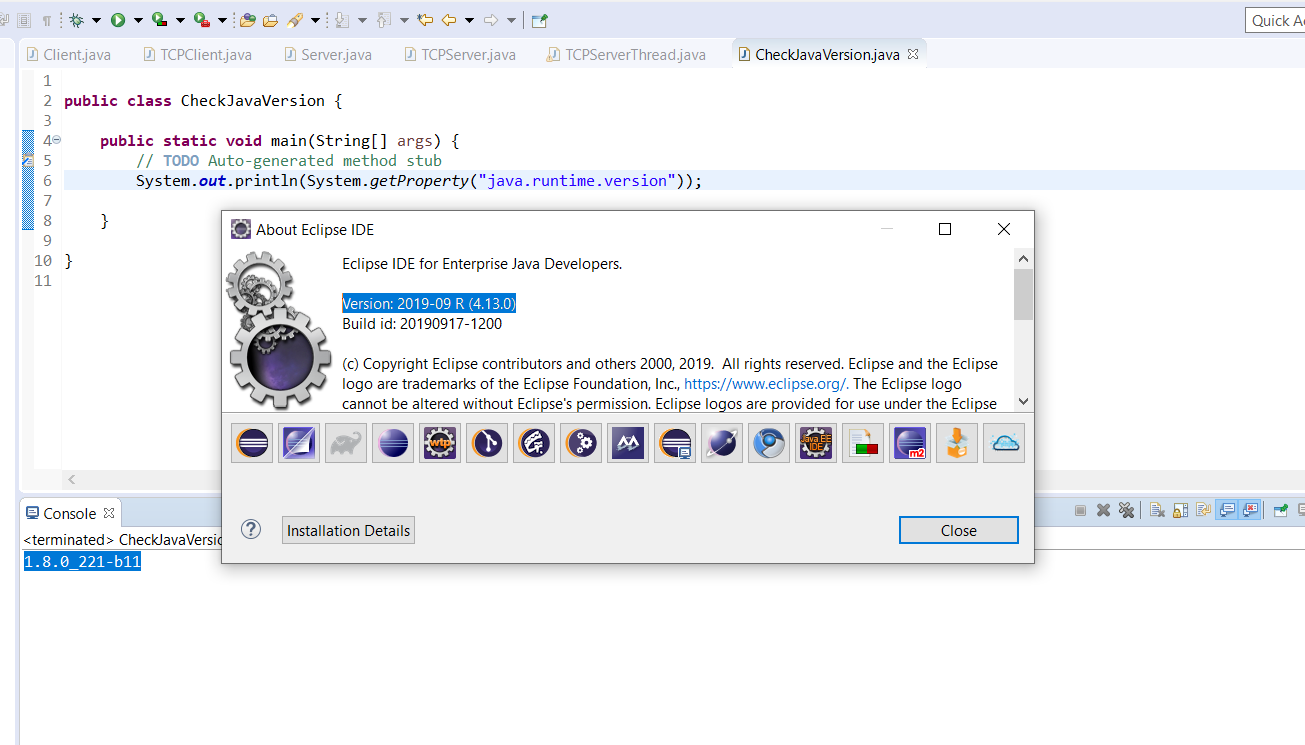
**Kenneth Goh, Student Number: 1970596**

1. Programming language and version:

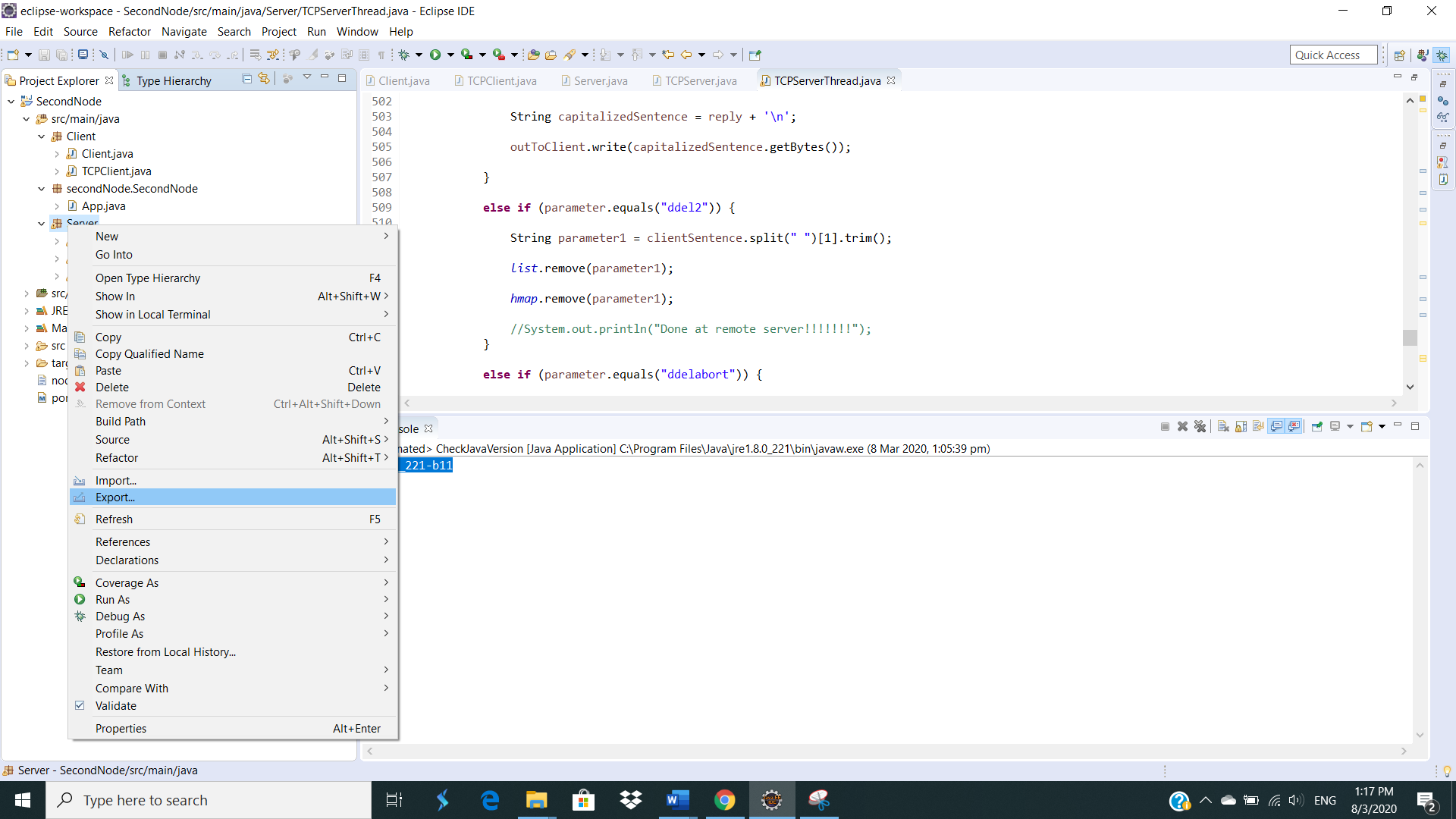
Java 1.8.0\_221-b11

1. Integrated development environment (IDE) or editor used to develop the project

Eclipse IDE for Enterprise Java Developers. Version: 2019-09 R (4.13.0)



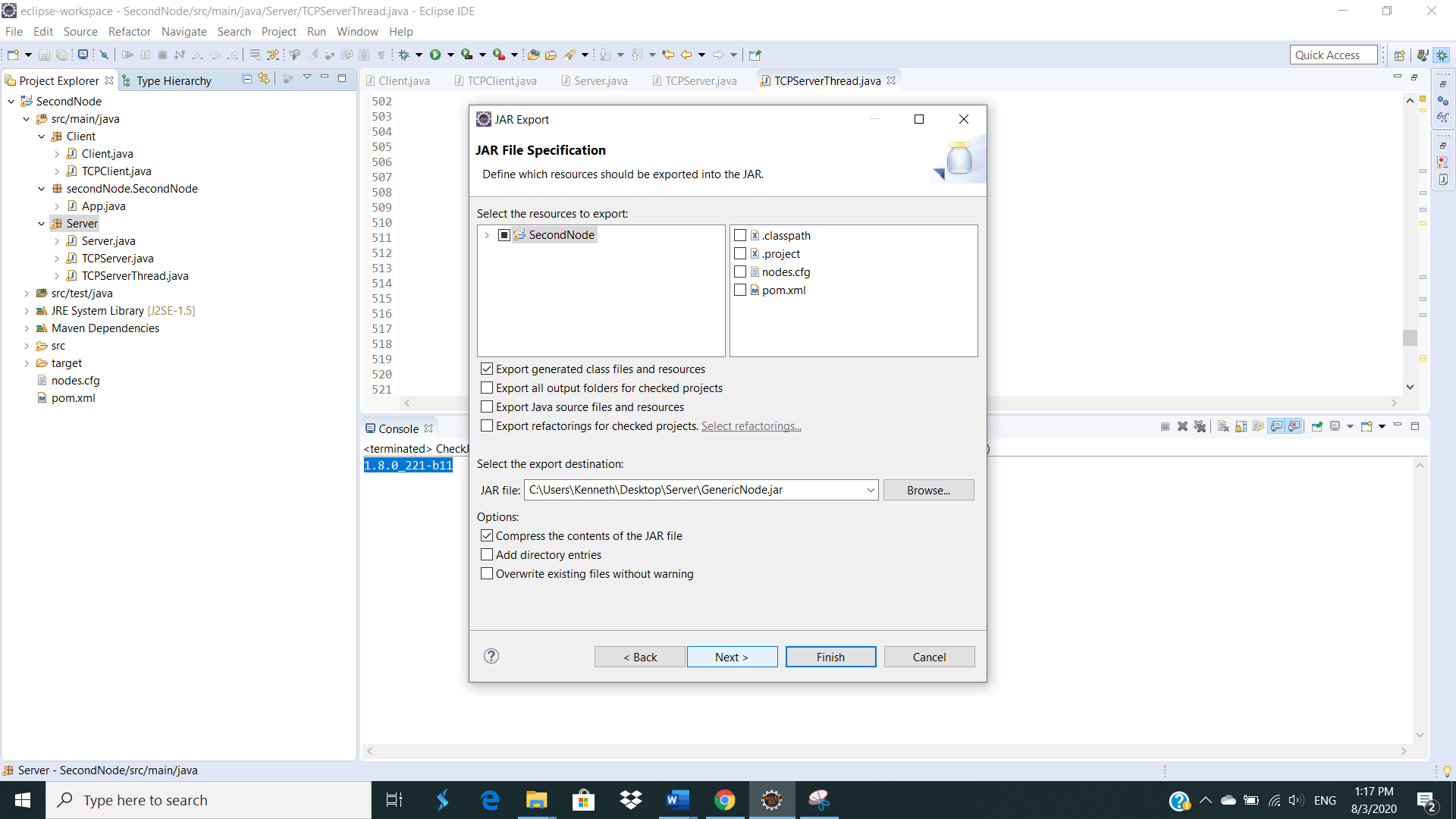
1. Exporting the jar file, GenericNode.jar of the server



