6.005 Project #2 Team Contract

Olutayo Falase, Jonathan Trevino, Kaivan Wadia

GOALS:

- Mission statement: to develop a working IM client and server using Java
- Each teach member will make their best effort to contribute equally to the success of the project
- Anticipated obstacles include technical mishaps, time restrictions due to classes, and scheduling conflicts
- All members agree to work towards the goal of achieving an A on this project
- If we have a suitable design and code base, we will try to go beyond the specification and compete for an award

MEETING NORMS:

- Meetings will be scheduled by vote via email according to the most preferable and reasonable location and time
- We anticipate that we will be meeting at least twice a week
- Meetings will be used to assign tasks and discuss design specifications.
- In-class time will be used to meet with the TAs and work on implementation of the project
- Food and drinks are allowed during the meeting so long as the team member stays productive and on task
- Formal minutes of the meeting will not be recorded, but a general outline of ideas were presented will be stored on a Google-doc which will be shared amongst all team members.

WORK NORMS

- Work will be split as evenly as possible without forcefully splitting apart tasks that logically follow one another
- We estimate that each person will need to put in 5 to 6 hours per week for the completion of the project
- Work will be divided based on both skill and preference
- Deadlines shall be set in advance and chiefly revolve around the milestones as set forth by the 6.005 staff

DECISION MAKING:

- Decisions shall be made in consultation with all the team members to obtain a unanimous vote.

TEAM MEMBERS:

Olutayo Falase (tfalase@mit.edu)
Jonathan Trevino (trevino@mit.edu)
Kaivan Wadia (kwadia@mit.edu)

CONVERSATION DESIGN:

Conversation like a phone call with conference call functionality (in the future)

CLIENT SIDE :

```
ClientModel Class:
        Instance Variables :
                String serverIP
                int serverPort
                String username
                ArrayList<String usernames> activeUsers
        -----------------------------------------
        Public Methods :
                newChat(String otherUsername)
                addNewActiveUser(String username)
                otherUserLogsOut(String username)
                terminateMyConnection()

ChatModel Class:
        Instance Variables:
                HashMap<String username, String message> conversationHistory
                int chatID
        -----------------------------------------
        Public Methods:
                clientToServerChatMessage(String message)
                addUserToChat(String otherUsername)
                closeChat()

LoginGUI Class:
        Display for login information

ClientGUI Class:
        Displays all the people online

ChatGUI Class:
        Displays a chat between two people
```

SERVER SIDE:

```
ServerModel Class:
        Instance Variables:
                HashMap<String username, Client client> clientMap
        -----------------------------------------
        Public Methods:
                acceptClient()
                        creates a thread for a single user
                        Tells all users someone new is online

                newClientAccepted()
                        returns a list of active usernames

                inviteToChat()

                createChat(String user1, String user2)
```

serverToClientMessage()

endClientConnection(String username)

Tells all users someone has logged off

ClientHandler Class:

Handles a single client

Public Methods:

newChatHandler(String username1, String username2)

closeClient()

ChatHandler Class

Instance Variables:

int chatID

Public Methods:

endSingleUserConnection(String username)

closeChat()

CLIENT/SERVER PROTOCOL:

Client To Server:

CLIENTMESSAGE :== (SENDMESSAGE | ENDCHAT | TERMCON | ADDUSER) NEWLINE

SENDMESSAGE :== “message” SPACE CHATID SPACE MESSAGE

ENDCHAT :== “endchat” SPACE CHATID

TERMCON :== “terminateconnection”

ADDUSER :== “adduser” SPACE CHATID USERNAME

NEWLINE :== “\n”

SPACE :== “ “

MESSAGE :== .+

CHATID = [0-9]+

USERNAME :== [a-z0-9]+

Server To Client:

SERVERMESSAGE :== (RELAYMESSAGE | NEWUSER | USERTERM | USEREXITED | CONNECTIONACCEPTED) NEWLINE

RELAYMESSAGE :== “relaymessage” SPACE CHATID SPACE MESSAGE

NEWUSER :== “newuser” SPACE USERNAME

CONNECTEDACCEPTED :== “connectionaccepted”

