A Covid-19 viral transmission prevention system for embedded devices utilising deep learning

Mihai Penica
Electronic & Computer Engineering Department,
University of Limerick,
Ireland
Email: mihai.penica@ul.ie

Reenu Mohandas
Electronic & Computer Engineering Department,
University of Limerick,
Ireland
Email: Reenu.Mohandas@ul.ie

Mangolika Bhattacharya
Electronic & Computer Engineering Department,
University of Limerick,
Ireland
Email: mango.bhattacharya@ul.ie

Karl Vancamp
Electronic & Computer Engineering Department,
University of Limerick,
Ireland
Email: mihai.penica@ul.ie

Prof. Martin Hayes
Electronic & Computer Engineering Department,
University of Limerick,
Ireland
Email: martin.j.hayes@ul.ie

Dr Eoin O'Connell
Electronic & Computer Engineering Department,
University of Limerick,
Ireland
Email: eoin.oconnell@ul.ie

2021 32nd Irish Signals and Systems Conference (ISSC) | 978-1-6654-3429-4/20/$31.00 ©2021 IEEE | DOI: 10.1109/ISSC52156.2021.9467847

Abstract - The coronavirus pandemic (COVID-19) has created an urgent need for different monitoring systems to prevent viral transmission because of its severity and contagious aspect. This paper proposes design and implementation of a hardware-software solution that uses supervised machine learning algorithms to examine an individual and determine if he/she poses a viral transmission danger. The solution proposed was developed utilising an ARM embedded device along with different sensors to detect and monitor COVID-19 symptoms and, at the same time, to enforce wearing of a mask by using deep learning computer vision.

Index Terms - artificial intelligence, health monitoring, covid19 viral transmission, raspberry pi, COVID-19 symptoms

I. INTRODUCTION

The advancement of technologies today, especially in electronics and artificial intelligence, has made life much easier and simpler. But apart from making day to day life more manageable, these new technologies have a significant impact on the medical field.

In the current COVID-19 pandemic situation, and the absence of a known efficient treatment and the lack of vaccination available worldwide, there is a need for fast decisions on people isolation to reduce the spread of contagion [1].

The need to isolate an individual or deny access in a public space requires first determining if that person respects public health rules or presents the infection symptoms.

To determine whether an individual is infected or not, a person or a system should monitor the health parameters. Manual monitoring can sometimes be uncomfortable, and many people embrace mask-wearing and comply with public health rules. Some argue that wearing a mask does not prevent viral transmission and does not comply with respecting the health rules. Given today's facts, the health system is being overloaded due to the high number of coronavirus cases. The automated system proposed can gather data and determine if people respect the public health rules, and it is an excellent alternative to monitoring symptoms and health rules enforcement. All data collected is being stored in a cloud, and it is used to create a health profile for further analysis. Adopting an automated system like the one presented in this paper can relieve medical personnel's burden in monitoring compliance and symptoms related to the COVID-19 pandemic. This paper's proposed system is based on a Raspberry Pi embedded device used as the data collection and data delivery tool to gather information about an individuals' health and a deep learning computer vision system to detect his/her mask-wearing compliance.

One of the most common symptoms of COVID-19 infection is fever and respiratory problems [2]; that is why it was determined that the basic parameters to be measured are temperature, pulse rate, and oxygen level.

To determine body temperature, the MLX90614ESF-DCH Infra-Red Thermometer was used. The model used here is the medical version that has a high accuracy of 0.2 degrees Celsius in a wide temperature range. A SparkFun Pulse Oximeter and Heart Rate Sensor based on MAX30101 & MAX32664 chipsets were used as Pulse oximeter and to measure heart rate.

This paper describes the methodology and hardware components used to build the proposed system.

II. METHODOLOGY

Figure 1 presents a chart highlighting the process followed to develop and build the system proposed in this paper. This process contains the hardware solution design stage, the implementation and testing of the hardware solution, the software development process, the complete integration process between software and hardware, the design and training of deep learning networks and the integration and testing of software and hardware.

