Submission Worksheet

CLICK TO GRADE

https://learn.ethereallab.app/assignment/IT114-003-F2024/it114-milestone-3-rps-2024-m24/grade/cae6

Course: IT114-003-F2024
Assigment: [IT114] Milestone 3 RPS 2024 M24
Student: Chizorom E. (cae6)

Submissions:

Submission Selection

1 Submission [submitted] 12/11/2024 3:28:02 AM

Instructions

COLLAPSE

Implement the Milestone 3 features from the project's proposal document: https://docs.google.com/document/d/11SRMo7JkLAMM-PuuiGwI_Z-QXP3pyQ7xN3lRxwmcwCc/view Make sure you add your ucid/date as code comments where code changes are done All code changes should reach the Milestone3 branch Create a pull request from Milestone3 to main and keep it open until you get the output PDF from this assignment. Gather the evidence of feature completion based on the below tasks. Once finished, get the output PDF and copy/move it to your repository folder on your local machine. Run the necessary git add, commit, and push steps to move it to GitHub Complete the pull request that was opened earlier Upload the same output PDF to Canvas

Branch name: Milestone3

Group

Group: Basic UI
Tasks: 1
Points: 2

COLLAPSE

Task

Group: Basic UI
Task #1: UI Panels
Weight: ~100%
Points: ~2.00

COLLAPSE

Details:

All code screenshots must include ucid/date.

App screenshots must have the UCID in the title bar like the lesson gave.

Columns: 1

Sub-Task

100%

Group: Basic UI

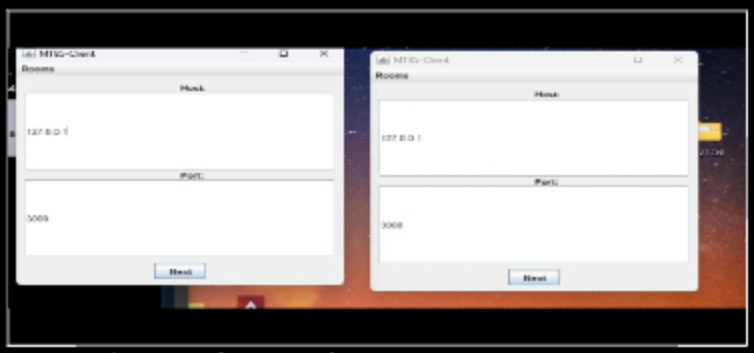
Task #1: UI Panels

Sub Task #1: Show the ConnectionPanel by running the app (should have host/port)

Task Screenshots

Gallery Style: 2 Columns

4 2 1



Connection panel screenshot

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Sub-Task

100%

Group: Basic UI

Task #1: UI Panels

Sub Task #2: Show the code related to the ConnectionPanel

Task Screenshots

Gallery Style: 2 Columns

4 2 1

code related to the connection panel

code related to the connection panel

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Briefly explain how it works and how it's used

Response:

So first a BorderLayout is created in the connection panel and then a host and port input field are created, and they already default the client to the local host and the port value. A Next button is also created and if the port value and localhost are valid, the client can move to the next card.

Sub-Task

100%

Group: Basic UI

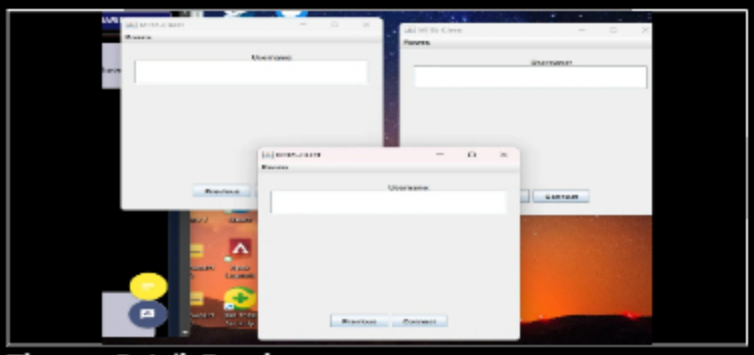
Task #1: UI Panels

Sub Task #3: Show the UserDetailsPanel by running the app (should have username)

