2018 International Internship Program

<Chatting Program U&I>

Kyubin Kyoung

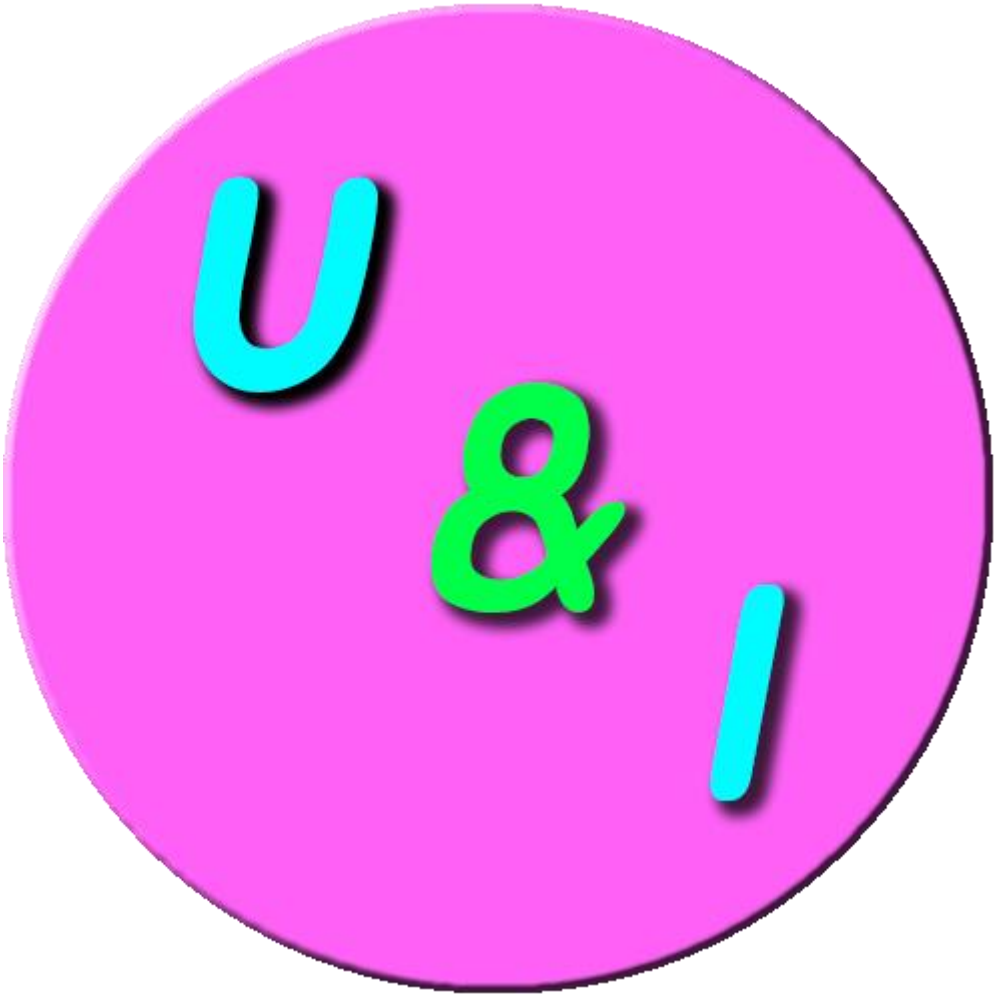


Table of Contents

Table of contents

Ⅰ. Introduction

Ⅱ. Design Overview

Ⅲ. Functionality

Ⅳ. References

Ⅰ. Introduction

a) Background

This is a chat messaging system based on TCP server and android application clients using multithreading, sockets and database. This system offers a chatting service between two people. This application can be installed on a smartphone which has a version above Marshmallow. Through this project I made up my mind to use this application with my girlfriend as a personal chatting program.

b) Scope

I made a server using java with eclipse and client using java, xml with Android studio. There are database of users and messages which are text files. User database contains user profile (name, user id, password, partner id). And message database is consisted of (sender id, receiver id, message, time stamp). When delivering message, users should click refresh button to get a message. There are more functions such as sign up, checking status through image, profile image, different UI between opponent and user.

Ⅱ. Design Overview

a) Technology used

TCP socket programming using AsyncTask. Most of the function is done by AsyncTask which is a simple version of threading. Also multiple users are able to enter the server because of multithreading. When sending data between client and server, I encrypted messages using specific keywords. I saved data using file IO. Also sending data between activities is consisted of intent function. The time information is offered by global calender.

b) Design architecture

There are 4 scenes(we call it activities in android), two object classes(User and message), client handler, file IO class and server.

