Submission Worksheet

CLICK TO GRADE

https://learn.ethereallab.app/assignment/IT114-003-F2024/it114-milestone-3-chatroom-2024-m24/grade/js2637

Course: IT114-003-F2024
Assigment: [IT114] Milestone 3 Chatroom 2024 M24
Student: Jack S. (js2637)

Submissions:

Submission Selection

1 Submission [submitted] 11/24/2024 7:08:46 PM

Instructions

COLLAPSE

Implement the Milestone 3 features from the project's proposal document: https://docs.google.com/document/d/1ONmvEvel97GTFPGfVwwQC96xSsobbSbk56145XizQG4/view 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

Group

100%

Group: Basic UI
Tasks: 1
Points: 2

COLLAPSE

Task

100%

Group: Basic UI
Task #1: UI Panels
Weight: ~100%
Points: ~2.00

COLLAPSE

Details:
All code screenshots must include ucid/date.

App screenshots must have the UCID in the title bar like the lesson gave.

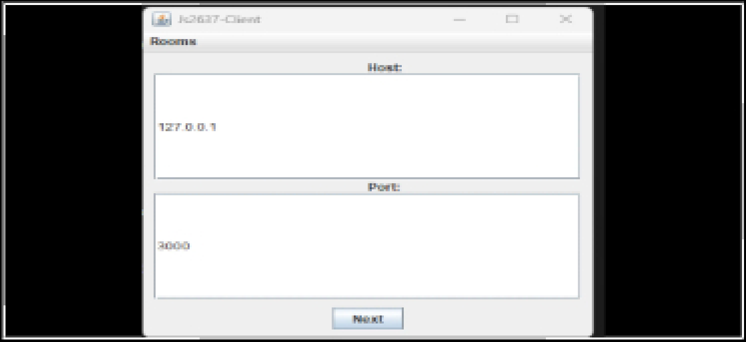
Columns: 1

Sub-Task
100%
Group: Basic UI
Task #1: UI Panels
Sub Task #1: Show the ConnectionPanel by running the app (should have host/port)

Task Screenshots

Gallery Style: 2 Columns

4 2 1



Host:

127.0.0.1

Port:

3000

UI panel

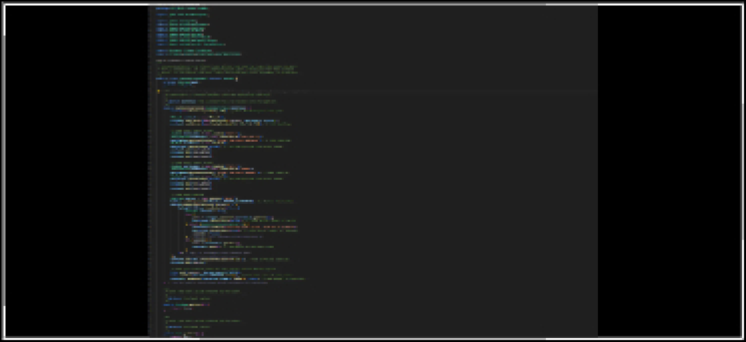
Caption(s) (required) ✓
Caption Hint: *Describe/highlight what's being shown*

Sub-Task
100%
Group: Basic UI
Task #1: UI Panels
Sub Task #2: Show the code related to the ConnectionPanel

Task Screenshots

Gallery Style: 2 Columns

4 2 1



[illegible]

ConnectionPanel

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Briefly explain how it works and how it's used

Response:

It auto fill in your host and port in the UI so your able to connect faster.

Sub-Task

100%

Group: Basic UI

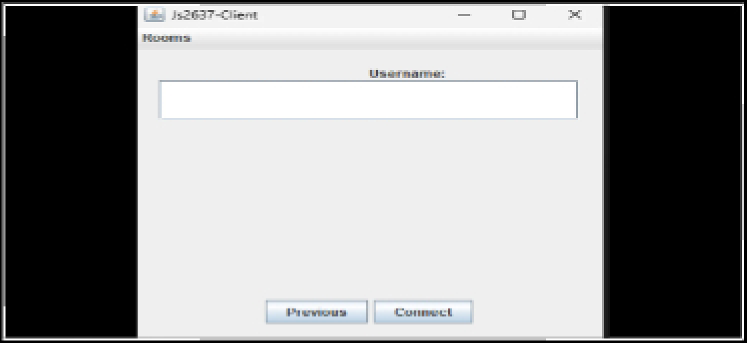
Task #1: UI Panels

Sub Task #3: show the UserDetailsPanel by running the app (should have username)

