Submission Worksheet

CLICK TO GRADE

IT114-002-S2024 - [IT114] Project Milestone 1

Submissions:

Submission Selection

1 Submission [active] 3/18/2024 1:42:05 AM

Instructions

^ COLLAPSE ^

Create a new branch called Milestone1
At the root of your repository create a folder called Project if one doesn't exist yet
    You will be updating this folder with new code as you do milestones
    You won't be creating separate folders for milestones; milestones are just branches
Create a pull request from Milestone1 to main (don't complete/merge it yet, just have it in open status)
Copy in the latest Socket sample code from the most recent Socket Part example of the lessons
    Recommended Part 5 (clients should be having names at this point and not ids)
    https://github.com/MattToegel/IT114/tree/Module5/Module5
Fix the package references at the top of each file (these are the only edits you should do at this point)
Git add/commit the baseline and push it to github
Create a pull request from Milestone1 to main (don't complete/merge it yet, just have it in open status)
Ensure the sample is working and fill in the below deliverables
    Note: The client commands likely are different in part 5 with the /name and /connect options instead of just "connect"
Generate the worksheet output file once done and add it to your local repository
Git add/commit/push all changes
Complete the pull request merge from step 7
Locally checkout main
git pull origin main

Branch name: Milestone1

Tasks: 9 Points: 10.00

● Start Up (3 pts.)
^COLLAPSE^

Task #1 - Points: 1

Text: Server and Client Initialization

Checklist

*The checkboxes are for your own tracking

#	Points	Details
☐ #1	1	Server should properly be listening to its port from the command line (note the related message)
☐ #2	1	Clients should be successfully waiting for input
☐ #3	1	Clients should have a name and successfully connected to the server (note related messages)

Task Screenshots:

Gallery Style: Large View

Small   Medium   Large

the screen shot

Checklist Items (0)

Task #2 - Points: 1

Text: Explain the connection process

ⓘ Details:

Note the various steps from the beginning to when the client is fully connected and able to communicate in the room.

Emphasize the code flow and the sockets usage.

#	Points	Details
☐ #1	1	Mention how the server-side of the connection works
☐ #2	1	Mention how the client-side of the connection works
☐ #3	1	Describe the socket steps until the server is waiting for messages from the client

Response:

In order to connect the client and the server you must first make sure each part is running then from the server type connect localhost:3000, localhost is how it sounds and means you are hosting it locally and the 3000 part is a port number that is usually available. next name the two clients and then connect both to the same port as the server.

● **Communication (3 pts.)**

^COLLAPSE ^

●

^COLLAPSE ^

Task #1 - Points: 1

Text: Add screenshot(s) showing evidence related to the checklist

#	Points	Details
☐ #1	1	At least two clients connected to the server
☐ #2	1	Client can send messages to the server
☐ #3	1	Server sends the message to all clients in the same room
☐ #4	1	Messages clearly show who the message is from (i.e., client name is clearly with the message)
☐ #5	2	Demonstrate clients in two different rooms can't send/receive messages to each other (clearly show the clients are in different rooms via the commands demonstrated in the lessons
☐ #6	1	Clearly caption each image regarding what is being shown

Task Screenshots:

Gallery Style: Large View

Small   Medium   Large

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL   PORTS   GITLENS

```
                          Thread-2 leaving room Lobby                                      at java.base/java.net.Socket.<
                          Thread[31]: Thread starting        merca@pear MINGW64 ~/am3485-it114-002 (   init>(Socket.java:324)
                          Thread-2 joining room Lobby        module5)                                   at M5.Client.connect(Client.ja
merca@pear MINGW64 ~/am3485-it114-002 (
```

```
module5)                          Thread[31]: Received from client: Type[        $ java M5/Client                    va:47)
$ java M5/Client                  CONNECT], Number[0], Message[null]         .                                            at M5.Client.processCommand(Cl
                            .     waiting for next client                    Listening for input                    ient.java:120)                    .
Listening for input               Client connected                          Waiting for input                             at M5.Client$1.run(Client.java
Waiting for input                 Thread[33]: Thread created                 /name Bill                             :160)
/connect localhost:3000           Thread-4 leaving room Lobby                Name set to Bill                       Waiting for input
You must set your name before you can c   Thread[33]: Thread starting        Waiting for input                      /connect localhost:3000
Users\merca\am3485-it114-002\M5\Client.class   Thread-4 joining room Lobby   /connect localhost:3000                Client connected
                                  Thread[33]: Received from client: Type[    Client connected                       Waiting for input
/name Alan                        CONNECT], Number[0], Message[null]         Waiting for input                      Debug Info: Type[CONNECT], Number[0],
Name set to Alan                  Thread[31]: Received from client: Type[    Debug Info: Type[CONNECT], Number[0], M  Message[connected]
Waiting for input                 MESSAGE], Number[0], Message[/createroo    essage[connected]                      *Bob connected*
/connect localhost:3000       .   m big']                                    *Bill connected*                       Debug Info: Type[DISCONNECT], Number[0
Client connected                  Room[Lobby]: Sending message to 3 clien    Debug Info: Type[DISCONNECT], Number[0]  ], Message[disconnected]
Waiting for input                 ts                                         , Message[disconnected]                *null disconnected*
Debug Info: Type[CONNECT], Number[0], M   Created new room: big'             *Alan disconnected*                    Debug Info: Type[CONNECT], Number[0],
essage[connected]                 Thread-2 leaving room Lobby                /joinroom big'                         Message[connected]
*Alan connected*                  Thread-2 joining room big'                 Waiting for input                      *Alan connected*
Debug Info: Type[DISCONNECT], Number[0]   Thread[33]: Received from client: Type[  Debug Info: Type[CONNECT], Number[0], M  Debug Info: Type[DISCONNECT], Number[0
, Message[disconnected]           MESSAGE], Number[0], Message[/joinroom    essage[connected]                      ], Message[disconnected]
*null disconnected*               big']                                      *Bill connected*                       *null disconnected*
Debug Info: Type[CONNECT], Number[0], M   Room[Lobby]: Sending message to 2 clien  hey all whats up                       Debug Info: Type[CONNECT], Number[0],
essage[connected]                 ts                                         Waiting for input                      Message[connected]
*Bill connected*                  Thread-4 leaving room Lobby                Debug Info: Type[MESSAGE], Number[0], M  *Bill connected*
/createroom big'                  Thread-4 joining room big'                 essage[hey all whats up]               Debug Info: Type[DISCONNECT], Number[0
Waiting for input                 Thread[33]: Received from client: Type[    Bill: hey all whats up  .              ], Message[disconnected]
Debug Info: Type[CONNECT], Number[0], M   MESSAGE], Number[0], Message[hey all wh  []                    .             *Alan disconnected*
essage[connected]                 ats up]                                                                            Debug Info: Type[DISCONNECT], Number[0
*Alan connected*                  Room[big']: Sending message to 2 client                                           ], Message[disconnected]
Debug Info: Type[CONNECT], Number[0], M   s                                                                          *Bill disconnected*  . .
essage[connected]                 Thread[29]: Received from client: Type[                                            where'd everyone go   .
*Bill connected*          .       MESSAGE], Number[0], Message[where'd ev                                            Waiting for input
Debug Info: Type[MESSAGE], Number[0], M   eryone go]                                                                 Debug Info: Type[MESSAGE], Number[0],
essage[hey all whats up]          Room[Lobby]: Sending message to 1 clien                                            Message[where'd everyone go]
Bill: hey all whats up            ts                                                                                 Bob: where'd everyone go
[]                                []
```

red = message shown blue = proves connection green = shows messages being sent from inside the same room yellow = shows different rooms can't see each others messages

Checklist Items (0)

●

Task #2 - Points: 1
Text: Explain the communication process

ⓘ Details:
How are messages entered from the client side and how do they propagate to other clients?

Note all the steps involved and use specific terminology from the code.
Don't just translate the code line-by-line to plain English, keep it concise.

Checklist   *The checkboxes are for your own tracking

#	Points	Details
☐ #1	1	Mention the client-side (sending)
☐ #2	1	Mention the ServerThread's involvement
☐ #3	1	Mention the Room's perspective
☐ #4	1	Mention the client-side (receiving)

Response:

from the client side all that you have to do is type something out and press enter, on the server thread side its forwards the message so people can see it, on the room side it makes it so that only people in the same room can see a message, and the receiving side they are displayed the message and who its from.

