CE719 Assignment 2

Part A

The first group I would like to interview are a selection of the Library staff. Being that the success of this project is dependent on meeting the needs of our user, it is imperative that we ascertain problems of the current system, and features that make the maintenance of the system easier for the staff to handle. Furthermore, interviewing the library staff gives my team the chance to preserve any features that are already a success.

The questions I would like to ask during this interview are the following:

1. What are the problems with the current system?

The most frequent users of this system are the library staff themselves. It's important that we know the problems with the current system including any features that need to be removed or updated.

2. Are there any features that you would like to preserve for the updated system?

As well as removing outdated and troublesome system features, there will be functions that work in the systems favour. It would be in libraries interest to preserve the functionality that makes the current system easy to use and would maintain the staff members understanding of how each process works.

3. What new features you would like the updated system to have?

Potentially the most important question in the interview, it is essential that me and my team has a clear understanding of what the library staff would like the new system to contain. What features would make the system more efficient? What features are missing that need to be included in production?

4. Who are you target audience?

A supplementary question to the previous ones. As well as taking a measure of the updated system specifications, we need to make sure we are aware of the libraries target audience. Assuming certain features are accessible by library members, what is the age range of those members? The system needs to look appropriate in response to the audience accessing the software.

As well as conducting staff interviews, it is vital that I have a chance to collect the views of other remote users. Assuming certain aspects of the software are made available to the public (Searching for books, renting books, listing currently owned books), the library members should have their chance to list requirements for the updated system; it needs to be as easy to use for the library members, as it is the library staff themselves. Although some of the interview questions are similar to the previous, I believe it necessary that we have suggestions from two audiences thus aiding us prioritize the most common response.

The questions I would like to ask during this interview are the following:

1. What do you find difficult when using the current library system?

Library members constantly need to utilize specific system functions i.e. searching for available books, checking when their books are due to be returned and listing books they have borrowed. For this reason I have designed a question to ascertain the members opinions as to the applications usability. It's important that we know what features are difficult to use as well as those that make the system generally hard to interpret.

2. How would you like the system to behave when you use it?

Similar to the previous interview, the library members should have a say in how the software is designed and how they want the system to behave. Working in combination with the previous responses, this question is designed to spark innovative functionality ideas to not only combat current issues, but to increase the efficiency and usability of the system as a whole.

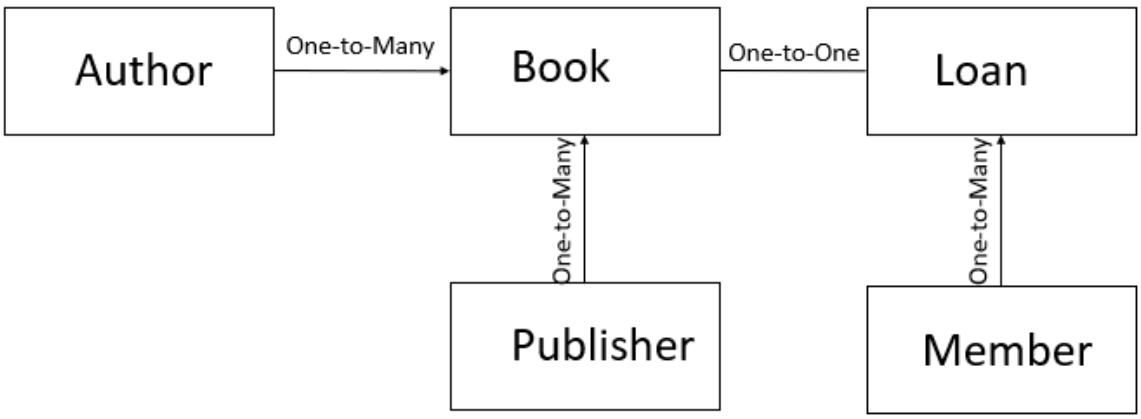
3. Would you like to access the system remotely / from home?

Being that the new system is a web-based, it should be decided whether the library members would like the chance to access the system remotely. If the functionality to search for available books in the library existed, would they use it? Would they like to see how long they can rent a particular book for? These are all micro questions that this interview is designed to answer without asking them directly.

4. What new features would you like the system to contain?

For similar reasons to the previous interview, not only must we obtain current weaknesses, but functionality that could improve the sustainability of the entire system. This question was designed to extract new system functions through the point of the view of remote users that have different experiences with the software.

Part B



ERD Explanations

One Author can write many books; however all books only have one Author.
One publisher can publish many books, but all books can only have one publisher.