Task Screenshots

Gallery Style: 2 Columns

4 2 1



User Detail Panel

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Sub-Task

100%

Group: Basic UI

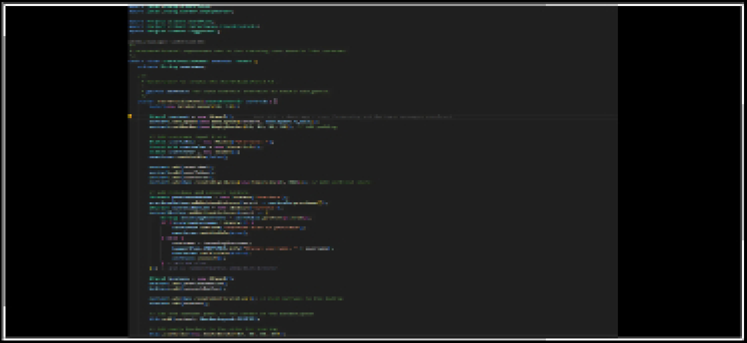
Task #1: UI Panels

Sub Task #4: Show the code related to the UserDetailsPanel

Task Screenshots

Gallery Style: 2 Columns

4 2 1



[illegible]

User Detail Panel

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Briefly explain how it works and how it's used

Response:

make sure the user has a name and is identifiable and make sure that the name isn't blinking and also remove the blink spaces with.trim()

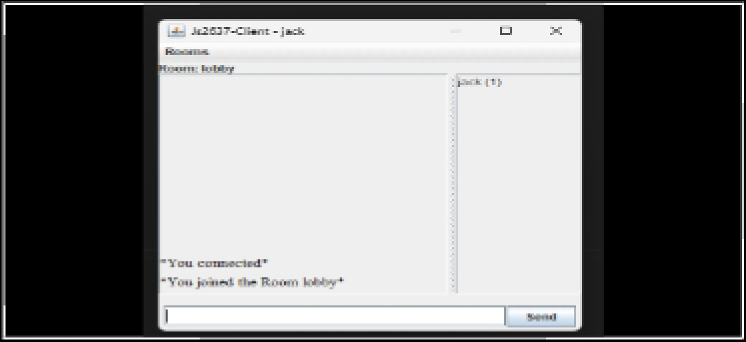
Sub-Task 100%

Group: Basic UI
Task #1: UI Panels
Sub Task #5: Show the ChatPanel (there should be at least 3 users present and some example messages)

Task Screenshots

Gallery Style: 2 Columns

4 2 1



Room: lobby

jack (1)

You connected

You joined the Room lobby

Chat Panel

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

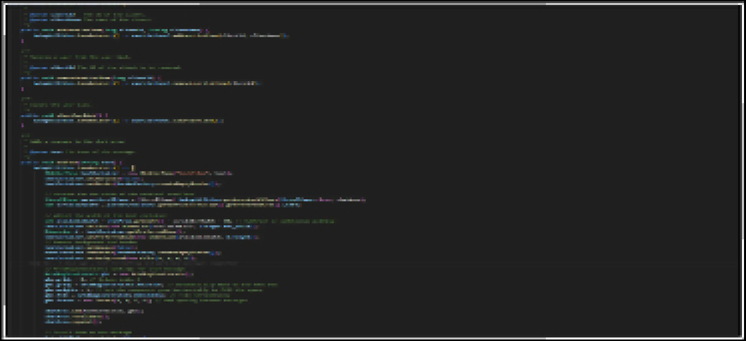
Sub-Task 100%

Group: Basic UI
Task #1: UI Panels
Sub Task #6: Show the code related to the ChatPanel

Task Screenshots

Gallery Style: 2 Columns

4 2 1



[illegible]

Chat Panel code

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Briefly explain how it works and how it's used (note the important parts of the ChatPanel)

Response:

Ensure the text is in HTLM so the code can be underlined, bolded, colored, etc. It also allows the user to get feedback and text from other users. it also lets user see past message that they have typed.

