Engineering Team Simulation

We are hiring engineers who are excellent at writing software and working on a team. This means understanding the relationships between servers and clients, making smart architecture decisions, creating code others can easily use/maintain, and explaining complicated technical ideas clearly and concisely in English.

The initial evaluation simulation below allows you to demonstrate your expertise by designing and implementing a simple server with basic network requirements. More importantly, it extends the opportunity to impress us with you how complete the task. Your completed simulation will provide the primary basis for discussion during the technical phone interview.

Part 1: The Code

Implement a chat server that allows connecting via telnet using a language of your choice.

Attached is a log of the communication between a client and the server, including a basic set of commands and responses. Re-implement the implied protocol exactly in your own server, then feel free to extend it with additional helpful functionality. *Ensure the given functionality continues to work correctly.*

Use this as an opportunity to showcase your ability to think critically, design strong architecture, write maintainable code, plan for expected (and unexpected) use cases, evaluate implicit criteria, and work within existing constraints. You should be confident that it will perform as expected and not break when we test it with a few users.

Your code will be looked at by our senior engineers, so make sure they are impressed and that it represents your best ability. Writing beautiful, well-organized code and implementing additional features is more valuable than quickly finishing the minimum assignment, so take your time to show us your stuff.

Part 2: The Essay - Champion Your Technical Decisions

As with your programming submission, use this as an opportunity to demonstrate your ability to craft a clear and well supported argument. You should write at least a few paragraphs explaining the decisions you made when designing your submission. Illuminate the major problems and how you solved them. Demonstrate how you approached the problem and why your solution is excellent at both a small and large scale.

Make it clear that you looked at the problem from all angles, planned ahead, implemented a great design, covered the edge cases, and that your code is production ready. Additionally, your essay should be easy to read, coherent, technically deep, and well written. Like your code, your essay should be ‘production’ quality.

Part 3: Submission and Review

Please host your server somewhere accessible from the internet (check out Amazon's free tier AWS offer if necessary), package your source code and essay, and return your submission to your point of contact within 48 hours. If you have any questions, please ask them immediately.

Additionally, please send us the following:
1) The IP address to connect to the chat server with telnet
2) .zip or a .tar.gz file including your source code
3) PDF of your text submission.

We will check the functionality by the next day. If functional, we will immediately pass your submission on to our senior engineers for review. We will inform you of the outcome.

Have fun with this exercise! We hope to be working with you soon!

Log from telnet client to chat server:

Below is an example of a short session using telnet to connect to a server implementing the basic requirements. Left arrows (<=) indicate a message from the server to the client and right arrows (=>) indicate a message from the client to the server. These are given to help you understand the log -- they are **not** part of the protocol.

```
gc$ telnet x.x.x.x 9399
Trying x.x.x.x...
Connected to x.x.x.x.
Escape Character is '^'

<= Welcome to the XYZ chat server
<= Login Name?
=> gc
<= Sorry, name taken.
<= Login Name?
=> gc_reviewer
<= Welcome gc_reviewer!
=> /rooms
<= Active rooms are:
<=  * thebestchat (5)
<=  * hottub (2)
<= end of list.
=> /join thebestchat
<= Entering room: thebestchat
<=  * gc
<=  * gc_reviewer (** this is you)
<=  * foo
<=  * user1
<=  * user2
<=  * y2kcrisis
<= end of list.
<= foo: welcome gc_reviewer!
=> hi there!
<= gc_reviewer: hi there!
<= * new user joined thebestchat: user7
=> gotta go!
<= gc_reviewer: gotta go!
=> /leave
<= * user has left chat: gc_reviewer (** this is you)
=> /quit
<= BYE
Connection closed by foreign host.
```