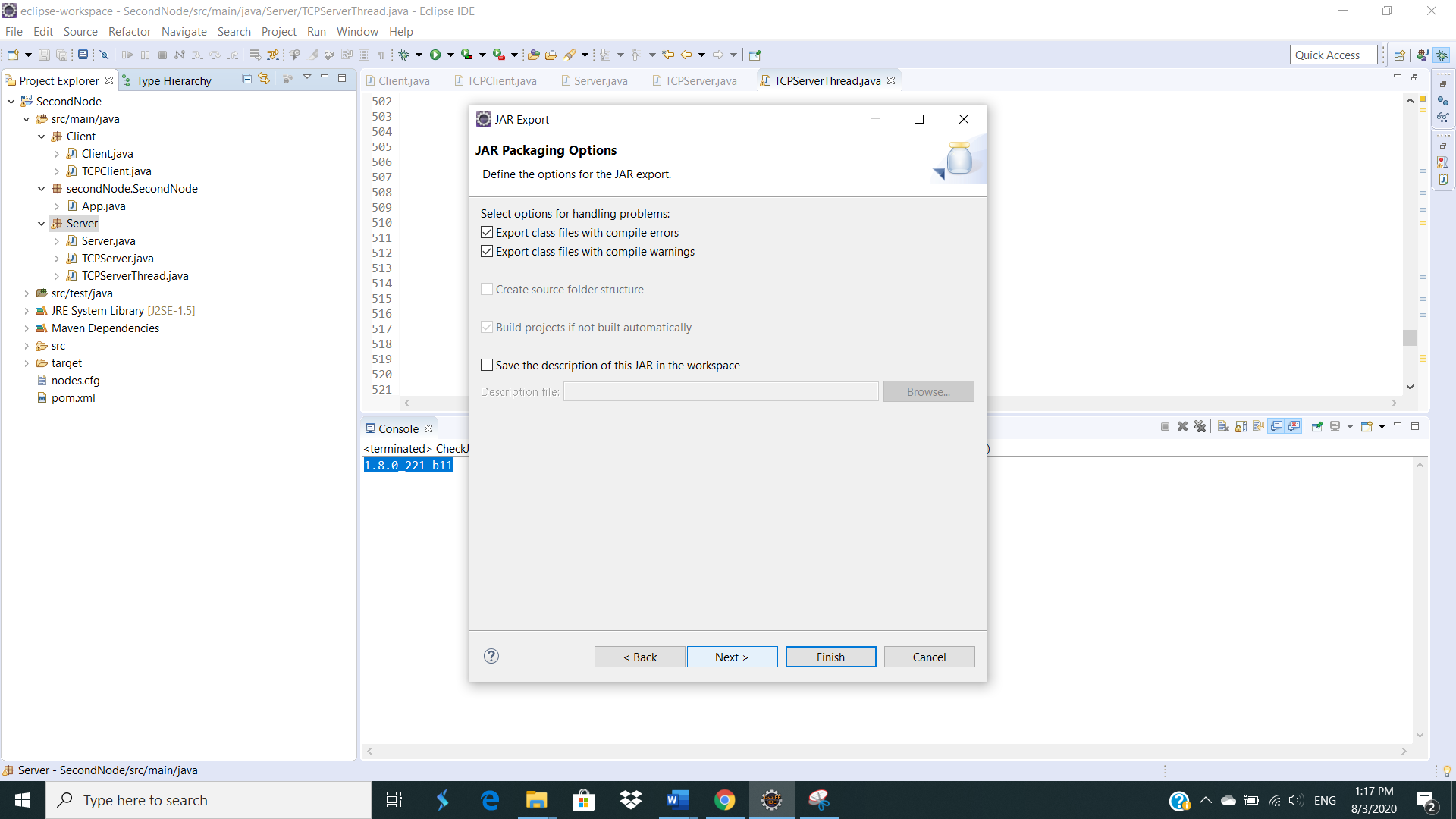
Right-click on the server package and click “Export”.



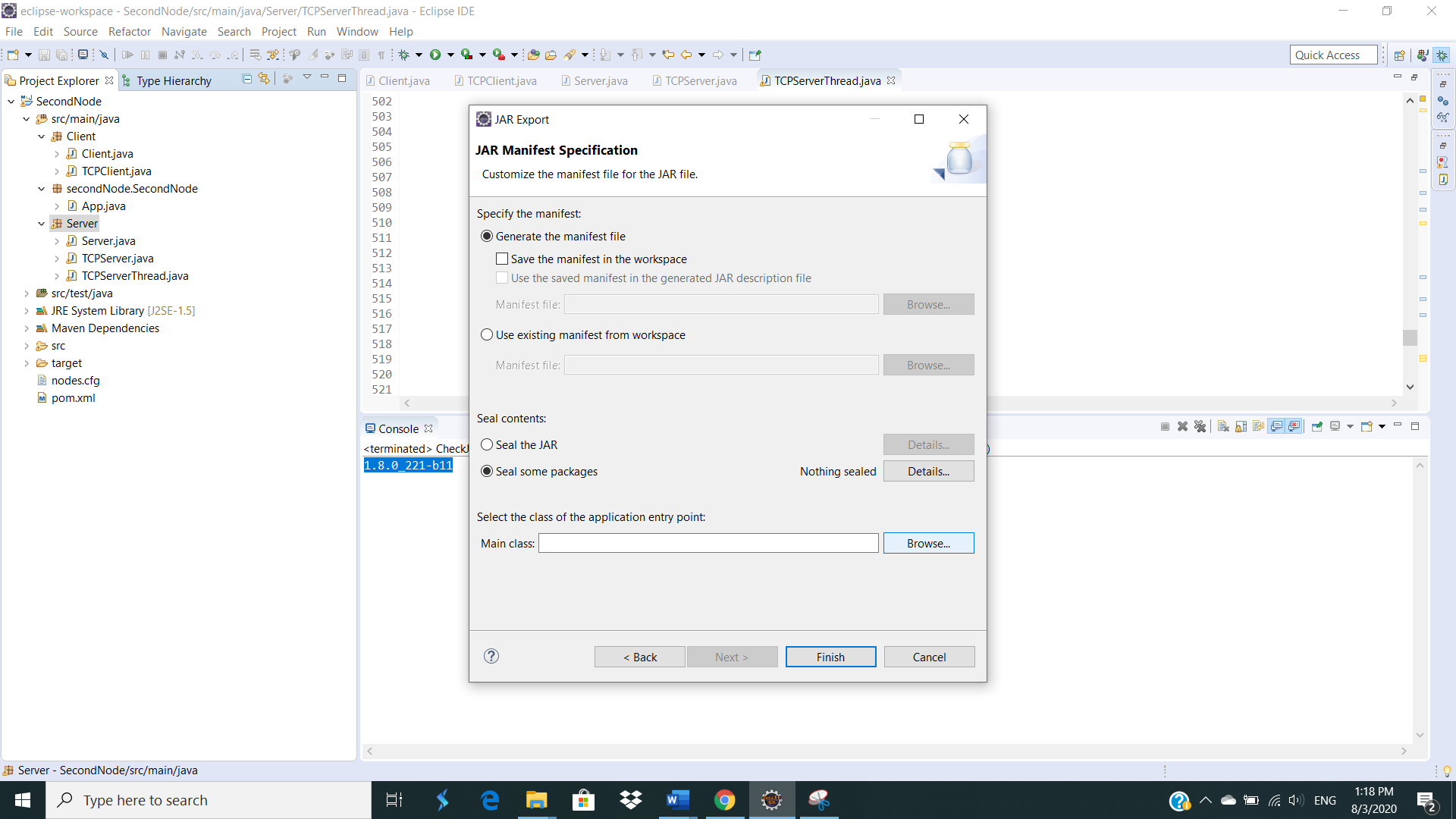
Select “JAR File”.



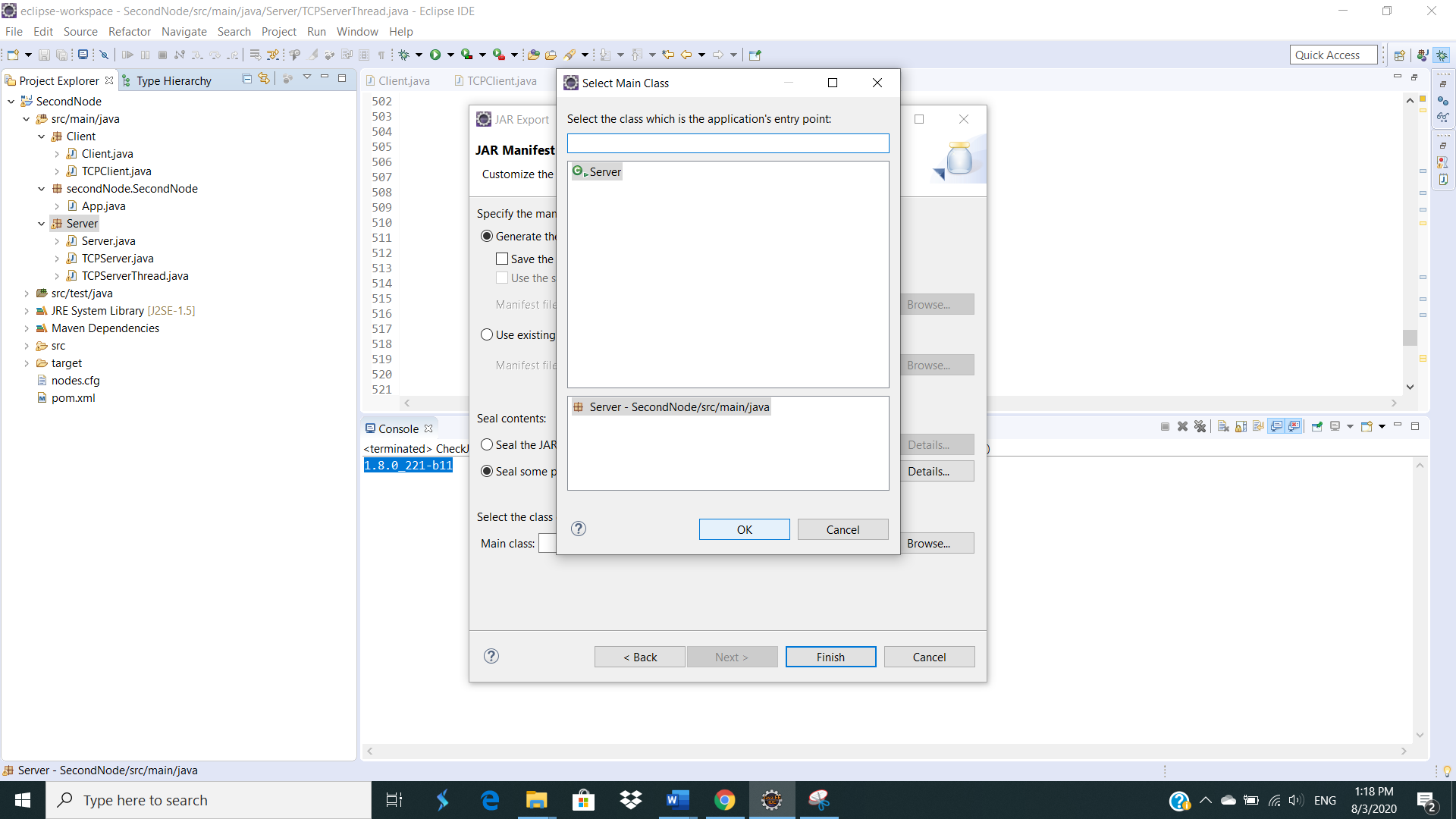
Specify the path to export the File jar and click “Next”.