End of Task 1

End of Group: Basic UI
Task Status: 1/1

Group

100%

Group: Build-up
Tasks: 2
Points: 3

COLLAPSE

Task

100%

Group: Build-up
Task #1: Results of /flip and /roll appear in a different format than regular chat text
Weight: ~50%
Points: ~1.50

COLLAPSE

Details:

All code screenshots must include ucid/date.

App screenshots must have the UCID in the title bar like the lesson gave.

Columns: 1

Sub-Task

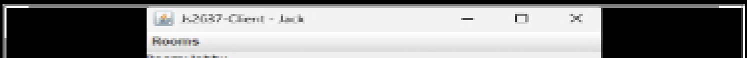
100%

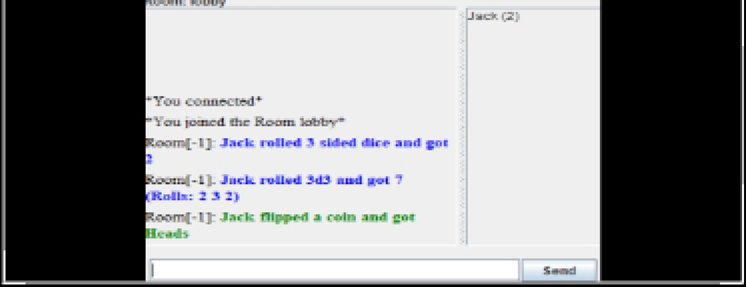
Group: Build-up
Task #1: Results of /flip and /roll appear in a different format than regular chat text
Sub Task #1: Show examples of it printing on screen

Task Screenshots

Gallery Style: 2 Columns

4 2 1




Room: lobby

Jack (2)

You connected

You joined the Room lobby

Room[-1]: **Jack rolled 3 sided dice and got 2**

Room[-1]: **Jack rolled 3d3 and got 7 (Rolls: 2 3 2)**

Room[-1]: **Jack flipped a coin and got Heads**

flip

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Sub-Task

100%

Group: Build-up

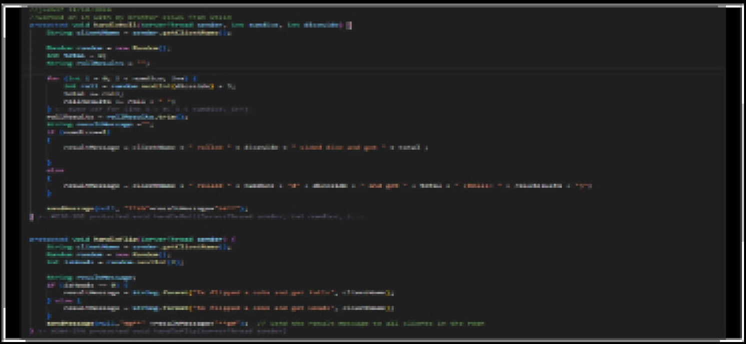
Task #1: Results of /flip and /roll appear in a different format than regular chat text

Sub Task #2: Show the code on the Room side that changes this format

Task Screenshots

Gallery Style: 2 Columns

4 2 1


[illegible]

Roll and Flip code

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain what you did and how it works

Response:

just added #g g#,#b b#, and ** ** to make the text bold and different colors, and then in the chat panel code I just changed the code to JEditorPane textContainer = new JEditorPane("text/html", text) so these will be caused the word to be bold and colored because of the text formatter method from milestone2.

End of Task 1

Task

100%

Group: Build-up

Task #2: Text Formatting appears correctly on the UI

Weight: ~50%

Points: ~1.50

COLLAPSE

ⓘ Details:

All code screenshots must include ucid/date.

App screenshots must have the UCID in the title bar like the lesson gave.