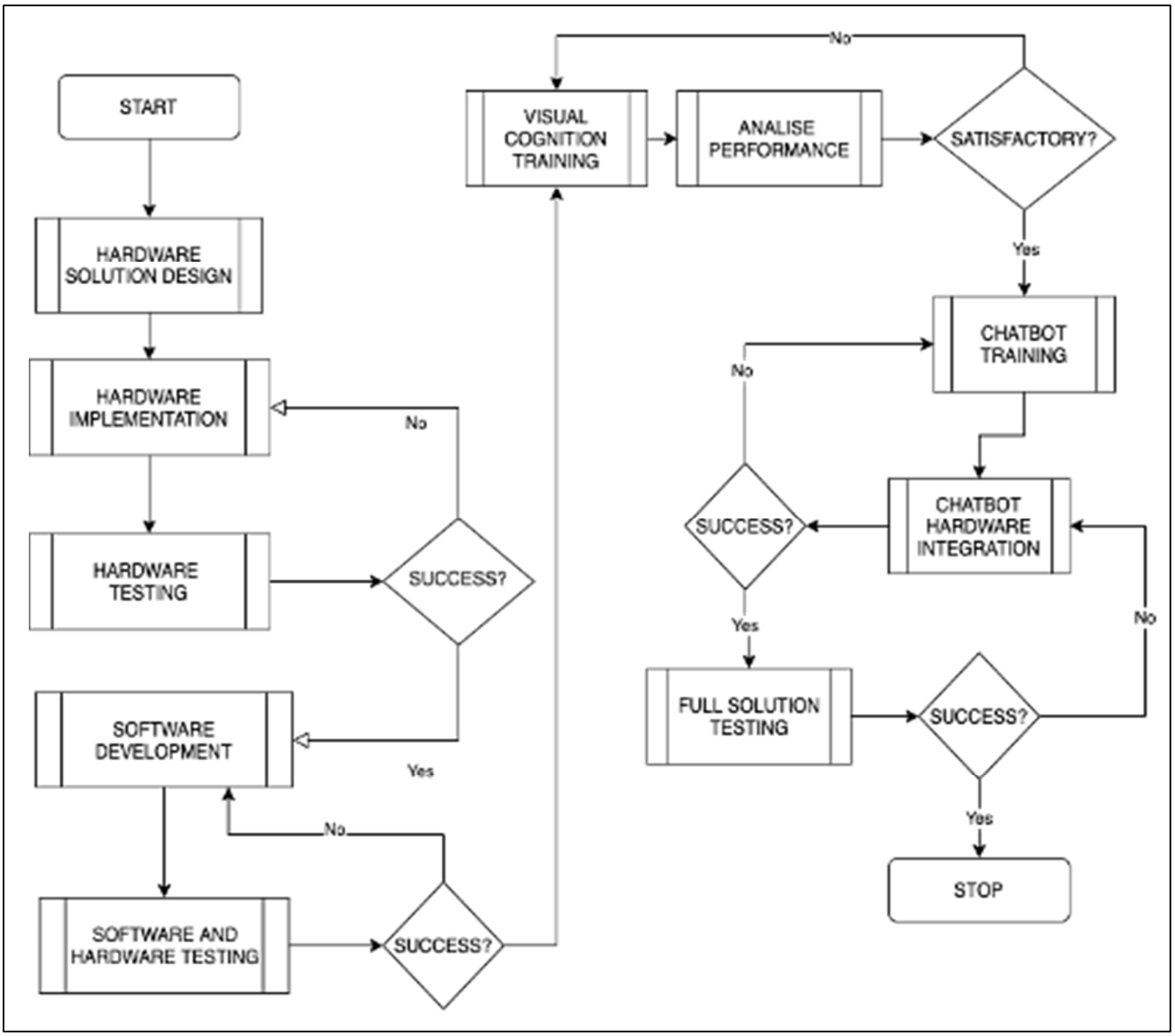

Figure 1 Methodology process

A. Hardware design

The hardware used for this prototype consists of: Raspberry

- Pi 4 Model B/4G has an ARM Cortex-A72 CPU powerful enough to sustain model training for the CAI and run deep learning detection and retrieve and manage the sensors' inputs.
- 7in Touchscreen display
- I2C-TO-39 non-contact infrared thermometer with a low noise 17-bit ADC and a powerful DSP unit allows the thermometer to achieve high accuracy and resolution of 0.14 Celsius degrees. It can measure temperature ranges: between -40˚C and +125˚C for sensor temperature and between -70˚C and +380˚C for object temperature.
- I2C - SparkFun Pulse Oximeter and Heart Rate Sensor (based on MAX30101 & MAX32664) for getting pulse and blood oxygenation
- Raspberry Pi, Camera Module, CSI-2 with 3280 x 2464 Resolution
- USB Stereo Audio Adaptor External Sound Card
- Arduino Nano Every
- 2 Speaker – 8ohm
- 1 External microphone

The Raspberry Pi acts as a bridge between the sensors and the cloud server. Its primary role is to collect all the measured data and send it to the cloud server for creating a health profile. Fig. 2 below shows the way the MLX90614 is connected to the Raspberry Pi board. The Arduino Nano board [4] retrieves the Sparkfun pulse oximeter data and forward it to the Raspberry Pi for processing. Fig. 3 presents the connection of the Sparkfun pulse oximeter to the Arduino board.

