Evolution in Electrical and Electronic Engineering Vol. 2 No. 2 (2021) 605-614

EEEE

Homepage: http://publisher.uthm.edu.my/periodicals/index.php/eeee
e-ISSN : 2756-8458

COVID-19 Mandatory Self-Quarantine Monitoring System

Tan Zhan Feng[1], **Wan Nurshawani Wan Zakaria**[1*]

[1]Faculty of Electrical and Electronic Engineering,
Universiti Tun Hussein Onn Malaysia, Batu Pahat, 86400, Johor, MALAYSIA

*Corresponding Author Designation

DOI: https://doi.org/10.30880/eeee.2021.02.02.072
Received 06 July 2021; Accepted 03 August 2021; Available online 30 October 2021

Abstract: In response to rising COVID-19 cases, the Malaysia government imposed multiple measures to prevent virus transmission. Among the efforts taken by MoH were the enforcement of a-14 day self-quarantine for PUI, PUS and positive Covid-19 patients without symptoms to curb local transmission. However, the violations of self-quarantine are still being reported due to the location data of the users that cannot be tracked by the available system. Thus, this project proposes a system to monitor and track the self-quarantine PUI and PUS in real time by implementing geolocation tracking and face recognition. The geolocation tracking is developed to monitor users to stay in their quarantine place within 30 meters radius, and prevent unauthorized movement by scanning the users' faces every two hours. User also requires updating their health status every day in the proposed system. If the user is classified as a very high risk dependent, admin and user will receive an alert notification for further action. The test result shows the developed map API can track multiple people with accuracy of 99% and the location of users can be displayed in real time in the map. In addition, the system can detect the user's face accurately in various conditions within 1 second. As a result, this system could be used by the authority in order to reduce the self-quarantine violation rate in which subsequently help to prevent the spread of COVID-19.

Keywords: COVID-19, Mandatory Self-Quarantine System, Geolocation Tracking System, Face Recognition System

1. Introduction

Nowadays, Coronavirus disease (COVID-19) is affecting 215 countries and regions worldwide. COVID-19 is defined as an infectious disease caused by a newly discovered coronavirus [1] [2]. Presentations of COVID-19 have ranked from asymptomatic symptoms to serious illness and death which can grow from 2 days to 2 weeks after exposure to the virus. According to current evidence, the COVID-19 virus is primarily transmitted between people through respiratory droplets and contact routes. Droplet transmission occurs when a person is in close contact with someone who has respiratory symptoms and is therefore at risk of having mucosae or conjunctiva exposed to potentially

*Corresponding author: shazwani@uthm.edu.my

infective respiratory droplets. Transmission may also occur through fomites in the immediate environment around the infected person. Therefore, positive Covid-19 patients, PUI and PUS should undergo self-quarantine to prevent the COVID-19 virus from spreading to others [3].

During the self-quarantine period, positive Covid-19 patients, PUI and PUS need to refrain from any contact with other individuals. For PUI and PUS, the self-quarantine should last for 14 days to allow time to confirm that the person is not infected. While positive Covid-19 patients are required to undergo self-quarantine until recover. However, self-quarantine is also difficult to implement because the quarantine period is quite long and people need requirements to live. For example, the whole Sivaganga cluster was started by a single person who ignored his home quarantine order. This non-adherence could be due to few factors, and one of them is due to the manual monitoring method. Therefore, COVID-19 Mandatory Self-Quarantine Monitoring System is needed to monitor positive Covid-19 patients, PUI and PUS staying in the quarantine place and refrain from any contact with other individuals.

To date, there are several existing systems that has been developed by different countries to monitor Covid-19 quarantine conditions such as MySejahtera, MyTrace, TraceTogether, Quarantine-Report•Self-Check, and StayHomeSafe. MySejahtera, MyTrace, and TraceTogether are not suitable for monitoring a person who undergoing self-quarantine at quarantine location. They work as a precaution to prevent users attend to hotspot of COVID-19, and inform the users if the users have been exposed to COVID-19 patients with close contact. By using Quarantine-Report•Self-Check, PUI and PUS must submit their health status every day for 14 days from the date of entry. Quarantine-Report•Self-Check will monitor the health status of users every day, while it does not monitor the location of users. StayHomeSafe is the only application in conjunction with wristband. The scanning QR code on the wristband in random time is a way to prevent users left their quarantine place without brings their phone.

