Style Guidelines for Final Year Project ReportsFOOD DONOR

Final Year Project

Session 2014-2018

A 4th Year Student

A project submitted in partial fulfilment of the

COMSATS-Lancaster Dual Degree

of

BSc. (Hons.)BS in Computer Science / Software Engineering (CIIT)

BS in Computing / Software Engineering (LU)

**C:\Documents and Settings\drzhabib\Desktop\lancaster-university-logo.png**

Department of Computer Science

COMSATS Institute of Information Technology, Lahore

04 January 2018

# Project Detail

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Type (Nature of project) | | | [ \*] **D**evelopment [ ] **R**esearch [ ] **R**&**D** | | |
| Area of specialization | | |  | | |
| **Project Group Members** | | | | | |
| Sr.# | Reg. # | Student Name | | Email ID | \*Signature |
| (i) | DDP-SP14-BSE-045 | M Bilal Kafayat | | bilalkafayat@gmail.com |  |
| (ii) | DDP-SP14-BSE-089 | Ali Haider | | omair0334@gmail.com |  |
| (iii) |  |  | |  |  |

\*The candidates confirm that the work submitted is their own and appropriate credit has been given where reference has been made to work of others

# Plagiarism Free Certificate

This is to certify that, I am \_M.Bilal Kafayat\_\_\_\_\_\_\_\_\_ S/D/o \_\_\_\_\_\_\_\_Kafayat Ullah\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, group leader of FYP under registration no CIIT/\_\_ddp-sp14-bse-045\_\_/LHR at Computer Science Department, COMSATS Institute of Information Technology, Lahore. I declare that my FYP proposal is checked by my supervisor and the similarity index is \_\_\_\_16\_\_\_\_% that is less than 20%, an acceptable limit by HEC. Report is attached herewith as Appendix A.

Date: \_\_\_04 Jan 2018\_\_ Name of Group Leader: \_\_\_\_M.Bilal Kafayat\_\_\_Signature: \_\_Bilal\_\_

Name of Supervisor: \_\_\_Dr.M.Hussnain\_\_\_\_\_\_ Co-Supervisor (if any):\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Designation: \_\_\_AP\_\_\_\_\_\_\_\_ Designation: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature: \_\_\_\_ Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

HoD: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Table of Contents

[FOOD DONOR 1](#_Toc502786176)

[Final Year Project 1](#_Toc502786177)

[Session 2014-2018 1](#_Toc502786178)

[Project Detail 2](#_Toc502786179)

[\*The candidates confirm that the work submitted is their own and appropriate credit has been given where reference has been made to work of others 2](#_Toc502786180)

[Plagiarism Free Certificate 2](#_Toc502786181)

[Abstract 9](#_Toc502786182)

[1 Introduction 10](#_Toc502786183)

[1.1 Objectives: 10](#_Toc502786184)

[1.2 Problem Statement: 10](#_Toc502786185)

[1.3 Assumptions and constraints: 11](#_Toc502786186)

[1.4 Project Scope: 11](#_Toc502786187)

[2 Chapter 2 : Requirements Analysis 12](#_Toc502786188)

[2.1 Literature Review: 12](#_Toc502786189)

[2.2 Stakeholders list: 13](#_Toc502786190)

[2.3 Requirements Elicitation: 14](#_Toc502786191)

[2.3.1 Functional Requirements 14](#_Toc502786192)

[2.3.2 Non-Functional Requirements 19](#_Toc502786193)

[2.3.3 Requirement Traceability Matrix 21](#_Toc502786194)

[2.4 Use case description: 28](#_Toc502786195)

[2.4.1 Login: 28](#_Toc502786196)

[2.4.2 Logout: 29](#_Toc502786197)

[2.4.3 Add Food: 30](#_Toc502786198)

[2.4.4 Update Food: 31](#_Toc502786199)

[2.4.5 Delete Food: 32](#_Toc502786200)

[2.4.6 View Food: 33](#_Toc502786201)

[2.4.7 Request Food: 34](#_Toc502786202)

[2.4.8 Fill the volunteer form: 36](#_Toc502786203)

[2.4.9 Volunteer Handle Receiver Requests: 37](#_Toc502786204)

[2.4.10 View Hunger Spot: 38](#_Toc502786205)

[2.4.11 Add Hunger Spot: 39](#_Toc502786206)

[2.5 Use Case Design: 40](#_Toc502786207)

[2.5.1 Donate Food User: 40](#_Toc502786208)

[2.6 Software Development Life Cycle Model: 44](#_Toc502786209)

[2.6.1 MODEL DIAGRAM 44](#_Toc502786210)

[3 Chapter 3: System Design: 45](#_Toc502786211)

[3.1 Work breakdown structure: 45](#_Toc502786212)

[3.2 Activity diagrams: 46](#_Toc502786213)

[3.2.1 Login : 46](#_Toc502786214)

[3.2.2 Logout : 47](#_Toc502786215)

[3.2.3 Add Food : 48](#_Toc502786216)

[3.2.4 Delete Food 49](#_Toc502786217)

[3.2.5 View Food 50](#_Toc502786218)

[3.2.6 Request Food 51](#_Toc502786219)

[3.2.7 Edit Food 52](#_Toc502786220)

[3.2.8 Add as a Volunteer : 53](#_Toc502786221)

[3.2.9 Fill Volunteer form 54](#_Toc502786222)

[3.2.10 Handle Receiver Requests 55](#_Toc502786223)

[3.2.11 Add hunger spot 56](#_Toc502786224)

[3.2.12 View hunger spot 57](#_Toc502786225)

[3.3 Sequence diagrams: 58](#_Toc502786226)

[3.3.1 Sequence Diagram Login : 58](#_Toc502786227)

[3.3.2 Sequence Diagram DashBoard : 59](#_Toc502786228)

[3.3.3 Sequence Diagram Food Donor 60](#_Toc502786229)

[3.3.4 Sequence diagram-Receiver 61](#_Toc502786230)

[3.3.5 Sequence diagram-Complete 63](#_Toc502786231)

[3.3.6 Software architecture: 64](#_Toc502786232)

[3.4 Database Diagram: 65](#_Toc502786233)

[3.5 Class diagram: 68](#_Toc502786234)

[3.6 Gantt Chart: 69](#_Toc502786235)

[3.7 Collaboration Diagram: 70](#_Toc502786236)

[4 Chapter 4 System Testing 71](#_Toc502786237)

[4.1 Test Cases 71](#_Toc502786238)

[4.1.1 Account registration 71](#_Toc502786239)

[4.1.2 Application Login 72](#_Toc502786240)

[*4.1.3* Donate Food 72](#_Toc502786241)

[4.1.4 Receive Food 73](#_Toc502786242)

[4.1.5 Volunteer 73](#_Toc502786243)

[4.1.6 Add as a Volunteer 74](#_Toc502786244)

[4.1.7 Hunger Spot 74](#_Toc502786245)

[4.2 Unit Testing : 75](#_Toc502786246)

[4.3 Integration testing: 75](#_Toc502786247)

[5 Chapter 5: Conclusion 76](#_Toc502786248)

[5.1 Problems faced and lessons learned: 76](#_Toc502786249)

[5.2 Future Work: 77](#_Toc502786250)

[5.3 Project Summary: 78](#_Toc502786251)

[5.4 Prototypes 79](#_Toc502786252)

**List of Tables**

[Table 1 Stakeholders list 13](#_Toc502786254)

[Table 2 User Registration 14](#_Toc502786255)

[Table 3 User Login 14](#_Toc502786256)

[Table 4 Manage User Profile 15](#_Toc502786257)

[Table 5 Donate Food 15](#_Toc502786258)

[Table 6 Receive Food 16](#_Toc502786259)

[Table 7 Volunteer 16](#_Toc502786260)

[Table 8 Hunger Spot 17](#_Toc502786261)

[Table 9 Alerts 18](#_Toc502786262)

[Table 10 Online Supported Client Platforms 18](#_Toc502786263)

[Table 11Online User Groups 18](#_Toc502786264)

[Table 12 Google Maps 19](#_Toc502786265)

[Table 13 Security 19](#_Toc502786266)

[Table 14 Usability 19](#_Toc502786267)

[Table 15Maintainability 20](#_Toc502786268)

[Table 16 Reliability 20](#_Toc502786269)

[Table 17 Efficiency 20](#_Toc502786270)

[Table 18 Portability 20](#_Toc502786271)

[Table 19 Interoperability 20](#_Toc502786272)

[Table 20 Availability 20](#_Toc502786273)

[Table 21 2.3.3 Requirement Traceability Matrix 27](#_Toc502786274)

[Table 22 Login 28](#_Toc502786275)

[Table 23 Logout 29](#_Toc502786276)

[Table 24 Add Food 30](#_Toc502786277)

[Table 25 Update Food 31](#_Toc502786278)

[Table 26 Delete Food 32](#_Toc502786279)

[Table 27 View Food 33](#_Toc502786280)

[Table 28 Request Food 34](#_Toc502786281)

[Table 29 Add as a Volunteer 35](#_Toc502786282)

[Table 30 Fill the volunteer form 36](#_Toc502786283)

[Table 31 Volunteer Handle Receiver Requests 37](#_Toc502786284)

[Table 32 View Hunger Spot 38](#_Toc502786285)

[Table 33 Add Hunger Spot 39](#_Toc502786286)

[Table 34 Account registration 71](#_Toc502786287)

[Table 35 Application Login 72](#_Toc502786288)

[Table 36 Donate Food 72](#_Toc502786289)

[Table 37 Receive Food 73](#_Toc502786290)

[Table 38 Volunteer 73](#_Toc502786291)

[Table 39 Add as a Volunteer 74](#_Toc502786292)

[Table 40 Hunger Spot 74](#_Toc502786293)

**List of Figures**

[Figure 1 Donate Food User 40](#_Toc502786296)

[Figure 2 Receive Food User 41](#_Toc502786297)

[Figure 3 Volunteer 42](#_Toc502786298)

[Figure 4 Hunger Spot User 43](#_Toc502786299)

[Figure 5 Waterfall Model 44](#_Toc502786300)

[Figure 6 Work breakdown structure 45](#_Toc502786301)

[Figure 7 Login 46](#_Toc502786302)

[Figure 8 Logout 47](#_Toc502786303)

[Figure 9 Add Food 48](#_Toc502786304)

[Figure 10 Delete Food 49](#_Toc502786305)

[Figure 11 View Food 50](#_Toc502786306)

[Figure 12 Request Food 51](#_Toc502786307)

[Figure 13 Edit Food 52](#_Toc502786308)

[Figure 14 Add as a Volunteer 53](#_Toc502786309)

[Figure 15 Fill Volunteer form 54](#_Toc502786310)

[Figure 16 Handle Receiver Requests 55](#_Toc502786311)

[Figure 17Add hunger spot 56](#_Toc502786312)

[Figure 18 View hunger spot 57](#_Toc502786313)

[Figure 19 Sequence Diagram Login 58](#_Toc502786314)

[Figure 20 Sequence Diagram DashBoard 59](#_Toc502786315)

[Figure 21 Sequence Diagram Food Donor 60](#_Toc502786316)

[Figure 22 Sequence diagram-Receiver **Error! Bookmark not defined.**](#_Toc502786317)

[Figure 23 Sequence diagram-Complete 63](#_Toc502786318)

[Figure 24 Software architecture 64](#_Toc502786319)

[Figure 25 DataBase Diagram 67](#_Toc502786320)

[Figure 26 Class diagram 68](#_Toc502786321)

[Figure 27 Gantt Chart 69](#_Toc502786322)

[Figure 28 Collaboration Diagram 70](#_Toc502786323)

# Abstract

This earth is our home. Humans are our family members. We cannot let anyone **sleep hungry**. Let’s change the world! This project is an android application which is a platform for the people to donate daily routine extra food which they wasted in no manners. Our aim is to reduce the food waste and feed the hungry people. People can use this application to donate their foods or to collect food for the hungry people. Millions of lives are starving for food around us, have a Tap to feed a life. Let’s work together to bridge the gap and reduce hunger and food waste in our beloved country.

# Introduction

FOOD DONOR is an android application that provides opportunities to people to donate food easily even they don’t need to search a hungry one. This app allows a layman to use technology for their ease in an efficient way. Integrated maps will be used in this application which automatically gets the location of the person who wants to donate food and display the location of the person who request for the food. Using location data, the app matches between the donor and a nearby organization to collect food. This app allows a layman to use technology for their ease in an efficient way. User can share the food to available Charity services. User can find Charities near to the location and can deliver them self, user will get the root map assistance to charity which will easy to reach the location. Otherwise all the user need to do is to upload the food images in the app and immediately all the volunteers will get notification and will contact you to pick up the food and serve to charity, the user can also track the status of food up to delivery. Those who ready to act as a volunteer can register as a volunteer in the app itself and can collect food from different donors and serve to Charity. This app includes 4 major modules Donate Food, Receive Food, Volunteer and Hunger Spot.

## Objectives:

The Goal of our project is to implement such a system, which is expected to be:

* Electronically efficient, less time consuming and free from complexities.
* Change culture of bagging system.
* Want to provide platform where people can easily donate/receive food.
* Helps Orphan Houses to request for food efficiently.
* Make the people to use the technology in a better way.

## Problem Statement:

There is no platform for the people where they can donate food. Basically people are promoting towards technology. That is the reason why Food Donor was introduced and keep on improving day by day. “Food Donation is quiet an issue nowadays for almost every individual.” Time is major issue for most of the people in today’s era. People nowadays are so busy that sometimes they do not have enough time to donate food and they waste their food in no sense. Food Donor will ease such problems for the people. It will provide them with the opportunity to donate and receive food quickly.

## Assumptions and constraints:

The project is chartered assuming that:

* An incorporated database framework will be outlined by us that will be administrated by the Ministry of Health who will keep record of each needy person who request for food.
* Access to people personal information (includes phone number, CNIC etc.)
* Constraints are that project will be under the policies of the university.

## Project Scope:

Mostly people when they want to donate some food they are in looking for some bagger to safe their food from waste while in restaurants they also waste food in daily routine. So, we want to safe the food waste and provide food to the hungry people and different food banks and different NGO’s where people are feed.

Internet and Smartphone is the major invention of this age and people are moving towards it very extensively so in that perspective our application will have the greatest scope in market. We are also using GPS system in our application which is the very fine feature of Smartphone so it also plays a major role in increasing the scope of our application.

