Rafiqul Huffazh: An Android Application using Quran Automatic Speech Recognition for Assisting Muroja'ah

Amal Qurany[1, a], Dessi Puji Lestari[1,b], and Yudistira Dwi Wardhana Asnar[1,c]

[1] School of Electrical Engineering and Informatics, Bandung Institute of Technology (ITB), Indonesia
[a] mal.qurany@gmail.com, [b] dessipuji@informatika.org, [c] yudis@informatika.org

Abstract

Muroja'ah is reciting Quran verse (ayat) in order to recall the verses that have been memorized. This activity is necessary for people who memorizes Quran (in whole or in part). Muroja'ah is better performed in front of a teacher or companion in order to obtain feedbacks and evaluations, but some people might not be able to find one due to some reasons. A mobile application using Automatic Speech Recognition (ASR) to recognize Quran recitation can be used as an alternative to replace a teacher's or companion's role in assisting muroja'ah. Currently, some Quran mobile applications using ASR are available, but none of these is suitable to be used for assisting muroja'ah. Therefore, a new mobile application using ASR for assisting muroja'ah - named Rafiqul Huffazh - is developed in this project. The main feature of Rafiqul Huffazh is to transcribe Quran recitation of user memorization, evaluate the transcription based on the type of error defined and calculate the user score in performing muroja'ah. The application back-end used to transcribe Quran recitation is also developed in this project. The application has passed all the functional testing. Usability testing has also been done and it is concluded that the application interaction is considered suitable for assisting muroja'ah, however its appearance and interaction need further development in order to be more user friendly.

Keywords: **Android application, Quran, automatic speech recognition, muroja'ah, Quran memorization.**

1. Introduction

Quran, a holy book of Islam, known to be authentic. The pronunciation of Quran verses (*ayat*) is said to be authentic because the *ayat* was brought down by God himself and guarded and passed down by people who memorized it and its memorization is guaranteed through a mechanism called *sanad*. Those who memorize Quran (in whole or in part) need to recall their memorization in order to keep the *ayat* in mind. This activity is known as muroja'ah.

Muroja'ah comes from Arabic word ro-ja-a which means call, recover or regain something. In the context of memorizing Quran, muroja'ah means to repeat or recall verses of the Quran that have been memorized. Muroja'ah is commonly performed in front of a tutor because it allows the reciter (the actor of muroja'ah) to gain feedback on their Quran recitation. Due to some reasons, some who memorize the Quran are not able to find a teacher or companion to perform muroja'ah. The use of mobile application using Automatic Speech Recognition technology can be an alternative to assist the reciters and give them feedbacks on their Quran recitation during muroja'ah.

Automatic Speech Recognition (ASR) technology has been applied for various purposes and is still being developed thus far. ASR has also been applied to recognize Quran recitation. The studies on ASR to recognize Quran recitation have been carried out for at least ten years and still being continued up to this day. Some recent studies about ASR for Quran are spoken query for Quranic information retrieval by Ridwan & Lestari (2017), development of acoustic model using Deep Neural Network with BLSTM architecture by Thirafi & Lestari (2018) and CNN architecture by Muslimin, Lestari, & Asnar (2018).

Some Quran mobile application using ASR have also been developed, but they still need an advancement in order to obtain more suitable application for assisting muroja'ah. QSpeec, Al-Moallem and Tasmee are examples of Quran mobile application using ASR that can be found in Google Play. The disadvantage of QSpeec application as a muroja'ah assistant is that it is unable to show the detail of user mistake during muroja'ah. Al-Moallem can show the detail of the user mistakes, but the user has to recite the verse within the time determined by the application even though the reciting speed among users differ. This recording method can cause the recording result to have incomplete recitation or long silences. In Tasmee application, the user can recite the verses without time limit, but it still has a disadvantage as muroja'ah assistant, the application cannot recognize mistakes regarding the verses order in user recitation. Moreover, in using Tasmee application false positive recognition causes user to be unable to continue its recitation.

Fathoni, Lestari, & Maulidevi (2014) have developed a prototype of mobile application to check Quran recitation and memorization using ASR, named Najmi. Najmi can show the comparison between transcription of user recognition and reference transcript. However, Najmi has some drawbacks if it is used as a muroja'ah assistance that are 1) its ASR input method using file instead of microphone, 2) limited coverage of Quran verses (only verses number 100 to 114), and 3) unsuitable application interaction for assisting muroja'ah.

2. System Overview

2.1 Architecture

At the top level, Rafiqul Huffazh consists of two subsystems. First subsystem is the mobile application itself that acts as the interface for the users to record Quran recitation and view its results. Second subsystem is the back-end of the application. It performs recognition of user's recitation. The back-end which is adopted from Kaldi GStreamer Server was modified to recognize Quran recitation recorded by users using mobile application. Kaldi GStreamer Server is used because it is very scalable and it was also successfully used by Adianto & Lestari (2017) in their project using Bahasa Indonesia model.

The mobile application consists of eight Android-activity components and three service components that can be seen in Figure 1 below.

