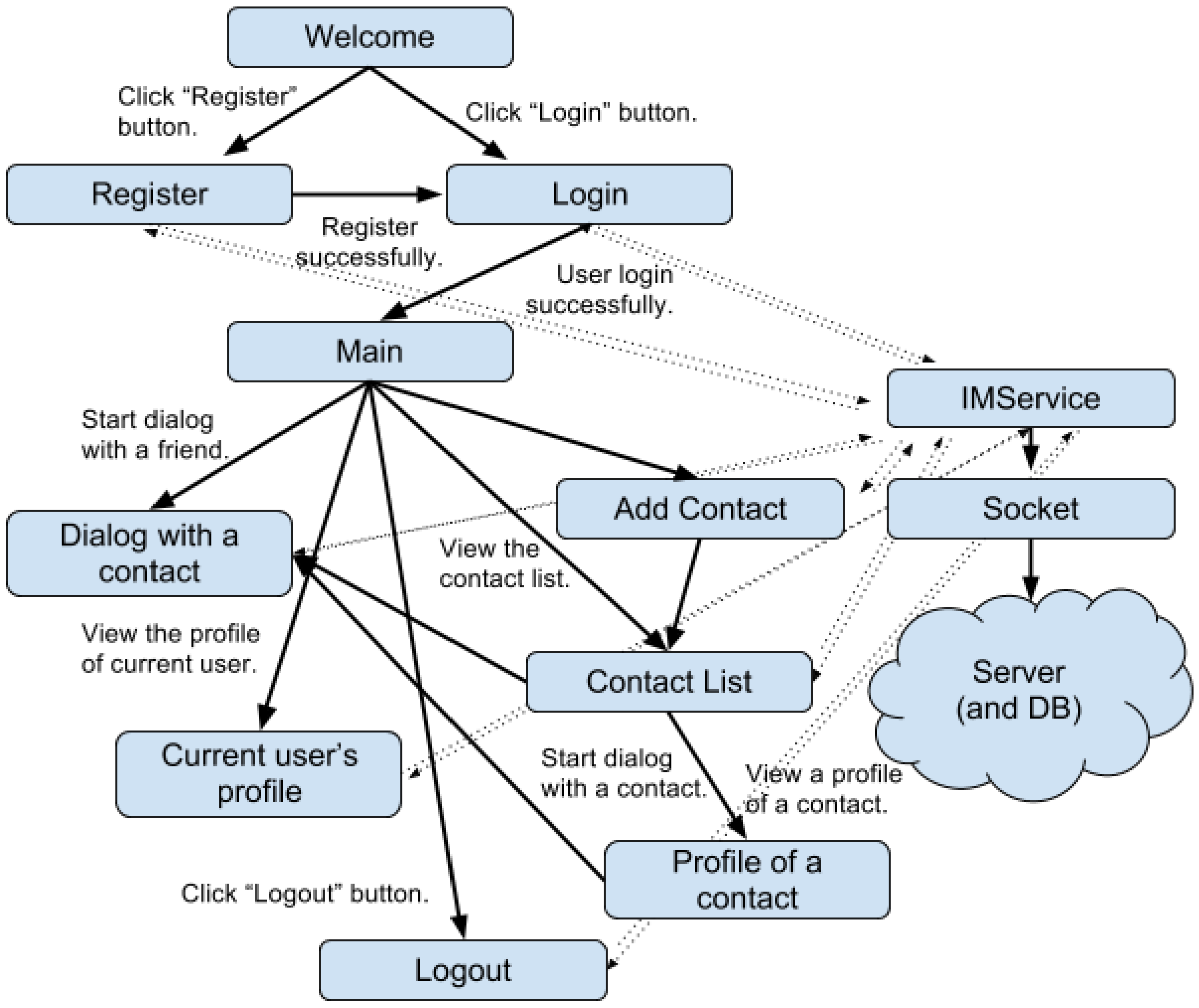
Incremental and Regression Testing

Team5

Classification of Components

Test form:

The form of incremental testing we are using is bottom up testing. Because driver is needed for bottom up testing, and driver modules are easier to produce than stub modules.



