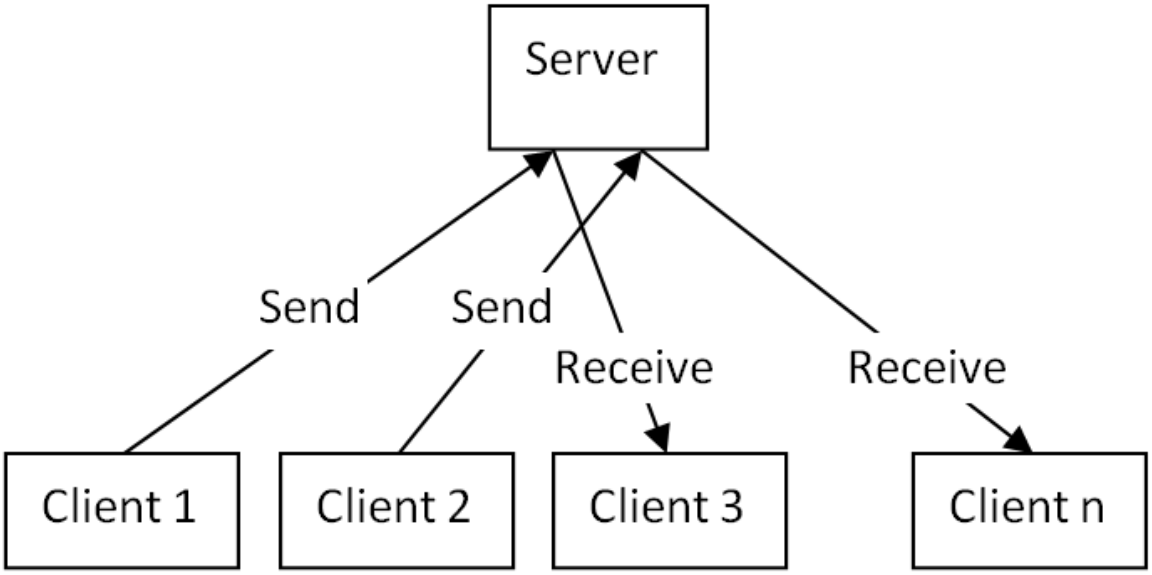
06/12/2017

Programming Languages
BLM2541
Term Project - V1.0
(Due 08/01/2018)

Important Note: **This document might be updated to answer your FAQ, so please frequently check website for updates and check the version number of document to see if it is updated.**

SUBJECT

You are required to implement a Multi-User MESSAGING application using C language. The project should include 2 different programs running independently on server-side and client-side. The server-side program would listen to the clients and capture messages and then deliver them to the corresponding client (user). The client-side programs would react like users and these programs could be run on different terminals simultaneously. The details of the system are given below. Please read them carefully and strictly conform to these details while implementing the system.



1. System Details

The server-side program would have the following capabilities:

It would

- listen to the clients continuously.
- deliver the messages to the corresponding user
- store all the user list of each client
- deliver the alert messages to the corresponding user

The client-side program would have the following capabilities:

User would

- display their contact list
- add/delete a user to/from contact list
- check message(s)
- write a message and send it to one of the users within the contact list
- receive a message and read it
- display the message history of each user within his/her contact list (should be ordered by date ascending)
- delete corresponding message(s)
- see alert messages including "Message Received", "Message Sent", "Message Read".

2. TIPS AND HINTS

This project should be partitioned into several modules and each module should be implemented separately. You should also clearly define inputs and outputs of each module.

- There would be two separate programs. server.c and client.c
- Both programs should run on localhost but separately.
- Only one server.exe would run at the same time whereas many clients.exe would run on different terminals simultaneously.
- Each client should login to the system by sending his/her unique userid. ----→ .\client userid
- The server-side program should compare the userid to the userlist saved on the server-side. If the user does not exist within the list, the server should add the user to the list and create a contact list for that user.
- Each client should send a request to the server by selecting "CHECK MESSAGES" item in the menu in order to get new message(s).
- There will be 3 different types of messages.
    - Request for Login
    - Request for Sending a Message
    - Request for Receiving a Message
- The following information of the users should also be stored at server side: **UserID(unique), Mobile Phone Number, Name, Surname**.

Do not forget to read documents about socket programming.

EXAMPLE

Each user should login to the system by typing UserID from terminal.

.\client 1

Client should send a message using the following format:
UserID of the receiver, message body.
For example in order to send a message to the user with UserID2, the client should type:

- 2 Nasilsin?

Your client program should have such a menu and should give the following outputs for the given inputs:

1. List Contacts
2. Add User
3. Delete User
4. Send Message
5. Check Messages

.

.

- Please type your choice? 1

  Your contacts:

Userid	PhoneNumber	Name Surname
1	5385754760	Ali Ates
2	5594932046	Derin Deniz
3	5835749292	Demir Bilek

- Please type your choice? 4

  2 Nasilsin?

- Please type your choice? 5

  You have messages from User 3
  You have messages from User 2
  Whose messages do you want to read? 3
  Biraz gecikecegim.
  Sen neredesin?