|  |  |
| --- | --- |
| Author | Thanh Vo |
| Date | 26-06-2018 |
| Version | 3.35 |
| Status | Reviewing |
| Project | Give and Receive Application |

**SRS\_GiVE\_AND\_RECEIVE APP**

**TABLE OF CONTENTS**

[1](#_1fob9te) INTRODUCTION 5

[1.1](#_3znysh7) Purpose 5

[1.2](#_2et92p0) Scope 5

[2](#_tyjcwt) REFERENCED DOCUMENTS 5

[2.1](#_3dy6vkm) Controlling Documents 5

[2.2](#_3rdcrjn) Controlled Documents 5

[2.3](#_26in1rg) Background Information 5

[3](#_35nkun2) ACTORS 7

[4](#_44sinio) OVERVIEW 7

[5](#_2jxsxqh) FEATURES 8

[5.1](#_z337ya) Device Orientation 8

[5.2](#_3j2qqm3) Startup 8

[5.2.1](#_1y810tw) User Interface 8

[5.2.2](#_4i7ojhp) Business Logic 8

[5.3](#_2xcytpi) Main 8

[5.3.1](#_1ci93xb) User Interface 8

[5.3.2](#_3whwml4) Business Logic 8

[5.3.3](#_2bn6wsx) User Cases 9

[5.4](#_qsh70q) Login 9

[5.4.1](#_3as4poj) User Interface 9

[5.4.2](#_1pxezwc) Business Logic 9

[5.4.2.1](#_49x2ik5) Login using Facebook/Google account 9

[5.4.2.2](#_2p2csry) Remember credentials 10

[5.4.2.3](#_2szc72q) Lost internet connection 10

[5.4.3](#_147n2zr) Use Cases 10

[5.5](#_32hioqz) Create/Modify/Delete item in giver’s list 11

[5.5.1](#_1hmsyys) User Interface 11

[5.5.2](#_41mghml) Business Logic 11

[5.5.2.1](#_2grqrue) Create new item 11

[5.5.2.2](#_184mhaj) Post items 11

[5.5.2.3](#_vx1227) Modify items 11

[5.5.2.4](#_3fwokq0) Delete items 12

[5.5.3](#_1v1yuxt) Use Cases 12

[5.6](#_28h4qwu) Claim items to Taker 13

[5.6.1](#_nmf14n) User Interface 13

[5.6.2](#_37m2jsg) Business Logic 13

[5.6.3](#_1mrcu09) Use Cases 13

[5.7](#_46r0co2) Request available items 14

[5.7.1](#_2lwamvv) User Interface 14

[5.7.2](#_111kx3o) Business Logic 14

[5.7.3](#_3l18frh) Use Cases 14

[5.8](#_4k668n3) Category 15

[5.8.1](#_2zbgiuw) User Interface 15

[5.8.2](#_1egqt2p) Business Logic 15

[5.8.3](#_3ygebqi) Use cases 15

[5.9](#_sqyw64) Remove violated items, ban users 15

[5.9.1](#_3cqmetx) User Interface 15

[5.9.2](#_1rvwp1q) Business Logic 15

[5.9.2.1](#_4bvk7pj) Manage a list of violated Items 15

[5.9.3](#_2r0uhxc) Use cases 16

[5.10](#_25b2l0r) Create/Modify/Delete category 17

[5.10.1](#_kgcv8k) User Interface 17

[5.10.2](#_34g0dwd) Business Logic 17

[5.11](#_1jlao46) Give comments/feedbacks 18

[5.11.1](#_43ky6rz) User Interface 18

[5.11.2](#_2iq8gzs) Business Logic 18

[5.11.3](#_xvir7l) Use cases 18

[5.12](#_3hv69ve) Create ranking system for Giver 18

[5.12.1](#_1x0gk37) User Interface 18

[5.12.2](#_4h042r0) Business Logic 19

[5.13](#_2w5ecyt) Rank the item/post 19

[5.13.1](#_1baon6m) User Interface 19

[5.13.2](#_3vac5uf) Business Logic 19

[5.14](#_2afmg28) Report the item/post 19

[5.14.1](#_pkwqa1) User Interface 19

[5.14.2](#_39kk8xu) Business Logic 19

[5.15](#_48pi1tg) Create/Update/Delete Post 19

[5.15.1](#_2nusc19) User Interface 20

[5.15.2](#_1302m92) Business Logic 20

[5.15.2.1](#_3mzq4wv) Create a post 20

[5.15.2.2](#_2250f4o) Edit a post 20

[5.15.2.3](#_haapch) Delete a post 20

[6](#_1gf8i83) Non-functional Requirements 22

[6.1](#_40ew0vw) Operating System Support 22

[7](#_2fk6b3p) ANALYZE AND DESIGN 22

[7.1](#_upglbi) Activity Diagram 22

[7.2](#_3ep43zb) Class Diagram 22

[7.3](#_1tuee74) Database 22

[8](#_4du1wux) APPENDIX A: [TBD] 22