Component - 1 Welcome

The welcome page is showed once this app is launched before users login. It allows users to choose “Register” (to register as new user) or “Login” (to login as an existing user).

Input: Click a specific button on the welcome page.

Output: The responding page is shown up.

Component - 2 Login

Once the user clicked the login button on the welcome screen, the login page will appear and it allows user to choose “OK” or “Cancel” and input username and password.

Input: username (String), password (String), specific button on login page.

Output:

1. Correct combination of username and password login success,redirect to main page..
2. Incorrect/invalid username or password, login fail.
3. Click on “Cancel”, program exits.

Component - 3 Register

Once the user clicked the register button on the welcome screen, the login page will appear and it allows user to choose “OK” or “Cancel” and input username and password.

Input: username (String), password (String), specific button on Register page.

Output:

1. Valid of username and password, click OK, redirect to main page
2. Invalid username or password, register fail.
3. Click on “Cancel”, program exits.

Component - 4 Main

Once the user is registered or clicked the OK button on the login screen, the Main page will appear and it allow user to click on specific dialog to enter the dialog page. The user can also choose to view his or her profile by clicking the “Profile” tab or browse the contact list by clicking the “Contact” tab. User can also choose to logout by clicking the logout button.

Input: Specific button/tabs on main page.

Output:

1. Redirect to Profile page by clicking the “Profile” tab.
2. Redirect to Contact page by clicking the “Contact” tab.
3. Redirect to Dialog page by click on specific dialog.
4. Redirect to Logout page clicking the logout button.

Component - 5 Contact

Once the user click the contact tab from the main page, the contact page will appear and it allows user to choose to view friend’s profile by clicking on specific friend’s tab. User are also allowed to click on “Add” button to add friends.

Input: specific button/tabs on Contact page.

Output:

1. Redirect to Add Contact page by clicking the “Add” button
2. Redirect to Profile page by clicking specific friend’s tab
3. Redirect to Main page by clicking Main tab

Component - 6 Add Contact

Add contact page will appear if user click on “Add” button on contact page. It will allow users to search friends that they want to add to their contact list by searching their username or email. Once the user select the friend they wanted to add, he or she can click on OK button.

Input: username/email (String), specific button on add contact page.

Output:

1. Enter valid username or email, search friend success.
2. Enter invalid username or email, search friend fail.
3. Click OK button after adding friend, redirect to contact page.
4. Click on “back”,redirect to contact page.

.

Component - 7 User Profile

User profile page is called when user click on “Profile” button on main page. User can change his or her personal information in this page. Or go back to the main page by clicking the back button.

Input: profile information (String), specific button on User Profile page.

Output:

1. Change valid profile information and click OK, change accepted.
2. Change invalid profile information and click OK, change failed.
3. Redirect to Main page by clicking Back button.

Component - 8 Profile

Once the user clicked the specific friends tab on Contact list, the profile page will appear and it allows user to choose “Send Message” or “Cancel” to go to dialog page or go back to Contact page

Input: specific button on Profile page.

Output:

1. Click “Send Message” redirect to Dialog page.
2. Click on “Back”,redirect to contact page.

Component - 9 Dialog

User can send message to friends on this page. Also a back button is provided so that user can go back to the main page

Input: Message (String), specific button on dialog page.

Output:

1. Valid message send after clicking OK, message sent
2. Invalid message send after clicking OK, message sending failed
3. Click on “Back”, redirect to main page.

Component - 10 Logout

User can logout on this page or click Back to go back to the main page

Input: specific button on logout page.

Output:

1. Click on “Back”, redirect to main page.
2. Click on “Logout”, program exits.

Component - 11 Server (and Database)

The server is listening for any command. If there is a command received, it will call responding method to handle it. Besides, the server will store and fetch data from database.

