A MOBILE APPLICATION MONITORING SYSTEM USING INTERNET OF THINGS (IOT) AND FIREBASE

NOR FARAHIDAH ZA'BAH [1*], ASYRAFUDDIN MOHAMAD

Department of Electrical and Computer Engineering, Kulliyyah of Engineering, International Islamic University Malaysia, Kuala Lumpur, Malaysia

**Corresponding author: adah510@iium.edu.my*

Received: 18th December 2020; Accepted: 30th January 2021
Published on-line: 30th April 2021

ABSTRACT: Dealing with consultations and appointments can be problematic if one of the parties does not commit to the appointment without any prior notice. Without any confirmation, whether he/she is in the office or not, students and other staff members may be indirectly affected with regards to their time management. Difficulty in determining this also affects students' motivation to seek consultation with their lecturers when there are problems regarding their studies. Using the "Lecturer Is In-oT" application, student will be able to know the status of the lecturers whether they are available in the room or not through the user interface. This application will be based on human monitoring application by using a proximity sensor to determine the presence of lecturer. Raspberry Pi is used as the IoT gateway to connect the data gathered from the sensor to the IoT platform known as Firebase. The mobile application act as a user interface to display the data from the Firebase for the user to view the lecturer's availability as 'IN' or 'OUT'. The application has successfully shown the data that indicate the lecturers' information by kulliyyah/department and his/her status.

***KEY WORDS*:** *Internet of things, proximity sensor, Raspberry Pi, Firebase*

1. INTRODUCTION

The fast-growing number of Internet of Things (IoT) technology has resulted in the modernization of our civilization. It can be defined as a network of devices, object and things that enable the transfer of data without human interaction (Olakunle et al., 2019). Monitoring system had become the main part of the IoT application such as a health monitoring system using an LM35 temperature sensor (Kumar et al., 2016; Deepika et al., 2019) or using passive infra-red (PIR) sensor to monitor movement inside a residence (Bassoli et al., 2017; Ooi et al., 2019). By connecting the monitoring system to the internet, people will be able to monitor remotely where the system can be accessed through a device using a specific application such as mobile application or web application.

2. RELATED WORK

Table 1 shows a summary on some of the previous works in using IoT for monitoring purpose.

Table 1: Summary on some of the previous work in using IoT for monitoring purpose

Related Work	Sensor	IoT Gateway	User Interface	Findings
Building automation & security using CAN and IoT (Halemani and Rajagopal, 2016)	Multiple sensors for energy monitoring	Beaglebone Black Board	Web Server	Depending on which node is triggered, it will send different kind of messages to the user.
IoT based monitoring and control for home automation (Pavithra and Balakrishnan, 2015)	PIR sensor	Raspberry pi	Web Apps -Apache Web Server	PIR sensor is used to detect presence of user and upon detection, electrical appliances will be turned on.
A smart system for face detection with spatial correlation improvement in IoT environment (Lu et al, 2017)	Raspberry Pi Camera Module V2	Raspberry Pi	Web and Mobile App	The algorithm will detect the face and a cropped image will be sent to the user.
Environmental monitoring as an IoT application in building smart campus of University Udayana (Sastra and Wiharta, 2016)	Visual sensor (camera) PIR sensor	ESP8266, Arduino mega XBee and Arduino	Thingspeak	Sensor will send information to Arduino, and ESP8266 will send the data to internet via Wi-Fi. It utilized two XBees where one will act as router and another one as coordinator.
Wireless water quality cloud monitoring system with self-healing algorithm (Ariffin et al, 2017)	Multiple sensors to detect the quality of water such as PH and Electrical Conductivity sensors	Arduino Atmega328p + XBee	Favoriot	All sensors will gather data and data will be processed by Arduino and sent to the Favoriot database to be stored inside the cloud

In this work, a monitoring system is used to observe lecturer's availability by knowing whether the lecturer is in his/her office. The problem faced by most students is to know the availability of their lecturers. Currently, when students plan to do any consultations, they have to inform the lecturer earlier. This can be a time-consuming process as some lecturers may be late in responding to a consultation e-mail. Hence, this system would be beneficial for the students as the lecturer's availability can be viewed in real time and remotely by using a mobile application.

By referring to Table 1, there are a number of sensors that can be used for monitoring purpose. In this work, a proximity sensor is used due to its low cost and

its ability to detect distance. The camera might be the best solution among the three sensors. However, it is costly and the algorithm associated to image and video processing can be a challenge.

As can be seen from the table, Raspberry Pi is the most used IoT gateway component. This is due to its built-in Wi-Fi connectivity and its low cost. The third component of an IoT system is the cloud platform to store the data. This work will be using Firebase and since it provides authentication services, a registration flow can be created for lecturers and students from different kulliyyahs and departments.

Therefore, the focus of this work is to ensure that whenever a lecturer enters his/her room, his/her presence will automatically be detected by a sensor and the information will be uploaded to the cloud computing via the IoT platform.

3. METHODOLOGY

Generally, there are four main components in an IoT system, as shown in Figure 1, which are sensor, IoT gateway, IoT or cloud platform and the user interface.

