Course: IT114-006-S2025
Assignment: IT114 Milestone 2 - RPS
Student: Muhammad K. (muk)
Status: Submitted | Worksheet Progress: 85%
Potential Grade: 6.00/10.00 (60.00%)
Received Grade: 0.00/10.00 (0.00%)
Grading Link: https://learn.ethereallab.app/assignment/v3/IT114-006-S2025/it114-milestone-2-rps/grading/muk

Instructions

1. Refer to Milestone2 of Rock Paper Scissors
    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.

100%

Section #1: (1 pt.) Payloads

100%

Task #1 (1 pt.) - Show Payload classes and subclasses

Combo Task:

Weight: *100%*
Objective: *Show Payloadclasses and subclasses*

Details:

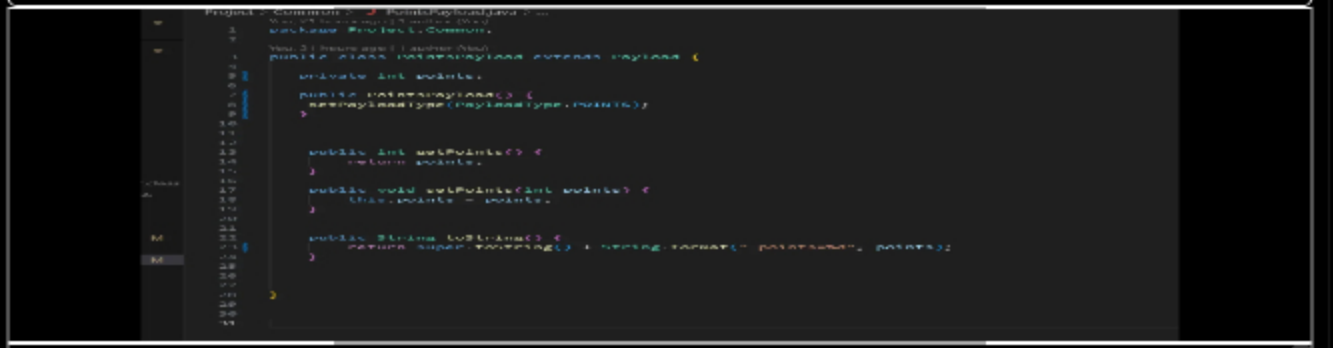
- Reqs from the document
  - Provided Payloadfor applicable items that only need client id, message, and type
  - PointsPayloadfor syncing points of players
  - Each payload will be presented by debug output (i.e, properly override the toString() method like the lesson examples)

Image Prompt

Weight: *50%*

Details:

- Show the code related to your payloads (Payload,PointsPayload,and any new ones added)
- Each payload should have an overriden toString() method showing its internal data



PointsPayload

Saved: 4/25/2025 5:29:42 PM

Text Prompt

Weight: *50%*

Details:

- Briefly explain the purpose of each payload shown in the screenshots and their properties

Your Response:

PointsPayload tracks and transfers the number of wins(points) each player will earn during the game.

Saved: 4/25/2025 5:29:42 PM

100% Section #2: (4 pts.) Lifecycle Events

100% Task #1 (0.80 pts.) - GameRoom Client Add/Remove

Combo Task:

Weight: *20%*
Objective: *GameRoom Client Add/Remove*

Image Prompt

Weight: *50%*

Details:

- Show the `onClientAdded()` code
- Show the `onClientRemoved()` code

```
/** {@inheritDoc} */
@Override
protected void onClientAdded(ServerThread sp) {
    // sync GameRoom state to new client
    syncCurrentPhase(sp);
    syncReadyStatus(sp);
    syncTurnStatus(sp);
}
```

Code for onClientAdded()

```
/** {@inheritDoc} */
@Override
protected void onClientRemoved(ServerThread sp) {
    // added after Summer 2024 Demo
    // Stops the timers so room can clean up
    LoggerUtil.INSTANCE.info("Player Removed, remaining: " + clientsInRoom.size());
    if (clientsInRoom.isEmpty()) {
        resetReadyTimer();
        resetTurnTimer();
        resetRoundTimer();
        onSessionEnd();
    }
}
```

