Applied Information Technology And Computer Science Vol. 3 No. 2 (2022) 241-256

AITCS

Homepage: http://publisher.uthm.edu.my/periodicals/index.php/aitcs
e-ISSN :2773-5141

Android Audio Recorder Mobile Application with Digital Watermarking

Lai Boon Shien[1], Shamsul Kamal Ahmad Khalid[1]*

[1]Faculty of Computer Science and Information Technology,
UniversitiTun Hussein Onn Malaysia, Parit Raja, BatuPahat, 86400, MALAYSIA

DOI: https://doi.org/10.30880/aitcs.2022.03.02.016
Received 07 October 2022; Accepted 09 October 2022; Available online 30 November 2022

Abstract: The copyright and ownership are much more important than previously as the idea of intellectual property is recognized in the world. It is important to protect own digital property including the audio file. Digital watermark is a way to provide strong protection for the audio file but not disturbing the audio quality. Hence, this project focuses on designing an Android application that helps the UTHM students protect their audio presentation and music copyright using digital watermarking. The chosen SDLC to develop the application is SDLC Object-Oriented Model. The application is developed using Python, Thunkable and Airtable. Then, the audio file is watermarked by using Least Significant Bit (LSB) algorithm which implemented by Python. The proposed application records the user text input for watermark and voice recorded and audio file uploaded. The watermarked audio file is also undergone a comparison testing on audio quality with two existing tool which are Audacity and AWT2. As the result stated, the quality of audio file processed by proposed application is close to the existing tool. However, due to several constraints, the proposed application has also some weakness to be improved in the future such as to restrict the creation and modifying of watermarking text to ensure the watermarking text is unique in the database, restrict the format of the registration password to ensure the users using strong password, to support more audio filetype and mobile operating system. In conclusion, the proposed application may help the UTHM students to secure their audio file from unauthorized users without cost and show their ownership to their audio file when the unauthorized using is happened.

Keywords: Watermark, Object-Oriented, Embedding, Detection, LSB

1. Introduction

The proposed project is an application proposed for UTHM students. It is proposed to protect the copyright of the music and voice recording produced by UTHM students. Nowadays, copyright is much more important than previously as the idea of intellectual property is recognized in the world [1]. Hence, watermarks, especially invisible or silent watermarks are frequently used to protect a product copyright. A silent watermark is the audio version of an invisible watermark which is used to hide the copyright information into an audio file without disturbing the audio quality of the original file [2]. An audio watermarking application is a mobile application used to hide the information as a watermark signal

*Corresponding author: shamsulk@uthm.edu.my

into the original audio signal. The hidden signal should not degrade the audio quality but should be detectable and indelible [3].

One of the weakness of non-watermarked product is it can be easily used by others without cost. This can weaken the creativity of the content creators if they know their products will be stolen. Then, the audio is more difficult to be recognized as someone's product compare to picture or video. People will simply know a picture or video is made by someone through drawing habit, photography style or editing style. However, an audio clip can only be recognized by listening and the listener may not have a clear image of the owner or creator. Next problem is the authentication of the ownership is hard in online world. People have limited evidence to prove they buy an audio file or know an audio file is sold legally. Hence, to protect the copyright and ownership of the audio file, watermark is needed for content creation process of UTHM students.

At the end of this work, the proposed application should be able to record the users' voice and embed a text input on the voice recording as a silent watermark to show the audio is copyright protected. Users should also be able to import other audio files into the application to be watermarked. Not only to encode, the application should be able to decode the watermarked audio to get the copyright information to show the audio file' copyright ownership. However, the application will also allow the normal recording without watermarking process if the users turn off the watermarking.

2. Related Work

2.1 Comparison of Audio Watermarking-Techniques

In this paper, several techniques of audio watermarking are compared with their different information hiding approaches in the audio-data and discussed in terms of robustness against audio-manipulations and signal suitability. There are three methods discussed in this paper which are spread spectrum, echo hiding and low frequency-based watermarking.

The spread spectrum watermarking technique was based on the direct sequence spread spectrum (DSSS) theory, which states that a transmitted signal is spread as widely as possible throughout the frequency spectrum by multiplying it with a code that is unrelated to the sent signal.

Four signals which are the original signal, the carrier wave, the key signal (also called Chip) and the binary string are required to encode a watermark by spread spectrum method. Figure 1 shows a schematic diagram of the synthesizing process. The coded waveform (watermark signal) is then added to the original data after being attenuated to around 0.5 of the original data's dynamic range [4]. The watermarked signal is the end outcome.

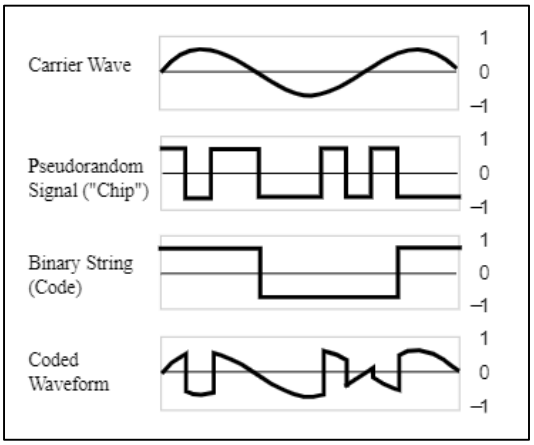

Figure 1: Watermark encoding with the DSSS method

In this method, the phase value, φ is interpreted as 0 while φ+1 as 1 to encode a watermark. It is necessary to ensure some important conditions are met during the detection process which are the key signal does not recur over an extended period of time, has a flat frequency spectrum, and is known to the decoder; the signals are synchronized in the time domain, and the beginning and end of the DSSS-data within the watermarked signal are also known. Aside from that, the key's and binary string's data rates, as well as the carrier frequency, must be understood. The decoding procedure uses binary phase shift keying (BPSK) since the phase of the coded waveform alternates with each key and watermark alternation [5]. As a conclusion, DSSS method is a complicated method but it will require longer acquisition time than other method.

