© Universiti Tun Hussein Onn Malaysia Publisher's Office

IJIE

Journal homepage: http://penerbit.uthm.edu.my/ojs/index.php/ijie
ISSN : 2229-838X e-ISSN : 2600-7916

The International Journal of Integrated Engineering

Internet of Things (IoT)-based Solution for Real-time Monitoring System in High Jump Sport

Muhammad Faris Roslan[1], Afandi Ahmad[1,2*]

[1]Reconfigurable Computing for Analytics Acceleration (ReCAA) Research Laboratory, Microelectronic and Nanotechnology-Shamsuddin Research Centre (MiNT-SRC), Universiti Tun Hussein Onn Malaysia (UTHM), P. O. Box 101, 86400 Batu Pahat, Johor, Malaysia.

[2]Department of Computer Engineering, Faculty of Electrical and Electronic Engineering, Universiti Tun Hussein Onn Malaysia (UTHM), P. O. Box 101, 86400 Batu Pahat, Johor, Malaysia.

*Corresponding author

DOI: https://doi.org/10.30880/ijie.2019.11.08.020
Received 16 September 2018; Accepted 8 May 2019; Available online 30 December 2019

Abstract: Samsung, Fitbit and Sony are advanced companies that have produced various wearable devices for years, especially in sports technology for the needs of athletes. The high-tech developments within the Internet of things (IoTs) have also given athletes and coaches a smart way to develop the way the athlete trains and plays which will contribute towards economic benefit. However, the high jump sport monitoring system still use high-speed camera which is non-economical to the athlete and coaches. Besides, the camera itself needs technical expertise to setup the devices and the results are hard to understand and not in real-time basis. Therefore, the aim of this paper is to develop an IoT-based solution for real-time monitoring system in high jump sport. The OpenHAB mobile application (app) is created to communicate between the wearable device and the server. This mobile app helps the athlete to monitor their performances in the smartphone through IoT-based solution rather than using the costly high-speed camera. The outcomes have shown promising results since all data are able to be visualise and monitor in real-time, history of the training can be retrieved via the log files and the benchmark data acts as a guide to the intermediate athlete to improve the performance.

Keywords: Mobile application, real-time, Internet of Thing (IoT), monitoring, OpenHAB, high jump sport.

1. Introduction

In recent years, the technological development within the Internet of Things (IoT) have given the athletes and coaches a smart way to develop the way the athletes train and play which contributes towards economic advantage. There are now variety of objects, devices and mobile applications (apps) that benefit the athletes to track their performance and monitor their progress.

High jump athletes need the mobile apps the most as for the past recent years, they gradually used high-speed camera [1]–[10] which is non-economical to the athlete and coaches. Besides, the camera itself needs technical expertise to setup the devices and the results are hard to understand and not in real-time. To overcome these issues, IoT-based solution for real-time monitoring system in high jump sport has been developed in this study. The mobile app was created to communicate to the wearable device and server. With this IoT-based solution, it will help the athletes to monitor their performances in the smartphone without using expensive tools like high-speed camera.

*Corresponding author: afandia@uthm.edu.my

This paper begins with an overview of the proposed system implementation for the IoT-based solution for real-time monitoring system in high jump sport. The background of the IoT-based solution is presented in Section 2. The experimental setup for the mobile app is explained in Section 3 while the experimental result and analysis are explained in Section 4. Finally, a brief summary is included in Section 5.

2. Background

Today, mobile app developers try their best to develop a unique mobile app. However, everybody knows that billions of apps are already available in the online market. Hence, to discover a unique idea is really a challenging task. The mobile app demand is growing day by day, but still, the developers are unable to fulfil the constraint. As a result, this mobile app demand is increasing on higher rate.

A huge amount of time was involved to develop mobile apps for various platforms and operating system (OS). For example, the developer needs to code in Objective C or Swift to develop mobile apps for Apple iPhone operating system (iOS) and the developer needs to code in Java for Android OS. These things increase the development time, labour and cost. Consequently, to overcome this situation, the developers must agree to take new technologies like a cross platform mobile app development such as OpenHAB platform. Through this technology, developers just need to code once and then they can use that code for multiple OS like iOS and Android. This technology reduces the time, labour and cost.

OpenHAB is known as a fully open source IoT-based cross platform mobile app development software that uses C language. It uses only a single user interface to operate in all platforms like web, Android, iOS and Windows. OpenHAB cloud services also offer advantages for the developers to use this platform. It gives free cloud services with unlimited data which could be stored in their cloud server. Fig. 1 presents the IoT-based flow diagram to build the mobile app. It comprises the Mosquitto software that was used as a tool to connect between things like wearable device, server and mobile app.

