University of the Philippines Manila
College of Arts and Sciences
Department of Physical Sciences and Mathematics

Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

A special problem in partial fulfillment
of the requirements for the degree of
Bachelor of Science in Computer Science

Submitted by:
John Bengemin Sih Uy
May 2018

Permission is given for the following people to have access to this SP:

Available to the general public	Yes
Available only after consultation with author/SP adviser	No
Available only to those bound by confidentiality agreement	No

ACCEPTANCE SHEET

The Special Problem entitled "Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application" prepared and submitted by John Bengemin Sih Uy in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science has been examined and is recommended for acceptance.

Vincent Peter C. Magboo, M.D., M.S.
Adviser

EXAMINERS:

	Approved	**Disapproved**
1. Gregorio B. Baes, Ph.D. (*cand.*)	_____	_____
2. Avegail D. Carpio, M.Sc.	_____	_____
3. Richard Bryann L. Chua, Ph.D. (*cand.*)	_____	_____
4. Perlita E. Gasmen, M.Sc. (*cand.*)	_____	_____
5. Marvin John C. Ignacio, M.Sc. (*cand.*)	_____	_____
6. Ma. Sheila A. Magboo, M.Sc.	_____	_____
7. Geoffrey A. Solano, Ph.D (*cand.*)	_____	_____

Accepted and approved as partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science.

Ma. Sheila A. Magboo, M.Sc.
Unit Head
Mathematical and Computing Sciences Unit
Department of Physical Sciences
and Mathematics

Marcelina B. Lirazan, Ph.D.
Chair
Department of Physical Sciences
and Mathematics

Leonardo R. Estacio Jr., Ph.D.
Dean
College of Arts and Sciences

Abstract

The Philippines' blood donor count at the present time is unable to reach quota needed to have sufficient blood supply due to the lack of blood donors which can be rooted to the low awareness of people with regards to blood donation. This project's purpose is to create a mobile application equipped with the information the Philippine National Red Cross provides to a prospective blood donor in order to donate blood consequently and to increase awareness for blood donation.

Keywords: Blood donation, Red Cross, Mobile Application

Contents

Acceptance Sheet i

Abstract ii

I. Introduction 1

A. Background of the Study . . . 1

B. Statement of the Problem . . . 3

C. Objectives of the Study . . . 4

D. Significance of the Project . . . 5

E. Scope and Limitations . . . 6

F. Assumptions . . . 6

II. Review of Related Literature 7

III. Theoretical Framework 10

A. Philippine Red Cross . . . 10

B. Blood Donation . . . 11

C. Blood Request . . . 13

D. Information System . . . 13

E. Mobile Application . . . 14

F. Firebase Application . . . 14

G. Global Positioning System . . . 14

IV. Design and Implementation 15

A. Use Case Diagram . . . 15

B. Flowchart: Web Application . . . 16

C. Flowchart: Mobile Application . . . 17

D. Entity Relationship Diagram . . . 18

V. Results 20

VI. Discussions 37

VII. Conclusions **40**

VIII. Recommendations **41**

IX. Bibliography **42**

X. Appendix **44**

..1 FAQ.java 44

..2 FAQAdapter.java 44

..3 FAQsActivity.java 45

..4 GMailSender.java 46

..5 HelpActivity.java 49

..6 History.java 49

..7 HistoryActivity.java 49

..8 HistoryAdapter.java 52

..9 InquiryActivity.java 53

..10 JSSEProvider.java 56

..11 MainActivity.java 56

..12 News.java 63

..13 NewsActivity.java 64

..14 NewsAdapter.java 66

..15 PRCBranch.java 67

..16 PRCBranchComparator.java 69

..17 Question.java 70

..18 QuestionActivity.java 71

..19 QuestionAdapter.java 73

..20 SplashActivity.java 75

..21 WaveHelper.java 77

XI. Acknowledgement **79**

I. Introduction

A. Background of the Study

Blood transfusion is a process used in the medical field to replace lost or damaged blood of various type of patients. For example, in events such as but not limited to patients who experienced road accidents, or even the need of organ transplants, hospitals require the relatives or guardians of the patients to gather a number of blood bags that will be used during the operation for the patient. Given such a situation, the whole blood transfusion involves two parties, namely, the blood donor, and the blood recipient. A blood recipient pertains to the patient who needs the blood bags for the operation he or she is about to undergo to. The blood recipient, normally is a patient suffering from severe blood loss, leukemic patients, hemophilic patients, etc [1]. On the other hand, a blood donor would be a standardized healthy person who could supply the source of blood that would be used in the said procedure. According to the Donor Recruitment Head of the Philippine Red Cross, the World Health Organization suggests that a country needs to have at least ten percent of its total population as blood donors to have an adequate blood supply, otherwise, not all blood recipient may receive their needed blood components [2].

Blood donation plays a vital role in blood transfusion as in fact, without a blood donor, the patient, since is not able to receive the needed blood components, may then be denied of his or her operation because of the lack of blood components needed to do the operation. In the worst case scenario, the denial of the operation may lead to the death of the patient. Because of this, organizations within the country are created with the goal of becoming blood banks, which would store numbers of blood components. After which, these organizations will also serve as suppliers to patients needing blood components.

In the Philippines, one of the organizations that handles most of the supply needed for blood transfusion needs is the Philippines Red Cross. According to the Department of Health in the Philippines, the Philippine Red Cross has supplied more than twenty five percent of blood units of the nation's total dispatched blood units in the previous year [3].

However, despite this achievement, it is also recorded that the Philippines is still unable to meet the standard of the World Health Organization that states the percentage of the nation's population to be blood donors [2]. Because of this, the Philippines Red Cross is unable to cater to everyone who asks for blood units, rejecting blood requests from various patients when there is a lack of supply of blood units.

The Philippine Red Cross, given that they engage in the supply chain for blood components, follow the Republic Act of 7719 that states everything about the service relating to blood transfusions. Republic Act of 7719 is also known as the National Blood Services Act of 1994, and it states its mission to ensure blood safety, blood adequacy, and efficient blood services and aims to develop a fully voluntary system for blood donation, increase the efficiency of blood service facilities by making the blood test and processing centralized, implement quality management system that includes good management system and management information system, maximizing blood usage through a rational use of the blood products and component therapy, lastly, creating a realistic but sustainable management and funding for the nationally coordinated blood network. In effect, given such missions and aims, the Department of Health in the Philippines does not encourage a donor-recipient contact, because it then may contradict the aim of the National Blood Services Act of 1994 to also let blood donors to do blood donation voluntarily [4].

Given such a setting, when one person is in need of blood components, according to the Philippine Red Cross, will be given by their doctor or hospital's nurse a signed blood request form. After that, they must go to a submit the blood request form to a Philippine Red Cross office. If the blood components requested are available, the requester will be asked to pay for handling or maintenance fee, only after is Philippine Red Cross able to dispatch the blood bag containing the needed components. Otherwise, in the event that the blood components are not available, the Philippine Red Cross advises the requester to request from other blood banks [2].

Moving on to the perspective of blood donors, when one desires to donate blood to the Philippine Red Cross, they must go to an office of the Philippine Red Cross. The

aspiring donor must then undergo a screening process based on his or her background and physical condition such as history of drug use, alcoholism, or tattooing and diseases like Syphilis, a sexually transmitted disease, Hepatitis B virus, Human Immunodeficiency virus or commonly known as HIV, just to name a few. If the prospect donor is able to pass the examinations, the Philippine Red Cross will take note of his or her blood type. Furthermore, if the donor has a very rare blood type RH^-, the Philippine Red Cross would ask for the donor's contact details so that the donor will be asked to donate a later time when his or her blood type is needed to prevent wasting the donor's blood donation. After which, the Philippine Red Cross further instructs the donor of post-donation education and finally taking the donor's first blood donation. According to Philippine Red Cross, they are capable of handling two methods of blood donations. These are conventional and apheresis donations or extraction methods. The key difference between the two is that conventional blood extraction takes in the entire blood which will be separated later into different unit components while apheresis blood extraction only takes in a specific blood component while the rest are returned to the blood donor [2].

The Philippine Red Cross, aware of the problem where the Philippines does not have the recommended number of blood donors, wants to increase the awareness of blood donation in the Philippines. Currently, they use traditional means like creating brochures, participating in outreach programs, and even engaging in the social media in order to increase awareness for blood donation, stating the benefits of blood donation, the requirements of being a blood donor, and the procedure in donating blood where the Philippine Red Cross ultimately hopes to resolve the lack of blood donors in the country [2].

B. Statement of the Problem

Even though the Philippine Red Cross has done various methods to increase awareness for the need of blood donors in the country, a lot of the citizens of the Philippines still are unaware of the current predicament which results to the country still not having the recommended number of blood donors for the circulating blood supply used in blood transfusion

over the country. The people of the Philippines are also unaware of the benefits of donating blood that also does not help in motivating Filipinos in becoming blood donors. Because of this, there is a lack of blood units in blood banks and consequently, not all blood recipients are given the blood components they need in their operations. Ultimately, the denial of the operation may cost the patient his or her life.

C. Objectives of the Study

This research provides a mobile application that equips users the information they need for donating blood and encourage them in doing so. Furthermore, the research also provides a tool allowing Philippine Red Cross employees to update information for the mobile application. Specifically, the mobile application and web application has the following functionalities:

1. Using the mobile application
    (a) Allows user to view details of the different Philippine Red Cross branches such as:
        i. How much blood unit a facility still has in terms of a graphical representation.
        ii. What blood extraction methods the facility is able to perform.
        iii. When the facility is open or closed.
        iv. What are the contact details of the facility
        v. Where it is located geographically.
    (b) Allows user to view various materials on blood donation and outreach programs from the Philippine Red Cross such as:
        i. News
        ii. Announcements
        iii. Promotional Materials
        iv. Frequently Asked Questions

(c) Allows user to ask other questions regarding blood donation or blood requesting by filling out and submitting a form.

(d) Allows user to view and answer a sample Donor History Questionnaire that will do either of the following:

i. Shows qualification of user to be a blood donor.

ii. Suggests things to do for the user to be a valid blood donor.

iii. Shows the things preventing the user in becoming a blood donor.

(e) Allows user (those who have passed the blood donation screening tests) to input serial code that will identify user as a blood donor and unlock more features such as:

i. Input latest blood donation and its type.

ii. View donation history.

iii. See if user can donate at the time of usage.

2. Using web tool (for Philippine Red Cross employees)

(a) Allows posting of various materials regarding blood donation for the mobile app:

i. Post news

ii. Post announcements

iii. Post promotional materials

iv. Post new frequently answered questions

v. Post questions for the donor history questionnaire

(b) Allows the editing of Philippine Red Cross Chapters' information

(c) Allows posting of new serial codes for mobile application users

(d) Allows the addition of new donation histories for mobile application users

D. Significance of the Project

Using the mobile and web tool, Philippine Red Cross has an easier time disseminating information involving any blood service in the Philippines. This helps in increasing awareness

regarding blood donation in the country that in turn increases the number of blood donors in the country. This leads to Philippines reaching the recommended number of blood donors to have a sufficient supply of blood units within the country. Ultimately, this allows the lessening of rejected blood requests and deaths due to not having enough blood units for operations.

E. Scope and Limitations

1. Information seen by the public is solely limited to the information released by the Philippine Red Cross.
2. Actual blood donation is not supervised by the application, verification of blood donation is manually done by Philippine Red Cross medical team.
3. Transaction between requester and Philippine Red Cross for handling the blood components is not be part of the system.
4. Personal information is not stored online as per request by the Philippine Red Cross
5. Every information in the mobile application is tailored with respect to the Philippine setting.
6. The verification of adding serial codes for mobile users is not included in the system.

F. Assumptions

1. Normal users have an Android smartphone having minimum version 4.1
2. Information is readily supplied by the Philippine Red Cross

II. Review of Related Literature

Blood is requested because of various reasons, from the data gathered in 2014, more than sixty percent of blood units were used for treating medical conditions such as anaemia, cancer and other blood disorders, around twenty percent were used for various types of surgeries, while less then ten percent were used to treat blood loss after child birth [5]. This just shows how great is the demand for blood components as it is used in a lot of ways.

The concept of using mobile applications in handling or encouraging blood donation is not new. In fact, a number of researchers have already created a compilation of existing applications in the Google Play Store, Apple Apps Store, Blackberry App World and Windows Mobile App Store that existed in 2015 which numbered around 169 mobile applications spread across various mobile operating systems [6].

In fact, using mobile applications for blood related services have continued to grow over the years that countries are starting to create their own official blood donation management mobile application. An example of this would be Bangladesh's mobile application that allows easy monitoring of blood donation activities by also being able to access donor records across the country [7]. This also shows how mobile blood donation applications can be useful in various ways as even different countries try to implement their blood related services with mobile applications.

Currently, these mobile applications designed for blood related services continue to evolve by using modern technology in handling said services. An example is using the current cloud computing design which houses data and information on the cloud where the host need not prepare a hardware for their own server but outsource companies that would provide servers for them to use. Such technology was used by Mostafa and his co-researchers when they created their Blood Donation System framework [8].

Some applications even use Global Positioning System capabilities of the smartphones to provide realtime location data for blood related services. An example of such would be

the mobile application developed by Fahim and his team that locates the nearest compatible blood donor in the area [9]. According to the researchers, this provided an easy and fast way to find a donor which in conclusion saves time and lives.

There is also a new type of blood related mobile applications where they implement gaming mechanics in the form of rewards and badges. In particular, there exists a study where the researchers compiled a number of gamified mobile applications that are aimed at handling blood related services. Aside from having functionalities such as tracking donors, these applications also give out badges and redeemable points to blood volunteers as incentive [10]. This is due to the fact that even though these mobile applications are convenient in finding volunteers, it will not change the fact that some people are not interested in donating blood.

This calls for needing to discuss motivating normal people to become blood donors. Chell and her group mates showed various types of incentives and analyzed which objects motivate people to become blood donors [11]. The conclusion during the research was that appreciating donors count as a low level of encouragement. People tend to get motivated given that movie tickets, health checks, or even blood tests are given to them as incentives.

Such an issue was needed to be solved because of the rising population of developed countries in this time and age. In fact, according to a study by Williamson and her team, it is predicted that using models that in the span of five to ten years, blood availability will once again be an issue [12]. Hence the need to discuss ways on how to attract more people to become donors or increase awareness regarding the need for blood donors.

Other issues why a low number of people are donating blood is discussed by Setiawan and Putra in their project where it is shown that distance of the prospect blood donor from the blood facility and the prospect blood donor not being asked to donate blood are some of many reasons why they do not donate blood [13]. In order to answer both of these problems, it is clear that one must be able to see the nearest locations for donating blood and to spread news or information regarding blood donation to various areas so that people

may be aware.

In the Philippines, the law that handles the blood related services is the Republic Act of 7719. Researches describe the current methodologies of said services as one that encourages voluntary, not reward based, donation of blood to ensure a sufficient supply of blood in the country [14]. An opposite to the status of blood donation at other countries where the foreign countries try to implement a reward system to ensure a sufficient blood supply whereas the Philippines insist on a non reward based blood donation system.

Another thing to be considered in the Philippines is the Data Privacy Act, which is stated in the Republic Act of 10173. When compared to other countries such as South Korea, is described as currently inoperative unless a data protection authority is present to take responsible actions [15]. At the moment, organizations are finding it hard to deploy applications that would require getting information from its users because of the Data Privacy Act implemented during the 2012.

III. Theoretical Framework

A. Philippine Red Cross

The Philippine Red Cross that was founded in 1947 is a non-profitable organization in the Philippines patterned around the International Red Cross, provides six major types of operations. Specifically, these are Disaster Management Services, Safety Services, Health Services, Social Services, Red Cross Youth and Volunteer Services, and lastly, Blood Services [16].

The Philippine Red Cross follows seven principles. Namely and concisely, these are the following. First, humanity which simply implies on being able to stop human suffering whenever possible. Second, impartiality which shows no discrimination to any topic that brings difference to people. Third, neutrality which prevents affiliated to Philippine Red Cross from taking any sides in a conflict. Fourth, independence which gives autonomy to the organization despite needing to follow the law of the country. Fifth, voluntary service which drives Red Cross volunteers to not let any desire be the core of their services. Sixth, unity which brings together all humanitarians of Philippine Red Cross. Lastly, universality which introduces the Red Cross as a worldwide community [17].

Currently, the number of chapters or branches of the Philippine Red Cross in the country numbers 89, with its main branch stationed here in Metro Manila. Each chapter is assigned a specific representative who would monitor assigned duties of the chapter and report to the headquarters. In relation to Blood Services, a report sheet is needed to be filled and passed every day that would require information regarding number of blood units dispatched, blood units collected, etc [2].

In order to spread awareness for Blood Services related tasks, the Philippine Red Cross uses various methods and materials. Among these, they have a website containing most information about their organization that also lists their contact information. They also use Facebook in order to answer questions from various Filipinos. Then, they also use collaterals such as posters and brochures when they go to outreach programs [2].

With respect to Blood Services, the Philippine Red Cross collects the following data: gender and age group of blood donors, number of blood donors, collected and dispensed units, average daily collection of blood units, target and actual collection of blood units, number of indigent blood units and unserved units [2].

B. Blood Donation

Blood donation is an act where a donor gives his or her blood components out with the purpose of those blood components replacing the blood components needed by a patient. This may come in two forms, either a standard blood donation where the donor gives out an entire blood unit that is composed of all blood components, or an apheresis blood donation where only specific blood components are taken from a blood donor [2].

Becoming a blood donor requires passing various examinations. Medically speaking, a potential blood donor must not have sexually transmitted diseases such as Syphilis, Hepatitis B, C, and E, and HIV. The blood donor must also not have a history of Malaria, T-cruzi, West Nile Virus, or Cytomegalovirus [18]. The list of infections and diseases that are checked are continually updated in order to make sure one is able to provide relatively healthy blood. Other minor factors that are checked are the medication the donor is currently taking, the presence of tattoos and body piercings the donor has, the latest infection the donor had, and other case to case basis situations such as state of pregnancy and blood pressure of the donor.

In the Philippine setting, blood donation is handled mostly by the Philippine Red Cross aside from hospitals. The process is as follows, a potential donor goes to an outreach program by the Philippine Red Cross or to any chapter of it. They then answer a donor history questionnaire and an interview that would determine if they can donate blood or not. Upon knowing that the potential blood donor passes the examination, the Philippine Red Cross takes a sample of the blood of the donor. The Philippine Red Cross next determines the blood type of the donor. After which, if the donor does not have a special blood type, they would conduct a standard blood donation procedure. Otherwise, if the donor has a

RH negative blood type, the Philippine Red Cross would ask for the contact details of the donor and would ask that the donor keep in touch in cases where the donor is needed [2].

Given by the Department of Health of the Philippines, all organizations handling blood services, including the Philippine Red Cross, has a frequently answered questions for those who would want to donate or request blood. For blood donors, things to keep in mind are the following statements. First, blood donors are not paid for their services. Second, it is mandatory to always answer the donor history questionnaire to ensure the quality of the blood component. Third, the donor can be rest assured that they will not contract any disease nor will they be harmed because of their blood donation because instead of being harmful, blood donation actually is beneficial for the blood donor as this allows the donor to renew their blood therefore allowing the donor's blood circulation and body resistance to be better than before. Fourth, the donor does not need to worry about working immediately after donating as only rigorous work should be avoided for a few hours after donating. Fifth, the donor can only donate after three months for standard blood donation and two weeks for apheresis blood donations. Fifth, the donor does not need to worry about having not enough blood for the donation since the body is capable of storing six liters of blood while the blood taken for the donation is only less than ten percent of this. Sixth, the donor does not need to worry about the process to be very time consuming as the entire process which contains the blood transfusion and the post treatment lasts only for an hour. Seventh, donors should take heed to eat and sleep properly before donating. Eighth, a donor may not be allowed to donate blood temporarily if the donor is pregnant, in a fever, drank alcohol lately, has body piercings and tattoos, and underwent surgery within the past month while the donor will not be allowed to donate permanently if the donor has a history of cancer, cardiac disease, severe lung disease, hepatitis B or C, sexually transmitted diseases, high risk occupations, and unexplainable weight loss. Finally, the blood taken during blood donation will be separated into various blood components which will be used in different types of operation [1].

C. Blood Request

Blood request, as opposed to blood donation, is where someone requests for blood components for a patient who is suffering from blood loss. They need to request from either hospitals where the patient is staying at, blood banks stationed across their country, or even look for willing blood donors to provide the components requested for the operation. Being unable to do so will normally make the patient unable to take their respective surgeries.

In the Philippine setting, one of the biggest providers of blood units is the Philippine Red Cross. People who would want to request blood components must first let a nurse or the doctor in the hospital fill up a blood request form containing the necessary information regarding the request which normally contains the name, needed blood components, and reason for the request. They would then go to a Red Cross chapter personally and hand this over to a volunteer. The request will be assessed if it could be supplied by the Philippine Red Cross. If deemed suppliable, the requester would be asked for a processing fee used to pay for the handling and preservation of the blood components requested which goes from at least 1000 to 15000 Philippine Peso depending on the components requested. Otherwise, explanation is given to the requester why Red Cross is unable to supply the blood request [2].

