

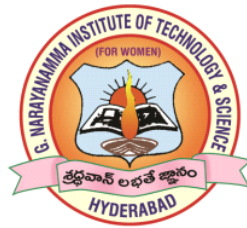
A Major Project Report on

SMART MANAGEMENT OF FOOD STORAGE AND WASTE REDUCTION

Submitted in partial fulfilment of the requirement for the award of the degree of
Bachelor of Technology in Information Technology

By

K CHAMUNDESHWARI



**G. Narayanamma Institute of Technology and Science
(For Women)**

Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad

Accredited by NBA, Shaikpet, Hyderabad – 500104, TS.

June - 2021

BATCH NO: 17A10

**SMART MANAGEMENT OF FOOD STORAGE AND
WASTE REDUCTION**

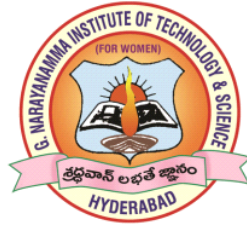
2021

SMART MANAGEMENT OF FOOD STORAGE AND WASTE REDUCTION

Submitted in partial fulfilment of the requirement for the award of the degree of
Bachelor of Technology in Information Technology

By

Name of the candidate	Registration Number
K CHAMUNDESHWARI	17251A1210



**G. Narayanamma Institute of Technology and Science
(For Women)**

Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad

Accredited by NBA, Shaikpet, Hyderabad – 500104, TS.

June - 2021

CERTIFICATE

This is to certify that the project report entitled “**SMART MANAGEMENT OF FOOD STORAGE AND WASTE REDUCTION**” is bonafide work done by

K CHAMUNDESHWARI 17251A1210

under the guidance of **U. Jyothi, Asst. Professor of IT**, during March 2021 to June 2021 in partial fulfilment for the award of degree in B. Tech in Information Technology, from G. Narayanamma Institute of Technology and Science.

Internal Guide

(U. Jyothi)

(Asst. Professor of IT)

Head of Department

(Dr. I. Ravi Prakash Reddy)

8-1-297/2/ I, Shaikpet, Hyderabad - 500 104, Telangana

☎ : 040-2356 5648 / 2356 5649 E-mail : principal@gnits.ac.in Website : www.gnits.ac.in

ACKNOWLEDGEMENT

We would like to express our sincere thanks to **Dr K. Ramesh Reddy, Principal**, G. Narayanamma Institute of Technology and Science for providing the working facilities in the college.

Our sincere thanks and gratitude to **Dr I. Ravi Prakash Reddy, Professor and HOD, Dept. of IT**, GNITS for all the timely support and valuable suggestions during the period of our Major project.

We are extremely thankful and indebted to our internal guide, **U. Jyothi Asst.Professor of IT**, GNITS for her constant guidance, continuous advice, encouragement and moral support throughout the Major project.

Finally, we would also like to thank all the faculty and staff of IT Department who have helped us directly or indirectly, parents and friends for their cooperation in completing the major project work.

ABSTRACT

Now a day's people are celebrating many events gloriously such as marriages, birthday parties, new organization launching parties, etc. There are a lot of food wastages at these kinds of parties and also at restaurants. So instead of trashing the food, it can be donated to needy people. Also, many people shift to their new homes due to reasons such as job change or purchase new houses and trash their furniture many times. So, this unwanted furniture can be donated to people in need similar to clothes and blood.

Many people in our society don't have proper clothes, food, furniture to wear, and many hospitals require the blood of certain groups. To achieve all the above requirements different kinds of interfaces were implemented like Apps, websites, etc. which are different for individual activities.

With the development of the proposed application, we can create an interface between donors, social service organizations, and volunteers. In our application donors can donate any item such as food items, clothes, furniture items and donate blood also. Donors can post their donate requests to different charities who so ever registered in this application. So that the donor posts, requests are visible only to the social service organizations and they can accept and take the items donated. Here donors need not physically visit the social service organization, therefore the donors can donate the food, furniture, clothes, blood from their places. Once the donors donate the needed items only the social service organization will get the sms notification of the available items and can accept/ reject the donation.

Here, the social service organizations can be charities or hospitals. When there is donation related to blood/plasma, the hospitals will get notified otherwise the charities will get notified. In this way, considering the current pandemic situations, many people will get benefitted. Later with the help of volunteers, they can pick the items from the donating person and distribute them to the needy people.

Table of Contents

S.No.	Name		Page No.
1.	Introduction		1-3
	1.1	General/Domain Description	1
	1.2	Objective & Scope of the project	1-2
	1.3	Project Definition	2
	1.4	Organization of Project Report	3
2.	Literature Survey		4-6
	2.1	Existing System	4-5
	2.2	Drawbacks in Existing System	5
	2.3	Motivation for Proposed System	5
	2.4	Proposed System	5-6
3.	Requirement Specification		7-14
	3.1	Overall description of the project (Use case Diagrams)	7-8
	3.2	Functional and Non- Functional requirements	9-10
	3.3	Design Specification(Class Diagrams/DFD/ E-R Diagrams)	10-13
	3.4	Software and Hardware Requirements	14
4.	Implementation		15-19
	4.1	Methodology (Flowcharts)	15
	4.2	System Architecture	16
	4.3	Modules Description	17
	4.4	Sequence Diagrams with timelines	18-19
5.	Testing		20-26
	5.1	Introduction and types of testing used	20-22
	5.2	Test Cases	23-26
6.	Conclusion and Future Scope		27
	6.1	Conclusion	27
	6.2	Future Scope	27
APPENDIX:			28-45

I.	Screenshots representing the flow of project	28-37
II.	Code	38-44
III.	References	45

List of Figures and Tables

Figure No.	Name	Page No.
3.1.1	Use Case Diagram (Donor)	8
3.1.2	Use Case Diagram (Charities)	8
3.3.1	Class Diagram	11
3.3.2	Data Flow Diagram	12
3.3.3	ER - Diagram	13
4.1	Flowchart	15
4.2	System Architecture	16
4.4.1	Sequence Diagram (Donor)	18
4.4.2	Sequence Diagram (Charities)	19
Table-1: Test cases		23-26

1. INTRODUCTION

1.1 General Description

It is a statistical fact that one-third of all food is wasted from farms, restaurants, grocery stores, weddings, parties etc. Yet there has been no efficient way for groups that have excess food to donate to the millions of hungry. On the other hand we have many Charity organizations that collect various products and provide food and other basic amenities for the poor. But they find it difficult to find donors. So we want to build an app that acts as a bridge between the donors and the charity organizations. We would like to make this app as a platform not just to donate food but also other products like clothes, blood, plasma, furniture items etc. In simple terms our app serves the Motto “Serving hands are better than praying lips”.

1.2 Objective & Scope of the project

Objectives

- 1) A donor can donate various commodities to the charities:
 - A user of this app should initially recognize himself as a donor or charity organization.
 - Once registered as a donor whenever there is anything he wants to donate he can post it by mentioning the item, quantity.
 - Here in this application we have a domain for hospital as well like if the donor wants to donate blood and plasma then they can select the hospital domain and they can donate the specific requirements to the hospital.
 - All the charities and hospitals will receive a notification that a particular donor is willing to donate a particular item.
 - The charitable organizations in the vicinity can accept the offering.
 - Once accepted the donor will get notified about the charity organization and hospitals that accepted the requests.
 - The charity organization that accepted the request gets to know the address and contact details of the donor and can approach him with the help of Google map which will be integrated with the app.
- 2) Charity organizations can find volunteers:

- When charity organizations require volunteers they can post all the details like what age group of candidates they are looking for, their expected roles etc.
- Interested individuals can approach the charity organizations for further details.

Scope of the project

The entire world celebrates the World Food Day, and India is leading the charge with most cities seeing people of all age groups donating food from their houses, restaurants, hotels and corporate. What could be better than people coming together uniting for a cause in such times? More than anything else people are realizing the importance of the fundamental right to food and the joy of sharing it with people who don't have enough. Overwhelmed with the support and love people have expressed for sharing food, we have developed the application named "Pack it and pass" and this would help in diminishing India's biggest social problem of hunger. Using this app, people can donate their leftover food to the receivers requested, thereby making food donation easier and more regular instead of a one day activity with the help of volunteers.

1.3 Project Definition

This project is developed for food redistribution which is an enormously successful social innovation that tackles food waste and food poverty. The admin collects the food from donors through their nearby agent then provides it to nearest orphanages and poor people. The donors can even donate clothes, furniture, blood and plasma.

1.4 Organization of the Project

The project Smart Management of Food Storage and Waste Reduction is structured in chapters, as follows:

1. Chapter 1 gives an introduction of what the proposed project is and the technologies included in that.
2. Chapter 2 explains about the drawbacks of the existing systems and how the proposed system works.

3. Chapter 3 deals with the requirements for the project.
4. Chapter 4 shows the workflow of project and description of modules.
5. Chapter 5 contains test plans and possible test cases of the project.
6. Chapter 6 includes result analysis, conclusion and future scope of the project.

2. LITERATURE SURVEY

- **Reduction of Food Wastage through Android Application – Make You Smile - By Mafishan Ali, Sana Sheikh, Yumna Sohail**
 - In this paper, application was to make reduction in food wastage as much as we can, and to feed those people who do not have enough food for one time to feed themselves.
 - Developing an android application, which aims to establish a link between restaurants and the NGOs or organizations who works for these cause/needy households to enable the access for food donation.
 - Application consists of three modules:
 - ✓ Admin Module
 - ✓ User Module
 - ✓ Rider/Driver Module
- **D' WORLD: Blood Donation App Using Android Dr. A. Meiappan¹, K. Logavignesh², R. Prasanna³, T.Sakthivel⁴**
 - This project is an online blood bank system that is achieved on an android platform. The android application will help users to view and access information such as the nearby hospitals and the blood banks and the availability of blood banks.
 - The App will contain names, address, contact and blood type of blood donors available nearby from the persons in need of blood.
 - This application uses GPS to track the donors and nearby blood banks.
 - The Donor is sorted out with following constraints such as Blood Group, Rh factor, and Last date of Donation and at last based on the Location.

2.1 Existing System

- Many people in our society do not have proper clothes, food, furniture and many hospitals are in the need of blood of certain groups.
- To achieve all the above requirements different kinds of interfaces were implemented like Apps, websites, etc. which are different for individual activities.