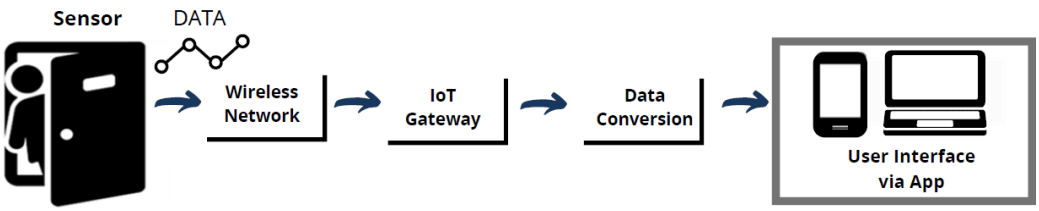

Fig. 1. IoT Components

In this work, a proximity sensor HC-SR 04 is used as the sensor to detect the lecturers' availability by using ultrasonic wave. When the sensor triggers an ultrasonic wave, the wave will travel until it hits an object, producing an echo which will be received back by the sensor (Vidhya et al., 2016). Using the speed of ultrasound and the time taken for the reflected wave to be received, the distance of the object can be determined. HC-SR 04 has four pins, V_{CC}, *Trig*, *Echo* and *Gnd*. *Trig* pin is used to trigger the ultrasonic wave and *Echo* pin will receive the reflected echo wave. The sensor is integrated with the IoT gateway using a Raspberry Pi (Model B+) with Python as the programming language. In order to configure the sensor, *Echo* pin will be connected to *GPIO 23* while the *Trig* pin will be connected to *GPIO 24* in the Raspberry Pi circuit. As soon as the ultrasonic wave is transmitted, the program will record the time interval until the *Echo* pin receives the transmitted wave. From the recorded time, the distance can be calculated. An LED is also connected to *GPIO 03* of the Raspberry Pi to indicate whether the calculated distance is more or less than the threshold distance that has been set in the program. The circuit configuration is shown in Figure 2.

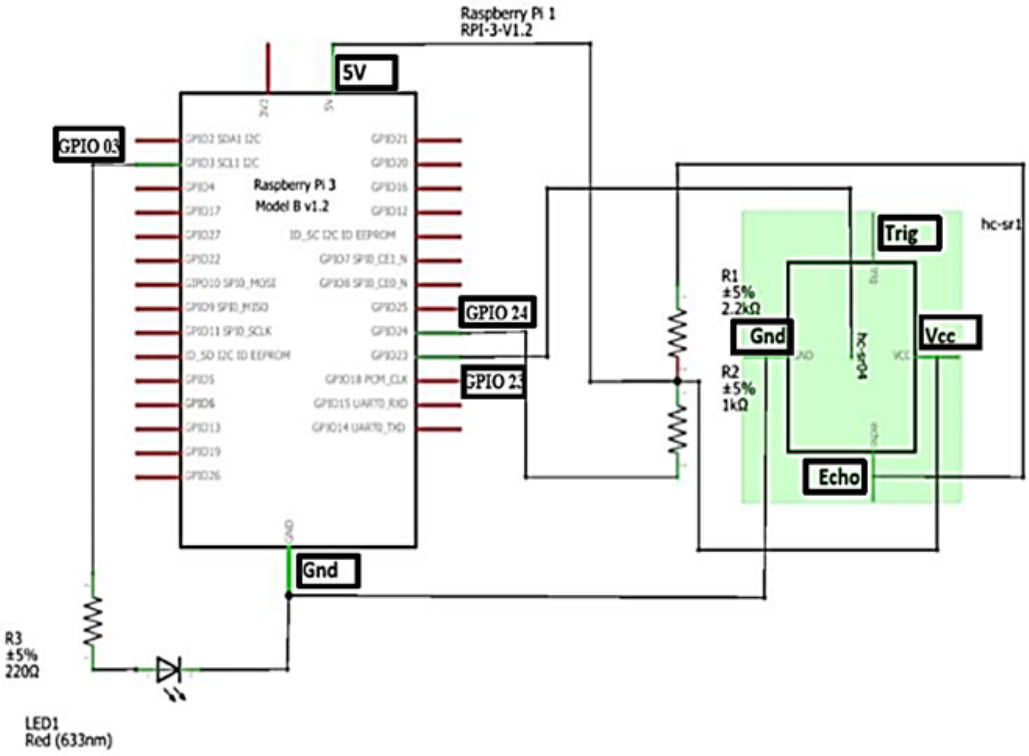

Fig. 2. Configuration of the sensor with Raspberry Pi

2.1. IoT Configuration with Cloud Platform

The data from the sensor and the microcontroller need to be stored in the internet in order for the user to be able to view the data. In this work, based on the flowchart in Figure 3, Firebase from Google will be used as a cloud platform to store all the data. Firebase is an IoT platform provided by Google for the developer to develop database application. By using REST API to read and write data, the user will be able to write data 20000 times per day and read data 50000 times per day by using the starter plan. Firebase also provides authentication services and by using Cloud Firestore, it allows an unlimited number of devices to be connected at the same time. However, for the starter plan, Firebase only allows 1GB of data storage. Some of the works that used Firebase as the IoT platform are A. Daramas et al. (2016), A. Alsalemi et al. (2017) and S. Sarkar et al. (2018). They used Firebase to provide notification services whenever the sensor is triggered.