Our project will be covering an Android application for android devices. GPS based location service will be used in our application to get the location of the donor and receiver will be provided to that person on the basis of its current location. Food donor and receiver services will be implementing in this application.

# Chapter 2: Requirements Analysis

## Literature Review:

There are many different ways to donate food and many different apps are playing their role in order to save the wastage of food.

There are many different ways to donate food that is used by the people. Usually people are in search of beggars whom which they give them extra food of daily routine. There are many people in Pakistan still wasting lots of food and there are also have some restaurants that waste food in no sense. We provide people a platform through our application where they can donate extra food of daily routine, and they make people happy by serving them food. In fast-moving time of today, when everyone is squeezed for time, the majority of people are fastidious when it comes to placing a food donate. Now-a-days people do not care of someone who is not afford food expenses and people should help them.

FOOD DONOR (online donating app) that is proposed here, greatly simplifies the request for food for both the Donor and the Receiver. This app allows a layman to use technology for their ease in an efficient way. App presents an interactive and up-to-date menu with all available food in an easy to use manner. This app will be user friendly, easy to use and makes the food donating procedure more time saving and efficient. Integrated maps will be used in this application which automatically gets the location of the donor/receiver and display the requests accordingly and user also able to check the Hunger Spots. Food availability is also an issue now-a-days in food banks which will be covered by this application. Receiver can request for 2 items per day as he wants and then the list of item will be updated. User can also view all the request details before checking out. In the end, Donor/Receiver gets request confirmation details. Once the request is placed it is entered in the database and retrieved in pretty much real time. This process will allow donor to quickly go through the requests as they are received and process all requests efficiently and effectively with minimum delays and disorientation.

## Stakeholders list:

|  |  |
| --- | --- |
| **Stakeholder** | Details |
| Donors | Donors are those who use the application to donate food and also get some information about NGO or Slum House. |
| Volunteers | Includes those people who are responsible to complete the Receivers requests and perform management related activities using the application. |

Table 1 Stakeholders list

## Requirements Elicitation:

### Functional Requirements

**FR01: User Registration**

|  |  |
| --- | --- |
| **Req. No** | **Functional Requirements** |
| FR01-01 | The app shall enable the user to Sign Up by providing his /her detailed information   * Name (Text Field) * Email (Text Field) * Mobile Number (Text Field) * Location (Button) * Address (Text Field) |
| FR01-02 | The framework might provide the appropriate error when the surrendered detail into Sign form is incorrect or not according to the format determined by the framework. |
| FR01-03 | The system shall generate and send a verification code to the user. |
| FR01-04 | After the verification user should get a message that his/her number has been verified successfully and the account has been created. |
| FR01-05 | The framework might store the client information into the database and provide related input for effective join into the framework |

Table 2 User Registration

**FR02: User Login**

|  |  |
| --- | --- |
| **Req. No.** | **Functional Requirements** |
| FR02-01 | The app shall enable the user to view the dashboard after login. |
| FR02-02 | The app shall enable the user to log in to the system by providing mobile number. Every user has a different authentication code. |
| FR02-03 | The system shall enable the user to press “Login” button to login to the application. |
| FR02-04 | The system shall enable the user to access all the other functionalities after login. |

Table 3 User Login

**FR03: Manage User Profile**

|  |  |
| --- | --- |
| **Req. No.** | **Functional Requirements** |
| FR03-01 | App shall enable the user to view his/her profile information. |
| FR03-02 | The app shall enable the user reset his/her information again. |
| FR03-03 | The app shall show the user his/her picture along with the name. |

Table 4 Manage User Profile

**FR04: Donate Food**

|  |  |
| --- | --- |
| **Req. No.** | **Functional Requirements** |
| FR04-01 | After login, the app must show the user their donate food option. |
| FR04-02 | The page must show the Heading and Title “Donate Food”. |
| FR04-03 | The app shall show the some information on the donate food. |
| FR04-04 | The app shall show button to donate food on click. |
| FR04-05 | The app must show the information fields about food on click  Donate food button. |
| FR04-06 | The app must show donation request summary in form of list after  Completing donation request. |
| FR04-07 | The app must show the notification to donor when someone works on his/her request. |
| FR04-08 | The app must show edit or delete buttons on food donation request to the donor. |

Table 5 Donate Food

**FR05: Receive Food**

|  |  |
| --- | --- |
| **Req. No.** | **Functional Requirements** |
| FR05-01 | After login, the app must show the user their Receive Food option. |
| FR05-02 | The page must show the Heading and Title “Receive Food”. |
| FR05-03 | The app shall show the list of all donation requests post by the donors in Receive Food. |
| FR05-04 | The app shall show “Pick Food” and “Contact to Volunteer” button on every nearby food request. |
| FR05-05 | The app must show map and “Get direction” button to collect food from donor on click “Pick Food” button. |
| FR05-06 | The app must send notification to donor when receiver click on  ‘Pick Food” button. |
| FR05-07 | The app must show the notification to nearby Volunteer when Receiver click on “Contact to Volunteer” button to complete his/her request. |
| FR05-08 | The app must expire food donation request after 12 hours or after day timings. |

Table 6 Receive Food

**FR06: Volunteer**

|  |  |
| --- | --- |
| **Req. No.** | **Functional Requirements** |
| FR06-01 | After login, the app must show the user their Volunteer option. |
| FR06-02 | The page must show the Heading and Title “Volunteer”. |
| FR06-03 | The app shall show the list of all Volunteers. |
| FR06-04 | The app must show “Add as Volunteer” button to every user of app  Except who already registered as Volunteer. |
| FR06-05 | The app must show call button on every list on Volunteer. |
| FR06-06 | The app must send notification to donor when Volunteer work receiver’s  Request. |
| FR06-07 | The app must show the Edit button to Volunteer when he/she wants to edit  His/her profile. |

Table 7 Volunteer

**FR07: Hunger Spot**

|  |  |
| --- | --- |
| **Req. No.** | **Functional Requirements** |
| FR07-01 | After login, the app must show the user their Hunger Spot option. |
| FR07-02 | The page must show the Heading and Title “Hunger Spot”. |
| FR07-03 | The app shall show map on which Hunger spots are visible to any user. |
| FR07-04 | The app must show “Add Hunger Spot” button on map to give details about some NGO or Slum house. |
| FR07-05 | The app must take details of NGO or Slum house on Click button from the user who want to add some Hunger Spot on map. |
| FR07-06 | The app must show information about NGO or Slum house when someone click on the target location on map. |
| FR07-07 | The app must show nearby Slum houses or NGOs to user on map who want to donate some food himself or get some information about that place. |

Table 8 Hunger Spot

**FR8: Alerts**

|  |  |  |
| --- | --- | --- |
| **Req. No.** | **Functional Requirements** | |
| FR8-01 | App shall show a notification with the name Alerts on Home screen of mobile of the user. | |
| FR8-02 | The app form  1. | will have different alerts along with some readings in tabular  Alerts (Low, Full, Half, ) |
|  | 2. | Description |
|  | 3. | Date |
|  | 4. | Time |
|  | 5. | Status |
| FR8-03 | The Alerts, Date, Time and Status show to the Volunteer as a Remainder  When some food request is near to expire. | |
| FR8-04 | The notification show to all user when some food request shall post by donor. | |
| FR8-05 | The donors shall receive alerts from users when someone shall start work on related person request | |

Table 9 Alerts

**FR9: Online Supported Client Platforms**

Food Donor is an android-based application and it can be accessed via the internet through platforms like Android Mobile Phones and Android Tablets.

|  |  |
| --- | --- |
| **Req. No.** | **Functional Requirements** |
| FR9-01 | The following list of versions shall be supported by the Android-  Application   * Android 4.x (Kit Kat) * Android 5.x (Lollipop) * Android 6.x (Marshmallow) * Android 7.x (Nougat) |

Table 10 Online Supported Client Platforms

**FR10: Online User Groups**

Food Donor Android application can be used and accessed by certain users.

|  |  |
| --- | --- |
| **Req. No.** | **Functional Requirements** |
| FR10-01 | The following list of users shall be entitled to use the system   * Any Household person * NGO Owners * Restaurant Owners * Slum House Owners * Drivers as a Volunteer |

Table 11Online User Groups

**FR11: Google Maps**

|  |  |
| --- | --- |
| **Req. No.** | **Functional Requirements** |
| FR11-01 | App shall show the user a Google Maps when user want to get direction of food request. |
| FR11-02 | The user will be able to click on the map to enlarge it. |
| FR11-03 | The map will show the location of the Food request by donor. |
| FR11-04 | User can click on the map to zoom in or out and see where the sensor actually exists. |
| FR11-05 | User can close the map to use other functionality of the app. |

Table 12 Google Maps

### Non-Functional Requirements

**NFR01: Security**

|  |  |
| --- | --- |
| NFR01-01 | Just approved clients might have the capacity to get to the framework through login module. |
| NFR01-02 | Framework should not give get to any client aside from the assigned client to refresh his/her information. |

Table 13 Security

**NFR02: Usability**

|  |  |
| --- | --- |
| NFR02-01 | Framework might be simple to handle and explore in the most expected path with no delays |
| NFR02-02 | It should take less the 1 hour of backward utilize for another client to become a specialist client of the application. |

Table 14 Usability

**NFR03: Maintainability**

|  |  |
| --- | --- |
| NFR03-01 | The App design is being finished in view of measured quality with the goal that practicality should be possible effectively. |
| NFR03-02 | So when items i.e. change in menu, change in functionalities, etc. will be refreshed with time then it will even now be kept up. |

Table 15Maintainability

**NFR04: Reliability**

|  |  |
| --- | --- |
| NFR04-01 | The crash rate of this application is 1% so it is reliable most of the time. |

Table 16 Reliability

**NFR05: Efficiency**

|  |  |
| --- | --- |
| NFR05-01 | Any android smartphone (works best with quad-core processor or above). |
| NFR05-02 | RAM 1 GB (Minimum). |
| NFR05-03 | Storage 50 MB (Minimum) |
| NFR05-04 | Installed GPS system |
| NFR05-05 | Running Internet connection |

Table 17 Efficiency

**NFR06: Portability**

|  |  |
| --- | --- |
| NFR06-01 | Android application will be developed for the android smartphones. |

Table 18 Portability

**NFR07: Interoperability**

|  |  |
| --- | --- |
| NFR07-01 | This whole system is connected to the central database so required data is easily available for all devices compatible with this system. |

Table 19 Interoperability

**NFR08: Availability**

|  |  |
| --- | --- |
| NFR08-01 | The framework might be up and running all day. |

Table 20 Availability

### Requirement Traceability Matrix

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ID** |  | **FR**  **-ID** | **FR-Description** | **Project**  **Objective** | **WB**  **S ID** | **Pro**  **duct Desi gn** | **Te st**  **Ca**  **se ID** | **Prio**  **rity** | **Verifi cation** | **Softw are**  **Modul**  **e** |
|  |  |  | **Android Based**  **Requirements** |  |  |  |  |  |  |  |
|  | **1.** | **FR**  **01-**  **01** | The application should empower the client to Sign Up by providing his/her point by point information | User Registration |  |  | T  C0  1 | H | Compl  eted | Registration |
|  | **2.** | **FR**  **01-**  **02** | The framework should provide the appropriate error when the surrendered detail into Sign form is incorrect or not according to the format indicated by the framework. | User Registration |  |  | T  C0  1 | H | Compl  eted | Registration |
|  | **3.** | **FR**  **01-**  **03** | The system shall generate and send a verification code to the user. | User Registration |  |  | T  C0  1 | H | Compl  eted | Registration |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | user in his/her details |  |  | CD-  01 |  |  |  |  |
| **4.** | **FR**  **01-**  **04** | After the verification user should get a message that his/her number has been verified successfully and the account has been created. | User Registration |  |  | T  C0  1 | H | Compl  eted | Registration |
| **5.** | **FR**  **01-**  **05** | The framework might store the client information into the database and provide related input for fruitful join into the framework. | User  Registration |  |  | T  C0  1 | H | Compl  eted | Registration |
| **6.** | **FR**  **02-**  **01** | The app shall enable the user to view the dashboard after login. | User Login |  |  | T  C0  2 | H | Compl  eted | Login |
| **7.** | **FR**  **02-**  **02** | The application should empower the client to log in to the framework by providing mobile number. Each client has an alternate authentication code. | User Login |  |  | T  C0  2 | H | Compl  eted | Login |
| **8.** | **FR**  **02-**  **03** | The system should give access to the user to press the ‘Login’ button to login into the application. | User Login |  |  | T  C0  2 | H | Compl  eted | Login |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | |  | |  |  | | |  | | | |  | | | |  | | | |  | |
| **9.** | **FR**  **02-**  **04** | App shall enable the user to view his/her profile information. | | | Manage User Profile |  |  | T  C0  3 | | | | H | | | | Compl  eted | | | | Profile Management | | | |
| **10** | **FR**  **02-**  **05** | The app shall enable the user to reset his/her information again. | | | Manage User Profile |  |  | | T  C0  3 | | | | H | | | | Compl  eted | | | | Profile Management | | |
| **11** | **FR**  **03-**  **01** | The app shall show the user his/her picture along with the name. | | | Manage User Profile |  |  | | | | T  C0  3 | | | | H | | | | Compl  eted | | | | Profile  Management | |
| **12** | **FR**  **04-**  **01** | After login, the app must show the user their donate food option. | | | Donate Food |  |  | | | | T  C0  4 | | | | M | | | | Compl  eted | | | | Donate Food | |
| **13** | **FR**  **04-**  **02** | | The page must show the Heading and Title “Donate Food”. | | Donate Food |  |  | | | | T  C0  4 | | | | M | | | | Compl  eted | | | | Donate Food | |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **14** | **FR**  **04-**  **03** | The app shall show the some information on the donate food. | Donate Food |  |  | T  C0  4 | M | Compl  eted | Donate Food |
| **15** | **FR**  **-**  **04-**  **04** | The app shall show button to donate food on click. | Donate Food |  |  | T  C0  4 | H | Compl  eted | Donate Food |
| **16** | **FR**  **-**  **04-**  **05** | The app must show the information fields about food on click Donate food buton. | Donate Food |  |  | T  C0  4 | H | Compl  eted | Donate Food |
| **17** | **FR**  **-**  **04-**  **06** | The app must show donation request summary in form of list after completing donation request. | Donate Food |  |  | T  C0  4 | H | Compl  eted | Donate Food |
| **18** | **FR**  **-**  **04-**  **07** | The app must show the notification to donor when someone works on his/her request. | Donate Food |  |  | T  C0  4 | H | Compl  eted | Donate Food |
| **19** | **FR**  **04-**  **08** | The app must show edit or delete buttons on food donation request to the donor. | Donate Food |  |  | T  C0  4 | H | Compl  eted | Donate Food |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  | |  | |
| **20** | **FR**  **-**  **05-**  **03** | The app shall show the list of all donation requests post by the donors in Receive Food. | Receive Food |  |  | T  C0  5 | H | Compl  eted | | | Receive Food |
| **21** | **FR**  **-**  **05-**  **04** | The app shall show “Pick Food” and “Contact to Volunteer” button on every nearby food request. | Receive Food |  |  | T  C0  5 | H | Compl  eted | | | Receive Food |
| **22** | **FR**  **-**  **05-**  **05** | The app must show map and “Get direction” button to collect food from donor on click “Pick Food” button. | Receive Food |  |  | T  C0  5 | H | Compl  eted | | | Receive Food |
| **23** | **FR**  **-**  **05-**  **06** | The app must send notification to donor when receiver click on  ‘Pick Food” button. | Receive Food |  |  | T  C0  5 | M | | Compl  eted | | Receive Food |
| **24** | **FR**  **-**  **05-**  **07** | The app must show the notification to nearby Volunteer when Receiver click on “Contact to Volunteer” button to complete his/her request. | Receive Food |  |  | T  C0  5 | H | Compl  eted | | | Receive Food |
| **25** | **FR**  **-**  **05-**  **08** | The app must expire food donation request after 12 hours or after day timings. | Receive Food |  |  | T  C0  5 | H | Compl  eted | | | Receive Food |
| **26** | **FR**  **-**  **06-**  **03** | The app shall show the list of all Volunteers. | Volunteers |  |  | T  C0  6 | M | Compl  eted | | | Volunteers |
| **27** | **FR**  **-**  **06-**  **04** | The app must show “Add as Volunteer” button to every user of app except who already registered as volunteer. | Volunteers |  |  | T  C0  6 | M | Compl  eted | | | Volunteers |
| **28** | **FR**  **-**  **06-**  **05** | The app must show call button on every list on Volunteer. | Volunteers |  |  | T  C0  6 | M | Compl  eted | | | Volunteers |
| **29** | **FR**  **-**  **06-**  **06** | The app must send notification to donor when Volunteer work receiver’s request. | Volunteers |  |  | T  C0  6 | M | Compl  eted | | | Volunteers |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **30** | **FR**  **-**  **07-**  **04** | The app must show the Edit button to Volunteer when he/she wants to edit his/her profile | Hunger Spot |  |  | T  C0  7 | M | Compl  eted | Hunger Spot |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **31** | **FR**  **-**  **07-**  **05** | The app must show “Add Hunger Spot” button on map to give details about some NGO or Slum house. | Hunger Spot |  |  | T  C0  7 | M | Compl  eted | Hunger Spot |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **32** | **FR**  **-**  **07-**  **06** | The app must take details of NGO or Slum house on Click button from the user who want to add some Hunger Spot on map | Hunger Spot |  |  | T  C0  7 | M | Compl  eted | Hunger Spot |

