|  |  |
| --- | --- |
| 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](#_Toc519841458)

[1.1 Purpose 6](#_Toc519841459)

[1.2 Scope 6](#_Toc519841460)

[2 REFERENCED DOCUMENTS 6](#_Toc519841461)

[2.1 Controlling Documents 6](#_Toc519841462)

[2.2 Controlled Documents 6](#_Toc519841463)

[2.3 Background Information 6](#_Toc519841464)

[3 ACTORS 7](#_Toc519841465)

[4 OVERVIEW 7](#_Toc519841466)

[5 FEATURES 8](#_Toc519841467)

[5.1 Device Orientation 8](#_Toc519841468)

[5.2 Start up 8](#_Toc519841469)

[5.2.1 User Interface 8](#_Toc519841470)

[5.2.2 Business Logic 8](#_Toc519841471)

[5.3 Login 9](#_Toc519841472)

[5.3.1 User Interface 9](#_Toc519841473)

[5.3.2 Business Logic 9](#_Toc519841474)

[5.3.2.1 Login using Facebook account 9](#_Toc519841475)

[5.3.2.2 Login using Google+ account 9](#_Toc519841476)

[5.3.2.3 Remember credentials 9](#_Toc519841477)

[5.3.3 Use Cases 10](#_Toc519841478)

[5.4 Post Management 12](#_Toc519841479)

[5.4.1 User Interface 12](#_Toc519841480)

[5.4.2 Business Logic 12](#_Toc519841481)

[5.4.3 Use Cases 13](#_Toc519841482)

[5.5 Request Management 14](#_Toc519841483)

[5.5.1 User Interface 14](#_Toc519841484)

[5.5.2 Business Logic 14](#_Toc519841485)

[5.5.3 Use Cases 14](#_Toc519841486)

[5.6 Conversation Management 16](#_Toc519841487)

[5.6.1 User Interface 16](#_Toc519841488)

[5.6.2 Business Logic 16](#_Toc519841489)

[5.7 Report management 16](#_Toc519841490)

[5.7.1 User Interface 16](#_Toc519841491)

[5.7.2 Business Logic 16](#_Toc519841492)

[5.7.2.1 User report a post 16](#_Toc519841493)

[5.7.2.2 Admin manage a list of reported posts 16](#_Toc519841494)

[5.7.3 Use cases 17](#_Toc519841495)

[5.8 Category Management 18](#_Toc519841496)

[5.8.1 User Interface 18](#_Toc519841497)

[5.8.2 Business Logic 18](#_Toc519841498)

[5.8.2.1 Admin manage category 18](#_Toc519841499)

[5.8.2.2 User choose a category to filtrate the posts 18](#_Toc519841500)

[5.9 Feedbacks Management 19](#_Toc519841501)

[5.9.1 User Interface 19](#_Toc519841502)

[5.9.2 Business Logic 19](#_Toc519841503)

[5.9.2.1 Quest create feedbacks 19](#_Toc519841504)

[5.9.2.2 Admin read feedbacks 19](#_Toc519841505)

[5.9.3 Use cases 19](#_Toc519841506)

[5.10 Ranking system 20](#_Toc519841507)

[5.10.1 User Interface 20](#_Toc519841508)

[5.10.2 Business Logic 20](#_Toc519841509)

[5.10.2.1 Ranking system for Giver 20](#_Toc519841510)

[5.10.2.2 Ranking system for Post 20](#_Toc519841511)

[6 Non-functional Requirements 21](#_Toc519841512)

[6.1 Operating System Support 21](#_Toc519841513)

[7 ANALYZE AND DESIGN 21](#_Toc519841514)

[7.1 Activity Diagram 21](#_Toc519841515)

[7.1.1 Login 21](#_Toc519841516)

[7.1.2 Item Management 24](#_Toc519841517)

[7.1.3 Post Management 27](#_Toc519841518)

[7.1.4 Request Management 30](#_Toc519841519)

[7.1.5 Report Management 33](#_Toc519841520)

[7.1.6 Category Management 34](#_Toc519841521)

[7.1.7 Feedbacks Management 37](#_Toc519841522)

[7.2 Class Diagram 38](#_Toc519841523)

[7.3 Database 38](#_Toc519841524)

[8 APPENDIX A: [TBD] 38](#_Toc519841525)

**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 |  | |

## Post Management

### User Interface

### Business Logic

Public posts screen is loaded after the application is loaded successfully. Users/guests can see all the posts that givers have posted. Besides that, users/guests can use filters to classify the posts to see what they concern. After users/guests choose a filter, the posts will be filtrated and sorted. There are 3 filter :

* Location (default is “Da Nang”)
* Category (default is “All category”)
* Sort by (Like number or Time)

Givers can create a new post by filling information: title, description, location and photos about the items they want to give. They can import the photos from their gallery or take a new picture. After they submit, this post will be create and update to system so all users/guests can see.

Givers modify their posts by changing post’s fields: title, description, location, status, add or remove photos. After they submit the changes, a notification will be sent to all takers who sent the requests to the post.

Users can only delete a post which has status “Sent”. If users want to delete a post in “Available” status, they must change its status first.

If users find a post containing useful items, they can like it, give a comment about it or even share it on Facebook, so their friends on Facebook can be able to know about it and can open the post in our application if they have already installed app in their phone. Or requiring install app to see the post if they have not installed app yet.

### 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 | |

|  |  |  |
| --- | --- | --- |
| 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, and notification is sent to Takers | |
| Alternative scenario 1 | Step 3) The Give and Receive app display error message :“Can’t delete” | |

## Request Management

### User Interface

### Business Logic

When usersfind a post containing useful items for them, they can send a request containing their message to the giver. User can only send the request to the posts in available status. After they send the request, a notification will be sent to the giver.

Givers can see all the requests of their posts including the request’s message and requester’s profile. Givers can claim that their items are sent to the right people by studying this information and decide who they want to give. Givers accept the request by sending contact information to the requester. Then they will negotiate by phone/ message.

After Givers send all their giving items to takers, they can change the status of their post to “Sent” so takers can aware that the post is no longer available to send a request.

Requesters will receive notification that their request is denied when givers decline their requests or givers change the status of the post to “sent” but didn’t accept their requests.