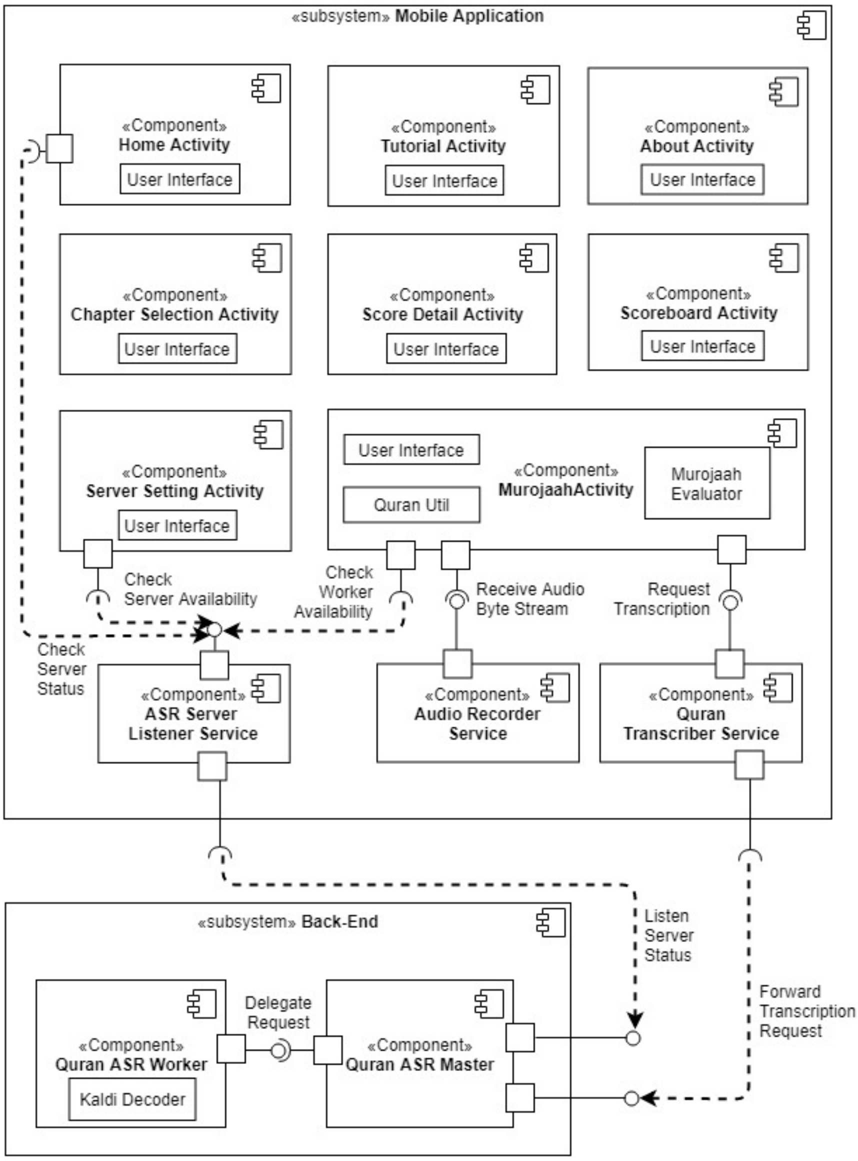

«subsystem» Mobile Application
«Component» Home Activity
User Interface
«Component» Tutorial Activity
User Interface
«Component» About Activity
User Interface
«Component» Chapter Selection Activity
User Interface
«Component» Score Detail Activity
User Interface
«Component» Scoreboard Activity
User Interface
«Component» Server Setting Activity
User Interface
User Interface
Quran Util
«Component» MurojaahActivity
Murojaah Evaluator
Check Server Availability
Check Worker Availability
Receive Audio Byte Stream
Request Transcription
Check Server Status
«Component» ASR Server Listener Service
«Component» Audio Recorder Service
«Component» Quran Transcriber Service
Listen Server Status
Forward Transcription Request
«subsystem» Back-End
«Component» Quran ASR Worker
Kaldi Decoder
Delegate Request
«Component» Quran ASR Master

Fig. 1. System Architecture of Rafiqul Huffazh

2.2 System and User Interaction

There are two endpoints on the back-end: status endpoint and speech endpoint. Status endpoint broadcasts server status that contain number of idle worker available and number of audios processed to all the client. Speech endpoint serves to receive audio data and sends the transcription to the user. The mobile application sends audio file to the back-end and the back-end process the audio file and send-back the message to the application that contains the transcription of audio file.

In using the application for muroja'ah, user recites and records its Quran recitation (from memorization) verse per verse. The recording files containing single verse recording are stored in the wav file format. Every time a new wav file is created, a new transcription request will be added to the queue. When there is any idle worker on the back-end, the queue will be processed by sending the recording file to the back-end. The back-end then transcribes the file and sends its transcription to the application. The transcriptions are collected and assessed on the application. After completing muroja'ah for certain surah, the score will be calculated and displayed to the user. The user then can choose to retry muroja'ah for same chapter, performing muroja'ah for next chapter, choose another chapter or exit from the application.

3. The Application Back-End

3.1 ASR Decoder

Decoder is the important part of an ASR system. Adjustment of Kaldi decoder was needed in order to decode Quran recitation. Two ASR model developed in previous research used to decode Quran recitation in Rafiqul Huffazh. First, an HMM-GMM model developed by Ridwan & Lestari (2017) and CNN-type-DNN model developed by Muslimin et al. (2018). The Kaldi decoder used in this project are hmmdecodefaster.sh for HMM-GMM model and decode.sh for DNN model.