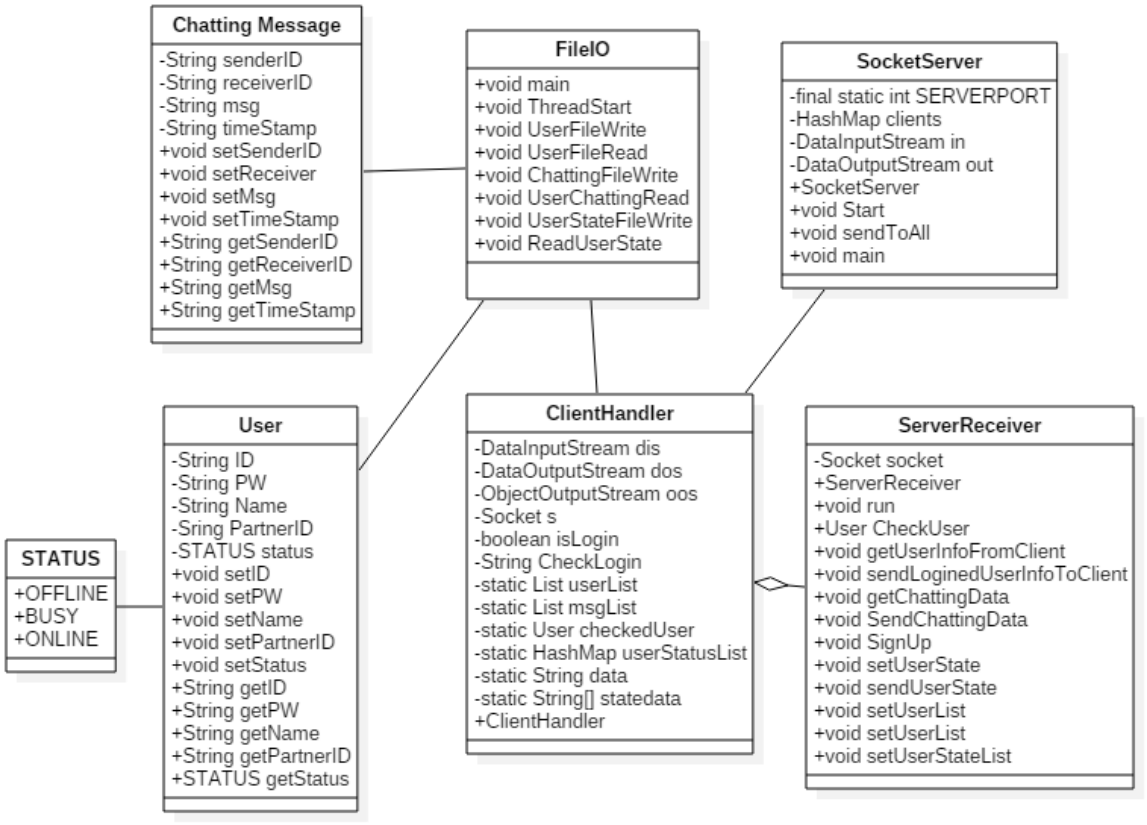


Figure 1, Server Class Diagram