Code for onClientRemoved()

Saved: 4/25/2025 10:55:35 PM

Text Prompt

Weight: *50%*

Details:

- Briefly note the actions that happen in `onClientAdded()` (app data should at least be synchronized to the joining user)
- Briefly note the actions that happen in `onClientRemoved()` (at least should handle logic for an empty session)

Your Response:

The actions that occur with onClientAdded() are that the player learns which round the game might be in, who might be present/ready, and which player is going. For onClientRemoved(), when a player leaves, the game will check how many players there are left and if there are none. All sessions end and the game timers will end.

Saved: 4/25/2025 10:55:35 PM

100% Task #2 (0.80 pts.) - GameRoom Session Start

Combo Task:

Weight: *20%*
Objective: *GameRoom Session Start*

Details:

- Reqs from document
  - First round is triggered
- Reset/set initial state

Image Prompt

Weight: *50%*

Details:

- Show the snippet of `onSessionStart()`

```
    // end timer handlers

    // lifecycle methods

    /** {@inheritDoc} */
    @Override
    protected void onSessionStart() {
        LoggerUtil.INSTANCE.info(message:"onSessionStart() start");
        changePhase(Phase.IN_PROGRESS);
        round = 0;
        LoggerUtil.INSTANCE.info(message:"onSessionStart() end");
        onRoundStart();
    }
```

Snippet of onSessionStart()

Saved: 4/26/2025 6:51:22 PM

Text Prompt

Weight: *50%*

Details:

- Briefly explain the logic that occurs here (i.e., setting up initial session state for your project) and next lifecycle trigger

Your Response:

The game will set up the session and take steps like allowing the first player to start, waiting for the player to ready up or showing "waiting for player." This is something that happens before the game begins.

Saved: 4/26/2025 6:51:22 PM

100%

Task #3 (0.80 pts.) - GameRoom Round Start

Combo Task:

Weight: *20%*
Objective: *GameRoom Round Start*

Details:

- Reqs from Document
  - Initialize remaining Players' choices to null (not set)
  - Set Phase to "choosing"
  - GameRoom round timer begins

Image Prompt

Weight: *50%*

Details:

- Show the snippet of `onRoundStart()`

```
/** {@inheritDoc} */
@Override
protected void onRoundStart() {
    LoggerUtil.INSTANCE.info(message:"onRoundStart() start");
    resetRoundTimer();
    resetTurnStatus();
    round++;
    relay(sender:null, String.format("Round %d has started", round));
    startRoundTimer();
    LoggerUtil.INSTANCE.info(message:"onRoundStart() end");
}
```

Snippet of onRoundStart()

Saved: 4/26/2025 10:51:31 PM

Text Prompt

Weight: *50%*

Details:

- Briefly explain the logic that occurs here (i.e., setting up the round for your project)

Your Response:

With onRoundStart() everything is set up to start the new round. Players choices are reset, phase set to waiting or choice(RPS). Timer will be set giving players limited time.

Saved: 4/26/2025 10:51:31 PM

100%

Task #4 (0.80 pts.) - GameRoom Round End

Combo Task:

Weight: *20%*
Objective: *GameRoom Round End*

Details:

- Reqs from Document
  - **Condition 1:** Round ends when round timer expires
  - **Condition 2:** Round ends when all active Players have made a choice
  - All Players who are not eliminated and haven't made a choice will be marked as eliminated
  - Process Battles:
    - Round-robin battles of eligible Players (i.e., Player 1 vs Player 2 vs Player 3 vs Player 1)
      - Determine if a Player loses if they lose the "attack" or if they lose the "defend" (this Player has two battles each round)
        - Give a point to the winning Player
        - Points will be stored on the Player/User object
        - Sync the points value of the Player to all Clients
      - Relay a message stating the Players that competed, their choices, and the result of the battle
      - Losers get marked as eliminated (Eliminated Players stay as spectators but are skipped for choices and for win checks)
    - Count the number of non-eliminated Players
      - If one, this is your winner (onSessionEnd())
      - If zero, it was a tie (onSessionEnd())
      - If more than one, do another round (onRoundStart())

