

# Submission Worksheet

CLICK TO GRADE

<https://learn.ethereallab.app/assignment/IT114-002-S2024/it114-chatroom-milestone-4-2024/grade/am3485>

IT114-002-S2024 - [IT114] Chatroom Milestone 4 2024

## Submissions:

Submission Selection

1 Submission [active] 4/30/2024 11:36:50 AM

## Instructions

^ COLLAPSE ^

Implement the Milestone 4 features from the project's proposal document: <https://docs.google.com/document/d/1ONmvEveI97GTFPGfVwwQC96xSsobbSbk56145Xi>  
Make sure you add your ucid/date as code comments where code changes are done  
All code changes should reach the Milestone4 branch  
Create a pull request from Milestone4 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: Milestone4

Tasks: 15 Points: 10.00



Demonstrate Chat History Export (2.25 pts.)

^ COLLAPSE ^



Task #1 - Points: 1

Text: Screenshots of code

Checklist			*The checkboxes are for your own tracking
#	Points	Details	

<input type="checkbox"/> #1	1	Show the code that gets the messages and writes it to a file (recommended to use a StringBuilder)
<input type="checkbox"/> #2	1	File name should be unique to avoid overwriting (i.e., incorporate timestamp)
<input type="checkbox"/> #3	1	Screenshots should include ucid and date comment
<input type="checkbox"/> #4	1	Each screenshot should be clearly captioned

Task Screenshots:

Gallery Style: Large View

SmallMediumLarge

```
@Override
public void onMessageReceive(Long clientId, String message) {
    if (currentCard.ordinal() >= CardView.CHAT.ordinal()) {
        String clientName = Client.INSTANCE.getClientNameFromId(clientId);
        chatPanel.addText(String.format("%s: %s", clientName, message));
        String currentDate = new SimpleDateFormat("MM/dd/yyyy").format(new Date());

        //am3485 4/29/24
        try{
            FileWriter writer = new FileWriter ("ChatHistory.html", true);
            counter++;
            BufferedWriter br = new BufferedWriter(writer);
            PrintWriter pr = new PrintWriter(br);
            if(counter == 1){
                pr.println(currentDate);
            }
            pr.println(String.format("%s: %s", clientName, message));
            pr.close();
            br.close();
            writer.close();
        } <- #195-207 try
        catch(Exception E){

        }

    } <- #189-211 if (currentCard.ordinal() >= CardView.CHAT.ordinal())
} <- #188-212 public void onMessageReceive(long clientId, String message)
```

handling the messages that are sent and making sure they are written to the file

Checklist Items (0)

^COLLAPSE ^

Task #2 - Points: 1  
Text: Screenshot of the file

Checklist			*The checkboxes are for your own tracking
#	Points	Details	
<input type="checkbox"/> #1	1	Show content with variation of messages (i.e., flip, roll, formatting, etc)	
<input type="checkbox"/> #2	1	It should be clear who sent each message	
<input type="checkbox"/> #3	1	Each screenshot should be clearly captioned	