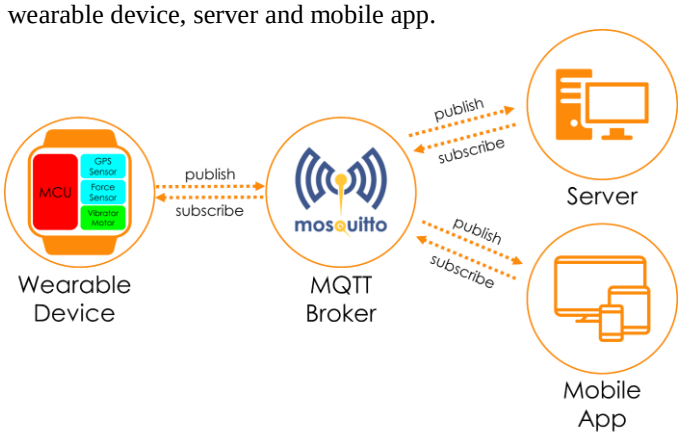


Fig. 1 - IoT-based flow diagram of overall system.

Other than that, the Mosquittto software also provides a lightweight server operation of the message queue telemetry transport (MQTT) broker protocol that is suitable for wireless connectivity. The MQTT acts as a wireless communication between the wearable device (client) to publish and host-PC (server) to subscribe the data from the wearable devices. Besides, the OpenHAB server is connected to the MQTT to communicate between the client and server when the server requested to subscribe the client. The output client from the wearable device is published in the server and mobile app after the OpenHAB server has been connected to the MQTT broker. The detailed explanation about the experimental setup implementation is described in the following section.

3. Experimental Setup

This section presents the overview of the proposed system applications for high jump sport mobile app that covers the following subsections item, rule, sitemap, cloud service and data logger of the mobile app.

3.1 Mobile Application Item

Item is a basic data type and has a state which can be read from, or written to, in order to interact with them. Fig. 2 illustrates an OpenHAB interface code to determine the item programmed by the mobile app. The item type describes which kind of state can be stored in that item and which instructions can be sent to it, such as string, number or binary switch. They are comparable with basic variable data types in the programming languages.

In this study, sensors from microcontroller units (MCUs) have been configured in this item file which defined in the “bindingconfig” to communicate the inputs between the gateway of the wearable devices and the host-PC

by using message queuing telemetry transport (MQTT) from Mosquitto software. Items are placed in the folder "${openhab.home}/conf/items".

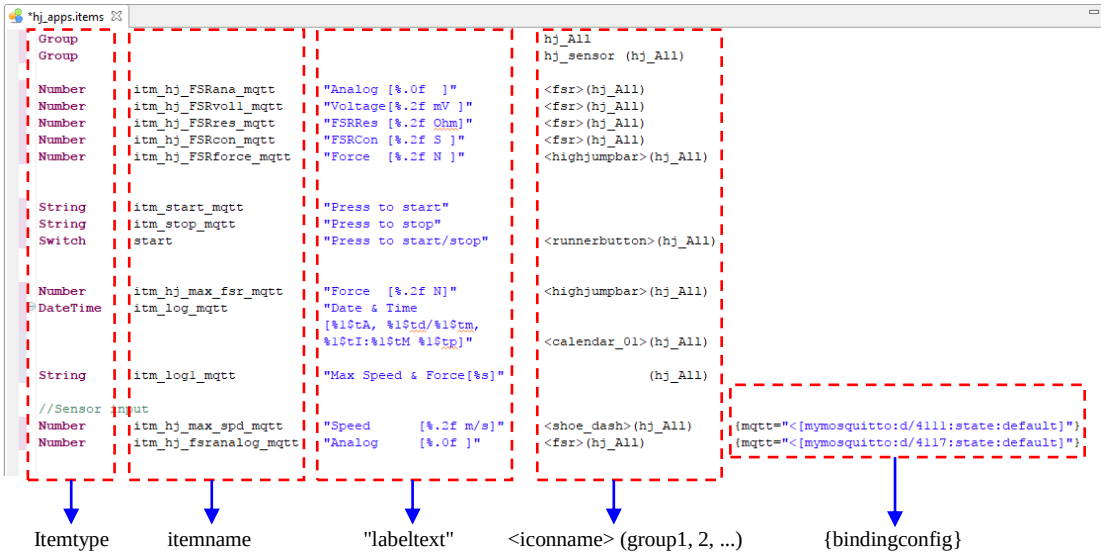


Fig. 2 - OpenHAB items interface code.