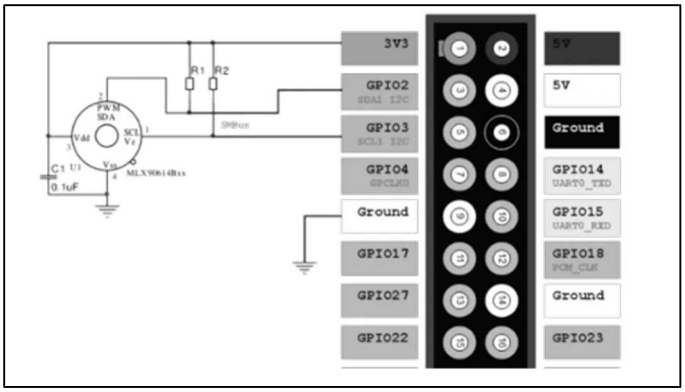

Figure 2 MLX90614 connection

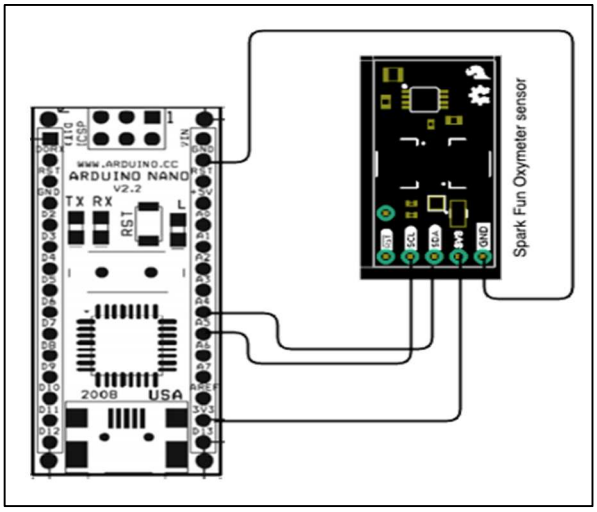

Figure 3 Arduino – SparkFun oximeter and heartbeat sensors

The SparkFun module will require the user to hold the thumb on the sensors to detect pulse and oxygen saturation in the blood; that is why after each usage, the sensor must be clean with an alcohol-based product to prevent further spread of infection.

The highest reading is considered as the guiding reference reading, which is recorded.

To clean the sensor after each use, a UV-C LED was installed in order to avoid viral transmission between the person using the device and the next person in line.

Still, because UV-C light is considered hazardous for the human body and can cause severe burns and eye injuries, it was disabled in this experiment.

The UV-C LED will be enabled in future experiments after developing safety mechanisms to cover the UV-C LED when running to prevent accidental radiation. Sensors are activated only on request by the CAI except for the temperature sensor that it is always running. Fig. 4 presents a block diagram of the hardware system modules interacting. All data from the sensor are stored first on the Raspberry pi storage card, and then if the server is available, it is uploaded to the cloud.

Data is stored in an encrypted way to be protected if a hacking or a security breach is attempted.

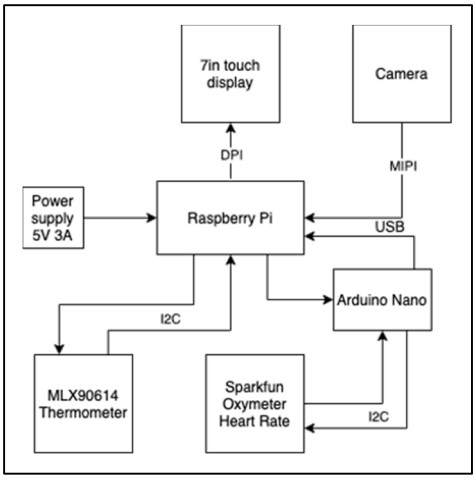

Figure 4 Hardware Design

A prototype of the system is shown in Fig 5, evidencing the hardware components presented in Fig. 4. Fig. 4 also shows the way the parts are installed, creating the system used for testing.

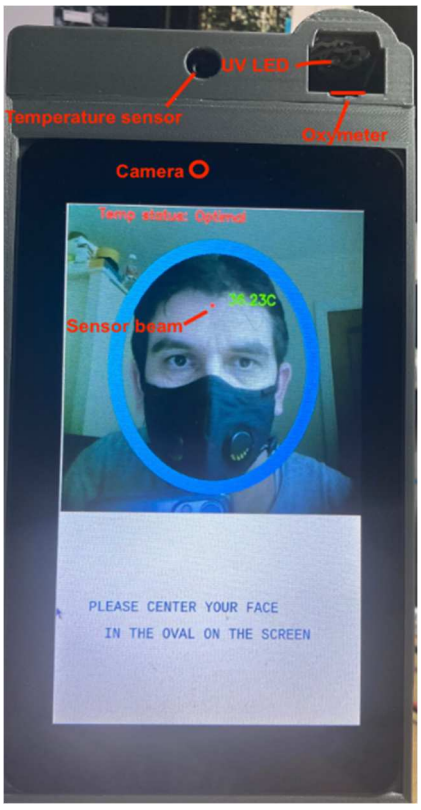

Figure 5 Prototype in use

B. Software design

As observed in Fig. 4, the hardware components proposed in this paper are used to assess if the person tested, presents COVID-19 symptoms. The software architecture proposed in this paper is modular-based, and it is intended to be scalable and provide new services and functionalities based on user needs.

