De La Salle University
College of Computer Studies
CSNETWK Machine Project – Message Board Demo Kit

Member 1:		Section:	
Member 2:			
Member 3:			

Machine Project – Message Board Demo Kit

A. Startup Process[2]	**SCORE**
Run the Server Application (IP: 127.0.0.1, Port:12345)[1]	
Run 3 instances of the Client Application (Users: Alice, Bob, and Charlie)[1]	

B. Command Test (Alice, Unjoined)[3]	**SCORE**
User Alice checks the command list by using "`/?`"[1]	
User Alice receives a command error from the client using "`/leave`"[2]	
User Alice receives a command error from the client using "`/register`"	
User Alice receives a command error from the client using "`/register Alice`"	
User Alice receives a command error from the client using "`/all`"	
User Alice receives a command error from the client using "`/all Hello World!`"	
User Alice receives a command error from the client using "`/msg`"	
User Alice receives a command error from the client using "`/msg Alice Hello World!`"	
User Alice receives a command error from the client using "`/abcde`"	
User Alice receives a command error from the client using "`join 127.0.0.1 12345`"	

C. User Join Process[5]	**SCORE**
User Alice successfully joins the server using "`/join 127.0.0.1 12345`"[1]	
User Bob unsuccessfully joins the server due to an incorrect port using "`/join 127.0.0.1 12346`"[1]	
User Bob unsuccessfully joins the server due to an incorrect port IP using "`/join 103.231.240.130 12345`"[1]	
User Bob successfully joins the server using "`/join 127.0.0.1 12345`"	
User Charlie unsuccessfully joins the server due to incorrect parameters using "`/join`"[2]	
User Charlie unsuccessfully joins the server due to incorrect parameters using "`/join abcde`"	
User Charlie unsuccessfully joins the server due to incorrect parameters using "`/join abcde 12345`"	
User Charlie unsuccessfully joins the server due to incorrect parameters using "`/join 127.0.0.1 abcde`"	
User Charlie unsuccessfully joins the server due to incorrect parameters using "`/join 127.0.0.1 12345 abcde`"	
User Charlie unsuccessfully joins the server due to incorrect parameters using "`/join abcde 127.0.0.1 12345`"	
User Charlie successfully joins the server using "`/join 127.0.0.1 12345`"	

D. Command Test (Bob, Joined, Unregistered)[5]	SCORE
User Bob checks the command list by using "`/?`"[1]	
User Bob receives a command error from the client using "`/join`"[2]	
User Bob receives a command error from the client using "`/all`"	
User Bob receives a command error from the client using "`/all Hello World!`"	
User Bob receives a command error from the client using "`/msg`"	
User Bob receives a command error from the client using "`/msg Alice Hello World!`"	
User Bob receives a command error from the client using "`/abcde`"	
User Bob unsuccessfully leaves the server due to incorrect parameters using "`/leave Bob`"[1]	
User Bob successfully leaves the server using "`/leave`"[1]	
User Bob successfully rejoins the server using "`/join 127.0.0.1 12345`"	

E. User Handle Registration Process[3]	SCORE
User Alice successfully registers the handle using "`/register Alice`"[1]	
User Bob unsuccessfully registers the handle due to an incorrect syntax using "`/register`"[1]	
User Bob unsuccessfully registers the handle due to an incorrect syntax using "`/register Bob abcde`"[1]	
User Bob successfully registers the handle using "/register Bob"	
User Charlie successfully registers the handle using "/register Charlie"	

F. Command Test (Charlie, Joined, Registered)[5]	SCORE
User Charlie checks the command list by using "`/?`"[1]	
User Charlie receives a command error from the client using "`/join`"[2]	
User Charlie receives a command error from the client using "`/register`"	
User Charlie receives a command error from the client using "`/register Alice`"	
User Charlie receives a command error from the client using "`/register Charlotte`"	
User Charlie receives a command error from the client using "`/register Charlotte abcde`"	
User Charlie receives a command error from the client using "`/all`"[2]	
User Charlie receives a command error from the client using "`/msg`"	
User Charlie receives a command error from the client using "`/msg Alice`"	
User Charlie receives a command error from the client using "`/abcde`"	
User Charlie unsuccessfully leaves the server due to incorrect parameters using "`/leave Charlie`"	
User Charlie successfully leaves the server using "`/leave`"	

