# Submission Worksheet

**IT114-006-S2024 - [IT114] Chatroom Milestone 4 2024**

## Submissions:

Submission Selection

1 Submission [active] 4/30/2024 6:39:26 PM ▼

**Instructions**

^ COLLAPSE ^

Implement the Milestone 4 features from the project's proposal document: https://docs.google.com/document/d/1ONmvEvel97GTFPGfVwwQC96xSsobbSbk56145Xi

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**
^COLLAPSE^
**Text: Screenshots of code**

**Checklist**                                          *The checkboxes are for your own tracking

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

| | | | |
|---|---|---|---|
| ☑ #1 | 1 | Show the code that gets the messages and writes it to a file (recommended to use a StringBuilder) | |
| ☑ #2 | 1 | File name should be unique to avoid overwriting (i.e., incorporate timestamp) | |
| ☐ #3 | 1 | Screenshots should include ucid and date comment | |
| ☑ #4 | 1 | Each screenshot should be clearly captioned | |

Task Screenshots:

Gallery Style: Large View

Small          Medium          Large



```java
41    public class ChatPanel extends JPanel {
46        public ChatPanel(ICardControls controls) {
              JButton button = new JButton(text:"Send");
68            // MS75 4-29-24
69            // New export button to export chat history using Java File chooser and java printwriter
70            // as well as java simple date format to have unique filename
71            JButton export = new JButton(text:"Export Chat");
72            input.add(export);
73            export.addActionListener((event) -> {
74            JFileChooser select = new JFileChooser();
75            int x = select.showSaveDialog(this);
76            if (x == JFileChooser.APPROVE_OPTION) {
77                File file = select.getSelectedFile();
78
79                SimpleDateFormat date = new SimpleDateFormat(pattern:"yyyyMMdd_hhmm a");
80                date.setTimeZone(TimeZone.getTimeZone(ID:"America/New_York"));
81                String timeStamp = date.format(new Date());
82                String fileNameWithTimestamp = "chatHistory_" + timeStamp;
83                File fileWithTimestamp = new File(file.getParent(), fileNameWithTimestamp);
84                try (PrintWriter print = new PrintWriter(fileWithTimestamp)) {
85                    for (Component component : chatArea.getComponents()) {
86                        if (component instanceof JEditorPane) {
87                            JEditorPane pane = (JEditorPane) component;
88                            String text = pane.getText();
89                            print.println(text.trim());
90                            print.println(x:"\n");
91                        } <- #86-91 if (component instanceof JEditorPane)
92                    } <- #85-92 for (Component component : chatArea.getComponents())
93                } catch (IOException e) {
94                    e.printStackTrace();
95                }
96            } <- #76-96 if (x == JFileChooser.APPROVE_OPTION)
97        }); <- #73-97 export.addActionListener
98
99
```

Screenshot shows my new jbutton called "Export Chat" that uses filewrite to create a new file with the chat history. As well as the use of java SimpleDateFormat to create a string of the data and time and the string fileNameWithTimeStamp is saved and used when saving 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** | |
| ☑ #1 | 1 | Show content with variation of messages (i.e., flip, roll, formatting, etc) | |
| ☑ #2 | 1 | It should be clear who sent each message | |

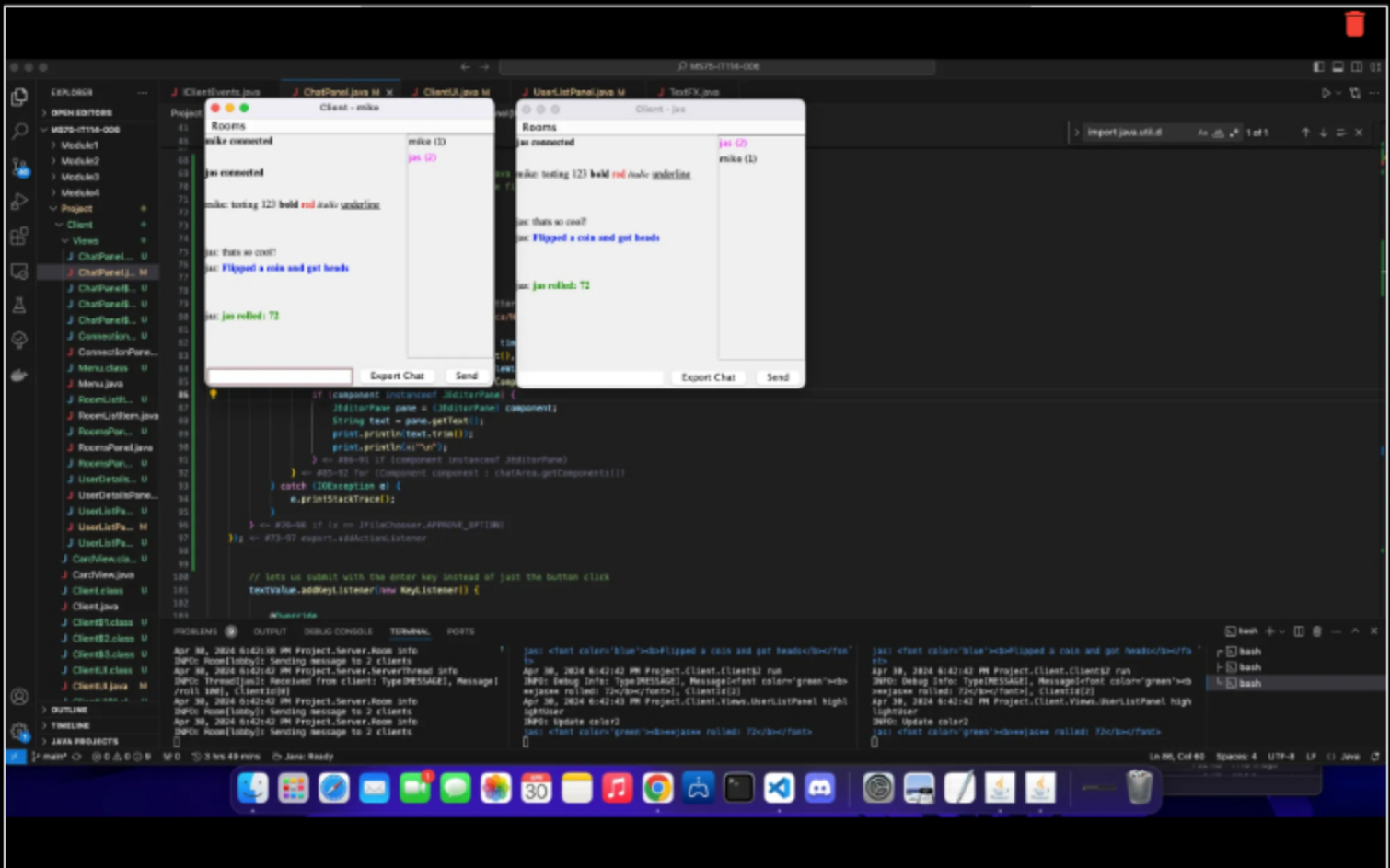| | #3 | 1 | Each screenshot should be clearly captioned |
|---|---|---|---|

