CMPE/ISYE/ITEC 431 Computer Network Design and Applications
Short Message Server Project Details (2018120401)

Objectives:
After finished this project students develop their own simple short message server and client applications.

Recommendations & Rules:

- The applications must be written in **C/C++** in **Linux** Platform.
- Server application should operate in **concurrent mode**. (So application serve more than one user at a time)
- Only **listed library** can be used for file parsing. Using other libraries or classes prohibited. Write your own functions if it's necessary.
- Unless otherwise explicitly specified, all written assignments or code that is submitted is to be entirely the student's own work. Using any code or copying any assignment from others is strictly prohibited without advance prior permission from the instructor.
- All students' work is their own. Students who do share their work with others are as responsible for academic dishonesty as the student receiving the material.
- Students who encourage others to cheat or plagiarize, or students who are aware of plagiarism or cheating and do not report it are also participating in academically dishonest behavior.
- In this case (academically dishonest behavior), students to be punished with no grade.

Allowed Libraries:

- Standart POSIX socket library.
- Ini parser, https://github.com/ndevilla/iniparser

Project Details:

1) Server should bind (listen) port that defined in application configuration file. This port uses for command exchange and data transfer purpose. All data exchange should use TCP transport layer protocol for reliability.

2) Server must have a configuration file (.ini file). This file includes configuration parameters of server. Configuration file example given in Appendix section (server.ini).

3) Message Server application should operate as defined below.

Server initially show greeting message as it defined in server.ini with ServerMsg.

Then client should authenticate itself with User Keyword. Given values must check from server.ini. If given values correct so client can pass next stage.
USER UserID Password
Possible Responses:
+**OK** UserID and password okay go ahead.
-**ERR** Either UserID or Password is wrong.

After authentication clients can give following commands:
LIST [num]
This command can be used for listing user message that available in server. Num is optional.
Possible Responses:
+OK listing follows.

Server list messages with date order. Format looks like;
<num> <from_id> <unix_time_stamp> <octets>
<num> : Message number that can be used with RET and DEL commands.
<from_id> : This is represent who was send this message.
<unix_time_stamp> : This is unix timestamp that starting 1970.
<octets> : Message size as byte.
Note: All message attributes should taken from message filename. Please check SEND command.

-ERR there are no message.

RET ID
This command can be used for reading user message with given ID.
Possible Responses:
+OK message follows.
-ERR no such message.

DEL ID
This command can be used for deleting user message with given ID.
Possible Responses:
+OK message deleted.
-ERR no such message.

SEND UserID TXT
This command can be used for sending text message (TXT) to given UserID. The sended message should store on the storage.

While message TXT is sended to UserID in this case server must store this message with using following structure.
Directory path should be ServerRoot/UserID
Ex: Path=/data/home/20110500/server/10000

Each message should store into this directory with following structure:
<Unix_Timestamp>_<From_UserID>_<Txt_size>.msg
Ex: 1368479000_10001_100.msg
This file format should be use in LIST command as well.

Possible Responses:
+OK message sended to <UserID>.
-ERR message can not send to <UserID>.

QUIT
This command can be used for disconnect from server.
Possible Responses:

+OK Bye <UserID>.
-ERR Error occurred Signoff not possible.

4) Appropriate response codes should be implemented.
5) All sended message should store to ServerRoot directory. But each user should have their own subdirectory.
6) Server should also log all client activities like Authentication (Good and Bad attempts), Send message from to whom etc.
7) Client must have a configuration file (ini file). This file includes configuration parameters of server. Configuration file example given in Appendix section (config.ini).
8) Following parts represent typical client server interaction. S represent as Server and C represent as Client.

S: +OK My Chat Server v0.1 Ready.
C: USER 10000 myPass

S: +OK UserID and password okay go ahead.	**S: -ERR** Either UserID or Password is wrong.

If Ret Code is OK:

C: LIST

S: +OK 3 messages (350 Octets) S: 1 10001 1368479000 100 S: 2 10002 1368479300 120 S: 3 10001 1368479500 130 S: . For checking size of 2.nd message. C: LIST 2 S: +OK 2 10002 1368479300 120 S: .	**S: -ERR** There are no message in your mailbox. **C:** LIST 4 **S: -ERR** no such message, only 4 messages in maildrop

C: RET 1	C: RET 3
S: +OK 10002 1368479300 100 Octets S: This is my first message that i got from this system. S: .	**S: -ERR** no such message available.

C: DEL 2	C: DEL 3
S: +OK message 1 deleted.	**S: -ERR** message 3 already deleted.

C: SEND 10001 “Hello world”	C: SEND 10011 “Hello world”
S: +OK message sended to 10001.	**S: -ERR** message can not send to 10011. User 10011 not available.

C: QUIT	C: QUIT
S: **+OK** Good Bye 10000.	**-ERR** Error occurred Sign-off not possible.

9) Client should have following features;

a) Client should read all configuration parameters from client.ini file. Ini parameters can be used as initial values. But can be changed in runtime time as well.
b) Client should give option to users to writing target UserID and Txt message interactively.
c) Client should give option to users for closing and terminating connection.(QUIT)
d) All application layer commands should hide by client application. Just informative messages should be show by clients.

Ex: Client should can be represent following menu for user

Hello 10001,
Please choose your option:
1) Read/Delete Messages
2) Write Message to User
3) Change config parameters
4) Quit

Option->

If you select 1 Option;

Step 1	Your Messages listed below: 1) From: 10001,Date: 2013-05-13 15:00, 100 Byte 2) From: 10002,Date: 2013-05-13 15:30, 120 Byte 3) From: 10001,Date: 2013-05-13 15:32, 130 Byte Type R<id> for Reading, D<id> for deleting ->
Step 2	R1
Step 3	Your Message : “This is my first message.”
Step 4	D1
Step 5	Do you want to delete message <1>?
Step 6	Y
Step 7	Your message was deleted.

If you select 2 Option;

Step 1	Give target UserID->
Step 2	10001
Step 3	Please type text->
Step 4	Hello world.
Step 5	Would you like to Send message to 10001? (Y/N)
Step 6	Y

If you select 3 Option;

Step 1	Would you change any following value? 1) TargetServer : 127.0.0.1 2) TargetPort : 1888 3) UserID : 10000 4) Passwd : myPass Option->
Step 2	1
Step 3	Please give new TargetServer->
Step 4	172.16.32.10
Step 5	Do you want to use this new value? (Y/N)
Step 6	Y

APPENDIX

server.ini:

```
# This is Chat server configuration file. Its have several attributes.
# The lines begin with # accepted as comment line.
# Start for server section
[server]
# ListenIp defines listen ip address of application
ListenIp=All ; If its defined as All its listen all ip address or if defined  ip address its only listen this ip address
# ListenPort defines listening port of application
ListenPort=1888

# ServerHomeDirectory  defines server root directory
ServerRoot=/data/home/20110500/server

# ServerMsg defines server greeting message to client
ServerMsg=”My Chat Server v0.1”

[users]
#UserID:Password
10000:myPass
10001:TesT
10002:Something

#end file
```

client.ini:

```
# This is chat client configuration file. Its have several attributes.
# The lines begin with # accepted as comment line.
# Start for client section
[client]
#  TargetServer defines target host that you would like to connect
TargetServer=127.0.0.1

# TargetPort defines remote port number
TargetPort=1888
# UserID defines client username
UserID=10000
# Passwd defines client password at authentication
Passwd=myPass

#end file
```