3.2 Mobile Application Rule

OpenHAB rule file is capable to have multiple rules. All rules of a file shared common execution context as it can access and exchange variables with each other. It therefore makes sense to have different rule files for different use-cases or categories. Fig. 3 shows an OpenHAB interface code to configure the rule programmed by the mobile app.

The imports section contains an import statement just like in Java. As in Java, the imported types are available without having to use the fully qualified name for them. The variable declaration section is used to declare variables that should be accessible to all rules in this file. The variables were declared with or without initial values and modifiable or read-only. The rules section contains a list of rules.

Each rule has the syntaxes as shown in Fig. 3 on the rules syntax section. A rule can have any number of TRIGGER CONDITION; however, it must at least have a minimum of one (1) TRIGGER_CONDITION to execute the rule file. The SCRIPT_BLOCK contains the code that should be executed when a TRIGGER_CONDITION is encountered. Rules are placed in the folder "${openhab.home}/conf/rules".

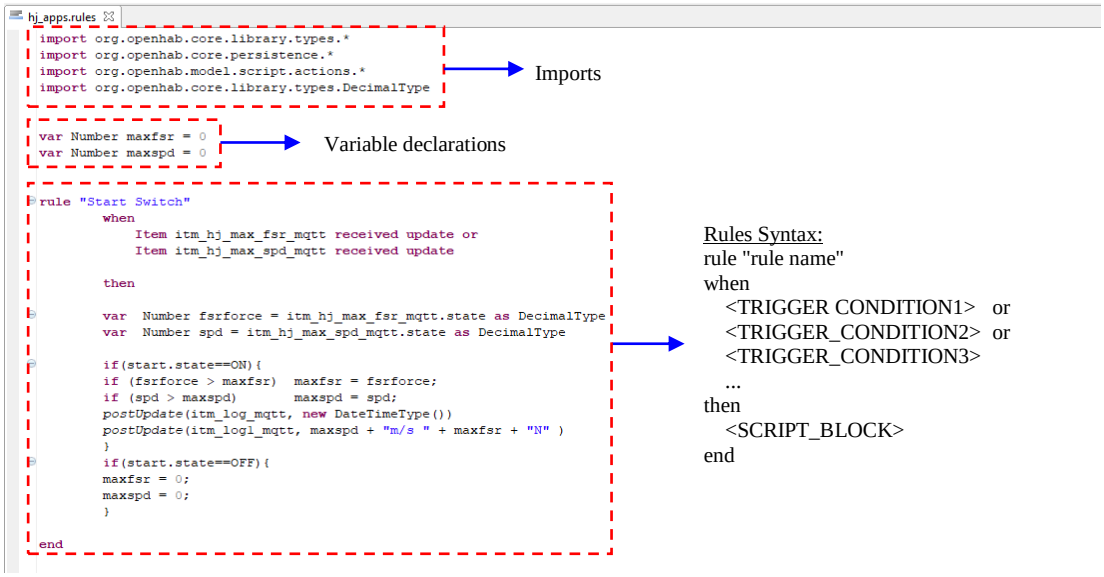


Fig. 3 - OpenHAB rules interface code.

3.3 Mobile Application Sitemap

A collection of things and items in OpenHAB represents physical or logical objects of the user's automation setup like wearable devices and host-PC. Sitemaps are used to select and prepare these elements in order to compose a user-oriented presentation of this setup for various front-ends, including GUI for the Android and iOS OpenHAB mobile app. A sitemap definition file is stored in the folder "${openhab.home}/conf/sitemaps" and has to have the ".sitemap" filename extension. For easy customisation, the OpenHAB Designer brings full integrated development environment (IDE) support for this file.

Fig. 4 demonstrates an OpenHAB interface code to configure the sitemap programmed for the GUI of mobile apps. It contains the mobile app title on the above side of the app. The SWITCH section functions to start the apps and the parameters that links from the wearable devices are also shown in real-time at the SENSOR section. The LOGS section presents the previous data which had been run by the users.