**CHANGE HISTORY**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Description** |
| 0.1 | 03-10-2016 | Robert Hendriksen | Initial version |
|  |  |  |  |

# INTRODUCTION

## Purpose

The purpose of this document is to describe the requirements and specification for the Scientific Workflows application on iPad

## Scope

This document is intended for anyone in direct relation to the Give and Receive application.

# REFERENCED DOCUMENTS

## Controlling Documents

This document is dependent on the following documents:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id | Document | Author | Version | Date |
|  | FEI Workflows – Mockups.pdf | Author Name |  | Aug 21 2016 |
|  | Database\_Design V2.1.jpeg | Author Name |  | Aug 21 2016 |
|  | XML file structure 1.4.docx | Author Name |  | Oct 06 2016 |
|  | UI propsosal\_0033645 UI design iPhone.pdf | Author Name |  | Apr 13 2018 |

## Controlled Documents

|  |  |  |
| --- | --- | --- |
| UI Design | Ky Pham | Sep 26 2016 |
| Reworked UI Design | Ky Pham | Feb 2 2017 |
|  |  |  |

## Background Information

The following documents are relevant to the context of the document but do not affect the contents in a direct way:

[SQS]: Sioux Quality System

Version 3.5  
August 7, 2012

# ACTORS

The following table gives an overview of the different human and non-human actors that interact with the system. The name between parentheses is the formal name further used in this document.

|  |  |
| --- | --- |
| **Actor** | **Description** |
| Operator | Who execute a workflow step by step with different instrument. |
| S3 | A server which holds all available workflows and its relevant files, results, images, and videos. |

# OVERVIEW

Description for overview of application

# FEATURES

## Device Orientation

**Give and Receive App** supports both Portrait and Landscape mode for supported devices:

iPad and iPhone with the ***minimum*** ***iOS 11.0***, multi-touch display with LED backlight and IPS technology.

And android too… need to have Mr Thanh’s slide to write this information.

* For ***iPad***: the minimum model supported is **iPad Air 2**.
* For ***iPhone***: the minimum model supported is **iPhone 5s** ***(Performance issues are acceptable)*** and the ***Standard Testing Device*** is **iPhone 6s**. The application also supports ***iPhone X***

## Startup

### User Interface

For Iphone

[TBD]

For Android:

[TBD]

### Business Logic

Splash screen is the first screen to show whenever user opens the application.

Splash screen remains during loading progress and close after the loading finished.

Once the app is finished loading, it will show the login screen.

## Index

Index is shown after the application is loaded successfully.

Guests and users can see all the posts about items to be given.

Users and guests can use filters sorter to filtrate and short the posts they want to see.

### User Interface

*[TBD]*

### Business Logic

* Main screens is loaded after the application is loaded successfully. In main screen, users can see all the posts of Giver.
* Besides that, users can use filter to classify the posts to see what they concern.
* The value of filters are pre-defined by dev team.
* If user apply filters, the result will be shown based on the filter, if not, the result will be all of the posts available on our system.

