
    Serial.print("C    |    ");
    Serial.print(TempF);
    Serial.println("F");
    Serial.println("");

    data["HeartBeat"] = Bpm;
    data["SpO2"] = Spo;
    data["TemperatureC"] = TempC;
    data["TemperatureF"] = TempF;
    data["Latitude"] = latitude;
    data["Longitude"] = lo;
    data.printTo(nodemcu);
    jsonBuffer.clear();
}
```

```
//STM Code
#include <Thermistor.h>
#include <NTC_Thermistor.h>
#include <SoftwareSerial.h>
#include <ArduinoJson.h>
#define SW PB4
#define SENSOR1_PIN PA0
#define SENSOR2_PIN PA1
#define REFERENCE_RESISTANCE 100000
#define NOMINAL_RESISTANCE 100000
#define NOMINAL_TEMPERATURE 30
#define B_VALUE 3950
#define STM32_ANALOG_RESOLUTION 4095

SoftwareSerial stm(18,17);
Thermistor* thermistor1;
Thermistor* thermistor2;

double celsius1;
double celsius2;

int celsius_1;
int celsius_2;
int flag1=0,flag2=0,flag3=0;
int sample_Array1[20] = {};
int sample_Array2[20] = {};
int sample_Counter = 0;
int count = 0;
int flag=0;
int mode = 0;
int SW_count = 0;
int S1_Temp1,S1_Temp2,S1_Temp3,S1_Temp4,S1_Temp5;
int S2_Temp1,S2_Temp2,S2_Temp3,S2_Temp4,S2_Temp5;
float calibration_value = 7.0;

void calibrate_sensors()
{
while(celsius_1 != celsius_2)
  {
  if(celsius_1 < celsius_2)
  {
  calibration_value = calibration_value - 1.0;
  read_temperature();
  }
```

```
  if(celsius_1 > celsius_2)
  {
  calibration_value = calibration_value + 1.0;
  read_temperature();
  }
  }
}

void setup()
{
  Serial.begin(115200);
  Serial1.begin(115200);
pinMode(LED,OUTPUT);
  pinMode(SW,INPUT);
  thermistor1 = new
NTC_Thermistor(SENSOR1_PIN,REFERENCE_RESISTANCE,NOMINAL_RESISTANCE,NOMINAL_TE
MPERATURE,B_VALUE,STM32_ANALOG_RESOLUTION);
  thermistor2 = new
NTC_Thermistor(SENSOR2_PIN,REFERENCE_RESISTANCE,NOMINAL_RESISTANCE,NOMINAL_TE
MPERATURE,B_VALUE,STM32_ANALOG_RESOLUTION);
  stm.begin(9600);
}

void loop()
{
    DynamicJsonBuffer jsonBuffer;
    JsonObject& data2 = jsonBuffer.createObject();

  digitalWrite(LED,HIGH);
  if(digitalRead(SW)==LOW)
  {
  digitalWrite(LED,HIGH);
  Serial.println("Calibrating Sensors..");
  calibrate_sensors();
  Serial.println("Calibration Completed !!!");
  digitalWrite(LED,LOW);
  while(digitalRead(SW)==LOW);
  }
  read_temperature();
  sample_Array1[sample_Counter] = celsius_1;
sample_Array2[sample_Counter] = celsius_2;
  sample_Counter++;
  if(sample_Counter == 6)
  {
```

```
  for(int i=0; i<5 ; i++)
  {
  if(i==0)
  {
  S1_Temp1 = sample_Array1[i];
  S2_Temp1 = sample_Array2[i];
  }
  if(i==1)
  {
  S1_Temp2 = sample_Array1[i];
  S2_Temp2 = sample_Array2[i];
  }
  if(i==2)
  {
  S1_Temp3 = sample_Array1[i];
  S2_Temp3 = sample_Array2[i];
  }
  if(i==3)
  {
  S1_Temp4 = sample_Array1[i];
S2_Temp4 = sample_Array2[i];
  }
  if(i==4)
  {
  S1_Temp5 = sample_Array1[i];
  S2_Temp5 = sample_Array2[i];
  }
  }


  if(S1_Temp1 == S2_Temp1 && S1_Temp2 == S2_Temp2 && S1_Temp3 == S2_Temp3 &&
S1_Temp4 == S2_Temp4 && S1_Temp5 == S2_Temp5)
  {
  count++;
  Serial.println(count);
  }

  if(S1_Temp1 != S2_Temp1 && S1_Temp2 != S2_Temp2 && S1_Temp3 != S2_Temp3 &&
S1_Temp4 != S2_Temp4 && S1_Temp5 != S2_Temp5)
  {
  Serial.println(count);
  flag=0;
  count=0;
  }
```