Image Prompt

Weight: *50%*

Details:

- Show the snippet of `onRoundEnd()`

```
/** {@inheritDoc} */
@Override
protected void onRoundEnd() {
    LoggerUtil.INSTANCE.info(message:"onRoundEnd() start");
    resetRoundTimer(); // reset timer if round ended without the time expiring

    LoggerUtil.INSTANCE.info(message:"onRoundEnd() end");
    if (round >= 3) {
        onSessionEnd();
    }
    else{
        onRoundStart();
    }
}
```

Snippet of onRoundEnd()

Saved: 4/27/2025 10:53:11 PM

Text Prompt

Weight: *50%*

Details:

- Briefly explain the logic that occurs here (i.e., cleanup, end checks, and next lifecycle events)

Your Response:

onRoundEnd() will reset the round timer. If timer runs out or all players have made their choice the round will end. If a player did not participate, they will be eliminated. The game will go through each RPS battles with each person making their choice. If a player wins a round, a point is addded to them. Points are stored and displayed to everyone. Any player who loses becomes a spectator. At the end if a player remains, he wins, if no players remain its a tie and game ends. But if players still remain, a new round will begin.

Saved: 4/27/2025 10:53:11 PM

100% Task #5 (0.80 pts.) - GameRoom Session End

Combo Task:

Weight: *20%*
Objective: *GameRoom Session End*

Details:

- Reqs from Document
  - **Condition 1:** Session ends when one Player remains (they win)
  - **Condition 2:** Session ends when no Players remain (this is a tie)
  - Send the final scoreboard to all clients sorted by highest points to lowest (include a game over message)
  - Reset the player data for each client server-side and client-side (do not disconnect them or move them to the lobby)
  - A new ready check will be required to start a new session

Image Prompt

Weight: *50%*

Details:

- Show the snippet of `onSessionEnd()`

```
/** {@inheritDoc} */
@Override
protected void onSessionEnd() {
    LoggerUtil.INSTANCE.info(message:"onSessionEnd() start");
    resetReadyStatus();
    resetTurnStatus();
    changePhase(Phase.READY);
    LoggerUtil.INSTANCE.info(message:"onSessionEnd() end");
}
// end lifecycle methods
```

Snippet of onSessionEnd()

Saved: 4/27/2025 11:02:09 PM

Text Prompt

Weight: *50%*

Details:

- Briefly explain the logic that occurs here (i.e., cleanup/reset, next lifecycle events)

Your Response:

For onSessionEnd() every players ready and turn statuses are reset. The game is turned back to Ready mode. Final scores are sent to all players and game over message is shown. The states of player are reset but they still remain inside the room. Every player has to ready to begin another game.

Saved: 4/27/2025 11:02:09 PM

0%

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

0%

Task #1 (2 pts.) - Choice Logic

Combo Task:

Weight: *50%*
Objective: *Choice Logic*

Details:

- Reqs from document
  - Command: `/pick <[r,p,s]>` (user picks one)
    - GameRoom will check if it's a valid option
    - GameRoom will record the choice for the respective Player
    - A message will be relayed saying that "X picked their choice"
    - If all Players have a choice the round ends

Image Prompt

Weight: *50%*

Details:

- Show the code snippets of the following, and clearly caption each screenshot
- Show the Client processing of this command (process client command)
- Show the ServerThread processing of this command (process method)
- Show the GameRoom handling of this command (handle method)
- Show the sending/syncing of the results of this command to users (send/sync method)
- Show the ServerThread receiving this data (send method)
- Show the Client receiving this data (process method)

