**COURSE PROJECT**

**Instructor:** Truong Ninh Thuan

**Project team:**

* Tran Quang Cuong Student.No: 13020049 K58CA
* Nguyen Thanh Trung Student.No: 13020463 K58CA
* Nguyen Vu Hoang Vuong Student.No: 13020763 K58CA

**Project requirement:** Analyze and design a documentation for any software project.

Table of contents

1. Social Network Requirements
   1. Problem statement
   2. Glossary
   3. Supplementary Specification
   4. Use-Case Model
2. Social Network Analysis
   1. Architectural Analysis
   2. Use-case analysis
      1. Use-case Realization Interaction Diagrams
      2. Use-case Realization View of Participating Classes (VOPCs)
3. Social Network Design
   1. Identify Design Elements
   2. Describe the Run-time Architect
   3. Describe Distribution
   4. Use-case design
   5. Subsystem Design
   6. Class Design
      1. Describe each class or interface
      2. Class diagram in total

SECTION 1: Social Network Requirements

* 1. **Problem statement**

Online social networking nowadays is extremely popular, attracts many users of all age, especially young ones. Although It is possible in person, workplace, universities, and high schools it is most popular online.

Social network is the mapping and measuring of relationships and flows between people, groups, organizations, computers, URLs, and other connected information/knowledge entities. The nodes in the network are the people and reoups while the links show relationships or flows between the nodes. Social network provides both a visual and a mathematical analysis of human relationships.

Social networking sites are not only for you to communicate or interact with other people globally but, this is also one effective way for business promotion. A lot of business minded people these days are now doing business online and use these social networking sites to respond to customer queries. It isn't just a social media site used to socialize with your friends but also, represents a huge pool of information from day to day living.

We are tasked with to developing a Social Network Website project where people can make friends as well as stay connected and interact with each other even if they are distance apart.

* 1. **GLOSSARY**
* ***Server***:

Where receives requests and send responses requests.

* ***User***:

A person who use our social network.

* ***Post***:

A Wall Post object works very similar to a Message object. The only diﬀerence is that the Wall Post objects are viewable to all friends of the user. Another diﬀerence is that a Wall Post object can hold comments. If a user adds a wall post to his or her own account, their status will change to the new wall post.

* ***Comment***:

A user’s writing that be under / following the post – felling about the post.

* ***Wall***:

A user’s home page - The wall is a place where the account owner can express his current feeling or thoughts with a wall status. Other users can add their thoughts by creating wall posts that show up below the status. It is basically a way for many users to communicate publicly or simply leave their thoughts about each other.

* ***Chatting***:

One of the advantages of having friends is the ability to have live communications with them via chat. When a friend is online, he/she will appear in the chat tab as an available friend to chat with. To start a chat session, simply click on the name of the friend which will begin a new chat session. The little bar at the bottom will change from chat disabled to chatting with friend name as seen in the ﬁgure below. Also, the friend receiving the chat will get a notiﬁcation that a new chat session has started. Then the two friends can chat with each other until one or both log out.

* ***Friend*** ***list***:

A management of friends. It contains all user's friends.

* ***Event***:

In addition to messaging your friends, social networking sites are a great place to alert your friends of important upcoming events. For example, if a user is throwing a birthday party, he/she could create an event an invite friends he/she wants to attend or simply make it a public for anyone. The process of creating an event is as simple as ﬁlling out a form.

* ***Privacy***:

All about privacy rules of User

* ***Media Uploading:***

As people use their social networking account, they will want to be able to upload funny or interesting images, video, music, etc., to share with their friends. The media upload section will be located at the media tab, where users are able to specify a ﬁle to upload as well as provide a short description of the ﬁle to be uploaded.

* ***Note***:

Notes are very similar to wall posts with one major exception. Unlike wall posts which can be created by anyone, only the account owner can create notes. It’s main purpose if for the account owner to create reminders for him/herself, or create announcements for any other users to see.

* ***Messaging***:

Unlike with chatting, users can send a message to any other user. For user A to send a message to user B, he/she simply goes to the message center tab and clicks compose message. He then ﬁlls out the form as shown in the ﬁgure below with the email of the recipient, the message title, and the message content. The new message will then be in User B’s inbox in the message center.

* 1. **SUPPLEMENTARY SPECIFICATION**

**Objectives**

The purpose of this document is to define requirements of the Social Networking Website. This Supplementary Specification lists the requirements that are not readily captured in the use cases of the use-case model. The Supplementary Specifications and the use-case model together capture a complete set of requirements on the system.

**Scope**

This Supplementary Specifications applies to the Social Networking Website, which will be developed by the ooad students.

This specification defines the non-functional requirements of the system; such as reliability, usability, performance, and supportability, as well as functional requirements that are common across a number of use cases. (The functional requirements are defined in the Use Case Specifications.)

**References**

None.

**Functionality**

The social network to be developed is a free website where people can access via a computer or a smartphone with Internet connection.

* Multiple user can access their homepage at the same time.
* Anyone can register an account. An account has a profile which contains provided user’s information. Some are private, such as email, mobile number; password, and the rest are public to other users. Changing profile is available.
* Users can share their story, their thought by writing a post. Unlike Facebook or Twitter, all posts are public.
* Any user can see what you have written, vote (like/dislike) and make comments. Posts are displayed in which we call a “wall” by newest post to oldest post. When one user goes to another user’s wall, that user’s wall should be appeared.
* Users can make friend easily by click Add friend to send request to their friends. Visit friends wall, chatting, even voice or video call.
* Users can chat with each other by a chat window. The chat window keep conversations include old ones between users. When a message came, users must be received a notification about it.

**Usability**

* The website has user-interface of html5, css (bootstrap) which user can easily to understand see.

**Reliability**

* The system will be available 24 hours a day and 7 days a week, with no more than 10% down time.

**Performance**

* The system shall support up to thousands users against the central database at any time.
* The system should provide access to the user’s database with no more than a 10 second latency.
* The system must be able to complete 80% of all transactions within 2 minutes.

**Supportability**

None.

**Security**

* The system must prevent user to do something wrong with the Privacy rules.

**Design Constraints**

* The system must have Web interface in stage 1 (and maybe have Windows app in stage 2).
* System interface must be good-looking, visually, easy to understand it’s features.
  1. **USE-CASE MODEL**

**1.4.1 Registration**

**Brief Description**

This use case describes how a user register an account of the Social Networking Website.

**Flow of Events**

*Basic Flow*

This use case starts when the actor wishes to register an account of the SOCIAL NETWORK WEBSITE.

1. The actor clicks on Register button.
2. The system response a webpage that include a form for actor. Actor need to full fill the form and click Submit button.
3. The system validates the entered name and password and other detail information that the actor had filled the form.
4. If success (all the information that actor provide to system are valid) the actor’s account should be stored in system database and redirect actor to Login page
5. If not success (one of these information is invalid, the system will return registration page with highlight which wrong or invalid information)

*Alternative Flows*

**Invalid provided Information**

If, in the Basic Flow, the actor enters an invalid name and/or password/ other information, the system displays an error message. The actor can choose to either return to the beginning of the Basic Flow or cancel the registration, at which point the use case ends.

