Submission Worksheet

CLICK TO GRADE

https://learn.ethereallab.app/assignment/IT114-004-S2024/it114-project-milestone-1/grade/grg

IT114-004-S2024 - [IT114] Project Milestone 1

Submissions:

Submission Selection

1 Submission [active] 3/24/2024 5:50:16 PM

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

COLLAPSE

Task #1 - Points: 1

Text: Server and Client Initialization

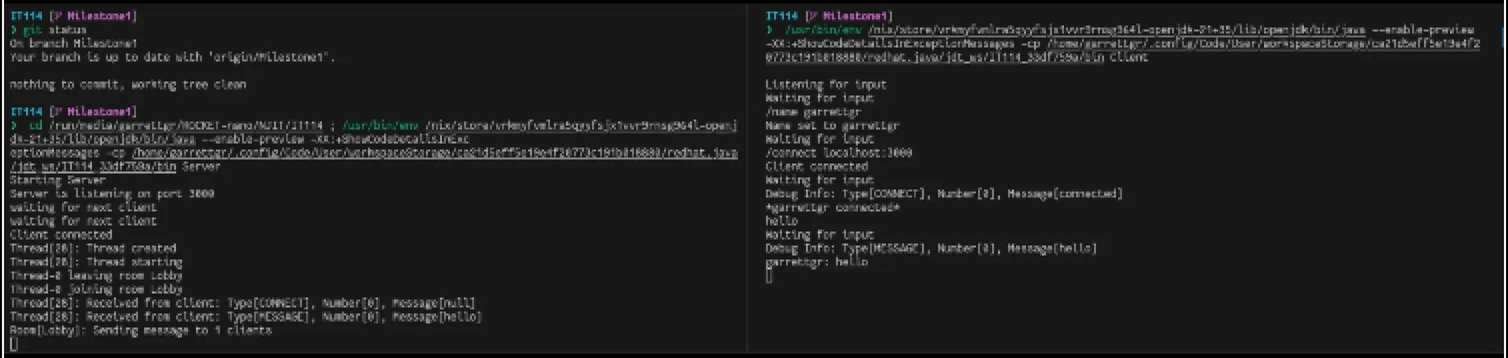
Checklist *The checkboxes are for your own tracking

#	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



```
IT114 [ Milestone1]
> git status
On branch Milestone1
Your branch is up to date with 'origin/Milestone1'.

nothing to commit, working tree clean

IT114 [ Milestone1]
>  cd /run/media/garrettgr/ROCKET-nano/NJIT/IT114 ; /usr/bin/env /nix/store/vrkmyfvmlra5qysfsjx1vvr9rnag364l-openj
dk-21+35/lib/openjdk/bin/java --enable-preview -XX:+ShowCodeDetailsInExc
eptionMessages -cp /home/garrettgr/.config/Code/User/workspaceStorage/ca21d5eff5e19e4f20773c191b818880/redhat.java
/jdt_ws/IT114_33df759a/bin Server
Starting Server
Server is listening on port 3000
waiting for next client
waiting for next client
Client connected
Thread[28]: Thread created
Thread[28]: Thread starting
Thread-0 leaving room Lobby
Thread-0 joining room Lobby
Thread[28]: Received from client: Type[CONNECT], Number[0], Message[null]
Thread[28]: Received from client: Type[MESSAGE], Number[0], Message[hello]
Room[Lobby]: Sending message to 1 clients
```

```
IT114 [ Milestone1]
>  /usr/bin/env /nix/store/vrkmyfvmlra5qyyfsjs1vvr9rnag364l-openjdk-21+35/lib/openjdk/bin/java --enable-preview
-XX:+ShowCodeDetailsInExceptionMessages -cp /home/garrettgr/.config/Code/User/workspaceStorage/ca21d5eff5e19e4f2
0773c191b818880/redhat.java/jdt_ws/IT114_33df759a/bin Client

Listening for input
Waiting for input
/name garrettgr
Name set to garrettgr
Waiting for input
/connect localhost:3000
Client connected
Waiting for input
Debug Info: Type[CONNECT], Number[0], Message[connected]
*garrettgr connected*
hello
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[hello]
garrettgr: hello
```

Server & Clients being started, connecting Client to server

Checklist Items (0)

COLLAPSE

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.

Checklist *The checkboxes are for your own tracking

#	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:

The server starts by creating a 'ServerSocket' object and listens for client connections request on a specific port (3000 by default)

```
ServerSocket serverSocket = new ServerSocket(portNumber);
```

The server then enters a loop waiting for a client connection request -- when it receives a valid request, it creates a new 'Socket' object and calls the 'accept' method

```
Socket clientSocket = serverSocket.accept();
```

As for the client, a 'Socket' object is created with the server's hostname and port, which both initiates the connection request to the server and handles communication if that request is accepted

```
Socket socket = new Socket(hostName, portNumber);
```

After communication is established, both the client and server create input and output streams to send and receive data (text data or objects)

```
PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
BufferedReader in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
```

The server generally waits for a message from the client, processes it, and sends a response back -- this loop continues until the client disconnects or the server is terminated.

