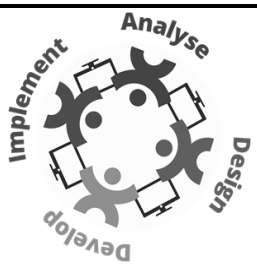
INF272: Homework Assignment 03



Themes: Programmatic and creative problem solving, reasoning and logic.
Due date: DUE: 03-11-2024 before 23:59

Notes & Instructions [65 Marks]

[+ 10 bonus points**]

** There are 10 bonus points allocated to this assignment. Please refer to the rubric at the end of the assignment.

PURPOSE AND OBJECTIVES.

The purpose and objective of this assignment is:

- **Objective 1:** Integrated programming.
- **Objective 2**: Programmatic problem-solving.
- **Objective 3:** Programmatic reasoning.
- **Objective 4:** Programmatic logic.
- **Objective 5:** Creative problem-solving.
- **Objective 6:** RAPID APPLICATION DEVELOPMENT. [1]

Read EVERYTHING.

This is a functionality review.
Do not skip sections or details.

MATERIAL BEING REFERENCED.

Students are only allowed to use technology taught in INF272 up to **Session 24 - Generating reports from processed data in MVC with C#, add-ins and modules**. Using any technology outside of these sessions will cost you marks.

READING OR RESEARCH THAT WILL BE NECESSARY.

Students will have to conduct additional independent research and reading on the topics listed below. The topics were discussed and covered in the Monday preparation theory sessions. Students who participated in the sessions will have a broad overview of the topics listed, and, as such, have a distinct advantage. The topics required is as follows:

- **Topic 1:** Recording your video demonstration and then compressing the demo for upload (self-study).
- **Topic 2:** MS SQL Server database.
- **Topic 3:** Bootstrap documentation and the application thereof.
- **Topic 4:** Bootstrap modal popups.
- **Topic 5:** Page merging.
- **Topic 6:** Asynchronous processing as extended with EF.
- **Topic 7:** jQuery and DOM manipulation with jQuery.
- **Topic 8:** LINQ/LAMBDA.
- **Topic 9:** The application of add-ons and third-party applications in MVC.
- **Topic 10:** Everything taught in INF272 that could be used in the assignment (session 1 up to session 24).
- **Topic 11:** **GitHub (no support will be provided with regards to this topic).** [2]

There shall be no assistance with regards to the completion of this assignment.

[1] "*Rapid application development is an agile software development approach that focuses more on ongoing software projects and user feedback and less on following a strict plan. As such, it emphasizes rapid prototyping over costly planning*". https://www.outsystems.com/glossary/what-is-rapid-application-development/

[2] Refer to the notes and details on marked as VERY IMPORTANT [NB].

NOTES ON TOPIC 11: GITHUB.

As discussed in one of the Monday preparation sessions, if, and only if a student makes use of GitHub as an assignment submission tool, then a student will obtain an additional 2% to their year average. This 2% will form part of the average calculated before the commencement of the exam. The exam mark is not subject to this bonus. If this submission option works, then the averages will be calculated as follows:

Final year mark contributions

Module Average [as calculated at the end of the year]	
• Practical activities and tasks	30%
• Projects and homework contribution	30%
• Semester Test 1 (S1) and Semester Test 2 (S2) contribution	40%
Final year average [40% required to write exam]	**100% + 2%**

Final year mark contributions and notes

Final Module Average [as calculated at the end of the exam]	
• Year mark	50%
• Exam mark	50%
Final module average [need 50% to pass module]	**100%**

Getting Started with GitHub:

S1L02 Session 02 - Getting started with Visual Studio and GitHub

Use git fetch, pull, push and sync for version control in Visual Studio:

https://learn.microsoft.com/en-us/visualstudio/version-control/git-fetch-pull-sync?view=vs-2022

Inviting collaborators to a personal repository:

https://docs.github.com/en/account-and-profile/setting-up-and-managing-your-personal-account-on-github/managing-access-to-your-personal-repositories/inviting-collaborators-to-a-personal-repository

If, and only if, the files and details are submitted through GitHub perfectly correctly, as defined by the details posted above, then and only then, will a student receive the bonus applicable to the pre-exam year module average.