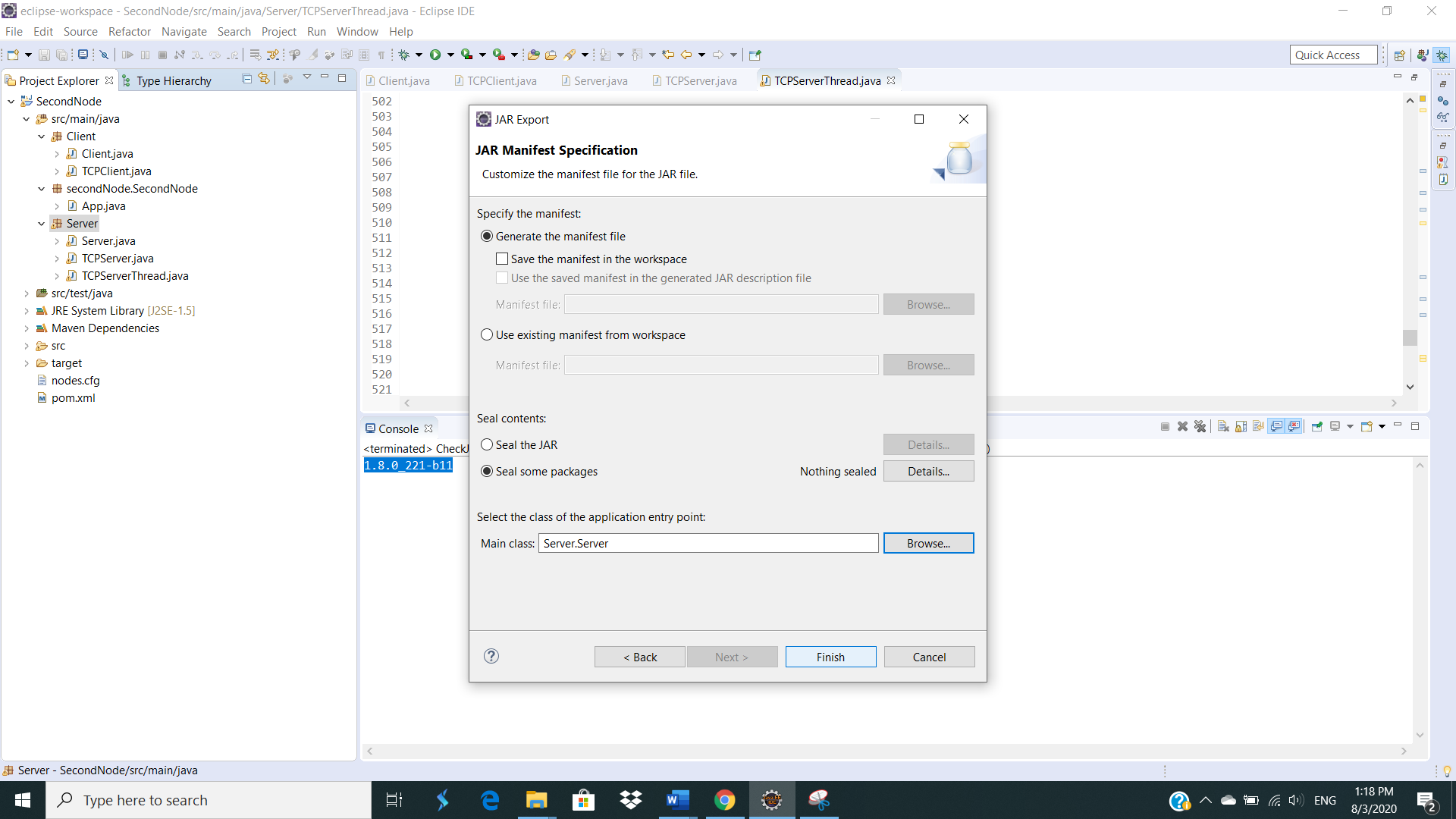
Click “Next”.



Click “Browse” to select the “Server” class.



Select “Server” class and click “OK”.

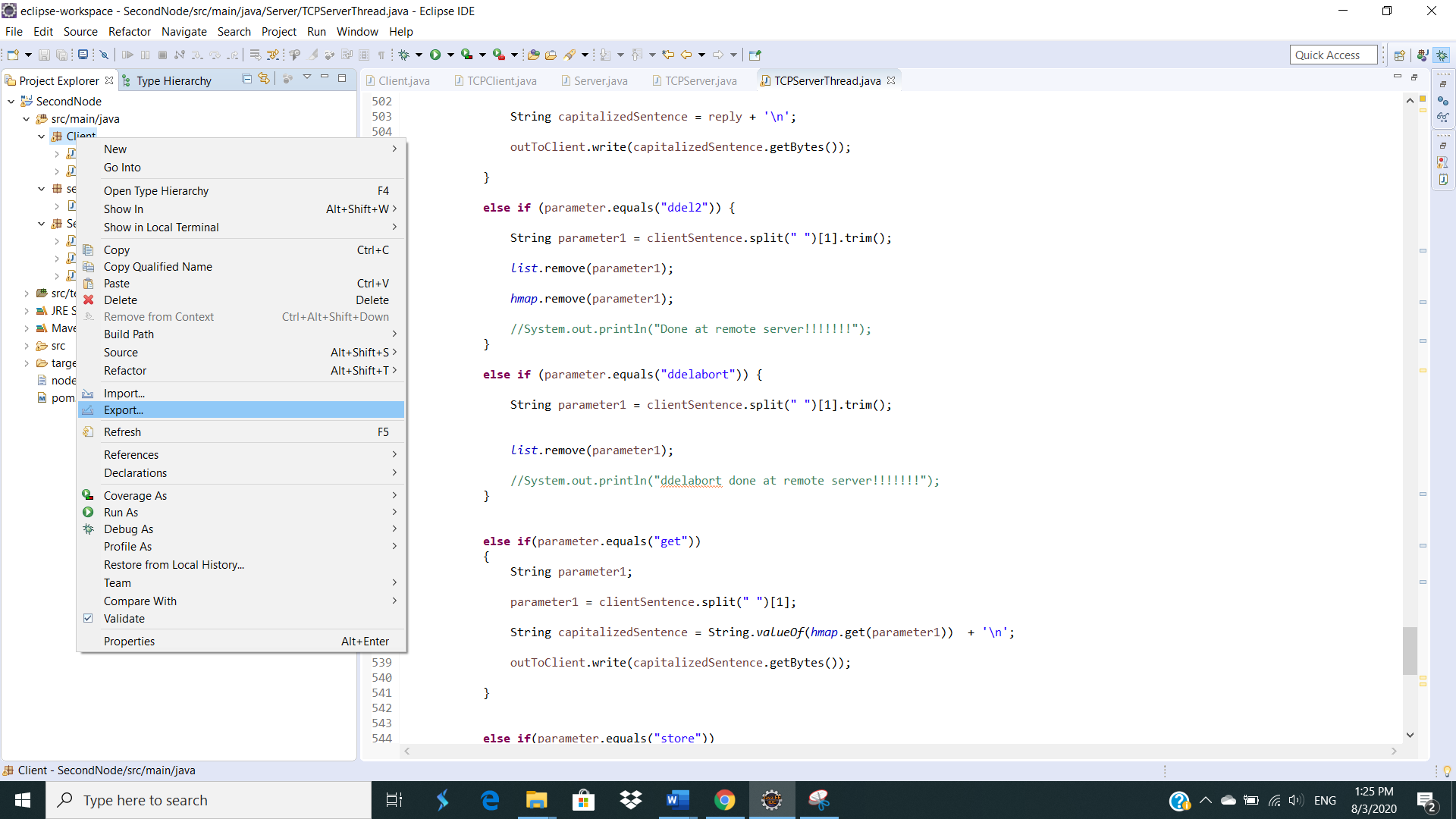


Click “Finish” and the named jar file, “GenericNode.jar” will be exported to desired location.

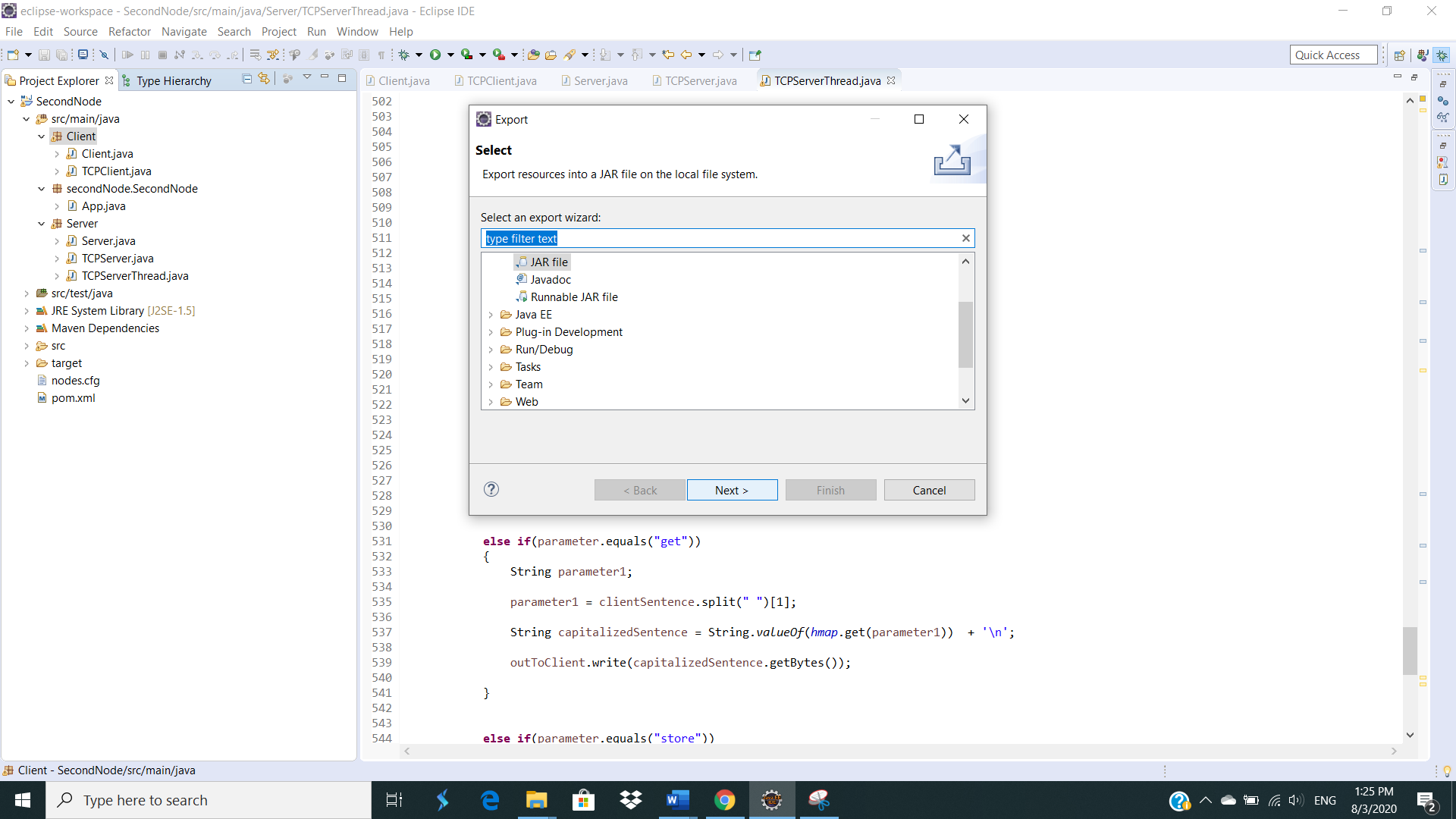
1. Exporting the jar file, Generic.jar of the client