G. User Broadcast Messaging Process[8]	**SCORE**
User Alice successfully sends a message to all using "`/all Hello World!`"[2] 1. User Alice successfully receives an echo back reply of "`Alice: Hello World!`" from the server 2. User Bob successfully receives "`Alice: Hello World!`" from the server 3. User Charlie successfully does not receive "`Alice: Hello World!`"	
User Charlie receives a command error from the client using "`/register Alice`"[1]	
User Charlie successfully joins the server using "`/join 127.0.0.1 12345`"	
User Bob successfully sends a message to all using "`/all Hi Alice!`"[1] 1. User Bob successfully receives an echo back reply of "`Bob: Hi Alice!`" from the server 2. User Alice successfully receives "`Bob: Hi Alice!`" from the server 3. User Charlie successfully does not receive "`Bob: Hi Alice!`"	
User Charlie receives a command error from the client using "`/register Bob`"	
User Charlie successfully registers the handle using "`/register Charlie`"	
User Alice successfully sends a message to all using "`/all Nice to meet you!`"[2] 1. User Alice successfully receives an echo back reply of "`Alice: Nice to meet you!`" from the server 2. User Bob successfully receives "`Alice: Nice to meet you!`" from the server 3. User Charlie successfully receives "`Alice: Nice to meet you!`" from the server	
User Charlie successfully sends a message to all using "`/all Glad to be here!`"[1] 1. User Charlie successfully receives an echo back reply of "`Charlie: Glad to be here!`" from the server 2. User Alice successfully receives "`Charlie: Glad to be here!`" from the server 3. User Bob successfully receives "`Charlie: Glad to be here!`" from the server	
User Charlie successfully sends a message to all using "`/all Wherever "here" is.`"[1] 1. User Charlie successfully receives an echo back reply of "`Charlie: Wherever "here" is.`" from the server 2. User Alice successfully receives "`Charlie: Wherever "here" is.`" from the server 3. User Bob successfully receives "`Charlie: Wherever "here" is.`" from the server	

H. User Unicast Messaging Process[11]	**SCORE**
User Alice successfully sends a unicast message to Bob using "`/msg Bob How are you?`"[2] 1. User Alice successfully receives an echo back reply of "`[To Bob]: How are you?`" from the server 2. User Bob successfully receives "`[From Alice]: How are you?`" from the server 3. User Charlie successfully does not receive any message	
User Alice successfully sends a unicast message to Charlie using "`/msg Charlie Nice weather right?`"[1] 1. User Alice successfully receives an echo back reply of "`[To Charlie]: Nice weather, right?`" from the server 2. User Charlie successfully receives "`[From Alice]: Nice weather, right?`" from the server 3. User Bob successfully does not receive any message	
User Bob unsuccessfully sends a unicast message to Charlotte using "`/msg Charlotte Good day!`"[1]	
User Bob successfully sends a unicast message to Alice using "`/msg Alice Doing great!`"[1] 1. User Bob successfully receives an echo back reply of "`[To Alice]: Doing great!`" from the server 2. User Alice successfully receives "`[From Bob]: Doing great!`" from the server 3. User Charlie successfully does not receive any message	
User Charlie successfully sends a unicast message to Alice using "`/msg Alice That's right!`"[2] 1. User Charlie successfully receives an echo back reply of "`[To Alice]: That's right!`" from the server 2. User Alice successfully receives "`[From Charlie]: That's right!`" from the server 3. User Bob successfully does not receive any message	
User Charlie successfully sends a unicast message to Bob using "`/msg Bob BRB, I have to reset my WiFi router.`"[1] 1. User Charlie successfully receives an echo back reply of "`[To Bob]: BRB, I have to reset my WiFi router.`" from the server 2. User Bob successfully receives "`[From Charlie]: BRB, I have to reset my WiFi router.`" from the server 3. User Alice successfully does not receive any message	
User Charlie successfully leaves the server using "`/leave`"	
User Bob unsuccessfully sends a unicast message to Charlie using "`/msg Charlie Sure, no problem!`"[1]	
User Bob successfully sends a unicast message to Alice using "`/msg Alice Hey, Charlie needs to do some networking stuff.`" 1. User Bob successfully receives an echo back reply of "`[To Alice]: Hey, Charlie needs to do some networking stuff.`" from the server 2. User Alice successfully receives "`[From Bob]: Hey, Charlie needs to do some networking stuff.`" from the server 3. User Charlie successfully does not receive any message	
User Alice successfully sends a message to all using "`/all It might have something to do with the server.`"[1] 1. User Alice successfully receives an echo back reply of "`Alice: It might have something to do with the server`" from the server 2. User Bob successfully receives "`Alice: It might have something to do with the server.`" from the server 3. User Charlie successfully does not receive any message	

I. Server Closing Test[2]	**SCORE**
Stop the Server Application	
User Alice unsuccessfully sends a message to all using "`/all Did the server go down?`"[1] 1. User Alice unsuccessfully receives an echo back reply from the server 2. User Bob unsuccessfully receives any message 3. User Charlie successfully does not receive any message	
User Alice unsuccessfully sends a unicast message to Bob using "`/msg Bob Are you still there?`"[1] 1. User Alice unsuccessfully receives an echo back reply from the server 2. User Bob unsuccessfully receives any message 3. User Charlie successfully does not receive any message	
Stop all 3 instances of the Client Applications	

J. Interoperability Test[6]	**SCORE**
Run the tests again using the same Server Application but different Client Application	
Run the tests again using a different Server Application but the same Client Application	
Optional: Check if the messages being sent by the Client to the Server and vice-versa is following the given format	