Table 21 2.3.3 Requirement Traceability Matrix

## Use case description:

### Login:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Use Case Id: Login** | | | | |
| **Users :** Data Entry User | | | | |
| **Use Case Summary :**  It is made to check whether clients are legitimate/valid or not. | | | | |
| **Pre-Condition :**  Client should have record of use and client must enter the right Number and code. | | | | |
| **Scenarios :** | | | | |
|  | **Steps** | **Flow** | **Alternative** |  |
| 1 | At the point when an actor opens the application, the 'Login dialogue box' will show up. |  |
| 2 | Actor will enter his phone number. |  |
| 3 | A code will be send and the actor will login successfully. | On the off chance that client number is correct then principle page will open else invalid message will show up. |
| **Alternative Scenario:**  On the off chance that client number is correct then principle page will open else invalid message will show up. | | | | |
| **Post-Condition:**  Therefore of login, if the client number did not coordinate then primary page will open else Error message will showed onscreen. | | | | |

Table 22 Login

### Logout:

|  |  |  |
| --- | --- | --- |
| **Use Case Id: Logout** | | |
| **Users:** Data Entry User | | |
| **Use Case Summary:**  It is made to sign out from application. | | |
| **Pre-Condition:**  User should be logged in. | | |
| **Scenarios:** | | |
| **Step#** | **Flow** | **Alternative** |
| 1 | User click the ‘Log out’ button |  |
| 2 | User will be automatically logout | If the user press ‘log out’ then message will appear on screen ‘Log out successfully’ and if user click no then user will logged in. |
| **Alternative Scenario:**  If the user press ‘log out’ then message will appear on screen ‘Log out successfully’ and if user click no then user will logged in. | | |
| **Post-Condition:**  Accordingly of logout, fundamental page of application will be close and login dialog box will show up on primary page of application. | | |

Table 23 Logout

### Add Food:

|  |  |  |
| --- | --- | --- |
| **Use Case Id: Add Food** | | |
| **Users:** Donor | | |
| **Use Case Summary:**  It is made to include food in the database. | | |
| **Pre-Condition:**  User should be signed in into the system. | | |
| **Scenarios:** | | |
| **Step#** | **Flow** | **Alternative** |
| 1 | User select the choice for Add Food. |  |
| 2 | User fill the required fields |  |
| 3 | Click ‘submit’ button | On the off chance that the expected fields to include food fill accurately at that point food will be included else error message will show up. |
| **Alternative Scenario:**  On the off chance that the expected fields to include food fill accurately at that point food will be included else error message will show up. | | |
| **Post-Condition:**  Because of included food, food item will be included into database and discourse message box will show up on screen with message 'Food included Successfully'. | | |

Table 24 Add Food

### Update Food:

|  |  |  |
| --- | --- | --- |
| **Use Case Id: Update Food** | | |
| **User :** Donor | | |
| **Use Case Summary:**  It is made to refresh/alter food in the database. | | |
| **Pre-Condition:**  User should be signed in into the system. | | |
| **Scenarios:** | | |
| **Step#** | **Flow** | **Alternative** |
| 1 | User select the alternative for refresh food |  |
| 2 | User will alter or change already filled fields |  |
| 3 | Click ‘submit’ button | In the event that the expected fields to refresh food fill effectively then data in the database will be refreshed else error message will show up. |
| **Alternative Scenario:**  In the event that the expected fields to refresh food fill effectively then data in the database will be refreshed else error message will show up. | | |
| **Post-Condition:**  Because of refresh food data ,the alter food will be included into database and discourse message box will show up on screen with message 'Food refresh Successfully'. | | |

Table 25 Update Food

### Delete Food:

|  |  |  |
| --- | --- | --- |
| **Use Case Id: Delete Food** | | |
|  | | |
| **Users:** Donor | | |
| **Use Case Summary:**  It is made to remove food from the database. | | |
| **Pre-Condition:**  User should be signed in into the system. | | |
| **Scenarios:** | | |
| **Step#** | **Flow** | **Alternative** |
| 1 | User will tap the food which he needs to erase. |  |
| 2 | User will click the ‘delete’ button |  |
| 3 | Click ‘yes or no’ button to confirm. | On the off chance that the client squeeze yes at that point food will be erased from database else it won't. |
| **Alternative Scenario:**  On the off chance that the client squeeze yes at that point food will be erased from database else it won't. | | |
| **Post-Condition:**  Because of erase food, Donor will erase previous food data effectively from the database and exchange message box will show up on screen with message 'Food Deleted effectively'. | | |

Table 26 Delete Food

### View Food:

|  |  |  |
| --- | --- | --- |
| **Use Case Id: View Food** | | |
| **Users:** Donor, Receiver | | |
| **Use Case Summary:**  It is made to view food from the end database. | | |
| **Pre-Condition:**  User should be signed in into the system. | | |
| **Scenarios:** | | |
| Step# | Flow | Alternative |
| 1 | Donor/Receiver will tap the food which he wants to view. |  |
| 2 | User will click the ‘view food’ button | User will press view button to view food otherwise food will not be viewed. |
| **Alternative Scenario:**  User will press view button to view food otherwise food will not be viewed. | | |
| **Post-Condition:**  When User will view food, then the information about the food will be viewed. | | |

Table 27 View Food

### Request Food:

|  |  |  |
| --- | --- | --- |
| **Use Case Id: Request Food** | | |
| **Users:**  Receiver | | |
| **Use Case Summary:**  Made to request food by the Receiver. | | |
| **Pre-Condition:**  User should be signed in into the system. | | |
| **Scenarios** | | |
| **Step#** | **Flow** | **Alternative** |
| 1 | User tap the ‘Request Food’ option. |  |
| 2 | User will request for food. |  |
| 3 | Click ‘submit’ button | If the user will request for food and submit it then request will be proceeded otherwise request will not be proceed. |
| **Alternative Scenario:**  If the user will request for food and submit it then request will be proceeded otherwise request will not be proceed. | | |
| **Post-Condition:**  When Receiver will request food, the request will be added in the request catalogue. | | |

Table 28 Request Food

**Add as a Volunteer:**

|  |  |  |
| --- | --- | --- |
| **Use Case Id: Add as a Volunteer** | | |
| **Users:** Volunteer | | |
| **Use Case Summary:**  It is made to check separate volunteers section. | | |
| **Pre-Condition:**  User should be signed in and press volunteer button. | | |
| **Scenarios:** | | |
| **Step#** | **Flow** | **Alternative** |
| 1 | User will press the ‘Volunteer section’ button. |  |
| 2 | User will click on add as a volunteer button. |  |
| 3 | User will receive a form. | If the required actions fill correctly to add as a volunteer then user will see the volunteer form otherwise not. |
| **Alternative Scenario:**  If the required actions fill correctly to add as a volunteer then user will see the volunteer form otherwise not. | | |
| **Post-Condition:**  When User will click the button, then he will showed the volunteer form on screen. | | |

Table 29 Add as a Volunteer

### Fill the volunteer form:

|  |  |  |
| --- | --- | --- |
| **Use Case Id: Fill the volunteer form** | | |
| **Users:** User | | |
| **Use Case Summary:**  It is made to save volunteers information. | | |
| **Pre-Condition:**  User should be signed and must press the fill volunteer form button. | | |
| **Scenarios:** | | |
| **Steps** | **Flow** | **Alternative** |
| 1 | User will press the ‘Fill volunteer form button. |  |
| 2 | User will fill all the expected fields in the form. |  |
| 3 | Click ‘submit’ button | In the event that the required fields fill effectively to include as a volunteer at that point client's information will be put away into the database generally mistake message will be shown. |
| **Alternative Scenario:**  In the event that the required fields fill effectively to include as a volunteer at that point client's information will be put away into the database generally mistake message will be shown. | | |
| **Post-Condition:**  When user fill all the necessary fields, then he will be added as a volunteer. | | |

Table 30 Fill the volunteer form

### Volunteer Handle Receiver Requests:

|  |  |  |
| --- | --- | --- |
| **Use Case Id: Volunteer Handle Receiver Requests** | | |
| **Users:** Volunteer | | |
| **Use Case Summary:**  It is made to handle receiver’s food requests stored in database. | | |
| **Pre-Condition:**  User should be signed in and see the receivers notifications. | | |
| **Scenarios:** | | |
| **Step#** | **Flow** | **Alternative** |
| 1 | Volunteer press the receive food button. |  |
| 2 | Volunteer see if any requests are there to handle. |  |
| 3 | If requests are there then user will proceed to them. | If requests are there then user will proceed to them, otherwise he will wait for the requests from the receivers if any. |
| **Alternative Scenario:**  If requests are there then user will proceed to them, otherwise he will wait for the requests from the receivers if any. | | |
| **Post-Condition:**  Volunteer will handle receiver’s requests. | | |

Table 31 Volunteer Handle Receiver Requests

### View Hunger Spot:

|  |  |  |
| --- | --- | --- |
| **Use Case Id: View Hunger Spot** | | |
| **Users:** Hunger Spot Users | | |
| **Use Case Summary:**  It is made to view the hunger spot in the application. | | |
| **Pre-Condition:**  User should be signed in and press Hunger Spot module. | | |
| **Scenarios:** | | |
| **Step#** | **Flow** | **Alternative** |
| 1 | User tap the ‘Hunger Spot’ module. |  |
| 2 | User tap view button to view the spot. |  |
| 3 | Press ‘ok’ | If the user proceed correctly to view hunger spot then spot will be viewed otherwise it will not be viewed. |
| **Alternative Scenario:**  If the user proceed correctly to view hunger spot then spot will be viewed otherwise it will not be viewed. | | |
| **Post-Condition:**  User has viewed the Hunger Spot. | | |

Table 32 View Hunger Spot

### Add Hunger Spot:

|  |  |  |
| --- | --- | --- |
| **Use Case Id: Add Hunger Spot** | | |
| **Users:** Hunger Spot Users | | |
| **Use Case Summary:**  It is made to add the hunger spot in the application. | | |
| **Pre-Condition:**  User should be signed in and press Hunger Spot module. | | |
| **Scenarios:** | | |
| **Step#** | **Flow** | **Alternative** |
| 1 | User press the ‘Hunger Spot’ module. |  |
| 2 | User press add button to add the spot. |  |
| 3 | Press ‘submit’ button. | If the user perform functions correctly to add hunger spot then spot will be viewed otherwise it error message will be displayed. |
| **Alternative Scenario:**  If the user perform functions correctly to add hunger spot then spot will be viewed otherwise it error message will be displayed. | | |
| **Post-Condition:**  User has added the Hunger Spot. | | |

