**Table of Contents**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **Name** | **Page No.** |
|  | **ABSTRACT** | **3** |
|  | **ACKNOWLEDGEMENT** | **4** |
|  | **DECLARATION** | **5** |
|  | **Chapter 1: Introduction** | **8-12** |
| 1.1 | Background | 8 |
| 1.2 | Objectives | 8 |
| 1.3 | Purpose, Scope, Applicability | 9 |
| 1.3.1 | Purpose | 9 |
| 1.3.2 | Scope | 10 |
| 1.3.3 | Applications | 11 |
| 1.4 | Achievements | 11 |
|  | **Chapter 2: Survey of Technologies** | **13-15** |
|  | **Chapter 3: Requirements and Analysis** | **16-22** |
| 3.1 | Problem Definition | 16 |
| 3.2 | Requirement Specification | 16 |
| 3.2.1 | Functional Requirements | 17 |
| 3.2.2 | Non-Functional Requirements | 17 |
| 3.3 | Planning and Scheduling | 18 |
| 3.4 | Software and Hardware Requirement | 20 |
| 3.5 | Preliminary Product description | 20 |
| 3.6 | Modules Description | 20 |
| 3.7 | Feasibility Study | 21 |
|  | **Chapter 4: System Design** | **23-31** |
| 4.1 | UML Diagrams | 23 |
| 4.1.1 | Data Flow Diagram | 23 |
| 4.1.2 | Use Case Diagram | 24 |
| 4.1.3 | Sequence Diagram | 25 |
| 4.1.4 | Entity-Relationship Diagram | 26 |
| 4.1.5 | Class Diagram | 27 |
| 4.2 | Data Design | 28 |
| 4.2.1 | Schema Design | 28 |
| 4.3 | Procedural Design | 29 |
| 4.3.1 | Algorithm Design | 29 |
| 4.4 | Test Case Design | 30 |
|  | **Chapter 5: Implementation and Testing** | **32-43** |
| 5.1 | Implementation and Approaches | 32 |
| 5.2 | Coding details and efficiency | 32 |
| 5.2.1 | Code Efficiency | 32 |
| 5.3 | Testing Approach | 32 |
| 5.3.1 | Manual Testing | 33 |
| 5.3.2 | Functional Testing | 38 |
|  | **Chapter 6: Result and Discussion** | **44-54** |
| 6.1 | Test Reports | 44 |
| 6.2 | User Documentation | 45 |
|  | **Chapter 7: Conclusion** | **55** |
| 7.1 | Conclusion | 55 |
| 7.2 | Future Scope of the project | 55 |
|  | **References** | **55** |

**List of Figures**

|  |  |  |
| --- | --- | --- |
| **Fig No.** | **Figure Name** | **Page No.** |
| Fig 1 | Features of online library management system | 9 |
| Fig 2 | Existing library | 10 |
| Fig 3,4 | Concept of Online Library | 12 |
| Fig 5 | Waterfall model | 19 |
| Fig 6 | Gantt chart | 19 |

# Chapter 1: Introduction

The Book Buffet is a software application developed to automate the process of managing a library. It provides an efficient and convenient way to handle various library tasks such as book cataloguing, member management, circulation of books, and generating reports. The system allows admin to maintain an organized record of books, track book availability, handle borrower information, etc.

## 1.1 Background

The library has grown immensely with collections expanded from 2000 to more than 30,000 volumes of books, e-books, as well as e-journals. Despite having come this far, their services have not yet fully automated their services hence a need to replace the system with a fully automated online system that can be accessed by users remotely, from the comfort of their locations.

Moreover, “Book Buffet” is designed to carry out operations in a smooth and efficient manner. No formal knowledge is needed for the user to use this system. Thus, it proves user friendly.

Overall, the Book Buffet simplifies and streamlines library operations, enabling librarians to efficiently manage books, borrowers, and administrative tasks. It enhances the user experience for both librarians and borrowers, resulting in improved library services and a well-organized library environment.

**Problems faced by Existing system:**

* When a user requests the book, one must physically check for the presence of a book in the library.
* Lack of security of data.
* Daily keeping a manual record of changes taking place in the library such as: books being issued, books being returned, etc. can become troublesome if the library size is bigger.

**Proposed system:**

* **Saves cost**: After the computerized system is implemented fewer human resources will be required to maintain the library thus reducing the overall cost.
* **Saves time:** Librarian can search record by using few clicks of mouse and few search keywords thus saving his valuable time.
* Greater efficiency and data security.
* User friendliness and interactive.

**1.2 OBJECTIVES:**

1. It excludes the use of paperwork by managing all the records electronically.
2. It saves human efforts and resources.
3. Cost-effectiveness: It eliminates the need for physical infrastructure, such as buildings and maintenance costs, and reduces expenses associated with printing.
4. It reduces manual paperwork, saves time, and enables librarians to focus on more value-added services.
5. It provides a centralized database where librarians can enter, update, and retrieve information about items in the library's collection.

The objectives of a Book Buffet revolve around improving efficiency, enhancing user experience, optimizing resource management, and leveraging technology to streamline library operations and services.

### 1.3 Purpose, Scope, and Applicability

#### 1.3.1 PURPOSE

In the modern age, libraries are suffering from many problems including a lack of space, ineffective staff, and improper management. In this cut-throat competitive world, the education system presents far more challenges and demands than ever before. In such a dynamic and volatile environment, the libraries, especially college and university libraries must learn to cater to such ever-changing demands.

Therefore, keeping in sync with the technological evolution around it, it is time for libraries to reduce their dependency on manual work and make avail of computers and sophisticated technology for its management and services.

The purpose of Book Buffet is to manage & track the daily work of the library such as issuing books, return books, due calculations, etc. and to efficiently manage and organize various aspects of library operations.

Another purpose is to operate a library with efficiency and at reduced costs. The system being entirely automated streamlines all the tasks involved in the operation of the library. The activities of book purchasing, cataloguing, indexing, circulation recording, and stock checking are done by the software. Such software eliminates the need for repetitive manual work and minimizes the chances of errors.

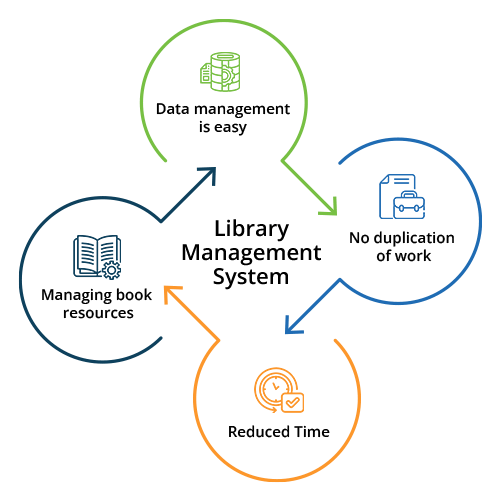


Fig 1: Features of online library management system

It helps in reducing operational costs. Managing a library manually is labor intensive and an immense amount of paperwork is involved. An automated system reduces the need for manpower and stationery. This leads to lower operational costs. The system saves time for both the user and the librarian. With just a click the user can search for the books available in the library. The librarian can answer queries with ease regarding the availability of books.

Adding, removing or editing the database is a simple process. Adding new members or cancelling existing memberships can be done with ease.

Stock checking and verification of books in the library can be done within a few hours. The automated system saves a considerable amount of time as opposed to the manual system. The Book Buffet software makes the library a smart one by organizing the books systematically by author, title and price. This enables admin to search for books quickly and effortlessly.



Fig 2: Existing library

#### 1.3.2 SCOPE

The scope of Book Buffet encompasses various functional and technical aspects that need to be considered during its development and implementation. The scope defines the boundaries and features of the system. Here are some key elements within the scope of Book Buffet:

* To utilize resources in an efficient manner by increasing productivity through automation.
* The system generates types of information that can be used for various purposes.
* It satisfies the user requirement.
* Easy to understand and operate for the user.
* Implement regular data backups and disaster recovery procedures to prevent data loss.
* Accessibility and Responsive Design: Design the system to be accessible to users with disabilities. Ensure the system's interface is responsive and usable on various devices.
* Resource Cataloguing and Classification: Add, edit, and delete library resources (books, journals, digital materials, etc.).
* Search and Discovery: Provide a user-friendly search interface for patrons to find library resources.
* Circulation Management: Handle checkouts, returns, renewals, and holds for library materials. Maintain borrower information and manage due dates. Generate notifications and alerts for overdue items.

**1.3.3 APPLICATION**

**Applications of LMS:**

1. It can be used in offices and modifications can be easily done according to requirements.

1. **Corporate Libraries**: Organizations can manage their internal resources, documents, and training materials using an online library system, ensuring easy access for employees.

1. **Media Organizations:** Media companies can manage their collection of audio-visual content, scripts, and media-related resources.

1. **Educational Institutions:** Schools, colleges, and universities can use online library systems to manage their physical and digital resources for students, faculty, and researchers.

1. **Remote and Virtual Libraries**: Online library systems can be used to create virtual libraries accessible to users across geographic boundaries.

1. **Government Libraries:** Government agencies can use online library systems to organize and provide access to legislative documents, reports, and policy resources.

1. **Nonprofit Organizations:** Nonprofits can maintain a digital library of resources related to their cause, making information easily accessible to stakeholders and the public.

1. **Medical Institutions:** Medical libraries can manage medical literature, research papers, and clinical resources for healthcare professionals.

### 1.4 Achievements

The contributions of Book Buffet to the chosen area can be significant and far-reaching, depending on the specific context in which it is implemented. Here are some potential contributions that such a project can make to various application areas:

* **Educational Institutions:**
* Enhanced Learning: Students can access a wider range of resources, improving their research and learning experiences.

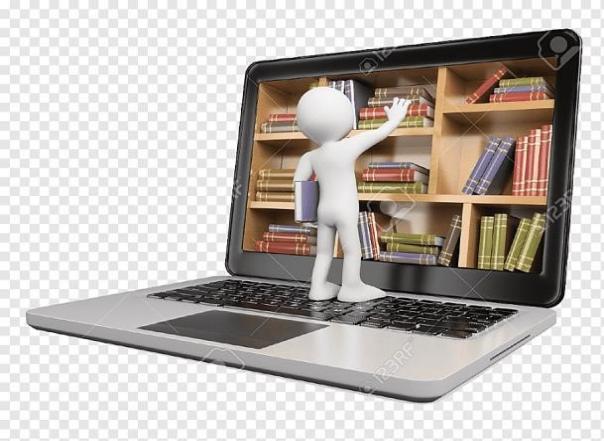
* **Public Libraries:**
* Increased Accessibility: Patrons can remotely access resources, expanding the library's reach to a broader community.
* Improved User Experience: Online catalogues and self-service features offer a convenient and user-friendly experience.

* **Corporate Libraries:**
* Efficient Resource Management: Employees can easily find and utilize internal documents and training materials.
* Knowledge Sharing: The system promotes collaboration and knowledge sharing within the organization.
* **Media Organizations:**
* Centralized Content Management: Media companies can efficiently manage and distribute their content and media assets.
* **Government Libraries:**
* Transparent Access: Citizens can easily access government documents and information, promoting transparency.
* Efficient Information Sharing: Government employees can access policy documents and resources more effectively.

* **Nonprofit Organizations:**
* Resource Dissemination: Nonprofits can share valuable information and resources related to their mission.
* Awareness and Advocacy: The system can help spread awareness and support for the organization's cause.

In essence, Book Buffet contributes by enhancing access, streamlining operations, facilitating collaboration, and providing valuable resources to users within their respective contexts.

The specific contributions may vary, but the overall goal is to improve the efficiency, effectiveness, and reach of library services in various domains.



**CHAPTER 2: SURVEY OF TECHNOLOGIES**

**Why Python?**

**Why would anyone choose python over any another language to implement this project?**

Choosing a programming language for implementing Book Buffet involves considering various factors such as the project's requirements, the developer's familiarity with the language, available libraries and frameworks, and the overall ecosystem. Here are some reasons why Python might be chosen for implementing this project:

1. **Readability and Ease of Development**: Python is known for its clear and concise syntax, which enhances code readability. This can lead to faster development and easier maintenance of the system.
2. **Vast Library Ecosystem:** Python boasts a rich ecosystem of libraries and frameworks that can expedite development.
3. **Cross-Platform Compatibility:** Python is platform-independent, meaning your application can run on various operating systems without significant modifications.
4. **Community and Resources:** Python has a large and active community, which means there are ample resources, tutorials, and forums available for troubleshooting and learning.
5. **Integration with Databases:** Python offers excellent support for various databases, both relational (e.g., PostgreSQL, MySQL) and NoSQL (e.g., MongoDB). This is important for managing and organizing the library's resources and user data.
6. **Open Source and Licensing:** Python is open source, which means there are no licensing costs associated with using it for your project. This can be advantageous for organizations with budget constraints.

**Why tkinter module?**

Benefits of using tkinter module of python for this project are as follows:

1. **Native Look and Feel:** tkinterprovides a native look and feel on various platforms, which can make the application interface familiar and comfortable for users.
2. **Availability:** tkinter is a standard library module that comes bundled with Python, which means you don't need to install any additional dependencies. This can simplify deployment and reduce potential compatibility issues.
3. **Cross-Platform:** tkinter applications can run on different operating systems without major modifications. This cross-platform compatibility is essential for reaching a wide range of users.