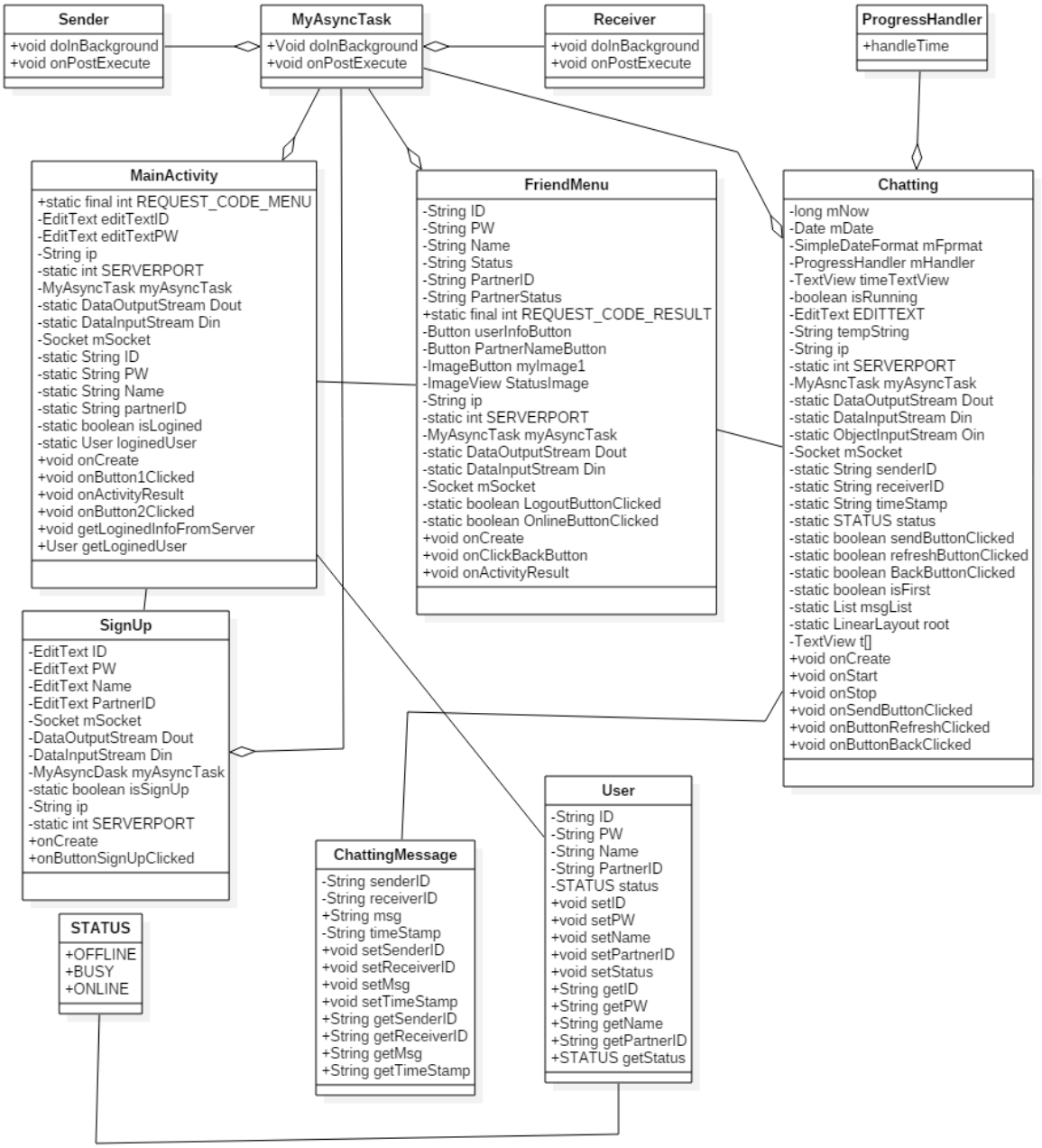