Table 33 Add Hunger Spot

## Use Case Design:

### Donate Food User:

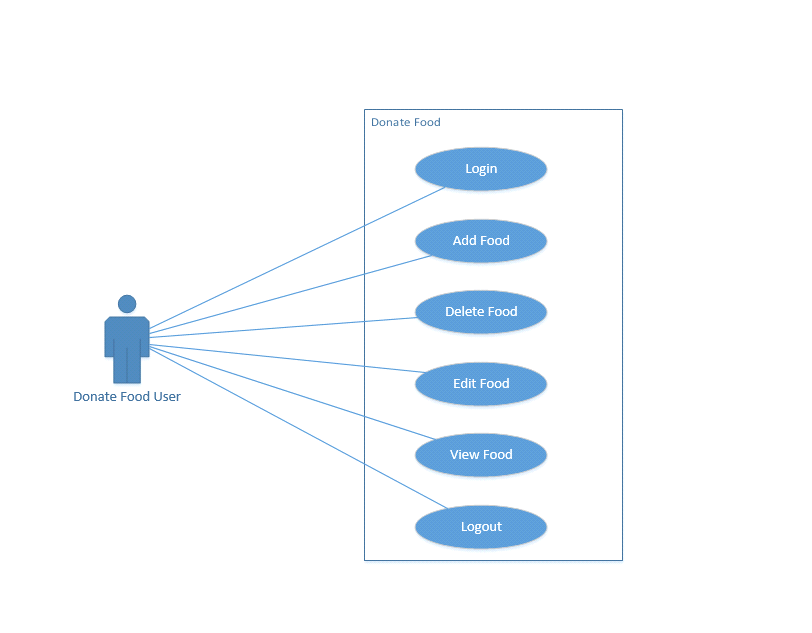


Figure 1 Donate Food User

**Receive Food User:**

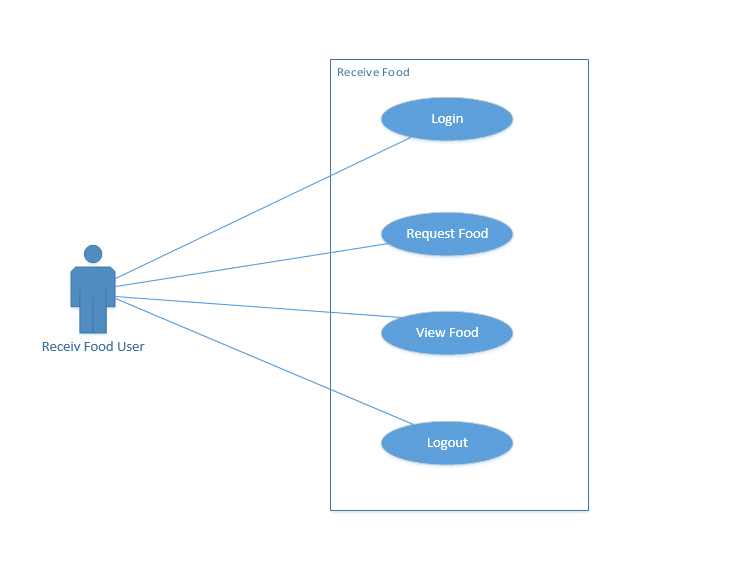


Figure 2 Receive Food User

**Volunteer:**

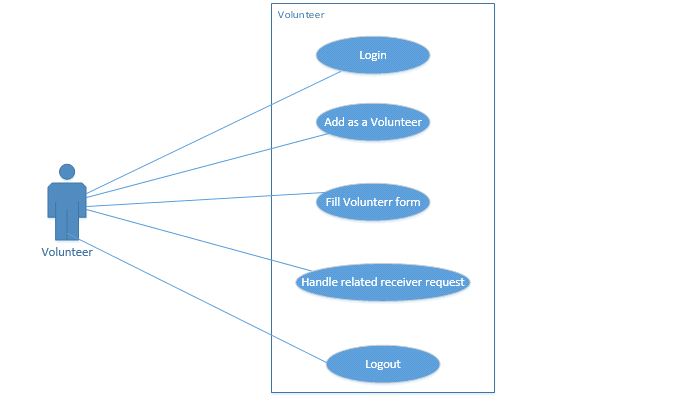


Figure 3 Volunteer

**Hunger Spot User:**

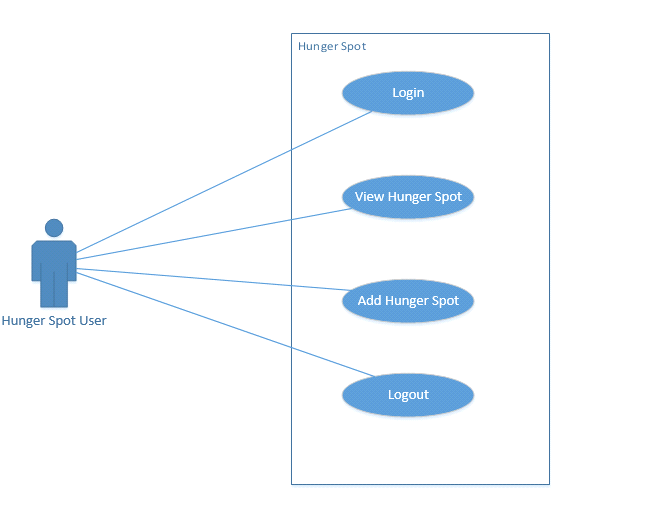


Figure 4 Hunger Spot User

## Software Development Life Cycle Model:

The software development life cycle which we utilized as a part of our project is the waterfall model.

Following are the reasons why we utilized this model:

* It is uncomplicated and reasonable to follow and use.
* The amount of resources required to actualize is negligible.
* In this model, output is created after each stage of implementation.
* It is extremely useful where quality is more favored than cost and plan.

### MODEL DIAGRAM

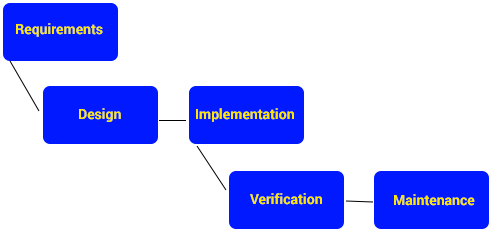


Figure 5 Waterfall Model

* The first step is to create functional requirements of the software. This is usually done by understanding the project very well.
* The second step is to create non-working prototype along with a user interface specifications.
* The third step is to implement non-working prototypes so that user can see the screens.
* The forth step is to install, test and debug the software.
* Last step is to provide maintenance to the software.