### Use Cases

|  |  |  |
| --- | --- | --- |
| Use Case | UC1: Create request to 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, and notification is sent to the Giver | |
| Alternative scenario 1 | Step 4) The Give and Receive app notifies required fields are missing. | |

|  |  |  |
| --- | --- | --- |
| Use Case | UC2: Delete (Cancel) 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. | |
| 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 (Accept or Decline Requests) |
| Actors | **Giver** |
| Pre-conditions | The user have logged in successfully |
| Post-conditions | One request is accepted or declined. The reply message (notification) is sent to the chosen one successfully |
| Main success scenario | 1. Giver launches request lists for a post. 2. Giver sees the detail of a request. 3. Givers accept the request. 4. Givers confirm reply message. 5. "Give and Take" displays list of requests again to user 6. "Give and Take" app sends notification to the Taker. |
| Alternative scenario 1 | 1. Givers receive a notification about a request.    1. Giver continues main step 2. |
| Alternative scenario 2 | 1. Giver accepts the request without see the detail. 2. Giver continues main step 3. |
| Alternative scenario 3 | 1. Giver declines the request. 2. The request is deleted and notification is sent to the Taker. |

## Conversation Management

### User Interface

### Business Logic

Giver and Taker can negotiate each other via conversation on system.

User can go to another user’s profile and start conversation with him/ her.

Message send to the user (and there is notification for this message), and the user Choose Reply or not.

The both users can delete the conversation. Once the conversation is deleted, notification will be sent to users involved.

## 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 all reviews of other Takers (from 1- 5 stars).

There is a list that shows top 50 generous givers.

There is a system to calculate the average number of stars Givers have and marks it on their profiles.

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

#### Ranking system for Post

User can click Like (Facebook) for the posts they think interesting and helpful.

The post which has the most number of Likes, will be on top.

