

Submission Worksheet

CLICK TO GRADE

<https://learn.ethereallab.app/assignment/IT114-002-S2024/it114-chatroom-milestone-3-2024/grade/ns87>

IT114-002-S2024 - [IT114] Chatroom Milestone 3 2024

Submissions:

Submission Selection

1 Submission [active] 4/15/2024 7:22:52 PM

Instructions

^ COLLAPSE ^

Implement the Milestone 3 features from the project's proposal document: <https://docs.google.com/document/d/1ONmvEvel97GTFPGfVwwOC96xSsobbSbk56145Xl>
Make sure you add your ucid/date as code comments where code changes are done
All code changes should reach the Milestone3 branch
Create a pull request from Milestone3 to main and keep it open until you get the output PDF from this assignment.
Gather the evidence of feature completion based on the below tasks.
Once finished, get the output PDF and copy/move it to your repository folder on your local machine.
Run the necessary git add, commit, and push steps to move it to GitHub
Complete the pull request that was opened earlier
Upload the same output PDF to Canvas

Branch name: Milestone3

Tasks: 14 Points: 10.00

Basic UI (2 pts.)

^ COLLAPSE ^



Task #1 - Points: 1

Text: Screenshots of the following

Checklist

*The checkboxes are for your own tracking

#

Points

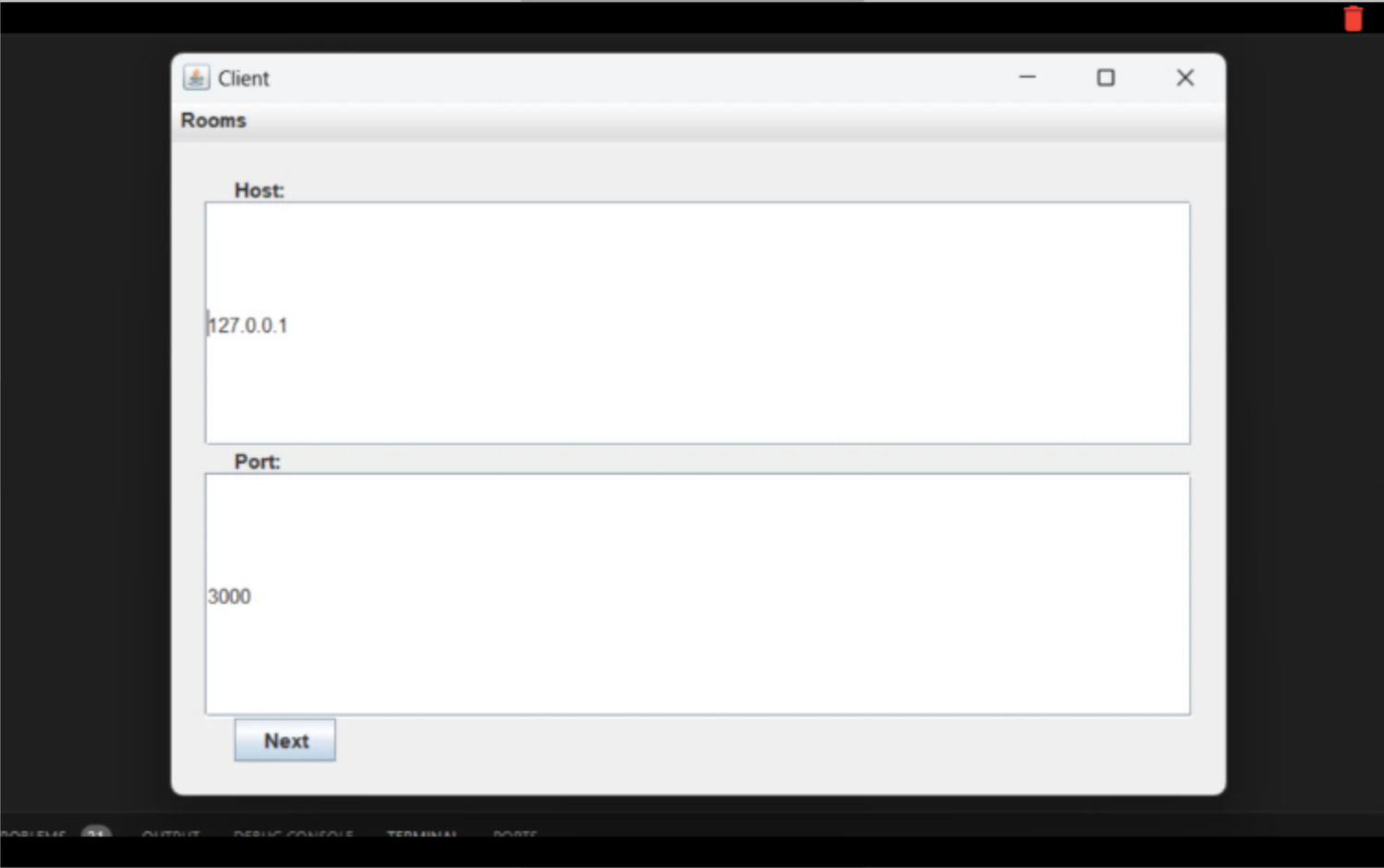
Details

#1	1	Connection Panel
#2	1	User Details Panel
#3	1	Chat Panel
#4	1	Clearly caption screenshots