# Chapter 3: System Design:

## Work breakdown structure:

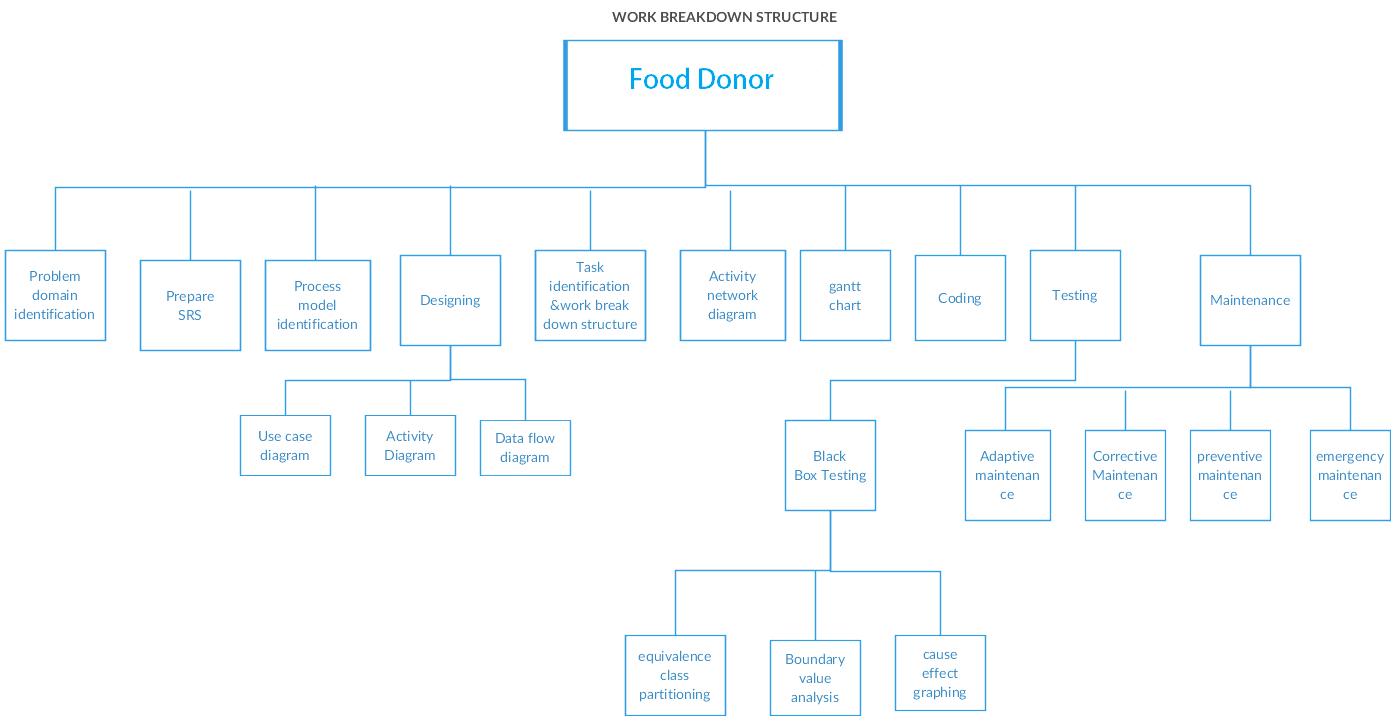


Figure 6 Work breakdown structure

## Activity diagrams:

### Login



Figure 7 Login

### Logout

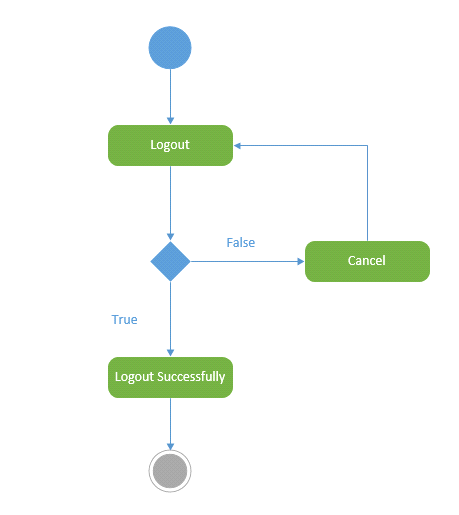


Figure 8 Logout

### Add Food



Figure 9 Add Food

### Delete Food

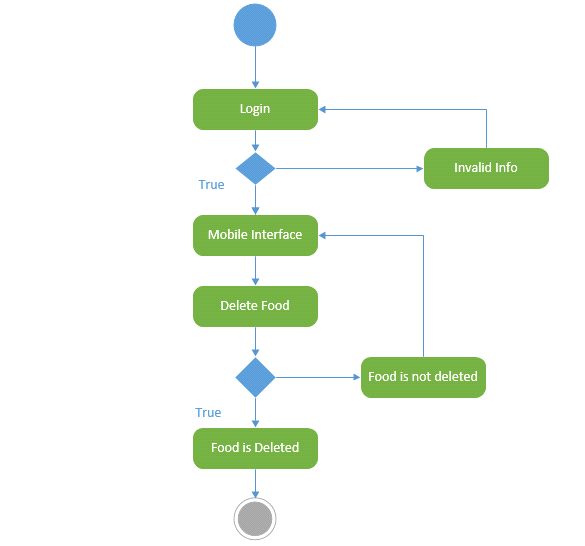


Figure 10 Delete Food

### View Food

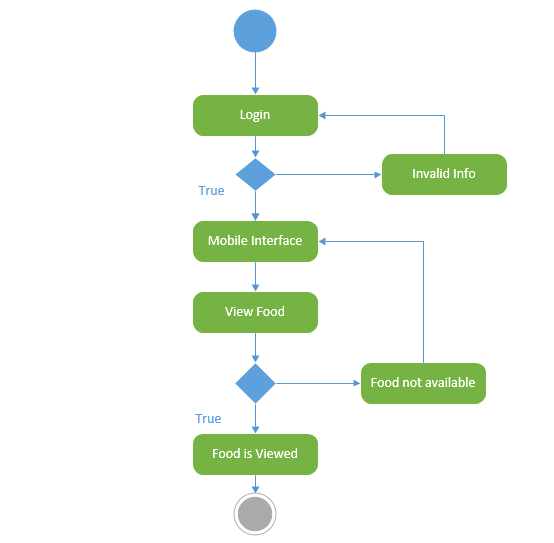


Figure 11 View Food

### Request Food

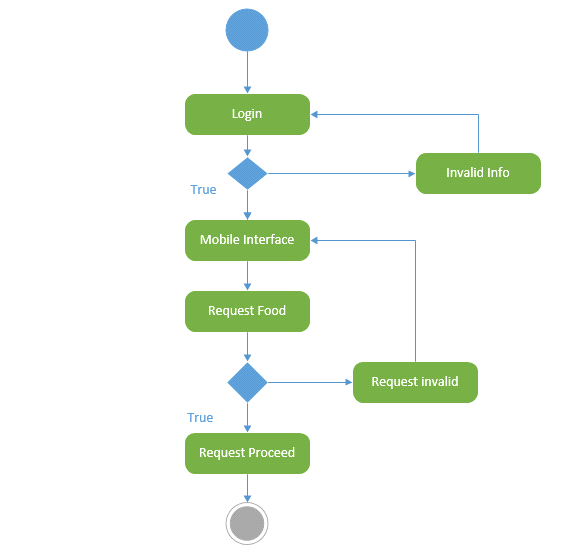


Figure 12 Request Food

### Edit Food

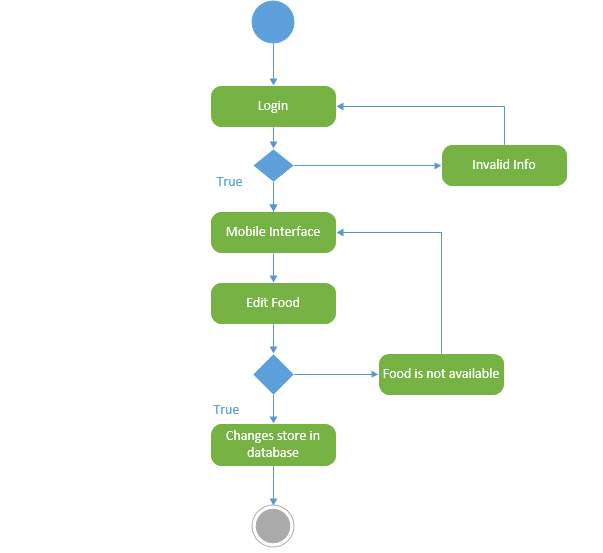


Figure 13 Edit Food

### Add as a Volunteer

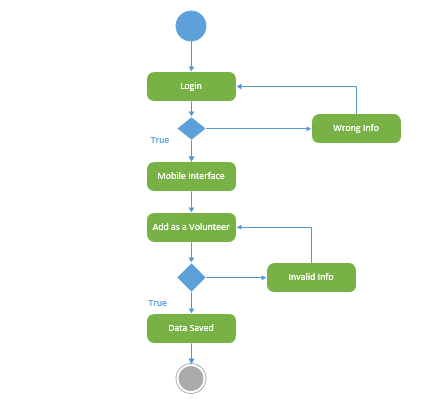


Figure 14 Add as a Volunteer

### Fill Volunteer form

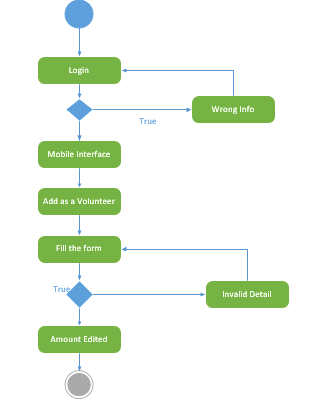


Figure 15 Fill Volunteer form

### Handle Receiver Requests

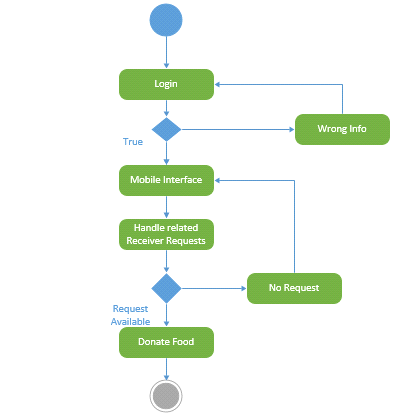


Figure 16 Handle Receiver Requests

### Add hunger spot

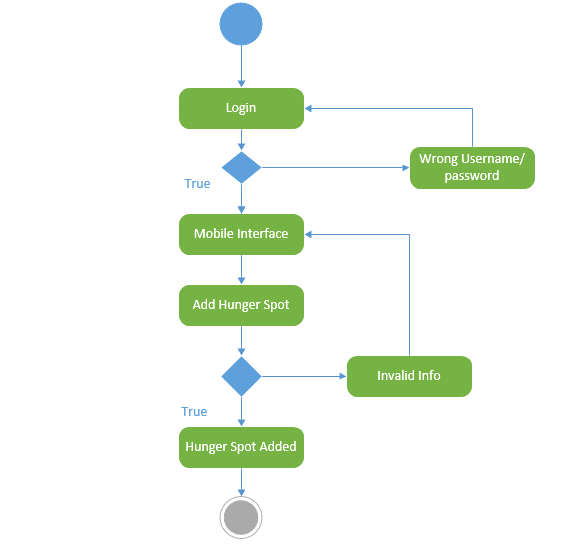


Figure 17Add hunger spot

### View hunger spot

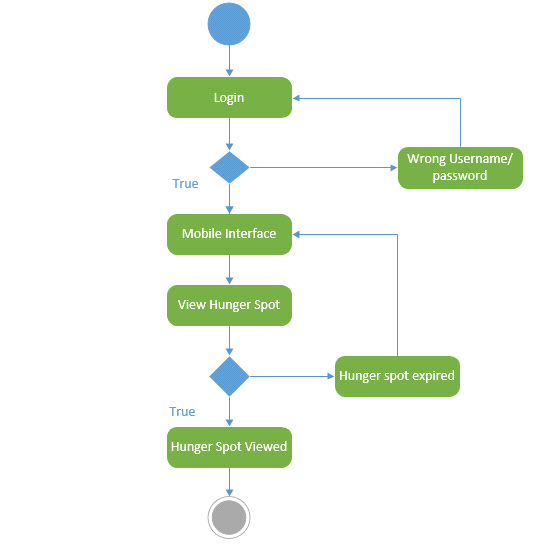


Figure 18 View hunger spot

## Sequence diagrams:

### Sequence Diagram Login:

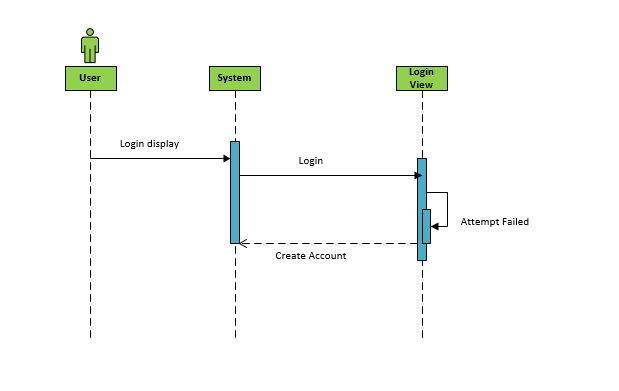


Figure 19 Sequence Diagram Login

### Sequence Diagram Dashboard:

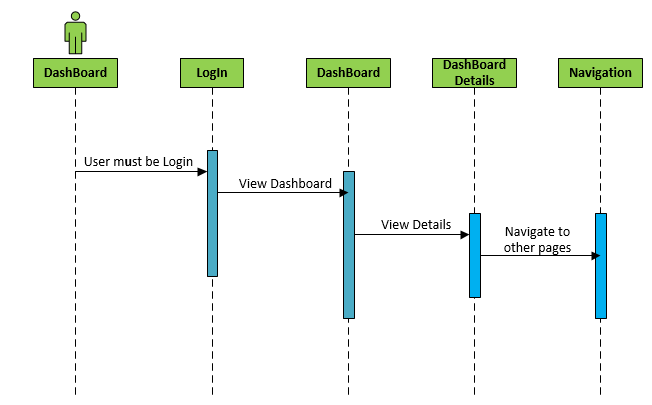


Figure 20 Sequence Diagram Dashboard

### Sequence Diagram Food Donor

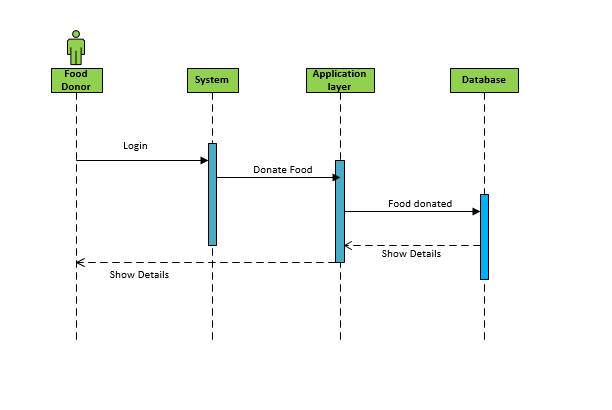


Figure 21 Sequence Diagram Food Donor

### Sequence diagram-Receiver

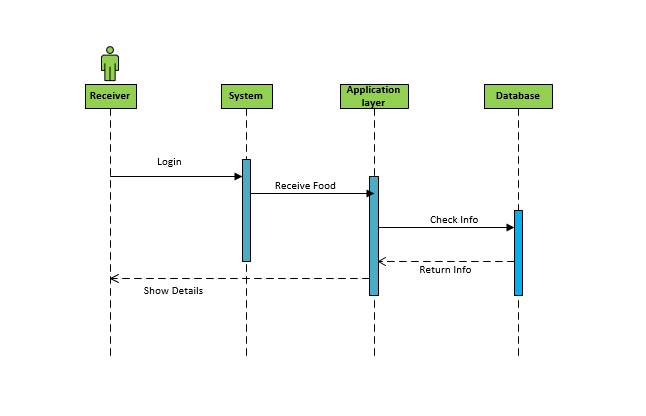


Figure 22 Sequence Diagram Receiver

### Sequence Diagram Volunteer

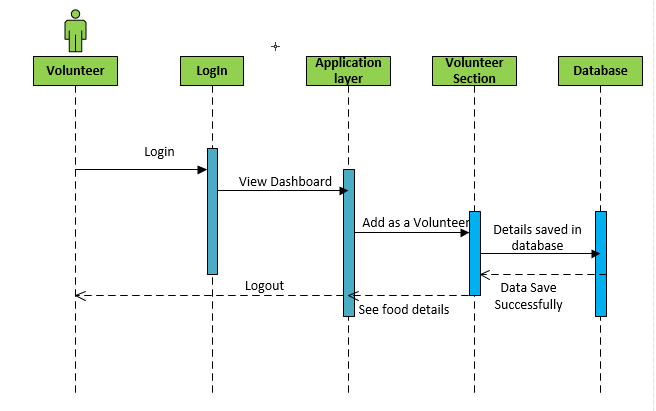


Figure 23 Sequence Diagram Volunteer

### Sequence diagram-Complete

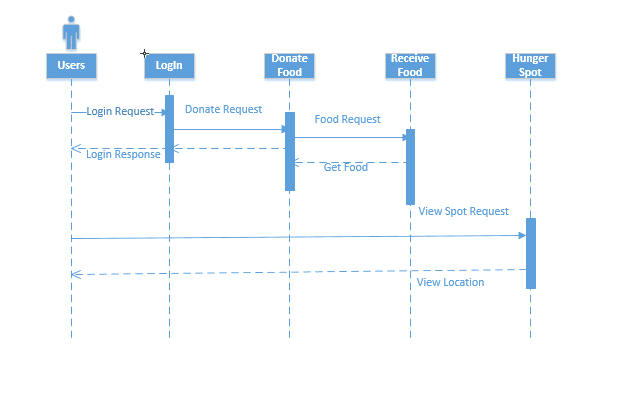


Figure 24 Sequence diagram-Complete

### Software architecture:

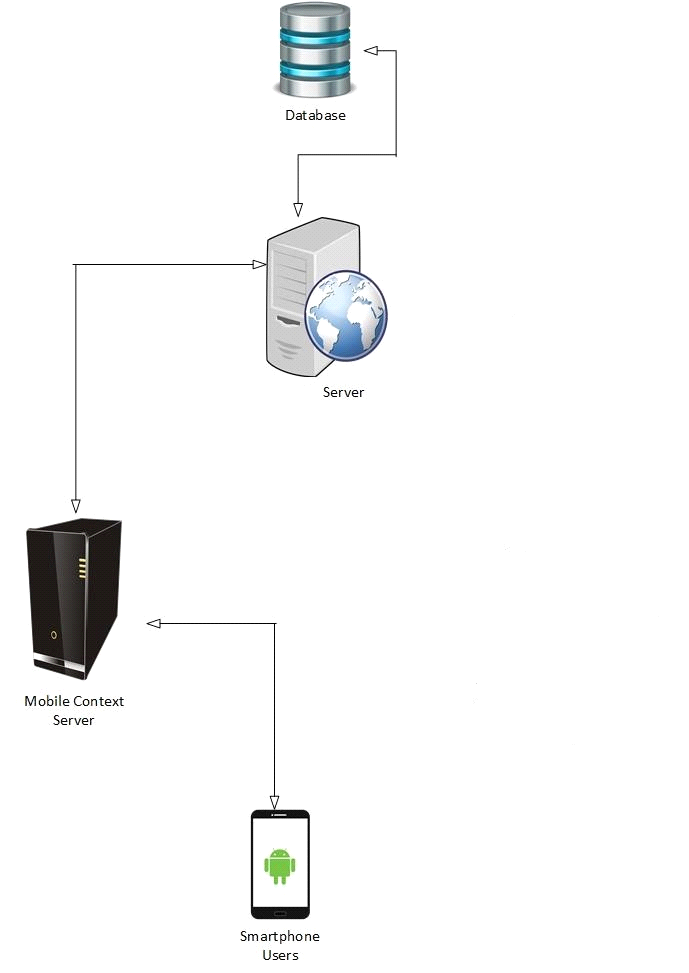
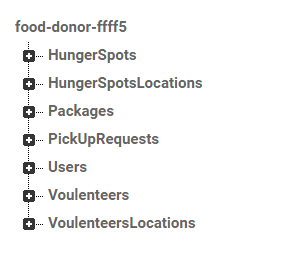
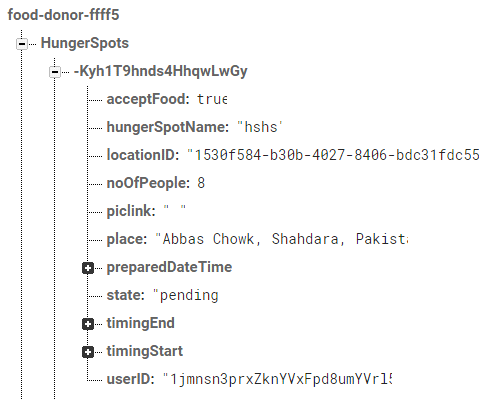


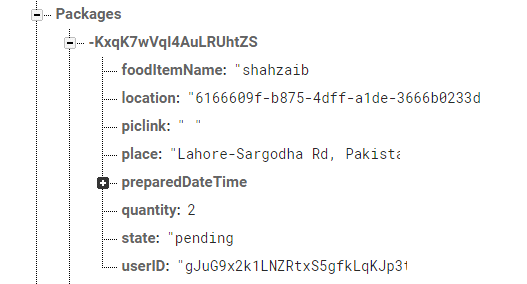
Figure 25 Software architecture

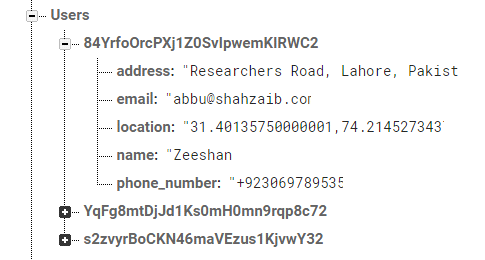
## Database Diagram:

All Firebase Real-time Database information is stored as JSON objects. You can think of the database as a cloud-hosted JSON tree. Not at all like a SQL database, there are no tables or records. At the point when you include information to the JSON tree, it becomes a node in the current JSON structure with an associated key.









