**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:** | 2CWK70 |
| **Assessment Weighting:** | 70% |
| **Assessment Title:** | Test Suite |
| **Type:** | Individual |
| **Hand-In Deadline:** | See Moodle |
| **Hand-In Format and Mechanism:** | Submission is online, via Moodle |

**Learning outcomes being assessed:**

|  |  |
| --- | --- |
| **LO1** | Use industry standard automated testing tools to conduct large volumes of software tests on code in development |
| **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](https://www.mmu.ac.uk/student-case-management/guidance-for-students/student-code-of-conduct/) and [Regulations for Undergraduate Programmes](https://www.mmu.ac.uk/student-life/course/assessments#ai-63930-1). 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](mailto: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](https://www.mmu.ac.uk/student-life/course/assessments#ai-63930-2) on the [Student Life web pages](https://www.mmu.ac.uk/news-and-events/news/story/?id=16239).

|  |  |
| --- | --- |
| **Formative Feedback:** | Students can obtain formative feedback in the scheduled lab session in Week 11. |
| **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*.* |

# Assessment Introduction

This assessment naturally follows on from the first assessment (1CWK30 – Test Plan), in which you produced a Test Plan for a System Under Test (SUT) based on a predefined scenario. This assessment is now concerned with conducting testing on the SUT, where it is envisaged that your previously developed test plan will guide your test case designs and your approach to testing the SUT. 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** for this assessment, detailing the expected elements.
* **Section 4 provides instructions** for submitting your assessment.
* **Section 5 details how you can get support** for this assessment throughout the unit.
* **Section 6 contains the summative mark scheme**, which contains the criteria you will be assessed against

# Assessment Scenario

Having completed your test plan for the Library Management System (LMS), you are now prepared to move on to the next crucial phase of the software testing lifecycle: executing the test cases. This assessment focuses on the design of test cases, their implementation and execution, and the subsequent reporting of results.

# Assessment Instructions

You are required to produce the elements in Table 1 for this assessment. Refer to the summative marking scheme in Section 6 for descriptions of things expected at each grade band.

Table Assessment Elements

|  |  |  |
| --- | --- | --- |
| **Element** | **Weighting** | **Description** |
| Test Case Design Specification Document | 30% | In this section, you are expected to create a comprehensive set of test cases that cover various scenarios, conditions, and input data. The test cases should be documented in a structured format (Excel is acceptable). Include details such as test case ID, description, prerequisites, steps, expected results, and other details as per standard test case specification design practices. |
| Implementation and Execution of Test Cases | 30% | This section involves the actual coding and execution of the test cases you've designed. You should use automated testing tools suitable for the SUT. Include the scripts, configurations, and any utilities you've developed as part of your test suite. |
| Results and Reporting | 40% | Produce **a report that contains two elements:**  **Test Summary:** should summarise the test activities, outcomes, and any relevant statistics. It should clearly indicate pass/fail status for each test case, and any deviations from expected results.  **Defect Log:** should list all the defects found, their severity, status, and other details as per standard defect reporting practices. |

# Submission Instructions

Your submission should be in the form of a ZIP file, with the following included:

* Your **Test Case Specification Document** (This can be in the form of a written document or an Excel spreadsheet).
* Your **Test** folder from your IDE, which should contain your test classes (your test suite)
* A Word or PDF file(s) containing your **1) Test Summary** and **2) Defect Log**

Your ZIP file should use the following naming convention: **Firstname\_Surname\_StudentID.zip** (e.g. Lewis\_Evans\_12345678.zip).

# Support

Support for this unit and this assessment can be sought from the Unit Leader, Dr Lewis Evans (Weeks 1- 7), Dr Ash Williams (Weeks 8+), during office hours, or by email/Teams message.

