A voice-based multi-platform first aid application using the Jaccard similarity index algorithm

Jenny Rose P. Guigue, Rey Gabrielle E. Pogado, Tiffany Claire C. Marata and *Meljohn V. Aborde
College of Computing Education, University of Mindanao, Davao City, Philippines
**Corresponding author:*
mjaborde@umindanao.edu.ph

Date received: September 12, 2019
Date accepted: December 4, 2019
Date published: December 20, 2019

ABSTRACT

The purpose of this study was to create a first aid application that would help users to be able to provide immediate assistance in times of emergencies. As the researchers developed the application, the researchers utilized the following technical tools React Native framework in creating a multi-platform app, React Native Voice API for voice recognition, Firebase Database, React Native Geolocation and Maps, React Native Immediate Phone Call and the implementation of Jaccard Similarity Index Algorithm. The researchers used the Agile Scrum Methodology for tracking down the development progress of the study. The researchers tested the application on both Android and iOS devices; the researchers also tested the app based on the location, distance, and how fast the app responded. As per the recommendation from the users, the researchers would suggest using Android mobile devices in installing the app since Android devices were flexible. It was also recommended to place the app near the user's mouth for better voice command detection, especially in crowded places.

***Keywords**: information technology; Jaccard similarity index algorithm; React Native; speech recognition, voice aid.*

INTRODUCTION

As today's time, technology plays a vital role in humans' daily lives, it aimed to make people's life more comfortable and hassle-free. In this generation, mobile phones became a necessity. Anyone can look for the things that he or she wanted and needed using just their mobile phones. Almost everything that people does right now has a corresponding application to make everyone's lives easier. One of technology's innovation is automatic speech recognition (ASR), also known as automatic voice recognition (AVR). Voice-controlled devices were popular these days, and a voice-driven application also became a trend; it is used to improve the traditional usage of mobile devices.

Accidents were unpredictable; it can happen anytime and anywhere. It may occur while working, studying, or even walking. Being knowledgeable about first aid was essential since it provided multiple benefits to people, such as being able to assist whenever accidents happen or facing emergencies (FAAE Team, 2018). Individuals who usually perform first aid were inclined with the medical field or had the grasp about the know-how in first aid. One of the main reasons for most deaths, especially involving kids, were caused by accidents not given first aid (Agoncillo, 2016). With these things at hand, problems related to first aid arose such as: First, was the correctness of the solution. When it comes to first aid, there was much information that can be found on the internet but does not give the assurance of its effectiveness. Also, solutions on the internet usually provided the user a general way of treating an injury

which does not apply to all injuries since some injury solutions vary from the situation such as injured patient's age, and its environment. Second, the availability of the solutions. In consideration to Davao city's state when it comes to internet connection, users do not have internet or data connection all the time which means that the user may not be able to look for a proper solution in treating the injury and may not be able to perform an immediate help to the injured person due to the absence of the internet. Next, the means of asking queries. When looking for solutions for specific injuries, the user's means of searching is by typing down their query or clicking through the set of queries provided on the screen which usually prolongs and slows down the user. Lastly, knowing the appropriate medical supplies. When accidents occur at home, and the user or the victim got a minor injury, in searching for solutions, results only provided the solutions alone excluding the medical supplies that the user may buy or used to perform the procedures of the solutions they have found.

The main goal of this research was to create a multi-platform application that would help people who lack the knowledge about first aid and to provide first aid solutions that individuals might follow whenever accidents occur, and whenever injured. The users can also use the app without worrying what language he/she used to come up with the correct solution. To attain the main objective, the following will be sought; Create an application that was accessible to both Android and iOS devices using React Native Framework; Recognize voice-driven commands to make it more convenient in making queries by using React Native Voice API; Recommend the possible best solution that the user may follow in doing the treatment based on the uttered query using the Jaccard Similarity Index Algorithm; Access the data from the Firebase database; Contact the user's chosen emergency contact either a friend or a family member that they may directly call when accidents occur by using the React Native Immediate Phone Call Plugin; Apply different languages aside from the English language such as Cebuano and Filipino for its voice commands to give comfort to the diverse users so that they may be able to choose the language they prefer, using Grouping of Common Words; And, Locate the nearby hospital or pharmacy locations to make it more useful to the users in guiding them where they may go when accidents occur or when they experience injury using React Native Geolocation and Maps API.