### User Cases

|  |  |  |
| --- | --- | --- |
| Use Case | UC1: Give and Receive App allows users to see all the items on the main screen. | |
| Actors | **Guests** | **User** |
| Pre-conditions | N/A | |
| Post-conditions | N/A | |
| Main success scenario | 1. The user launches Give and Receive app 2. Give and Receive app displays Launching screen 3. Give and Receive app displays Main screen 4. The user see all the posts in a list, scroll down and up to see the posts | |
| Alternative scenario 1 | 1. Launch Give and Receive app from background    1. Give and Receive app displays the last screen | |
| Business Rules |  | |
| Other requirements | Time of launching app is as short as possible (within 3s loading for items filled the screen) | |
| Open issues |  | |

### 

### 

|  |  |  |
| --- | --- | --- |
| Use Case | UC2: User use filter and sorter to classify the posts they concern. | |
| Actors | **Guests and Users** |  |
| Pre-conditions | In the Main Screen | |
| Post-conditions | Result shown on the screen is fixed with the filters and sorters. | |
| Main success scenario | 1. User applies filter to classify the posts they want 2. The user see all the posts classified by filters. | |
| Alternative scenario 1 | 1. Users do not apply any filter or sorter.    1. The Main Screen displays all of posts again. | |
| Business Rules |  | |
| Other requirements | Time of launching app is as short as possible (within 3s loading for items filled the screen) | |
| Open issues |  | |

## 

## Login

### User Interface

For Iphone

*[TBD]*

For Android:

*[TBD]*

### Business Logic

#### Login using Facebook account

The user login to “Give and Take” app using their Facebook account

* If the device is not yet log in to Facebook, the user must login to Facebook and they must confirm the privacy problem of Facebook. If the user agree, redirect back to the main screen.
* If the device is logged in to Facebook, skip the “log in to Facebook” step, redirect to the main screen.
* If the user account already synchronized with “Give and Take” app, skip “Confirm privacy” step.

#### Login using Google+ account

The user login to “Give and Take” app using their Google+ account

* They must confirm privacy problem of Google, which will display by the user’s phone browser. If the user agree, redirect back to the main screen.
* If the user want to use another Google+ account, they must choose “Login to another account”, then fill all the mandatory field needed, and use that account to login. After that, redirect to confirm privacy step.
* If the user account already synchronized with “Give and Take” app, skip “Confirm privacy” step.

#### Remember credentials

It is possible to remember the credentials. In this case the credentials information (Facebook/Google account that linked with app) will be used for next login purpose. User will not need to “Sign in with Facebook” or “Sign in with Google+” again, the App will be logged in automatically.

### Use Cases

|  |  |  |
| --- | --- | --- |
| Use Case | UC1: Login using Facebook account | |
| Actors | **Guest** | Operator **User** |
| Pre-conditions | N/A | |
| Post-conditions | Application successfully synchronize with the user Facebook account | |
| Main success scenario | 1. The user’s phone have logged in to Facebook. 2. The user confirm privacy. 3. The application receive access token of the user’s Facebook account from Facebook services. 4. The application get all the information of the user. 5. Redirect to the main screen. | |
| Alternative scenario 1 | Step 1: The user’s phone not yet logged in to Facebook.   * + 1. The user must login to Facebook. Then continue from main step 2. | |
| Alternative scenario 2 | Step 2: The user don’t confirm privacy.  2.2.1) Redirect to the login screen and start from main  step 1. | |
| Alternative scenario 3 | Step 2: The user’s Facebook account has already synchronize with “Give and Take” application  2.3.1) The application update all the user information through their Facebook account and continue from main step 4. | |
| Business Rules |  | |
| Other requirements | Time of launching app is as short as possible (3 - 5s) | |
| Open issues |  | |

