Design Document OS project 2018 – Code by Ultan Kearns

Student no: G00343745
Email: G00343745@gmit.ie

Sections completed of project – 1 – 3 + 5 – 6

Server Side

The Menu

On the server side I used a run method to keep the connection opened I put the code inside an infinite while so that the connection stays open until user teriminates app. I send prompts to user before entering while loop and after. The user is presented with option 1 for register or 2 for login initially. Once logged in the user is presented with options 3-8 as per project specs.

1. Registering User

If the user attempts to register the method checks that the employee id and the email are unique by calling the method userUnique which returns an int if 0 if not unique or 1 if unique. UserUnique uses a scanner to take in each token in file and checks if it matches the parameters passed into the method. if the employee id and email are unique it then calls a synchronized method writeUserFile which takes in userFile as a parameter as well as all the strings entered in registerUser. WriteUserFile then appends the values to the bottom of file using a bufferedReader and a printWriter. The method is synchronized to prevent overwriting by different clients.

2. Login

The login method uses a buffered reader to read each user in the userFile and if parameters in file are equal to the parameters passed into it then it sets an int login to 1, int login is then returned by the method and I use this returned integer to send the client more options if they are logged in and I check this integer when the user attempts to call options 3-8 when not logged in so that they do not exetcute.

3. Register Bug

Register bug asks user for input as per project spec it then calls writeBugFile() which is basically the same as writeUserFile() but it takes in the parameters from registerBugFile() and then uses a counter to keep track of the unique ID for the bug.

5. View All Unassigned files

Show unassignedBugFile() method is called and takes a File f as a parameter. I pass the absolute path to the unassignedBugFile() method which is synchronized. This method uses a bufferedReader() to read through the file and if the current line ends with “Open” then I increment a counter after the file is read I close the reader then reopen it and then I use a while to sendMessage() to the client which I use to print out any bug that has a status of Open. The reason I use a counter is so that I know how many bugs I recieve in the client side.

6. View all bugs

I have a bug file on the server which I read from. I invoke the method readBugFile() which is synchronized but doesn’t necessarily need to be. I use a bufferedReader() and count the lines in the file using an int. I then close and reopen the bufferedReader() and use another while to send all the bugs to the user using sendMessage(). The reason I use a counter is so that I know how many bugs I recieve in the client side.

Client Side

In the client side I prompt the user to enter the IP and port address. The client then uses a scanner to send a string message to the server and the server returns a response eg. 1. to register or 2. to login. Options 3 – 7 are not available unless logged in and on the client side I also check if the user is logged in so I can print out the additional message from the server.

The client has a sendMessage method which takes in a string and then sends it out to user. The client also takes in response messages from server and sends info to server all depenedent on what response is sent to client and what the client sends to server.