Since the positive Covid-19 patients are monitored by health worker, so this project is primarily concerned on PUI and PUS. This project proposes an intelligent-based COVID-19 Mandatory Self-Quarantine Monitoring System. WiFi and GPS must be turned on, so users will receive an alert message from the system if the distance between users and their quarantine place exceeds 30m. Users are required to scan their faces to verify themselves every 2 hours in the period 8 a.m. until 10 p.m. and then the face recognition system will verify the identity of users [4]. The final phase of the project is the development of application system which can be installed on the smartphone. The system will monitor the users and track for the quarantine period. The overview of the project flow is illustrated in Figure 1.

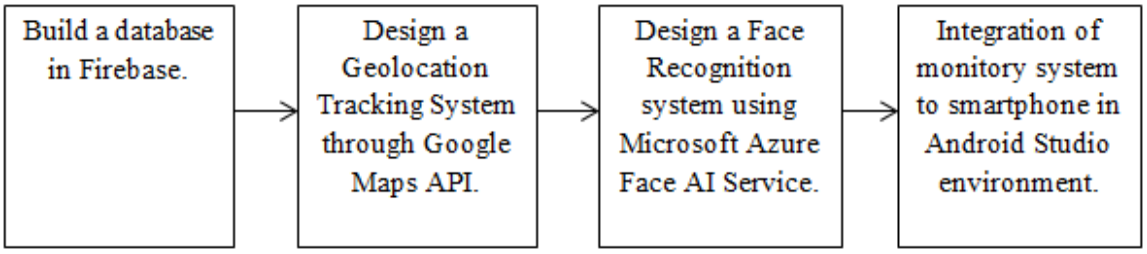

Figure 1: Overall Project Flow of COVID-19 Mandatory Self-Quarantine Monitoring System

2. Materials and Methods

2.1 Implementation of COVID-19 Mandatory Self-Quarantine Monitoring System

A COVID-19 Mandatory Self-Quarantine Monitoring System is designed as a software application that can be installed on the smartphone. Firebase is used to create a real-time database that delivers intelligence built-in, and the coding in the Java Language is built by using Android Studio.

The location of users is always showed in the Google Map on the monitoring system and the history of the location is saved to the database [5]. By using the free Google Maps API for tract visualization, there was no need to develop a map solution [6]. Hence, the host can monitor the live location of users in any time. The Azure Face service is used to provide AI algorithms that detect, recognize, and analyze human faces in images. Users will receive an alert message when they are going out from a quarantine place or have not scanned their face on time. Figure 2 shows the block diagram of COVID-19 Mandatory Self-Quarantine Monitoring System