D. Information System

Information systems manage and processes information. These information may come in the form of data of a user, product, or service in a given situation. The processes, specifically are the following: handling the capture, transmission, storage, retrieval, manipulating, and display of the information. Examples of which are hospital information systems that processes data of its patients, doctors, employees, or even sickness profiles.

In the context of the project, the information system to be produced holds data that speaks of news, articles and announcements from the Philippine Red Cross, status of data related to the Blood Services gathered by Philippine Red Cross as stated earlier, and do-

nation history data gathered when used with the partner mobile application.

E. Mobile Application

Mobile applications are another specific case of information system that uses an application to display data or information in a smartphone. Currently, the top operating systems of smartphones are either in Android developed by Google, iOS by Apple, or Windows by Microsoft Corporation. Mobile applications are currently being created as an alternative to some desktop applications due to smartphones being convenient to bring around at all places. Furthermore, mobile applications are preferred over web applications for general public use due to mobile applications being capable of working without the need of being connected to the Internet.

F. Firebase Application

Firebase applications are web or mobile applications that use the Google's Firebase API. The Firebase API allows easier deployment of the application as Google also provides various tools such as a real-time database, hosting services, cloud messaging, and analytics for the application. The use of the Firebase API allows easier maintenance as well as Google also provides tools for back-end updating so that new code will not crash the main servers [19].

G. Global Positioning System

Global Positioning System is the use of the system created by the United States in order to provide real-time location services to a user. In a nutshell, a satellite created by the United States broadcasts radio signals that would pinpoint the location of the user, this signal travels through space and is finally received by a GPS capable device in order to show current location and time with respect to the whole world [20].

IV. Design and Implementation

A. Use Case Diagram

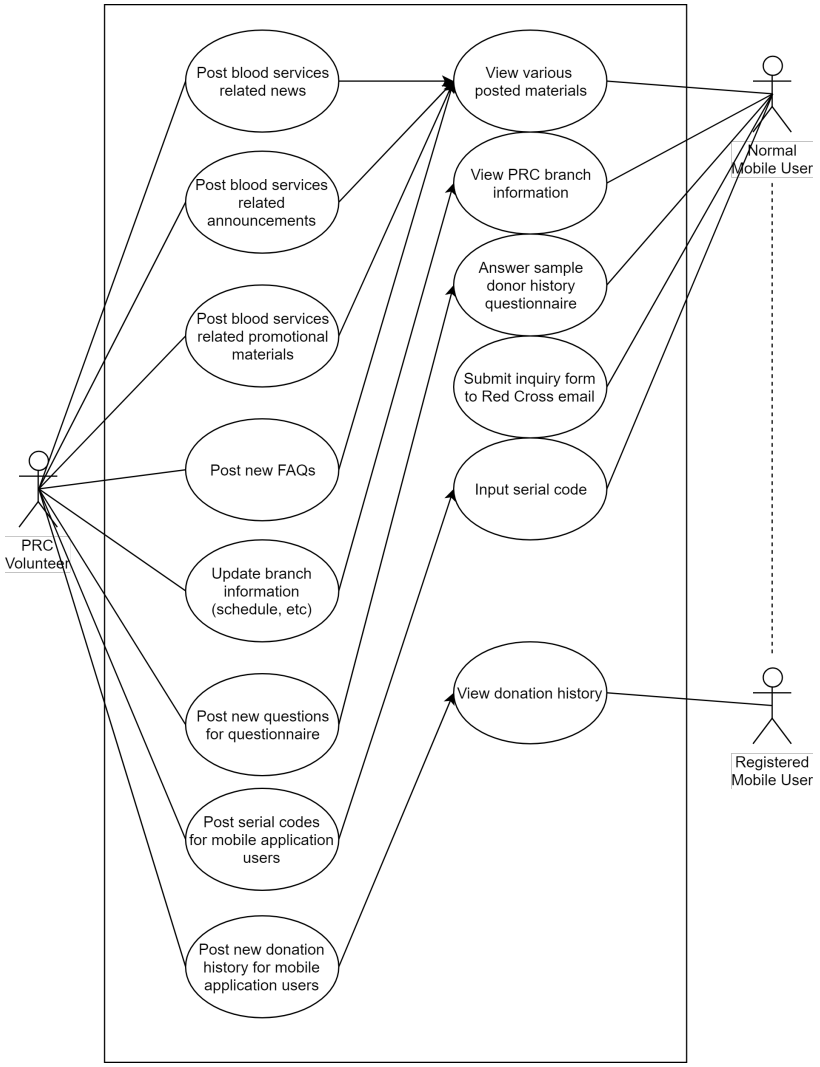

Figure 1: Use Case Diagram, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

Using the web application, Philippine Red Cross employees can simply add, delete, or remove various materials from the database. This includes the creation, editing, and removal of news, announcements, promotional materials, frequently asked questions, questionnaire questions, serial codes, and donation histories given a serial code.

Using the mobile application, the user may also view the materials posted by the Philip-

pine Red Cross content supervisors. They may also view specific Philippine Red Cross branch information such as their schedule and location. Furthermore, users can also answer a sample donor history questionnaire and receive immediate feedback from it. Users can then also ask Philippine Red Cross questions using the inquiry form in the mobile application. Lastly, users can input serial codes which unlock other features for the mobile application. When registered, the mobile application user can now input latest blood donation, check their donation history, and even check if they are eligible to donate again with respect to their latest donation at the time of checking their donation history.

B. Flowchart: Web Application

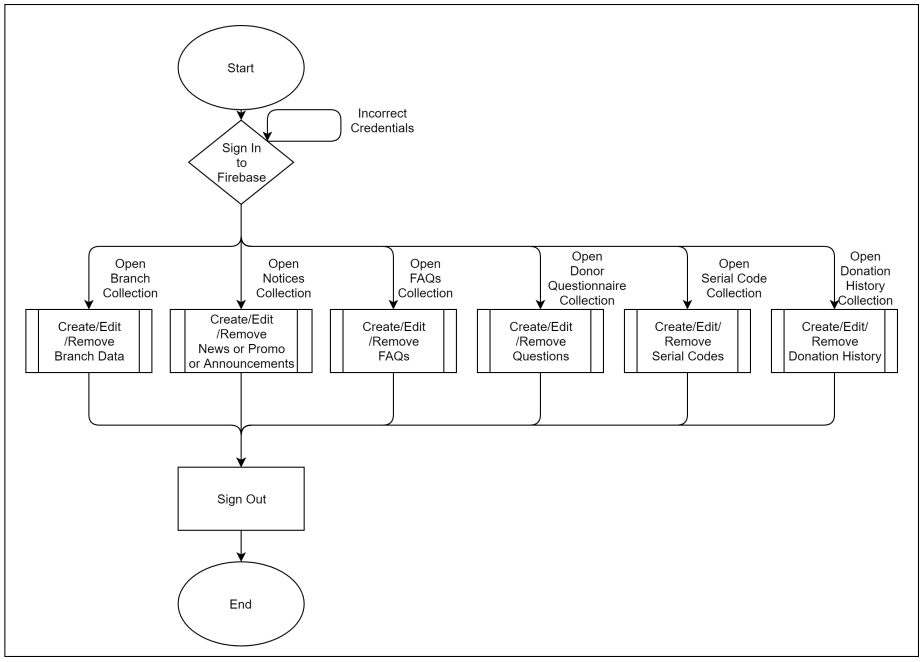

Figure 2: Web Application Flowchart, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

The Philippine Red Cross employee must first sign in to Firebase using their Google powered email then he or she must open console.firebase.com, where the database is stored. From here on, the Philippine Red Cross employee can view the collections for the respective materials for the mobile application. Viewing one collection allows the employee to either create a new document under that collection, edit existing documents, or delete documents.

All changes will immediately be seen in the mobile application due to the nature of Firebase.

C. Flowchart: Mobile Application

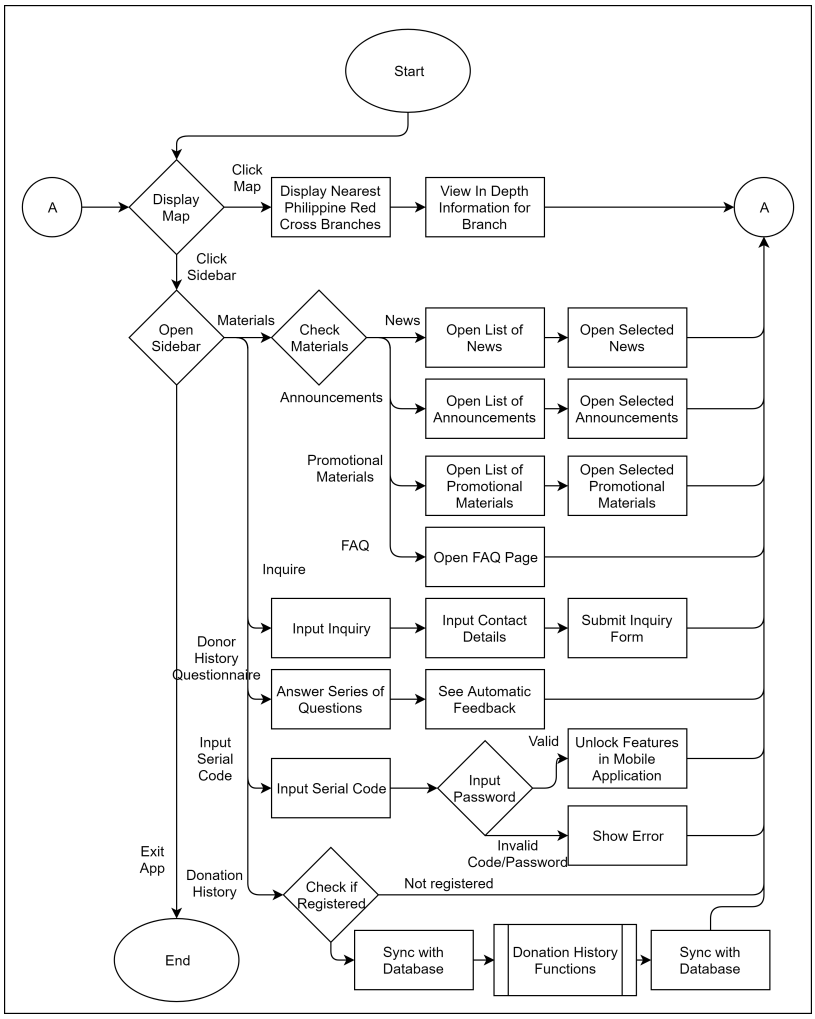

Figure 3: Flowchart: Mobile Application (Non-registered Users), Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

Opening the mobile application, the first thing users will see is the map centered at the nearest Philippine Red Cross chapter to their location. If the user's location cannot be determined, the map will be centered at the Philippine Red Cross headquarters. In the said screen, users can either click on the map to focus on the Philippine Red Cross branch and show its details or open the sidebar for other functions of the mobile application. The sidebar contains checking the materials, sending an inquiry form, answering a sample donor

history questionnaire, inputting a serial code, accessing donation history functions if the user is registered as a donor, and exiting the application. In checking materials, whatever type of material it may be, the application will be displaying via a list. In submitting an inquiry form, the user must input both their question and contact details. In answering the donor history questionnaire, the user will first answer a series of questions and receive feedback according to the user's inputs. In submitting a serial code, other mobile features will only be unlocked after submitting a valid serial code and password.

Once registered, mobile application users can have access to donation history related functions. The mobile application checks the user's latest donation from the database. If the donor can donate given the details of his or her previous donation, the mobile application shows an indicator for this. Donation history of the user is updated in realtime as long as the user is connected to the internet.

D. Entity Relationship Diagram

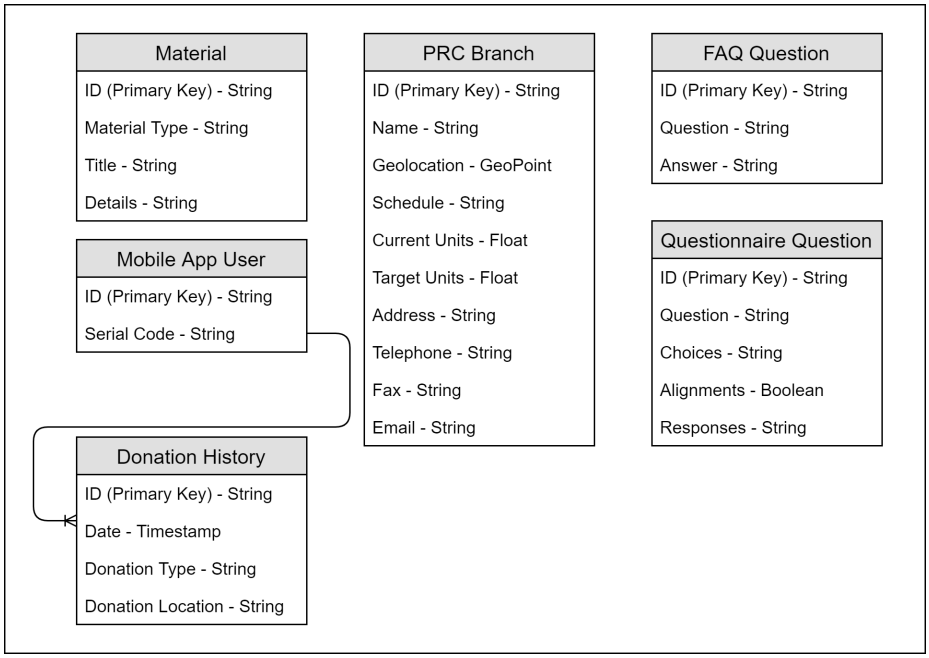

Figure 4: Diagram: Entity Relationship Diagram, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

The database is comprised of six tables. First is for the materials, where their type could be news, announcement, or promotional material, with their corresponding title, details, and attached image if there is any. The second is composed of the users where they have a unique serial code. Each user will have their own donation history, the third table, where each donation history contains the date for the donation, the donation type, and the location for the blood donation. The fourth table is for the information regarding the branches of the Philippine Red Cross, with each info being there in order for them to be contacted more easily. Fifth table is composed of the questions frequently asked, managed by employees of the Philippine Red Cross. Last but not least, there is the questionnaire questions table. This includes the question, choices, their respective alignments and responses.

V. Results

The following are screenshots from the created mobile application and web tool.

Figure 5 shows the very first thing the user will see when running the mobile application. This is where the mobile app prepares the assets and resources to be used later. Furthermore, here is where the app requests permissions from the user to allow the mobile application to pinpoint the location of the user.

Figure 5: Screenshot: Final Splash Activity Asking for Permission, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

Figure 6 displays the default android permission dialog where the mobile asks for permission from the user to allow the application in finding the user using the GPS of the phone. This is due to the mobile application later on using the Maps and Directions API from Google which requires the said permission. Failure to securing this permission would result to the user being unable to access one of the core functions of the application because of the mobile application being unable to determine the location of the user. Since this permission is really important in order for one of the core functions to function, denying to give the permission would remind the user of the consequences of not giving due permissions. If the user still does not want to give permission by then, the mobile application disable getting the location of the user throughout the application. However, this does not mean that other functions will also be locked out. Only locating the location of the user and giving directions to a Philippine Red Cross Chapter will be disabled. The succeeding functions are still working.

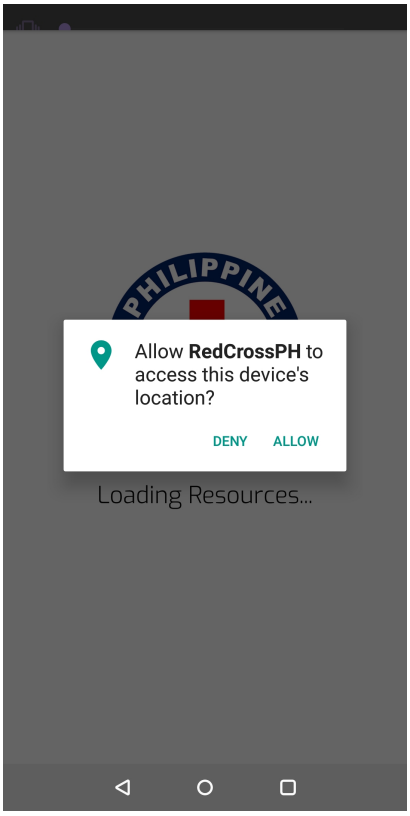

Figure 6: Screenshot: Final Splash Activity Asking For Permission, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

Figure 7 shows the user entering the main activity of the mobile application. Assuming that the user did not give their permission to get their location, what they would see is a blank map with only the markers for the Philippine Red Cross chapters. The mobile application is unable to check for the location of the user. Hence, is unable to check for nearby chapters. Without the permission being given, the users can only find the Philippine Red Cross chapters by manually looking for them in the map or by filtering them through the use of the search bar.

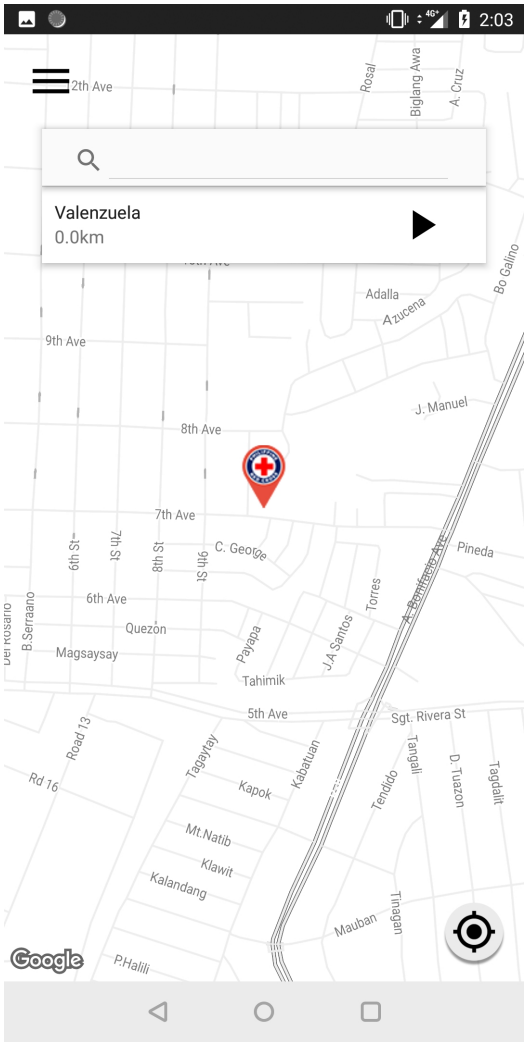

Figure 7: Screenshot: Final Main Activity Without Permission, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

If the user gave their permission for the mobile application to pinpoint their location via the GPS, the resulting screenshot is as seen in Figure 8. Using their location, the mobile application calculates the distance of the user to the different Philippine Red Cross chapters and automatically looks for the nearest Philippine Red Cross chapter and shows the direction from the user to the Philippine Red Cross chapter. This still applies when the user filters out a wanted chapter to be visited. Using the search bar, the user may filter the Philippine Red Cross they want to visit and the application will automatically show the direction from the user to that filtered Philippine Red Cross chapter.

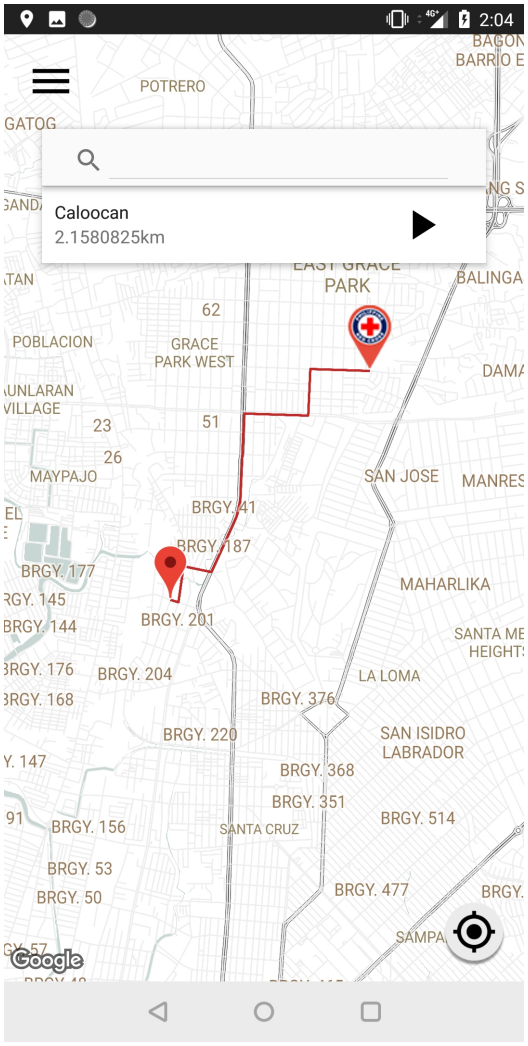

Figure 8: Screenshot: Final Main Activity With Permission, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

In Figure 9, user is in the main activity. This popup shows up after clicking the go button beside the highlighted location from the previous two screenshots. Through this, the user is able to see specific information regarding the highlighted Philippine Red Cross chapter. This includes the graphical representation of the status of the blood supply of the chapter, its name, capabilities, and more. The details shown here is tied with the database such that given changes from the database, the mobile application still updates the details in realtime, assuming that the user is connected to the Internet.

