Course: IT114-010-S2025
Assignment: IT114 Module 4 Sockets Part3 Challenge
Student: Chihao T. (ct52)
Status: Submitted | Worksheet Progress: 83%
Potential Grade: 8.72/10.00 (87.20%)
Received Grade: 0.00/10.00 (0.00%)
Grading Link: https://learn.ethereallab.app/assignment/v3/IT114-010-S2025/it114-module-4-sockets-part3-challenge/grading/ct52

Instructions

1. Ensure you read all instructions and objectives before starting.
2. Create a new branch from `main` called `M4-Homework`
    1. `git checkout main` (ensure proper starting branch)
    2. `git pull origin main` (ensure history is up to date)
    3. `git checkout -b M4-Homework` (create and switch to branch)
3. Copy the template code from here: GitHub Repository - M4 Homework
    - It includes Sockets `Part1`, `Part2`, and `Part3`. Put all into an `M4` folder or similar if you don't have them yet (adjust `package` reference at the top if you chose a different folder name).
    - Make a copy of `Part3` and call it `Part3HW`
        - [ ] Fix the `package` and `import` references at the top of each file in this new folder
    - Immediately record to history
        - [ ] `git add .`
        - [ ] `git commit -m "adding M4 HW baseline files"`
        - [ ] `git push origin M4-Homework`
        - [ ] Create a Pull Request from `M4-Homework` to `main` and keep it open
4. Fill out the below worksheet
    - Each Problem requires the following as you work
        - [ ] Ensure there's a comment with your UCID, date, and brief summary of how the problem was solved
        - [ ] Code solution (add/commit periodically as needed)
        - [ ] Hint: Note how `/reverse` is handled
5. Once finished, click "Submit and Export"
6. Locally add the generated PDF to a folder of your choosing inside your repository folder and move it to Github
    1. `git add .`
    2. `git commit -m "adding PDF"`
    3. `git push origin M4-Homework`
    4. On Github merge the pull request from `M4-Homework` to `main`
7. Upload the same PDF to Canvas
8. Sync Local
    1. `git checkout main`
    2. `git pull origin main`

Section #1: (3 pts.) Challenge 1 - Coin Flip

Task #1 (3 pts.) - Implement a Coin Flip Command

Combo Task:

Weight: *100%*
Objective: *Implement a Coin Flip Command*

Details:

- `Client` must capture the user entry and generate a valid command per the lesson details
  - Command format must be `/flip`
- `ServerThread` must receive the data and call the correct method on `Server`
- `Server` must expose a method for the logic and send the result to everyone
  - The message must be in the format of `<who> flipped a coin and got <result>` and be from the Server
- Add code to solve the problem (add/commit as needed)

Image Prompt

Weight: *40%*

Details:
Multiple screenshots are expected

1. Snippet of relevant code showing solution (with ucid/date comment) from `Client`
   - Should only need to edit `processClientCommands()`
2. Snippet of relevant code showing solution (with ucid/date comment) from `ServerThread`
   - Should only need to edit `processCommand()`
3. Snippet of relevant code showing solution (with ucid/date comment) from `Server`
   - Should only need to create a new method and pass the result message to `relay()`
4. Show 5 examples of the command being seen across all terminals (2+ Clients and 1 Server)
   1. This can be captured in one screenshot if you split the terminals side by side

```
} else if ("/flip".equalsIgnoreCase(text.trim())) {
    String[] commandData = {Constants.COMMAND_TRIGGER, "flip"};
    sendToServer(String.join(",", commandData));
    wasCommand = true;
}//ct52 4/22
```

Missing Caption

```
case "flip":
    server.handleCoinFlip(this);
    wasCommand = true;
    break;//ct52 4/22
```

Missing Caption

```
public void handleCoinFlip(ServerThread sender) {
    String result = Math.random() < 0.5 ? "Heads" : "Tails";
    String message = "Coin flip result for client" + sender.getClientId() + ": " + result;
    relay(sender.getClientId(), message);
}//ct52 4/22
```

Missing Caption

Saved: 4/25/2025 2:45:30 PM

Url Prompt

Weight: *20%*

Details:
Direct link to the file in the homework related branch from Github (should end in `.java`)

URL #1
https://github.com/ct52/IT114-010-s2025/tree/M4-Homework/M4/Part3HW

url
https://github.com/ct52/IT114-010

Saved: 4/25/2025 2:45:30 PM

Text Prompt

Weight: *40%*

Details:

Briefly explain how the code solves the challenge (note: this isn't the same as what the code does)

Your Response:

What was done was to extend and add code to the existing framework. First, it introduces a new /flip command that will integrate seamlessly with the established client-server protocol. By adding a new case of flipping in the ServerThread, the server can recognize and route the command to the method handleCoinFlip located in the server class. This method can broadcast the result.