COLLAPSE

Communication (3 pts.)

COLLAPSE

Task #1 - Points: 1

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

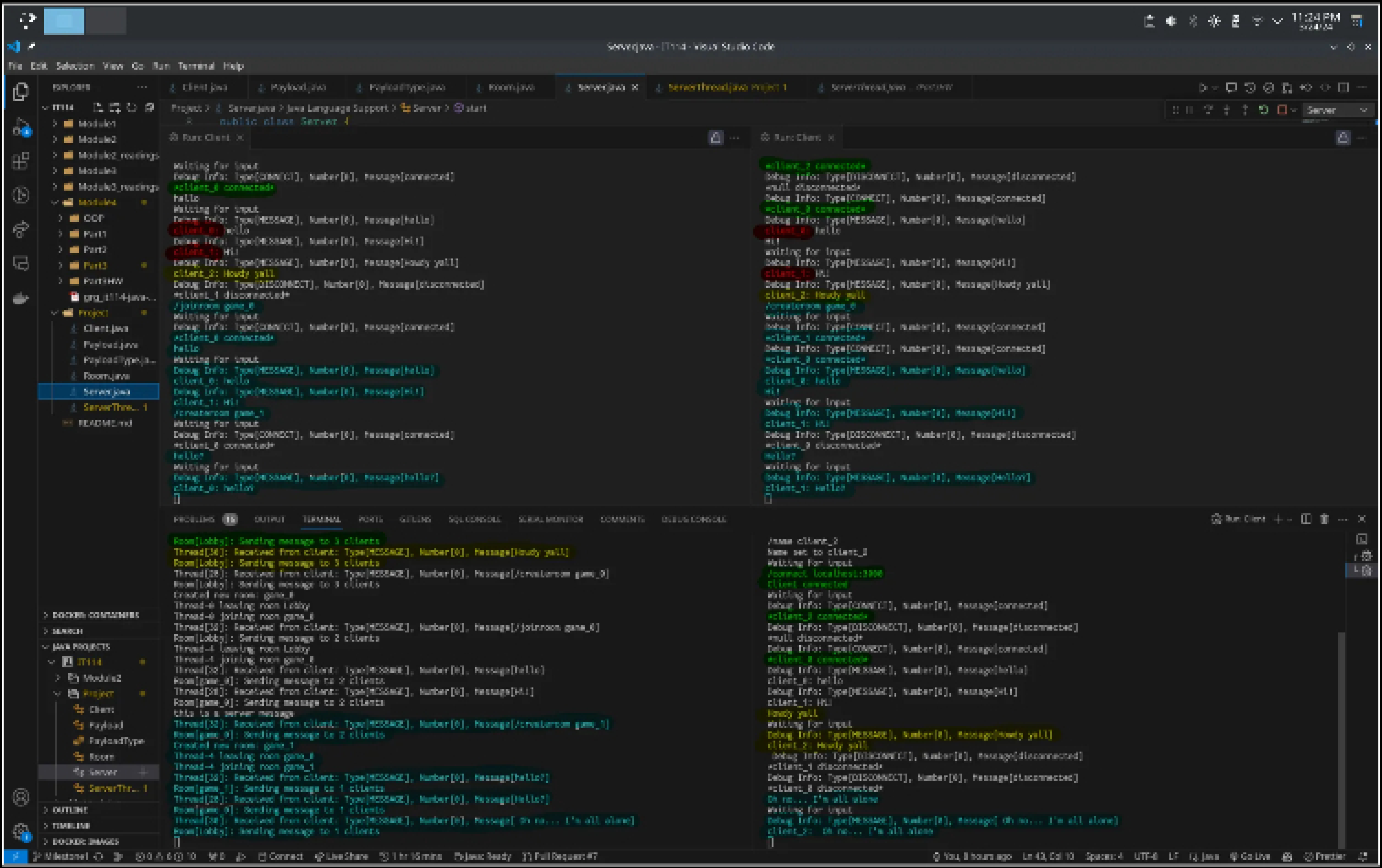
Checklist *The checkboxes are for your own tracking