3.2 Application Programming Interface (API)

The server uses two APIs to communicate with client as can be seen in the Figure 1. Both APIs uses websocket protocol with the description as the following:

1. *client/ws/status*: The client connects to this endpoint to get server status. The server broadcasts its status containing number of idle worker available and number of transcribed files to all connected clients.
2. *client/ws/speech*: The client uses this endpoint to send audio files and the server send back the transcription of that audio through this endpoint.

3.3 Transcriber System

Transcriber system used to recognize Quran recitation was adopted from Kaldi GStreamer Server developed by Alumae (2014). The model used in Kaldi GStreamer Server need to be replaced by Quran model from previous research.

4. The Mobile Application

4.1 Functionality

Following are the main functionalities of Rafiqul Huffazh application for assisting muroja'ah:

1. *Verse selection*. Users can use Rafiqul Huffazh application to perform muroja'ah in certain verse of the Quran, therefore the application provide a menu to select the verse before muroja'ah.
2. *Record Quran recitation*. The application can record Quran recitation from user and save the recording with wav file type using Audio Record class.
3. *Cancel and retry recording the verse*. In some case in muroja'ah, the actor needs to cancel and retry the recitation of particular verse. This functionality purposed to fulfil that case.
4. *Show hint*. When the actor of muroja'ah is not able to continue reciting the verse, the companion in muroja'ah usually give a hint by reciting the next part of the verse (that the actor is not able to recite/remember). In this application, the hint is given by showing the text of the verse and in addition (if needed) playing its audio.
5. *Transcribe the recording*. The application transcribes the wav file using transcription service provided by the server.
6. *Evaluate verse recitation*. The application evaluates verse recitation by comparing the transcription received from the server with the reference script in same verse.
7. *Calculate user score*. When muroja'ah finished, the application calculates user score in performing muroja'ah for that verse using particular algorithm and show the score to the user.
8. *Play recorded and reference recitation*. The user can play its recitation in performing muroja'ah on that session and also reference recitation (if audio file available) for certain verse.

Besides the functionalities mentioned above, there also supporting functions: ASR server setting, usage tutorial and server status indicator.

4.2 Application Architecture

The application designed and developed using Android Architecture Component library and MVVM pattern. Figure 2 below is the architecture of Rafiqul Huffazh application.

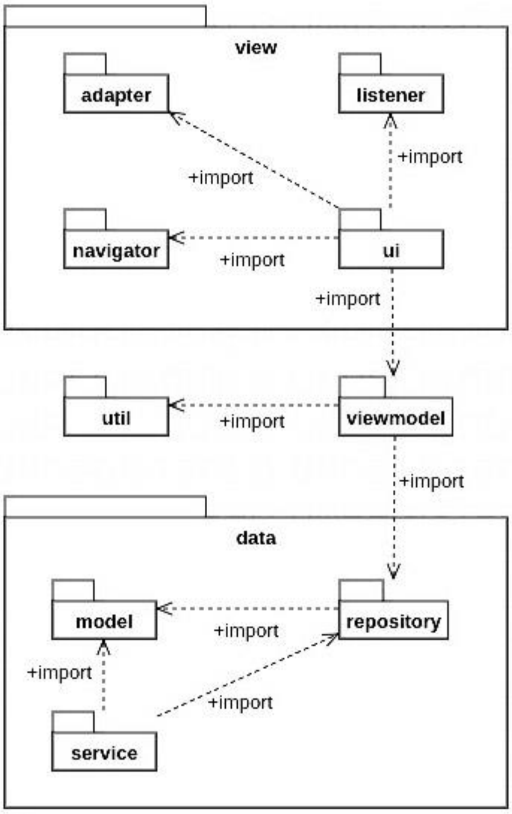


Fig. 2. Package Diagram of Rafiqul Huffazh Application

The packages on the above diagram is explained as below:

1. Package view
   This package stores classes and interfaces that are responsible for handling view elements. This package consists of four sub packages as follow:
   a. Sub package ui: contains activity classes of the application.
   b. Subpackage adapter: contains adapter classes for handling list or recycler view elements.
   c. Subpackage listener: contains listener classes for handling user action to particular views.
   d. Subpackage navigator: contains navigator interfaces that define navigation functions in particular activity.
2. Package viewmodel
   This package stores classes inherited from AndroidViewModel and responsible for supplying data to the view elements.
3. Package data
   This package handle data-related stuff and consist of three sub packages as follows:
   a. Subpackage model: contains model classes used in the application.
   b. Subpackage repository: contains EvaluationRepository class that responsible to store evaluation data of user recitation.