The scalability is possible due to the modular software and hardware architecture that allows new sensors and software functionalities without changing the initially developed system. The developed architecture's logic is intended to retrieve necessary information from the users and sensors parameters readings and store it in the database for further processing. Based on the acquired data, the system can determine the risk of viral transmission and suggest the following action to be taken by the tested person showing infection signs. The modules included in the proposed architecture are intended to work independently and have different functions like hardware interaction, logging, model training, CAI interaction. The number of modules is scalable, and a new one can be implemented without affecting the existing ones. The system has three major components: the CAI user interface, the Data logging service, and the API services. The API services receive the message from the data logging system in a JSON schema and decide which module to call to get the correct response data. Fig. 6 presents an overview of the modular infrastructure proposed for the platform.

An advantage of this modular infrastructure is that it is scalable, and any new module can be developed and tested without interfering with the other services. Also, a new type of sensor can be attached without the requirement to restart the services.

The communication between the Raspberry Pi Data logging system and cloud API is being undertaken using the HTTPS protocol and a VPN tunnel for extra security.

The CAI interfaces and sensor interactions that reside on the Raspberry Pi were developed using python, an object-oriented high-level programming language. For the API system and the dashboard interfaces, we used PHP, a server-side scripting programming language. For the dashboard user interface, the typescript Angular 8 framework is used. All these interfaces for API and dashboard interface.

1) NLP CAI training and results: The CAI can be defined as a software program that can maintain a coherent conversation with humans using natural language. In this section, it will be shown how the dataset was created using the designed web interfaces. The model is trained on a custom dataset built on request by using the developed web interfaces. The model training is web-based, relying on TensorFlow.js and Nodejs. The model adheres to the bag-of-words type that uses the collected data to design vocabulary and create binary vectors to provide scoring based on the number of times and frequency of the word's occurrence. The system presented in this paper consists of two stages, Training and Results.

a) CAI Training: Because the hardware and software are specifically designed to harvest COVID-19 data, a web interface for building the dataset that will feed the Seq2Seq [6] model is available for training. The sentences are preprocessed by lower casing, removing punctuation, question marks, or other characters, and tokenised.[6]. A Seq2Seq model using two-layered LSTM as encoder and decoder is used. The former encodes the input sentence into a context vector while the later decodes the vector into the target one, which is the prediction or response to the input [5]. A dataset of 100 questions and answers about COVID19 symptoms has been used to train the system.

b) Results: Although the dataset may be considered unsatisfactory for NLP CAI training results, the purpose of this paper is to demonstrate the usage of an NLP CAI together with hardware elements to prevent viral transmission, and it does not focus on the model training. Further research in this area is going to be conducted in the future.

The summary of the model is presented in Fig. 7, with 59667 parameters evaluated during training. The model was optimised using adam optimiser, and it was executed for 40 epochs with a batch size of 8. In this case, the network was overtrained. The loss of the model was calculated with categorical cross-entropy loss, and it was recorded that after epoch 15, the loss decreases and the accuracy increases; the maximum accuracy was reached at epoch 24.

The model's performance was measured using a BLEU score that measures the similarity between the model's output and the human reference sentences [7]. The BLEU score is in the interval of [0, 100]. The higher the BLEU score, the more the output correlates to the human reference sentences. A BLEU score of 30 or more is commonly considered to have a good quality that highly correlates to the human reference sentences.[6]

In our testing scenarios and based on the small dataset's evaluation, a BLUE score of 46.72 was achieved, which is considered acceptable for this experiment.

2) Mask detection training and results: This section's main objective is to explain how MobileNet CNN was used to classify and determine if the person in front of our device is complying with the public health rules by wearing a mask. The SSD is a feed-forward based convolutional network that produces a fixed-size collection of bounding boxes and scores for the presence of object class instances in those boxes [10]. There are two essential components for SSD: [10] The first one is that it progressively reduces the volume size in deeper layers the same way as a standard CNN. The second one is that each of the CONV layers connects to the final detection layer. That is a very important aspect because it allows the detection and localisation of objects at different scales.

a) Mask detection algorithm training: The most time-consuming job for any object detection algorithm is to create a training dataset. We labelled 1135 pictures containing faces of people wearing masks and 1220 pictures of people not wearing a mask. Our dataset has 231 pictures of people with mask-wearing glasses, 452 pictures of people with mask-wearing a cap, and 125 pictures of people with mask-wearing both. The rest of the images represents people not wearing any of them. Results about the training and performance of the model are going to be published in a forthcoming paper.

b) Results: To determine the system's performance and mask detection success, ten volunteers with different looks were asked to use the system and record the results. Each volunteer had to execute ten different scenarios and wear various accessories, like wearing a cap or wearing glasses.