#	Points	Details
☐ #1	1	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



```
Waiting for input
Debug Info: Type[CONNECT], Number[0], Message[connected]
*client_0 connected*
hello
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[hello]
client_0: hello
Debug Info: Type[MESSAGE], Number[0], Message[Hi!]
client_1: Hi!
Debug Info: Type[MESSAGE], Number[0], Message[Howdy yall]
client_2: Howdy yall
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
*client_1 disconnected*
/joinroom game_0
Waiting for input
Debug Info: Type[CONNECT], Number[0], Message[connected]
*client_0 connected*
hello
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[hello]
client_0: hello
Debug Info: Type[MESSAGE], Number[0], Message[Hi!]
client_1: Hi!
/createroom game_1
Waiting for input
Debug Info: Type[CONNECT], Number[0], Message[connected]
*client_0 connected*
hello?
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[hello?]
client_0: hello?
```

```
*client_2 connected*
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
*null disconnected*
Debug Info: Type[CONNECT], Number[0], Message[connected]
*client_0 connected*
Debug Info: Type[MESSAGE], Number[0], Message[hello]
client_0: hello
Hi!
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[Hi!]
client_1: Hi!
Debug Info: Type[MESSAGE], Number[0], Message[Howdy yall]
client_2: Howdy yall
/createroom game_0
Waiting for input
Debug Info: Type[CONNECT], Number[0], Message[connected]
*client_1 connected*
Debug Info: Type[CONNECT], Number[0], Message[connected]
*client_0 connected*
Debug Info: Type[MESSAGE], Number[0], Message[hello]
client_0: hello
Hi!
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[Hi!]
client_1: Hi!
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
*client_0 disconnected*
Hello?
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[Hello?]
client_1: Hello?
```

```
Room[Lobby]: Sending message to 3 clients
Thread[30]: Received from client: Type[MESSAGE], Number[0], Message[Howdy yall]
Room[Lobby]: Sending message to 3 clients
Thread[28]: Received from client: Type[MESSAGE], Number[0], Message[/createroom game_0]
Room[Lobby]: Sending message to 3 clients
Created new room: game_0
Thread-0 leaving room Lobby
Thread-0 joining room game_0
Thread[30]: Received from client: Type[MESSAGE], Number[0], Message[/joinroom game_0]
Room[Lobby]: Sending message to 2 clients
Thread-4 leaving room Lobby
Thread-4 joining room game_0
Thread[32]: Received from client: Type[MESSAGE], Number[0], Message[hello]
Room[game_0]: Sending message to 2 clients
Thread[28]: Received from client: Type[MESSAGE], Number[0], Message[Hi!]
Room[game_0]: Sending message to 2 clients
this is a server message
Thread[32]: Received from client: Type[MESSAGE], Number[0], Message[/createroom game_1]
Room[game_0]: Sending message to 2 clients
Created new room: game_1
Thread-4 leaving room game_0
Thread-4 joining room game_1
Thread[32]: Received from client: Type[MESSAGE], Number[0], Message[hello?]
Room[game_1]: Sending message to 1 clients
Thread[28]: Received from client: Type[MESSAGE], Number[0], Message[Hello?]
Room[game_0]: Sending message to 1 clients
Thread[30]: Received from client: Type[MESSAGE], Number[0], Message[ Oh no... I'm all alone]
Room[Lobby]: Sending message to 1 clients
```

```
/name client_2
Name set to client_2
Waiting for input
/connect localhost:3000
Client connected
Waiting for input
Debug Info: Type[CONNECT], Number[0], Message[connected]
*client_2 connected*
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
*null disconnected*
Debug Info: Type[CONNECT], Number[0], Message[connected]
*client_0 connected*
Debug Info: Type[MESSAGE], Number[0], Message[hello]
client_0: hello
Debug Info: Type[MESSAGE], Number[0], Message[Hi!]
client_1: Hi!
Howdy yall
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[Howdy yall]
client_2: Howdy yall
 Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
*client_1 disconnected*
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
*client_0 disconnected*
Oh no... I'm all alone
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[ Oh no... I'm all alone]
client_2:  Oh no... I'm all alone
```

#1 - Green highlight #2 - Yellow highlight #3 - Yellow highlight #4 - Red highlight #5 - Blue highlight

Checklist Items (0)

COLLAPSE

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:

The client sends a message by creating a 'Payload' object', settings its type to 'MESSAGE' and setting the content. This is then sent to the server using the 'ObjectOutputStream' associated with the client's socket

On the server side, each client is associated with a 'ServerThread' object -- this thread is responsible for reading the incoming 'Payload' object -- when the 'payload's type is message, the processMessage method is called -- which then calls the 'sendMethod' method on the 'Room' that the client is currently in, passing itself and the message content

The 'Room' class represents a chat room -- it has a method 'sendMessage' used for broadcasting messages to all clients in that room.

Once a message is sent, the client recieves a 'payload' object on its 'ObjectInputStream' -- the client then extracts the message's content from the 'Payload' and displays it to the user

● **Disconnecting/Termination** (3 pts.)