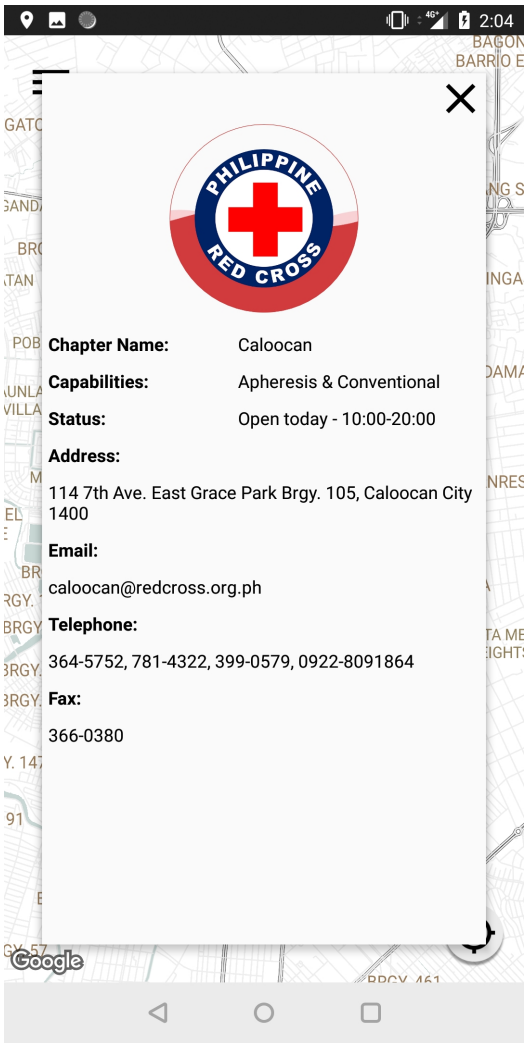

Figure 9: Screenshot: Final Main Activity Showing Specific Information, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

Figure 10 shows the main activity of the mobile application. This time, showing the search or the filter function where the user can type in the search bar and the mobile application filters which closest Philippine Red Cross chapter contains said input. When no chapter is found using the filter, there will be no chapter highlighted and no direction will be shown.

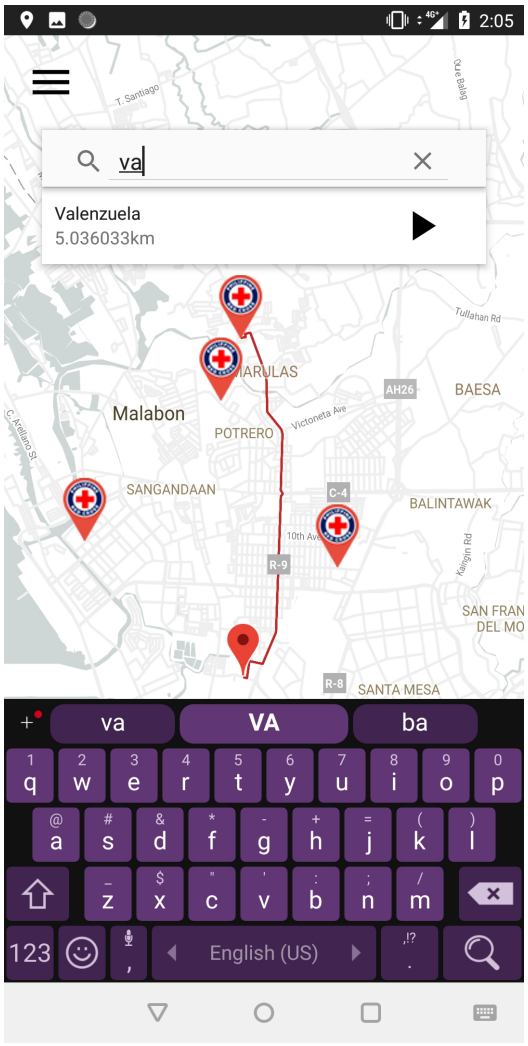

Figure 10: Screenshot: Final Main Activity Using Filter, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

Figure 11 shows the user seeing the navigation bar used to navigate through the other activities. The navigation drawer holds the buttons in order to access the other functions of the app that will be showed later.

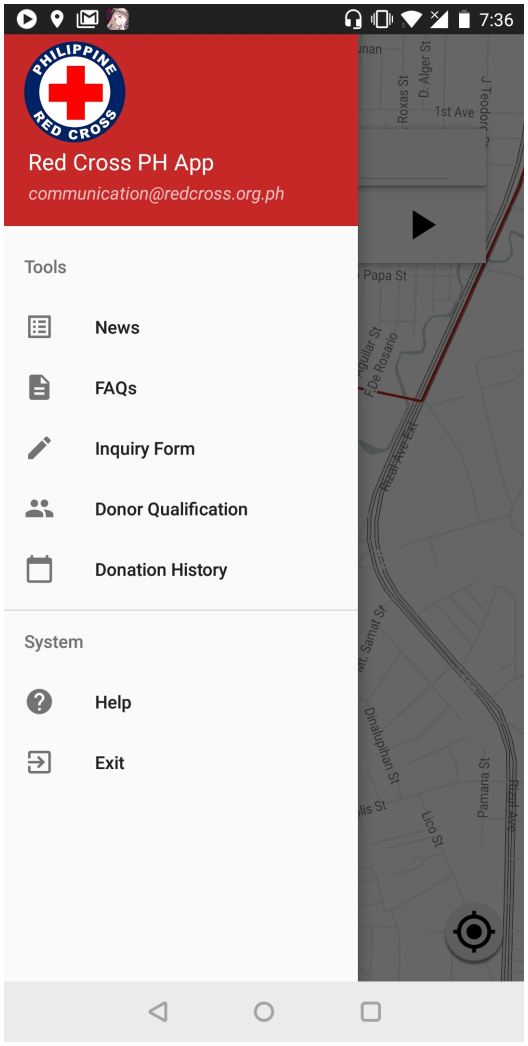

Figure 11: Screenshot: Final Navigation Bar, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

Figure 12 shows the news activity of the mobile application. In this screenshot, the mobile application displays all the news, announcements, and promotional materials the Philippine Red Cross inputted into the database. The mobile application properly distinguishes the three from one another by showing respective tags to each type. Furthermore, these materials are displayed according to the date they were added into the database with the user first seeing the latest material from the Philippine Red Cross. The user can also choose to use the search bar in order to filter out the news that the user is able to see.

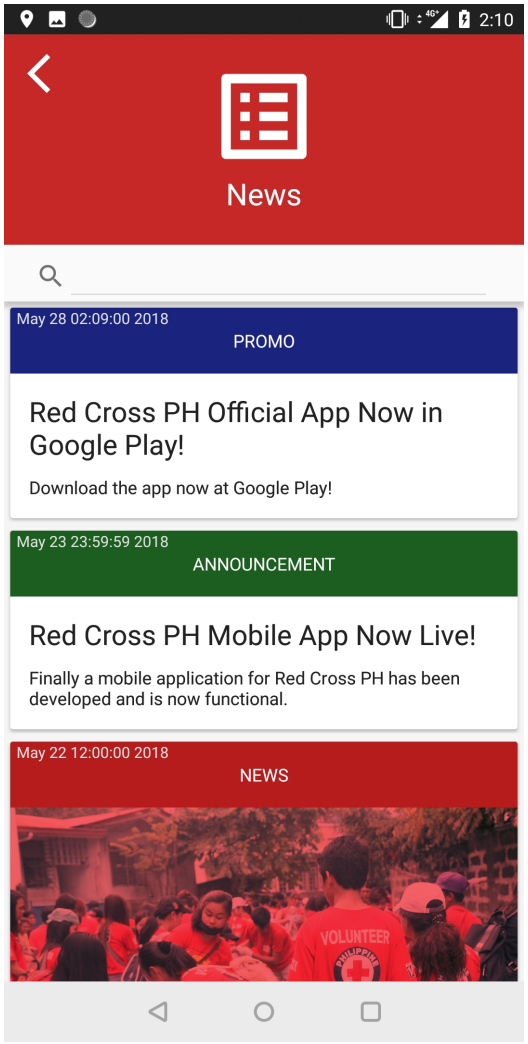

Figure 12: Screenshot: Final News Activity, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

The frequently asked questions and their respective answers directed to the Philippine Red Cross is shown in Figure 13. Like the news activity, this section lists the entire set of question and answers from the database but can be filtered by the user using the search bar. Questions and answers seen here is also updatable using the web tool and updates could be seen in realtime like the previous activities.

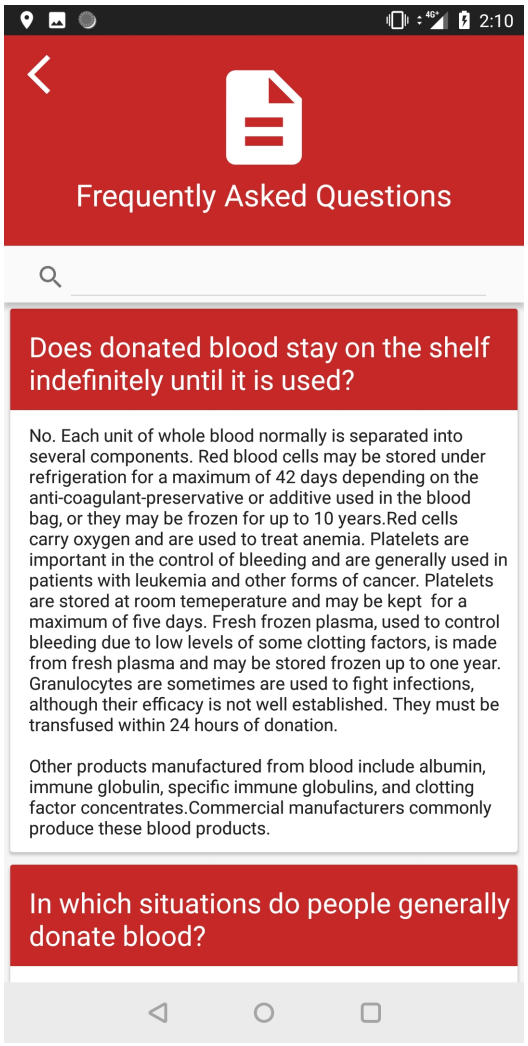

Figure 13: Screenshot: Final FAQs Activity, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

The inquiry activity where the user can choose to input their email or their contact number and their inquiry is shown in Figure 14. They can only submit this form if and only if the user gives either an email or a contact number and an inquiry. When the user finally submits this form, the user can see from the submit button whether that inquiry is still being sent or has finally been sent successfully.

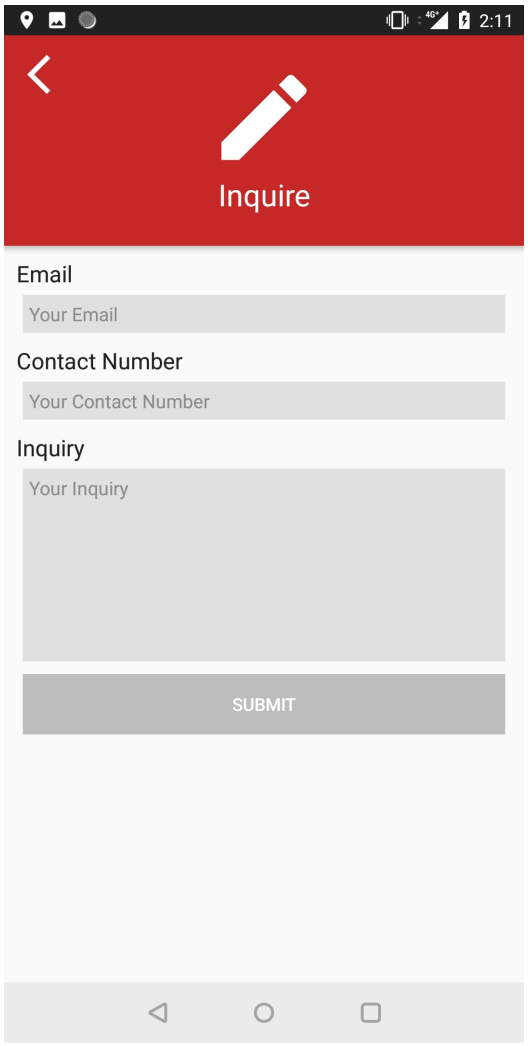

Figure 14: Screenshot: Final Inquiry Activity, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

The sample questions from the donor history questionnaire from the Philippine Red Cross where the user can answer questions and get immediate and overall feedbacks from it is highlighted in Figure 15. Like the FAQs and the various materials from the Philippine Red Cross, these questions again can be filtered using the search bar.

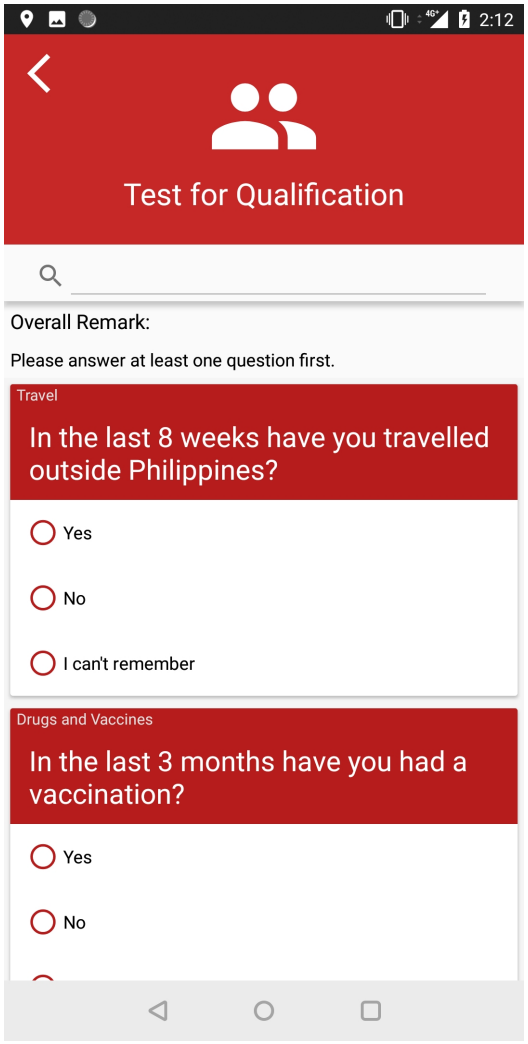

Figure 15: Screenshot: Final Questionnaire Activity, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

This screenshot shows an example where the user has given a positive answer given a question (see Figure 16). Both the overall remark and the specific remark show that the answer the user gave is positive and does not disqualify the user from becoming a blood donor.

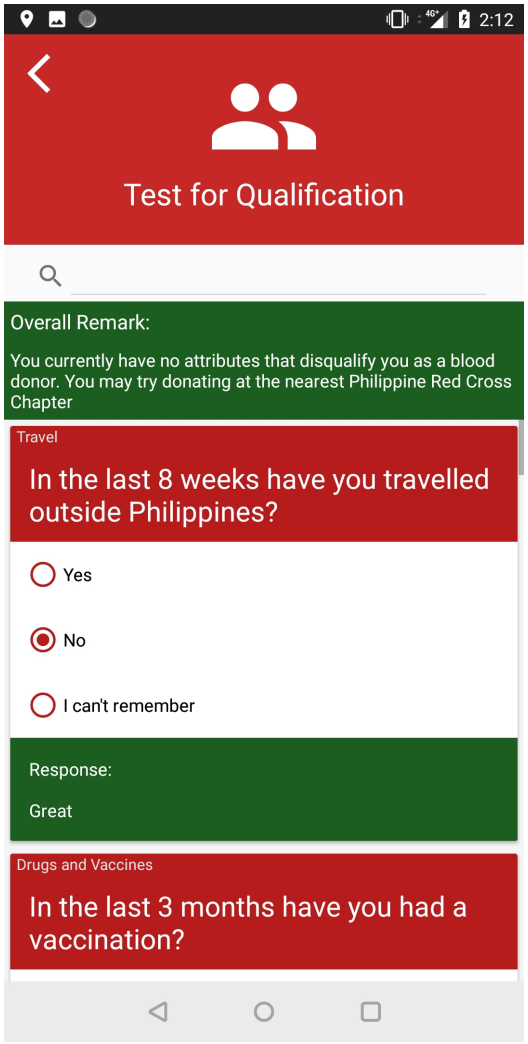

Figure 16: Screenshot: Final Questionnaire Activity with Positive Response, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

This screenshot shows an example where the user has given a negative answer given a question (see Figure 17). Both the overall remark and the specific remark show that the answer the user gave is negative and the overall remark shows that the user must first resolve the specific negative remarks before continuing.

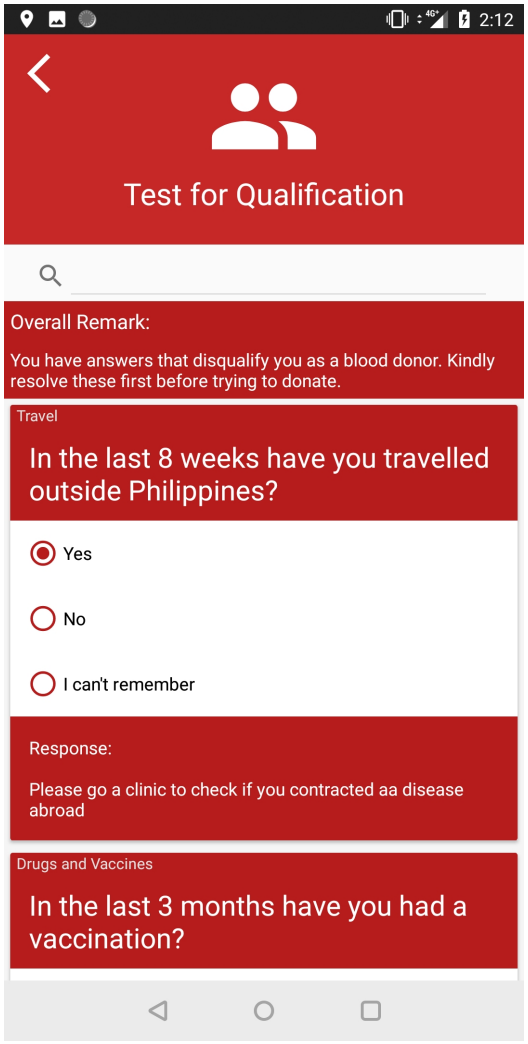

Figure 17: Screenshot: Final Questionnaire Activity with Negative Response, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

The first use of the donation history function of the mobile application where the mobile application asks for a serial code that should be given to the user by the Philippine Red Cross during their first donation is noted in Figure 18. Being unable to provide a serial code would not allow the user from proceeding on using the donation history function of the app.

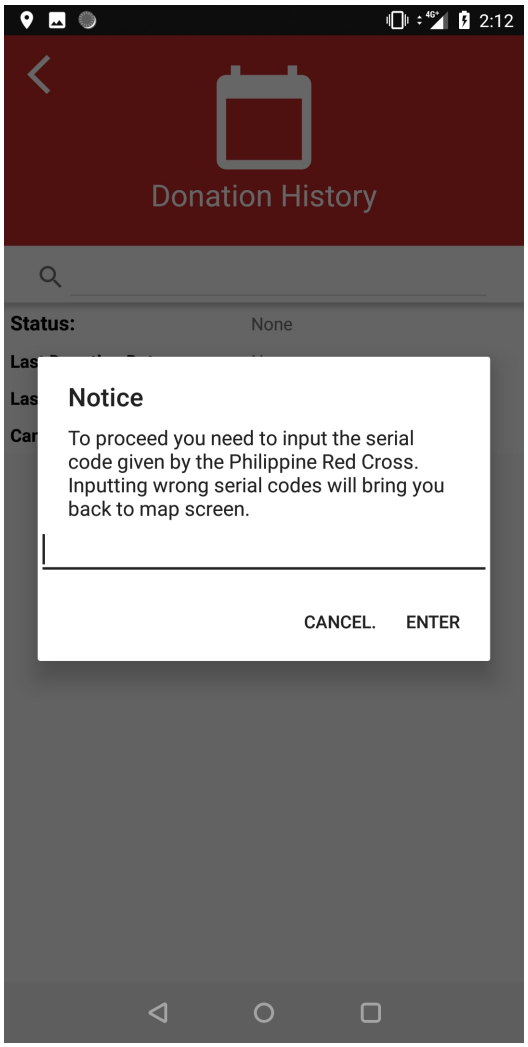

Figure 18: Screenshot: Final Donation History Activity, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

The next screenshot (see Figure 19) shows the user inputting a serial code but is not part of the database. The mobile application shows that an incorrect serial code is used and asks the user to try again next time.