METHOD

The application was a cross-platform mobile app wherein it can both run on IOS and Android operating system. A React Native framework would be used to build a cross-platform native mobile app utilizing JavaScript language. The application received input through voice. The application would be able to receive and interpret dictation or to identify spoken word command/s using speech recognition; this was done using react native voice recognition library for iOS and Android with offline support. The detected speech input would be processed using Jaccard Similarity Index Algorithm, wherein it would find which was the closest solution based on the user's query from the predefined datasets, the closest predefined set outputted as the possible best solution or treatment. Given the solution, the user could rate on how helpful the solution and these ratings were stored into a Firebase Database, a cloud-hosted real-time synchronization database.

The application also gives a 2D map navigational view that showed the nearest hospital and pharmacy locations. Furthermore, the application also has an Emergency Contact feature that allowed the user to add a person to call during an emergency in just one click and a Q and A game that could test the user's knowledge about first aid treatments.

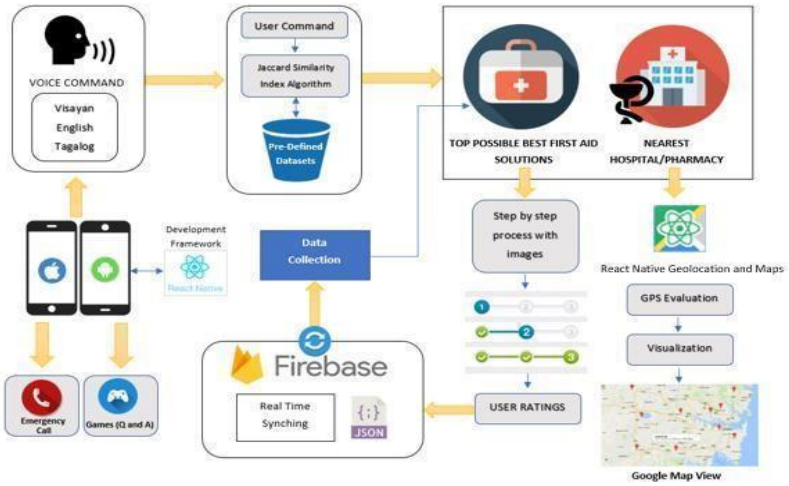

Figure 1. Conceptual Framework

The following development tools were used to develop the application:

React Native. The researchers used React Native for building a cross-platform mobile app. React Native was a framework for building native apps using React and JavaScript. Using React Native, it was easier to create a cross-platform (Android and IOS) application by just coding once. At its core, it provided some built-in components and APIs that were used in building the application.

Node.js and NPM. React Native required Node.js to had it installed. It was a JavaScript runtime environment that runs on various platforms. Installing Node.js, the Node Package Manager (NPM) program is installed on the computer. NPM was a command-line interface (CLI) which allowed installing other libraries.

Android SDK. The researchers utilized Android Software Development Kit (SDK), it was a tool for compiling the code to be able to run and test the application in a real Android operating system device created in react native.

XCode. XCode was an Integrated Development Environment for Mac OS developed by Apple. Just like Android SDK, XCode would be used to launch a mobile app in the iOS simulator.

Visual Studio Code. The researchers used the Visual Studio Code as the development platform. Visual Studio Code was a source code editor used for debugging and writing codes. The Visual Studio Code was used in editing codes for React Native in building a cross-platform native mobile app.

Firebase Database. The researchers used a Firebase database for storing and retrieving data. Firebase was a cloud-hosted real-time database that synced data to every user as long as internet connectivity was available.

Building a Native Cross-Platform application with React Native
The researchers used the concept of write once, run anywhere (WORA) using React Native Framework in creating an application by just coding once and running it on multiple platforms, especially Android and iOS. The purpose of this was to save time and effort on the programmer's part, thus creating a cross-platform app was a complicated thing to do. Utilizing React native framework in developing a native app

would be less hassle. React native allowed to create a native Android and iOS applications by only coding once. React Native also exposes JavaScript interfaces for platform APIs, so that the react-native apps could access platform features like user location, microphone, and many more.

There are four essential pieces in React Native architecture, first was the JavaScript where the developer would mostly write, next, the JavaScript Core and React Native which served as the bridge, lastly, the platform codes. The JavaScript could communicate with the native side through the bridge, which sent events from the JS side. Since JavaScript was an interpreted language, a JavaScript core runs the app, then entered the React and React Native code which transformed the React app into a tree of components to render and represented by unobstructed native views (Ballester, 2018).