Task Screenshots

Gallery Style: 2 Columns

4 2 1



The userDetailsPanel

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Sub-Task

100%

Group: Basic UI

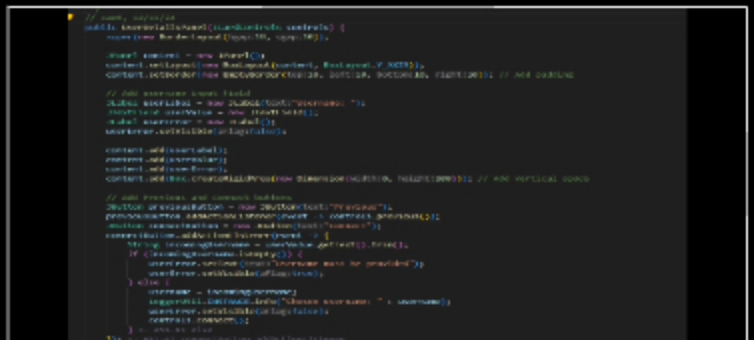
Task #1: UI Panels

Sub Task #4: Show the code related to the UserDetailsPanel

Task Screenshots

Gallery Style: 2 Columns

4 2 1



Code related to the userdetails panel

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Briefly explain how it works and how it's used

Response:

A border layout for the user details panel is created and then an input field is also created and this is where clients can input their username. There is a previous button which allows users to return to the previous card and there is also connect button that connects them to the server if they enter in a valid and non null username.Once they enter in their username they will be taken to the lobby.

End of Task 1

End of Group: Basic UI
Task Status: 1/1

Group

100%

Group: Game Area
Tasks: 5
Points: 7

COLLAPSE

Task

100%

Group: Game Area
Task #1: ReadyCheck UI Panel
Weight: ~20%
Points: ~1.40

COLLAPSE

Details:

All code screenshots must include ucid/date.
App screenshots must have the UCID in the title bar like the lesson gave.

Columns: 1

Sub-Task

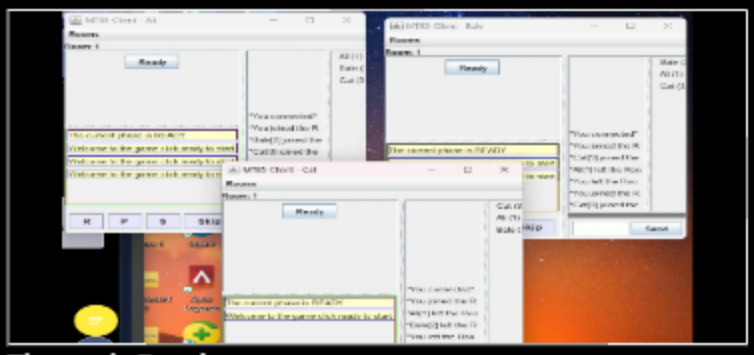
100%

Group: Game Area
Task #1: ReadyCheck UI Panel
Sub Task #1: Show the screen with the ready panel open in a fresh session

Task Screenshots

Gallery Style: 2 Columns

4 2 1



The ready Panel

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Sub-Task

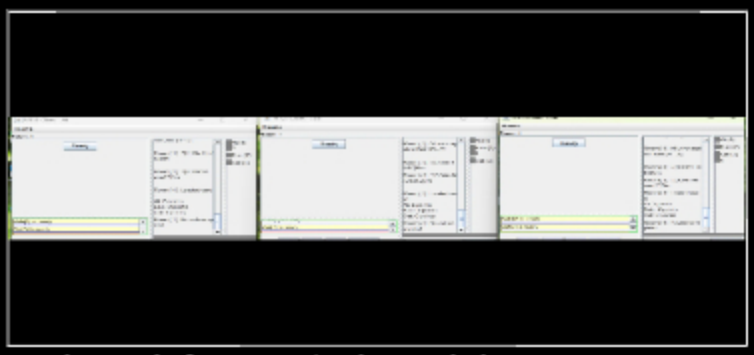
100%

Group: Game Area
Task #1: ReadyCheck UI Panel
Sub Task #2: Show the screen with the ready panel open after a session ends (there should be output in other parts of the UI showing this)

Task Screenshots

Gallery Style: 2 Columns

4 2 1



Ready Panel after a session has ended

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

End of Task 1

Task

100%

Group: Game Area
Task #2: User List
Weight: ~20%
Points: ~1.40

COLLAPSE

Details:

All code screenshots must include ucid/date.
App screenshots must have the UCID in the title bar like the lesson gave.