●

∧COLLAPSE ∧

Task #1 - Points: 1

Text: Add screenshot(s) showing evidence related to the checklist

Checklist   *The checkboxes are for your own tracking

#	Points	Details
☐ #1	1	Show a client disconnecting from the server; Server should still be running without issue (it's ok if an exception message shows as it's part of the lesson code, the server just shouldn't terminate)
☐ #2	1	Show the server terminating; Clients should be disconnected but still running and able to reconnect when the server is back online (demonstrate this)
☐ #3	1	For each scenario, disconnected messages should be shown to the clients (should show a different person disconnected and should show the specific client disconnected)
☐ #4	1	Clearly caption each image regarding what is being shown

Task Screenshots:

Gallery Style: Large View

Small   Medium   Large

red disconnected from yellow

Checklist Items (0)

when server disconnects

Checklist Items (0)

●

^COLLAPSE^

Task #2 - Points: 1
Text: Explain the various Disconnect/termination scenarios

ⓘ Details:
Include the various scenarios of how a disconnect can occur. There should be around 3 or so.

Checklist   *The checkboxes are for your own tracking

#	Points	Details
☐ #1	1	Mention how a client gets disconnected from a Socket perspective
☐ #2	1	Mention how/why the client program doesn't crash when the server disconnects/terminates.
☐ #3	1	Mention how the server doesn't crash from the client(s) disconnecting

Response:

the scenarios of a disconnect are quiting the client, terminating the server, or quitting the client

Misc (1 pt.)

∧COLLAPSE ∧

∧COLLAPSE ∧

Task #1 - Points: 1

Text: Add the pull request link for this branch

URL #1

https://github.com/alanpear/am3485-it114-002/pull/10

∧COLLAPSE ∧

Task #2 - Points: 1

Text: Talk about any issues or learnings during this assignment

ⓘ Details:

Few related sentences about the Project/sockets topics

Response:

I had no issues with this milestone it was more of an assignment that helped me get familiar with the process and helped me recount what i learned in class.

∧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.
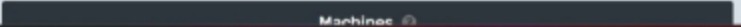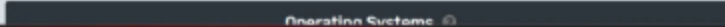
Task Screenshots:
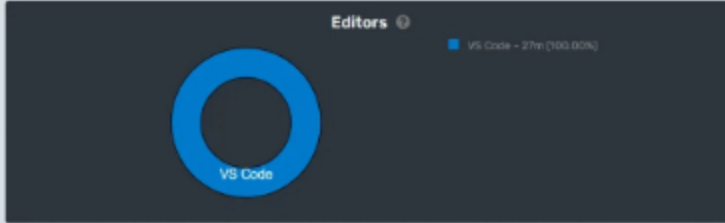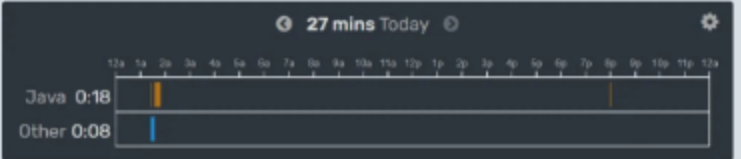
Gallery Style: Large View

Small   Medium   Large

27 mins over the Last 7 Days. ⓐ

Wed Mar 13th 2024
Total   0s

27 mins Today ⚙

am3485-it114-002 0:27

12a 1a 2a 3a 4a 5a 6a 7a 8a 9a 10a 11a 12p 1p 2p 3p 4p 5p 6p 7p 8p 9p 10p 11p 12a

27 mins Today ⚙

Java 0:18

Other 0:08

12a 1a 2a 3a 4a 5a 6a 7a 8a 9a 10a 11a 12p 1p 2p 3p 4p 5p 6p 7p 8p 9p 10p 11p 12a

Editors ⓘ

VS Code – 27m (100.00%)

VS Code

Languages ⓘ

Java – 19m (69.32%)
Other – 8m (30.68%)

Other

Java

Operating Systems ⓘ   Machines ⓘ
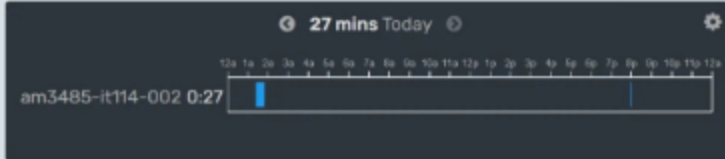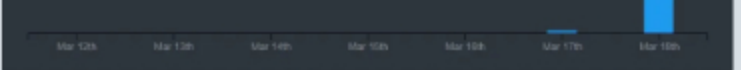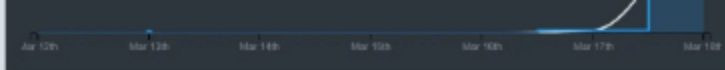
waka time ss

End of Assignment