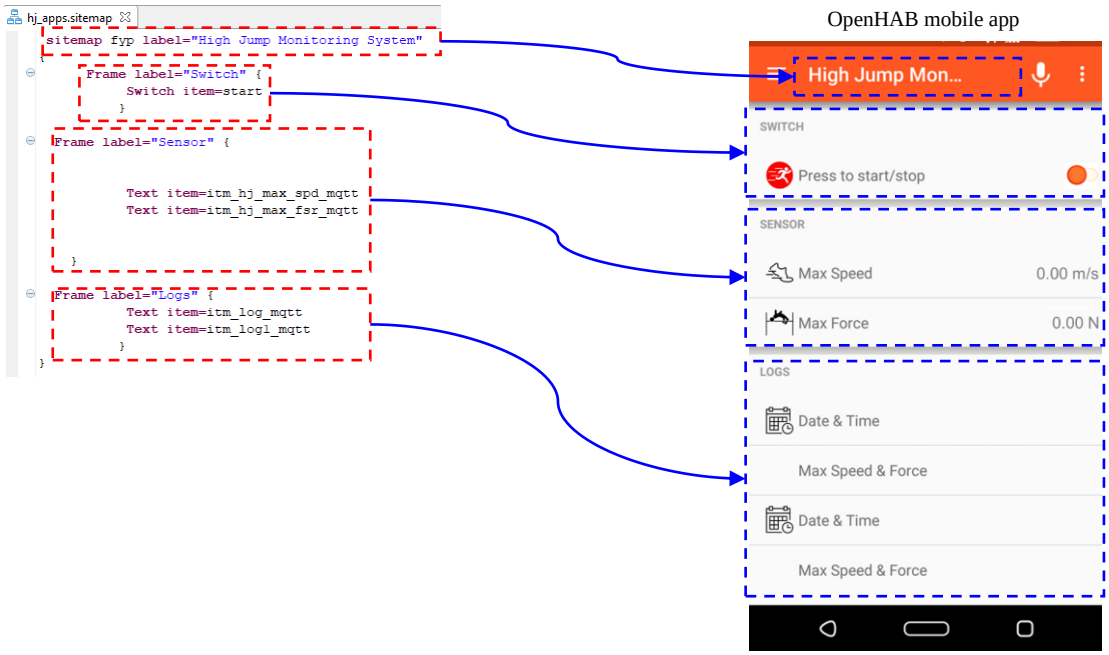


Fig. 4 - OpenHAB sitemap interface code and GUI for Android and iOS.

3.4 Mobile Application Cloud Service

OpenHAB has a simple cloud service named myOpenHAB. It is an instance of the OpenHAB cloud service, which is introduced by the OpenHAB foundation. This service is completely free to be used and meant to allow users to quickly check out its features without having to setup and host a personal instance. The registration had been made on the link "https://myopenhab.org/login" to connect the host-PC to cloud service, as shown in Fig. 5.

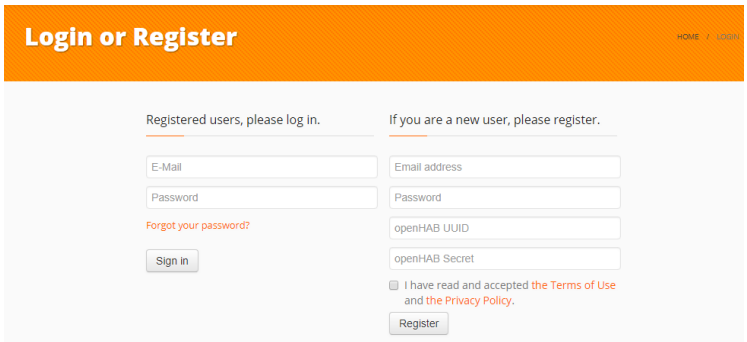
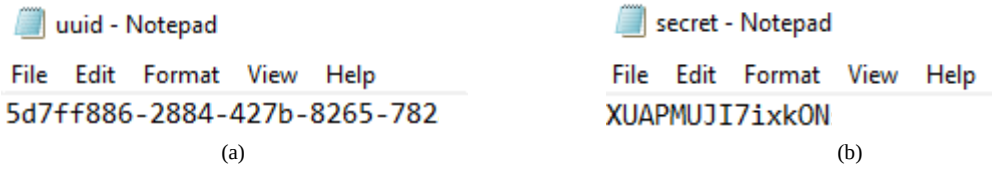


Fig. 5 - Registration setup for OpenHAB cloud service.

