**Distributed Systems - Project 1**

**DESCRIPTION:**

As part of Distributed Systems Project 1, we have implemented file operations includes upload, download, rename and delete operations which are based on message oriented, client-server communication and a computation service using remote procedure call (RPC) based communication.

**Question 1:**

We have implemented all the programs using python. For Question 1, we have implemented a single threaded server which connects with the help of sockets. A socket is bound to a port number so that the TCP layer can identify the application that data is destined to be sent to. Single threaded means we open a single connection and server can respond to only single client at a time. In this program we have implemented four basic operations Download, Delete, Rename and Upload. We have used a connection-oriented protocol in which the client and server first establishes a network connection, negotiates the operation to be performed and carry out the file transfer in the same connection.

**Question 2:**

In Question 2, we have implemented a multi-threaded server which connects with sockets and multithread. In multi-threaded server, server socket can respond to multiple clients at a time. The multithread is executed with the help of locking mechanism. In distributed systems, a locking mechanism is helpful when two or more than two clients are trying to access the same file, only one can access it. Each request is working on its own thread. Whenever multiple threads act on the same resource, then threads are locked to work first in first out. Distributed locks are helpful to improve the efficiency of services or implement the absolute mutual exclusion of accesses. Mainly, locking mechanism used in deleting and renaming process. The main challenge we faced in converting single thread server to multi thread server is to add threads and lock mechanism to single socket thread program. We also faced difficulty in debugging the code as we also had to maintain concurrency. When one of the clients is renaming one of the files another client will not be able to change or modify the file. We have also tested several test cases by involving both the clients. Example: one client deleted the file from the server, another client cannot download or rename the same file.

**Question 3 and 4:**

In Question 3 and 4, we have implemented synchronous and asynchronous Remote Procedure Call (RPC). Remote Procedure Call (RPC) is a technique for building distributed systems. Basically, it allows a program on one machine to call a subroutine on another machine without knowing that it is remote. RPC is not a transport protocol: rather, it is a method of using existing communications features in a transparent way.

**ISSUES:**

The major issues found in implementing asynchronous RPC, as I need to implement without using modules. First thing needs to understand how the code should be written that can perform RPC process efficiently.

The challenges faced in converting single threaded to multithreaded is that to understand how the process should for more than one client and the server should be in loop that can give responses according to the client input.

The purpose of a lock is to ensure that among several nodes that might try to do the same piece of work, only one does it (at least only one at a time). That work might be to write some data to a shared storage system, to perform some computation, to call some external API