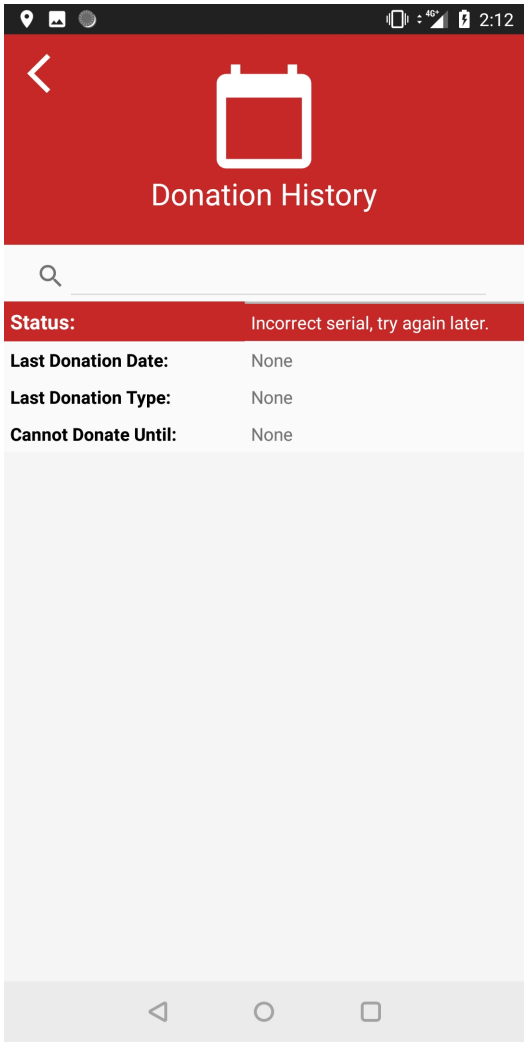

Figure 19: Screenshot: Final Donation History Activity Incorrect Serial Code, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

The screenshot (see Figure 20) shows that the user has inputted a correct serial code. In this case, the mobile app fetches the donation history for the respective serial code. Listing all donations done by the user. The mobile application takes a step further by also checking the current status of the user, whether the user is able to donate blood at the time they checked this activity. If the user is still not eligible, then the application shows when they can donate again, using the standard of the Philippine Red Cross as calculation for their next blood donation.

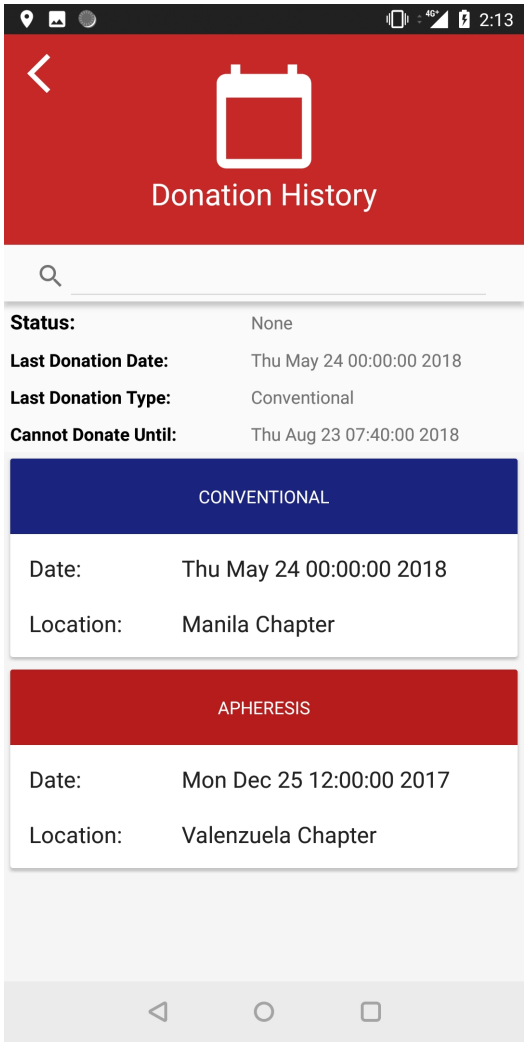

Figure 20: Screenshot: Final Donation History Activity Correct Serial Code, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

This screenshot shows the collections in the database. The Philippine Red Cross staff in charge of updating these can choose a specific collection, then choose to edit existing documents under that collection, or remove them from the list of saved documents, or they can also create new documents for that collection. In any case, all changes done to the collection will be effectively seen in the mobile application when the mobile app users connect to the Internet (see Figure 21).

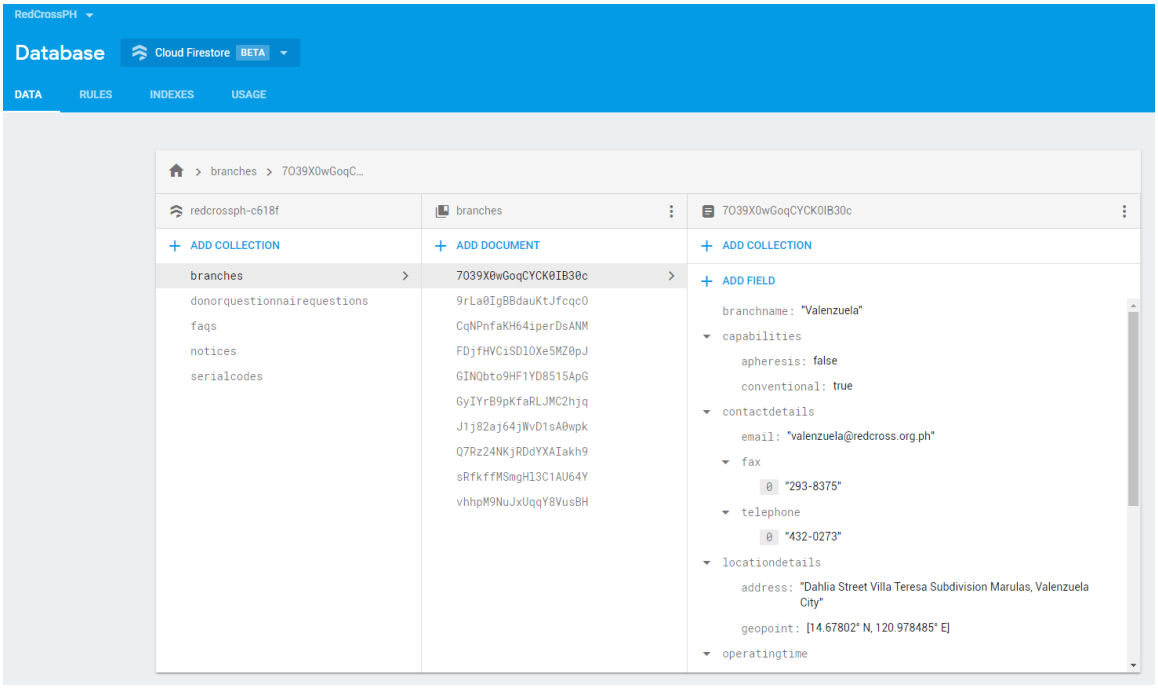

Figure 21: Screenshot: Google Firebase, Disseminating Information Regarding Blood Donation In the Philippines Using Mobile Application

VI. Discussions

The mobile application, Red Cross PH Official App, currently is able to provide ample information with regards to the blood services in the Philippines using the data that the Philippine Red Cross provides. First, the mobile application is now able to direct users where to go by locating the different chapters of the Philippine Red Cross and even show relevant information regarding the chosen chapter. Next, the mobile application is also updated with the news, announcements, and promotional materials provided by the Philippine Red Cross so that users will be notified if the Philippine Red Cross is about to go to an event or such that could also be useful for the users. Third, the mobile application is able to give answers to frequently asked questions using the FAQ activity. Fourth, the mobile application also allows users to contact the Philippine Red Cross using an inquiry form. Next, the mobile application is also able to provide feedback to users using the sample questions from the donor history questionnaire. Lastly, the mobile application is able to show the donation history of a blood donor and is able to show the next latest date when the donor can donate again.

Looking at all perspectives, the strength of the mobile application created could be seen from the overall information it provides. Depending on the variety of the information the Philippine Red Cross provides, the application can be used by potential blood donors so that they may self educate themselves using the information from the mobile application. Existing blood donors can now also be empowered to monitor the frequency of their donations using their respective donation histories. Even Philippine Red Cross volunteers can choose to use the mobile application when teaching others of blood donation. Overall, the mobile application is not restricted to one type of citizen but instead, the mobile application can be used by anyone.

Currently, the data used displayed in the mobile application is not from the Philippine Red Cross but is compiled from various credible locations. The news, announcements, and promotional materials displayed are just all dummy data. However, the frequently asked questions are all taken from the webpage provided by the Department of Health in the

Philippines [21] whereas the questions used are from the Canadian Blood Services website [22].

Specifically, the usage of mobile phone as the platform for this application is explained by the statistics of the total number of people using smartphones in the year 2018 as provided by statista.com. According to statista, the total number of smartphone users currently number 2.53 billion [23]. Furthermore, mobile phones are more portable than laptops and more so than desktop computers. Hence it is wiser to implement this type of application in a mobile phone in order to maximize the number of people who would use it.

Another thing to note is the location, GPS, maps, and directions functions of the mobile application. This set of functions are extremely helpful specially when directing someone where to go. In this case, these functions point out and highlight the different Philippine Red Cross Chapters. This is helpful as according to studies made by the Philippine Red Cross [2], one of the key factors why people do not donate is because they do not know where they can donate. This is solved by these functions.

The next functions, whether in showing the latest news, or the frequently asked questions, and even to answering the sample donor history questionnaire is also useful for the masses. From interviewing the Donor Recruitment Head of the Philippine Red Cross. One big problem as to why there is a low count of blood donors in the Philippines is due to the misconceptions with regards to blood donation [2]. Many people think that blood donation is harmful. Most people think that after donating you cannot go to work for a few days. A lot of people do not know if they are qualified to become blood donors. These are rooted to not knowing the nature of blood donation is. Hence, using these functions, the mobile application will provide the information needed so that one person may be educated in the nature of blood donation.

Lastly, with regards to the donation history function. This will help existing blood donors in being able to donate blood as soon as they are reminded that they can donate again. Due to conventional blood donations requiring donors to have an interval of at least

two months before donating again, most of the time, blood donors forget if two months have already passed. Being afraid of repercussions of donating without properly replenishing lost blood nutrients, blood donors wait a long time before donating again. This is solved by storing the donation history of the donor, so that the donor may check from time to time if he or she is able to donate again.

However, these strengths become a weakness assuming that the information Philippine Red Cross is not enough. Currently, all information seen in the app is solely limited to the information released by the Philippine Red Cross. Only Philippine Red Cross chapters are marked in the map, only materials from the Philippine Red Cross is displayed, only frequently asked questions compiled by the Philippine Red Cross is seen, and only donors that donate to the Philippine Red Cross see their donation history. If the information provided by the Philippine Red Cross is insufficient, or if the person handling the release of information is not active. The mobile application will not be useful for anyone.

As for the mobile application being compared to other countries' blood services related mobile applications. Looking at the mobile application developed in Bangladesh [7], Bangladesh's mobile application is not only for potential blood donors, but also for blood requesters. Even though the idea of requesting blood and looking for potential blood donors through the app is a good idea, the Philippine government prohibit such a behavior hence this function cannot be implemented in our system. This is because unlike other countries that depend on incentives for motivating blood donors [11], the Philippine government wants to create a fully voluntary system for blood donation.

VII. Conclusions

Overall, the mobile application that disseminates information regarding blood donation in the Philippines, at the moment is helpful in terms of spreading information. Normal users can use the mobile application for self education so that illiteracy towards blood donation can be solved. Blood donors can use the mobile application to have self monitoring so that they may maximize their frequency of donating blood. Philippine Red Cross volunteers can use the mobile app in teaching others in their activities so that they do not need to print brochures like before. The mobile application is very information packed and is beneficial to anyone wanting to engage in the blood supply chain of the country.

The effects of using this mobile app is still unclear as it is still not released to the public. But assuming that the information from the Philippine Red Cross is enough and the distribution of the application is properly done, the problem of having not much blood donors in the country because of being ignorant on the blood services in the country will be solved, consequently raising the numbers of blood donors in the country and ultimately allowing the Philippines to have enough blood donors as recommended by the World Health Organization.

VIII. Recommendations

The mobile application can still be improved in various ways. For one, instead of only having the Philippine Red Cross as the supplier of information for the mobile application, one can also integrate other parties in the blood supply chain of the Philippines in the mobile app as well. This goes in line with the goal of the Republic Act of 7719 [4] to have a centralized network for the blood supply. Consequently, in order to fulfill this without one party interfering with another party in their operations of providing data for the mobile app. A web application having its own authorization system depending on from which organization the informant comes from, may have their own collections within Firebase and can only edit the assigned collections to them. With this, Philippine Red Cross may not try and tamper on data inputted by other organizations, and vice versa. Another idea is to allow live conversations between a prospective blood donor and a Philippine Red Cross staff, in order to provide more interactive communication. Lastly, a notification system may also be implemented such that donors may be notified automatically if they are able to donate or if the blood banks are low in blood unit count. These are some ways that could improve this system and overall efficiency in terms of providing information to the public.

IX. Bibliography

[1] "Voluntary blood donation." http://www.doh.gov.ph/book/export/html/1416.

[2] "Interview with philippine red cross donor recruitment head hazel sentones," 2018.

[3] "How to donate." https://www.redcross.org.ph/get-involved/give-blood/how-to-donate.

[4] "Republic act no. 7719." http://www.officialgazette.gov.ph/1994/05/05/republic-act-no-7719/.

[5] "How blood is used." https://www.blood.co.uk/why-give-blood/how-blood-is-used/.

[6] S. Ouhbi, J. L. Fernndez-Alemn, A. Toval, A. Idri, and J. R. Pozo, "Free blood donation mobile applications," *Journal of Medical Systems*, vol. 39, p. 52, Mar 2015.

[7] A. H. M. S. Islam, N. Ahmed, K. Hasan, and M. Jubayer, "mhealth: Blood donation service in bangladesh," *Informatics, Electronics and Vision*, 2013.

[8] A. M. Mostafa, A. E. Youssef, and G. Alshorbagy, "A framework for a smart social blood donation system based on mobile cloud computing," *CoRR*, vol. abs/1412.7276, 2014.

[9] M. Fahim, H. I. Cebe, J. Rasheed, and F. Kiani, "mhealth: Blood donation application using android smartphone," *Digital Information and Communication Technology and its Applications*, 2016.

[10] L. Sardi, A. Idri, and J. L. Fernández-Alemán, "Gamified mobile blood donation applications," in *Bioinformatics and Biomedical Engineering* (I. Rojas and F. Ortuño, eds.), (Cham), pp. 165–176, Springer International Publishing, 2017.

[11] K. Chell, T. E. Davison, B. Masser, and K. Jensen, "A systematic review of incentives in blood donation," *Transfusion*, vol. 58, no. 1, pp. 242–254, 2018.

[12] L. M. Williamson and D. V. Devine, "Challenges in the management of the blood supply," *The Lancet*, 2013.

[13] M. A. Setiawan and H. H. Putra, "Bloodhub: A context aware system to increase voluntary blood donors' participation," *TICST*, 2015.

[14] H. Y. Lam, V. Y. Belizario, N. R. Juban, M. M. Alejandria, N. C. Carandang, E. Arcellana-Nuqui, M. A. Mirasol, C. P. Cordero, O. T. Sison, and L. Tayao, "Current practices of blood service facilities in the philippines," *The UPManila Journal*, 2013.

[15] G. Greenleaf and W. il Park, "South korea's innovations in data privacy principles: Asian comparisons," *Computer Law and Security Review*, 2014.

[16] "About the philippine red cross." https://www.redcross.org.ph/about-the-philippine-red-cross.

[17] "Fundamental principles." https://www.redcross.org.ph/fundamental-principles.

[18] "Tests we carry out." https://www.blood.co.uk/the-donation-process/further-information/tests-we-carry-out/.

[19] "Firebase." https://firebase.google.com/.

[20] "How gps works." https://www.gps.gov/multimedia/poster/.

[21] "Voluntary blood donation." https://www.doh.gov.ph/book/export/html/1416.

[22] "What is the donor questionnaire." https://blood.ca/en/blood/what-is-the-donor-questionnaire.

[23] "Number of smartphone users worldwide from 2014-2020 (in billions)." https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/.

X. Appendix

..1 FAQ.java

```
package upmanila.bengeminuy.redcrossph;

public class FAQ {

    private String id;
    private String question;
    private String answer;

    public FAQ(){
        id = "";
        question = "";
        answer = "";
    }

    public void setId(String id) {
        this.id = id;
    }

    public void setAnswer(String answer) {
        this.answer = answer;
    }

    public void setQuestion(String question) {
        this.question = question;
    }

    public String getId() {
        return id;
    }

    public String getAnswer() {
        return answer;
    }

    public String getQuestion() {
        return question;
    }

    @Override
    public boolean equals(Object obj) {
        FAQ newFAQ = (FAQ) obj;
        return this.id.compareTo(newFAQ.getId()) == 0;
    }
}
```

..2 FAQAdapter.java

```
package upmanila.bengeminuy.redcrossph;

import android.content.Context;
import android.support.annotation.NonNull;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

import java.util.List;

public class FAQAdapter extends RecyclerView.Adapter<FAQAdapter.ViewHolder> {

    private List<FAQ> faqsModel;
    private Context context;

    public FAQAdapter(List<FAQ> faqs, Context context){
        this.faqsModel = faqs;
        this.context = context;
    }

    @NonNull
    @Override
    public FAQAdapter.ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        View v = LayoutInflater.from(parent.getContext()).inflate(R.layout.faq_card, parent, false
            );
        return new FAQAdapter.ViewHolder(v);
    }

    @Override
    public void onBindViewHolder(@NonNull FAQAdapter.ViewHolder holder, int position) {
        FAQ faq = faqsModel.get(position);

        holder.questionText.setText(faq.getQuestion().replace("\\n", "\n"));
        holder.answerText.setText(faq.getAnswer().replace("\\n", "\n"));
```

```
    }

    @Override
    public int getItemCount() {
        return faqsModel.size();
    }

    @Override
    public int getItemViewType(int position) {
        return position;
    }

    public class ViewHolder extends RecyclerView.ViewHolder{

        public TextView questionText;
        public TextView answerText;

        public ViewHolder(View itemView) {
            super(itemView);

            questionText = itemView.findViewById(R.id.faqCardQuestion);
            answerText = itemView.findViewById(R.id.faqCardAnswer);

        }
    }

}
```

..3 FAQsActivity.java

```
package upmanila.bengeminuy.redcrossph;

import android.content.Context;
import android.support.annotation.Nullable;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.util.Log;
import android.view.View;
import android.widget.SearchView;

import com.google.firebase.FirebaseApp;
import com.google.firebase.firestore.DocumentChange;
import com.google.firebase.firestore.EventListener;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.FirebaseFirestoreException;
import com.google.firebase.firestore.QuerySnapshot;

import java.util.ArrayList;

public class FAQsActivity extends AppCompatActivity {

    private Context context;
    private FirebaseFirestore database;
    private ArrayList<FAQ> databaseFaqs, filteredFaqs;
    private RecyclerView faqRecyclerView;
    private RecyclerView.Adapter faqRecyclerViewAdapter;
    private SearchView faqSearchView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_faqs);

        context = getApplicationContext();
        faqRecyclerView = findViewById(R.id.parentFAQRecyclerView);
        faqRecyclerView.setLayoutManager(new LinearLayoutManager(this));
        faqSearchView = findViewById(R.id.faqSearchView);
        faqSearchView.setIconifiedByDefault(false);

        FirebaseApp.initializeApp(context);
        database = FirebaseFirestore.getInstance();

        databaseFaqs = new ArrayList<>();
        filteredFaqs = new ArrayList<>();

        faqRecyclerViewAdapter = new FAQAdapter(filteredFaqs,this);
        faqRecyclerView.setAdapter(faqRecyclerViewAdapter);

        faqSearchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
            @Override
            public boolean onQueryTextSubmit(String query) {
                filterFilteredFAQS();
                return true;
            }

            @Override
            public boolean onQueryTextChange(String newText) {
                filterFilteredFAQS();
```

```
                return true;
            }
        });
    }

    @Override
    protected void onStart() {
        super.onStart();

        faqSearchView.setIconified(false);
        faqSearchView.clearFocus();

        loadFAQs();
    }

    private void loadFAQs(){

        database.collection("faqs").addSnapshotListener(new EventListener<QuerySnapshot>() {
            @Override
            public void onEvent(@Nullable QuerySnapshot querySnapshot,
                                @Nullable FirebaseFirestoreException e) {
                if (e != null) {
                    return;
                }
                for (DocumentChange change : querySnapshot.getDocumentChanges()) {
                    if (change.getType() == DocumentChange.Type.MODIFIED) {
                        FAQ updatedFAQ = new FAQ();
                        updatedFAQ.setId(change.getDocument().getId());
                        FAQ editedFAQ = databaseFaqs.get(databaseFaqs.indexOf(updatedFAQ));
                        editedFAQ.setQuestion(change.getDocument().getString("question"));
                        editedFAQ.setAnswer(change.getDocument().getString("answer"));
                    } else if (change.getType() == DocumentChange.Type.ADDED) {
                        FAQ newFAQ = new FAQ();
                        newFAQ.setId(change.getDocument().getId());
                        if(!databaseFaqs.contains(newFAQ)){
                            newFAQ.setQuestion(change.getDocument().getString("question"));
                            newFAQ.setAnswer(change.getDocument().getString("answer"));
                            databaseFaqs.add(newFAQ);
                        }
                    } else if (change.getType() == DocumentChange.Type.REMOVED) {
                        FAQ updatedFAQ = new FAQ();
                        updatedFAQ.setId(change.getDocument().getId());
                        int index = databaseFaqs.indexOf(updatedFAQ);
                        databaseFaqs.remove(index);
                    }
                }
                Log.i("COMPLETION", "Finished Updating Data");
                filterFilteredFAQS();
            }
        });
    }

    private void filterFilteredFAQS(){
        filteredFaqs.clear();
        for(FAQ currentFAQ : databaseFaqs){
            String query = faqSearchView.getQuery().toString().toLowerCase();
            if(currentFAQ.getQuestion().toLowerCase().contains(query) || currentFAQ.getAnswer().
                toLowerCase().contains(query)){
                filteredFaqs.add(currentFAQ);
            }
        }
        faqRecyclerViewAdapter.notifyDataSetChanged();
    }

    public void faqsPressBack(View view){
        finish();
    }

}
```

