|  |  |
| --- | --- |
| 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 INTRODUCTION 6](#_Toc519747579)

[1.1 Purpose 6](#_Toc519747580)

[1.2 Scope 6](#_Toc519747581)

[2 REFERENCED DOCUMENTS 6](#_Toc519747582)

[2.1 Controlling Documents 6](#_Toc519747583)

[2.2 Controlled Documents 6](#_Toc519747584)

[2.3 Background Information 6](#_Toc519747585)

[3 ACTORS 7](#_Toc519747586)

[4 OVERVIEW 7](#_Toc519747587)

[5 FEATURES 8](#_Toc519747588)

[5.1 Device Orientation 8](#_Toc519747589)

[5.2 Start up 8](#_Toc519747590)

[5.2.1 User Interface 8](#_Toc519747591)

[5.2.2 Business Logic 8](#_Toc519747592)

[5.3 Login 9](#_Toc519747593)

[5.3.1 User Interface 9](#_Toc519747594)

[5.3.2 Business Logic 9](#_Toc519747595)

[5.3.2.1 Login using Facebook account 9](#_Toc519747596)

[5.3.2.2 Login using Google+ account 9](#_Toc519747597)

[5.3.2.3 Remember credentials 9](#_Toc519747598)

[5.3.3 Use Cases 10](#_Toc519747599)

[5.4 Item Management 12](#_Toc519747600)

[5.4.1 User Interface 12](#_Toc519747601)

[5.4.2 Business Logic 12](#_Toc519747602)

[5.4.2.1 Create new item 12](#_Toc519747603)

[5.4.2.2 Modify items 12](#_Toc519747604)

[5.4.2.3 Delete items 12](#_Toc519747605)

[5.4.3 Use Cases 13](#_Toc519747606)

[5.5 Post Management 14](#_Toc519747607)

[5.5.1 User Interface 14](#_Toc519747608)

[5.5.2 Business Logic 14](#_Toc519747609)

[5.5.2.1 View public posts 14](#_Toc519747610)

[5.5.2.2 Giver create a Post 14](#_Toc519747611)

[5.5.2.3 Giver modify a Post 14](#_Toc519747612)

[5.5.2.4 Giver delete a Post 14](#_Toc519747613)

[5.5.2.5 Share a Post 14](#_Toc519747614)

[5.5.2.6 User comment in a Post 15](#_Toc519747615)

[5.5.3 Use Cases 15](#_Toc519747616)

[5.6 Request Management 16](#_Toc519747617)

[5.6.1 User Interface 16](#_Toc519747618)

[5.6.2 Business Logic 16](#_Toc519747619)

[5.6.2.1 Taker create a Request (Request a post) 16](#_Toc519747620)

[5.6.2.2 Taker delete (Cancel) a Request 16](#_Toc519747621)

[5.6.2.3 Giver claim item to Taker (Accept the Request) 16](#_Toc519747622)

[5.6.2.4 Giver marked a request as “Delivered”. 16](#_Toc519747623)

[5.6.2.5 Giver decline requests 16](#_Toc519747624)

[5.6.3 Use Cases 17](#_Toc519747625)

[5.7 Report management 19](#_Toc519747626)

[5.7.1 User Interface 19](#_Toc519747627)

[5.7.2 Business Logic 19](#_Toc519747628)

[5.7.2.1 User report a post 19](#_Toc519747629)

[5.7.2.2 Admin manage a list of reported posts 19](#_Toc519747630)

[5.7.3 Use cases 20](#_Toc519747631)

[5.8 Category Management 21](#_Toc519747632)

[5.8.1 User Interface 21](#_Toc519747633)

[5.8.2 Business Logic 21](#_Toc519747634)

[5.8.2.1 Admin manage category 21](#_Toc519747635)

[5.8.2.2 User choose a category to filtrate the posts 21](#_Toc519747636)

[5.9 Feedbacks Management 22](#_Toc519747637)

[5.9.1 User Interface 22](#_Toc519747638)

[5.9.2 Business Logic 22](#_Toc519747639)

[5.9.2.1 Quest create feedbacks 22](#_Toc519747640)

[5.9.2.2 Admin read feedbacks 22](#_Toc519747641)

[5.9.3 Use cases 22](#_Toc519747642)

[5.10 Ranking system 23](#_Toc519747643)

[5.10.1 User Interface 23](#_Toc519747644)

[5.10.2 Business Logic 23](#_Toc519747645)

[5.10.2.1 Ranking system for Giver 23](#_Toc519747646)

[5.10.2.2 Ranking system for Item/ Post 23](#_Toc519747647)

[6 Non-functional Requirements 24](#_Toc519747648)

[6.1 Operating System Support 24](#_Toc519747649)

[7 ANALYZE AND DESIGN 24](#_Toc519747650)

[7.1 Activity Diagram 24](#_Toc519747651)

[7.1.1 Login 24](#_Toc519747652)

[7.1.2 Item Management 27](#_Toc519747653)

[7.1.3 Post Management 30](#_Toc519747654)

[7.1.4 Request Management 33](#_Toc519747655)

[7.1.5 Report Management 36](#_Toc519747656)

[7.1.6 Category Management 37](#_Toc519747657)

[7.1.7 Feedbacks Management 40](#_Toc519747658)

[7.2 Class Diagram 41](#_Toc519747659)

[7.3 Database 41](#_Toc519747660)

[8 APPENDIX A: [TBD] 41](#_Toc519747661)

**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 multiplatform.

## 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** |
| Admin | Who control CMS administration page |
| User | The user who can be both Giver and Taker in system |
| Guest | Who use the system but are not login. |

# 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***

## Start up

### 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 public posts screen.

## Login

### User Interface

For iPhone

*[TBD]*

For Android:

*[TBD]*

### Business Logic

By default, guests can only view public posts. When they want to create a new post/item or send a request to Givers, they must login first.

#### 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 agrees, 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 agrees, redirect back to the main screen.
* If the user wants to use another Google+ account, they must choose “Login to another account”, then fill all the mandatory fields 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, but the App will be logged in automatically.

### Use Cases

|  |  |  |
| --- | --- | --- |
| Use Case | UC1: Login using Facebook account | |
| Actors | **Guest** |  |
| 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 confirms privacy. 3. The application receive access token of the user’s Facebook account from Facebook services. 4. The application gets 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 updates all the user information through their Facebook account and continues 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** |  |
| Pre-conditions | N/A | |
| Post-conditions | Application successfully synchronize with the user Google+ account | |
| Main success scenario | 1. The user uses their existing Google+ account. 2. The user confirms privacy. 3. The application receive access token of the user’s Google+ account from Google services. 4. The application gets 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, continues 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 |  | |

## Item Management

### User Interface

[TBD]

### Business Logic

Users can create/modify/delete their items.

There are 3 collections of items.

* 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

#### Create new item

Users 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.
* Item's description.
* Quantity.
* Choose category.

After that, the item will be stored into available list.

#### Modify items

Users can modify items. For the items being in posts have requests, after the giver modifies it, a notification will be sent to all the requesters.

#### Delete items

Users can only delete items which don’t appear in “Giving” list.

For the items which appear in “Giving” list, the App will show an error message: “You can’t delete this item”. If users want to delete the item which appear in “Giving” list, they must remove it from the post it is attached.