1. **Customizable Widgets:** tkinter offers a variety of customizable widgets like buttons, labels, text fields, and more, allowing you to create a user-friendly interface for your library management system.
2. **Responsive and Lightweight:** tkinter applications are generally responsive and lightweight, suitable for creating efficient user interfaces.
3. **Integration with Python:** Since tkinter is a Python library, you can easily integrate it with the rest of your Python codebase. This can streamline the development process.
4. **Flexibility:** While tkinter might be considered a simpler GUI toolkit, it still provides flexibility for creating complex and interactive user interfaces.
5. **Community and Documentation:** tkinterhas an active community and a wealth of online resources and tutorials. This can be beneficial for getting help and finding solutions to common problems.

Among various GUI Frameworks, Tkinter is the only framework that is built-in into Python's Standard Library**.**

**To create a tkinter app:**

* Importing the module – tkinter
* Create the main window (container)
* Add any number of widgets to the main window
* Apply the event Trigger on the widgets.

Importing tkinter is same as importing any other module in the Python code.

Pythonwith Tkinter provides a **faster** and more **efficient way** to build useful desktop applications that would have taken much time if you had to program directly in C/C++ with the help of native OS system libraries.

**DATABASE USED**

**What Is SQLite?**

* SQLite is an embedded, server-less relational database management system.
* It is an in-memory open-source library with zero configuration and does not require any installation.
* Also, it is very convenient as it’s less than 500kb in size, which is significantly lesser than other database management systems.
* SQLite is a lightweight, server-less, and self-contained relational database management system that is often used for smaller-scale applications, prototypes, or projects with modest data storage requirements.

**Why SQLite?**

There are several reasons why you might choose to use SQLite in your project:

1. **Ease of use:** SQLite is very easy to get started with, as it requires no setup or configuration. You can simply include the library in your project and start using it.

1. **Lightweight:** SQLite is a very lightweight database engine, with a small library size (typically less than 1MB). This makes it well-suited for use in applications where the database is embedded directly into the application binary, such as mobile apps.

1. **Server-less:** As mentioned earlier, SQLite is a server-less database engine, which means there is no need to set up and maintain a separate database server process. This makes it easy to deploy and manage, as there are no additional dependencies to worry about.

1. **Cross-platform:** SQLite is available on many platforms, including Linux, MacOS, and Windows, making it a good choice for cross-platform development.

1. **Standalone:** SQLite stores all the data in a single file on the filesystem, which makes it easy to copy or backup the database.

1. **High reliability:** SQLite has been widely tested and used in production systems for many years and has a reputation for being a reliable and robust database engine.

1. **Integration with Python: SQLite** can be easily integrated with Python using the built-in sqlite3 module. This module provides functions and methods to execute SQL queries, manage connections, and work with the SQLite database.

1. **Security Considerations: Implement** proper access controls and authentication mechanisms to protect the SQLite database from unauthorized access.

**CHAPTER 3: REQUIREMENTS AND ANALYSIS**

## 3.1 Problem Definition

The problem definition for implementing Book Buffet involves identifying the challenges and needs that the system aims to address.

Libraries are essential repositories of knowledge, offering resources and services to diverse user communities. However, traditional library operations often face inefficiencies and limitations, such as manual cataloguing, slow circulation processes, and restricted access to physical resources.

**Challenges faces by existing system:**

* **Inefficient Resource Management:** Manual cataloguing and tracking of library resources lead to errors, duplication, and difficulties in locating materials.
* **Limited Accessibility:** Patrons face barriers in accessing library resources due to geographical constraints, limited opening hours, and physical presence requirements.
* **Slow Circulation Processes:** Manual checkout, return, and renewal processes result in delays and inconvenience for both patrons and library staff.
* **Lack of Real-time Information:** Patrons lack real-time information about resource availability, due dates, and holds, leading to confusion and inefficiency.
* **Data Discrepancies:** Manual data entry increases the likelihood of discrepancies in borrower records, fines, and transaction histories.

To overcome these challenges and enhance the overall library experience, the problem at hand is the implementation of a comprehensive online library management system.

**DRAWBACKS OF EXISTING SYSTEM**

After studying the existing system, we come across certain drawbacks, which are discussed below: -

1. The records stored manually can be altered by an unauthorized person.
2. Searching for any book is difficult, since all the compartments are to be checked.
3. Generation of fine on late return is difficult to calculate.
4. Problem arises if any of the records get misplaced.
5. Overload on staff members.

## 3.2 Requirements Specification (SRS)

Requirement specification for Book Buffet outlines the functional and non-functional requirements that the system should meet. Here's a possible requirement specification for the project: -

**3.2.1 Functional Requirements:**

1. **Login Authentication**

* Only the authorized person should be able to login with his login credentials.
* No other unauthenticated person should be able to access any data.

1. **Resource Management: -**

* Admin should be able to add, edit, and delete resources.
* Resources should have attributes such as title, author, Book ID, price, and availability

1. **Search and Discovery: -**

* Admin should be able to search for resources using keywords like Book ID.
* Search results should display detailed resource information, including availability status.

1. **Circulation and Transactions: -**

* Admin should be able to issue, renew, and return resources.
* The system should calculate and apply fines for overdue resources.

1. **Reporting and Analytics: -**

* Generate reports on resource circulation, user activity and fines.
* Provide visualizations to facilitate data analysis and decision-making.

**3.2.2 Non-Functional Requirements**:

1. **Usability and User Experience: -**

* The system should have an intuitive and user-friendly interface.
* Response times for search queries and transactions should be fast.

2. **Security and Privacy: -**

* User data, including personal information and borrowing history, should be securely stored and transmitted.
* Access controls should be implemented to ensure data confidentiality.

3. **Scalability: -**

* The system should be able to handle increasing numbers of users and resources without significant performance degradation.

4. **Integration and Compatibility: -**

* The system should integrate with existing library systems and external services (e.g., authentication, payment gateways).

**5. Data Backup and Recovery: -**

* Regular automated backups of the database should be performed to prevent data loss.
* Mechanisms for data recovery in case of system failures should be in place.

**6. Performance and Reliability: -**

* The system should exhibit stable performance even during peak usage.
* Downtime for maintenance should be minimal and planned.

**Constraints:**

* 1. The system should be implemented using Python and the `tkinter` library for the user interface.
  2. The system's budget and resources should align with the project's scope and requirements.

This requirement specification serves as a comprehensive guide to ensure that the Book Buffet is designed and developed to meet the needs of both library staff and users, while also considering technical and operational constraints.

## 3.3 Planning and Scheduling

Planning and scheduling are an important part of project development. Planning makes the work easy to handle and scheduling makes how the project will go, how much time it will take to complete the project. Planning and Scheduling is just an estimation which will play an important role in project development.

My first task in project planning was the Project Initiation in which I had defined project objectives and scope.

Next, for Requirement Analysis, I reviewed the problem definition and requirement specification. Break down requirements into detailed user stories or use cases. Prioritize features based on their importance and impact.

After that for the System Design I defined the technology stack, including programming languages, frameworks, and libraries.

Later for the Development and testing I will begin coding the various components of the system, following the design specifications.

Next, I will implement database tables, queries, and interactions using SQLite.

Conduct testing for individual components to ensure they function as intended.

❖ **WATERFALL MODEL**

The Waterfall model is a traditional software development approach that follows a linear and sequential progression through different phases of the project. Here's how the Waterfall model can be applied to the implementation of Book Buffet:

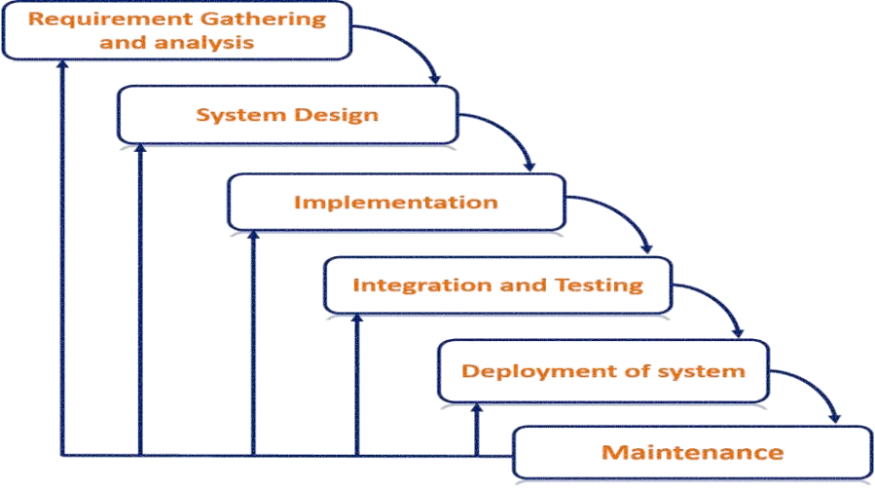


Fig 5: Waterfall model

**Advantages of the Waterfall Model in implementing library system:**

**Clear Documentation**: The Waterfall model emphasizes thorough documentation, which is essential for a complex project like an online library management system.

**Structured Approach:** The linear progression through phases ensures a systematic approach to development, allowing for better control and management.

**Detailed Planning:** The requirements and design phases allow for detailed planning of the system's functionalities and user interactions.

❖ **GANTT CHART**

A Gantt chart may be a sort of chart that illustrates a project schedule. This chart lists the tasks to be performed on the vertical axis, and time intervals on the horizontal axis. The width of the horizontal bars within the graph shows the duration of every activity. Gantt charts are widely used in project management to communicate project plans and progress. Gantt charts are valuable for project management because they provide a clear overview of the project's timeline, tasks, and progress. They aid in planning, coordination, communication, and identifying potential bottlenecks or delays in the project.

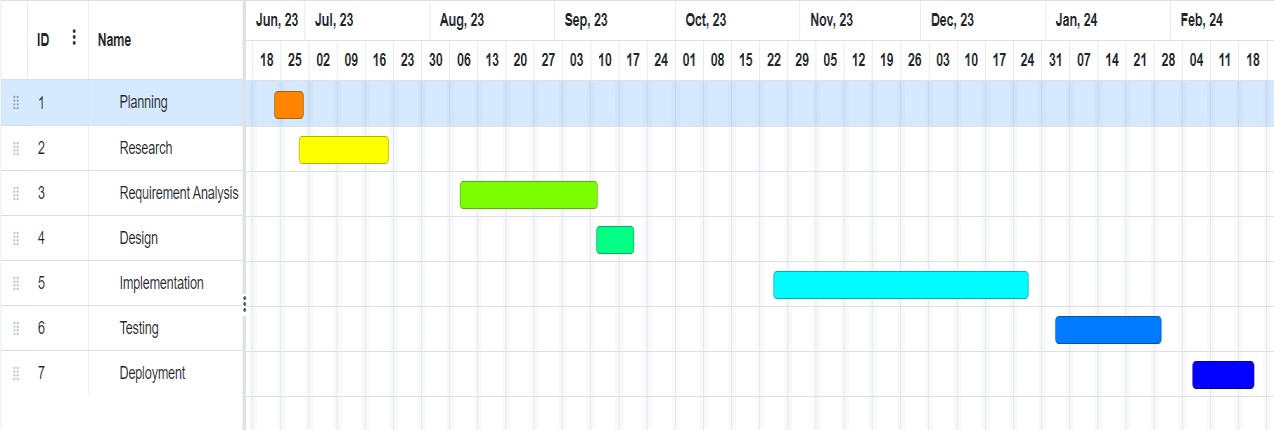


Fig 6. Gantt chart

## 3.4 Software and Hardware Requirements

**3.4.1** **Hardware Requirements: -**

Hardware is a term that refers to all the physical parts that make up a computer.

The internal hardware devices that make up the computer. Various devices which are essential to form hardware are called components.

Following are the hardware specifications that are required at client side to implement this project:

* A personal laptop or a desktop computer is needed.
* RAM: 1.00 GB or higher for smooth working of application.
* System type: 64-bit operating system, x64-based processor
* Hard Disk: 120 GB hard disk or above.

**3.4.2** **Software Requirements: -**

Computer software, or simply software, is a collection of data or computer instructions that tell the computer how to work. This contrasts with physical hardware, from which the system is built and performs the work.

Following are the software requirements

* Operating System: Windows 7 or higher.

## 3.5 Preliminary Product Description

The system is only designed for librarians, which is the ADMIN in this case. The system will ask for the Login as Admin. After correctly putting the login credentials the system will jump on the Dashboard Page where there are multiple options like add, update, delete, issue or search a book. Further the Admin can also add new members, check if any are overdue, or add books which are being returned by the library members. At a time, the admin can perform only one operation. After he has finished doing his work, he can easily log out of the system.

## 3.6 Modules Description

1. **Book Module:** Admin has the authority to add, modify, and delete books from the library database. Each book entry includes details such as title, author, ID, number of copies available.

1. **Search Module**: The system provides a search feature to locate books based on book ID. That means it indicates the availability status of each book, helping librarians determine whether a book is currently available or not.

1. **Login Module**: The system includes a login system to ensure only authorized librarians can access and modify library data.

1. **Book Issuance and Return Module**: The system facilitates the borrowing and returning of books. Librarians can check the availability of a book, issue it to a member,

and update the status accordingly. Similarly, when a member returns a book, the system updates the availability status and makes the book available for other borrowers.

1. **New Member:** The system allows librarians to maintain a list of registered members. Librarians can add new members, update their information.

1. **Report Module:** Librarians can generate reports on book inventory, borrower details, fine collection, and other relevant statistics. These reports provide insights into library operations and aid in decision-making.

## 3.7 Feasibility Analysis

A system undergoes several feasibility analyses to ascertain if it can be acquired and eventually adopted. It involves a continuous process that looks at the operational, technical, economic, cultural, legal, and schedule feasibility of a system among other factors.