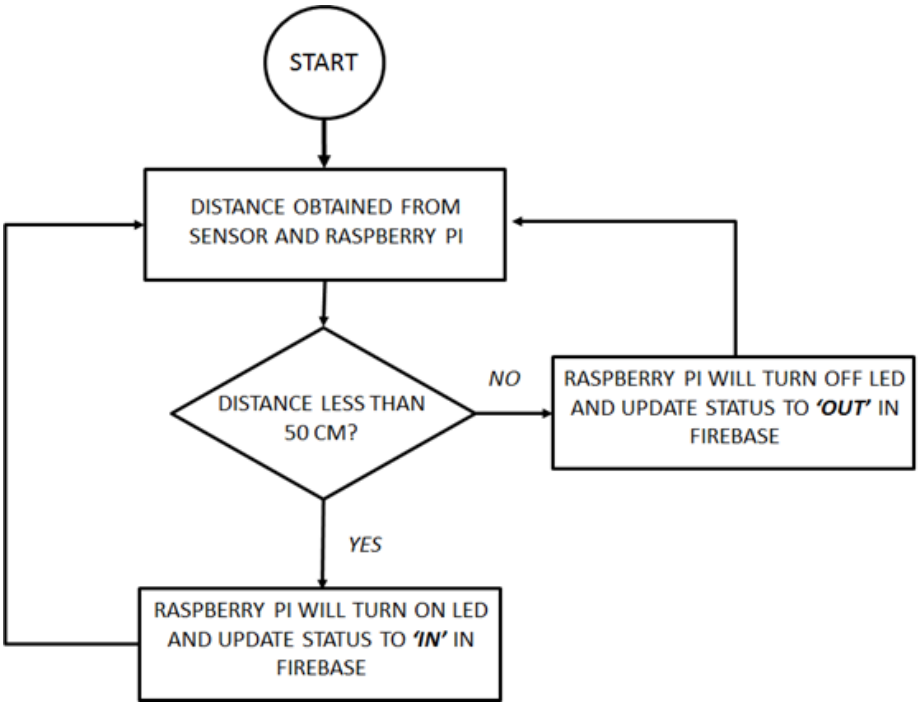

Fig. 3. IoT Configuration Flowchart

Based on the flowchart, the distance between the lecturer and the sensor is set to 50 cm. If LED is off, it means that the distance detected is more than 50 cm, which indicates that the sensor is unable to detect the lecturer. The Raspberry Pi will send the status 'OUT' to Firebase and to the user interface. Once the distance is less than 50 cm, LED will turn on, which means that the sensor is detecting the presence of the lecturer, and Raspberry Pi is ready to send the 'IN' status to the server. This cycle will be repeated at every 5 minutes interval.

In order to integrate Firebase with Raspberry Pi, the Firebase module needs to be installed by using the following command in Raspberry Pi terminal:

Pip install firebase-admin

Then, Firebase credential files need to be downloaded in order to link Raspberry Pi and Firebase. The following command is used to link Firebase and Raspberry Pi:

cred = credentials.Certificate('credential file path')

firebase_admin.initialize_app(cred)

The credential file contains information to access the right database. If the credential file is corrupted or damaged, the data will not be sent to Firebase successfully. Several collections will be used in Firebase to divide the lecturers according to kulliyyah and department. The following code shows how the lecturer's data will be sent and stored according to kulliyyah and department:

**Divide data according to kulliyyah and department:

doc_ref = db.collection(u'iium').document(u'engineering')
.collection(u'dept').document(u'electrical and computer')
.collection(u'lecturer').document(u'nor farahidah za'bah')

**Updating command:

doc_ref.update({status='IN'})

By referring to the code, the data will be divided into three collections, 'iium', 'dept' and 'lecturer'. Each of the collections contains the list of kulliyyahs (faculties), departments and lecturers which will be decided by using 'document' to access the right location. Once this is completed, the lecturer's status will be sent to Firebase by using the 'update' command. Hence, based on the flow of the program, the lecturer is required to create the data first by registering in the application.

2.2 User Interface – Mobile Application

In this work, the Ionic Framework is used to develop the mobile application. Ionic is a hybrid application which can be used on Android and IOS devices. This framework used web technology such as HTML, CSS and Javascript. The current version of the Ionic Framework supports the latest Javascript Framework like Angular.js, which will be used in this work. Figure 4 shows the overall flow for the mobile application. The system begins when a user, which can be a student or a lecturer, sign in using the student's matric number or the lecturer's staff number. It is important to highlight that the status of the user is determined based on the number of digits that are keyed in during the registration process. For a student, the number of digits is 7 (matric number) while for a staff, the number of digits is 4 (staff number).