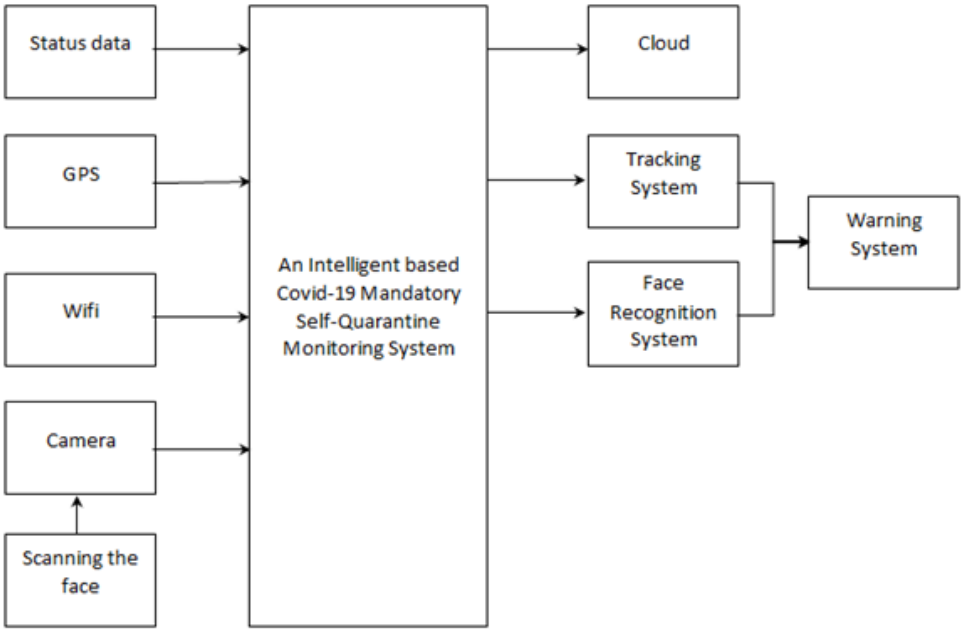

Figure 2: Block diagram of COVID-19 Mandatory Self-Quarantine Monitoring System

2.2 Design of the database in Firebase

The Firebase Real-time Database is a cloud-hosted database. Data is stored as JSON and synchronized in real-time to every connected client. Firebase Real-time Database lets developers store and sync data in real-time. This makes developers easy to access their data from any device, Web, or mobile, and it helps users collaborate with one another. When the data is updated, it stores the data in the cloud and simultaneously notifies all interested devices in milliseconds. The database can be structured by the developer, and the developer can specify who has access to what pieces of data. In the proposed system, users are specified to access only their own data. When the users lose their connection, the database STK will serve and store the changes. Users should update their health status to the database every day in the proposed system.

2.3 Development of Geolocation Tracking using Google Map

Google Maps is a web mapping service developed by Google. Google Maps provides the maps API and Geolocation API service. Maps API can display maps as images or interactive maps in system. Markers are the fundamental way to show the live location and quarantine location of users on a map. By using Geolocation API, the current location of the user can be tracked and display on the map. The latitude and longitude of location can be obtained and save to the database then the distance between two marker points is calculated.

2.3.1 Accuracy of Tracking Location

Accuracy of Tracking Location is the ability of the system to measure the accurate value of the user's location. The higher accuracy shows the closer measured location to the actual location of the

device. The accuracy of tracking location can be affected by many conditions like high buildings, under tree cover inside tunnels, atmosphere delay, and Ionospheric delay [7]. With Maps API, Android Developers provide a data class representing a geographic location. Location data are generated by the Location Manager and guaranteed to have a valid latitude, longitude, and accuracy [8]. The accuracy also can be calculated by using Eq.1:

$$Accuracy = \frac{Measured\ value - Actual\ value}{Actual\ value} \qquad Eq.1$$

2.3.2 Proposed Methods to Improve Accuracy

To improve positioning accuracy, the following two issues has been considered:

a. Determining a more accurate reference point.

b. Reducing rounding errors using a coordinate translation process.

GPS signals may be blocked by a signal obstruction. To obtain the reference point, we use long-term averaging, which is performed on the fixed position data in order to determine the reference position of that receiver [9]. The latitude and longitude of the user are collected over a long duration, and then the long-term average value of latitude and longitude are calculated by using Equation (2):

$$AVG_x = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad AVG_y = \frac{1}{N}\sum_{i=1}^{N} y_i \qquad Eq.2$$

where x is the latitude, y is the longitude, and N is the number of timestamps.

Rounding errors occur when the infinite real numbers are converted into finite numbers, and then cause the loss of data when rounding value [10]. The data calculation in GPS generally uses values in a decimal form converted from real data, which induces rounding error. To reduce rounding errors, coordinate translation is used and presented in degrees, minutes, and seconds based on the global coordinate center [0, 0]. Coordinate translation relocates the coordinate center from [0, 0] to any required local coordinate center. Therefore, all positions in the current navigation area are calculated relative to the local coordinate center.

2.4 Development of Face Recognition by using Microsoft Azure

Face recognition is the main biometric used by human beings [11]. In automatic face recognition, it is required to either identify or verify one or more persons in still or video images of a scene by means of a stored database of faces [12]. A face recognition system needs to identify or verify one or more persons found in a still or video image using a stored database of faces as accurately and as quickly as possible [13].

The Azure Face service provides AI algorithms that detect, recognize, and analyze human faces in images. Microsoft Azure Face API offers advanced algorithms that detect human faces in digital images. The Verify operation takes a face ID from the detected face and determines whether they belong to the same person. The machine-learning-based predictions of facial features are used to recognize a face by analyzing age, gender, emotion, pose, smile, and facial hair, in addition to 27 other landmarks for each face identified in the given image [14].

