Operating Systems project:

This project uses client server socket programming allowing two separate programs to communicate over a network.

This is achieved by the client-side application connecting to the server application via the server sides ip address.

Once the ip address has been verified the line of communication is opened between the two applications allowing for data to be received and sent to each other via input and output object streams.

Each stream of data is either sent of received one at a time which has a top down flow of information between the two applications.

For this project I chose to use an object orientated approach to develop this application. I had considered using a database connection to store the user details and records entered but instead chose to use text files using buffered writers and buffered readers to create and parse the text files.

On the server-side application there holds the following files:

1.EchoServer class

This class controls the information received and sent to the client-side application.

It also creates instances of the User, FitnessRecord and MealRecord classes.

These instances are then stored in array lists to be used for the creation of the below text files.

When a user needs to login he/she is asked for their user name first and then there pps number.

The user text file is then parsed with a while loop and all lines are set to the variables a new instance of User class.

Each instance of User is then stored in an array list for checking against the user login input requested. A for each loop is then used to iterate over the array list of user objects. If They user name and pps number both match any of the objects in the loop the user gains access to the logged in menu, if not they are returned to the main menu.

2.User class

Holding the User name, address, pps number, age, height and weight variables.

3.MealRecord class

Holding Meal type and meal description variables.

4.FitnessRecord class

Holding Activity mode and duration variables.

5. User text file

Once a user registers with the system by entering all the prompted details from the server each stream entered is stored in a variable to be used for writing to a text file.

A Buffered writer is then used to write each captured variable to the text file.

When written each stream is added (in the order hey have been streamed) to a new line in the text file (name, address, pps number etc).

Every time a new user registers with the system they are added to the end of the users.txt text file. This allows for a complete list of valid users and is used for verification of each user should they wish to login to the system.

6.fitness record file

Is created and records are added and deleted by the Echo server class parsing the file. The file is created only when a user is logged in and a record is entered. When this is saved or created by the logged in user. The users pps number which is unique is appended to the text file name allowing only that user to alter the text file while logged in.

To add to this text file the user once logged in first selects the menu option for adding a fitness record. The buffered writer is then used to write a new file or append to it if it already exists.

To view the last ten records of the users file the text file is parsed and stored in an array list.

A string builder is then used to get each element of the array list and store it in a complete string.

This string is then sent to the client side.

To delete a record the index of the array is input by the user and is removed.

The text file is cleared of all text, so it can only receive the updated array list data.

The array list is then reconstructed written to the text file and sent back out the user.

The array list is then reversed to display the last record entered as the first in the list.

7.Meal record file

Is created and records are added and deleted by the Echo server class parsing the file. The file is created only when a user is logged in and a record is entered. When this is saved or created by the logged in user. The users pps number which is unique is appended to the text file name allowing only that user to alter the text file while logged in.

To add to this text file the user once logged in first selects the menu option for adding a meal record. The buffered writer is then used to write a new file or append to it if it already exists.

To view the last ten records of the users file the text file is parsed and stored in an array list.

A string builder is then used to get each element of the array list and store it in a complete string.

This string is then sent to the client side.

To delete a record the index of the array is input by the user and is removed.

The text file is cleared of all text, so it can only receive the updated array list data.

The array list is then reconstructed written to the text file and sent back out the user.

The array list is then reversed to display the last record entered as the first in the list.