GENERAL INSTRUCTIONS.

- This is an individual assignment.
- The assignment is based on assessment objectives and requirements.
- If an objective has been achieved a mark will be allocated. Please refer to the rubric.

SUBMISSION DUE: 03-11-2024 before 23:59.

- There shall be no extensions to the deadline.
- All work will be marked offline by the lecturers and the assistant lecturers.
- Verify the completeness of uploaded files.
- Incomplete uploads will be considered unsubmitted work.
- If homework submissions are uploaded too late then upload errors WILL happen.
- Do not wait to the last minute to complete the assignment.
- There are multiple upload opportunities enabled to ensure that uploads are complete.
- Email submissions **WILL NOT** be accepted.
- Late submissions **WILL NOT** be accepted.
- **NO EXCEPTIONS WILL BE MADE FOR ANYONE.**
- Do not request any form of an exception by email.

- **Emails requesting an exception will not receive a reply**.
- **No special concessions will be made regardless of the situation**.
- There shall be no "negotiation" with regards to the mark allocation regarding missed or incomplete submissions.

SUBMISSION INSTRUCTIONS.

- In this assignment, students are provided with high-level requirements and these requirements can implemented in a context as seen fit. Requirements are not optional. Requirements are mandatory.
- Make sure to upload the entire project. A student should ensure that they select, zip and then upload the complete project. This was clearly discussed in one of the preparation session in the Monday theory class in semester 1. If only the *.sln is uploaded, then it will be considered and unsubmitted application and a zero will be awarded.
- **Source Code:** Zip source code files together and name it **uXXXXXXXX_HW03.zip**, where the XXXXXXXX is a student's UP student number, e.g., u12345678_HW03.zip. If the zipped project is less than 100 Mb, then you can upload the zipped file on ClickUP. The upload will be slow and will appear as though the upload is not working. An upload directly to ClickUP is time consuming, and a student should already be aware of this fact. If there is a time-out during the ClickUP upload, and the date and time is after that of the cut-off time, then the upload link will have closed and, as such, the source code would not been loaded. If the source code was loaded correctly, ClickUP should send a submission receipt to a student's email address. If a student chooses to make use of the **GitHub submission** option, then **the project should not be zipped**. A student will have to make use of the push or sync option. In the case of a GitHub submission, a student SHOULD share their GitHub repository with the **inf272marker@gmail.com** as a **COLLABORATOR**. If a student submits work by means of a **Google Drive link**, a student should make sure to zip the files and share the drive link as "**ANYONE WITH THE LINK". If a student is uncertain if one of the options might be problematic, use all three options indicated. If the source files are missing or inaccessible, a zero will be awarded.**
- **Video Demo: DO NOT** Zip your Video Demo. Name the video demo **uXXXXXXXX_HW03.mp4**, where the XXXXXXXX is a student's UP student number, e.g., u12345678_HW03.mp4. If a student submits work by means of a **Google Drive link**, make sure NOT TO zip the files and share the drive link as "**ANYONE WITH THE LINK". If a student chooses to make use of a YouTube link submission, set the demo as UNLISTED and not as Private. If the demo is missing or inaccessible, a zero will be awarded.**
- **Make sure that all uploads are completed at least 3 hours before the final cut-of time. Budget time to include:**
    1) **coding of the application,**
    2) **the recording of the demonstration,**
    3) **the compression of the demonstration,**
    4) **reviewing the completeness of the files that will be uploaded and then,**
    5) **the subsequent upload,**
    6) **the verification of the correctness and completeness of the uploads and then finally,**
    7) **correcting any upload or submission errors.**

NOTES ON DEMO SOFTWARE.

