Project 2

Assignment overview

The purpose of this assignment, as I understand it, is to expand on the previous single-threaded key-value store server by creating a multi-threaded server that can handle concurrent client requests through Remote Procedure Calls (RPC) using Java RMI. This multi-threaded design allows the server to perform PUT, GET, and DELETE operations in parallel, improving efficiency in a distributed environment where multiple clients may access the server simultaneously. Additionally, Docker containers were used to package the server and client, ensuring that the system runs consistently across different environments. Shell scripts were also developed to streamline the build and run process for easy deployment and testing of both single and multi-threaded clients.

Technical impression

Developing this multi-threaded client-server system with Java RMI and Docker provided a deeper understanding of RPC and concurrency in distributed applications. The main challenge was implementing the Java RMI framework in Docker, especially ensuring that the client could locate and communicate with the server across containers. Setting up a custom Docker network helped, but I also needed to adjust the client's lookup address to the server container's name instead of localhost to enable inter-container communication. This debugging process showed the importance of networking configurations in containerized applications.

Handling concurrency on the server also presented a new layer of complexity, as I had to make sure each client request was managed properly without conflicts. Using Java's built-in synchronization methods and thread-safe collections, I was able to ensure that the server responded accurately and consistently, even when accessed by multiple clients. Testing with MultiClientTest simulated real-world concurrent access and underscored the value of thread-safe programming. Additionally, by using Docker and shell scripting, the deployment and testing of the system became far simpler and more reliable. This assignment has improved my confidence in setting up multi-threaded, distributed systems and strengthened my skills in handling inter-process communication and concurrency challenges.