|  |  |  |
| --- | --- | --- |
| Use Case | UC2: Login using Google+ account | |
| Actors | **Guest** | Operator **User** |
| Pre-conditions | N/A | |
| Post-conditions | Application successfully synchronize with the user Google+ account | |
| Main success scenario | 1. The user use their existing Google+ account. 2. The user confirm privacy. 3. The application receive access token of the user’s Google+ account from Google services. 4. The application get all the information of the user. 5. Redirect to the main screen. | |
| Alternative scenario 1 | Step 1: The user want to use different Google+ account.   * + 1. The user must login to Google+ using their different account. The continue to main step 2. | |
| Alternative scenario 2 | Step 2: The user don’t confirm privacy.  2.2.1) Redirect to the login screen, and start from main step 1. | |
| Alternative scenario 3 | Step 2: The user’s Google+ account has already synchronize with “Give and Take” application  2.3.1) The application update all the user information through their Google+ account and continue from main step 4. | |
| Business Rules |  | |
| Other requirements | Time of launching app is as short as possible (3 - 5s) | |
| Open issues |  | |

## Manage items in giver’s list

Users can create/modify/delete their items.

There are 3 list of items, each list is a collection of items that have the same item’s status:

- Giving: items that are available on platform, all people can see it and request to receive it  
- Available: items that giver uploaded but not ready to give away  
- Delivered: items that giver has given to receiver

### User Interface

[TBD]

### Business Logic

#### Create new item

User can create a new item by filling all required information :

- take a picture of the item or upload the picture in their gallery

- name of the item that they want to give away

- the item's description

After that, there are some options for the user :

- save the item to their private list and modify/edit/post it later. The item's status is changed to “Available”

- post the item to the platform so that all the other users can be aware about it. The item's status is changed to “Giving”

- cancel

#### Modify items

User can’t modify items that have “Delivered” status. For the items having “Giving” status, after giver modify it, a notification will be sent to all the receiver that are requesting this item.

#### Delete items

User can only delete items that have no request. For the items that have status “Delivered” or the items that have requests, the App will show a warning message: “You can not delete this item”. If user want to delete the selected item, he must accept/decline all the requests.

### Use Cases

|  |  |  |
| --- | --- | --- |
| Use Case | UC1: Give and Receive App allows Givers to Create a new item | |
| Actors | **Primar User** | Operat |
| Pre-conditions | The user have logged in successfully | |
| Post-conditions | The user create a new item successfully | |
| Main success scenario | 1. User requests to create a new item 2. The Give and Receive app displays Create Item screen 3. User fills in all the required informations of the item 4. User requests to save the item to the private list. 5. The Give and Receive app creates an item with all the information filled by user and status field is “Available” 6. The Give and Receive app updates item to server | |
| Alternative scenario 1 | Step 4. User requests to post the item to public.  4.1.1. The Give and Receive app creates an item with all the information filled by user and status field is “Giving”  4.1.2. The Give and Receive app updates the item to server  4.1.3. User fills in required informations of the post  4.1.4. The Give and Receive app creates a post with the informations filled by user  4.1.5. The Give and Receive app updates the item and the post to server | |
| Alternative scenario 2 | Step 5. The Give and Receive app notifies required fields are missing | |

|  |  |  |
| --- | --- | --- |
| Use Case | UC2: Give and Receive App allows Givers to Modify items | |
| Actors | **Primar User** | Operat |
| Pre-conditions | The user have logged in successfully | |
| Post-conditions | The user modify an item successfully | |
| Main success scenario | 1. User chooses an item to modify 2. User requests to modify 3. The Give and Receive app displays Modify item screen 4. User modifies fields 5. User submits 6. The Give and Receive app updates item to server | |
| Alternative scenario 1 | Step 3. The Give and Receive app display error message :“This item can’t be modified” | |
| Alternative scenario 2 | Step 6. The Give and Receive app notifies required fields are missing. | |

|  |  |  |
| --- | --- | --- |
| Use Case | UC3: Give and Receive App allows Givers to Delete items | |
| Actors | **Primar User** | Operat |
| Pre-conditions | User has logged in successfully | |
| Post-conditions | User deletes an item/ group of items successfully | |
| Main success scenario | 1. User chooses items to delete 2. User requests to delete 3. The Give and Receive app deletes item | |
| Alternative scenario 1 | Step 3. The Give and Receive app display error message :“This item can’t be deleted” | |

## Claim items to Taker

Giver claim items to Taker by managing request from Taker for items on his (her) post.