# Non-functional Requirements

## Operating System Support

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

# ANALYZE AND DESIGN

## Activity Diagram

### Login

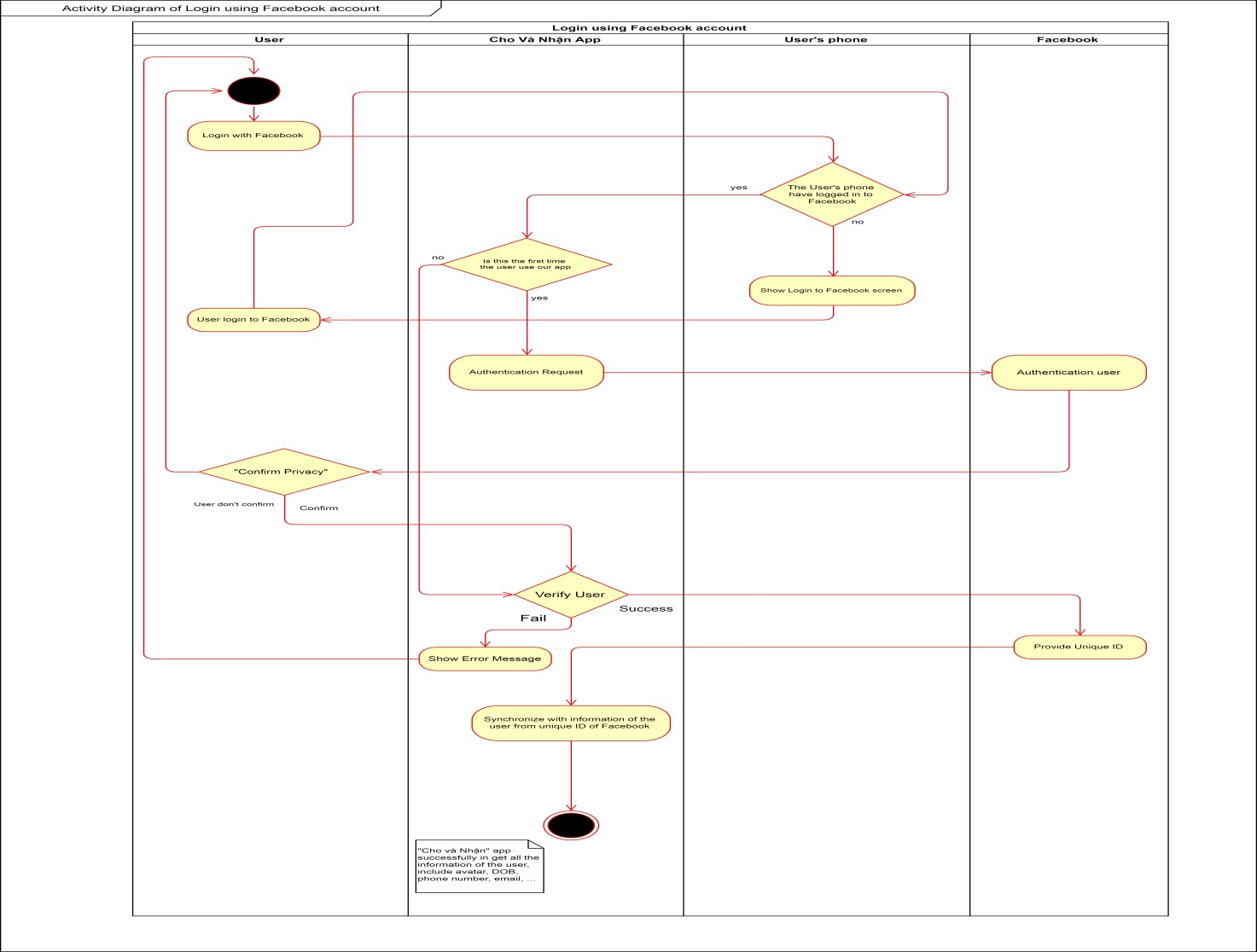


Figure ‑ Activity Diagram\_ Login Using Facebook

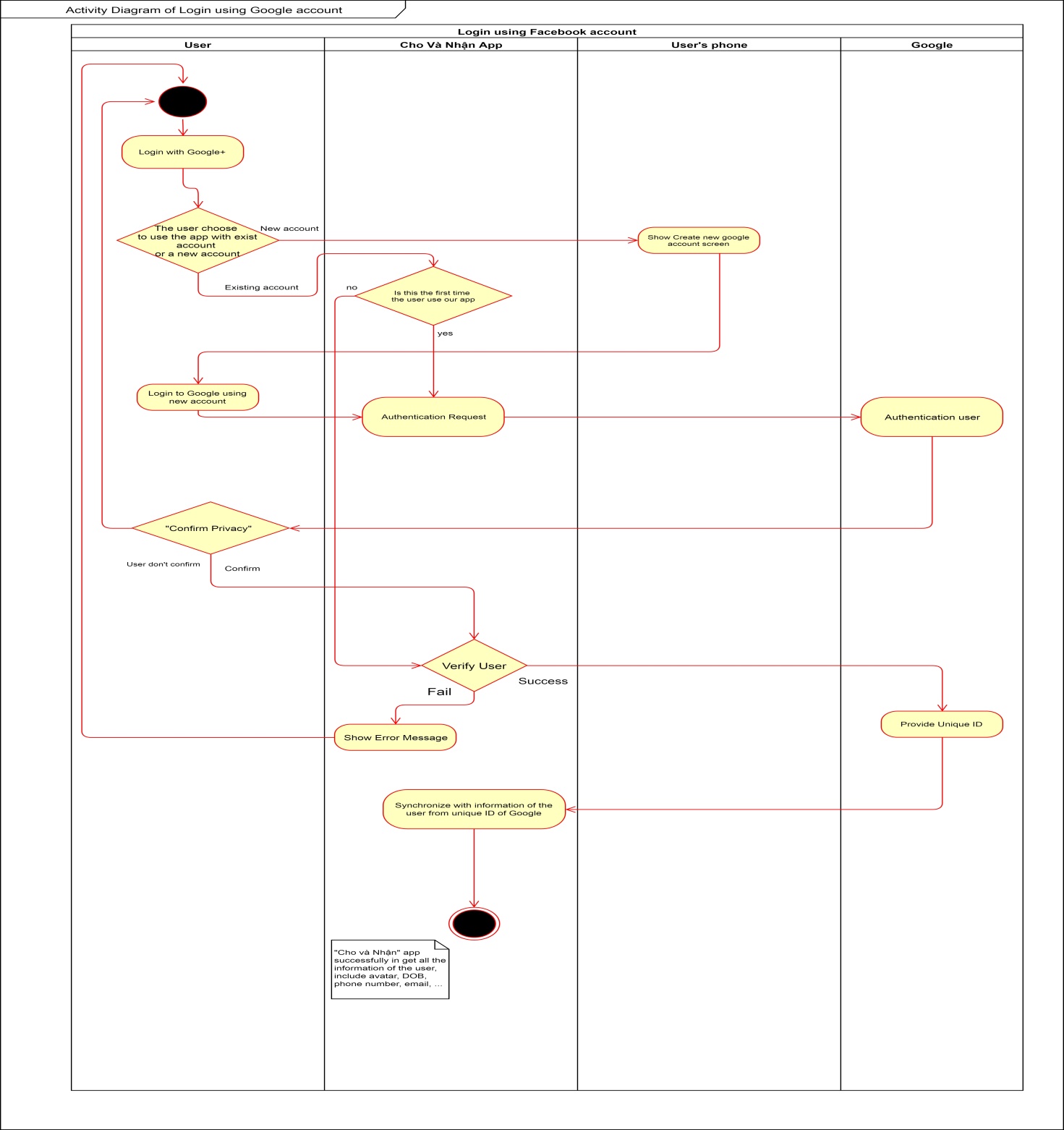