- Use desktop recording software as suggested by the assignment. Change the compression ratio and Frames Per Second (FPS) of the desktop recording software that was chosen so that the file size of the demo can be reduced.
- **DO NOT use a phone to record the demo**. Phone recordings creates unnecessarily large files making the video demo upload problematic. ClickUP will reject the upload of a file that is larger than 100Mb. Make sure that files do not reach the 100Mb file size limit. Desktop recording software streamlines the process and creates smaller files that are easier to upload. Phone recordings create files larger than the upload limit, and, as such, will be rejected by ClickUP.
- Students can make the video demo with OBS Studio [https://obsproject.com/], ScreenPal (Formerly Screencast-O-Matic) [https://screenpal.com/] or any other reasonable desktop recording software.
- Please make sure to compress the demo as much as possible. A student can use free software called Handbrake [https://handbrake.fr/] for enhanced compression.

VIDEO DEMO INSTRUCTIONS.

- Make sure that everything is running when the demonstration video recording starts. If the project crashes during the demo, pause the recording and then continue with recording when the project is running again. The video should not be longer than 10 minutes showing the items in the Checklist. When the 10 minute limit is reached, there shall be no further marking. The maximum limit is 10 minutes and no more. The demonstration may be less than 10 minutes as long as all the required details are presented.
- Demonstrate the behaviour and functionality of the program by running the program. Complete the transactions necessary to demonstrate the functionality while the program runs.
- In this project, we are not interested in code. The majority of the code is prefabricated scaffolded code. **We are only interested in the behaviour of the application. Do not spend any time on explaining the code.**
- A student should make use of the given demonstration time to present as many full and complete transactions for every part of this project.
- If the program does not run, or crashes during the demo, do not waste time by apologising. Run the program again after there is a crash, and then continue with the demo in another section of the application that does not crash.
- **Demonstrate sections that work! This is a functionality evaluation and not a code evaluation.**
- Refer to the rubric at the end of this document. Sequence discussions according to the rubric. Consider the rubric as the script for the demo.
- **Follow the sequence of the rubric with regards to the demonstration. Clearly state the section of the rubric being presented, demonstrate the necessary functionality, and then move on to the next section as defined by the rubric. The rubric is the clear demonstration sequence. There is no need to explicitly discuss the aesthetics and appearance of the project, as this will be clearly visible in the demonstration of the application's functionality. With regards to creativity in the application itself, explain what was creative. Aesthetics (how good the application looks), does not need to be pointed out. If a student doesn't state and demonstrate a section as defined by the rubric, then it will be assumed that the section is dysfunctional.**
- In this assignment, students should STILL upload the application code as well as the demonstration. If one of the other is missing, then the submission will not be marked and a student shall receive a zero. BOTH the Video Demo and Homework Source Code should be submitted correctly so as to be evaluated and receive a mark.
- **If the Video Demo or the Homework Source Code is missing, a student will get a zero.**
- If files or links are uploaded to the wrong upload area, we will not go and look for the upload. Uploads should be submitted correctly as indicated. Incorrect uploads will lead to a zero being allocated. There shall be no "negotiation" or discussion about the allocation of a zero for incomplete uploads. If a student sends an email regarding such an instance, then the email shall not receive a reply.
- If a student is caught plagiarising or sharing applications or application code, we will give the said student a 0 and will subsequently be reported for plagiarism or academic dishonesty.

NOTE ON MARKING.

- If a student is found to demonstrate another students application, then BOTH students will receive a zero.
- If one student assists another student with regards to their assignment, make sure not to share code but rather explain an approach, a thinking process and help other students find potential problems and how to approach solving these problems.
- Do not try to complete this program in "one sitting". Do not start late. Do not upload late.
- There is a bonus mark allocated to this assignment. The normal mark allocation is out of 65. If a student successfully complete the bonus mark section, then a student may achieve 75 / 65.

VERY IMPORTANT [NB] ← READ THIS.

This is a Rapid Application Development assignment. A student will need to use all of the skills learnt throughout the year to complete this application as a time challenge. This aspect of the module has always been part of the plan for INF272.