### User Interface

[TBD]

### Business Logic

* Each request just for an item. Taker can also choose many items to request as the same time. After that, system will spread automatically into n different requests for n different items
* After Taker request to receive a items, the requests will be sent to Giver
* Giver will receive n notification for n items which Taker requested.
* Giver will check information of per request and then they will decide who will take those items and quantity.

### Use Cases

|  |  |
| --- | --- |
| Use Case | UC1: Giver claim item to Taker. |
| Actors | **User** |
| Pre-conditions | The user have logged in successfully |
| Post-conditions | The confirmation is sent to Taker. |
| Main success scenario | 1. Giver launch Giving Items List, choose a particular item in a post, the app will show the request list for this item. Then, user want to see a request detail. 2. "Give and Take" app displays detailed request for this item. 3. Giver adjust quantity and accept. 4. "Give and Take" displays list of requests again to user 5. "Give and Take" app changes item’s state to be given or updates item’s quantity and the post. 6. "Give and Take" app sends notification to Taker requesting the item on the pos to let them know the changes about items and quantity. (Only Taker who requests for this item receive notification). 7. A message will be sent to chosen Taker to notify the confirmation of Giver and telephone number of Giver to contact and receive the items. |
| Alternative scenario 1 | 1. Giver receive a notification about a request.    1. continue main step 2. |
| Alternative scenario 2 | 1. Giver decline.    1. The request is removed from Receiver’s screen    2. A notification about rejection is sent to Taker who own the request. |

## Request available items

Taker making a request to receive items from Giver

### User Interface

[TBD]

### Business Logic

* Taker see the post and items they need on it.
* Taker has to fill all mandatory fields such as: telephone number, item name, quality, reason... in request form to send to Giver.
* As for telephone number, Taker can reuse their default telephone number on the profile, or they can fill the new number.
* As for item name, Taker only choose the name of items in the post (Multichoice with predefined value)
* As for item quality, Taker can choose the number (Maximum is the max value of this item in the post).
* If one of mandatory fields is empty, an error message will be shown to Taker.
* Each request is just only for an item. Taker can create request for multiple items (After that, system will separate this request into n requests for n items)
* Taker can’t change the sent requests.

### Use Cases

|  |  |  |
| --- | --- | --- |
| Use Case | UC1: Create Request And Send to Giver | |
| Actors | **User** |  |
| Pre-conditions | The user have logged in successfully | |
| Post-conditions | The request created successfully | |
| Main success scenario | 1. Taker launch ‘Create Request’ screen for a post 2. Taker fill the form with mandatory fields. 3. Taker submit the form. 4. Taker will be navigated into ‘Request Management’ screen | |
| Alternative scenario 1 | 1. Taker does not fill all mandatory fields    1. An error message will be shown to ‘Taker’, continue with main step 2. | |
| Alternative scenario 2 | 1. Taker “Cancel” creating request.    1. App navigates back to the Post detail screen | |
| Other requirements |  | |
| Open issues |  | |

## Category

Category screen helps user to choose the category of items they want to see.

### User Interface

[TBD]

### Business Logic

* Category values is defined by admin.
* User choose a particular category, the result fixed with this category will be shown.

### Use cases

|  |  |  |
| --- | --- | --- |
| Use Case | UC1: Choose the category | |
| Actors | **User** |  |
| Pre-conditions | N/A | |
| Post-conditions | The result fixed with this category will be shown. | |
| Main success scenario | 1. Taker launch Category screen. 2. Taker choose a particular category on the screen. 3. Main screen is shown with result filtrated by chosen category. | |
| Other requirements |  | |
| Open issues |  | |

## Remove violated items, ban users

### User Interface

[TBD]

### Business Logic

#### Manage a list of violated Items

In CMS, violated post are displayed as a list. Each row includes these information:

* Index
* Post’s link
* Number of report
* Detail of reports (a link)
* Name of violated user is accompanied by number of warnings
* Block User (block violated user)
* Warn User
* Delete Post
* Solved or unsolved state

Admin can see detail of the post, then checking the violated item.