Figure ‑ Activity Diagram\_ Login Using Google

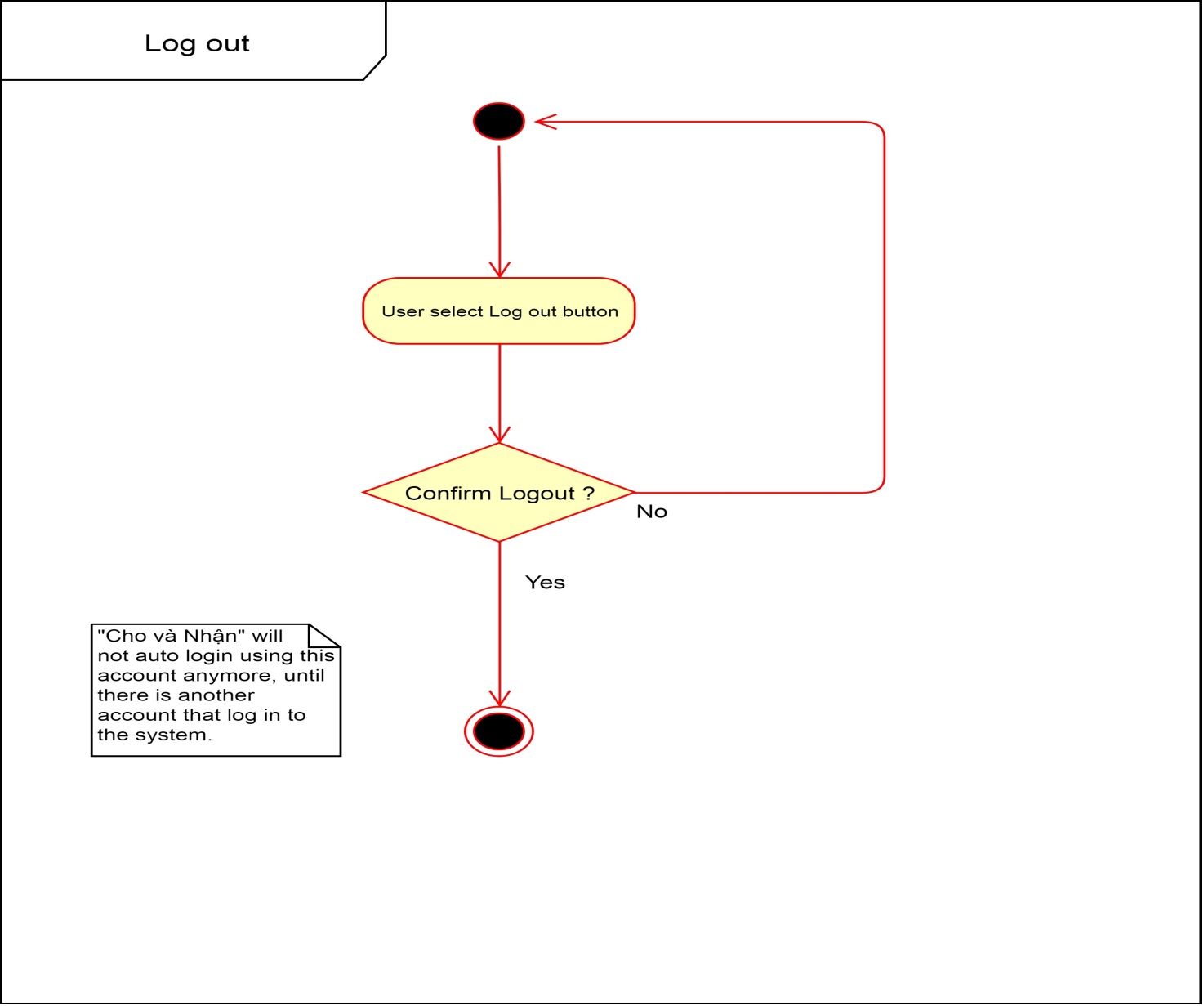


Figure ‑ Activity Diagram\_ Logout

### Item Management

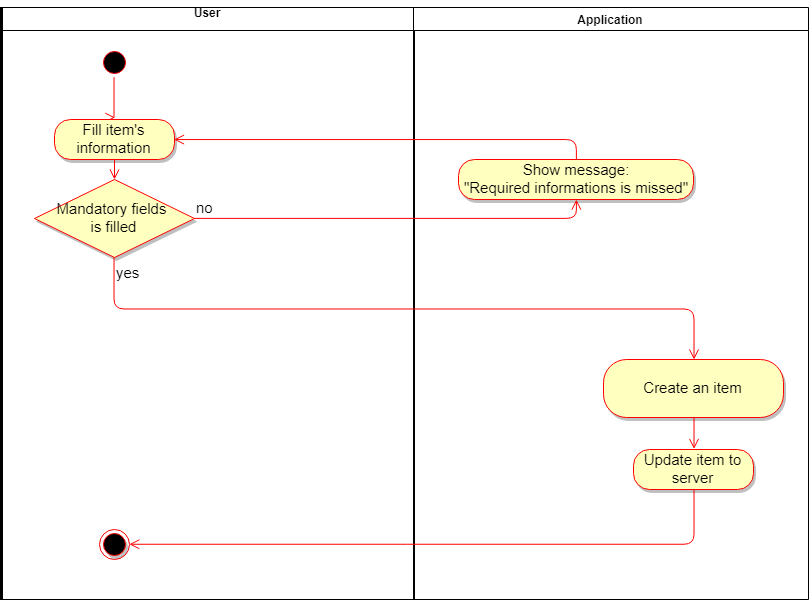


Figure ‑ Activity Diagram\_ Create Item

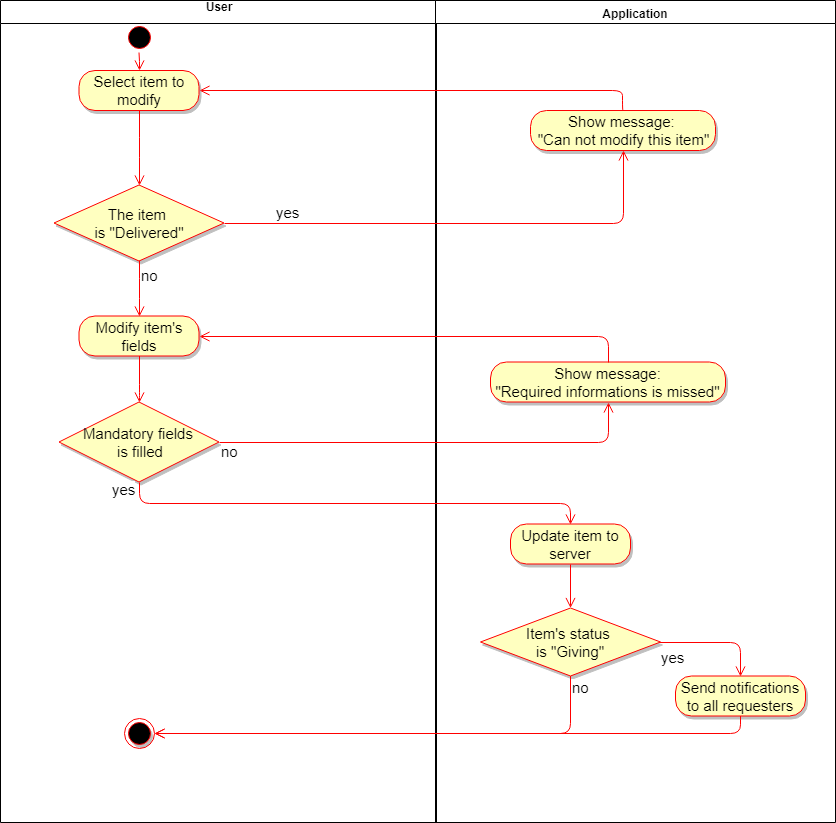


Figure ‑ Activity Diagram\_ Modify Item

