Manchester Metropolitan University

Department of Computing and Mathematics

ASSESSMENT COVER SHEET 2023/24

Unit Code and Title:	6G6Z0035: Software Testing and Quality Assurance
Assessment Set By:	Lewis Evans
Assessment ID:	1CWK30
Assessment Weighting:	30%
Assessment Title:	Test Plan
Type:	Individual
Hand-In Deadline:	See Moodle
Hand-In Format and Mechanism:	Submission is online, via Moodle

Learning outcomes being assessed:

LO1 Evaluate the appropriateness of a selection of testing techniques to software systems and their components to design comprehensive testing plans.

LO2 Apply testing techniques at different stages of the software development process to ensure developed software is correct and robust.

Note: it is your responsibility to make sure that your work is complete and available for marking by the deadline. Make sure that you have followed the submission instructions carefully, and your work is submitted in the correct format, using the correct hand-in mechanism (e.g., Moodle upload). If submitting via Moodle, you are advised to check your work after upload, to make sure it has uploaded properly. If submitting via OneDrive, ensure that your tutors have access to the work. Do not alter your work after the deadline. You should make at least one full backup copy of your work.

Penalties for late submission

The timeliness of submissions is strictly monitored and enforced.

All coursework has a late submission window of 7 calendar days, but any work submitted within the late window will be capped at 40%, unless you have an agreed extension. Work submitted after the 7-day late window will be capped at zero unless you have an agreed extension. See 'Assessment Mitigation' below for further information on extensions.

Please note that individual tutors are unable to grant extensions to assessments.

Assessment Mitigation

If there is a valid reason why you are unable to submit your assessment by the deadline you may apply for assessment mitigation. There are two types of mitigation you can apply for via the unit area on Moodle (in the 'Assessments' block on the right-hand side of the page):

- **Self-certification**: does **not** require you to submit evidence. It allows you to add a short extension to a deadline. This is not available for event-based assessments such as in-class tests, presentations, interviews, etc. You can apply for this extension during the assessment weeks, and the request must be made **before** the submission deadline.

- **Evidenced extensions:** requires you to provide independent evidence of a situation which has impacted you. Allows you to apply for a longer extension and is available for event-based assessment such as in-class test, presentations, interviews, etc. For event-based assessments, the normal outcome is that the assessment will be deferred to the Summer resit period.

Further information about Assessment Mitigation is available on the dedicated Assessments page: https://www.mmu.ac.uk/student-life/course/assessments#ai-69991-0

Plagiarism

Plagiarism is the unacknowledged representation of another person's work, or use of their ideas, as one's own. Manchester Metropolitan University takes care to detect plagiarism, employs plagiarism detection software, and imposes severe penalties, as outlined in the Student Code of Conduct and Regulations for Undergraduate Programmes. Poor referencing or submitting the wrong assignment may still be treated as plagiarism. If in doubt, seek advice from your tutor.

As part of a plagiarism check, you may be asked to attend a meeting with the Unit Leader, or another member of the unit delivery team, where you will be asked to explain your work (e.g. explain the code in a programming assignment). If you are called to one of these meetings, it is very important that you attend.

If you are unable to upload your work to Moodle

If you have problems submitting your work through Moodle, you can email it to the Assessment Team's Contingency Submission Inbox using the email address submit@mmu.ac.uk. You should say in your email which unit the work is for, and provide the name of the Unit Leader. The Assessment team will then forward your work to the appropriate person. If you use this submission method, your work must be emailed **before the published deadline**, or it will be logged as a late submission. Alternatively, you can save your work into a single zip folder then upload the zip folder to your university OneDrive and submit a Word document to Moodle which includes a link to the folder. **It is your responsibility to make sure you share the OneDrive folder with the Unit Leader, or it will not be possible to mark your work.**

Assessment Regulations

For further information see Assessment Regulations for Undergraduate/Postgraduate Programmes of Study on the Student Life web pages.

Formative Feedback:	*Students can obtain formative feedback in the scheduled lab session in Week 4.*
Summative Feedback:	*You will receive written feedback on your work within 20 working days of submission, in the form of a commented assessment grid identical to the one found in Section 5 of this document.*

1. Assessment Introduction

The purpose of this assessment is to provide you with hands-on experience in planning and documenting a robust testing strategy for a software system. You will be required to develop a testing plan of approximately 1,500 words, outlining the various types of testing that the system will undergo at different stages of its development. The focus should be on selecting appropriate testing techniques to verify the system's correctness effectively.

The rest of this assessment brief is structured as follows:

- **Section 2 gives an overview of the scenario** you are being asked to immerse yourself in as part of this assessment.
- **Section 3 provides instructions that you should follow** in order to make a good attempt at this assessment.
- **Section 4 provides instructions** for submitting your assessment.
- **Section 5 contains the summative marking scheme**, which contains the criteria you will be assessed against

2. Assessment Scenario & Objective

This assessment is scenario-based where you will assume the role of a Junior Software Tester immersing yourself in the scenario below.

You are currently employed as a Junior Software Tester (JST) at InfinitiCode Solutions, a software development company that produces software of various sizes and complexity. A significant number of projects the company takes on are fairly small in terms of scale and are usually developed by junior software developers under the mentorship of more senior colleagues.

The company has recently launched a pilot scheme, whereby junior developers are permitted to take advantage of Generative Artificial Intelligence (GenAI) to aid in their software development activities with the hopes this will lead to junior developers requiring less mentorship and guidance from senior developers, allowing the senior developers to spend more time on the more complex projects.

Naturally, this pilot scheme has been met with some scepticism by some executives within the company, in that GenAI is still considered to be in its infancy with respect to being able to generate robust, secure, and scalable code. Genuine concerns have been raised that serious reputational damage such a scheme could cause the company if projects that are developed with such technology fail to meet the high standards set by previous projects.

It is therefore imperative that any project that has used GenAI as part of its development is thoroughly tested before it is shipped to the customer, regardless of the project's size and complexity. To this end, you have been the JST assigned to lead the testing of a system (further details provided in the next section) produced as part of this pilot scheme. Your first task [and the task this assessment is concerned with] is to produce a robust test plan by analysing the documentation you have been provided to ensure a robust testing strategy is adopted and practised when testing of the system commences.

3. Assessment Instructions

Your test plan needs to be written based on your inspection of the documents provided in Appendices A-F, which are outlined below:

Appendix A – Overview of the System Under Test (SUT) – includes what the system is, what the principal functionality is, and expected behaviours.

Appendix B – System Requirements Specification (SRS) of the SUT. This document outlines the functional and non-functional requirements of the SUT, including any constraints, assumptions, and dependencies.

Appendix C – Database Schema – This document contains the table definitions and an Entity-Relationship Diagram (ERD) of the current database implemented within the SUT.

Appendix D – List of Implemented and Pending Features – As the system is currently in development, this document will outline which features have been implemented and are ready to test, and those currently in development. It is expected your test plan will use this information to assist in scoping the test plan.

Appendix E – UI Design & Copy – This document contains UI mockups of the different interfaces of the LMS.

Appendix F – Wireframes & Journeys – This document contains wireframes depicting the visual representation of the User Interface and the journey a user can take to accomplish certain tasks in the SUT.

The structure of your test plan is entirely up to you. However, it is expected to follow a similar structure of test plans presented during this unit, such as:

1. Introduction
2. Test Objectives
3. Scope of Testing
    - 3.1 In-Scope
    - 3.2 Out-of-Scope
4. Test Criteria
    - 4.1 Entry Criteria
    - 4.2 Exit Criteria
5. Test Environment
6. Resource Allocation
7. Test Data Requirements
8. Test Schedule
9. Testing Strategy
    - 9.1 Unit Testing
    - 9.2 Integration Testing
    - 9.3 System Testing
    - 9.4 User Acceptance Testing
10. Testing Tools
11. Test Deliverables
12. Test Responsibilities
13. Risks and Assumptions
14. Approvals
15. Appendices

Note: Although the system you are developing a test plan for was produced by a Junior Developer with assistance from Generative AI technologies, under no circumstances should you use such technologies to write your test plan for you.