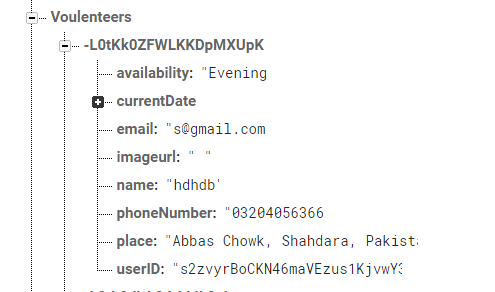
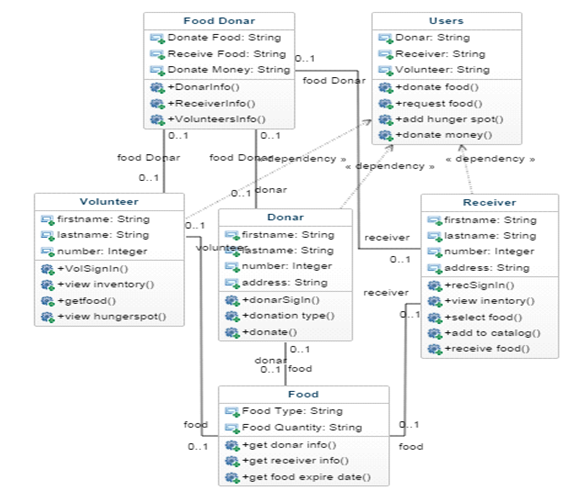


Figure 26 Database Diagram

## Class diagram:



*Figure 27 Class diagram*

## Gantt chart:

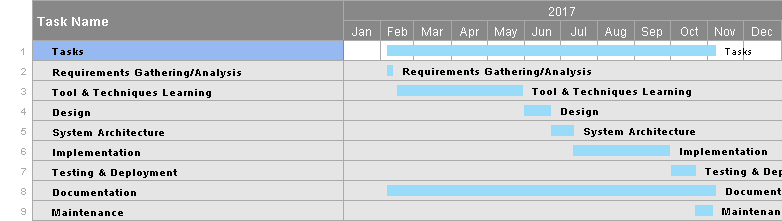


Figure 28 Gantt chart

## Collaboration Diagram:

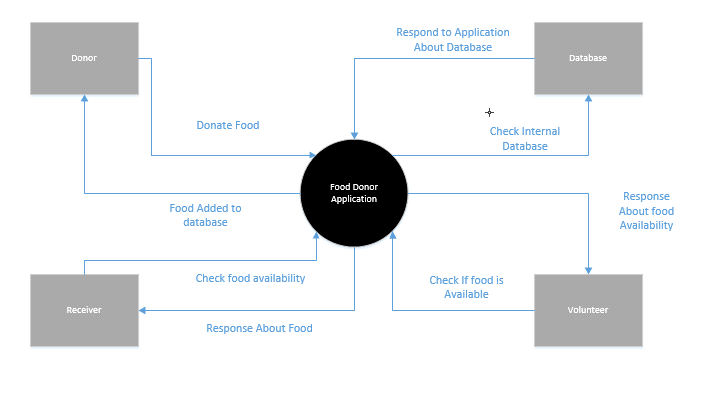


Figure 29 Collaboration Diagram

# Chapter 4 System Testing

## Test Cases

### Account registration

|  |  |
| --- | --- |
| Test case name | v-01 |
| Use case(s) | Account registration |
| Input summary | Name, phone, email, address |
| Output summary  Success:  User will be verified by Number & will get security number for verification. When the account is created successfully user will be directed to the login page.  Failure:  Error will occur if user already exists. | |
| Pre-conditions | User just installed the application |
| Post-conditions | Session for the user will be created |

Table 34 Account registration

### Application Login

|  |  |
| --- | --- |
| Test case name | v-02 |
| Use case(s) | Application Login |
| Input summary | Mobile Number |
| Output summary  Success:  System successfully authenticates the user.  Failure:  Because of wrong Number or format authentication will be denied. | |
| Pre-conditions | User already have an account |
| Post-conditions | User has the access to the functionality of the application |

Table 35 Application Login

### Donate Food

|  |  |
| --- | --- |
| Test case name | v-03 |
| Use case(s) | Donate Food |
| Input summary | User will click this button to add food |
| Output summary  Success:  User will be redirected to the new page to where status he wants to add food detail.  Failure:  Error will occur if user add something wrong. | |
| Pre-conditions | User is logged in. |
| Post-conditions | Food Stored to database will be displayed to the user. |

Table 36 Donate Food

### Receive Food

|  |  |
| --- | --- |
| Test case name | v-04 |
| Use case(s) | Receive Food |
| Input summary | User will click the button to get food |
| Output summary  Success:  User will be redirected to the receive food page.  Failure:  Error will occur if user didn’t add location. | |
| Pre-conditions | User never request for more than 2 food items. |
| Post-conditions | User request is updated |

Table 37 Receive Food

### Volunteer

|  |  |
| --- | --- |
| Test case name | v-05 |
| Use case(s) | Volunteer |
| Input summary | User will click the volunteer button. |
| Output summary  Success:  User will now see the volunteers list.  Failure:  User forget to press the volunteer button. | |
| Pre-conditions | User is on the dashboard |
| Post-conditions | User can view the volunteer group list. |

Table 38 Volunteer

### Add as a Volunteer

|  |  |
| --- | --- |
| Test case name | v-06 |
| Use case(s) | Add as a Volunteer |
| Input summary | User will click the volunteer button. |
| Output summary  Success:  User has successfully enter as a volunteer.  Failure:  User forget to press the volunteer button. | |
| Pre-conditions | User is on the dashboard |
| Post-conditions | User can view the volunteer group list. |

Table 39 Add as a Volunteer

### Hunger Spot

|  |  |
| --- | --- |
| Test case name | v-07 |
| Use case(s) | Hunger Spot |
| Input summary | User will click this button to add hunger spot and view hunger spot near to his location. |
| Output summary  Success:  Menu will be displayed to the user  Failure:  No menu will be displayed to user on current location | |
| Pre-conditions | User must be located in hunger spot that is registered in the application |
| Post-conditions | User can add hunger spot himself. |

Table 40 Hunger Spot

## Unit Testing:

The objective of unit testing is to check each & every component or code pieces of an application separately and check its working before running the whole application and testing it unit testing is performed to resolve the minor issues of an application. For example code for user login & code for admin login in an application is tested separately in this way both interfaces are verified correctly.

Components may be required to be disconnected from each other while unit testing.

## Integration testing:

The goal of integration testing is to test the modules in an application that are connected with each other and their functionality gives a mutual unique result. Integration testing is always done after the application is done with unit testing. In integration testing all requirements are checked according to documentation. For examples: order module and admin module are connected and they give a mutual output in result. Their connection is checked by integration testing and their result is analysed.

After unit testing, we continued incorporating units to give legitimate shape to item. After every unit coordinated, we performed joining testing on our framework. It was a period taking procedure and we did it precisely in light of the fact that it is essential piece of any venture. We had a few issues while performing it and incorporating it however we unravel our issues effectively.

# Chapter 5: Conclusion

## Problems faced and lessons learned:

**1. Creating an app that is different from existing apps:**

Due to immense competition in android application market it is a big hurdle for developers and idea generators to create an app that makes a difference. Designing an application that looks different, creative, have usable features and intuitive design. It should provide a wholesome experience.

**2. Dealing with android versions & device sizes:**

Users have devices of different sizes and software versions. Creating an app that targets only one version of android will be a biggest problem and will down rate the application in market. Keeping in view the limitations based on screen sizes, intensities, OS requirements.

**3. Making an interactive application:**

In every application interactions are equally important as responsive design. Making sensors in an application will be a plus however dealing with sensors will be challenging during design phase. Consider successful applications with sensors & extracting ideas from them will be a challenge.

**4. Performance vs. Battery usage:**

Making an interactive app with attractive design is one side but making an app that gives a fast performance & uses minimum battery is a big challenge for developers. Making an app that runs equally well on every version of device is a considerable thing for developers because most apps that work well on latest devices and have sensors may not work on old versions. Running test cases and prototyping is a good solution to this issue.

**5. Mobile content management:**

As content in the app is ever growing and every app includes images. Altogether content takes a lot of power & bandwidth. Wireless connectivity failures and frequent connection drops may disturb the content of application. Content must be simplified according to the need & reliable content must be available to get good performance from the application.

**6. Patent issues:**

Several android features may give rise to patent issues. So, using features and implementing ideas should be considered as a challenge before developing an application.

**7. Android market search engine:**

Android application market is ever growing and versions of android devices change rapidly there are more than 8 million apps and creating an app that is different & innovative in android market is a biggest challenge.

## Future Work:

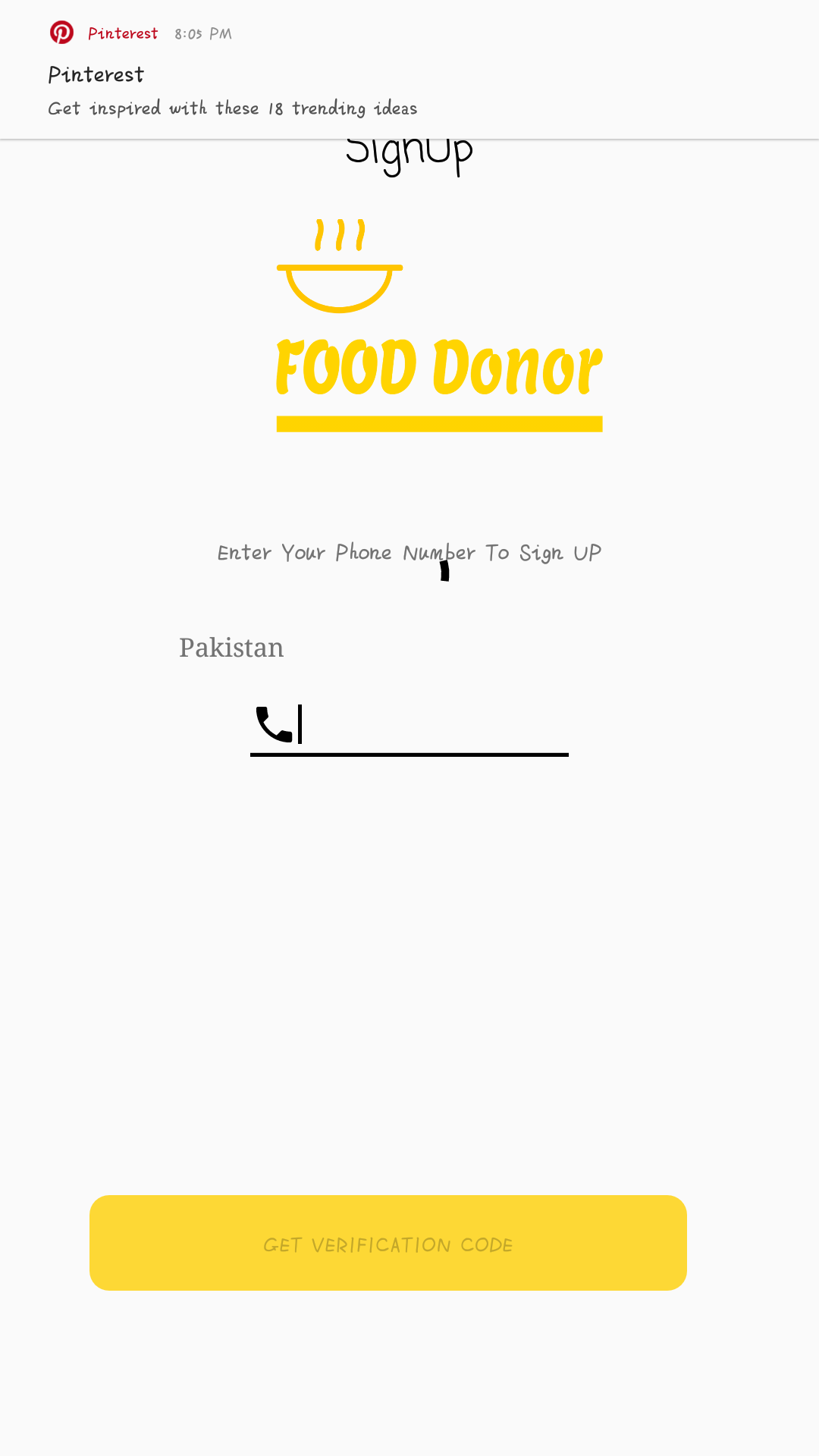
Our future work will be fixing a donating device on every food bank that will be controlled by the admin and each food bank will have its own record. Our future work will be fixing the problem of people to donate food. Each Donor or Receiver will have its own record. This will make managing the volunteers easier for people who are new in record. We will make Web application to provide more ease to the users. In future this app would also be available on IOS platform. More in our future work we'll be linking our application with social media platforms like Instagram & Facebook. This project will be leading many other food donating techniques and ideas.