- In existing system if anyone have extra food because of any function or in their home it will be become waste because instantly there is no way to share with anyone if they are having lots of food.
- Even if they want to give that extra food, clothes and furniture to any orphanage or poor people they don't have time or don't have an idea about that.

2.2 Drawbacks in Existing System

- Food is lost or wasted due to an unstable market, Overbuying, poor planning causes a lot of food wastage.
- Some of the websites may be fraudulent or unauthentic.
- You don't know who shows up at your door.
- When food ends up in landfills and decomposes, it creates methane, a harmful greenhouse gas that accelerates climate change.

2.3 Motivation for Proposed System

This application deals with the idea of developing an interface between donors, social service organizations to help needy people with different types of donations.

2.4 Proposed System

- In this application donors can donate any item such as food items, clothes, furniture items and donate blood and plasma as well.
- Donors can
 - Select which type of donation they want to make.
 - Post their donate requests to different charities and hospitals.
 - Once the donors donate the needed items, then the social service organization will get the sms notification of the available items and can accept/ reject the donation.
 - In the similar fashion if the donor wants to donate the modules included in the hospital domain such as blood and plasma he can donate it further and the hospitals will get notified of the available blood or plasma and then the hospitals can accept or reject the donation.
 - Here donors need not physically visit the social service organization.

- Later with the help of volunteers, charities can pick up the items from the donating person.

3. REQUIREMENT SPECIFICATION

3.1 Overall Description of the Project (Use Case Diagrams)

Home Page of this application consists of two fragments, donor and charities. Here the donors and charities have a separate registration and login forms.

If the user is donor, then the donor can create an account using register option and then by entering details like full name, user name, password, mobile number, address. After that they can login into the app using their username & password. They can donate funds to charity by using donate option and posting a request. Status of the request is shown in status option. A sms notification is sent to the registered charities whenever donor posts a request. If the request is related to blood/plasma, then the registered hospitals will get notified and if it is related to food, furniture or clothes then charities will get notified. They can also delete their request and can change their password.

At charities side, there are two users namely charity and hospitals. The user can create an account by entering details like selecting the type of organization (charity or hospital), name of the organization, user name, password, mobile number and address. After that they can log on to the app using their username & password. They can see the requests posted by donors using the donor's request option based on type of the organization. They can see their accepted requests and can also change their password. After accepting requests, respected donor will be notified through sms.

Use case diagram:

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved. Use cases are used during the analysis phase of a project to identify system functionality. They separate the system into actors and use cases.

Use case (donor):

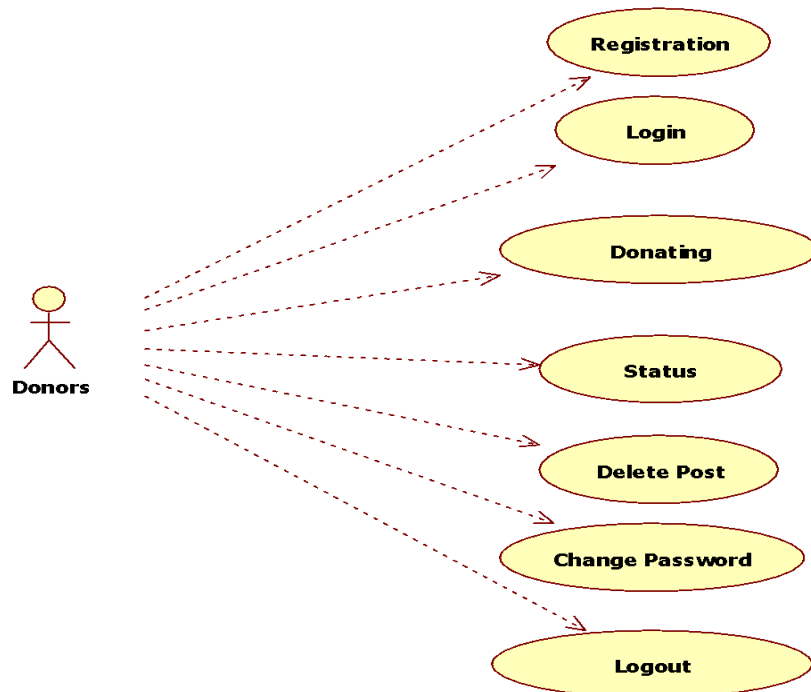


Fig 3.1.1 Use Case Diagram (Donor)

Use case (charities):

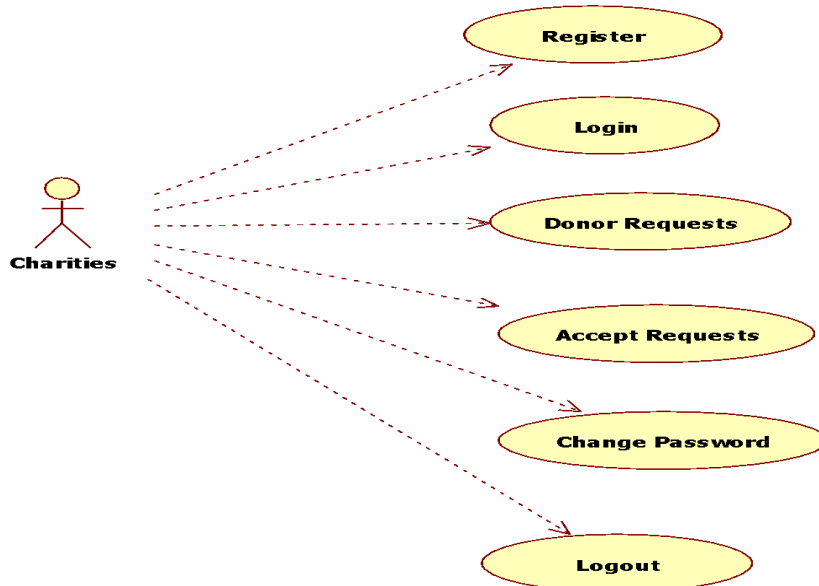


Fig 3.1.2 Use Case Diagram (Charities)

3.2 Functional & Non-functional Requirements Specification

Functional Requirements

A functional requirement defines a function of a system or its component. There are 2 main modules in this system.

- Donor Module
- Charities Module

Donor Module

- Donors Registration

Donors must be able to create an account by providing required information.

- Donors Login

Donors must be able to login into the app by giving user name & password.

- Donating

The registered donors must be able to post requests.

- Change Password

Donor must be able to change their password.

Charities Module

- Charities Registration

New Charities must be able to create an account by providing all the required information.

- Charities Login

Existing charities must be able to login into the app by giving the user name & password.

- Donor Request

Charities must be able to see donor requests.

- Accept Request

Charities must be able to accept requests.

Non- Functional Requirements

- **Usability:**

Usability requirements are documented expectations and specifications designed to ensure that a product, service, process or environment is easy to use. Requirements can be provided in a broad variety of formats by business units, customers and subject matter experts.

- **Serviceability:**

Serviceability requirements are a set of conditions under which a foundation structure is considered to be useful. It is the second category of performance after the strength requirement. Serviceability requirements of foundations include settlement, heave, tilt, vibration, lateral movement, and durability.

- **Manageability:**

Manageability deals with system monitoring of the QoS requirements and the ability to change the system configuration to improve the QoS dynamically without changing the system. Your architecture must have the ability to monitor the system and allow for dynamic system configuration.

- **Security:**

A security requirement is a goal set out for an application at its inception. Every application fits a need or a requirement. For example, an application might need to allow customers to perform actions without calling customer service.

3.3 Design Specification

3.3.1 Class Diagram

In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

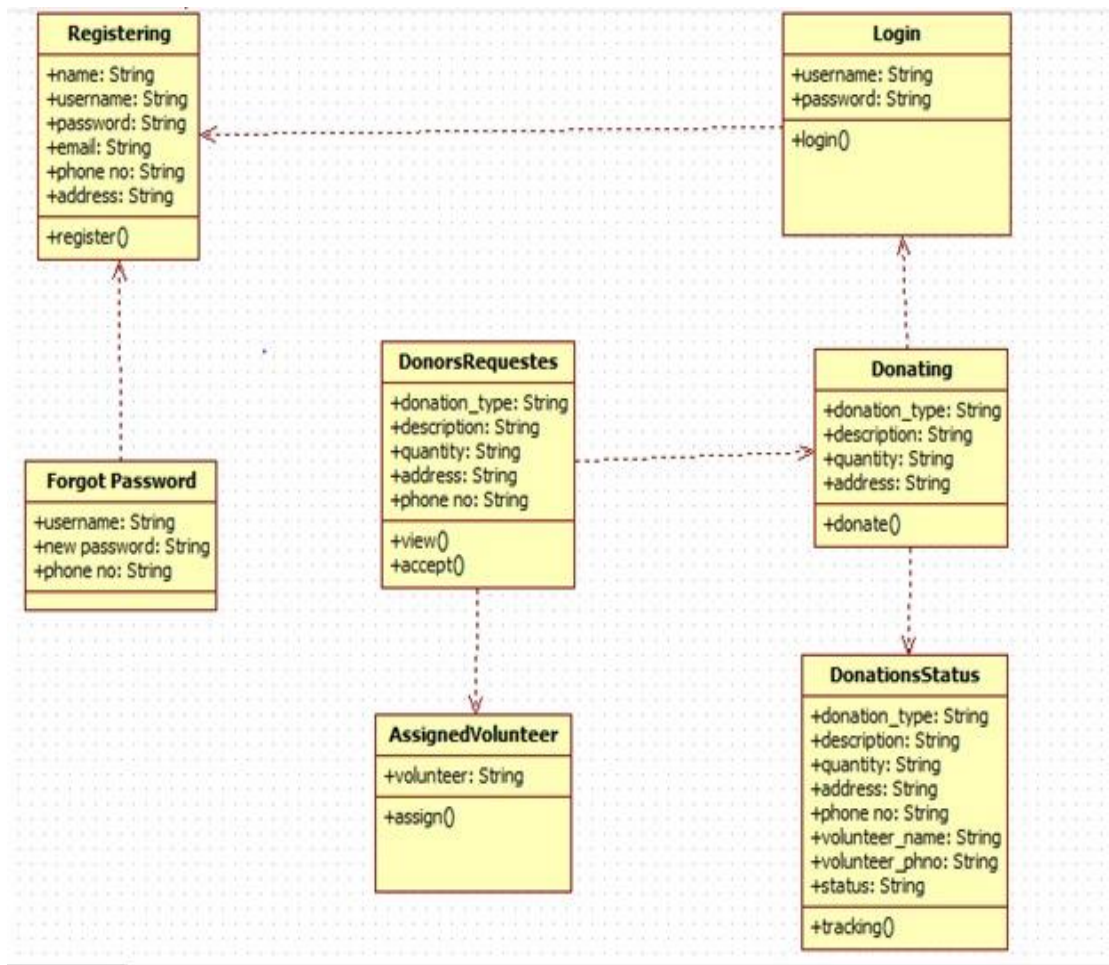


Fig 3.3.1 Class Diagram

3.3.2 Dataflow Diagram

A data-flow diagram (DFD) is a way of representing a flow of a data of a process or a system (usually an information system) The DFD also provides information about the outputs and inputs of each entity and the process itself. A data-flow diagram has no control flow; there are no decision rules and no loops.