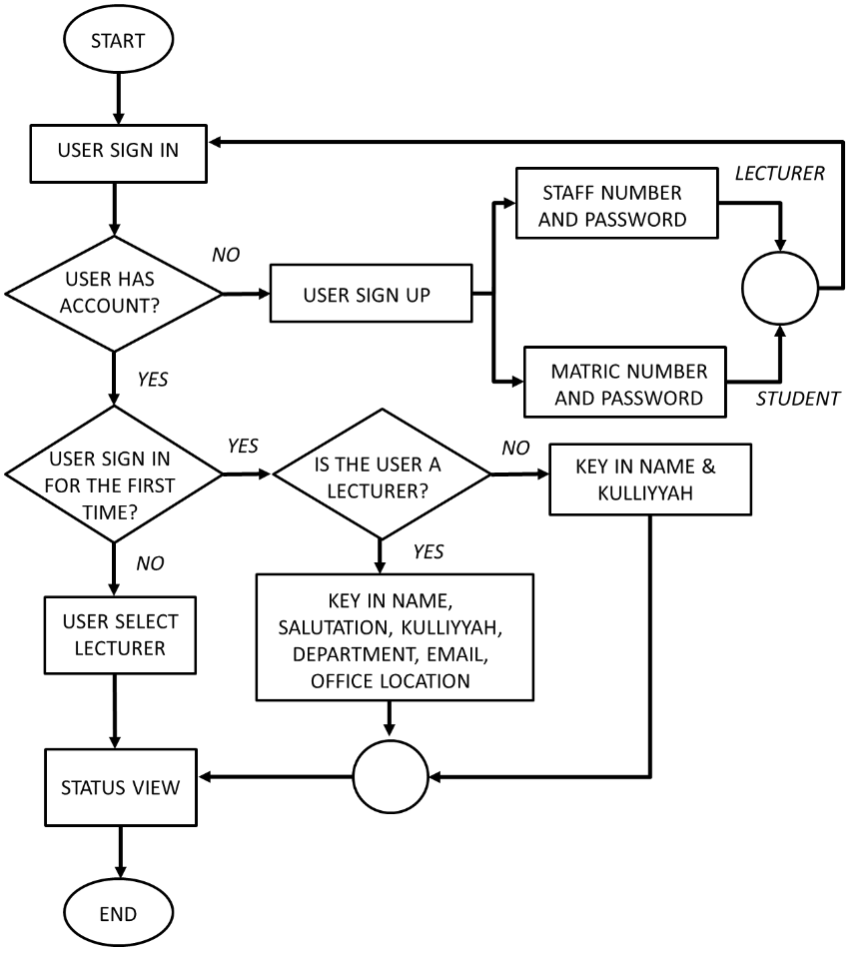

Fig. 4. The flowchart for mobile application

This status will control the application flow during the first sign in and will determine the system's application especially for lecturers. After completing all the details, every data will be stored in the Firebase. Lecturer's status will be displayed as either 'IN' or 'OUT' which is determined by the sensor's output data to the Firebase system.

4. RESULTS

Each successful registered user will have his/her unique User Identification (UID). This UID is auto-generated when the user has successfully registered with Firebase. Once the registration is complete, the user is required to sign in for the first time in order to identify the application flow (student or lecturer) as documented in Section 2.2 based on the number of digits. The UID will be stored inside the variable named '*data*'. Then the UID will be used as a reference to create a new database for the user. Two '*values*' will be sent to the register page, which is the type of user (lecturer or student) and the UID as depicted in Figure 5.

```
this.authf.auth.signInWithEmailAndPassword(this.email+'@live.iium.ed
u.my' ,this.password).then(data=>{
this.db.collection('user').doc(data.user.uid).get().toPromise().then
((doc)=>{
console.log(doc.data())
if(doc.data()==null && this.email>1000000 && this.email<1900000){

this.router.navigate(['register'+'/'+'student'+'/'+data.user.uid]);
        }
else if(doc.data()==null && this.email>1000 && this.email<10000){

this.router.navigate(['register'+'/'+'lecturer'+'/'+data.user.uid]);
        }
        else{
          this.router.navigate(['home']);
        }
      })
     })
```

value

Fig. 5. Program output when a user login

The information that was provided during registration will be stored inside a default database referred to as the '*user*' database. However, for a lecturer, on top of the default database, the information will also be stored in a '*sensor*' database as shown in Figure 6.