4. Submission Instructions

You are required to submit your test plan as a Microsoft Word document to the unit Moodle area. Your test plan will automatically be scanned for plagiarism using TurnItIn. Your file should be named using the convention below:

Firstname_Lastname_StudentIDNumber (**Example**: Lewis_Evans_12345678.pdf)

5. Summative Marking Scheme

Criteria (Weighting)	0-19%	20-39%	40-49%	50-59%	60-69%	70-79%	80-100%
Test Strategy and Approach (40%) ***Types of Testing****: Does the plan clearly identify types of testing to be conducted (e.g., unit, integration, system)?* ***Tools and Techniques:*** *Are testing tools and techniques appropriately chosen and justified?* ***Risk Assessment:*** *Does the plan include a risk assessment and strategies to mitigate those risks?* ***Resource Allocation****: Are resources like time, personnel, and equipment properly allocated and justified?*	The test plan presents a vague and incoherent test strategy.	The test plan provides a basic, generic testing approach.	The test plan mentions relevant techniques but lacks clear justification.	The test plan details a clear approach, somewhat tailored to the SUT.	The test plan outlines a well-defined and justified strategy.	The test plan presents a methodical and highly relevant testing approach.	The test plan articulates an exceptionally detailed and impeccable testing approach.
Coverage and Completeness (40%) ***Scope of Testing:*** *Does the plan outline how all the system's features and functionalities will be subjected to testing?* ***Coverage Metrics:*** *Does the plan outline metrics that will be used to measure the extent of coverage?*	The test plan overlooks significant areas of the SUT.	The test plan covers a few parts, leaving major areas untested.	The test plan touches upon some SUT components but lacks depth.	The test plan provides decent coverage of most SUT areas.	The test plan achieves thorough coverage of nearly all SUT components.	The test plan ensures comprehensive and meticulous coverage.	The test plan guarantees exhaustive and impeccable coverage of the SUT.
Organisation and Clarity (20%) **Structure**: Is the test plan well-organised, following a logical structure? **Language and Terminology:** Is the language clear, and are technical terms appropriately used and explained? **Documentation:** Are tables, diagrams, or other supplementary materials used effectively to enhance clarity?	The test plan is disorganised and lacks a clear structure.	The test plan has a rudimentary structure but lacks clarity in many sections.	The test plan is organised to some extent but could benefit from clearer delineation.	The test plan is clear with reasonably well-defined sections.	The test plan is effectively organised with a logical flow.	The test plan has a very well-structured format with seamless transitions.	The test plan exemplifies outstanding organisation with crystal-clear presentation.

Appendix A – System Under Test (SUT) Overview

This document aims to offer a high-level overview of the Library Management System (LMS), which is currently undergoing a testing phase. This system is not intended for educational institutions but is designed to assist small, local libraries in effectively managing their resources and services. This overview serves as a foundational document, delineating the main aspects of the system, to guide the subsequent testing strategies and protocols.

Introduction

The Library Management System (LMS) is a specialised software solution tailored for small, local libraries. Its principal function is to automate and streamline various library operations to improve efficiency and reduce human error. By digitising tasks such as cataloguing books, managing member records, issuing loans, and calculating fines, the LMS aims to modernise the library's operational framework. In the broader context of community services, this system serves as a pivotal tool to foster a more organised, accessible, and user-friendly environment for both staff and library members.

System Architecture

The LMS is constructed using a modular architecture, employing a three-tier design that comprises the Presentation Layer, Business Logic Layer, and Data Access Layer:

Presentation Layer
User Interface: Developed using JavaFX, this is the graphical front-end of the system where interactions with users occur.
Interface Controllers: These are Java classes responsible for handling user inputs and updating the UI.

Business Logic Layer
Core Services: Implemented in Java, this layer encapsulates the primary functionalities like book and member management, loan processing, and fine calculations.
Utilities: Additional helper classes and methods that support core services, such as date calculations for fines.

Data Access Layer
Database Management: SQLite serves as the database engine, storing all persistent data related to books, members, loans, and fines.
Data Access Objects (DAOs): Java classes using JDBC to perform CRUD operations on the SQLite database.

User Categories and Roles

The LMS caters to two primary categories of users:

1. **Library Staff:**
   Administrators: These users have full administrative rights, including system settings and user management.
   Librarians: They have access to all features except administrative settings. They manage books, members, loans, and fines.

2. **Members:**
   Adult Members: Full access to loan services.
   Junior Members: Access to loan services but restricted to children's literature.

Core Functionalities

The LMS is feature-rich, aiming to cover all aspects of library management:

Book Management
Cataloguing: Adding new books to the system, including metadata like ISBN, title, author, and genre.
Inventory Management: Tracking the status of each book—whether it's available, reserved, or loaned out.

Member Management
Registration: Creating new member records with details like name, contact information, and membership type (Adult or Junior).
Profile Management: Updating or deactivating existing member profiles.

Loan Management
Issuance: Allowing staff to issue books to members and automatically updating the book's status.
Returns: Processing the return of loaned books and updating inventory status.

Fine Management
Fine Calculation: Automatic generation of fines based on the overdue period.
Payment Processing: Recording fine payments and updating the member's account.

Operational Context

The LMS is a standalone application that operates independently of other systems. It is generally installed on a local server or computer within the library premises and accessed through workstations connected to the same network. While the current version of the system does not offer any external integrations, future iterations could potentially link with community portals or payment gateways for enhanced functionality.

Appendix B – System Requirements Specification (SRS)

B.1 Introduction

The primary objective of this SRS document is to serve as an exhaustive blueprint for the Library Management System (LMS). It aims to provide:

- A clear understanding of the system's functionality and constraints.
- A roadmap for developers to build the system.
- A guide for testers to validate the system.
- A reference document for stakeholders and project managers.

B.2 Functional Requirements

B.2.1 User Authentication

FR2.1.1: The system shall allow librarians to log in using a username and password.
FR2.1.2: The system shall lock accounts after three unsuccessful login attempts.

B.2.2 Book Management

FR2.2.1: Librarians should be able to add new books to the system, capturing attributes such as title, author, and ISBN.
FR2.2.2: The system shall generate a unique Book ID for each new book entry.

B.2.3 Member Management

FR2.3.1: Librarians should be able to add, update, or delete member records.
FR2.3.2: The system should validate the member's email format.

B.2.4 Loan Management

FR2.4.1: The system shall allow librarians to issue loans to members.
FR2.4.2: The system shall automatically set the loan duration and calculate due dates.

B.2.5 Fine Management

FR2.5.1: The system should automatically calculate fines for late returns based on rules. These rules include a £1 fine for 1 to 7 days late, £5 for 8 to 14 days late, and £1 additional for each day after 14 days.
FR2.5.2: Librarians should be able to mark fines as paid, which should update the database accordingly.

B.3 Non-Functional Requirements

B.3.1 Performance

NFR3.1.1: Response time for queries should not exceed 2 seconds.

B.3.2 Security

NFR3.2.1: All user passwords shall be encrypted using industry-standard algorithms, such as SHA-256.
NFR3.2.2: The system shall have role-based access control to restrict unauthorised actions.

B.3.3 Usability

NFR3.3.1: The system shall be user-friendly, requiring minimal training for librarians.
NFR3.3.2: Error messages should be clear, instructive, and localised.

B.3.4 Scalability

NFR3.3.4: The system should be scalable to support the addition of new modules or features without requiring a complete system overhaul.

B.4 Constraints and Assumptions

B.4.1 Constraints

Constraint 4.1.1: The system shall be developed using Java and SQLite, as these technologies are already in use by the library.

Constraint 4.1.2: The system must operate on Windows 10 and above, as these are the operating systems installed on library computers.

B.4.2 Assumptions

Assumption 4.2.1: All librarians have basic computer skills and can navigate the Windows operating system.

Assumption 4.2.2: The local library has a reliable internet connection for software updates and cloud backup.