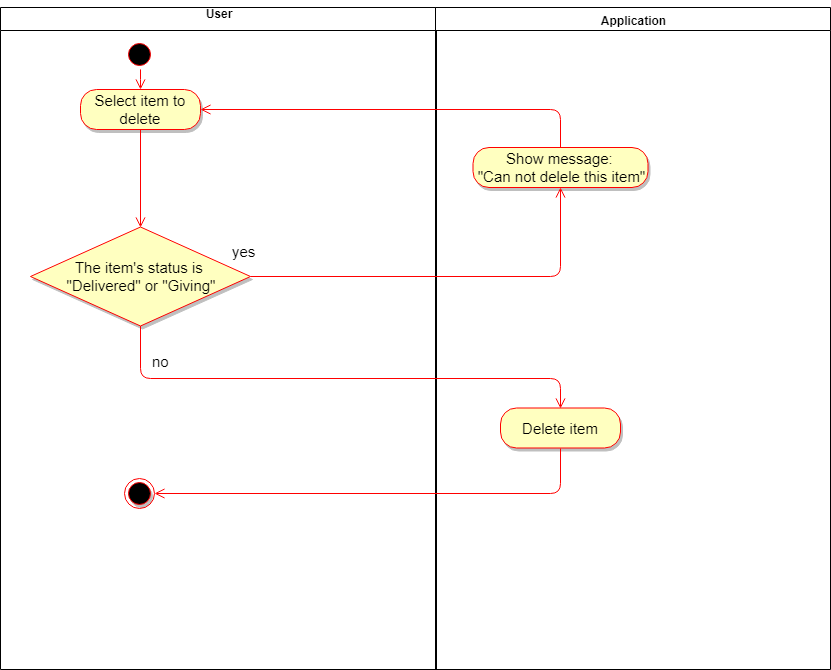


Figure ‑ Activity Diagram\_ Delete Item

### Post Management

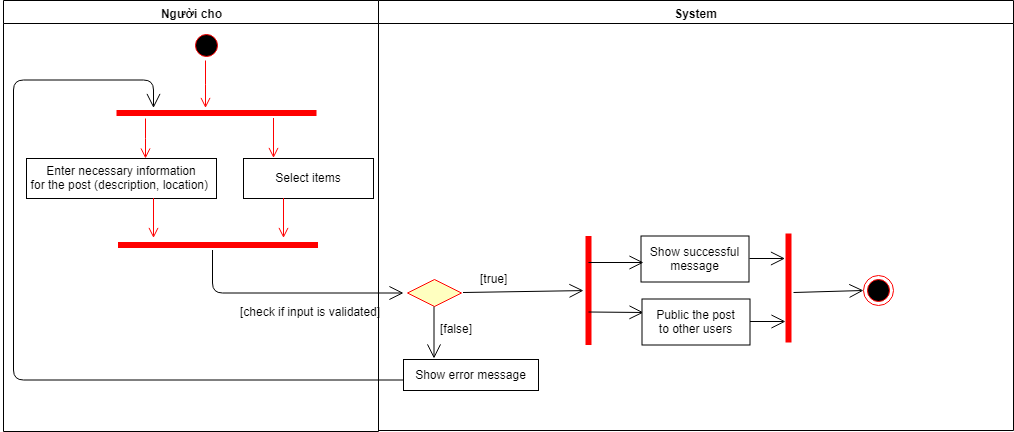


Figure ‑ Activity Diagram\_ Create Post

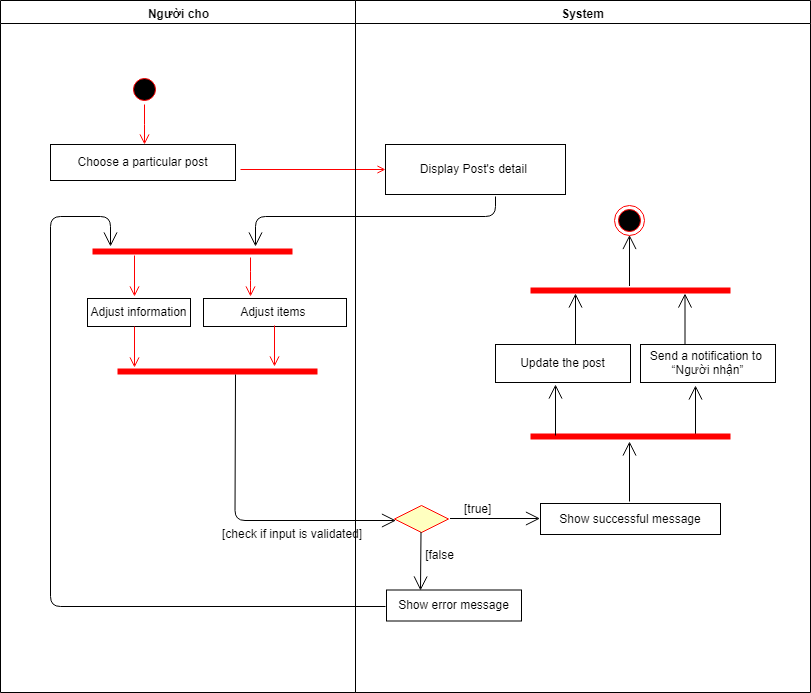


Figure ‑ Activity Diagram\_ Edit Post

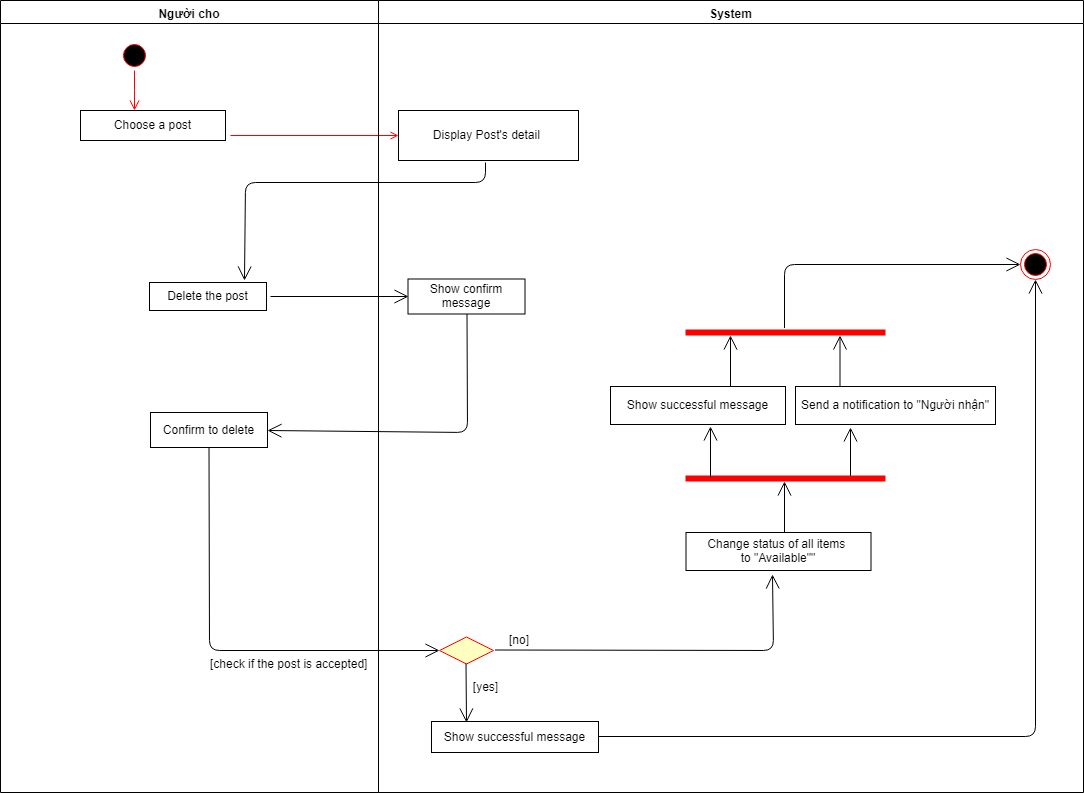