If one of these provided information is invalid, the system will return registration page with highlight which wrong or invalid information.

**Special Requirements**

None.

**Pre-conditions**

The system is in the registration state and has the registration screen displayed.

**Post-conditions**

If the use case was successful, the actor’s account will be stored at system database. If not, the system state is unchanged.

**Extension Points**

None.

**1.4.2 Login**

**Brief Description**

This use case describes how a user logs into the Social Networking Website.

**Flow of Events**

*Basic Flow*

This use case starts when the actor wishes to log into the SOCIAL NETWORK WEBSITE.

1. The actor enters his/her name and password.
2. The system validates the entered name and password and logs the actor into the system.

*Alternative Flows*

**Invalid Name/ Password**

If, in the Basic Flow, the actor enters an invalid name and/or password, the system displays an error message. The actor can choose to either return to the beginning of the Basic Flow or cancel the login, at which point the use case ends.

If actor enter her/his name or password that wrong over 5 times, server will response a message that “Only can login after 15 minutes later.”

**Special Requirements**

None.

**Pre-conditions**

The system is in the login state and has the login screen displayed.

**Post-conditions**

If the use case was successful, the actor is now logged into the system. If not, the system state is unchanged.

**Extension Points**

None.

**1.4.3 Update User Profile**

**Brief Description**

This use case describes how a user Update their profile for an account of the Social Networking Website.

**Flow of Events**

*Basic Flow*

This use case starts when the actor wishes to Update Profile for an account of the SOCIAL NETWORK WEBSITE.

1. The actor clicks on Edit Profile button.
2. The system response a webpage that include a form for actor. Actor need to full fill the form or edit the information that have in form and click Submit button.
3. The system requires account password to makes any changes to logged account.
4. Actor should enter password correctly.
5. The system validates the entered name and password and other detail information that the actor had filled the form.
6. If success (all the information that actor provide to system are valid) the actor’s account should be update in system database and redirect actor to Login page
7. If not success (one of these information is invalid, the system will return Edit Profile Page with highlight which wrong or invalid information)

*Alternative Flows*

**Invalid provided Information**

If, in the Basic Flow, the actor enters an invalid name and/or password/ other information, the system displays an error message. The actor can choose to either return to the beginning of the Basic Flow or cancel the Updating profile, at which point the use case ends.

If success (all the information that actor provide to system are valid) the actor’s account should be update in system database and redirect actor to Login page

If not success (one of these information is invalid, the system will return Edit Profile Page with highlight which wrong or invalid information)

**Special Requirements**

None.

**Pre-conditions**

The system is in the logged state and has the Wall screen displayed.

**Post-conditions**

If the use case was successful, the actor’s account will be updated at system database. If not, the system state is unchanged.

**Extension Points**

None.

**1.4.4 Friend Manager**

**Brief Description**

This use case describes how a user manager friend list on the Social Networking Website.

**Flow of Events**

*Basic Flow*

This use case starts when the actor wishes to **add / delete / block** another SNW user on the SOCIAL NETWORK WEBSITE.

1. Add friend:
   * 1. Actor X click “Add friend” button on the other wall or on the pop up when hover mouse other actor avatar.
     2. System send “Add friend request” to other actor and respone to actor X that “Friend request sent”. Then update the database.
2. Delete friend:
   * 1. Actor X click “Delete friend” button on the other (Y) wall or on the pop up when hover mouse other actor avatar.
     2. System makes a warning that “Do you really want to delete this friend?” asked for “Yes” or “No”.
     3. If actor select Yes, System immediately delete Y from friend list of X then update the database.
3. Block friend:
   * 1. Actor X click “Block” button on the other (Y) profile.
     2. System update database a constraint that X and Y are hidden with each other.

*Alternative Flows*

**Add friend**

If, in the Basic Flow, “Add friend request” not be accept from receiver two actor will not be friend.

**Delete friend**

If in the Basic Flow, actor select “No”, nothing changes to actor friend list.

**Special Requirements**

None.

**Pre-conditions**

The system is in the logged state and has the wall screen displayed.

**Post-conditions**

If the use case was successful, the actor friend list is updated. If not, the system database is unchanged.

**Extension Points**

None.

**1.4.5 Posting**

**Brief Description**

This use case describes how a user post text on the Social Networking Website.

**Flow of Events**

*Basic Flow*

This use case starts when the actor wants to post a text on the SOCIAL NETWORK WEBSITE.

1. The actor fill the post text-box and maybe can add something such as image, video, audio then click Post.
2. Server receive text, store into database and then send response to the actor that the text is posted. On other hand server send notifications to all the actor's follower who is following the actor.

*Alternative Flows*

**Invalid file extensions or Network errors**

If the post is valid that does not contains anything which due to the errors or warning of system. Text – post will be store in system database and then show on place where actor post in.

If the post contains invalid file name or file extensions, server will show messages to actor what invalid is.

**Invalid Posting place**

If the post is posted in a group or page that actor have not permission to post there, system should return an warning message: “You are not allowed to post here.”

**Special Requirements**

None.

**Pre-conditions**

The system is in the logged state.

Wall screen displayed.

**Post-conditions**

If the use case was successful, actor can view their post on wall.

**Extension Points**

None.

**1.4.6 View Post**

**Brief Description**

This use case describes how a user view a post on the Social Networking Website.

**Flow of Events**

*Basic Flow*

This use case starts when the actor wants to view a post on the SOCIAL NETWORK WEBSITE.

1. The actor Click on their name or the other user’s name to show their wall. In other way, click on homepage.
2. Server returns a list of post and event on main screen of homepage, wall.

The posts are shown shortly almost.

1. User click on “Read more…” to expand the post to view in detail.

*Alternative Flows*

**Not have permission**

If the posts do not belong to the user that was friend with the actor the posts do not show defaultly.

**Bad internet**

If the post’s data is too long to load or load slowly, the post should not be view by any user.

**Special Requirements**

**None**

**Pre-conditions**

The system is in the logged state.

**Post-conditions**

If the use case was successful, actor can view their post on wall.

**Extension Points**

None

**1.4.7 Edit/Delete Post**

**Brief Description**

**Flow of Events**

**Special Requirements**

**Pre-conditions**

**Post-conditions**

**Extension Points**

**1.4.8 Like Post**

**Brief Description**

**Flow of Events**

**Special Requirements**

**Pre-conditions**

**Post-conditions**

**Extension Points**

**1.4.9 Comment Post**

**Brief Description**

**Flow of Events**

**Special Requirements**

**Pre-conditions**

**Post-conditions**

**Extension Points**

**1.4.10 View Notifications**

**Brief Description**

**Flow of Events**

**Special Requirements**

**Pre-conditions**

**Post-conditions**

**Extension Points**

**1.4.11 Chatting**

**Brief Description**

This use case describes how a user chatting with their friends on the Social Networking Website.

