Intro. to Network Programming 2020 Fall

Homework 2 - Bulletin Board System: Part 2

Description

Continuing the first part, you are asking to implement boards and posts in the Bulletin Board System (BBS). The server should be **multiprocessing / multithreading**, and the following function will be connected by **TCP**. Your posts should maintain in the shared memory, that is, all operations related to posts should be done in shared memory, and other operations have no limitation.

Requirement

The service accepts the following commands and **at least 10 clients**:

When client enter command incompletely, that is, missing parameters, the server should show **command format** for client.

If command is in the right format, the first failure message will have the higher priority, for example of the “create-board” command, the user didn’t login, then the result should be “Please login first”.

Command format	Description	Result	
create-board <name>	Create a board which named <name>. <name> must **be unique.** If Board's name is already used, show failed message, otherwise it is success. Must be logged in when creating board’s name. There is no limit to do the operation by shared memory or database.	Success	Create board successfully.
		Fail (1)	Please login first.
		Fail (2)	Board already exists.
create-post <board-name> --title <title> --content <content> (command is in the same line)	Create a post which title is <title> and content is <content>. Use --title and --content to separate titles and content. <title> **can have space but only in one line**. <content> can have space, and key in ** ** to indicate a new line. Do the operation by shared memory. Assign a unique serial number to each post. The serial number will start from 1 and increase by creating post.	Success	Create post successfully.
		Fail (1)	Please login first.
		Fail (2)	Board does not exist.

list-board	List all boards in BBS. There is no limit to do the operation by shared memory or database.	Index Name Moderator <Index1> <Name1> <Moderator1>	
list-post <board-name>	List all posts in a board named <board-name> Do the operation by shared memory. Display the date when the post creates.	Success	
		S/N Title Author Date < S/N 1> <Title 1> <Author 1> <Date1>	
		Fail	Board does not exist.
read <post-S/N>	Show the post which S/N is <post-S/N>. Do the operation by shared memory.	Success	Author: <Author1> Title: <Title1> Date: <Date1> -- <content> -- <User1>: <Comment1>
		Fail	Post does not exist.
delete-post <post-S/N>	Delete the post which S/N is <post-S/N>. Only the post owner can delete the post. If the user is not the post owner, show failed message, otherwise it is success. Do the operation by shared memory.	Success	Delete successfully.
		Fail (1)	Please login first.
		Fail (2)	Post does not exist.
		Fail (3)	Not the post owner.
update-post <post-S/N> --title/content <new>	Update the post which S/N is <post-S/N>. Use -- to decide which to modify, title or content, and replaced by <new>. Only the post owner can update the post. If the user is not the post owner, show failed message, otherwise it is success. Do the operation by shared memory.	Success	Update successfully.
		Fail (1)	Please login first.
		Fail (2)	Post does not exist.
		Fail (3)	Not the post owner.
comment <post-S/N> <comment>	Leave a comment <comment> at the post which S/N is <post-S/N>. Do the operation by shared memory.	Success	Comment successfully.
		Fail (1)	Please login first.
		Fail (2)	Post does not exist.
exit	Close connection.		

Scenario

```
bash$ client 127.0.0.1 7890
********************************
** Welcome to the BBS server. **
********************************
% create-board NP_HW
Please login first.
% register Bob bob@qwer.asdf 123456
Register successfully.
% register Sam sam@qwer.com 654321
Register successfully.
% login Bob 123456
Welcome, Bob.
% create-board NP_HW
Create board successfully.
% create-board NP_HW
Board already exists.
% create-board OS_HW
Create board successfully.
% create-board FF
Create board successfully.
% list-board
Index        Name          Moderator
1            NP_HW         Bob
2            OS_HW         Bob
3            FF            Bob
% create-post NCTU --title About NP HW_2 --content Help!<br>I have some problem!
Board does not exist.
% create-post NP_HW --title About NP HW_2 --content Help!<br>I have some problem!
```

```
Create post successfully.

% create-post NP_HW --title HW_3 --content Ask!<br>Is NP HW_3 Released?

Create post successfully.

% list-post NP

Board does not exist.

% list-post NP_HW

S/N            Title            Author            Date

1              About NP HW_2    Bob               10/26

2              HW_3             Bob               10/26

% read 100

Post does not exist.

% read 1

Author: Bob

Title: About NP HW_2

Date: 10/26

--

Help!

I have some problem!

--

% update-post 100 --title NP HW_2

Post does not exist.

% update-post 1 --title NP HW_2

Update successfully.

% read 1

Author: Bob

Title: NP HW_2

Date: 10/26

--
```

```
Help!

I have some problem!

--

% logout

Bye, Bob.

% login Sam 654321

Welcome, Sam.

% update-post 1 --content Ha! ha! ha!

Not the post owner.

% delete-post 1

Not the post owner.

% comment 100 Ha! ha! ha!

Post does not exist.

% comment 1 Ha! ha! ha!

Comment successfully.

% read 1

Author: Bob

Title: NP HW_2

Date: 10/26

--

Help!

I have some problem!

--

Sam: Ha! ha! ha!

% create-board Hello

Create board successfully.

% list-board

Index        Name          Moderator
```

```
1               NP_HW           Bob

2               OS_HW           Bob

3               FF              Bob

4               Hello           Sam

% logout

Bye, Sam.

% login Bob 123456

Welcome, Bob.

% delete-post 1

Delete successfully.

% read 1

Post does not exist.

% logout

Bye, Bob.

% exit
```

Grade (100%)

- **TCP connection**. - (5%)
- **Multiprocessing / Multithreading**. - (10%)
- **Shared memory**. - (10%)
- **create-board** command. - (10%)
- **create-post** command. - (10%)
- **list-board** command. - (10%)
- **list-post** command. - (10%)
- **read** command. - (10%)
- **delete-post** command. - (5%)
- **update-post** command. - (10%)
- **comment** command. - (10%)

Submission

Please upload a zip file called “hw2_student_id.zip”, for example, “hw2_123456789.zip” that includes your **source code** (server and client) and **executable file** (alternative). If you are using python, it might be no executable file. You can submit your source code only, but you should make sure that you can run the bash script by your python file. **You should upload on time!** Submission that don’t follow the rule will **get 20% punishment** on the grade. You will get **0 points** on this project for **plagiarism**. Please don’t copy-paste any code!

Demo

There is script called **nphw2.sh**. Before you submit your executable to New E3, please check whether your code is able to run by the script. As same as homework1 we will provide a script called **npdemo2.sh** for you to download the script and run it. If you **don’t know how to run npdemo2.sh** at demo time, TA will take **10 points** off. For those don’t have a laptop, we provide a VirtualBox environment (link on New E3) for you to run your code, or you can use remote desktop this time, the software will be TeamViewer tentative. **You can’t modify your code after the deadline of submission.** TA will ask some questions about your code. If you can’t answer TA’s questions or you are not clear on your code, TA will take **10 points** off for **each question**. You can execute the script by following, but there are no regulations:

Change mode of nphw2.sh: chmod u+x nphw2.sh

- C/C++: ./nphw2.sh ./client 127.0.0.1 7890
- Java: ./nphw2.sh “java client” 127.0.0.1 7890
- Python2: ./nphw2.sh “python client.py” 127.0.0.1 7890
- Python3: ./nphw2.sh “python3 client.py” 127.0.0.1 7890

Reference

1. C/C++ Socket
2. SQLite C/C++ Interface
3. Linux socket SELECT
4. Linux semaphore
5. Linux fork