- Show the username and id of each Player

Columns: 1

Sub-Task

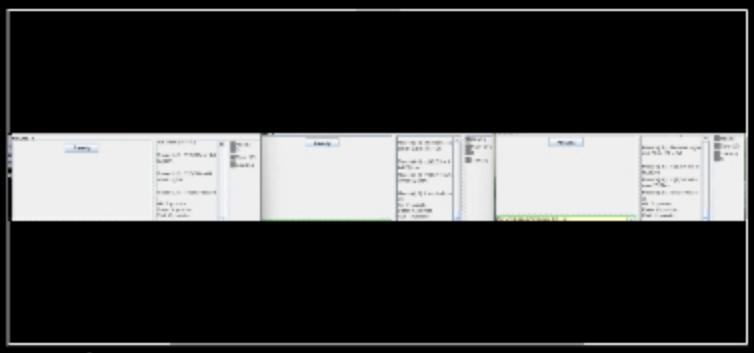
100%

Group: Game Area
Task #2: User List
Sub Task #1: Show the username/id of each player, current points, pending-to-pick indicator, elininated-indicator (list should appear in score order across all clients) (show a few examples)

Task Screenshots

Gallery Style: 2 Columns

4 2 1



User List Area

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Sub-Task

100%

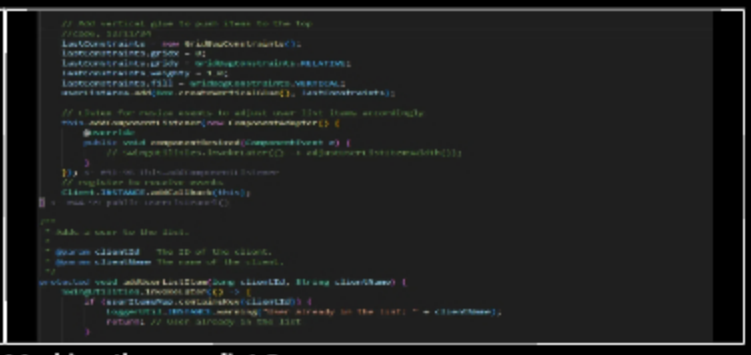
Group: Game Area
Task #2: User List
Sub Task #2: Show the related code (from server-side to UI) that marks the user list item properly