**Task Screenshots:**

Gallery Style: Large View
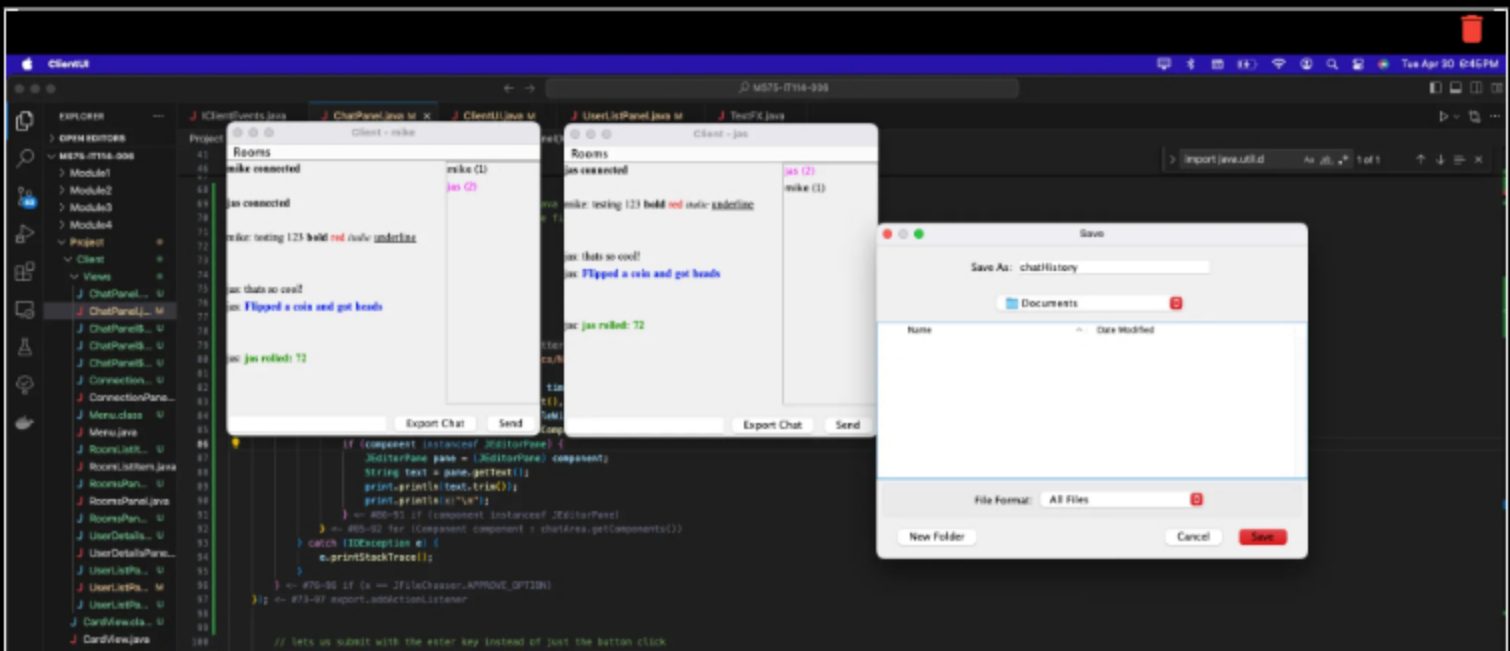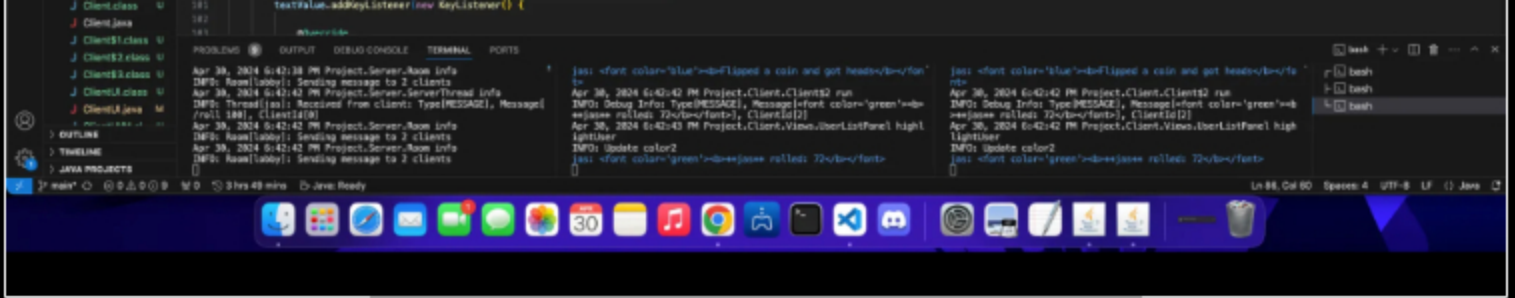
Small          Medium          Large