**3.7.1** **Operational feasibility**

Operational feasibility evaluates whether a system is relevant to operating in a particular environment.

The Book Buffet presents several features making it operationally feasible. Reports are generated automatically by a function that retrieves relevant information from the database and presents them to a user in organized format. The system is easy to use and navigate, hence enables any user with minimal computer skills to use it.

The system has incorporated the use of security features and access levels that allow only authorized users to login. The system is navigable since it presents users with several options to click on and accomplish their functions. It presents an easier way to capture user input for eventual processing. End users don’t need to have technical skills to use the system.

**3.7.2** **Technical feasibility**

The system is technically feasible in several ways. It is developed using several readily available web development tools. Coding is done using Python Programming language. The system made use of SQLite database which is open source and cross platform independent.

**3.7.3** **Cultural Feasibility**

Cultural feasibility looks at the feelings of the system end users. Several issues raised by the end users were put into consideration while designing this system.

The system is culturally feasible in several ways. The end user (here in this case is the admin) can access the system using their respective unique password and user ID.

The users have come to appreciate the system due to its attractive and user-friendly nature. The system grants more access privileges to the admin unlike other users of the system due to his administrative nature of tasks.

**3.7.4**  **Legal Feasibility**

This system has minimal licensing issues since it was developed using open-source software. This system is meant to operate in the library, hence it’s subject to the rules and regulations governing the institution.

**3.7.5** **Economic Feasibility**

This feasibility looks at the costs incurred in operating the system versus the revenue and accumulated returns. Library Book Management Systems is economically feasible in the sense that; it’s cheaper to acquire, it saves on the cost spend on paper since it operates automatically, the number of employees needed to operate the system is reduced hence lowering labour costs, and it saves time by facilitating quicker services to the end users.

**CHAPTER 4: SYSTEM DESIGN**

**4.1** **UML Diagrams:**

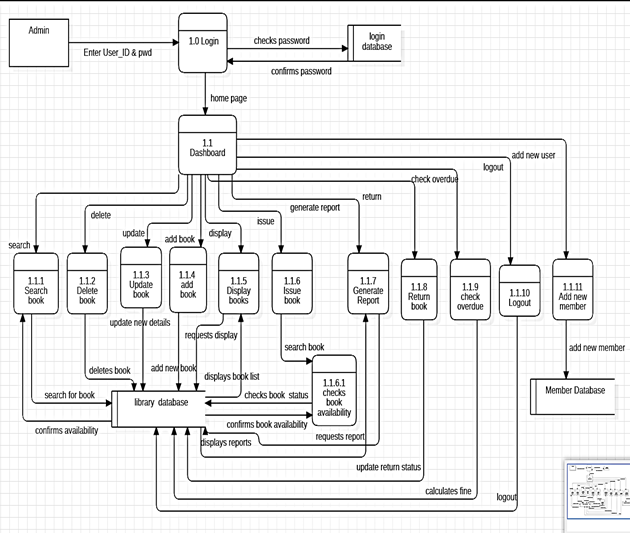
### 4.1.1 Data Flow Diagram

A Data Flow Diagram (DFD) is a graphical representation of the flow of data within a system or process. It illustrates how data is input, processed, stored, and output within the system. DFD consists of several components that together illustrate how data flows within a system or process. These components help in visually representing the flow of data, processes, data stores, and external entities. Here are the main components of a DFD:

* **External Entities:** These entities interact with the system and can be sources or destinations of data. In Book Buffet, the external entity is the Admin.
* **Processes:** Processes are activities or functions within the system that manipulate data. They represent an operation performed on data. In our system, processes include checking book availability, checking overdue, adding new members, generating reports, etc.
* **Data Flows:** Data flows represent the movement of data between external entities, processes, and data stores. They are typically represented by arrows and show the direction of data flow. Data flows indicate what data is transmitted from one component to another.
* **Data Stores:** Data stores represent repositories where data is stored and retrieved. They can include databases, files, or any storage mechanism. In our system, data stores include Library Database (for book related transactions) and Login Database (for storing Login credentials).
* **Data Flow Labels:** Data flow labels provide information about the data being transmitted between components. They describe the content of the data and may include data names or descriptions.

Thus, DFDs are a valuable tool for system analysis and design, as they provide a visual representation of how data moves through a system and help in understanding its functionality and interactions.

In the context of Book Buffet, following is the simple DFD to represent the key data flows and processes:



### 4.1.2 Use Case Diagram

A Use Case Diagram is a visual representation of the interactions between various actors (users) and the system under consideration.

Following are the **purposes** of a use case diagram given below:

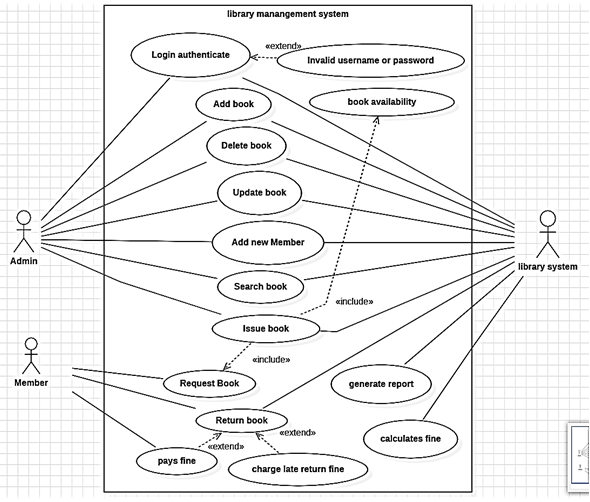
* It gathers the system's needs.
* It depicts the external view of the system.
* It recognizes the internal as well as external factors that influence the system.
* It represents the interaction between the actors.

Following are the **components** of use case diagram:

1. **Actors:** The users that interact with a system. They must be external objects that produce or consume data. Here, Admin, Library system and Member represent actors.
2. **Use cases:** Horizontally shaped ovals that represent the different uses that a user might have.
3. **Associations:** A line between actors and use cases. In complex diagrams, it is important to know which actors are associated with which use cases.
4. **System boundary boxes**: A box that sets a system scope to use cases. All use cases outside the box would be considered outside the scope of that system.

1. **Relationships:**
   * **Include Relationship: This** relationship indicates that one use case includes the functionality of another use case.
   * **Extend Relationship: This** relationship indicates optional or conditional behaviour that extends the basic functionality of a use case.

In the context of Book Buffet, following is a Use Case Diagram:



### 4.1.3 Sequence Diagram

A Sequence Diagram is a type of UML diagram that illustrates the interactions and the order of messages exchanged between objects or components in a system over a specific period. It portrays the communication between any two lifelines as a time-ordered sequence of events, such that these lifelines took part at the run time.

1. **Lifeline**

* A lifeline represents an individual participant in the Interaction.

1. **Activations**

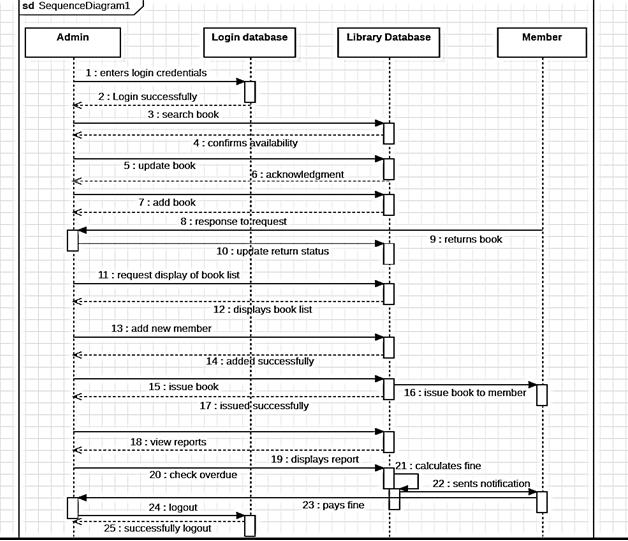
* A thin rectangle on a lifeline represents the period during which an element is performing an operation.
* Represents the time needed for an object to complete a task.

1. **Call Message**

* A message defines a particular communication between Lifelines of an Interaction.

1. **Reply Message –**

* Reply messages are used to show the message being sent from the receiver to the sender. We represent a return/reply message using an open arrowhead with a dotted line.

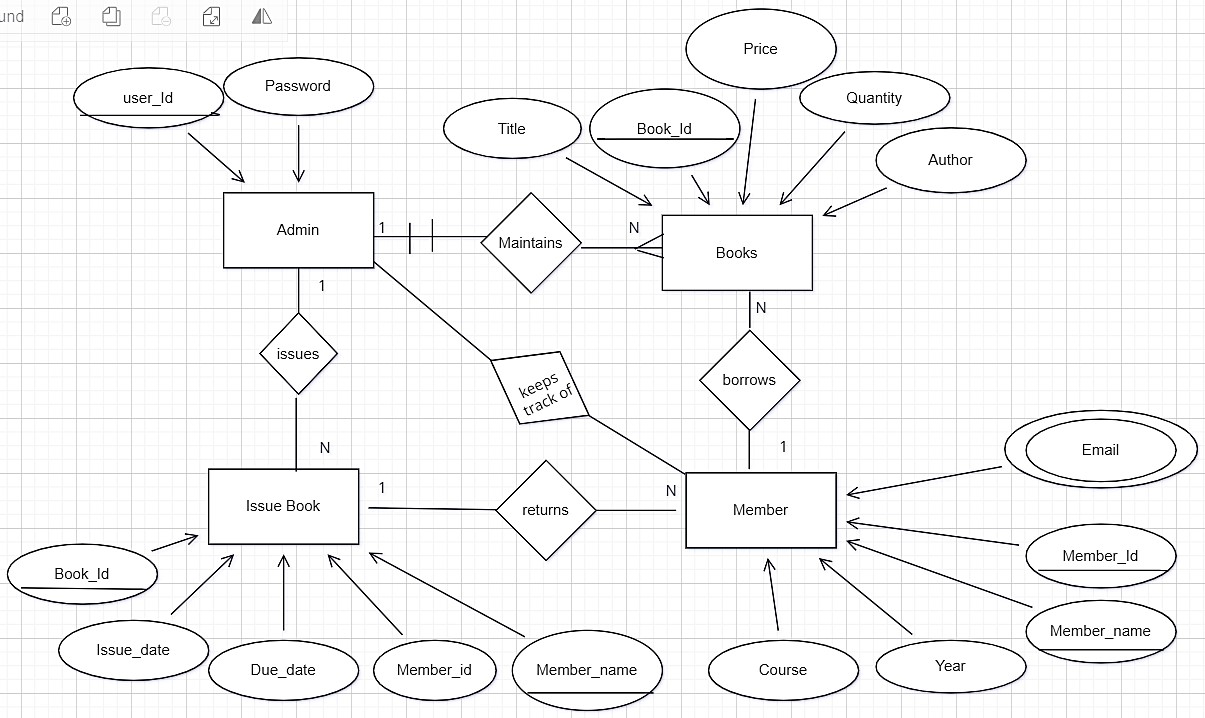


### 4.1.4Entity-Relationship Diagram

ER Diagram is known as Entity-Relationship Diagram, it is used to analyse the structure of the Database. It shows relationships between entities and their attributes. An ER Model provides a means of communication.

The Book Buffet database keeps track of readers with the following considerations –

* The system keeps track of the admin with a single point authentication system comprising login Id and password.
* Admin maintains the book catalogue with its Book ID, Book title, price, author name, and the quantity available.
* Members are registered with their name, unique ID, email, year of studying in the college (1st/2nd/3rd) and the course which they are enrolled in. The admin keeps track of these members.
* Members can return/issue books that stamps with their name & ID, Book ID, issue date and return date. If not returned within the prescribed time, it may have a due date too.



### 4.1.5Class Diagram

In Object-Oriented modelling, the main building block generally represents different objects in a system, their attributes, their different functions, and relationships among objects.

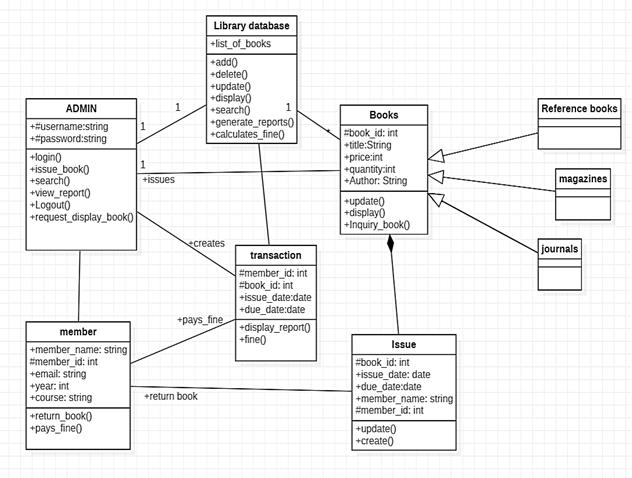
These building blocks are known as **Class Diagram**.

Class diagrams are generally used for conceptual modelling of static view of a software application, and for modelling translating models into programming code in a detailed manner. At time of developing or construction software systems, a class diagram is widely used. They are also used for data modelling.

It is used to show classes, relationships among them, interface, association, etc. Class in a class diagram simply is a blueprint of an object. It simply describes and explains different types of objects in a system, and different types of relationships that exist between them.