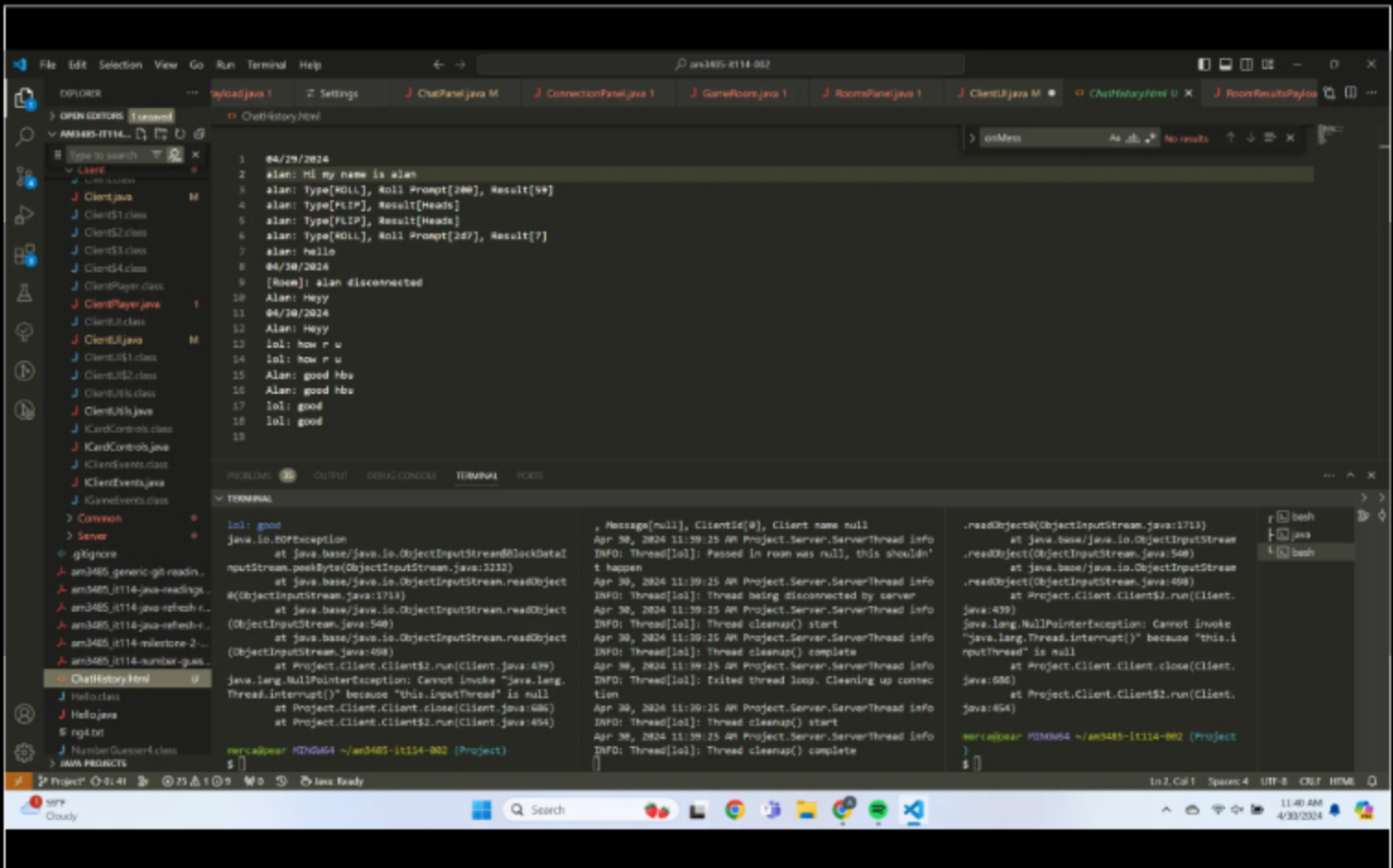
Task Screenshots:

# Gallery Style: Large View

Small

Medium

Large



The screen shot shows how everything is stored including the date the conversation started, messages, flip, and both variations of roll.

## Checklist Items (0)



COLLAPSE

Task #3 - Points: 1

Text: Explain solution

## Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Mention where the messages are stored and how you fetched them
<input type="checkbox"/> #2	1	Mention how the file is generated and populated

## Response:

The messages are stored using writers like File Writer, Buffered Writer, and Print Writer this was the easiest way i found to write to the file without clearing previous history. I also used a counter to only print out the date the first time the file is accessed. As for the method i used to fetch the messages, I know that all the messages must appear on the chat panel which is part of the ui so i went into the ClientUI file and found the function onMessageReceive which send the message to the chat panel and just added the functionality mention before that would write the history to a file.