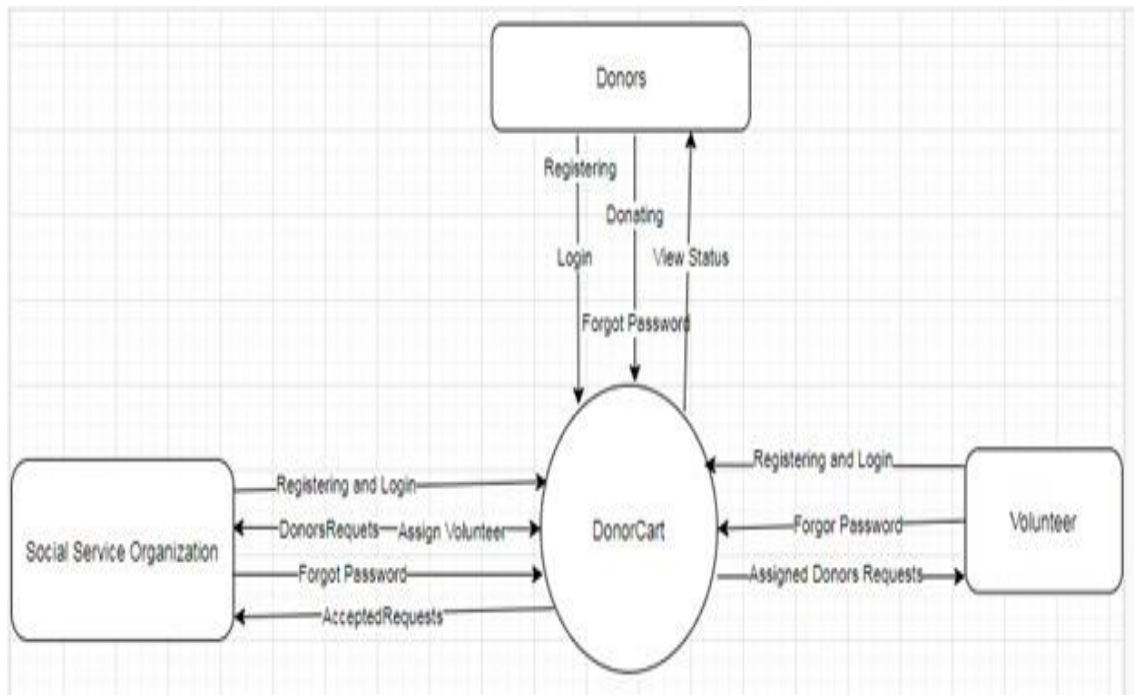


Fig 3.3.2 Data Flow Diagram

3.3.3 E-R Diagram

An entity-relationship diagram (ERD) is a data modelling technique that graphically illustrates an information system's entities and the relationships between those entities. An ERD is a conceptual and representational model of data used to represent the entity framework infrastructure.

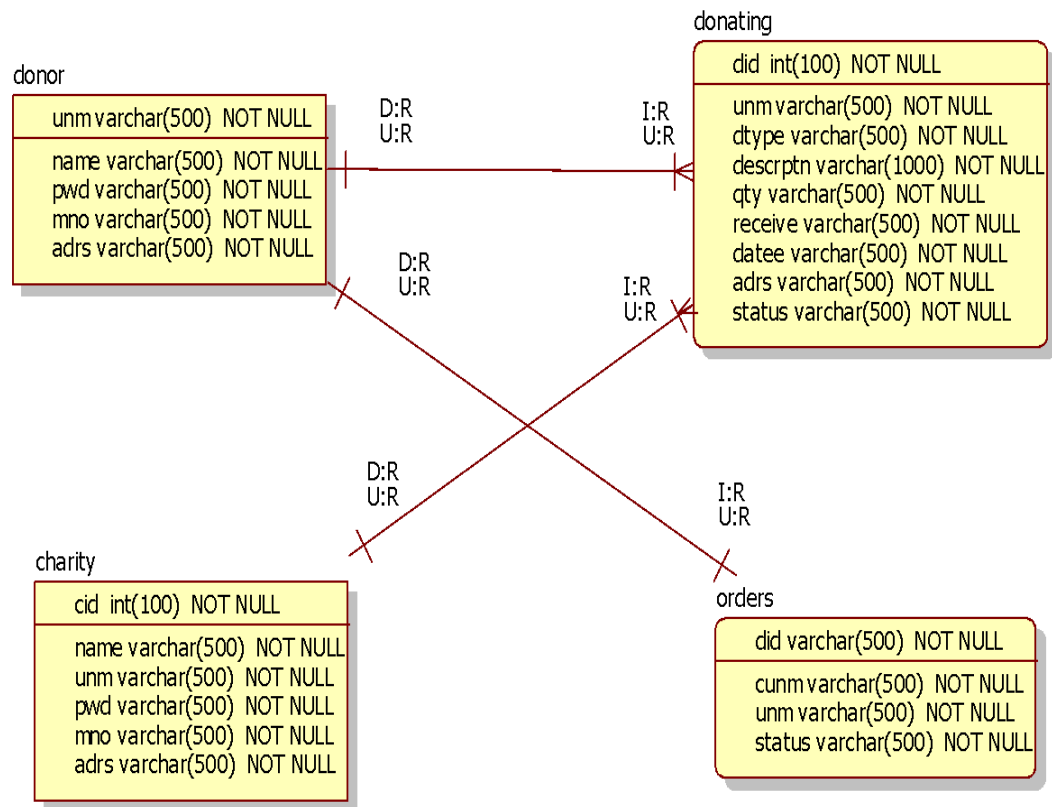


Fig 3.3.3 ER Diagram

3.4 Software and Hardware Requirements

Hardware requirements

The section of hardware configuration is an important task related to software development. Insufficient random-access memory may adversely affect the speed and efficiency of the entire system. The process should be powerful enough to handle the entire operation. The hard disk should have sufficient capacity to store the file and application.

- Processor : Intel core i5 8th gen
- Hard Disc Capacity : 1TB
- RAM Capacity : 8GB

Software requirements

A major element in building a system is the section of compatible software since the software in the market is experiencing geometric progression. Selected software should be acceptable by the firm and one user as well as it should be feasible for the system. This document gives a detailed description of the software requirement specification. The study of requirement specification is focused especially on the functioning of the system. It allows the developer or analyst to understand the system, function to be carried out, the performance level to be obtained and corresponding interfaces to be established.

- Platforms : Android 4.1.3
- Back-end : MySQL 5.5
- Front-end : Java 8
- Development Tool : Android Studio

4. IMPLEMENTATION

4.1 Methodology (with Flow Chart)

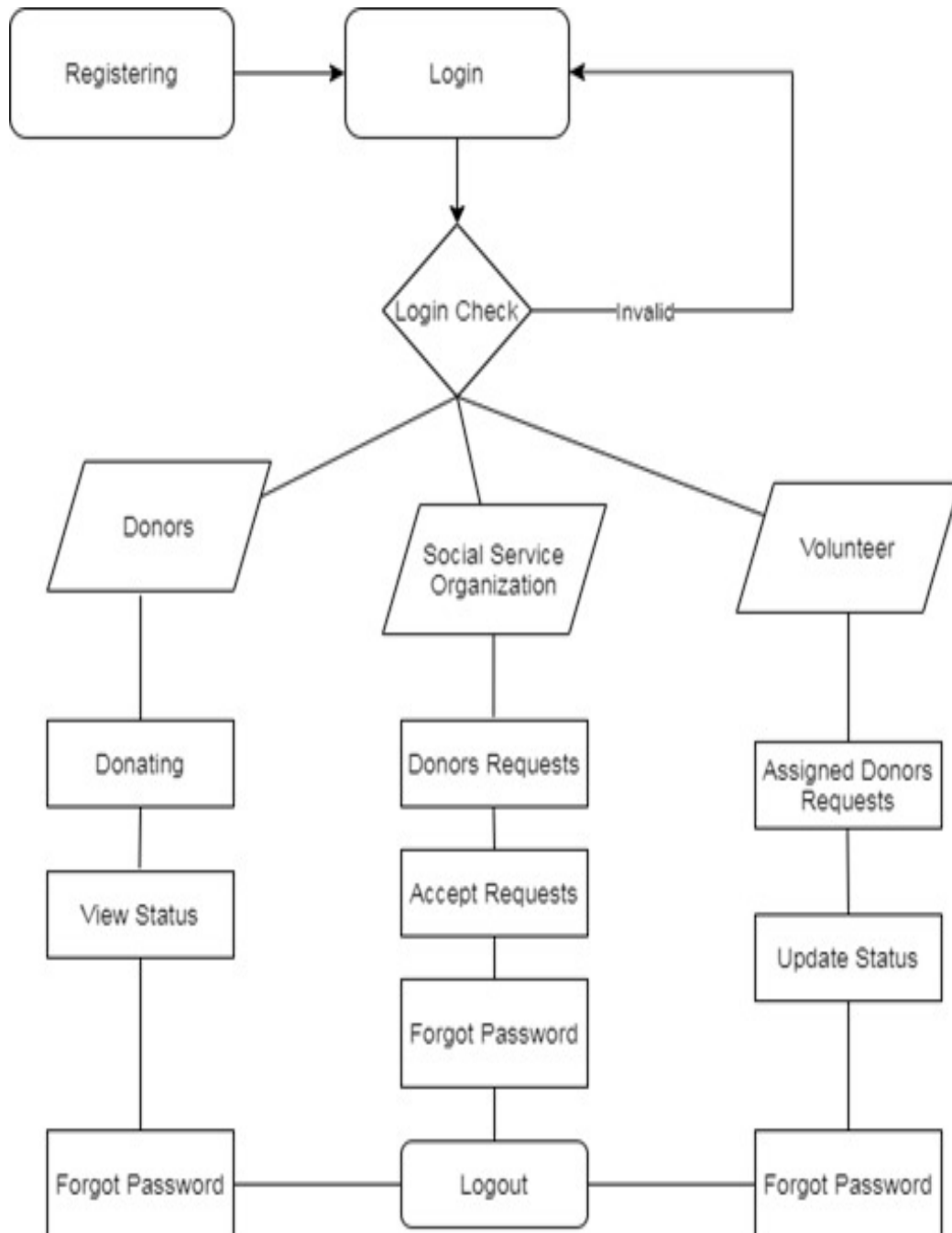


Fig 4.1 Flow Chart

4.2 System Architecture

A system architecture or systems architecture is the conceptual model that defines the structure, behaviour, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviour of the system.