screenshot just shows the user mike typing a message that has regular text, bold text, colored text, italic and underlined text, it also shows a /flip and /roll in 2 different colors being used.
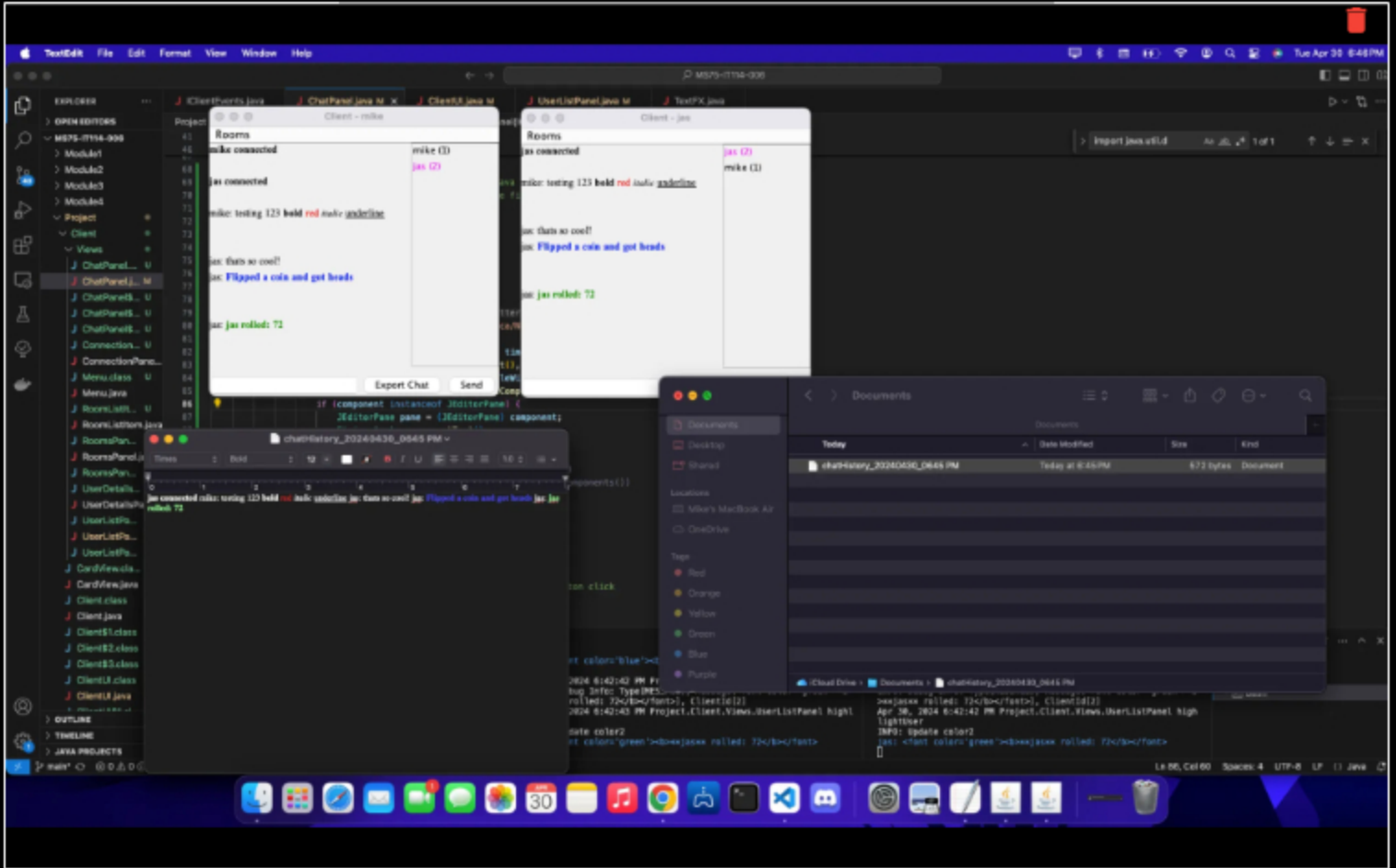
**Checklist Items (1)**

#1 Show content with variation of messages (i.e., flip, roll, formatting, etc)

Screenshot shows me saving the chat by pressing the export chat button and shows me saving it to my Documents directory, i had to enter the filename as chatHistory

Checklist Items (0)



Screenshot shows the file being saved into. my Documents directory with the date and time stamp on the filename to avoid overwriting files.

Checklist Items (0)

## Task #3 - Points: 1

**Text: Explain solution**

^COLLAPSE ^

### Checklist

*The checkboxes are for your own tracking

| # | Points | Details |
|---|--------|---------|
| ☐ #1 | 1 | Mention where the messages are stored and how you fetched them |
| ☐ #2 | 1 | Mention how the file is generated and populated |

Response:

The messages are stored in the chatArea as a content object. Each content component of the chatArea is basically a message that the user has sent along with the clientName. So each message is fetched by going through all the components in lines 84-91. The printwriter is created in a printwriter try catch block where the components from chatArea are fetched using /getComponents and then the string from each jeditor pane are ran through the .getText method to retrieve the text from each component. This .getText stores the text from the pane as a string variable called text and the string variable is then printed out using printWriter.

● **Demonstrate Mute List Persistence (2.25 pts.)**

∧COLLAPSE ∧

● **Task #1 - Points: 1**

∧COLLAPSE ∧

**Text: Screenshots of the code**

**Checklist**                                    *The checkboxes are for your own tracking

| # | Points | Details |
|---|--------|---------|
| ☐ #1 | 1 | Show the code that saves the mute list to a file with the name of the user it belongs to |
| ☐ #2 | 1 | Show the code that loads the mute list when a ServerThread is connected |
| ☐ #3 | 1 | Screenshots should include ucid and date comment |
| ☐ #4 | 1 | Each screenshot should be clearly captioned |

Task Screenshots:

Gallery Style: Large View

Small          Medium          Large

I had no progress on my mute feature since milestone 3 where I could not get it to work. I uploaded the attempted code in Milestone 3.

Checklist Items (0)

🟢

**COLLAPSE**

## Task #2 - Points: 1

**Text: Screenshots of the demo**

**Checklist**                                                          *The checkboxes are for your own tracking

| # | Points | Details |
|---|--------|---------|
| ☐ #1 | 1 | Show a user muting another user, disconnecting, reconnecting, and still having that user muted (same should be possible if the server restarts) |
| ☐ #2 | 1 | This should also be reflected in the UI per related feature in this milestone |
| ☐ #3 | 1 | Each screenshot should be clearly captioned |

Task Screenshots:

Gallery Style: Large View

Small          Medium          Large

Checklist Items (0)

## Task #3 - Points: 1

**Text: Explain solution**

^COLLAPSE ^

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

| # | Points | Details |
|---|--------|---------|
| ☐ #1 | 1 | Mention how you got the mute list to save and load |
| ☐ #2 | 1 | Discuss the steps to sync the data to the client/ui |

Response:

I had no progress on my mute feature since milestone 3 where I could not get it to work. I uploaded the attempted code in Milestone 3.

## Demonstrate Mute/Unmute notification (2.25 pts.)

^COLLAPSE ^

## Task #1 - Points: 1

**Text: Screenshots of the code**

^COLLAPSE ^

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

| # | Points | Details |
|---|--------|---------|
| ☐ #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) |
| ☐ #2 | 1 | Screenshots should include ucid and date comment |
| ☐ #3 | 1 | Each screenshot should be clearly captioned |

Task Screenshots:

Gallery Style: Large View

Small          Medium          Large

I had no progress on my mute feature since milestone 3 where I could not get it to work. I uploaded the attempted code in Milestone 3.

