

# Submission Worksheet

CLICK TO GRADE

<https://learn.ethereallab.app/assignment/IT114-002-S2024/it114-project-milestone-1/grade/mth39>

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

Submissions:

Submission Selection

1 Submission [active] 3/13/2024 2:39:45 PM

Instructions

^ COLLAPSE ^

1. Create a new branch called Milestone1
  2. At the root of your repository create a folder called Project if one doesn't exist yet
  3. You will be updating this folder with new code as you do milestones
  4. You won't be creating separate folders for milestones; milestones are just branches
  5. Create a pull request from Milestone1 to main (don't complete/merge it yet, just have it in open status)
  6. 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>
  7. Fix the package references at the top of each file (these are the only edits you should do at this point)
  8. Git add/commit the baseline and push it to github
  9. Create a pull request from Milestone1 to main (don't complete/merge it yet, just have it in open status)
  10. Ensure the sample is working and fill in the below deliverables
  11. Note: The client commands likely are different in part 5 with the /name and /connect options instead of just "connect"
  12. Generate the worksheet output file once done and add it to your local repository
  13. 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
<input type="checkbox"/> #1	1	Server should properly be listening to its port from the command line (note the related message)
<input type="checkbox"/> #2	1	Clients should be successfully waiting for input
<input type="checkbox"/> #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

Screenshot showing server listening to its port (terminal all the way on the left), and clients successfully connected, with a name, and waiting for input (last three terminals).

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
<input type="checkbox"/> #1	1	Mention how the server-side of the connection works
<input type="checkbox"/> #2	1	Mention how the client-side of the connection works
<input type="checkbox"/> #3	1	Describe the socket steps until the server is waiting for messages from the client

#### Response:

The server side of the connection is responsible for managing rooms. The default will be the lobby, from there, rooms can be created, joined, or removed depending on the method used.

The client side of the connection allows users to chat with each other. They can see the messages of other clients that are in the same room, this can either be the lobby or rooms that are created.

The server initializes with a default port of 3000 and creates a ServerSocket to listen for incoming connections. It also establishes a default room named "Lobby" for managing client interactions. Upon client connection, the server accepts the connection and spawns a new ServerThread to handle communication with the client. This thread manages the interaction with the client, allowing the server to wait for messages from the client. Overall, the server architecture enables continuous listening for client connections and seamless communication management through individual threads, ensuring efficient client-server interaction.



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
<input type="checkbox"/> #1	1	At least two clients connected to the server
<input type="checkbox"/> #2	1	Client can send messages to the server
<input type="checkbox"/> #3	1	Server sends the message to all clients in the same room
<input type="checkbox"/> #4	1	Messages clearly show who the message is from (i.e., client name is clearly with the message)

<input type="checkbox"/> #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)
<input type="checkbox"/> #6	1	Clearly caption each image regarding what is being shown

