Razakaar

Amna Islam & Rehan Ahmad

Session: 2013 – 2017

Roll No. 388-BSCS-13, 436-BSCS-13

Department of Computer Sciences

GC UNIVERSITY LAHORE

Razakaar

Submitted to GC University Lahore
in partial fulfillment of the requirements
for the award of degree of

Bachelors

IN

Computer Science

By

Amna Islam and Rehan Ahmad

SESSION : 2013 - 2017

Roll No. 388-BSCS-13, 436-BSCS-13

DEPARTMENT OF COMPUTER SCIENCE

GC UNIVERSITY LAHORE

DECLARATION

We, Ms. Amna Islam, Roll No. 388-BSCS-2013 and Mr. Rehan Ahmad, Roll No. 436-BSCS-2013, students of Government College University Lahore. In the subject of session 2013-2017, hereby declare that the matter printed in the thesis titled “Razakaar” is our own work and has not been printed, published and submitted as research work, thesis or publication in any form in any University, Research Institution etc in Pakistan or abroad.

Dated: ____________

____

Signatures of Deponent

RESEARCH COMPLETION CERTIFICATE

Certified that the research work contained in this thesis titled “Razakaar” has been carried out and completed by Ms. Amna Islam, Roll No. 388-BSCS-13 and Mr. Rehan Ahmad Roll No. 436-BSCS-13 under my supervision.

Date

Supervisor Signature
Yahya Khurram

Submitted Through

Dr. Syed Asad Raza Kazmi
Incharge
Department of Computer Science
GC University Lahore.

Controller of Examinations
GC University Lahore.

Abstract

Razakaar is a humanitarian technology based android application which helps regulate our extra food in particular and anything extra in our house, in general. The application is developed with complete solution from donation to its delivery to deserving through technology. The delivery of donation is made possible with the idea of volunteer service. There are two different application for donor and volunteer and both applications works in integration make the project successful.

Donor donate through Razakaar and Volunteer collects through Razakaar Volunteer. A list of Needy is provided by the admin. Volunteer can also add needy. Volunteer deliver the collected donation the Needy.

Google Map and its APIs are used for location. The development has been done in android studio. Adobe Illustrator is used for logo and UI.

The ultimate purpose of this project is to achieve sustainable development goal: Zero Hunger. Its vision is to practice technology for humanity. Its immediate goal is to introduce community services in society.

Contents

Chapter 1

INTRODUCTION TO PROJECT

1.1. Introduction

Razakaar is an android application based on humanitarian technology, which helps preventing food wastage by collecting extra food donation from donor and delivers to needy through volunteer service. There are two separate applications for donor and volunteer. For both applications, user signs up first.

After logging in, a map is displayed. On the map, volunteer will receive the location of donors around and donors will be shown the locations of volunteers around.

The donor donates extra food while filling out some basic information about it such as time to pick up donation and amount of food. The application generates alerts for volunteers and in the volunteer's application, location of donation is highlighted. The volunteer decides to collect the donation according to the feasible donor location on their route and confirms it. Donor gets the alert about a certain volunteer being on their way. The volunteer then collects the donation.

The collected donation is handed over to the specific places in record such as NGOs, shelter homes and other places. The information of these places is provided by the admin after complete verification. Volunteer can also forward request to add a certain needy and admin approves the request after reviewing it. Volunteer choses any place of delivery from the provided list according to his route of travelling. When the donation is successfully dropped to the delivery place. Donor gets an alert of successful delivery. The record is updated in each individual profile of donor and volunteer as successful donation and volunteer services record, respectively.

Razakaar cover other types of donation as well which includes:

1. Clothing
2. Footwear
3. Books

Currently, the focus is on the food in the context of technical management, practical implementation in the society and the generalized stability of the project. Whereas, the product we have developed is fully functional for other types of donation as well.

1.2. Objectives:

Razakaar is developed to implement the social and moral responsibility of donation through the means of technology. The idea is “to prevent the extra being wasted”. The main purpose is to introduce the technology for solving different social problems. It is a step to achieve one of the United Nation’s Sustainable Development Goals i.e. Zero Hunger. It is also an introduction to appreciate and improve the involvement of people of society in volunteer services and using their energies and resources for the better purposes.

1.3. Problem Statement:

A huge amount of food goes wasted daily. A number of people go to sleep hungry. Many people wish to help hungry or needy people but does not have time for community service. Many people wish to help needy people but does not have resources. To cater all these problems, Razakaar project has been developed.

1.4. Project Scope:

Razakaar is deployed on android platform, initially. There can also be a web version as well as an ios version, later on.

Razakaar is a project, which is designed for the global impact. Therefore, its scope is considerably broader. Major stakeholders include Government, Common Man, NGOs, Non-Profit Organizations, Social workers, Commercial Kitchens and Corporate Sector having Kitchens.

In future, popular food chains can be partnered with to initiate incentive schemes for volunteer as appreciation.

1.5. Constraints:

To use this application, smart phone (IPhone, Android) is the first requirement. Internet connectivity is essentially a second requirement.

1.6. Tools and Technology:

For the development of Razakaar Project, following tools and technologies have been used:

- Android Studio
- Google Maps
- Google APIs
- Adobe Illustrator
- Adobe Photoshop
- XML
- Java
- PHP

Chapter 2

LITERATURE REVIEW

2.1. Existing Products:

There are a few application, which have been developed for same purpose of saving extra food. However, the approach and strategy used to achieve the purpose are completely different or not at all implemented. These applications are mainly producing alerts or providing data of needy and donors. The scope of these applications are extremely limited.

Existing Food Saving applications are:

- Zero Percent
- Food Cowboy
- Pair up
- Waste No Food

2.2. Comparison and Analysis:

Each application share same purpose whereas play differently to solve the problem. One similarity between these applications is that none of these provides technology based solution to deliver the extra food. The maximum effort done on their part is the information provided about the needy and donor for extra food. Food is delivered to needy by social workers or government, physically by contacting the donor and needy on their own. Contrary to this, our application not only provides a platform to donate food, it also gives a solution based on technology to deliver extra food through volunteer services. Moreover, it includes more agents of society to be an active part according to the feasibility and increase a moral sense of responsibility in society.

Chapter 3

REQUIREMENT SPECIFICATIONS

3.1. Purpose:

Gathering requirements is the most important part for a software developer in order to understand not only the correct idea of demand but also to analyze the procedure of development according to the requirements. In this chapter, all the function and non-functional requirements are presented which makes it possible to understand and develop Razakaar and Razakaar Volunteer.

3.2. Stake Holders:

Stake Holders are said to be the people or agents involved in the execution of a software project. This project has its own stakeholders, which are as following:

- Donor: Any entity who wish to donate their extra food, clothing or books will sign up as donor in Razakaar application.
- Volunteer: Any entity who wish to perform volunteer service of collecting donation and delivering them to the needy will sign up as Volunteer in Razakaar Volunteer application.
- Needy: Admin will provide a list of places where donation can be delivered.
- Admin: The owner of this project at server side.

3.3. Functional Requirements:

3.3.1. Login to Razakaar using Email and Phone Number:

Req. No.	Functional Requirements
3.3.1-01	The system shall allow the donor to log in with correct email and phone number.
3.3.1-02	If the email or phone number are correct to authenticate him or her then the system should grant access to game play.

Table 1

3.3.2. Sign Up to Razakaar:

Req. No.	**Functional Requirements**
3.3.2 -01	The system shall allow the new user.
3.3.2-02	If all the fields are correctly filled in the form, the system should register user.

Table 2

3.3.3. Donate:

Req. No.	**Functional Requirements**
3.3.3-01	User should install Razakaar application.
3.3.3-02	If the user has signed up at Razakaar, they should be allow to make a donation.
3.3.3-03	If the registered donor has made a donation, system should save it successfully in the database
3.3.3-04a	If the donation has been made successfully, an alert should be generated for volunteer.
3.3.3-04b	If the donation has been mage successfully, location of donation should be highlighted in the map.

Table 3

3.3.4. Collect:

Req. No.	Functional Requirements
3.3.4-01	The user should download Razakaar Volunteer application.
3.3.4-02	If the user has signed up at Razakaar Volunteer, the system should allow them to collect donation.
3.3.4-03	If the donation has been made, the system should allow Volunteer to collect.
3.3.4-04	If the Volunteer presses Collect button, an alert should be generated for the donor.
3.3.4.-05	If the Volunteer presses the Collect button, they should collect donation as decided.
3.3.4.-06	If the donation has been collected by Volunteer, it should be delivered at Needy.
3.3.4.-07	If the donation has been delivered, an alert should be generated for donor.
3.3.4.-08	If the donation has been delivered, the count in the history should be updated.

Table 4

3.3.5. Add Needy:

Req. No.	Functional Requirements
3.3.5-01	The user should download Razakaar Volunteer application.
3.3.5-02	If the user has signed up at Razakaar Volunteer, the system should allow them to Add Needy.
3.3.5-03	If the Needy is already in the list, the system should inform volunteer.
3.3.5-04	If the Needy is new, admin should approve the information and add to list.

Table 5

3.3.6. History:

Req. No.	Functional Requirements
3.3.6-01	If the user has performed some services, the system should show the count of services they have performed.

Table 6

3.3.7. Settings:

Req. No.	Functional Requirements
3.3.7-01	If the user needs to change some information, system should allow.
3.3.7-02	If the user change any information in his profile, the databases should be updated.

Table 7

3.4. Non-Functional Requirements

3.4.1. Performance:

Req. No.	Non-Functional Requirements
3.2.1-01	Average load time of the location must be less than 10 seconds.
3.2.1-02	Average delivery time taken by volunteer should not be more than 1 hour for food.
3.2.1-03	Average admin response time should not be greater than 24 hours.

Table 8

3.4.2. Defects-Maintenance:

Req. No.	Non-Functional Requirements
3.2.2-01	Post Release defects of the system must not exceed 1 or 2 critical bug per month.
3.2.2-02	Post Release bugs fixing should not take more than 8-10 hours.

Table 9

3.4.3. Documentation:

Req. No.	Non-Functional Requirements
3.2.3-01	Documentation must be complete in providing information about each and every module and functionality provided by the system to help the developers and interested stakeholders.
3.2.3-02	The documentation must be easily accessible to the developers or field experts if some problem arise to review the details and to find out errors/bugs.

3.2.3-03	The documentation must be written using minimal complex technical terms, any technical terms used must be additionally defined at the end or in the start of the document.
3.2.3-04	The code must be readable and well-structured for future use.

Table 10

3.4.4. Efficiency:

Req. No.	Non-Functional Requirements
3.2.4-01	The system should utilize minimum resources of any android phone or web browser for the processing of data or to access data.

Table 11

3.4.5. Maintainability:

Req. No.	Non-Functional Requirements
3.2.5-01	The system should be changeable if some module needs to be modified without effecting functionality of other modules, for this the system must have the concept of modularity.
3.2.5-02	The system should be open for reusability and testability to remove any bugs or errors.
3.2.5-03	The system should be extendable (i.e. adding new functionality in the system) without effecting the functionality and performance of rest of the system.

Table 12

3.4.6. Availability:

Req. No.	Non-Functional Requirements
3.2.6-01	The system must be available to use and to test 24/7.

Table 13

Chapter 4

PROJECT DESIGN

4.1. Methodology:

The methodology selected to develop Razakaar is Agile Modelling. The reason to pursue this process model is that the project in planning phase could not hold the concept completely. It was essential to produce small modules with prototypes to get the endorsement from customer. Therefore, Agile Modelling was selected to adapt change incrementally.

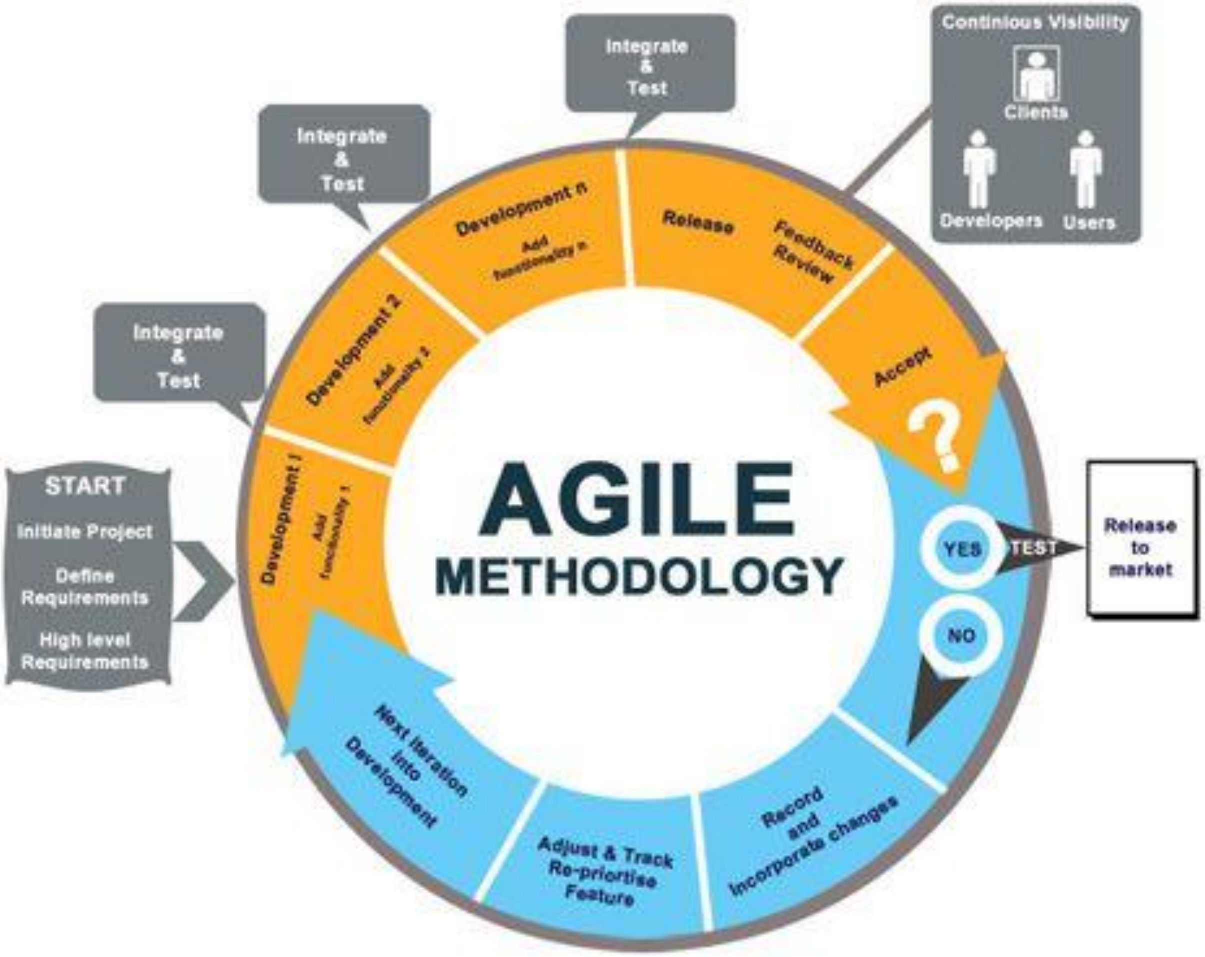


Figure 1

4.2. Design Use Case

4.2.1. Use Case for Installing Application Razakaar:

Primary Actors	Donor and Volunteer
Goal in context	Install the application on your device.
Preconditions:	Application has been programmed for registration of donor.
Trigger:	The user of application chooses to register as a donor or a volunteer.
Scenario:	1 Donor: selects installing of application. 2. Donor: After installation, will open the application.
Exceptions	1. Internet connection is not available. 2. You do not own Android Phone.
Frequency of use	Multiple times per day
Channel to actor:	Via device connected with internet
Secondary actors	Database, Buttons
Channels to secondary actors:	Touch Screen
Post Condition	User View Login Screen

