Criterion E: Evaluation

Client Evaluation

Methods of Data Collection:

- Observation [O]
- Interview [I]

Within the second interview with my client, my client stated that they were very satisfied with the system overall and that it was very self explanatory. Additionally it also allowed the client to do everything that they had originally asked for in the first interview and previous discussions. He liked the fact that the system would prompt him with what format to input certain pieces of data in, and whenever he did it wrong the system would restate and clarify how to input the data correctly (Khan, 2023)

Evaluation of System against Success Criteria

The following table summarises the client's evaluation of the system by evaluating the system against the success criteria listed in criteria A.

Success Criteria	**Evaluation**
The client must be able to input the name of the exercise, weight/resistance used, number of sets, and number of repetitions. Which should all be stored into different sessions containing multiple exercises.	[O] The client was observed to be able to input the name of the exercise, weight/resistance used, number of sets and reps as well as store multiple exercises into sessions through the addExercise and addSession methods. [I] Response to the question was: "Yeah I could"
The inputted information must be readily available and accessible for the client to view	[O] The client was able to view the sessions, the exercises within the sessions and the data within each exercise readily through the viewSessions and viewExercises methods. [I] Response to the question was: "Yeah I could view the sessions and exercises at any point"
The system must be able to store data from older sessions and keep them saved so that the system does not reset every time the application is closed	[O] The client was able to save and load all data such as previous sessions as well as the exercises completed within each session using the loadSessions and saveSessions methods. All data was retained.

	[I] Response to the question was: “Yeah so long as I remember to save the sessions it does”
The system should be able to prevent the input of invalid data, incorrectly formatted data, or wrong data; and any data that does not match the requirements should be correctly managed	[O] The system did not crash when incorrect data was inputted, all errors had been validated before and were not accepted unless they were verified by the system that the inputted data was the correct format, data type, and made sense. [I] Response to the question was: “Well yeah whenever I would accidentally type a random letter into the sets or reps by accident it would stop me and ask me to enter it again. But yes it didn’t let me input any invalid data like a date that didn’t exist for example.”
The system should be able to request to repeat inputs of data should they be inputted incorrectly.	[O] Upon an invalid entry of data the system would prompt the client to enter data again. The client had found this a very useful feature along with the prompting of formats during data input. [I] Response to the question was: “Yeah just like with the sets and reps I just talked about.”
The system should be able to search through the sessions for certain exercises, and organise/sort the exercises based on muscle group.	[O] The system was able to search and sort through the searchExercises and sortExercises methods, after which the client was able to see the results of the searching and sorting. The client found the sorting very helpful due to it being able to provide the client with insight as to what muscle groups the client needed more exercises for. [I] Response to the question was: “Yeah I could, and I really loved being able to sort the exercises by muscle group, it makes it super handy to look at what muscle groups I need to target more.”
The client should be able to store the data of the sessions and the exercises on a local file.	[O] By using the saveSessions file the system would use serialisation to save the sessions to a .ser file which could then be deserialized and loaded into the system with the loadSessons method. [I] Response to the question was: “Well I’m not too sure if it was saved to a file, let me just check… Oh yeah it did but I can’t open the file normally”, after which I assured the client that this was to protect the data of the file.

The Java System must be less than 750MB total.	[O] As seen by appendix entry 4 the java system can be seen to be less than 750 MB.

Recommendations for Further Development:

Client Recommendations:

Saving Files

My client suggested that the files should be saved in a format that can be opened with external applications such as Notepad, Word, or Excel. Hence. In order to do this instead of using serialization and deserialization for loading and saving the file I could instead save the file to a .csv file locally and load this file. As by using a .csv file it could be viewed in Excel as well.

GUI

My client recommended that it would be better if the menu screen was "cleaner and better" and that it had interfaces similar to that of a mobile application. In order to do this I could use the IntelliJ Ultimate GUI designer in order to create a simple GUI that would give the system a more clean and professional look.

Sessions per week function

My client suggested that the system would be better if there was a way for him to easily see how many sessions he had already completed in the week so far, and compare that to a goal that the client sets in order to see if they are going to the gym regularly enough. In order to do this I could create a counter that increments for every session added and compare this to a value that the user inputs for their target, which would reset each week.

Self Recommendations:

Progression Graph

For each exercise I could implement python methods into my code which could create simple graphs to show the progression of weight in each exercise over time so that my client can see the progressive overload easier.

Workout Splits

For each session I would create a variable in which the client could input a name of the session and could place each session into different splits which the client may use. Such as categorising different sessions into a “Push, Pull, Legs split” or “Än Arnold split” and name each session something like “Push Day” or “Arm Day” so it is easier for the client to see what their target was for each session.

[Word Count: 469]