Saved: 4/25/2025 2:45:30 PM

Section #2: (3 pts.) Challenge 2 - Private Message

Task #1 (3 pts.) - Implement a Private Message Command

Combo Task:

Weight: *100%*
Objective: *Implement a Private Message Command*

Details:

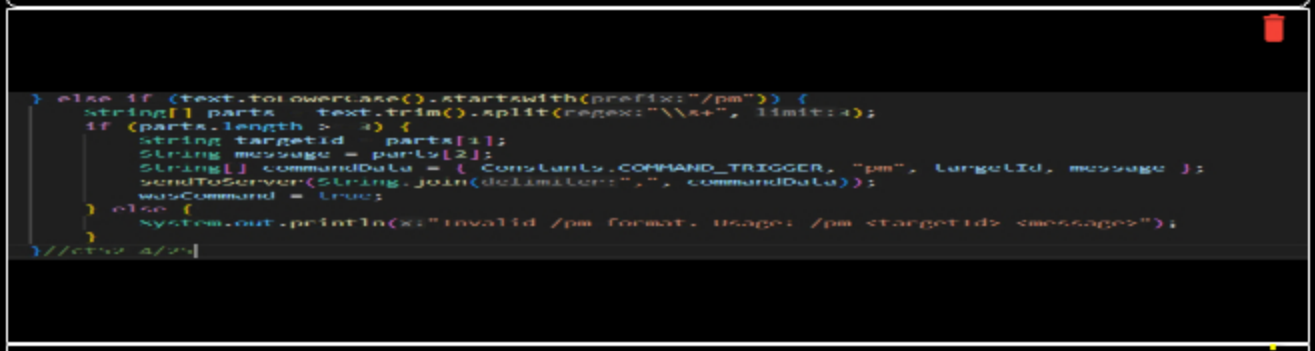
- `Client` must capture the user entry and generate a valid command per the lesson details
  - Command format must be `/pm <target id>`
- `ServerThread` must receive the data and call the correct method on `Server`
- `Server` must expose a method for the logic
  - The message must be in the format of `PM from <who>: <message>` and be from the Server
  - The result must only be sent to the original sender and to the receiver/target
- Add code to solve the problem (add/commit as needed)

Image Prompt

Weight: *40%*

Details:
Multiple screenshots are expected

1. Snippet of relevant code showing solution (with ucid/date comment) from `Client`
    - Should only need to edit `processClientCommands()`
2. Snippet of relevant code showing solution (with ucid/date comment) from `ServerThread`
    - Should only need to edit `processCommand()`
3. Snippet of relevant code showing solution (with ucid/date comment) from `Server`
    - Should only need to create a new method and pass the result message to `relay()`
4. Show 3 examples of the command being seen across all terminals (3+ Clients and 1 Server)
    1. This can be captured in one screenshot if you split the terminals side by side
    2. Note: Only the sender and the receiver should see the private message (show variations across different users)



Missing Caption

Missing Caption

Missing Caption

Saved: 5/1/2025 1:28:25 AM

Url Prompt

Weight: *20%*

Details:
Direct link to the file in the homework related branch from Github (should end in `.java`)

URL #1
https://github.com/ct52/IT114-010-s2025/blob/main/M4/Part3HW/Server.java
Url: https://github.com/ct52/IT114-010

URL #2
https://github.com/ct52/IT114-010-s2025/blob/main/M4/Part3HW/ServerThread.java
Url: https://github.com/ct52/IT114-010

URL #3
https://github.com/ct52/IT114-010-s2025/blob/main/M4/Part3HW/Client.java
Url: https://github.com/ct52/IT114-010