**Flow of Events**

*Basic Flow*

This use case starts when the actor wants to chatting with their friend on the SOCIAL NETWORK WEBSITE.

1. The actor clicks on Name of the friend which will begin a new chat session.
2. The little bar at the bottom will change from chat disabled to chatting with friend which named by Friend's name.
3. An actor type a text into the text box then click Send or Enter. The text will be sent to server.
4. Server receive text, store into database and then send response to the other one who will be receive the messages.
5. The system validates the entered name and password and other detail information that the actor had filled the form.

*Alternative Flows*

**None**

**Special Requirements**

None.

**Pre-conditions**

The system is in the logged state.

Two actor is friend on system. Actors do not block each other.

**Post-conditions**

If the use case was successful, the actor 2 can live chatting with actor 1, system save all messages automatically by default.

**Extension Points**

None.

SECTION 2: Social Network Analysis

2.1 Architectural Analysis

2.1.1 Key Abstractions

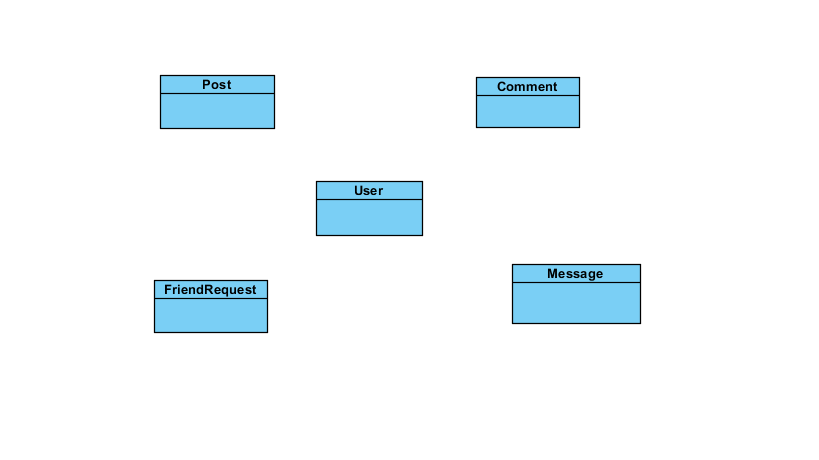


Figure Key Abstractions

2.1.1.1 Key Abstraction Definitions

User:

Post:

Comment:

FriendRequest:

Message:

2.1.2 Upper-Level Components and Their Dependencies

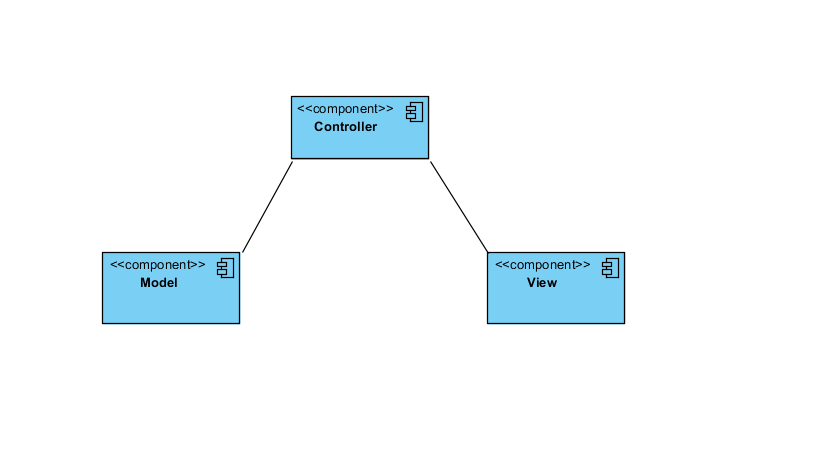


Figure The upper layer architecture

2.1.2.1 Component Definitions

Controller: Keep navigation tasks received requests from user and call the accordance methods to handle them. For example, this component will receive the request from url and form to manipulate directly with database.

Model: This is the component that contains the static data (database), method to access, query and process with it.

View: Assume the information display, interaction with users, which contains all the GUI objects such as text boxes, images, e.t.c. Understand a simple way, it is a set of form or HTML files.