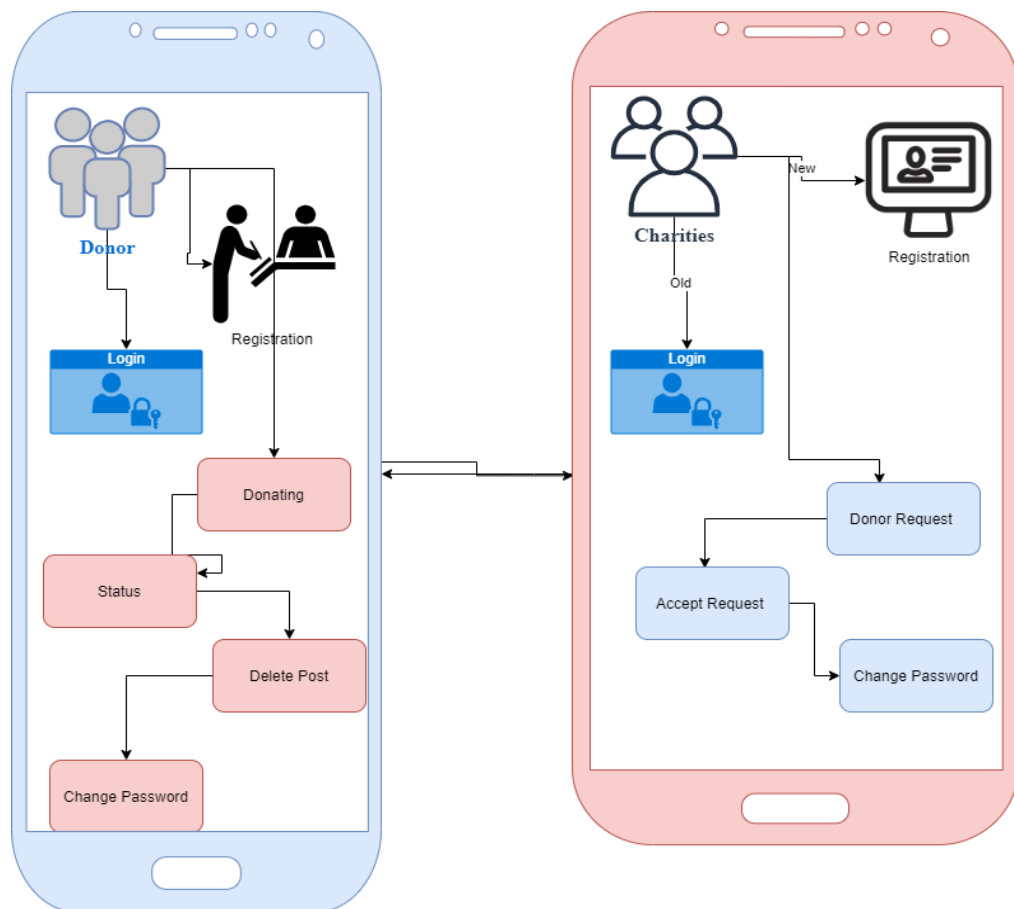


Fig 4.2 System Architecture

4.3 Modules Description

This application has five modules namely login & registration, donor, notification, charities and volunteer module.

Login &Registration:

- This involves login & registration for both the organization and users.
- The user's details are maintained confidential by maintaining separate account for each user.

Donor:

- Donor gives the desired items like food, clothes, furniture and also blood.
- The donor can view the charity details and receiver details.

Notification:

- This involves the notification to the organizations by the users.
- Also, the user will send the notification about the request via sms.

Charities:

- The charities can accept or reject the request made by the donor.
- Can collect the items by one of the volunteers.

Volunteer:

- Volunteer will have the organization details and donor details.
- After collecting the food the volunteer gives the alert message to the donor.

4.4 Sequence Diagram with timelines

Sequence is what happens next. Sequence diagram are closely related to collaboration diagram. The main difference between sequence and collaboration diagram is that sequence diagram shows time-based interaction while collaboration diagram shows objects associated with each other.

A sequence diagram shows, as parallel vertical lines (lifelines), different processes or objects that live simultaneously and as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner. The sequence diagram for the system is as follows: it consists of different objects namely user, system and database.