Appendix C – Database Schema

The **current** database schema for the LMS is as follows:

Members Table

Attribute	Data Type	Notes
MemberID	INTEGER	PRIMARY KEY, AUTOINCREMENT
FirstName	TEXT	NOT NULL
LastName	TEXT	NOT NULL
DateJoined	DATE	NOTNULL, DEFAULT CURRENT_DATE
PhoneNumber	TEXT	
Email	TEXT	

Books Table

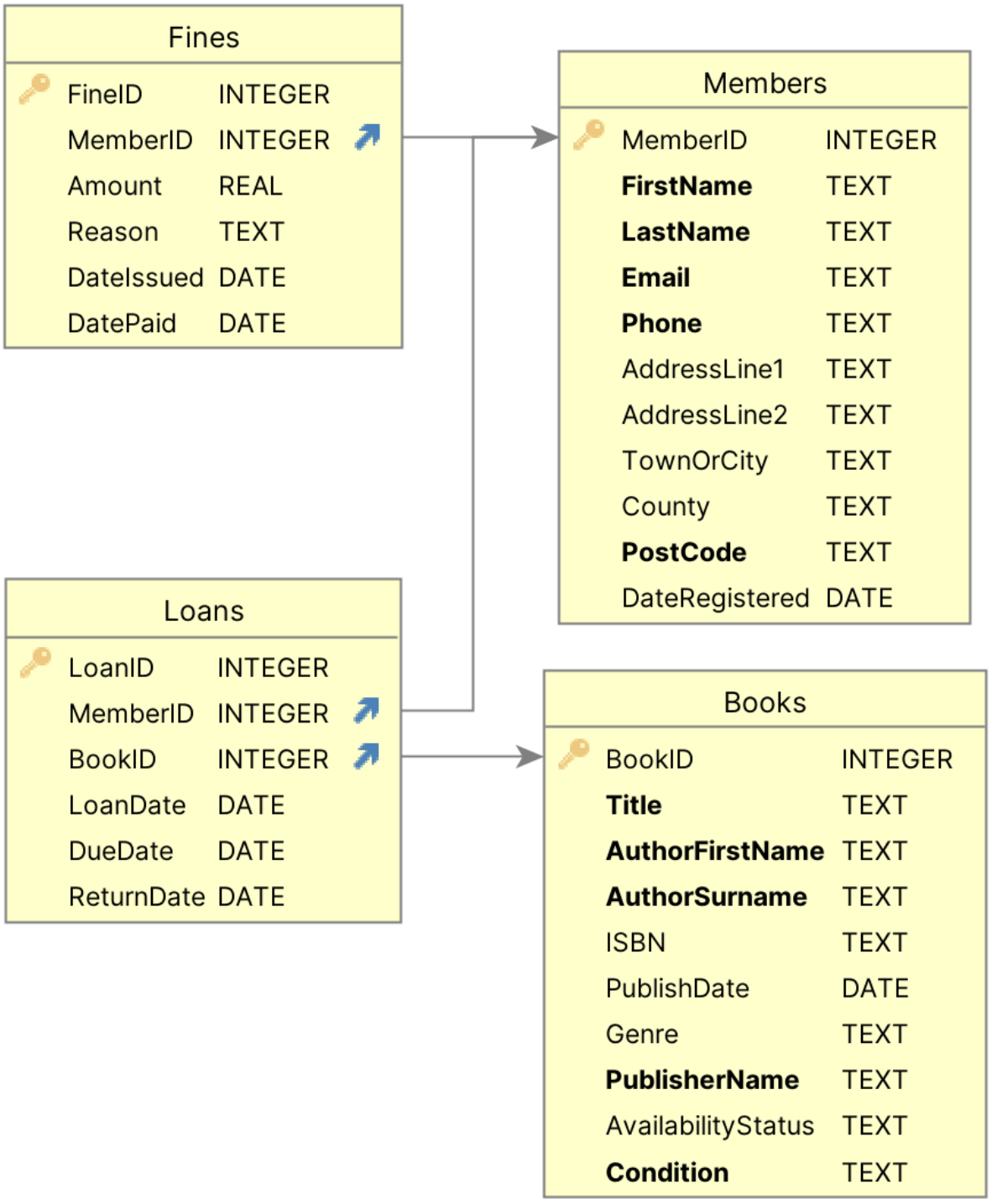
Attribute	Data Type	Notes
BookID	INTEGER	PRIMARY KEY, AUTOINCREMENT
Title	TEXT	NOT NULL
Author	TEXT	NOT NULL
ISBN	TEXT	
Published	DATE	
Genre	TEXT	
Status	TEXT	NOT NULL

Loans Table

Attribute	Data Type	Notes
LoanID	INTEGER	PRIMARY KEY, AUTOINCREMENT
MemberID	INTEGER	FOREIGN KEY REFERENCES Members(MemberID)
BookID	INTEGER	FOREIGN KEY REFERENCES Books(BookID)
DateBorrowed	DATE	NOT NULL, DEFAULT CURRENT_DATE
DateDue	DATE	NOT NULL

Fines Table

Attribute	Data Type	Notes
FineID	INTEGER	PRIMARY KEY, AUTOINCREMENT
MemberID	INTEGER	FOREIGN KEY REFERENCES Members(MemberID)
Amount	REAL	
Reason	TEXT	
DateIssued	DATE	DEFAULT CURRENT_DATE
DatePaid	DATE	


Fines
FineID INTEGER
MemberID INTEGER
Amount REAL
Reason TEXT
DateIssued DATE
DatePaid DATE
Members
MemberID INTEGER
FirstName TEXT
LastName TEXT
Email TEXT
Phone TEXT
AddressLine1 TEXT
AddressLine2 TEXT
TownOrCity TEXT
County TEXT
PostCode TEXT
DateRegistered DATE
Loans
LoanID INTEGER
MemberID INTEGER
BookID INTEGER
LoanDate DATE
DueDate DATE
ReturnDate DATE
Books
BookID INTEGER
Title TEXT
AuthorFirstName TEXT
AuthorSurname TEXT
ISBN TEXT
PublishDate DATE
Genre TEXT
PublisherName TEXT
AvailabilityStatus TEXT
Condition TEXT

Appendix D – List of Implemented and Pending Features

Based on the current status of the project, Table 1 provides a list of features and their current status in terms of implementation.

Table 1 Feature Status

Feature ID	Feature Description	Sub-Features	Implementation Status
F1.1	User Authentication	User Login	Not Implemented
		Account Locked after 3 failed attempts	Not Implemented
		Password Reset	Not Implemented
F2	Book Management	Viewing all current books	Implemented
		Add new book	Implemented
		Generating unique Book IDs	Implemented
		Cataloguing (ISBN, title, author, genre)	Implemented
		ISBN validation	Implemented
		Updating book information	Implemented
		Searching for books	Implemented
F3	Member Management	Viewing all members	Implemented
		Add new member	Implemented
		Generating unique Member IDs	Implemented
		Updating member profiles	Implemented
		Email format verification	Implemented
F4	Loan Management	Viewing all loans	Implemented
		Issuing loans	Implemented
		Automatic loan duration and due date calculation	Implemented
		Generating unique Loan IDs	Implemented
		Return a book that was on loan	Implemented
F5	Fine Management	Viewing all fines imposed	Implemented
		Automatic calculation of new fines based on book being returned late	Implemented
		Marking a fine as paid	Implemented
F6	Bulk Operations	Bulk addition of books	Not Implemented
		Bulk deletion of books	Not Implemented
		Bulk updates of member information	Not Implemented
F7	System Settings	Configurable default loan durations	Not Implemented
		Configurable fine amounts	Not Implemented
		Backup and restore database	Not Implemented
F8	Security	Password hashing using SHA-256	Not Implemented

