Software Design Specification

for

Donatio

A Donation Collection Application

Prepared by

Uzair Khan K152183

FAST-NUCES

November 7, 2017

Contents

1 Introduction

1.1 Purpose

This document will define the design of an Android application whose objective is specified below. This document contains information about the implementation of the classes in the program , as well as the design of the classes, their attributes and their functions. The interaction between classes are shown in the model diagrams at the end of this document.

1.2 Scope

This design specification is to be used to implement Donatio, a donation Collection Application.

1.3 Objective

Donatio is a donation collection application designed to implement an efficient system where donors are able to be connected with those who require donations. Other features of the application include the ability for users to view donation collection centres nearest to their location.

2 Overall Description

2.1 Product Perspective

2.1.1 Design Method

The design of this product utilizes an object-oriented approach.

2.1.2 User Interfaces

The normal user of this product will be interfacing with the android application. Admins of donation centres and charities will also be interacting with a website for this system.

2.1.3 Hardware interfaces

The application can run on most android devices.

2.1.4 Software Interfaces

The software will run on the Android JVM. The website will run on a web server.

2.1.5 Memory Constraints

This application is designed to not be memory intensive.

2.2 Product Functions

- Showing donation centres-This function shows the user donation centres nearest to their location filtered according to their choices.
- Become a volunteer-This function allows the user to peruse the list of registered charities and NGO's and apply to be a volunteer for them.
- Request donations-This function allows users is need to ask for donations for themselves or for some cause. Their request is verified by one of the many NGO's charities or other donation centers registered with the application.
- View volunteers-Admins of NGO's and charities can view the list of volunteers that were registered to them through this product.
- Issue emergency notification- Admins of NGO's, charities and donation centers can issue emergency donation requests in case of a city-wide emergency situation.

2.3 User Characteristics

The general characteristics of the intended users, include

- technical literacy- Ability to use a phone application and optionally a website.

2.4 Constraints

This application can only run on most android phones.

2.5 Apportioning of Requirements

There are requirements apportioned to later releases of the product.

3 Design Considerations

3.1 Operating Environment

Donatio is intended to be used on an Android device. Website will be hosted on MySQL server.

3.2 Fault Tolerant Design

Application errors will be handled by common fault detection services.

3.3 Design Conventions

The Donatio software design uses the Object Oriented methodology described in ”The Unified Software Development Process” by Ivar Jacobsen, Grady Booch and James Rumbaugh. (Booch, 1999)

3.4 Architectural Design

The software capabilities and requirements specified in the Donatio Software Requirements Specification are transformed into programs that will execute on an Android device. Software items are partitioned into classes and packages using Object Oriented methodology to maximize encapsulation and minimize interfaces. Packages are then built (compiled and linked) into executable programs.

3.5 User Interface

The user or simulator operator interfaces via a text input screen. The user is prompted for several values in order to perform the calculations.

3.5.1 Expected Input

The user is prompted to:

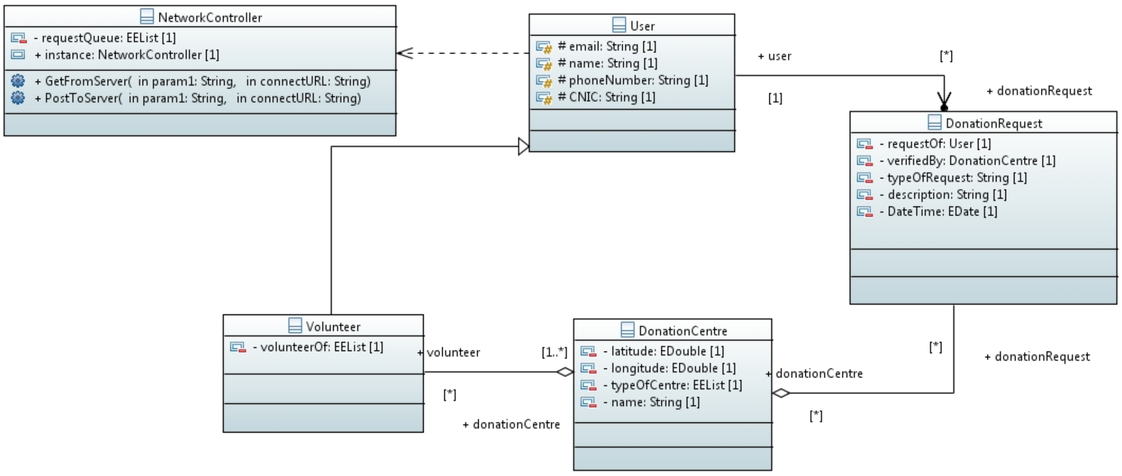
- Sign in using their Google account
- Pick the types of donation centres they want to see on the map
- Enter their phone number and their CNIC if they want to volunteer or request a donation
- Enter other relevant information related to their donation request

3.5.2 Output

The user can see multiple outputs according to the different functions. They can, most commonly, see Google Maps with markers for the respective donation centres. Output can also be in the form of messages from the server in case the user had sent a volunteer or donation request.

4 System Architecture

4.1 View of Classes (Figure 1)


NetworkController
- requestQueue: EEList [1]
+ instance: NetworkController [1]
+ GetFromServer(in param1: String, in connectURL: String)
+ PostToServer(in param1: String, in connectURL: String)
User
# email: String [1]
# name: String [1]
# phoneNumber: String [1]
# CNIC: String [1]
+ user
[*]
[1]
+ donationRequest
DonationRequest
- requestOf: User [1]
- verifiedBy: DonationCentre [1]
- typeOfRequest: String [1]
- description: String [1]
- DateTime: EDate [1]
Volunteer
- volunteerOf: EEList [1]
+ volunteer
[1..*]
[*]
+ donationCentre
DonationCentre
- latitude: EDouble [1]
- longitude: EDouble [1]
- typeOfCentre: EEList [1]
- name: String [1]
+ donationCentre
[*]
[*]
+ donationRequest

4.2 Individual Classes of System

4.2.1 User

The user class is the base class of user types. It is used to store the information of the user if the user is not a volunteer. It is also used to populate lists of donation requests as every donation request has a reference of its user.

1. Attributes of User

    - email : Email holds the email address of the user that they used for Google Sign in.
    - name : The name of the current user. Obtained from the Google API client with the default scope when the user uses Google Sign in.
    - phoneNumber : Optional information that the user can provide. Their current contact number.
    - CNIC : Also optional information that the user can provide. The identity code of a citizen of Pakistan.

4.2.2 Volunteer

The volunteer class is extended from the User base class. In this class, it is necessary for the volunteer to have provided a phone number and a valid CNIC.

1. Attributes of Volunteer

    - volunteerOf: A list of the charities and NGO's that this user is a volunteer of.

4.2.3 DonationCentre

This class represents the donation centres, charities and NGO's that are a part of Donatio.

1. Attributes of DonationCentre
    - latitude: Latitude co-ordinates that specify the physical location of the Centre.
    - longitude: Longitude co-ordinates that specify the physical location of the Centre.
    - typeOfCentre: A list of the types of donations that this Centre or Charity accepts. Types include Food, Money, Blood, Clothes and Books.
    - name : The name of the donation Centre.

4.2.4 DonationRequest

This class holds the data of requests made by users for donations. It can also represent requests made by admins of donation centres in case of an emergency.

1. Attributes of DonationRequest
    - requestOf : The user who made this request.
    - verifiedBy: The donation centre that verified this request and approved it.
    - typeOfRequest: The type of donation that the user requests.
    - Description: Additional optional information supplied by the user who made the request.
    - DateTime: The date and time when the user made the request.