Task Screenshots:

Gallery Style: Large View

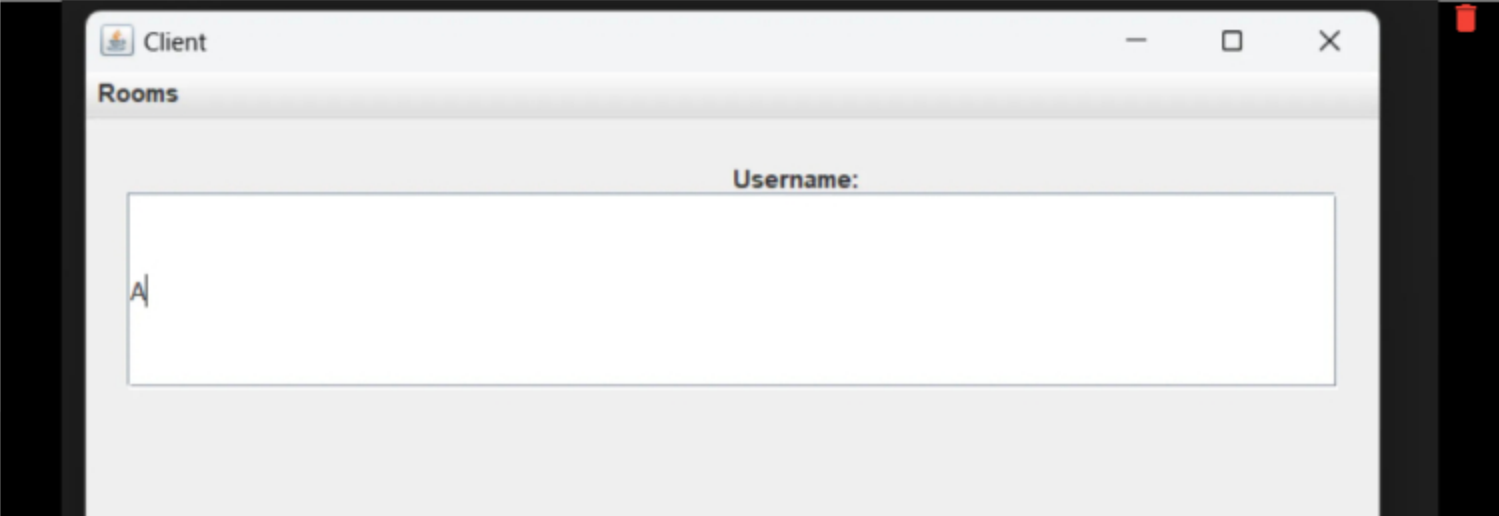
SmallMediumLarge



Connection Panel

Checklist Items (1)