### Use Cases

|  |  |  |
| --- | --- | --- |
| Use Case | UC1: Givers create a new item | |
| Actors | **Giver** |  |
| 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 information of the item 4. The item will be created with all the information filled by user and stored in “Available” list. | |
| Alternative scenario 1 | Step 4. The Give and Receive app notifies required fields are missing | |

|  |  |  |
| --- | --- | --- |
| Use Case | UC2: Giver modify an item | |
| Actors | **Giver** |  |
| 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 see detail 2. User modify the item 3. User modifies fields 4. User submits 5. The item is updated 6. A notification is sent to all requesters (If the item appear on posts have requests.) | |
| Alternative scenario 1 | Step 6. The Give and Receive app notifies required fields are missing. | |

|  |  |  |
| --- | --- | --- |
| Use Case | UC3: Giver delete an item | |
| Actors | **Giver** |  |
| Pre-conditions | User has logged in successfully | |
| Post-conditions | User deletes an item successfully | |
| Main success scenario | 1. Users choose an item to delete 2. Users commit to “delete” 3. The item is deleted in system. | |
| Alternative scenario 1 | Step 3. The Give and Receive app display error message :“This item can’t be deleted” | |

## Post Management

### User Interface

### Business Logic

#### View public posts

User can views all public posts which Givers posted.

* Public posts screen is loaded after the application is loaded successfully. Users/ guests can see all the posts that Givers has posted.
* Besides that, users/guests can use filters to classify the posts to see what they concern:
  + Location (default is “Nationwide”)
  + Category (default is “All category”)
* After users/guests choose a filter, the posts will be filtrated and sorted based on the time the post created: latest from top to bottom.

#### Giver create a Post

User can create a new post by selecting items in their “Available” list and filling post’s information:

* Title
* Description
* Location (select default’s locations or input a new one)

#### Giver modify a Post

Users modify their posts by changing post’s fields:

* Change Post’s title, description, category, location
* Add/Remove item in the post.
* Change item’s quantity for the post. The quantity must be smaller or equal with the initial quantity of the item.

After changing the post, a notification will be sent to all requesters.

For remove item, all requests to this post will lost this item.

**For change item’s quantity, the new quantity must be smaller or equal the total number between available quantity of this item and giving quantity in this post.(\*)**

#### Giver delete a Post

Users can only delete a post that has no request. If users want to delete a post that have requests, they must decline all of them first.

#### Share a Post

User can share a post on social sites, so people can open the post in our application if they already installed app in their phone. Or requiring install app to see the post if they have not installed app yet.

#### User comment in a Post

User can comment below the post.

### Use Cases

|  |  |  |
| --- | --- | --- |
| Use Case | UC1: Giver Create post | |
| Actors | **Giver** |  |
| Pre-conditions | The user have logged in successfully | |
| Post-conditions | The post has been created successfully | |
| Main success scenario | 1. Givers pick items. 2. Givers “Create new post”. 3. Givers fill the post’s information: title, description, location, categories. 4. Givers “Submit”. 5. The post is created and public. | |
| Alternative scenario 1 | Step 5) The Give and Receive app notifies required fields are missing. | |

|  |  |  |
| --- | --- | --- |
| Use Case | UC2: Giver modify post | |
| Actors | **Giver** |  |
| Pre-conditions | The user have logged in successfully | |
| Post-conditions | The post has been modified successfully and notification is sent to all requesters. | |
| Main success scenario | 1. Givers select a post to modify 2. Givers choose “Edit”. 3. Givers edit information of the post. 4. Givers “Save”. 5. The post is updated. 6. The Give and Receive server send notifications to all requesters | |
| Alternative scenario 1 | Step 3) Giver remove an item which has requests   1. Giver “Save” 2. All the requests to the post delete this item, continue main step 5. | |
| Alternative scenario 2 | Step 3) Giver changes an item’s quantity, and this quantity conforms the ([\*](#_Giver_modify_a)) condition, then continue the main step 4. | |
| Alternative scenario 3 | Step 3) Giver change an item’s quantity, and this quantity doesn’t conform the ([\*](#_Giver_modify_a)) condition.   1. Giver “Save”. 2. The Give and Receive app display an error message: “This quantity is not suitable”. | |

|  |  |  |
| --- | --- | --- |
| Use Case | UC3: Giver delete post | |
| Actors | **Giver** |  |
| Pre-conditions | The user have logged in successfully | |
| Post-conditions | The post has been deleted successfully | |
| Main success scenario | 1. Giver select a post to delete 2. Giver commit to “Delete”. 3. The post is deletes in system. | |
| Alternative scenario 1 | Step 3) The Give and Receive app display error message :“Can’t delete” | |

## Request Management

### User Interface

### Business Logic

#### Taker create a Request (Request a post)

When users see a post containing items that they wanted, they can send a request to the Giver by filling the “Message” field and submit.

#### Taker delete (Cancel) a Request

Taker can cancel their request if they don’t need the items on a post anymore. The request will be deleted in system and can’t be reverted.

If the cancelled request is accepted, all the items and quantity on this request will be returned into the post.

#### Giver claim item to Taker (Accept the Request)

Giver can see all the requests of a post including the request’s message and requester’s profile. Giver can claim that their items are sent to the right people by studying the information and accepting to the most suitable one. Then they will negotiate by phone/ message.

After accepting a request, the post will be updated about items and quantity based on items and quantity on the request.

In a post, Giver can accept many requests as long as the total quantity of an item in all requests is not bigger than the maximum quantity of this item in this post.

#### Giver marked a request as “Delivered”.

After a request is accepted, the Taker will receive items on the request. Then, Giver marks this request as “Delivered” to confirm that the Taker has received items already.

After that, items and their quantity will be moved to “Delivered” list.

#### Giver decline requests

Giver can decline requests if they see that Takers unsuitable to receive their items.

### Use Cases

|  |  |  |
| --- | --- | --- |
| Use Case | UC1: Request a post | |
| Actors | **Taker** |  |
| Pre-conditions | The user have logged in successfully | |
| Post-conditions | The request is created successfully | |
| Main success scenario | 1. Takers request a post they wanted 2. Takers fill in the message they want to send to the Giver 3. Takers submit the form. 4. The Give and Receive app creates and updates the request to server | |
| Alternative scenario 1 | Step 4) The Give and Receive app notifies required fields are missing. | |

|  |  |  |
| --- | --- | --- |
| Use Case | UC2: Delete a request | |
| Actors | **Taker** |  |
| Pre-conditions | The user have logged in successfully | |
| Post-conditions | The request is deleted successfully | |
| Main success scenario | 1. Takers launch request management. 2. Takers delete a particular request. 3. Takers receive a warning, and confirm the warning. 4. The request is deleted in system, and all items and quantity in this request will be returned back to the post. | |
| Alternative scenario 1 | Step 4) Takers receive a warning, and cancel the warning.   1. Return to the main step 1. | |

|  |  |
| --- | --- |
| Use Case | UC3: Giver claim item to Taker. |
| Actors | **Giver** |
| Pre-conditions | The user have logged in successfully |
| Post-conditions | One request is accepted. The reply message is sent to the chosen one successfully |
| 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, users want to see a request detail. 2. "Give and Take" app displays detailed request for this item. 3. Givers adjust quantity and accept. 4. "Give and Take" displays list of requests again to user 5. "Give and Take" app updates item’s quantity and the post. 6. "Give and Take" app sends notification to Taker requesting the item on the post to let them know the changes about items and quantity. 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. Givers receive a notification about a request.    1. Giver continues main step 2. |

|  |  |  |
| --- | --- | --- |
| Use Case | UC4: Giver mark a request as “Delivered” | |
| Actors | **Giver** |  |
| Pre-conditions | The user have logged in successfully | |
| Post-conditions | The request is created successfully | |
| Main success scenario | 1. Givers see an accepted request. 2. Givers mark this request as “Delivered”. 3. All items and their quantity are moved into “Delivered” list. | |
| Alternative scenario 1 | Step 4) The Give and Receive app notifies required fields are missing. | |

|  |  |
| --- | --- |
| Use Case | UC5: Giver decline a request |
| Actors | **Giver** |
| Pre-conditions | The user have logged in successfully |
| Post-conditions | One request is declined, and a notification is sent to the Taker who owns this request. |
| 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, users want to see a request detail. 2. "Give and Take" app displays detailed request for this item. 3. Givers decline the request, and the request is deleted in system. 4. A notification is sent to the Taker who owns this request. |

## Report management

### User Interface

### Business Logic

#### User report a post

Users can report a post that violated the terms of the App by filling the message field and submit the form. The report will be sent to admin to handle.

