**Chat App**

The application consists of a server application and a client application.

Server App:

Consists of 5 modules:

1- Server Class:

It extends the Thread class to handle having many clients connecting to the server app.

Class constructor accepts the socket instance.

The run function:

it runs per thread instance created. It reads It calls handshake function in the user handler module which will check for user name and password of the client.

After authentication the function will start reading messages from the client and append it to a string builder calls fullMessage.

When accepting null from the client means the the client program has stopped or end v conversation message "Bye Bye" the server saves the massege text and stats and breaks out of the loop.

2-Main Class:

The main function:

it sets the key store file for the server and its password. Then create SSL Server socket through Socket Factory.

after that for each connection the socket accepts an instance of the server thread will be created to handle this connection.

3-UserHandler Class:

Contains functions to handle users' processes:

1- getUserCred: getting the user credentials (user name and password) if he is new he

2-checkAuth: check user cred against the saved ones and return true or false

3-handShake: loops until the user gives valid inputs or create new ones.

4-FileHandler Class:

Contains functions that deal with files:

1- readFile: reads a given file and returns its content as a String builder object.

2- getJsonMag: get the jso from a string builder object and put it into hashmap and return it.

3-writeJson: write a hashmap to a json file

5-EndConversationHandler:

1-saveConversationText: saves the whole conversation of the user in file.txt called by his user name.

2-freqOfWords: compute the freaquency of words in the conversation and returns a hashmap object contains it.

3-saveConversationState: to save the frequency the text to files.

Client App:

Main Class:

1- setup the SSL connection properties like trust store and password

2- create secured connection to the server

3- get user input user name and password to authenticate with the server

4- get user input message line by line and send it to the server