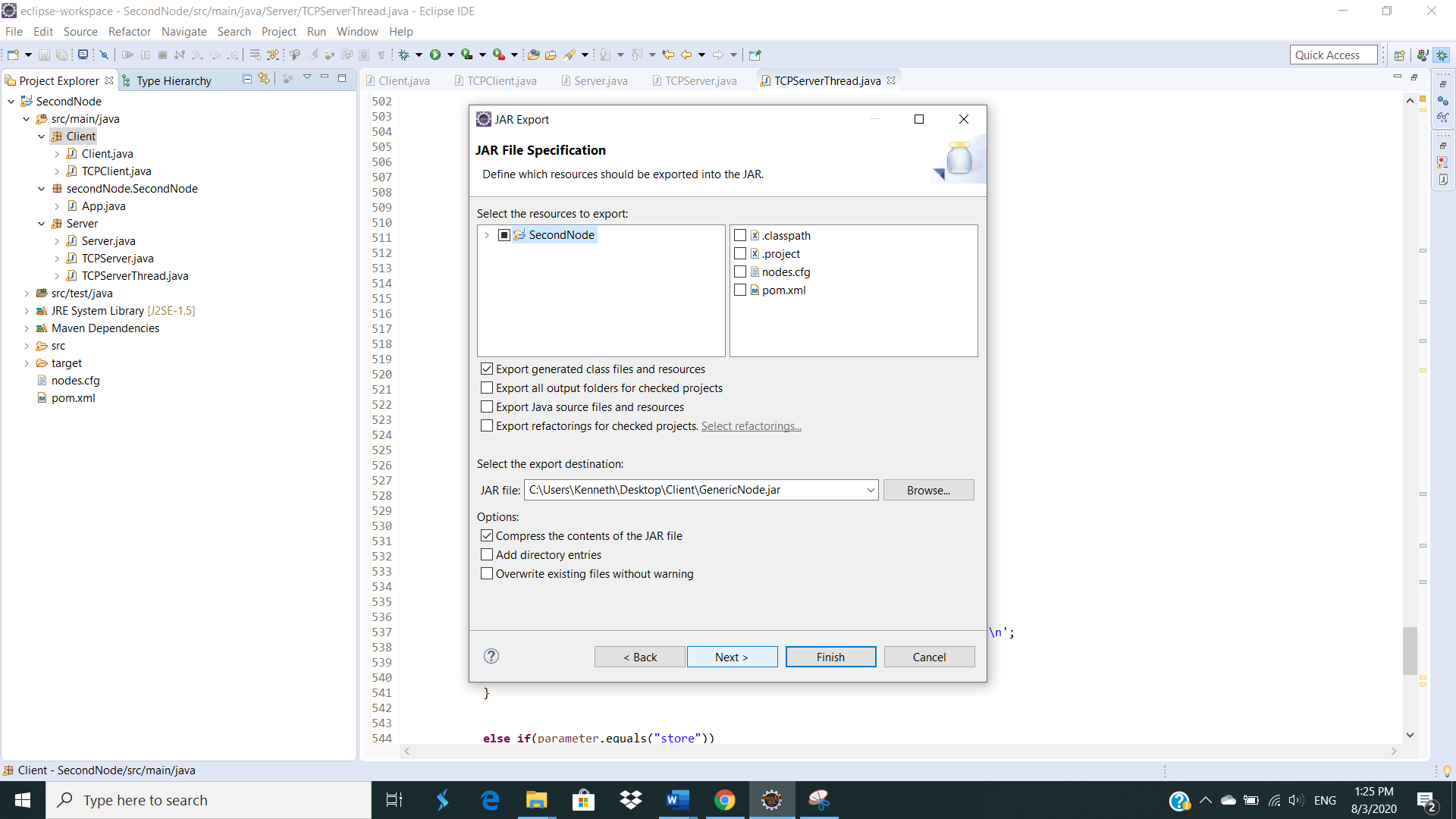
Click on the “Client” class and right click on it.



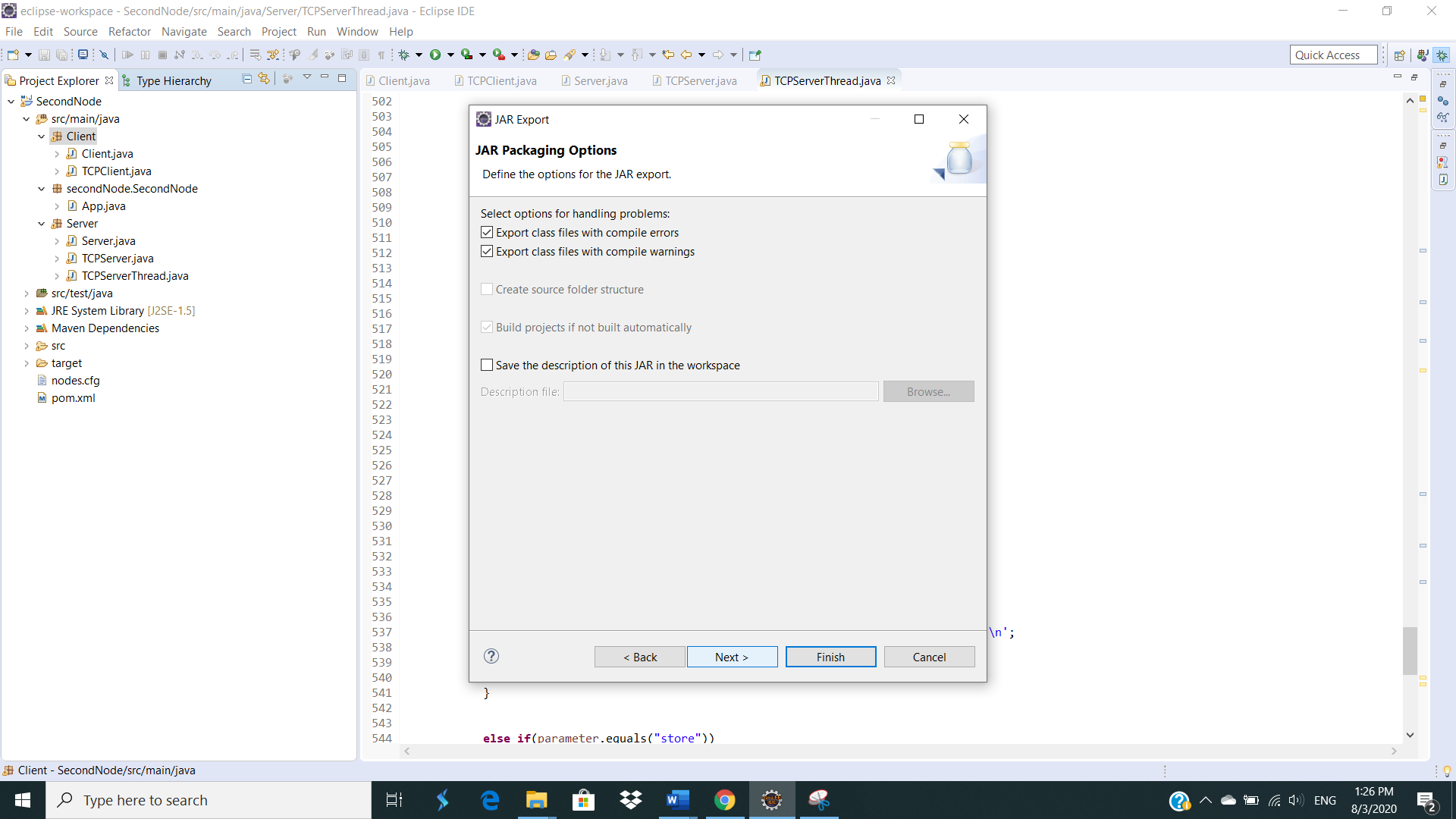
Click on “Export”.



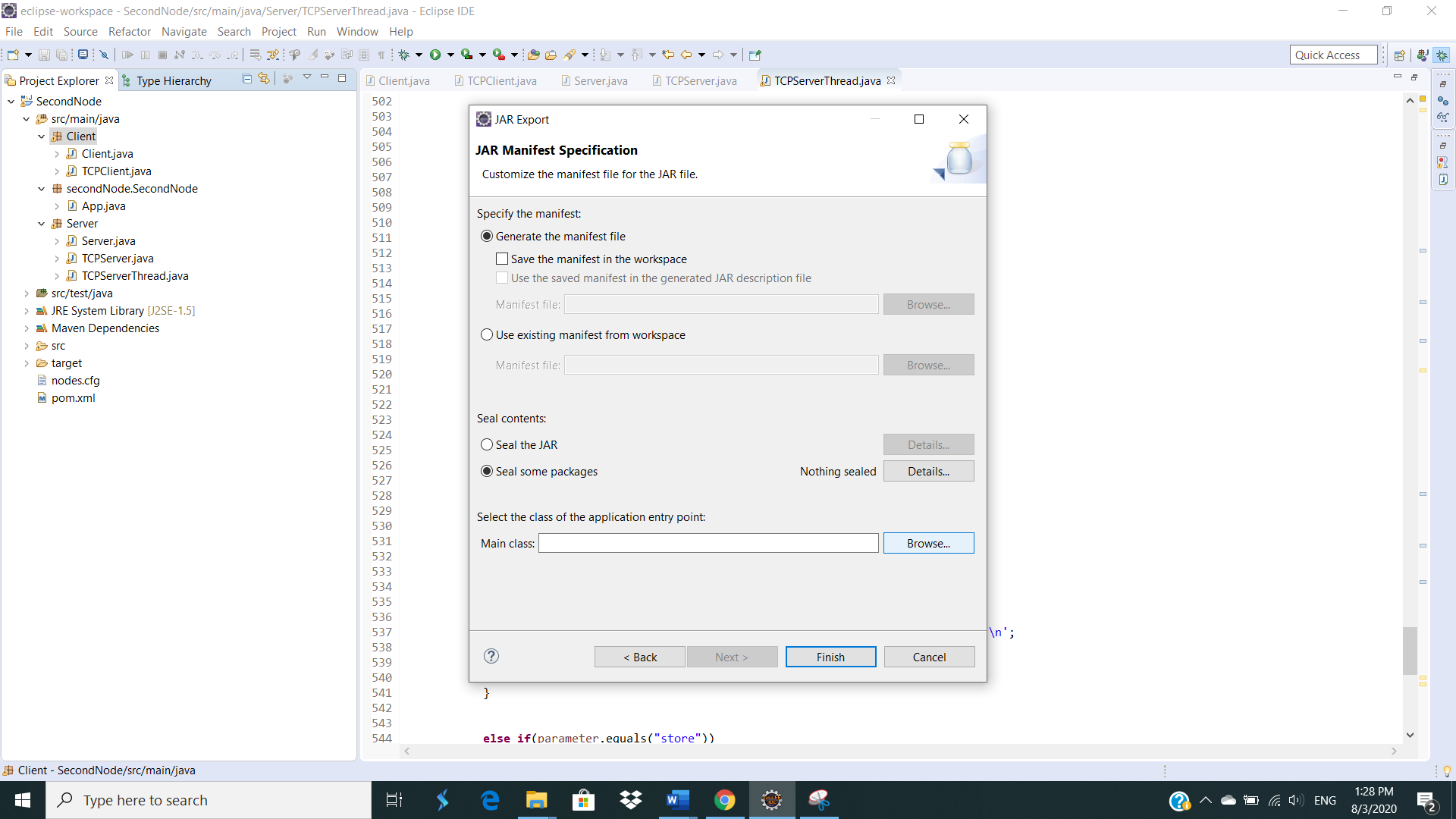
Choose “JAR file” as the type of file to be exported and then click “Next”.