For echo hiding, this method embeds data bits into the temporal domain of the original audio signal by producing an echo.

Two unique delay times, representing either a " δ1" or a " δ0," identify the type of embedded watermark bit. Thus, two system functions (kernels) are utilized to embed binary data into the original audio signal, each comprising two impulses from discrete time exponentials as shown as Figure 2. The first impulse copies the original signal by convolving one of the kernels with it, while the second impulse forms an echo. Different echo delay times and amplitudes are utilized for each kernel which are one for a binary zero (offset with initial amplitude) and one for a binary zero (offset + delta with decay rate) as shown as Figure 3 [4].

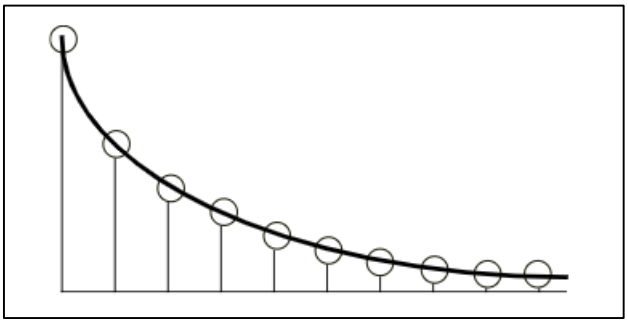

Figure 2: Discreet time exponential

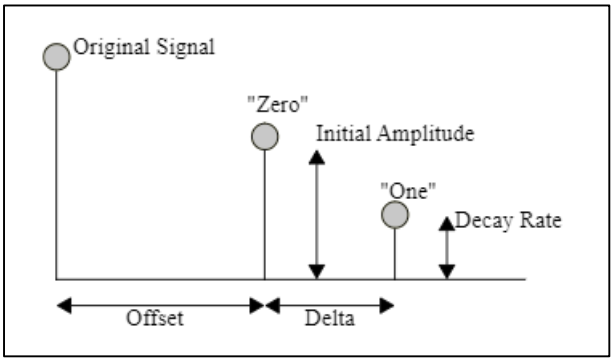

Figure 3: Adjustable parameters for the single echo hiding method

When the value of offset is decreasing, the distance between the original signal and the echo is increasing, and the echo is perceived as extra resonance to the human auditory system (HAS). The delay times δ 0 and δ 1 are chosen to be below the HAS time domain threshold for discriminating between the echo and the original signal. Furthermore, the echo amplitudes are below the HAS's hearing threshold [5].

Due to the strength of echo concealing techniques, buried information is kept intact even when additional information is inserted via LSB coding or spread spectrum. However, this costs as low payload capacity. Hence, use the LSB coding directly become one choice for the proposed application.

Low frequency-based watermarking is analogous to spread spectrum watermarking in several ways [5]. The original signal is divided into N segments with N samples each. It generates a PN sequence of length N having the values -1 or +1, which is used as the watermark key, and is generated via a chaotic map. This PN sequence is then modulated with the original audio signal's segment and attenuated with a predefined factor for each segment. Unlike DSSS, the result is subsequently low pass filtered with a Hamming filter, yielding an "inaudible watermark" [6] when coupled with the original signal. The filtered signal segment is combined with the original signal segment. The watermarked signal is created after this method is performed for each segment.

2.2 A New Audio Watermarking Method Based on Discrete Cosine Transform with a Gray Image

In this paper, two audio watermarking algorithms are compared with their grey image which are Discrete Wavelet Transform and Discrete Cosine Transform. These two algorithms are discussed on their strengths and weakness as potential algorithms for the proposed application.

Discrete Wavelet Transform is widely used in DSP applications including audio compression, data exchange via the Internet, pattern recognition, and numerical analysis. This transform is particularly good at representing signals with localized variations.

Each level in DWT is referred to as an octave, which at least in a 1-D case can be built as a pair of finite impulse response (FIR) filters, a low pass filter (LPF) and a high pass filter (HPF) as shown in Figure 4. Two down-sampling blocks work within an octave, each following a filter. These two reduce the octave's output samples by half, resulting in a reduced calculation load. The inverse of DWT, on the other hand, can be generated as an inverse LPF (ILPF) and an inverse HPF (IHPF) after two up-sampling blocks as shown in Figure 5 [4].

