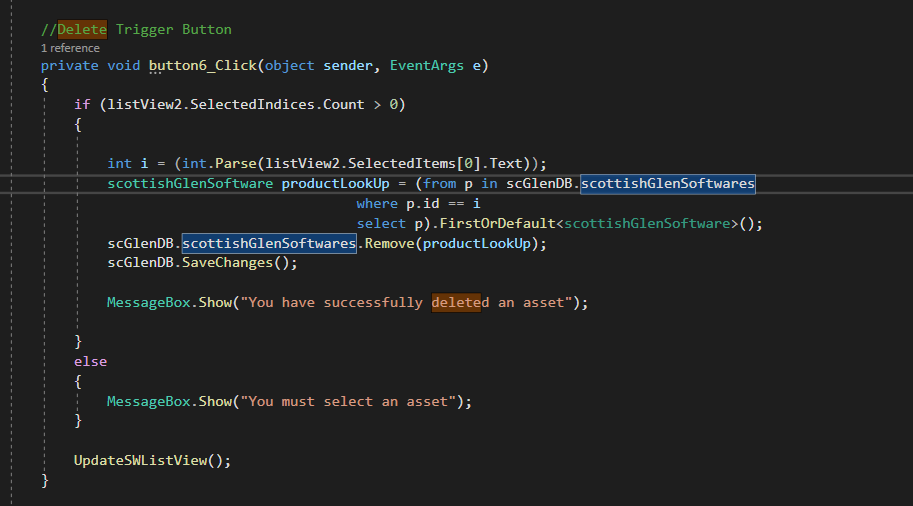
Process Development – Reliability, Secure Coding and Testing.

When it comes to the development process itself, I believe I have learned many new approaches and have experienced several difficulties along the way. At the beginning, when the requirements were only for Hardware Assets and just adding an asset, my implementation was very simple. Not only that but the visual application was also very simple as it only visualized an “Add” button and some input fields which would turn into data in the database. At that moment, I did not really understand and think about reliability, secure coding, or even edge-case testing. I saw that I could successfully add an asset and believed I was doing a great job. However, when the client updated the requirements and many more features were added, I experienced many difficulties and obstacles which I believe led me to a far better project outcome.

To start off, with the growth of the algorithm many things started to get out of hand. For example, there were many flying functions, spaghetti codes, not used variables and very ugly and not used panels and UI components. Not only that but the whole communication between the database server and the application was not trustworthy. The reason for this is that I started implementing without planning out the different logics and thinking in terms of testing and validating. Nevertheless, when many bugs appeared I understood that I had been going in the wrong direction and I had missed some very important methods of testing. Proudly after analyzing, I started to work towards making my algorithm more reliable. One method I used was implementing if checkers in all functions which will communicate with the database and list view. For example, if I want to delete an Item, the application must first know which item I have selected, otherwise an error will break the code and the application would potentially crash. This should not happen to the client. So going through my implementations and checking where I need to add such code was very critical throughout my development process. More to it, I added message alerts which indicate the state/response after an event triggering. This way the user would also be to understand why an event has failed or at least confirm that the event was successful.



Another very important approach in terms of reliability that I have taken is the try and catch functionality. Especially, when we are sending a request via a URL link and waiting for response, it is mandatory to cover the logic in a try-catch algorithm. The reason for this is that many errors could occur, which could potentially make the application crash. And that would be a showstopper for the client. So, using such an approach could potentially save many headaches and problems which could occur during showcasing or even in production.

Text

Description automatically generated

The approaches I have described above, could also fit into the “security” section. As in a way, showing the user that there could be a way to crash the program, would mean that the implementation is not professional, so there could be many vulnerabilities which could be used by hackers to attack and manipulate the whole system or even the database server. Focusing more on security, as the application includes account creation and use, there should be some type of verification and authorization. Since the client has stated that the use and manipulation of the database-stored assets must only be available to users which have registered, my application is not available to guests. All the panels which allow event triggering and data manipulation are disabled unless the user has logged into an account (or registered). Not only that but when the user creates and account, for security measures, password must be at least 8 characters long. After the user has written a password, the application will take the raw text and encrypt it by salting and hashing it. After that process, it would be stored in the database. This way even staff who can see the accounts form the database would not be able to check what the raw password is. Another security measure which is implemented is the change of password and log out system. When the user logs out, all account data is erased from the application itself, leaving no trace for the previous user. And if a user is concerned that the current password is somehow a potential risk, there is an option to change the password. Moreover, for security measures a password can only be changed if the current one is validated, so in a case where the user has left his/her account open, another user will not be able to change the password and steal the account.

Text

Description automatically generated

Text

Description automatically generated

Lastly but not least, the scope of testing has been increasing towards the finalization of my application. The methodology that I have been taking during the development process of this project and overall module assessments has been Agile. Initially, I had defined my functional and non-functional requirements and created the whole specification document, to be sure of what I have to do. After that I started to implement the different features and during the development itself, I tested the functionalities. However, as the functionalities increased from the update from the client specification, the implementation got more “aggressive”, as the deadline caused tension. In order to relieve stress, I split my work into several iterations and after each iteration I went through the code with a critical code review and some validation and testing. This approach of splitting into iterations and testing several times with the intention of optimizing and making the application more efficient, increased my handling of the stress to meet the deadline. Adding to this methodology I had several friends test my application to gain feedback and understanding as to what needs to be improved and is the UI friendly enough to be understandable by users.