#### Admin manage a list of reported posts

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

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

After admins see details of the reported posts and the message that Takers submitted. By basing on the level of the violation they will decide to send a warning message to the reported Giver or delete the post.

Admin can delete violated term posts, 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** |  |
| 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. | |

## Category Management

### User Interface

[TBD]

### Business Logic

#### Admin manage category

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

* Index
* Category’s name
* Description
* Action (Delete/ Edit)

And Create is above the list.

Admin can modify, delete per category. After deleting, category will be removed in system.

Admin can create a new category and modify category.

Admin can only delete a category which doesn’t have any items on it.

#### User choose a category to filtrate the posts

* Category values are defined by admin.
* User chooses a particular category.
* The result will be shown with filter for this category in result screen.

## Feedbacks Management

### User Interface

[TBD]

### Business Logic

#### Quest create feedbacks

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.

Feedbacks which were submitted are shown to Admin.

#### Admin read feedbacks

Feedbacks which were submitted are shown to Admin. Admin read feedbacks and have a plan to improve the app in the future.

### Use cases

|  |  |  |
| --- | --- | --- |
| Use Case | UC1: Create feedback | |
| Actors | **Quest** |  |
| 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. The Give and Receive app notifies message field are missing. | |

## Ranking system

### User Interface

[TBD]

### Business Logic

#### Ranking system for Giver

The rank of Giver is defined base on the number of items that the user has been given, the rate of the post, the number of posts that user has been reported… There is a list that shows 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 colour for their name.

#### Ranking system for Item/ Post

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 calculated by the average score of those who have evaluated.

# Non-functional Requirements

## Operating System Support

Give and Take App supports minimum iOS 11, Android 5.0.