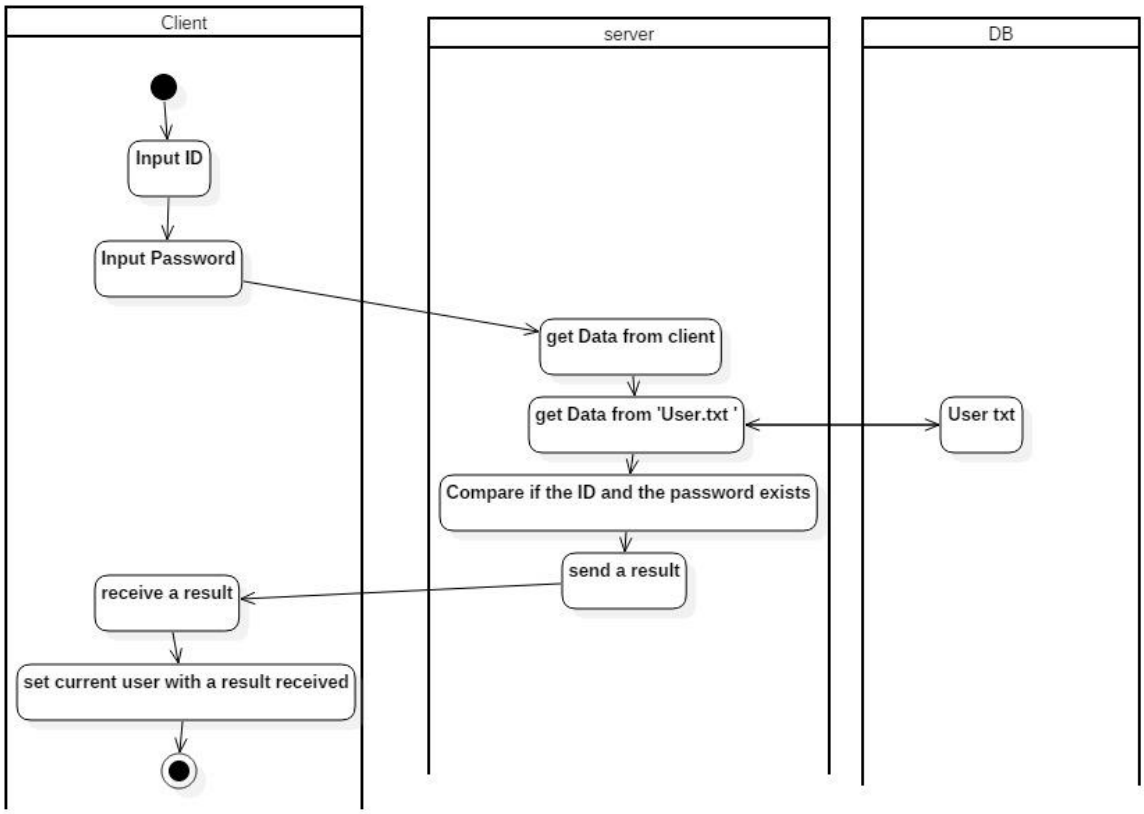
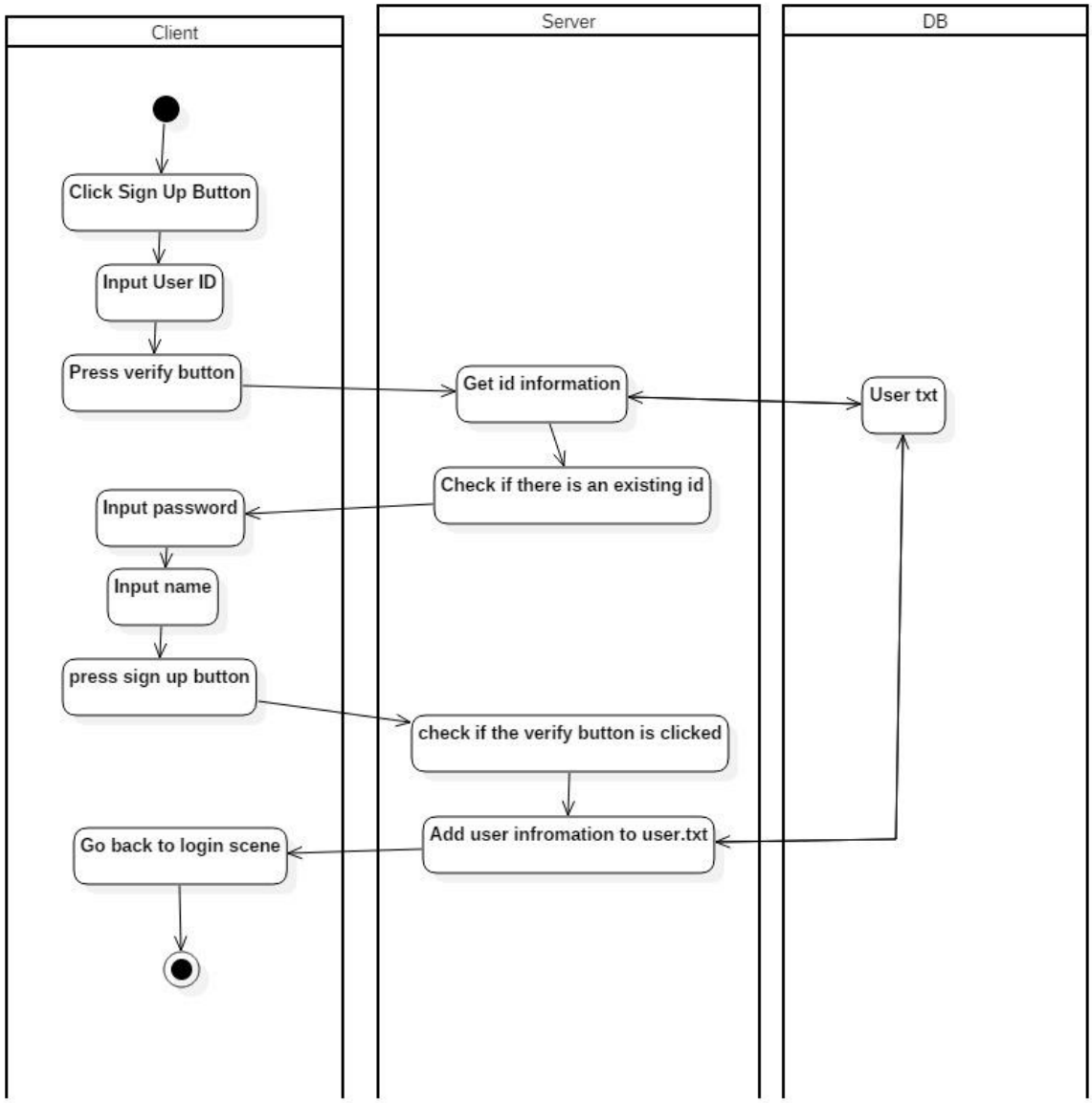