React Native Voice API for Voice Recognition

The speech recognition was the ability of a machine to recognize a spoken word or phrases and converted it into a machine-readable format. It was commonly used in electronic devices to perform voice commands as input without pressing any button, keyboards, touching the screen, and so on. Speech recognition works using algorithm and modeling that recognized the temporal pattern in speech and must go through in several complicated steps to make a high accuracy of guessing a spoken word or phrase. Fortunately, the researchers do not need to go through with those complicated steps as React Native already has a voice recognition support for both iOS and Android.

Figure 2 showed the flow of how speech recognition works. To be able to accept voice as an input, the microphone permission was granted. Using the React Native Voice API, the inputted voice command was automatically processed and analyzed and was converted to text. The Jaccard Similarity Index Algorithm processed the converted text for formulating the solutions.

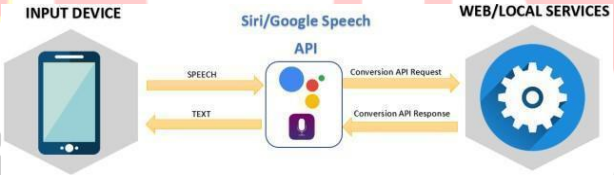

Figure 2. React Native Voice Recognition

Figure 3 showed the primary process of React Native for accessing Siri and Google Speech API. Initially, the React Native Voice library wrapped the official API by Apple and Google; thus, algorithms were all proprietary assets of Apple and Google each which was not open to the public. Automatic Speech Recognition may vary on the platform used; Android relied on Google while iOS relies on Siri. The primary process of recognizing speech was through accessing these API's, but more importantly, permission must be granted first to access this functionality. Once the device recognizes a speech input, the API would automatically convert this request to the services. There were two parts of services, the local and the web; these services only differ upon connectivity, the web could is acquired if the device has online connectivity while local when not. When it came to performance, web services were more efficient since more data were stored in the server, unlike local service that varies on internal offline speech recognition support (React Native Voice Recognition Library for iOS and Android (Online and Offline Support), n.d.).

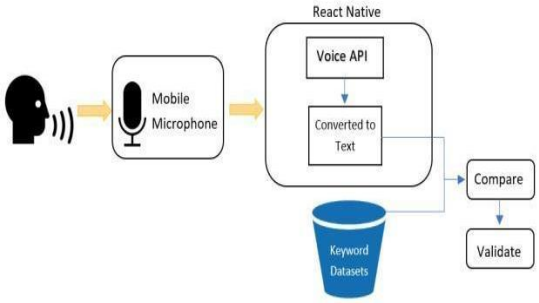

Figure 3. Access to Speech API Process

Algorithm for finding the best solution

Figure 4 showed the process of finding the best solution using the Jaccard Similarity Index Algorithm. It was done by comparing the user input to the predefined sets individually using the Jaccard Similarity Index to determine the set that satisfies the user input from the predefined sets. Every predefined set had a similar solution, and then set with the highest result would be declared as the best solution. But in cases that would have a multiple set with highest result and the object in the predefined set doesn't occur in the user input, a follow up question would be prompted from the application of which missing object from the set needed to be answered to eliminate sets and to attain only a single set with the highest result.

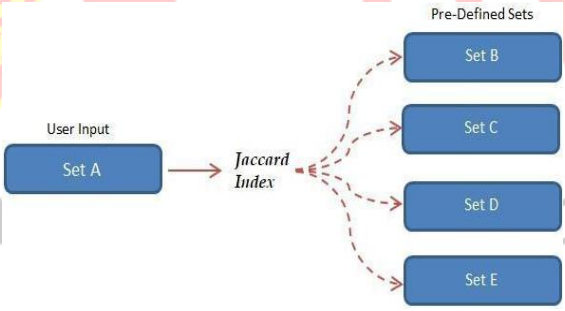

Figure 4 The process flow for finding the best solution

Table 1 showed the computed Jaccard Coefficient of each sample predefined sets from set B to E and its similar percentage. The computed Jaccard Coefficient of each set was used to compare all the results to determine the best solution. The set B has 57%, set C has 57%, set D has 75% and set E has 75%. It showed that both set D and E has the highest percentage, which means that either set D and E is the possible best solution based on the calculated Jaccard Coefficient. Since there are two highest results, there is a follow-up question for the user enable to have one highest result and to give the best first solution. The question is given based on the user input dataset, it was then compared in the question injury dataset, and the choices were based on the objects from highest dataset results that did not occur in the user input dataset. So the user input dataset is updated and repeat the process from computing the updated input dataset to all datasets in injury datasets.