- If a student was diligent during the year, then this assignment should take a student a couple of hours to complete.
- If a student wasn't diligent during the year (this was a student's free choice), then this assignment should take a student a couple of days to complete.
- If a student copied from other students during the year, then the time factor, scope and size of this assignment will make it very difficult to find a functional copy that such as student can use.

The complexity with regards to this application is not that high, and it makes use of a lot of prefabricated and standardised code, packages and existing add-in bundles. A student would still need to understand the aforementioned even if it is prefabricated, otherwise it is extremely easy to make unnecessary mistakes. Examples and portions of the technology applied in this assignment will be found distributed in between examples and practical work completed during the year.

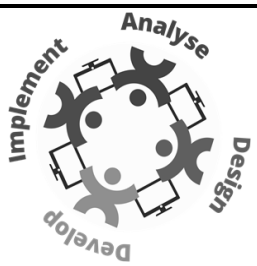
DO NOT leave the assignment to the day of the due date. Students already know that they will regret this choice. DO NOT make the mistake of leaving the assignment to the day of the due date. The complexity is not that high, but the assignment still needs the appropriate amount of time to complete.

This is a FUNCTIONALITY ASSESSMENT and NOT A CODE ASSESSMENT.

In INF370, mostly functionality is reviewed. The INF370 lecturers and assistants may on occasion review small snippets of code. In this last assignment, we will, as such, only **conduct a functionality review**. Students will STILL have to upload their code. We will download code to check on what a student is demonstrating. If a student's code has not been loaded, such as student WILL receive a zero. If a student uploads an empty project, then such as student will receive a zero.

There will be no negotiation regarding terms and conditions listed in the aforementioned sections of this assignment. Emails pertaining to additional time, upload errors, incomplete uploads, the allocation of a zero due to assignment upload details missing etc, SHALL not receive an email reply.

INF272: Homework Assignment 03



Themes: Programmatic and creative problem solving, reasoning and logic.
Due date: DUE: 03-11-2024 before 23:59

Case Study - [65 Marks]

[+ 10 bonus points**]

** There are 10 bonus points allocated to this assignment. Please refer to the rubric at the end of the assignment.

1. HIGH-LEVEL CASE DESCRIPTION.

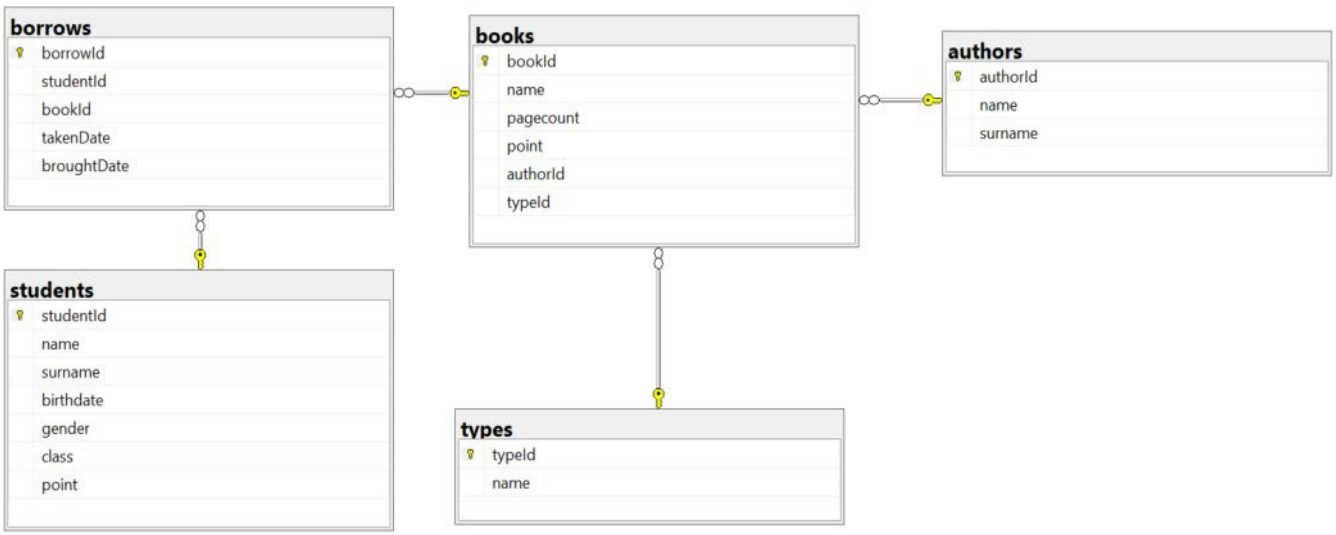
The project involves developing a library management system to streamline the handling of students, book catalogues, genres, and authors. Key features include a borrowing mechanism, book status viewer, and a reporting module for insights into the library. The system should feature a document archive for report storage and retrieval and a user-friendly interface throughout. This system aims to enhance user experience, simplify administrative tasks, and promote a culture of knowledge-sharing and learning within the library environment.

