**Project Report 3**

**I have neither received nor authorized assistance on this work.**

**Sign: 1001959376, 1002039215 Date: 10/06/2022**

**Project Overview:**

This project implements the 2-phase distributed commit(2PC) protocol and uses controlled and randomly injected failures to understand how the protocol handles node crashes.

**Implementation:**

This project consists of 1 client, 1 transactionController and 2 nodes-> nodeA and nodeB, each running processes listening on ports 2020, 2021, 2022 and 2023 respectively. Each have 2 methods listen() and send() to communicate both ways with each other.

Client requests the transaction controller to increment the value of a variable-> data. Transaction controller then sends this request to nodeA and nodeB. transactionController tries to get lock initially and then sends PREPARE to both the nodes. The nodes are then expected to respond with acknowledgment >YES, if any node fails to do so, the entire transaction is rolled back, and the data value is not incremented.

(Refer the below diagram to refer the control flow between each entity)

CLIENT

NODE A

TRANSACTION CONTROLLER

NODE B

GET

PUT

PREPARE

PREPARE

YES

YES

COMMIT

COMMIT

ACK

ACK

**Learning:**

* 2 Phase commit protocol
* Socket programming
* Git
* Exception handling
* Adding timeout for sockets in case no response received

**Issues Encountered:**

* Initial project structure
* Adding timeout in transaction controller

**Labor Division:**

* **Devyani:** TC failure handling
* **Akanksha:** Node failure handling

**References:**

* <https://docs.oracle.com/javase/7/docs/api/java/net/Socket.html#setSoTimeout(int)>
* <https://martinfowler.com/articles/patterns-of-distributed-systems/two-phase-commit.html>
* <https://www.ibm.com/docs/en/i/7.1?topic=communications-socket-programming>