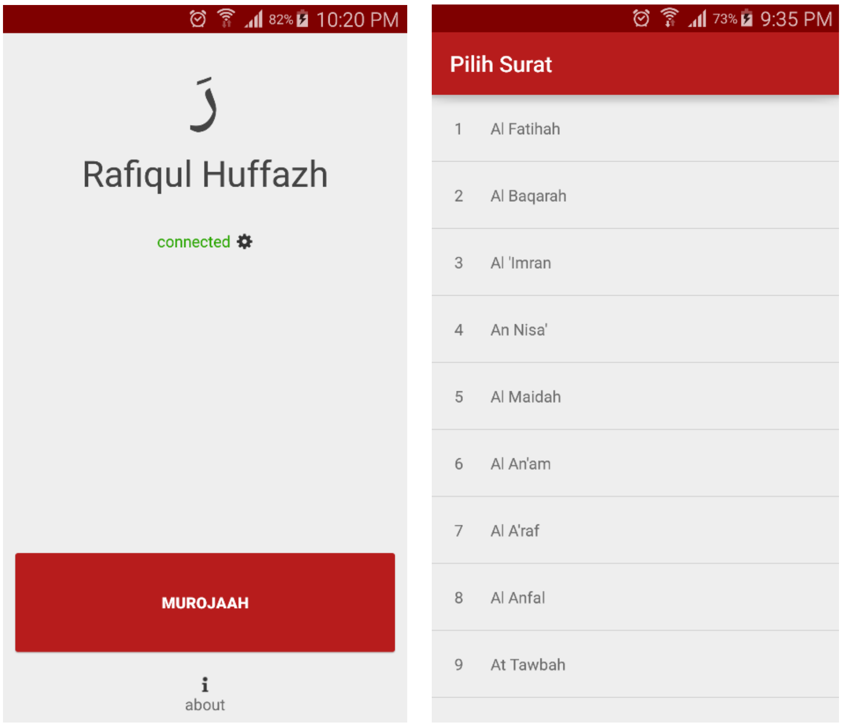
   c. Subpackage service: contains service classes used in the application including ASRStatusListener, AudioRecorder, and QuranTranscriber.

4. Package util
   This package contains utility classes used in the application. Those classes are; a) diff_match_patch class that is the library used to calculate the difference between verse transcription and its reference, b) MurojaahEvaluator that used to evaluate user recitation in muroja'ah, c) QuranFactory class that used to get Quran script, and d) QuranScriptConverter class that used to convert QScript[1] to Arabic text.

4.3 User Interface (UI)

There are eight activity classes in Rafiqul Huffazh, as can be seen on the architecture diagram, that have their own user interface. Following is the explanation of some important UIs.

1. Home and Chapter Selection UI
   On the main screen, there are three options; 1) configuring ASR Server, 2) showing application information and 3) getting muroja'ah assistance. By clicking muroja'ah button in the home view, the chapter selection view will be shown. Figure 3 below shows the view of home and chapter selection activity.



Pilih Surat

1	Al Fatihah
2	Al Baqarah
3	Al 'Imran
4	An Nisa'
5	Al Maidah
6	Al An'am
7	Al A'raf
8	Al Anfal
9	At Tawbah

Fig. 3. Home view (left) and chapter selection view (right)

2. Tutorial and Muroja'ah UI
   By selecting a chapter in the chapter selection activity, the user will be brought into tutorial activity. User may read or skip the tutorial, then go to muroja'ah activity. Figure 4 below shows the view of tutorial and muroja'ah activity.

[1] Stands for Quran Script, a Quran transliteration created by Yuwan (2015)

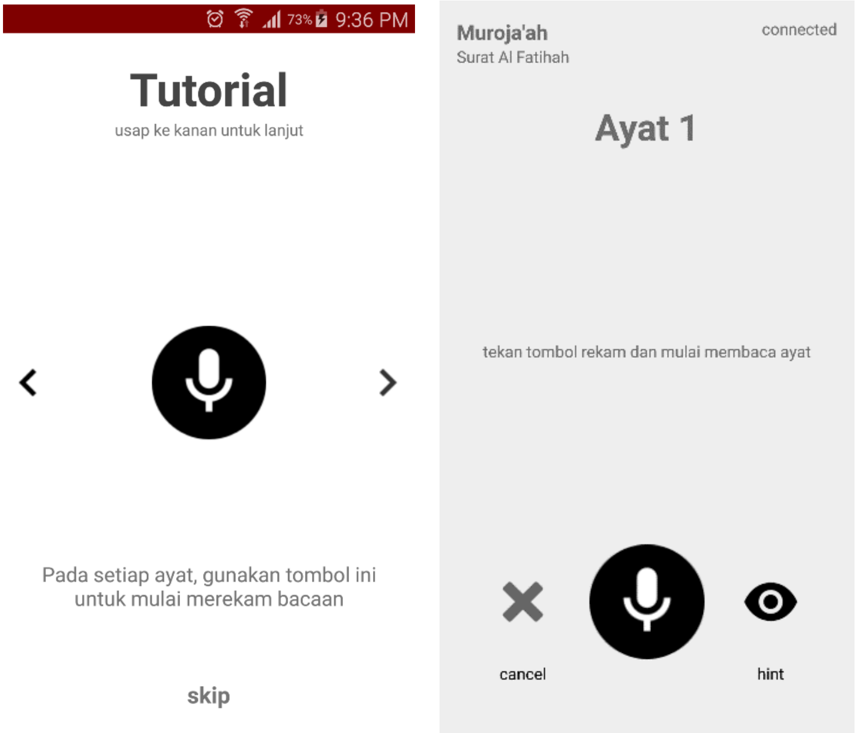
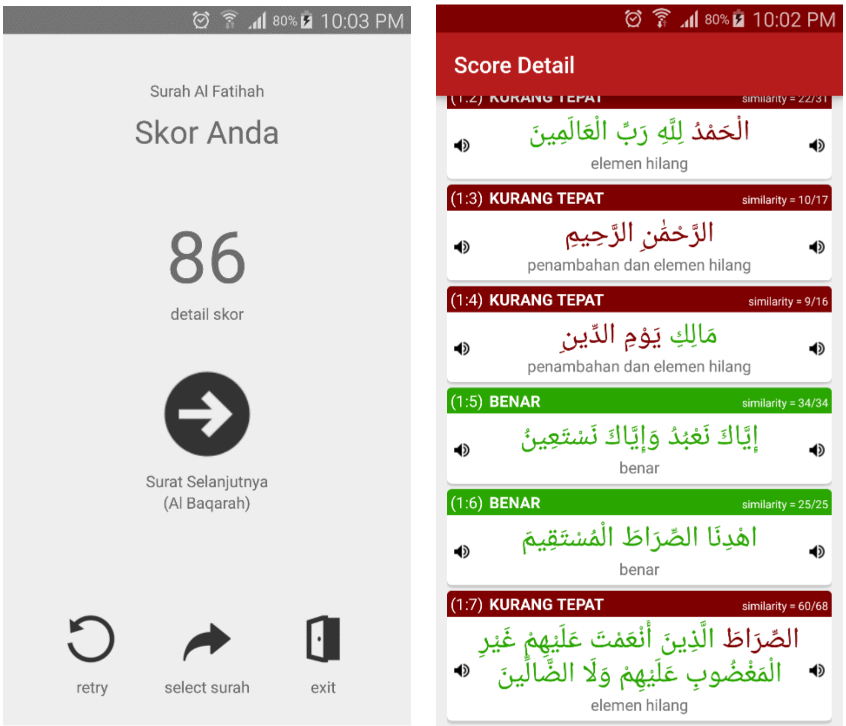