Table 1 depicts the testing and the system's results detecting people wearing masks in each situation, like wearing only glasses, wearing only a cap wearing both or wearing none.
All tests were performed using the prototype developed using the methods described in this paper.

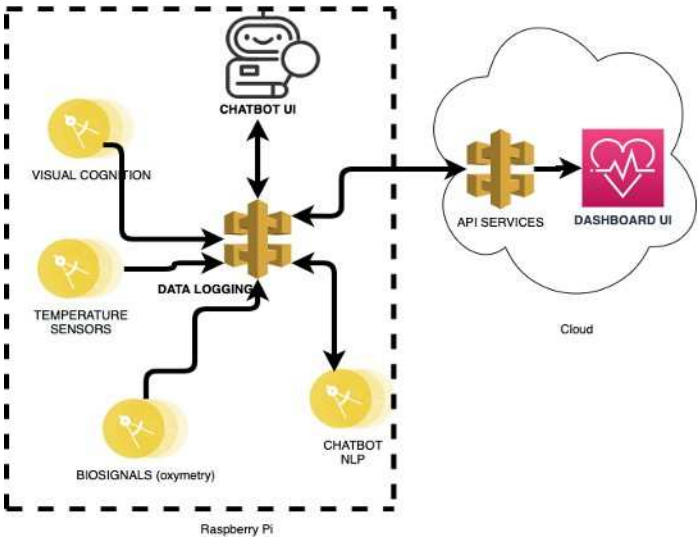

Figure 6 Overview of the software architecture

Layer (type)	Output shape	Param #
dense_Dense1 (Dense)	[null,256]	24320
dropout_Dropout1 (Dropout)	[null,256]	0
dense_Dense2 (Dense)	[null,128]	32896
dropout_Dropout2 (Dropout)	[null,128]	0
dense_Dense3 (Dense)	[null,19]	2451

Total params: 59667
Trainable params: 59667
Non-trainable params: 0

Figure 7 Model details

TABLE I
Detection Results

No	Test subject	Not wearing anything	Wearing glasses	Wearing cap	Wearing glasses & cap
1	Person1	93%	84%	82%	88%
2.	Person2	89%	88%	75%	80%
3.	Person3	82%	78%	70%	75%
4.	Person4	85%	72%	75%	79%
5.	Person5	90%	78%	72%	82%
6.	Person6	92%	75%	77%	82%
7.	Person7	93%	83%	78%	85%
8.	Person8	95%	92%	89%	93%
9.	Person9	88%	74%	79%	87%
10.	Person10	83%	75%	81%	81%

It was determined that the reason of low detection accuracy among people wearing glasses and cap is that the training data contains fewer pictures of people wearing both glasses and caps.

3) Data Collection: The data collection flow is presented in Fig. 9. The process starts with activating the temperature sensor and displaying instruction on how the user should be positioned. The user is asked to place his face in the oval on the screen, and the system starts collecting readings from the thermometer. The highest reading is considered as the guiding reference reading, which is recorded. If the temperature is below 35°C, the process will not continue. The CAI will inform the user that there is an error with the readings and ask him to follow the instructions on the screen and position himself accordingly. It is imperative to detect if the device is indeed a person or a picture in front of the device, so we use the temperature sensor to check if that is the case and start the mask detection code.

By positioning the face into the oval on the screen and filling it, the user is placed at 30 cm from the temperature sensor giving high accuracy on the readings.

The second step in the process that occurs after determining that the temperature is over 35.0 °C is to start the mask detection process. If the mask is detected, the system sends the data to the cloud and continues the process. The blood oxygenation level is read. Low Spo2 level and fever level can be inputted using the cloud interface. The settings are then downloaded to the device internal storage using the API's and used as computational reference values. If the oxygen level is lower than 90% or lower than the cloud server's settings, the user is informed about his health condition, and he/she is advised to look for medical advice as soon as possible. Simultaneously, options are displayed on the screen, and those give the user the possibility to retake the oxygen test. If the oxygen level value continues to stay low, a message to seek medical attention and 'access denied' is displayed on the screen.

When the temperature is over 37.9°C, but the oxygen level is over 90%, the system will display a list of symptoms asking the user to identify if he/she has experienced any of those. If the answer is yes on any of them, the CAI will save the information into the database. Symptoms indicated by the user are sent to the cloud along with bio readings for data analysis. The user is advised to look for medical advice as soon as possible, and an 'access denied' message is displayed.