**Interactions:** The term "interactions" refers to the various relationships and links that can exist in class and object diagrams. Some of the most common interactions include:

* **Inheritance:** The process of a child or sub-class taking on the functionality of a parent or superclass, also known as **generalization**. Here, the objects reference book, magazines, journals would inherit all the attributes of Books like (title, price, author, id) and methods (create, update) of the parent class (Books).
* **Bidirectional association:** The default relationship between two classes. Both classes are aware of each other and their relationship with the other. This association is represented by a straight line between two classes. (Here, class Admin and class Books are inter-related)
* **Composition:** It portrays the dependency between the parent and its child, which means if one part is deleted, then the other part also gets discarded.



### 4.2 Data Design

#### 4.2.1 Schema Design

* + **For Admin Login**

**Table Name: UserLogin**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Datatype** | **Constraints** | **Null** | **size** |
| User\_id | varchar | Primary Key | Not Null | 15 |
| Password | varchar | Unique | Not Null | 20 |

* + **For Books**

**Table Name: Book**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Datatype** | **Constraints** | **Null** | **size** |
| Book\_Id | varchar | Primary Key Unique | Not Null | 10 |
| Title | varchar |  | Not Null | 30 |
| Price | integer |  | Not Null | 4 |
| Author | varchar |  | Not Null | 30 |
| Quantity | integer |  | Not Null |  |

* + **For member**

**Table Name: member**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Datatype** | **Constraints** | **Null** | **size** |
| Member\_Id | integer | Primary Key Unique | Not Null | 10 |
| Member\_Name | varchar |  |  | 20 |
| Email | varchar |  |  | 35 |
| Year | integer |  |  | 4 |
| course | varchar |  |  | 15 |

* + **To keep track of expected return date**

**Table name: Expected\_Return\_Dates**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Datatype** | **Constraints** | **Null** | **Size** |
| MemberID | integer | Primary Key Unique | Not Null | 10 |
| BookID | varchar |  | Not Null | 10 |
| ExpectedReturnDate | varchar |  |  |  |

* + **Issue table to keep track of books issued**

**Table Name: issue**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field Name** | **Datatype** | **Constraints** | **Null** | **size** |
| Member\_Id | integer | Primary Key Unique | Not Null | 10 |
| Book\_Id | varchar | Primary Key Unique | Not Null | 10 |
| Issue\_date | Date |  |  |  |
| Member\_Name | varchar |  |  | 20 |

**4.3 Procedural Design**

Procedural design for Book Buffet involves breaking down the system's functionality into procedures or functions that perform specific tasks. These procedures are organized in a structured manner to ensure efficient and maintainable code.

#### 4.3.1 Algorithms Design

Designing an algorithm for Book Buffet involves creating a set of step-by step instructions to perform specific tasks or operations within the system. Below, I'll outline a high-level algorithm for one common task: issuing a book from the library. This algorithm assumes a simplified scenario and may need to be adapted to user’s specific system and requirements.

* **Algorithm for Issuing a Book:**

**Input data:**

* User ID and password (Library Admin)
* Book ID or Title

**Output data:**

* Member’s Account &the changes reflected in library database

* **Steps:**

**1. Prompt for User Identification:**

* Request the user (Library Admin) to enter their ID or login credentials to verify their identity.
* Authenticate the user by checking their credentials against the user database.

2. **Search for the Book:**

* Prompt the user to enter the ID or title of the book they wish to search for information about.
* Search the library's resource database to find the book based on the information provided.

**3. Check Book Availability:**

- Verify if the book is available for issuing by checking its availability status.

**4. Record the Transaction:**

* Create a new issuing transaction record in the library's database.
* Associate the transaction with the member and the issued book.
* Update the book's available quantity and include the due date for return.

**5. Update Member's Account:**

- Display the issued book against the user's account, indicating that they have borrowed this book.

**6. Complete Transaction:**

* Conclude the issuing process.

* **Exception Handling:**

* If the user is not authenticated, display an error message and return to step 1.
* If the book is not found in the library database, inform the user by displaying an appropriate message. - Handle any database errors or system failures gracefully, providing appropriate error messages.

This algorithm is a simplified representation of the book borrowing process in a library management system.

### 4.4 Test Cases Design

Test case design is the process of creating specific scenarios or test cases that test the functionality, features, and behaviours of a software system.

These test cases are designed to verify that the software behaves correctly and meets its specified requirements. Test case design is a critical part of software testing and quality assurance, and it helps ensure the reliability and correctness of a software application.

Test case design for Book Buffet involves creating a set of test scenarios to ensure that the system functions correctly and meets its requirements. Test cases should cover various aspects of the system, including user interactions, data management, and error handling.

Below, I have outlined some test case scenarios for Book Buffet:

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No.** | **Test Case** | **Input** | **Expected Result** |
| 1. | User Authentication:  Test Case 1: Valid Admin Login | Valid username and password | Admin should be successfully logged in and directed to their dashboard. |
|  | Test Case 2: Invalid Admin Login | Incorrect username or password | Users should receive an error message indicating invalid credentials. |
| 2. | Add New Resource (Books) | Book details: title, author, id, etc.) | The resource should be successfully added to the library catalogue. |
| 3. | Edit book details | Book ID and updated  details | Resource details should be updated without errors. |
| 4. | Delete book | Book ID to be deleted | The resource should be removed from the catalogue. |
| 5. | Search for Book | Search query (e.g., book ID) | The system should return a list of matching resources. |
| 6. | Issue book | Book ID & Member ID | The book quantity and the member’s borrowing history should be updated. |
| 7. | Return book | Book ID & Member ID | The book quantity and the member’s borrowing history should be updated. |
| 8. | Generate Library  Report | Book ID | A report on library operations and book usage for the specified period should be generated. |
| 9. | Error Handling | Simulate various error scenarios (e.g., database connection failure, book not found) | The system should handle errors gracefully, providing appropriate error messages and not crashing. |
| 10. | Add new member | Member details (name,  ID, email, course, etc.) | Members should be successfully added to the member’s database. |

# CHAPTER 5: IMPLEMENTATION AND TESTING

**5.1 IMPLEMENTATION AND APPROACHES:**

A project implementation plan is a document that covers a combination of strategy, tactics, assignees, dependencies, and actionable steps required for completing a project. It involves directly managing a project to ensure it meets the objectives outlined in the planning phase. In other words, a project implementation plan will outline what is necessary to achieve a project goal.

Project implementation approaches refer to the methodologies or strategies used to execute and complete a project. The chosen approach depends on the nature of the project, its goals, and the organizational context. When selecting an implementation approach, it's crucial to consider the project's size, complexity, requirements, and the organization's culture and preferences.

**5.2 CODING DETAILS AND EFFICIENCY:**

**5.2.1 CODE EFFICIENCY:**

Writing efficient code is important for ensuring that software performs well, particularly as the scale and complexity of projects increase.

It refers to the measure of how well a program or piece of code utilizes system resources, such as processor time, memory, and disk space, to perform its intended tasks.

It involves optimizing the code to minimize resource consumption and improve overall performance. This can be achieved through techniques like algorithmic improvements, efficient data structures, minimizing redundant operations, and avoiding unnecessary resource allocation.

The goal of code efficiency is to ensure that the program executes quickly and effectively with minimal waste of system resources.

**5.3 TESING APPROACH:**

A testing approach refers to the systematic and planned way testing activities are conducted to evaluate the quality and functionality of a software application or system. There are various testing approaches that can be employed based on the project requirements and objectives.

## 5.3.1 Manual Testing

Manual testing refers to the process of manually verifying and validating software applications to ensure they behave as expected and meet the specified requirements. In manual testing, testers execute test cases without using any automation tools or scripts. Instead, they interact with the application's user interface, input data, and observe the outputs to identify defects or ensure the software's correctness.

**1. Add New Book:**

**Test Steps:**

* Manually navigate to the "Add New Book" feature on the dashboard page.
* Enter valid data into the input fields, such as book title, author, price etc.
* Submit the form and verify that the new book appears in the system.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test case**  **ID** | **Test Scenario** | **Test Case** | **Test Data** | **Expected Result** | **Actual Result** | **Status**  **(Pass/**  **Fail)** |
| T01 | User should be able to  **add new Book** in library database | If all the details are properly filled and user clicks on add button | Book ID,  Title,  Author,  Price and  Quantity | The new book’s details should be successfully added to the library’s catalogue and should prompt a success message as  “Book added successfully” | The new book’s details were successfully added to the library’s catalogue and the system prompted a success message as “Book added successfully” | Pass |
| T01 | User should be able to  **add new Book** in library database | If any of the field is empty and user clicks on add button | Book ID,  Title,  Author,  Price and  Quantity | The system should prompt a warning message as “Please fill out all the details” | The system prompted a warning message  to fill out all the  details | Pass |
| T01 | User should be able to  **add new Book** in library database | If user enters any character or a letter in the price or quantity field and clicks on add button | Price or Quantity | The system should prompt an error message as “Price and Quantity should be an integer value” | The system prompted an error message to enter an integer number in both price and quantity fields | Pass |
| T01 | User should be able to  **add new Book** in library database | If user enters the same Book ID which is already present in library’s database | Book ID | The system should prompt a warning message as “A book with same ID already exist” | The system prompted a warning message as “A book with same ID already exist” | Pass |

**2. Update Existing Book Details:**

**Test Steps:**

* Manually navigate to the "Update Book Details" feature.
* Select an existing book by entering a valid BookID and edit its details, such as title, author, or availability.
* Save the changes and verify that the updated book details are reflected in the system.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test case**  **ID** | **Test Scenario** | **Test Case** | **Test Data** | **Expected Result** | **Actual Result** | **Status**  **(Pass/**  **Fail)** |
| T02 | User should be able to **update**  **existing**  **book**  details | If user enters a valid book ID and clicks on submit button | Book ID | Details of that book should be fetched and display in a new window and, when click on save button it should prompt a success message as “Details updated successfully” | Details of that book were fetched and displayed in a new window and, when clicked on save  button it prompted a success message as “Details updated successfully | Pass |
| T02 | User should be  able to **update**  **existing book**  details | If user enters an Invalid book ID and clicks on submit button | Book ID | The system should prompt a warning message as “Book does not exist in the database” | The system prompted a warning message as “Book does not exist in the  database” | Pass |
| T02 | User should be able to **update existing**  **book**  details | If the field is  empty and user clicks on submit button | --------- | The system should prompt a warning message as “Please provide a value” | The system prompted a warning message as “Please provide a value” | Pass |

**3. Add New Member:**

**Test Steps:**

* Manually navigate to the "Add New Member" section.
* Enter valid member details, such as name, email, and year of studying, etc.
* Submit the form and verify that the new member is successfully added to the system.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test case**  **ID** | **Test Scenario** | **Test Case** | **Test Data** | **Expected Result** | **Actual Result** | **Status**  **(Pass/**  **Fail)** |
| T03 | User should be able to  **add new**  **members** in the library’s database | If all the details are properly filled and user clicks on save button | Member  ID,  Member name, Email ID, Year of studying, Course | The new member’s details should be successfully added to the member’s database and should prompt a success message as “Member added successfully” | The new member’s  details were successfully added to the member’s database and the system prompted a success message as “Member added successfully” | Pass |
| T03 | User should be able to  **add new**  **members** in the library’s database | If any of the field is  empty and user clicks on save button | Member  ID,  Member name, Email ID, Year of studying, Course | The system should prompt a warning message as “Please fill out all the details” | The system prompted a warning message to fill out all the details | Pass |
| T03 | User should be able to  **add new members** in the library’s database | If user enters any character or  letters in year and  member ID  fields and clicks on save button | Member  ID, Year of studying | The system should prompt an error message as “ID and year should be an integer value” | The system prompted an error message to enter an integer number in both id and year fields | Pass |
| T03 | User should be able to  **add new members**  in the library’s database | If user enters a number which is not in a range of 1 to 4 and click on save button | Year of studying | The system should prompt a warning message as “Year should be between 1 and  4” | The system prompted a warning message as  “Year should be between 1 and 4” | Pass |

**4. Delete Existing Book:**

**Test Steps:**

* Manually navigate to the "Delete Book" functionality.
* Select an existing book by entering a valid BookID and initiate the deletion process.
* Confirm the deletion and verify that the book is removed from the system.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test case**  **ID** | **Test Scenario** | **Test Case** | **Test Data** | **Expected Result** | **Actual Result** | **Status**  **(Pass/**  **Fail)** |
| T04 | User should be able to **delete existing book** in library | If user enters a  valid book ID and clicks on delete button | Book ID, User confirmation: Yes or No | System asks confirmation whether to delete that book. If yes, that book will be deleted permanently. If no, it should display info message as “Deletion cancelled” | System asks confirmation whether to delete that book. If clicked yes, that book will be deleted permanently. If no, it displays info message as “Deletion cancelled” | Pass |
| T04 | User should be able to **delete existing book** in library | If the field is  empty and user clicks on delete button | Book ID | The system should prompt a warning message as  “Please enter Book ID” | The system prompted a warning message as  “Please enter Book ID” | Pass |
| T04 | User should be able to **delete existing book** in library | If user enters an Invalid book ID and clicks on delete button | Book ID | The system should prompt a warning message as “Book does not exist in the database” | The system prompted a warning message as “Book does not exist in the  database” | Pass |