Columns: 1

Sub-Task 100%

Group: Build-up

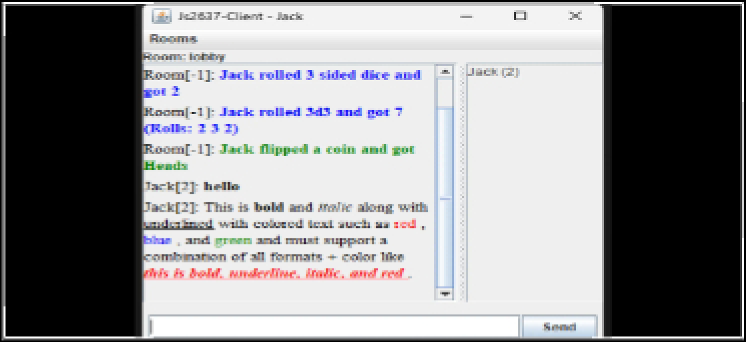
Task #2: Text Formatting appears correctly on the UI

Sub Task #1: Show examples of bold, italic, underline, each color implemented and a combination of bold, italic, underline, and one color in the same message

Task Screenshots

Gallery Style: 2 Columns

4 2 1



Room[-1]: **Jack rolled 3 sided dice and got 2**

Room[-1]: **Jack rolled 3d3 and got 7 (Rolls: 2 3 2)**

Room[-1]: **Jack flipped a coin and got Heads**

Jack[2]: **hello**

Jack[2]: This is **bold** and *italic* along with underlined with colored text such as red , blue , and green and must support a combination of all formats + color like ***this is bold, underline, italic, and red*** .

format text

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Sub-Task 100%

Group: Build-up

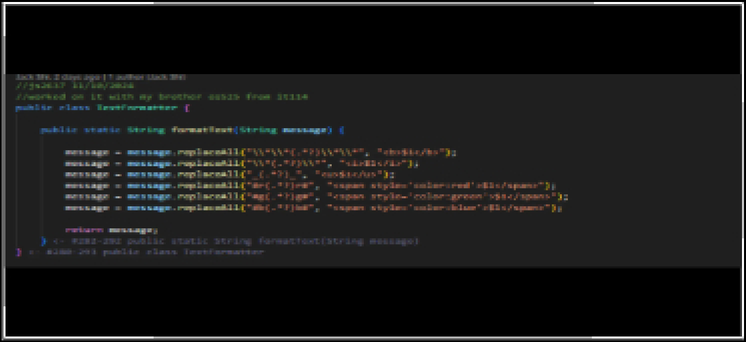
Task #2: Text Formatting appears correctly on the UI

Sub Task #2: Show the code changes necessary to get this to work

Task Screenshots

Gallery Style: 2 Columns

4 2 1



```
Jack Sht, 2 days ago | 1 author (Jack Sht)
//js2637 11/10/2024
//worked on it with my brother ossis from it114
public class TextFormatter {

    public static String formatText(String message) {

        message = message.replaceAll("\\*\\*(.*?)\\*\\*", "<b>$1</b>");
        message = message.replaceAll("\\*(.*?)\\*", "<i>$1</i>");
        message = message.replaceAll("_(.*?)_", "<u>$1</u>");
        message = message.replaceAll("#r(.*?)r#", "<span style='color:red'>$1</span>");
        message = message.replaceAll("#g(.*?)g#", "<span style='color:green'>$1</span>");
        message = message.replaceAll("#b(.*?)b#", "<span style='color:blue'>$1</span>");

        return message;
    } <- #282-292 public static String formatText(String message)
} <- #280-293 public class TextFormatter
```

format text

Caption(s) (required)

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Briefly explain what was necessary and how it works

Response:

Similar to milestone 2 but had to change it because the colors were not working. it replaced the **,_, and #b. with the correct format so that the text would be bold, underlined, etc when it printed out in HTML format.

End of Task 2

End of Group: Build-up
Task Status: 2/2

Group

100%

Group: New Features
Tasks: 2
Points: 4

COLLAPSE

Task

100%

Group: New Features
Task #1: Private messages via @username
Weight: ~50%
Points: ~2.00

COLLAPSE

Details:

All code screenshots must include ucid/date.

App screenshots must have the UCID in the title bar like the lesson gave.