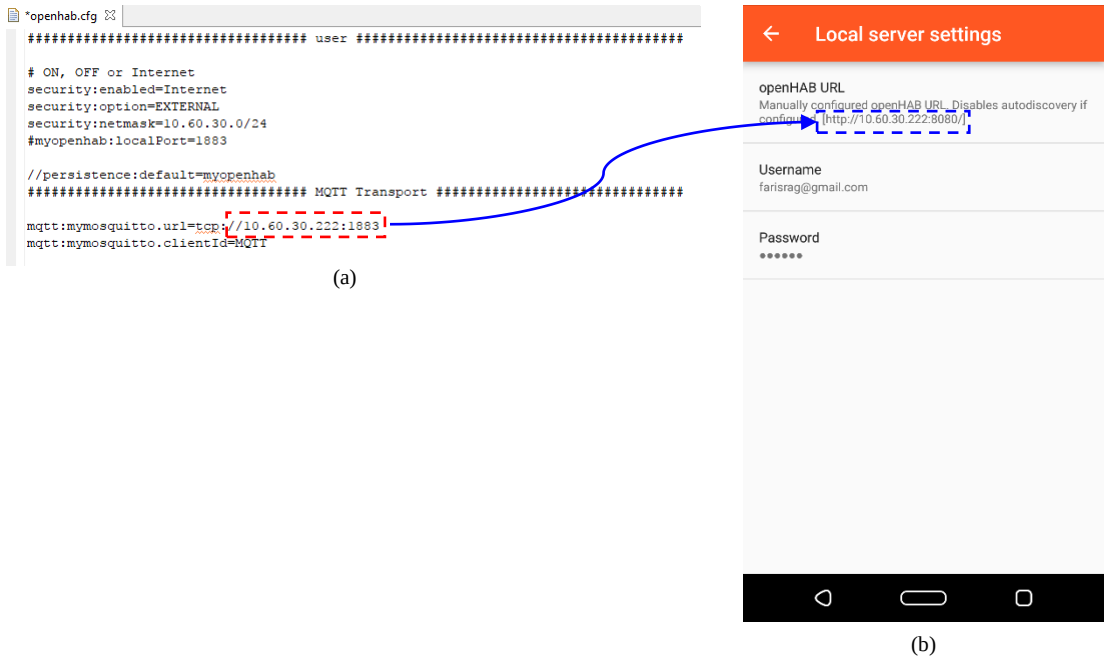
The email address and password were set by the user itself. Basically, the OpenHAB universal unique identifier (UUID) and Secret are able to discovered at path "${openhab.home}/conf/webapps/static" in the host-PC. Both are built in different mixed number and alphabet as shown in Fig. 6 to avoid other OpenHAB user to have same secret code to access the cloud service.



(a) (b)

Fig. 6 - OpenHAB (a) UUID and (b) Secret code registered in OpenHAB cloud service.

Once the registration setup had been set by the user, the user needs to set the Internet protocol (IP) address in the host-PC and mobile apps. To do that, the IP address in the "openhab.cfg" file at path "${openhab.home}/conf" and OpenHAB mobile apps server setting at path "${openhab.mobile.apps}/Settings/Local server settings" are required to have the same IP address from host-PC as illustrated in Fig. 7. With these configurations at both host-PC and mobile apps, the data are capable to synchronise from the wearable device to the host-PC via Internet in real-time.



(a) (b)

Fig. 7 - OpenHAB server settings between the (a) host-PC and (b) mobile apps.

3.5 Mobile Application Data Logger

The log file is a file that archives all the events that occurred during the operating of mobile apps. The events were recorded and saved to the log files during synchronisation between wearable devices and host-PC using the OpenHAB software. The advantages of log files are the previous events are able to be recalled and analysed by athletes and coaches. Other than that, previous events could also be plotted in graph to compare and analyse the results taken by the athletes.

In OpenHAB, the log file was configured in "logging.persist" file as shown in

Fig. 8 by setting up the transmission time to every seconds, selected items for each parameters, name for the log file and logging time were set for every second. When the wearable devices are operated, this setup automatically creates all the events for the items to be saved in the log files as indicated in Fig. 9. The events were saved in sequences starting from dates, times, items and input data from the wearable devices. These log files are placed in the folder "${openhab.home}/logs".

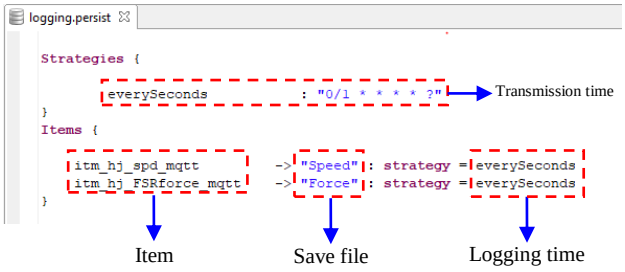


Fig. 8 - Configuration setup for OpenHAB log files.