Task Screenshots

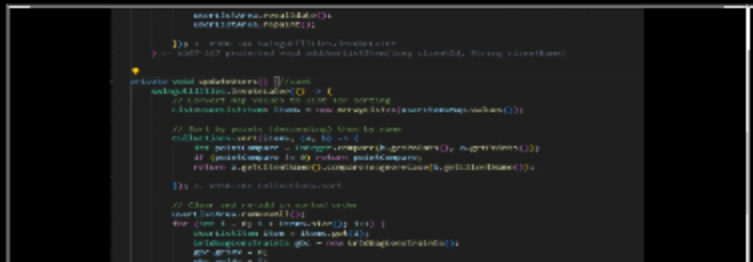
Gallery Style: 2 Columns

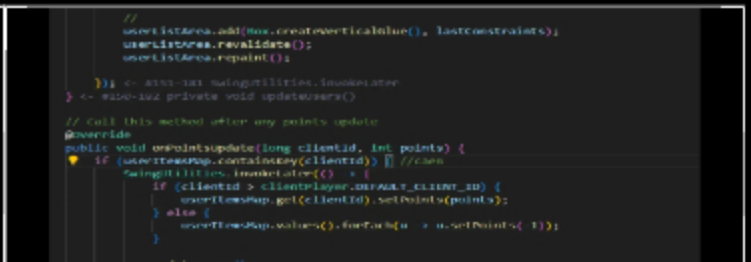
4 2 1

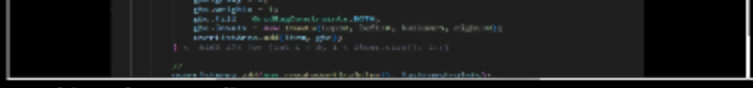
Marking the user list 1



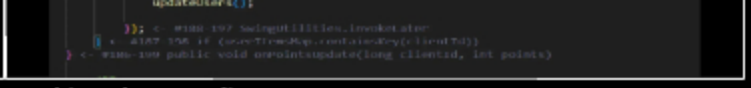
Marking the user list 2



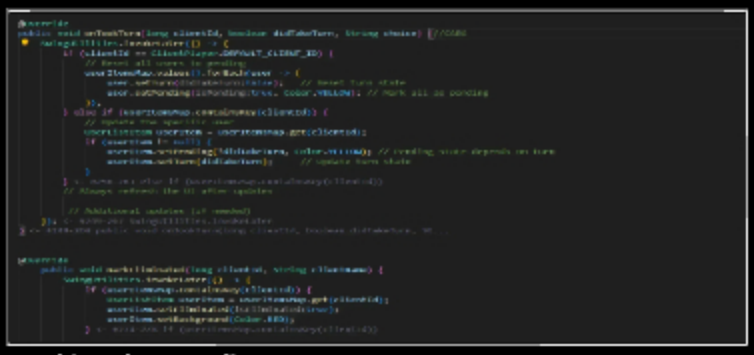




Marking the user list 3



Marking the user list 4



Marking the user list 5

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

When the game begins users are first added via addUserListItem. The add user list item first checks the the client ID to verify the client and avoid duplicates. Then a userlist item is created with the user's username and their client ID. Then that Userlistitem gets added to the UserListArea before, along with being added to the user items map. Then everything gets revalidated and repainted in order to show the changes that where made to the user list area. On took turn first checks if all users are valid users, and if they are the turn and pending state of the user gets updated. If the user didn't take their turn, they are listed as pending. If they did take their turn, then they are not shown to be marked as pending. Update points makes sure the user exists, then it changes the points based on that information, then it calls on update users which reorganizes the list based on points descending and then rebuilds the the user list area. The markEliminated makes sure the user is in the useritemsmap and then it sets the users eliminated state to true. I had a few issues with this part when it came to sorting users in descending order based on their points, allowing clients to see other clients points at the end, and marking users as eliminated and pending.

End of Task 2

Task

100%

Group: Game Area
Task #3: GameEventPanel
Weight: ~20%
Points: ~1.40

COLLAPSE

Details:

All code screenshots must include ucid/date.
App screenshots must have the UCID in the title bar like the lesson gave.

Columns: 1

Sub-Task

100%

Group: Game Area

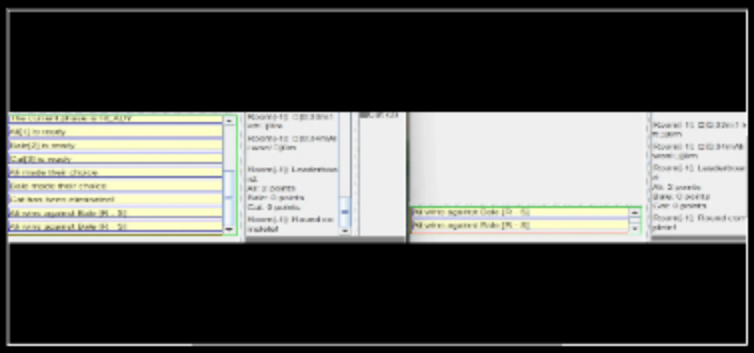
Task #3: GameEventPanel

Sub Task #1: Show the picking choices and battle log messages from Milestone 2

Task Screenshots

Gallery Style: 2 Columns

4 2 1



Picking choices and battle log messages from Milestone 2

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Note how the battle win/lose is determined/handled

Response:

First, a border layout is created for the game event panel. The text is set and so are scrolling capabilities, so that the client may scroll down on any new messages.The first thing called is onReceivePhase which adds text to the panel that shows the phase of the game. The recieveReady method adds text based on whether or not the client is ready. The OnTookTurn will get the client's name from their ID and then display that they have taken their turn along with their choice. The timer update, shows the time at the top of the game events panel. The onMessagerecieve receives messages from the battle log and then adds it text. If player one wins the game will return one and if player one doesn't win it returns negative 1. Any player who wins gets a point and the other player or players get eliminated and their name and who they beat gets displayed. That is how a winner gets determined.