Saved: 5/1/2025 1:28:25 AM

Text Prompt

91%

Section #3: (3 pts.) Challenge 3 - Shuffle Message

91%

Task #1 (3 pts.) - Implement a Shuffle Message Command

Combo Task:

Weight: *100%*
Objective: *Implement a Shuffle Message Command*

Details:

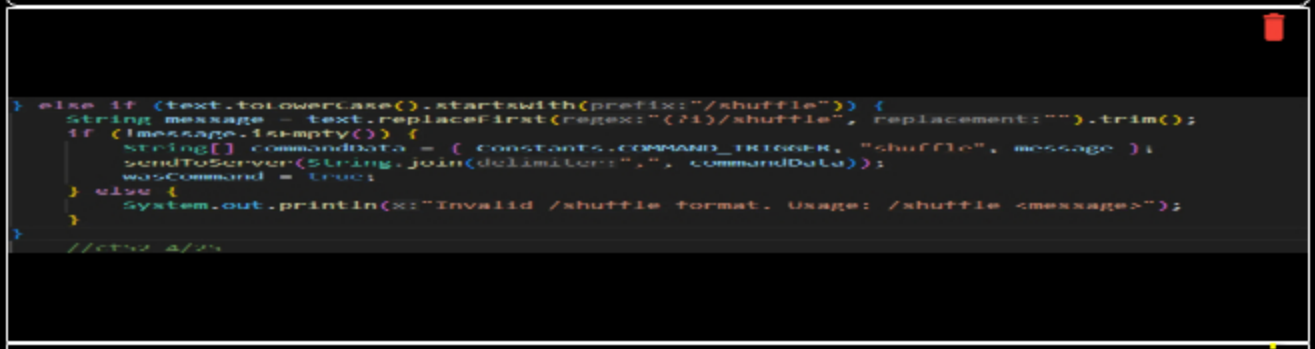
- `Client` must capture the user entry and generate a valid command per the lesson details
  - Command format must be `/shuffle <message>`
- `ServerThread` must receive the data and call the correct method on `Server`
- `Server` must expose a method for the logic and send the result to everyone
  - The message must be in the format of `Shuffled from <who>: <shuffled_message>` and be from the Server
- Add code to solve the problem (add/commit as needed)

Image Prompt

Weight: *40%*

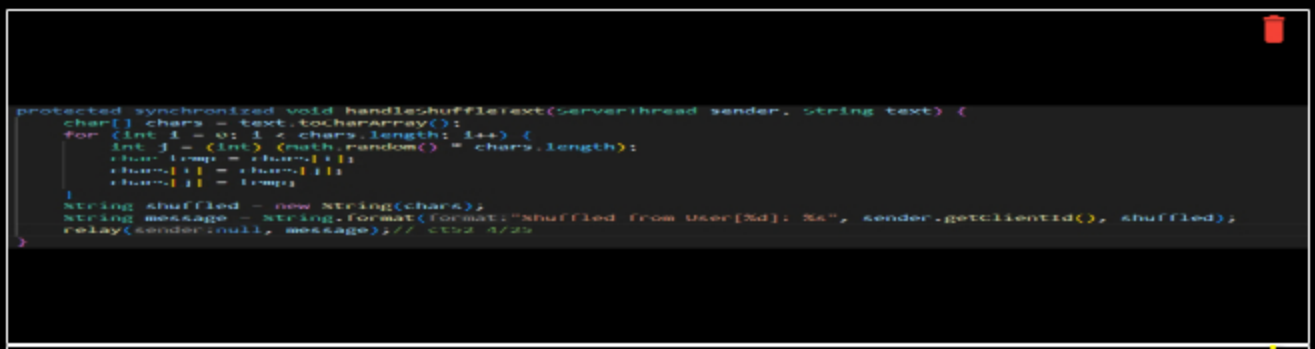
Details:
Multiple screenshots are expected

1. Snippet of relevant code showing solution (with ucid/date comment) from `Client`
   - Should only need to edit `processClientCommands()`
2. Snippet of relevant code showing solution (with ucid/date comment) from `ServerThread`
   - Should only need to edit `processCommand()`
3. Snippet of relevant code showing solution (with ucid/date comment) from `Server`
   - Should only need to create a new method and do similar logic to `relay()`
4. Show 3 examples of the command being seen across all terminals (2+ Clients and 1 Server)
   1. This can be captured in one screenshot if you split the terminals side by side