# Summative Marking Scheme

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Criteria (Weighting)** | **0-19%** | **20-39%** | **40-49%** | **50-59%** | **60-69%** | **70-79%** | **80-100%** |
| **Test Case Design (30%)**  *Coverage of functional and non-functional requirements, clarity, prioritisation* | No test cases are present, and requirements are ignored. | Test cases are minimal and irrelevant, providing insufficient coverage. | The set of test cases is incomplete and lacks a clear focus and coverage. | An adequate range of test cases targets primary functionalities. | A comprehensive set of test cases includes some non-functional requirements. | A nearly complete set of test cases is strategically prioritised and includes most requirements. | An almost exhaustive and highly focused set of test cases provides full coverage of all requirements. |
| **Implementation and Execution (30%)**  Quality of code, execution process, tool utilisation | Neither test implementation nor execution is present. | Code quality is rudimentary, executing only a few tests. | Code quality is low, with limited execution and tool utilisation. | Code quality is satisfactory, executing essential test cases. | Code is efficient, including the majority of test cases and employing basic tools. | Code quality is high, with extensive test execution and effective tool utilisation (e.g. Parameterized Tests). | Code quality is exceptional, with full test execution and optimal use of advanced tools (e.g. Parameterized Tests utilising external test data). |
| **Results and Reporting (40%)**  *Analysis, documentation, recommendations* | No Test Summary is submitted, or the report lacks critical details.  No Defect Log is submitted, or the report lacks critical details. | The Test Summary is incomplete and does not accurately summarise the testing activities.  The Defect Log is incomplete and does not accurately capture the defects found. | The Test Summary provides a basic summary but lacks depth in analysis or completeness.  The Defect Log provides basic information on defects but lacks depth in analysis or completeness. | The Test Summary is mostly complete, summarising most of the essential test activities.  The Defect Log is mostly complete, capturing most of the essential defect details. | The Test Summary is complete and includes thoughtful analysis but may lack some minor details.  The Defect Log is complete and includes all defects found with thoughtful analysis. | The Test Summary is comprehensive, offering insights into test effectiveness and coverage.  The Defect Log is comprehensive, providing full details including defect severity, status, and more. | The Test Summary is exemplary, offering deep insights and full coverage of all test activities.  The Defect Log is exemplary, providing a full, detailed, and insightful account of all defects found. |

**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.

1. **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 |
| Email | TEXT |  |
| Phone | TEXT |  |
| AddressLine1 | TEXT |  |
| AddressLine2 | TEXT |  |
| TownOrCity | TEXT |  |
| Country | TEXT |  |
| PostCode | TEXT |  |
| DateRegistered | DATE | NOTNULL, DEFAULT CURRENT\_DATE |

**Books Table**

|  |  |  |
| --- | --- | --- |
| **Attribute** | **Data Type** | **Notes** |
| BookID | INTEGER | PRIMARY KEY, AUTOINCREMENT |
| Title | TEXT | NOT NULL |
| AuthorFirstName | TEXT | NOT NULL |
| AuthorSurname | TEXT |  |
| ISBN | TEXT |  |
| PublishDate | DATE |  |
| AvailabilityStatus | TEXT |  |
| Condition | TEXT |  |

**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) |
| LoanDate | DATE | NOT NULL, DEFAULT CURRENT\_DATE |
| ReturnDate | 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 |  |

A diagram of a document

Description automatically generated

**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  A screenshot of a computer  Description automatically generated | <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  A screenshot of a computer  Description automatically generated | <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**  A screenshot of a computer  Description automatically generated | <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**  **A screenshot of a phone  Description automatically generated** | <Toolbar> No Selection  <Body> Please select a book from the table to edit  <Button 1> OK |
| **Delete Book Window**  **A screenshot of a message  Description automatically generated** | <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**  **A screenshot of a computer error message  Description automatically generated** | <Toolbar> Success  <Body 1> Book Deleted  <Body 2> Book was successfully deleted  <Button 1> OK  <Button 2> Cancel |
| **Loan Book Window**  **A screenshot of a loan book  Description automatically generated** | <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**  **A screenshot of a computer  Description automatically generated** | <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 |
| **Add Member Window**  **A screenshot of a computer  Description automatically generated** | <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 |
| **Edit Member Window**  A screenshot of a computer  Description automatically generated | <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**  **A screenshot of a phone  Description automatically generated** | <Toolbar> No Selection  <Body> Please select a member from the table to edit  <Button 1> OK |
| **Delete Member Window**  **A screenshot of a message  Description automatically generated** | <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**  **A screenshot of a computer error  Description automatically generated** | <Toolbar> Success  <Body 1> Member Deleted  <Body 2> Member was successfully deleted  <Button 1> OK  <Button 2> Cancel |
| **Loan Manager Screen**  **A screenshot of a computer  Description automatically generated** | <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  <Field 7> Fine Amont  <Button 1> Add |
| **Return Book Window**  **A screenshot of a computer  Description automatically generated** | <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 | *A screenshot of a computer screen  Description automatically generated* |
| Member Manager Screen | *A screenshot of a computer  Description automatically generated* |
| Loan Manager Screen | *A screenshot of a computer  Description automatically generated* |