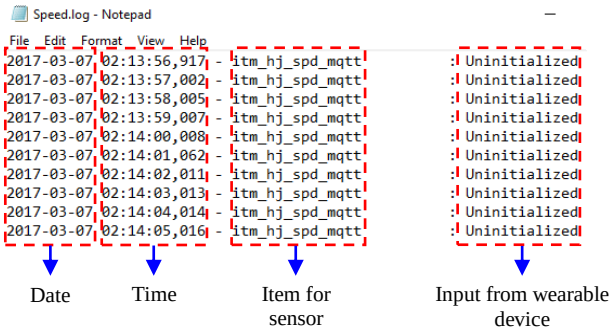


Fig. 9 - Sensor logs file from OpenHAB host-PC.

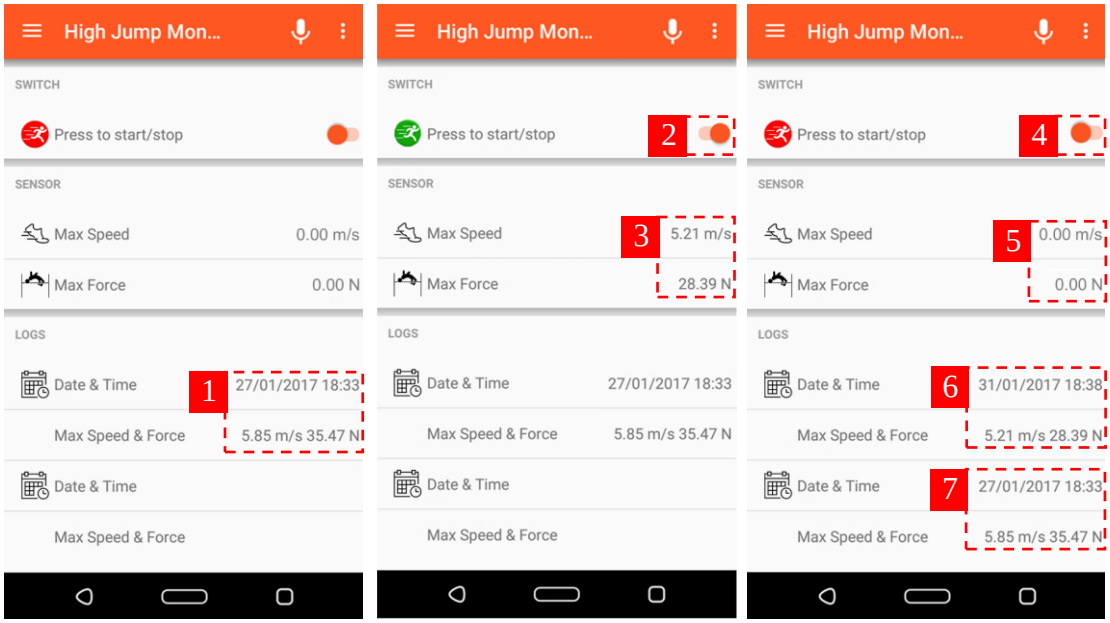
4. Results and Analysis

This experiment was conducted by testing the mobile app developed for high jump monitoring system on the smartphone by using wearable devices as the input data.

4.1 Mobile Application Graphical User Interface (GUI)

The main GUI of the mobile app was created to ensure the high jump athletes are able to monitor their performances in the smartphone. This GUI is performed as a tool to monitor the results of maximum speed during the run at Stage I and maximum force during the take-off jump at Stage II. Fig. 10 presents the main GUI of high jump monitoring system that has been developed to be used by the high jump athletes. The description of the developed GUI in Fig. 10 is explained in

Table **1**.



(a) Before operation (b) During operation (c) After operation

Fig. 10 - Main GUI of the mobile app.

Table 1 - Process flow of the mobile app.

Operation	Process
Before	1. Mobile app loaded the previous data from the saved log files.
During	2. The high jump athlete is required to press the start switch manually to activate the system.
	3. The server started saving the data in the log files and only maximum sensors value were visualised in the SENSOR section.
After	4. The switch will automatically off, if there is no reading from the wearable device
	5. The server stopped saving the data in the log files
	6. The latest data were published in the LOGS section.
	7. The previous data were published below the latest data in the LOGS section.

