Submission Worksheet

Submission Data

Course: IT114-005-F2025
Assignment: IT114 Milestone 2 - Hangman
Student: Nilkanth D. (nhd5)
Status: Submitted | **Worksheet Progress:** 100%
Potential Grade: 10.00/10.00 (100.00%)
Received Grade: 0.00/10.00 (0.00%)
Started: 11/23/2025 11:26:15 PM
Updated: 11/25/2025 12:22:18 AM
Grading Link: https://learn.ethereallab.app/assignment/v3/IT114-005-F2025/it114-milestone-2-hangman/grading/nhd5
View Link: https://learn.ethereallab.app/assignment/v3/IT114-005-F2025/it114-milestone-2-hangman/view/nhd5

Instructions

1. Refer to Milestone2 of Hangman / Word guess
   1. Complete the features
2. Ensure all code snippets include your ucid, date, and a brief description of what the code does
3. Switch to the `Milestone2` branch
   1. `git checkout Milestone2`
   2. `git pull origin Milestone2`
4. Fill out the below worksheet as you test/demo with 3+ clients in the same session
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 Milestone2`
   4. On Github merge the pull request from `Milestone2` to `main`
7. Upload the same PDF to Canvas
8. Sync Local
   1. `git checkout main`
   2. `git pull origin main`

- Complete each section and task sequentially.
- Review the details and validation criteria for each task.
- Ensure subtasks are completed before the parent task.

Section #1: (1 pt.) Payloads

Progress: 100%

--- Section Collapsed ---

Section #2: (4 pts.) Lifecycle Events

Progress: 100%

--- Section Collapsed ---

Section #3: (4 pts.) Gameroom User Action And State

Progress: 100%

--- Section Collapsed ---

Section #4: (1 pt.) Misc

Progress: 100%

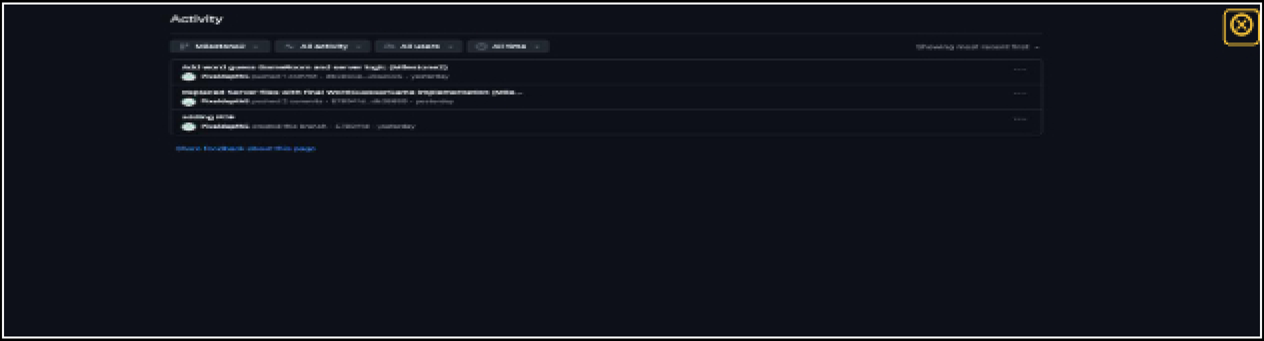
Task #1 (0.33 pts.) - Github Details

Progress: 100%

Part 1:

Progress: 100%

Details:
From the Commits tab of the Pull Request screenshot the commit history



Activity

Add word game GameRoom and server logic (Milestone3)
[illegible]

Replaced Server files with final WordGuesserGame implementation (Mile...
[illegible]

adding [illegible]
[illegible]

[illegible]

history

Saved: 11/25/2025 12:17:06 AM

Part 2:

Progress: 100%

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

URL #1
https://github.com/Pixeldepth5/nhd5-IT114/pull/5/

URL
https://github.com/Pixeldepth5/nl

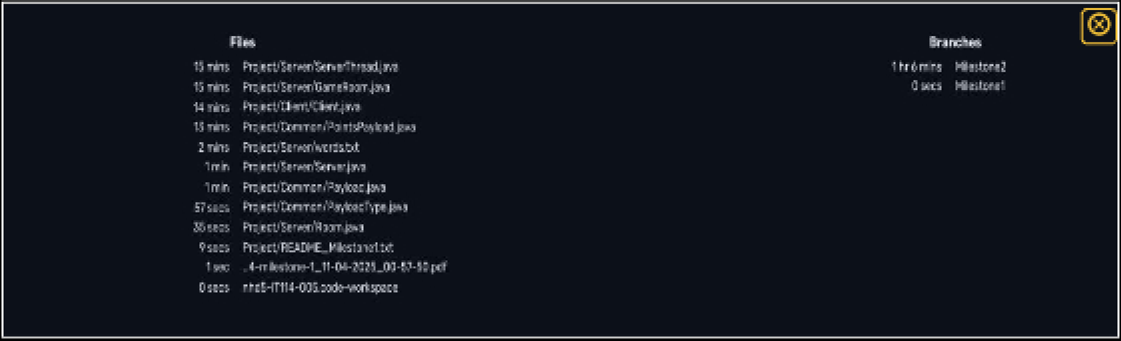
Saved: 11/25/2025 12:17:06 AM

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

Progress: 100%

Details:

- 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



```
Files                                                      Branches
15 mins  Project/Server/ServerThread.java                 1 hr 6 mins  Milestone2
15 mins  Project/Server/GameRoom.java                     0 secs       Milestone1
14 mins  Project/Client/Client.java
15 mins  Project/Common/PointsPayload.java
2 mins   Project/Server/words.txt
1 min    Project/Server/Server.java
1 min    Project/Common/Payload.java
57 secs  Project/Common/PayloadType.java
35 secs  Project/Server/Room.java
9 secs   Project/README_Milestone1.txt
1 sec    ...4-milestone-1_11-04-2025_00-57-50.pdf
0 secs   nhd5-IT114-005.code-workspace
```

Wakwaktime history

Saved: 11/25/2025 12:18:26 AM

Task #3 (0.33 pts.) - Reflection

Progress: 100%

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

Progress: 100%

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

Your Response:

I learned how all the moving parts of a networked word game fit together, from the client typing a command to the server updating game state and sending payloads back. I finally understood the "lifecycle" pattern in the GameRoom - session start → round start → turn start → turn end → round end → session end - and how each stage has its own responsibilities. Given the nature of /guess, /letter, and /skip, their implementation showed me how to reuse the same message pipeline but change the game logic depending on the command. The more practical side is that I got more comfortable compiling and running a Java project from the Mac terminal with multiple clients connected at the same time, which is easy.

Saved: 11/25/2025 12:19:47 AM

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

Progress: 100%

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

Your Response:

Once the structure became clear, the easiest part of the assignment for me was wiring up the basic commands and payloads. After I had the Payload, PayloadType, and PointsPayload classes set up, reusing that pattern for different actions like /guess, /letter, and /skip felt pretty straightforward. It was also easy to add debug toString() methods and see exactly what was being sent across between the client and server. Once the pipeline was working, testing commands from multiple clients became more of a routine process than a challenge.

Saved: 11/25/2025 12:21:02 AM

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

Progress: 100%

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

Your Response:

The toughest aspect of this assignment was to get all the lifecycle logic properly working across multiple clients at the same time. Ensuring that a game flowed in the right order-session start, round start, turn start, turn end, round end, and session end-required a lot of careful thinking and debugging. I also had to be very precise about whose turn it was, when strikes or points should be applied, and when to end a round or the whole session. Coordinating that with the networking code and testing it using several terminal windows at once was definitely the most challenging part.

Saved: 11/25/2025 12:22:18 AM