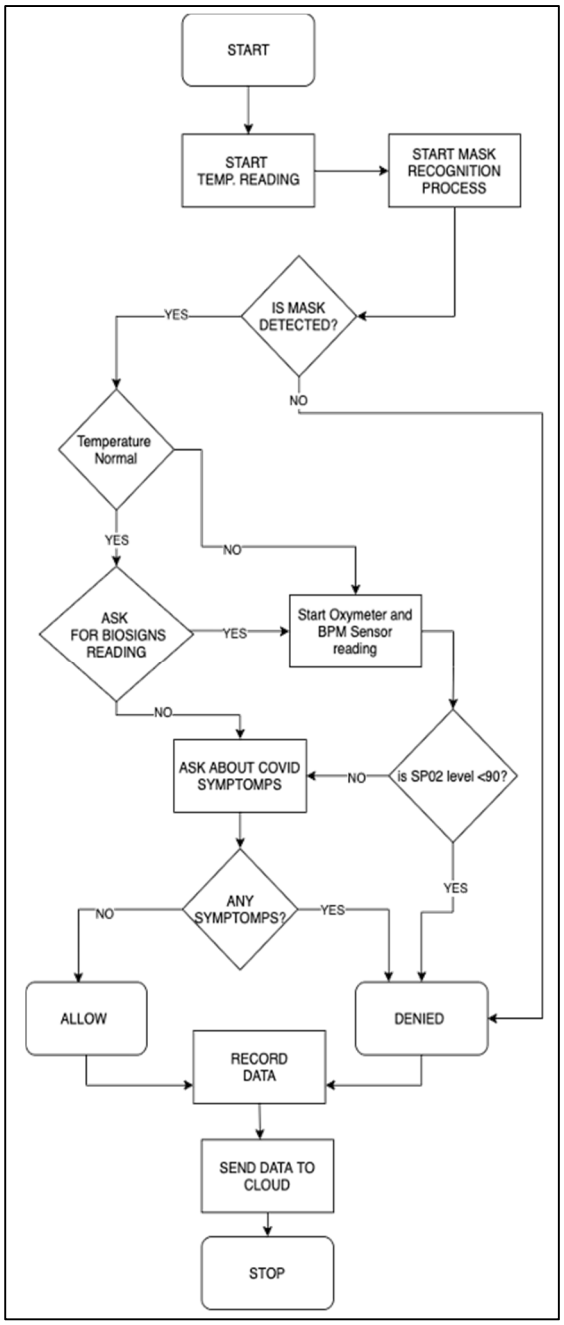

Figure 8 Data collection

C. Full solution Testing

As shown in Fig. 1, there are three testing stages: Hardware testing, Software testing and Full solution testing.

1) Hardware testing: The first test is the connectivity from the IR MLX9014 thermometer to the Raspberry Pi and the accessibility of the IR thermometer from the Raspberry Pi. The fact that the thermometer can read the environment temperature indicates that it is working correctly. To test the accuracy of the IR MLX9014 thermometer, a digital medical thermometer is used as a comparison to get the error rate. The measurements are obtained in a closed room with an air temperature of 25°C and a humidity of 35%. The second test that is performed is with the SparkFun Pulse Oximeter and Heart Rate Sensor. As in the same case as the IR thermometer, the connectivity is tested by checking if the red LED is on, and data can be received onto the ARDUINO board. The third testing done is the connectivity between the ARDUINO BOARD and the Raspberry Pi board via USB. In this case, the connectivity is checked by sending some sample data from the ARDUINO board, and it is tested if that data is received correctly into the Raspberry Pi. Each test is repeated 30 times for each sensor to ensure no discontinuity lies in the data transmission.

2) Complete solution software testing: Once the hardware testing is performed, the data retrieval is tested by manually calling the APIs and checking the sensors' returned data. If received data was recorded correctly into the database and the cloud API returns a correct response, the testing is considered a success. The next stage of testing is performed.

3) Complete solution testing: After the CAI training is completed, full solution testing is required. Therefore, a comprehensive testing scenario that simulated a real person data acquisition is performed. The CAI interacted with a volunteer asking the prescribed question and completing the required testing. Temperature measurement, blood oxygen level and heartbeat measurement, along with data recording about COVID19 symptoms, are tested. The test performed measures the accuracy of the sensors, data acquisition, and data reporting.

4) Complete solution results: a) Temperature data retrieval results: The retrieval of the body temperature is performed to determine the person's condition. Fever over 38°C is one of the signs of COVID 19 infection, and therefore it was needed to benchmark the MLX90614 sensor to determine the error differential (or offset). The accuracy of the MLX sensor was determined by comparing the MLX readings against the readings obtained from a Fluke 561 thermometer [11]. The results are presented in Table 2.