#1 Connection Panel



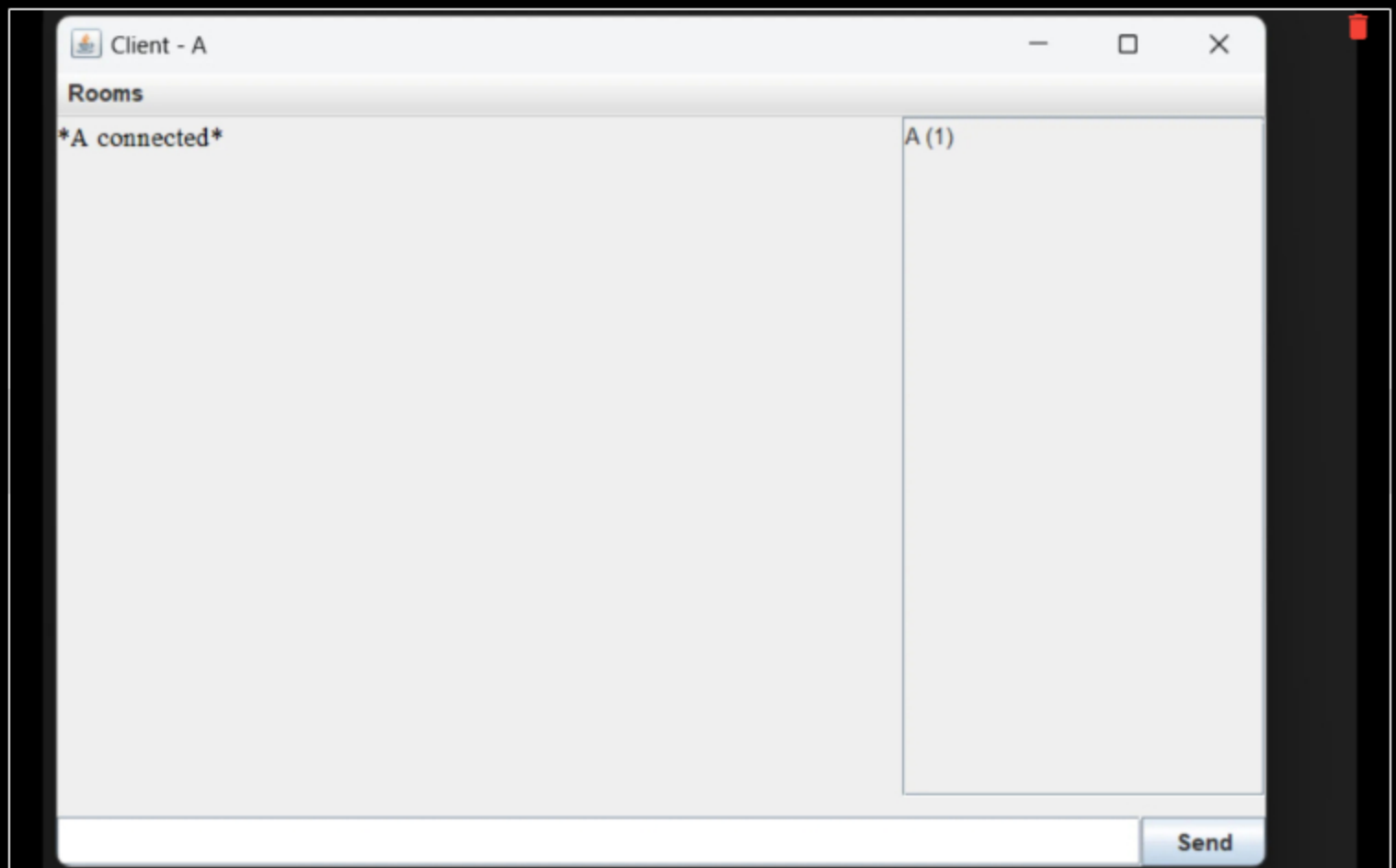
Previous

Connect

User Details Panel

Checklist Items (1)

#2 User Details Panel



Chat Panel

Checklist Items (1)

#3 Chat Panel

● Formatting (2 pts.)

^COLLAPSE ^



Task #1 - Points: 1

^COLLAPSE ^

Checklist

*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Flip output in a different format than normal messages
<input type="checkbox"/> #2	1	Roll # output in a different format than normal messages
<input type="checkbox"/> #3	1	Roll #d# output in a different format than normal messages
<input type="checkbox"/> #4	1	Clearly caption screenshots

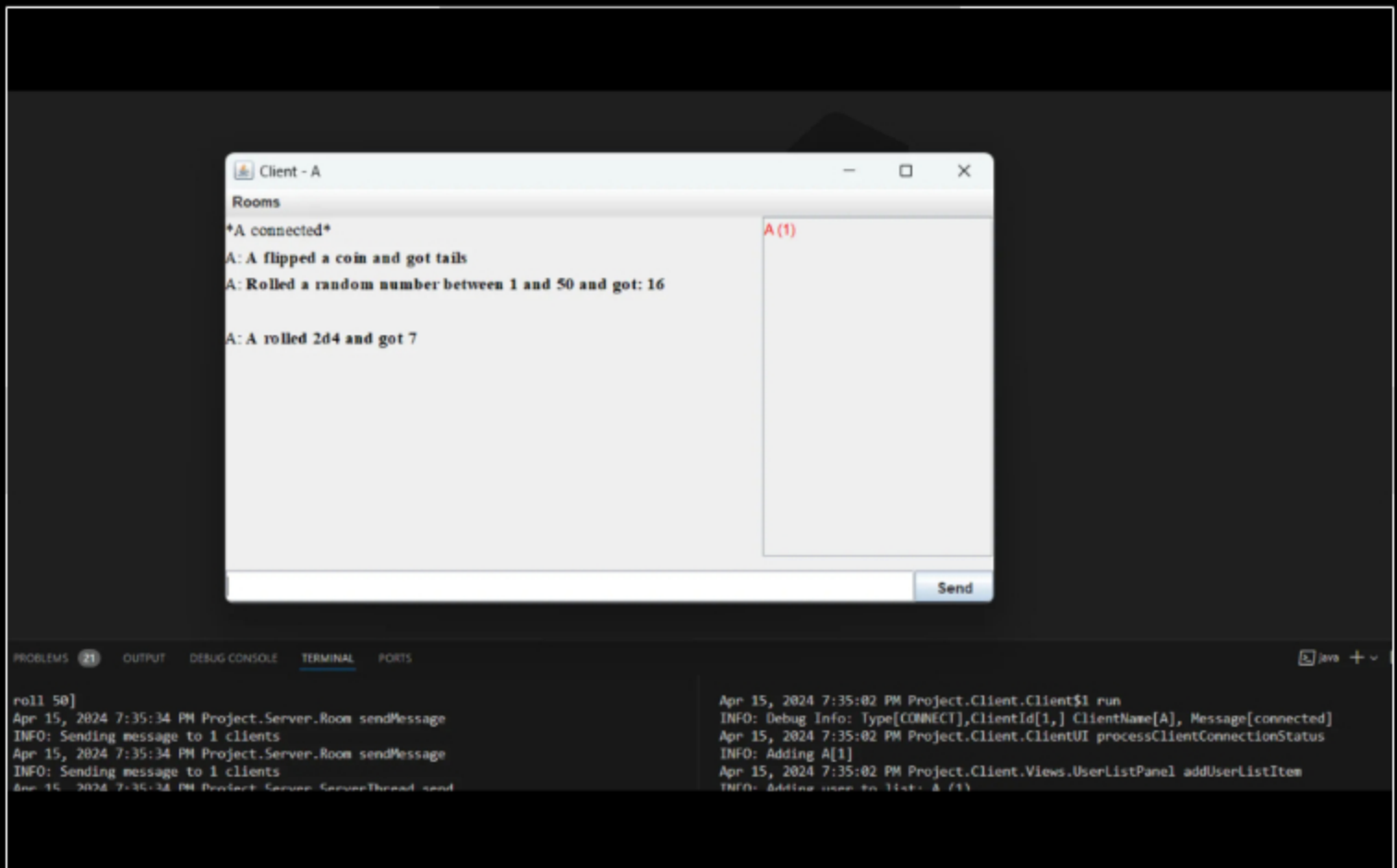
Task Screenshots:

Gallery Style: Large View

Small

Medium

Large



Screenshots demoing flip and roll commands

Checklist Items (4)

#1 Flip output in a different format than normal messages

#2 Roll # output in a different format than normal messages

#3 Roll #d# output in a different format than normal messages

#4 Clearly caption screenshots

Task #2 - Points: 1

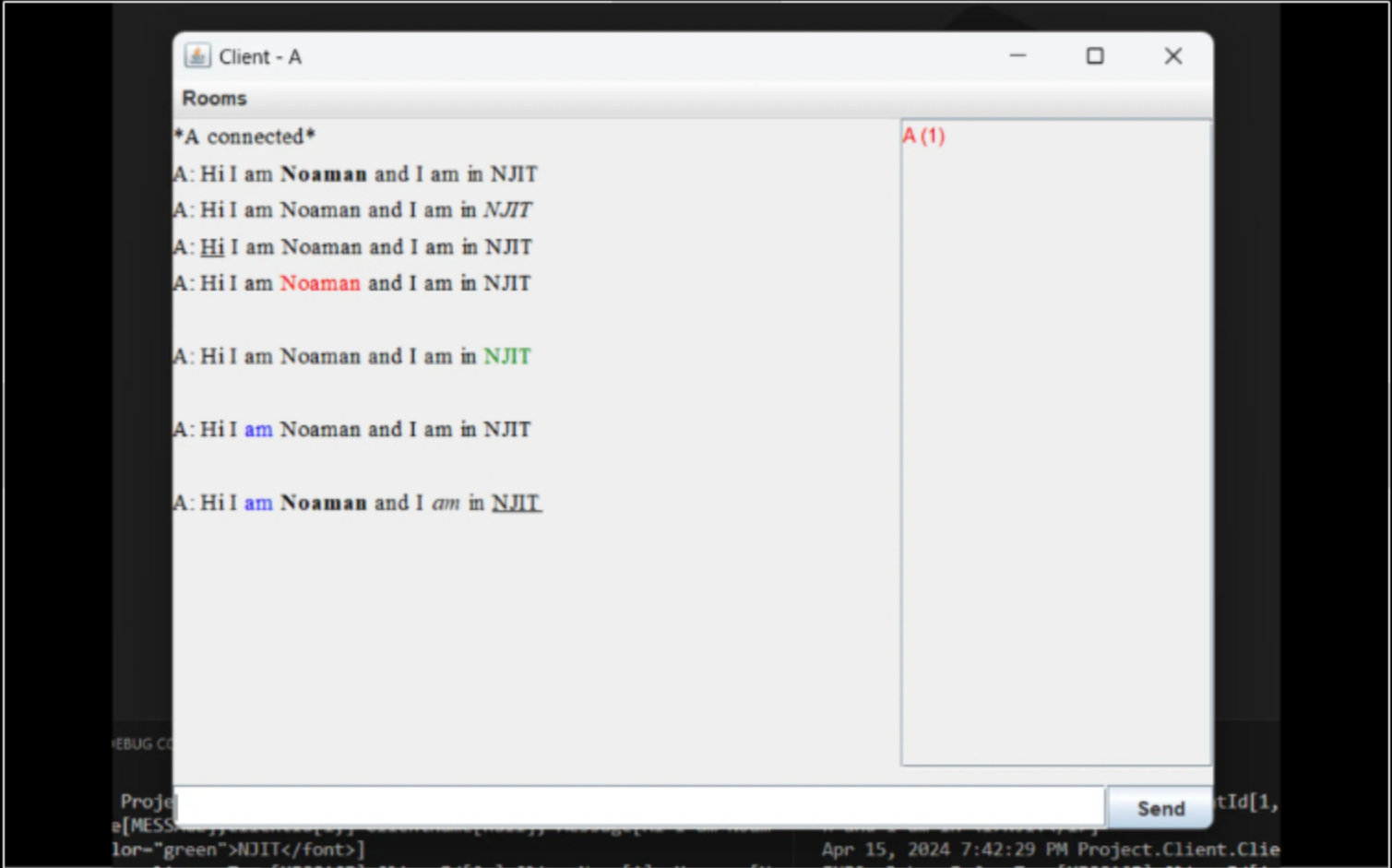
Text: Screenshots demoing custom text formatting

