Assignment Overview

The Paxos Key-Value Store assignment is designed to provide practical experience with distributed systems and consensus algorithms, specifically the Paxos protocol. The primary goal of this assignment is to implement a simple key-value store using Paxos to ensure consistency and fault tolerance across multiple servers. By doing so, it simulates how distributed systems maintain data integrity even when some nodes fail or behave unpredictably. The system is expected to store key-value pairs, handle concurrent read and write requests, and maintain consistency across replicas in the presence of failures. The Paxos algorithm is used to manage the consensus between multiple servers when agreeing on a value for a key. This assignment allows students to learn the inner workings of distributed consensus and implement an efficient fault-tolerant system, simulating real-world use cases of distributed storage systems.

Technical Impression

While carrying out this assignment, I gained valuable insights into the complexities of distributed systems and the Paxos consensus algorithm. Initially, implementing the Paxos protocol was challenging due to its intricate nature, especially in terms of handling network failures, message ordering, and ensuring consistency across servers. However, through careful study of the protocol's phases—prepare, propose, and accept—I was able to implement a basic version of Paxos that allowed multiple nodes to reach consensus and maintain data consistency.

One of the most rewarding aspects of the assignment was seeing how the system could handle various failure scenarios. For example, when one server crashed or failed to respond, the remaining servers were able to continue functioning, with the consensus process still ensuring that the data remained intact and consistent. Although the assignment provided a solid foundation for distributed consensus, there were areas where improvements could be made, such as handling network partitioning and optimizing the protocol's performance under high loads. I would also recommend further clarification on edge cases, particularly regarding message timeouts and retry mechanisms, as these could affect system behavior in real-world applications. Overall, this assignment provided a hands-on approach to understanding the principles of distributed systems and how they apply to building fault-tolerant services.