TABLE II
ERROR RATE TEMPERATURE

#	NAME	Body Temperature (°C) MLX90164 THERMOMETER	DIGITAL THERMOMETER	DIFFERENCE (°C)	ERROR RATE %
1	TEST 1	36.9	37.10	-0.2	0.5391
2	TEST 2	36.1	36	0.1	0.2778
3	TEST 3	36.9	36.8	0.1	0.2717
4	TEST 4	36.7	36.5	0.2	0.5479
5	TEST 5	37.2	37	0.2	0.5405
6	TEST 6	36.4	36.2	0.2	0.5525
7	TEST 7	36.7	36.8	-0.1	0.2717
8	TEST 8	37	36.6	0.4	1.0929
9	TEST 9	36.8	36.7	0.1	0.2725
10	TEST 10	37.1	36.9	0.2	0.5420
AVERAGE		36.78	36.66	0.12	0.3273

b) Oxygen level results: Radiographic or CT scans revealed that pneumonia and lung infections are some of the coronavirus symptoms; by measuring the SpO2 level in the blood, respiratory problems can be detected in a subject. To determine the accuracy of the SparkFun Pulse Oximeter, the readings were compared with the readings taken by a medical pulse oximeter [12] and are presented in Table 3.

c) Cloud storage results: After the data gathering process is finished and the user answered all questions about symptoms and all readings were taken; the next step of sending the data to the cloud server is performed.

TABLE III
ERROR RATE OXYGEN LEVEL

#	NAME	Oxygen Level (%)		ERROR RATE %
		SparkFun Oximeter	Medical Oximeter	
1	TEST 1	94	97	3.0928
2	TEST 2	98	97	1.0309
3	TEST 3	97	97	0.0000
4	TEST 4	98	97	1.0309
5	TEST 5	97	98	1.0204
6	TEST 6	96	97	1.0309
7	TEST 7	97	97	0.0000
8	TEST 8	95	95	0.0000
9	TEST 9	96	97	1.0309
10	TEST 10	96	96	0.0000
AVERAGE		96.4	96.8	0.4113

It can be observed that temperature and oxygen levels are recorded as well as the symptoms answers. A few tests were done following a written testing scenario to verify if the data is recorded correctly in the cloud, and the results have been a success. To perform a complete scenario testing where the user's oxygen level is being taken, the temperature threshold for triggering the oximeter hardware was lowered to 35 degrees Celsius.

The data recorded in the cloud are:

- User photo containing a label with accuracy detection.
- The location where the device is installed
- Date and time when the test was performed
- Temperature recorded
- Oxygen level recorded
- Pulse recorded
- Mask compliance

Figure 9 USER RECORDED DATA TO CLOUD

III. DATA SECURITY AND PRIVACY

At this moment, there are a lot of requirements on data security and data privacy due to regulations and ethical guidelines. Ethical guidelines are to ensure the responsibility of collection and analysis of data for any purpose.[13]

A summary of possible risks that the system poses and possible solutions to understand the privacy risks when collecting data.

1. Risk: Data may contain personal information, and it is subject to EU – GDPR

Possible solution:

It is advisable to get consent from the person using the device. Use the device in private areas.

2. Risk: Using 'all the data' for 'unknown purposes'

Possible solution: Limit access to user data to a reasonably necessary level.

3. Risk: Hacking Risks

Possible solution: Take proper security actions like data encryption, network security, hardware security.

There are three relevant security techniques that we have used while performing this research regarding security and risks.

These technics are:

- Cryptographic security
- Access control security
- Network Security

Cryptographic security: Cryptography [14] is an essential data security technique against interception, tampering, and unauthorised reading.

The development of the system presented in this paper focused on engineering the functionality while maintaining security and data protection.

This principle works by acquiring data from the sensors, encrypting the data using cryptography with an OTP algorithm unique for each acquisition time, then splitting it and fragmenting it using different micro-services to keep the data safe. The key needed to decrypt the data sits on the physical device itself, and it is formed from the unique hardware configuration. The key is being transmitted to the cloud server for each record in the database if there is a need for information. Another security method implemented in this case is that one packet of data is fragmented into two or more pieces making the decryption harder or almost impossible. The only way of decrypting it is by using the hardware key residing on the device itself.

If a data breach occurs, the databases' information is expected, and the user details are not readable. Another level of encryption is applied when the data is saved into the database. In this case, the key resides on a separate server and is available on request. By performing all this data encryption and fragmentation, the user data is kept safe in any situation.

Access control security – The system presented in this paper uses an encryption mechanism where part of the decryption key is kept on the Raspberry pi (the device itself). The device must be physically secured from allowing any external connection to it. Because of this necessity, all external ports were covered and disabled.

Network security - Network security maintains the security and the privacy of the data-in-transit transmitted from the device to the cloud server and from the cloud server to the device. It is maintained through security protocols and

standards such as Secure Socket Layer (SSL), Transport Layer Security (TLS), Secure HTTP, secure IP (IPsec), and Secure Shell (SSH). TLS and SSL provide transport-level security [16].