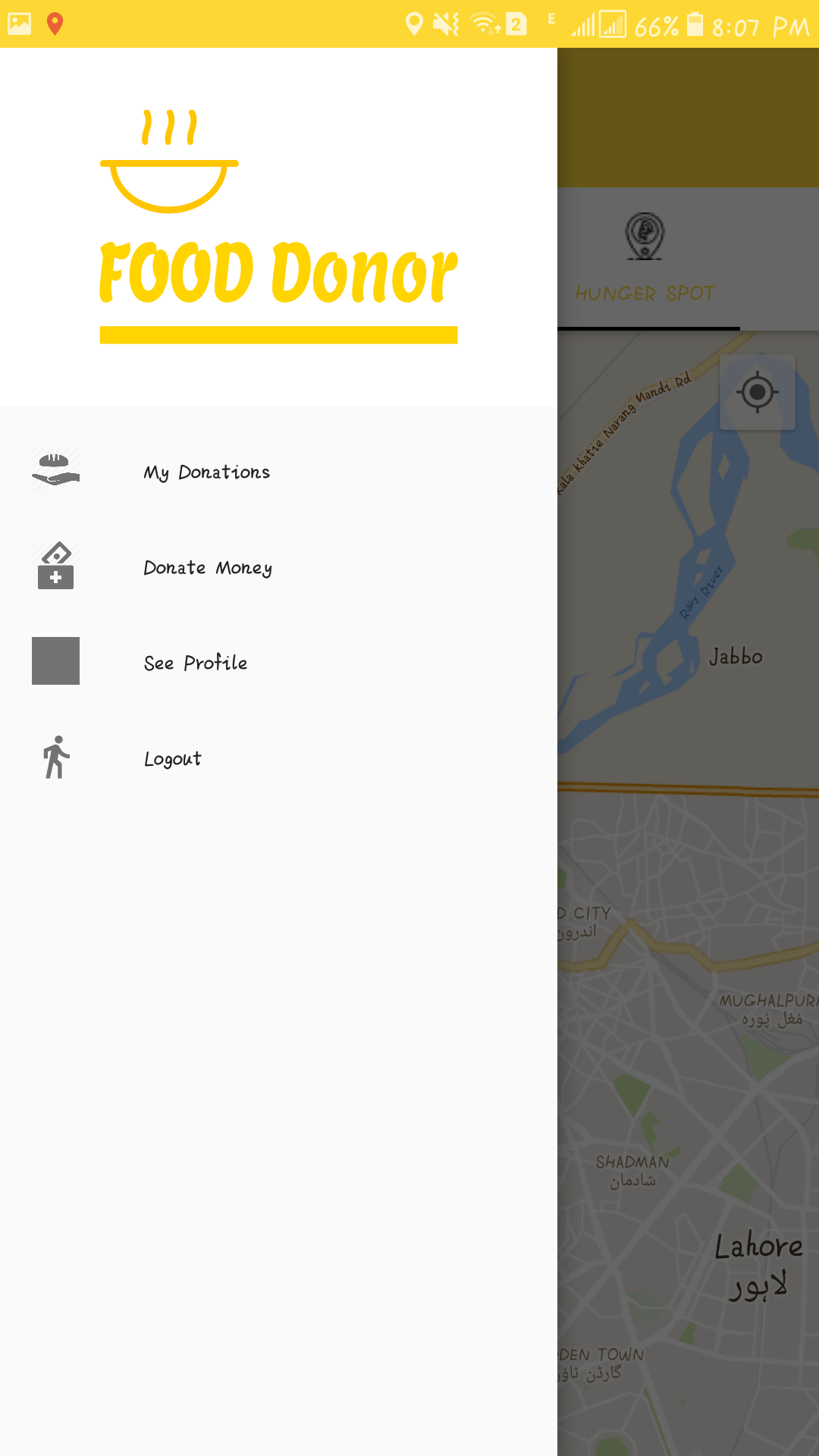
In future this app would also be available on IOS platform.

## Project Summary:

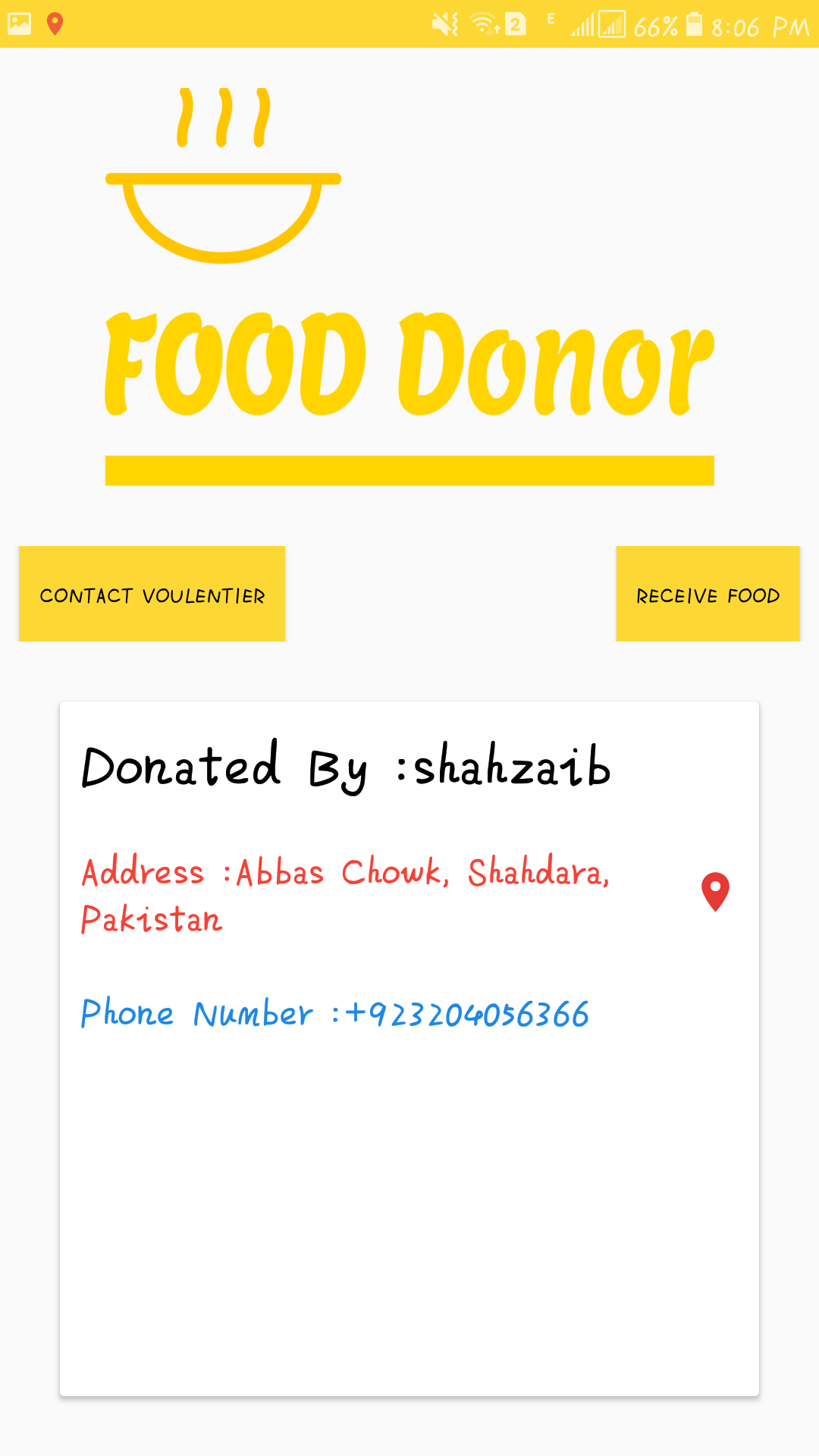
FOOD DONOR is an android application that provides opportunities to people to donate food easily even they don’t need to search a hungry one. This app allows a layman to use technology for their ease in an efficient way. Integrated maps will be used in this application which automatically gets the location of the person who wants to donate food and display the location of the person who request for the food. Using location data, the app matches between the donor and a nearby organization to collect food. This app allows a layman to use technology for their ease in an efficient way. As considering & analysing the ever-growing market of android applications and their usage this application will be a very helpful tool for food donation too. This application will be equally usable on both end i.e. donor and receiver.