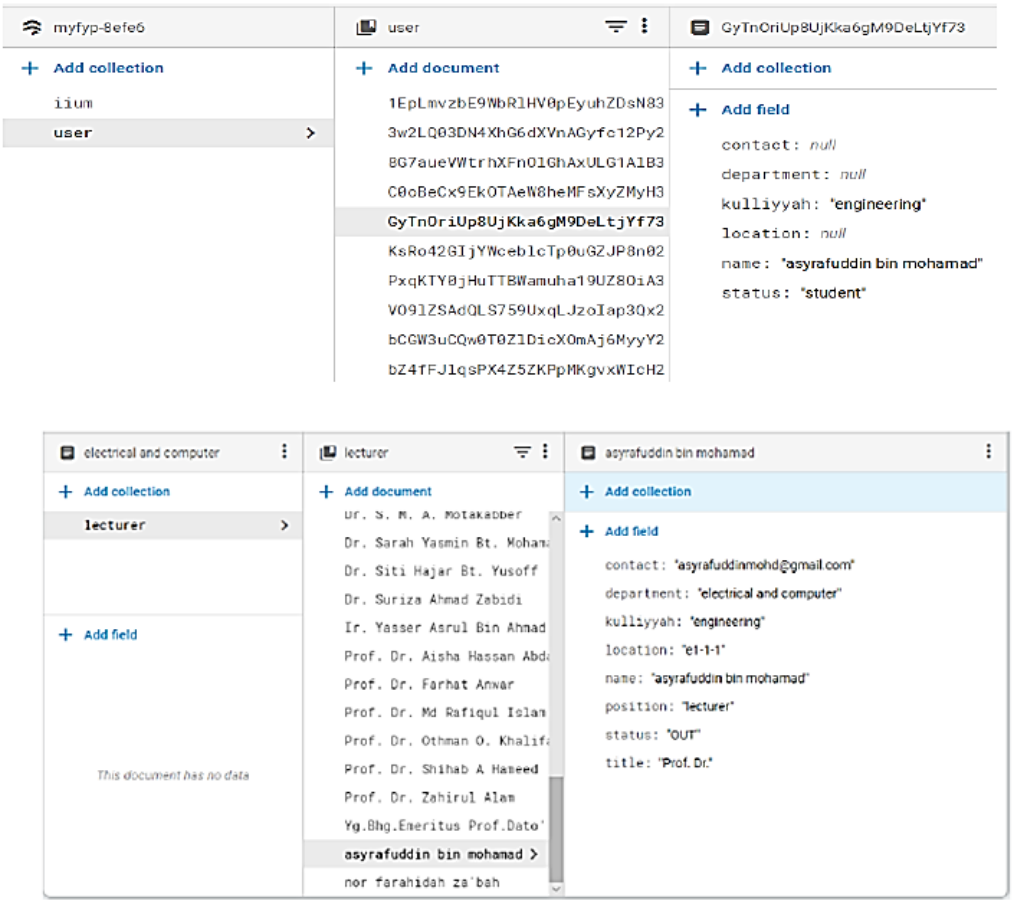

Fig. 6. Default database (Top) and Sensor Database for lecturers (Bottom)

This data will be extracted when the user wants to view his/her information on the profile page. Figure 7 shows the different interface for student and lecturer after the first sign in to when a user's profile has been created.

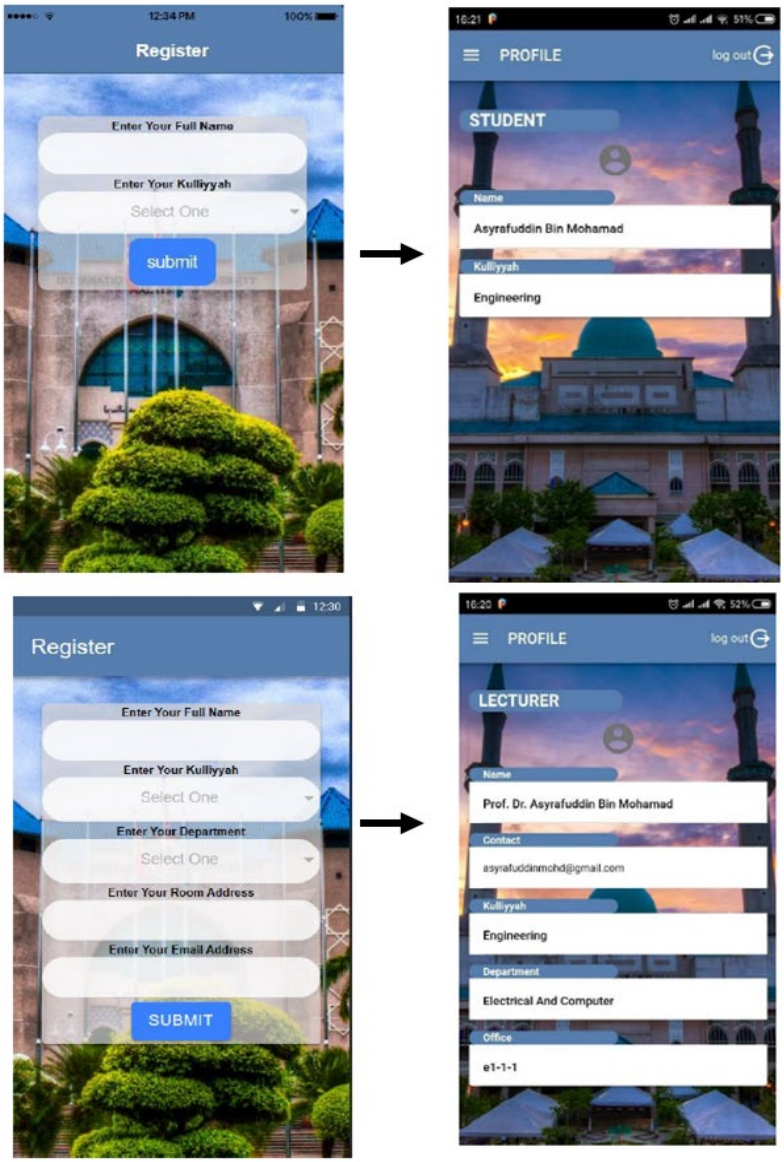

Fig 7. Student's Interface (Top) and Lecturer's Interface (Bottom) when user sign in for the first time and when the users' profile has been created

3.1 Status Update

Figure 8 shows the sensor's output (via the LED) when a lecturer's presence is detected. When the distance between the sensor and the lecturer is more than 50 cm, GPIO will set LED to LOW causing the LED to turn off. On the other hand, the LED is turned on when there is a presence detected in the range of 50 cm.