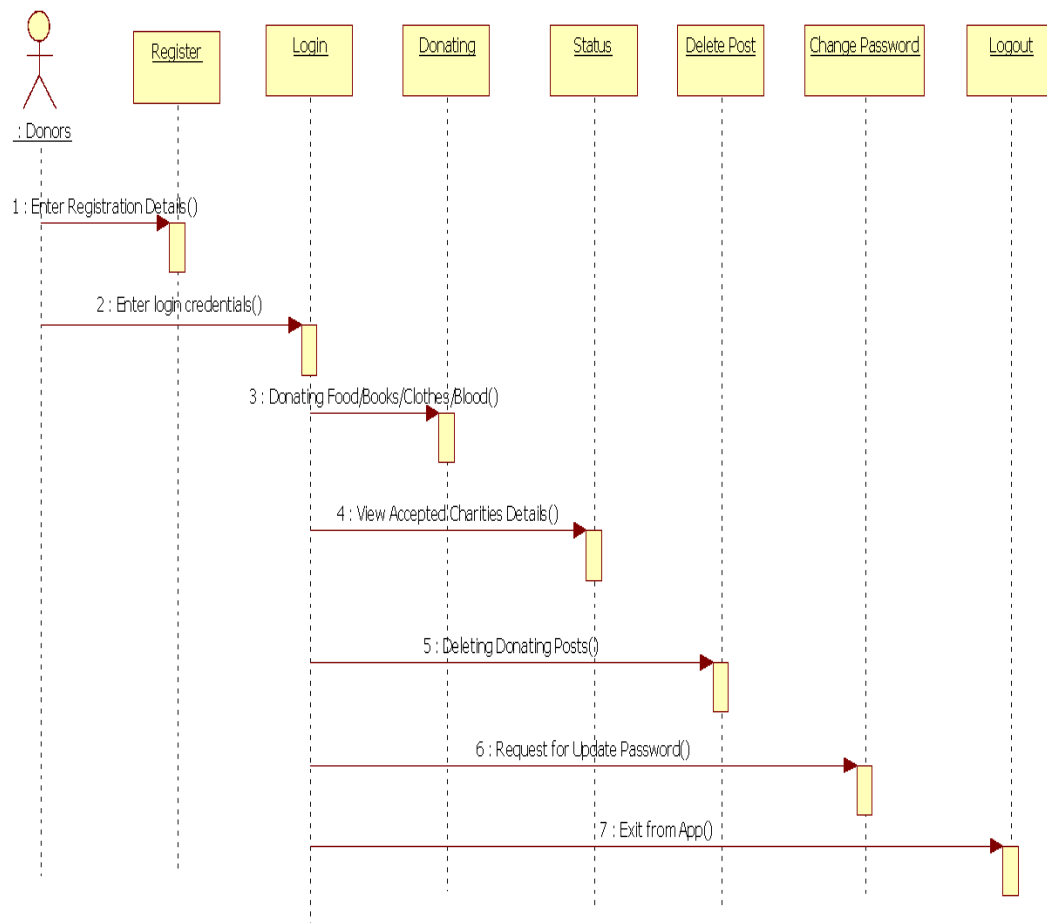


Fig 4.4.1 Sequence Diagram of Donor

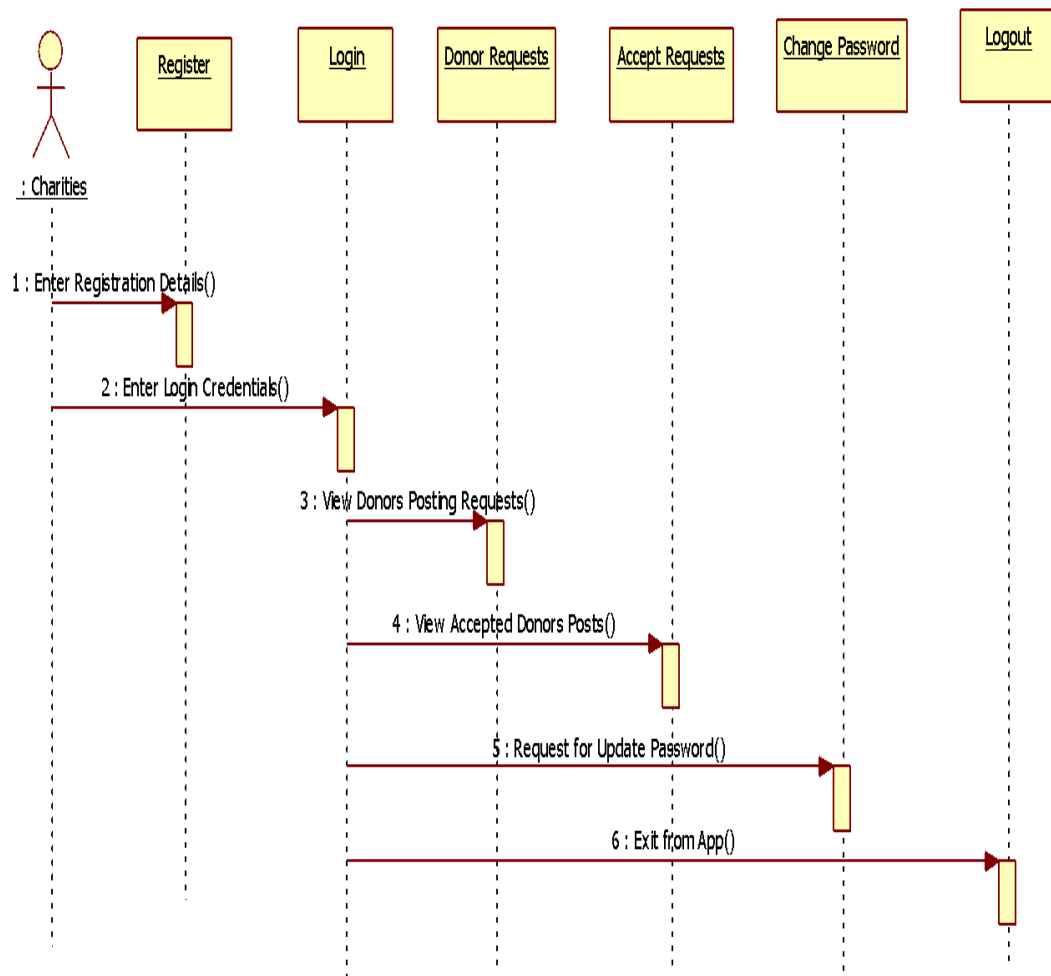


Fig 4.4.2 Sequence Diagram of Charities

5. TESTING

5.1 Introduction and Types of Testing used

Software Testing is the process used to help identify the correctness, completeness, security, and quality of developed computer software. Testing is a process of technical investigation, performed on behalf of stakeholders, that is intended to reveal quality-related information about the product with respect to the context in which it is intended to operate. This includes, but is not limited to, the process of executing a program or application with the intent of finding errors. Quality is not an absolute; it is value to some person. With that in mind, testing can never completely establish the correctness of arbitrary computer software; testing furnishes a criticism or comparison that compares the state and behaviour of the product against a specification. An important point is that software testing should be distinguished from the separate discipline of Software Quality Assurance (SQA), which encompasses all business process areas, not just testing.

There are many approaches to software testing, but effective testing of complex products is essentially a process of investigation, not merely a matter of creating and following routine procedure. One definition of testing is "the process of questioning a product in order to evaluate it", where the "questions" are operations, the tester attempts to execute with the product, and the product answers with its behaviour in reaction to the probing of the tester. Some of the common quality attributes include capability, reliability, efficiency, portability, maintainability, compatibility and usability. A good test is sometimes described as one which reveals an error; however, more recent thinking suggests that a good test is one which reveals information of interest to someone who matters within the project community.

Levels of testing:

Unit Testing

In computer programming, unit testing is a method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage Procedures, and operating procedures are tested to determine if they are fit for use. Intuitively, one can view a unit as the smallest testable part of an application.

Integration Testing

Integration testing is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing. Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing.

System Testing

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic.

Acceptance Testing

Acceptance testing is a test conducted to determine if the requirements of a specification or contract are met. Acceptance Tests are usually created by business customers and expressed in a business domain language.

Black Box Testing

Black-box testing is a method of software testing that examines the functionality of an application without peering into its internal structures or workings. This method of test can be applied to virtually every level of software testing: unit, integration, system and acceptance. It typically comprises most if not all higher- level testing, but can also dominate unit testing as well.

Typical black-box test design techniques include:

- Decision table testing
- All-pairs testing
- State transition tables
- Equivalence partitioning
- Boundary value analysis

White Box Testing

White-box testing is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality. In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs.

Testing Objectives

- To ensure that during operation the system will perform as per specification.
- To make sure that system meets the user requirements during operation.
- To make sure that during the operation, incorrect input, processing and output will be detected.
- To see that when correct inputs are fed to the system the outputs are correct.
- To verify that the controls incorporated in the same system as intended.
- Testing is a process of executing a program with the intent of finding an error.
- A good test case is one that has a high probability of finding an as yet undiscovered error.
- The software developed has been tested successfully using the following testing strategies and any errors that are encountered are corrected and again the part of the program or the procedure or function is put to testing until all the errors are removed. A successful test is one that uncovers an as yet undiscovered error.

5.2 Test Cases

Test Case1: Registration

Test Case ID	Test Case Name/Objective	Prerequisites // pre-condition	Test Description	Expected Result	Actual Result	Pass/Fail	Action/Notes
1	Users give their first and last name.	User must install the app	The user has to provide his/her details.	Pop showing email verification id.	Enter valid email/password	Fail	There should be a permanent backup of data
2	Submitting the form without entering any details	User must install the app	The app shows user details as invalid and does not accept the request.	Pop showing email verification message	Enter Email/ password.	Fail	
3	User enters invalid format of email id	User must install the app	The app shows an error.	Pop showing email verification message	Enter valid email id	Fail	
4	User enters a phone number with < 10 digits	User must install the app	The app shows an error.	Pop showing email verification message	Enter valid phone number	Fail	
5	Entering valid username and password	User must install the app	The user is registered successfully.	Pop showing email verification message	Pop showing email verification message	Pass	

Test case 2: Login

Test Case ID	Test Case Name/ Objective	Prerequisites // pre-condition	Test Description	Expected Result	Actual Result	Pass /Fail	Action/ Notes
1	User gives an email or password of <6 characters	User must be registered.	Shows an error message.	User logged in.	Enter valid email/ password.	Fail	
2	.Submitting the form without entering the details.	User must be registered.	Shows an error message.	User logged in.	Enter email/ password.	Fail	
3	User enters the wrong email or password.	User must be registered.	Shows an error message.	User logged in.	Enter correct email/password	Fail	

Test case 3: Search

Test Case	Test Case Name/	Prerequisites // pre-condition	Test Description	Expected Result	Actual	Pass/Fail	Action/Notes
-----------	-----------------	--------------------------------	------------------	-----------------	--------	-----------	--------------

e ID	Objective				Result		
1	User enters a value and clicks on search.	User must be registered.	Shows the value entered if available.	Results in the form of url	Nothing happens	Fail	

Test case 4: User Modules

Test Case ID	Test Case Name/ Objective	Prerequisites // pre-condition	Test Description	Expected Result	Actual Result	Pass /Fail	Action/ Notes
1	Click on the history tab.	User must be registered and logged in.	Shows the user's history.	Shows relevant history.	Get information.	pass	
2	Click on the activities tab.	User must be registered and logged in.	Shows the activities performed.	Get some tabs here.	Enter context users search keys.	pass	

Test case 5: Revisitation

Test Case ID	Test Case Name/Objective	Prerequisites // pre-condition	Test Description	Expected Result	Actual Result	Pass/Fail	Action/Notes
1	Revisitation.	User must be registered and logged in.	Shows all the features.	Enter context keywords.	calculate	Fail	

6. CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

The proposed application shall reduce food wastage and also fulfil other requirements like clothes, books, utensils, blood and plasma which have always been the common necessities for hospitals as well as of needy organizations. As mentioned above in the description there is a lot of food wastage that occurs daily at restaurants and cafes. Instead of throwing away the same as trash (which usually is the scenario), it can be used to feed the homeless. Also, since the pickup is arranged for by the enterprise, the restaurants/cafes need not worry about it. Benefits will be both the restaurants/cafés (reducing the carbon footprint and wastage), and the needy.

6.2 Future Scope

In future work, there was no standard food information system on food packages that gives the user the information of both the name of the food, as well as its expiry date. The viable improvement would be to get the food name from the product barcode and read the expiry date using OCR tools. However, the level of ease of using this option is only slightly greater than using the manual option of filling the food information. Some companies have started trials using QR code on their food packages to provide detailed information. Notwithstanding, there are still a lot of hurdles to pass for it to become a standard. But for the meantime, this application presents a viable and effective solution.

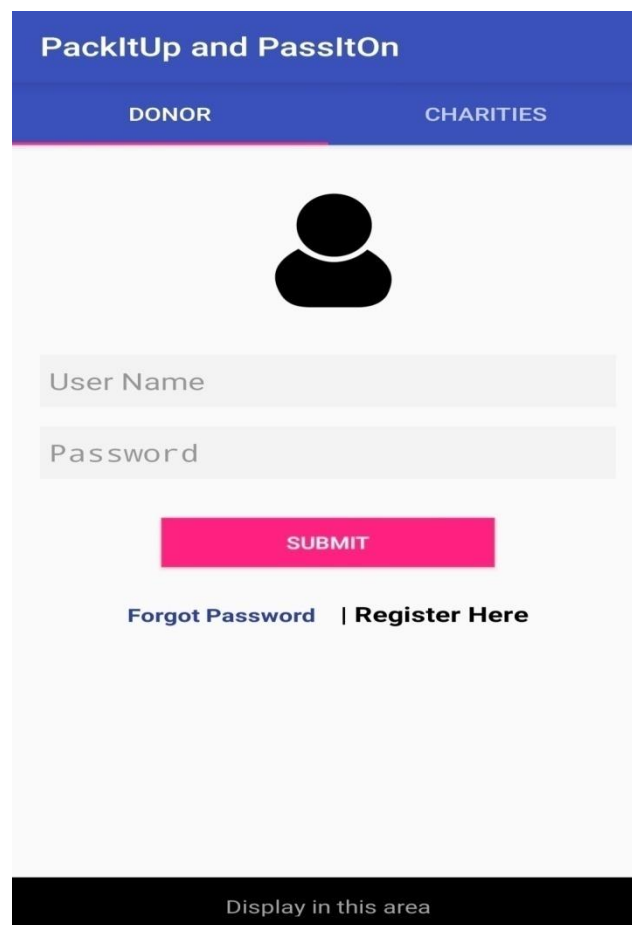
Even in the recent times we have not seen any such application which would act as an interface between the hospitals and the donors, Hence this application would become an ease for the hospitals as well as donors in fulfilling the requirements for both the ends. In this application the notification of the donor's request of donation would be sent directly to the hospitals and the hospitals could accept or reject the request as per their requirements. We have worked on the application authentication where if it seems to be insecure we have the facility of changing the password and resetting them in times.