Columns: 1

Sub-Task

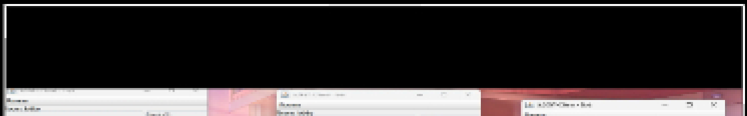
100%

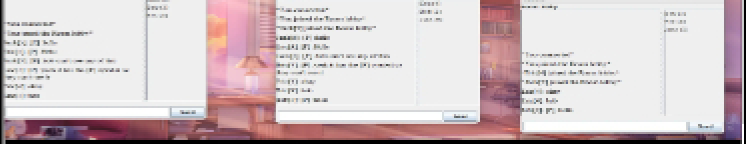
Group: New Features
Task #1: Private messages via @username
Sub Task #1: Show a few examples across different clients (there should be at least 3 clients in the Room)

Task Screenshots

Gallery Style: 2 Columns

4 2 1





[illegible]

private messages

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Sub-Task

100%

Group: New Features

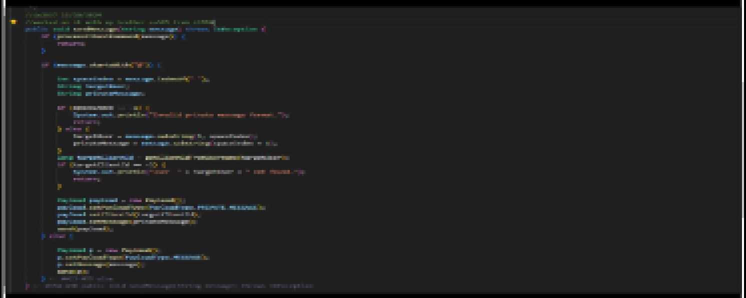
Task #1: Private messages via @username

Sub Task #2: Show the client-side code that processes the text per the requirement

Task Screenshots

Gallery Style: 2 Columns

4 2 1



```
[illegible]
```

private message client sided code

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

this code checks to see if the message starts with an @ so if it does it sets the payload to a private message and sends the payload out if it doesn't start with an @ it sends a regular message.

Sub-Task

100%

Group: New Features

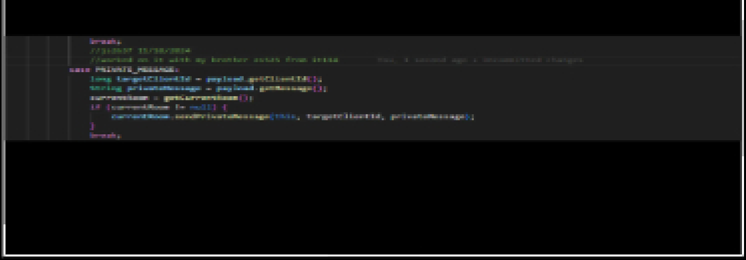
Task #1: Private messages via @username

Sub Task #3: Show the ServerThread code receiving the payload and passing it to Room

Task Screenshots

Gallery Style: 2 Columns

4 2 1



```
        break;
        //jc2637 11/18/2024
        //worked on it with my brother assas from issue        You, 1 second ago • Uncommitted changes
    case PRIVATE_MESSAGE:
        long targetClientId = payload.getClientId();
        String privateMessage = payload.getMessage();
        currentRoom = getCurrentRoom();
        if (currentRoom != null) {
            currentRoom.sendPrivateMessage(this, targetClientId, privateMessage);
        }
        break;
```

private message case

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

this code processes a private message by identifying the person and sending the message if the user is in a chat room.

Sub-Task

100%

Group: New Features

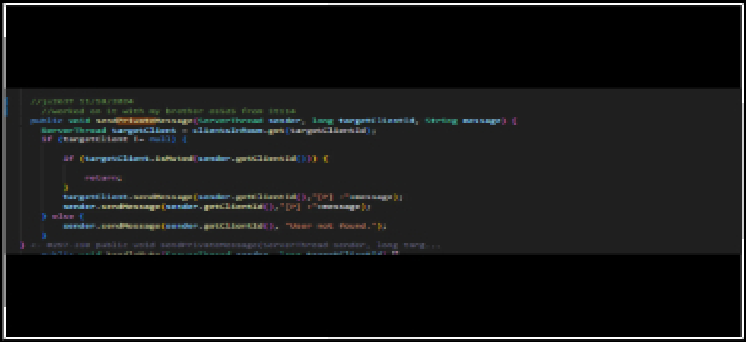
Task #1: Private messages via @username

Sub Task #4: Show the Room code that verifies the id and sends the message to both the sender and receiver