Appendix E – UI Design & Copy

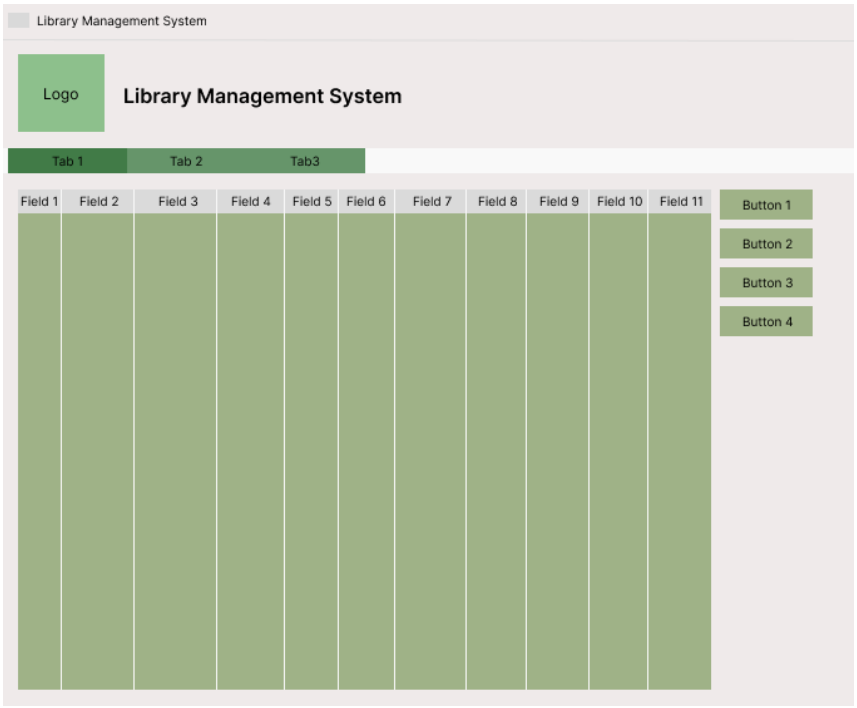
Copywriter: Margaret Jones

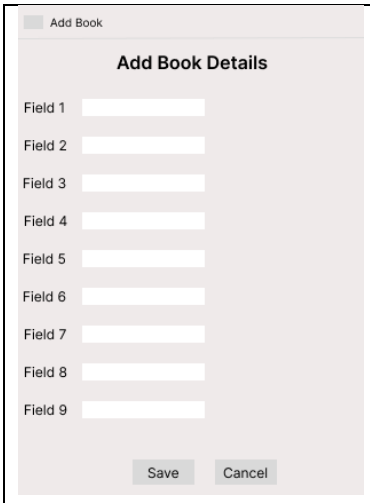
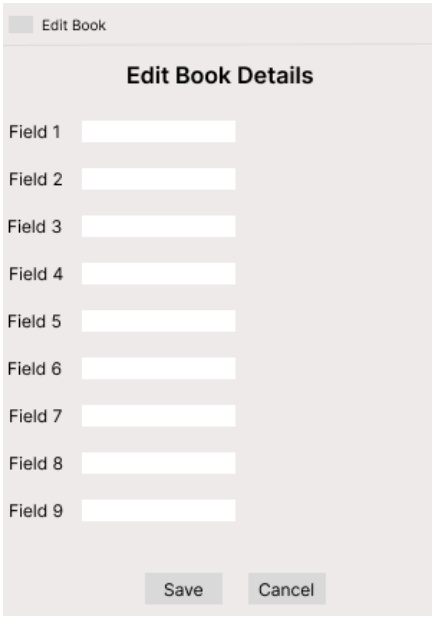
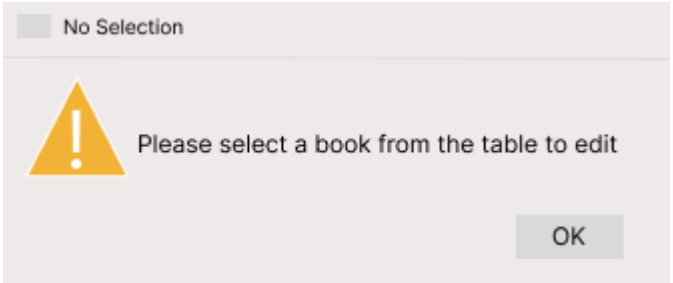
Version Control

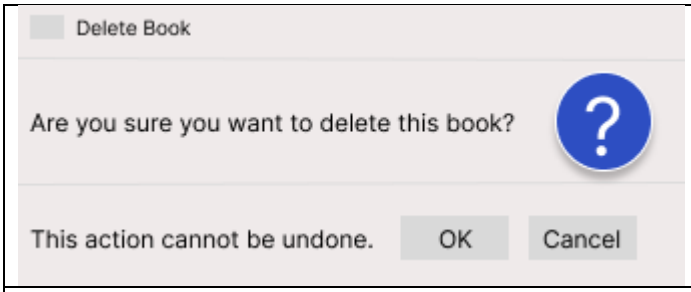
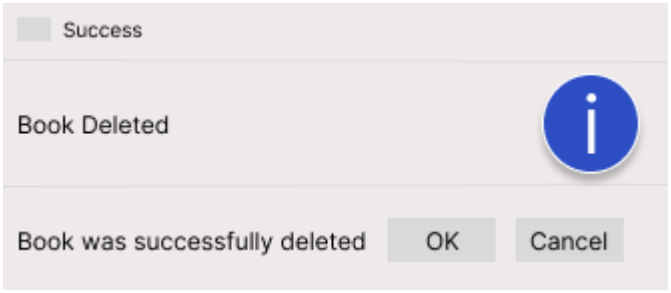
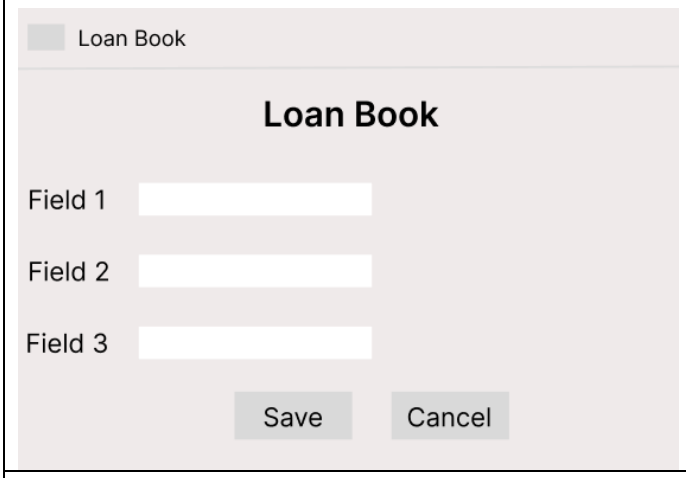
Version	Date	Changed by	Comments
1	26/09/2023	Margaret Jones	Copy document created

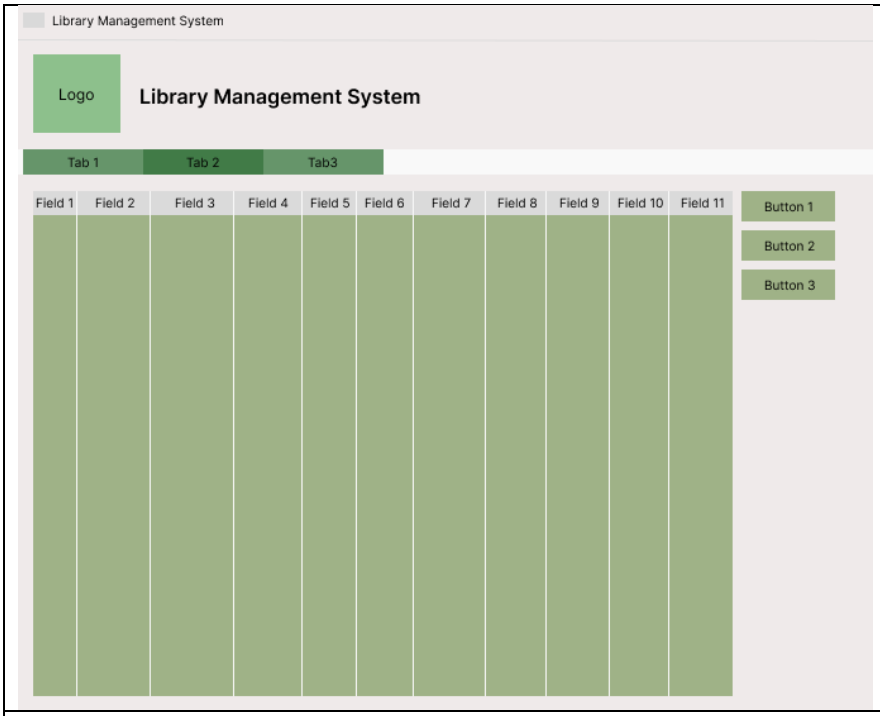
Stakeholders

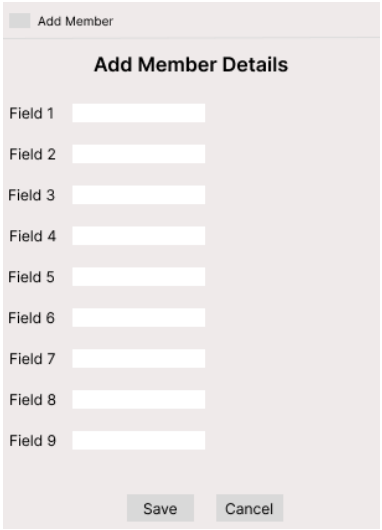
Name	Role	Approval	Sign-off (attach email)
Raj Patel	Project Manager	Sign-off	