Input: A string in a format as, COMMAND VARIABLES (e.g. REGISTER USERNAME PASSWORD).

Output:

1. For supported command, a string in defined format will be return (e.g. for "REGISTER" command, it should return "REGISTER TRUE" indicating register successes or return "FALSE" indicating register fails).
2. For unsupported command, nothing should be return.

* All supported command is listed below.

Input	Success Output	Failure Output
REGISTER YourUsername YourPassword	REGISTER TRUE	REGISTER FALSE (Invalid username and password)
LOGIN YourUserName YourPassword	LOGIN TRUE	LOGIN FALSE (Invalid username/password)
GETINFO UserID	INFO UserId InfoLIst (Information of user with specific id.)	GETINFO UserID FALSE (User not exist.)
STARTDIALOG CurrentUserID UserID	CONNECTED CurrentUserID UserID	STARTDIALOG FALSE (Networking problem)
SEND FROM CurrentUserID TO UserID Message (space is allowed in Message)	SENT (And then the server connected to the user with UserID and send a command as "SEND Message FROM CurrentUserID")	SEND FALSE (Message is too large or Networking problem)
ADD CurrentUserID UserID	ADD TRUE	ADD FALSE (User not exist)

LOGOUT	LOGOUT TRUE	N/A

Incremental Testing

We are using bottom-up incremental testing here.

product	Server (and Database)		
Date	09/13/2013		
Defect#	Description	Severity	How corrected
1	For messaging containing white space, it is cut into several string	1	Changed the command format. Put the message as the last variable. While retrieving the last variable, the server just get the substring from the end of the second last variable to the end of the whole string, which makes it possible to support white spaces in message.
2	Unhandled exception when client side passing parameters do not meet the required format of the	2	Implemented condition statements to control passed-in

	server side functions.		parameters from client side.

product	Server (and Database) + Socket		
Date	09/13/2013		
Defect#	Description	Severity	How corrected
1	Get an IO Exception while trying to connect.	2	Catched the IO Exception and print error information into the log.
2	Socket and Server was not able to be connected, therefore there wouldn’t be any communication between them.	1	We modified the code to ensure their connection.

product	Server (and Database) + Socket + IMService		
Date	09/14/2013		
Defect#	Description	Severity	How corrected
1	Some Socket functions couldn’t be properly called by IMService class, since they	3	We simply modified those functions to be public instead of

	were set to be private functions		private.
2	Unhandled exception is thrown when basic error or warning occurs in XML parser.	1	Catch the exception and prints the error information.

product	Server (and Database) + Socket + IMService + Welcome		
Date	09/14/2013		
Defect#	Description	Severity	How corrected
1	The welcome page is not shown up while trying to connect to the server.	1	Show the welcome page at first and try to establish the connection between server while needed.
2	Everytime already login user launches this application, the user needs to login again.	2	Add a timer and login status on the server side. The login status will maintain as true in a specific time. Once the login status is true, the user does not need to login again.

Product	Server (and Database) + Socket + IMService + Login		
Date	09/15/2013		
Defect#	Description	Severity	How Corrected
1	It happens when a user uses a password that is slightly different from the correct password and manages to login.	2	We changed the way password is stored in the server.
2	some class variables of IMService class were set to be private and not accessible by Login class.	3	Simply modified private to public.

product	Server (and Database) + Socket + IMService + Register		
Date	09/16/2013		
Defect#	Description	Severity	How corrected
1	User was able to register two accounts using the same email address.	1	we added handler in the Server to ensure this won't happen.
2	User could be able to register offline, though all the information hasn't been	2	We added handler in Server so that when user register, it will

	saved to database for later use.		check network connection.