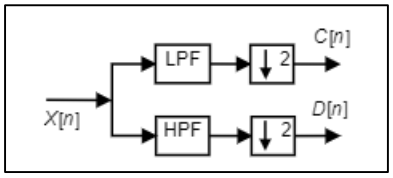

Figure 4: Discrete Wavelet Transform Block

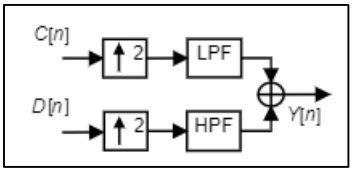

Figure 5: Inverse Discrete Wavelet Transform Block

As a result, the number of wavelet octave output samples is half that of the input mean signal. It's worth noting that the output mean/detailed signals can be used to reconstruct the input mean signal to an octave. This means that DWT simply has to maintain all detailed signals and the last output mean signal to reconstruct the original audio signal [7]. Therefore, DWT is expected more accurate in displaying high frequencies compared to the transforms analyzing signals in the time-frequency domain.

Discrete Cosine Transform is a transform that represents a signal as a set of coefficients derived from the sum of cosine functions oscillating at various frequencies and amplitudes. DCT, like the other transforms, is used to reduce correlation between signal elements.

For a 1-D case, DCT is expressed as

$$C(m) = a(m) \sum_{n=0}^{N-1} f(n) \cos\left[\frac{\pi(2n+1)m}{2N}\right]$$

m = 0, 1, 2, …, N-1

while for reverse DCT is expressed as

$$f(n) = \sum_{m=0}^{N-1} a(m)C(m) \cos\left[\frac{\pi(2m+1)n}{2N}\right]$$

m = 0, 1, 2, …, N-1

and a(m) is expressed as

$$a(m) = \begin{cases} \sqrt{\frac{1}{N}}, if\ m = 0 \\ \sqrt{\frac{2}{N}}, if\ m \neq 0 \end{cases}$$

One of the criteria for comparing transform performance is the ability to compress the signal's energy into a small number of coefficients [7]. When quantizing, the transform is allowed to ignore the coefficients with low amplitudes without losing precision when reconstructing the signal from its coefficients because DCT is one of the best in terms of compressing capability.

2.3 Audio steganography using LSB encoding technique with increased capacity and bit error rate optimization

In this paper, the Least significant bit (LSB) algorithm is discussed detailly. The LSB algorithm is one of the easiest ways to embed information in a digital audio file [8]. By substituting the least significant bit of each sampling point with a binary message, LSB algorithm allows for a large amount of data to be encoded.

The receiver needs access to the sequence of sample indices used in the embedding process in order to decode a secret message from an LSB encoded audio file. The secret message to be encoded often has a shorter length than the entire number of samples in an audio file. The next step is to decide how to select the subset of samples that will include the hidden message and to inform the recipient of that choice. One simple method is to begin the audio file at the beginning and perform LSB coding, leaving the subsequent samples unmodified, until the message has been entirely inserted.

The LSB watermarking algorithm using a two- step approach [9] was tested on 11 audio sequences from different music styles (pop, rock, techno, jazz). The audio excerpts were selected so that they represent a broad range of music genres, which has different dynamic and spectral characteristics. All music pieces have been watermarked using the proposed LSB watermarking algorithm. The overall results showed that the bit error rate is high when 64 kbps sample audio was used. But when the kbps of sample audio become higher, the bit rate is lesser and it is minimized noticeably [10]. This mean the kbps of the audio file should be ensured high enough if LSB algorithm is used in the proposed application.

3. Methodology/Framework

To develop an audio recording and watermarking application, object-oriented model is chosen as the systems development life cycle model because it is data-focused and easy to work with problem domains. The object-oriented software development life cycle consists of three main processes which are object-oriented analysis, object-oriented design and object-oriented implementation as shown in Figure 6. In the analysis phase, a use case model and carry out the object analysis. After the test on the analysis, the project will be moved on to the design phase. In this phase, the class of the project is designed to define the attributes and methods used in the project. Then, the object model, user interface and prototype are built and tested. Enter implementation phase, the project will be implemented using object-oriented programming language such as Python and undergone the user and usability test. Moreover, testing phase will be include in the software development life cycle to ensure the application run well and discover any weakness of the application.

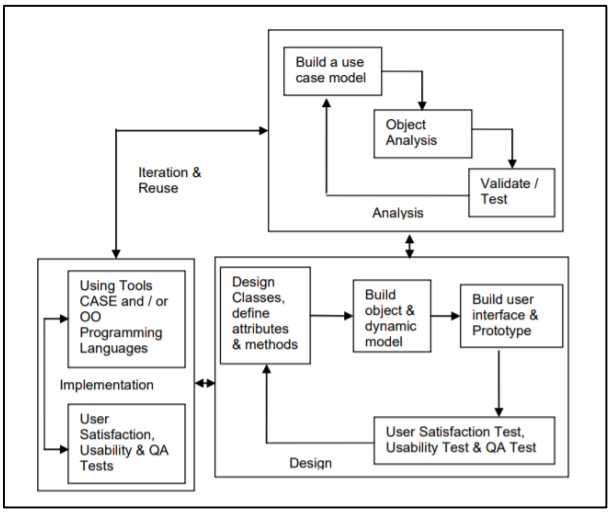

Figure 6: Object-oriented software development life cycle

3.1 Object-oriented Analysis Phase

Object-oriented analysis is focused with determining system requirements and identifying classes and their links to other classes in the problem domain during software development. Hence, functional requirement, non-functional requirement and class diagram is stated in this phase. Moreover, the software required will be analyzed in this phase. As a result, the software required is Thunkable and MySQL while the programming used to implement the proposed application is python.

a. Functional Requirement

A functional requirement is a description of the service or function which must be provided by the proposed application. Hence, the functional requirement of the proposed application is listed in Table 1 based on the module of proposed application.