Figure ‑ Activity Diagram\_ Delete Post

### Request Management

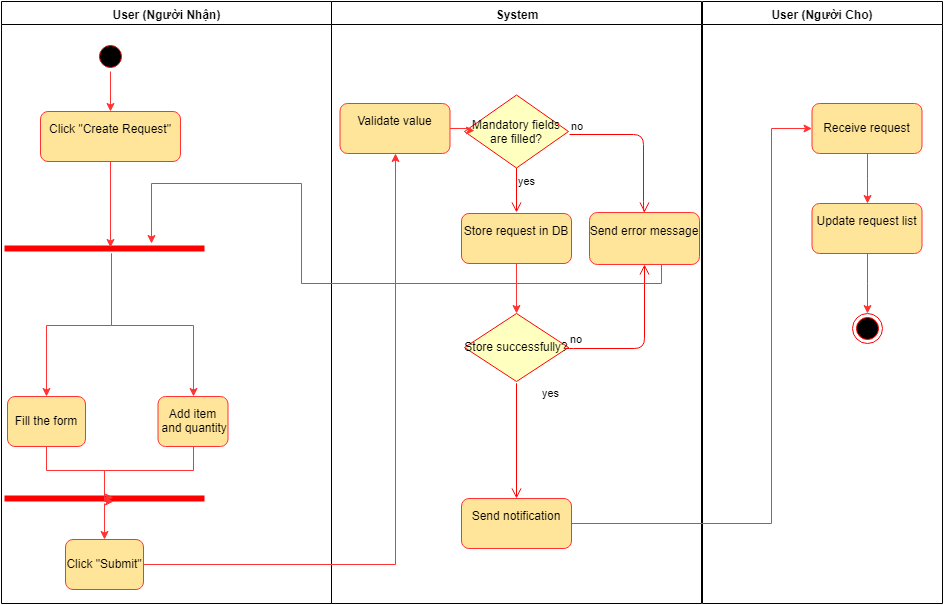


Figure ‑ Activity Diagram\_ Create Request

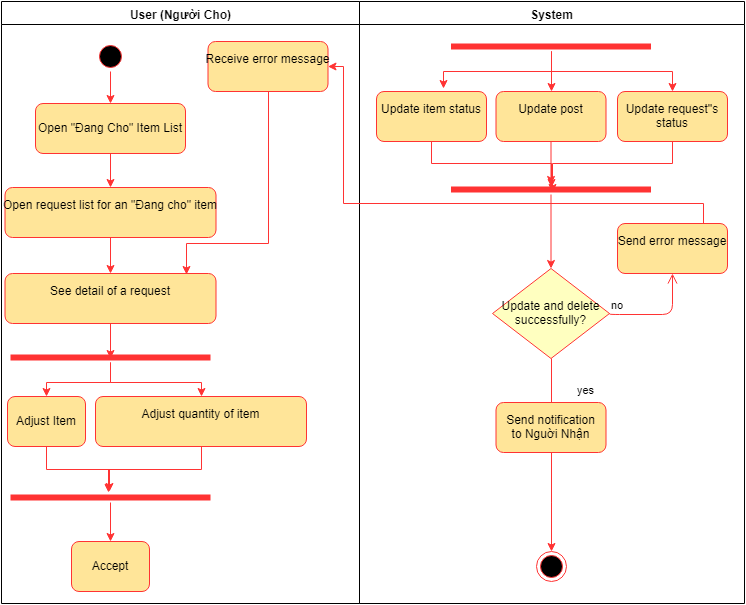


Figure ‑ Activity Diagram\_ Accept Request

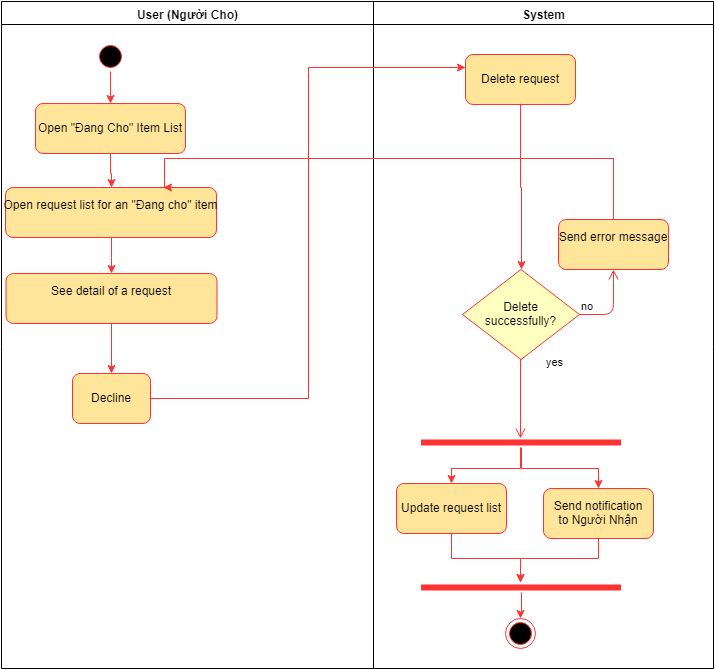


Figure ‑ Activity Diagram\_ Decline Request