Task Screenshots

Gallery Style: 2 Columns

4 2 1



```
 //jc2637 11/18/2024
   //worked on it with my brother assas from issue
 public void sendPrivateMessage(ServerThread sender, long targetClientId, String message) {
   ServerThread targetClient = clientsInRoom.get(targetClientId);
   if (targetClient != null) {

       if (targetClient.isMuted(sender.getClientId())) {

           return;
       }
       targetClient.sendMessage(sender.getClientId(),"[P] :"+message);
       sender.sendMessage(sender.getClientId(),"[P] :"+message);
   } else {
       sender.sendMessage(sender.getClientId(), "User not found.");
   }
} // end sendPrivateMessage ...
```

private message in room

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

check to see if the client is mute if not it will send the message to the receiver and the messenger with the message but will have [P] in front of the text so they know it's a private message.

End of Task 1

Task

Task

100%

Group: New Features
Task #2: Mute and Unmute
Weight: ~50%
Points: ~2.00

COLLAPSE

Details:
All code screenshots must include ucid/date.
App screenshots must have the UCID in the title bar like the lesson gave.

- Client-side will implement a /mute and /unmute command (i.e., /mute Bob or /unmute Bob)

Columns: 1

Sub-Task

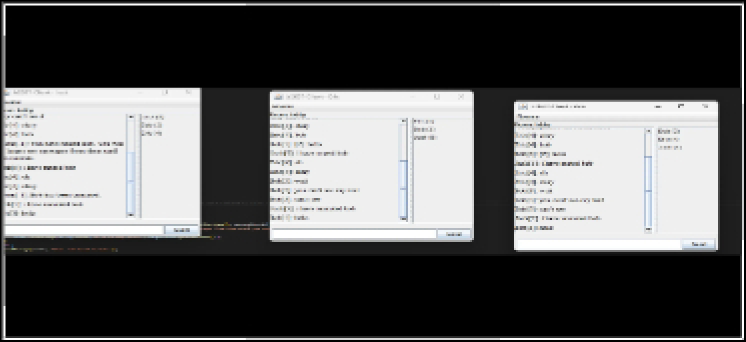
100%

Group: New Features
Task #2: Mute and Unmute
Sub Task #1: Show a few examples across different clients (there should be at least 3 clients in the Room)

Task Screenshots

Gallery Style: 2 Columns

4 2 1



mute/ unmute

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Sub-Task

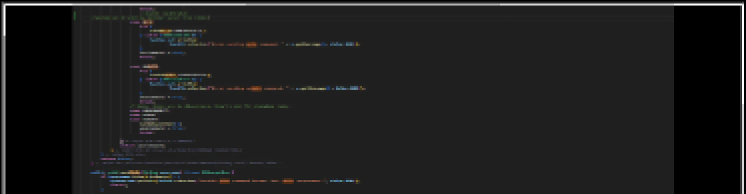
100%

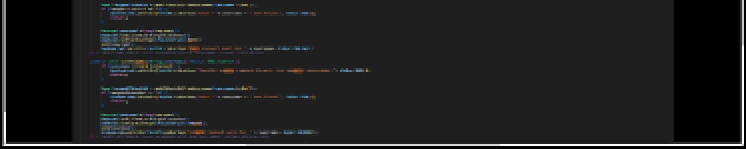
Group: New Features
Task #2: Mute and Unmute
Sub Task #2: Show the client-side code that processes the text per the requirement

Task Screenshots

Gallery Style: 2 Columns

4 2 1





mute/ unmute code in client side

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

the mute and unmute functions recognize commands and call sendMute or sendUnmute with the username. They validate input, retrieve the client ID, create a Payload (with type MUTE or UNMUTE), and send the command. Both handle errors during the process.