Missing Caption

Missing Caption



Missing Caption

Saved: 5/1/2025 1:40:19 AM

Url Prompt

Weight: *20%*

Details:
Direct link to the file in the homework related branch from Github (should end in `.java`)

URL #1
https://github.com/ct52/IT114-010-s2025/blob/main/M4/Part3HW/Server.java
Url: https://github.com/ct52/IT114-010

URL #2
https://github.com/ct52/IT114-010-s2025/blob/main/M4/Part3HW/Client.java
Url: https://github.com/ct52/IT114-010

URL #3
https://github.com/ct52/IT114-010-s2025/blob/main/M4/Part3HW/ServerThread.java
Url: https://github.com/ct52/IT114-010

Saved: 5/1/2025 1:40:19 AM

Text Prompt

62%

Section #4: (1 pt.) Misc

87%

Task #1 (0.33 pts.) - Github Details

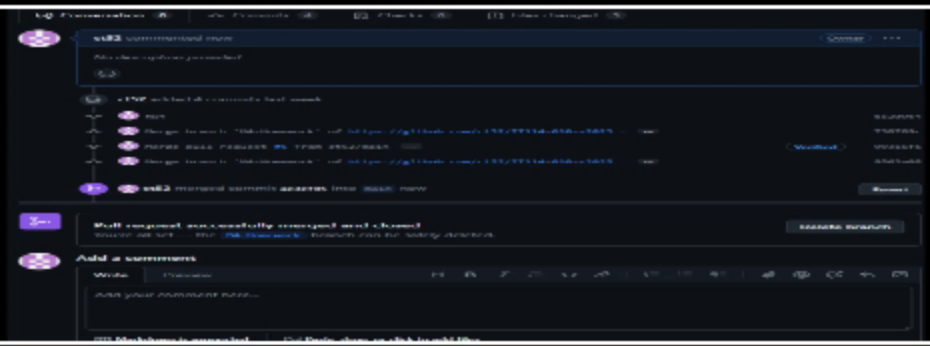
Combo Task:

Weight: *33.33%*
Objective: *Github Details*

Image Prompt

Weight: *60%*

Details:
From the Commits tab of the Pull Request screenshot the commit history Following minimum should be present



Missing Caption

Saved: 5/1/2025 1:43:46 AM

Url Prompt

Weight: *40%*

Details:
Include the link to the Pull Request (should end in `/pull/#`)

URL #1
https://github.com/ct52/IT114-010-s2025/pull/6

Url
https://github.com/ct52/IT114-010

Saved: 5/1/2025 1:43:46 AM

0%

Task #2 (0.33 pts.) - WakaTime - Activity

Image Prompt

Weight: *33.33%*
Objective: *WakaTime - Activity*

Details:

- Visit the WakaTime.com Dashboard

- Visit the WakaTime.com Dashboard
- Click `Projects` and find your repository
- Capture the overall time at the top that includes the repository name
- Capture the individual time at the bottom that includes the file time
- Note: The duration isn't relevant for the grade and the visual graphs aren't necessary

Missing Caption

Not saved yet

100% Task #3 (0.33 pts.) - Reflection

Weight: *33.33%*
Objective: *Reflection*

Sub-Tasks:

100% Task #1 (0.33 pts.) - What did you learn?

Text Prompt

Weight: *33.33%*
Objective: *What did you learn?*

Details:
Briefly answer the question (at least a few decent sentences)

Your Response:

I built a working networked application using sockets, handling input/output streams, and understanding how data flows between multiple clients and a central server for client server communication.

Saved: 5/1/2025 1:45:04 AM

Task #2 (0.33 pts.) - What was the easiest part of the assignr

Text Prompt

Weight: *33.33%*
Objective: *What was the easiest part of the assignment?*

Details:
Briefly answer the question (at least a few decent sentences)

Your Response:

it was writing the flip method

Saved: 5/1/2025 1:45:23 AM

Task #3 (0.33 pts.) - What was the hardest part of the assign

Text Prompt

Weight: *33.33%*
Objective: *What was the hardest part of the assignment?*

Details:
Briefly answer the question (at least a few decent sentences)

Your Response:

iso the game login in server class

Saved: 5/1/2025 1:45:39 AM