**Checklist Items (0)**

## Task #2 - Points: 1

^COLLAPSE ^
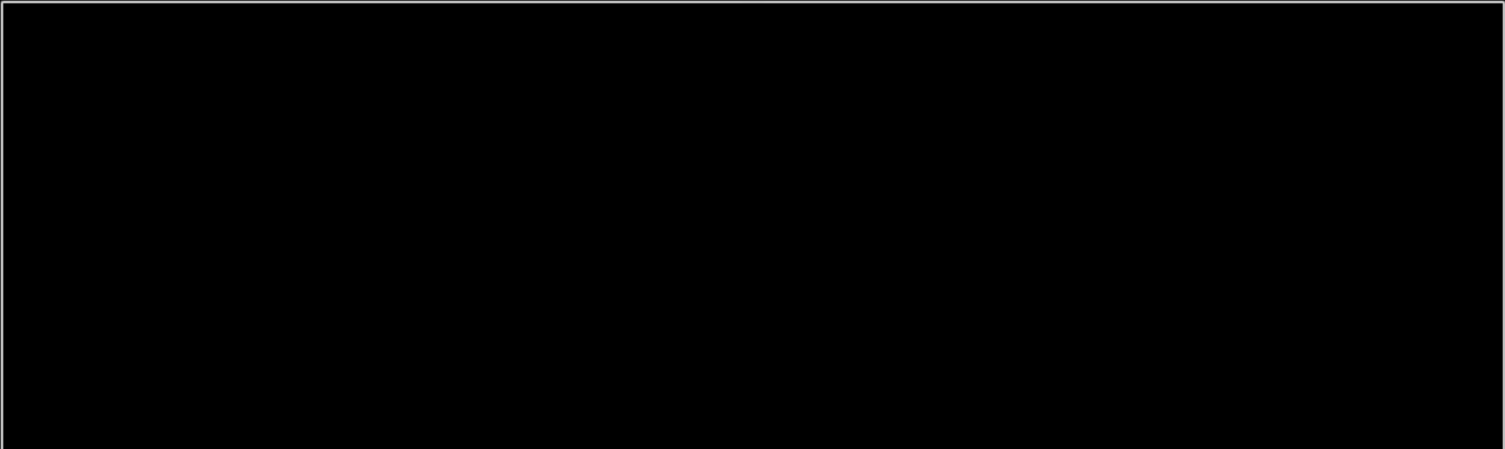
**Text: Screenshots of the demo**

**Checklist**　　　　　　　　　　　　　　　　　　　　　　*The checkboxes are for your own tracking

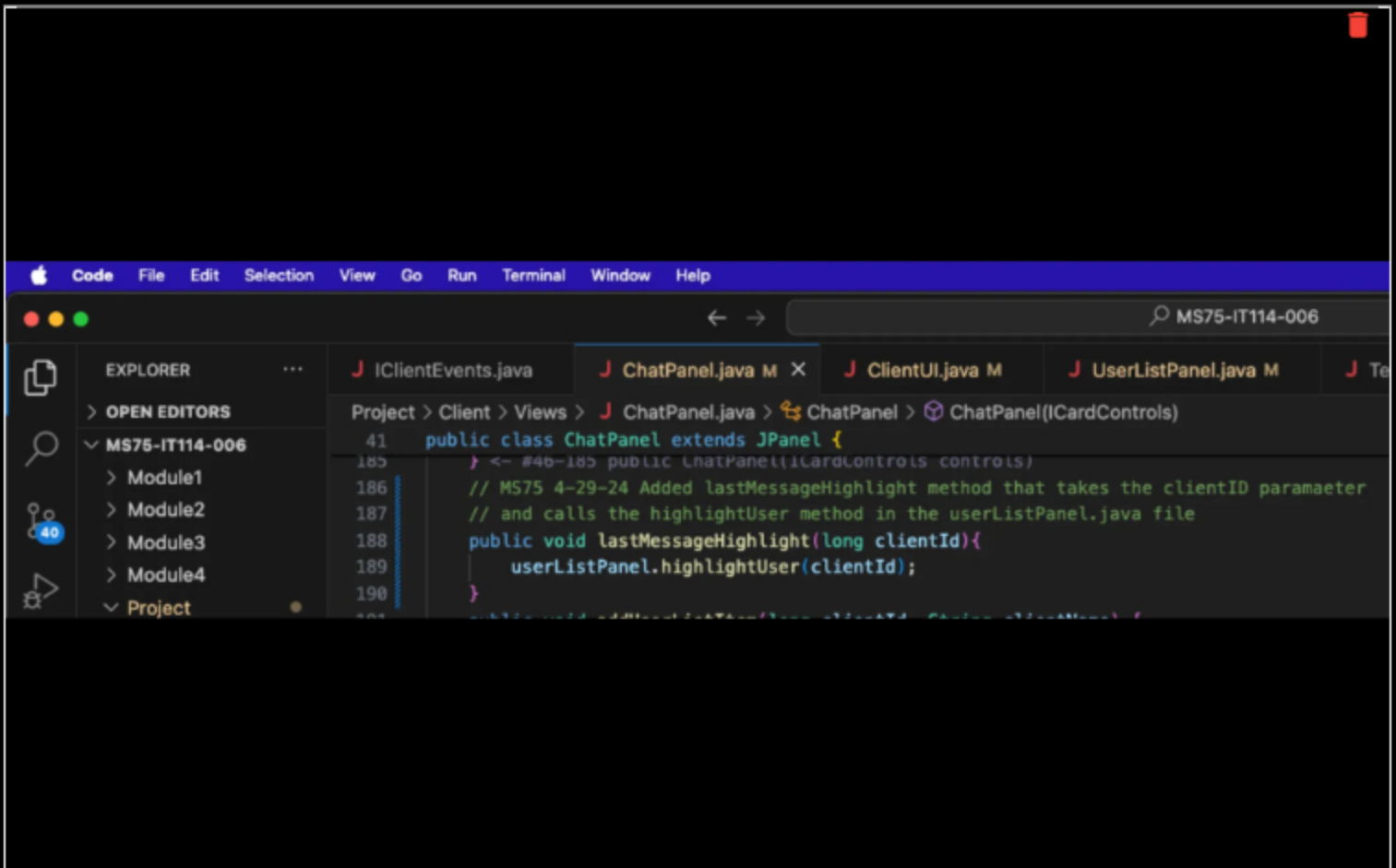| # | Points | Details |
|---|--------|---------|
| ☐ #1 | 1 | Show examples of doing /mute twice in succession for the same user only yields one message |
| ☐ #2 | 1 | Show examples of doing /unmute twice in succession for the same user only yields one message |
| ☐ #3 | 1 | Each screenshot should be clearly captioned |