One book can only have one loan at a time, and there is one loan record for each book. One member can have many loans, but each loan belongs to only one member.

Requirement: B1 List all the books in the system	Users: Library Staff, Library Members
Functional Requirement: List all the books available in the system in alphabetical order **Some descriptions of the functional requirement.** The system must include a function to list all the available books in the system. The list of books must be in alphabetical order and must contain information included in its database table: BookID, Title, PageLength, Rating, PublisherID, AuthorID. As well as the books unique information, the system must display the books availability and whether it can be loaned.	
Non-functional Requirement: Response Time (2 Seconds) **Text for non-functional requirement.** This particular function requires a large amount of data to be accessed and displayed. For this reason, it must be ensured that the function demonstrates a minimum of 2 second response time to keep the system running as efficiently as possible.	

Requirement: B2 Add a book to the system	Users: Library Staff
Functional Requirement: This function must allow library staff to add a book and its encompassing information to the system. **Some descriptions of the functional requirement.** It should be noted that library staff are the only personnel that able to access functions that modify the system in anyway so as to keep the system as secure as possible. This particular function should add an individual book, or a group of books to the particular area in the system to which it belongs (as well as the books details).	
Non-functional Requirement: Scalable (5TB of reserved database memory) **Text for non-functional requirement.** Being that books are constantly added to the system, we want to make sure that the system's memory can handle a potential overload of additions. For this reason the function must ensure that there is always 5TB of reservable memory in case the system happens to run out of space for books to be added.	

Requirement: B3 Edit a book in the system	Users: Library Staff
Functional Requirement: Edit a books origin information using a series of inputs provided by the library staff. **Some descriptions of the functional requirement.** Similar to the previous function, accessibility should be secure when modifying the system. This function should prompt the user to choose which fields of the book need modifying, followed by entering the specific information of which to exchange the original data for. This should ideally be completed via a series of input text boxes.	

Non-functional Requirement:
Availability (08:00 – 18:00, Monday – Friday)
Text for non-functional requirement.
This function should be accessible during the opening hours of the library establishment, these hours include 08:00 – 18:00, during the week. This is to ensure that any immediate changes that need to be made, can be done as quickly as possible.

Requirement: B4 Delete a book from the system	Users: Library Staff
Functional Requirement: When a library member indicates a book to be deleted, the book should be deleted from the system. **Some descriptions of the functional requirement.** Upon compilation, this function should prompt the user for their input. This input should pertain to a book that is currently stored in the system. Once the input is obtained, the system should search for the book and remove it from the system entirely. Optimally this function would contain a warning message that checks for validity as this action causes information being permanently deleted.	
Non-functional Requirement: Error handling in case of missing record errors **Text for non-functional requirement.** In case books that are not contained in the system are provided to the function, a try – except block should be called to ensure that the system does not fail to compile and return an error message.	

Requirement: B5 Search for a book in the system	Users: Library Staff, Library members
Functional Requirement: After prompting the user for a book title, the system should search for that particular book or most similar matches in the system. **Some descriptions of the functional requirement.** Similar to previous functions that require books to be searched for in the system, a text prompt with the ability to input book information should be provided to the user. This book should be searched for using exact matches or similar cases which would overcome any spelling or syntax errors in the user's input. Books must be displayed alphabetically by default in all cases.	
Non-functional Requirement: Search Time – Optimally O(log(N)), worst case O(1) **Text for non-functional requirement.** Being that we are dealing with such a large system database with such a large selection of books, it needs to be ensured that all records are being obtained in a minimum of O(1) time complexity. However, if we are using SQL to access our database, we can assume that the optimal time for queries to be return is in O(log(N)) time.	

The requirements for members module are similar to the book module as the functionality is seemingly identical (List, Add, Edit, Delete, Search)

Requirement: M1 List all the members in the system	Users: Library Staff, Library Members
Functional Requirement: List all the members in the system alphabetically **Some descriptions of the functional requirement.** Similarly to B1, this function should quickly list all of the members linked with the library in alphabetical order. The information displayed should be public information and nothing sensitive to the average member (phone number or address). Both first and last name should be displayed with the ability to scroll through the system as required.	
Non-functional Requirement: Response Time (1 seconds) **Text for non-functional requirement.** Although this particular function requires a large amount of data to be accessed and displayed, it is not as much information as the function B1 requires therefore the response time should be lower. The function should demonstrate a maximum of 1 second response time so as to keep the system running as efficiently as possible.	