Checklist			*The checkboxes are for your own tracking
#	Points	Details	
<input type="checkbox"/> #1	1	Custom text formatting for bold working (Part of the message should appear bold)	
<input type="checkbox"/> #2	1	Custom text formatting for italic working (Part of the message should appear italic)	
<input type="checkbox"/> #3	1	Custom text formatting for underline working (Part of the message should appear underline)	
<input type="checkbox"/> #4	1	Custom text formatting for red working (Part of the message should appear red)	
<input type="checkbox"/> #5	1	Custom text formatting for blue working (Part of the message should appear blue)	
<input type="checkbox"/> #6	1	Custom text formatting for green working (Part of the message should appear green)	
<input type="checkbox"/> #7	1	Custom text formatting for combined bold, italic, underline, and a color working (Part of the message should have all 4 formats applied at once)	
<input type="checkbox"/> #8	1	Clearly caption screenshots	

Task Screenshots:

Gallery Style: Large View

SmallMediumLarge



Checklist Items (0)

^COLLAPSE ^

Task #3 - Points: 1

Text: Screenshot of the code solving the formatting display

Checklist			*The checkboxes are for your own tracking
#	Points	Details	
#1	1	Show each relevant file this was done in (may be one or more)	
#2	1	Include ucid and date comment	
#3	1	Clearly caption screenshots	

Task Screenshots:

Gallery Style: Large View

SmallMediumLarge



Screenshot of the code solving the formatting display

Checklist Items (3)

- #1 Show each relevant file this was done in (may be one or more)
- #2 Include ucid and date comment

#3 Clearly caption screenshots

```
//UCID:NG87
//Date: 4.1.24

if (message.contains(s:"_")){
    String[] tEffects = message.split(regex:"");
    message = "";
    int count = 0;
    int count2 = 0;
    int indexcount = 0;
    for (int i = 0; i < tEffects.length; i++){
        if (tEffects[i].equals(wordject("_"))){
            count++;
            count2++;
            if (count == 1){
                indexcount = i;
                tEffects[i] = "u";
            }
            if (count == 2){
                tEffects[i] = "x/us";
                count = 0;
            }
        }
    }
    if (count2%2 == 1){
        tEffects[indexcount] = "_";
    }
    for (String i: tEffects){
        message+= i;
    }
}

//UCID:NG87
//Date: 4.1.24

if (message.contains(s:"X")){
    String[] tEffects = message.split(regex:"");
    message = "";
    int count = 0;
    int count2 = 0;
    int indexcount = 0;
    for (int i = 0; i < tEffects.length; i++){
        if (tEffects[i].equals(wordject("X"))){
            count++;
            count2++;
            if (count == 1){
                indexcount = i;
                tEffects[i] = "<font color='red'>";
            }
            if (count == 2){
                tEffects[i] = "</font>";
                count = 0;
            }
        }
    }
    if (count2%2 == 1){
        tEffects[indexcount] = "X";
    }
    for (String i: tEffects){
        message+= i;
    }
}
```

Screenshot of the code solving the formatting display

Checklist Items (0)

```
//UCID:NG87
//Date: 04.01.24

if (message.contains(s:"*")){
    String[] tEffects = message.split(regex:"");
    message = "";
    int count = 0;
    int count2 = 0;
    int indexcount = 0;
    for (int i = 0; i < tEffects.length; i++){
        if (tEffects[i].equals(wordject("*"))){
            count++;
            count2++;
            if (count == 1){
                indexcount = i;
                tEffects[i] = "<font color='green'>";
            }
            if (count == 2){
                tEffects[i] = "</font>";
                count = 0;
            }
        }
    }
    if (count2%2 == 1){
        tEffects[indexcount] = "*";
    }
    for (String i: tEffects){
        message+= i;
    }
}

//UCID:NG87
//Date: 4.01.24

if (message.contains(s:"*")){
    String[] tEffects = message.split(regex:"");
    message = "";
    int count = 0;
    int count2 = 0;
    int indexcount = 0;
    for (int i = 0; i < tEffects.length; i++){
        if (tEffects[i].equals(wordject("*"))){
            count++;
            count2++;
            if (count == 1){
                indexcount = i;
                tEffects[i] = "<font color='blue'>";
            }
            if (count == 2){
                tEffects[i] = "</font>";
                count = 0;
            }
        }
    }
    if (count2%2 == 1){
        tEffects[indexcount] = "*";
    }
    for (String i: tEffects){
        message+= i;
    }
}
```

Checklist Items (0)

Task #4 - Points: 1

Text: Explain how the formatting was made to be visible/rendered in the UI

Details:
Note each scenario

Response:

Changed the addText method in Chatpanel and made the output show text/html.

Private Message with @ (2 pts.)

Task #1 - Points: 1

Text: Screenshots demoing private message

Checklist

*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Should have 3 clients in the same room
<input type="checkbox"/> #2	1	Demo a private message where only the sender and target see the message
<input type="checkbox"/> #3	1	Clearly caption screenshots

