Assignment: Songs Retrieval System
Networks and Systems – Networks

This assignment is to be completed and handed in via DUO. All source code should be written in Java or python3.5 (or above) and must be called **Server.java/py** and **Client.java/py**. Furthermore, you have to provide one example input file for the server as .txt file as well as bash scripts per program to use your system and test its functionality.

Assessment tasks
You are required to implement a Client-Server system. Your system should include the following:

Server Program.

- Write a server program in Java/Python. Name your program **Server.java** or **server.py** depending on the programming language you used for your implementation.
- Once the server starts, it is passed a text file (please use the file provided to you on DUO (feel free to take one from http://textfiles.com/music/ for testing your system) enlisting artists plus their songs which is read in and held in a map, i.e. the map holds tuples of type artist-song.
- The server must listen at a predefined host and port (you can specify the server name (local host) and port in your program) for incoming TCP/IP requests.
- Client program(s) connecting to the server will communicate with it via the local host and the specified port.
- The server receives an artist name from client and finds all the song(s) associated to that artist.
- The server informs client(s) that it received the request successfully.
- The server retrieves the required information (songs associate to a given artist) from the text file and returns the result to the client.
- The server should handle situations such as unavailable/busy ports, client requesting a non-existing artist in the input file and no songs associated to a given artist.

Client program

- Write a client program in Java/python. Name your program **Client.java** or **client.py,** depending on the programming language you used for your implementation.
- Once the client program is running, it automatically connects to the server via the predefined host and port number that the server accepts connections from. For this assignment, the server should run on the same machine as the client program.
- The client informs the user that server connection is successful, otherwise it should display appropriate error message(s) to the user.
- The client prompts the user for the name of an artist.
- The client receives the artist name from the user and sends it to the server.
- The client receives a response (which should contain song(s) associated to the artist) from the server and displays them on the screen (e.g., the terminal) for the user. Note the server program should retrieve the songs from the supplied text file.
- Your setup should handle situations that may occur prior, during and after client-server interaction. As minimum the client program should handle situations such as:
  - the server is not running/unavailable;
  - the port number is busy;
  - user did not input artist name; and
  - server response is not received.

  You should print out those problems on screen.

Log files

- The server should record the following events to a log file and name the file **server.log**:
    - the date and time the server has started;
    - the date and time of incoming client connection request;
    - whether connection was successful or not;
    - the received artist name from client; and
    - the length of time the server was connected to a client (the period between client connection and disconnection).
- The client program should record the following events that occur during the client-server communication in a log file and name the file **client.log**:
    - server response time for a request (e.g., “It took 0.0123 ms to receive a response from the server for the request to get songs for ‘artist name’”);
    - server response length (e.g., “The response length was *n* bytes”); and
    - the date and time the client received a response from the server (e.g., “Server response received on day/mm/yyyy H:mm:s”).

Bash files

- Write a bash file to contain instructions to start the server program. The bash file should be executed from a terminal. Name the file **server.sh**.
- Write a bash file to contain instructions to start the client program. The bash file should be executed from a terminal. Name the file **client.sh**.

Terminate client-server connection

- After the client sent a request, received and processed the server’s response for the user, it should prompt the user to disconnect from the server.
- The user should send the command (quit) to the server to disconnect from the server.
- The server should close the connection.
- The client should confirm to the user that the connection is closed.

Other requirements:

- The assignment requirements (e.g., program and file naming requirements, submission requirements, etc.) have been followed.
- Clear instructions to use the system.
- Your system is compatible for different operating systems (Windows, Mac and Linux).
- The system can operate on any university machine.

Submission details:

Important information:

- Submission date: 10/12/2018 at 14:00
- Submission mode: via DUO
- Feedback date: 28/01/2019

Important requirements:

Compress all your files in a zip file. Name your zip file as “bannerID.zip” (***Make sure to replace ‘bannerID’ with the anonymous banner ID given to you by the university***). The zip file should contain the following:

- The server and client source code.
- The required system log files with correct information.

- The required bash files with correct commands for running your programs. Your bash files should compile your source code (if you opt to use Java) and run the system with the appropriate input file, host name, port, etc. for the server program and any default information for the client program. The input file should be the file that you used to test your program and it should be included in your submission. ***Make sure your programs have the correct path for the required files for read/write*** purposes.
- A readme.txt (less than 200 words) stating any details that we should be aware of in order to run your program. Failure to provide the required information for successfully testing the system may result in partial, or full, mark loses.

Collaboration policy
You can discuss your work with anyone, but you must avoid collision and plagiarism. ***<u>Your work will be assessed for collision and plagiarism through plagiarism detection tools.</u>***

Feedback sheet

Criterion	**Mark**	Comment
Sever program is implemented: Code realises all the functional requirements.	(/20)	
Client program is implemented. Code realises all the functional requirements.	(/20)	
Server-client disconnection is implemented and code realises all functional requirements.	(/15)	
Log files are generated automatically by the programs and contain correct information.	(/10)	
Bash files contain correct instructions and execute successfully.	(/5)	
Error handling: The programs handle all errors appropriately.	(/10)	
Code quality: • Program code is sufficiently commented. • Programs are appropriately structured (e.g., correct and consistent indentation style). • Appropriate naming convention is used for variables, methods/functions and classes throughout the project. • Appropriate use of external libraries and there is no evidence of redundant code, e.g. unnecessarily importing Java/Python classes. • Correct/appropriate use of conditional statement and iterative statements.	(/10)	
All of the **‘Other requirements’** of the assignment have been met.	(/10)	
Total	**(/100)**	