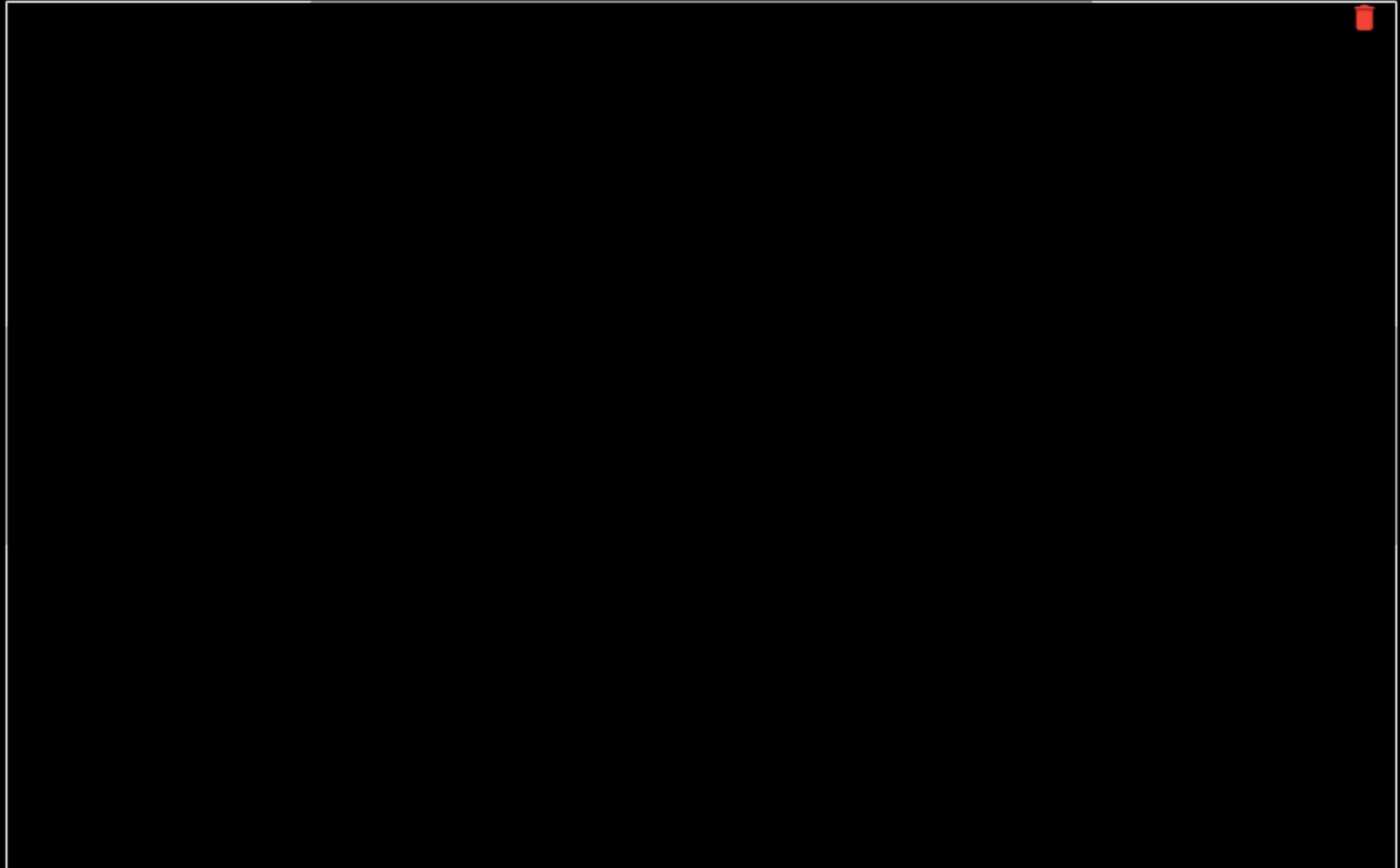
#### Task Screenshots:

#### Gallery Style: Large View

Small

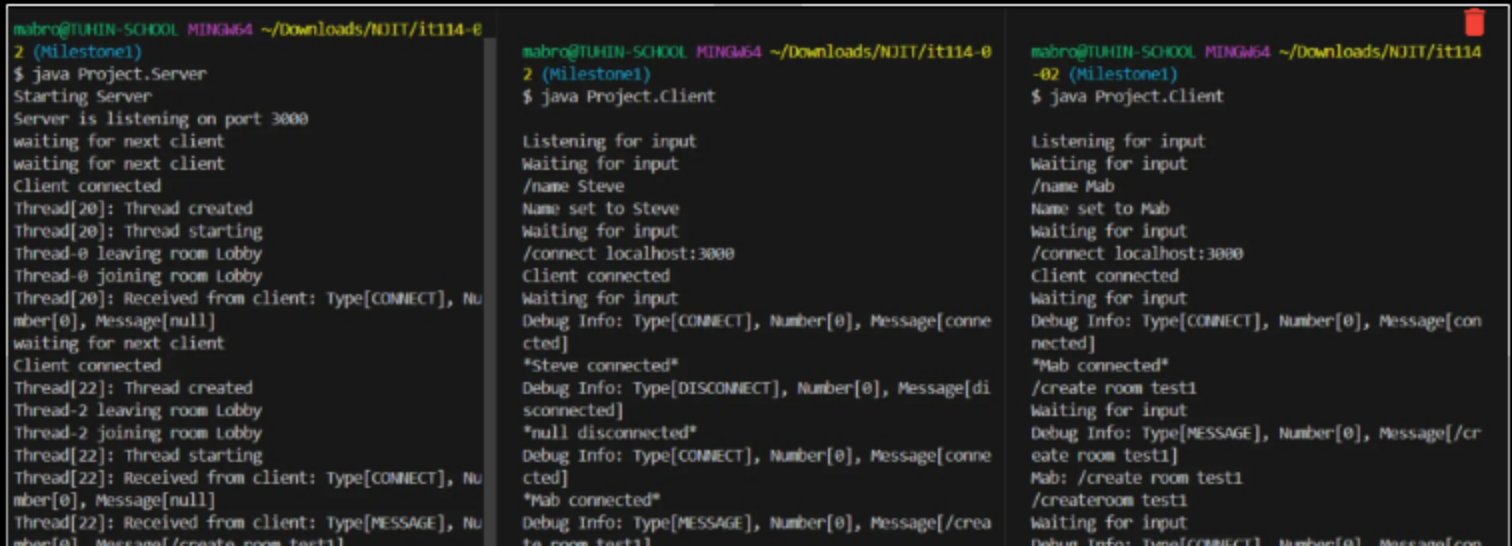
Medium

Large



Screenshot showing two clients connected to the server, sending messages to the server, which the server sends to all clients in the same room. Messages clearly show who the message is from (i.e., client name is clearly with the message)

#### Checklist Items (0)



```
Room[0]: Received from client: Type[MESSAGE], Number[0], Message[createroom test1]
Room[Lobby]: Sending message to 2 clients
Created new room: test1
Thread-2 leaving room Lobby
Thread-2 joining room test1
Thread[20]: Received from client: Type[MESSAGE], Number[0], Message[createroom test2]
Room[Lobby]: Sending message to 1 clients
Created new room: test2
Thread-0 leaving room Lobby
Thread-0 joining room test2
Thread[20]: Received from client: Type[MESSAGE], Number[0], Message[hi]
Room[test2]: Sending message to 1 clients
[]

Mab: /create room test1
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
*Mab disconnected*

Steve connected*
hi
Waiting for input
Debug Info: Type[CONNECT], Number[0], Message[connected]
*Steve connected*
hi
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[hi]
Steve: hi
[]
```

Screenshot showing two clients in separate rooms. When Steve sends the message "hi", Mab does not receive it because he is in another room.

## 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
<input type="checkbox"/> #1	1	Mention the client-side (sending)
<input type="checkbox"/> #2	1	Mention the ServerThread's involvement
<input type="checkbox"/> #3	1	Mention the Room's perspective
<input type="checkbox"/> #4	1	Mention the client-side (receiving)

#### Response:

Once a client enters the server, they are automatically placed in the lobby, if they're the only one in the lobby, the room is going to clean itself up using the cleanup method so that it's not wasting any resources in case anyone wants to create their own room. Once the user is in a room, they will only get messages from whoever is in the same room as them based on an if else statement. To create or delete rooms, the processCommand method is used with a switch case statement to allow the user to make the room or join one if it exists and if it doesn't, a message will pop up saying that the room doesn't exist. There are also cases that allow the user to disconnect from the room once the user types out; disconnect, logout, or logoff. The serverthread is there to replace the server reference with the room reference so instead of sending the message to the server, it can send the message to the room.