### Report Management

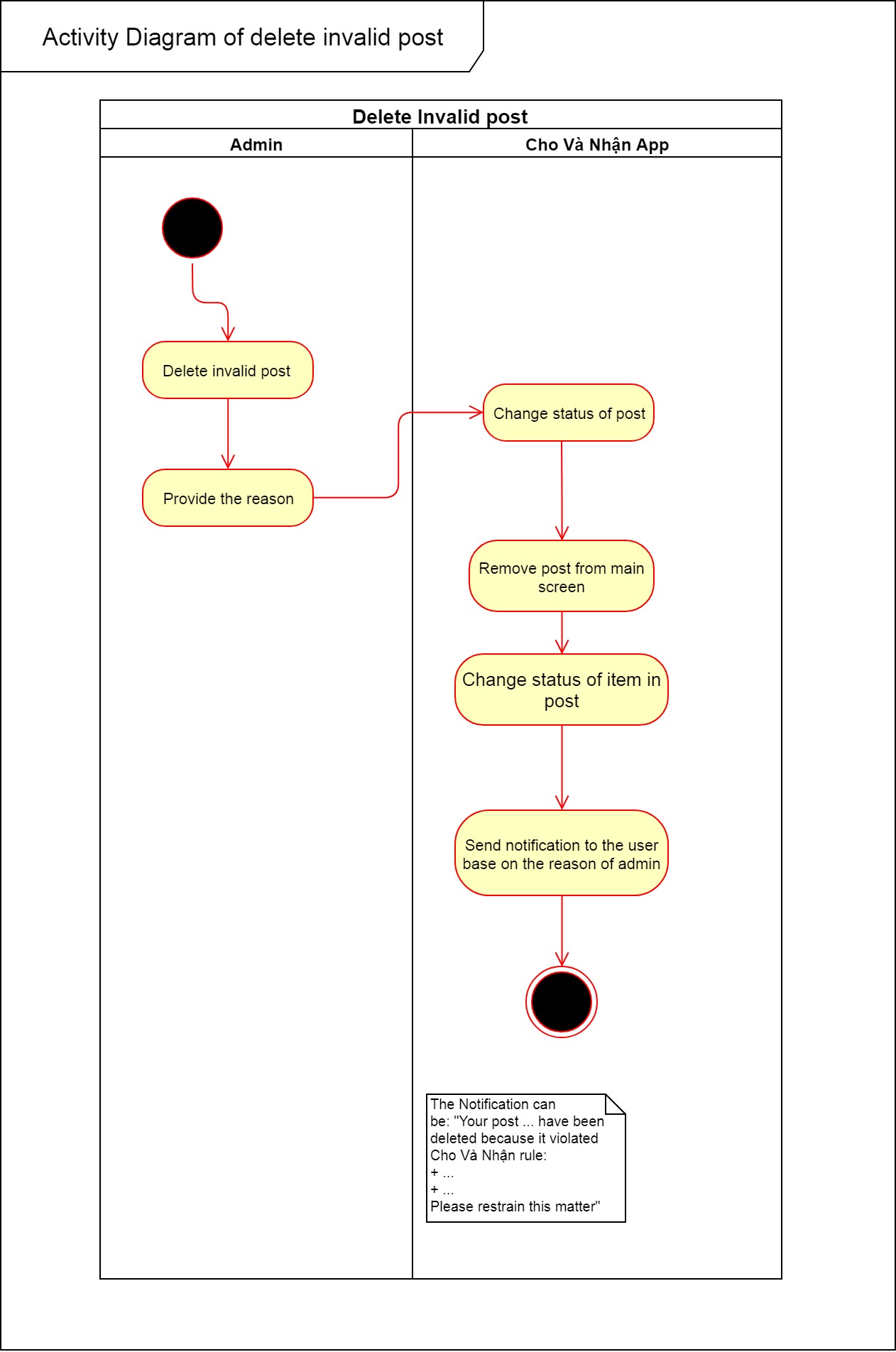


Figure ‑ Activity Diagram\_ Delete Invalid Post

### Category Management

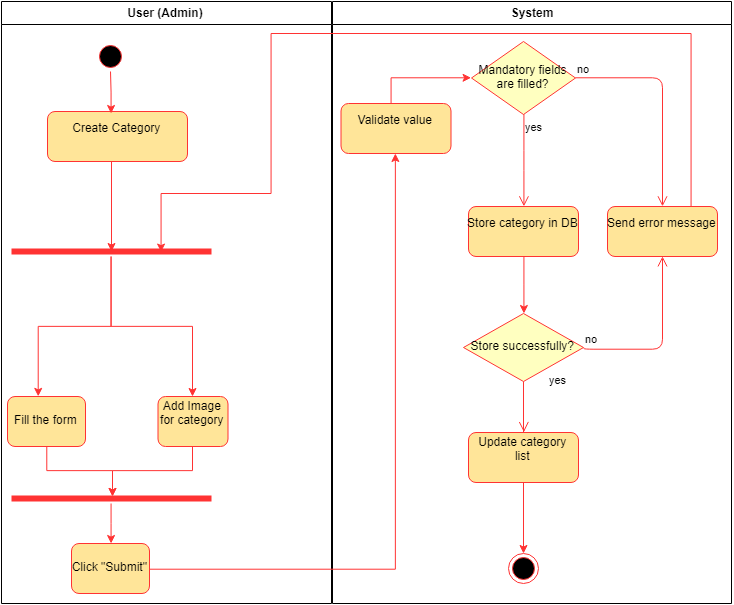


Figure ‑ Activity Diagram\_ Create Category

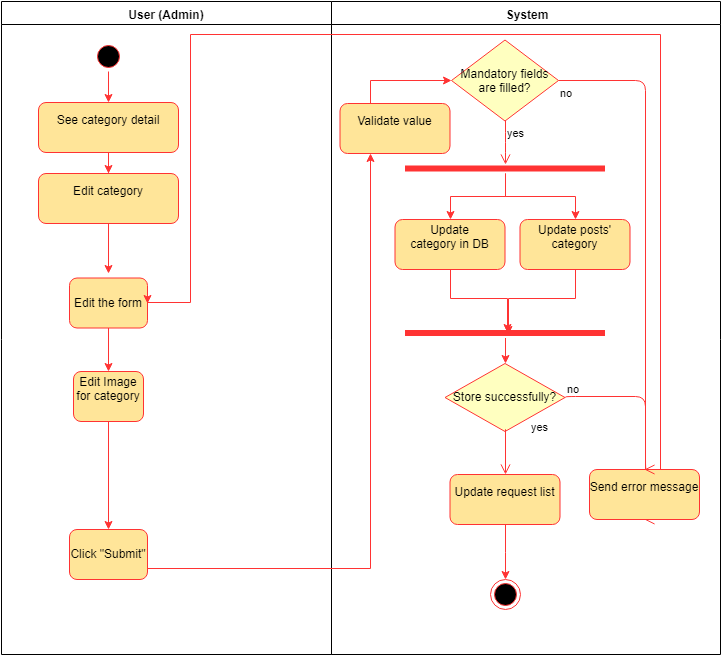


Figure ‑ Activity Diagram\_ Edit Category

