CSE5306, Distributed Systems
Fall 2022, Project 3
Due date: 11:59pm, Dec. 8, submission through Canvas.

Please read this:
1) Project assignments are to be completed by teams.
2) Total points possible: 100 pts.
3) Please add the following statement in the beginning of your submission.
   I have neither given or received unauthorized assistance on this work
   Signed: *Date:*

Assignment-1 (50pts): We have discussed the basic Paxos protocol that selects a single proposed value from multiple proposers. In this assignment, research on the multi-value extension of the basic Paxos protocol, i.e., multi-paxos, by reading tutorials, online postings, technical papers and answer the following questions.

1. How does the multi-paxos protocol ensure multiple nodes to agree on a consistent ordering of a sequence of values and how is it different from running the basic Paxos protocol for multiple rounds?
2. What are the performance and scalability issues of the multi-paxos protocol?

List all the references you used in your answer.

Assignment-2 (50pts): Implement a 2-phase distributed commit (2PC) protocol and use controlled and randomly injected failures to study how the 2PC protocol handles node crashes. Assume one coordinator and at least three participants in the 2PC protocol. Similar to the previous projects, we use multiple processes to emulate multiple nodes. Vote requests and responses should be carried out using communications. Each node (both the coordinator and the participants) devises a time-out mechanism when no response is received and transits to either the *abort* or *commit* state. Design a controlled failure test to evaluate whether the implemented 2PC protocol leads to consistent states across the coordinator and participants. For simplicity, you can assume that only one node fails in the controlled test. Evaluate different possibilities of failures (e.g., coordinator fails before or after sending *vote-commit*). To emulate a failure, you can impose a much longer delay at a failed node than the time-out period used by other healthy nodes. Node print their states before termination. Verify all nodes converge to the same state regardless of the failure.

Furthermore, evaluate the 2PC protocol by randomly injecting failures to any nodes (e.g., a node may be delayed emulating a failure with a probability at any point during execution). Verify the terminal state to ensure consistency.