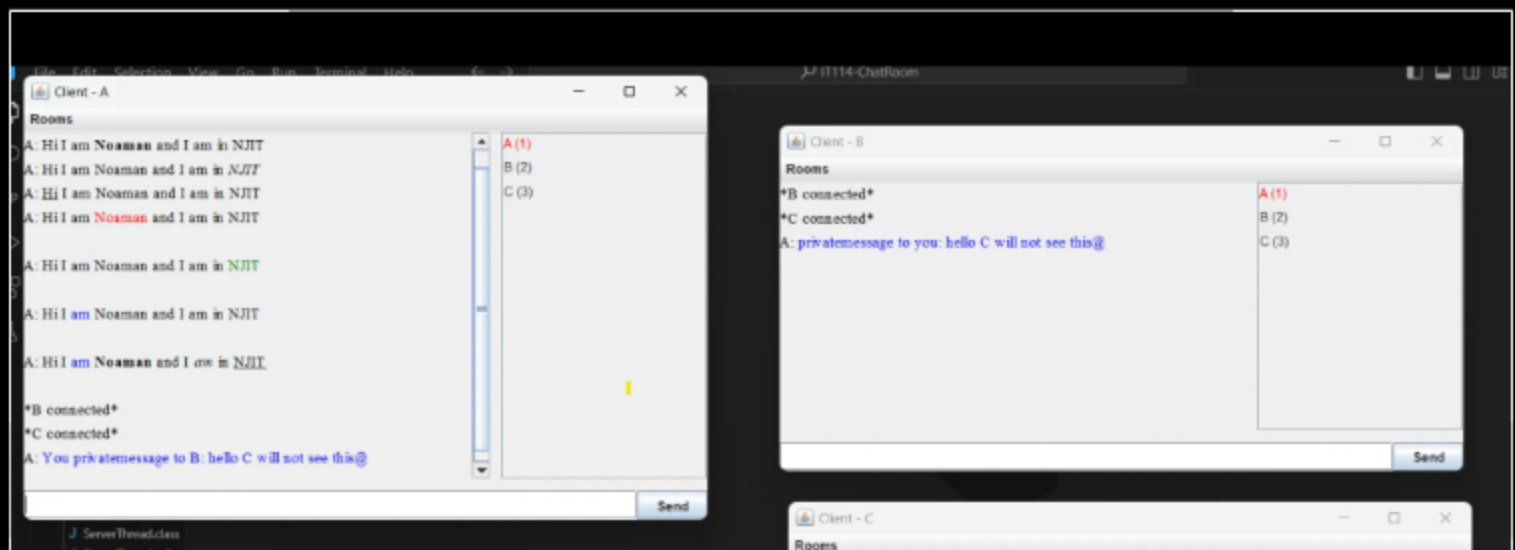
Task Screenshots:

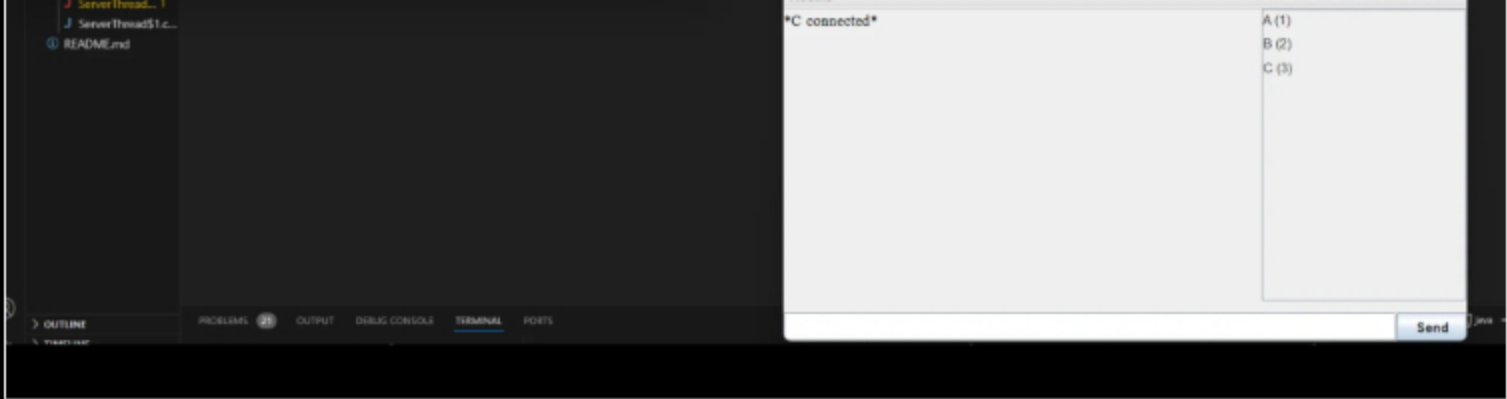
Gallery Style: Large View

Small

Medium

Large





Screenshots demoing private message

Checklist Items (3)

- #1 Should have 3 clients in the same room
- #2 Demo a private message where only the sender and target see the message
- #3 Clearly caption screenshots