Requirement: M2 Add a member to the system	Users: Library Staff
Functional Requirement: Add a new member that wishes to join the library to the correct index in the system. **Some descriptions of the functional requirement.** Using a series of input statements provided to the staff who are accessing the function, the members personal information should be used as the entry. This information should be securely stored in the correct alphabetical index in the system ensuring that later access is possible by M5.	
Non-functional Requirement: Scalable (5TB of reserved database memory) **Text for non-functional requirement.** Being that members are constantly added to the system, we want to make sure that the system's memory can handle a potential overload of additions. For this reason the function must ensure that there is always 5TB of reservable memory in case the system happens to run out of space for members to be added.	

Requirement: M3 Edit a member in the system	Users: Library Staff
Functional Requirement: Search for a member in the system and then choose which piece of information to access and edit using a series of inputs. **Some descriptions of the functional requirement.** Once the function is called, the staff member must stipulate which member needs their information changing. Once the member has been selected, it is up to the staff member to further select the piece of personal information that needs changing. All changes are made using inputs provided to the user in the form of text boxes.	

Non-functional Requirement:
Availability (08:00 – 18:00, Monday – Friday)
Text for non-functional requirement.
Due to the sensitivity of public information and the security of the user, it is vital that all information about the members is kept correct at all times. This is done by keeping the function active and available during the opening times of the library (between 08:00 and 18:00 on weekdays)

Requirement: M4 Delete a member from the system	Users: Library Staff

Functional Requirement:
Search for a member in the system and then delete them (after a confirmation warning) from the system.
Some descriptions of the functional requirement.
Similarly to previous functions, M4 should start by calling M5 and searching for a particular member in the system. Once found, the option should be provided to the user to delete the member from the system. It should be noted that a warning and confirmation message is vital so as to prevent "miss clicks" or mistaken button pressing.

Non-functional Requirement:
Error handling in case of missing record errors
Text for non-functional requirement.
In the case of members that are not contained or have already been deleted from the system being provided to the function, a try – except block should be called to ensure that the system does not fail to compile and return an error message.

Requirement: M5 Search members in the system by name	Users: Library Staff

Functional Requirement:
Using the first and second name provided by the library staff members, the system should search the database using a similar search algorithm to B5 and display information to the user.
Some descriptions of the functional requirement.
After being prompted with a series of input textboxes, the user should start by entering the required members first name and second name. Using an identical or "similar to" search method (combat spelling errors) the system should find the member record in the database and display it to the user with the option to reveal personal information as required.

Non-functional Requirement:
Search Time – Optimally O(log(N)), worst case O(1)
Text for non-functional requirement.
Being that we are dealing with such a large system database with such a large selection of members (with meta information included), it needs to be ensured that all records are being obtained in a minimum of O(1) time complexity. However, if we are using SQL to access our database, we can assume that the optimal time for queries to be return is in O(log(N)) time.

Requirement: L1 List all the loans in the system	Users: Library Staff
Functional Requirement: All current loans displayed to the user including the meta information provided by its database table. **Some descriptions of the functional requirement.** Upon compilation, the system should display all the information pertaining to all current library loans including all of the meta information provided by its database table. This information includes: The loan ID, the book that has been loaned, the member that took the loan, the loan amount, and the loan length.	
Non-functional Requirement: Response Time (2 Seconds) **Text for non-functional requirement.** Similarly to previous listing functions, L1 requires a lot of memory / computational execution time due to the shear number of records that need to be displayed to the user. For this reason a response time threshold should be kept ensuring that information returned from the function is displayed in a reasonable time.	

Requirement: L2 Check out a book	Users: Library Staff, Library members
Functional Requirement: The status of a chosen book should be changed from available to unavailable and a loan period assigned. **Some descriptions of the functional requirement.** Members and library staff members should have the ability to loan any book of their choosing. For this reason, L2 is developed to select a book of the users choosing and change the status of the book from available to unavailable (Because it has now been loaned). The loan should be assigned with the appropriate time period and all the relevant system information updated as well.	

Non-functional Requirement:
Simplistic learnability and an average 3-minute interface completion time
Text for non-functional requirement.
Being that this function is available to both normal customers and library members, it is crucial that the interface used to loan books is as simplistic as possible. The system should boast a maximum of 3-minutes completion time from the opening screen, to the completion of a loan form. This could be done by reimplementing functionality from the previous system that the users are already familiar with.

Requirement: L3 Extend a loan for a book	Users: Library Staff

Functional Requirement:
Extend the length of time that a book can be loaned to a particular library member.
Some descriptions of the functional requirement.
This function is responsible for extending the amount of time a book has been loaned to a member. Optimally the user should be able to select the amount of time that they would like the extension to be however to ensure fairness in the library loan system, the staff members are responsible for changing this option. It should be noted that this option should not be available if there is a queue waiting to acquire the book.