COLLAPSE

● COLLAPSE

Task #1 - Points: 1

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

Checklist *The checkboxes are for your own tracking

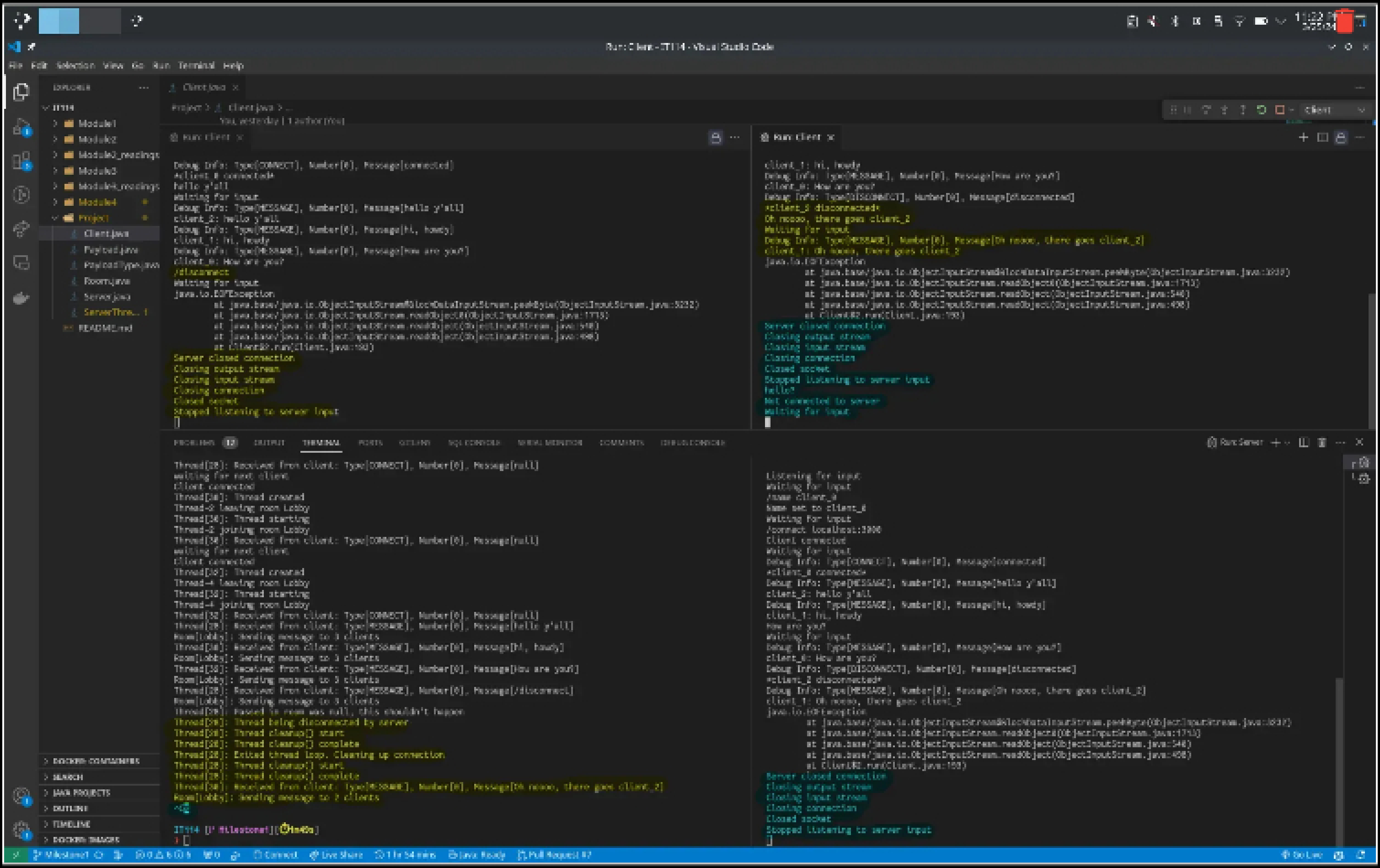
#	Points	Details
☐	1	Show a client disconnecting from the server; Server should still be running without issue (it's ok if an

#1	1	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



```
Debug Info: Type[CONNECT], Number[0], Message[connected]
*client_0 connected*
hello y'all
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[hello y'all]
client_2: hello y'all
Debug Info: Type[MESSAGE], Number[0], Message[hi, howdy]
client_1: hi, howdy
Debug Info: Type[MESSAGE], Number[0], Message[How are you?]
client_0: How are you?
/disconnect
Waiting for input
java.io.EOFException
        at java.base/java.io.ObjectInputStream$BlockDataInputStream.peekByte(ObjectInputStream.java:3222)
        at java.base/java.io.ObjectInputStream.readObject0(ObjectInputStream.java:1715)
        at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:540)
        at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:498)
        at Client$2.run(Client.java:192)
Server closed connection
Closing output stream
Closing input stream
Closing connection
Closed socket
Stopped listening to server input
```