Fig. 8. Proximity sensor's output when a lecturer's presence is detected

In addition, the output from the sensor will also be sent to Firebase to update the status as 'IN' or 'OUT' and this cycle is performed every 5 minutes interval. Figure 9 shows the data from the sensor being sent to Firebase through Raspberri Pi when the process starts.

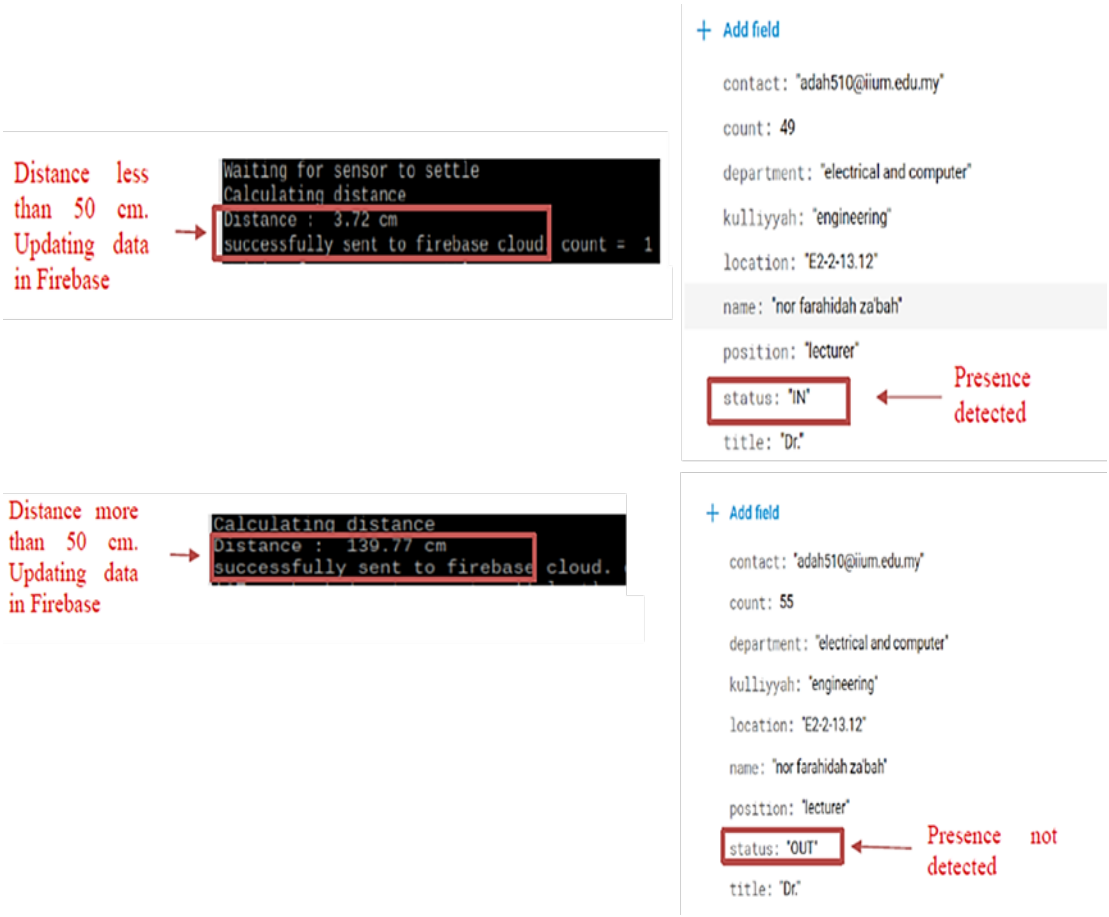

Fig. 9. The data from the sensor is sent to Firebase and the status of the lecturer is updated and sent to the mobile application. Status = 'IN' (Top) and Status = 'OUT" (Bottom)

3.2 Viewing the status

As documented earlier, the database is used to store the status of lecturers and their other information. This information is stored when a lecturer registers for the first time. When a user (either student or another lecturer) would like to view the status of a particular lecturer, other information will also be displayed such as e-mail and the location of the lecturer's office. By using an API call, only the data requested will be sent to the application. Since the data is divided into several collections, a registered user needs to choose the correct kulliyyah, department and lecturer before he/she can view the information. By using Active Route from the Angular.js to retrieve all three variables in the status page, Firebase command query can be setup to obtain the right data. Figure 10 shows how the variables were sent to the mobile application status page using Active Route.

```
Passing variables from previous page:
                          Name of next page        variables
      this.router.navigate(['lecturer'+'/'+name+'/'+kul+'/'+dept])

viewing page get the variables:

      this.lectName=this.route.snapshot.params['id'];
      this.kulliyyah=this.route.snapshot.params['id2'];
      this.dept=this.route.snapshot.params['id3'];
                                                    New page gets variables
```

Fig. 10. Variables were sent to the mobile application status page using Active Route.

Finally, Figure 11 shows the results when a user logs in to obtain the status of a lecturer.