# ANALYZE AND DESIGN

## Activity Diagram

### Login

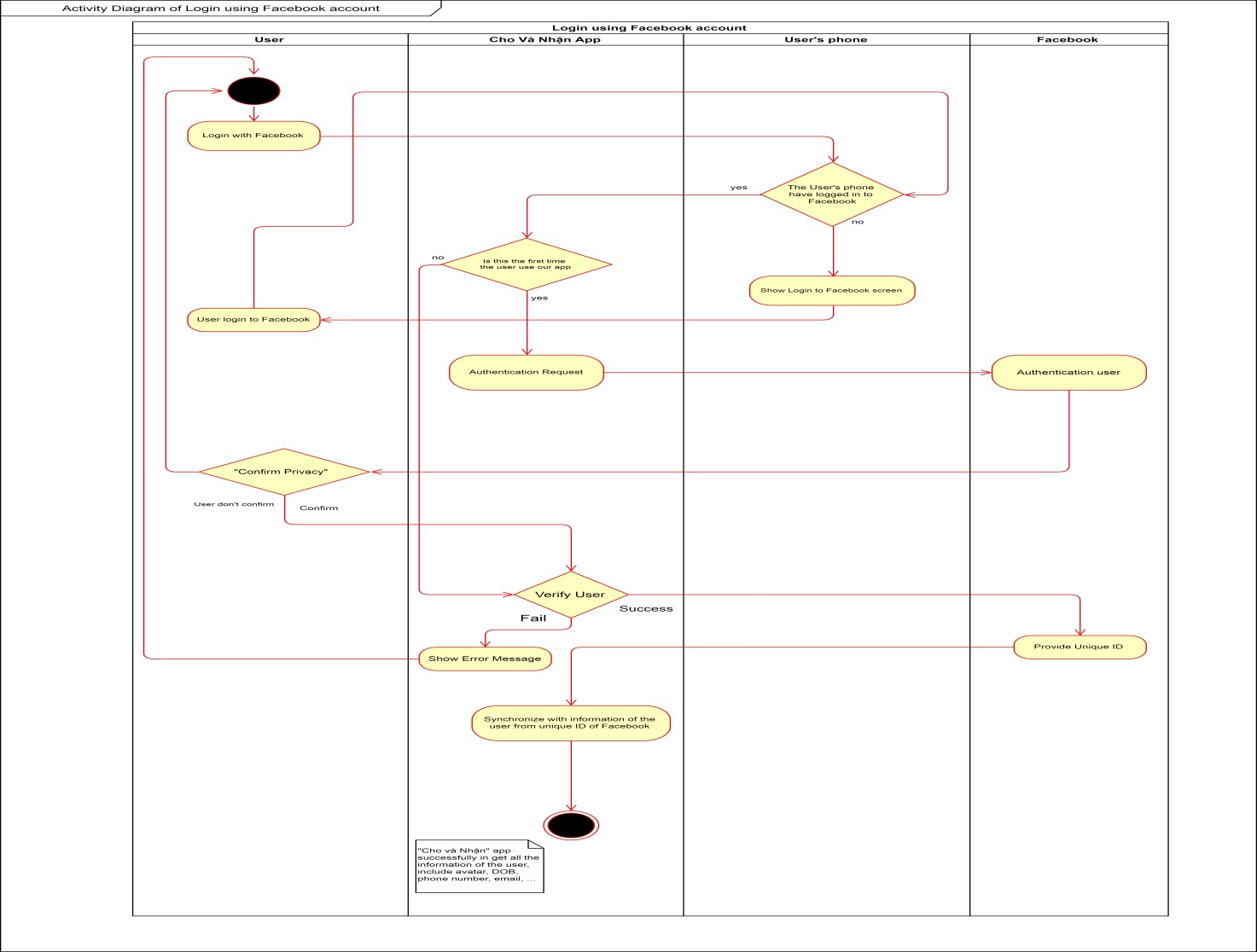


Figure ‑ Activity Diagram\_ Login Using Facebook

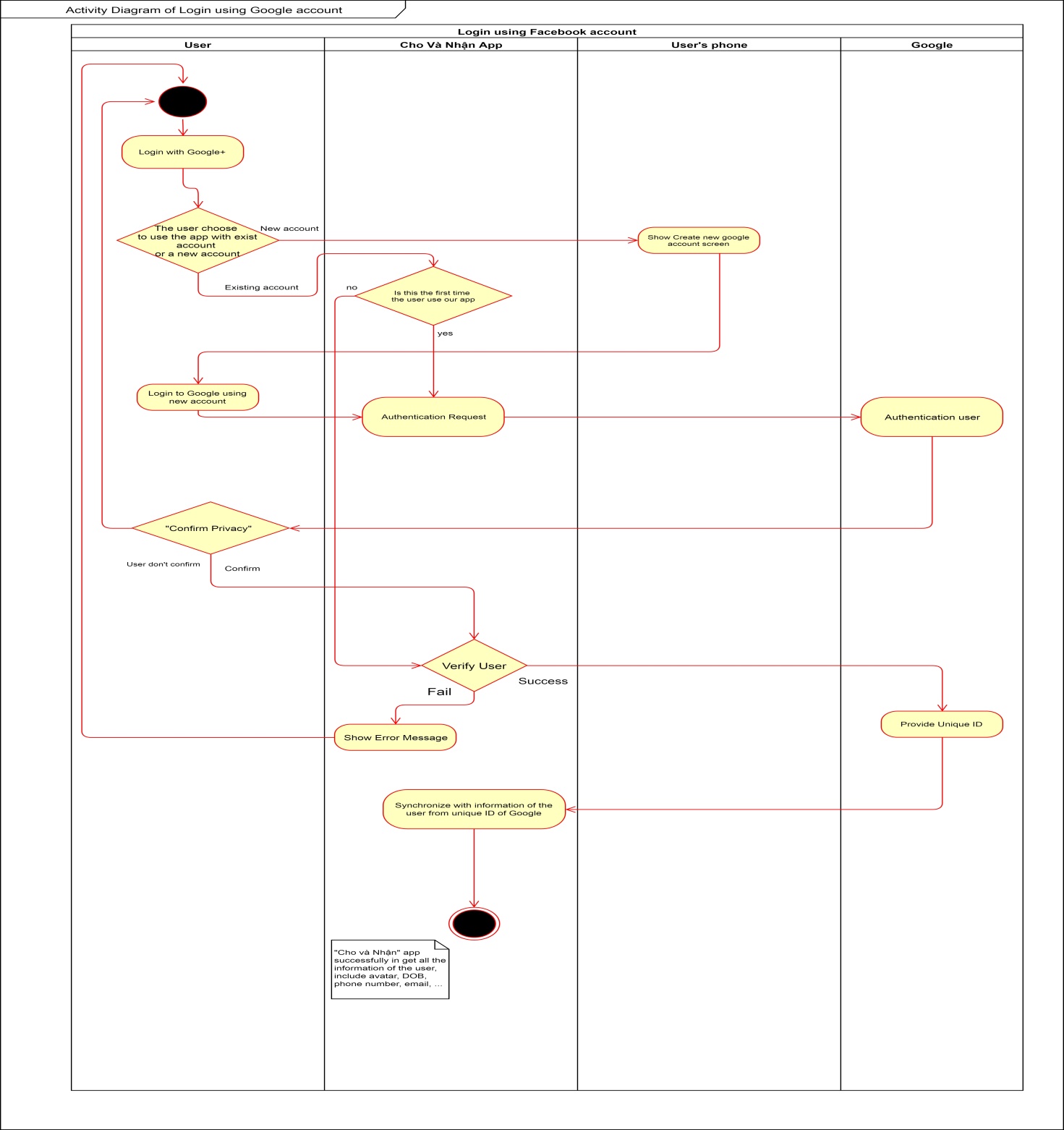


Figure ‑ Activity Diagram\_ Login Using Google

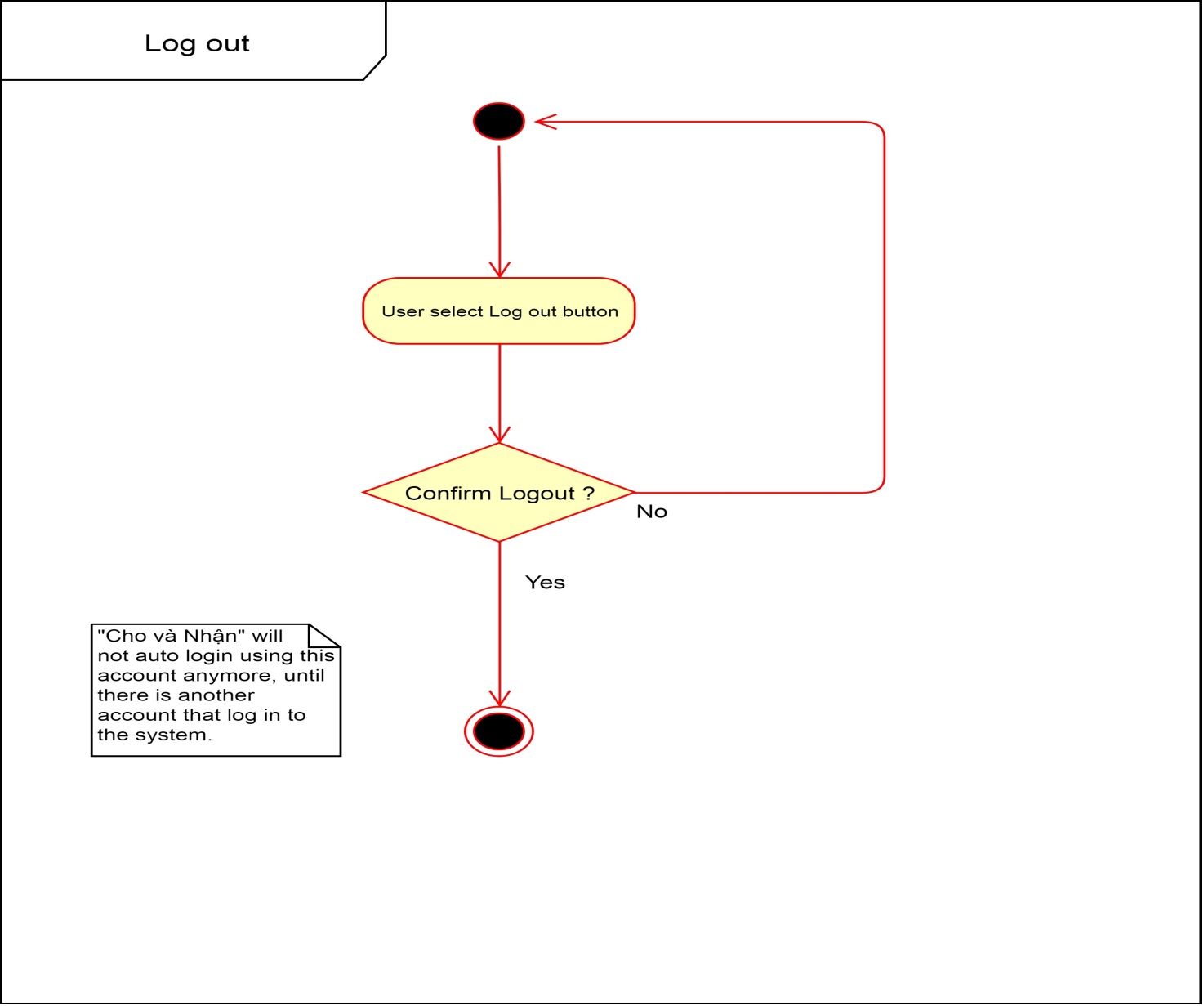


Figure ‑ Activity Diagram\_ Logout

### Item Management

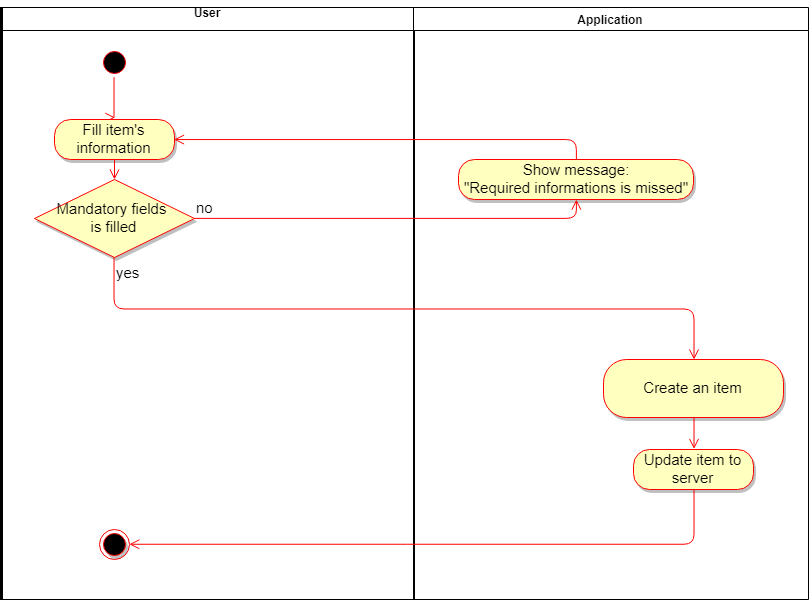