2.2 Use-case Analysis

2.2.1 Use-case Realization Interaction Diagrams

2.2.1.1 Registration

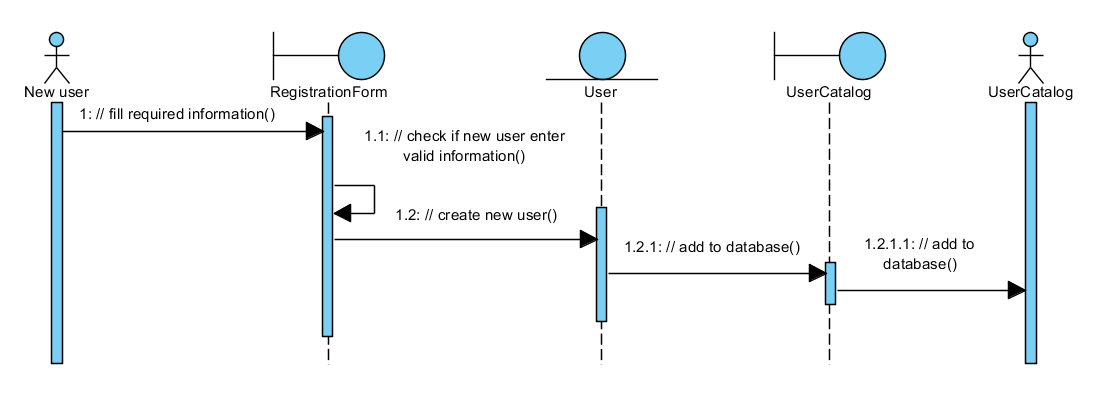


Figure Registration - Basic Flow

2.2.1.2 Login

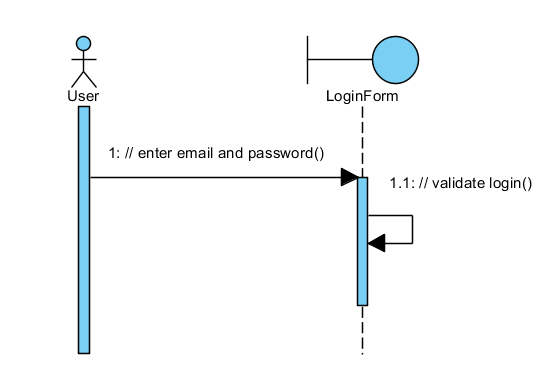


Figure Login - Basic Flow

2.2.1.3 Update User Profile

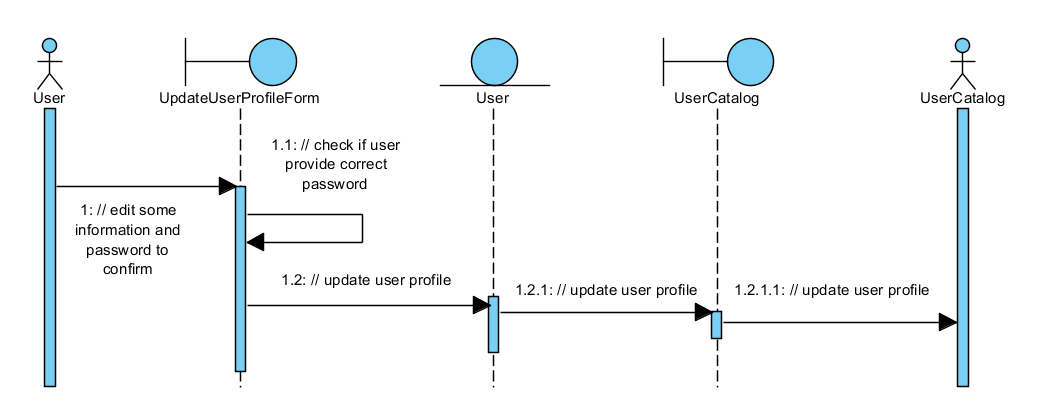


Figure Update User Profile

2.2.1.4 Friend Manager

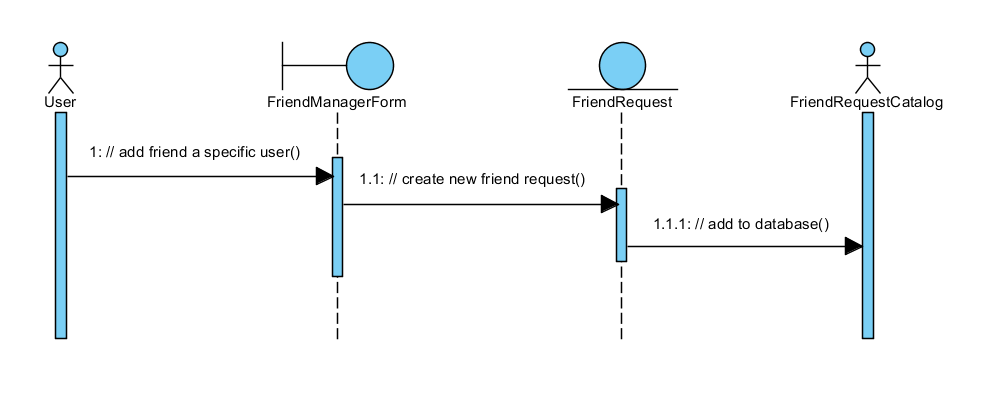


Figure Add Friend - Basic Flow

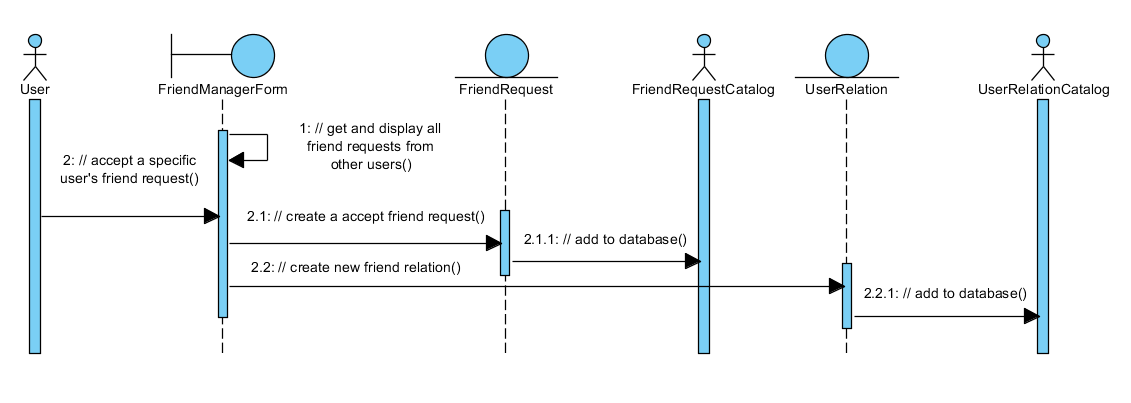


Figure Accept Friend Request - Basic Flow

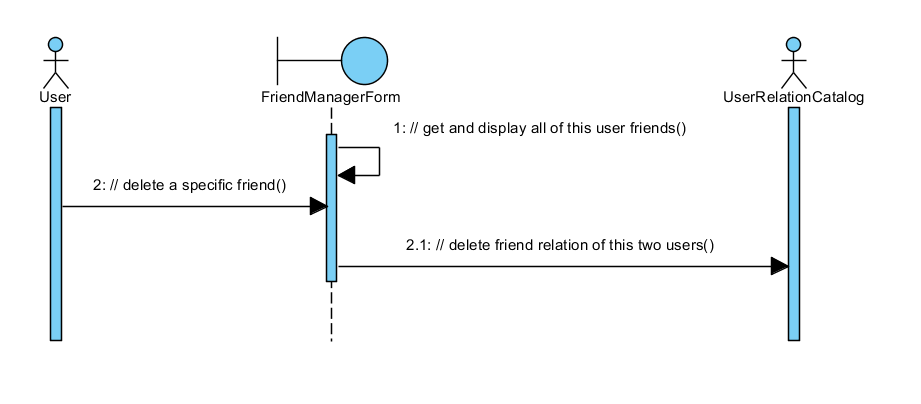


Figure Unfriend – Basic Flow

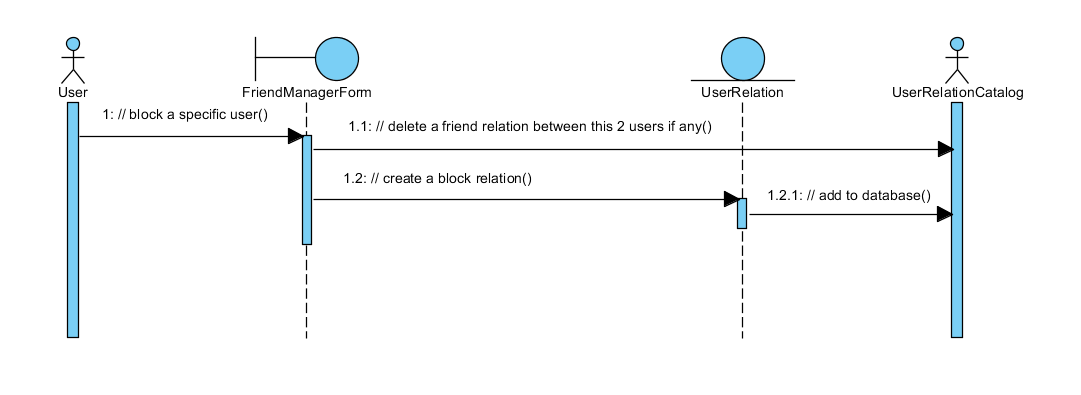


Figure Block a user - Basic Flow

2.2.1.5 Posting

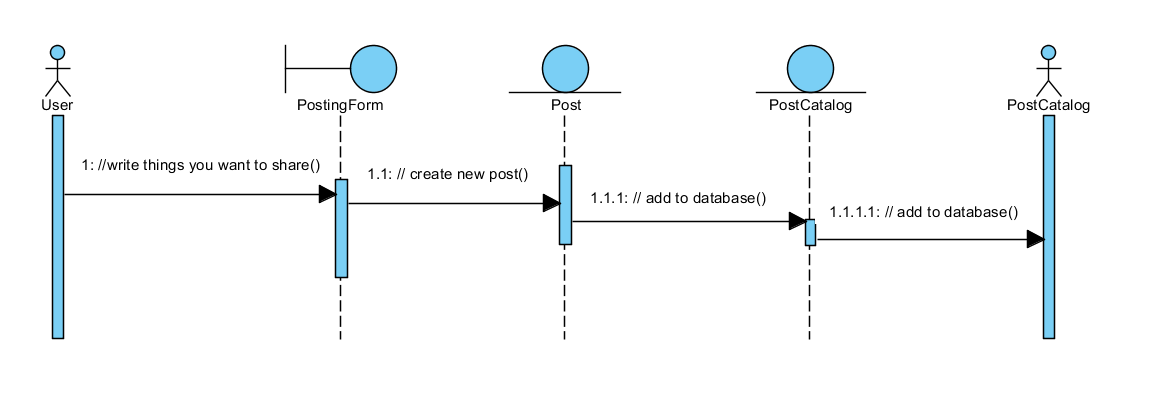


Figure Posting - Basic Flow