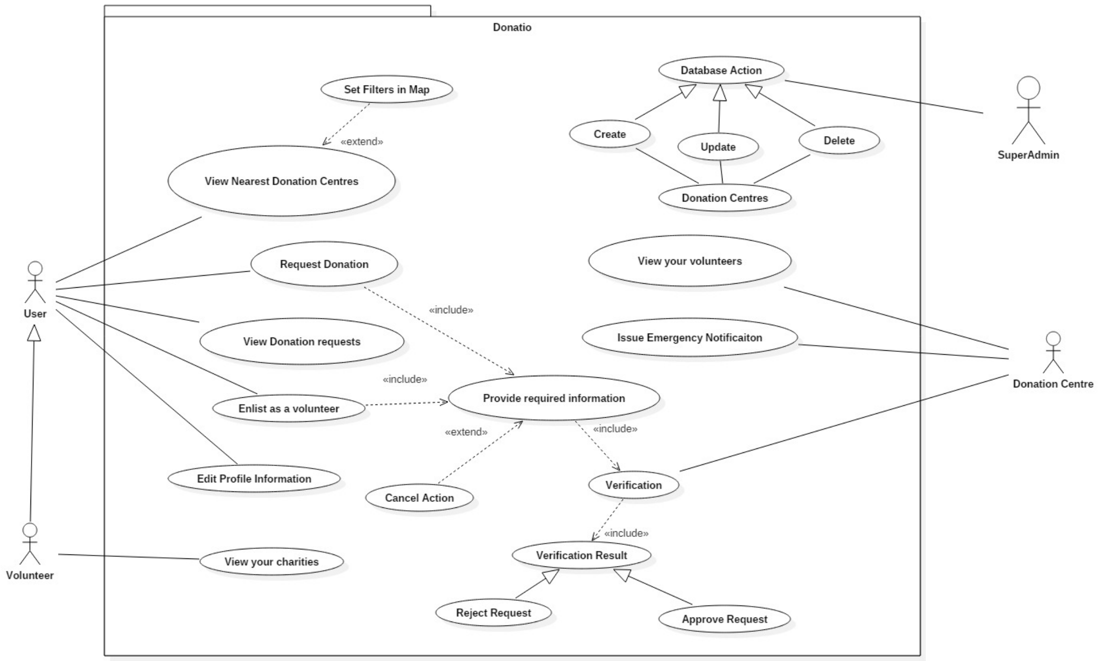
4.2.5 NetworkConnector

A singleton connector class used by the application. Does network operations asynchronously and callbacks with the results to the calling thread.

1. Attributes of NetworkConnector
    - requestQueue ; Holds the list of network requests..
    - instance: The context for the NetworkConnector as this a singular instance of this class handles all network operations.
2. Functions of NetworkConnector
    - GetFromServer : Receives the parameters to append to URL to query server and get result.
    - PostToServer : Posts data to server.

