CS 410/510
Large Scale Systems

Assignment 2:
Implementing Commit Protocol

Overview:
In this course, our goal is to create **MiniSpanner**. MiniSpanner takes inspiration from **Google's Spanner** database. Despite being created nearly two decades ago, Spanner still acts as the backbone for all Google's applications.

In this course, over **a span of four assignments**, you will be creating MiniSpanner. Thus, all the assignments are **incremental** as you will get an opportunity to use your existing code in your current assignment.

Goal:
The goal of your second assignment is to implement a commit protocol on top of Apache ResilientDB (incubating).

CS 410 students will implement **Two Phase Commit (2PC)** protocol.
CS 510 students will implement **Three Phase Commit (3PC)** protocol.

Key Challenges:
- ResilientDB, by default, assumes a replicated system. However, the commit protocols are run on a partitioned or sharded system.
- In a partitioned/sharded system, any partition can receive the client transaction. The partition that receives client transaction acts as the coordinator, initiates concurrency control and commit protocol.
- ResilientDB, by default, runs PBFT protocol and does not have any notion of a concurrency control and commit protocol. ResilientDB's PBFT implementation assumes that the proxy collects all the client requests and forwards them to the leader to initiate PBFT consensus. Post PBFT consensus, each replica of ResilientDB executes the transaction and replies to the client.

Key Expectations:
- You are **not expected** to modify the replicated design of ResilientDB to a partitioned design. Of course, if you want to, you can!

- You **can assume** that you are using the by default configuration of ResilientDB with 4 replicas and 1 proxy.
- You **can assume** all the transactions are sent to the proxy and the replica that proxy forwards the transactions is the coordinator for the transactions. So **only one coordinator** for all transactions.
- You **can assume** that each transaction requires all the replicas to participate in the commit protocol. Essentially, you are assuming that each transaction requires access to 4 partitions.
- You are **not expected** to run or implement any concurrency control protocol.
- You **can assume** that as soon as the coordinator receives a transaction, it starts the commit protocol.
- You **do not need** to modify the execution and reply to client steps. Post commit protocol, let every replica execute the transaction.
- You **can assume** each transaction always commits (no aborts).

Assignment Tasks:

Following are the tasks that you need to perform as part of this assignment.

1. Implement your commit protocol.
2. Measure the throughput of your commit protocol at the coordinator and the participants.

Deliverables:

As part of this assignment, you are expected to provide the following deliverables.

1. A latex report explaining how you implemented commit protocol. Please provide details of what functions you modified or added to the existing codebase.
2. In your latex report include snapshots of any function that you modified. These snapshots should show your changes.
3. Please state your machine configuration and the observed throughput/latency.