Sub-Task 100%

Group: New Features

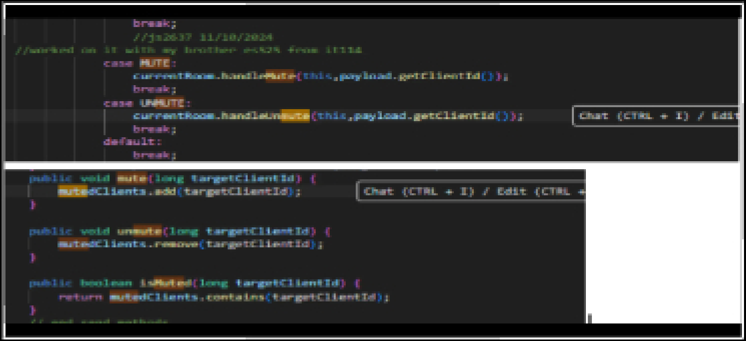
Task #2: Mute and Unmute

Sub Task #3: Show the ServerThread code receiving the payload and passing it to Room

Task Screenshots

Gallery Style: 2 Columns

4 2 1



```
                break;
                //js2637 11/10/2024
//worked on it with my brother es525 from it114
            case MUTE:
                currentRoom.handleMute(this,payload.getClientId());
                break;
            case UNMUTE:
                currentRoom.handleUnmute(this,payload.getClientId());
                break;
            default:
                break;
```

```
    public void mute(long targetClientId) {
        mutedClients.add(targetClientId);
    }

    public void unmute(long targetClientId) {
        mutedClients.remove(targetClientId);
    }

    public boolean isMuted(long targetClientId) {
        return mutedClients.contains(targetClientId);
    }
```

mute/unmute

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

In the MUTE case, the code calls handleMute with the client ID. For UNMUTE, it calls handleUnmute. The mute method adds a client ID to the mutedClients list, while unmute removes it. The isMuted method checks if a client ID is in the mutedClients list.

Sub-Task 100%

Group: New Features

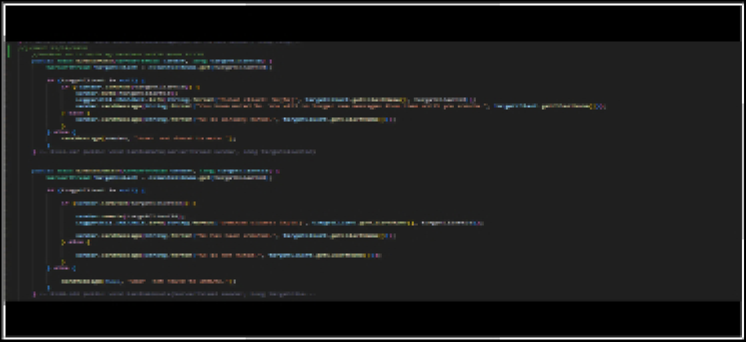
Task #2: Mute and Unmute

Sub Task #4: Show the Room code that verifies the id and add/removes the muted name to/from the ServerThread's list

Task Screenshots

Gallery Style: 2 Columns

4 2 1



```
[illegible]
```

mute code in room

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

The handleMute method checks if the target client exists. If they do and are not muted, it mutes them, logs the action, and notifies the sender. If already muted, it informs the sender. If the target client is not found, it sends an error message. The handleUnmute method similarly checks for the target client. If they exist and are muted, it unmutes them, logs the action, and notifies the sender. If not muted, it informs the sender. If the target is not found, it sends an error message.

Sub-Task

100%

Group: New Features

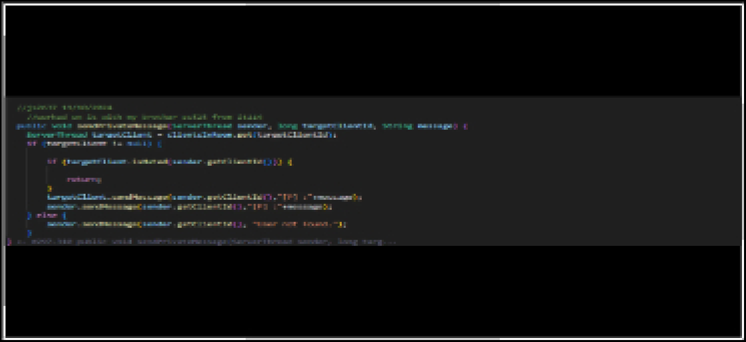
Task #2: Mute and Unmute

Sub Task #5: Show the Room code that checks the mute list during send message. private message, and any other relevant location

Task Screenshots

Gallery Style: 2 Columns

4 2 1



```
[illegible]
```

send message

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

It check to see if the target client is muted in private messaging and if they are they won't receive the message.

