IEEE 830 Template

vSewa

Group_no-23

Software Requirements Specifications

Submitted By:
Md Asif Anwar 2017TT10922
Dipanshu Sharma 2017EE10306
Mohammed Zia Kamran 2017TT10925
Poojith Gundelli 2016PH10581

1. Introduction

1.1 Purpose

Our purpose is to provide help locally to old age people, PWD, needy by connecting with people who want to volunteer, through the app and NGOs and increase awareness in people by motivating them with incentives like discount coupons and goodies.

1.2 Scope

Initially, we release this in local area to test the app and expand it further after fixing bugs (if any). This can be extended for the NGOs and other social service camps where people nearby can sign up to volunteer for organizations or even join them. This can be extended for helping people in emergency situations (accidents, travel needs, etc) if they are stuck in remote areas. A reward system helps in the active participation of volunteers and increases users by ensuring motivation in terms of rewards. It can be used to search for blood donors. Social media apps notification via linked account can be used to alert the volunteer. Further a web portal can be made to volunteer. The use cases can be increased in various directions, but the core of the platform is to connect volunteers and needy individuals, NGOs.

1.3 Definitions, Acronyms and Abbreviations

- API—Application Programming Interface
- AD---Active Directory
- CRM---Customer Relationship Management
- SAP---System Applications and Products
- VSS----Visual Source Safe
- UI---User Interface
- REST-- Representational State Transfer

1.4 References

"Computing Acronyms and Abbreviations". All Acronyms. Retrieved 2008-07-12.
https://firebase.google.com/ Firebase

1.5 Overview

We are using firebase to store user's data, image, profile pic and to authenticate user's while login and signup. Firebase provides easy management of database, authentication. Firebase easily create database table and it's easy to use in android studio, no need to host any server for database. Firebase notify the data updating/deletion in real time.

We are making an android app which help people to find volunteer to help them. It has messaging service, Map navigation, rating system, point system and many more features. We are using java language as a backend. For database and authentication, we are using Firebase (as it's gives free 1 GB real time data storage).

2. Overall Description

2.1 Product Perspective

The main segments in the app are:

1. registration/login
2. Needy or user can raise a request for help
3. Nearest Volunteer get the notification using google map API
4. Messaging environment to connect volunteer and needy
5. NGOs can interact with volunteer to recruit them

This android application allows users (Needy) to raise help request through the app to find the nearest volunteer who are available to help. This app intends to remove the difficulties of needy, old age, NGOs, when they required voluntarily help. Now they can easily find this help.

2.1.1 System Interface

The firebase database is stored as JSON and is synchronized with every user connected in real time
Documentation- https://firebase.google.com/docs/database
Unlike SQL database, Firebase has no tables or records. When data is added it becomes a node.
Documentation-
https://firebase.google.com/docs/database/web/structure-data
We can use the cloud Firestore managed export and import service to recover from accidental deletion of data and to export data for offline processing. Data can be export to another cloud firestore also.
Documentation-https://firebase.google.com/docs/firestore/manage-data/export-import

2.1.2 User interface`

- Signup/Login Page-Here user login to the app
- Raising request Page- Here needy raise the request for help, then nearest volunteer get the notification
- Map Page-Here volunteers can see the path of the needy or vice versa. SOS button for help on map page.
- Rating for a volunteer Page-Here user can see the ratings of -area, volunteer-user, needy-user.
- Messaging Page- (personal or group) messaging interface to communicate between two users

2.1.3 Hardware Interface

The hardware interfacing is taken care by the Android API and we don't have to deal with the details of hardware interfacing. We will update in case of any hardware interfacing requirement rising during the project.

2.1.4 Software Interface

Firebase – Database management, user Authentication, Hosting
Documentation- https://firebase.google.com/
Android OS- Application developed on Android operating system to handle operating functions

Documentation- https://developer.android.com/
Google Maps API- Calculating shortest distance and ratings
Documentation-https://developers.google.com/maps/documentation
Firebase cloud Messaging API-We are using this API for messaging service
Documentation- https://firebase.google.com/docs/cloud-messaging

2.1.5 Communication Interfaces

The communication interfaces are also handled by Android system API and Firebase and are handled in a abstract way by them and we don't have to deal with them.