UI Screen	Copy
Book Manager Screen 	<Toolbar> Library Management System <Header> Library Management System <Tab 1> Book Manager <Tab 2> Member Manager <Tab 3> Loan Manager <Field 1> Book ID <Field 2> Title <Field 3> Author First Name <Field 4> Author Surname <Field 5> ISBN <Field 6> Publish Date <Field 7> Genre <Field 8> Publisher Name <Field 9> Availability Status <Field 10> Condition <Button 1> Add <Button 2> Edit <Button 3> Delete <Button 4> Loan Book
Add Book Window	<Toolbar> Add Book <Header> Add Book Details

	<Field 1> Title: <Field 2> Author First Name: <Field 3> Author Surname: <Field 4> ISBN: <Field 5> Publish Date: <Field 6> Genre: <Field 7> Publisher Name: <Field 8> Availability Status: <Field 9> Condition: <Button 1> Save <Button 2> Cancel
Edit Book Window 	<Toolbar> Edit Book <Header> Edit Book Details <Field 1> Title: <Field 2> Author First Name: <Field 3> Author Surname: <Field 4> ISBN: <Field 5> Publish Date: <Field 6> Genre: <Field 7> Publisher Name: <Field 8> Availability Status: <Field 9> Condition: <Button 1> Save <Button 2> Cancel
Edit Book (No Book Selected) Window 	<Toolbar> No Selection <Body> Please select a member from the table to edit <Button 1> OK
Delete Book Window	<Toolbar> Delete Book <Body 1> Are you sure you want to delete this book? <Body 2> This action cannot be

	undone. <Button 1> OK <Button 2> Cancel
Delete Book Successful Window 	<Toolbar> Success <Body 1> Book Deleted <Body 2> Book was successfully deleted <Button 1> OK <Button 2> Cancel
Loan Book Window 	<Toolbar> Loan Book <Header> Loan Book <Field 1> Book ID: <Field 2> Member ID: <Field 3> Loan Duration (days): <Button 1> Save <Button 2> Cancel
Member Manager Screen	<Toolbar> Library Management System <Header> Library Management System <Tab 1> Book Manager <Tab 2> Member Manager <Tab 3> Loan Manager <Field 1> Member ID <Field 2> First Name <Field 3> Last Name <Field 4> Email <Field 5> Phone

<table>
<tr>
<td> 
</td>
<td><Field 6> Address Line 1
<Field 7> Address Line 2
<Field 8> Town/City
<Field 9> County
<Field 10> Postcode
<Field 11> Date Registered

<Button 1> Add
<Button 2> Edit
<Button 3> Delete</td>
</tr>
<tr>
<td>Add Member Window
 
</td>
<td><Toolbar> Add Member

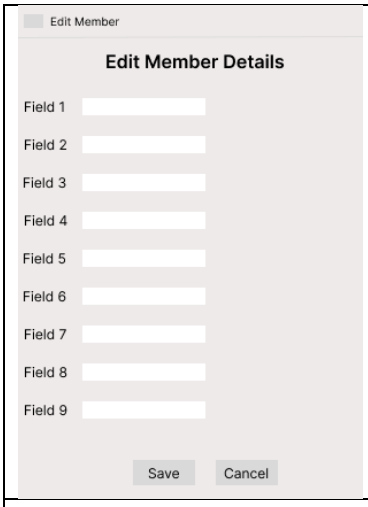
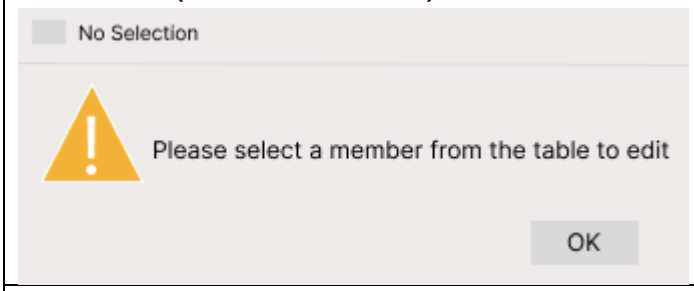
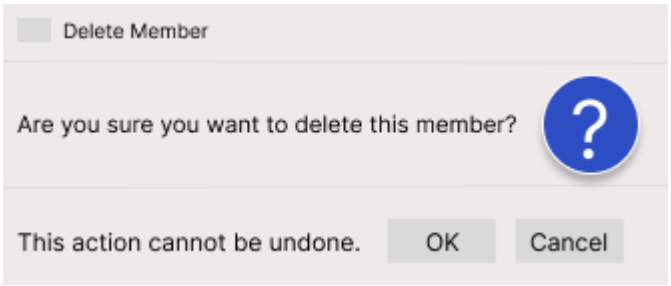
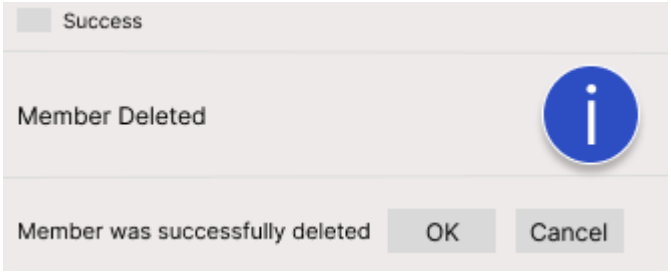
<Header> Add Member Details

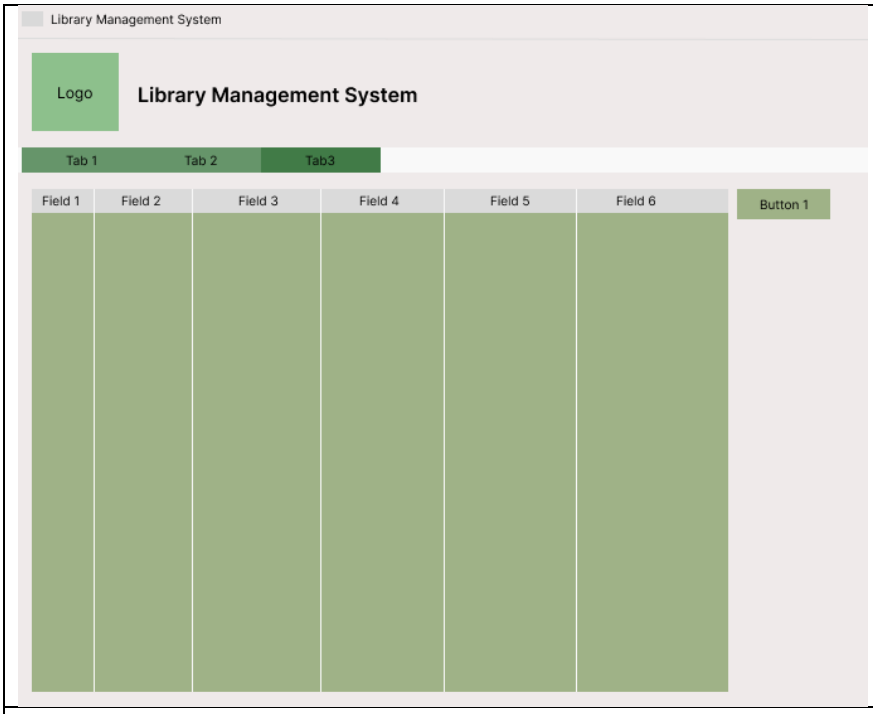
<Field 1> First Name:
<Field 2> Last Name:
<Field 3> Email:
<Field 4> Phone:
<Field 5> Address Line 1:
<Field 6> Address Line 2:
<Field 7> Town/City:
<Field 8> Postcode:
<Field 9> Date Registered:

<Button 1> Save
<Button 2> Cancel</td>
</tr>
<tr>
<td>Edit Member Window</td>
<td><Toolbar> Edit Member

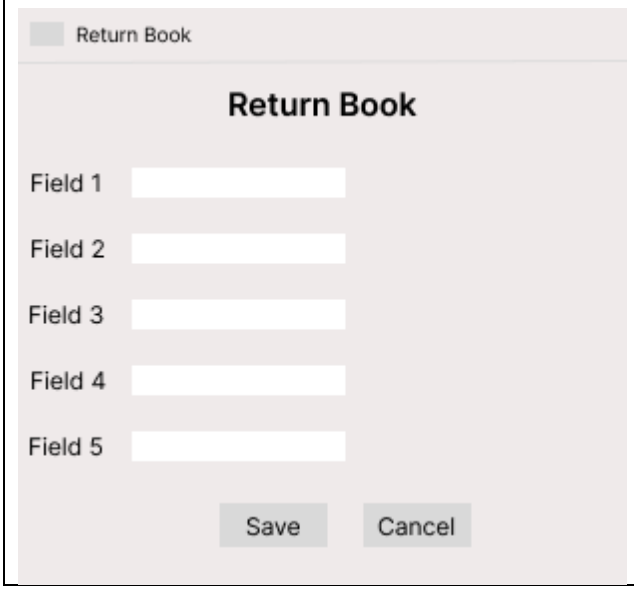
<Header> Edit Member Details

<Field 1> First Name:
<Field 2> Last Name:
<Field 3> Email:
<Field 4> Phone:
<Field 5> Address Line 1:
<Field 6> Address Line 2:
<Field 7> Town/City:</td>
</tr>
</table>

	<Field 8> Postcode: <Field 9> Date Registered: <Button 1> Save <Button 2> Cancel
Edit Member (No Member Selected) Window 	<Toolbar> No Selection <Body> Please select a member from the table to edit <Button 1> OK
Delete Member Window 	<Toolbar> Delete Member <Body 1> Are you sure you want to delete this member? <Body 2> This action cannot be undone. <Button 1> OK <Button 2> Cancel
Delete Member Successful Window 	<Toolbar> Success <Body 1> Member Deleted <Body 2> Member was successfully deleted <Button 1> OK <Button 2> Cancel
Loan Manager Screen	<Toolbar> Library Management System <Header> Library Management System <Tab 1> Book Manager

<table>
<tr>
<td> 
</td>
<td><Tab 2> Member Manager
<Tab 3> Loan Manager

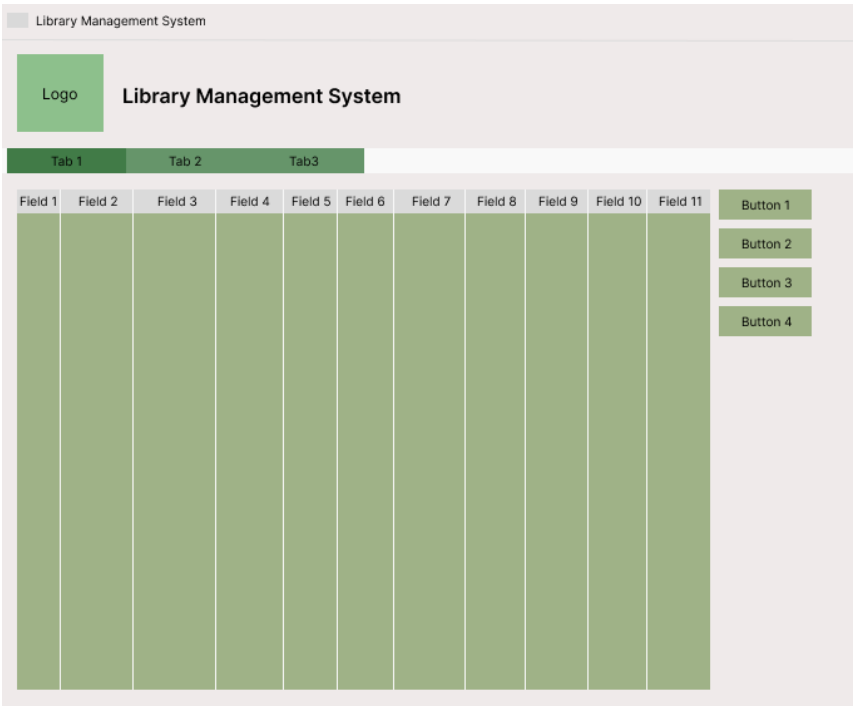
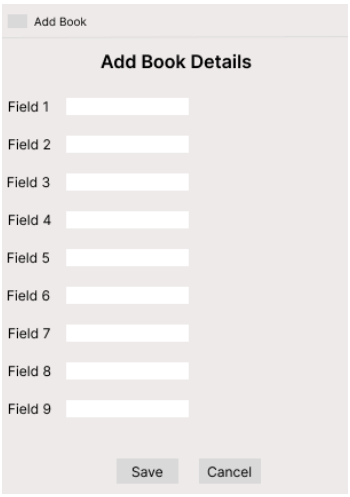
<Field 1> Loan ID
<Field 2> Member ID
<Field 3> Book ID
<Field 4> Loan Date
<Field 5> Due Date
<Field 6> Return Date

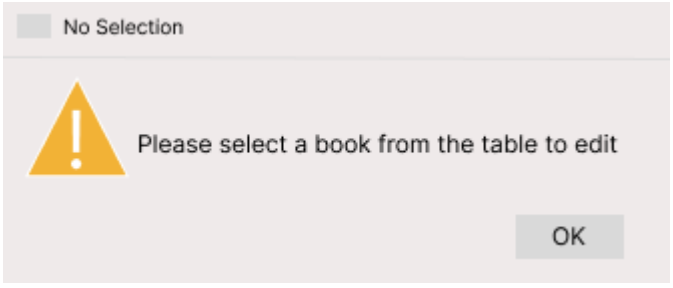
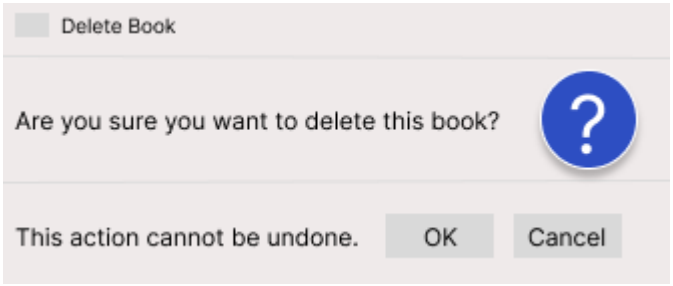
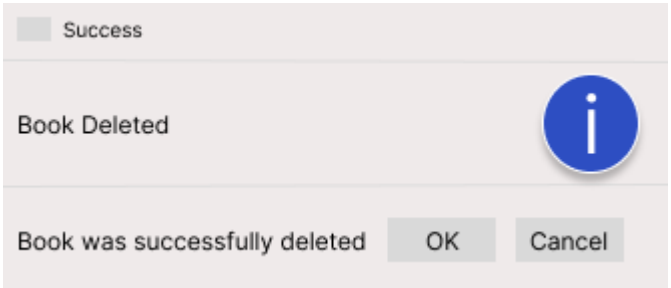
<Button 1> Add</td>
</tr>
<tr>
<td>Return Book Window
 