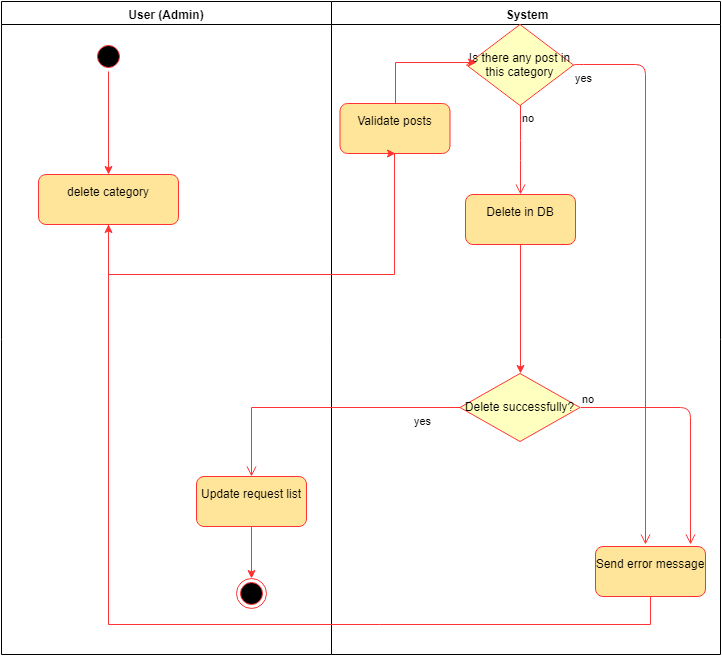


Figure ‑ Activity Diagram\_ Delete Category

### Feedbacks Management

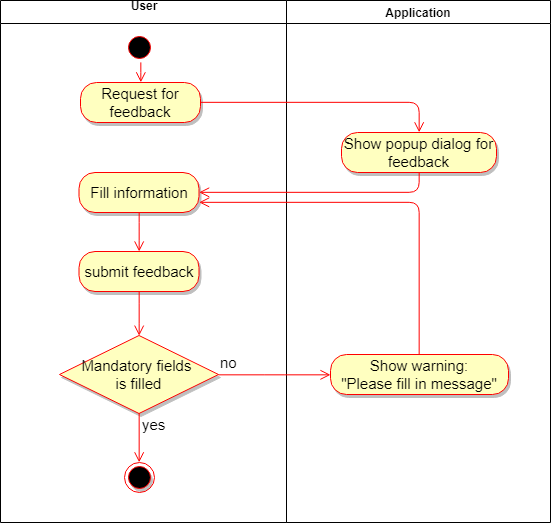


Figure ‑ Activity Diagram\_ Create Feedbacks

## Class Diagram

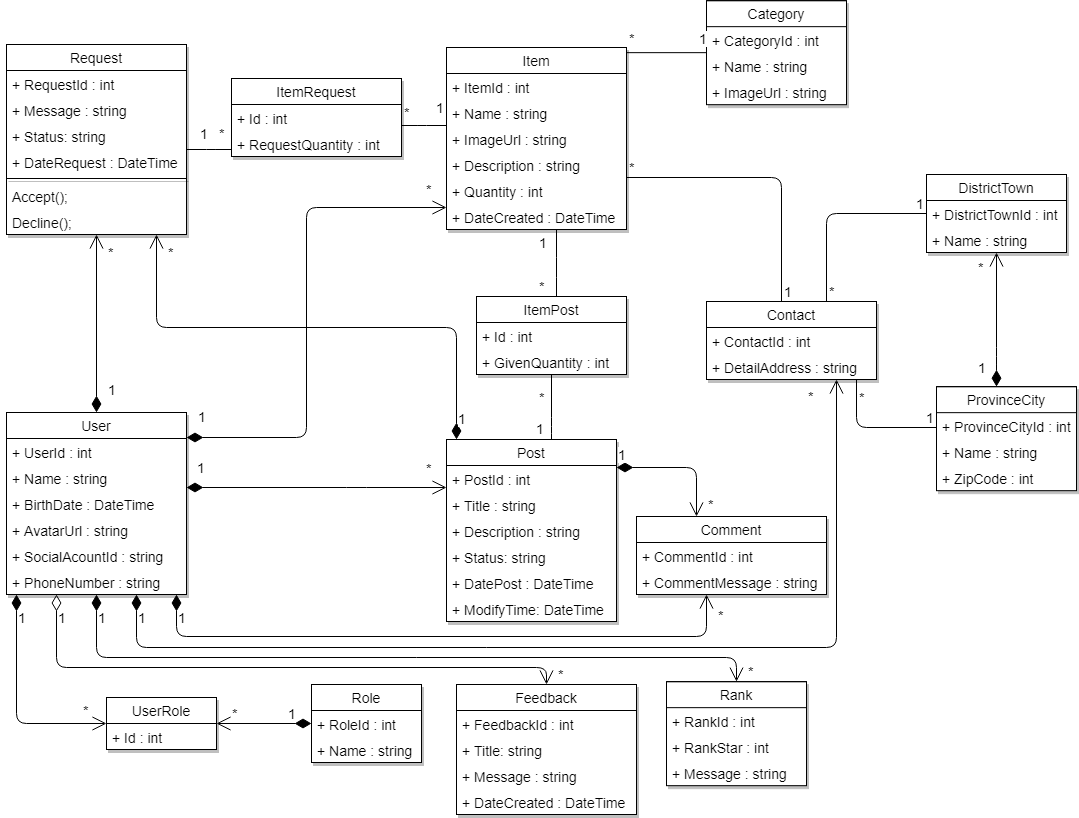


Figure ‑ Class Diagram

## Database

…

# APPENDIX A: [TBD]