..4 GMailSender.java

```
package upmanila.bengeminuy.redcrossph;

import javax.activation.DataHandler;

import javax.activation.DataSource;

import javax.activation.FileDataSource;

import javax.mail.BodyPart;

import javax.mail.Message;

import javax.mail.Multipart;

import javax.mail.PasswordAuthentication;

import javax.mail.Session;

import javax.mail.Transport;
```

```
import javax.mail.internet.InternetAddress;

import javax.mail.internet.MimeBodyPart;

import javax.mail.internet.MimeMessage;

import javax.mail.internet.MimeMultipart;

import java.io.ByteArrayInputStream;

import java.io.IOException;

import java.io.InputStream;

import java.io.OutputStream;

import java.security.Security;

import java.util.Properties;


public class GMailSender extends javax.mail.Authenticator {

    private String mailhost = "smtp.gmail.com";
    private String user;
    private String password;
    private Session session;

    private Multipart _multipart = new MimeMultipart();

    static {

        Security.addProvider(new JSSEProvider());

    }

    public GMailSender(String user, String password) {

        this.user = user;
        this.password = password;

        Properties props = new Properties();
        props.setProperty("mail.transport.protocol", "smtp");
        props.setProperty("mail.host", mailhost);
        props.put("mail.smtp.auth", "true");
        props.put("mail.smtp.port", "465");
        props.put("mail.smtp.socketFactory.port", "465");
        props.put("mail.smtp.socketFactory.class", "javax.net.ssl.SSLSocketFactory");
        props.put("mail.smtp.socketFactory.fallback", "false");
        props.setProperty("mail.smtp.quitwait", "false");
        session = Session.getDefaultInstance(props, this);

    }


    protected PasswordAuthentication getPasswordAuthentication() {

        return new PasswordAuthentication(user, password);

    }


    public synchronized void sendMail(String subject, String body, String sender, String
         recipients) throws Exception {

        try {

            MimeMessage message = new MimeMessage(session);
            DataHandler handler = new DataHandler(new ByteArrayDataSource(body.getBytes(),"text/
                 plain"));
            message.setSender(new InternetAddress(sender));
            message.setSubject(subject);
            message.setDataHandler(handler);

            BodyPart messageBodyPart = new MimeBodyPart();
            messageBodyPart.setText(body);
            _multipart.addBodyPart(messageBodyPart);

            // Put parts in message

            message.setContent(_multipart);
            if (recipients.indexOf(',') > 0)
                message.setRecipients(Message.RecipientType.TO, InternetAddress.parse(recipients))
                    ;
            else
                message.setRecipient(Message.RecipientType.TO, new InternetAddress(recipients));
            Transport.send(message);

        } catch (Exception e) {
```

```
    }

}


public void addAttachment(String filename) throws Exception {

    BodyPart messageBodyPart = new MimeBodyPart();
    DataSource source = new FileDataSource(filename);
    messageBodyPart.setDataHandler(new DataHandler(source));
    messageBodyPart.setFileName("download image");

    _multipart.addBodyPart(messageBodyPart);

}


public class ByteArrayDataSource implements DataSource {

    private byte[] data;
    private String type;




    public ByteArrayDataSource(byte[] data, String type) {

        super();

        this.data = data;
        this.type = type;

    }


    public ByteArrayDataSource(byte[] data) {

        super();
        this.data = data;

    }




    public void setType(String type) {

        this.type = type;

    }


    public String getContentType() {

        if (type == null)
            return "application/octet-stream";

        else
            return type;

    }


    public InputStream getInputStream() throws IOException {

        return new ByteArrayInputStream(data);

    }


    public String getName() {

        return "ByteArrayDataSource";

    }


    public OutputStream getOutputStream() throws IOException {

        throw new IOException("Not Supported");
```

```
        }

    }

}
```

..5 HelpActivity.java

```
package upmanila.bengeminuy.redcrossph;

import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class HelpActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_help);
    }
}
```

..6 History.java

```
package upmanila.bengeminuy.redcrossph;

import com.google.firebase.Timestamp;

public class History {

    private String id;
    private String type;
    private Timestamp donationDate;
    private String branch;

    public History(){
        id = "";
        donationDate = null;
        branch = "";
    }

    public void setId(String id) {
        this.id = id;
    }

    public void setType(String type) {
        this.type = type;
    }

    public void setBranch(String branch) {
        this.branch = branch;
    }

    public void setDonationDate(Timestamp donationDate) {
        this.donationDate = donationDate;
    }

    public String getId() {
        return id;
    }

    public String getType() {
        return type;
    }

    public String getBranch() {
        return branch;
    }

    public Timestamp getDonationDate() {
        return donationDate;
    }

    @Override
    public boolean equals(Object obj) {
        History history = (History) obj;
        return id.compareTo(history.getId())==0;
    }
}
```

..7 HistoryActivity.java

```
package upmanila.bengeminuy.redcrossph;

import android.content.Context;
import android.content.DialogInterface;
import android.content.SharedPreferences;
```

```
import android.support.annotation.NonNull;
import android.support.annotation.Nullable;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.util.Log;
import android.view.View;
import android.widget.EditText;
import android.widget.SearchView;
import android.widget.TextView;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.FirebaseApp;
import com.google.firebase.Timestamp;
import com.google.firebase.firestore.DocumentChange;
import com.google.firebase.firestore.EventListener;
import com.google.firebase.firestore.FieldValue;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.FirebaseFirestoreException;
import com.google.firebase.firestore.QuerySnapshot;

import java.util.ArrayList;
import java.util.Collections;
import java.util.Date;

public class HistoryActivity extends AppCompatActivity {

    private Context context;
    private FirebaseFirestore database;
    private ArrayList<History> databaseHistory, filteredHistory;
    private RecyclerView historyRecyclerView;
    private RecyclerView.Adapter historyRecyclerViewAdapter;
    private SearchView historySearchView;
    private EditText serialCodeText;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_history);

        context = getApplicationContext();
        historyRecyclerView = findViewById(R.id.parentHistoryRecyclerView);
        historyRecyclerView.setLayoutManager(new LinearLayoutManager(this));
        historySearchView = findViewById(R.id.historySearchView);
        historySearchView.setIconifiedByDefault(false);

        FirebaseApp.initializeApp(context);
        database = FirebaseFirestore.getInstance();

        databaseHistory = new ArrayList<>();
        filteredHistory = new ArrayList<>();

        historyRecyclerViewAdapter = new HistoryAdapter(filteredHistory,this);
        historyRecyclerView.setAdapter(historyRecyclerViewAdapter);

        historySearchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
            @Override
            public boolean onQueryTextSubmit(String query) {
                filterHistory();
                return true;
            }

            @Override
            public boolean onQueryTextChange(String newText) {
                filterHistory();
                return true;
            }
        });
    }

    @Override
    protected void onStart() {
        super.onStart();

        historySearchView.setIconified(false);
        historySearchView.clearFocus();

        SharedPreferences serialPreferences = getSharedPreferences("upmanila.bengeminuy.redcrossph
            .serialcode",Context.MODE_PRIVATE);
        if(serialPreferences.getString("serial","").length()==0){
            serialCodeText = new EditText(this);
            AlertDialog.Builder builder = new AlertDialog.Builder(this);
            builder.setMessage("To proceed you need to input the serial code given by the
                Philippine Red Cross. Inputting wrong serial codes will bring you back to map
                screen.").setTitle("Notice");
            builder.setView(serialCodeText);
            builder.setPositiveButton("Enter", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int id) {
                    checkSerial();
```

```
            }
        });
        builder.setNegativeButton("Cancel.", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int id) {
                finish();
            }
        });
        AlertDialog dialog = builder.create();
        dialog.show();
    }
    else{
        loadHistory();
    }
}

private void checkSerial(){
    database.collection("serialcodes").whereEqualTo("serial", serialCodeText.getText().
        toString()).limit(1).get().addOnCompleteListener(new OnCompleteListener<QuerySnapshot
        >() {
        @Override
        public void onComplete(@NonNull Task<QuerySnapshot> task) {
            if(task.isSuccessful()){
                if(!task.getResult().isEmpty()){
                    SharedPreferences sharedPrefs = getApplicationContext().
                        getSharedPreferences("upmanila.bengeminuy.redcrossph.serialcode",
                        Context.MODE_PRIVATE);
                    SharedPreferences.Editor editor = sharedPrefs.edit();
                    editor.putString("serial", serialCodeText.getText().toString());
                    editor.apply();
                    loadHistory();
                }
                else{
                    TextView statusText = findViewById(R.id.statusText);
                    statusText.setText("Incorrect serial, try again later.");
                    statusText.setBackgroundColor(getResources().getColor(R.color.navheader));
                    statusText.setTextColor(getResources().getColor(R.color.
                        cardview_light_background));
                    findViewById(R.id.statusLabel).setBackgroundColor(getResources().getColor(
                        R.color.navheader));
                    ((TextView)findViewById(R.id.statusLabel)).setTextColor(getResources().
                        getColor(R.color.cardview_light_background));
                }

            }
            else{ }
        }
    });
}

private void loadHistory(){
    SharedPreferences serialPreferences = getSharedPreferences("upmanila.bengeminuy.redcrossph
        .serialcode",Context.MODE_PRIVATE);
    String serial = serialPreferences.getString("serial", "");
    Log.i("SERIAL", serial);
    database.collection("serialcodes").document(serial).collection("donationhistory").
        addSnapshotListener(new EventListener<QuerySnapshot>() {
        @Override
        public void onEvent(@Nullable QuerySnapshot querySnapshot,
                            @Nullable FirebaseFirestoreException e) {
            if (e != null) {
                return;
            }
            for (DocumentChange change : querySnapshot.getDocumentChanges()) {
                if (change.getType() == DocumentChange.Type.MODIFIED) {
                    History updatedHistory = new History();
                    updatedHistory.setId(change.getDocument().getId());
                    History editedHistory = databaseHistory.get(databaseHistory.indexOf(
                        updatedHistory));
                    editedHistory.setDonationDate(change.getDocument().getTimestamp("
                        donationdate"));
                    editedHistory.setType(change.getDocument().getString("type"));
                    editedHistory.setBranch(change.getDocument().getString("branch"));

                } else if (change.getType() == DocumentChange.Type.ADDED) {
                    History newHistory = new History();
                    newHistory.setId(change.getDocument().getId());
                    if(!databaseHistory.contains(newHistory)){
                        newHistory.setId(change.getDocument().getId());
                        newHistory.setDonationDate(change.getDocument().getTimestamp("
                            donationdate"));
                        newHistory.setType(change.getDocument().getString("type"));
                        newHistory.setBranch(change.getDocument().getString("branch"));
                        databaseHistory.add(newHistory);
                    }
                } else if (change.getType() == DocumentChange.Type.REMOVED) {
                    History updatedHistory = new History();
                    updatedHistory.setId(change.getDocument().getId());
                    databaseHistory.remove(databaseHistory.indexOf(updatedHistory));
                }
            }
            updateLatestHistory();
            filterHistory();
```

```
            }
        });
    }

    private void filterHistory(){
        filteredHistory.clear();
        for(History currentHistory : databaseHistory){
            String query = historySearchView.getQuery().toString().toLowerCase();
            if(currentHistory.getBranch().toLowerCase().contains(query) || currentHistory.
                getDonationDate().toString().toLowerCase().contains(query)){
                filteredHistory.add(currentHistory);
            }
        }
        Collections.reverse(filteredHistory);
        historyRecyclerViewAdapter.notifyDataSetChanged();
    }

    private void updateLatestHistory(){
        TextView latestDonationDate = findViewById(R.id.latestDonationDate);
        TextView latestDonationType = findViewById(R.id.latestDonationType);
        TextView nextDonationDate = findViewById(R.id.nextDonationDate);
        if(databaseHistory.size() > 0){
            latestDonationDate.setText(databaseHistory.get(databaseHistory.size()-1).
                getDonationDate().toDate().toString().replace("GMT+08:00 ",""));
            latestDonationType.setText(databaseHistory.get(databaseHistory.size()-1).getType());
            FieldValue.serverTimestamp();
            long currentTime = System.currentTimeMillis();
            long latestDonationTime = databaseHistory.get(databaseHistory.size()-1).
                getDonationDate().getSeconds()*1000;

            if(latestDonationType.getText().toString().toLowerCase().compareTo("apheresis")==0){
                if(currentTime-latestDonationTime > 1209600000){
                    nextDonationDate.setText("Now Ready.");
                }
                else{
                    long nextTime = 1209600000 + latestDonationTime;
                    Date nextDate = new Date(nextTime);
                    nextDonationDate.setText(nextDate.toString().replace("GMT+08:00 ",""));
                }
            }
            else{
                if((currentTime-latestDonationTime)/1000 > 7890000){
                    nextDonationDate.setText("Now Ready");
                }
                else{
                    long nextTime = latestDonationTime + 7890000000L;
                    Date nextDate = new Date(nextTime);
                    nextDonationDate.setText(nextDate.toString().replace("GMT+08:00 ",""));
                }
            }
        }
        else{
            latestDonationDate.setText("None");
            latestDonationType.setText("None");
            nextDonationDate.setText("Now Ready.");
        }
    }

    public void historyPressBack(View view){
        finish();
    }

}
```

..8 HistoryAdapter.java

```
package upmanila.bengeminuy.redcrossph;

import android.content.Context;
import android.support.annotation.NonNull;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

import java.util.List;

public class HistoryAdapter extends RecyclerView.Adapter<HistoryAdapter.ViewHolder> {

    private List<History> historyModel;
    private Context context;

    public HistoryAdapter(List<History> histories, Context context){
        this.historyModel = histories;
        this.context = context;
    }

    @NonNull
    @Override
    public HistoryAdapter.ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
```

```
        View v = LayoutInflater.from(parent.getContext())
                .inflate(R.layout.history_card, parent, false);
        return new HistoryAdapter.ViewHolder(v);
    }

    @Override
    public void onBindViewHolder(@NonNull HistoryAdapter.ViewHolder holder, int position) {
        History history = historyModel.get(position);

        holder.donationTypeText.setText(history.getType().toUpperCase());

        if(history.getType().toLowerCase().compareTo("apheresis")==0){
            holder.donationTypeText.setBackgroundColor(context.getResources().getColor(R.color.
                newstag));
        }
        else{
            holder.donationTypeText.setBackgroundColor(context.getResources().getColor(R.color.
                promotag));
        }

        holder.dateText.setText(history.getDonationDate().toDate().toString().replace("GMT+08:00
            ",""));
        holder.locationText.setText(history.getBranch() + " Chapter");
    }

    @Override
    public int getItemCount() {
        return historyModel.size();
    }

    @Override
    public int getItemViewType(int position) {
        return position;
    }

    public class ViewHolder extends RecyclerView.ViewHolder{

        public TextView donationTypeText;
        public TextView dateText;
        public TextView locationText;

        public ViewHolder(View itemView) {
            super(itemView);

            donationTypeText = itemView.findViewById(R.id.donationtypeText);
            dateText = itemView.findViewById(R.id.dateText);
            locationText = itemView.findViewById(R.id.locationText);

        }
    }

}
```

..9 InquiryActivity.java

```
package upmanila.bengeminuy.redcrossph;

import android.content.Context;
import android.content.DialogInterface;
import android.content.SharedPreferences;
import android.os.Looper;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.Editable;
import android.text.TextUtils;
import android.text.TextWatcher;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

public class InquiryActivity extends AppCompatActivity {

    Context context;
    EditText emailText, contactText, inquiryText;
    Button submitButton;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_inquiry);

        emailText = findViewById(R.id.emailText);
        contactText = findViewById(R.id.contactText);
        inquiryText = findViewById(R.id.inquiryText);
        submitButton = findViewById(R.id.submitButton);
        context = getApplicationContext();
    }

    @Override
    protected void onStart() {
```

```
        super.onStart();

        SharedPreferences sharedPrefs = getApplicationContext().getSharedPreferences("upmanila.
            bengeminuy.redcrossph.flags", Context.MODE_PRIVATE);
        if(!sharedPrefs.getBoolean("acceptedinquiryterms",false)){
            AlertDialog.Builder builder = new AlertDialog.Builder(this);
            builder.setMessage("Your email and contact number won't be stored in a database but
                 will only be used as details for your message.").setTitle("Disclaimer");
            builder.setPositiveButton("I Understand", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int id) {
                    SharedPreferences sharedPrefs = getApplicationContext().getSharedPreferences("
                        upmanila.bengeminuy.redcrossph.flags", Context.MODE_PRIVATE);
                    SharedPreferences.Editor editor = sharedPrefs.edit();
                    editor.putBoolean("acceptedinquiryterms", true);
                    editor.apply();
                }
            });
            builder.setNegativeButton("I Refuse.", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int id) {
                    finish();
                }
            });
            AlertDialog dialog = builder.create();
            dialog.show();
        }

        emailText.clearFocus();
        contactText.clearFocus();
        inquiryText.clearFocus();

        submitButton.setBackgroundColor(getResources().getColor(R.color.disabled));
        submitButton.setEnabled(false);

        addEditListeners();
    }

    private void addEditListeners(){
        emailText.addTextChangedListener(new TextWatcher() {
            @Override
            public void beforeTextChanged(CharSequence s, int start, int count, int after) {}

            @Override
            public void onTextChanged(CharSequence s, int start, int before, int count) {
                EditText emailText = (EditText) findViewById(R.id.emailText);
                EditText contactText = (EditText) findViewById(R.id.contactText);
                EditText inquiryText = (EditText) findViewById(R.id.inquiryText);

                boolean validEmail = !TextUtils.isEmpty(emailText.getText().toString());
                boolean validContact = !TextUtils.isEmpty(contactText.getText().toString());
                boolean validInquiry = !TextUtils.isEmpty(inquiryText.getText().toString());

                if((validEmail || validContact) && validInquiry){

                    Button submitButton = (Button) findViewById(R.id.submitButton);
                    submitButton.setEnabled(true);
                    submitButton.setBackgroundColor(getResources().getColor(R.color.navheader));
                    submitButton.setText("Submit");
                }
                else{
                    Button submitButton = (Button) findViewById(R.id.submitButton);
                    submitButton.setEnabled(true);
                    submitButton.setBackgroundColor(getResources().getColor(R.color.disabled));
                    submitButton.setText("Submit");
                }
            }

            @Override
            public void afterTextChanged(Editable s) {}

        });

        contactText.addTextChangedListener(new TextWatcher() {
            @Override
            public void beforeTextChanged(CharSequence s, int start, int count, int after) {}

            @Override
            public void onTextChanged(CharSequence s, int start, int before, int count) {
                EditText emailText = (EditText) findViewById(R.id.emailText);
                EditText contactText = (EditText) findViewById(R.id.contactText);
                EditText inquiryText = (EditText) findViewById(R.id.inquiryText);

                boolean validEmail = !TextUtils.isEmpty(emailText.getText().toString());
                boolean validContact = !TextUtils.isEmpty(contactText.getText().toString());
                boolean validInquiry = !TextUtils.isEmpty(inquiryText.getText().toString());

                if((validEmail || validContact) && validInquiry){

                    Button submitButton = (Button) findViewById(R.id.submitButton);
                    submitButton.setEnabled(true);
                    submitButton.setBackgroundColor(getResources().getColor(R.color.navheader));
                    submitButton.setText("Submit");
```

```
            }
            else{
                Button submitButton = (Button) findViewById(R.id.submitButton);
                submitButton.setEnabled(true);
                submitButton.setBackgroundColor(getResources().getColor(R.color.disabled));
                submitButton.setText("Submit");
            }
        }

        @Override
        public void afterTextChanged(Editable s) {}

    });

    inquiryText.addTextChangedListener(new TextWatcher() {
        @Override
        public void beforeTextChanged(CharSequence s, int start, int count, int after) {}

        @Override
        public void onTextChanged(CharSequence s, int start, int before, int count) {
            EditText emailText = (EditText) findViewById(R.id.emailText);
            EditText contactText = (EditText) findViewById(R.id.contactText);
            EditText inquiryText = (EditText) findViewById(R.id.inquiryText);

            boolean validEmail = !TextUtils.isEmpty(emailText.getText().toString());
            boolean validContact = !TextUtils.isEmpty(contactText.getText().toString());
            boolean validInquiry = !TextUtils.isEmpty(inquiryText.getText().toString());

            if((validEmail || validContact) && validInquiry){

                Button submitButton = (Button) findViewById(R.id.submitButton);
                submitButton.setEnabled(true);
                submitButton.setBackgroundColor(getResources().getColor(R.color.navheader));
                submitButton.setText("Submit");
            }
            else{
                Button submitButton = (Button) findViewById(R.id.submitButton);
                submitButton.setEnabled(true);
                submitButton.setBackgroundColor(getResources().getColor(R.color.disabled));
                submitButton.setText("Submit");
            }
        }

        @Override
        public void afterTextChanged(Editable s) {}

    });
}

public void inquiryPressBack(View view){
    finish();
}

public void sendEmail(View view){

    new Thread(new Runnable() {

        public void run() {

            try {

                submitButton.setText("Sending Inquiry...");

                GMailSender sender = new GMailSender(

                        "redcrossphtestee@gmail.com",

                        "testAccount");

                String message = "Email: " + emailText.getText().toString() + "\n\nContact
                    Number: " + contactText.getText().toString() + "\n\nInquiry:\n\n\t" +
                    inquiryText.getText().toString();

                sender.sendMail("[RED CROSS PH APP] INQUIRY", message,

                        "redcrossphtestee@gmail.com",

                        "bengeminuy@gmail.com");

                submitButton.setText("Finished Sending.");

            } catch (Exception e) {

                submitButton.setText("Error Sending... Please try again later.");

            }

        }

    }).start();

}
```

```
}
```