2.1.6 Memory Constraints

We have storage of 1 GB memory for storing all the data as provided by firebase in real-time database.
Documentation- https://developer.android.com/

2.1.7 Operations

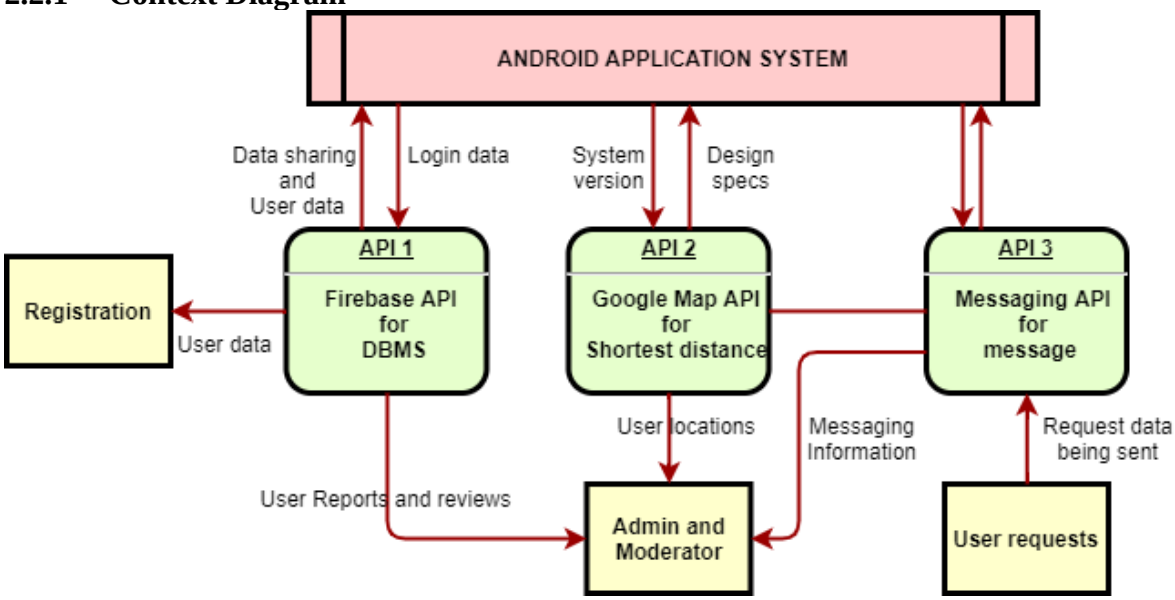
Needy-user ask for help by initiating the operations, then nearest volunteer (who are active) get the notification
Volunteer-user can opt to volunteer or activate DND mode
Organisation-user can initiate operations to get the help from volunteer to organise an event.
User cans get the history of their volunteering and rating rated by another user.
User can download the data as a backup in android studio in json format (Volunteering History, Volunteer Required history, rating, rated etc)

