Library Management System

Submitted By

Student Name	**Student ID**
Mahfuzur Rahman	221-15-5270
Md. Rakibul Islam	221-15-5829
Md. Fahimur Rahman	221-15-5953
Md Anaf fakir	221-15-4811
Md. Jahidul Islam Shuvo	221-15-5003

MINI LAB PROJECT REPORT

This Report Presented in Partial Fulfillment of the course
CSE324:Operating Systems Lab in the Computer Science and Engineering Department

DAFFODIL INTERNATIONAL UNIVERSITY

Dhaka, Bangladesh

December 11, 2024

DECLARATION

We hereby declare that this lab project has been done by us under the supervision of **Md Hefzul Hossain Papon**, **Lecturer**, Department of Computer Science and Engineering, Daffodil International University. We also declare that neither this project nor any part of this project has been submitted elsewhere as lab projects.

Submitted To:

____

Md Hefzul Hossain Papon
Lecturer
Department of Computer Science and Engineering Daffodil
International University

Submitted by

Mahfuzur Rahman SID:221-15-5270 Dept. of CSE, DIU	
Md. Rakibul Islam SID:221-15-5829 Dept. of CSE, DIU	Md. Fahimur Rahman S ID:221-15-5953 Dept. of CSE, DIU
Md Anaf fakir SID:221-15-4811 Dept. of CSE, DIU	Md. Jahidul Islam Shuvo SID:221-15-5003 Dept. of CSE, DIU

COURSE & PROGRAM OUTCOME

The following courses have course outcomes as follows:

Table 1: Course Outcome Statements

CO's	Statements
CO1	Experiment with Unix commands and shell programming.
CO2	Able to build shell programs for process and file system management with system calls.
CO3	Able to implement and analyse the performance of different algorithms of Operating systems like CPU scheduling algorithm, page replacement algorithms, deadlock avoidance, detection algorithm and so on.
CO4	Able to design and develop a course project that can have a positive impact on the environment or society or mankind.

Table 2: Mapping of CO, PO, Blooms, KP and CEP

CO	PO	Blooms	KP	CEP
CO1	PO1	C1, C2	KP3	EP1, EP3
CO2	PO2	C2	KP3	EP1, EP3
CO3	PO3	C4, A1	KP3	EP1, EP2
CO4	PO3	C3, C6, A3, P3	KP4	EP1, EP3

.

Table of Contents

Table of Contents

Chapter 1

Introduction

This chapter introduces the **Shell-Based Library Management System (LMS)**, outlining its purpose, motivation, and objectives. It discusses the system's feasibility, highlights gaps in existing solutions, and presents the expected outcomes.

1.1 Introduction

Libraries play a crucial role in organizing resources and providing access to knowledge, but many lack the resources for complex software solutions. We aim to solve that problem by creating a lightweight, shell-based LMS capable of handling book inventory, user accounts, and transaction records through Unix/Linux shell scripting. By leveraging the command-line interface (CLI), the system provides a flexible and efficient method for executing library operations without the need for specialized hardware or software.This Project describes a Library Management System (LMS) developed entirely using shell commands.

1.2 Motivation

The motivation for developing this shell command-based Library Management System (LMS) stems from the need for a resource-efficient library system that performs essential operations without requiring advanced software or hardware. Unlike traditional systems that rely on GUIs, this LMS operates solely on a command-line interface, reducing overhead and allowing for easier maintenance and customization.Though we are not bound to develop an GUI but we will develop a simple GUI for our simulation purposes. It is particularly useful for small libraries with limited budgets and resources, as well as for students and professionals who want to enhance their skills in Unix/Linux and shell scripting while working on a practical project that addresses real-world needs.

1.3 Objectives

The main objectives of the shell command-based LMS project are as follows:

1. **Efficient Book Management:** To allow library administrators to add, edit, and delete book records with ease.
2. **User Account Management:** To provide users with a mechanism to borrow and return books, track due dates, and manage account history.
3. **Transaction Logging**: To record all borrowing and returning activities to maintain a clear history of transactions.
4. **Command-Based Interface:** To ensure ease of use on Unix/Linux systems, making it accessible and simple to operate without additional software.The interface will be simple but effective.
5. **Customization and Resource Efficiency:** To deliver a system that is lightweight, customizable, and easy to update, suited for small-scale libraries.
6. **User Friendly:** To deliver seamless user friendly environments and experience without facing any kinds of difficulties .

.

