*Only 500-800 words required overall, very short document

Dylan – T00252785

As SCRUM leader for this project organizing sprints, I feel like I learnt a lot about how the iterative process works in software development. I feel as if we organized smoothly as a team and team members did their assigned tasks without problems. There wasn't much noticeable suggestion or improvement I could make to improving the teamwork process in the future, aside from perhaps more status updates from team members would help the SCRUM leader stay more in touch with the progress of the project.

One challenge faced was initially coordinating the development of a basic C# Visual Studio Forms prototype using Visual Studio 2022, but we quickly managed to develop an efficient system that allowed everyone to contribute meaningfully. This system involved each team member developing 2-3 different forms functionality at a time specified by the SCRUM leader, which allowed each member to contribute changes and develop freely without conflicting with each other very often.

On one occasion there was a conflict with one of the uploads, as someone had changed the Main Menu forum for example while someone else was still working on an older version. But this was resolved quickly through Gits pull request acceptance system, making a quick edit to reject the old version.

Donnacha – T00248511

For this group project, I feel that despite not doing a lot, I still had an impact on the project’s outcome over the two sprints we did. My main tasks were to develop the UML Class Diagram, build the returnBook() Windows Form for the library, and run the commits and pushes for the group’s GitHub Repository. The UML class diagram was based on the C# classes EBook, Book, Customer, and Library. The returnBook() page allows a user to select a customer’s name. Once a name is selected, the checked-out book will be displayed in the textbox. If satisfied, the user would then click the ‘Return Book’ button to then reset the status of the book to ‘’. A message box pops up confirming the return, and the form then refreshes. It requires the Customer C# files for it to function.

The GitHub repository allows the group to store, push, pull, and clone files. However, there were a few challenges encountered during the project’s development. Firstly, we had to figure out how to clone and push files to the GitHub repository. We resolved it by playing around with the GitHub repository to get used to using it. However, during the project’s development, we encountered an issue with the Main Menu not existing in the Visual Studio solution folder. This was resolved in a rollback.

Calvin Wong – T00247209

For this group project, I have been mainly doing the Customer class file to add and update Customer details by Name, Age, Email Address, Phone No., Membership Status. The Membership Status consists of three memberships - Junior, Senior, and Adult. The Customer class consists of getters and setters for each variable, including a 'getNextAccountID ()' function to generate the next AccountID in the LinkedList. The LinkedList contains all the customer details so when a new customer is added, all customer details are saved into it. This also applies to when you are updating customer details from the LinkedList.

In the NewCustomer WinForms, all textboxes must be filled and validated, so all the details entered are saved and added to the LinkedList of Customers. When validating, this also checks to see if the customer being added has the same email address and telephone number with the only exception that the customer has the same surname as the person in the LinkedList. Upon completion, the AccountID is auto generated in the background in the Customer class, increasing the current AccountID by 1, every time a new account is created.

In the EditCustomer WinForms, the librarian would search up the customer's name or email address and select any one of the matching results in the search textbox. Upon selecting a customer, all textboxes below the search grid will appear to its corresponding function like getName, getAge, etc. Upon completion, the updated data will then be saved and overwritten in the LinkedList.

Robert Smyth – T00145976

From a team perspective, I think we worked well together, especially considering that everyone was doing different things in the project. One thing that stood out to me was how quickly we all organised the structure of the project and how to work on it together. We all used GitHub as our main source and version control system. We did have a couple issues with this e.g. merge conflicts, refactor issues, but we managed to resolve them through communication through WhatsApp, email, and in person. I found that communicating with each other on even smaller issues e.g. class names etc. Regularly prevented big issues from occurring. I found it helpful that we all collaborated well with each other, from small mistakes, bugs, and improvements to possible fixes. We would send small messages on potential fixes and improvements, and I feel this kind of information sharing was positive in our group.

One improvement I would suggest would be have more regular meetings and meetups, even quick one maybe between classes, so we could stay more closely aligned with each other's work and what we were currently working on and if we needed something more of a priority from another member e.g. if we needed someone working on a class to insert a specific variable or something else. I think the teamwork was still effective and supportive, and it made the project more manageable and enjoyable.