</td>
<td><Toolbar> Return Book

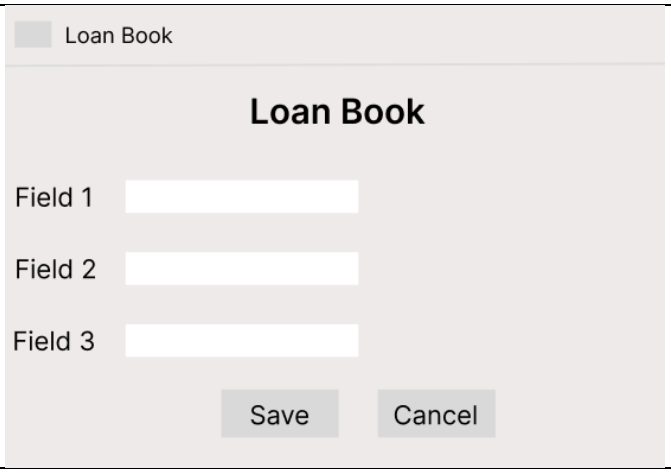
<Header> Return Book

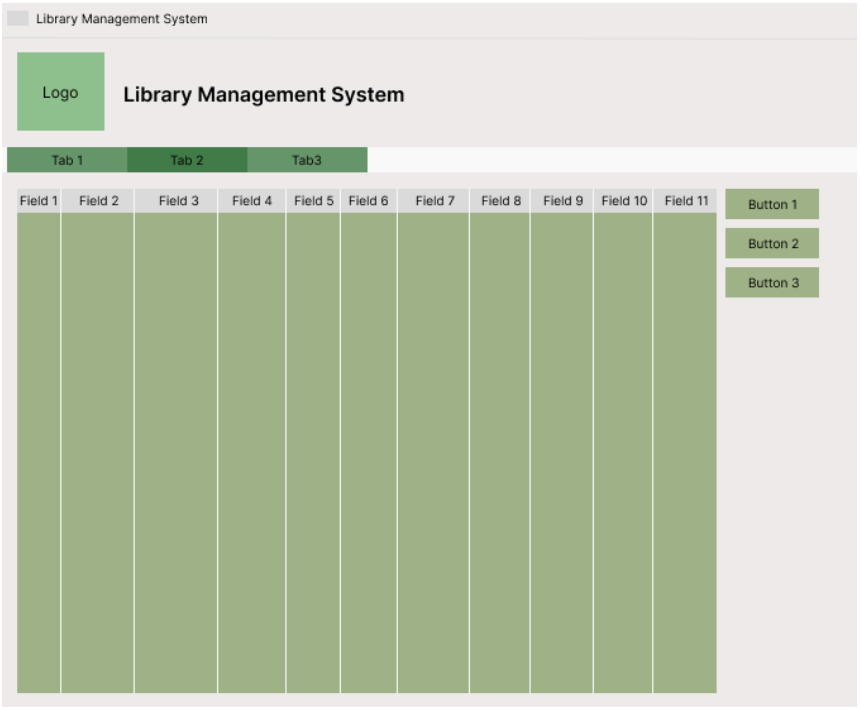
<Field 1> Loan ID:
<Field 2> Member ID:
<Field 3> Book ID:
<Field 4> Loan Date:
<Field 5> Return Date:

<Button 1> Save
<Button 2> Cancel</td>
</tr>
</table>

UI Screen	Copy
Book Manager Screen 	<Toolbar> Library Management System <Header> Library Management System <Tab 1> Book Manager <Tab 2> Member Manager <Tab 3> Loan Manager <Field 1> Book ID <Field 2> Title <Field 3> Author First Name <Field 4> Author Surname <Field 5> ISBN <Field 6> Publish Date <Field 7> Genre <Field 8> Publisher Name <Field 9> Availability Status <Field 10> Condition <Button 1> Add <Button 2> Edit <Button 3> Delete <Button 4> Loan Book
Add Book Window 	<Toolbar> Add Book <Header> Add Book Details <Field 1> Title: <Field 2> Author First Name: <Field 3> Author Surname: <Field 4> ISBN: <Field 5> Publish Date: <Field 6> Genre: <Field 7> Publisher Name: <Field 8> Availability Status: <Field 9> Condition: <Button 1> Save <Button 2> Cancel
Edit Book Window	<Toolbar> Edit Book

	<Header> Edit Book Details <Field 1> Title: <Field 2> Author First Name: <Field 3> Author Surname: <Field 4> ISBN: <Field 5> Publish Date: <Field 6> Genre: <Field 7> Publisher Name: <Field 8> Availability Status: <Field 9> Condition: <Button 1> Save <Button 2> Cancel
Edit Book (No Book Selected) Window 	<Toolbar> No Selection <Body> Please select a member from the table to edit <Button 1> OK
Delete Book Window 	<Toolbar> Delete Book <Body 1> Are you sure you want to delete this book? <Body 2> This action cannot be undone. <Button 1> OK <Button 2> Cancel
Delete Book Successful Window 	<Toolbar> Success <Body 1> Book Deleted <Body 2> Book was successfully deleted <Button 1> OK <Button 2> Cancel
Loan Book Window	<Toolbar> Loan Book <Header> Loan Book <Field 1> Book ID:

<table>
<tr>
<td> 
</td>
<td><Field 2> Member ID:
<Field 3> Loan Duration (days):

<Button 1> Save
<Button 2> Cancel</td>
</tr>
<tr>
<td>Member Manager Screen
 
</td>
<td><Toolbar> Library Management System

<Header> Library Management System

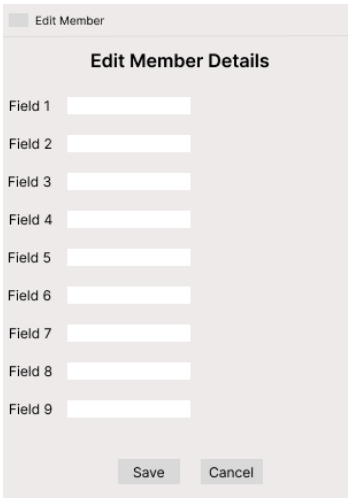
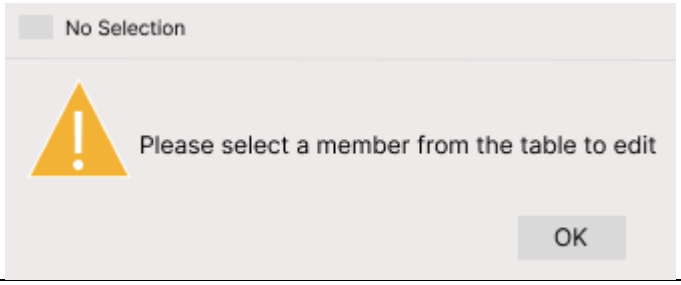
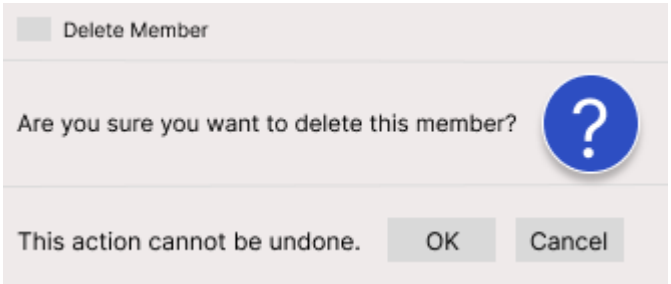
<Tab 1> Book Manager
<Tab 2> Member Manager
<Tab 3> Loan Manager