Table 1. Comparing all sample predefined data sets based on the computed Jaccard coefficient

Set	Object/Keywords	Jaccard Coefficient	Result
B	Drowning, child, unconscious	.57	57%
C	Drowning, child, conscious	.57	57%
D	Choke, child, conscious	.88	88%
E	Choke, child, unconscious	.88	88%

The process from voice command to the first-aid solution

Figure 5 showed the process from voice command to the first-aid solution. To deeply understand the process, the researcher would provide sample data with the user speech input "nachoke ang bata." The first phase of the process was the user speech input would be converted into text using React Voice API. After converting the speech into text, the converted text would go through into word checking using the group of common words that the researchers had set and checked the word if it exists and got the equivalent English interpretation to form a new user input dataset which would be compared to all injury sets using Jaccard Similarity Index Algorithm to find the best solution. After the word checking process, the updated user input dataset was {choke, child} and it would get the primary injury in user input dataset used in later part of the process, and compared to the injury datasets using the Jaccard Similarity Index Algorithm.

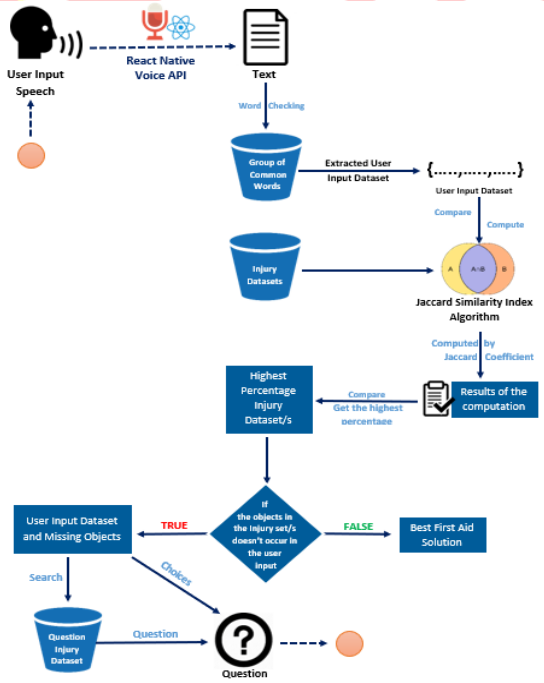

Figure 5. The process from voice command to the first-aid solution

Based on the result from table 1 both set D and E has the highest percentage of similarity from the user input but to give which one of the two sets was the best first aid solution there would be a follow-up question to the user. The basis deriving questions for the user is through comparing input dataset to the question injury dataset. Also, for the choices, the primary injury in user input that occurs in the injury dataset that got the highest result was the only objects used that does not occur from the user input dataset, and those objects would be defined as a missing object. In the given sample data of user input {choke, child} and the two predefined sets with the highest result set D and B with missing object {unconscious, conscious} those missing objects by the choices of the question "What is the state of the casualty?". This time the user must answer the question base to the given choices. After answering the question, the user input dataset becomes up-to-date, and it would repeat the process from word checking until the highest predefined set has no more missing object, and the result of its Jaccard coefficient may have a result of 100%, or until the result has only one highest injury dataset it means that would be the best solution based on the user input.

Firebase Integration

Firebase was a cloud-hosted, real-time synchronization database storage engine and offline data modification.

Figure 6 showed the process of data synchronization of the application to firebase. It was real-time data synchronization, and when the user's device goes offline, the data would be stored locally from the last update. Thus, there was still persistence in the data. When the user device came back online, it automatically synchronized local data to the new update. Upon giving the solution to the user, there would be a rating for evaluating how helpful the solution was. Those ratings from different users in that specific solution automatically are updated in the firebase database.