Table 1: Functional Requirement

Module	Functionalities
Register	User must register as a new user to the application. New user must give a text input as own audio watermark when register.
Login	User must input valid username and password. User must update and confirm their own watermark.
Watermark Customization	User can customize the silent watermark as text input.

Table 1: (cont.)

Module	Functionalities
Voice Recording	User can record the voice as a wav file. User can pause and continue the recoding process. User can upload an existing audio file and skip the recording process. User can continue with watermarking process or direct export the audio file.
Watermark Embedding	User can embed silent watermark on the target audio file. User can check the quality of the watermarked audio file. User can view the spectrum diagram of the watermark. User can be redirected to watermark customization module to modify the watermark. User can export the audio as wav file with customized file name.
Watermark Detection	User can upload the existing audio file to the module. User can detect whether there is watermark exist in the audio file. User can decode the watermark embedded by the application itself.

b. Non-Functional Requirement

A non-functional requirement will specify the criteria that can be used to judge the operation of the proposed application. Hence, the discussion of the requirement is listed in Table 2 based on the non-functional requirement of the proposed application.

Table 2: Non-Functional Requirement

Requirement	Discussion
Performance of user interface and system flow	User should be redirected to the correct module depend on the user authorized.
Compatibility Readability	The proposed application should run in Android 11 or later version. The user interface should be clean and simple. All the text in the module should be clear and minimized.
Usability	The functional button should be clear. The flow of using the proposed application should be short and easy to understand.
Security	Users should register the account with correct username and password format. User should login with correct username and password. The password is encrypted using AES algorithm.
Testability	The function of proposed application can easily be test by watermark detector or audio editor with spectrum view.

3.2 Object-oriented Design Phase

In object-oriented design phase, the goal is to design the classes identified in the analysis phase. The object model will be built based on objects and their relationships. Then, the model will be refined in term of classes, attributes, methods, structures and associations. A class diagram will be constructed to give a clear image for implementation phase as shown as Figure 7. Moreover, the user case diagram and activity can help to know the application requirement and the data flow as shown as Figure 8 and Figure 9 respectively.

Figure 7 show the class diagram of the proposed application. There are three main class used to store the information of users, watermark and audio file. At the same time, there 2 sub class for Media class which used to store audio file which are Embedding and Detection.

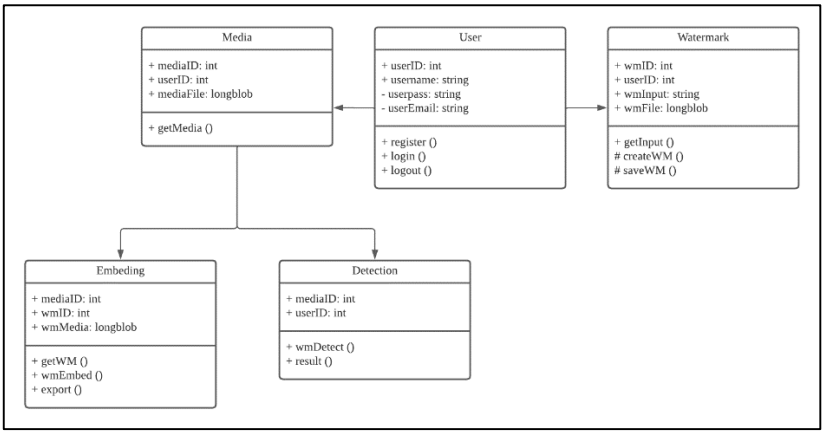

Figure 7: Class Diagram

Figure 8 show the user case diagram of the proposed application which name VWaterSMark. Users will first register or login into the application. For login user, a watermark text modifying step is required to confirm the watermark text. Then, user may choose to record the voice, encode the audio file or detect the digital watermark in the audio file. As long as no operation is performed, users may log out from the application.