There are several factors that will affect the result of face detection in this proposed system [15]. For the example, resolution of the image, the image format, and the face angle of the camera. These factors will lead to failure of the Face Recognition System. Hence, the limitations should be found out by testing the implementation of the proposed system. First, the range of acceptable resolution of Azure Face API is required to find out. Second, the different types of image formats should be used as the input to test the face detection. Last, the users' faces should be identified by changing the yaw angles from the camera.

2.5 Design of self-monitoring algorithm by using Android Studio

Android Studio is a fully integrated development environment that provides a unified environment for their users to build applications for Android phones. Java Language is used to develop algorithm by using Android Studio. Figure 3 shows the flow of coding in the program.

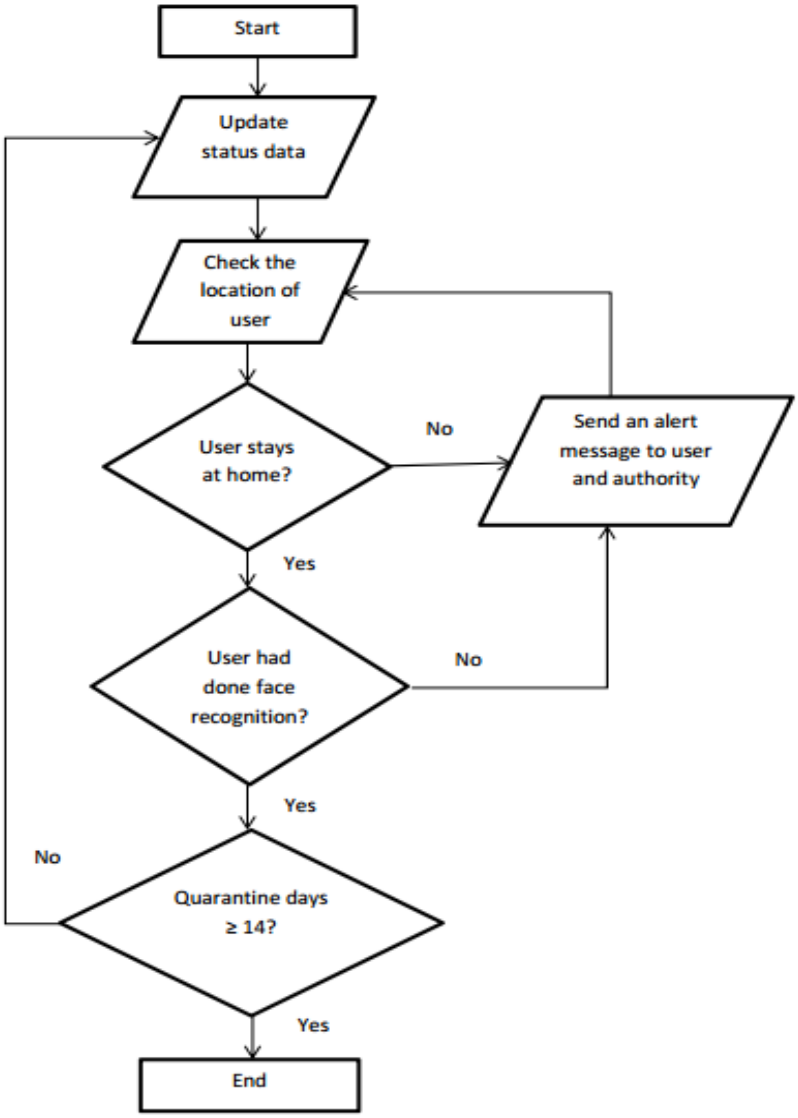

Figure 3: Flow Chart of Self-monitoring algorithm

2.6 Users interface of the system

There are two type of users in the system which are admin who monitoring PUI and PUS by using proposed system, and user who being monitored by admin. Hence, the interfaces of admin and user are different, they should login with their respective accounts. Figure 4 shows the main interface of admin and user. In the main interface of admin, there are a map, offenders button, logout button, and user list. Admin can choose the users' IC in the user list to show their current location and quarantine location on map. When admin clicks the offenders button, offenders' IC are shown in the offenders list, and double click the IC to know the detail of offenders. In the main interface of user, there are update, snap, and logout buttons. User must update their health status every day by answering all questions in the main interface. The snap button is used when user is required to verify identity by scanning face.