9:36 PM
73%
Tutorial
usap ke kanan untuk lanjut
Pada setiap ayat, gunakan tombol ini untuk mulai merekam bacaan
skip
Muroja'ah
Surat Al Fatihah
connected
Ayat 1
tekan tombol rekam dan mulai membaca ayat
cancel
hint

Fig. 4. Tutorial view (left) and muroja’ah view (right)

3. Scoreboard and score detail UI
   When the user finished muroja’ah, the application calculates user score and shows the scoreboard. To show the score details, the user can click ‘detail’ button on the scoreboard view. Figure 5 below shows the view of scoreboard and score detail activity.



(1:2) KURANG TEPAT similarity = 22/31
الْحَمْدُ لِلَّهِ رَبِّ الْعَالَمِينَ
elemen hilang

(1:3) KURANG TEPAT similarity = 10/17
الرَّحْمَٰنِ الرَّحِيمِ
penambahan dan elemen hilang

(1:4) KURANG TEPAT similarity = 9/16
مَالِكِ يَوْمِ الدِّينِ
penambahan dan elemen hilang

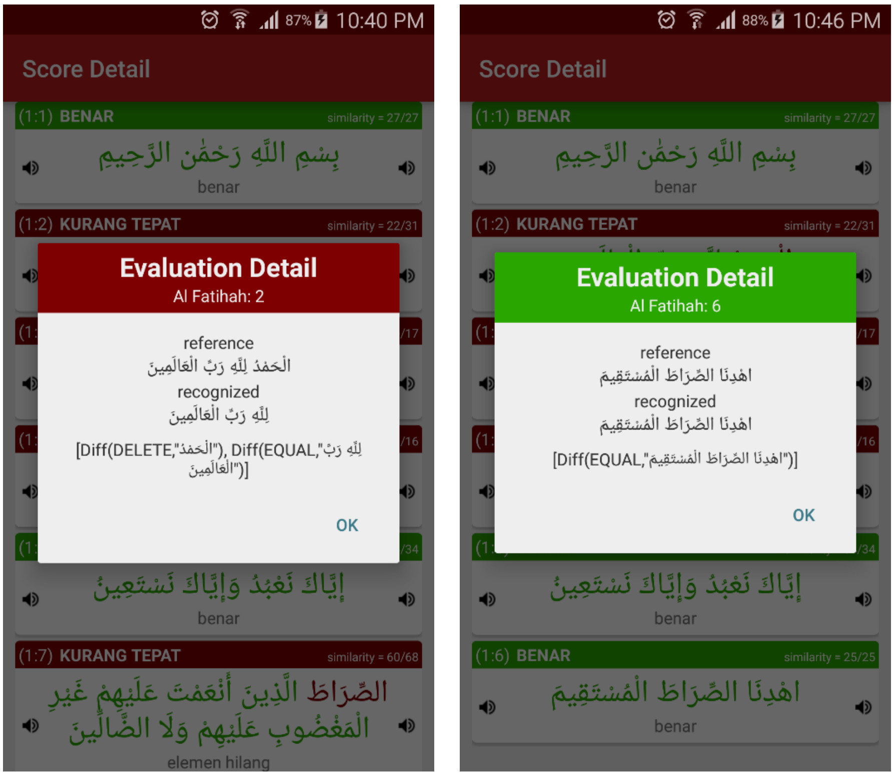
(1:5) BENAR similarity = 34/34
إِيَّاكَ نَعْبُدُ وَإِيَّاكَ نَسْتَعِينُ
benar