Figure ‑ Activity Diagram\_ Create Item

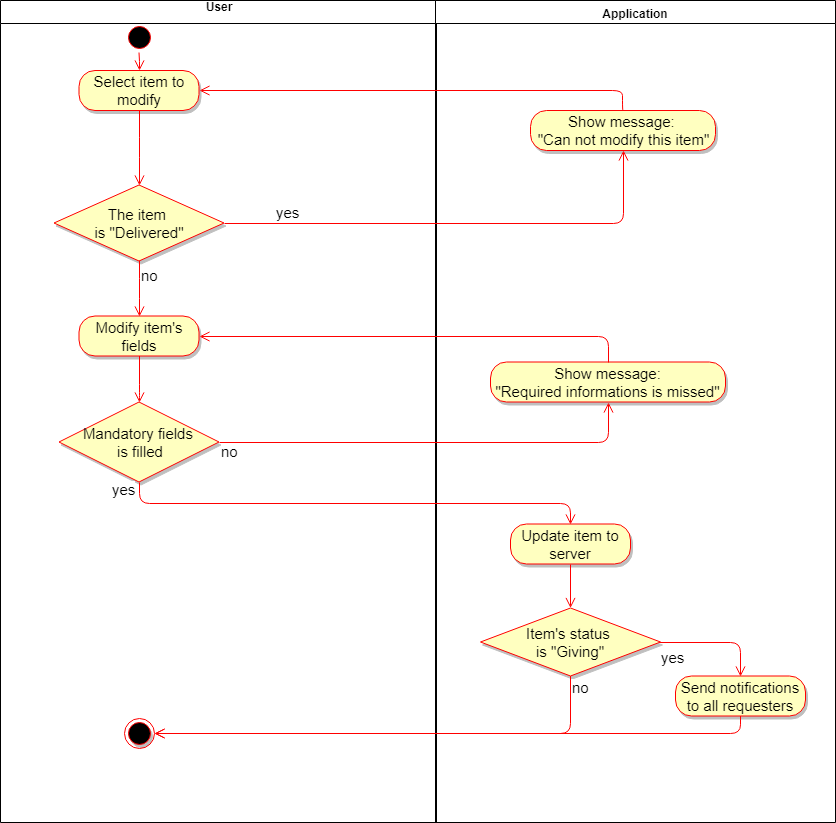


Figure ‑ Activity Diagram\_ Modify Item

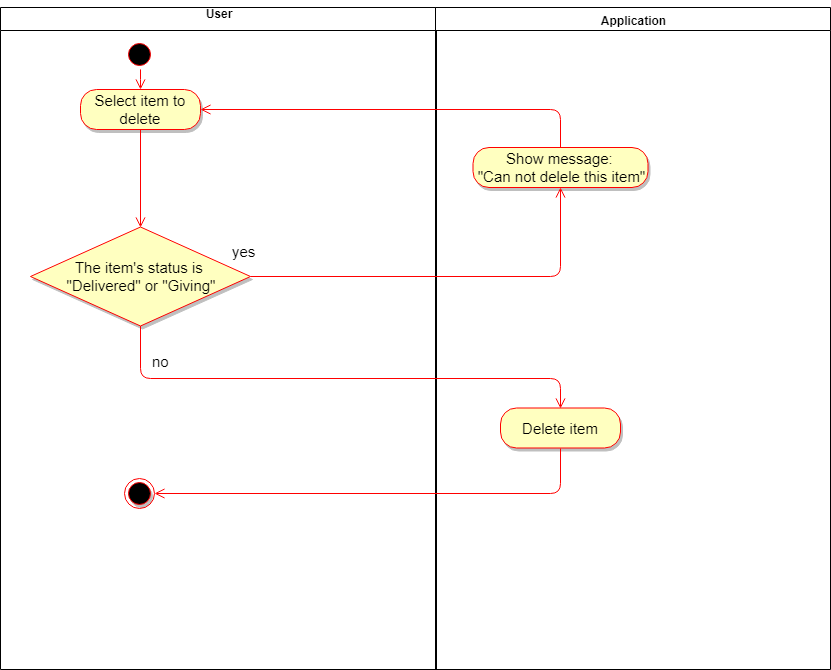


Figure ‑ Activity Diagram\_ Delete Item

### Post Management

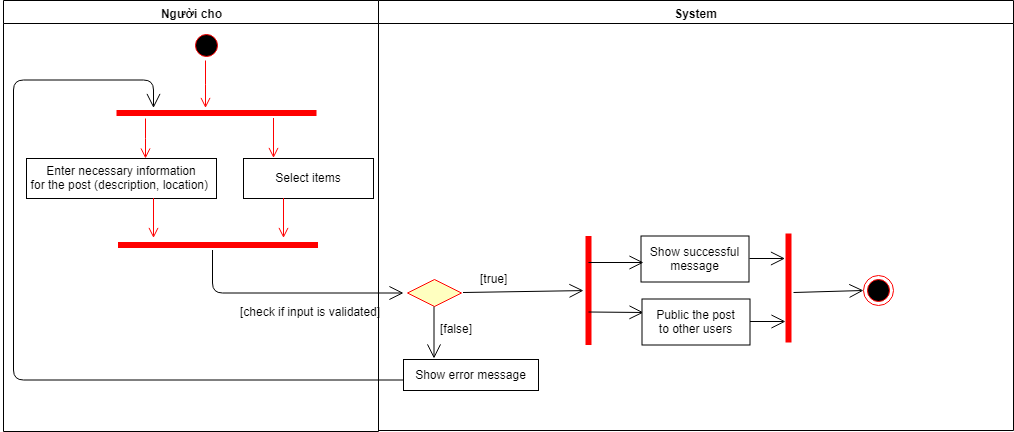


Figure ‑ Activity Diagram\_ Create Post