**5. Search All Details About Book:**

**Test Steps:**

* Manually access the search feature in the system.
* Enter relevant search criteria, such as book ID.
* Verify that the search results display accurate information about the book.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test case**  **ID** | **Test Scenario** | **Test Case** | **Test Data** | **Expected Result** | **Actual Result** | **Status**  **(Pass/**  **Fail)** |
| T05 | User should be able to **search all the details of the book** | If user enters a valid book ID and clicks on search button | Book ID | Details of that book should be fetched and display in a separate frame | Details of that book were fetched and displayed in a separate frame | Pass |
| T05 | User should be able to **search all the details of the book** | If the field is  empty and user clicks on search button | Book ID | The system should prompt a warning message  as “Please enter  Book ID” | The system prompted a warning message  as “Please enter  Book ID” | Pass |
| T05 | User should be able to **search all the details of the book** | If user enters an Invalid book ID and clicks on search button | Book ID | The system should prompt a warning message as “Book does not exist” | The system prompted a warning message as “Book does  not exist” | Pass |

* 1. **View Existing Members:**

**Test Steps:**

* Manually navigate to the "View Members" section.
* Verify that the list of existing members is displayed correctly, including their details.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test case**  **ID** | **Test Scenario** | **Test Case** | **Test Data** | **Expected Result** | **Actual Result** | **Status**  **(Pass/**  **Fail)** |
| T06 | User should be  able to **view existing**  **members**  of library | User should click on “Display existing  members” button | -------- | System should be able to fetch all the members details from the member’s database and display it in tabular format | System was able to fetch all the members details from the member’s database and displayed it in  tabular format | Pass |

**7. View Issued Books:**

**Test Steps:**

* Manually access the "Issued Books" section.
* Verify that the list of issued books is displayed accurately, including details about the borrowers.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test case**  **ID** | **Test Scenario** | **Test Case** | **Test Data** | **Expected Result** | **Actual Result** | **Status**  **(Pass/**  **Fail)** |
| T07 | User should be  able to **view issued books** | User should click on “Display issued books” button | ------ | System should be able to fetch all the issued book details from the book’s database and display it in tabular format | System was able to fetch all the issued book details from the book’s database and displayed it  in tabular format | Pass |

## 5.3.2 Functional Testing

It involves testing the software against its functional specifications or requirements to ensure that it behaves according to the intended functionality. The goal of functional testing is to validate that the software functions correctly from the end user's perspective and meets their requirements and expectations.

**1. Admin Login:**

* **Sub- Category:** Authentication Testing
* **Explanation:** Admin login involves verifying the authentication mechanism of the system. Functional Testing ensures that the login process works as expected, allowing authorized admins to access the system while preventing unauthorized access.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test case**  **ID** | **Test Scenario** | **Test Case** | **Test Data** | **Expected Result** | **Actual Result** | **Status**  **(Pass/**  **Fail)** |
| T1 | Verify **admin**  **login** | If both the fields are empty and user clicks login button | Username and Password | Display warning message to fill out required details. | Displays warning message as  “Please fill out required details” | Pass |
| T1 | Verify admin  login | If details filled are incorrect/ invalid | Username or  Password | Display Error message to enter correct credentials and clear both the  fields | Displays error message as “Please enter correct credentials and clears out both the fields | Pass |
| T1 | Verify admin  login | If details are correctly filled | Username and password | Display success message and jumps to dashboard page | Display success message as  “Login Successfully” and jumps to dashboard page | Pass |

**2. Logout from Dashboard Page:**

* **Sub- Category:** Session Management Testing
* **Explanation:** Session Management Testing ensures that user sessions are managed correctly within the system. Testing the logout functionality involves verifying that users can successfully end their session and return to the login page securely.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test case**  **ID** | **Test Scenario** | **Test Case** | **Test Data** | **Expected Result** | **Actual Result** | **Status**  **(Pass/**  **Fail)** |
| T2 | To **Logout from** the dashboard page | Ask confirmation whether user wants to logout | Yes or  No | If yes, display message as Logout successfully and dashboard page is closed and again Login Page is open.  If not, it will  display message as Logout cancelled. | If yes, it displays message as  “Logout successfully” and dashboard page is closed and again Login Page appears.  If not, it will display message as “Logout cancelled”. | Pass |

**3. Generate Library Report:**

* **Sub-Category:** Report Testing
* **Explanation:** This involves testing the functionality to generate reports within the Book Buffet. Functional Testing verifies that the generated reports contain accurate and relevant information, are formatted correctly, and meet the specified requirements.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test case**  **ID** | **Test Scenario** | **Test Case** | **Test Data** | **Expected Result** | **Actual Result** | **Status**  **(Pass/**  **Fail)** |
| T3 | Generate **Report** | It should generate report on number of books issued on specific date and total quantity of books available in library visually | Date in  (YYYY-  MM-  DD)  format or simply year  (YYYY) | Visual representation of total quantity of books available in library as well as total number of books issued on specific date | Visual representation of total quantity of books available in library as well as total number of books issued  on specific date | Pass |

**4. Create Database Backup:**

* **Sub-Category:** Reliability Testing
* **Explanation:** Reliability Testing focuses on ensuring the system's ability to perform consistently and reliably over time. Testing the creation of database backups involves verifying that the backup process is robust, dependable, and capable of preserving data integrity in various scenarios.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test case**  **ID** | **Test Scenario** | **Test Case** | **Test Data** | **Expected Result** | **Actual Result** | **Status**  **(Pass/**  **Fail)** |
| T4 | To create **Database backup** | It should create backup of databases in excel as well as pdf format | BookDB  and  MemDB  databases | Display success message and create backup\_folder with excel and pdf files of both databases | Displays success message as  “Backup created successfully” and create backup\_folder with excel and pdf files of both  databases | Pass |

**5. Open Web Browser:**

* **Sub-Category:** Compatibility Testing
* **Explanation:** Compatibility Testing ensures that the application functions correctly across different web browsers. Testing the ability to open a web browser involves verifying compatibility with various browsers to ensure consistent behaviour and appearance.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test case**  **ID** | **Test Scenario** | **Test Case** | **Test Data** | **Expected Result** | **Actual Result** | **Status**  **(Pass/**  **Fail)** |
| T5 | **Open web browser**  to purchase books | It should open Amazon website under books  category in chrome | ---- | It should open Amazon  website under books category inside chrome web browser | Opens  Amazon  website under books category  inside chrome web  browser | Pass |

**6. Display List of Existing Books:**

* **Sub-Category:** User Interface Testing
* **Explanation:** This involves testing the user interface to ensure that it correctly displays the list of existing books. Functional Testing verifies that the displayed information matches the data stored in the system's database.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test case**  **ID** | **Test Scenario** | **Test Case** | **Test Data** | **Expected Result** | **Actual Result** | **Status**  **(Pass/**  **Fail)** |
| T6 | To **display a list of existing books** in library | Display  Existing Books List in tabular format | BookDB  database | Display a list of existing books inside library database in tabular format | Displays a list of existing books inside library database in  tabular format | Pass |

**7. Return book and Book Issuance:**

* **Sub-Category:** Transaction Testing
* **Explanation: "Book return"** and **"book issuance"** functionalities in a Book Buffet fall under the category of Transaction Testing within Functional Testing. These functionalities involve specific actions or operations that users perform within the system, such as borrowing books from the library (issuance) and returning borrowed books (return).

* **Transaction Testing** focuses on verifying the integrity and correctness of transactions or operations within the software application.
* **Transaction Testing** ensures that transactions, such as book issuance and return, are processed correctly within the system, data is accurately recorded and updated, and any associated business rules or constraints are enforced effectively.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test case**  **ID** | **Test Scenario** | **Test Case** | **Test Data** | **Expected Result** | **Actual Result** | **Status**  **(Pass/**  **Fail)** |
| T7 | User should be  able to **return book** which was issued previously | If the both the fields are empty and user clicks on return button | Book ID, Member ID | The system should prompt warning message as “Please enter a value” | The system prompted a warning message as “Please enter a value” | Pass |
| T7 | User should be  able to **return book** which was issued previously | If user enters an Invalid  Member ID and valid Book ID and clicks on return button | Member ID, Book ID | The system should prompt an error message as “Member does  not exist” | The system prompted an error message as “Member does not  exist” | Pass |
| T7 | User should be able to **return**  **book** which was issued previously | If user enters an Invalid book ID and valid member ID and clicks on return button | Book ID, Member ID | The system should prompt an error message as “Book does not exist” | The system prompted an error message as “Book does not exist” | Pass |
| T7 | If specific book **wasn’t issued** to specified member | If user enters a valid book ID and member ID and clicks on return Button | Book ID,  Member ID,  Return date | System prompts a warning message as “book is not currently issued to specified member” | The system prompted a warning message as “book is not currently issued to specified  member” | Pass |
| T7 | User should be  able to **return book** which was issued previously | If user enters a valid book ID and member ID and clicks on return Button | Book ID,  Member ID,  Return date, User confirmation:  Yes or No            User confirmation:  Yes or No | System asks confirmation whether to return that book. If user clicks **No**, it prompts message as  “Return  Canceled”.  If user clicks **YES,** it prompts message whether to calculate fine: **Yes** or **No**. If user clicks **Yes**, it calculates fine for number of days from expected return date to return date entered by user on interface and displays message as  “Book returned successfully”. If user clicks **No**, it returns book without calculating fine. | System asks confirmation whether to  return that  book. When user clicked **No**, it prompted message as  “Return  Canceled”. When user clicked **YES**, it prompted message whether to calculate fine: **Yes** or **No**. If user clicked **Yes**, it calculated fine and displayed a message as “book returned successfully” If user clicked **No**, it returned book without calculating  fine | Pass |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| T8 | User should be able to **issue** **book** | If user enters a valid book ID and member ID and clicks on issue Button | Book ID, Member ID | The system should prompt a success message as “Book issued successfully” | The system prompted a success message as “Book issued successfully” | Pass |
| T8 | User should be able to **issue** **book** | If user enters an invalid  book ID or member ID and clicks on issue Button | Book ID, Member ID | The system should prompt warning message as  “Invalid  Member ID or  Book ID” | The system prompted a warning message as  “Invalid  Member ID or Book ID” | Pass |
| T8 | User should be able to **issue** **book** | If the both the fields are empty and user clicks on issue button | Book ID, Member ID | The system should prompt warning message as “Please enter  Member ID or  Book ID” | The system prompted a warning message as  “Please enter  Member ID or Book ID” | Pass |

**CHAPTER 6: RESULT AND DISCUSSION**

**6.1 TEST REPORTS:**

Test reports are documents that provide a comprehensive summary and analysis of the results obtained from testing activities performed during the software development life cycle. These reports serve as a crucial component of the quality assurance process, helping to assess the software's reliability and functionality to specified requirements.

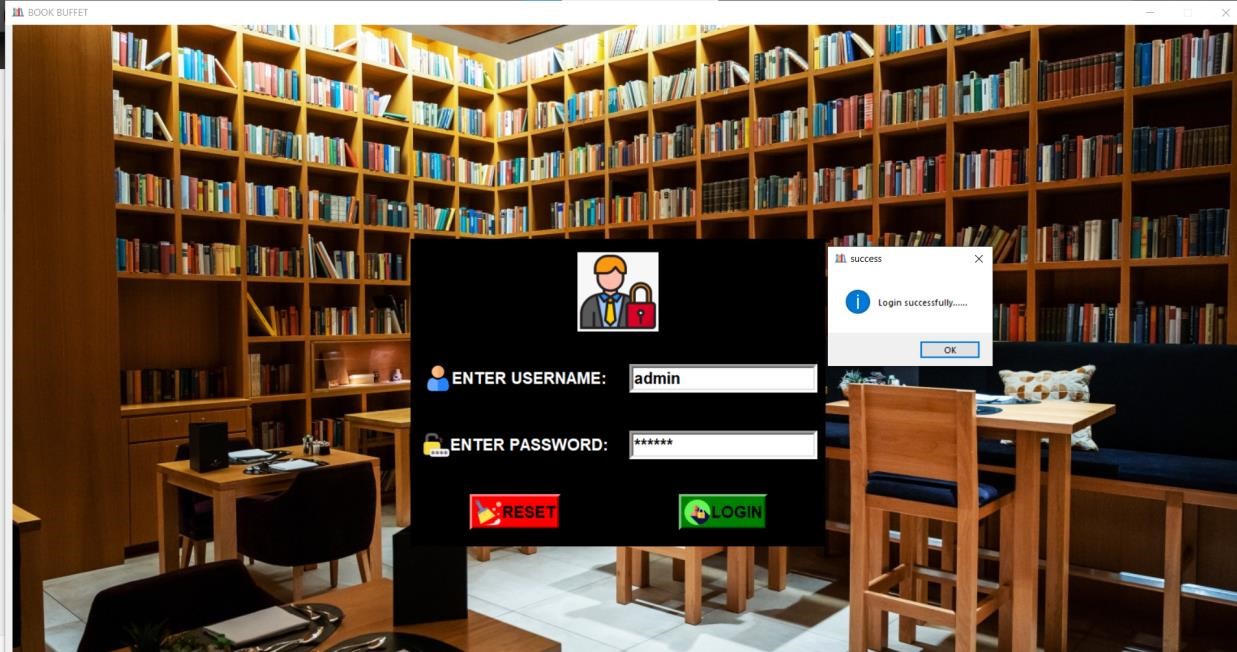
Test reports are essential for communication between different stakeholders, such as developers, testers, project managers, and clients. They help with making informed decisions about the quality of the software and any necessary corrective actions. Additionally, test reports provide a historical record of the testing process, which can be valuable for future projects and audits.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **TEST REPORT** | |  |  |  |
| ***Test***  ***Case ID*** | ***Description*** | ***Testing Approach*** | ***Total Test Cases***  ***Executed*** | ***% Test Case***  ***Passed*** | ***Priority*** |
| T01 | Add new book | Manual testing | 4 | 100% | High |
| T02 | Update existing book details | Manual testing | 3 | 100% | High |
| T03 | Add new member | Manual testing | 4 | 100% | High |
| T04 | Delete existing book | Manual testing | 3 | 100% | High |
| T05 | Search book details | Manual testing | 3 | 100% | High |
| T06 | View existing members | Manual testing | 1 | 100% | High |
| T07 | View issued books | Manual testing | 1 | 100% | High |
|  |  | |  |  |  |
| T1 | Verify Admin Login | Authentication testing | 3 | 100% | High |
| T2 | Logout from dashboard page | Session management testing | 1 | 100% | High |
| T3 | Generate library report | Report testing | 1 | 100% | High |
| T4 | Create database backup | Reliability testing | 1 | 100% | High |
| T5 | Open web browser | Compatibility testing | 1 | 100% | High |
| T6 | Display list of existing books | User interface testing | 1 | 100% | High |
| T7 | Return book | Transaction testing | 1 | 100% | High |
| T8 | Issue book | Transaction testing | 1 | 100% | High |

**6.2 User Documentation:**

User documentation, also known as user manuals or user guides, is a crucial component of software and product development. It provides users with information on how to use a product or software effectively. Well-written and comprehensive user documentation can enhance the user experience, reduce support requests, and contribute to overall customer satisfaction.