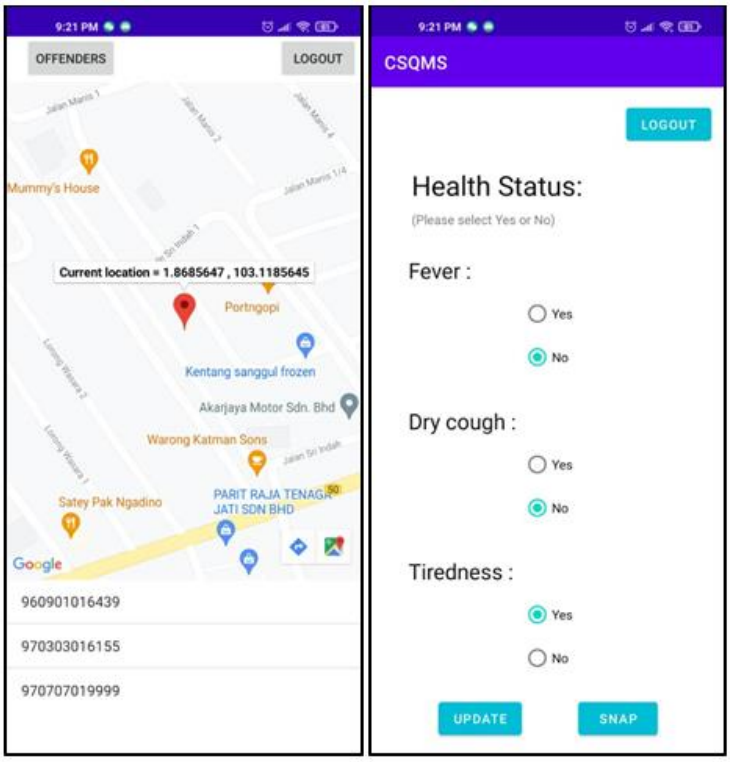

Figure 4: Main interfaces of admin and user

3. Results and Discussion

3.1 System Latency Test for Geolocation Tracking System

The proposed system is developed to track multiple people at the same time. There is no limitation on the number of users in the proposed system. However, when the number of users increases, more information is stored in the database at the same time which will make a delay when taking location information from the database. To avoid delay, the proposed system is targeted to track five people at the same time. Maps API will ask permission to track the location every 10 seconds. The time taken to track the location is analyzed by tracking one person and five people at the same time. Table 1 shows the tracking time for one person and five people.

From Table 1, the tracking time for five people is similar to the tracking time for one person. The delay when tracking multiple people is relatively low. It was found that the average tracking time for live location is approximately 10 seconds.

Table 1: Tracking time for one person and five people

No.	Tracking time for one person, second	Tracking time for five person, second
1.	10.05	10.31
2.	9.95	9.82
3.	9.99	10.03
4.	10.19	9.92
5.	10.25	10.47
AVG.	10.09	10.11

3.2 Accuracy of GPS

The proposed method focuses on determining a more accurate reference point and reducing rounding errors using a coordinate translation process to improve positioning accuracy. The percentage error is calculated by referring to the reference location and received location information. Table 2 shows the calculated percentage error of the proposed method.

Based on Table 2, the average received latitude and longitude are 1.86854258 and 103.1185174. Percentage error of latitude and longitude are 0.0030729% and 0.0000316%. This means there is a high accuracy of location which is around 99% will be tracked by the proposed system.