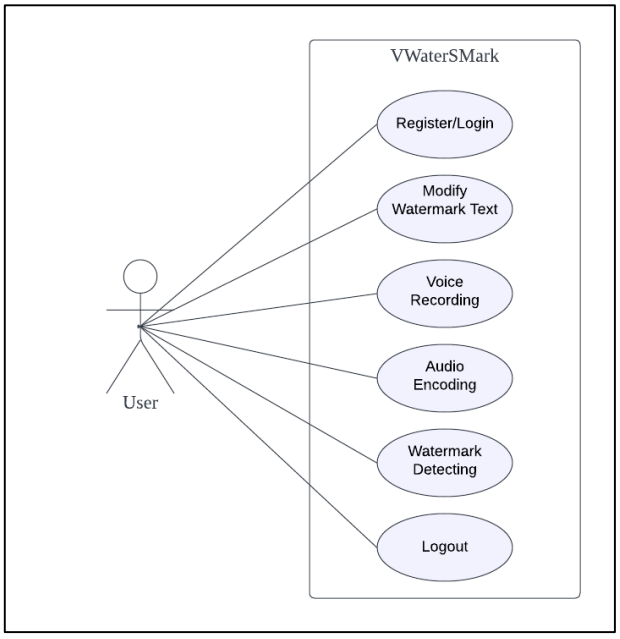

Figure 8: User Case Diagram

Figure 8 show the activity diagram which scale the system flow of the proposed application. The users will login or register to start the flow, after the first process, users will be redirected to watermark input page to create or update the watermark text. Then, users will finally enter the home page. Users can either redirect to voice recording page or watermark detection page. As users enter voice recording page, users need to record the voice or upload existing audio file. Users can either export the recorded voice directly or choose to watermark it. If users choose to watermark the recording or audio file, users will be redirected to watermark embedding page to watermark the audio file. For users chose the watermark detection page, they will get the detection result of no watermark or proposed application's watermark. After the operation is done, user will be redirected to the Home page again. Then, user can choose to log out or start the next watermarking or detection.

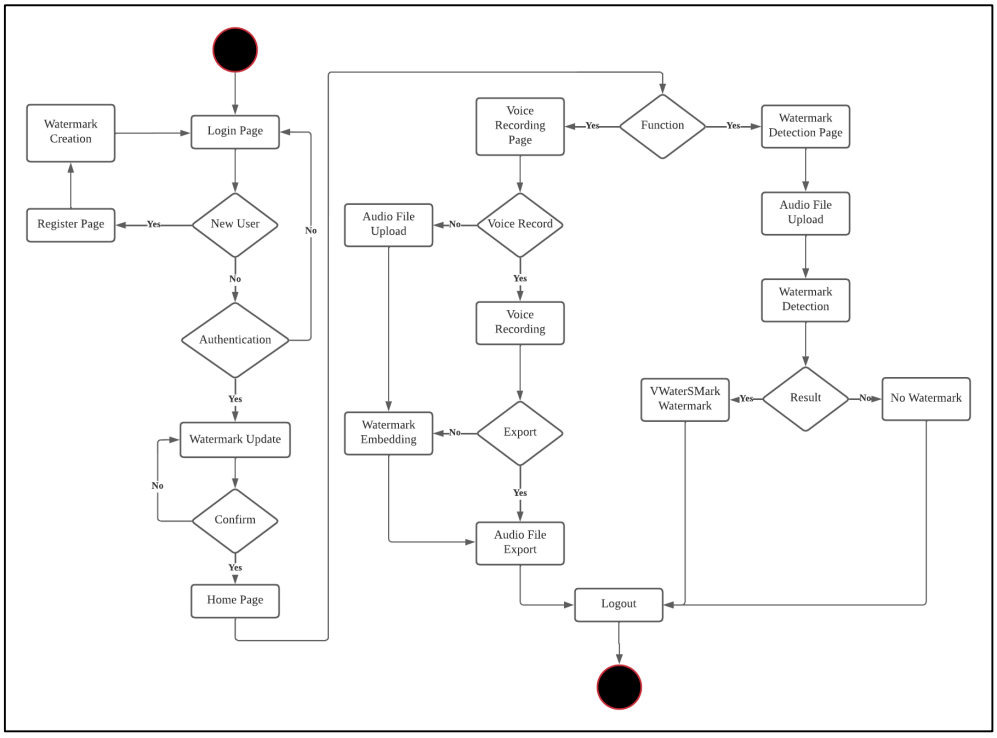

Figure 9: Data Flow Diagram Level 1

3.3 Object-oriented Implementation Phase

In object-oriented implementation phase, the proposed application starts to be implemented based on all the collected requirements and design which done in the previous phases. The interface will be developed using Thunkable and all the function will be developed using python programming language. The database management system used will be Airtable.

Figure 10 shows the Registration, Login and Watermark Confirmation Module of the proposed application. In the Registration Module, the users must fill in all the text fields which are email, password, confirm password, watermarking text and nickname to complete the registration step. If users are registered, users can be navigated to the login page by clicking on the Login bolded text. In the Login Module, the users must fill in all the text fields too which are email and password to complete the login step. If users are not registered yet, users can be navigated to the registration page by clicking on the Register bolded text. If the users' login successfully, they will be navigated to the Watermark Confirmation Module. In this module, the watermarking text shown as a modifiable text because this module enable user to modify their watermarking text. However, if users not decide to modify the watermarking text, users can click the Confirm button without doing anything to be navigated to the Home Module. If users decide to change the watermarking text, users can modify the text in the text field directly and click on the Confirm button to replace the text in the database and enter the Home Module.

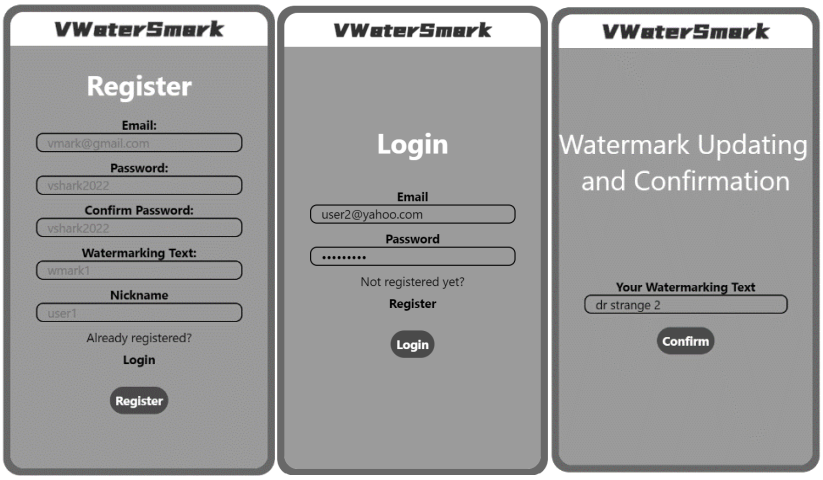

Figure 10: Registration, Login and Watermark Conformation Module

Figure 11 shows the interface of Home Module. As the Home Module is a module navigation center, users can choose to be navigated to Voice Recording Module, Watermark Encoding Module or Watermark Detection Module by clicking the buttons due to their needs at this module.