2.2.1.6 View Post

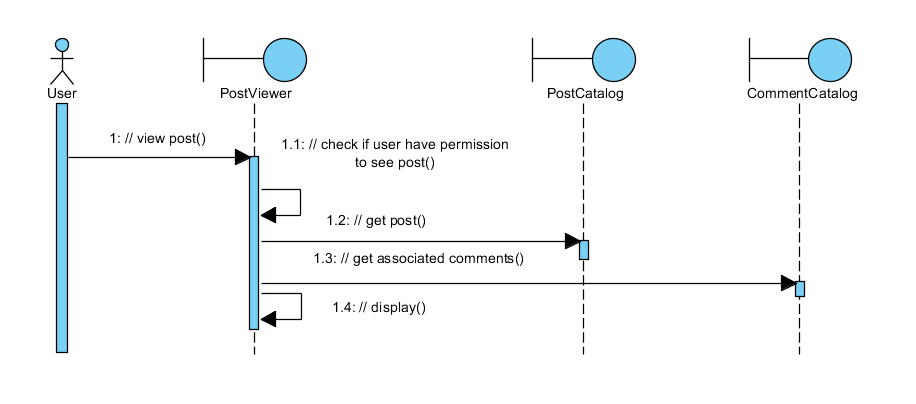


Figure View Post - Basic Flow

2.2.1.7 Edit/Delete Post

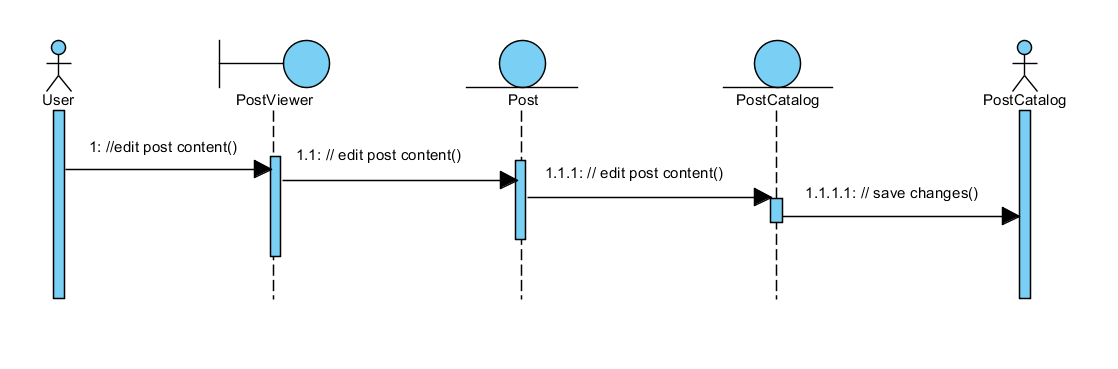


Figure Edit Post - Basic Flow

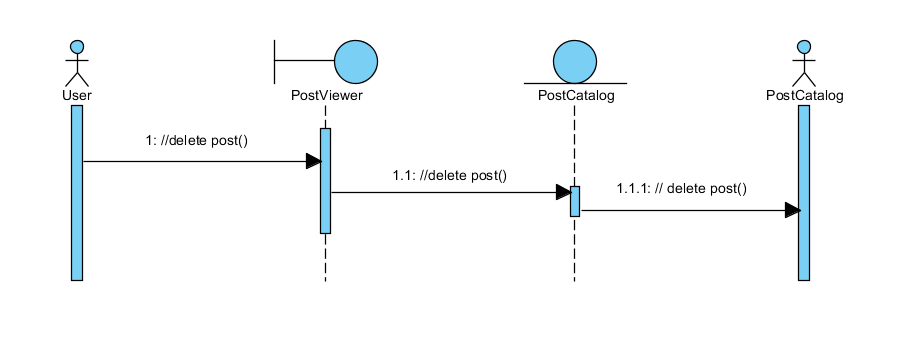


Figure Delete Post - Basic Flow

2.2.1.8 Like Post

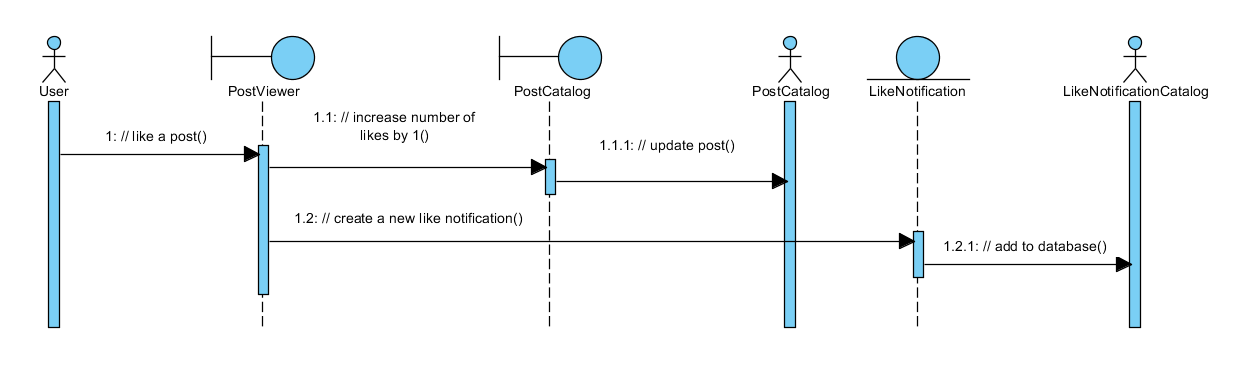


Figure Like Post - Basic Flow

2.2.1.9 Comment

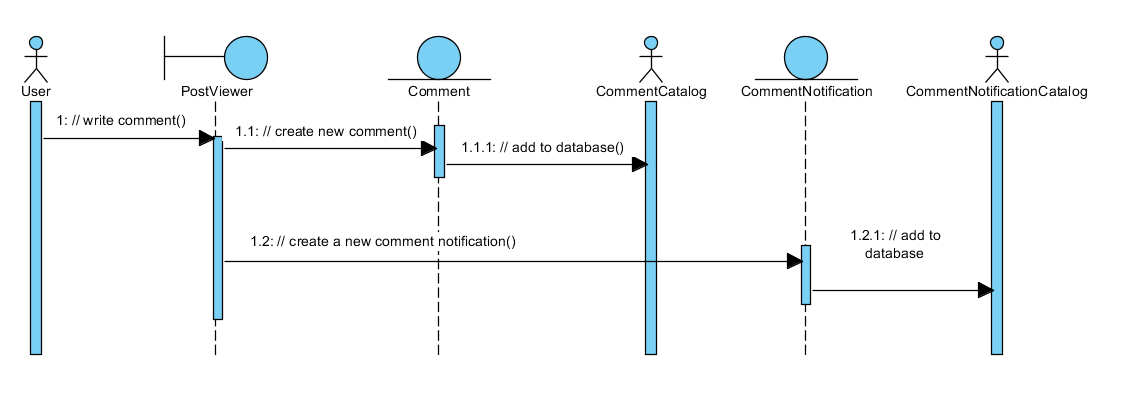


Figure Comment Post - Basic Flow

2.2.1.10 View Notifications

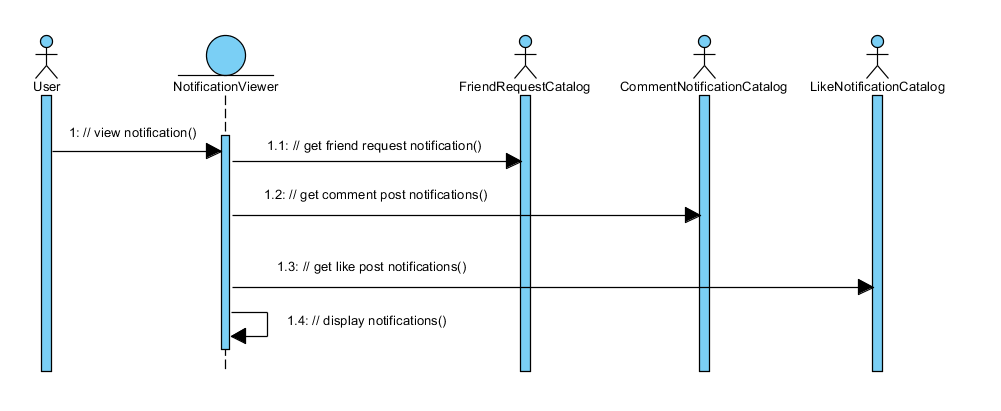


Figure View Notifications - Basic Flow

2.2.1.11 Chatting

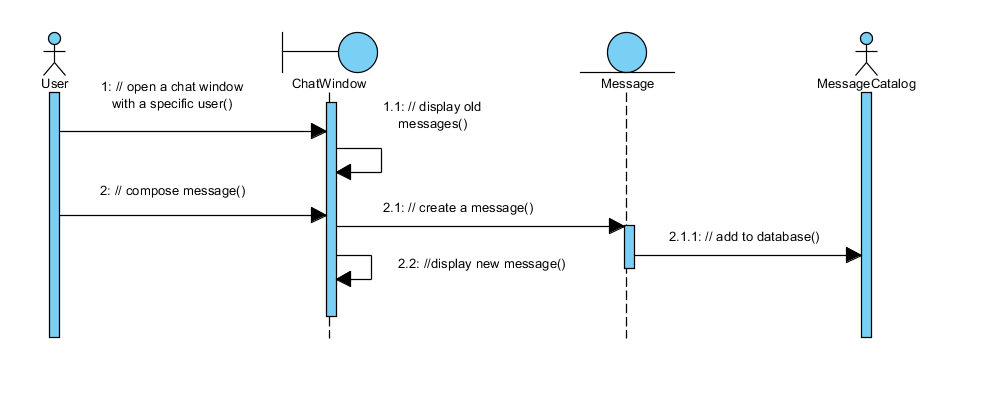


Figure Chatting - Basic Flow

2.2.2 Use-case Realization View of Participating Class (VOPCs)

2.2.2.1 Registration

