Progress and Difficulties encountered (and some resolved):

1. Had to ensure the functionality of starting screen music button
    a. Challenging to allow alternating on and off music just using HTML only
    b. W3 schools help me with the syntax of introducing audio in JS
    c. Wanted to just do autoplay music but not every user will like it and some browsers may not be compatible and may just block autoplay music (like mine)
    d. I decided to introduce the options of turning on and off → since there was an 'on' condition and 'off' condition, I could use an if-else loop which took a while to figure out
2. Username input sometimes gave null for output
3. Found my original starting screen's title very ugly, since there weren't as many fonts as I wanted
    a. decided to do an animated background using After Effects to look more like a game starting screen
    b. Couldn't get it to load because i rendered in MOV so i had to convert things to MP4 to get it to work
4. Darkened backgrounds of subsequent backgrounds afford users more focus and attention toward the more important, interactive parts of the game which is the foreground of the screen
5. Replaced URL background image with a local file as it was easier to manipulate CSS like this
6. Managed to add event listeners to the demo game page such that the clicks are registered, managed to get the score counter working
7. Centralized "3 2 1 Start" text in CSS with "text-align: center;"
8. Decided to add background video for the background of the main game so it models rhythm games better, would be less boring for users
9. Made "Re-try Demo" button functional, edited CSS such that it doesn't overlap with other elements on the html
10. Played around with CSS to change opacity for the moving beats so that it doesn't appear too early for users and crowd the screen (makes it overwhelming for users)
    a. done by using keyframes animation in CSS (inspiration: google dino game)
11. Fixed the main game mechanics: needed event listener to listen to clicks, introduced conditions for the different accuracies of clicks ("perfect" for almost 100% accuracy, "good" for relatively high accuracy and "missed" for beats that are missed / inaccurate), introduced counters to tabulate the scores for the respective categories, put them into the score summary html
    a. Needed 3 conditions other than if-else, did a quick google search and found out there was an if, else-if, else syntax i could use
    b. The original code was very messy, with most of the functions and variables being stuck in click event listeners where i listed individually for every single beat → decided to take them out and generalize them into functions which i can apply to specific beats and event listeners
    c. Relied on console.log to make sure my beat was printing on the screen

12. Had to find top position of the beats to determine the height at which the click should be made to be considered accurate
13. Added song selection menu so that user has more choices and variety of songs to play
    a. The pictures for the menu overlapped with one another initially, decided to put them in table form and added a hover so that users know what they are selecting
14. Decided to make blank beats with hole in the middle so its easier for user to aim
15. Couldnt add a nice glow or hover to the beat pictures so i decided to do text animation to indicate to users that they have managed to click and interact with the game