The speed and force data of the athletes have been saved in the log files described in Fig. 11. Inputs from wearable devices were recorded in date, time and sensor item to be easily recalled. These log data were used to analyse the data by plotting it into line graph. Fig. 12 illustrates the approach speed and Fig. 13 indicated the take-off jump force line graph that have been plotted from the logs file of speed and force sensor. Professional athlete's data were used as the benchmark data for intermediate athlete to improve their performances.

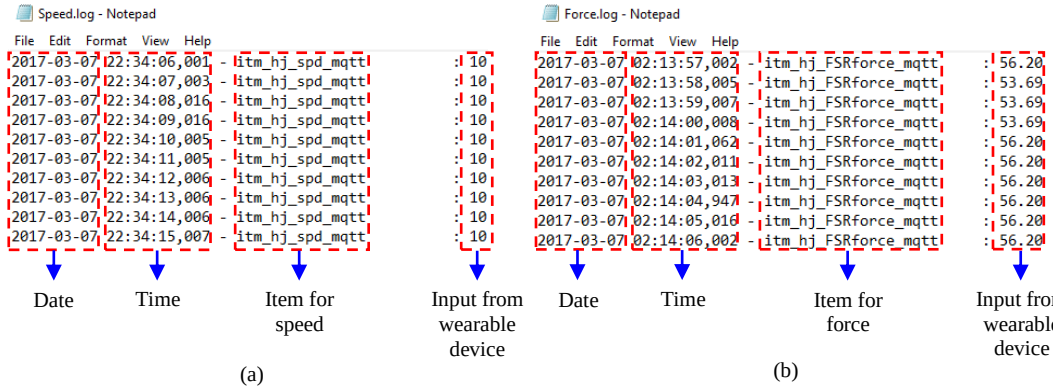


Fig. 11 - (a) Speed and (b) force logs file results from OpenHAB host-PC.

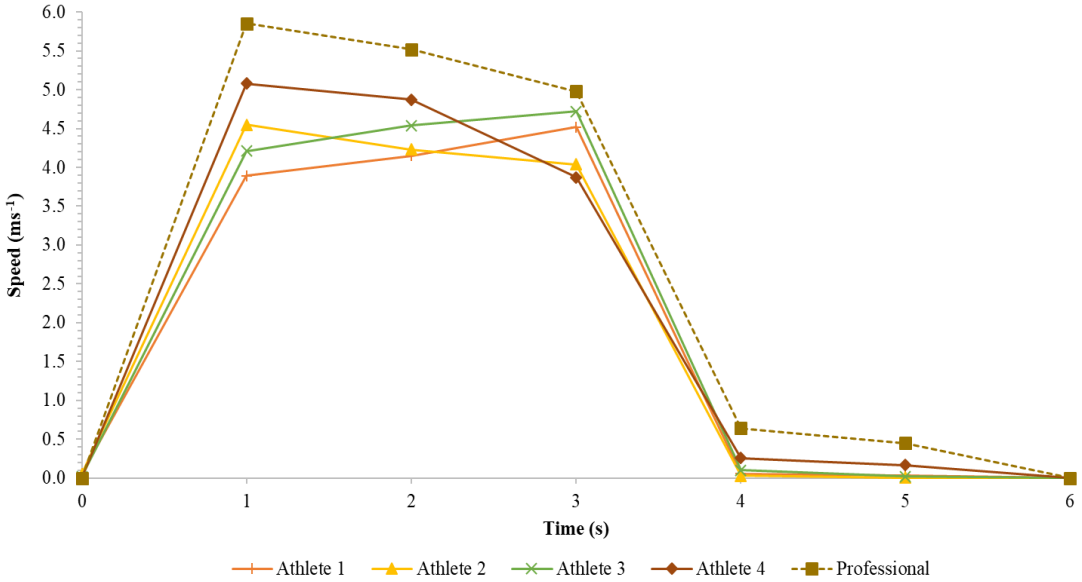


Fig. 12 - Approach speed data loaded from the logs file.