..10 JSSEProvider.java

```
package upmanila.bengeminuy.redcrossph;

import java.security.AccessController;

import java.security.Provider;


public final class JSSEProvider extends Provider {

    public JSSEProvider() {

        super("HarmonyJSSE", 1.0, "Harmony JSSE Provider");

        AccessController

                .doPrivileged(new java.security.PrivilegedAction<Void>() {

                    public Void run() {

                        put("SSLContext.TLS", "org.apache.harmony.xnet.provider.jsse.
                            SSLContextImpl");

                        put("Alg.Alias.SSLContext.TLSv1", "TLS");

                        put("KeyManagerFactory.X509", "org.apache.harmony.xnet.provider.jsse.
                            KeyManagerFactoryImpl");

                        put("TrustManagerFactory.X509", "org.apache.harmony.xnet.provider.jsse.
                            TrustManagerFactoryImpl");

                        return null;

                    }

                });

    }

}
```

..11 MainActivity.java

```
package upmanila.bengeminuy.redcrossph;

import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.content.pm.PackageManager;
import android.graphics.Color;
import android.location.Location;
import android.support.annotation.Nullable;
import android.support.design.widget.NavigationView;
import android.support.v4.app.ActivityCompat;
import android.support.v4.view.GravityCompat;
import android.support.v4.widget.DrawerLayout;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.MenuItem;
import android.view.View;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.SearchView;
import android.widget.TextView;

import com.gelitenight.waveview.library.WaveView;
import com.google.android.gms.location.FusedLocationProviderClient;
import com.google.android.gms.location.LocationCallback;
import com.google.android.gms.location.LocationRequest;
import com.google.android.gms.location.LocationResult;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MapStyleOptions;
import com.google.android.gms.maps.model.MarkerOptions;
import com.google.android.gms.maps.model.PolylineOptions;
import com.google.firebase.FirebaseApp;
import com.google.firebase.firestore.DocumentChange;
import com.google.firebase.firestore.EventListener;
```

```
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.FirebaseFirestoreException;
import com.google.firebase.firestore.QuerySnapshot;
import com.google.maps.DirectionsApi;
import com.google.maps.DirectionsApiRequest;
import com.google.maps.GeoApiContext;
import com.google.maps.model.DirectionsLeg;
import com.google.maps.model.DirectionsResult;
import com.google.maps.model.DirectionsRoute;
import com.google.maps.model.DirectionsStep;
import com.google.maps.model.EncodedPolyline;

import java.util.ArrayList;
import java.util.Collections;
import java.util.Date;
import java.util.List;

import static android.Manifest.permission.ACCESS_FINE_LOCATION;

public class MainActivity extends AppCompatActivity implements OnMapReadyCallback {

    private Context context;
    private LocationCallback mLocationCallback;
    private GoogleMap googleMap;
    private FirebaseFirestore database;
    private ArrayList<PRCBranch> branchInformation, filteredBranch;
    private DrawerLayout mDrawerLayout;
    private SearchView searchView;
    private String globalquery;
    private LinearLayout mainCard;
    private WaveView waveView;
    private WaveHelper mWaveHelper;
    private List<LatLng> path;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager().
            findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);

        SharedPreferences flagPrefs = getApplicationContext().getSharedPreferences("upmanila.
            bengeminuy.redcrossph.flags", Context.MODE_PRIVATE);
        SharedPreferences.Editor flagEditor = flagPrefs.edit();
        flagEditor.putBoolean("firstlocationflag", false);
        flagEditor.apply();

        context = getApplicationContext();
        mDrawerLayout = findViewById(R.id.drawer_layout);
        searchView = findViewById(R.id.mainSearchView);
        searchView.setIconifiedByDefault(false);
        mainCard = findViewById(R.id.mainCard);
        waveView = findViewById(R.id.wave);
        mWaveHelper = new WaveHelper(waveView);

        FirebaseApp.initializeApp(context);
        database = FirebaseFirestore.getInstance();

        branchInformation = new ArrayList<>();
        filteredBranch = new ArrayList<>();
        globalquery = "";
        path = new ArrayList<>();

        addFunctionsToNavBar();

        searchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
            @Override
            public boolean onQueryTextSubmit(String query) {
                globalquery = query;
                filterBranch();
                return true;
            }

            @Override
            public boolean onQueryTextChange(String newText) {
                globalquery = newText;
                filterBranch();
                return true;
            }
        });
    }

    @Override
    protected void onStart() {
        super.onStart();

        searchView.setIconified(false);
        searchView.clearFocus();

        mainCard.setVisibility(View.GONE);
```

```
        waveView.setShapeType(WaveView.ShapeType.CIRCLE);
        waveView.setWaveColor(getResources().getColor(R.color.navheader), Color.parseColor("#28
            f16d7a"));
        waveView.setBorder(1,getResources().getColor(R.color.navheader));
        mWaveHelper.start();
    }

    @Override
    public void onBackPressed() {
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setMessage("Are you leaving the app?");
        builder.setPositiveButton("Exit", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int id) {
                finish();
            }
        });
        builder.setNegativeButton("Go Back", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int id) {
            }
        });
        AlertDialog dialog = builder.create();
        dialog.show();
    }

    @Override
    public void onMapReady(final GoogleMap googleMap) {
        this.googleMap = googleMap;
        googleMap.setMapStyle(MapStyleOptions.loadRawResourceStyle(this, R.raw.style_json));
        googleMap.getUiSettings().setCompassEnabled(false);
        googleMap.getUiSettings().setRotateGesturesEnabled(false);

        getLastLocation();
        prepareLocationUpdates();

        getBranchInformation();

        searchView.setIconified(false);
        searchView.clearFocus();
    }

    private void getBranchInformation() {
        database.collection("branches").addSnapshotListener(new EventListener<QuerySnapshot>() {
            @Override
            public void onEvent(@Nullable QuerySnapshot querySnapshot,
                                @Nullable FirebaseFirestoreException e) {
                if (e != null) {
                    return;
                }
                for (DocumentChange change : querySnapshot.getDocumentChanges()) {
                    if (change.getType() == DocumentChange.Type.MODIFIED) {
                        PRCBranch updatedBranchInformation = new PRCBranch();
                        updatedBranchInformation.setId(change.getDocument().getId());
                        PRCBranch editedBranchInformation = branchInformation.get(
                            branchInformation.indexOf(updatedBranchInformation));
                        editedBranchInformation.setBranchName(change.getDocument().getString("
                            branchname"));
                        editedBranchInformation.setApheresis(change.getDocument().getBoolean("
                            capabilities.apheresis"));
                        editedBranchInformation.setConventional(change.getDocument().getBoolean("
                            capabilities.conventional"));
                        editedBranchInformation.setEmail(change.getDocument().getString("
                            contactdetails.email"));
                        editedBranchInformation.setFax((ArrayList<String>) change.getDocument().
                            get("contactdetails.fax"));
                        editedBranchInformation.setTelephone((ArrayList<String>) change.
                            getDocument().get("contactdetails.telephone"));
                        editedBranchInformation.setAddress(change.getDocument().getString("
                            locationdetails.address"));
                        editedBranchInformation.setGeoPoint(change.getDocument().getGeoPoint("
                            locationdetails.geopoint"));
                        editedBranchInformation.setMonday(change.getDocument().getBoolean("
                            operatingtime.operatingdays.Monday"));
                        editedBranchInformation.setTuesday(change.getDocument().getBoolean("
                            operatingtime.operatingdays.Tuesday"));
                        editedBranchInformation.setWednesday(change.getDocument().getBoolean("
                            operatingtime.operatingdays.Wednesday"));
                        editedBranchInformation.setThursday(change.getDocument().getBoolean("
                            operatingtime.operatingdays.Thursday"));
                        editedBranchInformation.setFriday(change.getDocument().getBoolean("
                            operatingtime.operatingdays.Friday"));
                        editedBranchInformation.setSaturday(change.getDocument().getBoolean("
                            operatingtime.operatingdays.Saturday"));
                        editedBranchInformation.setSunday(change.getDocument().getBoolean("
                            operatingtime.operatingdays.Sunday"));
                        editedBranchInformation.setOperatingHours(change.getDocument().getString("
                            operatingtime.operatinghours"));
                        editedBranchInformation.setCurrentUnits(Double.parseDouble(change.
                            getDocument().getString("units.current")));
                        editedBranchInformation.setTargetUnits(Double.parseDouble(change.
                            getDocument().getString("units.target")));
                    } else if (change.getType() == DocumentChange.Type.ADDED) {
```

```
                        PRCBranch newBranchInformation = new PRCBranch();
                        newBranchInformation.setId(change.getDocument().getId());
                        newBranchInformation.setBranchName(change.getDocument().getString("
                            branchname"));
                        newBranchInformation.setApheresis(change.getDocument().getBoolean("
                            capabilities.apheresis"));
                        newBranchInformation.setConventional(change.getDocument().getBoolean("
                            capabilities.conventional"));
                        newBranchInformation.setEmail(change.getDocument().getString("
                            contactdetails.email"));
                        newBranchInformation.setFax((ArrayList<String>) change.getDocument().get("
                            contactdetails.fax"));
                        newBranchInformation.setTelephone((ArrayList<String>) change.getDocument()
                            .get("contactdetails.telephone"));
                        newBranchInformation.setAddress(change.getDocument().getString("
                            locationdetails.address"));
                        newBranchInformation.setGeoPoint(change.getDocument().getGeoPoint("
                            locationdetails.geopoint"));
                        newBranchInformation.setMonday(change.getDocument().getBoolean("
                            operatingtime.operatingdays.Monday"));
                        newBranchInformation.setTuesday(change.getDocument().getBoolean("
                            operatingtime.operatingdays.Tuesday"));
                        newBranchInformation.setWednesday(change.getDocument().getBoolean("
                            operatingtime.operatingdays.Wednesday"));
                        newBranchInformation.setThursday(change.getDocument().getBoolean("
                            operatingtime.operatingdays.Thursday"));
                        newBranchInformation.setFriday(change.getDocument().getBoolean("
                            operatingtime.operatingdays.Friday"));
                        newBranchInformation.setSaturday(change.getDocument().getBoolean("
                            operatingtime.operatingdays.Saturday"));
                        newBranchInformation.setSunday(change.getDocument().getBoolean("
                            operatingtime.operatingdays.Sunday"));
                        newBranchInformation.setOperatingHours(change.getDocument().getString("
                            operatingtime.operatinghours"));
                        newBranchInformation.setCurrentUnits(Double.parseDouble(change.getDocument
                            ().getString("units.current")));
                        newBranchInformation.setTargetUnits(Double.parseDouble(change.getDocument
                            ().getString("units.target")));
                        branchInformation.add(newBranchInformation);
                    } else if (change.getType() == DocumentChange.Type.REMOVED) {
                        PRCBranch updatedBranchInformation = new PRCBranch();
                        updatedBranchInformation.setId(change.getDocument().getId());
                        branchInformation.remove(branchInformation.indexOf(
                            updatedBranchInformation));
                    }
                }
                googleMap.clear();
                getLastLocation();
                updateUserDistance();
                filterBranch();
                updateNearestChapters();
                addBranchMarkers();
            }
        });
    }

    private void getLastLocation() {
        SharedPreferences sharedPrefs = context.getSharedPreferences("upmanila.bengeminuy.
            redcrossph.userlocation", Context.MODE_PRIVATE);
        Double latitude = Double.parseDouble(sharedPrefs.getString("userlatitude", "14.647248"));
        Double longitude = Double.parseDouble(sharedPrefs.getString("userlongitude", "120.991966")
            );
        LatLng currentLocation = new LatLng(latitude, longitude);
        if (ActivityCompat.checkSelfPermission(MainActivity.this, ACCESS_FINE_LOCATION) ==
            PackageManager.PERMISSION_GRANTED) {
            googleMap.addMarker(new MarkerOptions().position(currentLocation).title("Latest
                Location"));
        }
        googleMap.animateCamera(CameraUpdateFactory.newLatLngZoom(currentLocation, 16));
    }

    private void prepareLocationUpdates() {
        if (ActivityCompat.checkSelfPermission(MainActivity.this, ACCESS_FINE_LOCATION) !=
            PackageManager.PERMISSION_GRANTED) {
            return;
        }
        FusedLocationProviderClient mFusedLocationClient = LocationServices.
            getFusedLocationProviderClient(getApplicationContext());
        mLocationCallback = new LocationCallback() {
            @Override
            public void onLocationResult(LocationResult locationResult) {
                if (locationResult == null) {
                    return;
                }
                Location latestLocation = locationResult.getLastLocation();
                Context context = getApplicationContext();
                SharedPreferences sharedPrefs = context.getSharedPreferences("upmanila.bengeminuy.
                    redcrossph.userlocation", Context.MODE_PRIVATE);

                SharedPreferences.Editor editor = sharedPrefs.edit();
                editor.putString("userlatitude", String.valueOf(latestLocation.getLatitude()));
                editor.putString("userlongitude", String.valueOf(latestLocation.getLongitude()));
```

```
                editor.apply();

                LatLng currentLocation = new LatLng(latestLocation.getLatitude(), latestLocation.
                    getLongitude());
                googleMap.clear();

                addBranchMarkers();

                googleMap.addMarker(new MarkerOptions().position(currentLocation).title("Current
                    Location"));

                SharedPreferences flagPrefs = getApplicationContext().getSharedPreferences("
                    upmanila.bengeminuy.redcrossph.flags", Context.MODE_PRIVATE);

                if (!flagPrefs.getBoolean("firstlocationflag", false)) {
                    SharedPreferences.Editor flagEditor = flagPrefs.edit();
                    flagEditor.putBoolean("firstlocationflag", true);
                    flagEditor.apply();
                    googleMap.animateCamera(CameraUpdateFactory.newLatLngZoom(currentLocation, 16)
                        );
                }
                updateUserDistance();
                filterBranch();
                updateNearestChapters();
            }
        };
        mFusedLocationClient.requestLocationUpdates(getLocationRequest(), mLocationCallback, null)
            ;
    }

    private LocationRequest getLocationRequest() {
        LocationRequest locationRequest = new LocationRequest();
        locationRequest.setInterval(10000);
        locationRequest.setFastestInterval(5000);
        locationRequest.setPriority(LocationRequest.PRIORITY_HIGH_ACCURACY);
        return locationRequest;
    }

    private void updateUserDistance(){
        if (ActivityCompat.checkSelfPermission(MainActivity.this, ACCESS_FINE_LOCATION) !=
            PackageManager.PERMISSION_GRANTED) {
            return;
        }
        for(PRCBranch currentBranch : branchInformation){
            SharedPreferences sharedPrefs = getApplicationContext().getSharedPreferences("upmanila
                .bengeminuy.redcrossph.userlocation", Context.MODE_PRIVATE);
            Double userlatitude = Double.parseDouble(sharedPrefs.getString("userlatitude",
                "-1.00"));
            Double userlongitude = Double.parseDouble(sharedPrefs.getString("userlongitude",
                "-1.00"));
            float[] distance = new float[1000];
            Location.distanceBetween(userlatitude,userlongitude,currentBranch.getGeoPoint().
                getLatitude(),currentBranch.getGeoPoint().getLongitude(),distance);
            currentBranch.setUserDistance(distance[0]);
        }
    }

    private void addBranchMarkers() {
        for (PRCBranch currentBranchInformation : branchInformation) {
            Double latitude = currentBranchInformation.getGeoPoint().getLatitude();
            Double longitude = currentBranchInformation.getGeoPoint().getLongitude();
            googleMap.addMarker(new MarkerOptions()
                    .draggable(false)
                    .position(new LatLng(latitude, longitude))
                    .title(currentBranchInformation.getBranchName() + " Chapter")
                    .icon(BitmapDescriptorFactory.fromResource(R.drawable.prcmarker50))
            );
        }

        if (ActivityCompat.checkSelfPermission(MainActivity.this, ACCESS_FINE_LOCATION) !=
            PackageManager.PERMISSION_GRANTED) {
            return;
        }
        if (path.size() > 0) {
            PolylineOptions opts = new PolylineOptions().addAll(path).color(getResources().
                getColor(R.color.navheader)).width(5);
            googleMap.addPolyline(opts);
        }
    }

    public void centerUserLocation(View view) {
        if (ActivityCompat.checkSelfPermission(MainActivity.this, ACCESS_FINE_LOCATION) !=
            PackageManager.PERMISSION_GRANTED) {
            return;
        }
        SharedPreferences sharedPrefs = getApplicationContext().getSharedPreferences("upmanila.
            bengeminuy.redcrossph.userlocation", Context.MODE_PRIVATE);
        Double userlatitude = Double.parseDouble(sharedPrefs.getString("userlatitude", "-1.00"));
        Double userlongitude = Double.parseDouble(sharedPrefs.getString("userlongitude", "-1.00"))
            ;
        googleMap.animateCamera(CameraUpdateFactory.newLatLng(new LatLng(userlatitude,
            userlongitude)));
```

```
}

public void openNavBar(View view) {
    mDrawerLayout.openDrawer(GravityCompat.START);
}

private void addFunctionsToNavBar() {
    NavigationView navigationView = findViewById(R.id.nav_view);
    navigationView.setNavigationItemSelectedListener(
        new NavigationView.OnNavigationItemSelectedListener() {
            @Override
            public boolean onNavigationItemSelected(MenuItem menuItem) {
                mDrawerLayout.closeDrawers();
                if(menuItem.getItemId()==R.id.nav_news){
                    Intent newsActivity = new Intent(getApplicationContext(), NewsActivity.
                        class);
                    startActivity(newsActivity);
                }
                else if(menuItem.getItemId() == R.id.nav_faqs){
                    Intent faqsActivity = new Intent(getApplicationContext(), FAQsActivity.
                        class);
                    startActivity(faqsActivity);
                }
                else if(menuItem.getItemId() == R.id.nav_form){
                    Intent inquiryActivity = new Intent(getApplicationContext(),
                        InquiryActivity.class);
                    startActivity(inquiryActivity);
                }
                else if(menuItem.getItemId() == R.id.nav_apply){
                    Intent questionActivity = new Intent(getApplicationContext(),
                        QuestionActivity.class);
                    startActivity(questionActivity);
                }
                else if(menuItem.getItemId() == R.id.nav_donor){
                    Intent historyActivity = new Intent(getApplicationContext(),
                        HistoryActivity.class);
                    startActivity(historyActivity);
                }
                else if(menuItem.getItemId() == R.id.nav_help){
                    Intent helpActivity = new Intent(getApplicationContext(), HelpActivity.
                        class);
                    startActivity(helpActivity);
                }
                else if(menuItem.getItemId() == R.id.nav_exit){
                    finish();
                }

                return true;
            }
        });
}

public void filterBranch(){
    filteredBranch.clear();
    for(PRCBranch currentBranch : branchInformation){
        if(currentBranch.getBranchName().toLowerCase().contains(globalquery)){
            filteredBranch.add(currentBranch);
        }
        else if(currentBranch.getAddress().toLowerCase().contains(globalquery)){
            filteredBranch.add(currentBranch);
        }
    }
    updateNearestChapters();
    addBranchMarkers();
}

private void updateNearestChapters(){
    TextView nearestChapter = findViewById(R.id.nearestChapterName);
    TextView nearestChapterDistance = findViewById(R.id.nearestChapterDistance);
    ImageView highlightButton = findViewById(R.id.highlightNearestChapter);

    ArrayList<PRCBranch> prcBranchCopy = new ArrayList<>();
    prcBranchCopy.addAll(filteredBranch);
    Collections.sort(prcBranchCopy,new PRCBranchComparator());

    if(prcBranchCopy.size() >= 1 ) {
        nearestChapter.setText(prcBranchCopy.get(0).getBranchName());
        nearestChapterDistance.setText(String.valueOf(prcBranchCopy.get(0).getUserDistance() /
            1000) + "km");
        nearestChapter.setVisibility(View.VISIBLE);
        nearestChapterDistance.setVisibility(View.VISIBLE);
        highlightButton.setVisibility(View.VISIBLE);

        waveView.setWaterLevelRatio(prcBranchCopy.get(0).getCurrentUnits().floatValue()/
            prcBranchCopy.get(0).getTargetUnits().floatValue());

        TextView branchName = findViewById(R.id.branchName);
        branchName.setText(prcBranchCopy.get(0).getBranchName());

        TextView statusText = findViewById(R.id.statusText);
        String currentDay = (new Date(System.currentTimeMillis())).toString().substring(0,3);
        if(currentDay.compareTo("Mon")==0){
```

```
    if(prcBranchCopy.get(0).getMonday()){
        statusText.setText("Open today - " + prcBranchCopy.get(0).getOperatingHours())
            ;
    }
    else{
        statusText.setText("Closed today");
    }
}
else if(currentDay.compareTo("Tue")==0){
    if(prcBranchCopy.get(0).getTuesday()){
        statusText.setText("Open today - " + prcBranchCopy.get(0).getOperatingHours())
            ;
    }
    else{
        statusText.setText("Closed today");
    }
}
else if(currentDay.compareTo("Wed")==0){
    if(prcBranchCopy.get(0).getWednesday()){
        statusText.setText("Open today - " + prcBranchCopy.get(0).getOperatingHours())
            ;
    }
    else{
        statusText.setText("Closed today");
    }
}
else if(currentDay.compareTo("Thu")==0){
    if(prcBranchCopy.get(0).getThursday()){
        statusText.setText("Open today - " + prcBranchCopy.get(0).getOperatingHours())
            ;
    }
    else{
        statusText.setText("Closed today");
    }
}
else if(currentDay.compareTo("Fri")==0){
    if(prcBranchCopy.get(0).getFriday()){
        statusText.setText("Open today - " + prcBranchCopy.get(0).getOperatingHours())
            ;
    }
    else{
        statusText.setText("Closed today");
    }
}
else if(currentDay.compareTo("Sat")==0){
    if(prcBranchCopy.get(0).getSaturday()){
        statusText.setText("Open today - " + prcBranchCopy.get(0).getOperatingHours())
            ;
    }
    else{
        statusText.setText("Closed today");
    }
}
else if(currentDay.compareTo("Sun")==0){
    if(prcBranchCopy.get(0).getSunday()){
        statusText.setText("Open today - " + prcBranchCopy.get(0).getOperatingHours())
            ;
    }
    else{
        statusText.setText("Closed today");
    }
}

TextView capabilitiesText = findViewById(R.id.capabilitiesText);
if(prcBranchCopy.get(0).getApheresis() && prcBranchCopy.get(0).getConventional()){
    capabilitiesText.setText("Apheresis & Conventional");
}
else if(prcBranchCopy.get(0).getApheresis()){
    capabilitiesText.setText("Apheresis Only");
}
else if(prcBranchCopy.get(0).getConventional()){
    capabilitiesText.setText("Conventional Only");
}
else{
    capabilitiesText.setText("None");
}

TextView email = findViewById(R.id.emailText);
email.setText(prcBranchCopy.get(0).getEmail());

TextView address = findViewById(R.id.addressText);
address.setText(prcBranchCopy.get(0).getAddress());

TextView telephone = findViewById(R.id.telephoneText);
telephone.setText(prcBranchCopy.get(0).getTelephone().toString().substring(1,
     prcBranchCopy.get(0).getTelephone().toString().length()-1));

TextView fax = findViewById(R.id.faxText);
fax.setText(prcBranchCopy.get(0).getFax().toString().substring(1,prcBranchCopy.get(0).
     getFax().toString().length()-1));

SharedPreferences sharedPreferences = getSharedPreferences("upmanila.bengeminuy.
```

```
                redcrossph.userlocation", Context.MODE_PRIVATE);
            String latitude = sharedPreferences.getString("userlatitude", "14.647248");
            String longitude = sharedPreferences.getString("userlongitude", "120.991966");

            String destlatitude = String.valueOf(prcBranchCopy.get(0).getGeoPoint().getLatitude())
                ;
            String destlongitude = String.valueOf(prcBranchCopy.get(0).getGeoPoint().getLongitude
                ());

            if (ActivityCompat.checkSelfPermission(MainActivity.this, ACCESS_FINE_LOCATION) !=
                PackageManager.PERMISSION_GRANTED) {
                return;
            }
            path.clear();
            GeoApiContext context = new GeoApiContext.Builder().apiKey("AIzaSyC-
                JAwpBItOb9FIaQkWHUGNr3L001I0yws").build();
            DirectionsApiRequest req = DirectionsApi.getDirections(context, latitude + "," +
                longitude, destlatitude + "," + destlongitude);
            try {
                DirectionsResult res = req.await();
                if (res.routes != null && res.routes.length > 0) {
                    DirectionsRoute route = res.routes[0];
                    if (route.legs !=null) {
                        for(int i=0; i<route.legs.length; i++) {
                            DirectionsLeg leg = route.legs[i];
                            if (leg.steps != null) {
                                for (int j=0; j<leg.steps.length;j++){
                                    DirectionsStep step = leg.steps[j];
                                    if (step.steps != null && step.steps.length >0) {
                                        for (int k=0; k<step.steps.length;k++){
                                            DirectionsStep step1 = step.steps[k];
                                            EncodedPolyline points1 = step1.polyline;
                                            if (points1 != null) {
                                                List<com.google.maps.model.LatLng> coords1 =
                                                    points1.decodePath();
                                                for (com.google.maps.model.LatLng coord1 : coords1
                                                    ) {
                                                    path.add(new LatLng(coord1.lat, coord1.lng));
                                                }
                                            }
                                        }
                                    } else {
                                        EncodedPolyline points = step.polyline;
                                        if (points != null) {
                                            List<com.google.maps.model.LatLng> coords = points.
                                                decodePath();
                                            for (com.google.maps.model.LatLng coord : coords) {
                                                path.add(new LatLng(coord.lat, coord.lng));
                                            }
                                        }
                                    }
                                }
                            }
                        }
                    }
                }
            } catch(Exception ex) {}
        }
        else{
            nearestChapter.setVisibility(View.GONE);
            nearestChapterDistance.setVisibility(View.GONE);
            highlightButton.setVisibility(View.GONE);
        }
    }

    public void showNearestBranchDetails(View view){
        mainCard.setVisibility(View.VISIBLE);
    }

    public void hideNearestBranchDetails(View view){
        mainCard.setVisibility(View.GONE);
    }

}
```