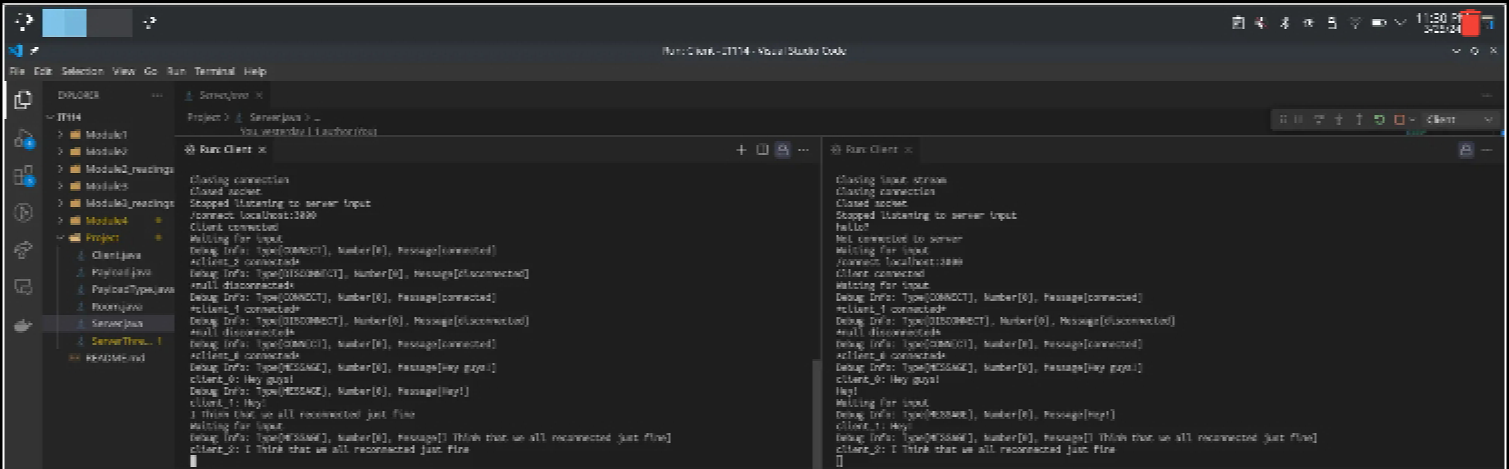
```
client_1: hi, howdy
Debug Info: Type[MESSAGE], Number[0], Message[How are you?]
client_0: How are you?
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
*client_2 disconnected*
Oh noooo, there goes client_2
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[Oh noooo, there goes client_2]
client_1: Oh noooo, there goes client_2
java.io.EOFException
        at java.base/java.io.ObjectInputStream$BlockDataInputStream.peekByte(ObjectInputStream.java:3237)
        at java.base/java.io.ObjectInputStream.readObject0(ObjectInputStream.java:1713)
        at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:540)
        at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:498)
        at Client$2.run(Client.java:192)
Server closed connection
Closing output stream
Closing input stream
Closing connection
Closed socket
Stopped listening to server input
hello?
Not connected to server
Waiting for input
```

```
Thread[28]: Received from client: Type[CONNECT], Number[0], Message[null]
waiting for next client
Client connected
Thread[30]: Thread created
Thread-2 leaving room Lobby
Thread[30]: Thread starting
Thread-2 joining room Lobby
Thread[30]: Received from client: Type[CONNECT], Number[0], Message[null]
waiting for next client
Client connected
Thread[32]: Thread created
Thread-4 leaving room Lobby
Thread[32]: Thread starting
Thread-4 joining room Lobby
Thread[32]: Received from client: Type[CONNECT], Number[0], Message[null]
Thread[28]: Received from client: Type[MESSAGE], Number[0], Message[hello y'all]
Room[Lobby]: Sending message to 3 clients
Thread[30]: Received from client: Type[MESSAGE], Number[0], Message[hi, howdy]
Room[Lobby]: Sending message to 3 clients
Thread[32]: Received from client: Type[MESSAGE], Number[0], Message[How are you?]
Room[Lobby]: Sending message to 3 clients
Thread[28]: Received from client: Type[MESSAGE], Number[0], Message[/disconnect]
Room[Lobby]: Sending message to 3 clients
Thread[28]: Passed in room was null, this shouldn't happen
Thread[28]: Thread being disconnected by server
Thread[28]: Thread cleanup() start
Thread[28]: Thread cleanup() complete
Thread[28]: Exited thread loop. Cleaning up connection
Thread[28]: Thread cleanup() start
Thread[28]: Thread cleanup() complete
Thread[30]: Received from client: Type[MESSAGE], Number[0], Message[Oh noooo, there goes client_2]
Room[Lobby]: Sending message to 2 clients
^C
IT114 [Milestone1][4m40s]
> 
```

```
Listening for input
Waiting for input
/name client_0
Name set to client_0
Waiting for input
/connect localhost:3000
Client connected
Waiting for input
Debug Info: Type[CONNECT], Number[0], Message[connected]
*client_0 connected*
Debug Info: Type[MESSAGE], Number[0], Message[hello y'all]
client_2: hello y'all
Debug Info: Type[MESSAGE], Number[0], Message[hi, howdy]
client_1: hi, howdy
How are you?
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[How are you?]
client_0: How are you?
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
*client_2 disconnected*
Debug Info: Type[MESSAGE], Number[0], Message[Oh noooo, there goes client_2]
client_1: Oh noooo, there goes client_2
java.io.EOFException
        at java.base/java.io.ObjectInputStream$BlockDataInputStream.peekByte(ObjectInputStream.java:3237)
        at java.base/java.io.ObjectInputStream.readObject0(ObjectInputStream.java:1713)
        at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:540)
        at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:498)
        at Client$2.run(Client.java:192)
Server closed connection
Closing output stream
Closing input stream
Closing connection
Closed socket
Stopped listening to server input
```