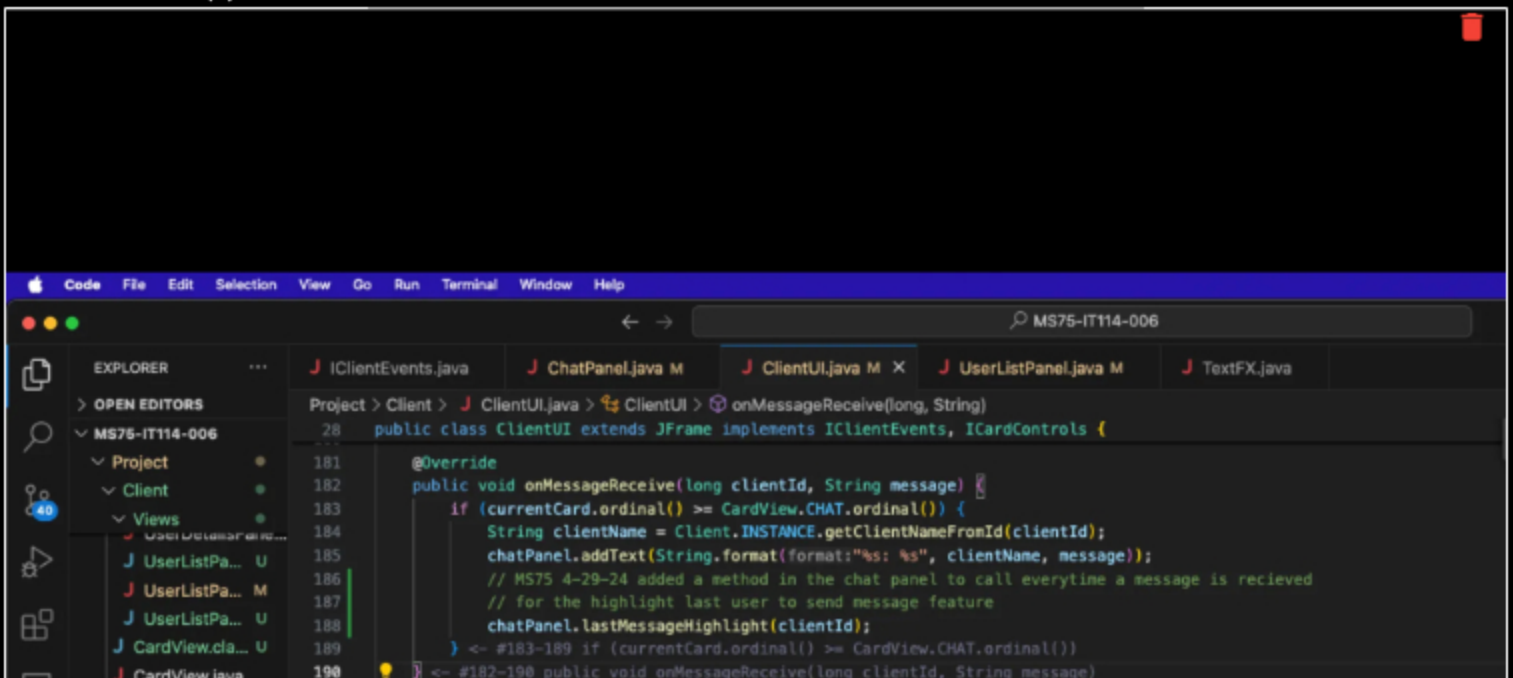Task Screenshots:

Gallery Style: Large View

Small　　　　Medium　　　　Large

I had no progress on my mute feature since milestone 3 where I could not get it to work. I uploaded the attempted code in Milestone 3.

Checklist Items (0)

● 
ᐱCOLLAPSE ᐱ **Task #3 - Points: 1**

**Text: Explain solution**

| Checklist | | *The checkboxes are for your own tracking |
|---|---|---|

| # | Points | Details |
|---|---|---|
| ☐ #1 | 1 | Mention how you limit the messages in each scenario |
| ☐ #2 | 1 | Discuss how you find the correct user to send the message to |

Response:

I had no progress on my mute feature since milestone 3 where I could not get it to work. I uploaded the attempted code in Milestone 3.

● **Demonstrate user list visual changes** (2.25 pts.)
ᐱCOLLAPSE ᐱ

● 
ᐱCOLLAPSE ᐱ **Task #1 - Points: 1**

**Text: Screenshots of the code**

| Checklist | | *The checkboxes are for your own tracking |
|---|---|---|

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

Task Screenshots:

Gallery Style: Large View

```java
public class ChatPanel extends JPanel {
    } <- #46-185 public ChatPanel(ICardControls controls)
    // MS75 4-29-24 Added lastMessageHighlight method that takes the clientID paramaeter
    // and calls the highlightUser method in the userListPanel.java file
    public void lastMessageHighlight(long clientId){
        userListPanel.highlightUser(clientId);
    }
```

I had no progress on my mute feature since milestone 3 where I could not get it to work. I uploaded the attempted code in Milestone 3. I did get the highlighting the last client to talk feature to work and this picture shows the added code in the ChatPanel.java class. All that was added was the lastMessageHighlight method

Checklist Items (0)



```java
public class ClientUI extends JFrame implements IClientEvents, ICardControls {
    @Override
    public void onMessageReceive(long clientId, String message) {
        if (currentCard.ordinal() >= CardView.CHAT.ordinal()) {
            String clientName = Client.INSTANCE.getClientNameFromId(clientId);
            chatPanel.addText(String.format(format:"%s: %s", clientName, message));
            // MS75 4-29-24 added a method in the chat panel to call everytime a message is recieved
            // for the highlight last user to send message feature
            chatPanel.lastMessageHighlight(clientId);
        } <- #183-189 if (currentCard.ordinal() >= CardView.CHAT.ordinal())
    } <- #182-190 public void onMessageReceive(long clientId, String message)
```

This code shows the addition to the onMessageRecieve() method in the ClientUI.java file. All that was added was the last line that called the lastMessageHighlight method in the chatPanel class: chatPanel.lastMessageHighlight(clientId);