..12 News.java

```
package upmanila.bengeminuy.redcrossph;

import com.google.firebase.Timestamp;

public class News {

    private String id;
    private String type;
    private String title;
    private String description;
    private String imageURL;
    private Timestamp timestamp;

    public News(){
```

```
        id = "";
        type = "";
        title = "";
        description = "";
        imageURL = "";
    }

    public void setId(String id) {
        this.id = id;
    }

    public void setType(String type) {
        this.type = type;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public void setDescription(String description) {
        this.description = description;
    }

    public void setImageURL(String imageURL) {
        this.imageURL = imageURL;
    }

    public void setTimestamp(Timestamp timestamp) {
        this.timestamp = timestamp;
    }

    public String getId() {
        return id;
    }

    public String getType() {
        return type;
    }

    public String getTitle() {
        return title;
    }

    public String getDescription() {
        return description;
    }

    public String getImageURL() {
        return imageURL;
    }

    public Timestamp getTimestamp() {
        return timestamp;
    }

    @Override
    public boolean equals(Object obj) {
        News news = (News) obj;
        return id.compareTo(news.getId())==0;
    }
}
```

..13 NewsActivity.java

```
package upmanila.bengeminuy.redcrossph;

import android.content.Context;
import android.support.annotation.Nullable;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.view.View;
import android.widget.SearchView;

import com.google.firebase.FirebaseApp;
import com.google.firebase.firestore.DocumentChange;
import com.google.firebase.firestore.EventListener;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.FirebaseFirestoreException;
import com.google.firebase.firestore.QuerySnapshot;

import java.util.ArrayList;
import java.util.Collections;

public class NewsActivity extends AppCompatActivity {

    private Context context;
    private FirebaseFirestore database;
    private ArrayList<News> databaseNews, filteredNews;
    private RecyclerView newsRecyclerView;
```

```
private RecyclerView.Adapter newsRecyclerViewAdapter;
private SearchView newsSearchView;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_news);

    context = getApplicationContext();
    newsRecyclerView = findViewById(R.id.parentNewsRecyclerView);
    newsRecyclerView.setLayoutManager(new LinearLayoutManager(this));
    newsSearchView = findViewById(R.id.newsSearchView);
    newsSearchView.setIconifiedByDefault(false);

    FirebaseApp.initializeApp(context);
    database = FirebaseFirestore.getInstance();

    databaseNews = new ArrayList<>();
    filteredNews = new ArrayList<>();

    newsRecyclerViewAdapter = new NewsAdapter(filteredNews,this, getResources());
    newsRecyclerView.setAdapter(newsRecyclerViewAdapter);

    newsSearchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
        @Override
        public boolean onQueryTextSubmit(String query) {
            filterFilteredNews();
            return true;
        }

        @Override
        public boolean onQueryTextChange(String newText) {
            filterFilteredNews();
            return true;
        }
    });
}

@Override
protected void onStart() {
    super.onStart();

    newsSearchView.setIconified(false);
    newsSearchView.clearFocus();

    loadNews();
}

private void loadNews(){

    database.collection("notices").addSnapshotListener(new EventListener<QuerySnapshot>() {
        @Override
        public void onEvent(@Nullable QuerySnapshot querySnapshot,
                            @Nullable FirebaseFirestoreException e) {
            if (e != null) {
                return;
            }
            for (DocumentChange change : querySnapshot.getDocumentChanges()) {
                if (change.getType() == DocumentChange.Type.MODIFIED) {
                    News updatedNews = new News();
                    updatedNews.setId(change.getDocument().getId());
                    News editedNews = databaseNews.get(databaseNews.indexOf(updatedNews));
                    editedNews.setTimestamp(change.getDocument().getTimestamp("timestamp"));
                    editedNews.setTitle(change.getDocument().getString("title"));
                    editedNews.setType(change.getDocument().getString("type"));
                    editedNews.setImageURL(change.getDocument().getString("imageURL"));
                    editedNews.setDescription(change.getDocument().getString("details"));

                } else if (change.getType() == DocumentChange.Type.ADDED) {
                    News newNews = new News();
                    newNews.setId(change.getDocument().getId());
                    if(!databaseNews.contains(newNews)){
                        newNews.setTitle(change.getDocument().getString("title"));
                        newNews.setTimestamp(change.getDocument().getTimestamp("timestamp"));
                        newNews.setType(change.getDocument().getString("type"));
                        newNews.setImageURL(change.getDocument().getString("imageURL"));
                        newNews.setDescription(change.getDocument().getString("details"));
                        databaseNews.add(newNews);
                    }
                } else if (change.getType() == DocumentChange.Type.REMOVED) {
                    News updatedNews = new News();
                    updatedNews.setId(change.getDocument().getId());
                    databaseNews.remove(databaseNews.indexOf(updatedNews));
                }
            }
            filterFilteredNews();
        }
    });
}

private void filterFilteredNews(){
    filteredNews.clear();
```

```
        for(News currentNews : databaseNews){
            String query = newsSearchView.getQuery().toString().toLowerCase();
            if(currentNews.getTitle().toLowerCase().contains(query) || currentNews.getDescription
                ().toLowerCase().contains(query)){
                filteredNews.add(currentNews);
            }
        }
        Collections.reverse(filteredNews);
        newsRecyclerViewAdapter.notifyDataSetChanged();
    }

    public void newsPressBack(View view){
        finish();
    }

}
```

..14 NewsAdapter.java

```
package upmanila.bengeminuy.redcrossph;

import android.content.Context;
import android.content.res.Resources;
import android.graphics.Color;
import android.net.Uri;
import android.support.annotation.NonNull;
import android.support.v7.widget.RecyclerView;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;

import com.bumptech.glide.Glide;
import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.StorageReference;

import java.util.List;

public class NewsAdapter extends RecyclerView.Adapter<NewsAdapter.ViewHolder> {

    private List<News> newsModel;
    private Context context;
    private Resources resources;
    private FirebaseStorage storage;
    private StorageReference reference;

    public NewsAdapter(List<News> news, Context context, Resources resources) {
        this.newsModel = news;
        this.context = context;
        this.resources = resources;
        storage = FirebaseStorage.getInstance();
        reference = storage.getReference();
    }

    @NonNull
    @Override
    public NewsAdapter.ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        View v = LayoutInflater.from(parent.getContext())
                .inflate(R.layout.news_card, parent, false);
        return new ViewHolder(v);
    }

    @Override
    public void onBindViewHolder(@NonNull final NewsAdapter.ViewHolder holder, int position) {
        News news = newsModel.get(position);
        holder.tagText.setText(news.getType().toUpperCase());

        if(news.getType().compareToIgnoreCase("news")==0){
            holder.tagText2.setBackgroundColor(resources.getColor(R.color.newstag));
            holder.tagText.setBackgroundColor(resources.getColor(R.color.newstag));
            holder.dateText.setBackgroundColor(resources.getColor(R.color.newstag));
        }
        else if(news.getType().compareToIgnoreCase("announcement")==0){
            holder.tagText.setBackgroundColor(resources.getColor(R.color.announcetag));
            holder.dateText.setBackgroundColor(resources.getColor(R.color.announcetag));
            holder.tagText2.setBackgroundColor(resources.getColor(R.color.announcetag));
        }
        else if(news.getType().compareToIgnoreCase("promo")==0){
            holder.tagText.setBackgroundColor(resources.getColor(R.color.promotag));
            holder.dateText.setBackgroundColor(resources.getColor(R.color.promotag));
            holder.tagText2.setBackgroundColor(resources.getColor(R.color.promotag));
        }

        if(news.getImageURL().length() > 0){
            StorageReference imageReference = reference.child(news.getImageURL());

            Log.i("IMAGE", news.getImageURL());
```

```
            imageReference.getDownloadUrl().addOnSuccessListener(new OnSuccessListener<Uri>() {

                @Override
                public void onSuccess(Uri uri) {
                    String imageURL = uri.toString();
                    Glide.with(context).load(imageURL).into(holder.imageView);
                }

            }).addOnFailureListener(new OnFailureListener() {

                @Override
                public void onFailure(@NonNull Exception exception) {
                }

            });
        }

        holder.dateText.setText(news.getTimestamp().toDate().toString().replace(" GMT+08:00","").
            substring(4));
        holder.titleText.setText(news.getTitle().replace("\\n", "\n"));
        holder.detailText.setText(news.getDescription().replace("\\n", "\n"));
    }

    @Override
    public int getItemCount() {
        return newsModel.size();
    }

    @Override
    public int getItemViewType(int position) {
        return position;
    }

    public class ViewHolder extends RecyclerView.ViewHolder {

        public TextView tagText, tagText2;
        public TextView dateText;
        public ImageView imageView;
        public TextView titleText;
        public TextView detailText;

        public ViewHolder(View itemView) {
            super(itemView);

            dateText = itemView.findViewById(R.id.dateText);
            tagText2 = itemView.findViewById(R.id.newsTagText2);
            tagText = itemView.findViewById(R.id.newsTagText);
            imageView = itemView.findViewById(R.id.newsImage);
            titleText = itemView.findViewById(R.id.newsTitle);
            detailText = itemView.findViewById(R.id.newsDetail);
        }
    }

}
```

..15 PRCBranch.java

```
package upmanila.bengeminuy.redcrossph;

import com.google.firebase.firestore.GeoPoint;

import java.util.ArrayList;

class PRCBranch {

    private String id;
    private String branchName;
    private boolean apheresis;
    private boolean conventional;
    private String email;
    private ArrayList<String> fax;
    private ArrayList<String> telephone;
    private String address;
    private GeoPoint geoPoint;
    private boolean monday, tuesday, wednesday, thursday, friday, saturday, sunday;
    private String operatingHours;
    private Double currentUnits;
    private Double targetUnits;
    private Float userDistance;

    public PRCBranch() {
        id = "";
        branchName = "";
        apheresis = false;
        conventional = false;
        email = "";
        fax = new ArrayList<>();
        telephone = new ArrayList<>();
        address = "";
        geoPoint = new GeoPoint(-1.0, -1.0);
```

```
        monday = false;
        tuesday = false;
        wednesday = false;
        thursday = false;
        friday = false;
        saturday = false;
        sunday = false;
        operatingHours = "";
        currentUnits = 0.0;
        targetUnits = 1.0;
        userDistance = new Float(0);
    }

    @Override
    public boolean equals(Object obj) {
        PRCBranch comparedBranch = (PRCBranch) obj;
        return this.id.compareTo(comparedBranch.getId()) == 0;
    }

    public void setId(String id) {
        this.id = id;
    }

    public void setBranchName(String branchName) {
        this.branchName = branchName;
    }

    public void setApheresis(boolean apheresis) {
        this.apheresis = apheresis;
    }

    public void setConventional(boolean conventional) {
        this.conventional = conventional;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public void setFax(ArrayList<String> fax) {
        this.fax.addAll(fax);
    }

    public void setTelephone(ArrayList<String> telephone) {
        this.telephone.addAll(telephone);
    }

    public void setAddress(String address) {
        this.address = address;
    }

    public void setGeoPoint(GeoPoint geoPoint) {
        this.geoPoint = geoPoint;
    }

    public void setMonday(boolean monday) {
        this.monday = monday;
    }

    public void setTuesday(boolean tuesday) {
        this.tuesday = tuesday;
    }

    public void setWednesday(boolean wednesday) {
        this.wednesday = wednesday;
    }

    public void setThursday(boolean thursday) {
        this.thursday = thursday;
    }

    public void setFriday(boolean friday) {
        this.friday = friday;
    }

    public void setSaturday(boolean saturday) {
        this.saturday = saturday;
    }

    public void setSunday(boolean sunday) {
        this.sunday = sunday;
    }

    public void setOperatingHours(String operatingHours) {
        this.operatingHours = operatingHours;
    }

    public void setCurrentUnits(Double currentUnits) {
        this.currentUnits = currentUnits;
    }

    public void setTargetUnits(Double targetUnits) {
```

```
        this.targetUnits = targetUnits;
    }

    public void setUserDistance(Float userDistance) {
        this.userDistance = userDistance;
    }

    public String getId() {
        return id;
    }

    public String getBranchName() {
        return branchName;
    }

    public boolean getApheresis() {
        return apheresis;
    }

    public boolean getConventional() {
        return conventional;
    }

    public String getEmail() {
        return email;
    }

    public ArrayList<String> getFax() {
        return fax;
    }

    public ArrayList<String> getTelephone() {
        return telephone;
    }

    public String getAddress() {
        return address;
    }

    public GeoPoint getGeoPoint() {
        return geoPoint;
    }

    public boolean getMonday() {
        return monday;
    }

    public boolean getTuesday() {
        return tuesday;
    }

    public boolean getWednesday() {
        return wednesday;
    }

    public boolean getThursday() {
        return thursday;
    }

    public boolean getFriday() {
        return friday;
    }

    public boolean getSaturday() {
        return saturday;
    }

    public boolean getSunday() {
        return sunday;
    }

    public String getOperatingHours() {
        return operatingHours;
    }

    public Double getCurrentUnits() {
        return currentUnits;
    }

    public Double getTargetUnits() {
        return targetUnits;
    }

    public Float getUserDistance() {
        return userDistance;
    }
}
```

..16 PRCBranchComparator.java

```
package upmanila.bengeminuy.redcrossph;
```

```
import java.util.Comparator;

public class PRCBranchComparator implements Comparator<PRCBranch> {

    @Override
    public int compare(PRCBranch o1, PRCBranch o2) {
        return o1.getUserDistance().compareTo(o2.getUserDistance());
    }

}
```

..17 Question.java

```
package upmanila.bengeminuy.redcrossph;

import java.util.ArrayList;

public class Question {

    private String id;
    private String question;
    private String category;
    private ArrayList<String> choices;
    private ArrayList<String> responses;
    private ArrayList<String> alignments;
    private int currentChoice = -1;
    private int currentChoiceID = -1;

    public Question(){
        id = "";
        question = "";
        category = "";
        choices = new ArrayList<>();
        responses = new ArrayList<>();
        alignments = new ArrayList<>();
    }

    public void setCurrentChoiceID(int currentChoiceID) {
        this.currentChoiceID = currentChoiceID;
    }

    public void setCurrentChoice(int currentChoice) {
        this.currentChoice = currentChoice;
    }

    public void setId(String id) {
        this.id = id;
    }

    public void setQuestion(String question) {
        this.question = question;
    }

    public void setCategory(String category) {
        this.category = category;
    }

    public void setChoices(ArrayList<String> choices) {
        this.choices = choices;
    }

    public void setResponses(ArrayList<String> responses) {
        this.responses = responses;
    }

    public void setAlignments(ArrayList<String> alignments) {
        this.alignments = alignments;
    }

    public int getCurrentChoiceID() {
        return currentChoiceID;
    }

    public int getCurrentChoice() {
        return currentChoice;
    }

    public String getId() {
        return id;
    }

    public String getQuestion() {
        return question;
    }

    public String getCategory() {
        return category;
    }

    public ArrayList<String> getChoices() {
```

```
        return choices;
    }

    public ArrayList<String> getResponses() {
        return responses;
    }

    public ArrayList<String> getAlignments() {
        return alignments;
    }

}
```