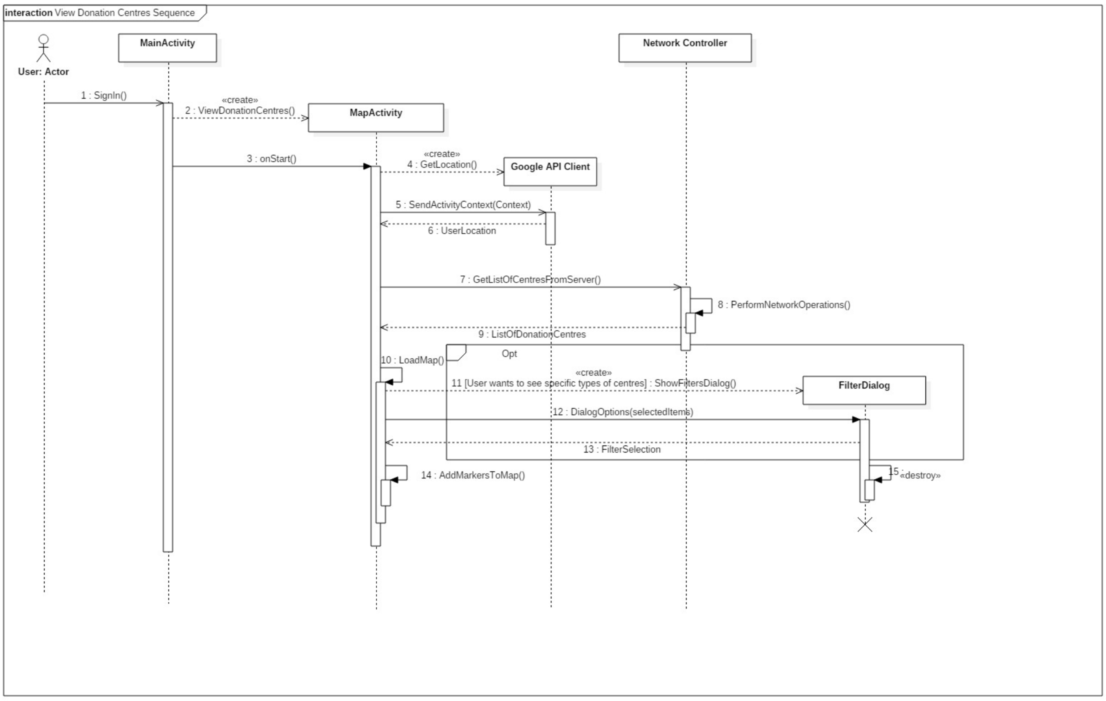
5 Figures

5.1 Use Cases (Figure 2)

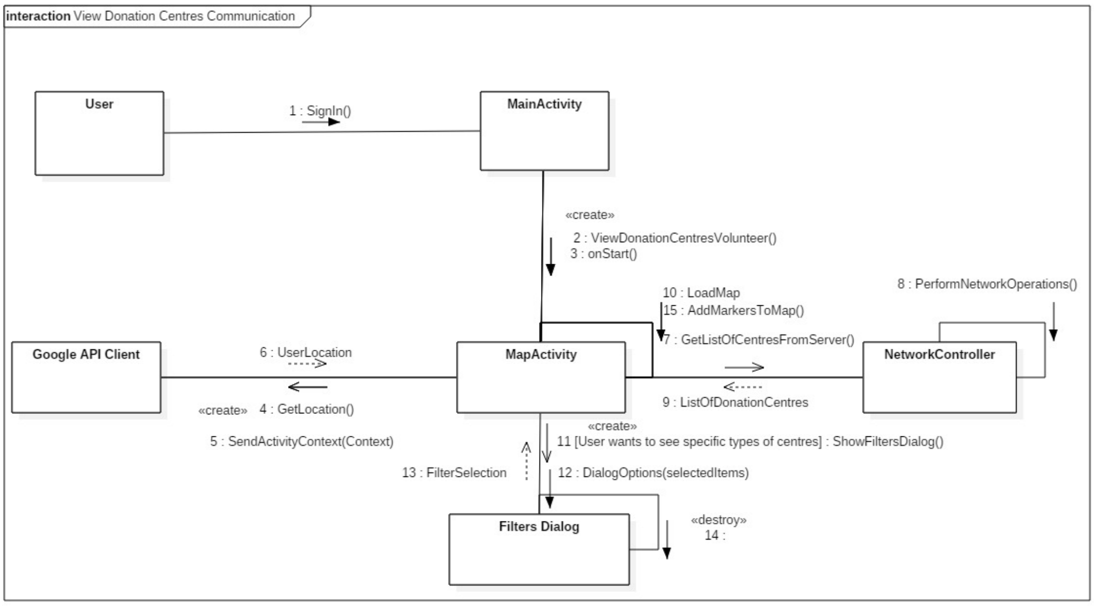


5.2 View Donation Centres

5.2.1 View Donation Centres Sequence (Figure 3)

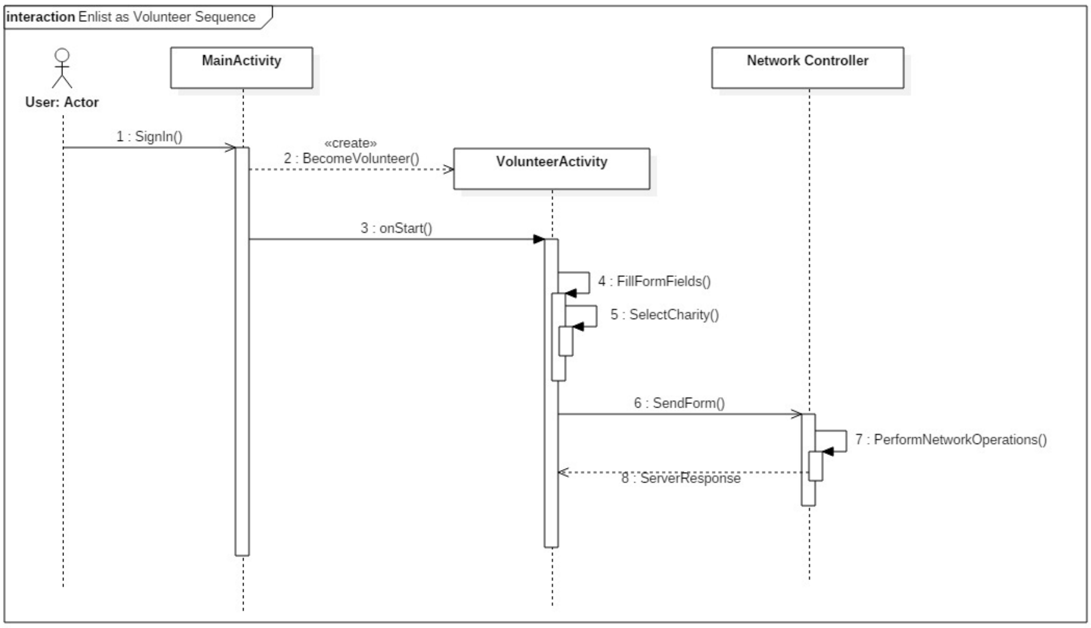

interaction View Donation Centres Sequence
User: Actor
MainActivity
MapActivity
Google API Client
Network Controller
FilterDialog
1 : SignIn()
«create»
2 : ViewDonationCentres()
3 : onStart()
«create»
4 : GetLocation()
5 : SendActivityContext(Context)
6 : UserLocation
7 : GetListOfCentresFromServer()
8 : PerformNetworkOperations()
9 : ListOfDonationCentres
Opt
10 : LoadMap()
«create»
11 [User wants to see specific types of centres] : ShowFiltersDialog()
12 : DialogOptions(selectedItems)
13 : FilterSelection
14 : AddMarkersToMap()
15 : «destroy»