Sub-Task

100%

Group: Game Area

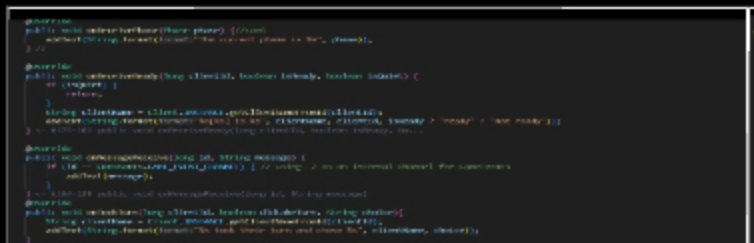
Task #3: GameEventPanel

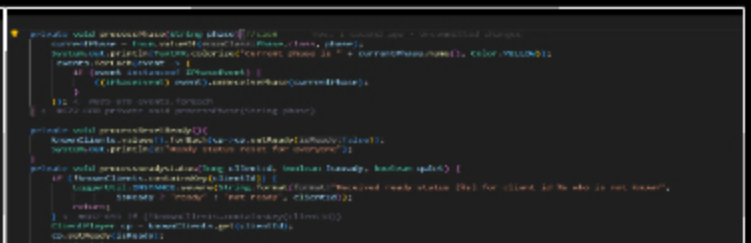
Sub Task #2: Show the code for the UI flow (Client receiving to UI) for each example

Task Screenshots

Gallery Style: 2 Columns

4 2 1





Code for the UI flow

Client receiving 1

client receiving 2

clients receiving 4

client receiving 5

Client receiving to ui 6

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

The client will receive points updates, messages, ontookTurn, whether or not the player was eliminated, the player phase, and whether or not the client was ready. The processing methods in the client now have new components that receive these updates, iterate through the event listeners, check if the listener implements the event, if it does the event gets cast and the method is called on, and the listener will handle the event. For instance, with process points, a client will get the points update, then it will find the IPointsEvent and then call on onPointsUpdate, the method in the listner, and then the listeners will take care of the update.

End of Task 3

Task

100%

Group: Game Area

Task #4: Choices Area

Weight: ~20%

Points: ~1.40

COLLAPSE

Details:

All code screenshots must include ucid/date.
App screenshots must have the UCID in the title bar like the lesson gave.

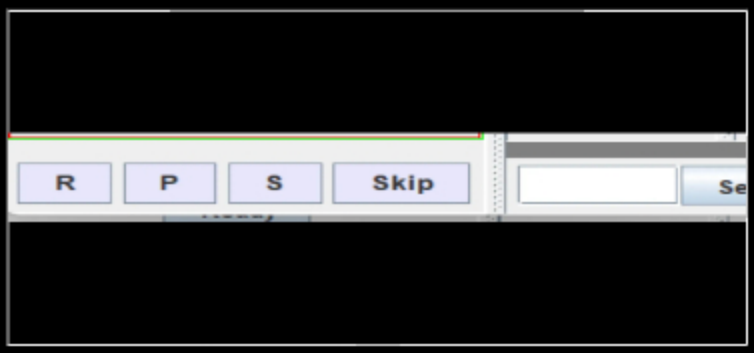
Columns: 1

Sub-Task
100%
Group: Game Area
Task #4: Choices Area
Sub Task #1: Show the UI representing the possible choices

Task Screenshots

Gallery Style: 2 Columns

4 2 1



UI representing the possible choices

Caption(s) (required) ✓

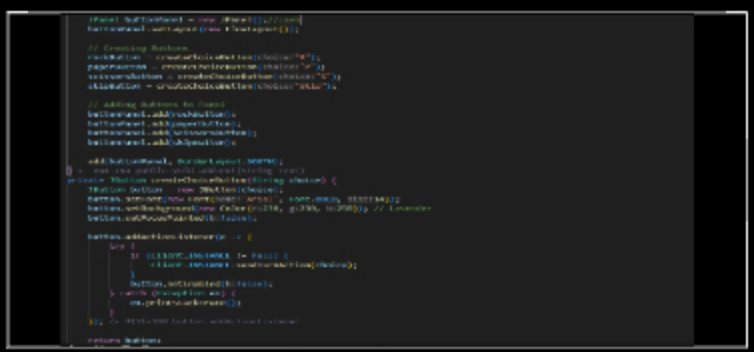
Caption Hint: *Describe/highlight what's being shown*