Table 2: Calculated percentage error of proposed method

No.	Received Latitude	Received Longitude	Percentage Error of Latitude, %	Percentage Error of Longitude, %
1.	1.86853880	103.1185181	0.0032752	0.0000309
2.	1.86854350	103.1185165	0.0030237	0.0000325
3.	1.86854240	103.1185174	0.0030825	0.0000316
4.	1.86854660	103.118517	0.0028578	0.0000320
5.	1.86854580	103.1185217	0.0029006	0.0000274
6.	1.86854310	103.1185228	0.0030451	0.0000264
7.	1.86853160	103.118523	0.0036605	0.0000262
8.	1.86854000	103.1185175	0.0032110	0.0000315
9.	1.86855260	103.1185061	0.0025367	0.0000426
10.	1.86854135	103.1185135	0.0031387	0.0000354
AVG	1.86854258	103.1185174	0.0030729	0.0000316

3.3 Range of Resolution

Resolution refers to the number of pixels in an image. Resolution is sometimes identified by the width and height of the image as well as the total number of pixels in the image. To measure the range of acceptable resolution of Azure Face API, images with different resolutions and face sizes are trained by using Microsoft Visual Studio. Table 3 shows the range of acceptable resolution of Azure Face API.

From Table 3, the ranges of image resolution between 7 dpi to 600 dpi are set as input in Microsoft Visual Studio. The images which are 7 dpi and 600 dpi are failed to be detected. Hence, the range of acceptable resolution of Azure Face API is concluded from 8 dpi to 500 dpi.

Table 3: Testing on different image resolution

No.	Image Resolution, dpi	Face Size, pixel	Result
1.	7	76 x 97	Face is failed to be detected.
2.	8	87 x 111	Face is detected.
3.	10	108 x 139	Face is detected.
4.	35	379 x 486	Face is detected.
5.	75	780 x 1000	Face is detected.
6.	150	1625 x 2038	Face is detected.
7.	300	3250 x 4166	Face is detected.
8.	400	4334 x 5556	Face is detected.
9.	500	5417 x 6945	Face is detected.
10.	600	6501 x 8331	Face is failed to be detected.

3.4 Format of Image

In this proposed system, a camera is used to take a photo which is used to verify their identity. The camera is capturing data, which creates a digital image, and then the data will be analyzed by using the algorithm in Azure Face Service. But there are many different types of image file formats that can be retrieved by the camera. Table 4 shows the result of different image formats detected by using Microsoft Visual Studio.

Based on Table 4, the different image formats including JPG, JPEG, PNG, BPM, GIF, TIFF, and PDF are set as input to be detected by using Microsoft Visual Studio. The result of PDF format is failed to detect. This means the proposed system cannot proceed with the image in PDF format.

Table 4: Testing of different image format

No.	Image Format	Result
1.	JPG	Face is detected.
2.	JPEG	Face is detected.
3.	PNG	Face is detected.
4.	BPM	Face is detected.
5.	GIF	Face is detected.
6.	TIFF	Face is detected.
7.	PDF	Face is failed to be detected.

3.5 System Latency and Accuracy Test for Face Recognition System

The proposed system is developed to detect the face of users to verify their identity. The system latency and accuracy test for Face Recognition are conducted by recording the time taken to identify the face of a user with the same angle several times. Table 5 shows the result of face detection and the time taken to detect the face.

The difference between the times taken in all testing are small, and the average time taken is 0.95s. Hence, the proposed system has high latency and accuracy in Face Recognition System.

Table 5: Result of face detection and time taken to detect face

No.	Face Detection	Time taken to detect face, s
1.	Face is detected.	0.79
2.	Face is detected.	1.18
3.	Face is detected.	0.80
4.	Face is detected.	1.10
5.	Face is detected.	0.92
6.	Face is detected.	0.92
7.	Face is detected.	0.89
8.	Face is detected.	0.99
	Average time taken:	0.95

3.6 Face Angle

Some faces might not be detected because of technical challenges. Extreme face can affect the detection of the proposed system. Frontal and near-frontal faces give the best results. The different face angles in yaw are detected to measure the efficiency of the Face Recognition System. Table 6 shows the different face angles and the result of the testing.

Based on Table 6, the proposed system can detect the face in the range of -60° to +60° in yaw. When the angle is greater than +75° or smaller than -75°, the face is failed to be detected.

Table 6: The different face angles and the result of testing

No.	Face Angle, °	Face Detection
1.	0	Face is detected.
2.	15	Face is detected.
3.	30	Face is detected.
4.	45	Face is detected.
5.	60	Face is detected.
6.	75	Face is failed to be detected.
7.	90	Face is failed to be detected.