## Demonstrate Mute List Persistence (2.25 pts.)

^COLLAPSE ^

### Task #1 - Points: 1

Text: Screenshots of the code

#### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Show the code that saves the mute list to a file with the name of the user it belongs to
<input type="checkbox"/> #2	1	Show the code that loads the mute list when a ServerThread is connected
<input type="checkbox"/> #3	1	Screenshots should include ucid and date comment
<input type="checkbox"/> #4	1	Each screenshot should be clearly captioned

Task Screenshots:

Gallery Style: Large View

Small

Medium

Large

Missing Caption

### Task #2 - Points: 1

^COLLAPSE ^

## Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Show a user muting another user, disconnecting, reconnecting, and still having that user muted (same should be possible if the server restarts)
<input type="checkbox"/> #2	1	This should also be reflected in the UI per related feature in this milestone
<input type="checkbox"/> #3	1	Each screenshot should be clearly captioned

Task Screenshots:

Gallery Style: Large View

Small

Medium

Large

Missing Caption

^COLLAPSE ^

Task #3 - Points: 1

Text: Explain solution

## Checklist


\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Mention how you got the mute list to save and load
<input type="checkbox"/> #2	1	Explain the logic to remove the data to the mute list


<input type="checkbox"/> #2	1	Discuss the steps to sync the data to the client/ui
-----------------------------	---	---

Response:

Missing Response

 Demonstrate Mute/Unmute notification (2.25 pts.)

^COLLAPSE ^

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

Checklist		*The checkboxes are for your own tracking
#	Points	Details
<input type="checkbox"/> #1	1	Show how the message is sent to the target user only if their mute/unmute state changes (i.e., doing mute twice for the same user shouldn't send two mute messages)
<input type="checkbox"/> #2	1	Screenshots should include ucid and date comment
<input type="checkbox"/> #3	1	Each screenshot should be clearly captioned

Task Screenshots:

Gallery Style: Large View

Small

Medium

Large

Missing Caption



^COLLAPSE ^

## Task #2 - Points: 1

Text: Screenshots of the demo

### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Show examples of doing /mute twice in succession for the same user only yields one message
<input type="checkbox"/> #2	1	Show examples of doing /unmute twice in succession for the same user only yields one message
<input type="checkbox"/> #3	1	Each screenshot should be clearly captioned

Task Screenshots:

Gallery Style: Large View

Small

Medium

Large

Missing Caption



^COLLAPSE ^

## Task #3 - Points: 1

Text: Explain solution

### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Mention how you limit the messages in each scenario
<input type="checkbox"/> #2	1	Discuss how you find the correct user to send the message to

Response:

Missing Response



Demonstrate user list visual changes (2.25 pts.)

^COLLAPSE ^



^COLLAPSE ^

Task #1 - Points: 1

Text: Screenshots of the code

### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Show the code related to "graying out" muted users and returning them to normal when unmuted
<input type="checkbox"/> #2	1	Show the code related to highlighting the user who last sent a message (and unhighlighting the remainder of the list)
<input type="checkbox"/> #3	1	Screenshots should include ucid and date comment
<input type="checkbox"/> #4	1	Each screenshot should be clearly captioned

Task Screenshots:

Gallery Style: Large View

Small

Medium

Large



Missing Caption

Task #2 - Points: 1

Text: Screenshots of the demo

#### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Show before and after screenshots of the list updating upon mute and unmute
<input type="checkbox"/> #2	1	Capture variations of "last person to send a message gets highlighted"
<input type="checkbox"/> #3	1	Each screenshot should be clearly captioned

Task Screenshots:

Gallery Style: Large View

Small

Medium

Large

Missing Caption

Task #3 - Points: 1

^COLLAPSE ^

Text: Explain solution

### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #1	1	Mention how you got the mute/unmute effect implemented
<input type="checkbox"/> #2	1	Mentioned how you got the highlight effect implemented (including unhighlighting the other users)

Response:

Missing Response



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/alanpear/am3485-it114-002/pull/14>



^COLLAPSE ^

Task #2 - Points: 1

Text: Talk about any issues or learnings during this assignment

Response:

A personal issue i had was my other coursework from other class like 201, 101, and 120 as well as this class lining up which mad it really hard for me to do things with the same amount of care as i normally would and because of that i feel like this wasn't my best work.