Table 14

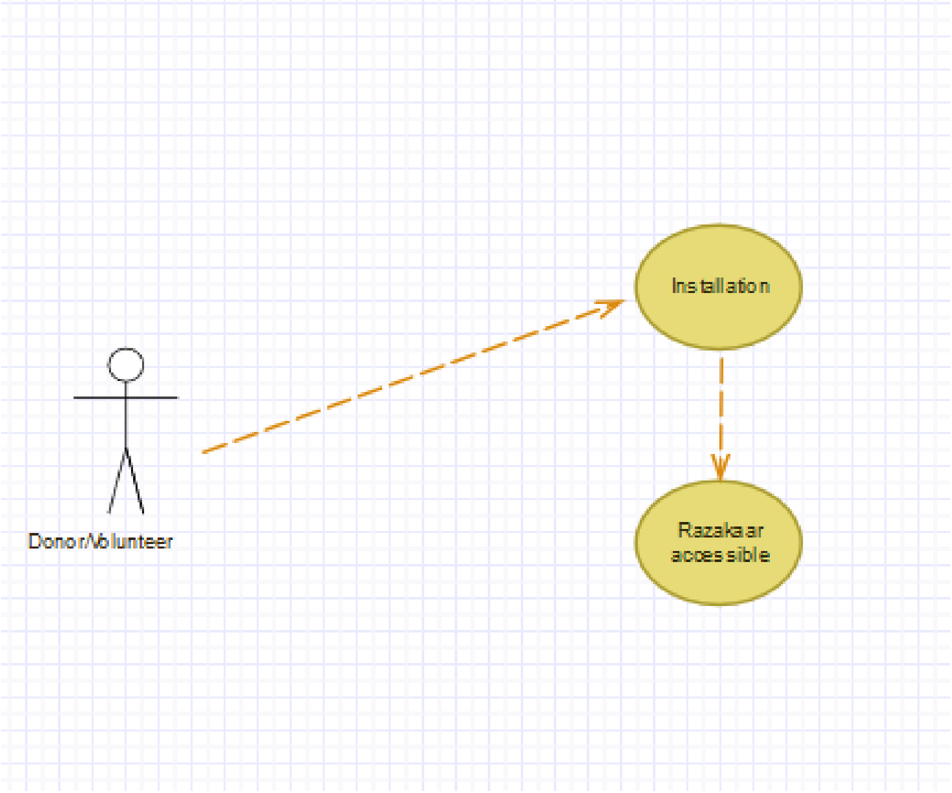


Figure 2

4.2.2. Use Case for Login:

Primary Actor	Donor and Volunteer
Goal in Context	Logging in to the registered account
Preconditions:	User has been registered
Scenario:	1. Donor/Volunteer: Enter Email. 2. Donor/Volunteer: Enter Mobile number.
Exceptions:	1. Internet connection is not available. 2. User is not registered.
Frequency of use	Multiple times per day
Channel to actor:	Via device connected with internet.
Secondary actors	Database, Buttons
Channels to secondary actors	Touch Screen
Post Condition	User View Home Page

Table 15

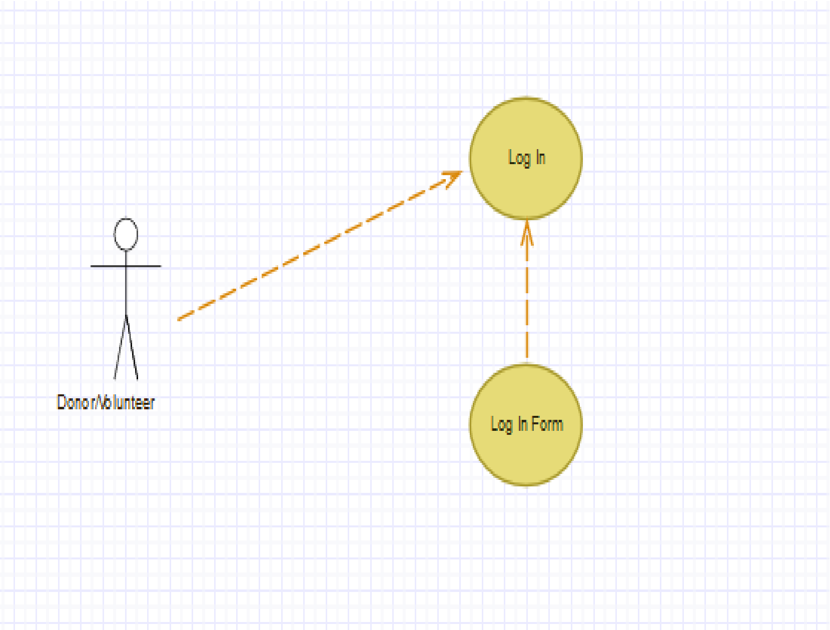


Figure 3

4.2.3. Use Case for Sign Up (Donor):

Primary actor	Donor
Goal in context	To register as Razakaar Donor
Preconditions:	Application has been programmed for registration of donor. User is new.
Trigger:	The user of application chooses to register as a donor
Scenario:	Donor: Press Button Sign Up on Login Page Donor: Fill out information on Signup form. Donor: Press Button Submit.
Exceptions:	1. Internet connection is not available. 2. User is already registered.
Frequency of use:	Multiple times per day.

Channel to actor	Via device connected with internet.
Secondary actors	Database, Buttons
Channels to secondary actors	Touch Screen
Post Condition	User View Home Page

Table 16

4.2.4. Use Case for Sign Up (Volunteer)

Primary actor	Volunteer
Goal in context	To register as Razakaar Volunteer
Preconditions	Application has been programmed for registration of Volunteer. User is new
Trigger	The user of application chooses to register as a Volunteer.
Scenario	1. Volunteer: Press Button Sign Up on Login Page. 2. Volunteer: Fill out information on Signup form. 3. Volunteer: Press Button Submit.
Exceptions	1. Internet connection is not available. 2. User is already registered.
Frequency of use:	Multiple times per day.
Channel to actor	Via device connected with internet.
Secondary actors	Database, Buttons
Channels to secondary actors	Touch Screen

Post Condition	User View Home Page

Table 17

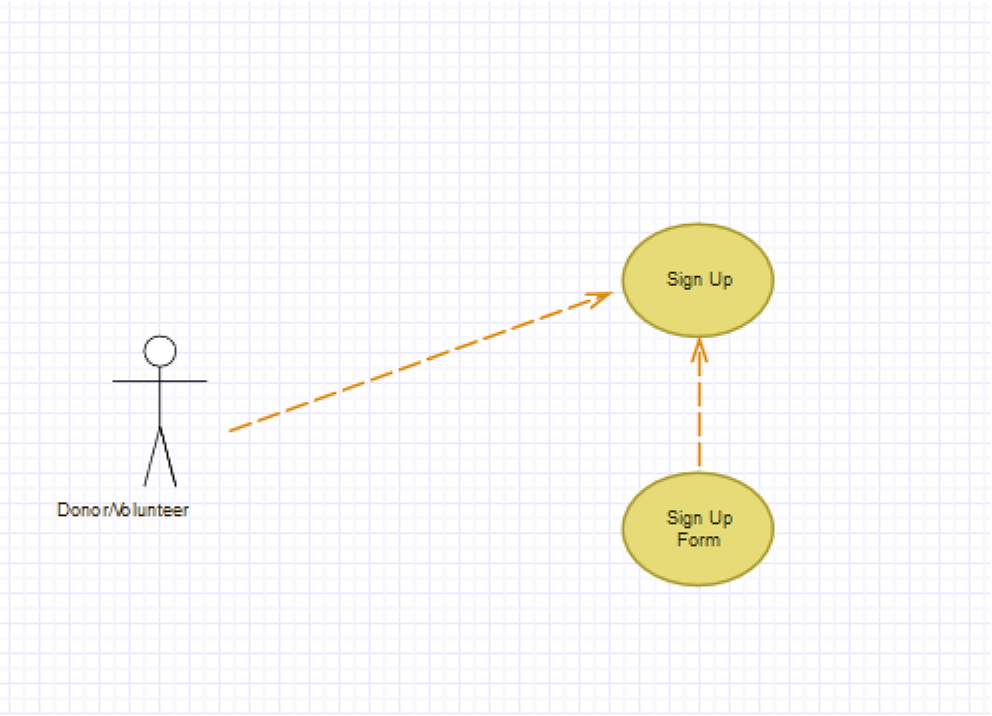


Figure 4

4.2.5. Use Case for Donate:

Primary actor	Donor
Goal in context	To make a Donation.
Preconditions:	Application is ready to make a Donation.
Scenario:	1. Donor: Press Dashboard button and selects type of Donation. 2. Donor: Fill out information on Donation form. 3. Donor: Press Button Submit.
Exceptions:	1. Internet connection is not available. 2. Donor has changed their mind.
Frequency of use	Multiple times per day.

Channel to actor	Via device connected with internet
Secondary actors	Database, Buttons, Forms
Channels to secondary actors	Touch Screen
Post Condition	User view highlighted location where donation has been made

Table 18

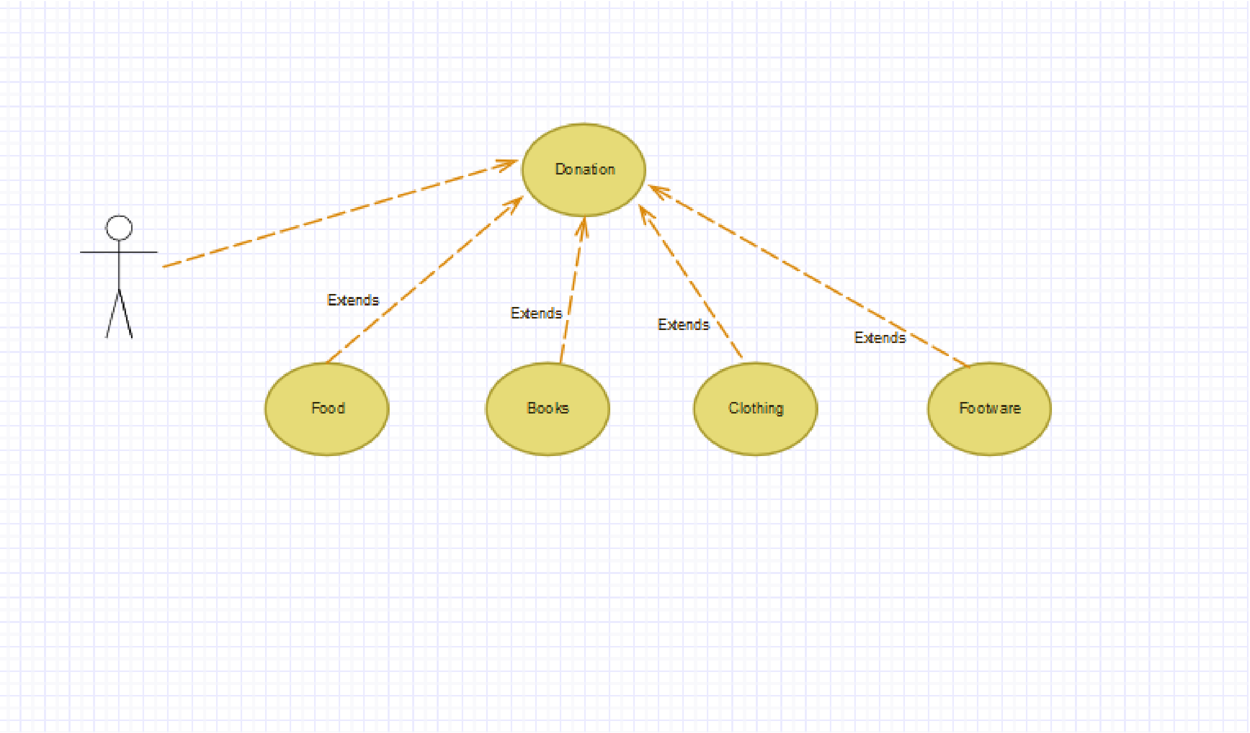


Figure 5

4.2.6. Use Case for Collect

Primary actor:	Volunteer
Goal in context	To collect donation and deliver it.
Preconditions:	A donation has been made. Volunteer received alert.
Scenario:	1. Volunteer: Checks map. Press highlighted donation. 2. Volunteer: Understand information and location.

	3. Volunteer: Checks needy near destination. 4. Volunteer: Press Confirm button. 5. Volunteer: Collect Donation and deliver it to the needy from the list.
Exceptions:	1. Internet connection is not available. 2. Donation has not been made.
Frequency of use	Multiple times per day.
Channel to actor	Via device connected with internet.
Secondary actors:	Database, Buttons, Forms
Channels to secondary actors	Touch Screen
Post Condition	Volunteer history is updated

Table 19

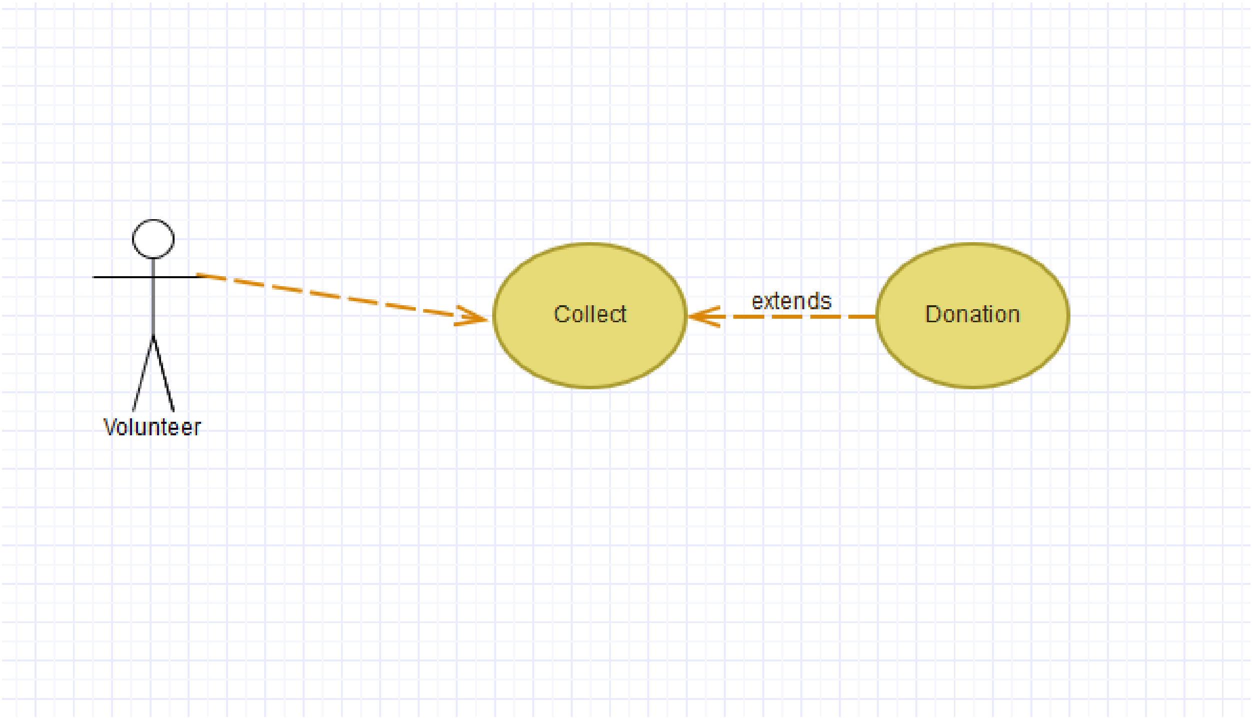


Figure 6

4.2.7. Use Case for My Profile

Primary actor	Donor, Volunteer.
Goal in context	To view profile, change information or log out.
Preconditions:	User has been registered and logged in.
Scenario:	**1. User:** Press My Profile Button form dashboard. **2. User:** Press Account Information or Logout Button. **3. User:** Changes information and saves it.
Exceptions	1. Internet connection is not available. 2. User is not registered.
Frequency of use:	Multiple times per day.
Channel to actor	Via device connected with internet.
Secondary actors	Database, Buttons, Forms
Channels to secondary actors:	Touch Screen
Post Condition:	User view profile updated message.