1.4 Feasibility Study

A feasibility study was conducted by our project's experts to determine if a shell-based LMS could serve as a viable alternative to more complex library systems. After analysis of similar systems revealed that most LMS platforms are designed for larger institutions, with graphical user interfaces and databases that require significant resources. Our project, however, is designed for minimal environments where a command-line approach can achieve similar functionality but more effective for the consumer . After a lot of study, it has been shown that a shell-based LMS can successfully handle basic library tasks, making it a feasible solution for smaller libraries that prioritize simplicity, efficiency, and low-cost implementation.

1.5 Gap Analysis

In existing library management systems, most solutions are focused on graphical interfaces, which, while user-friendly, require more resources and maintenance. There is a lack of minimalistic systems that operate solely on a Shell based CLI, which can be a major advantage in environments where advanced infrastructure is not feasible. Our particular project fills this gap by creating a system that functions entirely on shell commands, reducing overhead, improving accessibility on low-resource hardware, and providing a customizable, low-cost , lightweight alternative that still achieves the essential functions of library management.Moreover this system is much more effective and easy for maintanance .

1.6 Project Outcome

The expected outcome of our project is a fully functional, shell-based Library Management System that can efficiently handle core library operations like book management, user tracking, and transaction logging. This LMS will provide a simple, text-based interface and a GUI for the simulation that can be easily used by library staff with basic Unix command knowledge. It is a cost-effective and lightweight solution suited for smaller libraries or educational environments where simplicity and efficiency are critical. Additionally, our project aims to demonstrate how shell scripting can be applied to real-world applications, offering a learning tool for users interested in Unix/Linux systems.

Chapter 2

Proposed Methodology/Architecture

This chapter outlines the system's architecture, which combines shell scripting with a basic user interface for seamless interaction. The back end uses scripts to handle core operations like adding, deleting, and searching for books, while the front end employs menus and prompts for intuitive user navigation. The modular design ensures scalability and ease of maintenance, supported by a structured directory system for organizing data. The methodology emphasizes efficiency, clarity, and adaptability.

2.1 Requirement Analysis & Design Specification

The LMS uses a structured directory format where books are categorized into folders, and each book has a unique text file containing details such as title, author, and availability status. User accounts are similarly organized, with each user having a personal file that tracks borrowed books and due dates. The system design relies on a set of shell scripts for various tasks: adding books, borrowing and returning books, and updating transaction logs. This modular approach ensures that the system is efficient, easy to navigate, and quick to modify, as all operations are handled directly through CLI commands.

2.1.1 Overview

The LMS is designed to utilize Unix/Linux shell commands and directory structures to manage library resources. Key components of the system include book files, user files, and transaction logs, each organized within specific directories. Requirements include the ability to add, delete, and update books, manage user accounts, and record transactions. The system is developed using Bash scripts that execute file operations based on user input. Each functionality (e.g., borrowing a book) is mapped to specific commands and scripts, providing a modular, easily maintained system architecture that is adaptable to the library's needs.