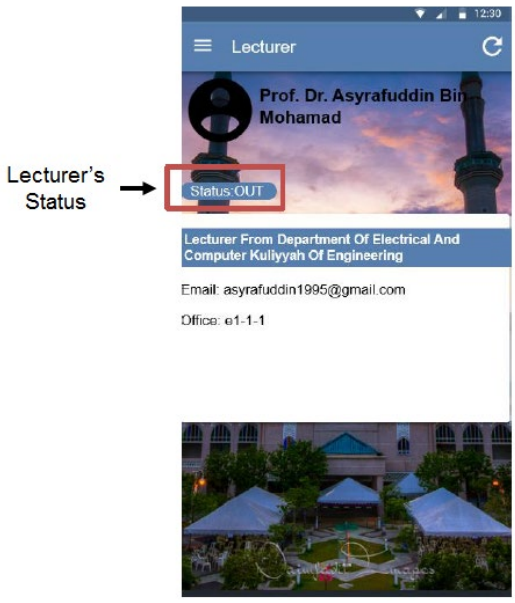

Fig. 11. The interface of the mobile application to view the status of a lecturer

3.3 Optimizing the positioning of the sensor

The positioning of the sensor is important in order to obtain an accurate reading. For this work, three different positions were tested as illustrated in Figure 12. During the test, the lecturer was asked to sit at his/her table and perform his/her task as usual. The detection cycle was performed at every minute for 1 hour. Hence, ideally, 60 detections should be obtained.

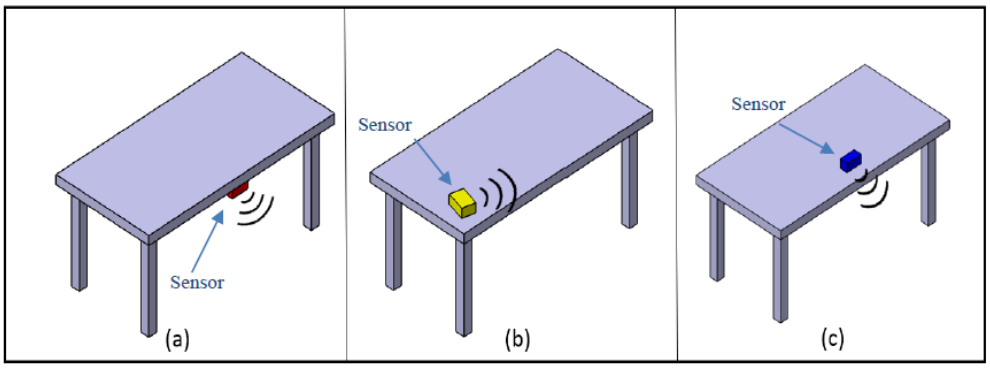

Fig. 12. Three proposed positions for the proximity sensor

For example, for position in Figure 12 (a), the result of the accuracy test is shown in Figure 13. It shows that the sensor had successfully detected the presence of the lecturer for 48 times or 80% accuracy. As can be seen from the figure, there are a few moments that the graph is flat indicating that at that minute, the sensor failed to detect the presence of the lecturer.

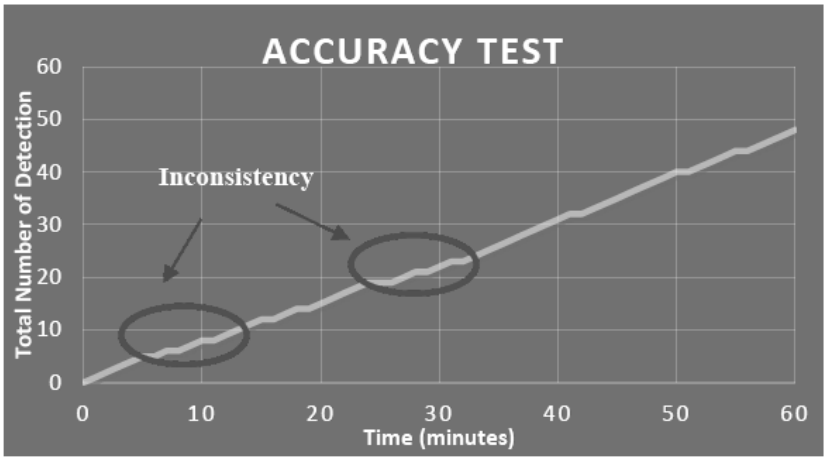

Fig. 13. The accuracy test data for the sensor's position at Figure 12(a).

Table 2 shows the number of counts for each position of the sensor. As a conclusion, position (b) produced the most inconsistencies. This may be due to the sensor being placed at a distance where the signal may be weakened when it reaches the receiver, causing inaccurate readings.

Table 2: Accuracy Test for all three positions

Position	Number of Detection	Percentage
(a)	48/60	80%
(b)	39/60	65%
(c)	48/60	80%

Based on these results, by using a proximity sensor and a much simpler algorithm, this work had produced comparable accuracy especially with regards to integrating the sensor to the IoT platform and cloud. In comparison with some of the related works, for example, the work by Lu et al, (2017) used a camera as the sensor, they had documented an average precision of 85.7%.