(1:6) BENAR similarity = 25/25
اهْدِنَا الصِّرَاطَ الْمُسْتَقِيمَ
benar

(1:7) KURANG TEPAT similarity = 60/68
الصِّرَاطَ الَّذِينَ أَنْعَمْتَ عَلَيْهِمْ غَيْرِ
الْمَغْضُوبِ عَلَيْهِمْ وَلَا الضَّالِّينَ
elemen hilang

Fig. 5. Scoreboard view (left) and score detail view (right)

In the score detail activity, user can play its recording by clicking left speaker icon and reference recitation (if the file is available in the storage) by clicking right speaker icon. User can also show the evaluation details by clicking one card view. Figure 6 below shows the view of the evaluation details for wrong and correct recitation.



87% 10:40 PM

Score Detail

(1:1) BENAR similarity = 27/27

بِسْمِ اللَّهِ رَحْمَٰنِ الرَّحِيمِ

benar

(1:2) KURANG TEPAT similarity = 22/31

Evaluation Detail

Al Fatihah: 2

reference

الْحَمْدُ لِلَّهِ رَبِّ الْعَالَمِينَ

recognized

لِلَّهِ رَبِّ الْعَالَمِينَ

[Diff(DELETE,"الْحَمْدُ"), Diff(EQUAL,"لِلَّهِ رَبِّ الْعَالَمِينَ")]

OK

إِيَّاكَ نَعْبُدُ وَإِيَّاكَ نَسْتَعِينُ

benar

(1:7) KURANG TEPAT similarity = 60/68

الصِّرَاطَ الَّذِينَ أَنْعَمْتَ عَلَيْهِمْ غَيْرِ
الْمَغْضُوبِ عَلَيْهِمْ وَلَا الضَّالِّينَ

elemen hilang

88% 10:46 PM

Score Detail

(1:1) BENAR similarity = 27/27

بِسْمِ اللَّهِ رَحْمَٰنِ الرَّحِيمِ

benar

(1:2) KURANG TEPAT similarity = 22/31

Evaluation Detail

Al Fatihah: 6

reference

اهْدِنَا الصِّرَاطَ الْمُسْتَقِيمَ

recognized

اهْدِنَا الصِّرَاطَ الْمُسْتَقِيمَ

[Diff(EQUAL,"اهْدِنَا الصِّرَاطَ الْمُسْتَقِيمَ")]

OK

إِيَّاكَ نَعْبُدُ وَإِيَّاكَ نَسْتَعِينُ

benar

(1:6) BENAR similarity = 25/25

اهْدِنَا الصِّرَاطَ الْمُسْتَقِيمَ

benar

Fig. 6. Evaluation detail view with wrong recitation (left) and correct recitation (right) of the verse

4. Evaluation of the Quran Recitation

The application performs evaluation of user's Quran recitation by comparing the transcriptions received from back-end with the reference scripts. These two strings are compared using Word Based Text Diff, which is an extension of the Diff Match Patch library that was tested by Muliati & Purwarianti (2010) to check some error cases in Quran recitation. There are two matters that was evaluated in user recitation that is verse order and verse element. The type of error due to tajweed[2] faulty are ignored in this application because of limitation of the ASR used. Therefore, following types of evaluation result are formulated.

a. *Correct*, when the transcription matched with the reference script in the corresponding verse
b. *Skipping verse*, when the transcription matched with the reference script of the next verse in that chapter
c. *Skipping some verses*, when the transcription matched with the reference script of the next two or more verses in that chapter
d. *Rewind*, when the transcription matched with the reference script of previous (one or more) verse in that chapter
e. *Insertion*, when none of the verse in that chapter matched with the transcription and there is any word in the transcription that is not in the reference script

[2] Set of rules governing the way in which the words of the Quran should be pronounced during its recitation

f. *Deletion*, when none of the verse in that chapter matched with the transcription and there is any word in the reference script that is not in the transcription

Both insertion and deletion, when none of the verse in that chapter matched with the transcription and there is any word in the transcription that is not in the reference script and also any word in the reference script that is not in the transcription.

