Personal Project – TypeRacing
Caden Kishimoto

TypeRacing Requirements Doc (Things I Want to Create/Implement)

This is a general doc for me (during the planning phase) to get my thoughts out there for what I want to build – and to double-check that I'm correctly meeting my own req's while implementing. It has since been finalized into a full game ruleset and list of design req's.

- Create a personal, custom version of TypeRacer (like the one on play.typeracer.com)
- Use Gradle to build the project
- Create a protocol using JSON
- Create a leader server that connects to game servers (like nodes)
    - Leader should listen for incoming connections from both clients and game servers simultaneously (in threads)
    - Each time a client is being connected to a game server, leader should check if it still has a connection to the game server, if it's not already full (can be checked on game server), and if a game isn't in progress (can be checked on game server)
    - Leader connects clients to the most populated, available game server (to fill one game server at a time)
- Game Rules
    - Up to 5 players per game
    - Sentences to type are in a text file called "prompts.txt" – which is a compilation of sentences from lipsum.com (the Lorem Ipsum description page)
    - A game has 1 sentence (varying in length)
    - 60 second time limit for each game
- Client (Player) will connect to leader server first, which will then connect the client to an available game server
- Client Stuff:
    - Client will first be prompted to either sign-up or login with leader server
        - For sign-up/login, get username and password (store new sign-up into system with leader server, and use "users.txt" to keep users' data persistent between server restarts)
    - Client will next have a menu to select to either join a new game, see the leaderboard, or exit (again communicating with leader server)
    - Leaderboard will show the specific client's info first, followed by all other registered players (ordered by decreasing win count). Each leaderboard entry will include specific username and their number of wins. Leaderboard is retrieved from the leader server based on users' info
    - Exit option will safely let client exit (and have goodbye message)
    - New game will let leader server match client with a game server and let play game from there (If no available servers, message will be given saying there's no servers available and to try again later.")

    - During game, player will see a sentence and needs to type all characters correctly. If incorrect, player will need to type it again (see Game Server Stuff for client handling)
    - Once game is finished (or leaderboard is displayed), menu displays again
    - If game server closes (for whatever reason), client should be redirected back to menu
    - If leader server closes, client should safely exit
    - Client should not internally keep game state/leaderboard (keep data server-side)
- Game Server Stuff
    - Each game server will have its own threaded connection to client
    - Once first client connects, 30 second countdown until game starts while more players join (broadcasts will be sent every 5 seconds, like "Game starts in 5 seconds!")
    - Once game has started, begin 60 second countdown and broadcast the same randomized sentence from "prompts.txt" to all clients (not every second is broadcast, only for 45/30/15 seconds remaining)
    - Game server will check client's input in its own threaded connection
        - If correct, game server will be notified and rank the player (sends client their ranking and a message of "Waiting for other players to finish…")
        - If incorrect, game server won't rank player for the session (yet) and will instead send a message to client to try again
    - Once all players finished (or time ran out), game server will broadcast rankings to all players (1^{st}-5^{th}) and close their connections. Game server should also send a connection to leader server saying who won (to update leaderboard). If a client/player disconnected mid-game without finishing, server should not rank client and should instead display DNF (did not finish)
- Protocol and networking should be robust with error handling. Clients, game servers, and leader servers should not crash if connections are disrupted between one another