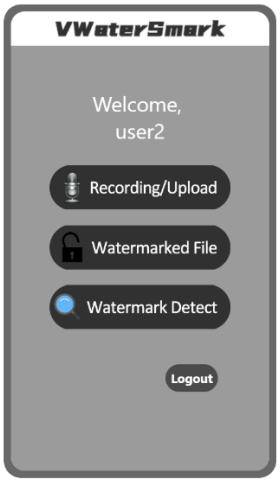

Figure 11: Home Module

Figure 12 shows the interface of Voice Recording Module. The Start button is used to start the voice recording while the Stop button is used to stop the recording. After the voice is recorded, the Play button can be used to play the recorded voice. For users who decide to watermark an existing audio file, the Upload button is used to upload the audio file. After the voice is recorded or the audio is uploaded, the Watermark button is used to send the request to watermark the audio file. The LSB algorithm used to watermark the audio file is shown as Figure 13. For the users who are not decided to watermark the voice, Download button can let user download the voice directly without the watermarking step. As the Download button is clicked, the block will show an alert box written with the Cloudinary media url of the audio file as shown as Figure 12. As the users click on the confirm button of the alert box, the users will be navigated to the preview interface of Cloudinary to download the audio file.

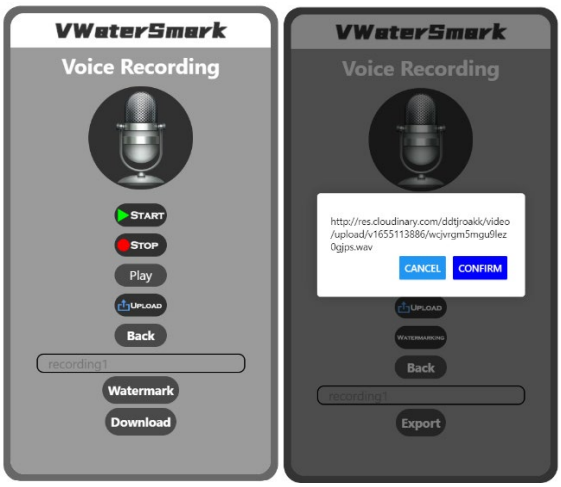

Figure 12: Voice Recording Module

```
    string = string + int((len(frame_bytes)-(len(string)*8*8))/8) *'#'
    bits = list(map(int, ''.join([bin(ord(i)).lstrip('0b').rjust(8,'0') for i in string])))
    for i, bit in enumerate(bits):
        frame_bytes[i] = (frame_bytes[i] & 254) | bit
    frame_modified = bytes(frame_bytes)
```

Figure 13: LSB Algorithm for Watermark Encoding

Figure 14 show the interface of Watermark Encoding Module. If the audio file is completely watermarked, the link will be displayed on this module, or else the module will display no record. The Back button is used to navigate back the Voice Recording Module while Download button used for audio file download purpose. As the same condition in the Voice Recording Module, when the Download button is clicked, the block will show an alert box written with the Cloudinary media url of the watermarked audio file to bring users to the preview interface of Cloudinary for download purpose.

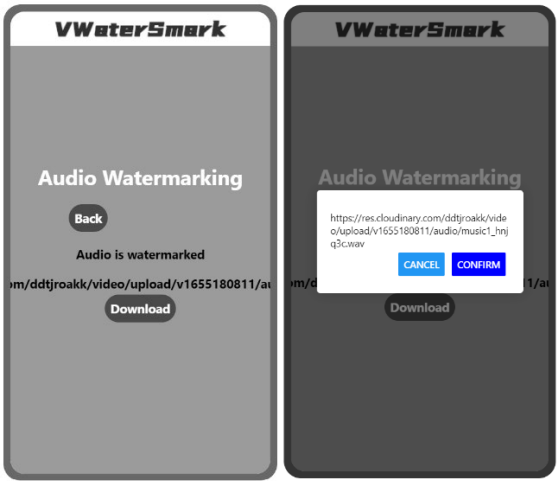

Figure 14: Watermark Encoding Module

Figure 15 shows the interface of Watermark Detection Module. This module request users to upload the audio file from device and decode the watermarking text in the audio file. If there is encoded text in the audio file, the decoded text will be showed in the result text field. If there is no watermark in the audio file, an error alert box will pop up. The LSB algorithm of watermark decoding is used to detect the digital watermark and showed as Figure 16.