5. Evaluation

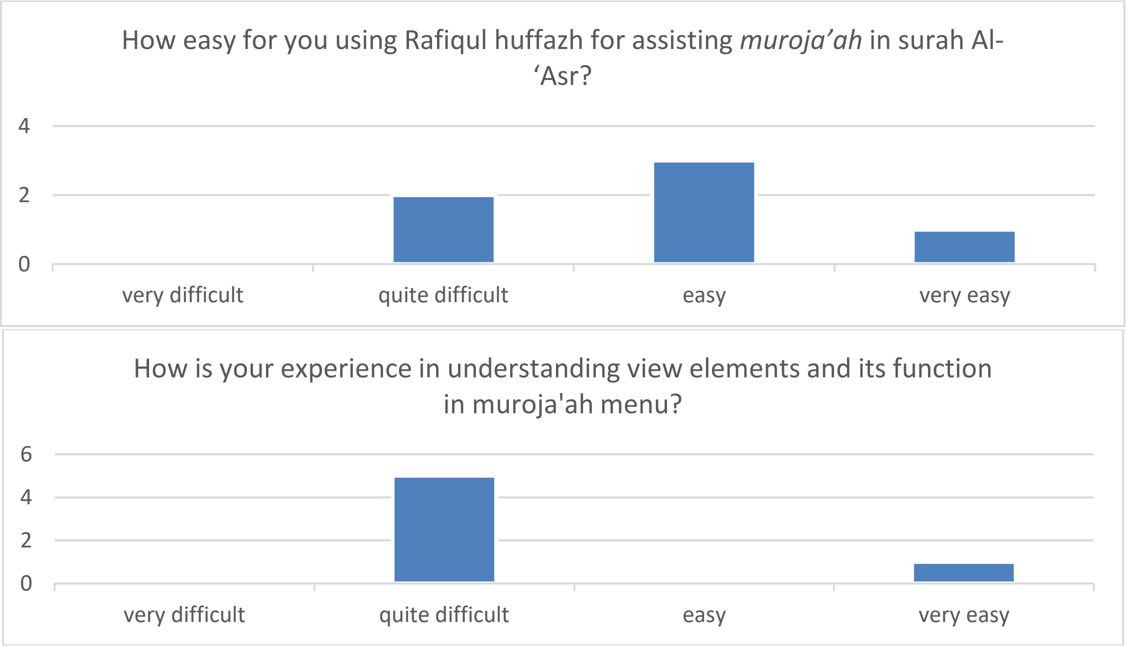
Evaluation of the Rafiqul Huffazh is conducted by performing three types of testing those are functional testing, non-functional testing and usability testing. The following is the explanation:

5.1 Functional Testing

To ensure that the application satisfy the functional mentioned before, following functional testing were performed; 1) verse selection test, 2) sound recorder with cancel and retry function test, 3) muroja'ah hint test, 4) Quran transcriber test, 5) transcription evaluation test, 6) score calculation test and 7) ASR server setting test. These functional testing were performed using JUnit 4 with Harmcrest Matcher and some manual checking method. All functionalities were pass the test.

5.2 Usability Testing

The usability testing was conducted to measure the suitability of the application interaction to assist user in performing muroja'ah as well as to identify which aspect of the Rafiqul Huffazh application that should be fixed and be improved. Usability testing was conducted to six participant that have criteria; 1) used in using smartphone and 2) familiar with muroja'ah activity. Following figures (in Figure 7) show questions asked to the participant and its results:



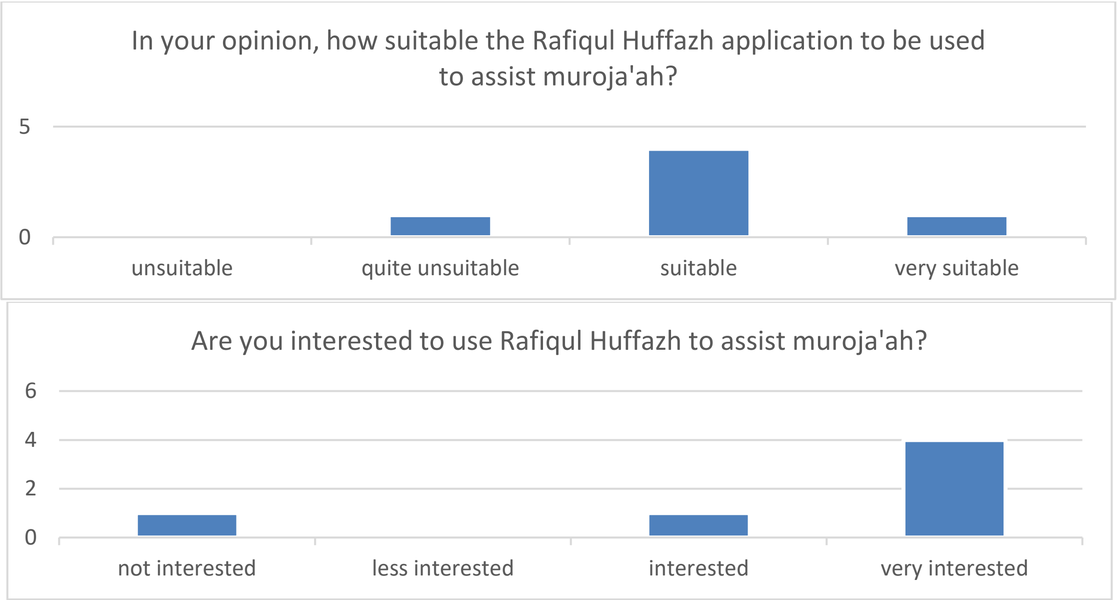


Fig. 7. The results of the usability testing questionnaire

The Rafiqul Huffazh application is considered suitable to be used as the assistance for muroja'ah. This is concluded from participant statement in usability test that out of six participant, one stated that the application is quite unsuitable, four stated that the application is suitable and one stated that the application is very suitable to assist muroja'ah. However, most of participants faced difficulty in using Rafiqul Huffazh during usability test. This is known from user's behavior and the results of usability testing that out of six participants, five participants stated that the view elements in the application are quite difficult to understand and one participant stated that that is very easy to understand.

The aspects of Rafiqul Huffazh application that should be fixed or improved based on usability testing results: improving accuracy of Quran transcriber; adding function to back to previous verse and revise the recitation; adding function to search the verse; adding function to start muroja'ah from certain page or juz, improving UI/UX; adding other feature related to memorizing or muroja'ah activity to the Rafiqul Huffazh application.

