STEPS TO GET PROGRAM WORKING:

Step 1) Run bash deploy.sh <OPTIONAL PORT> in the command line this will deploy the servers. The default port is 1099

Step 2) Run either bash run_client.sh <name> <OPTIONAL HOSTNAME> <OPTIONAL PORT>. Then default hostname and port are “my-hashserver” and 1099 respectively

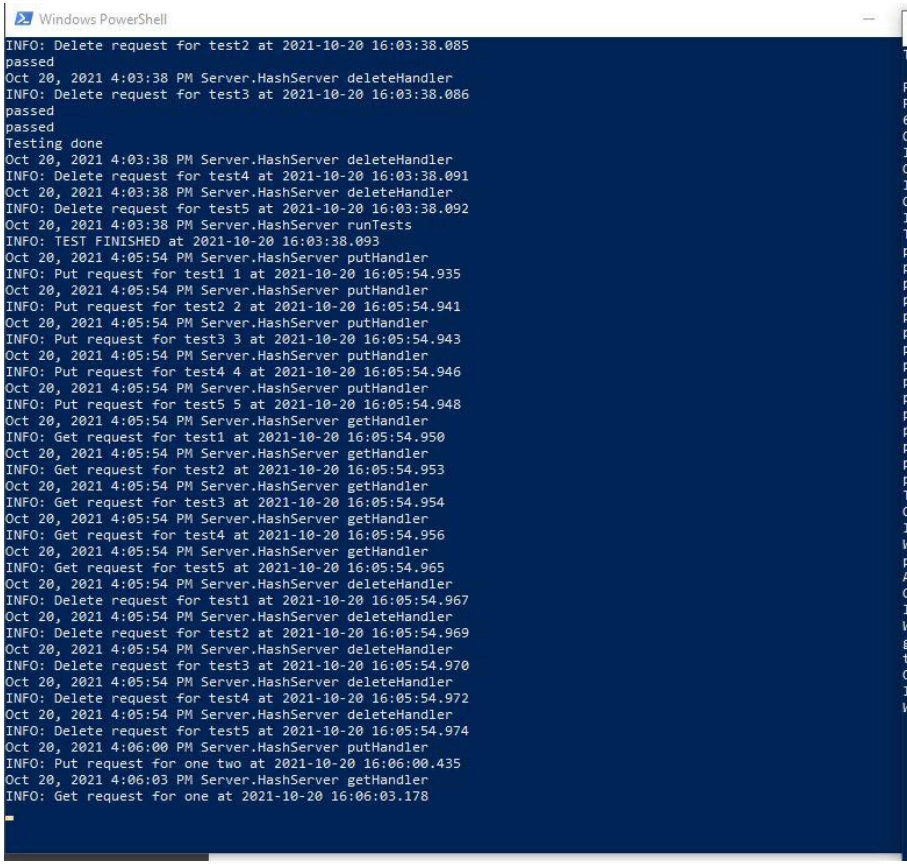
Step 3) The client will then prompt the user for a request which can be any of the following as formatted below:
PUT: “put <key> <value>”
GET: “get <key>”
DELETE: “delete <key>”
The server will respond to the client if the request had been completed or not, and then be requested to input another request. It’ll continuously ask for requests until the user quits the client and/or server.

Step 4) Checking the logs of a container run: docker container logs <container name> -f