Non-functional Requirement:
100% Reliability at all times.
Text for non-functional requirement.
This function should demonstrate 100% reliability at all times. If the function fails to compile or fails to update at any time when pertaining to the loan period, it can cause the false belief that a loan has not been returned in the adequate time frame. This in turn can lead to serious repercussions or potential persecution of the library members.

Requirement: L4 Search for a loan in the system by member id or book id	Users: Library Staff

Functional Requirement:
Using provided member or book ID's search for a loan in the system and display the relevant information to the user.
Some descriptions of the functional requirement.
Once the relevant ID for either a member or a book has been provided to the system, the system should search for the relevant Loan record in O(1) time complexity. All the information pertaining to that loan record including (The book loaned, the member who took the loan, the length of the loan, the return date) should be displayed on screen. It should also be noted that this function is only available to staff members to ensure there is no conflict between members wanting similar books.

Non-functional Requirement:
Search Time – Optimally O(log(N)), worst case O(1)
Text for non-functional requirement.
Similarly to previous functions that require any database wide search, the system must find the relevant record in a minimum of O(1) time complexity and a maximum of O(log(N)) time complexity. This ensures that the system runs as smoothly as possible especially as the size of the database consistently continues to grow due to exponential growth of members taking out loans.

Requirement: L5 Mark as loan returned	Users: Library Staff
Functional Requirement: The variable pertaining to a loans current status should be changed to "returned" once the relevant book has been returned to the library. **Some descriptions of the functional requirement.** When a book that has been loaned has been returned to the library and registered on the system, the Boolean variable that has been set to False (BookReturned = False), should be updated to True and the book should be displayed as available on the system for other members to borrow if they would like to.	
Non-functional Requirement: 100% Reliability at all times **Text for non-functional requirement.** Similarly to L3, the function should demonstrate a 100% reliability rate. In fact, any functionality that's error could cause a user to be wrongly incriminated should exercise 100% reliability. The failure of a loan's return status being updated would lead the system and the staff members to believe that a book has failed to be returned in its loan period, thus causing repercussions for the user who has correctly returned the book already.	

Part C

* - Indicates that the field represents the tables primary key

** - Indicates that the field represents the tables foreign key

All tables must have a primary key but may not necessarily have a foreign key as well.

Book Table

Fields: int BookID*, VARCHAR Title, int PageLength, int Rating, int PublisherID**, int AuthorID**

Each book has an encompassing author and publisher making it appropriate to designate two foreign keys that can link the tables.

Author Table

Fields: int AuthorID*, VARCHAR FirstName, VARCHAR SecondName

Publisher Table

Fields: PublisherID*, VARCHAR PublisherName, VARCHAR Country

Member Table

Fields: MemberID*, VARCHAR FirstName, VARCHAR SecondName, VARCHAR MembershipType, ContactNumber, VARCHAR LoanedBooks

Loan Table

Fields: int LoanID*,int BookID**, int MemberID**, int LoanAmount, Date LoanLength

For every loan, there must be a book and a member that has triggered the loan record to be created, therefore I have included foreign keys linking to both the member table, and the book table.

Part D

Table for Time elapsed for all team members on all related tasks that could be assigned to them:

The results for this table were calculated using the following formulae:

$$\text{Elapsed time} = \text{Effort time} \times \frac{100}{Availability\,\%} \times \frac{100}{Work\ rate\,\%}$$