**1. Login Page**



This is the first interface which is going to display when the app is opened. It includes 2 entry fields to enter username and password and 2 buttons, one to reset both the fields to empty and a login button to logged-in into the system.

Username and password are predefined, and user should enter both credentials correctly to log onto the system successfully. If the user enters any of the two credentials incorrect it will prompt the user to enter correct details. If the user left any of the two fields empty it will prompt a message to fill out required details.

Following details needs to be entered to logged-in successfully:

**Username: admin**

**Password: shruti**

## 2. Dashboard Page

After successful login dashboard page appears. The left side panel contains multiple options to choose one operation to perform at a time. On the top it displays a welcome message to the admin. On the top right corner, the current date and time is displayed.

Below there is a list of currently available books in the library. It is displayed in a tabular format with all the relevant details about the book including its price, name of the author, title, its unique ID to refer to that book and available copies of it.

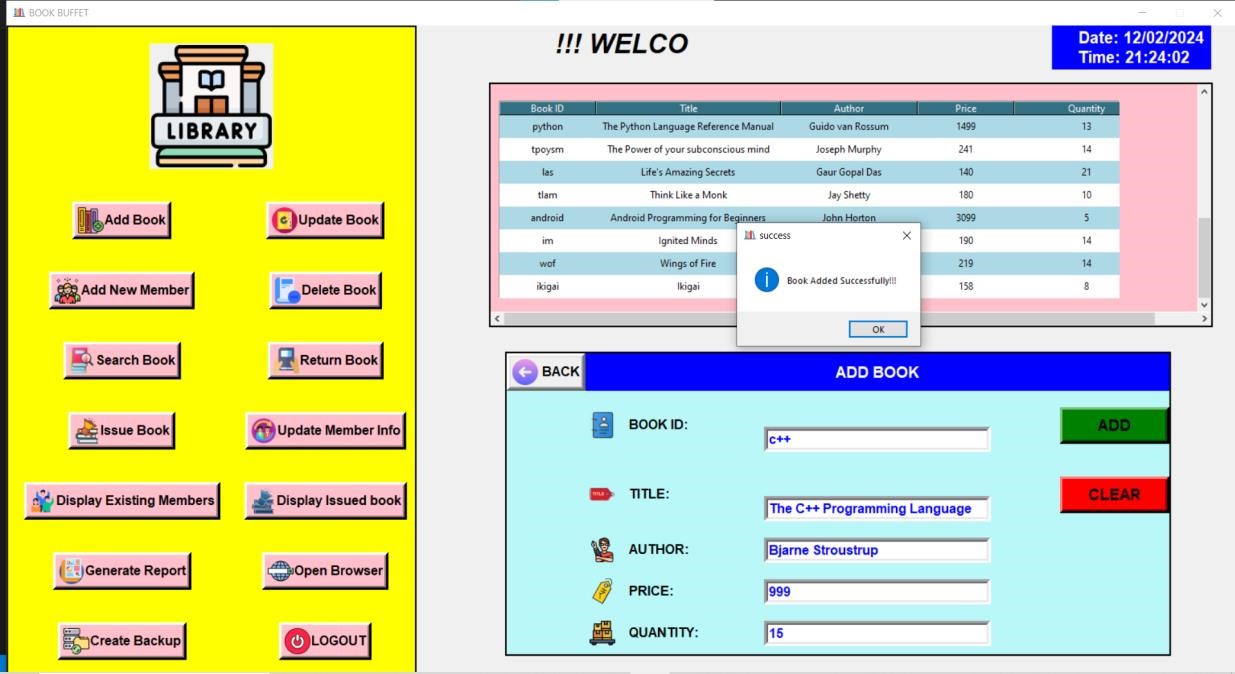


### 3) Add Book

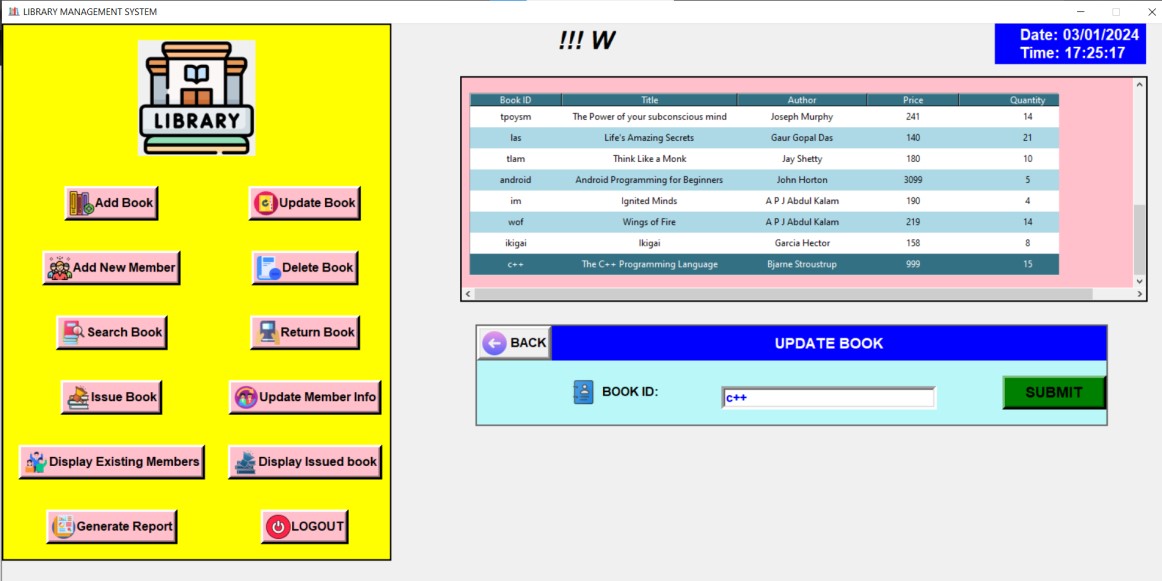
When the Add book button is clicked it displays a frame to enter the details about the book to add in the library’s database. Users can clear the fields by clicking the clear button all at once. If the user enters a character/letter in place of a number in the Price and Quantity field, it will throw an error with an appropriate message.

All the fields are required to be filled out otherwise it will not be going to proceed further and displays a warning message. When the user enters all the fields correctly and clicks on the add button it will prompt a success message and the book details are added in the library database and user can see that added book at the end of the table.

There is a back button on the left side of the heading to go back.

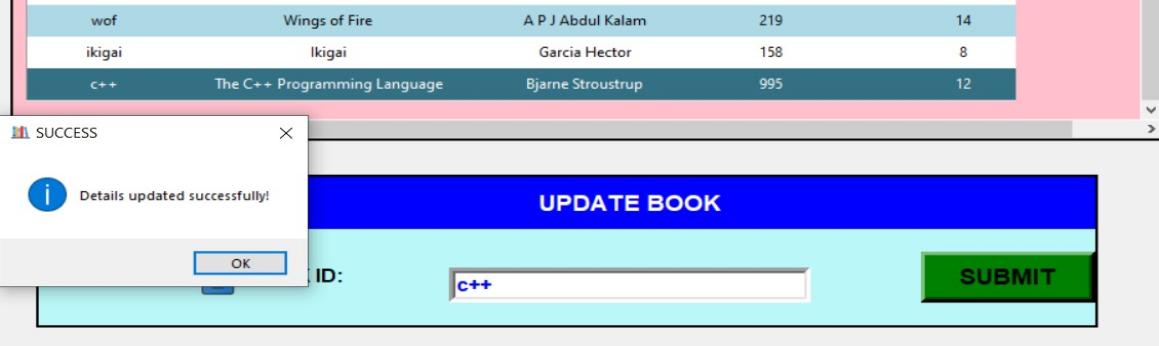
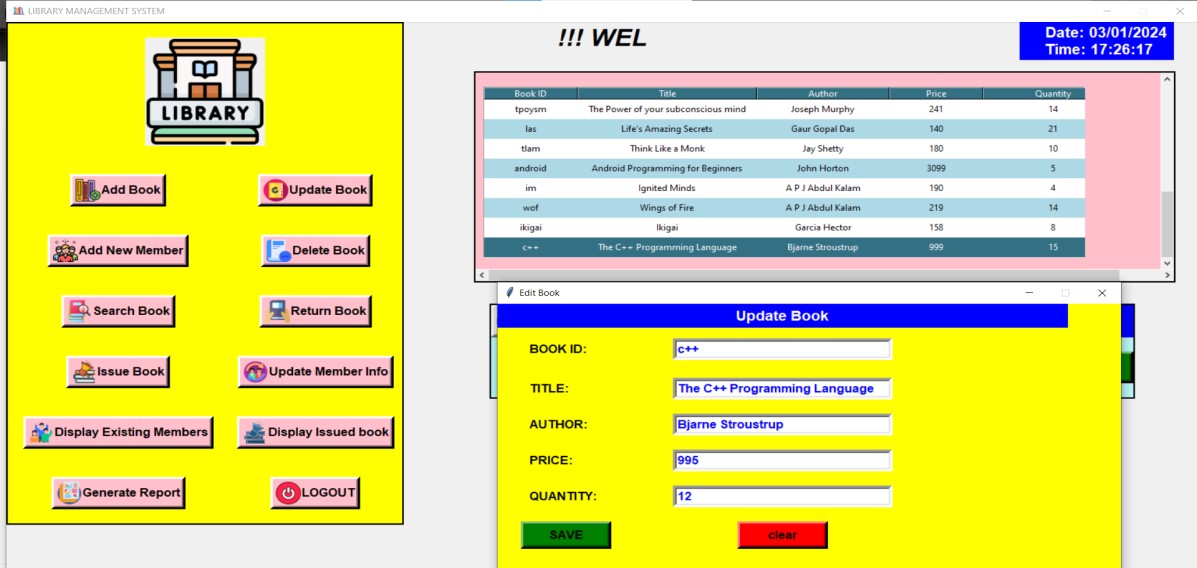


### 4) Update Book

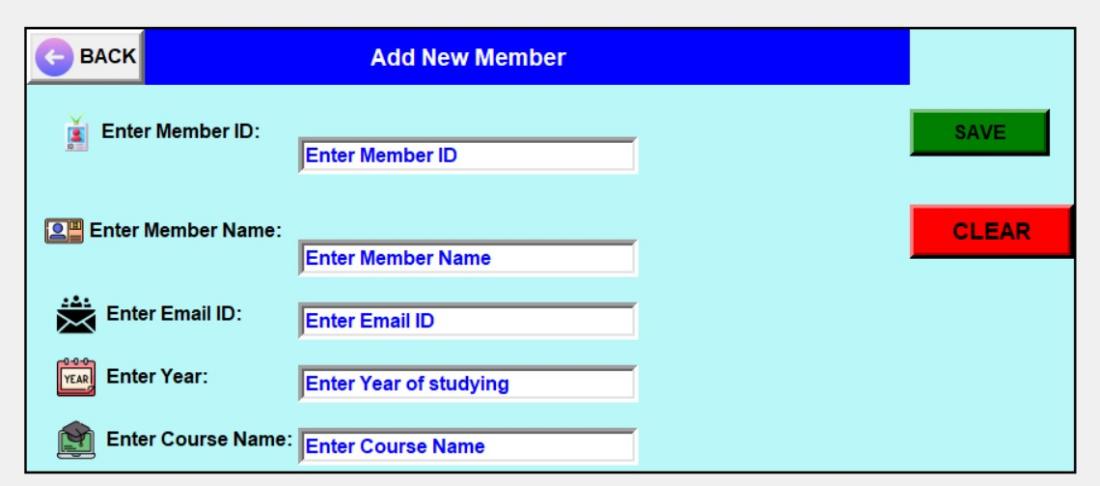


Users should enter the correct ID of the book to update which is present in library otherwise it will throw an error that the book does not exist in the database also this field should not be empty as it will not be able to proceed further for updating.

When user enters the correct book ID which is present and clicks the submit button it will fetch its entire details from the database and enable the user to update necessary details. Once clicked on the save button it displays a success message and changes get reflected in the database and the user can see the change in the table.