5. CONCLUSIONS

This work has successfully implemented the concept of Internet of Things and apply it as a systematic and manageable monitoring system for students and lecturers. By using Raspberry Pi as the IoT gateway to connect the proximity sensor to Firebase, the data is sent to the internet via a mobile application by using multilayer collection to separate the lecturers' information by kulliyyah and department. This work will not only propose an effective monitoring system to detect the presence of lecturer but it will also document the efficiency of the system by testing the accuracy of the sensor at different places within the space. As for future work, some features can be considered such as allowing the lecturer to set the status to 'busy' to inform students that the lecturer is unavailable for consultation although he/she is in the office. Another feature is to provide an analysis feature that can produce a pattern of the lecturer's availability which can provide students with some information on the schedule of the lecturer and also determine the most frequent time that the lecturer is in his/her office.

REFERENCES

A. Daramas, S. Pattarakitsophon, K. Eiumtrakul, T. Tantidham & N. Tamkittikhun (2016). HIVE: Home Automation System for Intrusion Detection. 2016 Fifth ICT International Student Project Conference (ICT-ISPC), 101-104.

https://doi.org/10.1109/ICT-ISPC.2016.7519246

Abdullah Alsalemi, Yahya Alhomsi, Mohammed Al Disi, Ibrahim Ahmed, Faycal Bensaali, Abbes Amira & Guillaume Alinier (2017). Real-Time Communication Network Using Firebase Cloud IoT Platform for ECMO Simulation. 2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), 178-182

https://doi.org/10.1109/iThings-GreenCom-CPSCom-SmartData.2017.31

Chee-Pun Ooi, Wooi-Haw Tan, Soon-Nyean Cheong, Yee-Lien Lee, V. M. Baskaran & Yeong-Liang Low (2019). FPGA-based embedded architecture for IoT home automation. Indonesian Journal of Electrical Engineering and Computer Science (IJEECS), 14 (2) 646-652.

http://doi.org/10.11591/ijeecs.v14.i2.pp646-652

D. Pavithra & R. Balakrishnan (2015). IoT based monitoring and control system for home automation. Global Conference on Communication Technologies, GCCT 2015, 169-173

https://doi.org/10.1109/GCCT.2015.7342646

D. S. Vidhya, D. P. Rebelo, C. J. D'Silva, Linford William Fernandes & C.J. Costa (2016). Obstacle Detection using Ultrasonic Sensors. IJIRST- International Journal for Innovative Research in Science & Technology, 2 (11), 316–320

Deepika N, M. Anand & F.Jerald (2019). A novel three tier internet of things health monitoring system. Indonesian Journal of Electrical Engineering and Computer Science, 15 (3), 631-637.

http://doi.org/10.11591/ijeecs.v15.i2.pp631-637

J. Lu, X. Fu & T. Zhang (2017). A smart system for face detection with spatial correlation improvement in IoT environment. 2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation, 1-4.

https://doi.org/10.1109/UIC-ATC.2017.8397550

Marco Bassoli, Valentina Bianchi, Ilaria De Munari & Paolo Ciampolini (2017). An unobtrusive Wi-Fi system for human monitoring. 2017 IEEE 7th International Conference on Consumer Electronics - Berlin (ICCE-Berlin).

https://doi.org/10.1109/ICCE-Berlin.2017.8210632

N. P. Sastra & D. M. Wiharta 2016. Environmental monitoring as an IoT application in building smart campus of Universitas Udayana. 2016 International Conference on Smart Green Technology in Electrical and Information Systems (ICSGTEIS), 85-88.

https://doi.org/10.1109/ICSGTEIS.2016.7885771

O. Elijah, Abdelmoneim A. bakhit, T. A. Rahman, T. H. Chua, S. F. Ausordin & Rifhan Narrissa Razali (2019). Production of strawberry using internet of things: A review. Indonesian Journal of Electrical Engineering and Computer Science (IJEECS), 15 (3), 1621-1628.

https://doi.org/10.11591/IJEECS.V15.I3.PP1621-1628

R. Halemani & A. Rajagopal (2016). Building automation and security using CAN and IoT. Proceedings of the 2015 International Conference on Applied and Theoretical Computing and Communication Technology, iCATccT 2015, 471-476

https://doi.org/10.1109/ICATCCT.2015.7456930

R. Kumar & M. Pallikonda Rajasekaran (2016). An IoT based patient monitoring system using Raspberry Pi. 2016 International Conference on Computing Technologies and Intelligent Data Engineering (ICCTIDE).

https://doi.org/10.1109/ICICCT.2018.8472957

S. H. S. Ariffin, M. A. Baharuddin, M. H. M. Fauzi, N. M. A. Latiff, S. K. S. Yusof & N. A. A. Latiff, (2017). Wireless water quality cloud monitoring system with self-healing algorithm. 2017 IEEE 13th Malaysia International Conference on Communications (MICC), 218-223.

https://doi.org/10.1109/MICC.2017.8311762

S. Sarkar, S. Gayen & S. Bilgaiyan (2018). Android Based Home Security Systems Using Internet of Things(IoT) and Firebase. 2018 International Conference on Inventive Research in Computing Applications (ICIRCA), 102-105.

https://doi.org/10.1109/ICIRCA.2018.8597197