CS 410/510
Large Scale Systems

Assignment 1:
Understanding Client-Server Architecture of ResilientDB

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 first assignment is to understand the client-server architecture of Apache ResilientDB. In this course, we make use of ResilientDB as it is incubating under Apache Software Foundation.

The key advantage of using ResilientDB is that it is actively maintained by a large community. Additionally, ResilientDB provides you access to a client server architecture and can be easily spawned over the cloud. Please note that the current goal of ResilientDB is to provide developers access to a Byzantine Fault-Tolerant (BFT) system. For this assignment, you do not need to worry about understanding a BFT system.

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

1. Form a group of 3 three members.
2. Select a name for your group.
3. Each member of your group needs to fork the Github repository of ResilientDB.
4. Next, install ResilientDB on your laptop. The installation script includes all the details that you need to know regarding the installation process.
5. Next, you should run ResilientDB with the following configuration: 4 replicas, 1 proxy, and 1 client.
6. There are two ways to run ResilientDB: (1) one client request at a time, and (2) deployment script. Your group should be able to run ResilientDB through the deployment script.

7. Next, note down the throughput (tput) at each replica and latency at the client/proxy. You will observe that ResilientDB runs in an endless manner. This is how any real world system runs; once started, the system only stops due to any failure or force stop. Hence, you should report the throughput and latency at the end of 100seconds.
8. Next, start understanding the client-server architecture. Your aim should be to make clients send Key-value pairs. Additionally, you should find where do replicas receive requests from the clients and how does the process of transaction management take place.
9. You should try to see what functions are called as part of the PBFT consensus protocol. Please note you do not need to understand the PBFT protocol; you are only trying to understand what functions are being run.

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

1. Forked Github repo of ResilientDB; we will manually check this.
2. A latex report explaining how you ran ResilientDB and what was the observed throughput and latency.
3. Please state your machine configuration as throughput/latency depends on the machine being run.
4. Summarize the implemented architecture of ResilientDB. In your summary, please include all the functions being touched and describe the functionality of each function in a line or two.