Checklist Items (0)
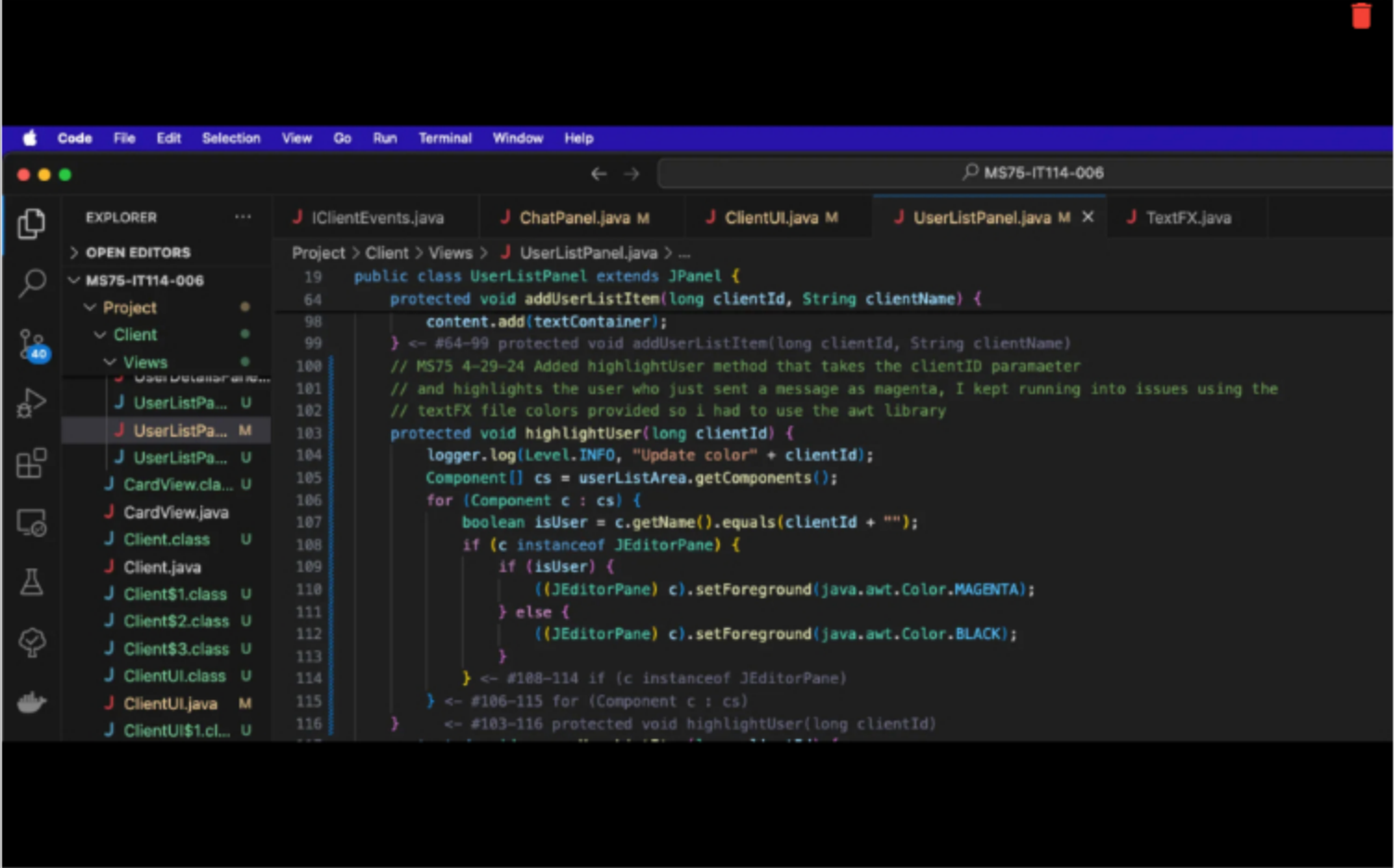


screenshot shows my highlightUser method that is called in the chatPanel.java file. The file gets all the components of the UserListPanel pane and sets the foreGround (front) color of the pane as MAGENTA or purple when called.
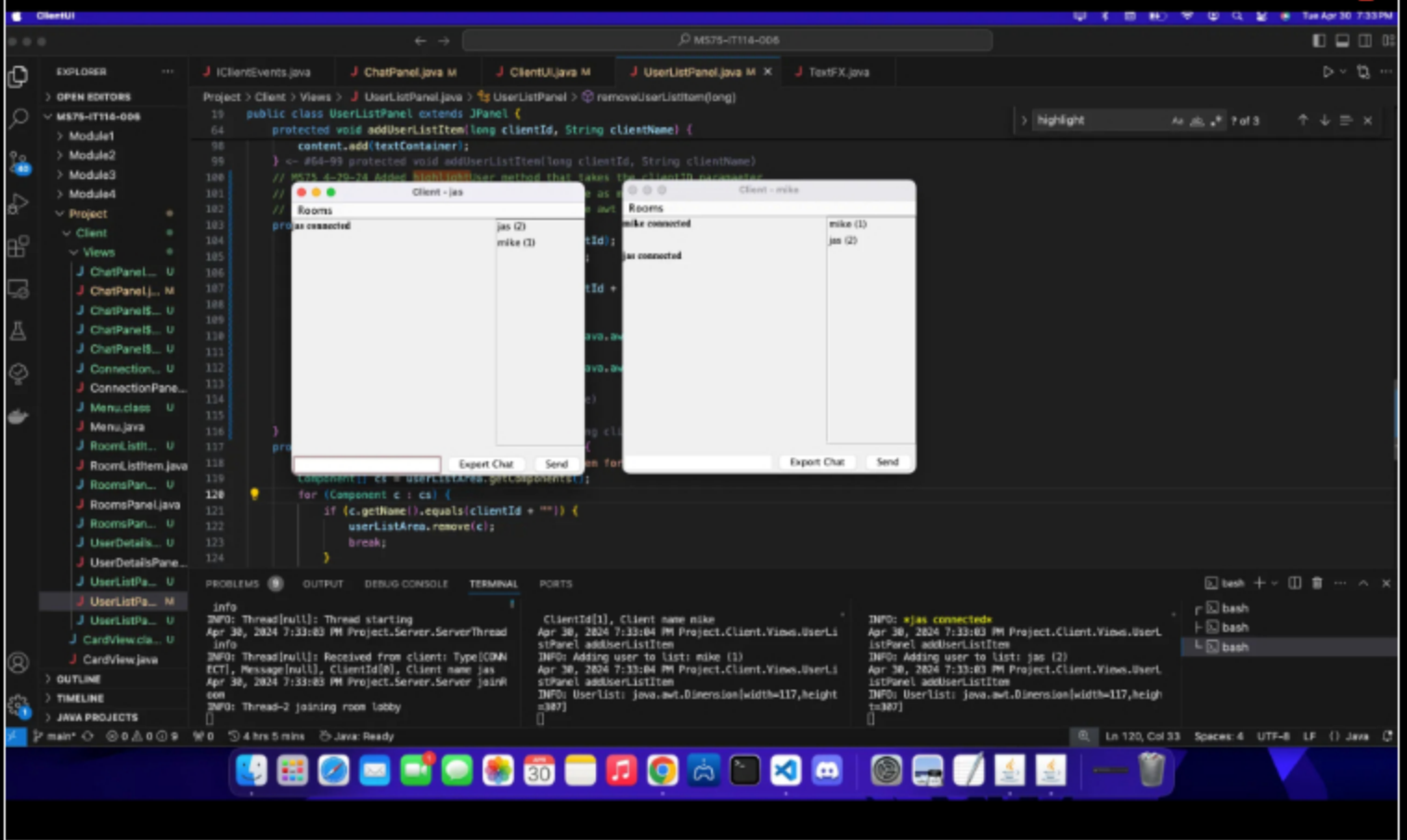
Checklist Items (0)

