Purpose – the purpose of the solution is to provide assistance to the client, a teacher, when they are keeping track of student grades, and help them calculate statistics off of this and make managing data inputted easier.

Evaluation criteria – refer to criteria 5

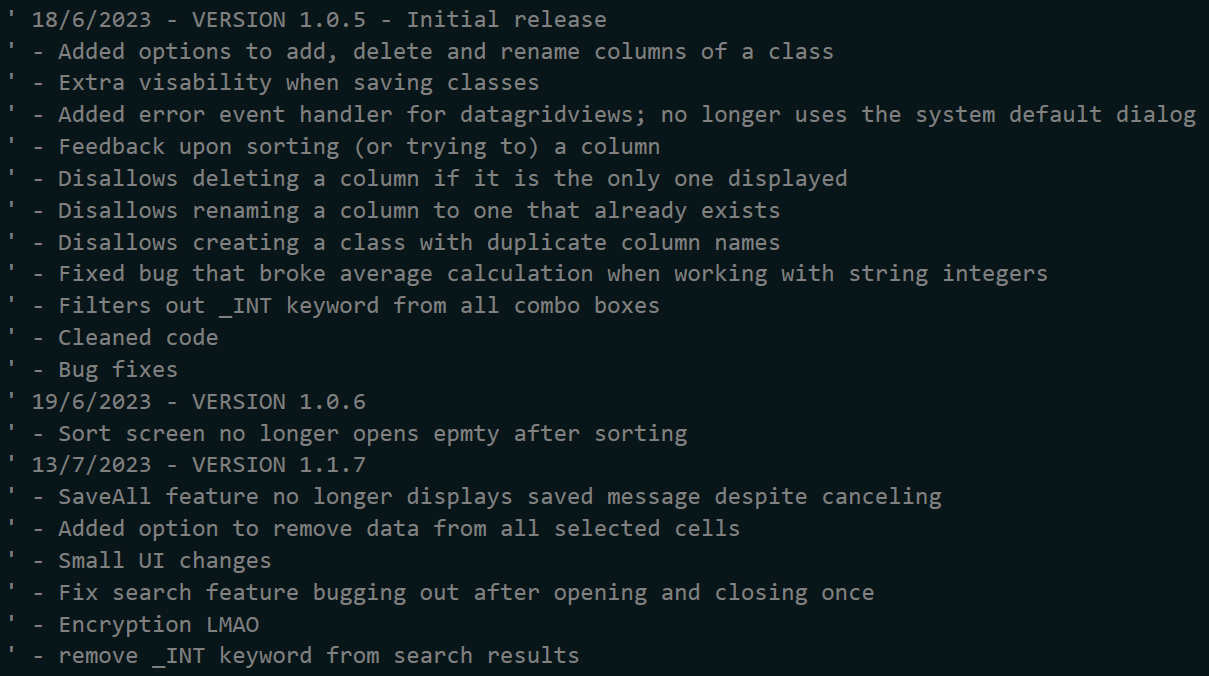
Evaluation strategy

|  |  |  |
| --- | --- | --- |
| Criteria | Strategy | Comments |
| (FR01) The user can login to access the rest of the system | The user was provided a working username and password and asked to log into the program.  Effectiveness:   * Does the solution accept the username and password that was setup beforehand? * Does inputting the correct username and password gain the user to the rest of the system?   Efficiency:   * How fast is it to log into the program? * How easy is it to set up a new username and password. | The users did not have any difficulty understanding or logging in with their username and password that was provided.  Effectiveness was met as the solution properly accepts the username and password and gives access to the rest of the system.  Efficiency was met as the entire login process took less than a minute and it is also very simple to set a new username and password by inputting them in an array within the program |
| (FR02) The user can quickly create a ‘class’ with unique parameters | The user is to create a class with new columns, including at least one integer column.  Effectiveness:   * Is the class created accurate with correct column types and names?   Efficiency:   * Is the time taken to create a class within a reasonable window? * Is the process self-explanatory and without the need for assistance? | All testers were able to create a new class quickly with unique parameters.  Effectiveness was met as the classes created were always accurate with the correct column types  Efficiency was met as the time taken to create a class generally ranged from 30 seconds to 2 minutes. It is also quite intuitive, with no need for any assistance or guidance whatsoever |
| (FR03) The user can quickly input student data into previously created or imported classes and modify it | To test this the user is to create an empty class and input any range of dummy data within it.  Effectiveness:   * Are the inputs accurately recorded within their selected cells with the correct data types?   Efficiency:   * Is the input smooth, with no/little delays? * Is it easy and intuitive to input the data? | All users were able to input student data within their previously created classes correctly.  Effectiveness was met as all data values were accurate and reflective of what the user inputted  Efficiency was met as it is very easy to input any data, requiring only a single click and to type the data into the cell, and users needed no guidance with this process. In extreme cases if the user is inputting very large amounts of data into a single cell the solution will lag slightly however this is not a concern as this would almost never happen under normal circumstances |
| (FR04) Data inputted into class parameters can be sorted and searched | The user is to create a new class with at least 1 column of each type, input a range of dummy data, and search/sort the class.  Effectiveness:   * Does the sorting arrange data correctly based on the row type? * Are the search results correct and relevant?   Efficiency:   * Is the searching and sorting fast, even with larger classes? | All users were able to create their classes as they were instructed previously and had no trouble inputting dummy data and then search/sort it.  Effectiveness was met as the sorting process correctly rearranges the class based on the selected column to be sorted, and search results being accurate and relevant to the task at hand.  Efficiency was met as the sorting and searching was almost instantaneous and without lag, even with very large classes of over 100 students |
| (FR05) Data can be saved or imported | The testers were asked to save their created classes, and then import their saved class.  Effectiveness:   * Does the save file accurately store all data and class parameters in the form of XML? * When importing a file does it read all file contents correctly and create an accurate data grid using it?   Efficiency:   * Does the solution save and import files in a timely frame? * Is the cost of file manipulation low? | All users were able to save their created classes easily and intuitively, and then import them again after.  Effectiveness was met as the files correctly contained the content to reflect the class that they were saved from and reads the files accurately and creates a new class precisely based on the contents within the file.  Efficiency was met as the time taken to save and import file was never longer than 3 seconds, even with very large classes. Apart from the login function, the save/open feature is the only other function within the solution that works with files, and only reads/writes once when the user performs relevant functions, meaning that the cost of file manipulation is very low and does not cost many resources. |