## Disconnecting/Termination (3 pts.)



## Task #1 - Points: 1

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

### Checklist

\*The checkboxes are for your own tracking

#	Points	Details
<input type="checkbox"/> #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)
<input type="checkbox"/> #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)
<input type="checkbox"/> #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)
<input type="checkbox"/> #4	1	Clearly caption each image regarding what is being shown

### Task Screenshots:

#### Gallery Style: Large View

Small Medium Large

```
Thread[22]: Thread created
Thread-2 leaving room Lobby
Thread-2 joining room Lobby
Thread[22]: Thread starting
Thread[22]: Received from client: Type[CONNECT], Number[0], Message[null]
Thread[22]: Received from client: Type[MESSAGE], Number[0], Message[/create room test1]
Room[Lobby]: Sending message to 2 clients
Thread[22]: Received from client: Type[MESSAGE], Number[0], Message[/createroom test1]
Room[Lobby]: Sending message to 2 clients
Created new room: test1
Thread-2 leaving room Lobby
Thread-2 joining room test1
Thread[20]: Received from client: Type[MESSAGE], Number[0], Message[/createroom test2]
Room[Lobby]: Sending message to 1 clients
Created new room: test2
Thread-0 leaving room Lobby
Thread-0 joining room test2
Thread[20]: Received from client: Type[MESSAGE], Number[0], Message[hi]
Room[test2]: Sending message to 1 clients
Thread[20]: Received from client: Type[MESSAGE], Number[0], Message[/disconnect]
Room[test2]: Sending message to 1 clients
Thread[20]: Passed in room was null, this shouldn't happen
Thread[20]: Thread being disconnected by server
Thread[20]: Thread cleanup() start
Thread[20]: Thread cleanup() complete
Removed empty room test2
Thread[20]: Exited thread loop. Cleaning up connection
Thread[20]: Thread cleanup() start
Thread[20]: Thread cleanup() complete
Thread[22]: Received from client: Type[MESSAGE], Number[0], Message[hello]
Room[test1]: Sending message to 1 clients
[]
```

```
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
*null disconnected*
Debug Info: Type[CONNECT], Number[0], Message[connected]
*Mab connected*
Debug Info: Type[MESSAGE], Number[0], Message[/create room test1]
Mab: /create room test1
Debug Info: Type[DISCONNECT], Number[0], Message[disconnected]
*Mab disconnected*
/createroom test2
Waiting for input
Debug Info: Type[CONNECT], Number[0], Message[connected]
*Steve connected*
hi
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[hi]
Steve: hi
/disconnect
Waiting for input
java.io.EOFException
    at java.base/java.io.ObjectInputStream$BlockDataInputStream.peekByte(ObjectInputStream.java:3232)
    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 Project.Client$2.run(Client.java:195)
Server closed connection
Closing output stream
Closing input stream
Closing connection
Closed socket
Stopped listening to server input
[]
```

```
mabro@TUHIN-SCHOOL MITIGAS4 ~/Downloads/NDIT/It114-02 (Milestone1)
$ java Project.Client

Listening for input
Waiting for input
/name Mab
Name set to Mab
Waiting for input
/connect localhost:3000
client connected
Waiting for input
Debug Info: Type[CONNECT], Number[0], Message[connected]
*Mab connected*
/create room test1
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[/create room test1]
Mab: /create room test1
/createroom test1
Waiting for input
Debug Info: Type[CONNECT], Number[0], Message[connected]
*Mab connected*
hello
Waiting for input
Debug Info: Type[MESSAGE], Number[0], Message[hello]
Mab: hello
[]
```

Screenshot showing client Steve disconnecting from the server, but the server is still running as Mab is able to send a message which server displays. The server also shows Steve disconnected message

### Checklist Items (0)

```
Room[test2]: Sending message to 1 clients
Thread[20]: Passed in room was null, this shouldn't happen
Thread[20]: Thread being disconnected by server
Thread[20]: Thread cleanup() start
```

```
ected]
*Mab connected*
java.net.SocketException: Connection reset
    at java.base/sun.nio.ch.NioSocketImpl.implRead(NioSocketImpl.java:318)
```

```
ected]
*Mab connected*
java.net.SocketException: Connection reset
    at java.base/sun.nio.ch.NioSocketImpl.implRead(NioSocketImpl.java:318)
```

```

Thread[20]: Thread cleanup() complete
Removed empty room test2
Thread[20]: Exited thread loop. Cleaning up connection
Thread[20]: Thread cleanup() start
Thread[20]: Thread cleanup() complete
Thread[22]: Received from client: Type[MESSAGE], Number[0], Message[hello]

mabro@TUHIN-SCHOOL MINE664 ~/Downloads/NDIT/it114-E
2 (Milestone1)
$ java Project.Server
Starting Server
Server is listening on port 3000
waiting for next client
waiting for next client
Client connected
Thread[20]: Thread created
Thread[20]: Thread starting
Thread-0 leaving room Lobby
Thread-0 joining room Lobby
Thread[20]: Received from client: Type[CONNECT], Number[0], Message[null]
waiting for next client
Client connected
Thread[22]: Thread created
Thread-2 leaving room Lobby
Thread[22]: Thread starting
Thread-2 joining room Lobby
Thread[22]: Received from client: Type[CONNECT], Number[0], Message[null]

mabro@TUHIN-SCHOOL MINE664 ~/Downloads/NDIT/it114-E
2 (Milestone1)
$