^COLLAPSE ^

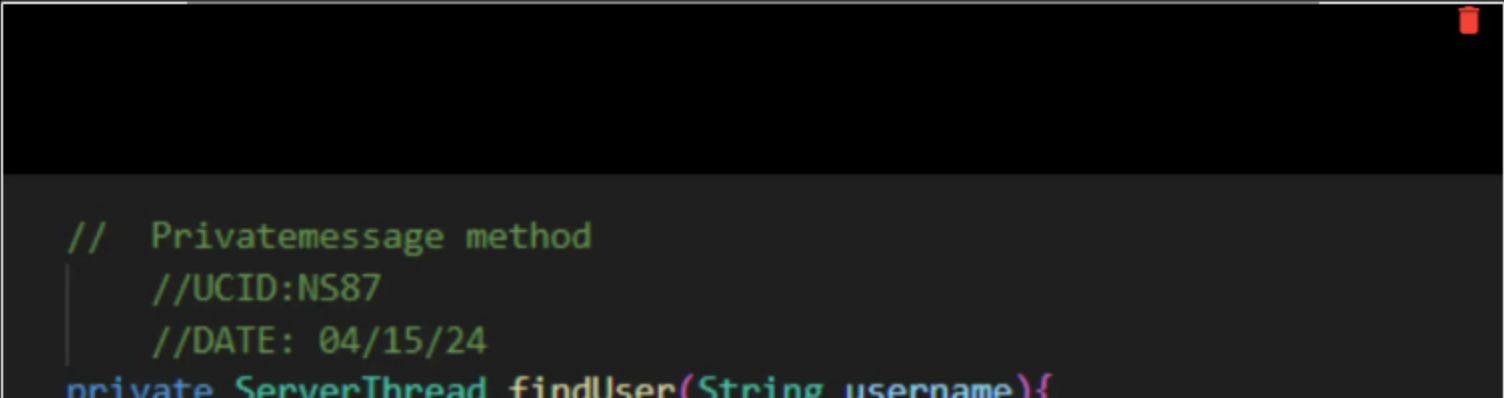
Task #2 - Points: 1

Text: Screenshots of the related code

Checklist			*The checkboxes are for your own tracking
#	Points	Details	
<input type="checkbox"/> #1	1	Show what code processes and handles the private message	
<input type="checkbox"/> #2	1	The message should only be sent to the receiver and the target	
<input type="checkbox"/> #3	1	The client should be targeting the username and the server side should be fetching the correct recipient	
<input type="checkbox"/> #4	1	Include ucid and date comment	
<input type="checkbox"/> #5	1	Clearly caption screenshots	

Task Screenshots:

Gallery Style: Large View



```

private ServerThread findUser(String username){
    for (ServerThread user : clients) {
        if(user.getClientName().equals(username))
            return user;
    }
    return null;
}

```

Screenshots of the related code

Checklist Items (0)

```

//UC10: NSM7
//Date: 4.15.2024
if (message.contains(":P")){
    String[] words = message.split(regex:"\\s+");
    for(String word : words){
        if (word.startsWith(prefix:"P")){
            String privateMessageName = word.substring(beginIndex:1);
            ServerThread targetUser = findUser(privateMessageName);

            if (targetUser != null){
                String senderMessage = "<font color='blue'>You private message to " + targetUser.getClientName() + ":" + message.substring(privateMessageName.length()+1) + "</font>";
                String receiverMessage = "<font color='blue'>private message to you: " + message.substring(privateMessageName.length()+1) + "</font>";
                targetUser.sendMessage(sender.getClientId(), receiverMessage);
                sender.sendMessage(sender.getClientId(), senderMessage);
                return;
            }
        }
    }
}

```

Screenshots of the related code

Checklist Items (0)

Task #3 - Points: 1

Text: Explain how private message works related to the code above

Checklist

*The checkboxes are for your own tracking

#	Points	Details
---	--------	---------

#1	1	Include how the sender and receiver are handled
#2	1	Include how the username is used to get the proper id

Response:

I added a method named findUser within the ServerThread to locate a specific client's name from a list of clients. In the sendMessage method, an if-statement was added to check if a message begins with an @ symbol. When this condition is met, the code proceeds into the if block. A string array called words is generated to remove all spaces from the message. The first index of this array represents the name, which is then assigned to the variable privatemessage. This privatemessage is processed through the findUser method to identify a matching client name. If a user is found, the remaining part of the string array is then utilized in sendMessage statements directed for both sender and receiver.

Mute/Unmute Users (3 pts.)

^COLLAPSE ^

Task #1 - Points: 1

^COLLAPSE ^

Text: Screenshots demoing feature working

Checklist			*The checkboxes are for your own tracking
#	Points	Details	
#1	1	Should have 3 clients in the same room	
#2	1	Demo mute preventing messages between the muter and the target	
#3	1	Demo mute also being accounted for with private messages	
#4	1	Demo unmute allowing the messages again from the target to the unmutter	

Task Screenshots:

Gallery Style: Large View

SmallMediumLarge

Client - Noaman

Rooms

Noaman connected

Ali connected

Saad connected

Noaman: Hi

Ali: Hello

Saad: Hey

Server: Ali has been muted

Server: Ali has been unmuted

Ali: Whatsup

Noaman: You privatemessage to Saad: Whatsup Saad@

Noaman (1)

Ali (2)

Saad (3)

Client - Saad

Rooms

Saad connected

Noaman: Hi

Ali: Hello

Saad: Hey

Ali: Whatsup

Noaman: privatemessage to you: Whatsup Saad@

Noaman (1)

Ali (2)

Saad (3)