Figure 2. Client class Diagram

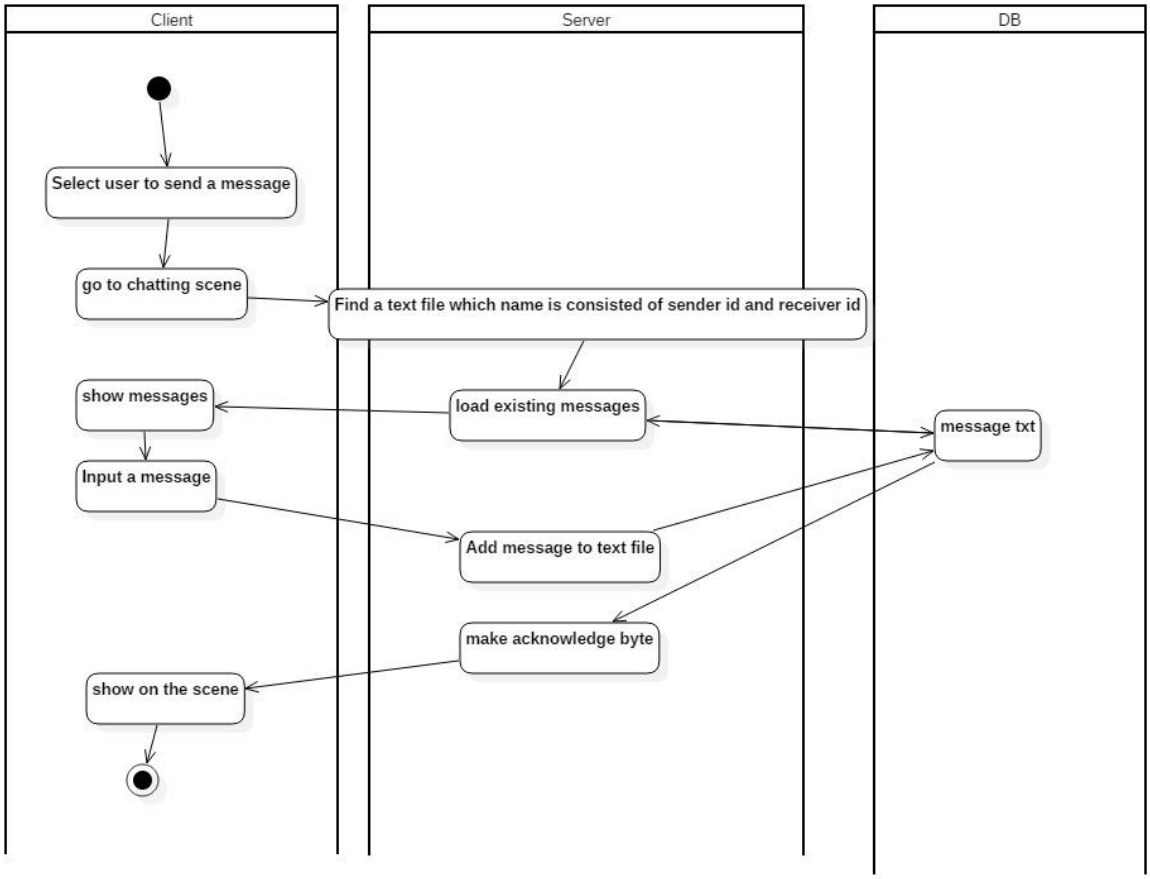
Login activity diagram



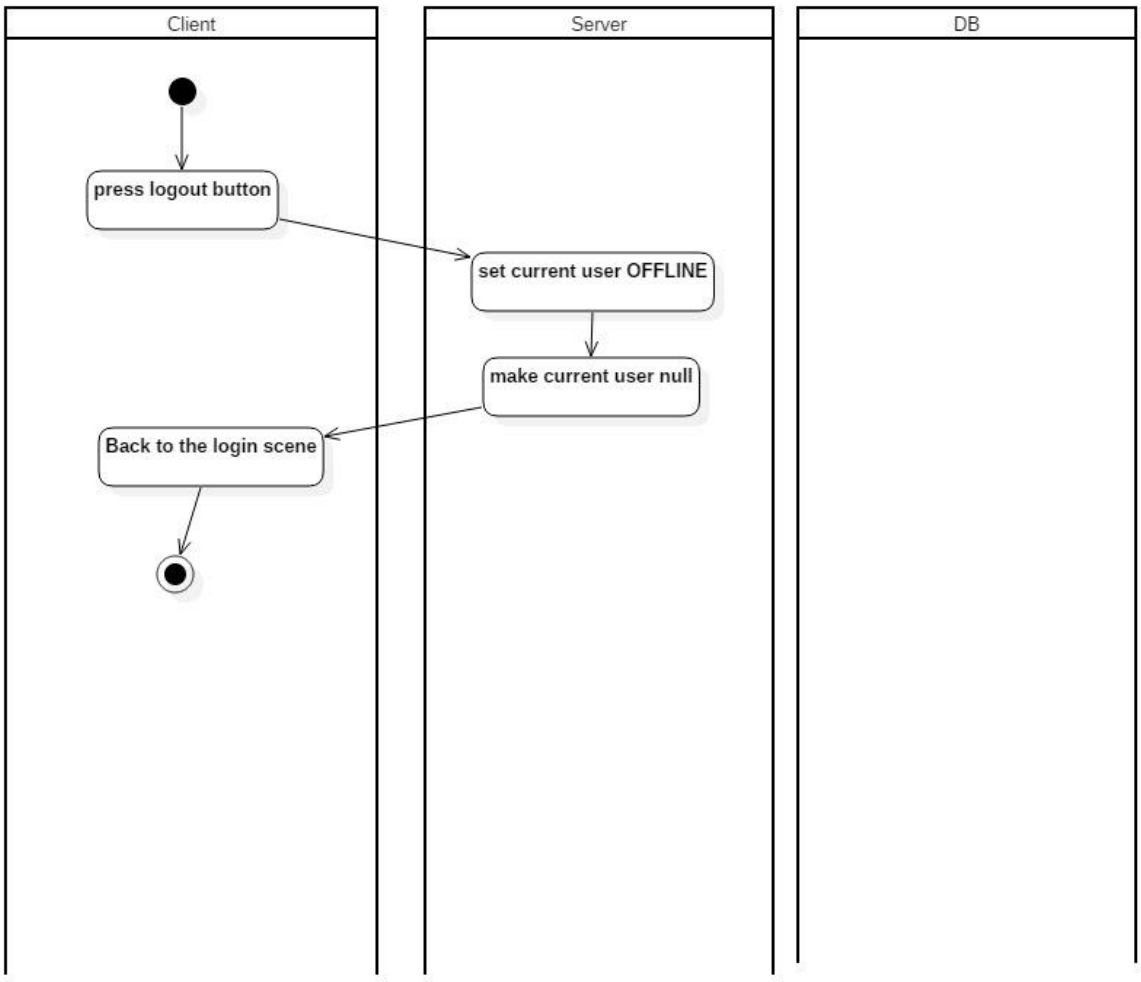
Sign up activity diagram



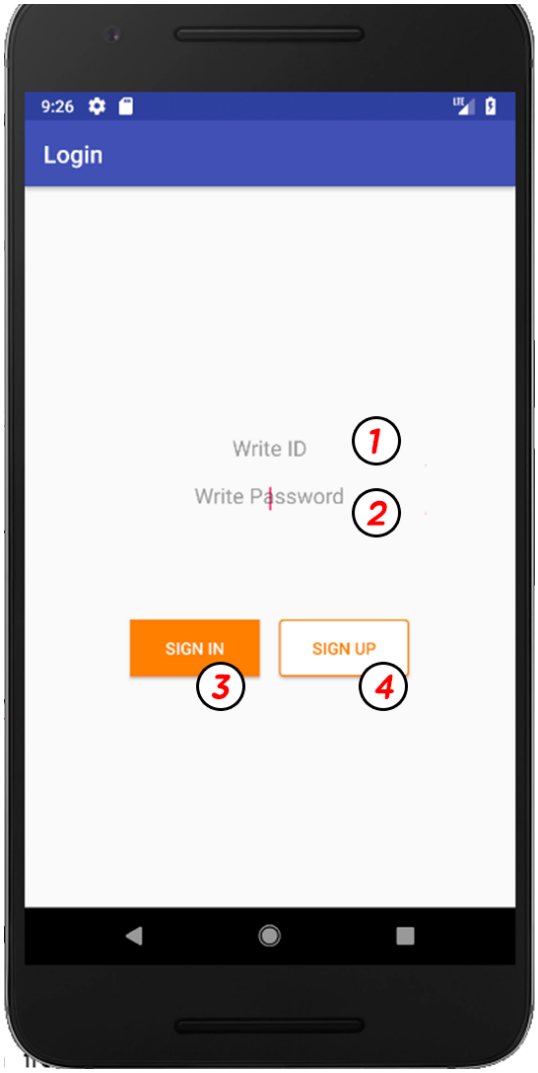
Send Message activity diagram



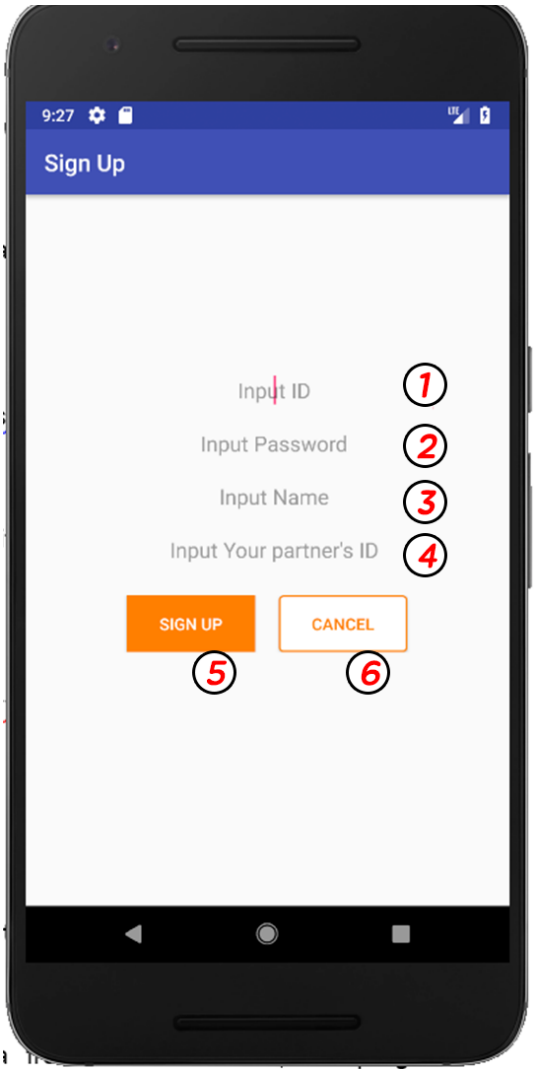
Logout activity diagram



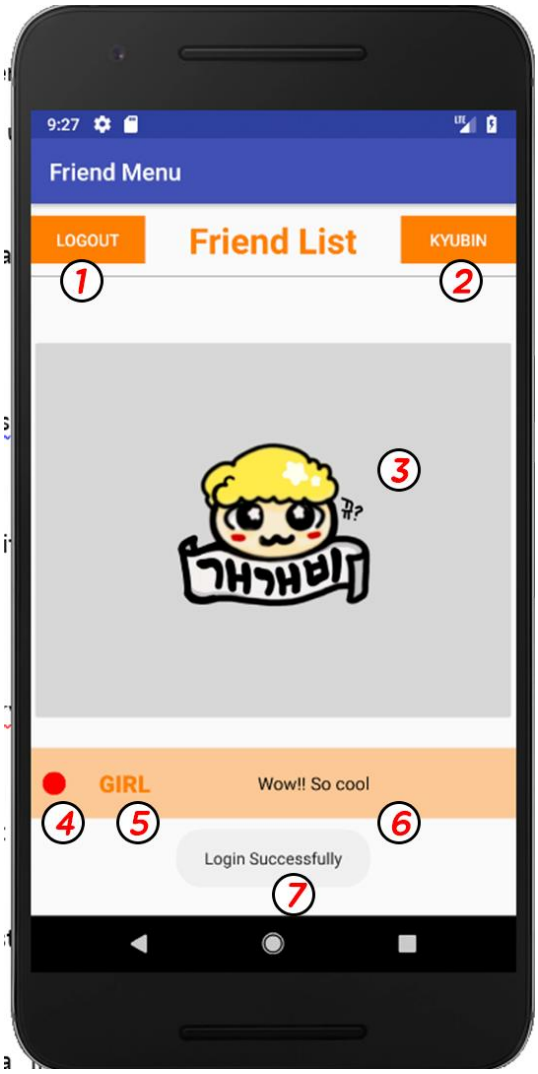
c) UI Flow



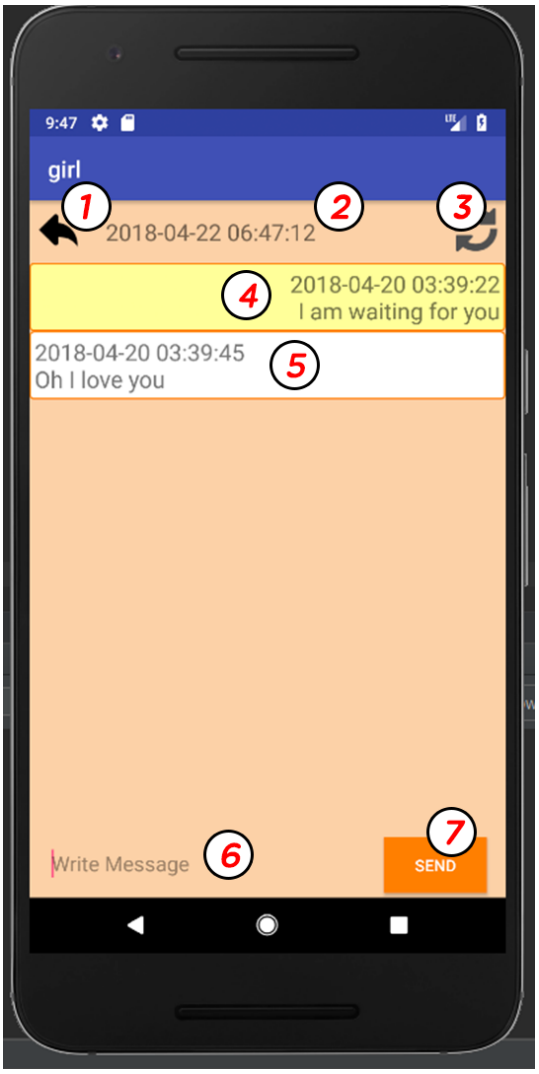
Login Activity	
①	Write ID to login with
②	Write Password to login with
③	When clicking this button, the information above will be sent to the server and authenticate.
④	When clicking this button, the sign up activity will appear. To make a new account.



SignUp Activity	
①	Write ID to make a new account
②	Write Password a new account
③	Write Name a new account
④	Write Partner ID a new account
⑤	When clicking this button, the information above will be sent to the server and authenticate. If the id is new, then a new data will be saved in "User.txt" file.
⑥	Go back to Login Activity



FriendList Activity	
①	Logout. Go back to Login Activity.
②	Information about the user.
③	Image of the user
④	State of the partner. Red means offline, yellow means busy and green means online.
⑤	ID of the partner. When clicking this button you can send a message to partner.
⑥	State message of your partner. Update later.
⑦	A toast message that the login was successful



Chatting Activity	
①	Go back to FriendList Activity.
②	Prints current time using thread
③	Refresh message list. Through this, users can get a new message.
④	Messages that user sent are printed at the right side of the scene.
⑤	Messages that user received are printed at the right side of the scene.
⑥	Write a message to send.
⑦	When clicking this button, the text will be sent to the server and save it into the text file. The text file's name is consisted of sender's ID and receiver's ID.