```

```

at java.base/sun.nio.ch.NioSocketImpl.read(
NioSocketImpl.java:346)
at java.base/sun.nio.ch.NioSocketImpl$1.read(
NioSocketImpl.java:796)
at java.base/java.net.Socket$SocketInputStr
eam.read(Socket.java:1099)
at java.base/java.net.Socket$SocketInputStr
eam.read(Socket.java:1093)
at java.base/java.io.ObjectInputStream$Peek
InputStream.peek(ObjectInputStream.java:2893)
at java.base/java.io.ObjectInputStream$Bloc
kDataInputStream.peek(ObjectInputStream.java:3220)
at java.base/java.io.ObjectInputStream$Bloc
kDataInputStream.peekByte(ObjectInputStream.java:32
30)
at java.base/java.io.ObjectInputStream.read
Object0(ObjectInputStream.java:1713)
at java.base/java.io.ObjectInputStream.read
Object(ObjectInputStream.java:540)
at java.base/java.io.ObjectInputStream.read
Object(ObjectInputStream.java:498)
at Project.Client$2.run(Client.java:195)
Server closed connection
Closing output stream
Closing input stream
Closed socket
Stopped listening to server input
hi
Not connected to server
Waiting for input

```

```

at java.base/sun.nio.ch.NioSocketImpl.read(
NioSocketImpl.java:346)
at java.base/sun.nio.ch.NioSocketImpl$1.rea
d(NioSocketImpl.java:796)
at java.base/java.net.Socket$SocketInputSt
ream.read(Socket.java:1099)
at java.base/java.net.Socket$SocketInputSt
ream.read(Socket.java:1093)
at java.base/java.io.ObjectInputStream$Peek
InputStream.peek(ObjectInputStream.java:2893)
at java.base/java.io.ObjectInputStream$Bloc
kDataInputStream.peek(ObjectInputStream.java:3220
)
at java.base/java.io.ObjectInputStream$Bloc
kDataInputStream.peekByte(ObjectInputStream.java:
3230)
at java.base/java.io.ObjectInputStream.rea
dObject0(ObjectInputStream.java:1713)
at java.base/java.io.ObjectInputStream.rea
dObject(ObjectInputStream.java:540)
at java.base/java.io.ObjectInputStream.rea
dObject(ObjectInputStream.java:498)
at Project.Client$2.run(Client.java:195)
Server closed connection
Closing output stream
Closing input stream
Closing connection
Closed socket
Stopped listening to server input
hi
Not connected to server
Waiting for input

```

Showing server terminated but client still running

## Checklist Items (0)

### 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
<input type="checkbox"/> #1	1	Mention how a client gets disconnected from a Socket perspective
<input type="checkbox"/> #2	1	Mention how/why the client program doesn't crash when the server disconnects/terminates.
<input type="checkbox"/> #3	1	Mention how the server doesn't crash from the client(s) disconnecting

### Response:

Once the user disconnects from the server, an IO exception is displayed which is expected since the connection got severed. The output stream, input stream, connection, and socket get closed so the user can still type things, but its not connected to the server anymore. Even when the clients disconnect, the server is still able to run since the server itself wasn't terminated.