Missing Caption

Not saved yet

Text Prompt

Weight: *50%*

Details:

- Briefly explain/list in order the whole flow of this command being handled from the client-side to the server-side and back

Your Response:

Missing Response

Not saved yet

0%

Task #2 (2 pts.) - Game Cycle Demo

Image Prompt

Weight: *50%*
Objective: *Game Cycle Demo*

Details:

- Show examples from the terminal of a full session demonstrating each command and progress output
- This includes battle outcomes, scores and scoreboards, etc
- Ensure at least 3 Clients and the Server are shown
- Clearly caption screenshots

Missing Caption

Not saved yet

100%

Section #4: (1 pt.) Misc

100%

Task #1 (0.00 pts.) - GitHub Details

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

Pull Request history

Saved: 4/28/2025 9:41:49 PM

Url Prompt

Weight: *40%*

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

URL #1
https://github.com/MuhammadKhan621/muk-IT114-006/pull/5

URL
https://github.com/MuhammadKha

Saved: 4/28/2025 9:41:49 PM

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

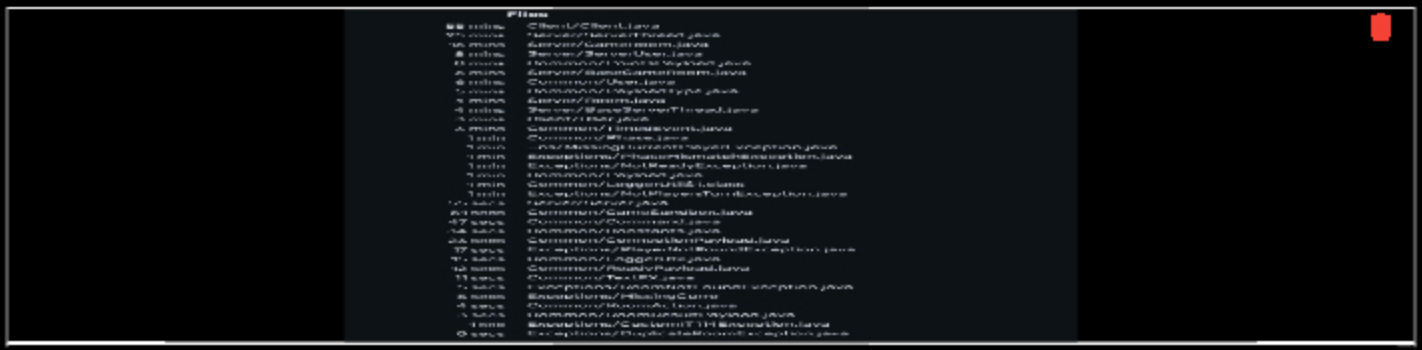
Image Prompt

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

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

Overall time



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 learned how to create pointsPayload. I learned the meaning behind each method from Section 1 to 2. Methods like onClientAdded(), OnClientRemoved(). I also learned about onSessionStart() on how it orginizes how players start during a session and whether they are ready or not. I also got to learn how to start up the server, create clients, create a room(Gameroom) and then get them ready witnessing each round pass.

Saved: 4/28/2025 9:27:53 PM

100%

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

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:

The easiest part of the assignment was Section 1 and section 2 because I was able to look at each method and understand what they did. I also found creating my gameRoom, creating clients, and starting the game(without RPS inside it) inside the terminal.

Saved: 4/28/2025 8:13:53 PM

100%

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:

The hardest part of the assignment was section 3 part 1 for me personally. That is because I couldnt figure out how to code most of it except for little. I had an idea on what I needed to put in for ServerThread. I also had an idea on how I would create if statements for the choice of RPS when a client chose it but was not able to complete it all. In honesty I find coding to be hard so all of this was challenging for me. But I did what I could to the best of my ability.

Saved: 4/28/2025 8:10:43 PM