Name:Bodapatla Jeevan Reddy
Id:1001949287

Execution steps:

1. At the beginning, run "server_b.py" to have Port 55555 open to communicate with incoming connection.

Command: "**python server_b.py**"

2. Then run `server_a.py` and type a port number so Therefore**,** PORTID is performed differently than 55556. Internally connect the socket to 5000 Get the data from `directory_b`.

Command:**"python server_a.py"**

3. Next we will execute file client.py all the data from server_a and server_b will be displayed in sorted form.

Command:**"python client.p"**

4. In next step we use commands to lock the file and unlock the file.

Lock the file Command:**`python client.py -lock -{file_index}`**

Unlock the file Command:**python client.py -unlock -{file_index}`**

The execution is done

Reference Links:

WATCHDOG: https://xiaoouwang.medium.com/create-a-watchdog-in-python-to-look-for-filesystem-changes-aaabefd14de4

https://xiaoouwang.medium.com/create-a-watchdog-in-python-to-look-for-filesystem-changes-aaabefd14de4

https://thepythoncorner.com/posts/2019-01-13-how-to-create-a-watchdog-in-python-to-look-for-filesystem-changes/

https://docs.oracle.com/javase/tutorial/networking/sockets/clientServer.html#later

https://realpython.com/python-sockets/

List of libraries:

For multiple threading <threading> std-lib
For request listening from client to socket <socket> std-lib
Acess to files and directories<os> std-lib
To process <json> std-lib
list of files from a directory <gt_list> function from <myutils> custom-lib
observer to capture file-system events <Observer> from
<watchdog.observers>
event-handler <FileSystemEventHandler> from <watchdog.events>
Deleting a file <remove> function from <os> std-lib
Coping a file <copyfile> function from <shutil> std-lib