5.3 ASR System Modularity Test

The Rafiqul Huffazh is expected to be easily maintained and developed, especially its ASR system. Therefore, the ASR system of Rafiqul Huffazh was developed in a modular manner so that the ASR can be modified and developed separately from the application. To evaluate the modularity of the ASR system, the modularity test was conducted.

ASR system modularity test was done by replacing ASR model used to recognize Quran recitation. The initial model that is HMM-GMM model developed by Ridwan & Lestari (2017) replaced into DNN model developed by Muslimin et al. (2018). Modularity of the ASR system have successfully passed the test with the assertion that the application can still be used after replacing the model without any changes to the application source code.

6. Conclusion and Future Works

An Android-based system, Rafiqul Huffazh, that can assist users in performing muroja'ah using Quran Automatic Speech Recognition have been developed. All functionality that are defined passed the test. The interaction of the application is considered suitable for assisting muroja'ah. It is concluded from participant's assessment in usability testing with four of six participants stated that the application was appropriate for assisting muroja'ah, while one participant stated that it was quite unsuitable and the other stated that it was very suitable. Although the application interaction is considered suitable for assisting muroja'ah, the appearance and interaction of the application is still quite difficult to understand. It is also concluded from usability testing in which five of six participants stated that application interaction is quite difficult and one stated that it was very easy to understand and use. The ASR system of Rafiqul Huffazh successfully passed modularity test with the assertion that the application can still be used after replacing ASR model in the back-end without any changes to the application source code.

The possible development of Rafiqul Huffazh includes improving the accuracy of Quran transcriber; adding function to back to previous verse and revise the recitation; adding a function to search the verse; start muroja'ah from certain page or juz; improving UI/UX as well as adding other feature related to memorizing or muroja'ah activity as the participants of usability testing mentioned. Furthermore, adding logger and performing beta testing can also be done to get more assessment data and insight about potential development or bug fixes.

References

Adianto, R., & Lestari, D. P. (2017). Transcriber: An Android application that automates the transcription of interviews in Indonesian. *International Conference on Advanced Informatics, Concepts, Theory, and Applications.*

Alumae, T. (2014). Full-duplex Speech-to-text System for Estonian. *Frontiers in Artificial Intelligence and Applications.*

Fathoni, Z., Lestari, D. P., & Maulidevi, N. U. (2014). Pengembangan Purwarupa Aplikasi Mobile untuk Pemeriksaan Bacaan dan Hafalan Ayat Al-Qur'an pada Sistem Operasi Android. *Jurnal Cybermatika*, Vol. 2 No. 2 Article 3.

Muliati, M., & Purwarianti, A. (2010). Pemeriksaan Urutan Hafalan Al Quran Memanfaatkan Pengenalan Suara Otomatis. *Konferensi Nasional Informatika 2010.* Bandung: Institut Teknologi Bandung.

Muslimin, I., Lestari, D. P., & Asnar, Y. D. (2018). Acoustic Modelling using Convolutional Neural Network for the Quran Automatic Speech Recognition. *6th International Conference on Islamic Applications in Computer Science And Technology*. Kuala Lumpur.

Ridwan, T., & Lestari, D. P. (2017). Spoken Query for Quranic Verses Information Retrieval. *Oriental Chapter of International Committee for Coordination and Standardization of Speech Databases and Assessment.* Seoul.

Thirafi, F., & Lestari, D. P. (2018). Hybrid HMM-BLSTM-Based Acoustic Modeling for Automatic Speech Recognition on Quran Recitation. *International Conference on Asian Language Processing (IALP).* Bandung: IEEE.

Biodata

	Amal Qurany received his B.E. degree in informatics engineering from Bandung Institute of Technology, Bandung, Indonesia, in October 2018. His final project is about development of mobile application using automatic speech recognition for assisting muroja'ah in Quran memorization. He is interested in Android application development and back-end development. He is also interested in education, Islamic civilization, comparative religion and psychology. Currently, he is studying the Quran in Bandung.

	Dessi Puji Lestari received the B.E. degree in informatics engineering from the Bandung Institute of Technology, Bandung, Indonesia, in 2003, and the M.Eng. and Ph.D. degrees in computer science from the Tokyo Institute of Technology (Titech) Tokyo, Japan, in 2007 and 2011, respectively. In 2012, she joined the Department of Informatics Engineering, School of Electrical Engineering and Informatics, Bandung Institute of Technology (ITB), as a Lecturer. Her current research interests include speech signal processing, speech recognition, speech synthesizer, speaker recognition, emotional recognition, and broad domain of human computer interaction. She is the general chair of Oriental Cocosda 2016.
	Yudistira Dwi Wardhana Asnar received the B.E. degree in informatics engineering from Bandung Institute of Technology, Bandung, Indonesia, in 2002, and Ph.D. degree in computer science from Università Degli Studi of Trento, Trento, Italy in 2009. In 2011, he joined the Department of Informatics Engineering, School of Electrical Engineering and Informatics, Bandung Institute of Technology (ITB), as a Lecturer. His expertise is in software and knowledge engineering, including risk management, software engineering, risk analysis, big data, software security, and cybersecurity.