**Task #2 - Points: 1**

**Text: Screenshots of the demo**

∧COLLAPSE ∧

**Checklist**                                                    *The checkboxes are for your own tracking

| # | Points | Details |
|---|--------|---------|
| ☐ #1 | 1 | Show before and after screenshots of the list updating upon mute and unmute |
| ☐ #2 | 1 | Capture variations of "last person to send a message gets highlighted" |
| ☐ #3 | 1 | Each screenshot should be clearly captioned |

Task Screenshots:

Gallery Style: Large View

Small          Medium          Large

This screenshot just shows 2 clients connected with no user highlight.
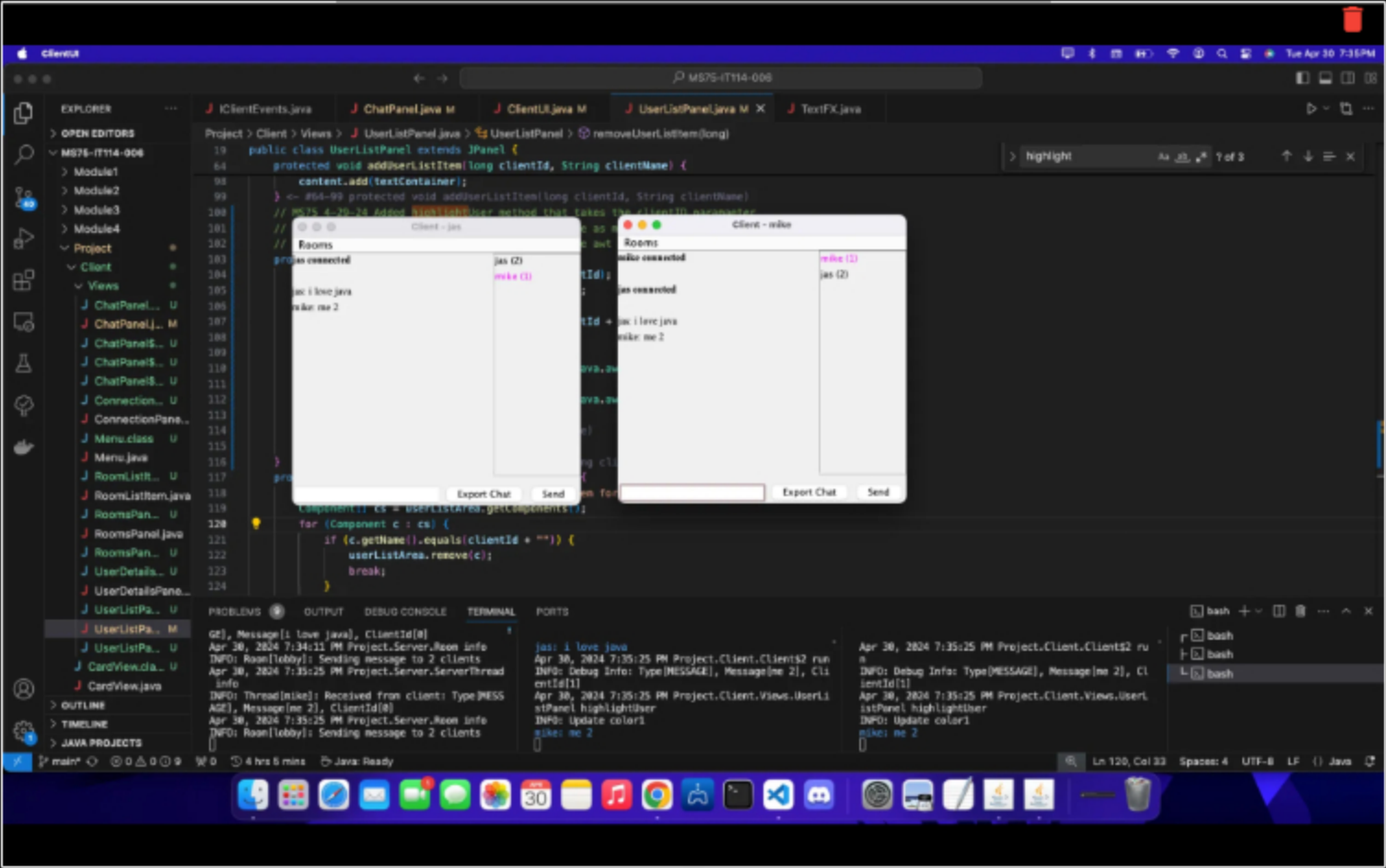
## Checklist Items (0)
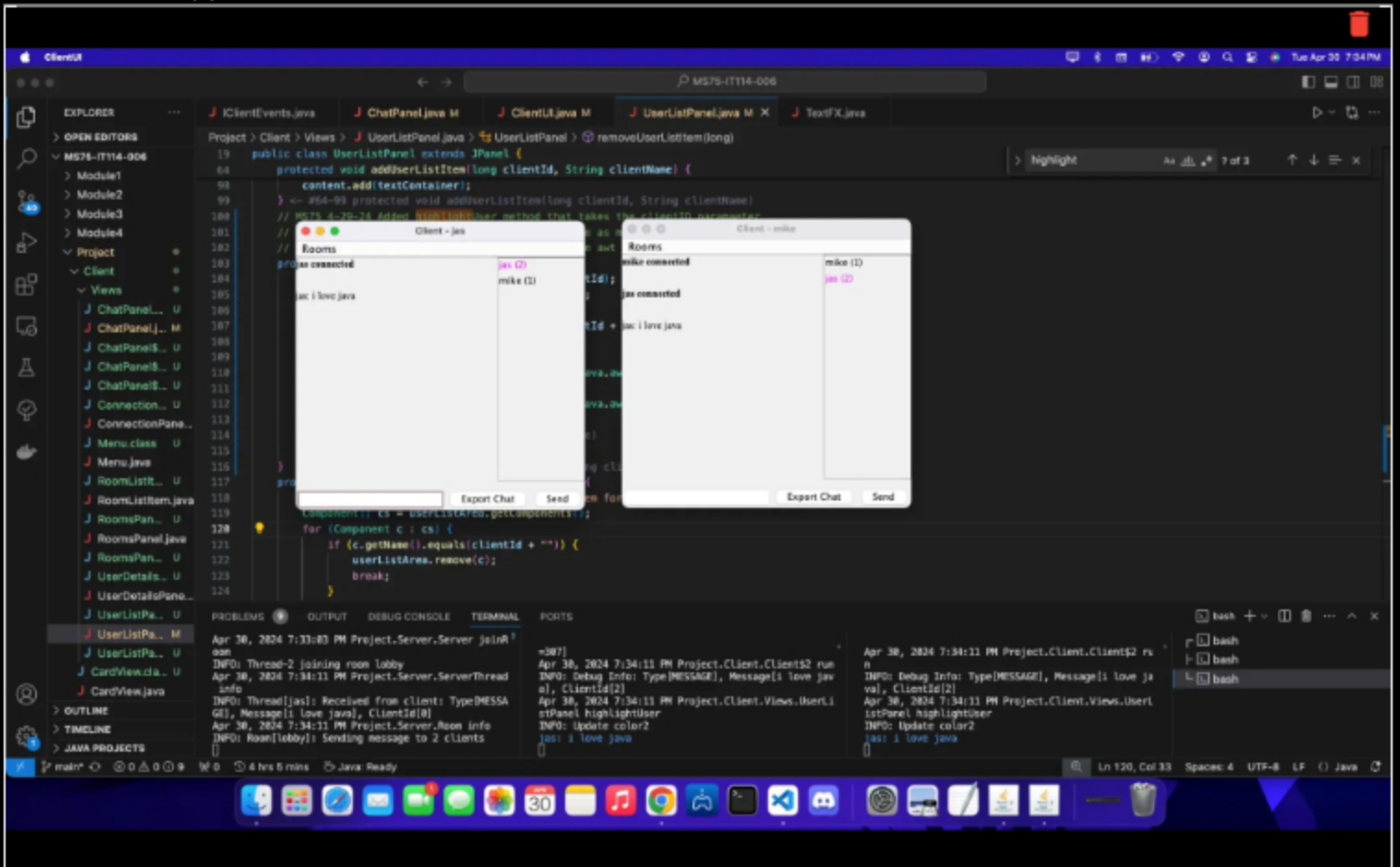


