Dhruv Duggal and Tanvi Mankame

Systems Programming

Assignment 3

Files and Design:

1. bankingServer.c:
    a. The server attempts to accept a connection from a client program through a socket descriptor. Once the socket descriptor is received, a thread for printing out the bank information is created and runs the whole time. Afterwards, another thread is created to accept a connection to each client. Within that thread function, after each client is accepted, another thread function is created for each specific client. The client thread function reads the input from a client and then runs the respective functions based on what the client says. The createAccount() function takes in the name of what the function should be and then creates a new account with that name. There is a semaphore lock on that function where the account is being created so that other clients cannot add while one is adding. The serveAccount() function activates an account and puts it in session so that the deposit, withdraw, balance, and end function can work. The withdraw() function allows the user to take money out of the bank account if the amount that is being withdrawn is less than the amount that is already in the account. The deposit() function deposits any amount of money into the bank. The query() function returns what the balance of the account that is currently being served is. It only

works if the account is in session at that time. The endAccount() function ends the current session that is running on the current account. The quit() function ends the connection between the client and the server.

2. bankingClient.c
    a. The client program attempts to set up a connection with the server program, it lets the user know when a connection is successful. Once the connection is made, there are two threads created. The first thread is for sending commands to the server in order to interact with the bank accounts. The second thread is for receiving a response from the bankingServer after giving a command. This is so that the client knows whether or not the command that was sent to the server was successful or not.
3. bankingClient.h
    a. This header file includes all the methods used in its respective c program with detailed summaries of what each method is supposed to do.
4. bankingServer.h
    a. This header includes the methods used in the bankingServer.c program with its own detailed summaries.

<u>Assumptions/Difficulties:</u>

- The Client file that will be used is the one that is made by us.
- The server is being done on an iLab machine.

- One difficulty that we had was keeping all the clients running simultaneously while also updating the bank server. This was overcome by adding a lock when each account was being created.
- Another difficulty that we had was interrupting the program every 15 seconds so that the bank can update the status of each account and print the values. This was overcome by using the sleep() function for 15 seconds at a time.

<u>How to Run the Code:</u>

1. To run the programs, type “make” into the terminal to compile the Client and the Server files.
2. Afterwards, make a new terminal for the client and type in
    a. ./bankingClient <machineName> <port number>
        i. ./bankingClient vi.cs.rutgers.edu 9999
3. On another Terminal, type in:
    a. ./bankingServer <port number>
        i. ./bankingServer 9999
4. Afterwards, you may use the commands needed in the terminals to run both programs.