**SERVICE-ORIENTED APPLICATION DEVELOPMENT**

**Hands on practice #2,3/6**

Socket Client-Server Chat Program

# Objectives:

After this lab, students will be able to:

* Create and manage chat applications using Java language
* Design and code client-server chat application based on Java language

# Requirements

* Java 8
* IDE for java programming: Eclipse

# Assessment descriptions

Students are assessed based on the level of completion in each practice section.

# Hands On Practices

## Reading guidance (45 mins)

Overview:

* The Java Chat application is launched from the command line. The server and clients can run on a local machine or on different computers in the same network, e.g. Local Area Network (LAN).
* There can be multiple clients connect to a server and they can chat to each other. There’s no private chat between two users (homework).
* After getting connected to the server, a user must provide his or her name to enter the chat. The server sends a list of currently online users to the new user.
* Every user is notified when a new user arrives and when a user has gone. Each message is prefixed with the username to keep track who sent the message. The user can say ‘bye’ to quit the chat.
* The application consists of two parts: server and client. Each part can run independently on separate computers.

Follow the guidance in the URL: https://www.codejava.net/java-se/networking/how-to-create-a-chat-console-application-in-java-using-socket

## Chat Server Program (90 mins)

* The server is implemented by two classes: ChatServer and another class **X** *(student named it)***extend** *Thread* class.
* The ChatServer class starts the server, listening on a specific port. When a new client is connected, an instance of X is created to serve that client.
* Since each connection is processed in a separate thread, the server is able to handle multiple clients at the same time.
* To keep track the names and threads of connected clients, Student can use **Set** collection
* The X (Thread) class is responsible for reading messages sent from the client and broadcasting messages to all other clients.
* First, it sends a list of online users to the new user. Then it reads the username and notifies other users about the new user.
* Some other methods in Chatserver class that student must implement : addUser, removeUser, getUserName, printUser,…

## Chat Client Program (90 mins)

* The client is implemented by three classes: ChatClient, ReadThread and WriteThread extend Thread class.
* The ChatClient starts the client program, connects to server . Once the connection is made, it creates and starts two threads ReadThread and WriteThread.
* The ReadThread is responsible for reading input from the server and printing it to the console repeatedly, until the client disconnects
* And the WriteThread is responsible for reading input from the user and sending it to the server, continuously until the user types ‘bye’ to end the chat

## Self-Implementation (90 mins)

Student design and code three features to the program: **Register, login and Private Chat**