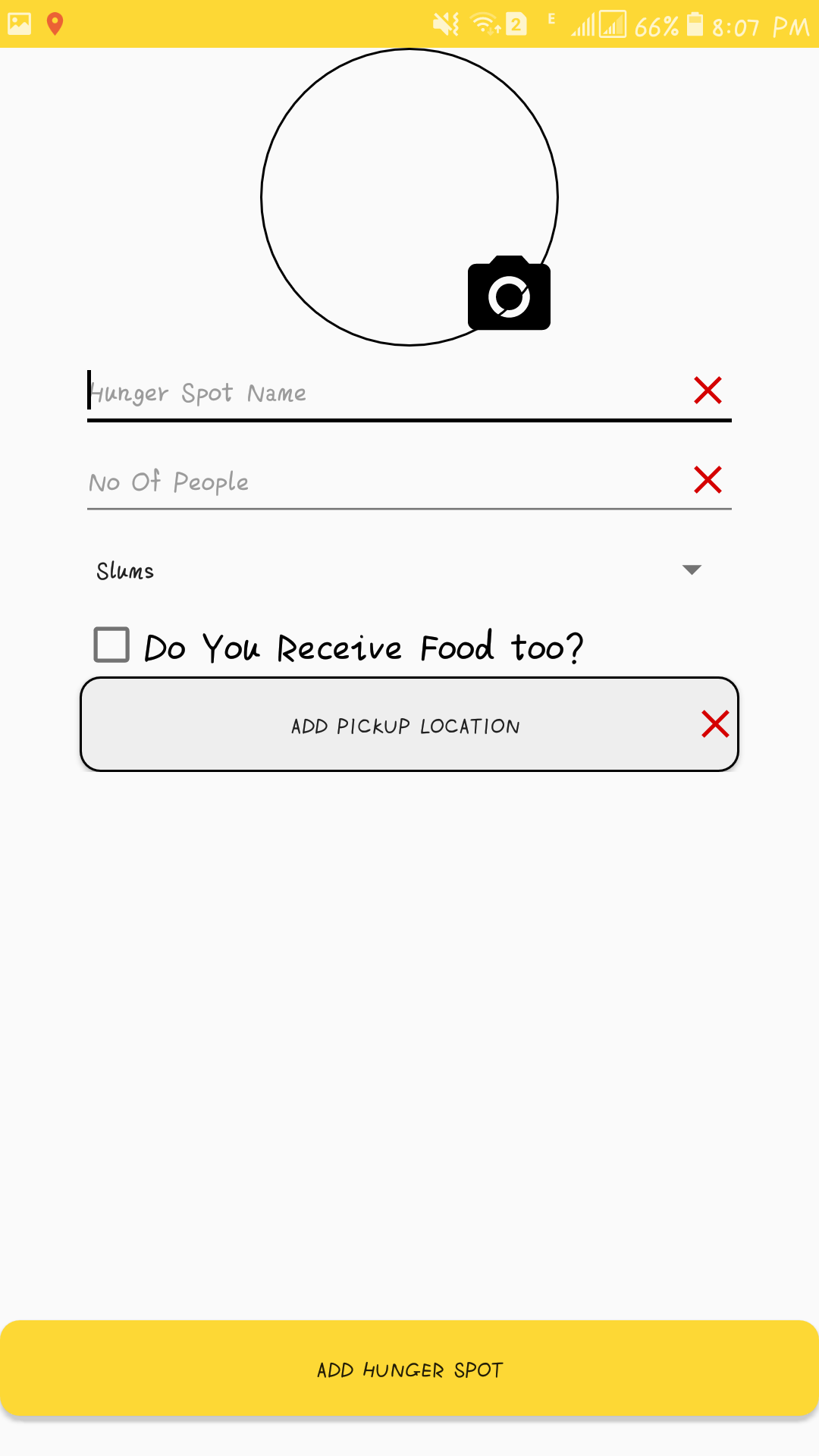
## Prototypes

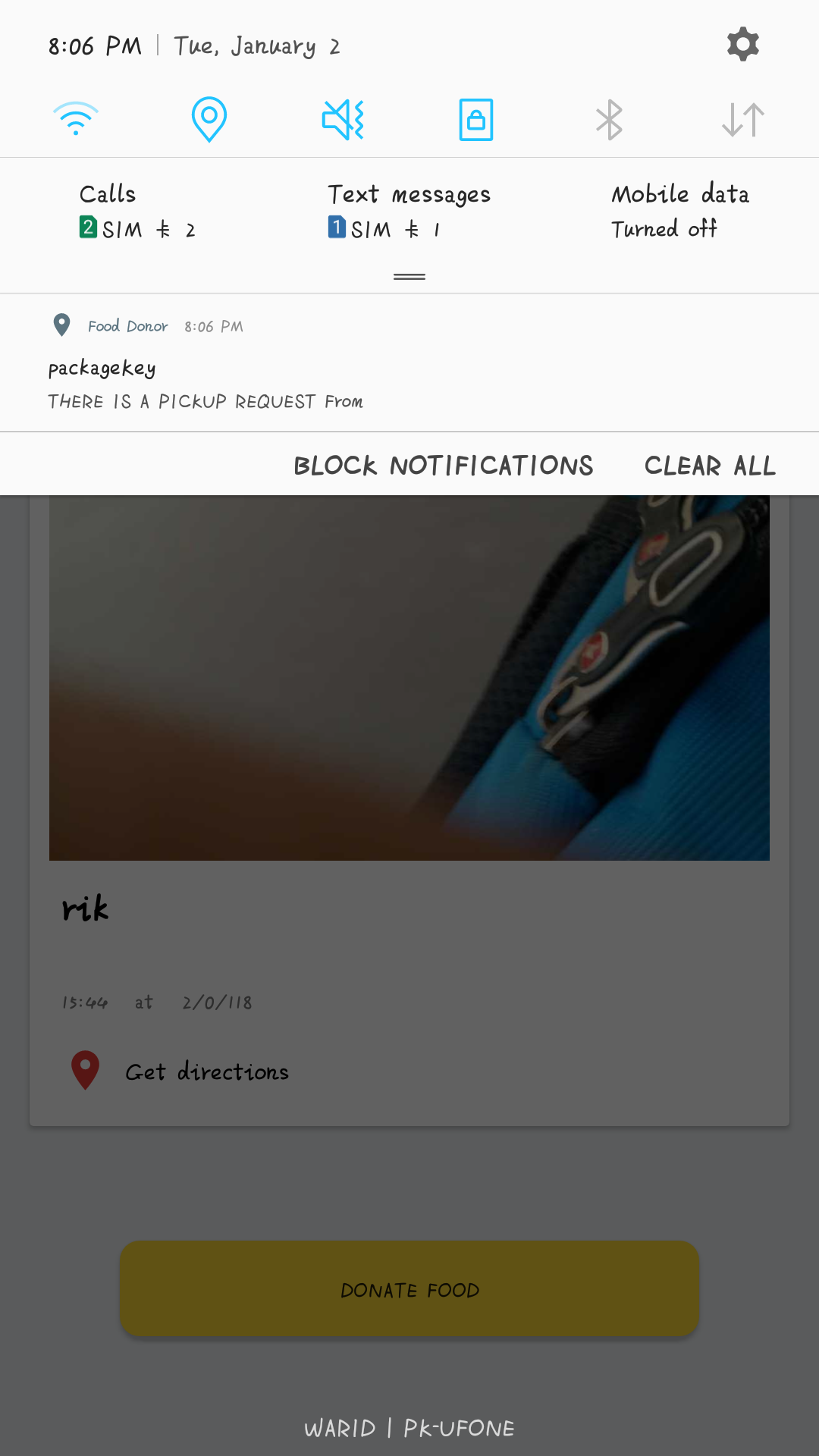


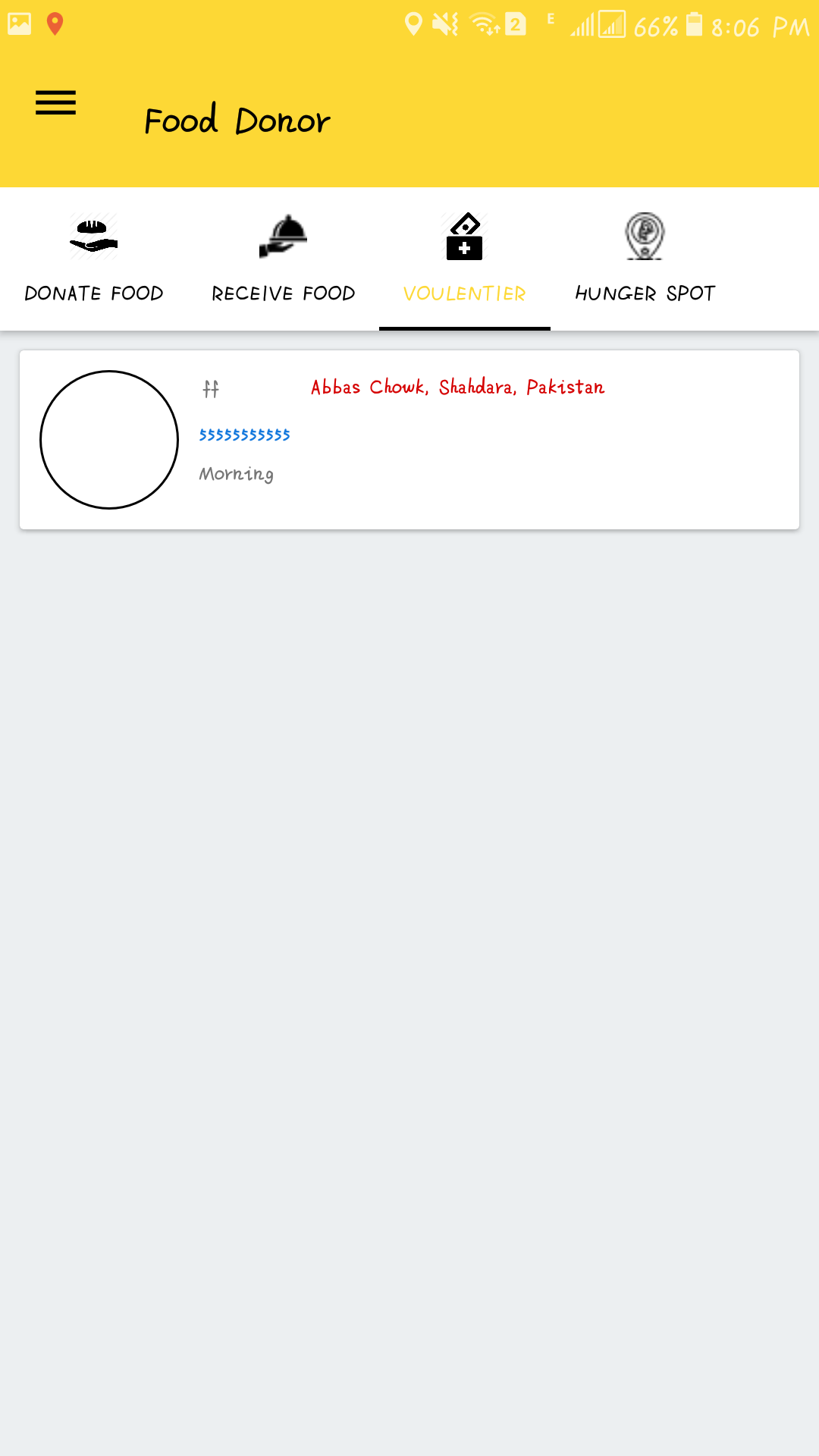


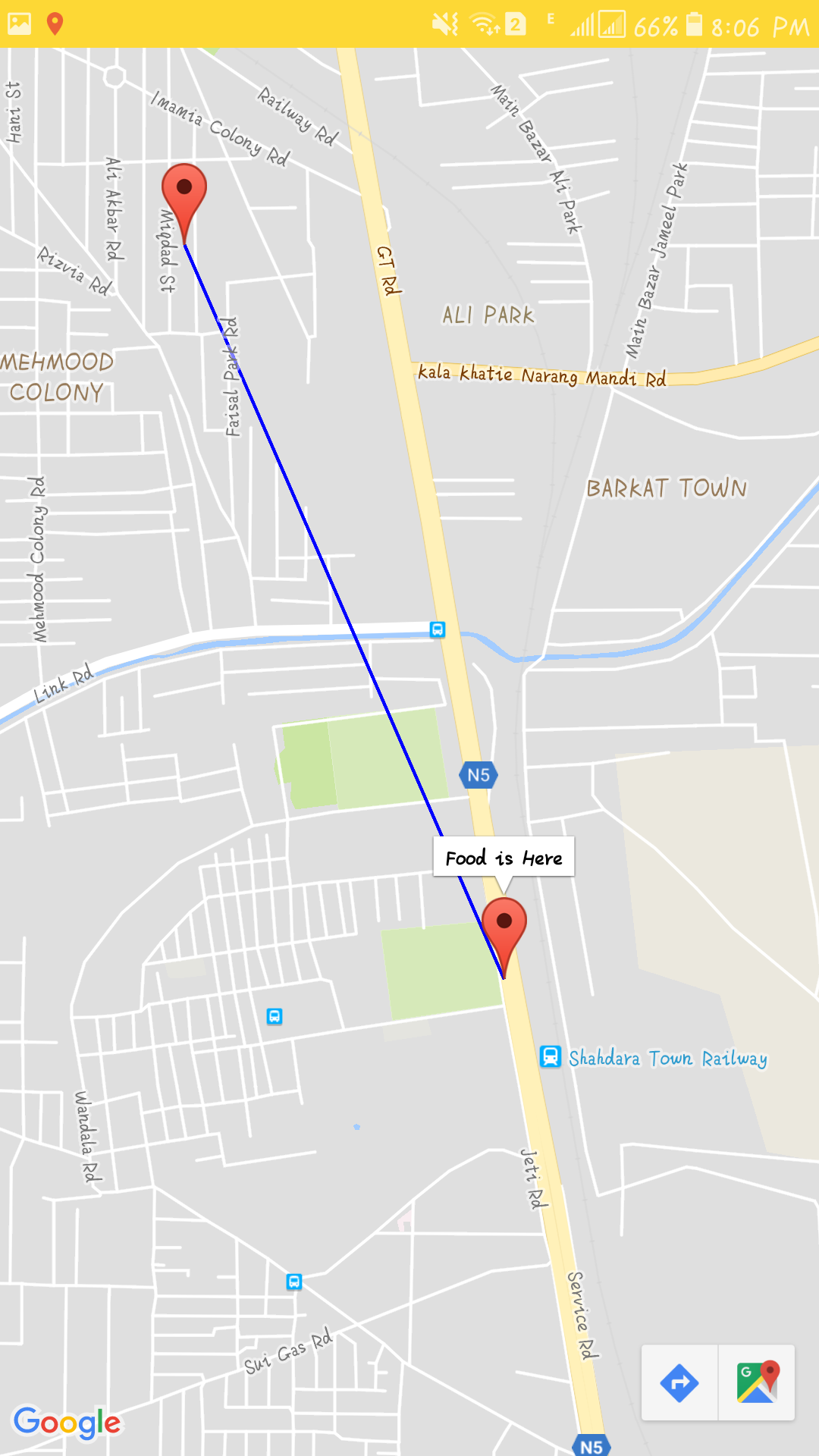
# C:\Users\TOSHIBA\Downloads\SEM 8\fyp\Prototypes\26613644_1194123844056251_96183038_o.png

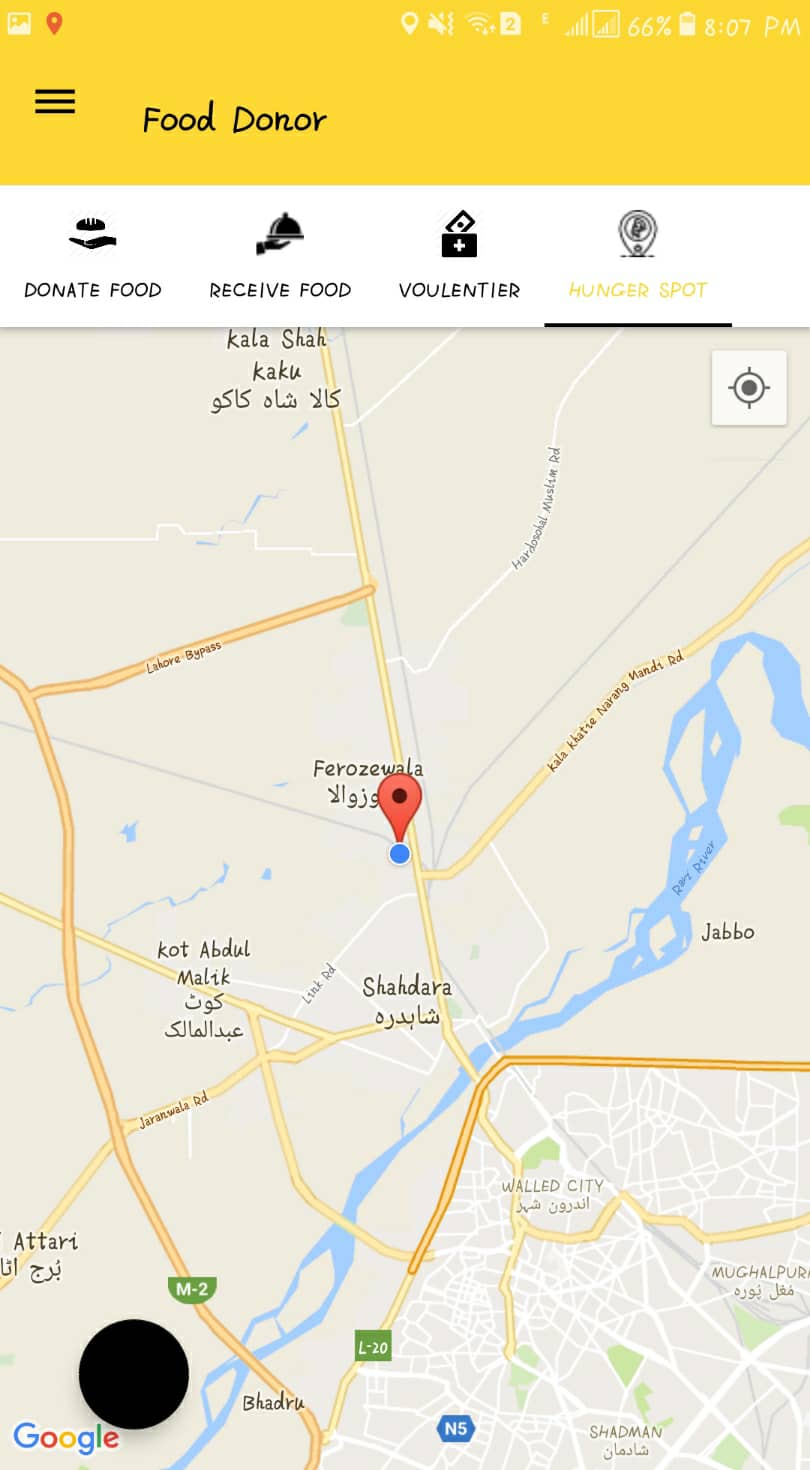












# References:

**Waste no food** releases mobile app for IOS and Android which is helping poor people to End Hunger in your community. They are providing three services donate, pick up or volunteer but not providing with location based services and interaction among different users to handle each other’s requests.

<http://wastenofood.org/>

**Zero Percent** aim to create a community system in order to help food businesses achieve ZERO food waste. The service will always be free for businesses and consumers eliminating food waste.

<https://www.zeropercent.us/>

**Surplus Food** is reducing waste to cut your grocery bills and help the planet.

<https://www.cnet.com/news/feeding-forward-app-delivers-food-to-homeless-shelters-in-real-time/>

**Food Donation Connection** manages food donation programs for food service companies interested in donating food. The donating process is based on donors receiving economic benefit through tax savings in addition to involvement with community and corporate goodwill.

<http://www.foodtodonate.com/>

**Food Bank** Australia works from paddock to plate, rescuing and sourcing food and groceries from farmers, manufacturers and retailers. Food bank then distributes these food and grocery items to front line charities around the country for people in need of food relief.

<https://www.foodbank.org.au/hunger-in-australia/hunger-report-2016/>

**COPIA** is an IOS based application for food donation. Use our [iOS app](https://appsto.re/us/oFB6_.i) or [website](https://app.gocopia.com/#/?init=signup) to schedule pickups of your surplus food. Prepare and package food for a seamless food recovery experience!

<https://www.gocopia.com/>

**Share the Meal** collects charity and makes a physical connection: between a regular daily activity and donating. They feed communities in specific areas like Yemen.

<https://sharethemeal.org/en/>

**Feedie** is an app created by The Lunchbox Fund that transforms your passion for sharing food photos into actually sharing food for children who need it most.

[**https://play.google.com/store/apps/details?id=org.thelunchboxfund.feedie&hl=en**](https://play.google.com/store/apps/details?id=org.thelunchboxfund.feedie&hl=en)

FYP2 Food Donor

ORIGINALITY REPORT

16% 2% 0% 16%

SIMILARITY INDEX INTERNET SOURCES PUBLICATIONS STUDENT PAPERS

|  |  |
| --- | --- |
| PRIMARY SOURCES |  |
| Submitted to Higher Education Commission 1  Pakistan  Student Paper | 12% |
| Submitted to Universiti Teknologi Petronas  2  Student Paper | 2% |
| Submitted to University of Newcastle upon  3 Tyne  Student Paper | 1% |
| Submitted to Informatics Education Limited 4  Student Paper | <1% |
| Submitted to Nottingham Trent University  5  Student Paper | <1% |
| firebase.google.com 6  Internet Source | <1% |
| Submitted to University of Bradford  7  Student Paper | <1% |
| Submitted to Polytechnic of Zagreb  8  Student Paper | <1% |

Submitted to University of Northampton

9 Student Paper <1%

Exclude quotes On Exclude matches Off

Exclude bibliography On