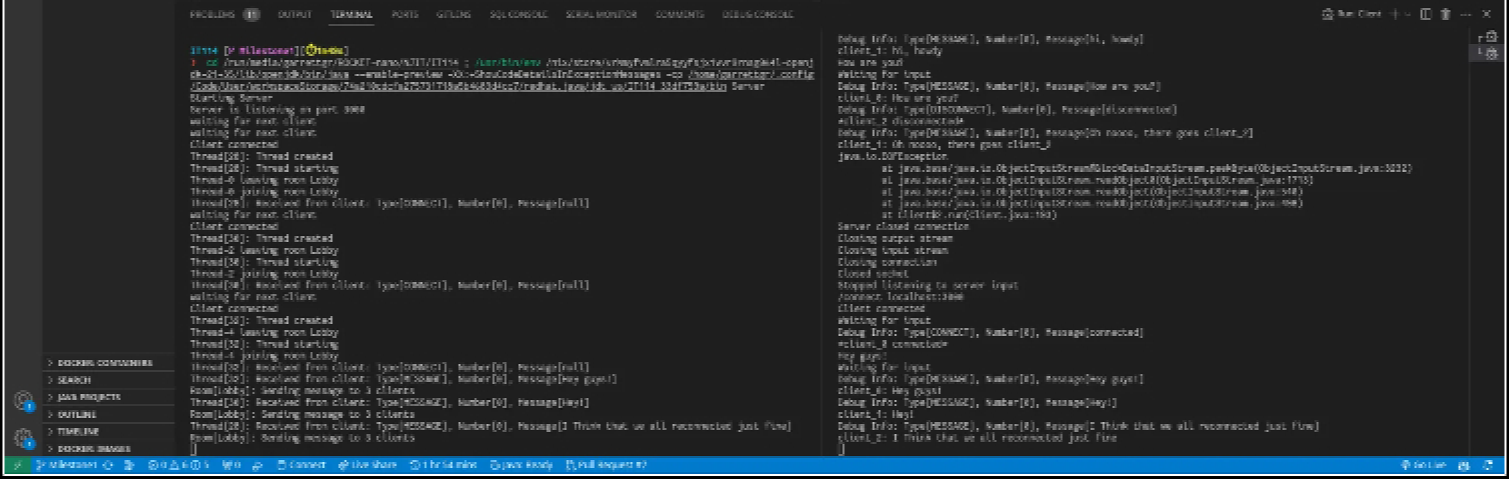
Shows 1 client disconnecting and the server continuing to run normally (yellow) -- shows the server being terminated and the clients getting a disconnect message

Checklist Items (0)



```
Closing connection
Closed socket
Stopped listening to server input
/connect localhost:3000
Client connected
Waiting for input
Debug Info: Type[CONNECT], Number[0], Message[connected]
*client_2 connected*
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
*null disconnected*
Debug Info: Type[CONNECT], Number[0], Message[connected]
*client_1 connected*
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
*null disconnected*
Debug Info: Type[CONNECT], Number[0], Message[connected]
*client_0 connected*
Debug Info: Type[MESSAGE], Number[0], Message[Hey guys!]
client_0: Hey guys!
Debug Info: Type[MESSAGE], Number[0], Message[Hey!]
client_1: Hey!
I Think that we all reconnected just fine
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[I Think that we all reconnected just fine]
client_2: I Think that we all reconnected just fine
```

```
Closing input stream
Closing connection
Closed socket
Stopped listening to server input
hello?
Not connected to server
Waiting for input
/connect localhost:3000
Client connected
Waiting for input
Debug Info: Type[CONNECT], Number[0], Message[connected]
*client_1 connected*
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
*null disconnected*
Debug Info: Type[CONNECT], Number[0], Message[connected]
*client_0 connected*
Debug Info: Type[MESSAGE], Number[0], Message[Hey guys!]
client_0: Hey guys!
Hey!
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[Hey!]
client_1: Hey!
Debug Info: Type[MESSAGE], Number[0], Message[I Think that we all reconnected just fine]
client_2: I Think that we all reconnected just fine
```



```
IT114 [ Milestone1][ 1m40s]
> cd /run/media/garrettgr/SOCKET-nano/NJIT/IT114 ; /usr/bin/env [illegible]
[illegible] --enable-preview -XX:+ShowCodeDetailsInExceptionMessages -cp [illegible] Server
Starting Server
Server is listening on port 3000
waiting for next client
waiting for next client
Client connected
Thread[28]: Thread created
Thread[28]: Thread starting
Thread-0 leaving room Lobby
Thread-0 joining room Lobby
Thread[28]: Received from client: Type[CONNECT], Number[0], Message[null]
waiting for next client
Client connected
Thread[30]: Thread created
Thread-2 leaving room Lobby
Thread[30]: Thread starting
Thread-2 joining room Lobby
Thread[30]: Received from client: Type[CONNECT], Number[0], Message[null]
waiting for next client
Client connected
Thread[32]: Thread created
Thread-4 leaving room Lobby
Thread[32]: Thread starting
Thread-4 joining room Lobby
Thread[32]: Received from client: Type[CONNECT], Number[0], Message[null]
Thread[32]: Received from client: Type[MESSAGE], Number[0], Message[Hey guys!]
Room[Lobby]: Sending message to 3 clients
Thread[30]: Received from client: Type[MESSAGE], Number[0], Message[Hey!]
Room[Lobby]: Sending message to 3 clients
Thread[28]: Received from client: Type[MESSAGE], Number[0], Message[I Think that we all reconnected just fine]
Room[Lobby]: Sending message to 3 clients
```

```
Debug Info: Type[MESSAGE], Number[0], Message[hi, howdy]
client_1: hi, howdy
How are you?
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[How are you?]
client_0: How are you?
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
*client_2 disconnected*
Debug Info: Type[MESSAGE], Number[0], Message[Oh noooo, there goes client_2]
client_1: Oh noooo, there goes client_2
java.io.EOFException
        at java.base/java.io.ObjectInputStream$BlockDataInputStream.peekByte(ObjectInputStream.java:3232)
        at java.base/java.io.ObjectInputStream.readObject0(ObjectInputStream.java:1715)
        at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:540)
        at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:498)
        at Client$2.run(Client.java:[illegible])
Server closed connection
Closing output stream
Closing input stream
Closing connection
Closed socket
Stopped listening to server input
/connect localhost:3000
Client connected
Waiting for input
Debug Info: Type[CONNECT], Number[0], Message[connected]
*client_0 connected*
Hey guys!
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[Hey guys!]
client_0: Hey guys!
Debug Info: Type[MESSAGE], Number[0], Message[Hey!]
client_1: Hey!
Debug Info: Type[MESSAGE], Number[0], Message[I Think that we all reconnected just fine]
client_2: I Think that we all reconnected just fine
```

