Implementation ;

- The daemon process will be acting as a serve, where it will be looking for other processes (clients in this case) to connect.
- We create multiple client processes using fork() command in the processes.cpp file. These processes try to connect with our server(daemon).
- Once a connection is established, a counter is incremented. This counter helps us in knowing the number of processes that are participating in the network.
- Daemon process will send its counter value to the process that has established the connection with it.
- Then the client process with calculate the difference between its own logical counter and the daemons logical counter.
- This difference is sent back to the daemon.
- Daemon will keep updating the avg time when ever it receives an offset from a new connection and updates the counter for the number of offsets it has received.
- Once this counter equals the number of processes that have actually established connection, it sends out the difference between the offset and the final average time.
- Each process will update its logical clock based on the difference it has received.
- And, finally daemon will also update its logical clock value.

Results:

- All the logical values of the respective processes before and after update are printed on the console.
- An image reference can be found in the Results folder of Assignment 1 folder.

Lessons learnt and Issues encountered:

- Took good amount of time that this can be implemented through concurrent web server -client approach.
- Had some trouble in getting an idea on how to know that we have calculated the final avg time and know we can formulate the difference.
- One of the main lesson learnt, any inter process communication that we are trying to establish can a server- client architecture.
- When we try to keep a counter, its good to call function which act as get functions, instead of directly checking for the value of the variable.