Product	Server (and Database) + Socket + IMService + AddFriend		
Date	09/17/2013		
Defect#	Description	Severity	How Corrected
1	added three friends, two showed up in friend list, which was due to improper handle of server, because it missed the last item in the friend's list.	3	we made modification of the server so that it will handle the list correctly.
2	Added the same person in friend list twice, it would appear in the friend list twice. which was because we didn't let server check the database to find if there were the same data already existing.	2	We added handler that deal with this situation.

product	Server (and Database) + Socket + IMService + Dialog with a Contact		
Date	09/18/2013		
Defect#	Description	Severity	How corrected

1	when a message contains some special tokens, such as multiple semicolons would not be properly processed by the server since server split the message string by semicolons, resulting in missing some characters.	2	we added extra code in the server part to handle this multiple semicolons case
2	When internet connection was down, the connection between server and database was also completely down, even if the connection was restored later.	1	we added code to the server part that periodically check internet connection, if connection is down, it will try to reconnect to database.
3	If an empty message was sent from one user to his or her friend, he or she may not receive the message, but the blank message would still be saved in the database.	3	we added handler to server so that it will not pass the message to database

Product	Server (and Database) + Socket + IMService + UnApprovedFriendList		
Date	09/19/2013		
Defect#	Description	Severity	How Corrected
1	It happens when two friends added each other but both appear in the UnapprovedFriendList	2	We updated how the UnapprovedFriendList is updated according to the

			server.
2	Once a friend is put in unapprovedFriendList, it does not go out of this list even after the user adds this friend.	3	We added code to update the FriendList and UnapprovedFriendList.

Product	Server (and Database) + Socket + IMService + UnapprovedFriendList		
Date	09/20/2013		
Defect#	Description	Severity	How Corrected
1	User was not able to delete an unapproved friend in the list.	2	We added a function in the server to handle it.
2	after user deleted a people in unapproved friend list, that guy reappeared in next login.	2	we add a global variable so that it can recorded if a people has been removed from list or not.

regression testing

Name	Regression Test login module - 01
Included components	Login & Server & database
Input	password input as “asdfg”
Expected Output	promt out “password incorrect”
Description	It happens when a user uses a password that is slightly different from the correct password e.g “asdfh”, and it should be an incorrect password and fail to login
Severity	2

Name	Regression Test friend list - 01
Included components	AddFriend & UnapprovedFriendList
Input	two clients add friends each other
Expected Output	appear in the approved friend list in other’s client
Description	When two friends added each other, they will appear in other’s approved friend list after updating Unapproved Friend List accordin to the server.
Severity	2

Name	Regression Test friendList - 02
Included components	server, friendlist
Input	Client Device A deletes friend Device B
Expected Output	Server removes Device B from Device A in friendList Table and update it own friend list
Description	Client A delete his friend in the friend list table and avoid error
Severity	1

Name	Regression Test client dialog - 01
Included components	server, dialog interface
Input	Click on “Send” button trying to send an non-empty text message that is no longer than 255 characters in the dialog window
Expected Output	Dialog box displayed the message you just sent
Description	Client A input a valid text in the dialog and make sure it send successfully, avoid error
Severity	3

Name	Regression Test messageHandler - 01
Included components	server, friendlist
Input	Server receives a message “hello” record from device A to device B on the condition that A is not in B’s friendList

Expected Output	device B is not expected to receive the message “hello”
Description	Client B should not receive any message from the Client which is not in its friendlist
Severity	1

Name	Regression Test Deleting Friend 01
Included components	server, friendlist, client
Input	Delete an existing friend in the friend list from Device
Expected Output	Update friendList in the database, update friend list in device
Description	The client should be successfully deleting the friend when action is done.
Severity	2

Name	Regression Test Client Register 01
Included components	Register, Password
Input	Enter a password with length of 17 and click “OK” button
Expected Output	Popup box appears saying “password length should be between 5 and 16”
Description	The password is stored in the database between the length of 5 anc 16 characters. Hence, password validity needs to be checked.
Severity	3

Name	Regression Test Client Device Request Profile 01
Included components	Client, Server, FriendList
Input	Client sends request to update profile
Expected Output	Update profile table in database, update profile on client device
Description	Server needs to be able to accept request from client to updata profile.
Severity	1