5.2.2 View Donation Centres Collaboration (Figure 4)

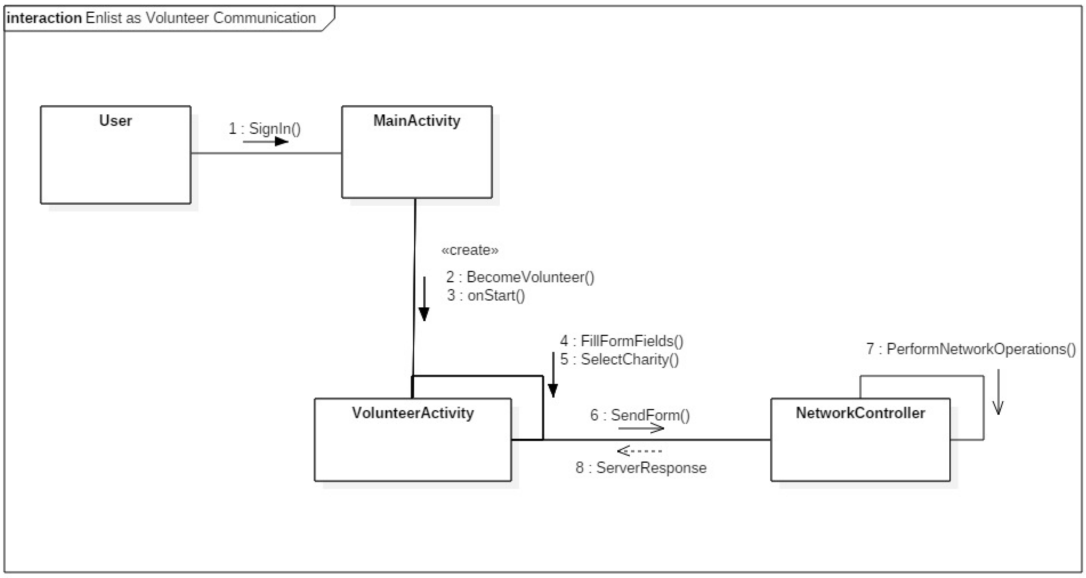

interaction View Donation Centres Communication
User
1 : SignIn()
MainActivity
«create»
2 : ViewDonationCentresVolunteer()
3 : onStart()
10 : LoadMap
15 : AddMarkersToMap()
8 : PerformNetworkOperations()
7 : GetListOfCentresFromServer()
Google API Client
6 : UserLocation
MapActivity
NetworkController
«create» 4 : GetLocation()
9 : ListOfDonationCentres
5 : SendActivityContext(Context)
«create»
11 [User wants to see specific types of centres] : ShowFiltersDialog()
13 : FilterSelection
12 : DialogOptions(selectedItems)
Filters Dialog
«destroy»
14 :