3.7 Light Intensity

The face can be detected only when the picture is taken with a particular light source to provide enough light for the image. This experiment is carried out by five different users in the environment with different light intensity. There are no light sources from bulbs when taking image in indoor. Table 7 shows the image was taken in the environment in different light intensities.

Based on Table 7, the proposed system can detect the face in the indoor and outdoor during day and night. The light source is provided by sun during day, while the flash light of phone is the light source at night. Hence, the Face Recognition System is unaffected by the light intensity if the phone has flash light.

Table 7: Testing of different intensity when taking picture

	Day		Night	
Users	Indoor	Outdoor	Indoor	Outdoor
1.	Face is detected.	Face is detected.	Face is detected.	Face is detected.
2.	Face is detected.	Face is detected.	Face is detected.	Face is detected.
3.	Face is detected.	Face is detected.	Face is detected.	Face is detected.
4.	Face is detected.	Face is detected.	Face is detected.	Face is detected.
5.	Face is detected.	Face is detected.	Face is detected.	Face is detected.

4. Conclusion

In conclusion, COVID-19 Mandatory Self-Quarantine Monitoring System is developed with Face Recognition System and Geolocation Tracking System. The location information of users can be tracked and shown on the map with high accuracy. Users are required to verify their identity by scanning their faces every 2 hours to prevent users to go out without bringing their phone. The purpose of the proposed system is to manage the spread of COVID-19 in Malaysia by monitoring the PUI and PUS to stay at their quarantine location. In this way, the percentage of PUI and PUS contact with others is decreasing. Although the system achieves all the objectives, there is still room for improvement so that user-friendliness and convenience can be increased.

Acknowledgement

The author would like to thanks the Ministry of Education Malaysia and Universiti Tun Hussein Onn Malaysia for providing support to this project.

References

[1] D. B. K. awn Chan, "New MySejahtera updates allow self-report of Covid-19 infections," New Straits Times, January 15, 2021 @ 9:49pm

[2] "Coronavirus," World Health Organization, [Online]. Available: https://www.who.int/health-topics/coronavirus#tab=tab_1

[3] "Coronavirus disease (COVID-19) advice for the public," World Healt Organization, 13 Oct 2020. [Online]. Available: https://www.who.int/emergencies/diseases/novel-coronavirus-2019/advice-for-public

[4] J. S. H. S. Bin Chen, "A Fast Face Recognition System on Mobile Phone," University of Electronic Science and Technology of China, Chengdu, China, 2012

[5] R. K. Moloo, "Low-Cost Mobile GPS Tracking Solution," University of Mauritius , Reduit, Mauritius, 2011

[6] M. D. Ms.A.Deebika Shree, "Real Time Bus Tracking And Location Updation System," 5th ICACCS, Tamil Nadu, India, 2019

[7] K. A. Gala, "REAL-TIME INDOOR GEOLOCATION TRACKING FOR ASSISTED HEALTHCARE FACILITIES," San Diego State University, United States, 2019

[8] I. F. Progri, "AN ASSESSMENT OF INDOOR GEOLOCATION SYSTEMS," WORCESTER POLYTECHNIC INSTITUTE, United States, May 2003

[9] M. Z. Parvez, "A Theoretical Model of GSM Network Based Vechicle Tracking System," 6th ICECE 2010, Dhaka, 18-20 December 2010

[10] Kim and Jong-Myon, "An Effective Approach to Improving Low-Cost GPS Positioning," Hindawi Publishing Corporation, Republic of Korea, 25 June 2014

[11] Pentland, "Face Recognition Using Eigenfaces," Universidad Federal de Pernambuco, 25 November 2009

[12] A. Nabatchian, "Human Face Recognition," University of Windsor , Windsor, Ontario, Canada, 2011

[13] C. Chee, "Gender and Ethnic Classification of Malaysia Face Image for Face Recognition," Multimedia University, Malaysia, November 2011

[14] "Face," Microsoft Azure, [Online]. Available: https://azure.microsoft.com/en-us/services/cognitive-services/face/. [Accessed 2020]

[15] Umm-e-Laila, "Comparative Analysis For a Real Time Face Recognition System Using Raspberry Pi," ICSIMA 2017, Karachi, November 2017