2.1.8 Site Adaption Requirements

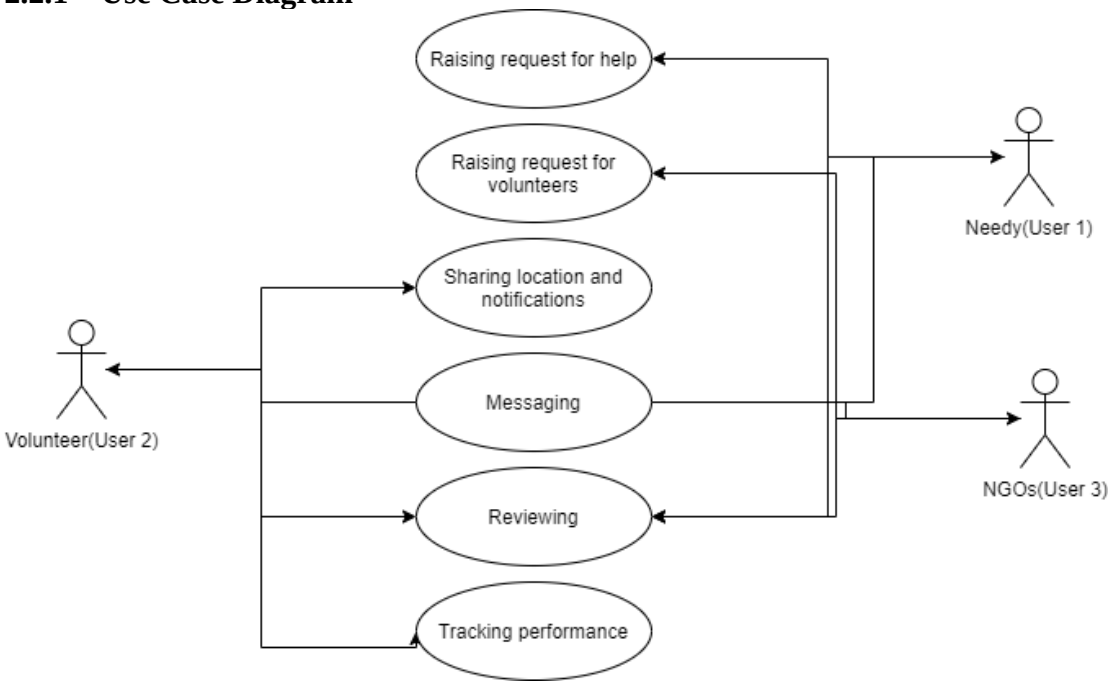
We can expand to web application, Ios also. As our data is on firebase so we can easily move to IOs or web application. Firebase update data in real time (store and sync) from any platform.
Documentation-https://firebase.google.com/docs/database

2.2 Product Functions

2.2.1 Context Diagram



2.2.1 Use Case Diagram



2.2.2 Use Case Descriptions / introductions

The use cases are described below as-

Emergency help: The needy or user who is in distress can raise a request on the application and he will get connected to the nearby people willing to help. The volunteers can connect with them and help.

Messaging: The users can interact with each other with the help of application's messaging feature and they have the privacy to hide their important details

Events: The NGOs can interact with volunteers to recruit for their organisational events and day-to-day functions.

2.3 User Characteristics

The users are generally characterised as Needy and Volunteer.

Needy- the user like Old, disable or any Organisation who wants any volunteer for some help in emergency or organising any kind of social event.

Volunteer- the users are mostly youth, those who'd like to serve for society can serve as volunteer for social events or help in emergencies.

NGOs- the organisations looking for volunteers for organising events and full-time recruitment to look at the day-to-day working of the organisation.

2.4 Constraints

There are things like user authentication, syncing data to server and other functions which gives constrain like compatibility with device used in and connections.

Regulatory policy: You can't sign up for make two account with same details.

Hardware limitation: We can store some data in device, we must reduce the space used by user activity detail in database. GPS need to be enabled for meeting the requirement for requests.

Criticality of the application: Can't use as both, needy and volunteer at same time.

Safety and security consideration: Firebases use password with special character and at least one upper case and one lower case and one number for login. In case of crash of software, we are using fire base to store data which gives privilege to recover the loss.

2.5 Assumptions and Dependencies

Let us assume that the operating system for the device in which it'll use is Android and the user is needy, an NGO requesting for volunteers:

It will search according to specification match to interest of volunteer and will have the ability to send personal invitation or request for joining the event.

Assuming, that the volunteer has attended events before which will decide the interest field and also there is similar option for volunteer that

NGO will host their event notification for all, and individual can request to attain.

2.6 Apportioning of Requirements

The requirement that may be delayed for the further version of the system are some fields like better suggestions for event organiser based on better algorithm, and good technique for the communication so that needy should not have to wait for the response or match after requesting for volunteer.

And link from social media so that every event attended by volunteer can be posted for the motivations.