Sub-Task
100%
Group: Game Area
Task #4: Choices Area
Sub Task #2: Show the code related to these buttons and their interaction

Task Screenshots

Gallery Style: 2 Columns

4 2 1



code related to the creation of button

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

First I created a Jpanel to hold the buttons and arranged the buttons horizontally using a flow layout. Then I created the buttons for rock, paper, scissors, and skip. For instance, rockButton = createChoiceButton("R");. Then I added the buttons to the panel by using buttonPanel.add and added them at the bottom of the layout by using borderlayout.south. Then I implemented logic for the button, so an action listener is used to determine when a button is clicked and if it is clicked, the selected choice is sent over to the client, then the button is disabled so the client can't click on it again in the same round.

End of Task 4

Task

100%

Group: Game Area
Task #5: Countdown Timer UI
Weight: ~20%
Points: ~1.40

COLLAPSE

Details:

All code screenshots must include ucid/date.
App screenshots must have the UCID in the title bar like the lesson gave.

Columns: 1

Sub-Task

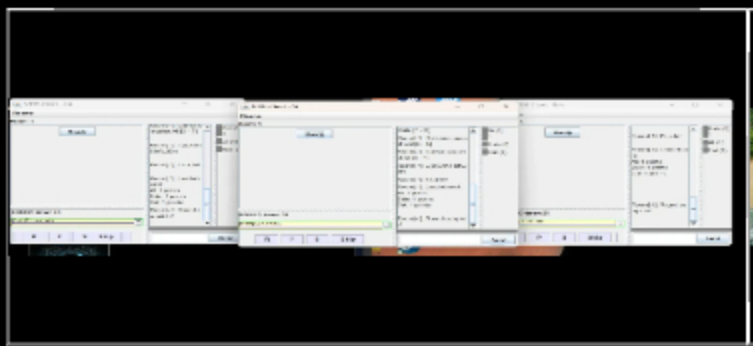
100%

Group: Game Area
Task #5: Countdown Timer UI
Sub Task #1: Show the UI of the countdown (few examples to show it changes)