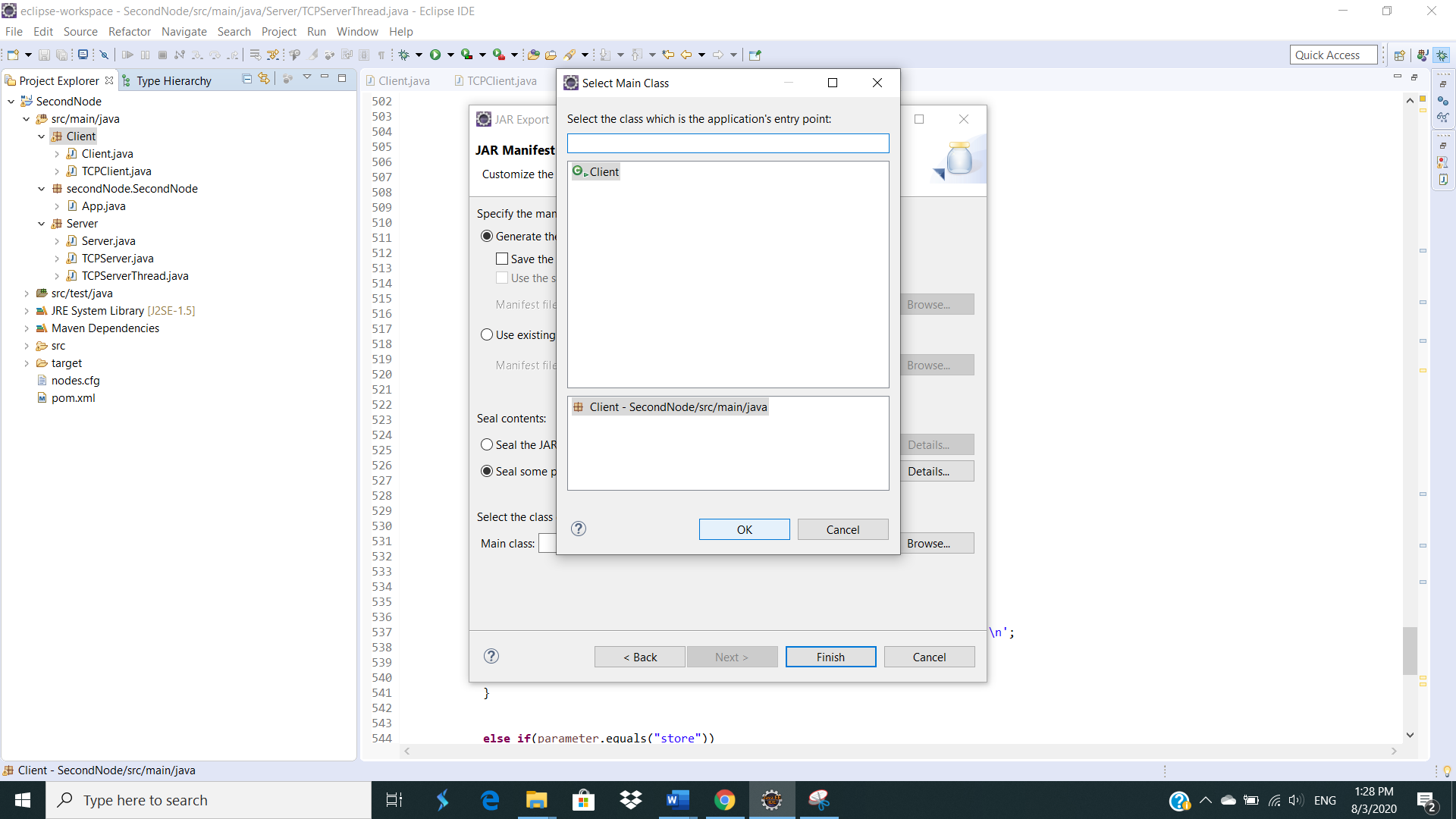
Specify the location to export the jar file to and click “Next”.



Click “Next”.



Click on “Browse” to select the class.



Select the client class “Client” and click “OK”.

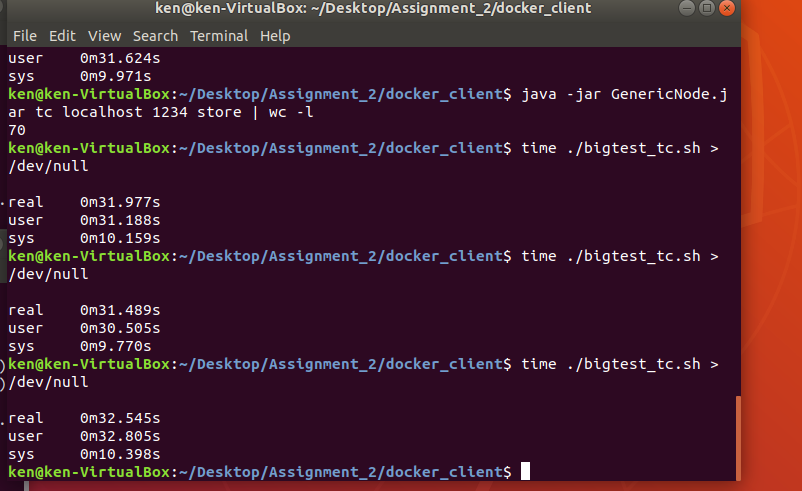


Click “Finish” and the server jar file, “GenericNode.jar” will be exported to the desired location.

1. Testing in a local Linux environment

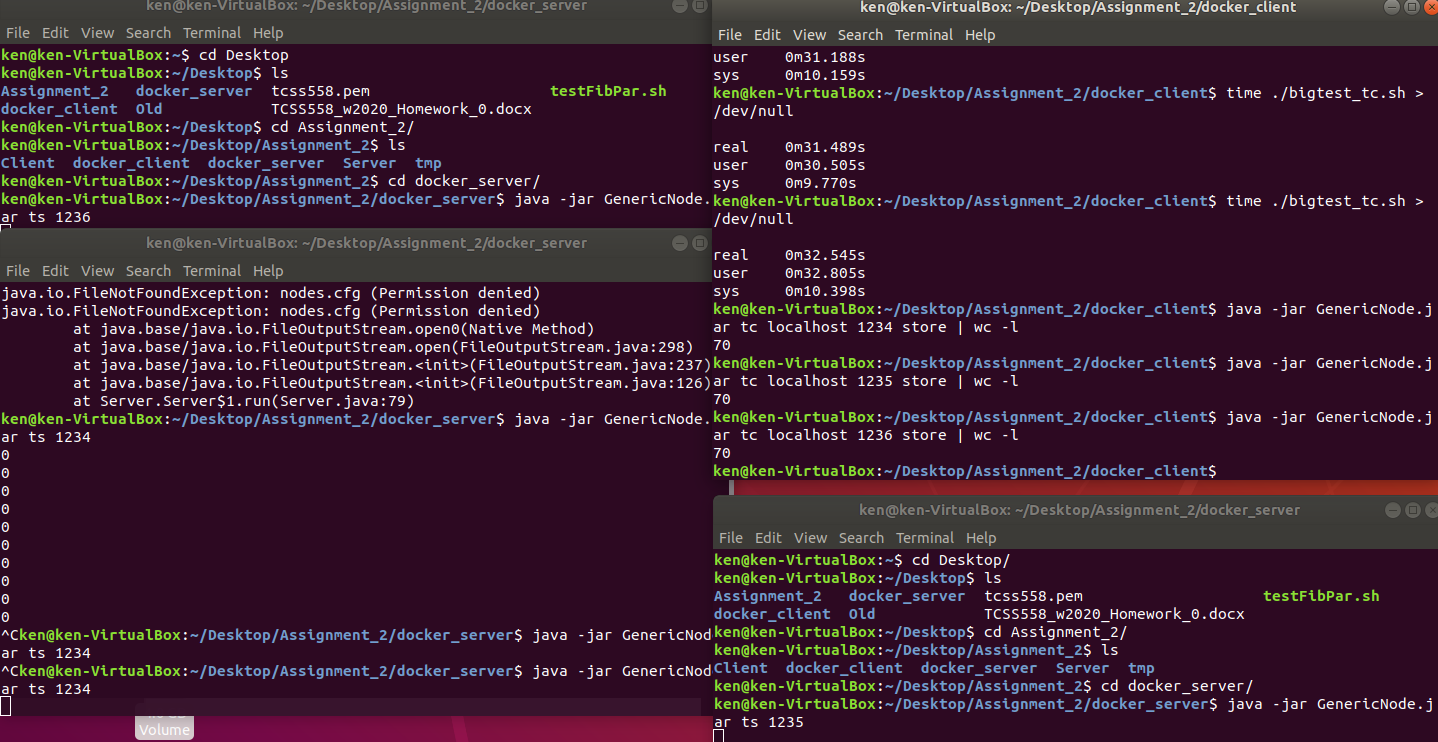
Run the “bigtest\_tc.sh” script after typing in “chmod 777 bigtest\_tc.sh”.

The command to run is “time ./bigtest\_tc.sh > /dev/null”. Below shows an example of running 3 instances or threads of the TCP server. The same applied to any number of instances or threads of the TCP server as well.



Printout of count = 70 for the 1 localhost TCP with ports 1234, 1235 and 1236 by typing in the command ”java -jar GenericNode.jar tc localhost 1234 store | wc -l”, ”java -jar GenericNode.jar tc localhost 1235 store | wc -l” and ”java -jar GenericNode.jar tc localhost 1236 store | wc -l”.

Testers can also input key or get key individual by “java -jar GenericNode.jar tc localhost 1234 put a 123” or “java -jar GenericNode.jar tc localhost 1234 get a”. Changed port according to test for port 1235 and 1235.

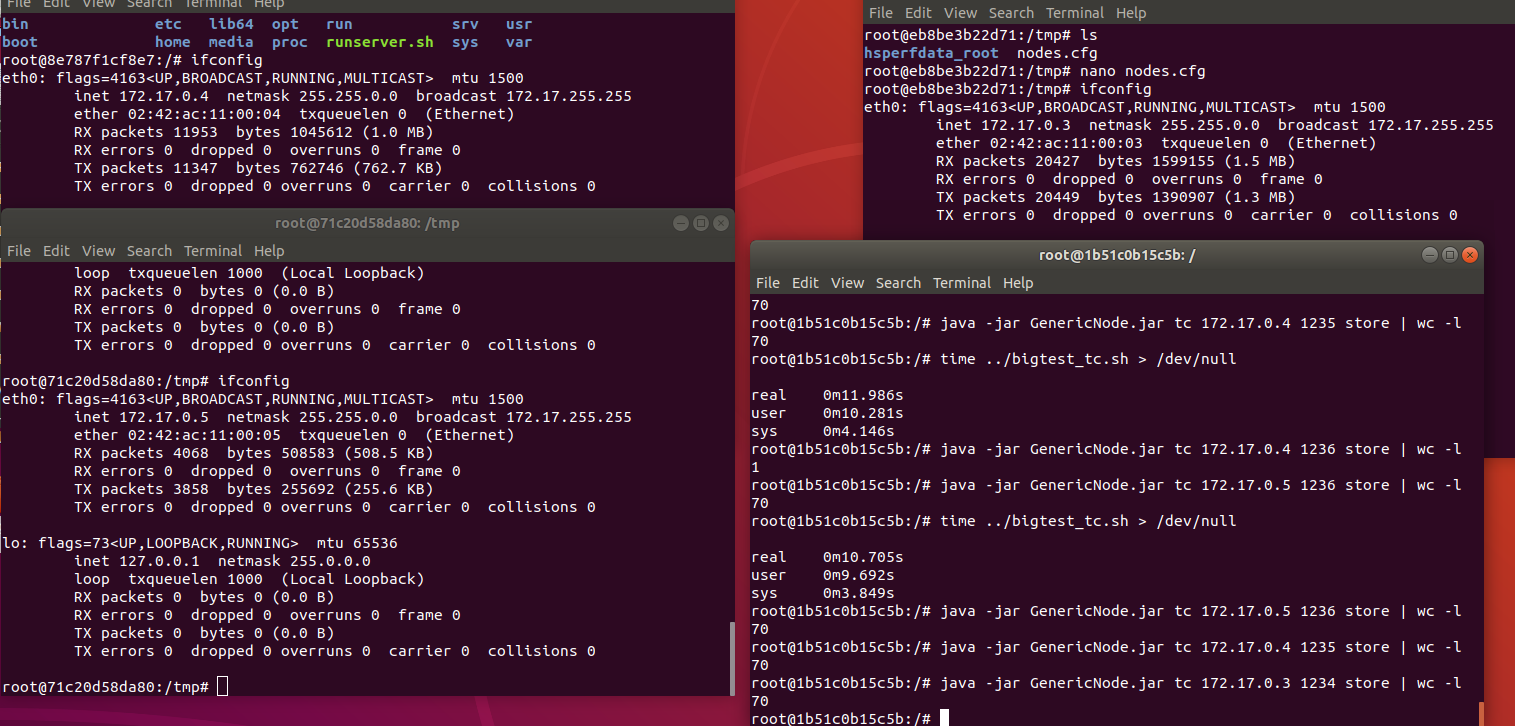


For local testing, it is sufficient to have just a local copy of the “nodes.cfg” stored in the local “/tmp” folder. The full address path is “/tmp/nodes.cfg”.

1. Testing in a local Linux environment on docker containers

Run the “bigtest\_tc.sh” script after typing in “chmod 777 bigtest\_tc.sh”.