3. Specific Requirements

3.1 External interface

3.2 Functional Requirements

3.2.1 Use Case 1

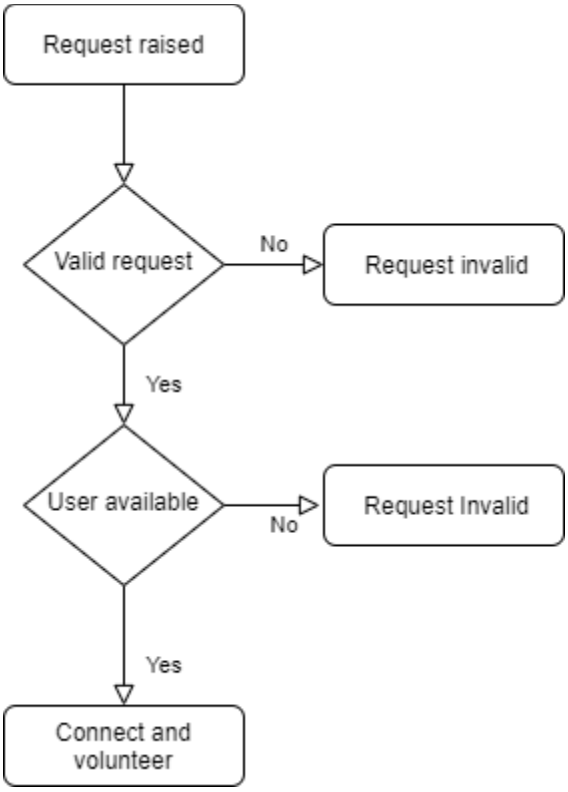
3.2.1.1 Use Case- Needy asking for help

Brief Description- The use case describes how the user(Old/PwD) will raise a request for help and the other users(Volunteers) can connect with them.
Actors- Needy(Old/PwD), Volunteer
Pre-conditions- There is an active internet connection with the needy. There are volunteers available in the vicinity.
Basic Flow of events-

1. The needy logs in and raises a request with the application with a specific query and location
2. Use case: Validating if it's a genuine request
3. The request is sent via notification to all the volunteers available in that area within a range pre-decided by the requestor
4. The volunteer interested connects with the needy and goes on to further chat with the needy
5. The volunteer confirms the location and the kind of help required and proceeds with helping
6. The volunteer goes on to help at the location provided by the application
7. The request is met and the needy and the volunteer can decide to stay in touch as friends or dicsconnect
8. The users rate each other based on the experience and block/report fake users/requests

Alternative flow-
Invalid user- The user trying to raise the request is a user with bad reviews or is not yet verified by the application as an Old/PwD person who needs help. We end such requests at this point
Invalid location- The location from which the request is being raised is not available to the system. The application will prompt the user to turn the location precisely ON.
Wrong Information- The volunteer available on the application is different from the volunteer present in person. The needy can block/report the volunteer and simultaneously send SOS signal
No response from needy- The volunteer has the option to cancel the request and report it as unresponsive to the admin
Quit- The needy can cancel the request at any point of time if they feel they don't need help or the request is fulfilled.
Key scenarios- No users available in the area
Post-conditions-
Successful condition-
The user has successfully received help and the request has been fulfilled
Failed condition-
The user has failed to receive help. The log has been noted in the system.

3.2.1.2 Process Flow Diagram / Data Flow Diagram



3.3 Performance Requirements

The platform will be first introduced in a small environment like inside a college and in this phase, it will be able to support more than 1000 users as firebase support 1 GB free data storage. it can be later introduced to website and then work across the whole country.

3.4 Logical Database Requirements

The Database would contain the information provided by all the users. Each user will have a unique user id. Each User's data would be deleted within 30 days, when user deactivated the account. We are using firebase as our database; it stores data as JSON and is synchronized with every user connected in real time. Firebase provides us with 1GB data initially which can be expanded later depending on our requirements.

3.5 Design Constraints

Android is used to develop our software so that it is used by most of the people in the world (99%). Only problem is we cannot reach users who do not have android mobiles which is very little percentage of population.