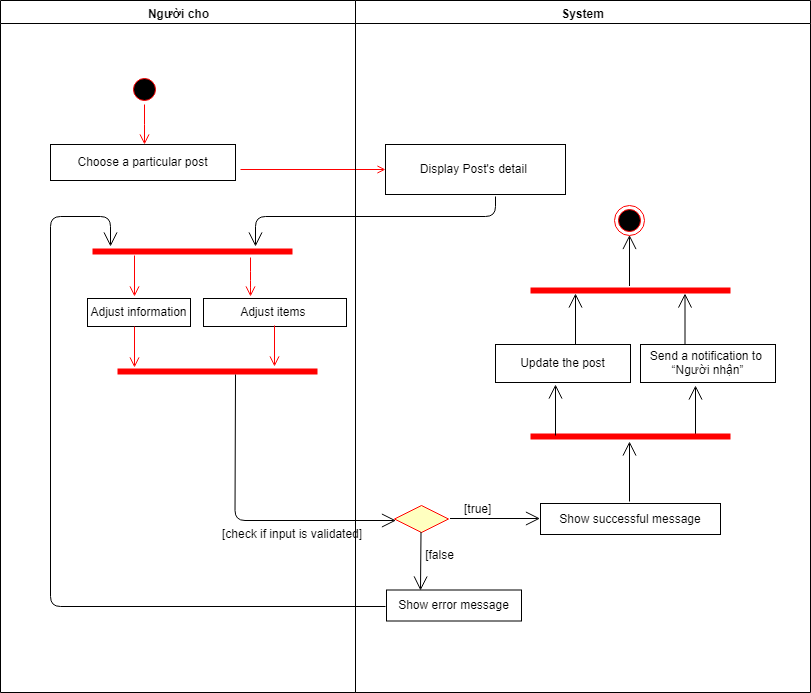


Figure ‑ Activity Diagram\_ Edit Post

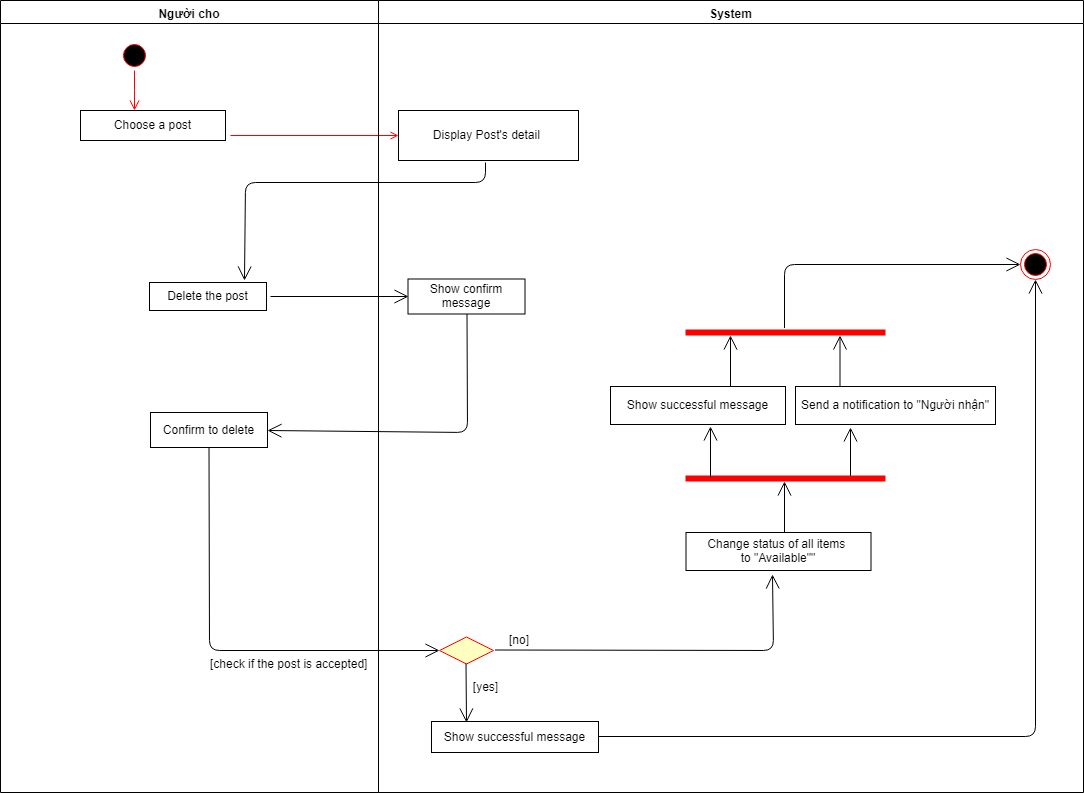


Figure ‑ Activity Diagram\_ Delete Post

### Request Management

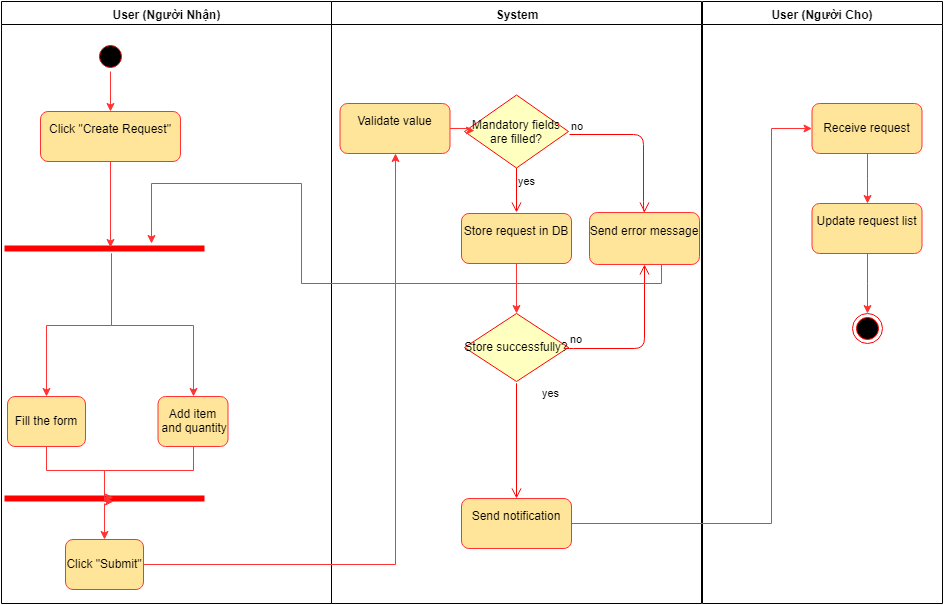


Figure ‑ Activity Diagram\_ Create Request

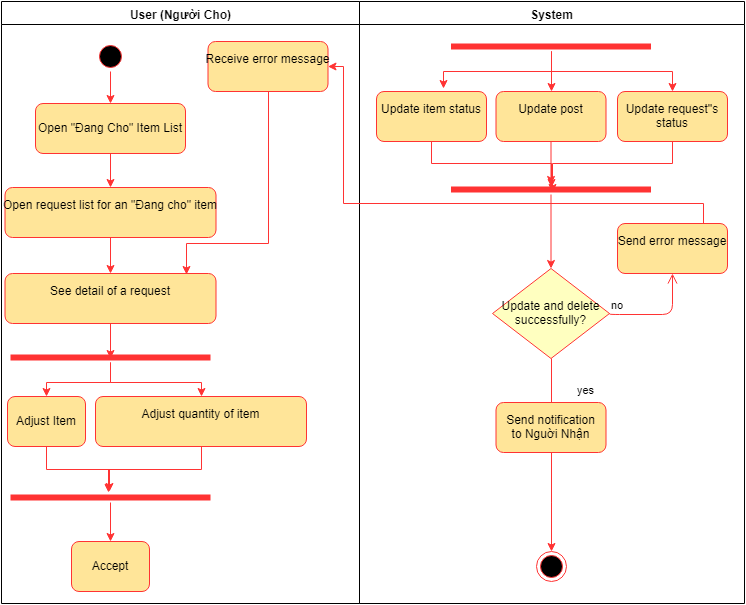


Figure ‑ Activity Diagram\_ Accept Request