The command to run is “time ../bigtest\_tc.sh > /dev/null”. Below shows an example of running 3 docker containers of the same TCP server. The same applied to any number of docker containers of the TCP server as well.

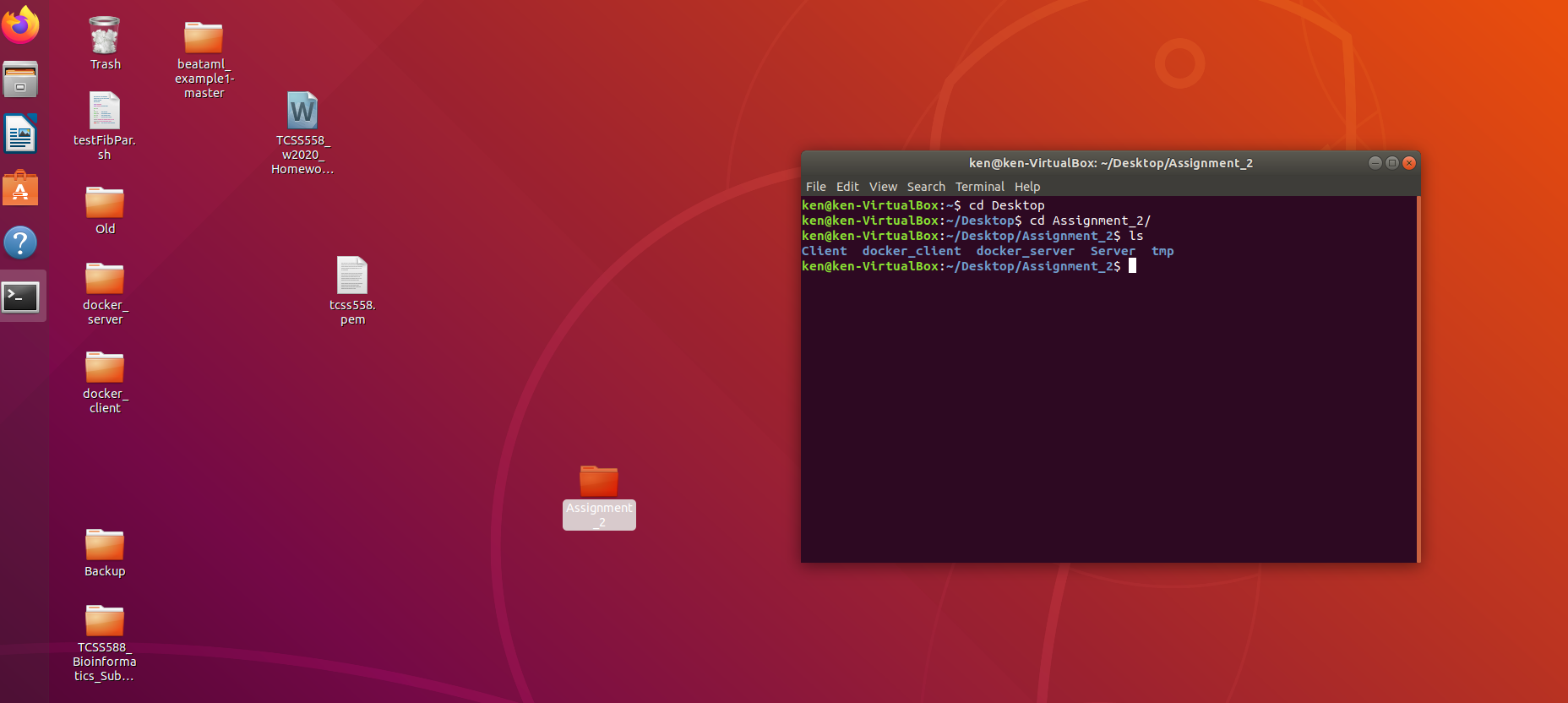


Printout of count = 70 for the 3 Container TCP with IP: 172.17.0.3 with port 1234, IP: 172.17.0.4 with port 1235, and IP: 172.17.0.5 with port 1236 by typing in the command ”java -jar GenericNode.jar tc 172.17.0.3 1234 store | wc -l”, ”java -jar GenericNode.jar tc 172.17.0.4 1235 store | wc -l” and ”java -jar GenericNode.jar tc 172.17.0.5 1236 store | wc -l”.

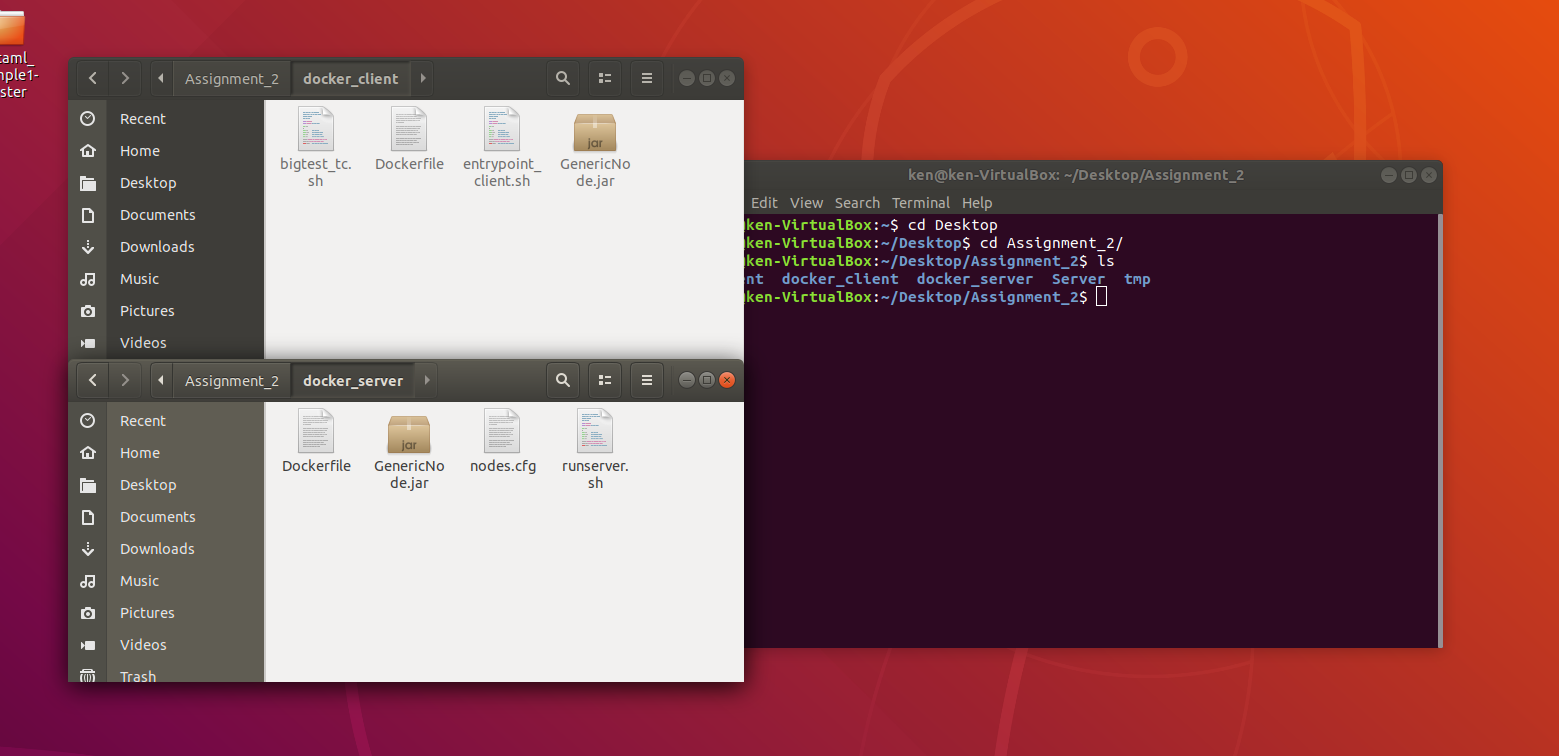
Testers can also input key or get key individual by “java -jar GenericNode.jar tc 172.17.0.3 1234 put a 123” or “java -jar GenericNode.jar tc 172.17.0.3 1234 get a”. Changed port according to test for port 1235 and 1235, and IP addresses “172.17.0.4” & “172.17.0.5” accordingly.

For local testing on docker containers, it is required to have a local copy of the “nodes.cfg” stored in the local “/tmp” folder of each docker container. The full address path is “/tmp/nodes.cfg”.

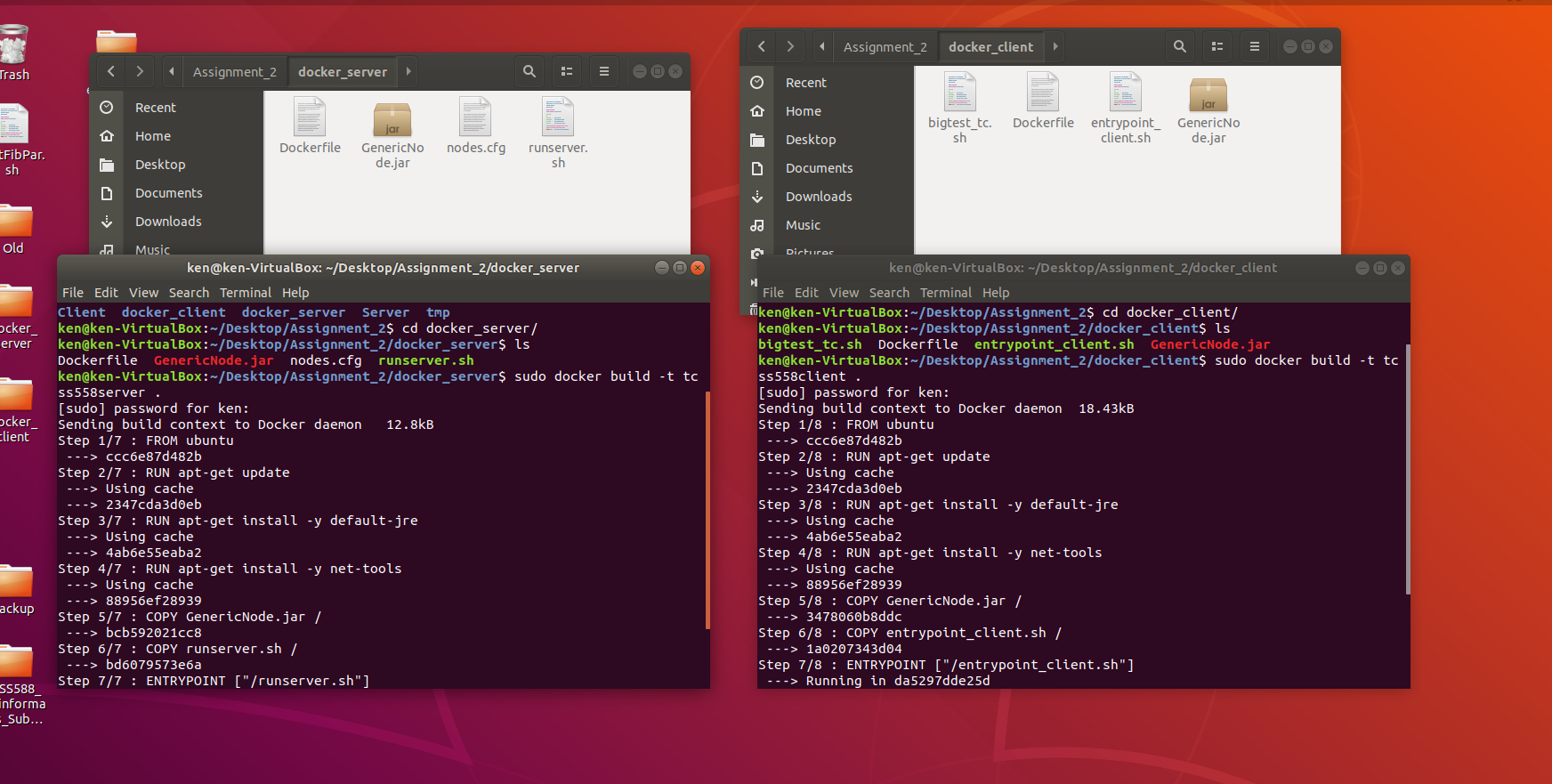
1. Setting up of the local docker containers (Refer to this section for more details on how to go about setting up each individual docker container.)