Sub-Task

100%

Group: New Features

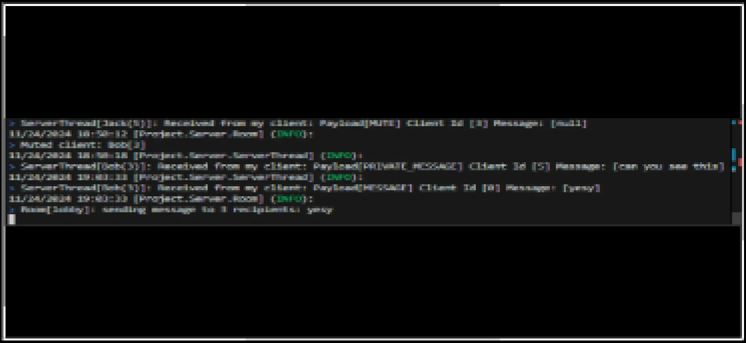
Task #2: Mute and Unmute

Sub Task #6: Show terminal supplemental evidence per the requirements (refer to the details of this task)

Task Screenshots

Gallery Style: 2 Columns

4 2 1



```
> ServerThread[Jack(5)]: Received from my client: Payload[MUTE] Client Id [3] Message: [null]
11/24/2024 18:50:12 [Project.Server.Room] (INFO):
> Muted client: Bob[3]
11/24/2024 18:50:18 [Project.Server.ServerThread] (INFO):
> ServerThread[Bob(3)]: Received from my client: Payload[PRIVATE_MESSAGE] Client Id [5] Message: [can you see this]
11/24/2024 19:03:33 [Project.Server.ServerThread] (INFO):
> ServerThread[Bob(3)]: Received from my client: Payload[MESSAGE] Client Id [0] Message: [yesy]
11/24/2024 19:03:33 [Project.Server.Room] (INFO):
> Room[lobby]: sending message to 3 recipients: yesy
```

mute payload

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

End of Task 2

End of Group: New Features

Task Status: 2/2

Group

100%

Group: Misc

Tasks: 3

Points: 1

COLLAPSE

Task

100%

Group: Misc

Task #1: Add the pull request link for the branch

Weight: ~33%

Points: ~0.33

COLLAPSE

Details:
Note: the link should end with /pull/#

Task URLs

URL #1
https://github.com/Jackshii/Js2637-IT114-003/pull/12

URL
https://github.com/Jackshii/Js2637-IT114-003/p

End of Task 1

Task

100%

Group: Misc
Task #2: Talk about any issues or learnings during this assignment
Weight: ~33%
Points: ~0.33

COLLAPSE

Task Response Prompt

Response:

I had a lot of issues with the mute and unmute functions. I would sometimes mute everyone or have the message that I mute someone but mute no one. I would approach it a different way and still have the same issue but was able to solve it at the end.

End of Task 2

Task

100%

Group: Misc
Task #3: WakaTime Screenshot
Weight: ~33%
Points: ~0.33

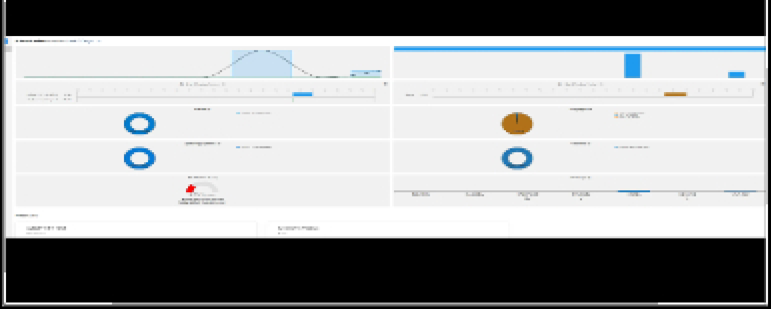
COLLAPSE

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: 2 Columns

4 2 1


waka time

End of Task 3

End of Group: Misc
Task Status: 3/3

End of Assignment