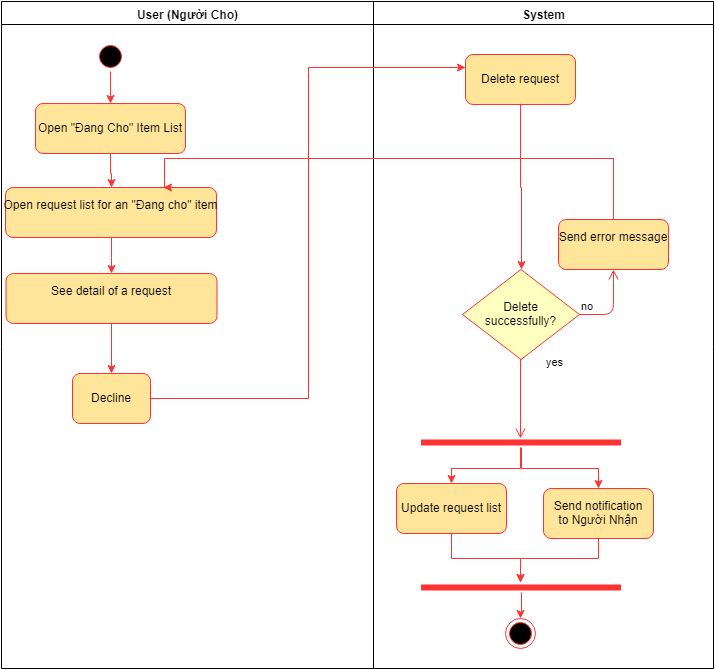


Figure ‑ Activity Diagram\_ Decline Request

### Report Management

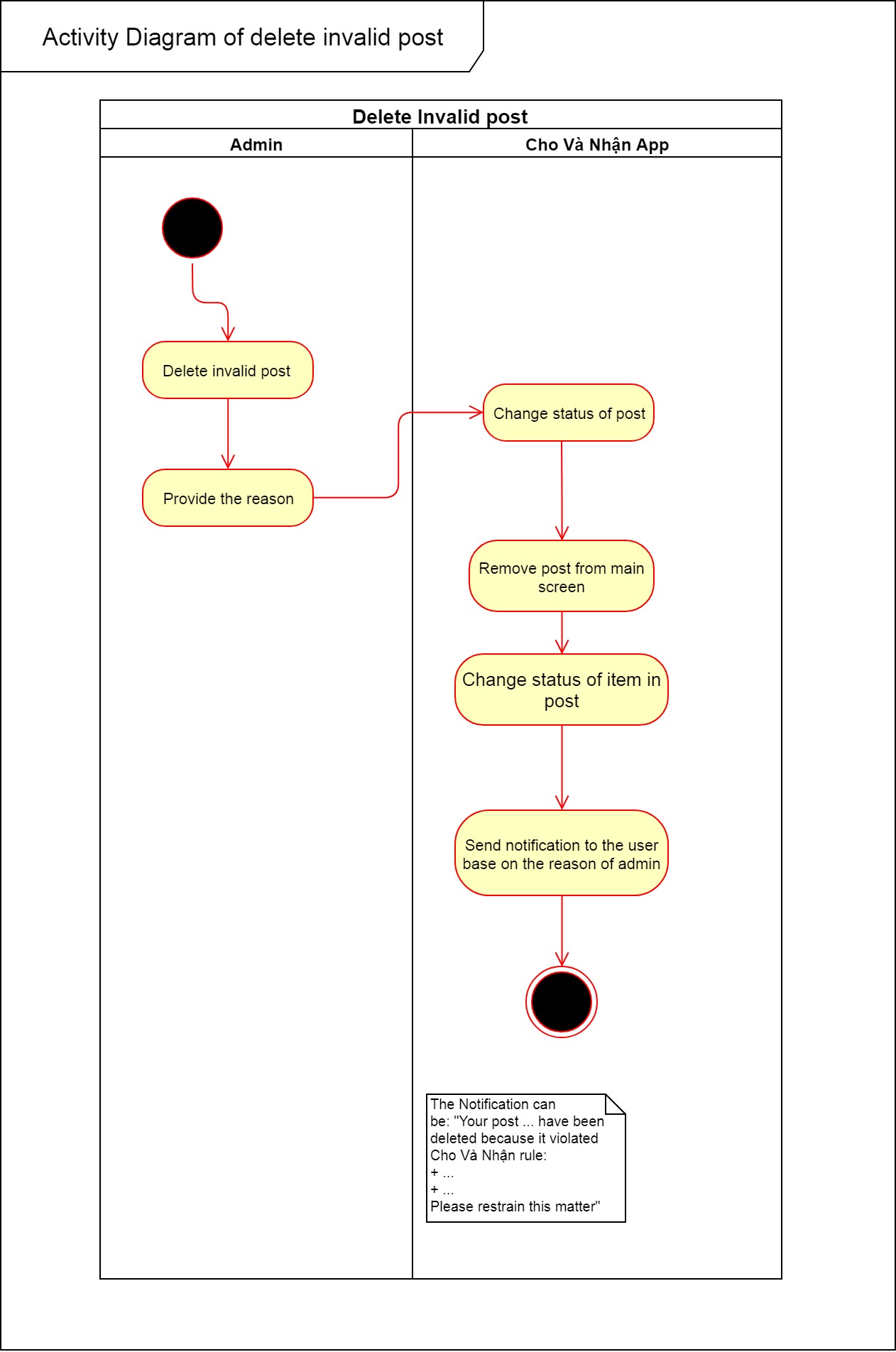


Figure ‑ Activity Diagram\_ Delete Invalid Post

### Category Management

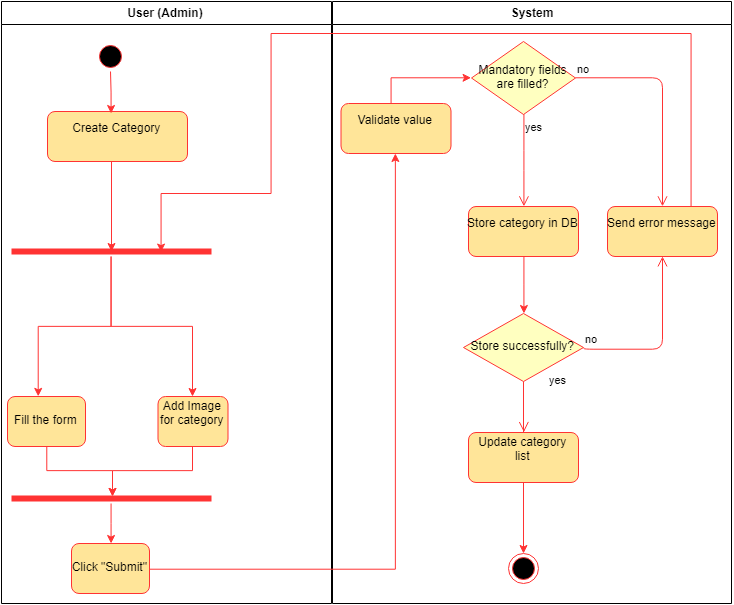


Figure ‑ Activity Diagram\_ Create Category

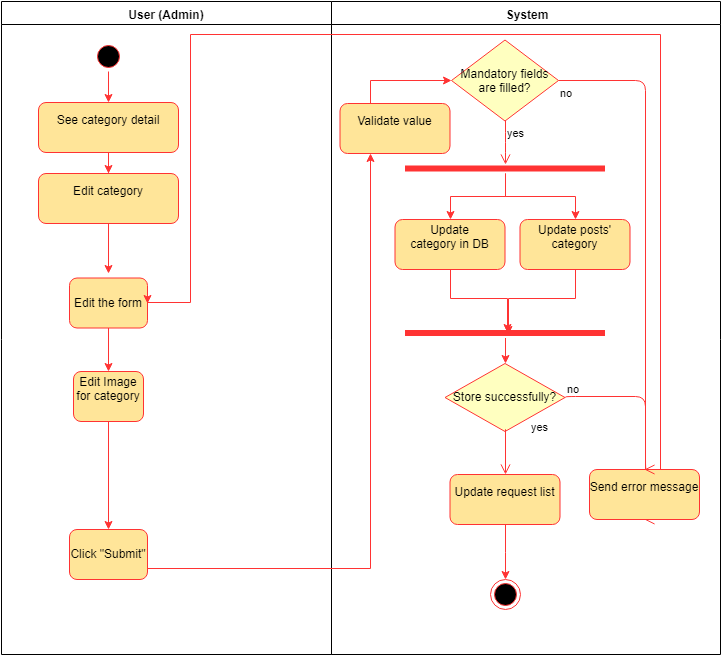