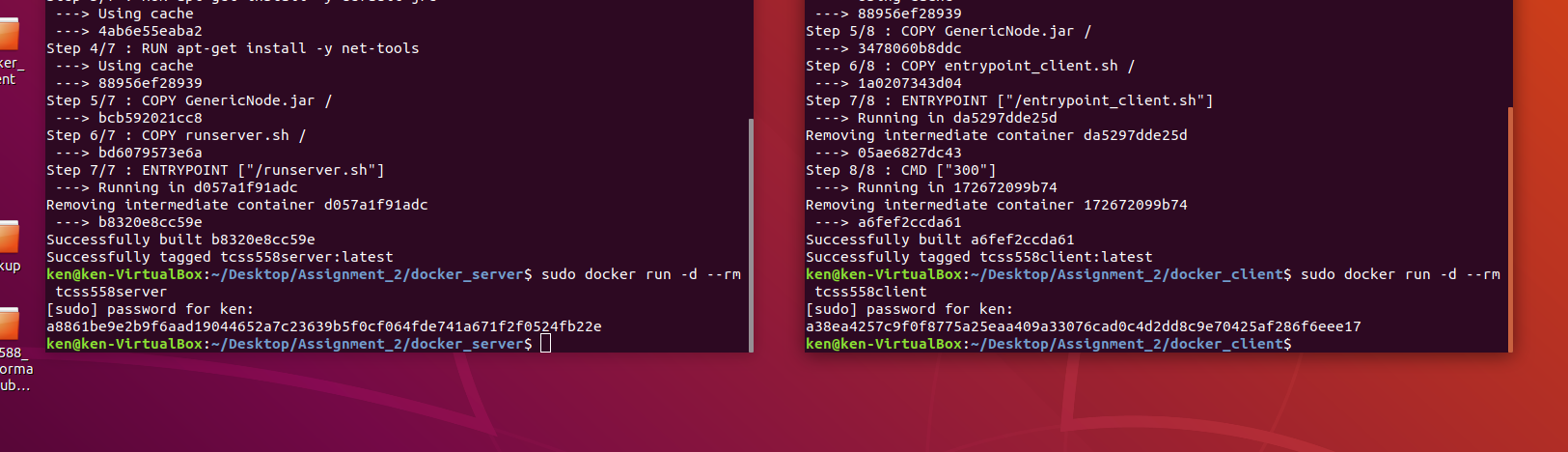
Use a command line and go into the working directory containing both docker client and docker server. In this case, it is located at “/Desktop/Assignment\_2”.



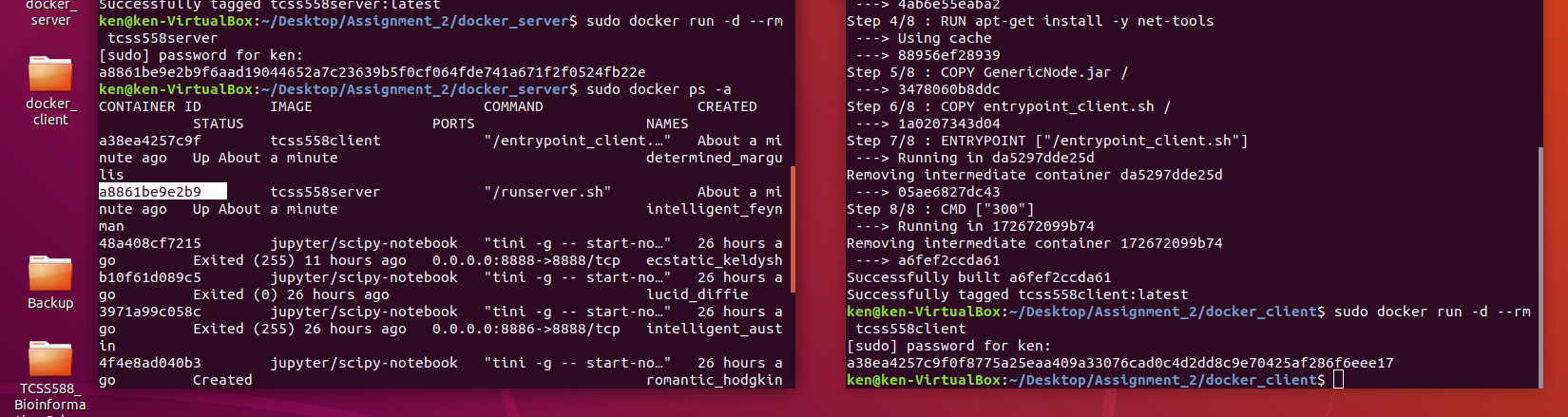
Make sure the jar file, “GenericNode.jar” for the server and client are in the respective directory before building the docker images for the server and client.



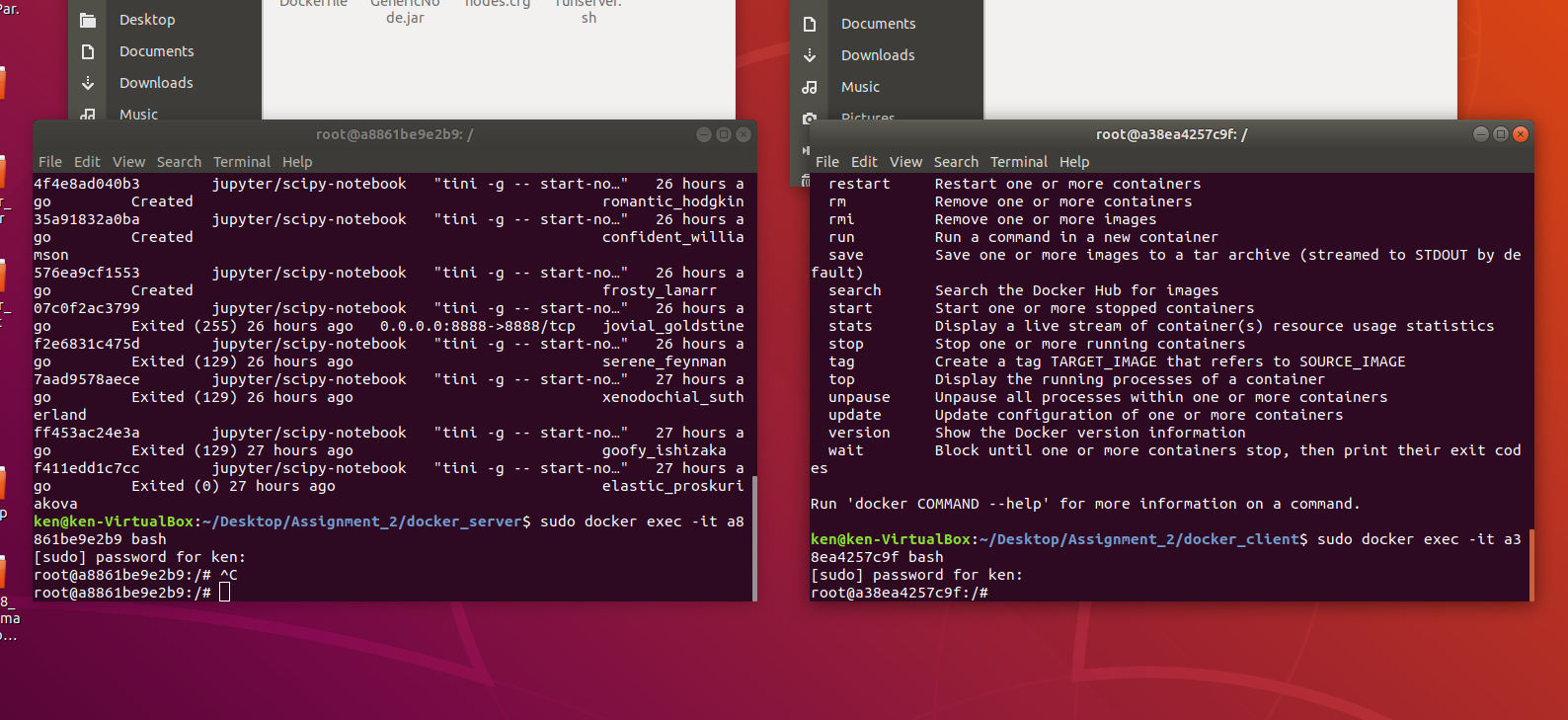
Build the docker images for the server and client respectively using the commands, “sudo docker build -t tcss558server .” and “sudo docker build -t tcss558client .”.



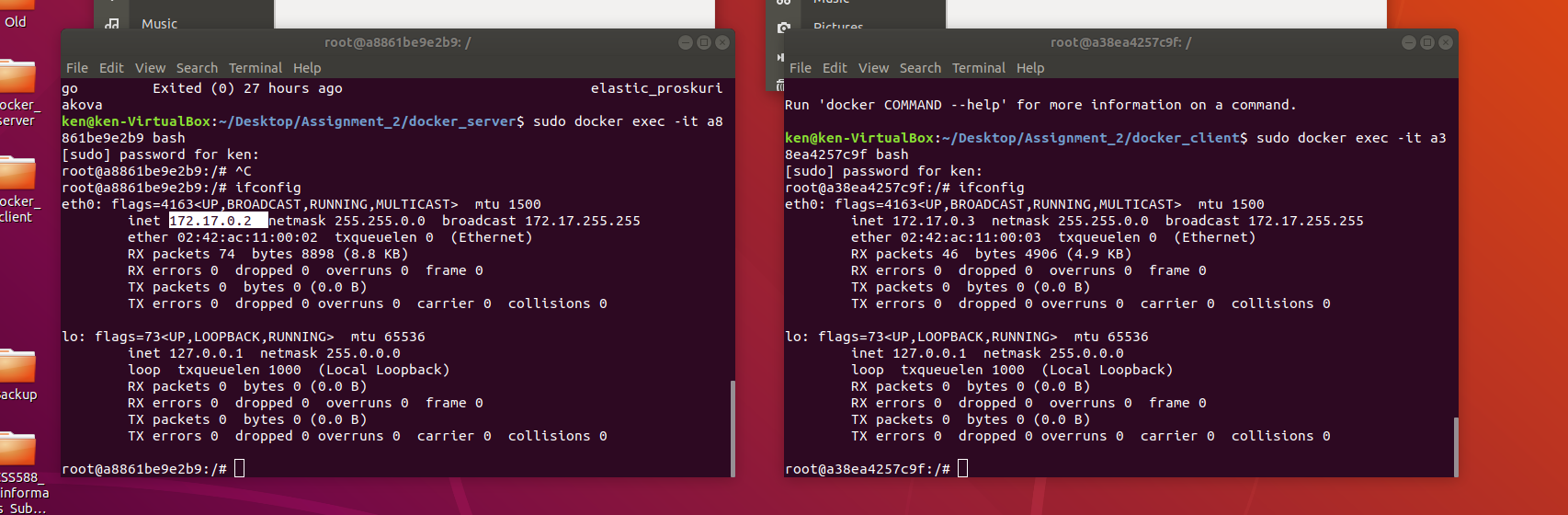
Run the two or more docker containers for the client and server by typing the commands, “sudo docker run -d --rm tcss558server” and “sudo docker run -d –rm tcss558client”.