2. STANDARD REQUIREMENTS:

The following is a list of mandatory requirements. The requirements are not optional.

- **Do not create a .NET Core project. It will not be marked.**
- **Ensure that you update the Bootstrap and jQuery NuGet packages that come with the initial project to the latest versions.**
- You must follow the guidance of the wireframes provided to complete the assignment.
- Your assignment does not have to look exactly like the wireframes. They serve mainly as guidance.
- You are allowed to choose any approach to complete the assignment. You must however stay within the technology parameters specified and taught in INF272. Exceeding these parameters will cost you marks.
- Marks will be allocated for the creativity and neatness of your assignment. Not displaying any original thinking in the layout and design of the pages will cost you marks.
- Do not add any functionalities that are not specified.
- **You must make use of Async methods for all the CRUD operations in this assignment.**
    - **You can also use async and await in Entity Framework with Asynchronous Scaffolding.**
    - **Do not make use of AJAX for this assignment.**
- Make sure that System.Threading.Tasks; is included when making use of async methods in the controller.
- You must make use of the SQL database provided for this assignment (Library.sql).

Here is the database diagram for this assignment:



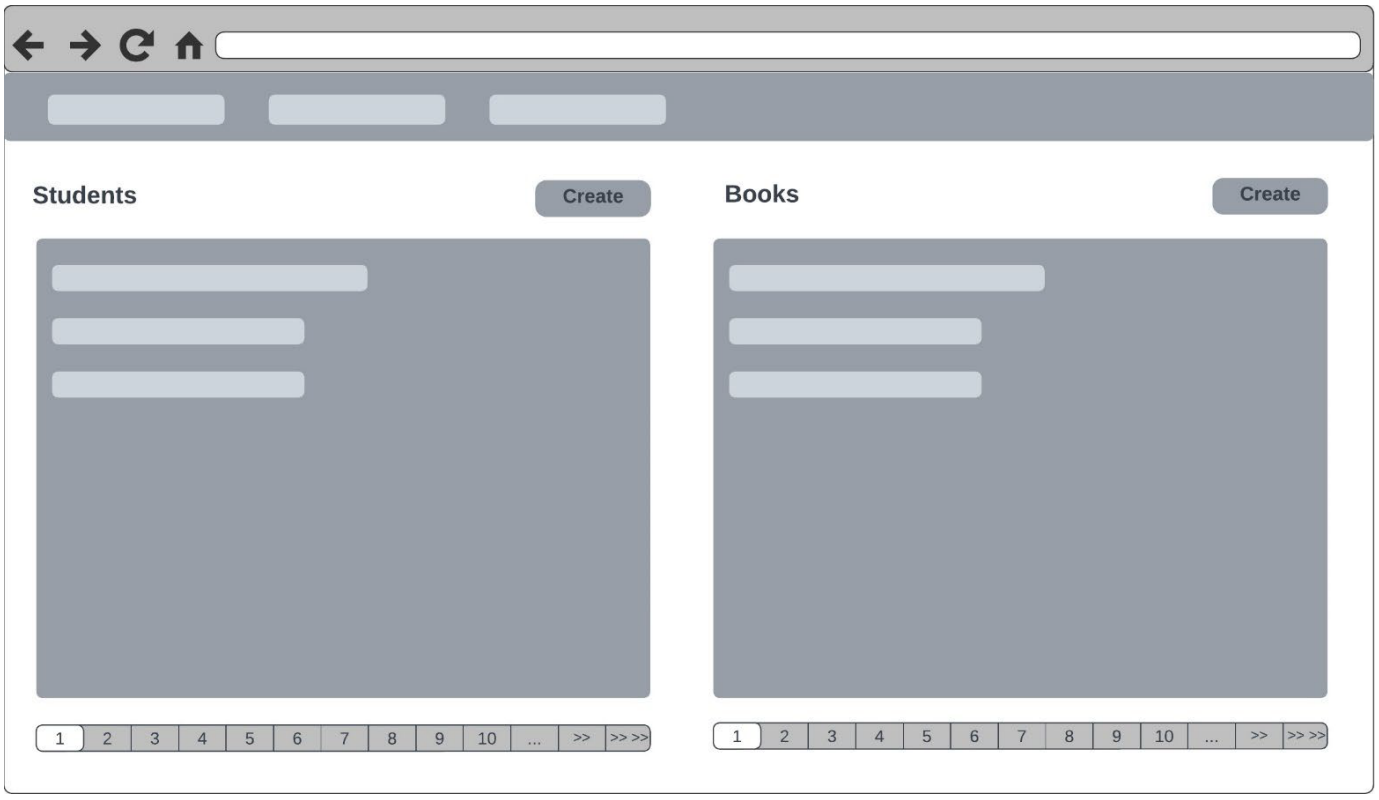
The following is the list of requirements that have been provided to you. They have been broken down per screen:

2.1. Navigation Bar

The requirements for the navigation bar are as follows:

- Each screen must contain a navigation bar.
- The design and function of the navigation bar is up to you.

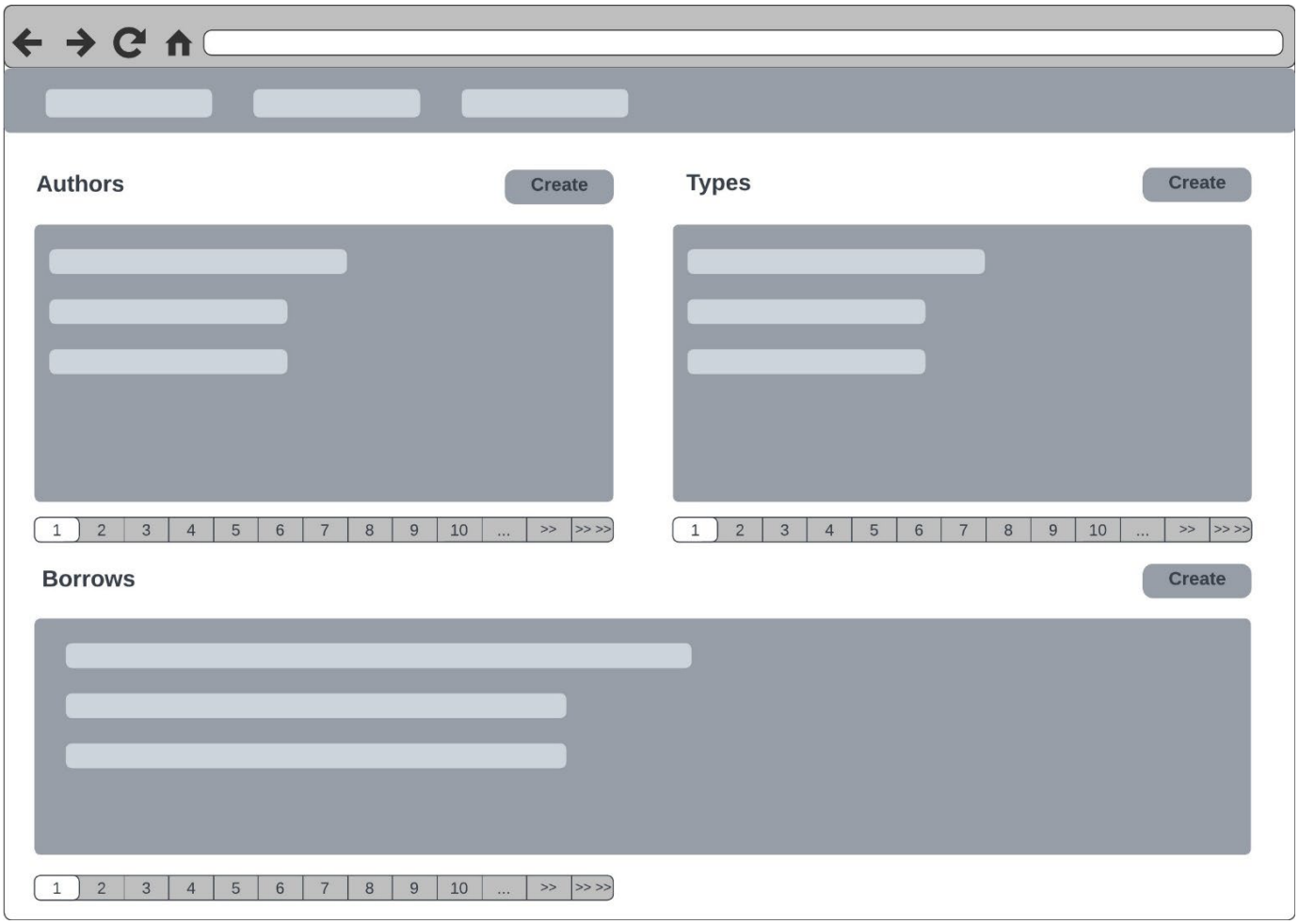
2.2. Home



The requirements for the Home page are as follows:

- This screen must serve as the home page for the application.
- This screen must make use of page merging. A user should see a list of **Students** and **Books**. Display all details of each student and book. **Do not display foreign keys but the content they reference**.
- Add a **Status** column for the book list. This column should show whether a book is currently 'Out' or 'Available' (reference the attributes in the **Borrows** table for this).
- A user must be able to **Create** new students and books (a user should not be able to update and delete on this screen).
- When a user clicks on the create button an appropriate **modal popup** must be displayed and used for the creation.
- Both sections must make use of its own **page list**.
- **Reminder:** You must make use of Async methods for all the CRUD operations in this assignment.
- See the wireframe above for some inspiration as to the layout and design.

2.3. Maintain



The requirements for the Maintain screen are as follows:

- A list of all the **Types, Authors, and Borrows** must be displayed. Display all details of the **Borrow** table. **Do not display foreign keys but the content they reference**.
- A user must be able to **Update, Delete, and Create** the **Types, Authors, or Borrows**.
- When a user **Updates, Deletes, or Creates** any of the records an appropriate **modal popup** must be displayed and used.
- All three sections must make use of its own **page list**.
- **Reminder:** You must make use of Async methods for all the CRUD operations in this assignment.
- See the wireframe above for some inspiration as to the layout and design.