3.6 Software System Attributes

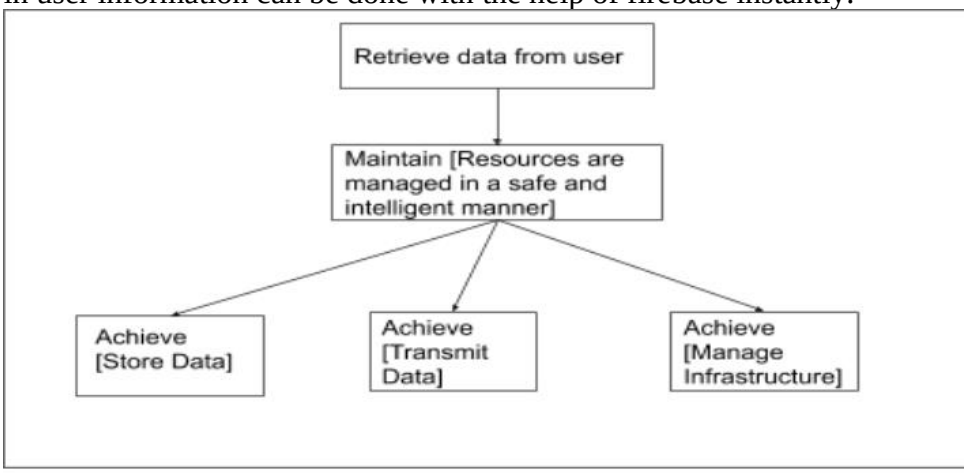
3.6.1 Reliability: Software is reliable to users once it achieves active participation of volunteers. This is done with the help of reward system.

3.6.2 Availability: Since all the information is stored in firebase the system can be recovered quickly by logging with user id and password after

logging into app from any other device. Any change in user profile is immediately stored in firebase and a checkpoint is created. All the information is easily accessed to user after restart as it is stored in internal storage.

3.6.3 **Security:** All personal information of the user is safe as we are using Gmail and firebase to login and store information. Unless google and firebase are hacked, we are safe. Unless google and firebase are hacked, we are safe.

3.6.4 **Maintainability:** Software can be regularly updated and maintained as each sub-system of the software is being developed as modules. Any update in user information can be done with the help of firebase instantly. Software can be regularly updated and maintained as each sub-system of the software is being developed as modules. Any update in user information can be done with the help of firebase instantly.



3.6.5 **Portability:** The software has good portability. It can be downloaded from google play store. So, user can use this app on any android phone. Though it is limited to android it can also be extended to iOS and as web application.

3.7 Organising Specific Requirements

3.8 Additional Comments

4. Supporting Information

4.1 Table of Contents & index

1. Introduction
    1.1. Purpose
    1.2. Scope
    1.3. Definitions, Acronyms and Abbreviations
    1.4. References
    1.5. Overview
2. Overall Description
    2.1. Product Perspective
        2.1.1. System Interface

2.1.2. User interface
2.1.3. Hardware Interface
2.1.4. Software Interface
2.1.5. Communication Interfaces
2.1.6. Memory Constraints
2.1.7. Operations
2.1.8. Site Adaption Requirements
2.2. Product Functions
2.2.1. Context Diagram
2.2.2. Use Case Diagram
2.2.3. Use Case Descriptions / introductions
2.3. User Characteristics
2.4. Constraints
2.5. Assumptions and Dependencies
2.6. Apportioning of Requirements
3. Specific Requirements
3.1. External interface
3.2. Functional Requirements
3.2.1. Use Case 1
3.2.1.1. Use Case
3.2.1.2. Process Flow Diagram / Data Flow Diagram
3.2.2. Use Case 2
3.3. Performance Requirements
3.4. Logical Database Requirements
3.5. Design Constraints
3.6. Software System Attributes
3.6.1. Reliability
3.6.2. Availability
3.6.3. Security
3.6.4. Maintainability
3.6.5. Portability
3.7. Organising Specific Requirements
3.8. Additional Comments
4. Supporting Information
4.1. Table of Contents & index
4.2. Appendixes
4.2.1. Screen Layouts (including validations)
4.2.2. Report Layouts

4.2 Appendixes

4.2.1 Screen Layouts (including validations)

4.2.2 Report Layouts