Table 20

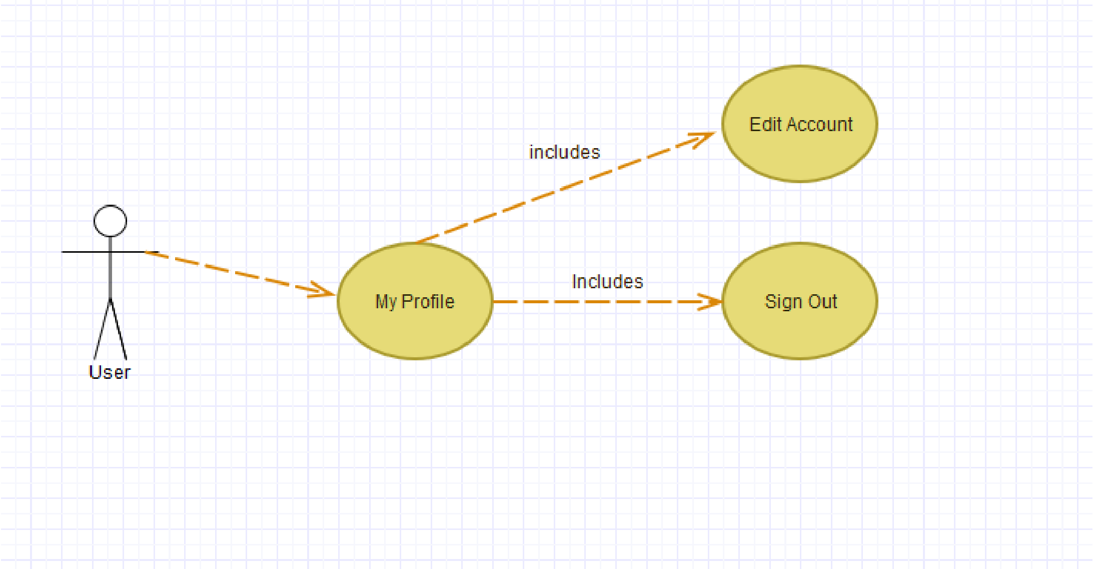


Figure 7

4.2.8. Use Case for Help:

Primary actor	Donor, Volunteer
Goal in context	To get help from admin.
Preconditions:	User is registered and logged in.
Trigger	A problem has occurred and user need help.
Scenario:	1. User: Press Help button form Dashboard. 2. User: Read FAQ. Otherwise call, message or Email
Exceptions:	1. Internet connection is not available. 2. User is not registered.
Frequency of use	Multiple times per day.
Channel to actor	Via device connected with internet.
Secondary actors	Database, Buttons
Channels to secondary actors	Touch Screen
Post Condition	Problem gets solved

Table 21

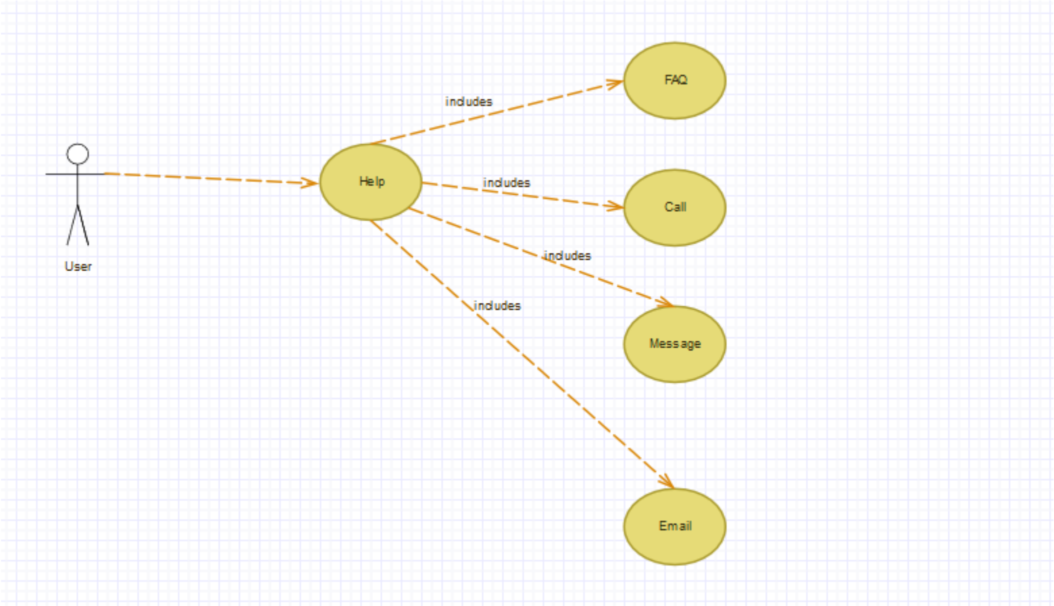


Figure 8

4.2.9. Use Case for History:

Primary actor	Donor. Volunteer
Goal in context	To view the record of user's services.
Preconditions:	User has registered and logged in.
Scenario:	1. User: Presses History button form Dashboard 2. User: View History.
Exceptions:	1. Internet connection is not available. 2. User is not registered.
Frequency of use	Multiple times per day
Channel to actor	Via device connected with internet
Secondary actors	Database, Buttons
Channels to secondary actors	Touch Screen
Post Condition	User get to view history

Table 22

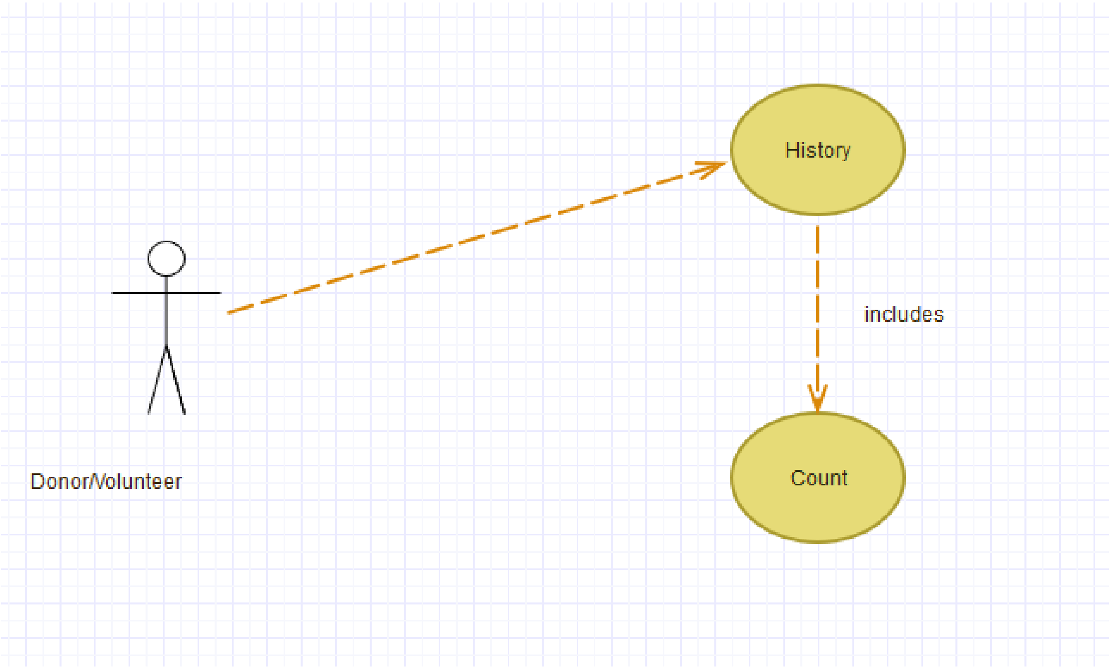


Figure 9

4.2.10. Use Case for Settings:

Primary actor	Donor, Volunteer
Goal in context	To change settings.
Preconditions	User has registered and logged in.
Scenario:	1. User: Presses Settings Button from Dashboard. 2. User: Change Settings. 3. User: Save Changes.
Frequency of use:	Multiple times per day.
Channel to actor	Via device connected with internet
Secondary actors:	Database, Buttons, Forms
Channels to secondary actors:	Touch Screen

Post Condition	User View Home Page

Table 23

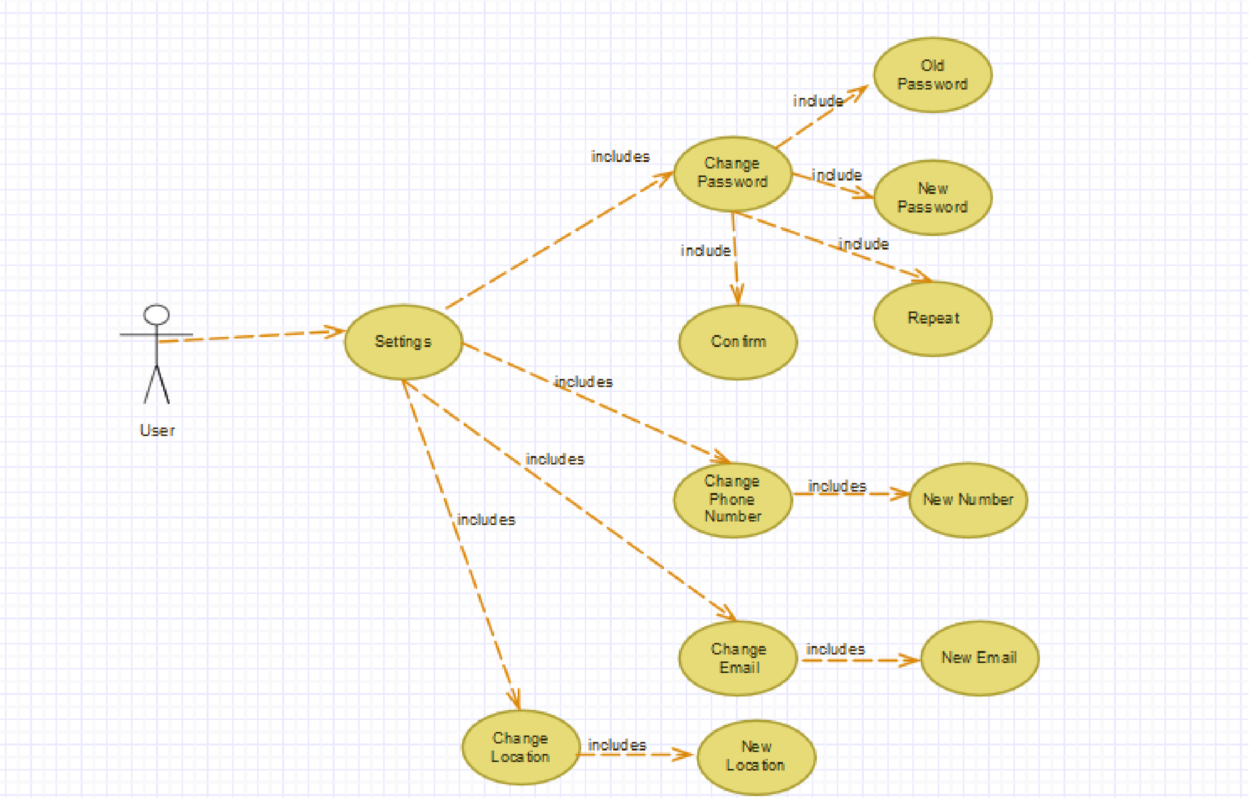


Figure 10

4.2.11. Use Case for Add Needy:

Primary actor	Volunteer
Goal in context	To add another Needy in the record
Preconditions	New Needy is not already in the list.
Scenario	1. Volunteer: Find a new needy and wants admin to provide service to them as well. 2. Volunteer: Request for new Needy by submitting Add Needy form. 3. Admin: Approves request after verification.

Exceptions	Admin already has the new needy information.
Frequency of use	Multiple times per day.
Channel to actor	Via device connected with internet.
Secondary actors	Database, Buttons, Forms, Admin
Channels to secondary actors	Touch Screen
Post Condition	New deserving Needy has been added

Table 24

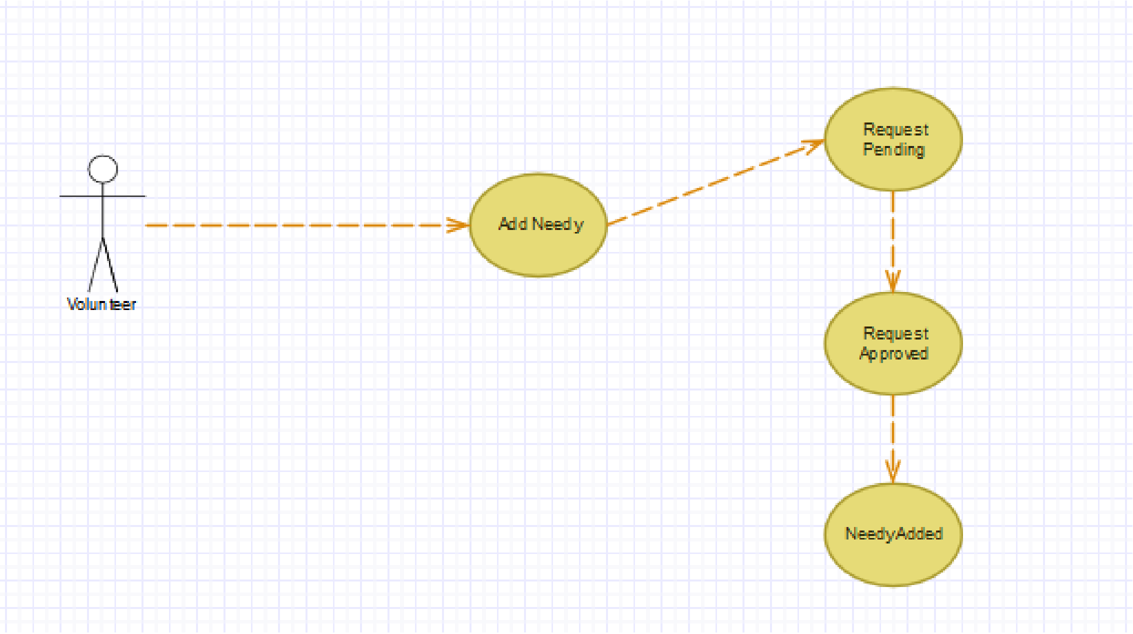


Figure 11

Chapter 5

IMPLEMENTATION

5.1. System Integration:

This project consists of two android application: Razakaar (Donor) and Razakaar Volunteer. Both application share a few common details such as Login, Sign Up, Help, Settings and implementation of User Interface. However, Both applications are designed separately as well as function on completely different grounds. The two application work together in integration to make this project completely successful. This link between the functionality of two application is called System Integration, which can be observed to be completely functional for Razakaar Project.