..18 QuestionActivity.java

```
package upmanila.bengeminuy.redcrossph;

import android.content.Context;
import android.support.annotation.Nullable;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.support.v7.widget.LinearLayoutManager;
import android.support.v7.widget.RecyclerView;
import android.util.Log;
import android.view.View;
import android.widget.SearchView;
import android.widget.TextView;

import com.google.firebase.FirebaseApp;
import com.google.firebase.firestore.DocumentChange;
import com.google.firebase.firestore.EventListener;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.FirebaseFirestoreException;
import com.google.firebase.firestore.QuerySnapshot;

import java.util.ArrayList;

public class QuestionActivity extends AppCompatActivity {

    private Context context;
    private FirebaseFirestore database;
    private ArrayList<Question> databaseQuestion, filteredQuestion;
    private RecyclerView questionRecyclerView;
    private RecyclerView.Adapter questionRecyclerViewAdapter;
    private SearchView questionSearchView;
    private TextView remarkText, remarkTitle;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_question);

        context = getApplicationContext();
        questionRecyclerView = findViewById(R.id.parentQuestionRecyclerView);
        questionRecyclerView.setLayoutManager(new LinearLayoutManager(this));
        questionSearchView = findViewById(R.id.questionSearchView);
        questionSearchView.setIconifiedByDefault(false);
        remarkText = findViewById(R.id.remarkText);
        remarkTitle = findViewById(R.id.remarkTitle);

        FirebaseApp.initializeApp(context);
        database = FirebaseFirestore.getInstance();

        databaseQuestion = new ArrayList<>();
        filteredQuestion = new ArrayList<>();

        questionRecyclerViewAdapter = new QuestionAdapter(filteredQuestion,this, remarkText,
             remarkTitle);
        questionRecyclerView.setAdapter(questionRecyclerViewAdapter);

        questionSearchView.setOnQueryTextListener(new SearchView.OnQueryTextListener() {
            @Override
            public boolean onQueryTextSubmit(String query) {
                filterFilteredQuestions();
                return true;
            }

            @Override
            public boolean onQueryTextChange(String newText) {
                filterFilteredQuestions();
                return true;
            }
        });
    }

    @Override
    protected void onStart() {
        super.onStart();

        questionSearchView.setIconified(false);
        questionSearchView.clearFocus();
```

```
    loadQuestions();
}

private void loadQuestions(){

    database.collection("donorquestionnairequestions").addSnapshotListener(new EventListener<
        QuerySnapshot>() {
        @Override
        public void onEvent(@Nullable QuerySnapshot querySnapshot,
                            @Nullable FirebaseFirestoreException e) {
            if (e != null) {
                return;
            }
            for (DocumentChange change : querySnapshot.getDocumentChanges()) {
                if (change.getType() == DocumentChange.Type.MODIFIED) {
                    Question updatedQuestion = new Question();
                    updatedQuestion.setId(change.getDocument().getId());
                    Question editedQuestion = databaseQuestion.get(databaseQuestion.indexOf(
                        updatedQuestion));
                    editedQuestion.setQuestion(change.getDocument().getString("question"));
                    editedQuestion.setChoices((ArrayList<String>)change.getDocument().get("
                        choices"));
                    editedQuestion.setAlignments((ArrayList<String>)change.getDocument().get("
                        alignment"));
                    if(((ArrayList<String>)change.getDocument().get("response"))!= null){
                        editedQuestion.setResponses((ArrayList<String>)change.getDocument().
                            get("response"));
                    }
                    else if(((ArrayList<String>)change.getDocument().get("responses"))!= null)
                        {
                        editedQuestion.setResponses((ArrayList<String>)change.getDocument().
                            get("responses"));
                    }
                    editedQuestion.setCategory(change.getDocument().getString("category"));
                } else if (change.getType() == DocumentChange.Type.ADDED) {
                    Question newQuestion = new Question();
                    newQuestion.setId(change.getDocument().getId());
                    if(!databaseQuestion.contains(newQuestion)){
                        newQuestion.setQuestion(change.getDocument().getString("question"));
                        newQuestion.setChoices((ArrayList<String>)change.getDocument().get("
                            choices"));
                        newQuestion.setAlignments((ArrayList<String>)change.getDocument().get
                            ("alignment"));
                        if(((ArrayList<String>)change.getDocument().get("response"))!= null){
                            newQuestion.setResponses((ArrayList<String>)change.getDocument().
                                get("response"));
                        }
                        else if(((ArrayList<String>)change.getDocument().get("responses"))!=
                            null){
                            newQuestion.setResponses((ArrayList<String>)change.getDocument().
                                get("responses"));
                        }
                        newQuestion.setCategory(change.getDocument().getString("category"));
                        databaseQuestion.add(newQuestion);
                    }
                } else if (change.getType() == DocumentChange.Type.REMOVED) {
                    Question updatedQuestion = new Question();
                    updatedQuestion.setId(change.getDocument().getId());
                    int index = databaseQuestion.indexOf(updatedQuestion);
                    databaseQuestion.remove(index);
                }
            }
            Log.i("COMPLETION", "Finished Updating Data");
            filterFilteredQuestions();
        }
    });
}

private void filterFilteredQuestions(){
    filteredQuestion.clear();
    for(Question currentQuestion : databaseQuestion){
        String query = questionSearchView.getQuery().toString().toLowerCase();
        if(currentQuestion.getQuestion().toLowerCase().contains(query.toLowerCase())){
            filteredQuestion.add(currentQuestion);
        }
        else if(isFilterInChoices(currentQuestion, query)){
            filteredQuestion.add(currentQuestion);
        }
        else if(currentQuestion.getCategory().toLowerCase().contains(query.toLowerCase())){
            filteredQuestion.add(currentQuestion);
        }
    }
    questionRecyclerViewAdapter.notifyDataSetChanged();
}

private boolean isFilterInChoices(Question question, String query){

    for(String choice : question.getChoices()){
        if(choice.toLowerCase().contains(query.toLowerCase())){
            return true;
        }
```

```
        }
        return false;
    }

    public void onQuestionBack(View view){
        finish();
    }

}
```

..19 QuestionAdapter.java

```
package upmanila.bengeminuy.redcrossph;

import android.content.Context;
import android.support.annotation.NonNull;
import android.support.v7.widget.RecyclerView;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.TextView;

import org.w3c.dom.Text;

import java.util.List;

public class QuestionAdapter extends RecyclerView.Adapter<QuestionAdapter.ViewHolder> {

    private List<Question> questionModel;
    private TextView remarkText, remarkTitle;
    private Context context;

    public QuestionAdapter(List<Question> questions, Context context, TextView remarkText,
     TextView remarkTitle){
        this.questionModel = questions;
        this.context = context;
        this.remarkText = remarkText;
        this.remarkTitle = remarkTitle;
    }

    @NonNull
    @Override
    public QuestionAdapter.ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType)
     {
        View v = LayoutInflater.from(parent.getContext()).inflate(R.layout.question_card, parent,
         false);
        return new QuestionAdapter.ViewHolder(v);
    }

    @Override
    public void onBindViewHolder(@NonNull QuestionAdapter.ViewHolder holder, int position) {
        final Question question = questionModel.get(position);

        holder.responseText.setVisibility(View.GONE);
        holder.categoryText.setText(question.getCategory());
        holder.questionText.setText(question.getQuestion().replace("\\n", "\n"));

        holder.choice1Text.setText(question.getChoices().get(0).replace("\\n", "\n"));
        holder.choice2Text.setText(question.getChoices().get(1).replace("\\n", "\n"));

        if(question.getChoices().size() >= 3){
            holder.choice3Text.setText(question.getChoices().get(2).replace("\\n", "\n"));
        }
        else{
            holder.choice3Text.setVisibility(View.GONE);
        }

        if(question.getChoices().size() >= 4){
            holder.choice4Text.setText(question.getChoices().get(3).replace("\\n", "\n"));
        }
        else{
            holder.choice4Text.setVisibility(View.GONE);
        }

        final QuestionAdapter.ViewHolder viewholder = holder;
        final RadioGroup choices = holder.choices;

        if(question.getCurrentChoice()!=-1){
            choices.check(question.getCurrentChoiceID());
            viewholder.responseText.setVisibility(View.VISIBLE);
            viewholder.responseText.setText("Response:\n\n" + question.getResponses().get(question
                 .getCurrentChoice()).replace("\\n", "\n"));
            if(question.getAlignments().get(question.getCurrentChoice()).compareToIgnoreCase("
                 negative")==0){
                viewholder.responseText.setBackgroundColor(context.getResources().getColor(R.color
                     .newstag));
            }
            else if(question.getAlignments().get(question.getCurrentChoice()).compareToIgnoreCase
                 ("neutral")==0){
```

```
                viewholder.responseText.setBackgroundColor(context.getResources().getColor(R.color
                    .promotag));
            }
            else if(question.getAlignments().get(question.getCurrentChoice()).compareToIgnoreCase
                ("positive")==0){
                viewholder.responseText.setBackgroundColor(context.getResources().getColor(R.color
                    .announcetag));
            }
        }

        choices.setOnCheckedChangeListener(new RadioGroup.OnCheckedChangeListener() {
            @Override
            public void onCheckedChanged(RadioGroup group, int checkedId) {
                question.setCurrentChoiceID(checkedId);
                View clickedChoice = choices.findViewById(checkedId);
                int index = choices.indexOfChild(clickedChoice);

                if(index >= 0){
                    question.setCurrentChoice(index);
                    viewholder.responseText.setVisibility(View.VISIBLE);
                    viewholder.responseText.setText("Response:\n\n" + question.getResponses().get(
                        index).replace("\\n", "\n"));
                    if(question.getAlignments().get(index).compareToIgnoreCase("negative")==0){
                        viewholder.responseText.setBackgroundColor(context.getResources().getColor
                            (R.color.newstag));
                    }
                    else if(question.getAlignments().get(index).compareToIgnoreCase("neutral")==0)
                        {
                        viewholder.responseText.setBackgroundColor(context.getResources().getColor
                            (R.color.promotag));
                    }
                    else if(question.getAlignments().get(index).compareToIgnoreCase("positive")
                        ==0){
                        viewholder.responseText.setBackgroundColor(context.getResources().getColor
                            (R.color.announcetag));
                    }
                }
                updateNegativeRemarks();
            }
        });

    }

    @Override
    public int getItemCount() {
        return questionModel.size();
    }

    @Override
    public int getItemViewType(int position) {
        return position;
    }

    public class ViewHolder extends RecyclerView.ViewHolder{

        public TextView categoryText;
        public TextView questionText;
        public RadioGroup choices;
        public RadioButton choice1Text;
        public RadioButton choice2Text;
        public RadioButton choice3Text;
        public RadioButton choice4Text;
        public TextView responseText;

        public ViewHolder(View itemView) {
            super(itemView);

            choices = itemView.findViewById(R.id.choices);
            categoryText = itemView.findViewById(R.id.categoryText);
            questionText = itemView.findViewById(R.id.questionCardTitle);
            choice1Text = itemView.findViewById(R.id.choice1Text);
            choice2Text = itemView.findViewById(R.id.choice2Text);
            choice3Text = itemView.findViewById(R.id.choice3Text);
            choice4Text = itemView.findViewById(R.id.choice4Text);
            responseText = itemView.findViewById(R.id.responseText);

        }
    }

    private void updateNegativeRemarks(){
        int negativeChoices = 0;
        for(Question currentQuestion : questionModel){
            if(currentQuestion.getCurrentChoice()!=-1) {
                if (currentQuestion.getAlignments().get(currentQuestion.getCurrentChoice()).
                    toLowerCase().compareTo("negative") == 0) {
                    negativeChoices++;
                }
            }
        }
        if(negativeChoices>0){
            remarkTitle.setBackgroundColor(context.getResources().getColor(R.color.newstag));
            remarkTitle.setTextColor(context.getResources().getColor(R.color.
```

```
                cardview_light_background));
            remarkText.setBackgroundColor(context.getResources().getColor(R.color.newstag));
            remarkText.setTextColor(context.getResources().getColor(R.color.
                cardview_light_background));
            remarkText.setText("You have answers that disqualify you as a blood donor. Kindly
                resolve these first before trying to donate.");
        }
        else{
            remarkTitle.setBackgroundColor(context.getResources().getColor(R.color.announcetag));
            remarkTitle.setTextColor(context.getResources().getColor(R.color.
                cardview_light_background));
            remarkText.setBackgroundColor(context.getResources().getColor(R.color.announcetag));
            remarkText.setTextColor(context.getResources().getColor(R.color.
                cardview_light_background));
            remarkText.setText("You currently have no attributes that disqualify you as a blood
                donor. You may try donating at the nearest Philippine Red Cross Chapter");
        }
    }

}
```

..20 SplashActivity.java

```
package upmanila.bengeminuy.redcrossph;

import android.Manifest;
import android.app.Activity;
import android.app.ActivityOptions;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.SharedPreferences;
import android.content.pm.PackageManager;
import android.os.AsyncTask;
import android.support.annotation.NonNull;
import android.support.v4.app.ActivityCompat;
import android.support.v7.app.AlertDialog;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.transition.Explode;
import android.view.animation.AlphaAnimation;
import android.view.animation.Animation;
import android.widget.ImageView;
import android.widget.TextView;

import java.net.InetSocketAddress;
import java.net.Socket;
import java.net.SocketAddress;

public class SplashActivity extends AppCompatActivity {

    TextView loadingTextView;
    ImageView logoImageView;
    private static final int MY_PERMISSIONS_REQUEST_FINE_LOCATION = 0;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_splash);

        //Link UI Objects to Layout Configuration
        loadingTextView = findViewById(R.id.loadingTextView);
        logoImageView = findViewById(R.id.logoSplashImageView);
    }

    @Override
    protected void onStart() {
        super.onStart();

        createAnimation();

        SharedPreferences flagPrefs = getSharedPreferences("upmanila.bengeminuy.redcrossph.flags",
            Context.MODE_PRIVATE);
        if(flagPrefs.getBoolean("remindedLocationPermission", false)){
            connectToInternet();
        }
        else{
            getPermissions();
        }
    }

    private void createAnimation(){
        final Animation logoFadeInAnimation = new AlphaAnimation(0.0f, 1.0f);
        logoFadeInAnimation.setDuration(3000);
        logoImageView.startAnimation(logoFadeInAnimation);

        final Animation fadeInAnimation = new AlphaAnimation(0.0f, 1.0f);
        fadeInAnimation.setDuration(3500);
        final Animation fadeOutAnimation = new AlphaAnimation(1.0f, 0.0f);
        fadeOutAnimation.setDuration(3500);
        fadeInAnimation.setAnimationListener(new Animation.AnimationListener() {
```

```
        @Override
        public void onAnimationStart(Animation animation) {
        }

        @Override
        public void onAnimationEnd(Animation animation) {
            loadingTextView.startAnimation(fadeOutAnimation);
        }

        @Override
        public void onAnimationRepeat(Animation animation) {
        }
    });
    fadeOutAnimation.setAnimationListener(new Animation.AnimationListener() {
        @Override
        public void onAnimationStart(Animation animation) {
        }

        @Override
        public void onAnimationEnd(Animation animation) {
            loadingTextView.startAnimation(fadeInAnimation);
        }

        @Override
        public void onAnimationRepeat(Animation animation) {
        }
    });
    loadingTextView.startAnimation(fadeInAnimation);
}

private void getPermissions(){
    ActivityCompat.requestPermissions(this, new String[]{Manifest.permission.
        ACCESS_FINE_LOCATION}, MY_PERMISSIONS_REQUEST_FINE_LOCATION);
}

@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String permissions[],
    @NonNull int[] grantResults) {
    switch (requestCode) {
        case MY_PERMISSIONS_REQUEST_FINE_LOCATION: {
            if (grantResults.length > 0 && grantResults[0] == PackageManager.
                PERMISSION_GRANTED) {
                //Permission Granted
                connectToInternet();
            }
            else {
                SharedPreferences flagPrefs = getSharedPreferences("upmanila.bengeminuy.
                    redcrossph.flags", Context.MODE_PRIVATE);
                if(flagPrefs.getBoolean("remindedLocationPermission", false)){
                    connectToInternet();
                }
                else{
                    requireLocationPermission();
                }
            }
        }
    }
}

private void requireLocationPermission(){
    final Activity splashActivity = this;
    AlertDialog.Builder builder = new AlertDialog.Builder(this);
    builder.setMessage("You need to allow the app to get your location to access the main
        features of the app.").setTitle("Warning");
    builder.setPositiveButton("I Understand", new DialogInterface.OnClickListener() {
        public void onClick(DialogInterface dialog, int id) {
            ActivityCompat.requestPermissions(splashActivity, new String[]{Manifest.permission
                .ACCESS_FINE_LOCATION}, MY_PERMISSIONS_REQUEST_FINE_LOCATION);
        }
    });
    builder.setNegativeButton("I Refuse.", new DialogInterface.OnClickListener() {
        public void onClick(DialogInterface dialog, int id) {
            SharedPreferences flagPrefs = getSharedPreferences("upmanila.bengeminuy.redcrossph
                .flags", Context.MODE_PRIVATE);
            SharedPreferences.Editor editor = flagPrefs.edit();
            editor.putBoolean("remindedLocationPermission", true);
            editor.apply();
            connectToInternet();
        }
    });
    AlertDialog dialog = builder.create();
    dialog.show();
}

private void connectToInternet(){
    loadingTextView.setText(R.string.TryInternetConnection);
    if(!hasInternetConnection()){
        alertNoInternet();
    }
    else{
        loadingTextView.setText(R.string.LoadResources);
```

```
            Intent i = new Intent(getApplicationContext(), MainActivity.class);
            startActivity(i);
            finish();
        }
    }

    private Boolean hasInternetConnection(){
        CheckInternetTask task = new CheckInternetTask();
        task.execute();
        try {
            return task.get();
        } catch (Exception e) {
            e.printStackTrace();
        }
        return false;
    }

    public void alertNoInternet(){
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setTitle("Warning");
        builder.setMessage("You have no internet connection, some of the details in the app won't
             update unless you have internet connection.");
        builder.setPositiveButton("OK", new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                loadingTextView.setText(R.string.LoadResources);

                Intent i = new Intent(getApplicationContext(), MainActivity.class);
                startActivity(i);
                finish();
            }
        });
        AlertDialog alertDialog = builder.create();
        alertDialog.show();
    }

    protected static class CheckInternetTask extends AsyncTask<Void,Void,Boolean> {

        @Override
        protected Boolean doInBackground(Void... voids) {
            try{
                int timeout = 3000;
                Socket sock = new Socket();
                SocketAddress socketAddress = new InetSocketAddress("8.8.8.8",53);
                sock.connect(socketAddress,timeout);
                sock.close();
                return true;
            }
            catch (Exception e){
                return false;
            }
        }

    }

}
```

..21 WaveHelper.java

```
package upmanila.bengeminuy.redcrossph;

import android.animation.Animator;
import android.animation.AnimatorSet;
import android.animation.ObjectAnimator;
import android.animation.ValueAnimator;
import android.view.animation.DecelerateInterpolator;
import android.view.animation.LinearInterpolator;

import com.gelitenight.waveview.library.WaveView;

import java.util.ArrayList;
import java.util.List;

public class WaveHelper {
    private WaveView mWaveView;

    private AnimatorSet mAnimatorSet;

    public WaveHelper(WaveView waveView) {
        mWaveView = waveView;
        initAnimation();
    }

    public void start() {
        mWaveView.setShowWave(true);
        if (mAnimatorSet != null) {
            mAnimatorSet.start();
        }
    }

    private void initAnimation() {
```

```
        List<Animator> animators = new ArrayList<>();

        ObjectAnimator waveShiftAnim = ObjectAnimator.ofFloat(
                mWaveView, "waveShiftRatio", 0f, 1f);
        waveShiftAnim.setRepeatCount(ValueAnimator.INFINITE);
        waveShiftAnim.setDuration(1000);
        waveShiftAnim.setInterpolator(new LinearInterpolator());
        animators.add(waveShiftAnim);

        ObjectAnimator waterLevelAnim = ObjectAnimator.ofFloat(
                mWaveView, "waterLevelRatio", 0f, 0.5f);
        waterLevelAnim.setDuration(10000);
        waterLevelAnim.setInterpolator(new DecelerateInterpolator());
        animators.add(waterLevelAnim);

        ObjectAnimator amplitudeAnim = ObjectAnimator.ofFloat(
                mWaveView, "amplitudeRatio", 0.0001f, 0.05f);
        amplitudeAnim.setRepeatCount(ValueAnimator.INFINITE);
        amplitudeAnim.setRepeatMode(ValueAnimator.REVERSE);
        amplitudeAnim.setDuration(5000);
        amplitudeAnim.setInterpolator(new LinearInterpolator());
        animators.add(amplitudeAnim);

        mAnimatorSet = new AnimatorSet();
        mAnimatorSet.playTogether(animators);
    }

    public void cancel() {
        if (mAnimatorSet != null) {
//            mAnimatorSet.cancel();
            mAnimatorSet.end();
        }
    }
}
```

XI. Acknowledgement

Thank you, my professors who have guided me from beginning until end. Thank you, my blockmates who have been there to support me during dire times. Thank you, my parents who have provided everything I needed to finish this. I would not have been able to finish the SP if it were not for all of you people.