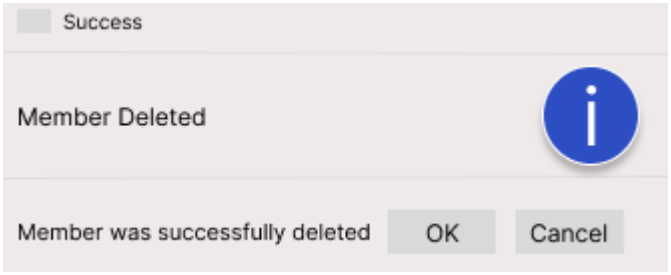
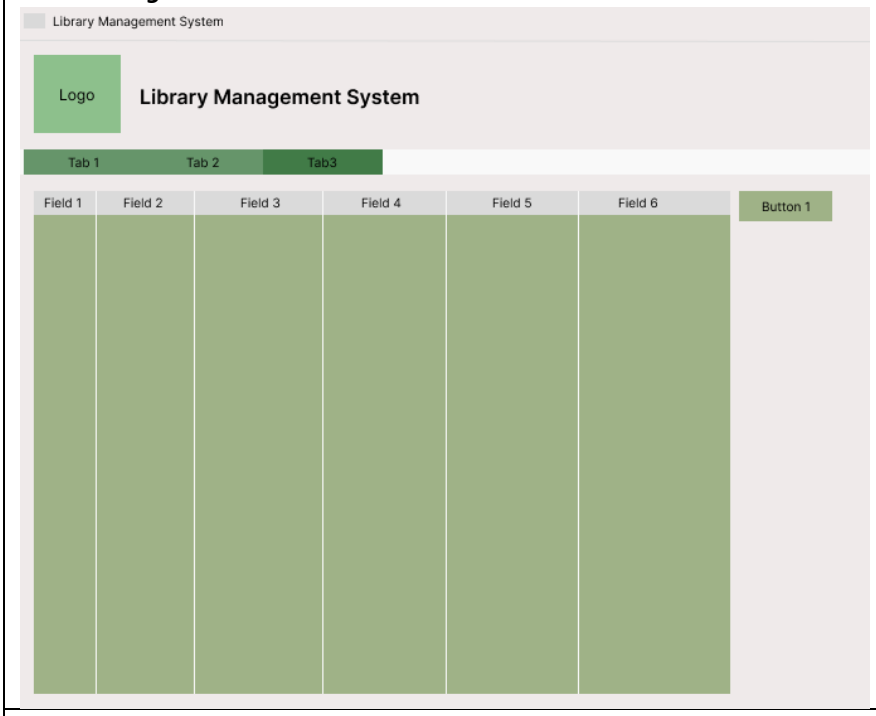
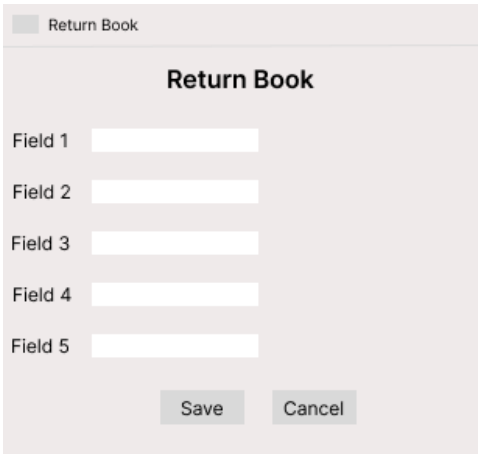
<Field 1> Member ID
<Field 2> First Name
<Field 3> Last Name
<Field 4> Email
<Field 5> Phone
<Field 6> Address Line 1
<Field 7> Address Line 2
<Field 8> Town/City
<Field 9> County
<Field 10> Postcode
<Field 11> Date Registered

<Button 1> Add
<Button 2> Edit
<Button 3> Delete</td>
</tr>
<tr>
<td>Add Member Window</td>
<td><Toolbar> Add Member

<Header> Add Member Details

<Field 1> First Name:
<Field 2> Last Name:
<Field 3> Email:
<Field 4> Phone:
<Field 5> Address Line 1:
<Field 6> Address Line 2:</td>
</tr>
</table>

	<Field 7> Town/City: <Field 8> Postcode: <Field 9> Date Registered: <Button 1> Save <Button 2> Cancel
Edit Member Window 	<Toolbar> Edit Member <Header> Edit Member Details <Field 1> First Name: <Field 2> Last Name: <Field 3> Email: <Field 4> Phone: <Field 5> Address Line 1: <Field 6> Address Line 2: <Field 7> Town/City: <Field 8> Postcode: <Field 9> Date Registered: <Button 1> Save <Button 2> Cancel
Edit Member (No Member Selected) Window 	<Toolbar> No Selection <Body> Please select a member from the table to edit <Button 1> OK
Delete Member Window 	<Toolbar> Delete Member <Body 1> Are you sure you want to delete this member? <Body 2> This action cannot be undone. <Button 1> OK

	<Button 2> Cancel
Delete Member Successful Window 	<Toolbar> Success <Body 1> Member Deleted <Body 2> Member was successfully deleted <Button 1> OK <Button 2> Cancel
Loan Manager Screen 	<Toolbar> Library Management System <Header> Library Management System <Tab 1> Book Manager <Tab 2> Member Manager <Tab 3> Loan Manager <Field 1> Loan ID <Field 2> Member ID <Field 3> Book ID <Field 4> Loan Date <Field 5> Due Date <Field 6> Return Date <Button 1> Add
Return Book Window 	<Toolbar> Return Book <Header> Return Book <Field 1> Loan ID: <Field 2> Member ID: <Field 3> Book ID: <Field 4> Loan Date: <Field 5> Return Date: <Button 1> Save <Button 2> Cancel

Appendix F – Journey Wireframes

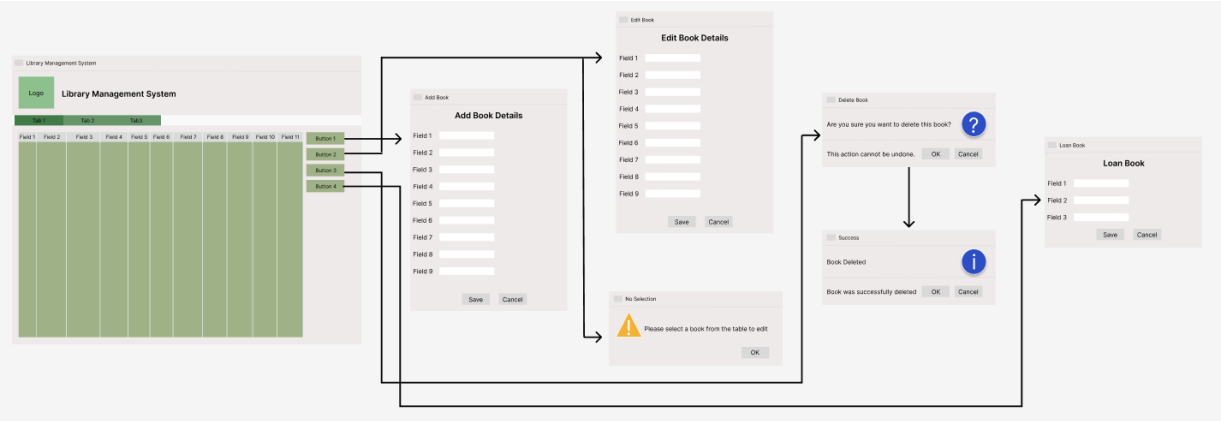
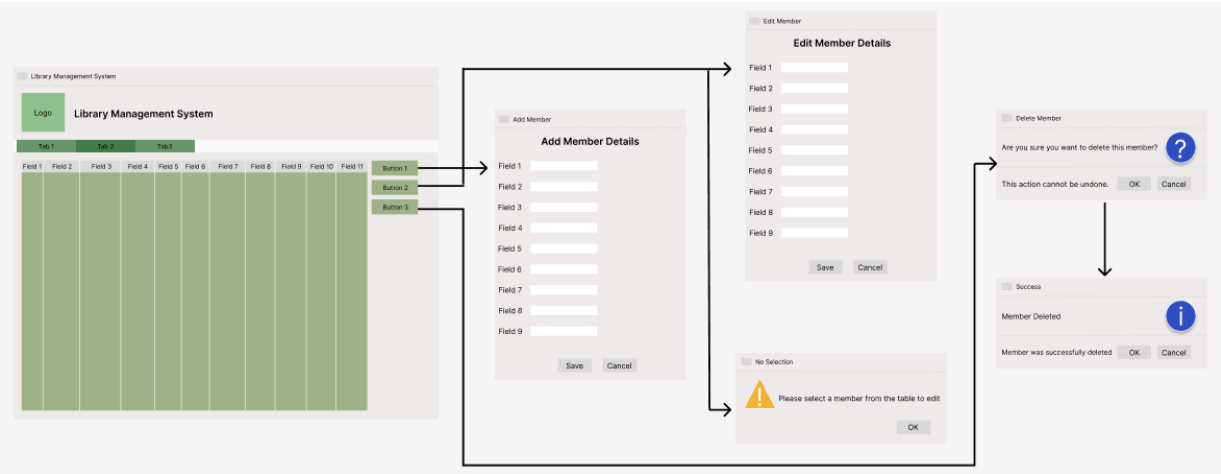
UI Designer: Peter Frost

Version Control

Version	Date	Changed by	Comments
1	26/09/2023	Peter Frost	Wireframe document created

Stakeholders

Name	Role	Approval	Sign-off (attach email)
Raj Patel	Project Manager	Sign-off	

Journey	Wireframe
Book Manager Screen	
Member Manager Screen	
Loan Manager Screen