5.2. Activity Diagrams:

5.2.1. Activity Diagram for Login:

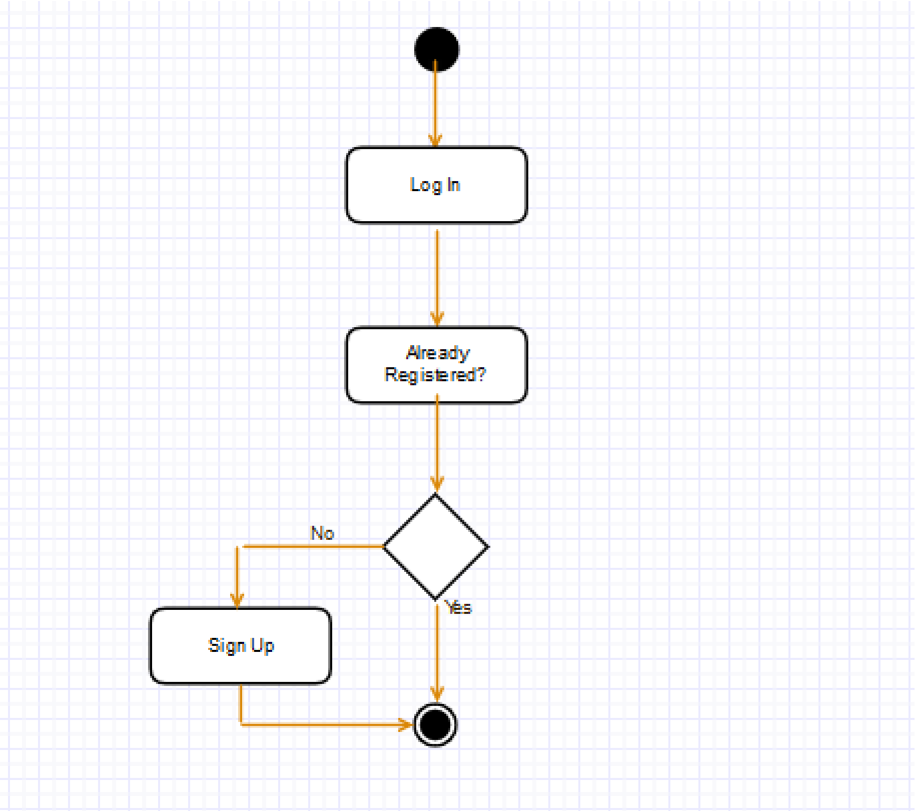


Figure 12

5.2.2. Activity Diagram for Sign Up:

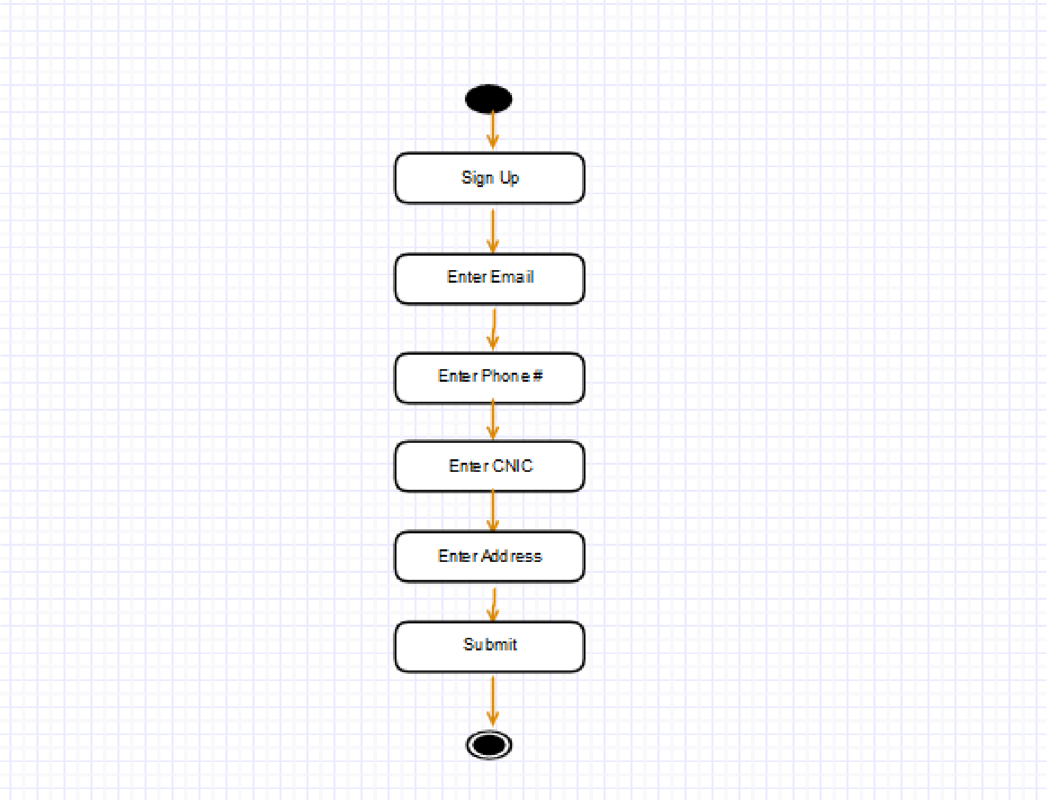


Figure 13

5.2.3. Activity Diagram for Donate:

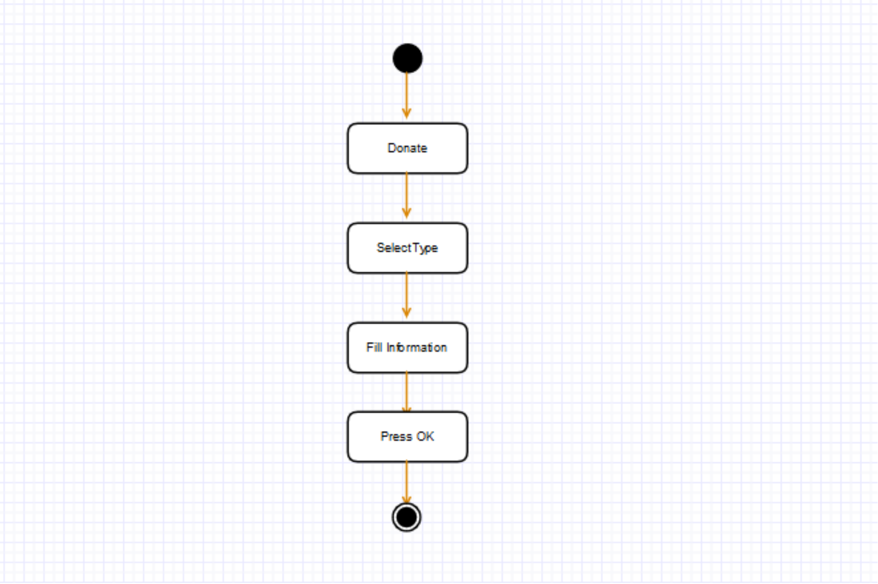


Figure 14

5.2.4. Activity Diagram for Collect:

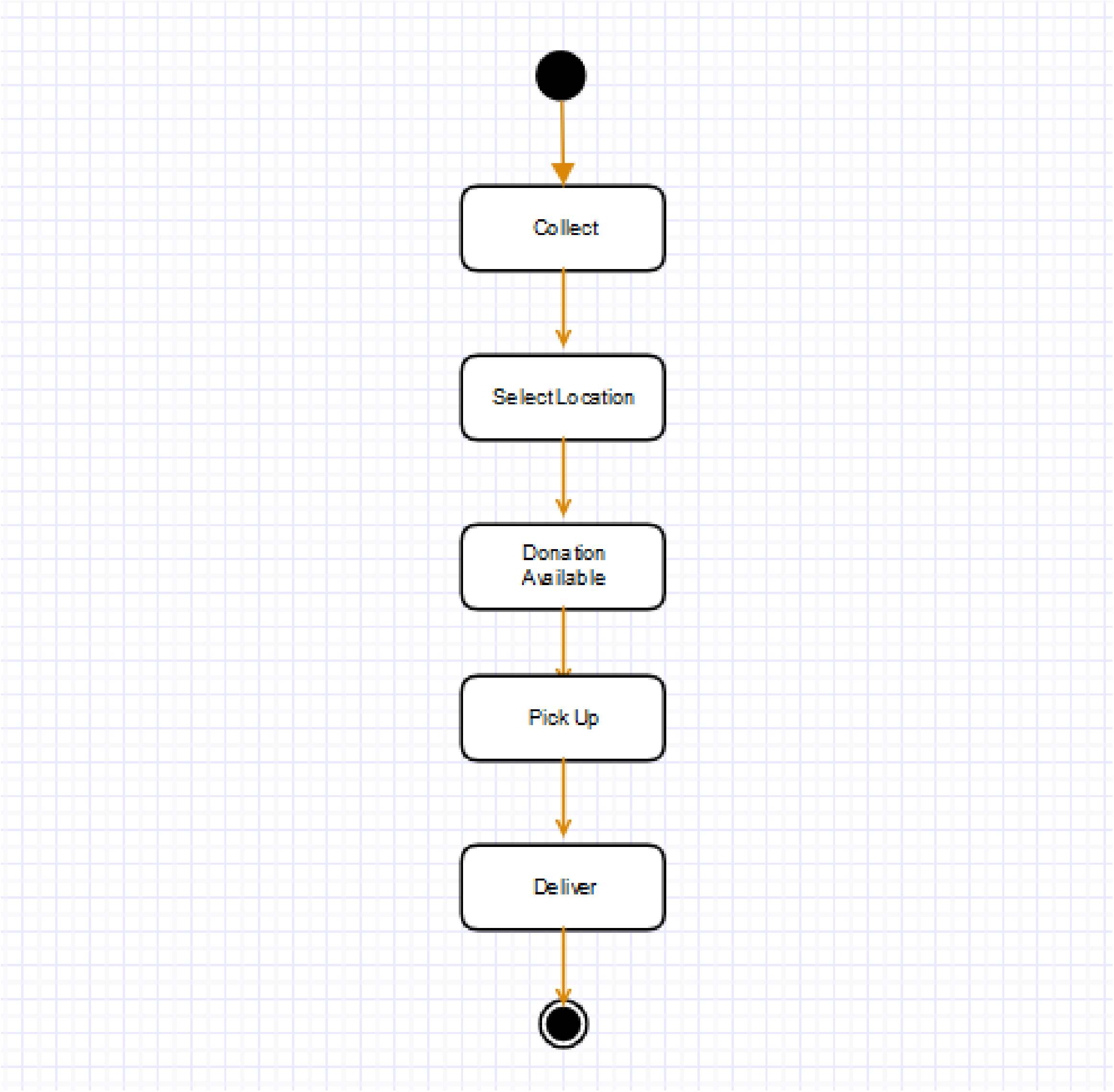


Figure 15