USERTERM :== “userterminated” SPACE USERNAME

USEREXITED :== “userexited” SPACE CHATID SPACE USERNAME

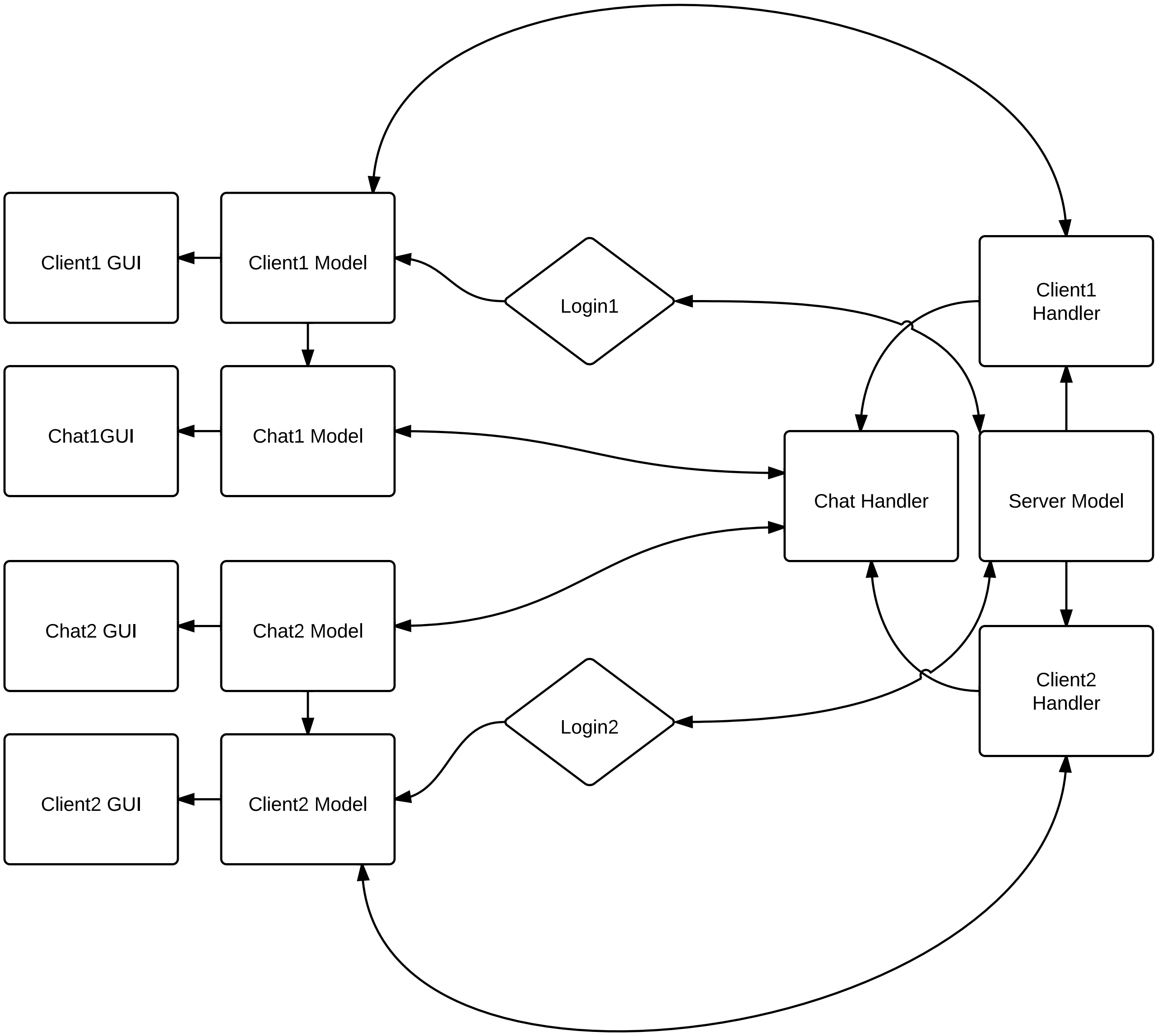
USERNAME :== [a-z0-9]+

CHATID = [0-9]+

MESSAGE :== .+

NEWLINE :== “\n”

SPACE :== “ “



Snapshot Diagram of Client/Server Interaction