2.1.2 Proposed Methodology/ System Design

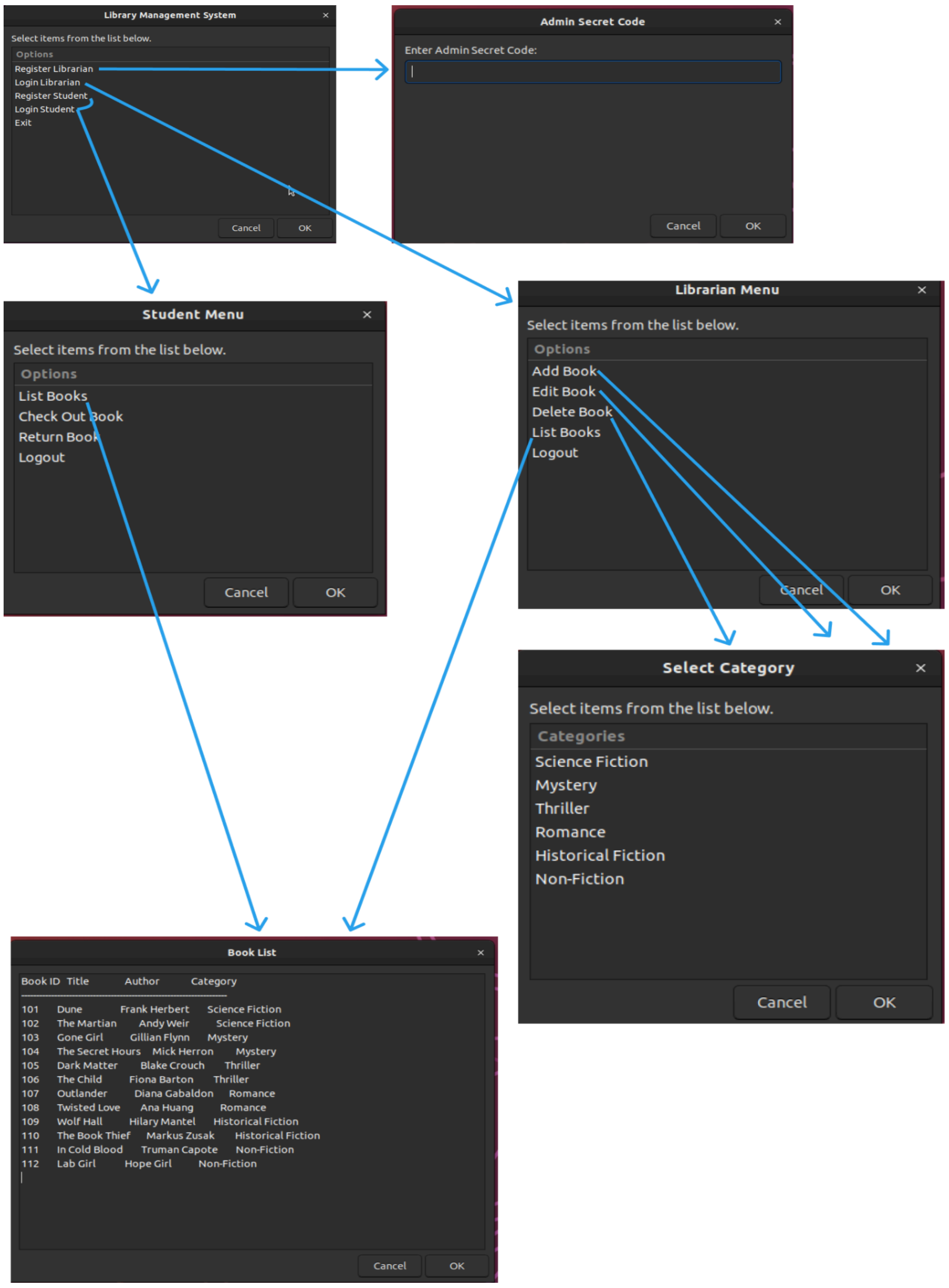
The system design includes:

- **Back End**: Shell scripts perform CRUD (Create, Read, Update, Delete) operations on book and user data. Each book and user has a dedicated file for storing relevant details.
- **Front End**: A menu-driven user interface built with tools like dialog (for CLI-based graphical menus) or basic GUI frameworks such as Tkinter or Zenity. This UI allows users to interact with the system without needing to remember commands.

2.1.3 UI Design

The UI consists of menus and forms for performing operations:

- **Main Menu**: Options for book management, user management, and transactions.
- **Book Management**: Add, search, delete, and update book records.
- **User Management**: Register new users, search user records, and manage borrowing history.
- **Transactions**: Borrow and return books, view transaction logs.



```
Book ID  Title        Author        Category
------------------------------------------------
101    Dune           Frank Herbert    Science Fiction
102    The Martian    Andy Weir        Science Fiction
103    Gone Girl      Gillian Flynn    Mystery
104    The Secret Hours   Mick Herron    Mystery
105    Dark Matter    Blake Crouch     Thriller
106    The Child      Fiona Barton     Thriller
107    Outlander      Diana Gabaldon   Romance
108    Twisted Love   Ana Huang        Romance
109    Wolf Hall      Hilary Mantel    Historical Fiction
110    The Book Thief   Markus Zusak   Historical Fiction
111    In Cold Blood  Truman Capote    Non-Fiction
112    Lab Girl       Hope Girl        Non-Fiction
```

2.2 Overall Project Plan

Our overall project plan consists of several stages:

1. **Requirement Gathering:** Identify system requirements and use cases.
2. **Design:** Develop system architecture and UI mockups.
3. **Development:** Write shell scripts and design the UI.
4. **Testing:** Validate functionality, usability, and performance.
5. **Deployment:** Implement the system in a test environment.

During the analysis stage, requirements for each library operation were identified. The design phase included directory and file structure planning, followed by writing Bash scripts for each functionality. Testing ensured the accuracy of book management and transaction records. The final implementation phase focused on deploying the LMS in a simulated environment to evaluate its usability and performance, ensuring a seamless experience for end-users.

Chapter 3

Implementation and Results

Implementation focuses on integrating shell scripts with a user interface to deliver robust functionality. Scripts handle specific tasks like book management and transaction updates, while the interface simplifies interaction. Performance testing highlights the system's speed and efficiency, making it suitable for managing thousands of records. The results demonstrate that the system meets its objectives effectively, with user feedback emphasizing its practicality and ease of use.