Figure 15: Watermark Detection Module

```
audio = wave.open("dcfile.wav", mode='rb')
frame_bytes = bytearray(list(audio.readframes(audio.getnframes())))
extracted = [frame_bytes[i] & 1 for i in range(len(frame_bytes))]
string = "".join(chr(int("".join(map(str,extracted[i:i+8])),2)) for i in range(0,len(extracted),8))
decoded = string.split("###")[0]

json_dump = json.dumps(decoded)

return json_dump
```

Figure 16: LSB Algorithm for Watermark Decoding

3.4 Object-oriented Testing Phase

In object-oriented testing phase, all modules in the application are tested based on the function and security. The tests are carried out to ensure the application is error-free, secured and accepted by user.

To evaluate the proposed application's performance, the watermarked audio files are tested on 2 parameters which are noise and audio gain ratio. The noise of the watermarked audio is directly measure by listening carefully the playing audio file. After that, the result of the watermarked audio noise test is listed in Table 3. All the audio file watermarked by using Audacity has little noise as it watermarking theory is direct merging the target audio file with the watermark audio file which converted from the image file. However, all the watermarked audio file processed by using AWT2 has no noise. For proposed application, no noise is found for the watermarking of uploaded audio file but little noise is found for watermarked voice recording file. The noise may cause by the conversion of the file type from webm to wav file when the voice recording undergoes the encoding process.

Table 3: Comparison of Watermarked Audio Noise Between Existing Tools and Proposed Application

Tested Audio File	Audacity	AWT2	Proposed Application (VWaterSMark)
m1.wav	Little noise	No noise	No noise
m2.wav	Little noise	No noise	No noise
m2.wav	Little noise	No noise	No noise
Voice Record	Little noise	No noise	Little noise

The audio gain ratio is calculated through the formula

$$\text{Ratio} = \frac{\text{Gain of Watermarked Audio}}{\text{Gain of Original Audio}}$$

while the gain of the audio file is detected by using online file loudness meter which is Youlean. Then, the result of the watermarked audio gain ratio test is listed in Table 4. As the result stated, all the gain of audio file watermarked by using Audacity remain or become stronger. However, all the gain of the audio file watermarked by using AWT2 and proposed application remain or become weaker.

Table 4: Comparison of Audio Gain Ratio Between Existing Tools and Proposed Application

Tested Audio File	Audacity	AWT2	Proposed Application (VWaterSMark)
m1.wav	1.0354 LUFS	0.9985 LUFS	0.9993 LUFS
m2.wav	1.0424 LUFS	0.9754 LUFS	0.9874 LUFS
m2.wav	1.0241 LUFS	0.9842 LUFS	1 LUFS
Voice Record	1 LUFS	1 LUFS	0.9653 LUFS

Table 5 illustrates the summary of functional testing results for all modules. The test plan conducts different test cases for the main functions and security feature provided by the application. By summarizing the functional test results, the application will perform as expected based on the tests performed.

Table 5: Functional Test Result

Description	Expected Result	Actual Result
Create the watermark: i. Save the text input and the watermark will be generated follow it.	i. The watermark will be generated when the keyed in text input is saved.	Pass
Modify the watermark: i. Press the modify button to change the text input of watermark. ii. Save the text input and the watermark will be change follow it.	i. The new text input will be saved when the modify button is pressed. ii. The new watermark will be generated when the new text input is saved.	Pass
View the watermark: i. View the changing of the watermark spectrum diagram when the text input is changed.	i. The spectrum diagram of new generated watermark will be displayed when every new text input is saved.	Fail
Record the voice: i. Start record the voice when press on start button. ii. Pause the recording when press on pause button. iii. Continue the recording when press on start button when the recording is paused. iv. End the recording when press on stop button.	i. The voice recording process will be started by the start button, paused by the pause button, restarted by the start button and ended by the stop button respectively.	Pass
Upload the audio file: i. Upload the audio file when press on upload button.	i. The wav file will be uploaded by the upload button.	Pass
Export the recorded voice: i. Export the voice recording directly. ii. Name the voice recording.	i. The voice recording will be exported directly with customized name if users do not want to continue with watermarking process.	Fail

Table 5: (cont.)

Description	Expected Result	Actual Result
Embed the watermark: i. Embed the watermark audio file to the target audio file.	i. The target audio file will be watermarked with the watermark audio file created before.	Pass
Export the audio file: i. Export the watermarked audio file. ii. Name the audio file.	i. The watermarked audio file will be exported with customized name.	Pass
Upload the audio file: i. Upload the audio file when press on upload button.	i. The wav file will be uploaded by the upload button.	Pass
Detect the watermark: i. Detect weather an audio file is watermarked or not when press on detect button.	i. The uploaded audio file will be scanned to determine whether the audio file is watermarked by the proposed application, other tools or not watermarked.	Pass
Ensure the text input used to create watermark should be unique.	i. The text input could not be saved when it is already used by other in database. ii. The text input should be combination of numerical and alphabetical letters.	Fail
Ensure the error message exist when login fail not direct indicate which part of the authentication data is incorrect.	i. The proposed application will just show invalid input whenever the username or password is incorrect.	Pass
Enforce the complexity of the password.	i. The password should be combination of numerical and alphabetical letters. ii. The password length should be restricted between 8 to 12.	Fail
Password should be obscured in the textbox.	i. The keyed in password will be shown in "*" form in the textbox.	Pass

4. Results and Discussion

This section will discuss and show the outcome of the user acceptance test of the proposed application. User acceptance testing is conducted on five UTHM students to collect their responses towards the performance of the proposed application. The test will be carried on the Thunkable platform through live test function as the publishing of the Thunkable app need to be paid. Table 6 summaries the results of the user acceptance test. Five volunteers agreed that the system executes from start to end. All of them also agree that user able to register, login, create and modify watermark, record and export the voice recording, upload the file, embed and detect the watermark, export the watermarked audio file, error message for login failed did not indicate the exact wrong authentication data and password is obscured in the textbox. However, all of the volunteers disagree that user able to view the watermark, the watermarking text used is unique and the complexity of the password is enforced.