| (FR06) Easy ability to switch in-between classes | The user is to create multiple different classes and attempt to switch between them to investigate the accuracy of each class switched.  Effectiveness:   * Is it easy to navigate between different classes? * Does the solution correctly retain the data within each class when switching?   Efficiency:   * is the process of switching classes smooth and free from delay? * Does the UI clearly show all available classes and allow easy selection? | The testers had no issues creating multiple classes and switch between them.  Effectiveness was met as navigation between different classes were quick and easy, and after cross-checking the data within the classes were correctly retained.  Efficiency was met as switching classes is lag free and instantaneous, and the UI clearly displays the different classes available, with selecting a different class only requiring one click. |
| (NFR01) Easy to use and understand with clean and simple UI | Interview the testers about their thoughts about the UI and how they felt after their experience.  Effectiveness:   * Is it easy for users to learn and use the solution without any prior training? * Is the interface visually appealing and free from clutter?   Efficiency:   * Is the UI responsive and lag free during interactions? * Does it feel logical navigating through different features and functionalities? * Is the UI easy to read and clear? * Is it accessible to all users? | Most users were pleased with the UI and the usability of the solution, however a small minority provided critical feedback regarding the UI aesthetics.  Effectiveness was mostly met, with all users not requiring any help or guidance when using the solution, and most users being pleased with the UI.  Efficiency was met as the UI is lag free and any actions performed were lag free, and navigating throughout the solution is self-intuitive and very logical, with no user having any issues with it. There seemed to be no problems with accessibility, with all users satisfied with the level of accessibility provided. |
| (NFR02) Calculations and stats shown by the software are correct | The testers are to perform a range of stat calculations for various column in their class.  Effectiveness:   * Are the calculations accurate and precise? * Are the stats shown useful and relevant?   Efficiency:   * Is the time taken to perform calculations done within a reasonable time frame, even for large classes? * Is it easy to read and interpret stats shown? | All users were able to perform a range of calculations using the statistics feature within the solution.  Effectiveness was not met, however very close with a few stats being inaccurate. The errors did not deviate too much from the correct answer, usually being a decimal point off. This implies that the either the rounding is wrong, or the index used to find values within the array was 1 off. Users were all pleased with the relevance of the stats shown, with most quoting that it would be quite helpful.  Efficiency was met, with the time taken to perform each individual calculation being almost instantaneous and without lag, and the stats shown easy to read and interpret. One or two users did mention that under certain themes it provides slight difficulties when reading the stats however overall it is still quite efficient. |
| (NFR03) The software can run-on low-end devices without using more than 30% CPU | Test the solution on a lower end device and measure the time taken to perform various functionalities across the program.    Effectiveness:   * Does the software run smoothly without overloading the CPU? * Are all functionalities fully accessible without any performance issues?   Efficiency:   * Does the solution load within a reasonable time and respond quickly to user actions? | There were no low-end devices available making it hard to test this criterion, however many of the tests were conducted on mid-end devices and should prove sufficient for this purpose.  Effectiveness was met, with the solution running very smoothly on a mid-end device, with zero issues whatsoever, and all functions working as intended without any negative impact on the performance of the solution. As tested the solution only takes up 10 percent of CPU on a mid-end device implying that when transferred to a lower end device it should no take up more than 30 percent under normal use.    Efficiency was met as there were no issues with loading or any lag time when performing actions, even with large classes. |