Shows all the clients reconnecting without issue

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:

A client can disconnect --- this closes the 'Socket connection' (by calling the Socket.close() method) -- this will throw a 'SocketException' which the server captures and handles by removing the ServerThread and sending a disconnect message to all other clients still connected and in the same room as the client (also handle cleanup, but not log an error or crash by detecting a 'disconnect' message)

A client can crash -- this will cause the client's socket to close unexpectedly and will also throw a 'SocketException' -- the server can detect that the client had not initiated a disconnection and handle the abrupt disconnection / crash accordingly (such as using the cleanup() method_

The server can crash / be terminated -- this will cause the client to attempt to read from / write to a socket, which will also throw a 'SocketException' -- the client can then close their input and output streams, close their socket and report the disconnect to the client

COLLAPSE

Misc (1 pt.)

COLLAPSE

Task #1 - Points: 1

Text: Add the pull request link for this branch

URL #1

https://github.com/GarrettGR/IT114/pull/7

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 did not encounter any issues

COLLAPSE

Task #3 - Points: 1

Text: WakaTime Screenshot

Details:

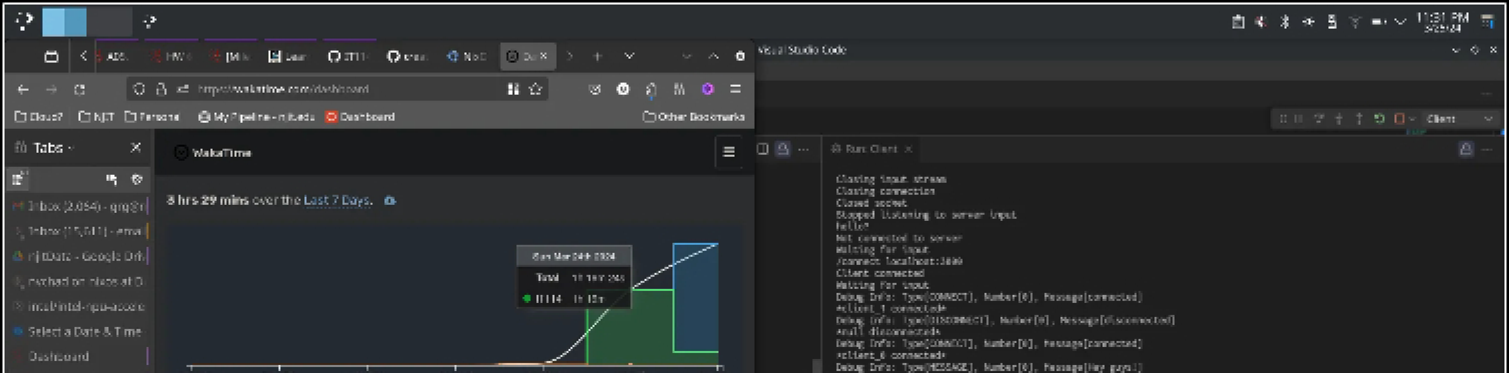
Grab a snippet showing the approximate time involved that clearly shows your repository.

The duration isn't considered for grading, but there should be some time involved.