Please note that we are not concerned about Delete Cascade. If a Cascaded Delete occurs, that is of no concern. Visual Studio 2022 Community Edition's Model Builder manages Delete SetNull poorly, and we did not show you how to use a method for the prevention of Delete Cascade. This default Model Builder Action is perfectly acceptable.

2.4. Report

The requirements for the Report page are as follows:

- This screen must allow a user to generate a report.
- The design of the report will be up to you. The report must make sense from a business standpoint (i.e., it must be useful). The report must contain a **graphical component**. A simple data dump (displaying only a single table) will not be viewed or considered a report. A report has business functionality and meaning behind it. Raw unprocessed data is not considered a report.
- Some ideas of what you could potentially create are:
    - **Current Loans Report**: List all books currently checked out and the student (s) who borrowed the book (s). This report helps track which books are out and who has them.
    - **Overdue Books Report**: Identify records where the borrow date is past due (less than the current date) and the book has not been returned. This report is crucial for managing overdue fines and sending reminders to students.
    - **Borrowing History Report**: Generate a report where one can see a student, all of the student's book borrow entries, the books borrowed and the dates when books were borrowed and returned (or not returned). This report is a student borrow history report which should present a profile of a student's borrow and book return behaviour.
    - **Popular Books Report**: Count the number of times each book appears in the Borrows table over a certain period, which helps in identifying the most frequently borrowed books. This report should assist in identifying books for which additional copies should be obtained and which books may require more frequent replacement due to potential damage.

