Design Document

Protocol design

- We use a decentralized design and use ISIS algorithm to ensure that the transactions are delivered in the right order. We fail our node by closing down the socket that has connection with other nodes. To realize the four cases in evaluation scenario part. We would fail some of our nodes 10 seconds or 100 seconds after each of them started.

Reliability

- The ISIS algorithm is a consensus algorithm used to achieve consistency in distributed systems. The primary principle behind the ISIS algorithm's ability to maintain total ordering is the use of an incrementing sequence number to identify each message. The algorithm operates in the following way:
  - Sender multicasts message to everyone.
  - Receiving processes:
    - Reply with proposed priority (sequence' no.) age
      - larger than all observed agreed priorities
      - Larger than any previously proposed (by self priority.
    - Store message in priority queue
      - ordered by priority (proposed or agreed).
    - Mark message as undeliverable
  - Sender chooses agreed priority, re-multicasts message id with agreed priority.
    - Maximum of all Droposed Driorities
  - Upon receiving agreed (final) priority for a message 'm'
    - Update m's priority to final, and accordingly reorder messages in queue.
    - Mark the message m as deliverable.
    - Deliver any deliverable messages at front of priority queue..
- We can prove the correctness of ISIS algorithm in this way:
  - Suppose we have a scenario involves messages m and m2, and two processes p and p'. Suppose that process p delivers message m before m2. When process p delivers message m, it is at the head of the queue. Message m2 is either already in p's queue and deliverable or not yet in p's queue.
    - If message m2 is already in p's queue and deliverable, then the final priority of message m1 is less than the final priority of message m2.

        - If message m2 is already in p's queue but not deliverable, then the final priority of message m1 is less than or equal to the proposed priority of message m2, which is less than or equal to the final priority of message m2.
        - If message m2 is not yet in p's queue, then the same argument applies, since the proposed priority of message m2 is greater than the priority of any delivered message.
    - Now suppose that process p' delivers message m before m1. By the same argument, the final priority of message m2 is less than the final priority of message m1.
- When node A sends a message to node B, the message's sequence number must be greater than the sequence number of the last message received by node B to ensure that node B processes the message in the correct order. If node A sends a message to node C before node B, but node C receives node A's message before node B, then node C processes node A's message first and then node B's message to ensure consistency in the processing order across all nodes. Therefore, It ensures that messages transmitted between nodes are processed in the same order, thereby guaranteeing the total ordering of transactions.

Handling node failures

- To simulate the four evaluation scenarios, we will close the socket that connects our node to other nodes, causing it to fail. We will deliberately make some of our nodes fail 10 or 100 seconds after they start.
- We ensure the system could still run with the following method:
    - When one or more nodes fail, they would lost connection to other nodes. At the same time, other nodes could detect their connection loss to the failed nodes. In our code, our node has a attribute called `self.n_connect`. When it detects a failed node, we would minus its connection count by 1. Due to changes in the connection count, the critical state of delivery in the queue will also change accordingly. Once replies are received from a sufficient number of nodes, the delivery status can be set to True. This ensures that the system can continue to operate even if some nodes fail.
    - When the head of queue was stucked, which means the sender of this message is dead and didn't receive any reply for more than 6 second, the we just pop this item out of the queue and ignore this transaction.