5.2.5. Activity Diagram for Add Needy:

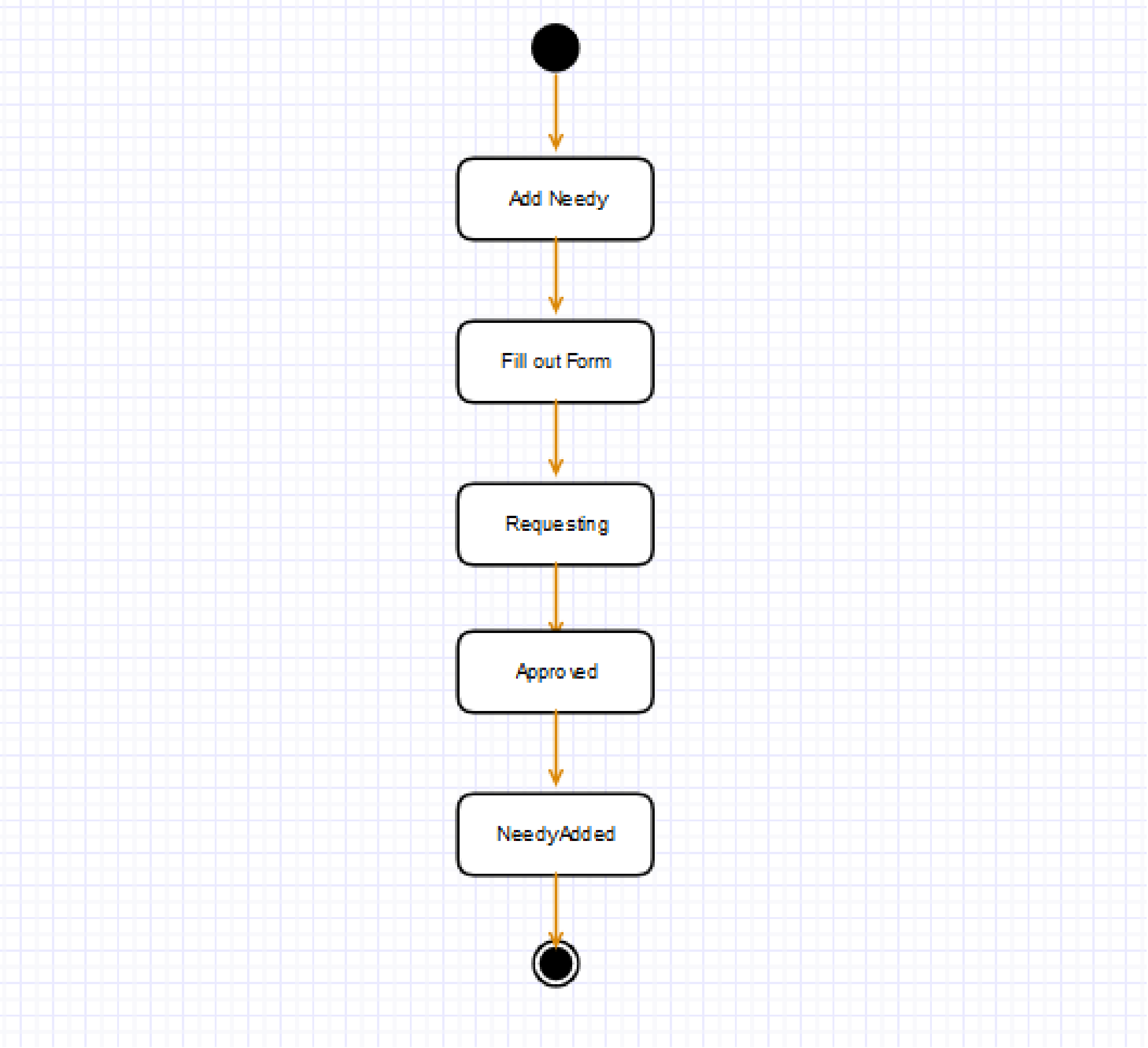


Figure 16

5.2.6. Activity Diagram for Main Activity:

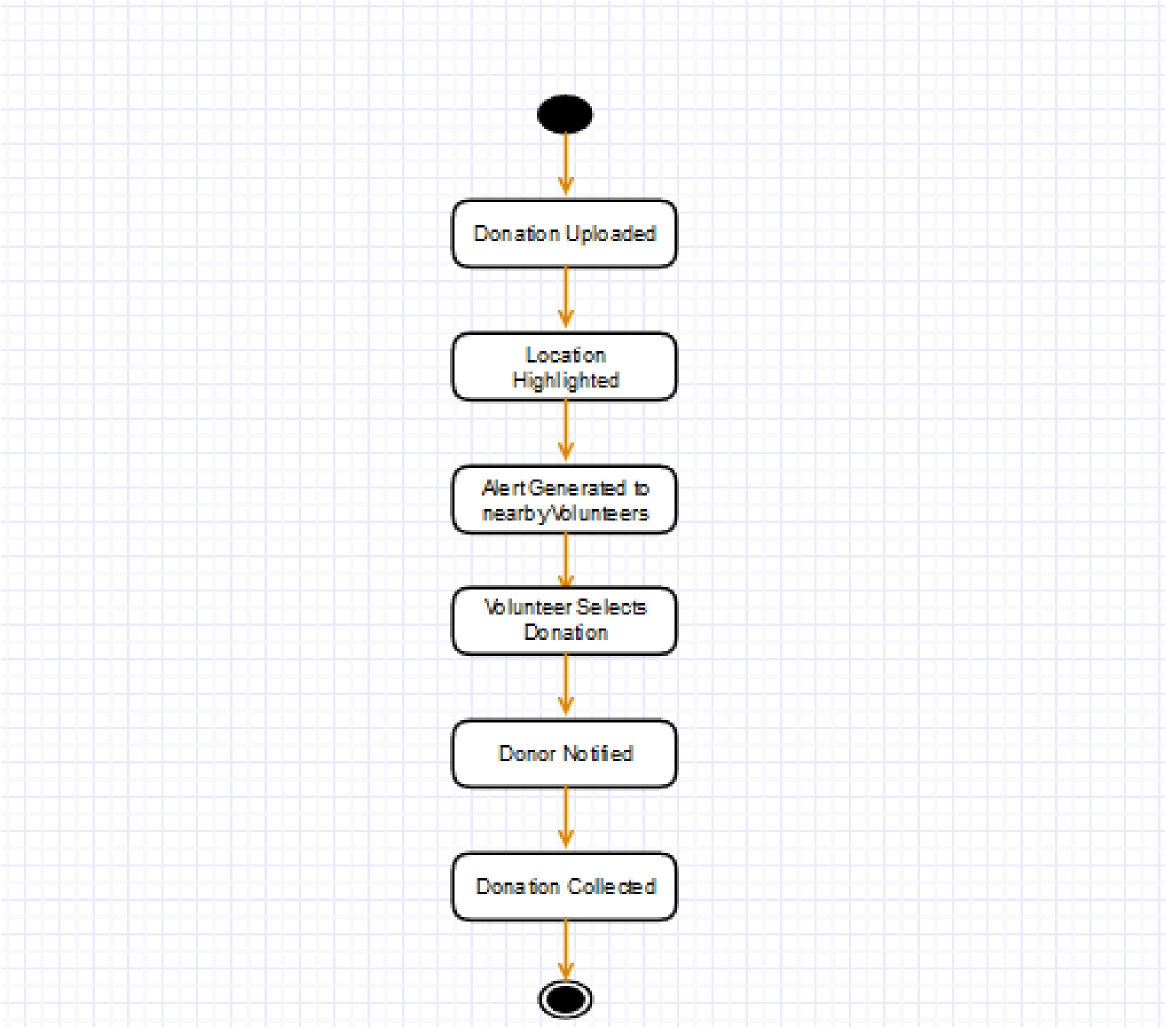


Figure 17

5.2.7. Activity Diagram for Help:

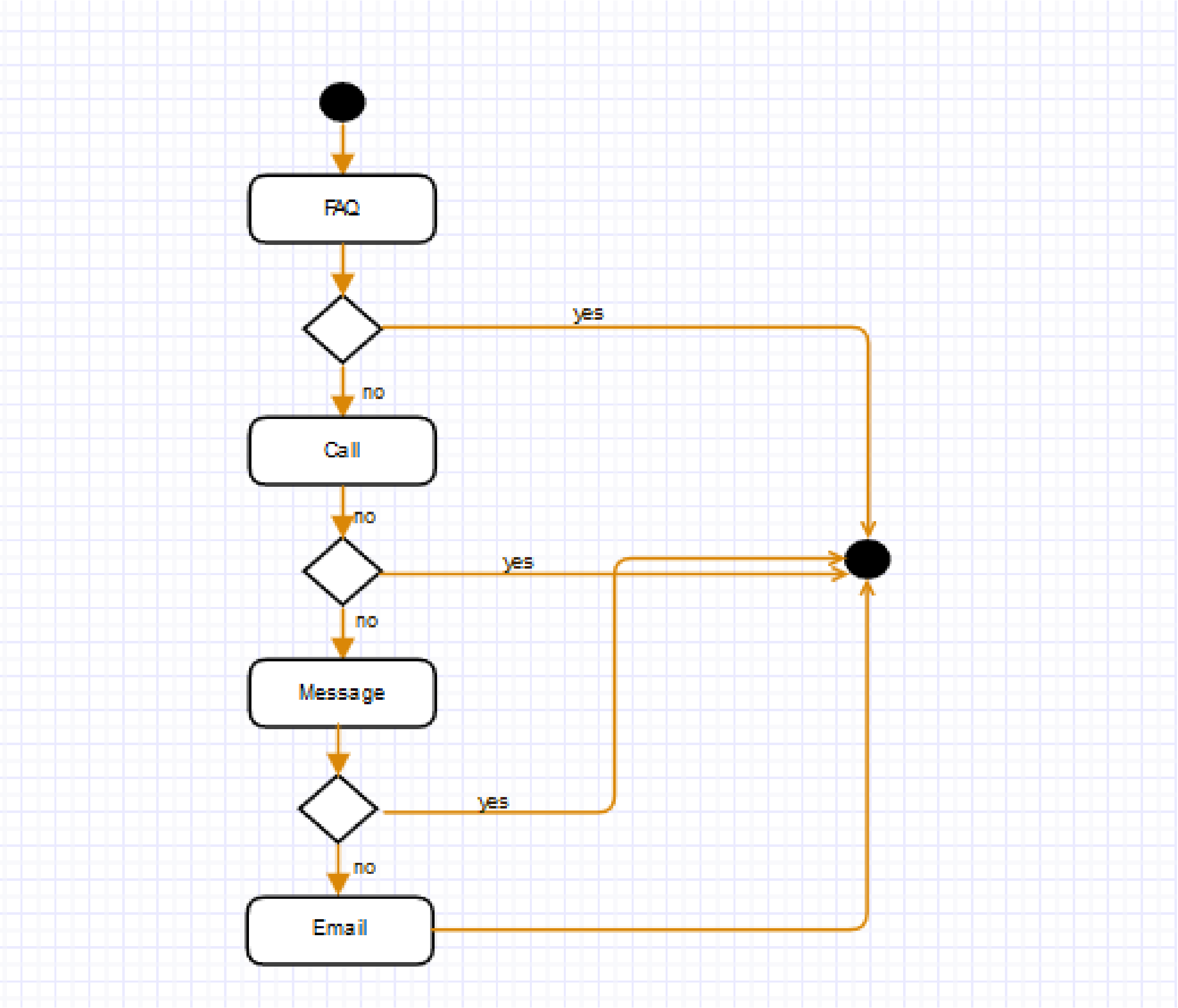


Figure 18

5.2.8. Activity Diagram for Settings:

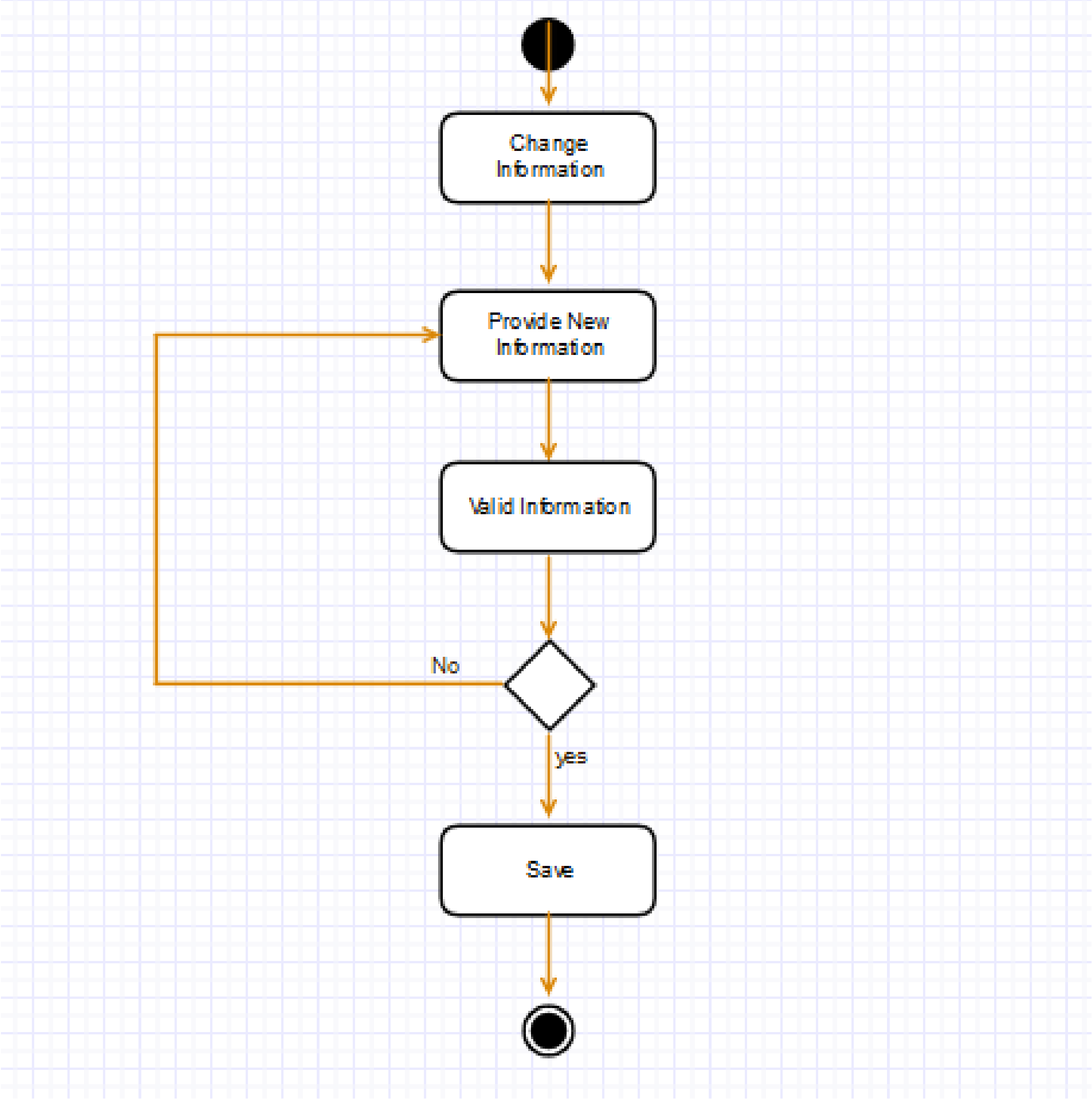


Figure 19

Sequence Diagram:

5.2.9. Sequence Diagram for Login:

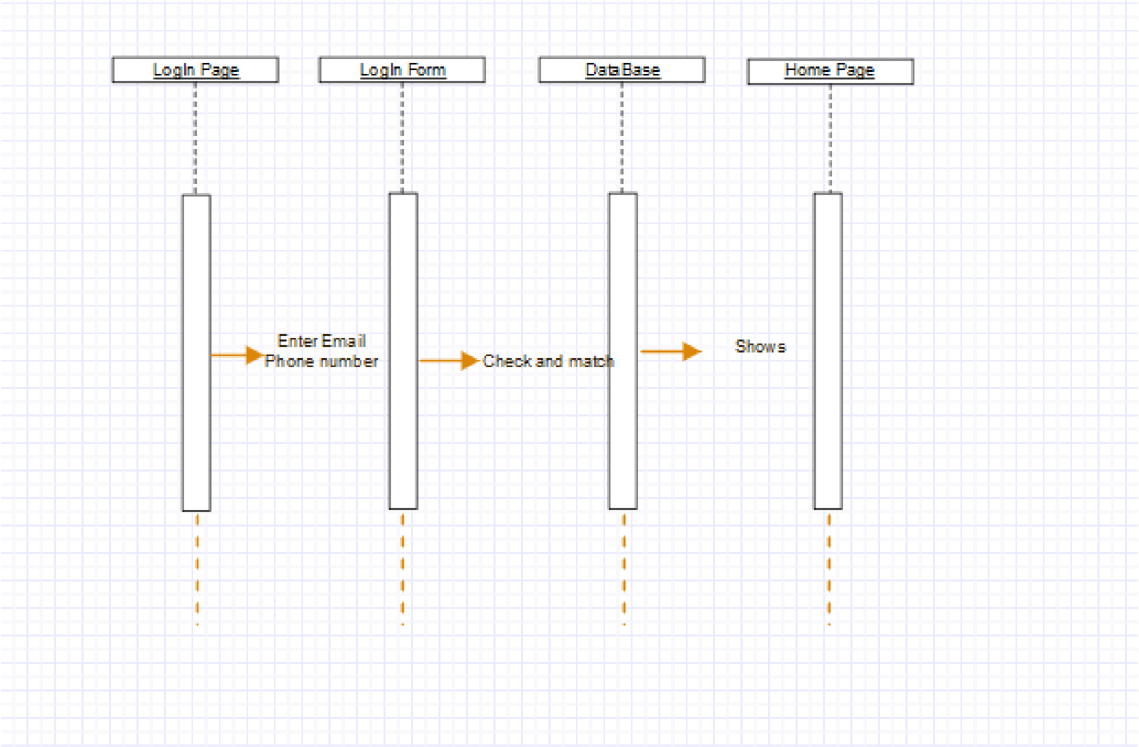


Figure 20

5.2.10. Sequence Diagram for Sign Up:

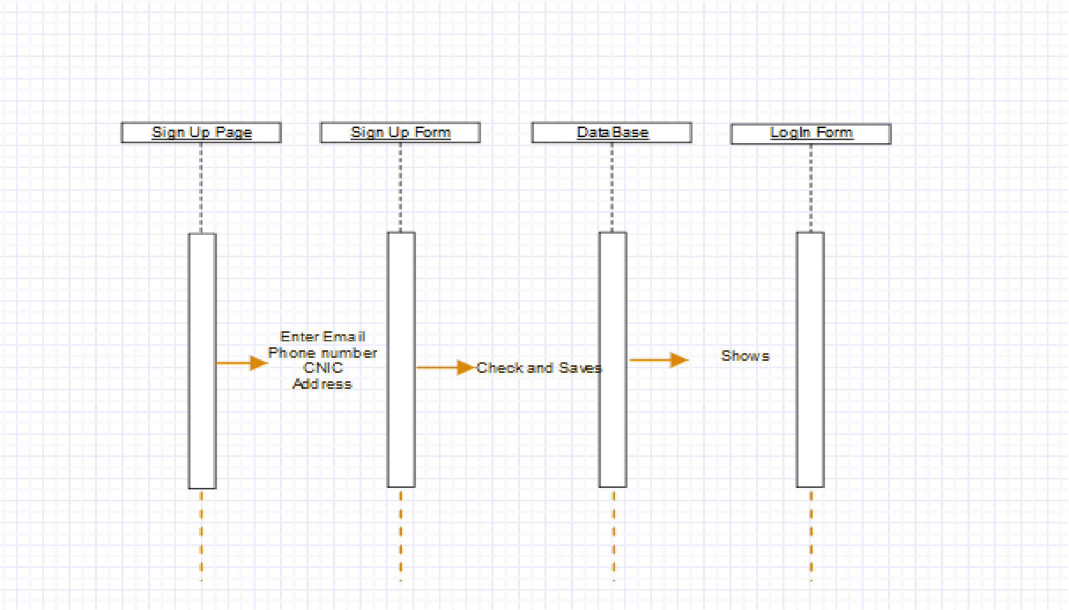


Figure 21

5.2.11. Sequence Diagram for Donate:

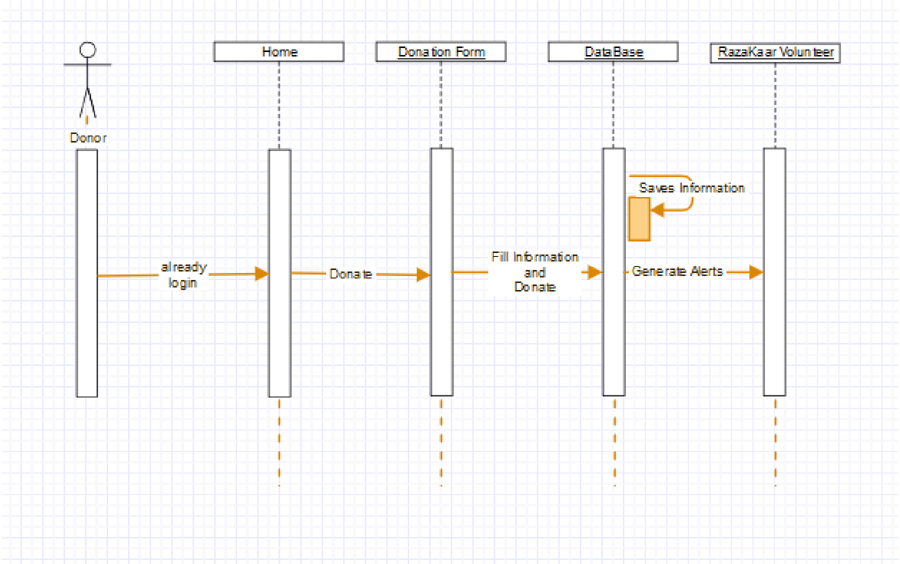


Figure 22

5.2.12. Sequence Diagram for Collect:

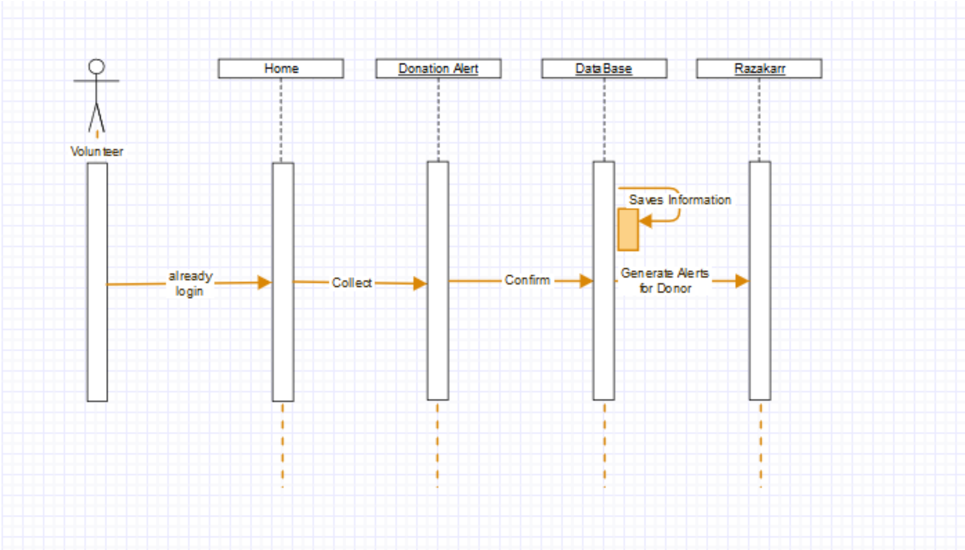


Figure 23

5.2.13. Sequence Diagram for Add Needy:

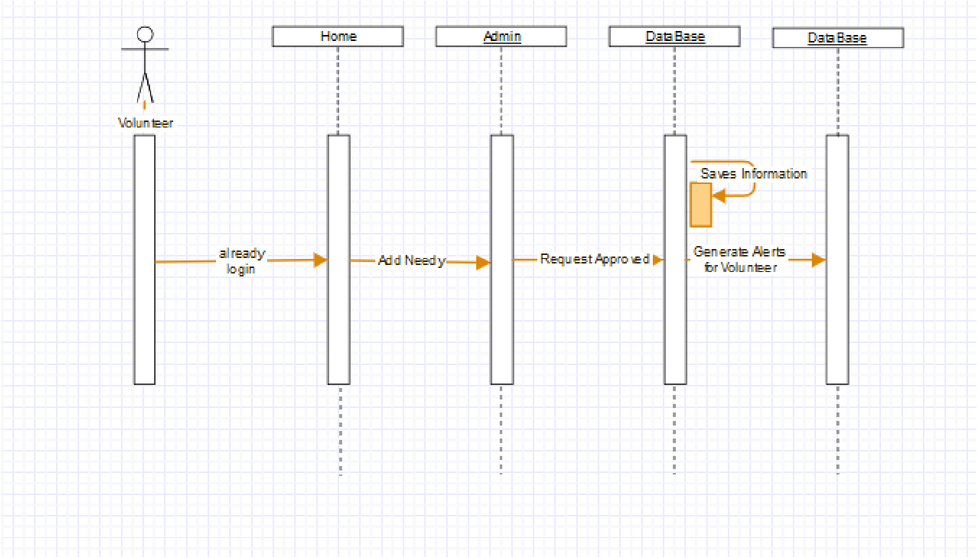


Figure 24

5.3. Class Diagram:

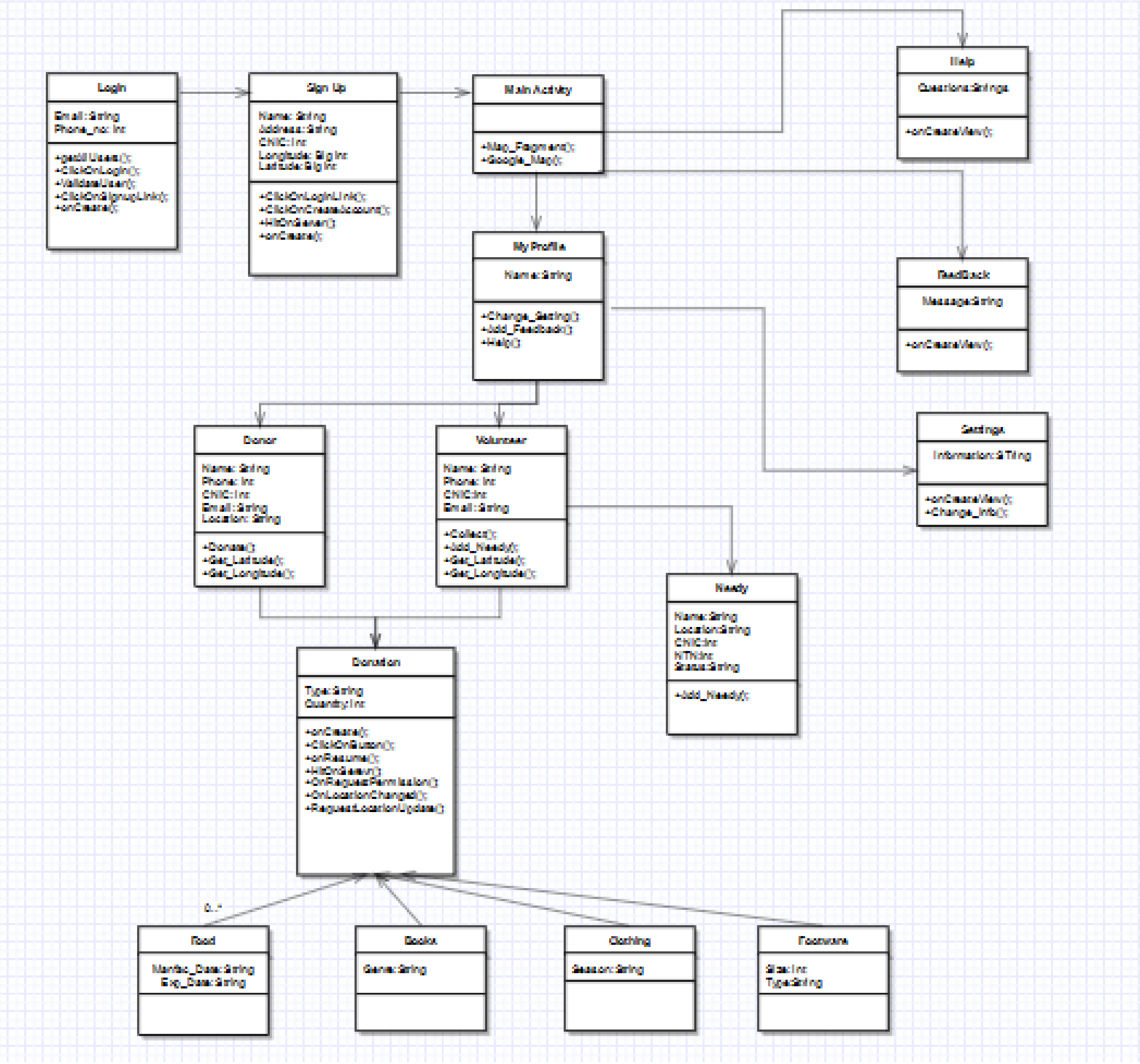


Figure 25

5.4. Component Diagram:

5.4.1. Component Diagram for Razakaar:

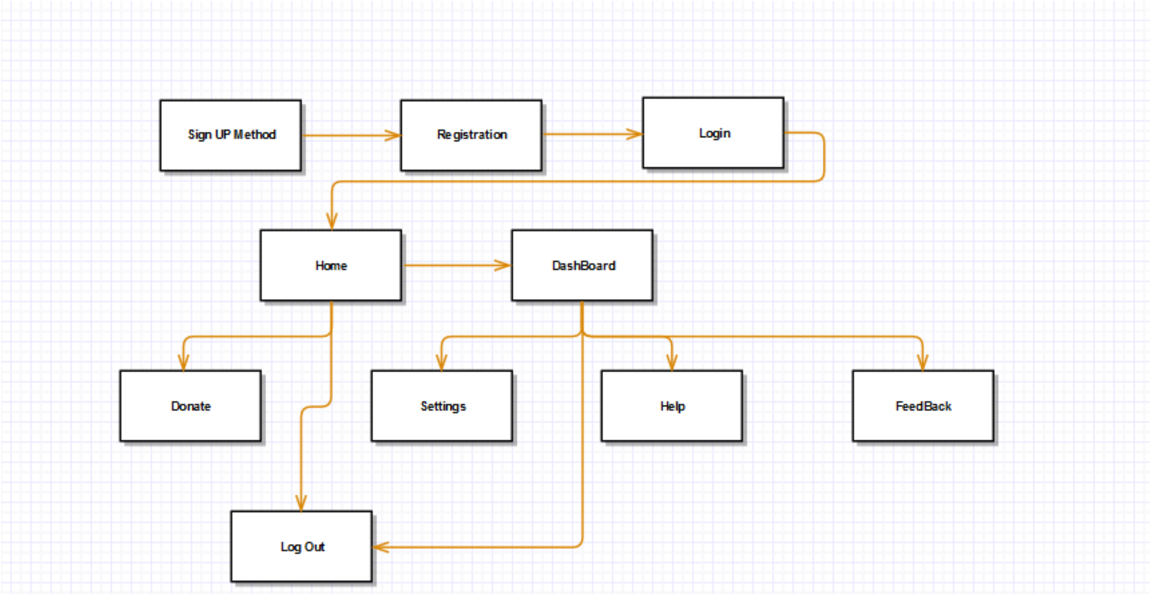


Figure 26

5.4.2. Component Diagram for Razakaar Volunteer:

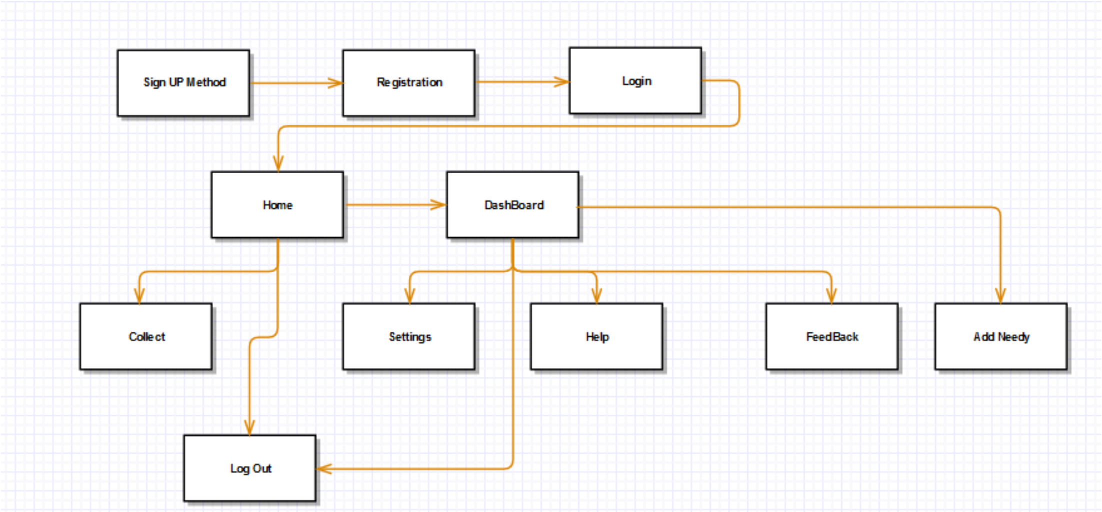
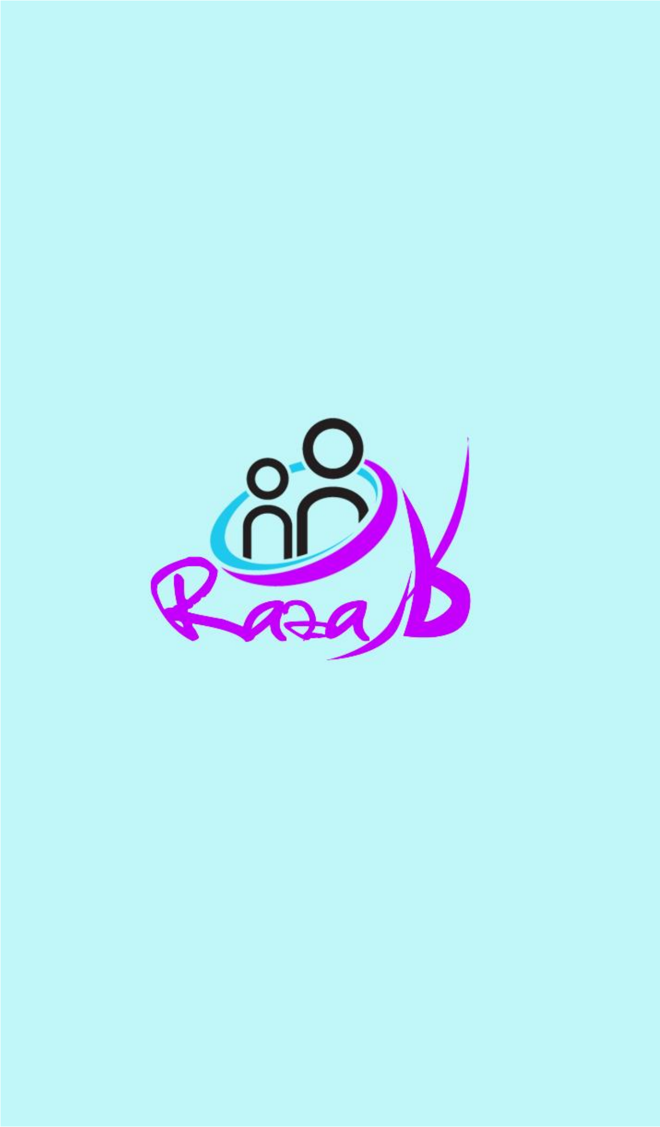