A VPN connection was created between the Raspberry pi and the cloud network. On top of the VPN network, the Raspberry pi uses secure HTTP to transfer its data to the cloud server. An IPTABLES firewall was activated to protect the device itself from external attacks.

Rules to block all incoming ports were in place to stop any connection attempt if the WIFI network was breached.

IV. CONCLUSION

This paper has demonstrated that it is possible to build a low-cost device capable of running deep learning models at the edge in a biometric testing use case. By accurately checking for proper mask wear in tandem with biometric testing, visual cognition system and a CAI that is informed by sensor feedback, it has been demonstrated that an early detection mechanism for infection prevention is possible. The research also revealed that a non-intrusive method to retrieve medical data is possible using ARM devices and low cost sensors.

The approach is standalone and the demonstrated sensor error rates and cognition model weighs using such a low power ARM device shows that cloud intervention is not required to make fast decisions.

It has also been demonstrated that it is possible to train and run a low weight model on an embedded device; in our case, the NLP CAI training is being done directly on the raspberry pi. Also, the mask detection model resides on the Raspberry pi. Another aspect demonstrated in this paper was that it is possible to gather, store and transfer sensitive data securely by splitting and fragmenting it using embedded devices to prevent any privacy issues in a breach.

V. ACKNOWLEDGEMENT

The authors would like to thank the staff of the Electronic and Computer Engineering Department at the University of Limerick for their assistance. The authors would also like to acknowledge resources made available from Science Foundation Ireland through the grant award (16/RC/3918) to Confirm Centre for Smart Manufacturing.

VI. REFERENCES

[1] Md. Martuza Ahamad, Sakifa Aktar, Md. Rashed-Al-Mahfuz, Shahadat Uddin, Pietro Liò, Haoming Xu, Matthew A. Summers, Julian M.W. Quinn, Mohammad Ali Moni, A machine learning model to identify early-stage symptoms of SARS-Cov-2 infected patients, Expert Systems with Applications, Volume 160,2020,

[2] Sijia Tian, Nan Hu, Jing Lou, Kun Chen, Xuqin Kang, Zhenjun Xiang, Hui Chen, Dali Wang, Ning Liu, Dong Liu, Gang Chen, Yongliang Zhang, Dou Li, Jianren Li, Huixin Lian, Shengmei Niu, Luxi Zhang, Jinjun Zhang,Characteristics of COVID-19 infection in Beijing,Journal of Infection,Volume 80, Issue 4,2020,Pages 401-406

[3] MLX90614 family datasheet. https://www.melexis.com /-/media/files/documents/datasheets/mlx90614-datasheet-mele xis.pdf (last accessed 2020/08/21)

[4] Arduino nano board https://components101.com/microc ontrollers/arduino-nano (last accessed 2021/02/14)

[5] Oriol Vinyals, Q.V.L.. A neural conversational model. Proceedings of the 31 st International Conference on Machine Learning 2015; Sequence to Sequence Model Performance for Education Chatbot

[6] Yogi Wisesa Chandra, Suyanto Suyanto,Indonesian Chatbot of University Admission Using a Question Answering System Based on Sequence-to-Sequence Model,Procedia Computer Science,Volume 157,2019,Pages 367 -374

[7] Kishore Papineni Salim Roukos, T.W.W.J.Z.. Bleu: a method for automatic evaluation of machine translation. Computational Linguistics (ACL) 2002,Pages 311–318.

[8] Liu W. et al. (2016) SSD: Single Shot MultiBox Detec- tor. In: Leibe B., Matas J., Sebe N., Welling M. (eds) Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science, vol 9905. Springer, Cham.

[9] Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Application

[10] Debojit Biswas, Hongbo Su, Chengyi Wang, Alek- sandar Stevanovic, Weimin Wang,An automatic traffic density estimation using Single Shot Detection (SSD) and MobileNet-SSD,Physics and Chemistry of the Earth, Parts A/B/C,Volume 110,2019,Pages 176-184

[11] Fluke 561 Multipurpose Thermometer http://www.farnell.com/datasheets/20887.pdf (last access 14.02.2021)

[12] CMS50D1 Pulse Oximeter https://docs.rs-online.com/0c84/A700000006918263.pdf (last access 14.02.2021)

[13] Chandra Thapa, Seyit Camtepe,Precision health data: Requirements, challenges and existing techniques for data security and privacy,Computers in Biology and Medicine,Volume 129,2021

[14] Handbook of applied cryptography A.J. Menezes, P.C. van Oorschot, S.A. Vanstone, in: Handbook of Applied Cryptography, fifth ed. ed., CRC Press, 2001.A.J. Menezes - Google Academic