Ⅲ. Functionality

a) Design consideration

1) Compatibility

A. This application should be available in every android version.

2) Extensibility

A. Making a couple memo board.

B. Making a web page using same server

3) Maintainability

A. As the client UI is based on xml, it is easy to modify UI due to user experience.

4) Fault Tolerance

A. It has self exceptions.

5) Modularity

A. 4 independent activity, lots of classes are well defined as a module...

6) Reusability

A. The code File IO, ServerSocket part can be used anywhere.

7) Robustness

A. As the client doesn't need to have any DB connection, users would think it is not a heavy application.

B. But server does most of the work.

8) Security

A. When sending data from client to server, the program encrypts the data.

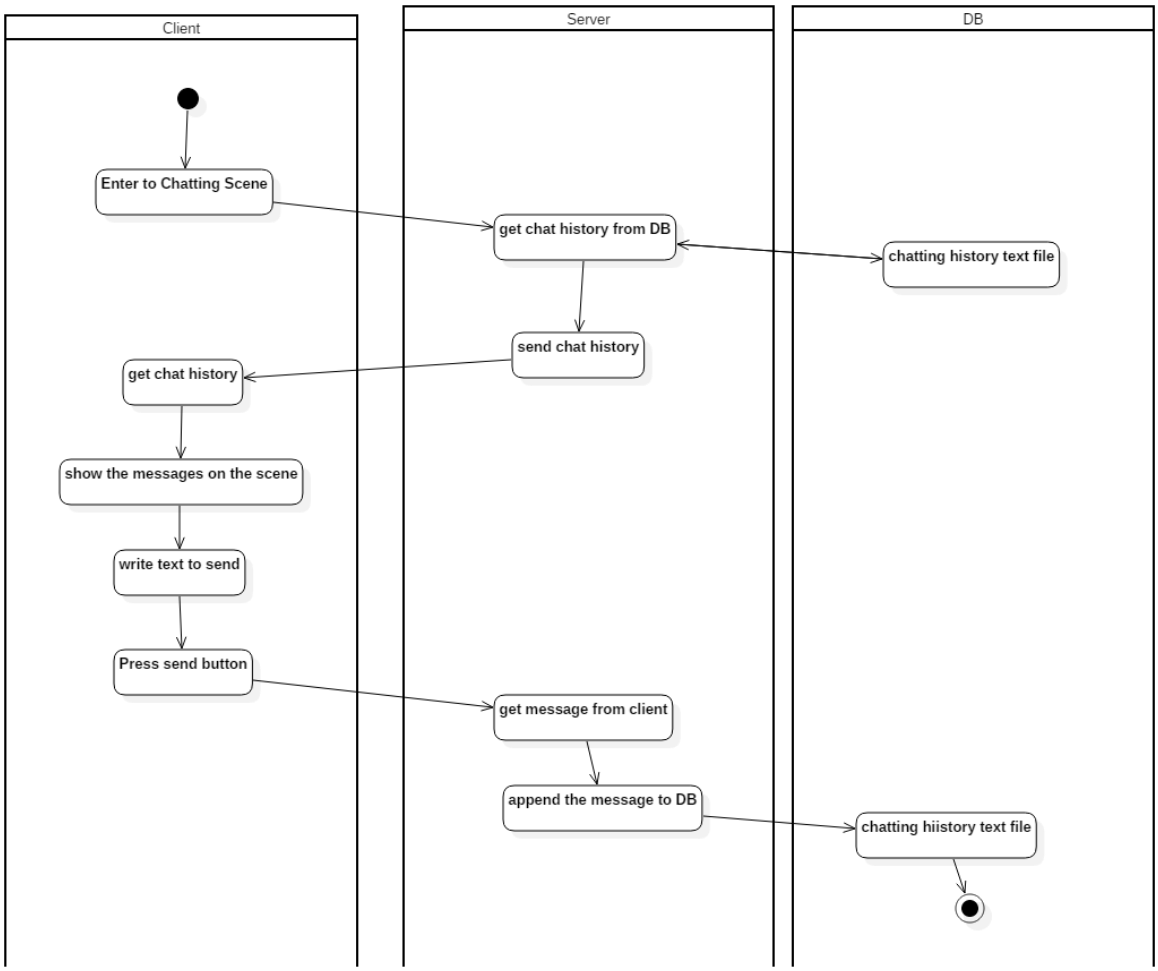
9) Interoperability

A. As there is an independent server and the server is handling most of the process, other kinds of clients can be developed and interact each other.

b) Constraints

- This application is only available in Android OS.
- The Android version must be above Marshmallow.
- To get a message from the opponent, refresh button needs to be clicked.
- Server should be executed in Eclipse.
- To make a server for a test, the ip address in the code should be modified.
- The implementation language should be JAVA.

c) Message Flows



IV. References

http://androidsrc.net/android-client-server-using-sockets-client-implementation/

Book: Do it Android App programming, Doctor Jung

https://stackoverflow.com/questions/4074808/java-socket-ioexception-permission-denied

http://server-engineer.tistory.com/207

https://developer.android.com/design/material/index.html?hl=ko