3.1 Implementation

The LMS is implemented through a series of Bash shell scripts that manage book records, user accounts, and transactions. Each script is responsible for a specific task: for example, adding a book, borrowing a book, or logging a return. Scripts use commands like **cat**, **awk**, **sqlit**e, **grep**, and **sed** to read, search, and update files. The book files store details like title, author, and availability, while user files track borrowed books and due dates. This modular scripting approach ensures that each functionality is isolated and easy to update, allowing for straightforward troubleshooting and enhancements.

3.2 Performance Analysis

The LMS focused on speed, resource usage, and ease of operations. Testing confirmed that the command-line approach provides a highly efficient solution, with operations executing quickly even with large numbers of books and user accounts. The system's minimal resource usage makes it suitable for low-spec machines or environments where software installations are limited. Additionally, the modularity of the scripts allows for individual components to be optimized or replaced as needed, ensuring long-term performance sustainability.

3.3 Results and Discussion

The shell-based LMS successfully handles core library functions, providing an efficient and user-friendly CLI solution. User feedback suggests that the system is easy to navigate for those familiar with basic Unix commands. The streamlined approach ensures that essential tasks such as book management and transaction logging are completed with minimal system overhead. We also have some areas for improvement, such as adding multi-user support and refining the search functionality for faster query processing.

Chapter 4

Engineering Standards and Mapping

This chapter examines the system's impact on society, the environment, and ethical considerations. The lightweight design promotes sustainability by minimizing energy use and hardware dependency. Ethical principles are upheld by ensuring data privacy and accessibility for all users. The system aligns with key engineering standards, tackling complex challenges like data consistency and multi-user operations while fostering teamwork and effective project management.

4.1 Impact on Society, Environment and Sustainability

The development of the **Shell-Based Library Management System (LMS)** has significant positive impacts on society and the environment, making it an efficient, low-resource solution for library management.

4.1.1 Impact on Life

This project creates a positive impact by providing an efficient, paperless, and eco-friendly way to manage library operations. It minimizes the need for physical records, reducing deforestation and waste. By digitalizing the process, it promotes literacy and education in an accessible manner.

○

- **Social Impact:** Enhances access to books and educational resources, fostering a learning culture.
- **Environmental Impact:** Reduces paper use and waste, contributing to environmental conservation.
- **Sustainability:** Encourages responsible use of resources, offering a long-term digital solution that can be expanded or improved.

4.1.2 Impact on Society & Environment

- **Society**: The system bridges gaps in resource allocation by simplifying book lending and tracking, ensuring that books reach more users without manual bottlenecks. It also modernizes traditional libraries, making them appealing to the tech-savvy generation.
- **Environment**: Transitioning to a digital system significantly reduces dependency on paper-based logs, thus aiding in the reduction of carbon footprint and conserving natural resources.

4.1.3 Ethical Aspects

This project adheres to ethical principles by ensuring:

- **Equitable Access:** Provides a fair opportunity for students and librarians to access library resources without bias.
- **Data Security**: Handles sensitive information, such as user credentials, responsibly by using authentication methods.
- **Transparency:** Logs all actions (e.g., book lending/returning) for accountability.
- **Privacy:** Protects the personal information of librarians and students, complying with ethical data handling practices.

4.1.4 Sustainability Plan

The sustainability of this system lies in its:

- **Scalability:** Can be easily modified or scaled to accommodate more users, books, or features (e.g., advanced analytics).
- **Maintenance:** Built using simple shell scripts, ensuring easy updates and debugging.
- **Eco-friendliness:** Eliminates paper-based processes, thus contributing to environmental sustainability.
- **Longevity:** Requires minimal resources to maintain and runs on virtually any system with a shell environment, ensuring a long lifespan.

4.2 Project Management and Teamwork

Planning: Divided into modules like librarian and student functionalities, login systems, and book management.
Roles: Team members were assigned specific responsibilities (like, interface design, functionality testing).
Communication: Weekly meetings ensured progress tracking and timely resolution of issues.
Testing: Extensive testing by multiple users ensured the robustness of the system.

4.3 Complex Engineering Problem

4.3.1 Mapping of Program Outcome

The project maps directly to the following program outcomes:

- Problem Solving: Tackles the inefficiencies of traditional library systems.
- Design and Development: Implements a user-friendly interface using Zenity dialogs and shell scripting.
- Sustainability: Promotes eco-friendly practices by reducing paper usage.
- Ethical Practices: Ensures privacy and responsible data management.