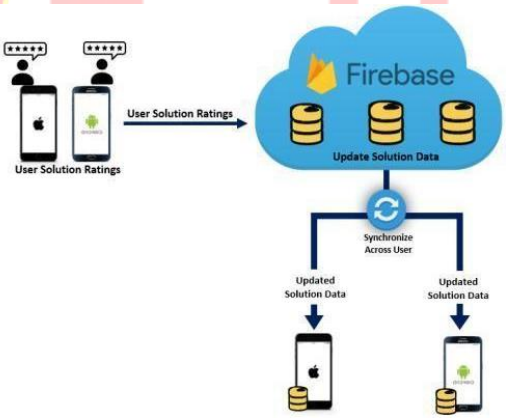

Figure 6. The process flow of Firebase Synchronization

React Native Geolocation and Maps

Figure 7 Locating nearby Hospital and Pharmacy was one of the features of the application. The researcher used React Native Geolocation, React Native Maps, Google Direction API, and Maps SDK for Android and iOS. The React Native Geolocation was used to get the current location of the device by calling the getCurrentPosition method. Geolocation used coordinates of longitude and latitude to find the user's current position/location. The React Native Maps was used to render Google map for displaying the maps and street view imagery where markers were plotted on the map depending on the latitude and

longitude coordinates. To get the routes to the hospital and pharmacy from the user's location and to calculate the direction between locations, the researcher would use Google Direction API. Since the application was a multi-platform, we used Maps SDK for each platform, which was the Maps SDK for Android and Maps SDK for iOS. These Maps SDK is used to add maps in the application based on Google Maps data. Maps SDK can automatically handle in accessing the Google Maps servers, downloading data, map displays, and response to any maps gesture. These objects also can give more additional map location data or information and enable user interaction with the map.

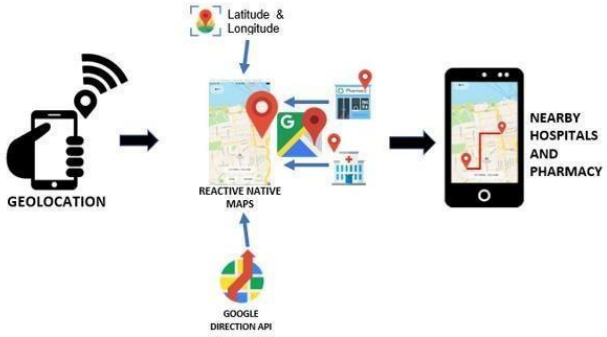

Figure 7. Process in getting the Nearby Hospital and Pharmacy

React Native Immediate Phone Call

The researchers used React Native Phone Call to make the emergency contact feature. React Native allowed to initiate an immediate phone call (without further user interaction) in the application without accessing the phone subsystem.

Figure 8 showed the methodology the researchers used in developing the application. The researchers used the Agile Scrum Methodology. The researchers started the project by determining and gathering all the data needed, aside from the app's content, the researchers also looked for what were the appropriate tools that they should use to make the application and what algorithm should they utilize to achieve the app's objectives. After gathering all the necessary data, the researchers proceeded in the designing phase; the researchers designed the structure of the application's database to properly store and manage the data that the app would require. Aside from designing the database, the researchers also designed the application's system flow to determine how the application would work and also to determine what were the pages/screens that they would be needing. Together with the database structure design and the system flow, the researchers also designed the application's interface since system interfaces played a vital role in attracting the user to utilize the app. Next, the researchers continued in developing the application. The researchers had done this phase by dividing the application into functions/features. Every function that the researcher would accomplish, they would proceed in testing it and looking for errors and bugs. Whenever the researchers would find some, they would proceed on implementing bug fixes on the found errors. The conducted the following tests to check the capability of the application created: android compatibility testing, iOS compatibility testing, app response testing, beta testing, alpha testing, and acceptance testing. This iterative process continues until all the features are satisfied. After implementing all the functionalities of the application, the researchers would make a review on the application's complete functionality before it would be published. The researchers published the application in Google Play after the development of the project.

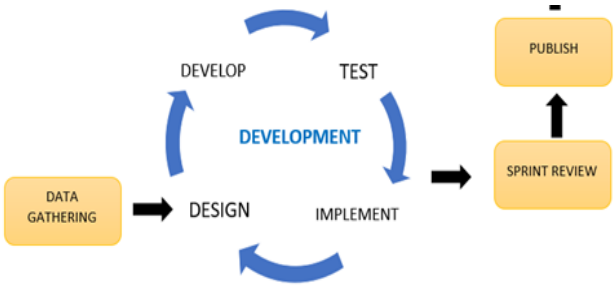

Figure 8. Agile Scrum Methodology

RESULTS

Android Compatibility Test Results

The researchers installed the application on different Android mobile devices with different OS versions. The app is compatible with all the devices mentioned in table 11. The application works efficiently on all the Samsung devices and Asus Zenfone3 that the researchers used for testing. There was a visible difference in the app's appearance. The text on the application was smaller in Samsung J7 Pro compare to the text in Samsung J7, J7 Prime, and Asus Zenfone3.

iOS Compatibility Test Results

While on Apple devices especially on iPad mini 4 where the app cannot be displayed in full screen and on iPhone 5s, the icon on the app where so small which makes it hard to click on it. As for the app's response, iPhone 5s was slower compared to iPad Mini 4. Lastly, for iPad Mini 1, the app cannot run on it since it keeps on crashing, the Apple device version must be more than version 8.2.