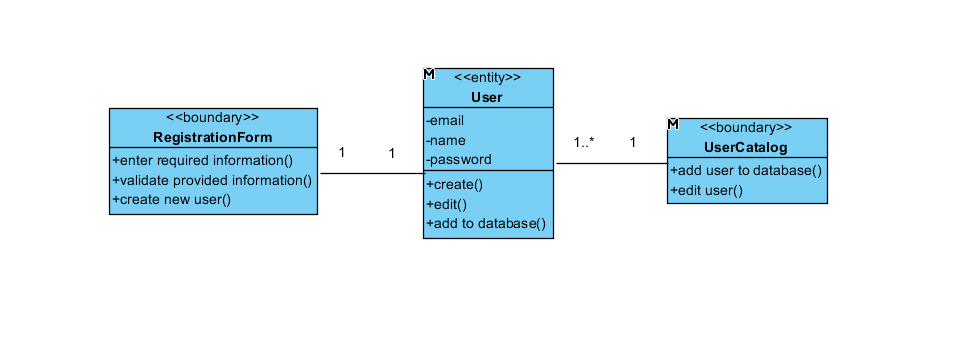


Figure Registration- VOPC

2.2.2.2 Login

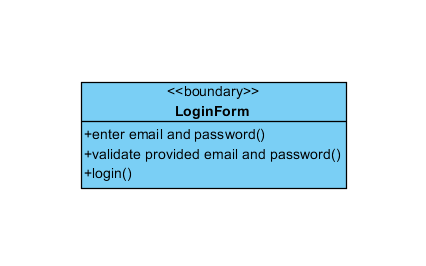


Figure Login - VOPC

2.2.2.3 Update User Profile

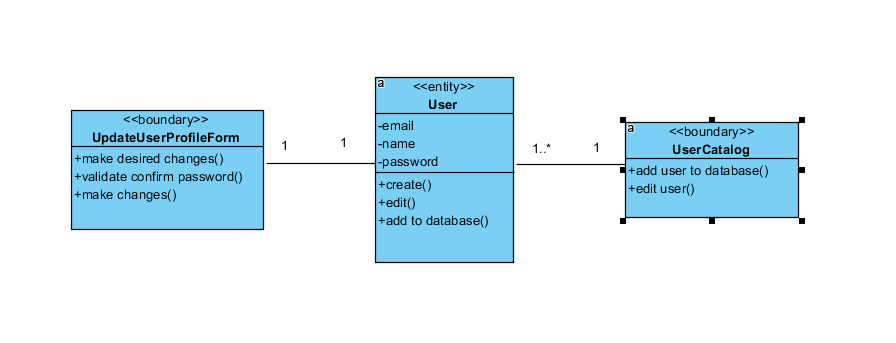


Figure Update User Profile - VOPC

2.2.2.4 Friend Manager

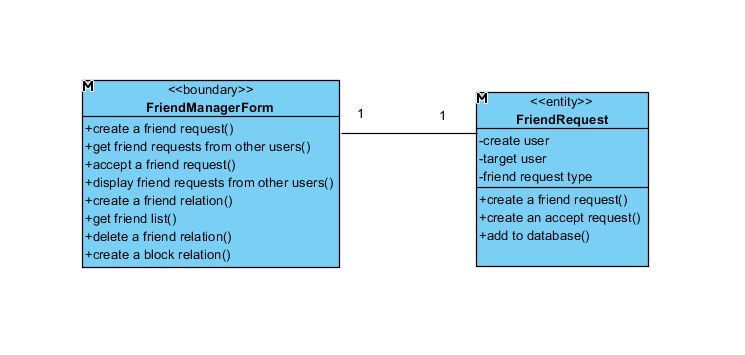


Figure Add Friend - VOPC

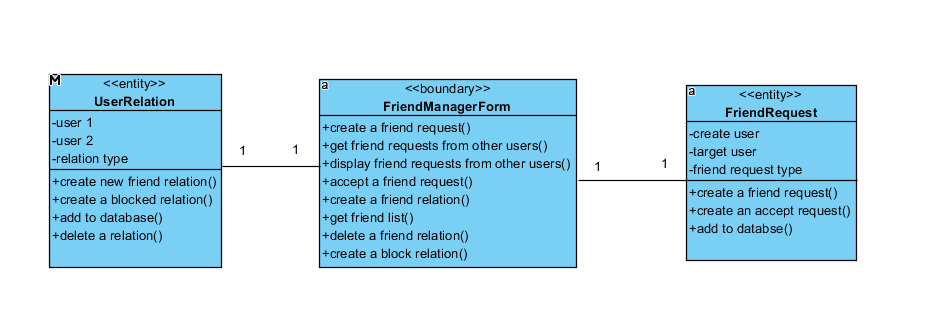


Figure Accept Friend Request - VOPC

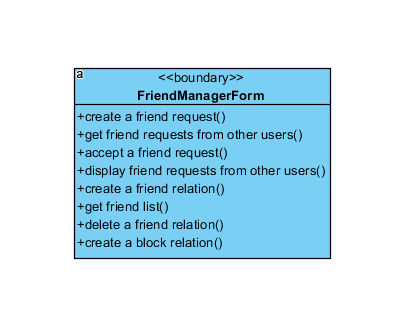


Figure Unfriend - VOPC

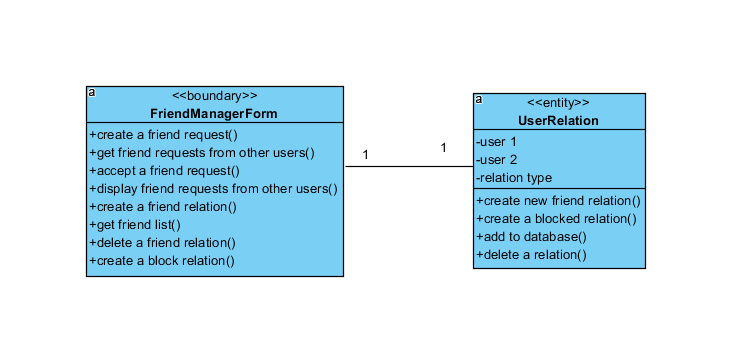


Figure Block a User - VOPC

2.2.2.5 Posting

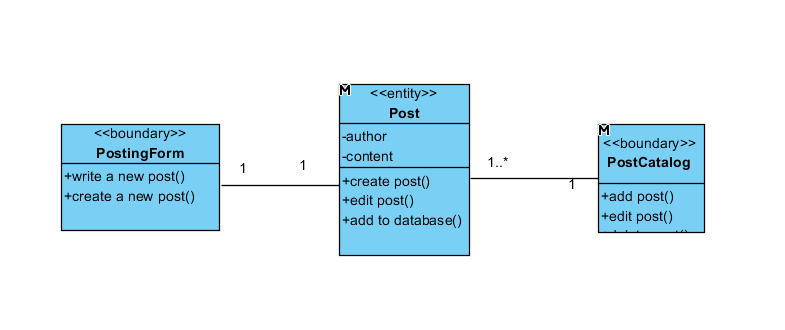


Figure Posting - VOPC

2.2.2.6 View Post

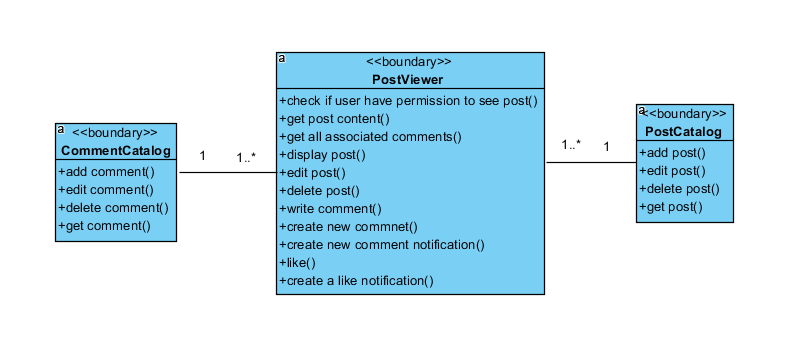


Figure View Post - VOPC

2.2.2.7 Edit/Delete Post