Admin can send warning notification to user or ban the user if number of warning more than 1.

Admin can delete violated post, all items that was attached to this post will be deleted as well.

### Use cases

|  |  |  |
| --- | --- | --- |
| Use Case | UC1: Send warning notification to user | |
| Actors | **Admin** | Operat |
| Pre-conditions | Login to CMS successfully | |
| Post-conditions | N/A | |
| Main success scenario | 1. Admin Warn user 2. System displays confirm box 3. Admin confirm that he want to warn user 4. System sends a warning notification to user’s phone and updates number of warnings of that user 5. Unsolved state changes to Solved | |
| Alternative scenario 1 | Step 3: Admin don’t confirm that he want to warn the user.  3.1.1 Close confirm box. Redirect to main manage screen. | |

|  |  |  |
| --- | --- | --- |
| Use Case | UC2: Block user | |
| Actors | **Admin** |  |
| Pre-conditions | Login to CMS successfully | |
| Post-conditions | N/A | |
| Main success scenario | 1. Admin Block User 2. System displays confirm box 3. Admin confirm that he want to block a user 4. System updates user’s state to blocked 5. Unsolved state changes to Solved | |
| Alternative scenario 1 | Step 3: Admin don’t confirm that he want to block the user.  3.1.1 Close confirm box. Redirect to main manage screen. | |

|  |  |  |
| --- | --- | --- |
| Use Case | UC3: Delete post | |
| Actors | **Admin** |  |
| Pre-conditions | Login to CMS successfully | |
| Post-conditions | N/A | |
| Main success scenario | 1. Admin Delete a post 2. System displays confirm box 3. Admin confirm that he want to delete that post 4. System removes the post and all items was attached to this post 5. System sends notification to user (information of post and all items deleted) 6. Unsolved state changes to Solved | |
| Alternative scenario 1 | Step 3. Admin don’t confirm that he want to delete that post  3.1.1 Close confirm box. Redirect to the main manage screen. | |

## Create/Modify/Delete category

### User Interface

[TBD]

### Business Logic

In CMS, categories are displayed as a list. Each row includes these information:

* Index
* Category’s name
* Description
* Edit
* Delete

And Create is above the list.

Admin can modify, delete per category. After deleting, category will be removed from the list, however, it still exists in database.

Admin can create a new category.

## Give comments/feedbacks

### User Interface

[TBD]

### Business Logic

User can give comments, feedback by filling what they think about the app, what is the advantages and disadvantages of the app, what do they think to improve the app better,... to the shout box and submit it.

### Use cases

|  |  |  |
| --- | --- | --- |
| Use Case | UC1: Create feedback/ comment | |
| Actors | **User** | Operat |
| Pre-conditions | N/A | |
| Post-conditions | The feedback has sent successfully | |
| Main success scenario | 1. User request to give a comment/feedback 2. User fill in comment/feedback 3. User submit to send | |
| Alternative scenario 1 | Step 3. User cancel | |
| Alternative scenario 2 | Step 3. The Give and Receive app notifies message field are missing. | |

## Create ranking system for Giver

### User Interface

[TBD]

### Business Logic

The rank of Giver is defined base on the number of items that the user has been given, the rate of the the post, the number of posts that user has been reported,.. There is a list that show top 50 generous givers.

There is a system to calculate the number of things that a user “Delivered” and then depend on that information to rank that person.

Each rank will have a label come with a special color for their name.

## Rank the item/post

### User Interface

[TBD]

### Business Logic

User can rank the items that are useful for other people can easier recognize it. User can rank from 1 star to 5 stars.

Rank of the item will display base on the number of bright stars.