```
  if(count == 5 && flag == 0)
  {
  flag = 1;
  }
  if(count == 5 && flag == 1)
  {
count = 0;
  }
  sample_Counter = 0;
  sample_Array1[20] = {};
  sample_Array2[20] = {};
  }
  delay(1000);

  data2["Resp"] = count;
  data.printTo(stm);
  jsonBuffer.clear();
}

void read_temperature()
{
  celsius1 = thermistor1->readCelsius();
  celsius2 = thermistor2->readCelsius();

  celsius1 = celsius1 - calibration_value;
  celsius1 = celsius1 + 26;

  celsius2 = celsius2 + 26;

  delay(500);
celsius_1 = celsius1;
  celsius_2 = celsius2;

}
```

```
//NodeMCU Code
#include <SoftwareSerial.h>
#include <ArduinoJson.h>
#include <ESP8266WiFi.h>
#include "ThingSpeak.h"

SoftwareSerial nodemcu(12,14);
SoftwareSerial stm(18,17);

const char *ssid = "Hotspot";
const char *password = "Password123";

unsigned long myChannelNumber = 1373125;
const char * myWriteAPIKey = "IVP1SA4RU6N0R";
const char* server = "api.thingspeak.com";

WiFiClient client;

void updateThingSpeak(float Bpm, float Spo, float TempC,float TempF, float latitude, float longitude, float
resp)
{
      ThingSpeak.setField(1,Bpm);
      ThingSpeak.setField(2,Spo);
      ThingSpeak.setField(3,TempC);
      ThingSpeak.setField(4,TempF);
      ThingSpeak.setField(5,latitude);
      ThingSpeak.setField(6,longitude);
      ThingSpeak.setField(7,resp);

      ThingSpeak.writeFields(myChannelNumber,myWriteAPIKey);
      delay(2000);
}

void setup()
{
   Serial.begin(9600);
   ThingSpeak.begin(client);
   WiFi.begin(ssid, password);

      while (WiFi.status() != WL_CONNECTED)
      {
            delay(2500);
            Serial.println("SEARCHING FOR WIFI");
      }
```

```
    Serial.println("");
    Serial.println("WiFi connected");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());
  nodemcu.begin(9600);
  delay(1000);
  stm.begin(9600);
  delay(1000);
}

void loop()
{
  DynamicJsonBuffer jsonBuffer;
  JsonObject& data = jsonBuffer.parseObject(nodemcu);

  DynamicJsonBuffer jsonBuffer;
  JsonObject& data2 = jsonBuffer.parseObject(stm);

   if (data == JsonObject::invalid())
  {
    jsonBuffer.clear();
    return;
  }

  Serial.print("Heart Beat : ");
  float Bpm = data["HeartBeat"];
  Serial.println(Bpm);
  Serial.print("Oxygen Saturation : ");
  float Spo = data["SpO2"];
  Serial.println(Spo);
  Serial.print("TemperatureC:");
  float TempC = data["TemperatureC"];
  Serial.println(TempC);
  Serial.print("TemperatureF:");
  float TempF = data["TemperatureF"];
  Serial.println(TempF);
  Serial.print("Latitude : ");
  String latitude = data["Latitude"];
  Serial.println(latitude);
  Serial.print("Longitude : ");
  String longitude = data["Longitude"];
  Serial.println(longitude);
  Serial.print("Breath Rate : ");
```

```
    float resp = data2["resp"];
    Serial.println(resp);

    if(client.connect(server,80))
      {
          updateThingSpeak(Bpm,Spo,TempC,TempF,latitude,longitude,resp);
      }
}
```