App Response Test Results

Samsung J7 Pro was the fastest to respond after the user uttered the command in an average loudness of the voice within a 1m distance and a noise decibel of 51.7dB. Next location was the noisy classroom with a decibel measurement of 70.6dB, Samsung J7 Pro was the fastest to respond to having the user to utter the commands loudly and adequately. For a running jeepney with a decibel measurement of 68.5dB, Samsung J7 Prime comes first on how fast the app responds to user commands, making the rest of the devices to make the user's voice louder than his/her voice natural loudness. Lastly for a busy fast food and a school gymnasium having a program with a decibel measurement of 80dB and 96.4dB respectively, on these locations no matter how loud moreover, explicit user's voice in uttering the commands the app cannot still correctly recognize the commands that is why the researchers would suggest navigating the app manually in these kinds of places. Based on the results provided, the researchers could conclude that Android devices are better in installing the app compared to iOS devices.

Acceptance Testing Result

The BLS Instructors from Red Cross tested the application. They were impressed by the features included in the application, and the list of injuries that are on the app was higher than the usual content on other existing first aid applications. They commended the app's language selection feature, which was unique because other application does not offer Filipino and Cebuano languages which hinders the users to appreciate the application because of the language barrier. They would like to emphasize the importance of proofreading because there were quite a few typographical errors on the prompts. They were not that satisfied with the app's responsiveness when it comes to voice command navigation, especially in crowded places because it does not detect the commands.

CONCLUSION

The researchers were able to develop an application that runs on the two most leading mobile platforms of today, the Android and iOS. The application can recognize voice commands that the researchers had set to the library that they had used for the voice recognition was not 100 percent accurate for Filipino and Cebuano but still the app functions well. The utilization of the Jaccard Similarity Index algorithm for the derivation of the best possible first aid solution for the injury. The users can cast votes the solutions they wanted to vote with the use of real-time data synchronization of Firebase Database. Users may choose the language they prefer in utilizing the app, and they can also add up to three contacts that they may directly call in times of emergency. Aside from the things mentioned, the app also provides hospital and pharmacy locations and the route in going there.

RECOMMENDATION

The researchers would recommend users to install the app in Android mobile phones since it is more compatible especially phones such as Samsung J7, J7 Prime and J7 Pro for it could detect commands way better than the other devices even if the user is far from the device and is in crowded places except for areas that are close and too noisy that made the app's voice detection capability non-functional. The researchers would also recommend using the app near the mouth of the user while uttering the commands well. The application works best on places that are not that close and crowded thus the app's voice command detection still works well in crowded places if the user would utter the commands clearly, louder and the devices must be near the user's mouth. As for the future researchers, it is recommended to create or have a library for voice recognition which may be explored more and used it the way with fewer restrictions. As the research goes and continues, the future researchers should be open to possible changes that the application, the user, the consultant, and validators would require. The researchers would also recommend to the future researchers to include the voice-over capabilities of the application in Filipino and Cebuano languages which was the app at the moment cannot do.

The application's validators which were BLS Instructors, recommended that the app's size must be reduced to encourage more users in downloading the app. They also recommended that the solution should be shown in a different manner catering all the ways on how people learn such as those who could learn with the help of images and text, auditory learners, kinetic learners, and audiovisual learners.

REFERENCES

Agoncillo, J. (2016, September 19). *Starting them Young: Red Cross Conducts First Aid Training for Kids.* Philippine Daily Inquirer.

Ballester, A. M. (2018, January 18). *Building Your Application with React Native*. Retrieved from https://www.youtube.com/watch?v=kitMA1szfcU

FAAE Team. (2018). *First Aid Accident and Emergency. Why is First Aid Important?* Retrieved from https://www.firstaidae.com.au/about-first-aid-ae/why-is-firstaid-important/

React Native Voice Recognition Lin=brary for iOS and Android (Online and Offline Support). (n.d.). Retrieved from https://github.com/weakesj/react-native-voice.