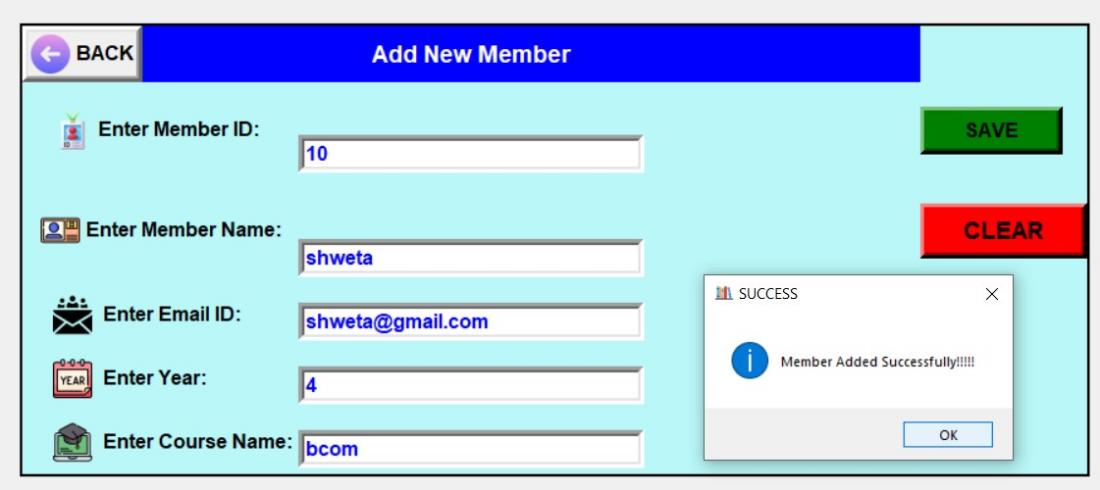


### 5) Add New Member

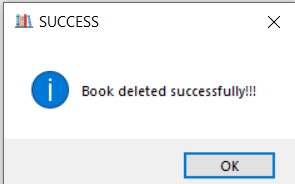
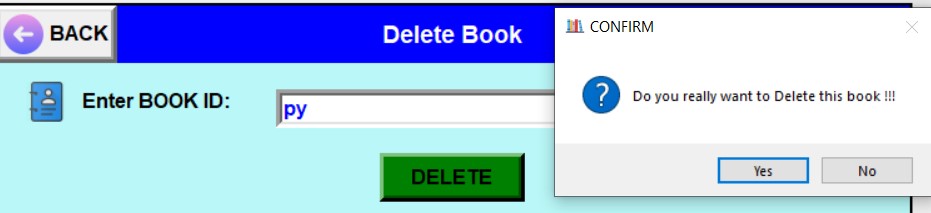


To add new members, users should enter all the fields otherwise it will display a warning message if any of the fields are left blank. Member ID and Year needs to be an integer value, if any character/string is entered then it will throw an error message. Year should be between 1 and 4 any other number other than that entered will display an error message.

Member ID should be unique for each member. Clear button clears all the fields all at once to enter new values. Save button adds new member in the library database and shows a success message upon addition.



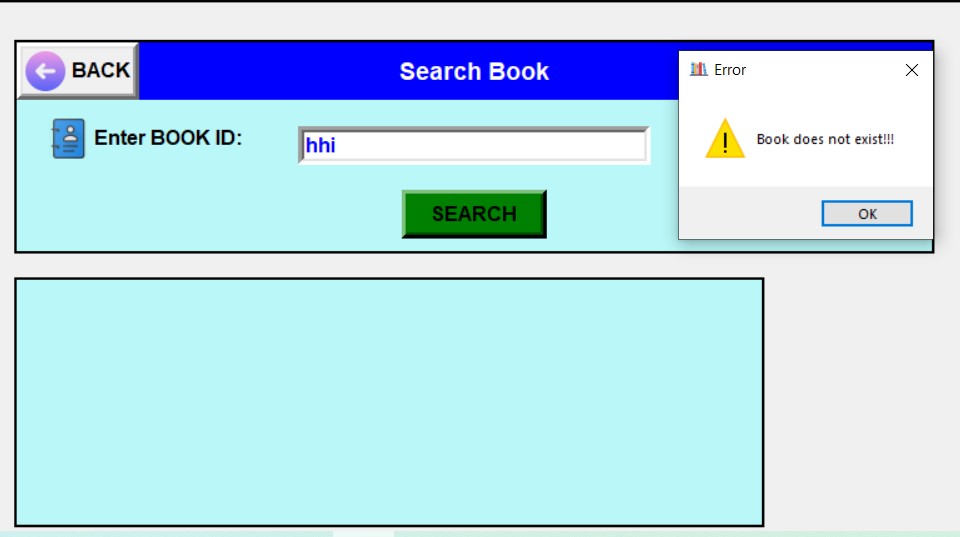
### 8. Delete Book



Users should enter the Book ID to delete all the available copies of that book which is present in library. The system checks whether the book is present in the database otherwise it will throw an error that the book does not exist in the library’s database also this field should not be empty otherwise it will prompt error message as please enter Book ID. If the book is present in the database, it will ask confirmation before deleting: Do u really want to delete this book?

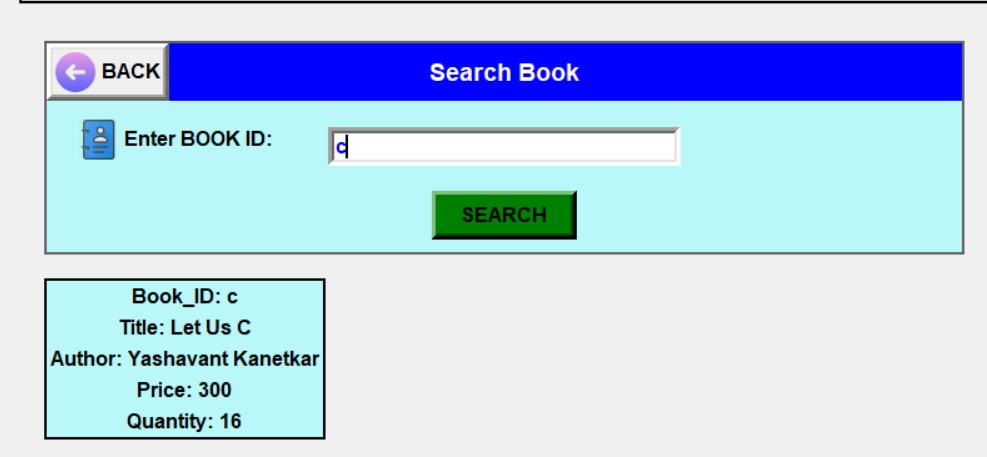
If the user clicks yes, it shows a success message and deletes the book permanently. If user click No, it displays a message as deletion cancelled.

### 9. Search Book



Users should enter a Book ID to search for the entire details of the book. If a book is not present in the database, it throws error message as book does not exist also if the field is empty, it will not be able to proceed further for search operation.

If the entered ID is present in the database, the system displays all the related details of that book.

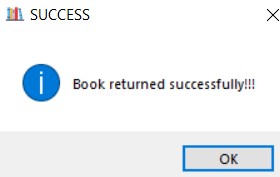
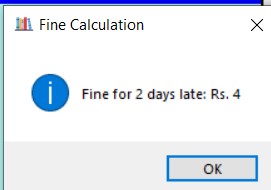
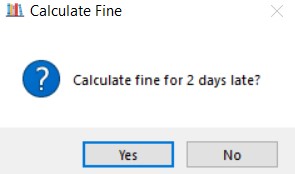
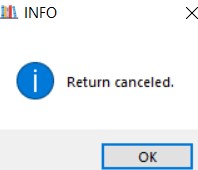
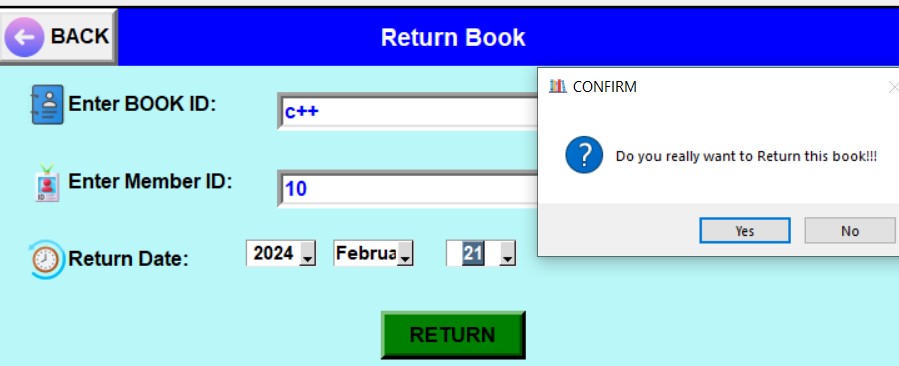


### 10. Return Book

To return a book which was issued previously to a member of the library, the user needs to enter the ID of that issued book and Member’s ID along with the return date of that book.

The system checks whether the member is a registered member or not. If not, it will display message as member does not exist. Next, it will check whether that book is issued to that specified member or not and if not, it will prompt an appropriate message.

If the book is issued to that specific member, it will ask whether you want to return this book and if yes it will calculate fine based on the return date and expected return date.



The system gets the actual return date from the GUI components (**return\_year**, **return\_month and**  **return\_date**).

Next, it will compare the actual return date with the expected return date to determine if the book is returned late.

If the book is returned late, calculate the fine based on the number of days late (fine rate is assumed to be Rs. 2 per day).

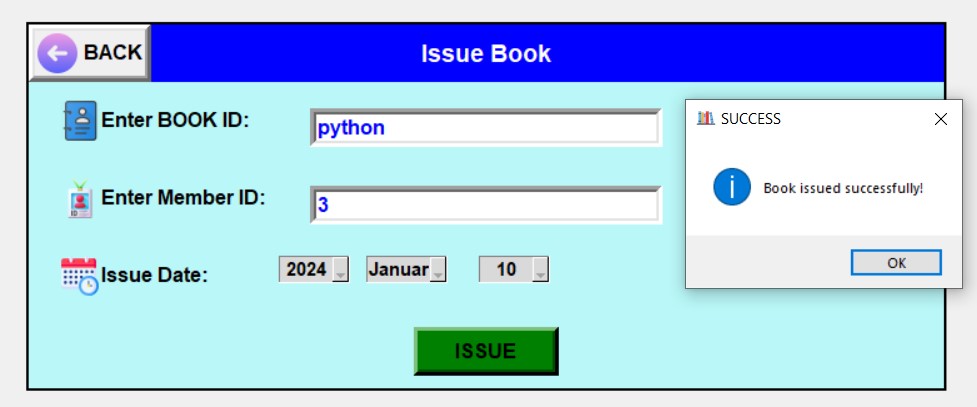
Display a confirmation message to calculate the fine and show the fine amount in a message box.

If the user confirms yes, delete the record from the "Expected\_Return\_Dates" table in the database.

If the book is returned on time, display a message indicating no fine.

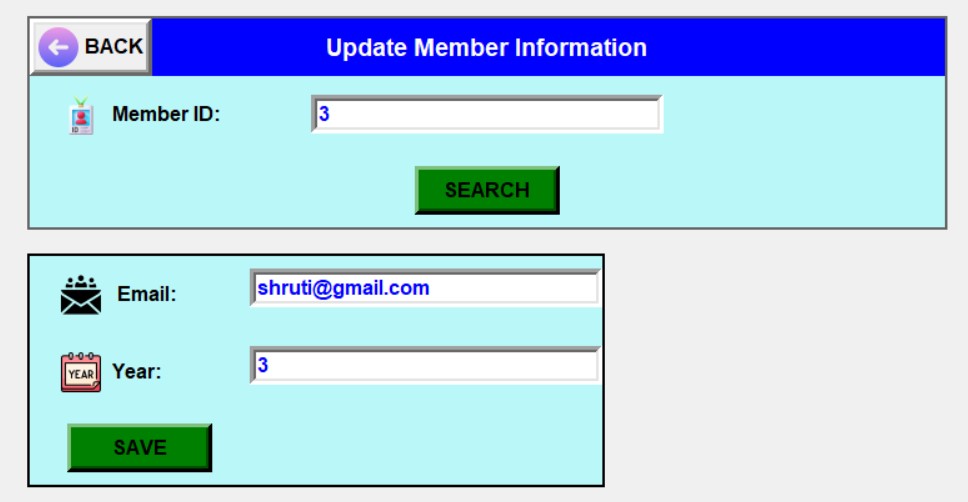
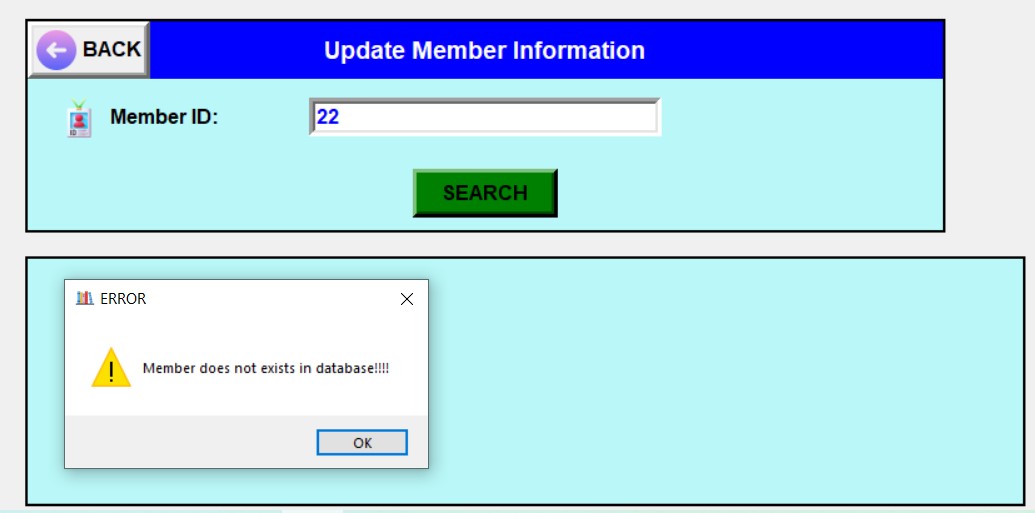
### 11. Issue book

To issue a book to a member the user needs to enter a valid Book ID and Member ID. User cannot change the issue date and when click on issue button it will display message as “book issued successfully”. If the book or member ID is not valid it will display message as “Invalid book id or member id”.