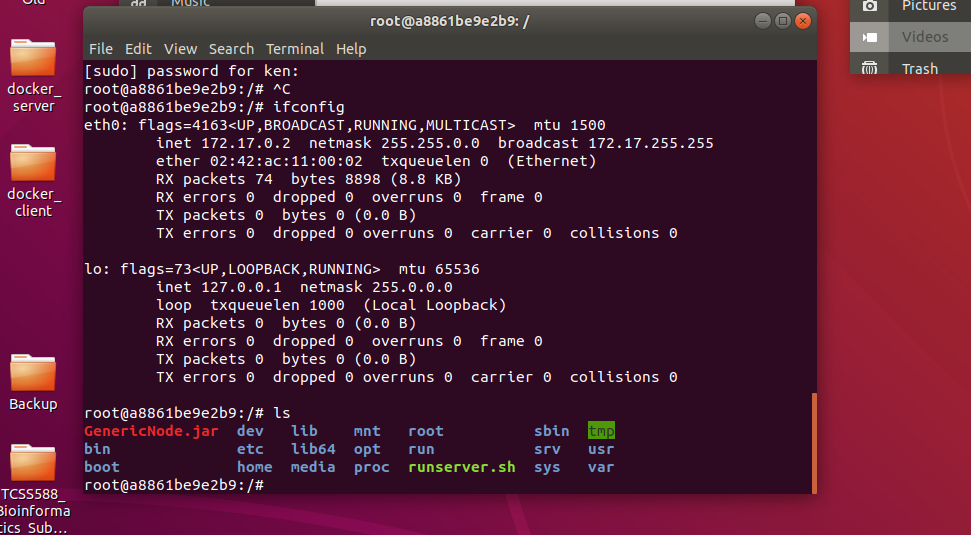
Get the container IDs of the docker client and server by typing “sudo docker ps -a”.

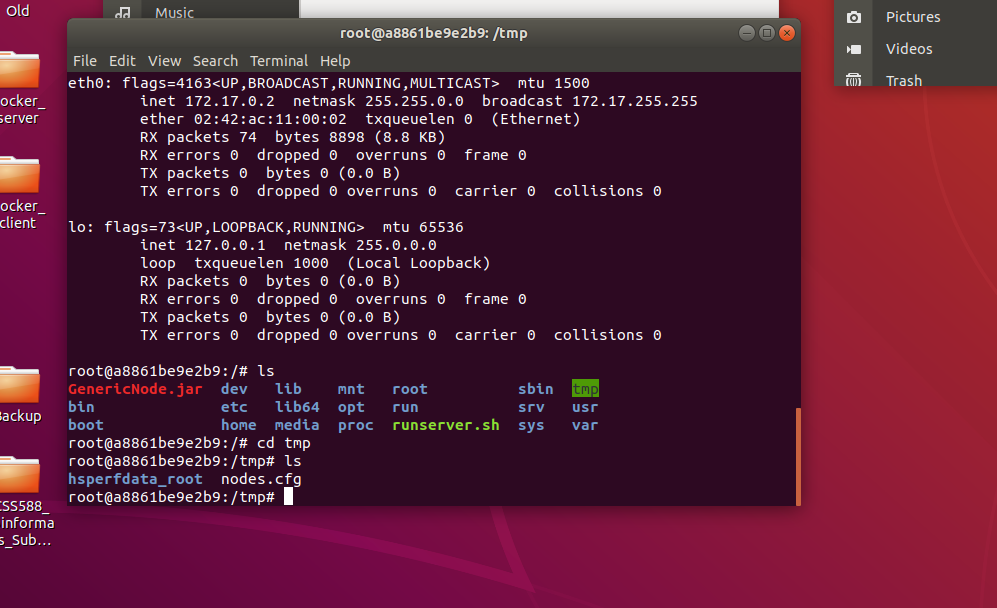


Then, execute bash iteratively on both containers by typing “sudo docker exec -it <container ID> bash”.

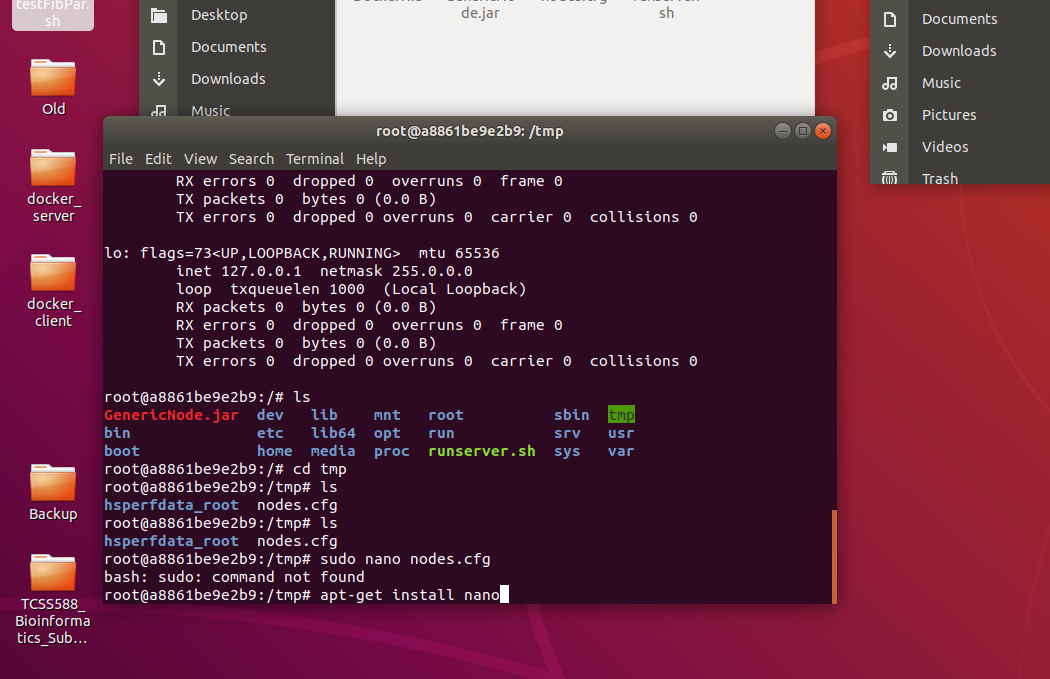


After which, type “ifconfig” to find out the IP addresses of each IP container.

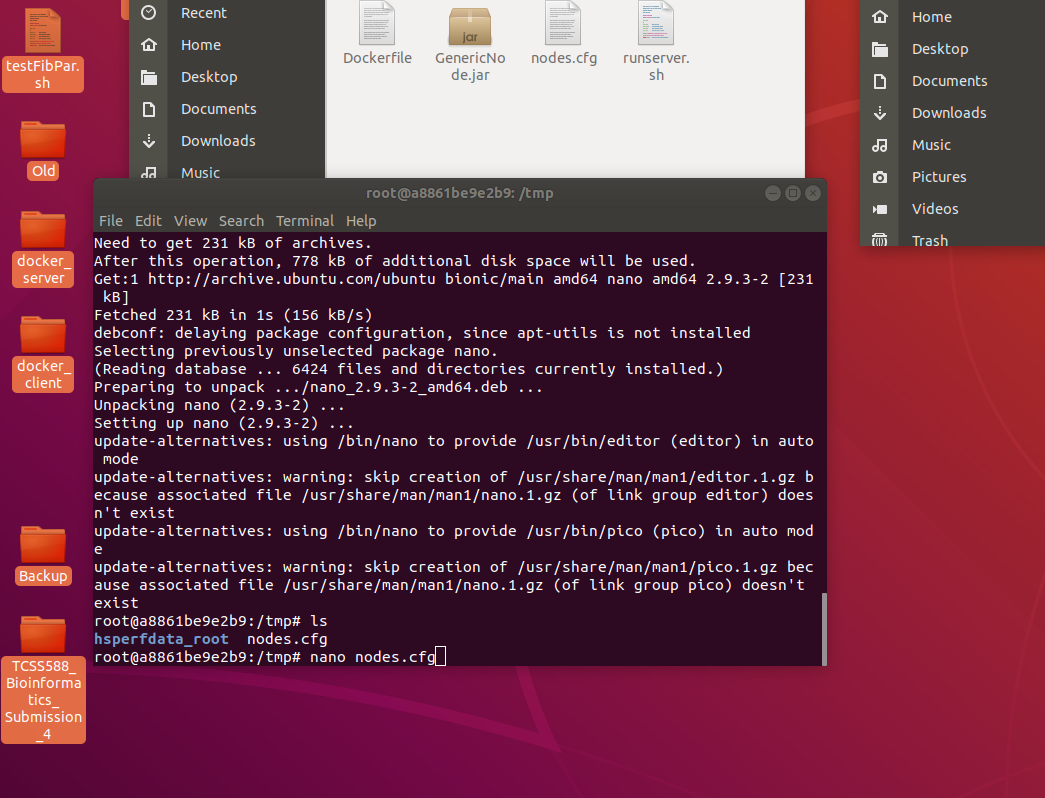


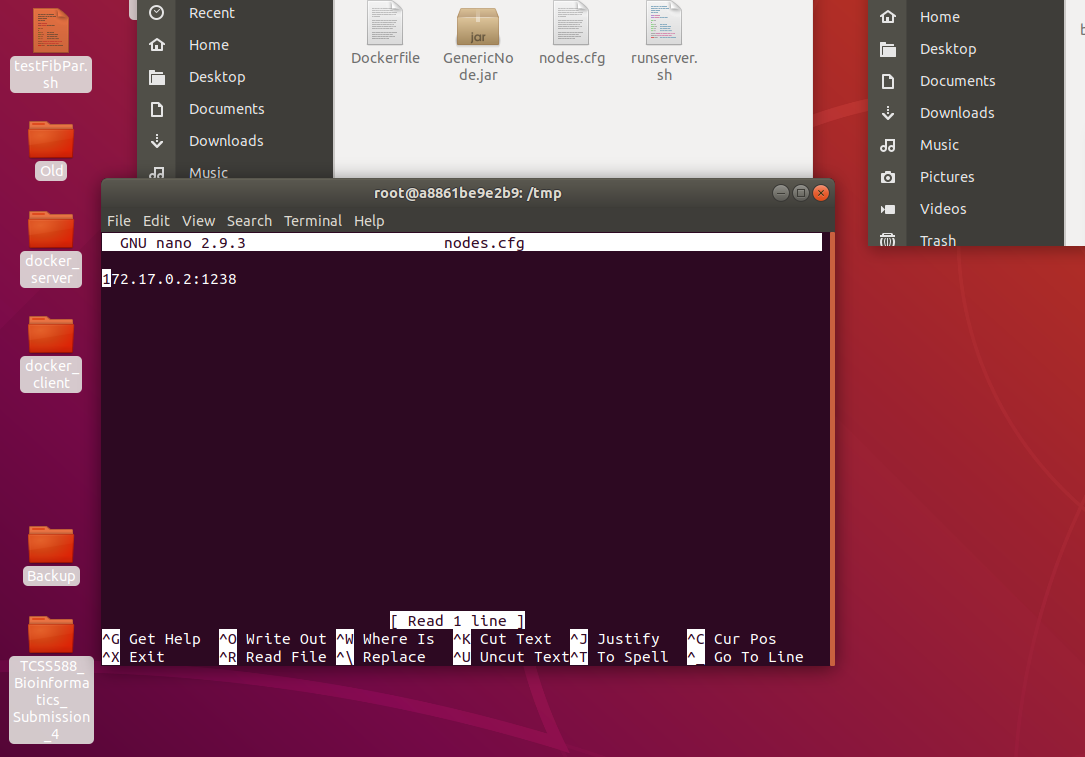


Go into the directory of each docker server to make sure that the “nodes.cfg” file is present in “/tmp/nodes.cfg”.



Then type in “apt-get install nano” in order to update the docker container and be able to modify the “nodes.cfg” file.





Type “nano nodes.cfg” to go into the “nodes.cfg” file of each docker server container. Type in the IP addresses and ports of each individual docker server container. (In this case of testing docker containers locally, the ports of each individual docker server container will be different.) Make sure that each docker server container contains the IP addresses and respective ports of the other docker server container/s.

(The server container is programmed to be event-driven in the sense that every time it receives a put or del command, it will reread the “nodes.cfg” file to get the membership list of docker server containers. It is important that users update the “nodes.cfg” file everytime the membership list changes.)