APPENDIX

I. Screenshots representing the flow

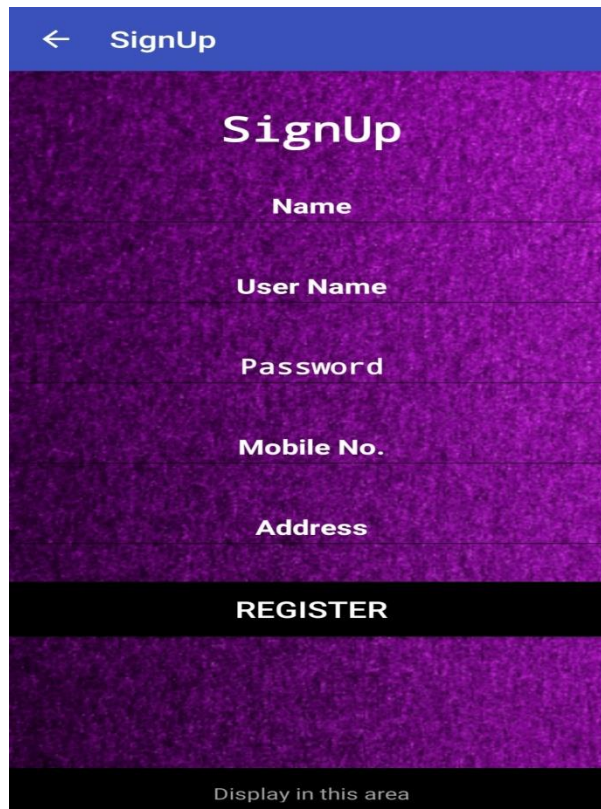
DONOR MODULE

Home page



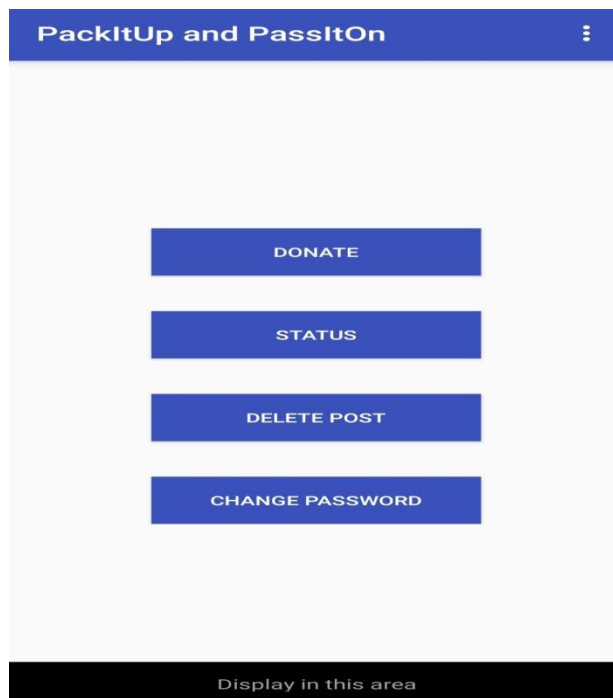
The screenshot displays the 'PackItUp and PassItOn' Donor Module home page. At the top, a blue header bar contains the app name and two tabs: 'DONOR' (active) and 'CHARITIES'. Below the header is a large black silhouette of a person. Underneath the silhouette are two input fields labeled 'User Name' and 'Password'. A prominent pink 'SUBMIT' button is positioned below the password field. At the bottom of the main content area, there are links for 'Forgot Password' and 'Register Here'. A black footer bar at the very bottom contains the text 'Display in this area'.

Sign up




A mobile app screen for signing up. The header is blue with a back arrow and the text "SignUp". The background is a purple textured pattern. The form contains five input fields with labels: "Name", "User Name", "Password", "Mobile No.", and "Address". Below the fields is a black button with the text "REGISTER". At the bottom, there is a black bar with the text "Display in this area".

Login Page



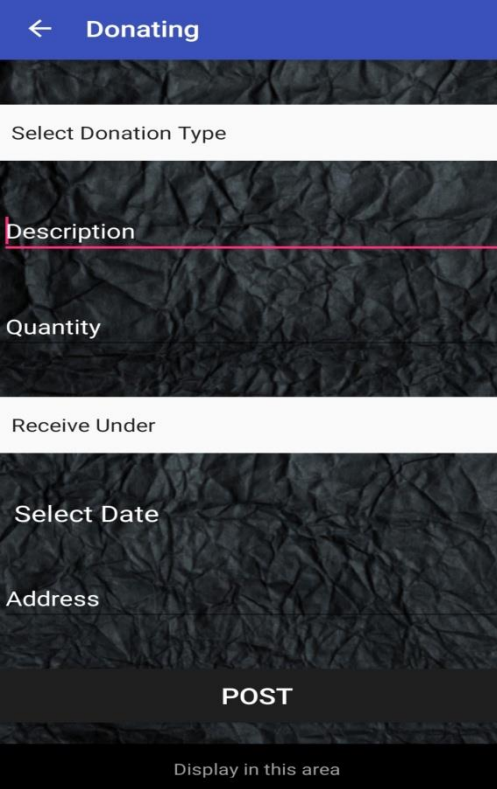
A mobile app screen for "PackItUp and PassItOn". The header is blue with the app name and a menu icon. The background is a light gray textured pattern. The screen displays four blue buttons with white text: "DONATE", "STATUS", "DELETE POST", and "CHANGE PASSWORD". At the bottom, there is a black bar with the text "Display in this area".

Forgot Password

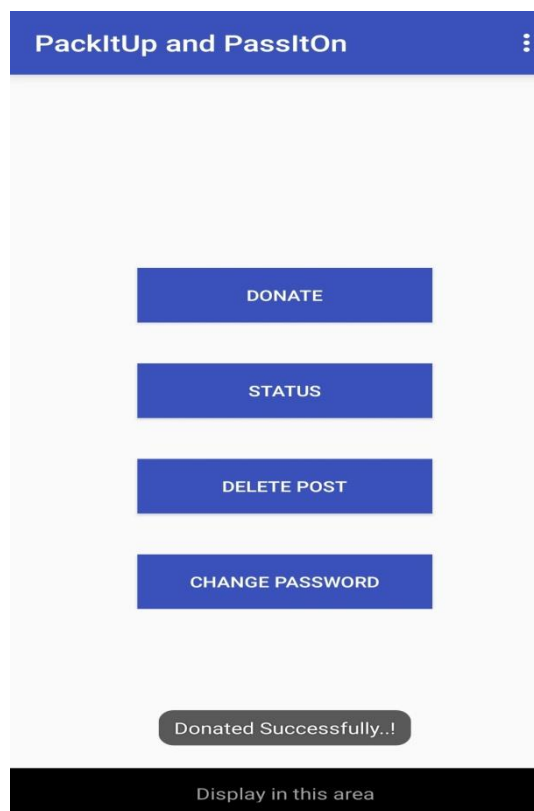
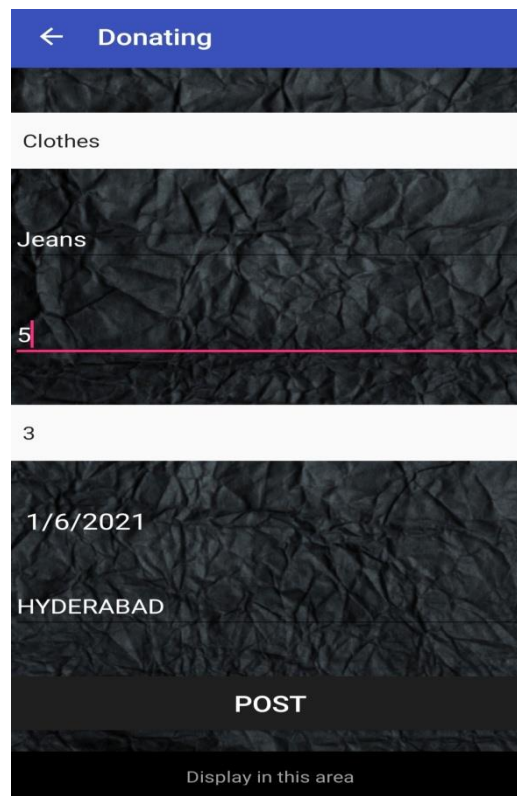


The screenshot shows a mobile application interface for the 'Forgot Password' screen. At the top, there is a blue header bar with a white back arrow and the text 'Forget Password'. Below the header, there is a logo featuring a blue question mark and the text 'I FORGET MY PASSWORD'. Underneath the logo, there are two input fields: 'User Name' and 'Mobile No.'. Below these fields is a blue button with the text 'GET PASSWORD'. At the bottom of the screen, there is a black bar with the text 'Display in this area'.

Donate Page



The screenshot shows a mobile application interface for the 'Donating' screen. At the top, there is a blue header bar with a white back arrow and the text 'Donating'. Below the header, there is a dark, textured background. The screen contains several input fields: 'Select Donation Type', 'Description', 'Quantity', 'Receive Under', 'Select Date', and 'Address'. At the bottom of the screen, there is a black bar with the text 'POST'. Below the 'POST' button, there is a black bar with the text 'Display in this area'.



Status Page

← Donating Status

chamu

Clothes

Desc:
Jeans

Qty: 5

Receive Time: 3 hr(s)

1/6/2021

HYDERABAD

80747

47841

Display in this area

← Charity Details

Charity Name

Owner Name

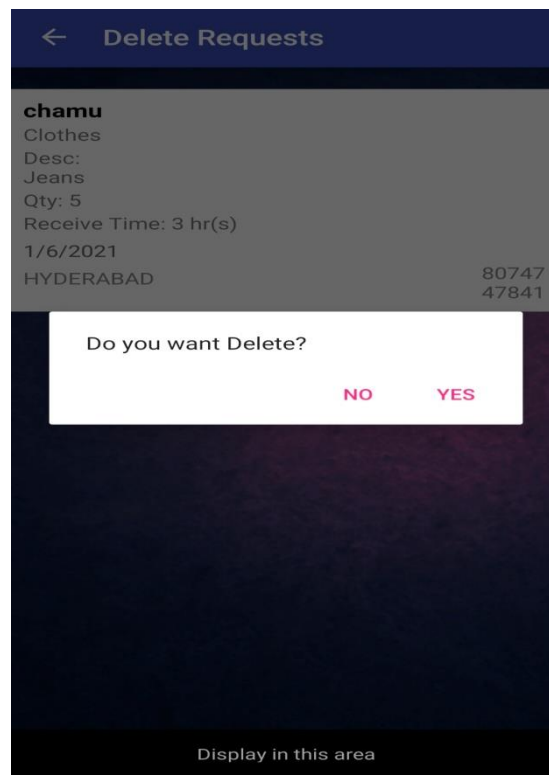
Mobile No.