* Elapsed time has been kept as days in the table

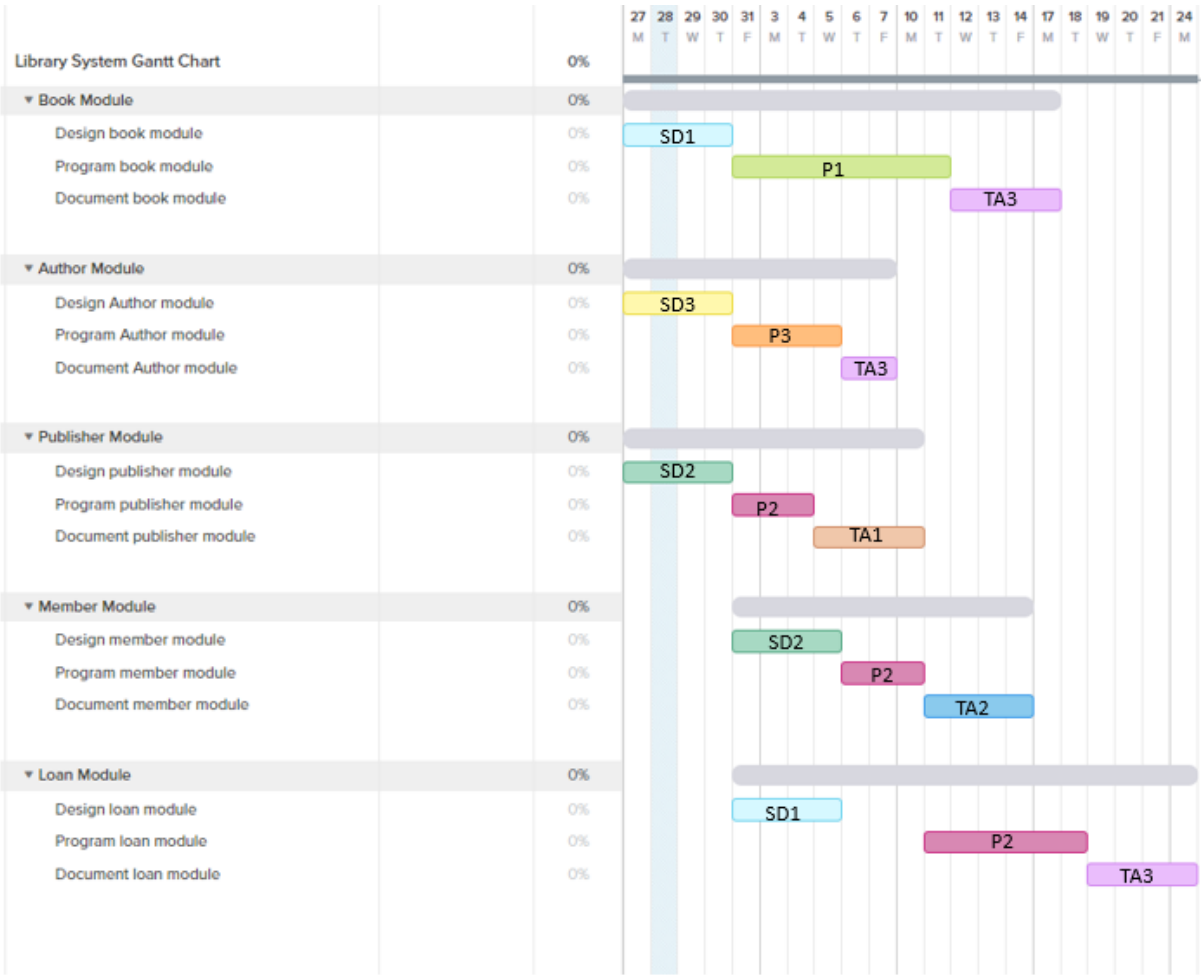
Tasks	SD1	SD2	SD3
Design book module	4	6.67	16
Design author module	1	1.67	4
Design publisher module	2	3.33	8
Design member module	2	3.33	8
Design loan module	4	6.67	16
	P1	P2	P3
Program book module	7.5	6	20
Program author module	1.25	1	3.33
Program publisher module	3.75	3	10
Program member module	3.75	3	10
Program loan module	7.5	6	20
	TA1	TA2	TA3
Document book module	5	6	3.75
Document author module	1.67	2	1.25
Document publisher module	3.33	4	2.5
Document member module	3.33	4	2.5
Document loan module	5	6	3.75

Gantt Chart

Below is a color coded Gantt chart representing the list of job undertaken per person and the timeline to the projects completion.

It should be noted that to represent a full and accurate working week, weekends have been removed from the timeline so as to keep the schedule as realistic as possible.

Furthermore tasks that have been calculated to a decimal time period have been rounded up. Being that it is inefficient to split days internally, the workers are given an entire day to complete their task. For example, a task that takes 3.33 days, is rounded to 4 complete days.



Book Module

Design – SD1: 4 days
Program – P1: 8 days
Documentation – TA3: 4 days
Completion time – 16 working days

Author Module

Design – SD3: 4 days
Program – P3: 4 days
Documentation – TA3: 2 days
Completion time – 10 working days

Publisher Module

Design – SD2: 4 days
Program – P2: 3 days
Documentation – TA1: 4 days
Completion time – 11 working days

Member Module

Design – SD2: 4 days
Program – P2: 3 days
Documentation – TA2: 4 days
No workers available time – 4 days
Completion time – 15 working days

Loan Module

Design – SD1: 4 days
Program – P2: 6 days
Documentation – TA3: 4 days
No workers available time – 7 days
Completion time – 21 working days