This screenshot shows that the client named Jas sent the first message in the room saying "I love java" and her name turned purple in the list of users.

## Checklist Items (1)

#2 Capture variations of "last person to send a message gets highlighted"



This screenshot shows Mike replying to Jas's message and saying "me 2" and now Mike's name in the user list panel is purple.

## Checklist Items (1)

#2 Capture variations of "last person to send a message gets highlighted"

∧COLLAPSE ∧

### Task #3 - Points: 1

**Text: Explain solution**

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

| # | Points | Details |
|---|--------|---------|
| ☐ #1 | 1 | Mention how you got the mute/unmute effect implemented |
| ☐ #2 | 1 | Mentioned how you got the highlight effect implemented (including unhighlighting the other users) |

Response:

I had no progress on my mute feature since milestone 3 where I could not get it to work. I uploaded the attempted code in Milestone 3.

## Misc (1 pt.)

∧COLLAPSE ∧

**Task #1 - Points: 1**

**Text: Add the pull request link for the branch**

∧COLLAPSE ∧

ℹ️**Details:**

Note: the link should end with /pull/#

URL #1

https://github.com/mikes1302/MS75-IT114-006/pull/17

**Task #2 - Points: 1**

**Text: Talk about any issues or learnings during this assignment**

∧COLLAPSE ∧

Response:

I had issues using the textFX colors for the highlight user name feature, I ended up using javas AWT library instead. I also made no progress on the Mute and Private message features, no code was changed from Milestone3 to 4 for those 2 features. I commented out the code for Mute and PrivateMessage. I would have tried to change them for Milestone4 but I ran out of time.

**Task #3 - Points: 1**

**Text: WakaTime Screenshot**

∧COLLAPSE ∧

ℹ️**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
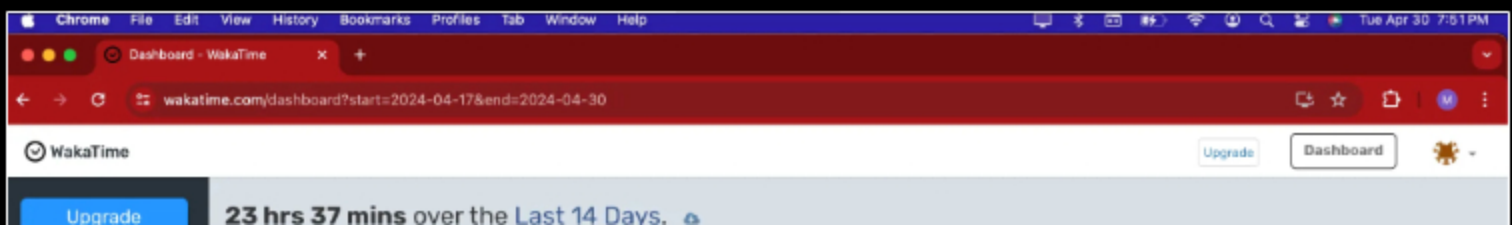
Task Screenshots:

Gallery Style: Large View

Small          Medium          Large

6 hrs 54 mins Mon Apr 29th 2024

MS75-IT114-006 6:28
Milestone2Copy 0:26

6 hrs 54 mins Mon Apr 29th 2024

Java 6:35
Bash 0:19
Text 0:00

**Editors**

VS Code - 23h 37m (100.00%)

VS Code

**Languages**

Java - 22h 56m (97.11%)
Bash - 56m (2.54%)
Markdown - 4m (0.34%)
Text - 0m (0.00%)

Java

**Screenshot shows wakatime time for past 14 days**

**End of Assignment**