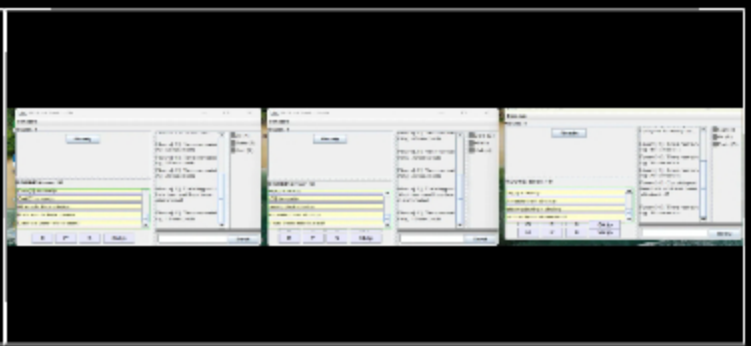
Task Screenshots

Gallery Style: 2 Columns

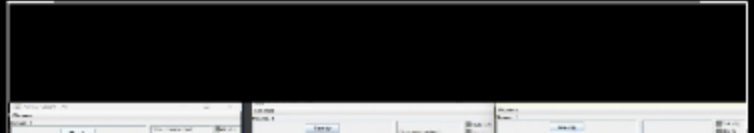
4 2 1

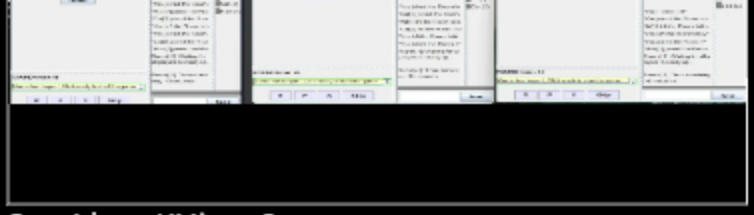


Countdown timer UI 1



Countdown timer UI 2





Countdown UI timer 3

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Sub-Task

100%

Group: Game Area

Task #5: Countdown Timer UI

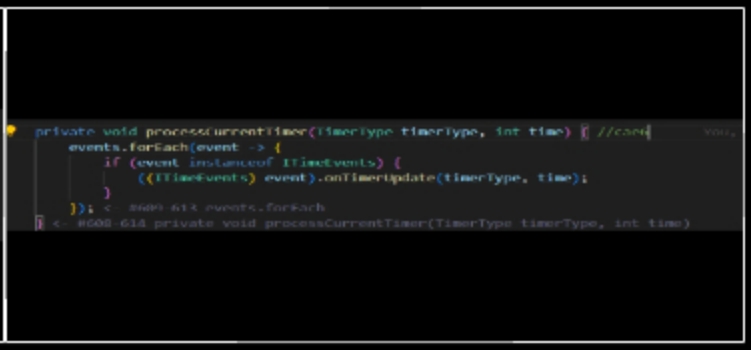
Sub Task #2: Show the code related to managing the timer

Task Screenshots

Gallery Style: 2 Columns

4 2 1

Code related to managing the timer



Code related to managing the timer 2

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works, also note if you're doing two separate timers or just syncing the ticks (or something else)

Response:

The process current timer method calls on the onTimerUpdate with the current timer type and time. The onTimerUpdate shows the timer type and the remaining time and if the timer is greater then zero, the timer will be visible.

End of Task 5

End of Group: Game Area

Task Status: 5/5

Group

100%

Group: Misc

Tasks: 3

Points: 1

COLLAPSE

Task

100%

Group: Misc
Task #1: Add the pull request link for the branch
Weight: ~33%
Points: ~0.33

COLLAPSE

Details:
Note: the link should end with /pull/#

Task URLs

URL #1
https://github.com/Cae6/cae6-IT114-003/pull/11

URL
https://github.com/Cae6/cae6-IT114-003/pull/11

End of Task 1

Task

100%

Group: Misc
Task #2: Talk about any issues or learnings during this assignment
Weight: ~33%
Points: ~0.33

COLLAPSE

Task Response Prompt

Response:

I had a few issues with this milestone a few of which, unfortunately, could not be solved in time. One of the first problems is that the same message may appear more than once in the game events panel, depending on which client made the action first. I'm also having a few problems with my User list panel. For points, I can only seem to get the points to apply for one client at a time and every client can only see their points in the user list panel. I also was not able to sort them in the panel according to their points due to my update user not working. When it comes to marking players eliminated and marking them pending, I also had issues because the colors for marking users as eliminated or pending do not show up, when certain events that call for them happen.

End of Task 2

Task

100%

Group: Misc
Task #3: WakaTime Screenshot
Weight: ~33%

Points: ~0.33

COLLAPSE

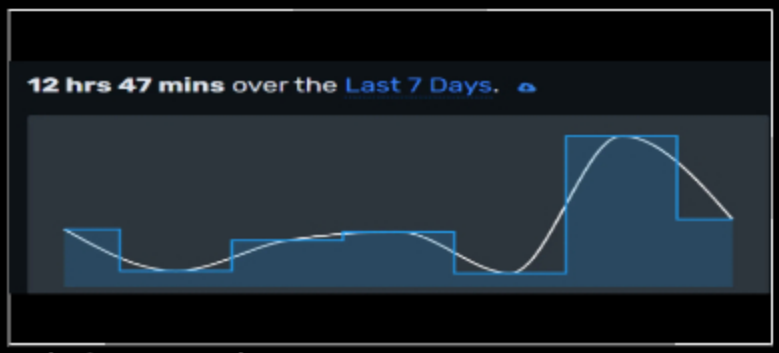
Details:

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: 2 Columns

4 2 1



Wakatime screenshot

End of Task 3

End of Group: Misc
Task Status: 3/3

End of Assignment