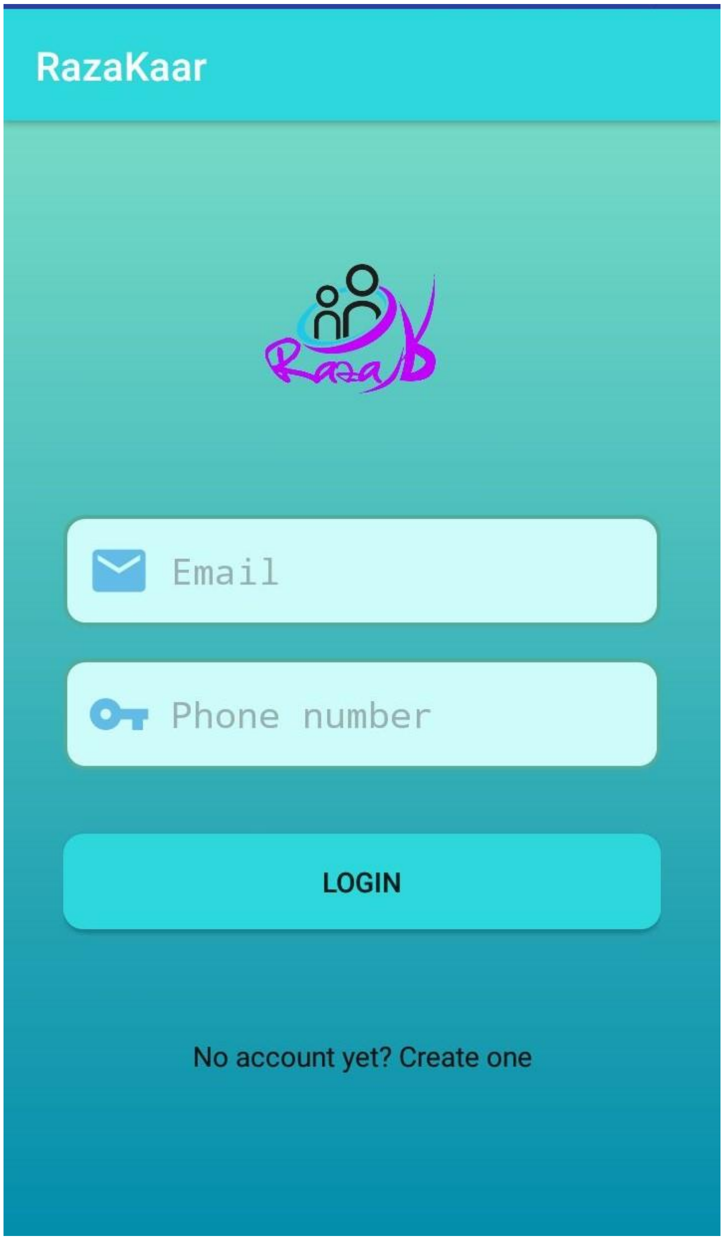
Figure 27

5.5. User Interfaces:

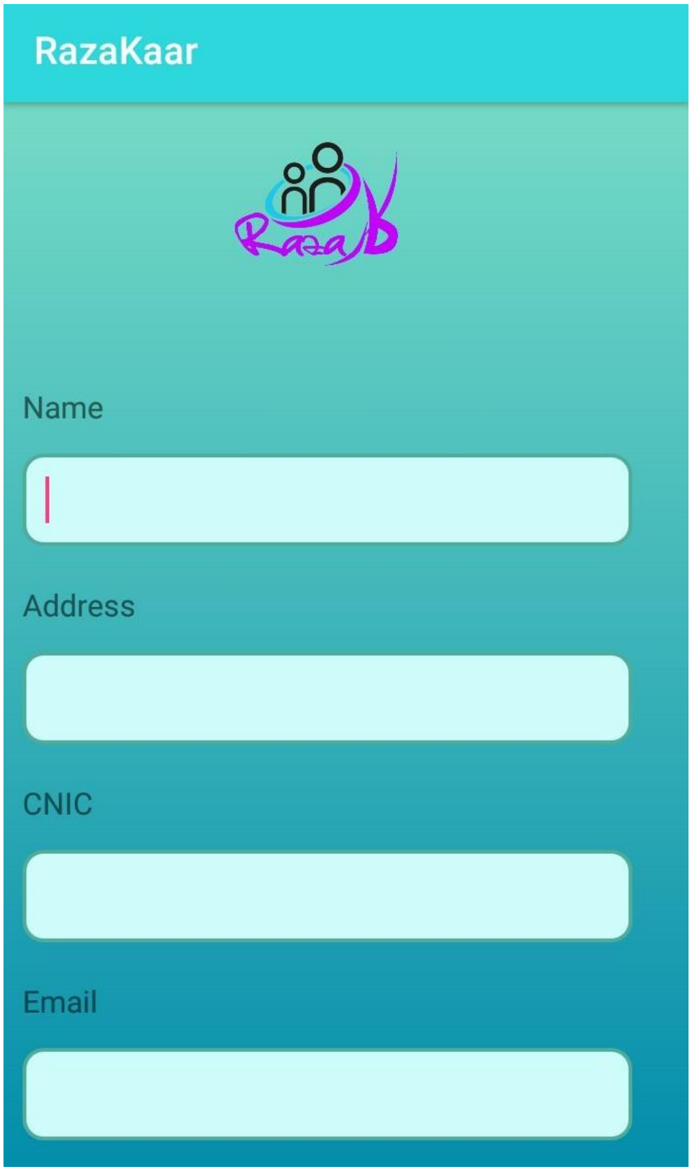
5.5.1. User Interface of Splash Screen:



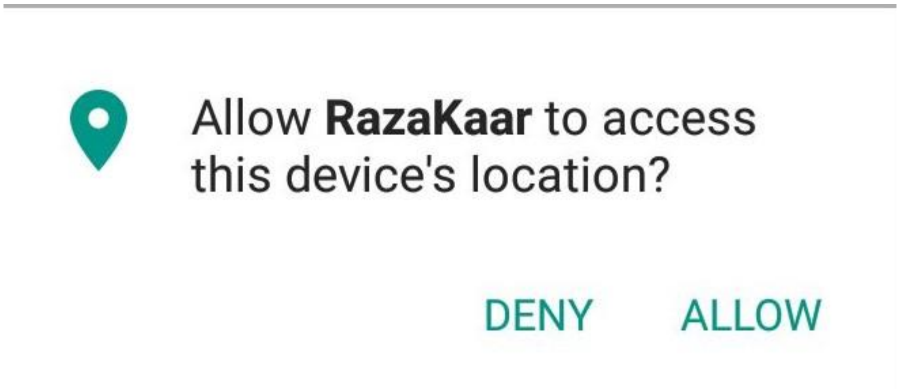
5.5.2. User Interface of Login:



5.5.3. User Interface for Sign Up:



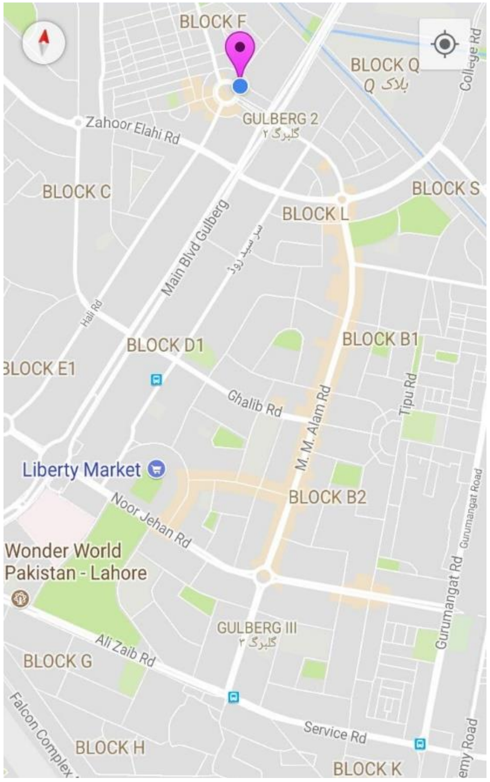
5.5.4. User Interface for Location Access:

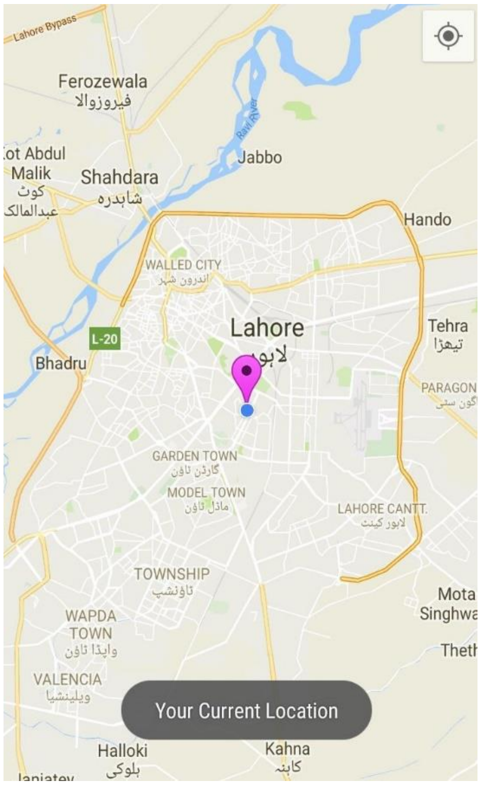


Allow **RazaKaar** to access this device's location?

DENY ALLOW

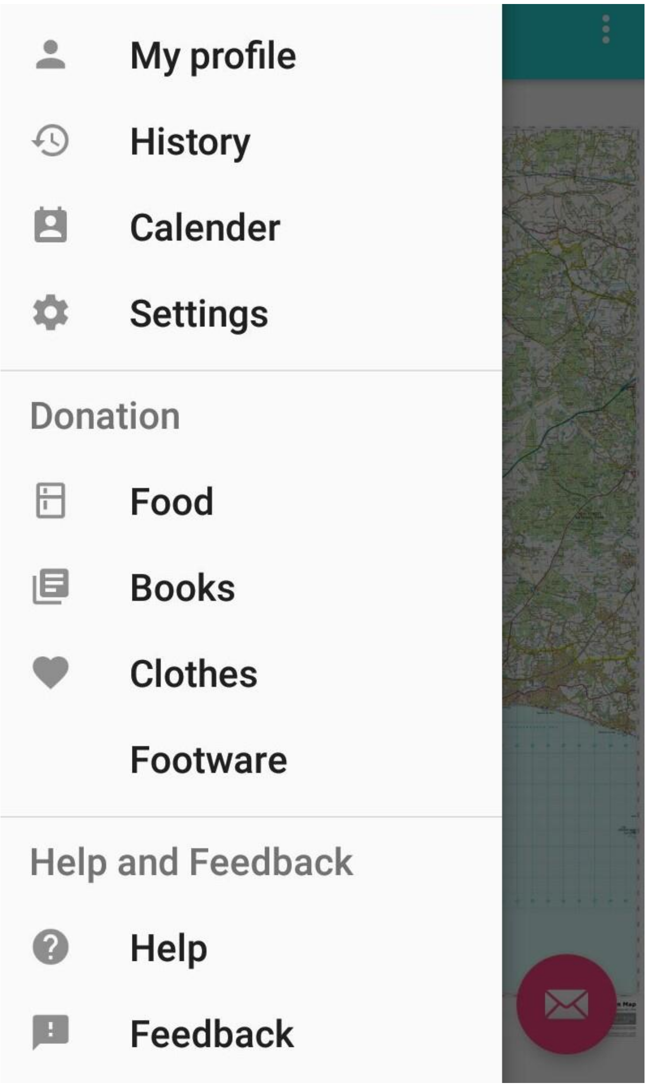
5.5.5. User Interface for Home Page:



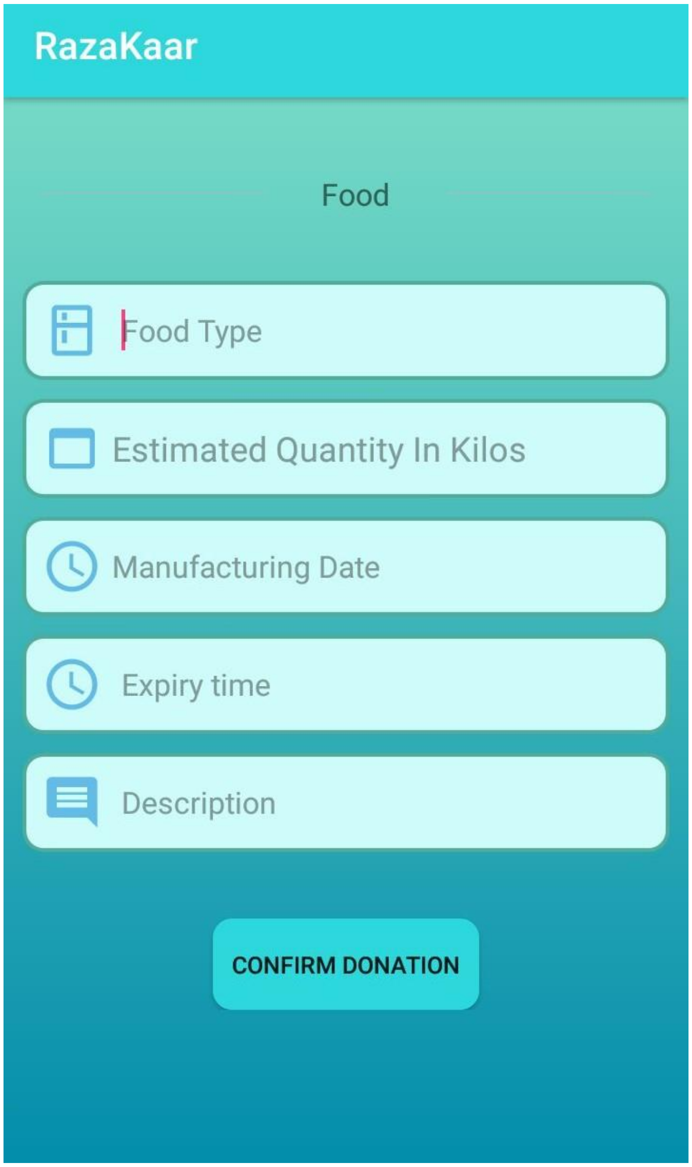


Your Current Location

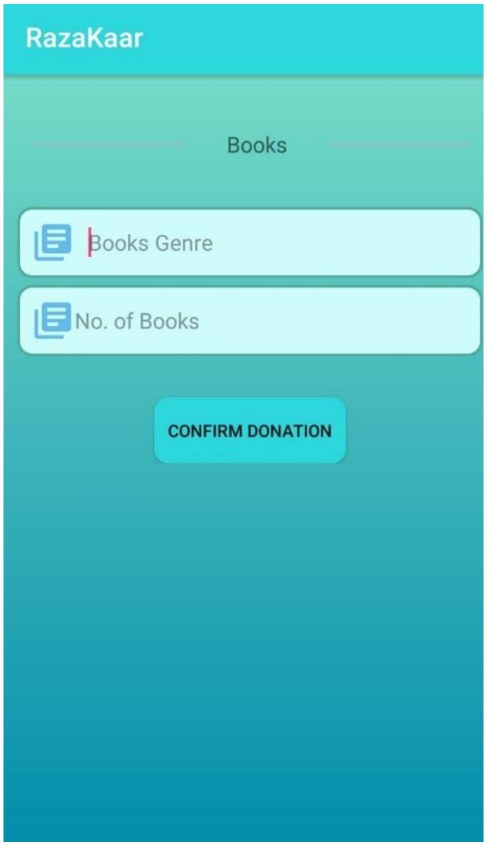
5.5.6. User Interface for Dashboard:

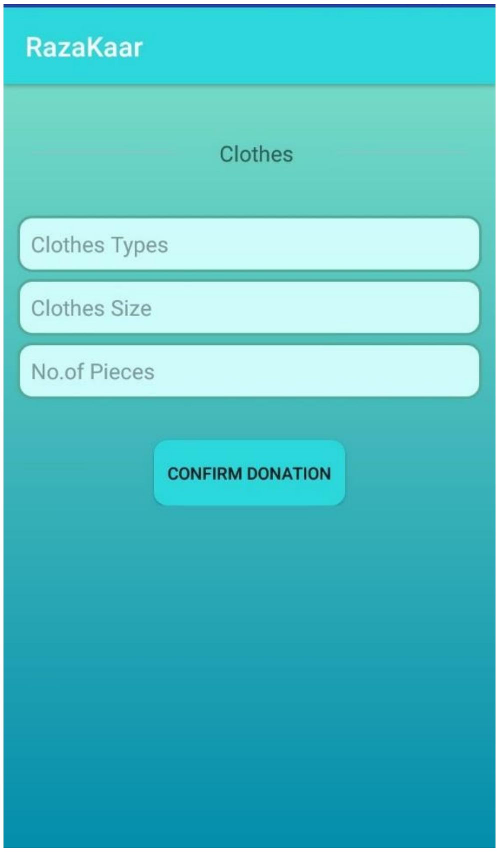


5.5.7. User Interface for Food Donation:

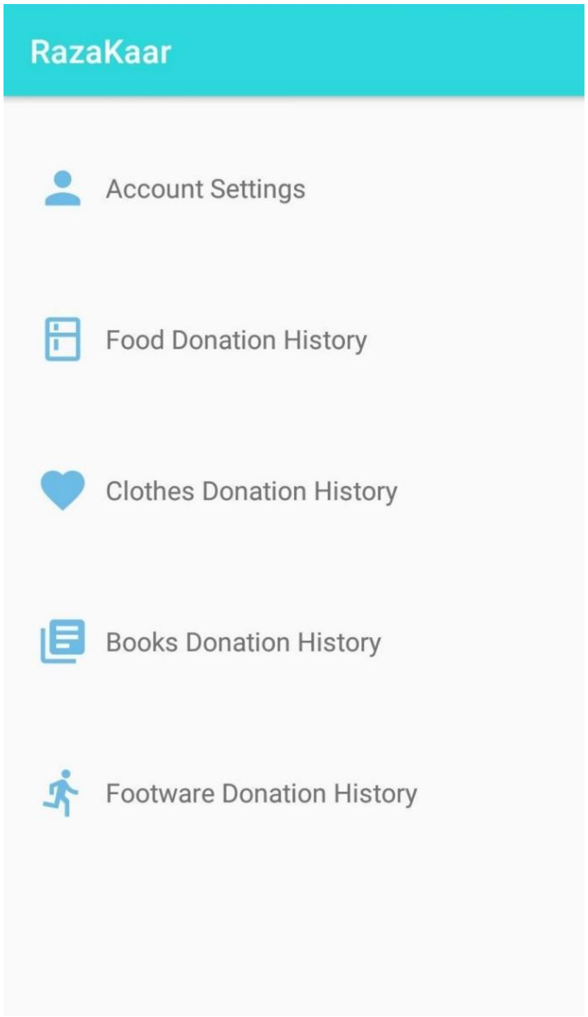


5.5.8. User Interface for Other Donation:

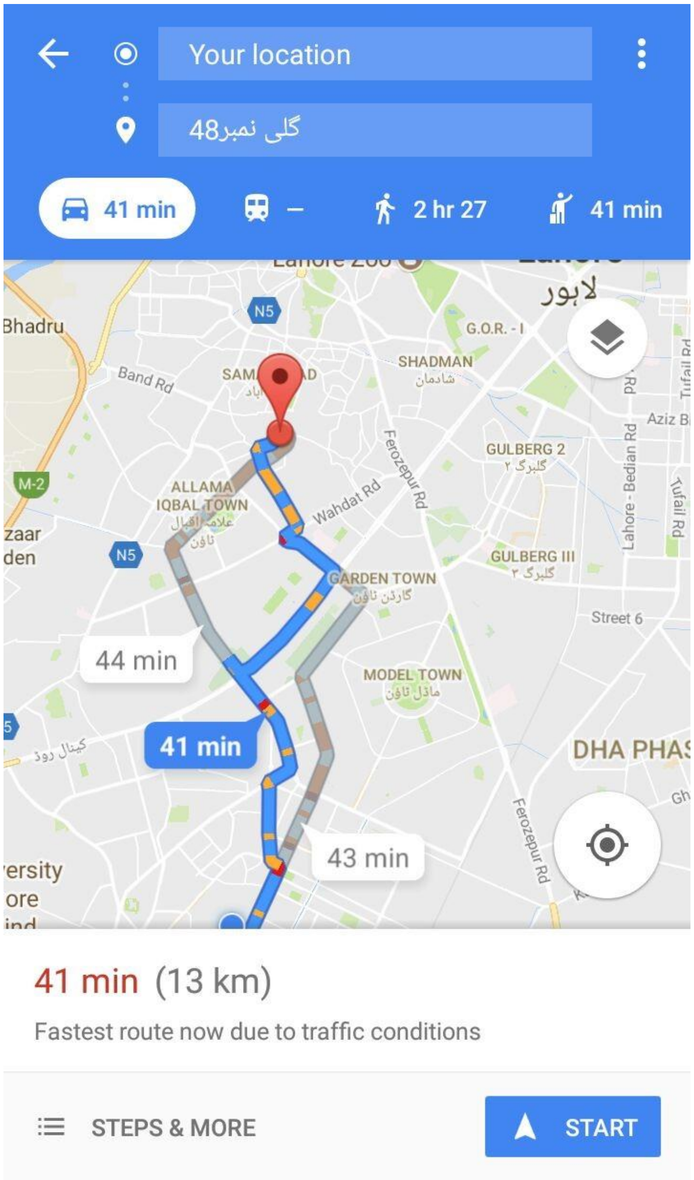




5.5.9. User Interface of My Profile:



5.5.10. User Interface of Map:



Your location

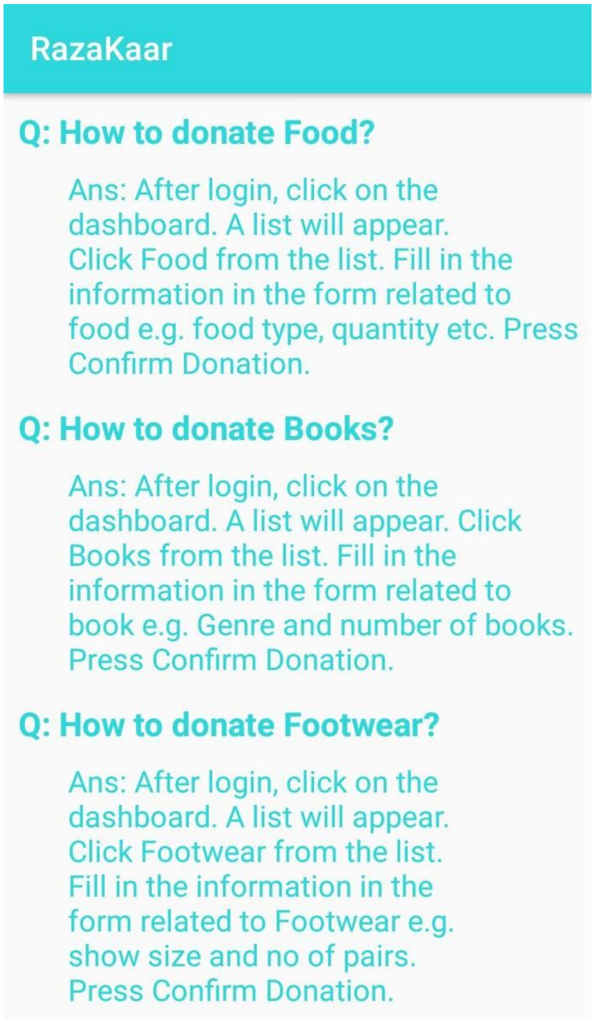
گلی نمبر48

41 min (13 km)

Fastest route now due to traffic conditions

Figure 5.6.1.10

5.5.11. User Interface of Help:



RazaKaar

Q: How to donate Food?

Ans: After login, click on the dashboard. A list will appear. Click Food from the list. Fill in the information in the form related to food e.g. food type, quantity etc. Press Confirm Donation.

Q: How to donate Books?

Ans: After login, click on the dashboard. A list will appear. Click Books from the list. Fill in the information in the form related to book e.g. Genre and number of books. Press Confirm Donation.

Q: How to donate Footwear?

Ans: After login, click on the dashboard. A list will appear. Click Footwear from the list. Fill in the information in the form related to Footwear e.g. show size and no of pairs. Press Confirm Donation.

Chapter 6

EVALUATION

6.1. Unit Testing

While development, each class and module is tested before moving further in development. Each test conducted was on unit level. A class is tested for all kind of functionality and technical complication individually as well as after being in integration with other classes in a module. This approach removes maximum possibility of any functionality regarding error.

6.1. Test Cases:

6.1.1. Test Case for User Login:

Ref	4.2.2.
Purpose	User login successfully.
Prerequisite	User must register to the system to be able to login.
Test data	1. Valid email and phone number. 2. Invalid email and valid phone number. 3. Valid email and incorrect phone number.
Steps	1. Go to Login Page. Enter email and phone number. Press login button. 2. Go to Login Page. Enter invalid email and correct phone number. Press login button. 3. Open Login Page. Enter valid email and incorrect phone number. Press login button.
Expected Results	1. User successfully login to system. 2. User not able to login to system. 3. User not able to login to system.
Actual Results	1. User successfully logged in. 2. Information is incorrect. 3. Information is incorrect.

Status	Pass

Table 25

6.1.2. Test Case for Sign up:

Ref	4.2.3
Purpose	To register new user.
Prerequisite	User must enter all fields.
Test data	1. User entered all fields. 2. User missed some fields. 3. User did not enter any field.
Steps	1. Open the sign up form. Enter information to all fields. Press Ok. 2. Open the sign up form. Enter information to some of the fields. Press Ok. 3. Open the sign up form. Enter information to none of the fields. Press Ok.
Expected Results	1. User successfully registered. 2. Enter all fields. 3. Enter all fields.
Actual Results	1. Admin at server successfully receive entry in database. 2. No new registration was received. 3. No new registration was received.
Status	Pass

Table 26

6.1.3. Test Case to Donate:

Ref	4.2.5.
Purpose	To successfully make a donation.
Prerequisite	User must login to the system to be able to make a donation.
Test data	1. User entered data to all fields. 2. User entered data to some of the fields. 3. User entered data to none of the fields.
Steps	1. Go to Donation Page. 2. Select any type of Donation. 3. Enter information. 4. Press Confirm Donation Button.
Expected Results	1. Donation Confirmed. 2. Fill all the fields. 3. Fill all the fields.
Actual Results	1. Admin on server side receives a new donation entry. 2. No new donation made. 3. No new donation made.
Status	Pass

Table 27

6.1.4. Test Case to Collect:

Ref	4.2.6.
Purpose	To Collect the Donation made earlier.
Prerequisite	User must have registered as volunteer and must be logged in.
Test data	1. Volunteer presses Collect for donation made earlier. 2. Volunteer presses Collect while there is no donation.
Steps	1. Open the Razakaar Volunteer application, go to the dashboard. Click Collect Button. 2. Open the Razakaar Volunteer application, go to dashboard, press Collect Button
Expected Results	1. Donor gets alert for corresponding volunteer.

	2. No Donation to Collect Message.
Actual Results	3. Successful alert has been generated at the backend. 4. No alert generated.
Status	Pass

Table 28

6.1.5. Test Case for Add Needy:

Ref	4.2.11
Purpose	To add a new Needy (Person OR Place)
Prerequisite	User has registered as Volunteer and must be logged in.
Test data	1. Add new needy information. 2. Add already existing Needy.
Steps	1. Login to the Razakaar Volunteer application, go to the dashboard. Press Add Needy button. Add information. 2. Login to the Razakaar Volunteer application, go to the dashboard. Press Add Needy button. Add information
Expected Results	1. New Needy Added Successfully. 2. Needy already exists.
Actual Results	1. New information received by admin at server side. 2. No new information received by admin at server side.
Status	Pass

Table 29

6.1.6. Test Case for History:

Ref	4.2.10.
Purpose	User need to check their record of services.
Prerequisite	User must be registered and logged in.
Test data	1. User has performed some services. 2. User did not perform any services.
Steps	1. Go to the dashboard, click History. 2. Go to the dashboard, click History.
Expected Results	1. Number of services performed are shown as a number. 2. Error, you have not performed any service.
Actual Results	1. Number of services are fetched from database. 2. No record in database.
Status	Pass

Table 30

Chapter 7

CONCLUSION & FUTURE WORK

a. Problems Encountered:

During the development of this project, following problems were faced:

- As Android Development is not part of any course being taught in university, so it took time to learn the environment alone.
- Learning and implementing the Google Map APIs for location was the most challenging task.
- Writing scripts and integration with user interface.
- Handle the UI panel settings in Android Studio.

b. Future Work:

This project is designed to implement in the society practically. With the addition of few new functionalities, this application will be introduced to the world through social media and Google Play Store. Moreover, hiring of volunteer will be a priority at first until people start registering themselves for volunteer services. Campaigns about food wastage and importance of community service will be held. Big names and corporate companies will be sent proposal to collaborate with us for the noble cause. Food chains will be sent proposal to collaborate for incentives for donors and volunteers as well as for extra food.

Food shelter will be established if necessary. Other team doing similar work will be sent proposal to use the application instead or join our team. The countries which already hold community services can use this application for more efficient results.

c. Conclusion:

The application is a practical step to start achieving sustainable development goals in Pakistan as well as all over the world. It is implemented to make good deed an ease and not a burden. With the technical implementation, it has been a great chance to learn everything we have studied especially object oriented programming and Java. Moreover, to design the complete package with user interfaces and handling with backend programming was a chance to explore many technical functionalities. In short, it was a very strong base to stand in the market as software developer.

APPENDIX

List of Tables:

List of Figures:

BIBLIOGRAPHY

- Build the Generation of Location Experience. Retrieved from Google Maps Developer Guide: https://developers.google.com/maps/

- Connectivity in Android. (n.d.). Retrieved from Android Developer : https://developer.android.com/guide/topics/connectivity/index.html

- Dietel, D. &. Java: How to Program. Pearson.

- Guide to Sequence Diagram. Retrieved from SmartDraw: https://www.smartdraw.com/sequence-diagram/

- How to make Activty diagram. Retrieved from gliffy: https://www.gliffy.com/

- Pressman, R. S. (2010). Software Engineering: A practitioner's Approach. Singapore: MacGrew Hills.