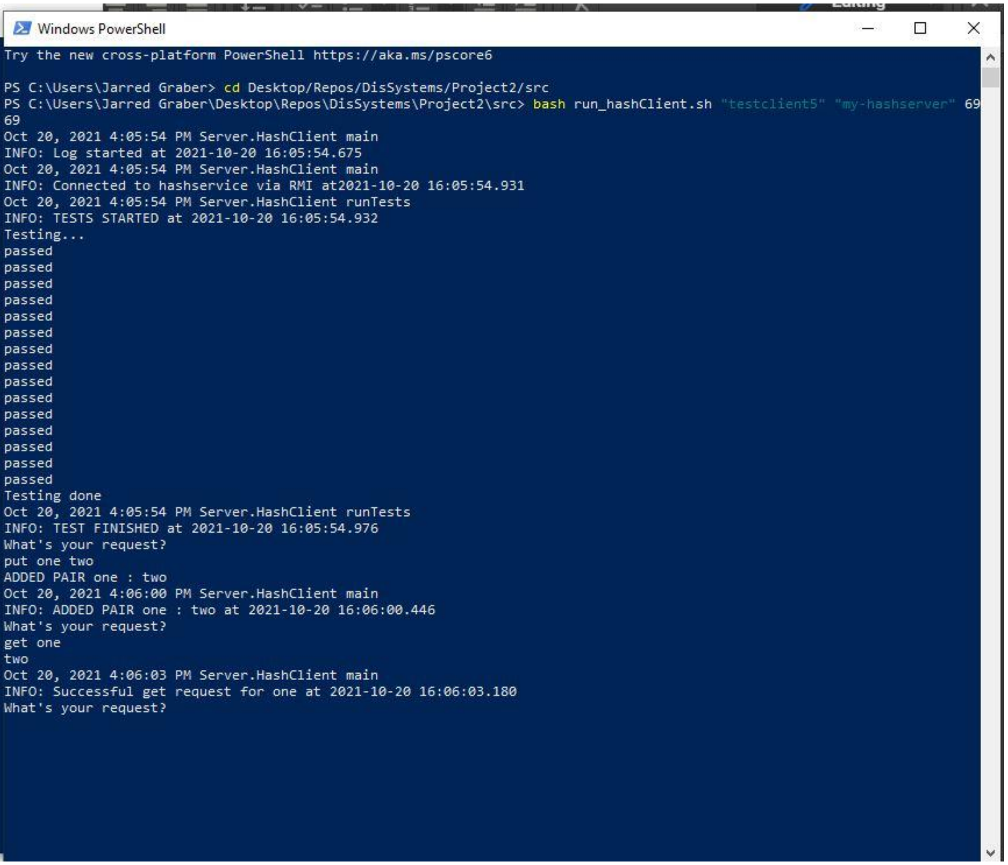
----
Part of your completed assignment submission should be an executive summary containing an “assignment overview” (1 paragraph, up to about 250 words) explaining what you understand to be the purpose and scope of the assignment and a “technical impression” (1–2 paragraphs, about 200–500 words) describing your experiences while carrying out the assignment. The assignment overview shows how well you understand the assignment; the technical impression section helps to determine what parts of the assignment need clarification, improvement, etc., for the future.

During this assignment I learned the fundamentals of RMI and how to connect computers together without using sockets. Instead, how to have computers connect via a port and their IP addresses via a registry. This is different from a TCP connection or UDP where the socket is needed to be known. With RMI another computer can access another computer’s functions and have it process them for us and return to another. This is the start of the foundation of modern day cloud computing, such as systems of Google’s Strava where a person can play high-intensity video games with very little hardware needed. In addition, this project showed how to properly handle multi-threaded projects with synchronized methods that make sure no overlapping of executions go on while the multiple threads are executing.