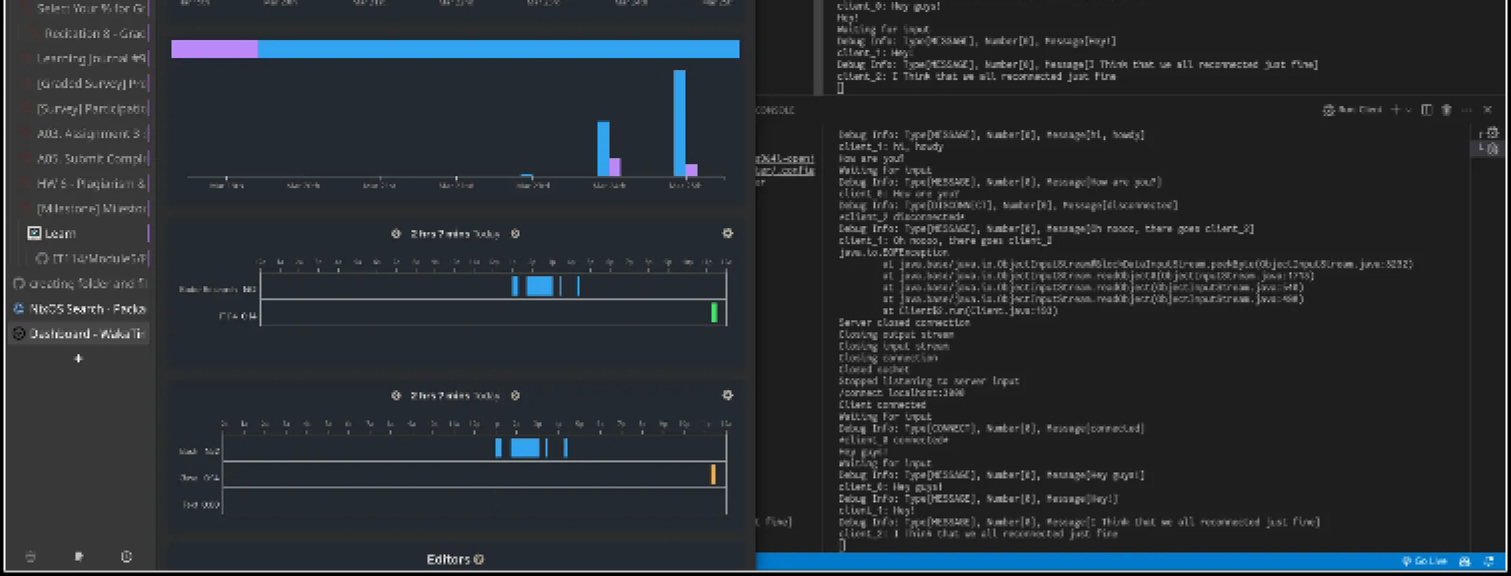
Task Screenshots:

Gallery Style: Large View

Small Medium Large



```
Closing input stream
Closing connection
Closed socket
Stopped listening to server input
hello?
Not connected to server
Waiting for input
/connect localhost:3000
Client connected
Waiting for input
Debug Info: Type[CONNECT], Number[0], Message[connected]
*client_1 connected*
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
*null disconnected*
Debug Info: Type[CONNECT], Number[0], Message[connected]
*client_0 connected*
Debug Info: Type[MESSAGE], Number[0], Message[Hey guys!]
```



```
client_0: Hey guys!
Hey!
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[Hey!]
client_1: Hey!
Debug Info: Type[MESSAGE], Number[0], Message[I Think that we all reconnected just fine]
client_2: I Think that we all reconnected just fine
[]
```

```
Debug Info: Type[MESSAGE], Number[0], Message[hi, howdy]
client_1: hi, howdy
How are you?
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[How are you?]
client_0: How are you?
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
*client_2 disconnected*
Debug Info: Type[MESSAGE], Number[0], Message[Oh noooo, there goes client_2]
client_1: Oh noooo, there goes client_2
java.io.EOFException
        at java.base/java.io.ObjectInputStream$BlockDataInputStream.peekByte(ObjectInputStream.java:3232)
        at java.base/java.io.ObjectInputStream.readObject0(ObjectInputStream.java:1733)
        at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:540)
        at java.base/java.io.ObjectInputStream.readObject(ObjectInputStream.java:498)
        at Client$2.run(Client.java:193)
Server closed connection
Closing output stream
Closing input stream
Closing connection
Closed socket
Stopped listening to server input
/connect localhost:3000
Client connected
Waiting for input
Debug Info: Type[CONNECT], Number[0], Message[connected]
*client_0 connected*
Hey guys!
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[Hey guys!]
client_0: Hey guys!
Debug Info: Type[MESSAGE], Number[0], Message[Hey!]
client_1: Hey!
Debug Info: Type[MESSAGE], Number[0], Message[I Think that we all reconnected just fine]
client_2: I Think that we all reconnected just fine
[]
```

Missing Caption

End of Assignment