Address

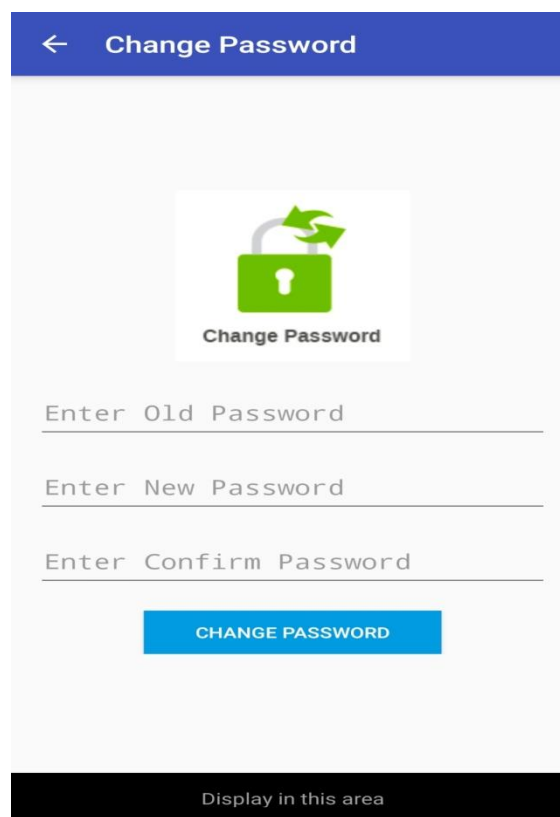
Status

Display in this area

Delete Request

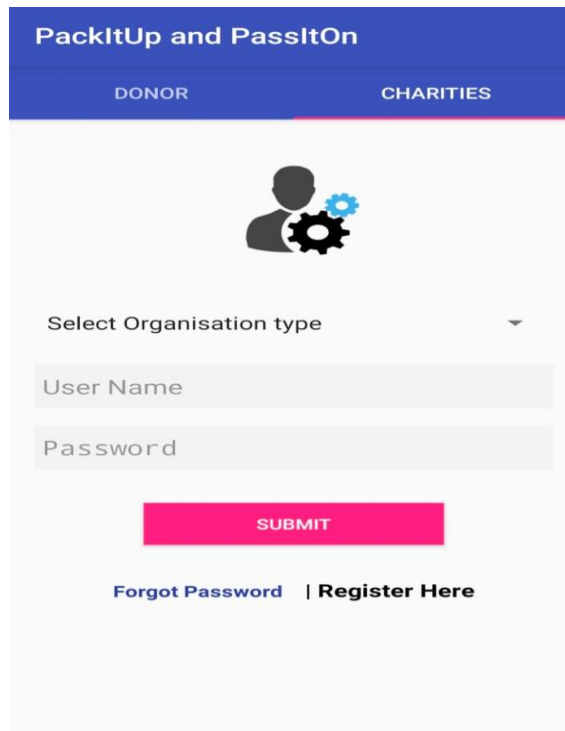


Change Password



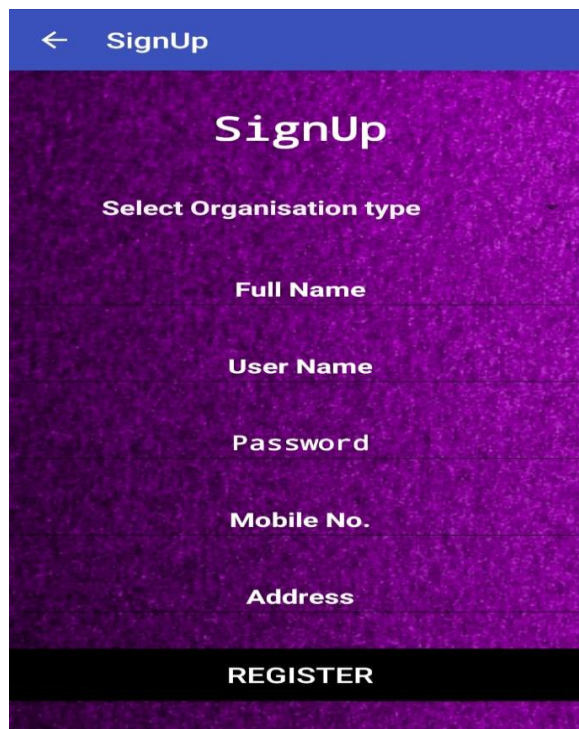
CHARITIES MODULE

Home Page



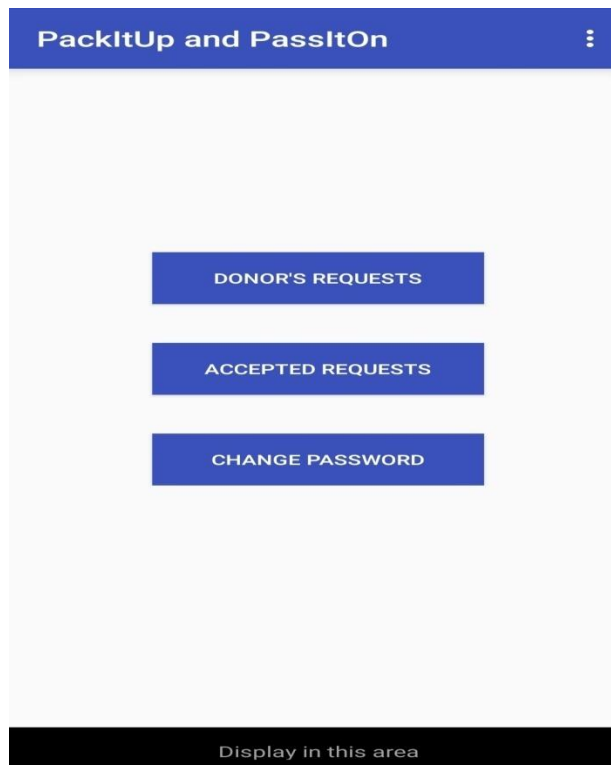
The screenshot shows the 'PackItUp and PassItOn' app interface. At the top, there is a blue header with the app name. Below the header, there are two tabs: 'DONOR' and 'CHARITIES'. The 'CHARITIES' tab is selected. In the center, there is a grey box containing a user icon with a gear, a dropdown menu labeled 'Select Organisation type', two input fields for 'User Name' and 'Password', a red 'SUBMIT' button, and two links: 'Forgot Password' and 'Register Here'.

Sign Up Page



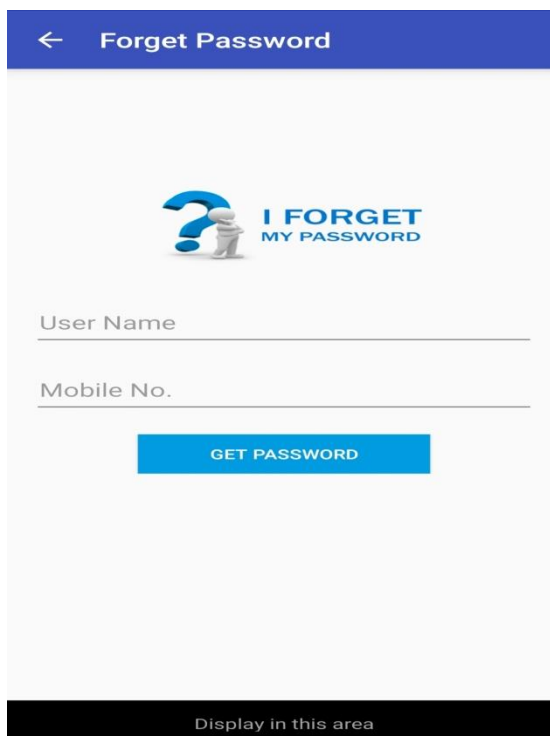
The screenshot shows the 'Sign Up' page of the app. At the top, there is a blue header with a back arrow and the text 'SignUp'. Below the header, there is a purple background with the text 'SignUp' in white. Underneath, there is a dropdown menu labeled 'Select Organisation type'. Below this, there are five input fields: 'Full Name', 'User Name', 'Password', 'Mobile No.', and 'Address'. At the bottom, there is a black button labeled 'REGISTER'.

Login Page



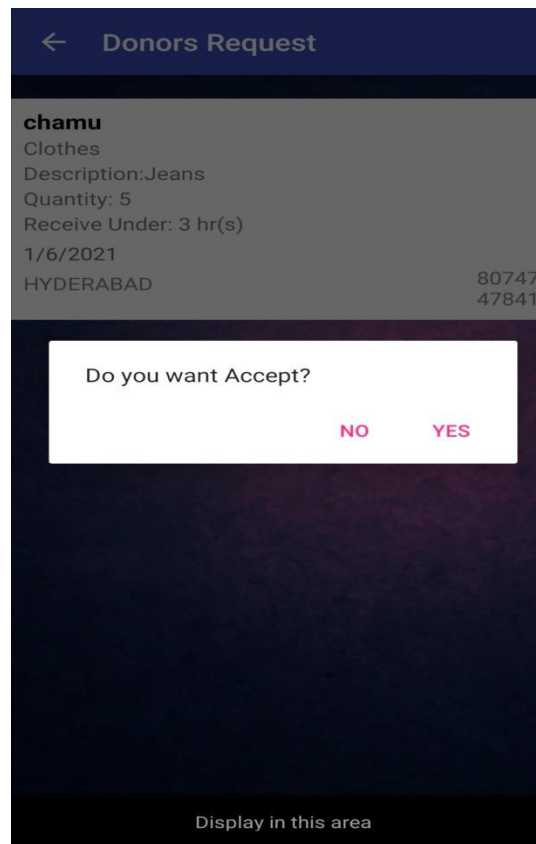
The image shows a mobile application interface for 'PackItUp and PassItOn'. At the top, there is a blue header bar with the text 'PackItUp and PassItOn' and a three-dot menu icon on the right. The main content area is light gray and contains three blue buttons stacked vertically: 'DONOR'S REQUESTS', 'ACCEPTED REQUESTS', and 'CHANGE PASSWORD'. At the bottom, there is a black footer bar with the text 'Display in this area'.

