1. Project Members:

Austin Swartley, Evan Binkley, Peter Schafer

1. Communication plan
   1. Snapchat, Discord, and Github will be used for general communicating and collaboration.
   2. Zoom and Discord will be the main method of conducting team meetings and focused collaboration efforts.
2. Task Management
3. Program design (Theorizing)
   1. JSON packet formatting and organization idea (Create format)
   2. Create possible schedule and due dates.
   3. Have consistent programming format practices for readability.
   4. Create sequence diagram.
4. Application Design (Writing)
   1. Client (Peter Schaefer)
5. Basic nickname sending
6. JSON packet management and sending
7. Server-to-Client push receiving and printing.
   1. Server (Austin Swartley & Evan Binkley)
8. Basic server user setup (Executable startup process)
9. Thread management for user connections
10. Stream handling per thread, and JSON packet handling
11. Server-to-Client message pushing and updating.
12. Signal handling and graceful closing
    1. Library management
13. Library (lib.rs) update as per sufficient needs (When applicable)
14. Programming Languages
    1. The client-side program for our project will be written and running in Rust.
    2. The server-side program for our project will also be written and running using Rust.
    3. The library for both programs will also be written using Rust, and shared between both programs for what is needed.
15. Project Scope
    1. Client-side message and nickname sending
    2. Server-side message receiving
    3. Server-side message management and message pushing/updating
    4. User inputted program termination
16. User interaction
    1. Client-side
17. User inputs username/nickname
18. User sends messages to terminal for processing
19. Client updates board of messages as server pushes and receives them
    1. Server-side
20. Server awaits port number for socket
21. Server processes information and pushes packet based data to clients based on receiving.
22. User can input termination signal for graceful close.
23. Shared functionalities
    1. JSON packet formatting (May not be necessary for including)
    2. Encoding and decoding sent and received packets
24. Application protocol
    1. Specifically designed JSON packets (Formatted in previously mentioned manner) in order to convey data sent from client to server, as well as server to client.
25. Embedded