Table 6: User Acceptance and Security Testing Result

No	Acceptance and Security Requirement	Agree	Disagree
1	The system of the proposed application executes from start to end.	5	
2	User able to register and login to the account.	5	
3	User able to create the watermark.	5	
4	User able to modify the watermark.	5	
5	User able to view the watermark:		5
6	User able to record the voice.	5	
7	User able to upload the audio file.	5	

Table 6: (cont.)

No	Acceptance and Security Requirement	Agree	Disagree
8	User able to export the recorded voice.	5	
9	User able to embed the watermark to the target audio file.	5	
10	User able to export the watermarked audio file.	5	
11	User able to detect the watermark in the audio file.	5	
12	The text input used to create watermark is unique.		5
13	The error message exists when login fail not direct indicate which part of the authentication data is incorrect.	5	
14	Enforce the complexity of the password.		5
15	Password is obscured in the textbox.	5	

5. Conclusion

The significance of the project is to provide strong protection for the audio file but not disturbing the audio quality. The project objectives are to design a mobile audio recorder application using digital watermarking, to develop an audio watermarking application using Python language for Android platform, and to test the audio quality of the watermarked audio. The proposed application which is VWaterSMark is designed and developed by using the Python programming languages and Thunkable block. At the same time, the functional and user acceptance test of VWaterSMark is passed for most of the requirements. As stated in the project objectives, all of them have been achieved. As compare to other application, VWaterSMark has several advantages such as enable users to embed the digital watermark on audio file without affect the audio quality, decode the digital watermark anytime to show the ownership of the audio file and easily modify the watermarking text used to encode the audio file. However, the proposed application still cannot achieve the most ideal and perfect results due to the constraints in time, budget and skill. Hence, there are some recommendations for future improvements of the proposed application to obtain better performance including to restrict the creation and modifying of watermarking text to ensure the watermarking text is unique in the database, restrict the format of the registration password to ensure the users using strong password and increase the supported filetype and mobile operating system.

Acknowledgment

The authors would like to thank the Faculty of Computer Science and Information Technology, Universiti Tun Hussein Onn Malaysia for their support and encouragement throughout the process of conducting this project.

References

[1] World Intellectual Property Organization, Understanding Copyright and Related Rights, 2nd ed. Reading, MA: WIPO, 2016. [E-book] Available: Google Books.

[2] Dhavale, Sunita and Deodhar, Rajendra and Patnaik, Lalit, " Lossless Audio Watermarking Based on the Alpha Statistic Modulation," in The International journal of Multimedia & Its Applications, vol. 4, pp. 109-119, 2012, doi: 10.5121/ijma.2012.4410.

[3] Dhavale, S., Deodhar, R., Pradhan, D., & Patnaik, L., "Adaptive Quantization Index Modulation Audio Watermarking based on Fuzzy Inference System," in International Journal of Image, Graphics and Signal Processing, vol 6, pp. 1-11, 02 2014, doi:10.5815/ijigsp.2014.03.01.

[4] W. Bender, D. Gruhl, N. Morimoto and A. Lu, "Techniques for data hiding," in IBM Systems Journal, vol. 35, no. 3.4, pp. 313-336, 1996, doi: 10.1147/sj.353.0313.

[5] Stephan Wiefling, "Comparison of Audio Watermarking-Techniques". Master Hauptseminar Medientechnologie WS 15/16, Mar. 2016.

[6] P. Bassia, I. Pitas and N. Nikolaidis, "Robust audio watermarking in the time domain," in IEEE Transactions on Multimedia, vol. 3, no. 2, pp. 232-241, June 2001, doi: 10.1109/6046.923822.

[7] Hooman Nikmehr and Sina Tayefeh Hashemy, "A New Approach to Audio Watermarking Using Discrete Wavelet and Cosine Transforms". 2021.

[8] Roy, S., & Manasmita, M, "A novel approach to format based text steganography, " in Proceedings of the 2011 International Conference on Communication, Computing & Security - ICCCS ’11, pp.511-516, 2011, doi:10.1145/1947940.1948046.

[9] Cedric, T. M. M., Adi, R. W., & Mcloughlin, I, "Data concealment in audio using a nonlinear frequency distribution of PRBS coded data and frequency-domain LSB insertion," in 2000 TENCON Proceedings, vol. 1, pp 275-278, 2000, doi:10.1109/tencon.2000.893586.

[10] Roy, S., Parida, J., Singh, A. K., & Sairam, A. S, "Audio steganography using LSB encoding technique with increased capacity and bit error rate optimization," in Proceedings of the Second International Conference on Computational Science, Engineering and Information Technology - CCSEIT ’12, pp. 372-376, 2012, doi:10.1145/2393216.2393279.