Rank of the item will be calculate by the average score of those who have evaluated.

## Report the item/post

### User Interface

[TBD]

### Business Logic

User can report the items/posts that are violated by filling the reason to the form and submit. The report will be sent to the admin for handling.

## Create/Update/Delete Post

Users (Receivers) manage their posts.

### User Interface

[TBD]

### Business Logic

#### Create a post from “Available” list

User can select items with the quantity from their Available list and then “Create a new post”.

User can write description for the post. User has to fill mandatory fields of the post like: title, description, categories, location. As for location, location on his/her profile is default value.

#### Edit a post

User can edit their post, we have some situations:

* Edit post’ s description -> send a notification to all Taker requested
* Edit item that was not requested in post -> don’t send a notification
* Edit item that was requested in post -> send a notification to Taker requested that item

#### Delete a post

User can delete their post, we have some situations:

* Delete post that does not have any request -> don’t send a notification
* Delete post that was requested in post -> Handle all the request (Accept or Decline) and send a notification to Taker requested that item.
  + 1. **Use Cases**

|  |  |  |
| --- | --- | --- |
| Use Case | UC1: User (Receiver) Create post | |
| Actors | **User** | Operat |
| Pre-conditions | The user have logged in successfully | |
| Post-conditions | The post has been created successfully | |
| Main success scenario | 1. User pick items and define quantity (Default is max number of this item). 2. User “Create new post”. 3. User fill the mandatory fields: title, description, location, categories. 4. User “Submit”. 5. The post and items will be public to other users (Takers). | |
| Alternative scenario 1 | 1. User fill all mandatory fields, then remove chosen items for the post    1. removed items disappear from the view. | |
| Alternative scenario 2 | 1. User does not fill all mandatory fields.    1. User “Submit”.    2. An error message will be shown to user. | |

|  |  |  |
| --- | --- | --- |
| Use Case | UC2: User (Receiver) modify post | |
| Actors | **User** | Operat |
| Pre-conditions | The user have logged in successfully | |
| Post-conditions | The post has been modified successfully and notification is sent to registered Takers. | |
| Main success scenario | 1. User launch “Giving” item list. 2. User see a detail of particular post. 3. User choose “Edit”. 4. User edit information of the post, including: Items and quantity. User can’t let one of mandatory fields empty. 5. User “Save”. 6. The post is updated, and app navigate back to”Giving” Item list. 7. Notification will push to all Giver who requested for updated items in the post. | |
| Alternative scenario 1 | 1. User let one of mandatory fields empty    1. User “Save”    2. An error message will be shown to user | |
| Alternative scenario 2 | 1. User delete an item which has requests.    1. an error message will be shown to Receiver    2. User must handle all the requests for this item (Accept or Decline).    3. If user accept for one or more requests, the item’s status will be changed to : ”Delivered”    4. If user decline all requests, user has 2 options: Move this item back to “Available” list or delete this item. Then, user continue main step 5. | |

|  |  |  |
| --- | --- | --- |
| Use Case | UC3: User (Receiver) delete post | |
| Actors | **User** | Operat |
| Pre-conditions | The user have logged in successfully | |
| Post-conditions | The post has been deleted successfully | |
| Main success scenario | 1. User launch “Giving” item list. 2. User see a detail of particular post. 3. User “Delete” the post. 4. The post is deleted, and app navigate back to”Giving” Item list. 5. Notification will push to all Giver who have request to the post. | |
| Alternative scenario 1 | 1. User delete a post has one more items which has requests.    1. An error message will be shown to user    2. User must handle all the requests for this item (Accept or Decline).    3. If user accept for one or more requests, the item’s status will be changed to : ”Delivered”    4. If user decline all requests, this item will be moved back to “Available” list. | |

# Non-functional Requirements

## Operating System Support

Scientific Workflows App supports iOS 11 and above.

# ANALYZE AND DESIGN

## Activity Diagram

## Class Diagram

## Database

…

# APPENDIX A: [TBD]