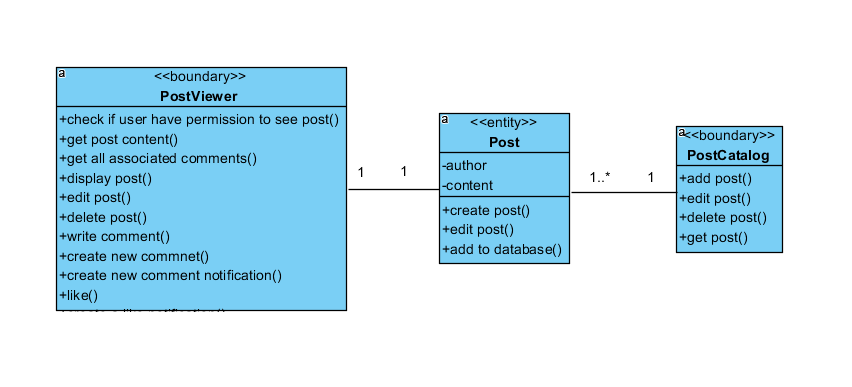


Figure Edit Post - VOPC

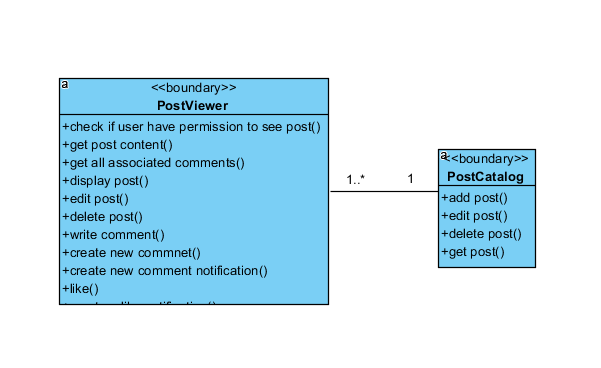


Figure Delete Post - VOPC

2.2.2.8 Like Post

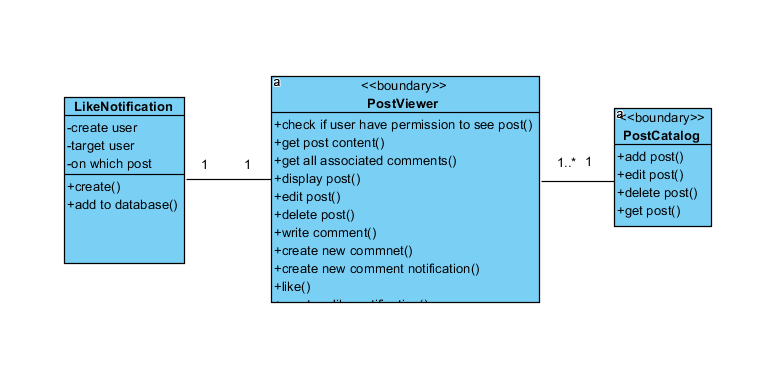


Figure Like Post - VOPC

2.2.2.9 Comment

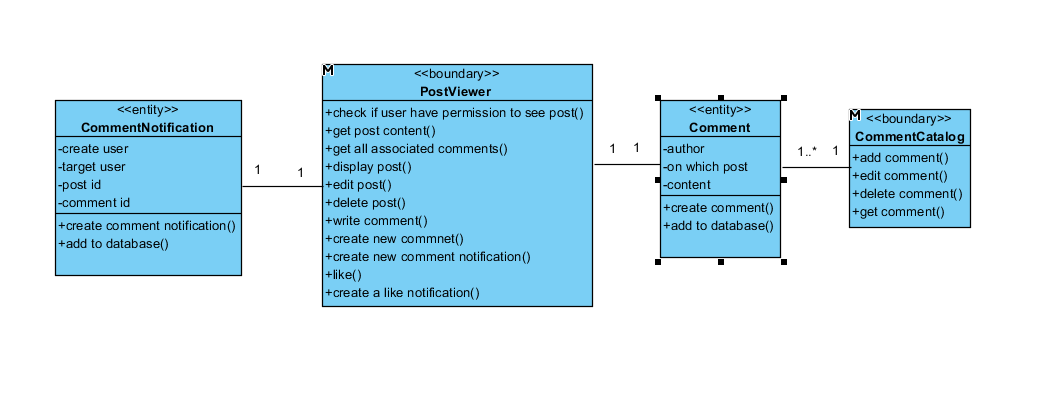


Figure Comment Post - VOPC

2.2.2.10 View Notifications

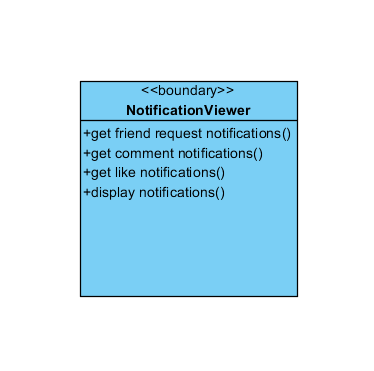


Figure View Notifications - VOPC

2.2.2.11 Chatting

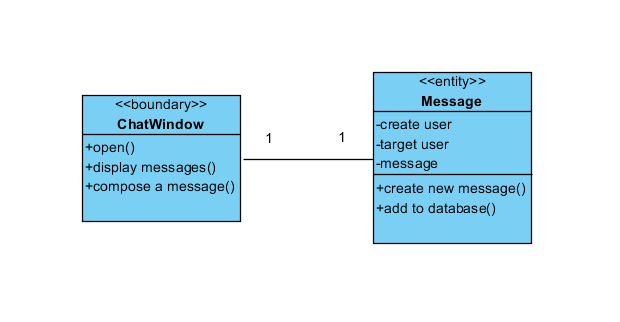


Figure Chatting - VOPC

2.2.3 Analysis-Class-To-Analysis-Mechanism Map

|  |  |
| --- | --- |
| Analysis Class | Analysis Mechanism(s) |
| RegistrationForm |  |
| User |  |
| UserCatalog |  |
| LoginForm |  |
| UpdateUserForm |  |
| FriendManagerForm |  |
| FriendRequest |  |
| UserRelation |  |
| PostingForm |  |
| Post |  |
| PostCatalog |  |
| PostViewer |  |
| CommentCatalog |  |
| LikeNotification |  |
| Comment |  |
| CommentNotification |  |
| NotificationViewer |  |
| ChatWindow |  |
| Message |  |

SECTION 3: Social Network Analysis

3.1 Identify Design Elements

3.1.1 Subsystem Context Diagram

3.1.1.1 Catalog Subsystem

3.1.1.2 Notification Subsystem

3.1.1.3 Chat Subsystem

3.1.2 Analysis-Class-to-Design-Element-Map

3.1.3 Design-Element-to-Owning-Package Map

3.1.4 Architectural Components and Their Dependencies

3.1.5 Packages and Their Dependencies

3.2 Describe the Run-time Architecture

3.3 Describe Distribution

3.4 Use-case Design

3.5 Subsystem Design

3.6 Class Design

3.6.1 Describe each class or interface

3.6.2 Class diagram in total