5.3 Enlist as Volunteer

5.3.1 Enlist as Volunteer Sequence (Figure 5)

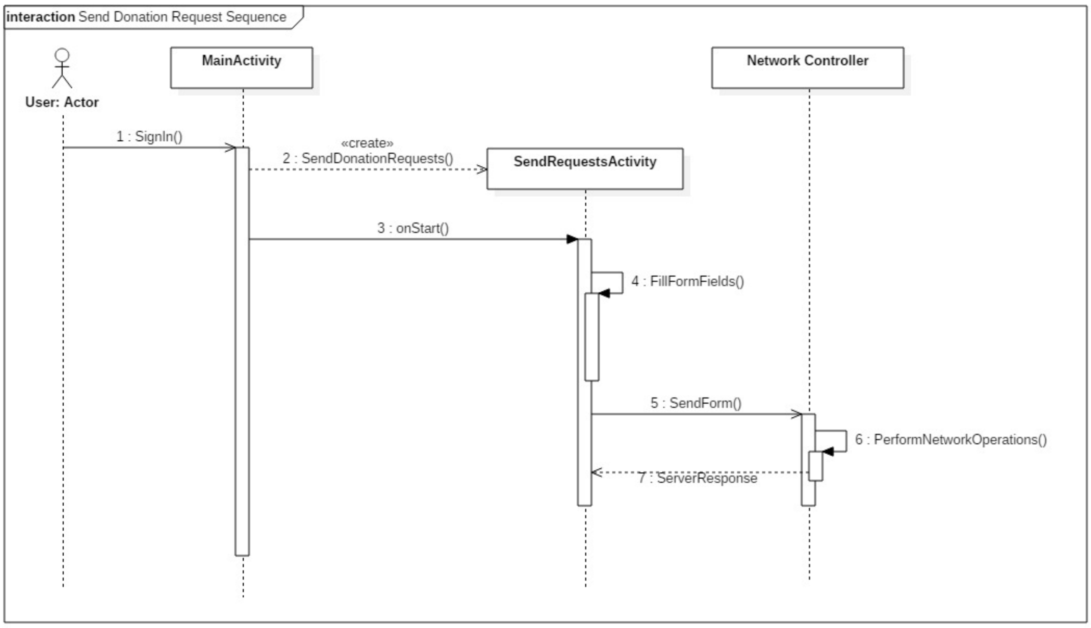


5.3.2 Enlist as Volunteer Collaboration (Figure 6)

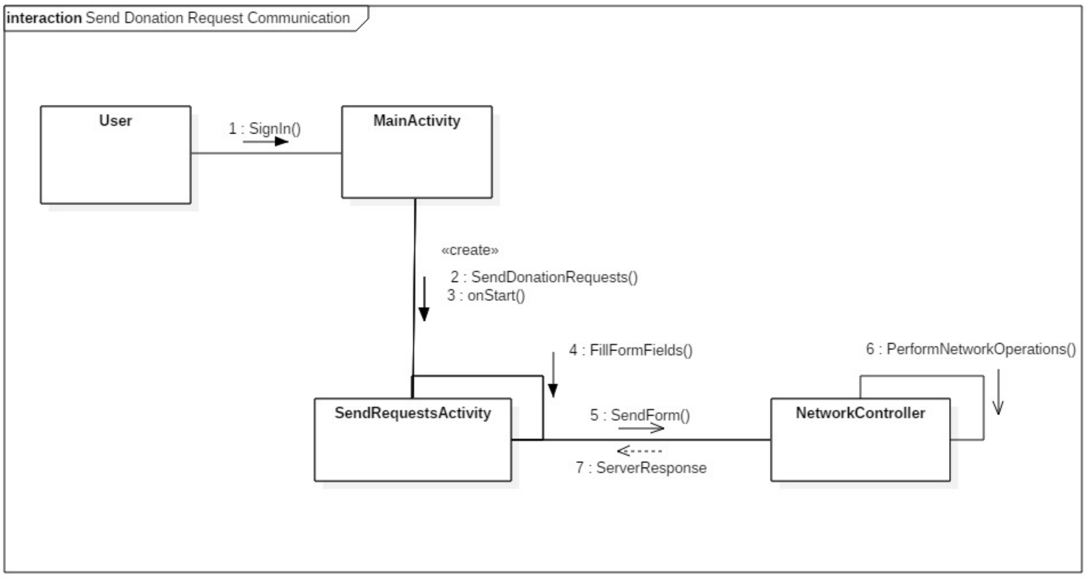


5.4 Send Donation Request

5.4.1 Send Donation Request Sequence (Figure 7)

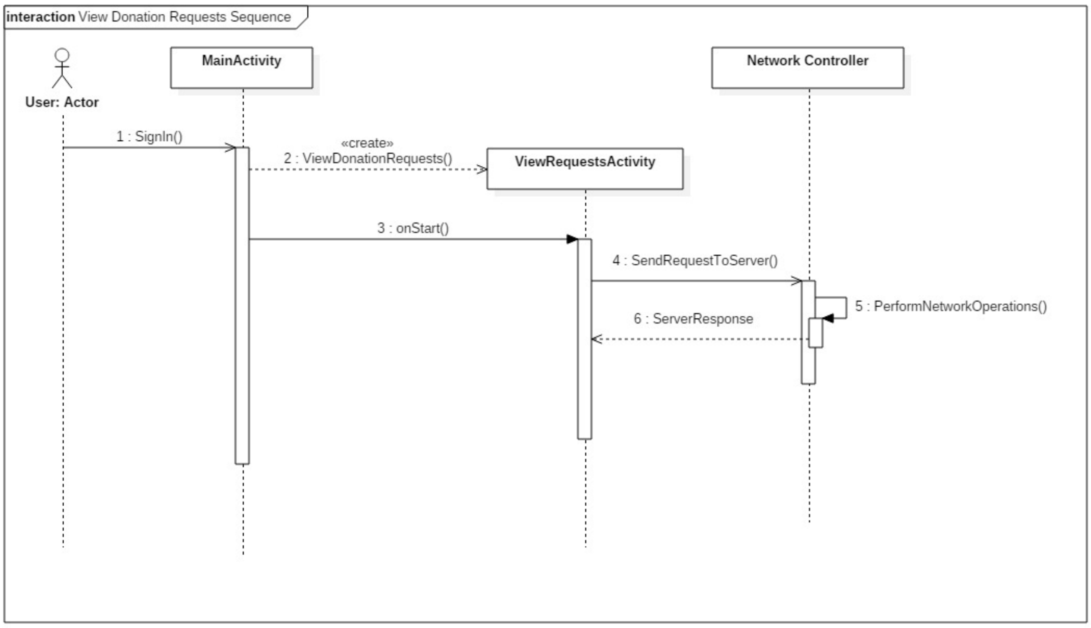


5.4.2 Send Donation Request Collaboration (Figure 8)

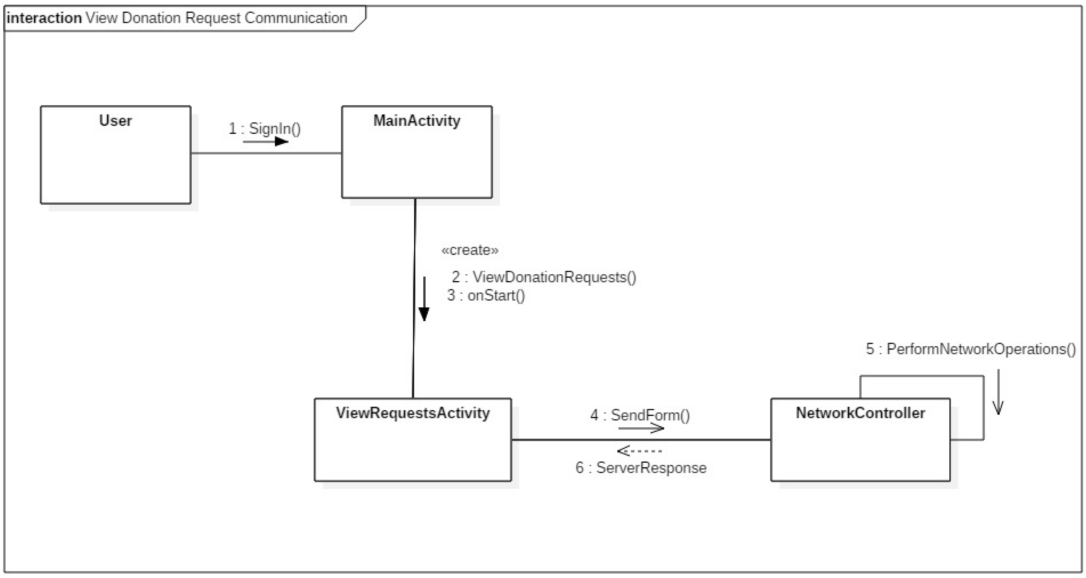


5.5 View Donation Request

5.5.1 View Donation Request Sequence (Figure 9)


interaction View Donation Requests Sequence
User: Actor
MainActivity
Network Controller
1 : SignIn()
«create»
2 : ViewDonationRequests()
ViewRequestsActivity
3 : onStart()
4 : SendRequestToServer()
5 : PerformNetworkOperations()
6 : ServerResponse

5.5.2 View Donation Request Collaboration (Figure 10)


interaction View Donation Request Communication
User
1 : SignIn()
MainActivity
«create»
2 : ViewDonationRequests()
3 : onStart()
5 : PerformNetworkOperations()
ViewRequestsActivity
4 : SendForm()
6 : ServerResponse
NetworkController