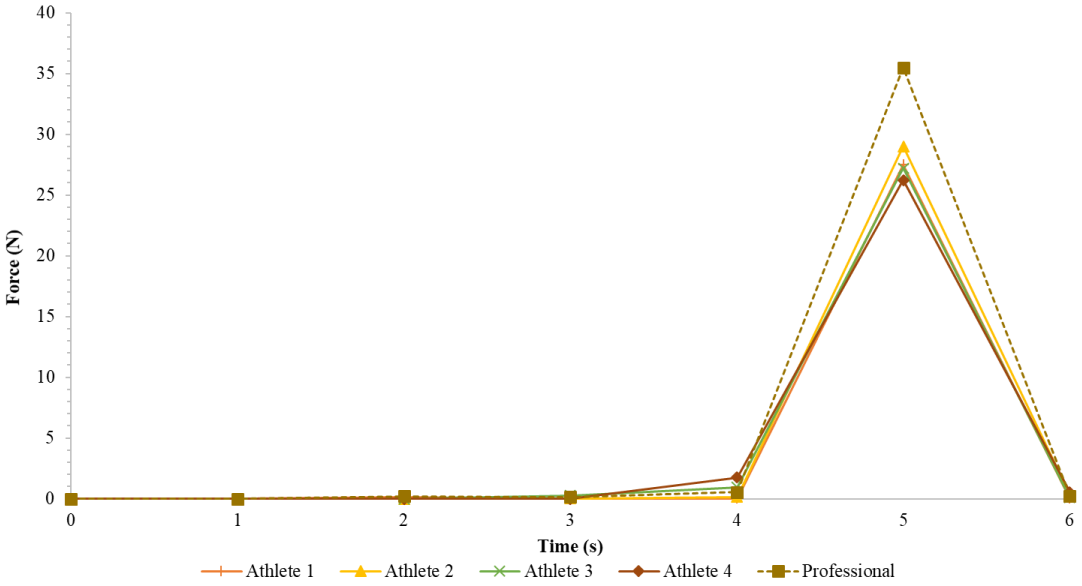


Fig. 13 - Take-off jump force data loaded from the logs file.

5. Conclusion

In conclusion, the high jump monitoring system mobile app has been discussed in this chapter. It has been developed and used by high jump athletes to monitor and guides their speed and force. The experimental setup has been provided to give more understanding on the methodology to build the OpenHAB mobile app. Besides, the mobile app also plays an important role to increase the efficiency by reducing the cost of buying hardware like high-speed camera and coaches remotely guided the athlete from the log files created by the athletes itself.

Last but not least, to improve this work, it is very convenient to study other mobile apps as another cloud storage for the high jump wearable devices system developed to decrease the number of apps stored in the smartphone. Besides, the other mobile apps GUI can be taken into consideration to improve the monitoring system interface of the existing device. Hence, other mobile apps with efficient design and implementation for sports application can be further experimented.

References

[1] M. P. Grieg and M. R. Yeadon, "The influence of touchdown parameters on the performance of a high jumper," *J. Appl. Biomech.*, vol. 16, no. 4, pp. 367–378, 2000.
[2] B. Van Gheluwe, P. Roosen, and K. Desloovere, "Rearfoot kinematics during initial take-off of elite high jumpers: estimation of spatial position and orientation of subtalar axis," *J. Appl. Biomech.*, vol. 19, pp. 13–27, 2003.
[3] I. Blažević, L. Antekolović, and M. Mejovšek, "Variability of high jump kinematic parameters in longitudinal follow-up," *Kinesiology*, vol. 38, no. 1, pp. 63–71, Jun. 2006.
[4] C. Wilson, M. M. R. Yeadon, and M. A. King, "Considerations that affect optimised simulation in a running jump for height," *J. Biomech.*, vol. 40, no. 14, pp. 3155–3161, Jan. 2007.
[5] J. Isolehto, M. Virmavirta, H. Kyrolainen, and P. Komi, "Biomechanical analysis of the high jump at the 2005 IAAF World Championships in Athletics," *New Stud. Athl.*, vol. 22, no. 2, pp. 17–27, 2007.
[6] M. Čoh, "Biomechanical characteristics of take-off action in high jump – A case study," *Serbian J. Sport. Sci.*, vol. 4, no. 4, pp. 127–135, 2010.
[7] C. Wilson, M. A. King, and M. Yeadon, "The effects of initial conditions and take-off technique on running jumps for height and distance," *J. Biomech.*, vol. 44, no. 12, pp. 2207–2212, Aug. 2011.
[8] R. M. Alexander, "Optimum take-off techniques for high and long jumps," *Philos. Trans. Biol. Sci.*, vol. 329, no. 1252, pp. 3–10, 1990.
[9] J. C. C. C. Tan and M. R. Yeadon, "Why do high jumpers use a curved approach?" *J. Sports Sci.*, vol. 23, no. 8, pp. 775–80, 2005.
[10] M. A. King, C. Wilson, and M. R. Yeadon, "Evaluation of a torque-driven model of jumping for height," *J. Appl. Biomech.*, vol. 22, no. 4, pp. 264–274, 2006.