Evaluation of development model implemented -

One of the key strengths of the waterfall model that is its simplicity and linear approach. The waterfall model gave a simple and easy way to follow through each stage as the solution’s scope was quite small and had well defined requirements from the very start. It allowed me a create a simple but effective Gantt chart at the start, outlining all the tasks required, their timeframe and their dependencies. The waterfall model’s strict order of tasks prevented any deviations or scope creep, making sure that the focus remained on the task at hand. As predicted the small project size did not require a complicated development model. This was reflected in the solution, being completed on time and achieved everything in scope, as well as a few things out of scope; most namely encryption. However it might’ve been more suitable to use a waterfall agile hybrid model, with the agile model being used during development, and waterfall for everything else. This is due to the strict linear approach was prompting slight difficulties during the development phase, where client feedback and the option of being able to go to a previous stage would’ve been quite beneficial. Overall the waterfall approach was a suitable choice for a simple project like the grading manager however for larger more complex projects a more flexible development model such as agile or spiral would be more appropriate.

Conclusion of evaluation –

Based off of the results shown in the evaluation strategy shown above, the solution meets almost all of the required criteria, with the only drawbacks being stat calculations (NFR02) being slightly off and some of the themes (NFR01) available being hard to read, both of which would be easy to improve if needed. However, it would not be possible to call the prototype a finished product, as it is still lacking in areas, and would need to be fixed before fully releasing the solution to the public if the client permits it. Based on the feedback from the client, as well as other users, it has been concluded that the solution as it is currently is fit for use for purely the client, as it was intended originally. Throughout the analysis and design stage of the problem-solving methodology some requirements were identified that could not be implemented within the final solution, however these were usually changed or improved in some extent so that the functional and nonfunctional requirements were still met. This is shown in the evaluation criteria created during the design stage of the problem solving methodology, as all criteria apart one being fully met, with that one being of smaller importance and overall not affecting the solutions functionality by much. Throughout each stage so far many improvements have been made to the solution based on feedback from the client and users, with the final evaluation stage providing even more possible changes. The changes made since the initial release are documented within the internal documentation of the solution and shown below: 

With the evaluation strategy several more possible changes are found:

* Improve clarity of the UI: some themes make reading text difficult, including the default theme that most users will be using. Fixing this will improve the clarity and readability of the solution.
* Fix the rounding errors for stat calculations: currently the feature makes minor errors when calculating statistics. Fixing this will improve the accuracy of the solution.