Client - Ali

Rooms

Ali connected

Saad connected

Noaman: Hi

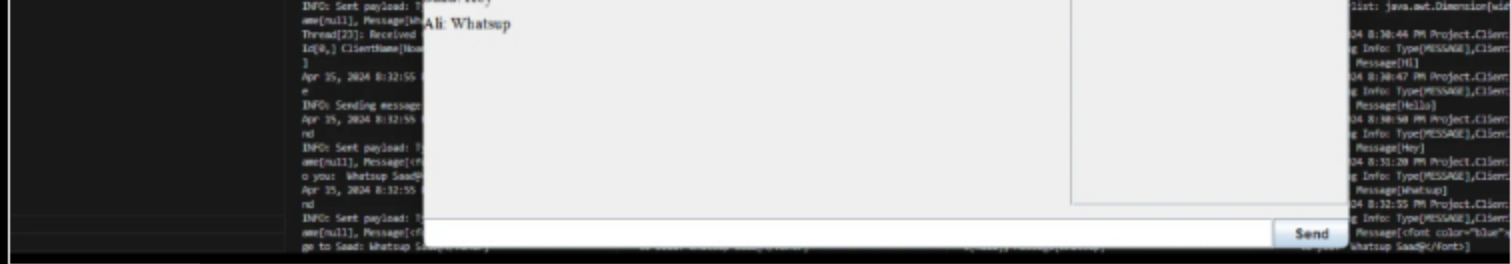
Ali: Hello

Saad: Hey

Noaman (1)

Ali (2)

Saad (3)



Screenshots demoing feature working

Checklist Items (4)

- #1 Should have 3 clients in the same room
- #2 Demo mute preventing messages between the muter and the target
- #3 Demo mute also being accounted for with private messages
- #4 Demo unmute allowing the messages again from the target to the unmuter

COLLAPSE

Task #2 - Points: 1
Text: Screenshots of the related code

Checklist			*The checkboxes are for your own tracking
#	Points	Details	
<input type="checkbox"/> #1	1	ServerThread should have a list of who they muted	
<input type="checkbox"/> #2	1	ServerThread should expose and add, remove, and is muted check to room	
<input type="checkbox"/> #3	1	Room should handle the mute list when receiving the appropriate payloads	
<input type="checkbox"/> #4	1	Room should check the mute list during send message and private messages	
<input type="checkbox"/> #5	1	Include ucid and date comment	
<input type="checkbox"/> #6	1	Clearly caption screenshots	

Task Screenshots:

Gallery Style: Large View

Small

Medium

Large

```
//mute/unmute feature
//UCID: NS87
//DATE: 4/15/24
public boolean sendMuteUser(String name){
    Payload p = new Payload();
    p.setPayloadType(PayloadType.MUTE);
    p.setClientName(name);
    return send(p);
}
public boolean sendUnmuteUser(String name){
```

```

        Payload p = new Payload();
        p.setPayloadType(PayloadType.UNMUTE);
        p.setClientName(name);
        return send(p);
    }

    public boolean isMuted(String name){
        for(String i: muteList){
            if(i.equals(name)){
                return true;
            }
        }
        return false;
    }
}

```

ServerThread

Checklist Items (2)

#1 ServerThread should have a list of who they muted

#2 ServerThread should expose and add, remove, and is muted check to room

```

// UCID: NS87
//DATE: 04/15/24

public ServerThread findMute(String username) {
    for(ServerThread user : clients) {
        if(user.getClientName().equals(username)) {
            return user;
        }
    }
    return null;
}

```

Room

Checklist Items (4)

#3 Room should handle the mute list when receiving the appropriate payloads

#4 Room should check the mute list during send message and private messages

#5 Include ucid and date comment

#6 Clearly caption screenshots



^COLLAPSE ^

Task #3 - Points: 1

Text: Explain how the mute and unmute logic works in relation to the code

Checklist

*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Explain how your mute list is handled
<input type="checkbox"/> #2	1	Explain how it's handled/processed in send message and private message

Response:

It is handled by three methods in ServerThread sendMuteUser , sendUnMuteUser and isMuted . The isMuted communicates with the room and processes payloads. A switch case is used; if the command is Mute it mutes the user if it is unmute it unmutes the client.



Misc (1 pt.)

^COLLAPSE ^



^COLLAPSE ^

Task #1 - Points: 1

Text: Add the pull request link for the branch

Details:

Note: the link should end with /pull/#

URL #1

<https://github.com/Noaman4/IT114/pull/3>



^COLLAPSE ^

Task #2 - Points: 1

Text: Talk about any issues or learnings during this assignment

Response:

This milestone was very insightful and provided me with more details and processes of Java and how applications with functionalities are made. Implementing the functionalities was a little hard for me but overall I have gained a lot of knowledge and hands-on experience while working on this milestone

COLLAPSE

Task #3 - Points: 1

Text: WakaTime Screenshot

Details:

Grab a snippet showing the approximate time involved that clearly shows your repository. The duration isn't considered for grading, but there should be some time involved.

Task Screenshots:

Gallery Style: Large View

Small

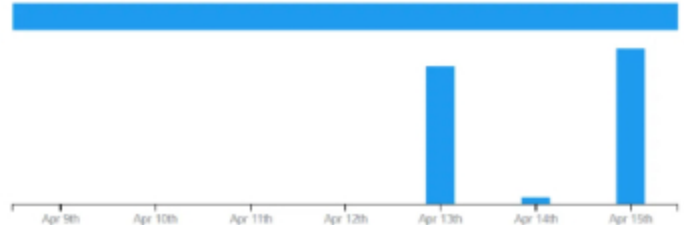
Medium

Large

Projects • IT114-ChatRoom

total 1 hr 18 mins

2 hrs 31 mins over the Last 7 Days in IT114-ChatRoom.



Languages



Editors



WakaTime Screenshot

End of Assignment