Table 4.1: Justification of Program Outcomes

PO's	**Justification**
PO1	The project applies engineering principles to solve a real-world problem, such as book inventory management and transaction handling.
PO2	The project requires analyzing library workflows, identifying bottlenecks, and designing efficient algorithms for operations.
PO3	The project involves designing a scalable system that automates manual processes, ensuring accuracy, efficiency, and usability.

4.3.2 Complex Problem Solving

The Library Management System project involves addressing several complex problems that arise in the context of system design, data handling, and concurrency management. These problems were analyzed and solved using systematic engineering approaches, aligning with the Complex Problem Solving framework. The following table maps the project's challenges to key problem-solving categories, along with the rationale for each mapping.

EP Table: Mapping with Complex Problem Solving

EP1: Dept of Knowledge	EP2: Range of Conflicting Requirements	EP3: Depth of Analysis	EP4: Familiarity of Issues	EP5: Extent of Applicable Codes	EP6: Extent of Stakeholder Involvement	EP7: Interdependene
✔	✔	✔	✔	✔		✔

Table 4.2: Mapping with complex problem solving.

4.3.3 Engineering Activities

Requirement Analysis: Identified the key needs of librarians and students, including book tracking and login functionalities.
System Design: Developed modules (e.g., book management, borrowing, and returning) using Zenity for user interaction.
Implementation: Created scripts ensuring seamless interaction between modules.
Testing and Debugging: Conducted rigorous tests to resolve errors and ensure smooth performance.
Documentation: Prepared clear documentation for future enhancements or user support.

Table 4.3: Mapping with complex engineering activities.

EA1: Range of Resources	EA2: Level of Interaction	EA3: Innovation	EA4: Consequences for Society and Environment	EA5: Familiarity
✔	✔	✔	✔	✔

Chapter 5

Conclusion

The Shell-Based LMS proves to be a reliable and efficient tool for library management, balancing functionality and simplicity. While current limitations include scalability and advanced user roles, proposed future enhancements like database integration, graphical UI upgrades, and automated features will address these gaps. The system provides a foundation for libraries seeking an affordable, efficient, and customizable management solution, with potential for significant future development.

5.1 Summary

The Shell-Based Library Management System successfully addresses the challenges of managing library operations by leveraging shell scripting for its back end and integrating a user-friendly interface for interaction. The system performs essential tasks, including book management, user account handling, and transaction tracking, with high efficiency. The inclusion of a graphical menu-based user interface ensures accessibility for non-technical users, while the modularity of the scripts allows easy customization and scalability. By using lightweight technologies, this system is particularly suited for small to medium-sized libraries with limited resources. Its simplicity, combined with robust functionality, makes it a practical solution for streamlining library operations. This project demonstrates the versatility of Unix/Linux systems and their ability to solve real-world problems effectively.

5.2 Limitation

Despite its efficacy, the system has some drawbacks that must be fixed for wider use. The inability to provide concurrent multi-user access is one major drawback. Due to the file-based technique used by the system, concurrent operations may result in inconsistent data, particularly in larger libraries with increased utilization. Scalability is another drawback; although effective for small libraries, the system may experience performance problems when managing extremely big databases, like thousands of books or people. Furthermore, even though the CLI-based user interface is effective, users who are not familiar with command-line operations could find it confusing. Finally, its usefulness is limited by the lack of sophisticated features like role-based access control, automatic notifications, and interaction with third-party tools like barcode scanners or data export capabilities.

5.3 Future Work

Utilizing all the necessary features there always has a corner to improve .So in here we will incorporate how we want to present our system and what extra features will be added to our system. The future integration are given below:

- **Database Integration:** For improved scalability and multi-user support, swap out the file-based system for a database such as SQLite or MySQL.
- **Advanced User Interface**: For better accessibility, switch to a graphical or web-based user interface (UI) using Flask or Tkinter.
- **Role-Based Access Control:** To improve security, provide users, employees, and librarians permissions.
- **Automated Notifications:** Include email or SMS notifications for books that are past due and impending deadlines.

- **Enhanced Search Filters:** Make it possible to filter by publication year, genre, author, and other parameters.
- **Integration with External Tools:** Enable data export in CSV/Excel formats and support barcode scanners for book check-ins and check-outs.
- **Data Security and Backups:** Automate routine backup procedures and encrypt critical data.
- **User Feedback Mechanism**: Incorporate feedback collection to guide future improvements.

References

[1] Jon Kleinberg and Eva Tardos, Algorithm Design, Pearson Education, 2006.
[2] "Unix Shell Scripting," The Linux Documentation Project, 2021.
[3] M. J. Loui, Understanding Systems Design, 2nd ed. Prentice Hall, 2018.