Requirements and Analysis for Good Deeds

## Authors

## Date

# Introduction

The Good Deeds application aims to target the need for a service or a need to be a good samaritan without any monetary transaction. The basic functionality is to be able to put out an add-like post if you want to get help with something or if you want to help your fellow human beings. This should be shown to users in a way that is easy to navigate so that the users can filter and find exactly what service they need.

The stakeholders in this application are anyone who needs help with something or wants to help others and want to do it in a more old fashioned way without any monetary transaction.

# Definitions, acronyms, and abbreviations

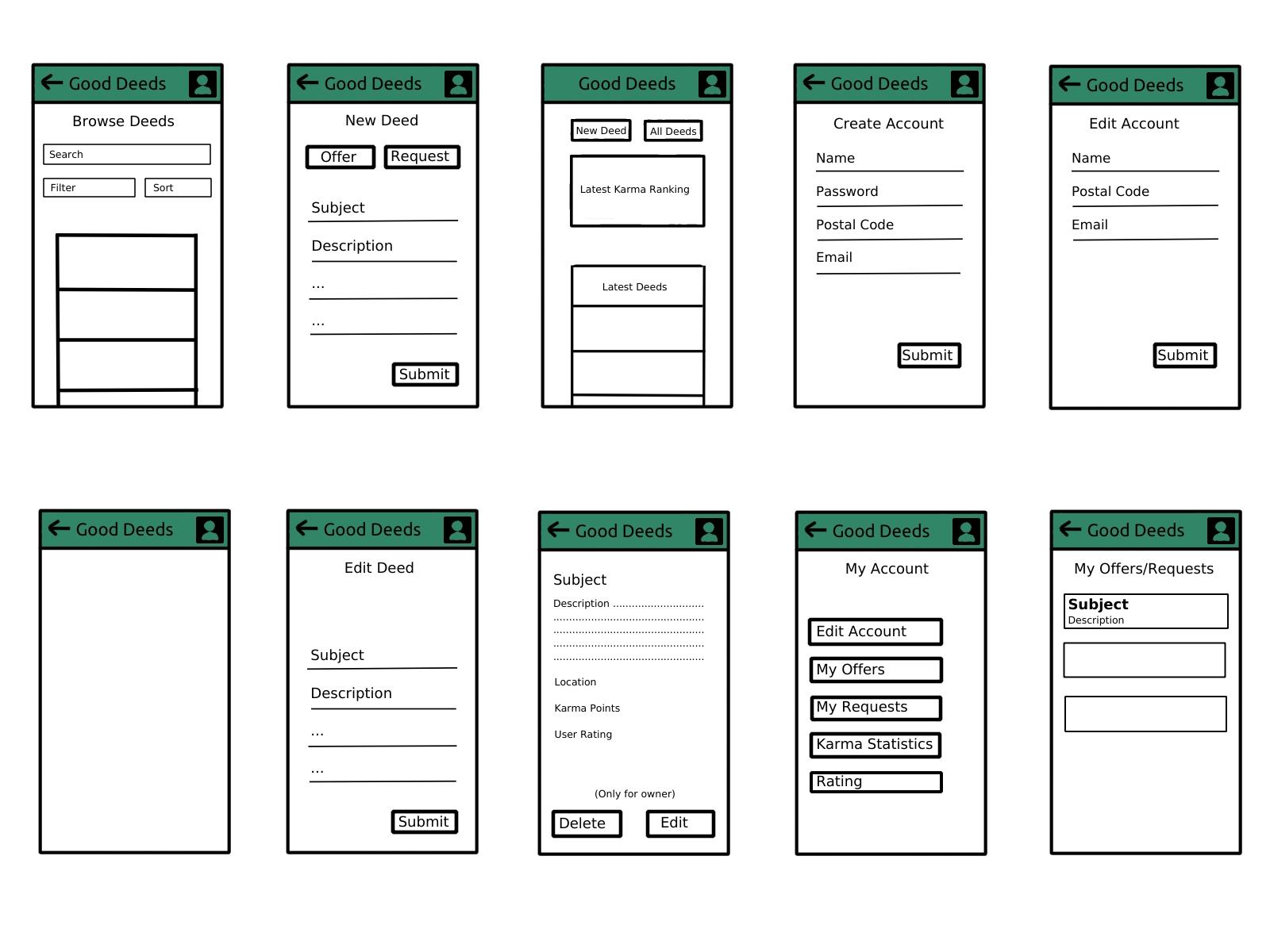
# Requirements

## User Stories

## Definition of Done

* Unit tests should all pass
* Code should have been reviewed by at least two developers
* Acceptance criteria should be met
* Code should be merged with master
* The user story should be marked as complete
* Domain design should be reviewed and updated if any change

## User interface



## Domain model

## 

## **Class responsibilities**

***-ActiveOffersActivity***

Responsible for showing active offers. Holds a recyclerview, adapter, layout manager.

***-DeedAdapter***

Responsible of taking Deeds and put them in the recyclerview.

***-CreateOffer***

Responsible for handling input in the view.

***-DeedController***

Handles input from views that handles deeds, and sends if necessary data to DeedRepository.

***-DeedRepository***

Handles the layer between service and database ( got no such yet).

***-IDeed***

Acts as a interface for Deed.

***-Deed***

Responsible for holding the “blueprint” for a Deed.

***-Good Deeds***

Acts as the model for the project, holds data which views might need to have or methods that’s necessary for the project to work properly.

***-Account***

Hold what is necessary for an Account to be created. And methods needed.

***-Account Repository***

Handles the layer between service and database ( Got no such yet).

***-Account Controller***

Responsible for handling the input from views that handles Accounts, and sends if necessary data to DeedRepository.

***-MainActivity***

Responsible for handling events in the MainActivity view.

***-Login***

Responsible for handling events in the Login view.

***-CreateAccount***

Responsible for handling events in the CreateAccount View.

***-EditAccount***

Responsible for handling events in the EditAccount View.

## References