## 12. Update Member Information

Two frames are created to organize and display different sections of the interface. User should enter a valid member ID and if not, it will display message as “Member does not exist”. The search button retrieves member information from the database based on the provided Member ID. And if member exists it will allow to change Email ID and Year of studying and then display message as “Member details updated successfully”.



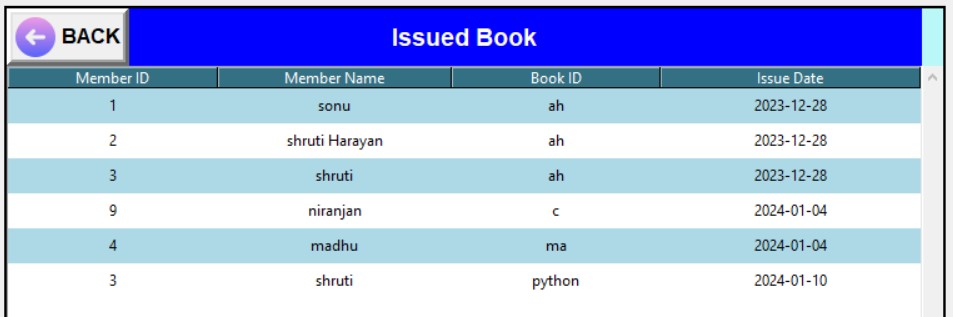
### 13. Display Existing Member

It will display all the existing members in the library in a tabular format which is created by using Treeview widget.



### 14. Display Issued Book

It will display all the books which are issued to the registered members in the library including issue date.

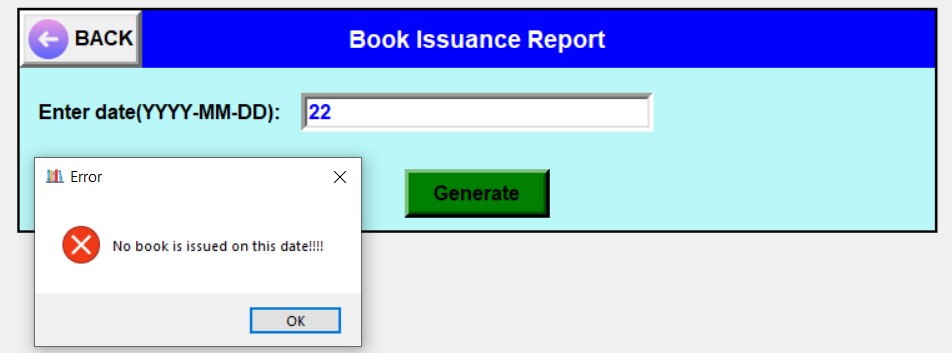


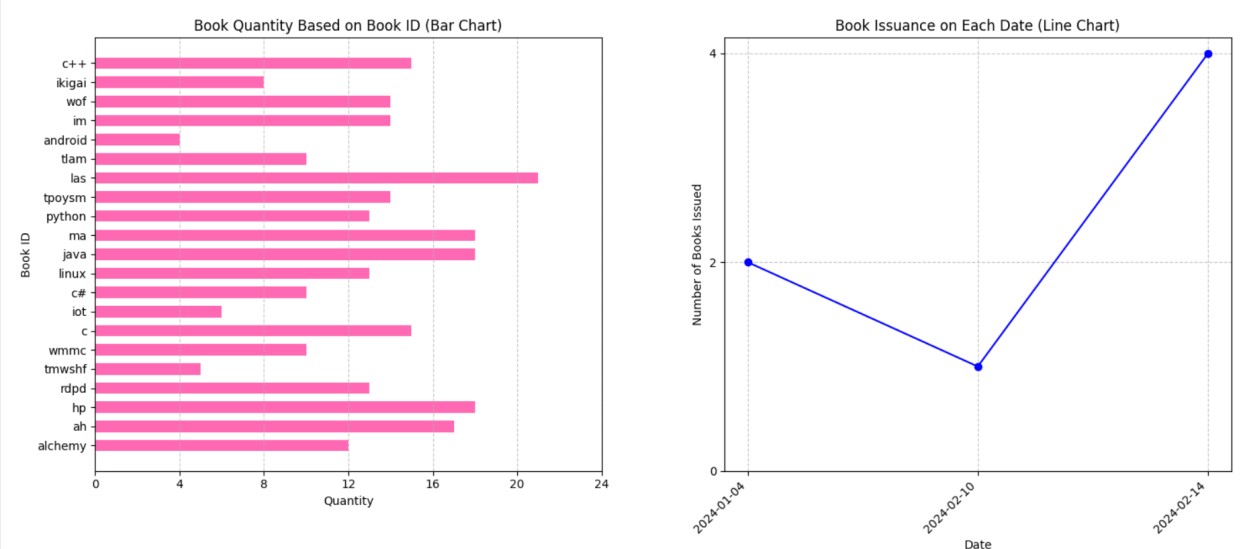
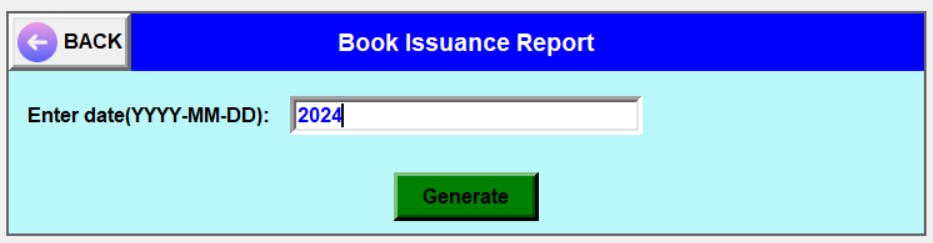
### 15. Book Issuance report

User needs to enter date in specific format to generate report on desired date. If that date is not in a valid format, it will display an error message. User can save the figure in desired location for further reference. This report includes 2 figures:

Fig 1: where user can check the quantity of all the books visually

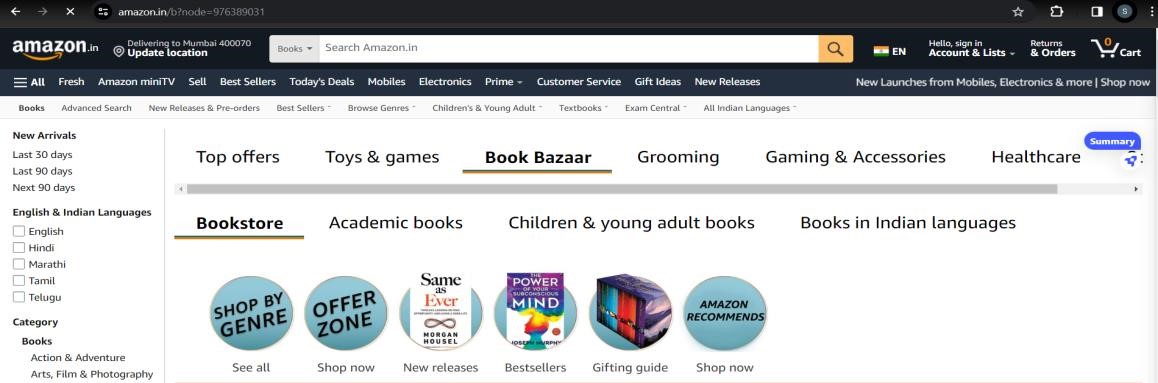
Fig 2: how many numbers of books are issued on desired date





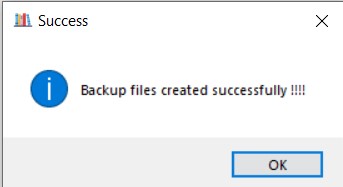
### 16. Open Browser

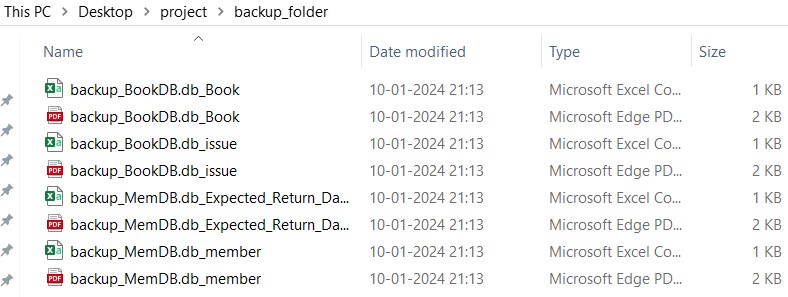
It will open Amazon website category books for the user to purchase different variety of books for the library.



### 17. Create Backup

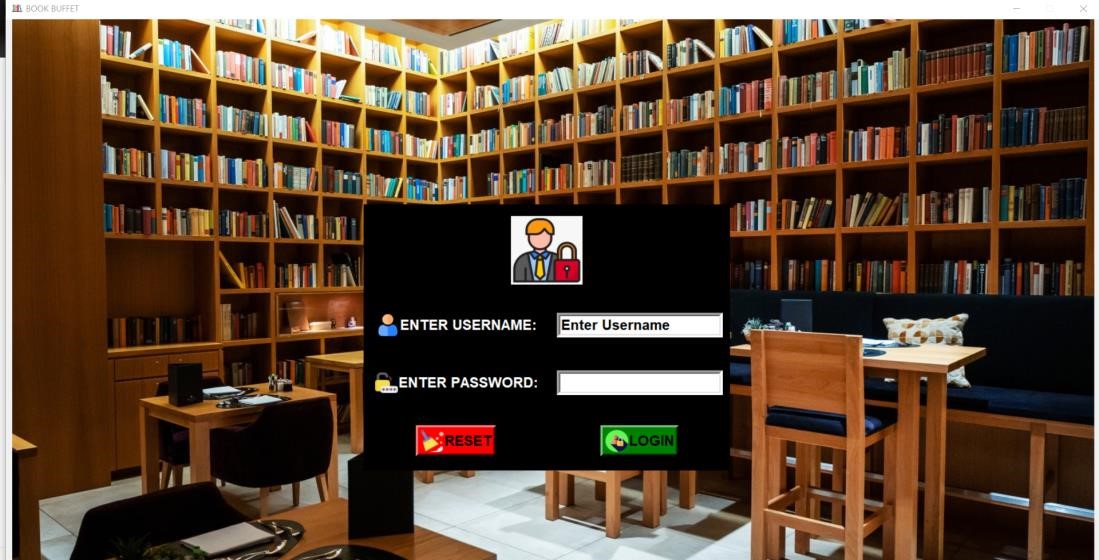
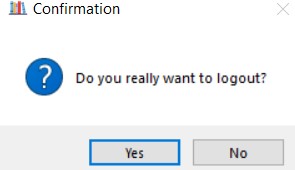
It will create a folder name backup\_folder and backup of all databases in CSV (Excel) and PDF format and display success message after successful creation.





### 18. Logout

It will ask the user whether he wants to logout and if the user clicks No, it will display message as logout cancelled and if user clicks Yes, it will again display login Page.



# CHAPTER 7: CONCLUSION

## 7.1 Conclusion

This software package allows storing the details of all the data related to the library. The system is strong enough to withstand regressive yearly operations under conditions where the database is maintained and cleared over a certain time span. The implementation of the system in the organization will considerably reduce data entry, time and provide readily calculated reports.

The following conclusions can be derived from the development of this project:

* Book Buffet improves efficiency.
* It provides a friendly graphical user interface which proves to be better when compared to the existing system.
* It gives appropriate access to the authorized users depending on their permissions.
* It effectively overcomes the delay in communications.
* System security, data security and reliability are the striking features.
* The System has adequate scope for modification in future if it is necessary.

## 7.2 Future Scope of The Project

In a nutshell, it can be summarized that the future scope of the project circles around maintaining information regarding:

♦ We can add printers in future.

♦ We can provide users and librarian login to access relevant parts of the library.

♦ We can give more advanced software for the Book Buffet, including more facilities.

♦ We will host the platform on online servers to make it accessible worldwide. Create the master and slave database structure to reduce the overload of the database queries.

## REFERENCES

* <https://www.geeksforgeeks.org/class-diagram-for-library-management-system/>
* [www.google.com](http://www.google.com/)
* <https://www.studocu.com/en-gb/document/university-of-east-london/advanced-software-engineering/library-management-system-final-report/10662691>
* <https://www.softwaretestingmaterial.com/manual-testing-tutorial/>
* [https://www.softwaretestingmaterial.com/test-case-template-with-explanation/#google\_vignette](https://www.softwaretestingmaterial.com/test-case-template-with-explanation/" \l "google_vignette)

Reference Book

* TKinter Widget Quick Reference Guide,2022 by John Elder