When completing this project I found the easiest portion to be the actual implementation of multithreading with RMI. Java has a keyword already in the system that helps with the multithreaded systems called “synchronized” that will automatically handle the synchronization of methods in a multithreaded project. Therefore, to implement the proper synchronization all that needed to be done was to add the keyword synchronized the methods needing to be synchronized, in addition to creating a concurrent arraylist that can manage and track the clients connecting to the server. That being said, the project did not require much time outside of the registry issues that went on. The project itself did not take much time as it involved creating interfaces, the server, the client, and then the implementation of said server. The actual coding/design could be copy and pasted for the most part from Project1 with some modifications. I ran into little to no issues except from the registry, which after some reading on the documentation, and help from the TAs it was deciphered that it was an issue of improper rebinding to a name that would not work. Once that was finished I ran into some docker issues running the code, but again after some help from the TAs it was deciphered that the issue was Docker changing the hostname/IP address. So by allowing the user to state what the hostname should be the docker issue can be circumvented by allowing custom hostnames to be made whenever creating a server, and it finally worked.



```
INFO: Delete request for test2 at 2021-10-20 16:03:38.085
passed
Oct 20, 2021 4:03:38 PM Server.HashServer deleteHandler
INFO: Delete request for test3 at 2021-10-20 16:03:38.086
passed
passed
Testing done
Oct 20, 2021 4:03:38 PM Server.HashServer deleteHandler
INFO: Delete request for test4 at 2021-10-20 16:03:38.091
Oct 20, 2021 4:03:38 PM Server.HashServer deleteHandler
INFO: Delete request for test5 at 2021-10-20 16:03:38.092
Oct 20, 2021 4:03:38 PM Server.HashServer runTests
INFO: TEST FINISHED at 2021-10-20 16:03:38.093
Oct 20, 2021 4:05:54 PM Server.HashServer putHandler
INFO: Put request for test1 1 at 2021-10-20 16:05:54.935
Oct 20, 2021 4:05:54 PM Server.HashServer putHandler
INFO: Put request for test2 2 at 2021-10-20 16:05:54.941
Oct 20, 2021 4:05:54 PM Server.HashServer putHandler
INFO: Put request for test3 3 at 2021-10-20 16:05:54.943
Oct 20, 2021 4:05:54 PM Server.HashServer putHandler
INFO: Put request for test4 4 at 2021-10-20 16:05:54.946
Oct 20, 2021 4:05:54 PM Server.HashServer putHandler
INFO: Put request for test5 5 at 2021-10-20 16:05:54.948
Oct 20, 2021 4:05:54 PM Server.HashServer getHandler
INFO: Get request for test1 at 2021-10-20 16:05:54.950
Oct 20, 2021 4:05:54 PM Server.HashServer getHandler
INFO: Get request for test2 at 2021-10-20 16:05:54.953
Oct 20, 2021 4:05:54 PM Server.HashServer getHandler
INFO: Get request for test3 at 2021-10-20 16:05:54.954
Oct 20, 2021 4:05:54 PM Server.HashServer getHandler
INFO: Get request for test4 at 2021-10-20 16:05:54.956
Oct 20, 2021 4:05:54 PM Server.HashServer getHandler
INFO: Get request for test5 at 2021-10-20 16:05:54.965
Oct 20, 2021 4:05:54 PM Server.HashServer deleteHandler
INFO: Delete request for test1 at 2021-10-20 16:05:54.967
Oct 20, 2021 4:05:54 PM Server.HashServer deleteHandler
INFO: Delete request for test2 at 2021-10-20 16:05:54.969
Oct 20, 2021 4:05:54 PM Server.HashServer deleteHandler
INFO: Delete request for test3 at 2021-10-20 16:05:54.970
Oct 20, 2021 4:05:54 PM Server.HashServer deleteHandler
INFO: Delete request for test4 at 2021-10-20 16:05:54.972
Oct 20, 2021 4:05:54 PM Server.HashServer deleteHandler
INFO: Delete request for test5 at 2021-10-20 16:05:54.974
Oct 20, 2021 4:06:00 PM Server.HashServer putHandler
INFO: Put request for one two at 2021-10-20 16:06:00.435
Oct 20, 2021 4:06:03 PM Server.HashServer getHandler
INFO: Get request for one at 2021-10-20 16:06:03.178
```



```
Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Jarred Graber> cd Desktop/Repos/DisSystems/Project2/src
PS C:\Users\Jarred Graber\Desktop\Repos\DisSystems\Project2\src> bash run_hashClient.sh "testclient5" "my-hashserver" 69
69
Oct 20, 2021 4:05:54 PM Server.HashClient main
INFO: Log started at 2021-10-20 16:05:54.675
Oct 20, 2021 4:05:54 PM Server.HashClient main
INFO: Connected to hashservice via RMI at2021-10-20 16:05:54.931
Oct 20, 2021 4:05:54 PM Server.HashClient runTests
INFO: TESTS STARTED at 2021-10-20 16:05:54.932
Testing...
passed
passed
passed
passed
passed
passed
passed
passed
passed
passed
passed
passed
passed
passed
passed
Testing done
Oct 20, 2021 4:05:54 PM Server.HashClient runTests
INFO: TEST FINISHED at 2021-10-20 16:05:54.976
What's your request?
put one two
ADDED PAIR one : two
Oct 20, 2021 4:06:00 PM Server.HashClient main
INFO: ADDED PAIR one : two at 2021-10-20 16:06:00.446
What's your request?
get one
two
Oct 20, 2021 4:06:03 PM Server.HashClient main
INFO: Successful get request for one at 2021-10-20 16:06:03.180
What's your request?
```