- **Borrowing Frequency Report**: Report on how many books are being borrowed monthly/weekly/daily by all students or a specific student. In other words, the report can have a student or students dimension to it. This report helps in understanding peak times and seasons in the library.
- **Duration Analysis Report**: Analyse the average time books are borrowed by calculating the difference between the date the book was borrowed and the date the book was returned (or not returned yet). This report can assist in understanding the average reading times and potentially adjusting borrowing periods.
- **Unreturned Books Report**: List all records where the book has not been returned (is NULL), regardless of the due date. Make sure that the unreturned book list shows who may still have the book. This will assist in finding books and understanding where a book might currently be located.
- **Student Borrowing Ranking**: Rank students by the number of books borrowed within a certain timeframe. This could be used for recognizing the most avid readers or identifying students who might need additional resources.

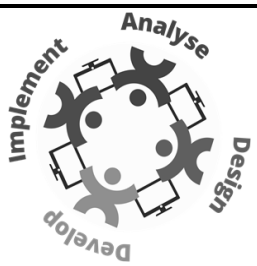
- The user must be able to save the entire report generated by entering a **Filename** and selecting the **File type** they would like to save the report as (refer to Session 10 - The application of on add-ons and third-party applications in MVC for this).
- This page must also contain a **document archive** section that shows a **list** of all the reports that have been saved by the user. The user must be able to **download** and **delete** files from this document archive.
- See the wireframe above for some inspiration as to the layout and design.

2.5. Bonus marks

The requirements to receive bonus marks is as follows:

- On the report screen add the ability for the user to add a description to the specific file that has been saved in the document archive.
- You must make use of a resizable rich text box for this (refer to Session 10 - The application of on add-ons and third-party applications in MVC for this).

INF272: Homework Assignment 03



Themes: Programmatic and creative problem solving, reasoning and logic.
Due date: DUE: 03-11-2024 before 23:59

Rubric [65 Marks]

[+ 10 bonus points**]

** There are 10 bonus points allocated to this assignment. Please refer to the rubric at the end of the assignment.

Requirement - Checklist Use the checklist as a script that would allow you to sequence your demonstration.		MAX MINUTES	DYSFUNCTIONAL	MAX MARK
1.	**Requirement 1 – Functionality.**	**2**	-----	-----
	Run the program and then go through one complete action.	-----	**0**	**10**
2.	**Requirement 2 – Home.**	**1**	-----	-----
	Students requirement met with adequate design and creativity.	-----	**0**	**5**
	Books requirement met with adequate design and creativity.	-----	**0**	**5**
3.	**Requirement 3 – Maintain.**	**3**	-----	-----
	Authors requirement met with adequate design and creativity.	-----	**0**	**5**
	Types requirement met with adequate design and creativity.	-----	**0**	**5**
	Borrow requirement met with adequate design and creativity.	-----	**0**	**5**
4.	**Requirement 4 – Report.**	**2**	-----	-----
	Report requirements met with adequate design and creativity.	-----	**0**	**10**
	Archive requirements met with adequate design and creativity.	-----	**0**	**10**
5.	**Requirement 7 – Typical Bootstrap Design, Appearance, and Creativity.**	**1**	**0**	**10**
6.	**Requirement 6 – Bonus.**	**1**	-----	-----
	Bonus mark requirements met with adequate **ENHANCED design and creativity.**	-----	**0**	**10**
Total (Without Bonus).		**10**	**0**	**65**
Total (With Bonus).		**10**	**0**	**75**