### Misc (1 pt.)



^COLLAPSE ^

### Task #1 - Points: 1

Text: Add the pull request link for this branch

#### URL #1

<https://github.com/MabroorHussan/it114-02/pull/8>



^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 a great time learning how chat rooms work in Java and the server side programming behind them.



^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

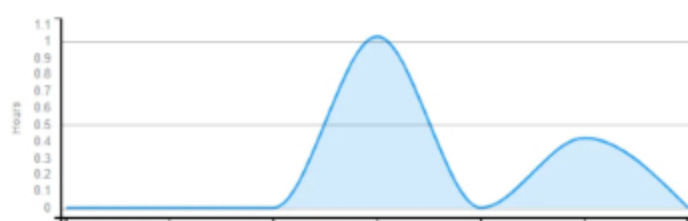
Large

## Projects • it114-02

total 7 hrs 41 mins

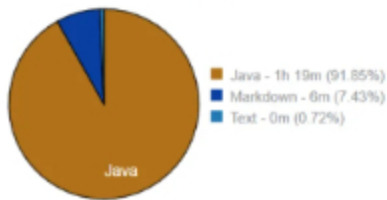


1 hr 26 mins over the Last 7 Days in it114-02 under all branches. 📊





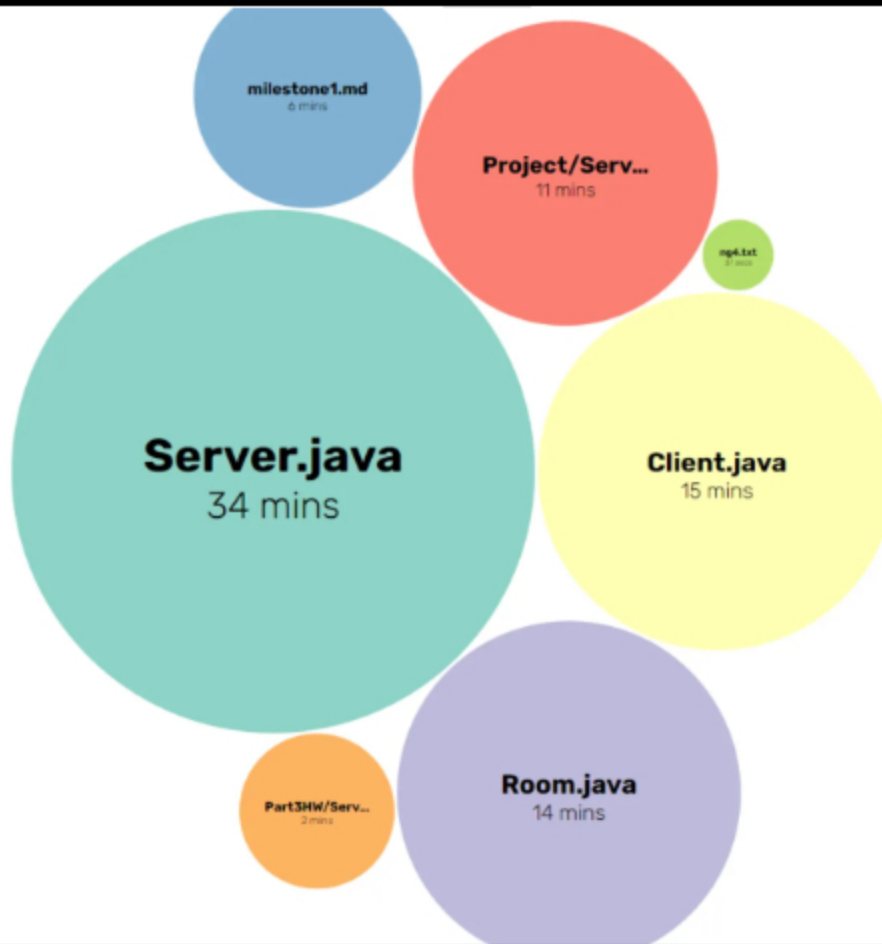
Languages



Editors



Screenshot showing Waka Time on repository "it114-02"



Screenshot showing the files Waka Time for the files I worked on

End of Assignment