Forgot Password

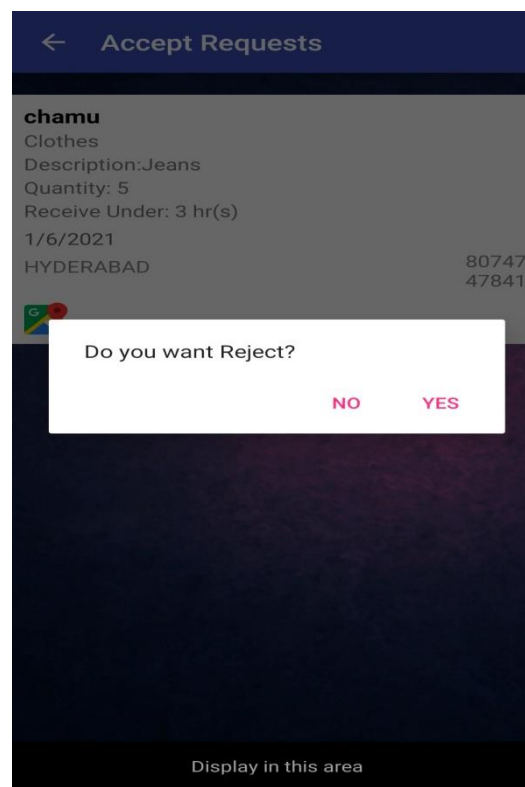
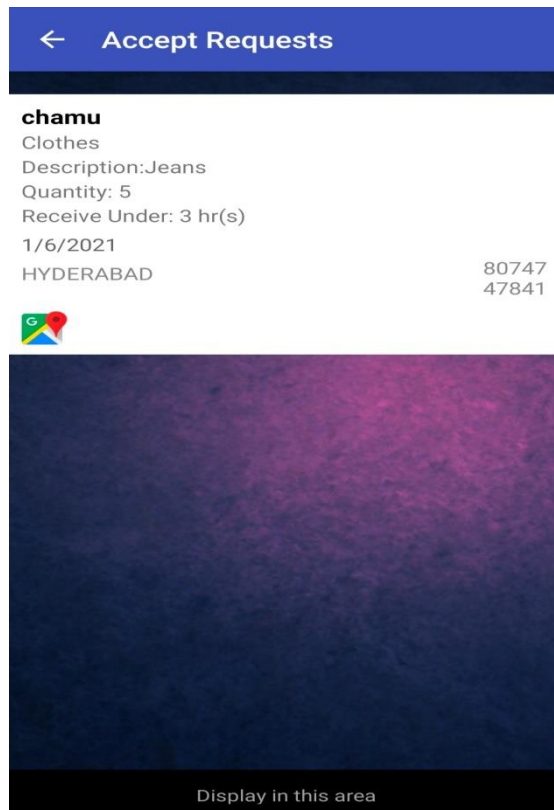


The image shows a mobile application interface for the 'Forgot Password' page. At the top, there is a blue header bar with a back arrow icon and the text 'Forgot Password'. The main content area is light gray and contains a logo with a blue question mark and the text 'I FORGET MY PASSWORD'. Below the logo, there are two input fields: 'User Name' and 'Mobile No.'. Below the input fields, there is a blue button labeled 'GET PASSWORD'. At the bottom, there is a black footer bar with the text 'Display in this area'.

Donor's Request Page



Accept Request Page



II. Code

Main:

```
package ct.donating;
import android.Manifest;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.os.Bundle;
import android.os.Handler;
import android.support.design.widget.TabLayout;
import android.support.v4.app.ActivityCompat;
import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentManager;
import android.support.v4.app.FragmentPagerAdapter;
import android.support.v4.content.ContextCompat;
import android.support.v4.view.ViewPager;
import android.support.v7.app.AppCompatActivity;
import android.support.v7.widget.Toolbar;
import android.widget.Toast;
import java.util.ArrayList;
import java.util.List;
public class MainActivity extends AppCompatActivity {
    private Toolbar toolbar;
    private TabLayout tabLayout;
    private ViewPager viewPager;
    Bundle extras=null; boolean doubleBackToExitPressedOnce = false;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        isSmsPermissionGranted();
        requestReadAndSendSmsPermission();
        viewPager = (ViewPager) findViewById(R.id.viewpager);
        setupViewPager(viewPager);
        tabLayout = (TabLayout) findViewById(R.id.tabs);
        tabLayout.setupWithViewPager(viewPager);
    }
    private void setupViewPager(ViewPager viewPager) {
        ViewPagerAdapter adapter = new
        ViewPagerAdapter(getSupportFragmentManager());
        //adapter.addFragment(new Home(), "Home");
        adapter.addFragment(new Donor(), "Donor");
        adapter.addFragment(new Charities(), "Charities");
```

```

        viewPager.setAdapter(adapter);
    }
    class ViewPagerAdapter extends FragmentPagerAdapter {
        private final List<Fragment> mFragmentManagerList = new ArrayList<>();
        private final List<String> mFragmentManagerTitleList = new ArrayList<>();
        public ViewPagerAdapter(FragmentManager manager) {
            super(manager);
        }
        @Override
        public Fragment getItem(int position) {
            return mFragmentManagerList.get(position);
        }
        @Override
        public int getCount() {
            return mFragmentManagerList.size();
        }
        public void addFragment(Fragment fragment, String title) {
            mFragmentManagerList.add(fragment);
            mFragmentManagerTitleList.add(title);
        }
        @Override
        public CharSequence getPageTitle(int position) {
            return mFragmentManagerTitleList.get(position);
        }
    }
    @Override
    public void onBackPressed() {
        if (doubleBackToExitPressedOnce) {
            super.onBackPressed();
            return;
        }
        this.doubleBackToExitPressedOnce = true;
        Toast.makeText(this, "Please click BACK again to exit",
        Toast.LENGTH_SHORT).show();
        new Handler().postDelayed(new Runnable() {
            @Override
            public void run() {
                doubleBackToExitPressedOnce = false;
            }
        }, 2000);
    }
    public boolean isSmsPermissionGranted() {

```

```

        return ContextCompat.checkSelfPermission(this,
Manifest.permission.READ_SMS) ==
PackageManager.PERMISSION_GRANTED;
    }
    /**
     * Request runtime SMS permission
     */
    private void requestReadAndSendSmsPermission() {
        if (ActivityCompat.shouldShowRequestPermissionRationale(this,
Manifest.permission.READ_SMS)) {
            // You may display a non-blocking explanation here, read more in the
documentation:
            // https://developer.android.com/training/permissions/requesting.html
        }
        ActivityCompat.requestPermissions(this, new
String[]{Manifest.permission.READ_SMS}, 1);
    }
}

```

Donor Side:

```

package ct.donating;

import android.app.ProgressDialog;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.ListView;
import android.widget.SimpleAdapter;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;

public class DonorHome extends AppCompatActivity {
    /** Called when the activity is first created. */

    @Override

```



```

public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.donorhome);
}

public void donating(View view){

    Intent i1=getIntent();
    Bundle b=i1.getExtras();
    String unm=b.getString("unm");
    Intent i = new Intent(DonorHome.this, Donating.class);
    i.putExtra("unm", unm);
    startActivity(i);

}

public void status(View view){

    Intent i1=getIntent();
    Bundle b=i1.getExtras();
    String unm=b.getString("unm");
    Intent i = new Intent(DonorHome.this, DonatingStatus.class);
    i.putExtra("unm",unm);
    startActivity(i);

}

public void delete(View view){

    Intent i1=getIntent();
    Bundle b=i1.getExtras();
    String unm=b.getString("unm");
    Intent i = new Intent(DonorHome.this, Delete.class);
    i.putExtra("unm",unm);
    startActivity(i);

}

public void cpwd(View view){

    Intent i1=getIntent();
    Bundle b=i1.getExtras();
    String unm=b.getString("unm");
    Intent i = new Intent(DonorHome.this, Changepwd.class);

```

```

        i.putExtra("unm",unm);
        startActivity(i);

    }

    // fill in the grid_item layout

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.main, menu);
        return true;
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            // action with ID action_refresh was selected
            case R.id.logout:
                Intent i=new Intent(DonorHome.this,MainActivity.class);
                startActivity(i);
                finish();
                break;
            // action with ID action_settings was selected
            default:
                break;
        }
        return true;
    }
}

```

Charities Side:

```

package ct.donating;
import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;

public class CharityHome extends AppCompatActivity {

    @Override
    public void onCreate(Bundle savedInstanceState) {

```

```

        super.onCreate(savedInstanceState);
        setContentView(R.layout.charityhome);
    }

    public void request(View view){

        Intent i1=getIntent();
        Bundle b=i1.getExtras();
        String unm=b.getString("unm");
        Intent i = new Intent(CharityHome.this, DonorsRequest.class);
        i.putExtra("unm", unm);
        startActivity(i);

    }

    public void status(View view){

        Intent i1=getIntent();
        Bundle b=i1.getExtras();
        String unm=b.getString("unm");
        Intent i = new Intent(CharityHome.this, AcceptRequests.class);
        i.putExtra("unm",unm);
        startActivity(i);

    }

    public void cpwd(View view){

        Intent i1=getIntent();
        Bundle b=i1.getExtras();
        String unm=b.getString("unm");
        Intent i = new Intent(CharityHome.this, Changepwd1.class);
        i.putExtra("unm",unm);
        startActivity(i);

    }

    //fill in the grid_item layout

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.main, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            // action with ID action_refresh was selected
            case R.id.logout:

```

```
        Intent i=new Intent(CharityHome.this,MainActivity.class);
        startActivity(i);
        finish();
        break;
        // action with ID action_settings was selected
        default:
            break;
    }
    return true;
}
```

III. References

- Reduction of Food Wastage through Android Application – Make You Smile - By Mafishan Ali, Sana Sheikh, Yumna Sohail
- D' WORLD: Blood Donation App Using Android Dr. A. Meiappane1, K. Logavignesh2, R. Prasanna3, T.Sakthivel4
- Prof. Snigdha et.al, “ Android Blood Bank “, International Journal of Advanced Research in Computer and Communication Engineering, Vol 4, No.11, November 2015, pp: 86-88.
- Narendra Gupta et .al, “MBB: A Life-Saving Application“, International Journal For Research in Emerging Science And Technology, Vol 2, No 1, March-2015, pp: 326-330, ISSN:2349- 7610.
- Sultan Turhan, “An Android Application Volunteer Blood Donors”, ICBB-2015, DOI:10.5121/csit .2015.51103, pp:23-3p
- Sayali Dhond et al., “Android-Based Health Application in Cloud Computing For Blood Bank”, International Engineering Research Journal (IERJ) Vol 1, Issue 9, 2015, pp: 868-870, ISSN 2395-1621