^COLLAPSE ^

Task #3 - Points: 1

Text: WakaTime Screenshot

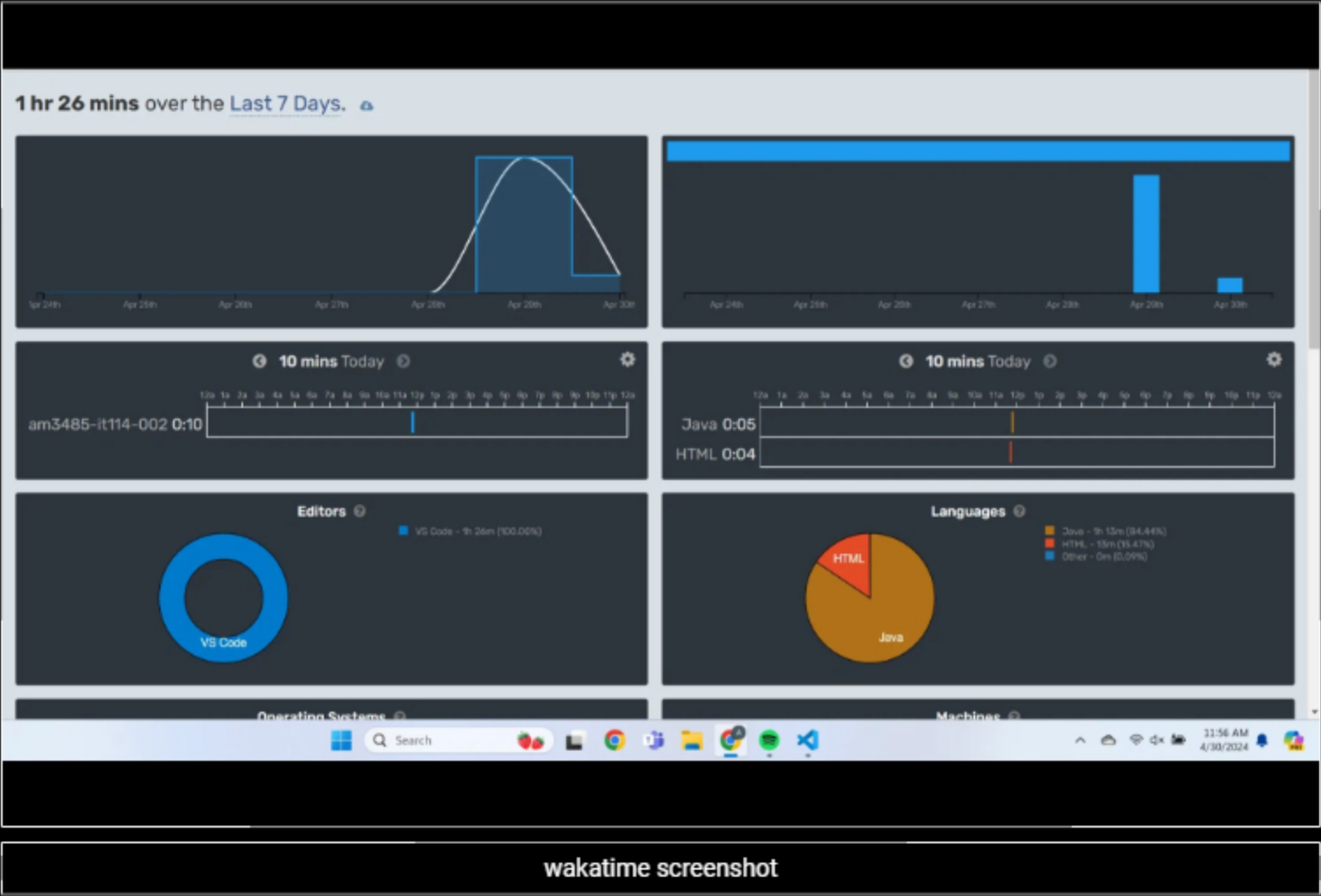
### Details:

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



End of Assignment