Figure ‑ Activity Diagram\_ Edit Category

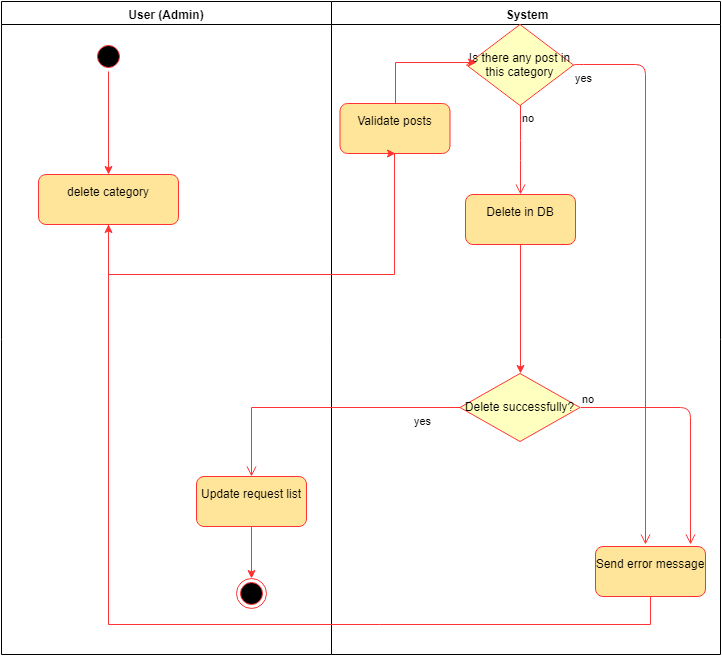


Figure ‑ Activity Diagram\_ Delete Category

### Feedbacks Management

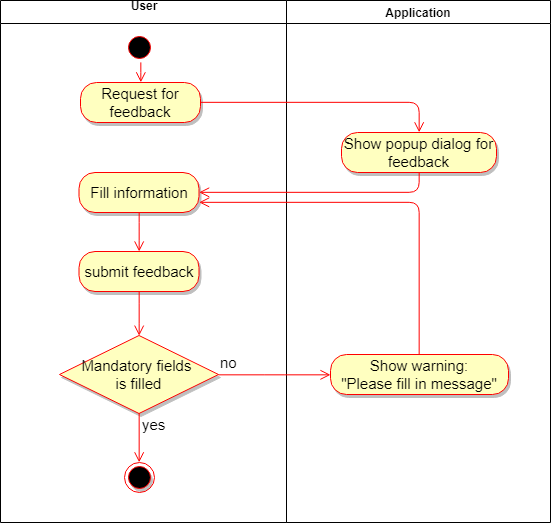


Figure ‑ Activity Diagram\_ Create Feedbacks

## Class Diagram

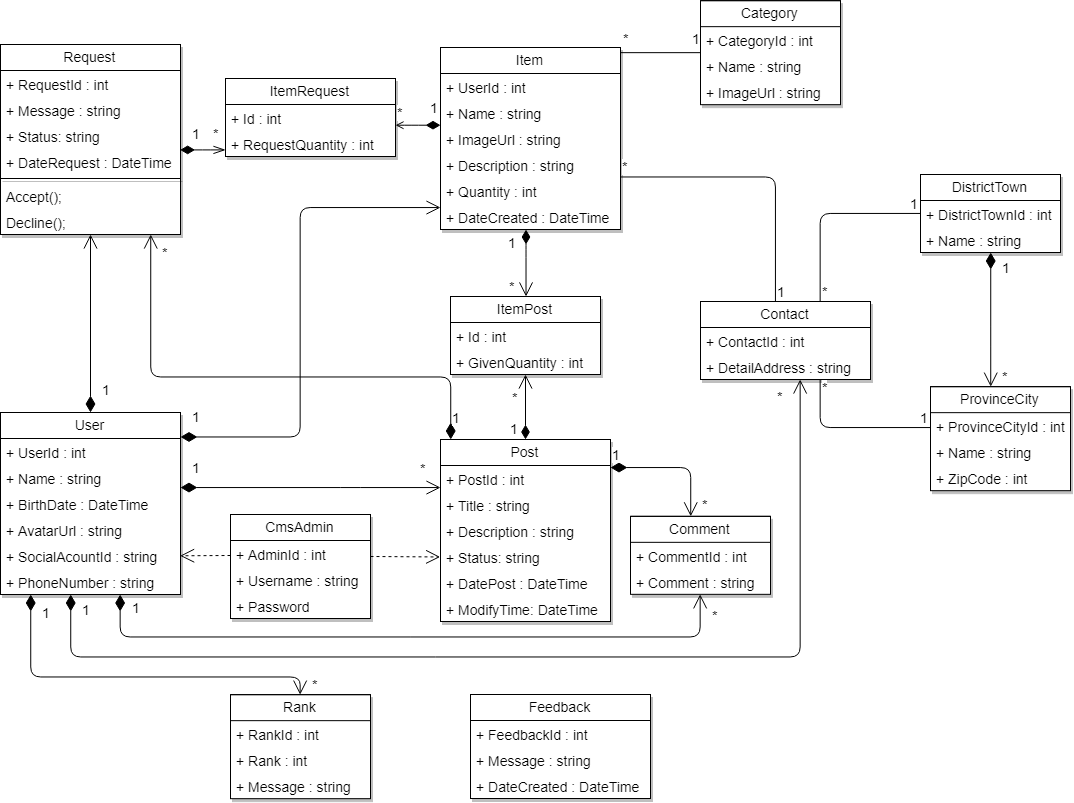


Figure ‑ Class Diagram

## Database

…

# APPENDIX A: [TBD]