To detail our program's thread synchronization requirements we'll just go over specific things and that should provide an overall detail on it as well as providing detail on the entire program. Also all the main commands you input(start, create, debit, etc.) have to be in lowercase. In the instructions all of it is in lowercase but I just wanted to write this part just in case.

bank.h has two structs that we use in the other files. In the Account struct there is the character array for the name, the balance, the flag(insession), and a mutex lock so that the accounts itself can be locked. The other struct carries the bank information such as having an array that can have 20 accounts, and other information pertaining to that which is how many accounts have been created(accountnum) and we also have a mutex lock in here as well. The other parts of bank.h are methods to be used in bank.c

For bank.c--In the bank file is where most of the bank related parts are done. So the methods related to account creation and starting the account are within this. When the program is creating an account(if successful) then it will lock, increment the number of accounts, add the account, and then will unlock again. When looking for which account to do the function on(debit, credit, etc.) it goes through the account array and finds the correct account(by name) and returns said account and. We thought at first to lock the mutex while doing this but we found out later it wasn't completely necessary so we took that part of the code out. Those two methods are the main part of bank.c

In client.c we have two methods that just print out the direction prompts in the client window. Another method is used to print out certain errors and also serves to close down the client once the server closes. The valid method is just to make sure what is inputted is a valid input and the main method serves to give the prompts the server.

In server.c we use one of the methods(periodic_action_cycle_thread) to print out the necessary stuff on the server every 20 seconds(semaphore/timer). When we're printing out the info the method checks if the bank is already locked because then it will have to hold on and if it is not locked then the bank will lock the mutex and then print the necessary information. After completion then it will unlock the mutex. One problem we had with this is we had a hard time getting the program to print out on the server window if the account was in service or not because we couldn't get it to work properly. Client_session_thread is used to connect with the client and it either calls the createAccount method and otherwise does the other parts(debit, credit, etc.) by itself. The main method is used to see if the server can start connecting with the client.