

# **MASTER OF COMPUTER APPLICATIONS**

## **LIBRARY MANAGEMENT SYSTEM**

**Submitted to:**



**Department of Computer Science, School of Engineering and Technology  
Islamic University of Science and Technology.**

# **LIBRARY MANAGEMENT SYSTEM**



**Project Submitted to the Department of Computer Sciences, School of Technology, Islamic University of Science and Technology in partial fulfilment of the requirements for the award of the degree of**

## **Master of Computer Applications**

Submitted by:

Pir Hilal Ahmad (MCA-21-25)

Zameer Ahmad Mir (MCA-21-61)

**DEPARTMENT OF COMPUTER SCIENCE,  
SCHOOL OF ENGINEERING & TECHNOLOGY**

**Islamic University of Science and Technology**  
**Awantipora, Pulwama, Kashmir, 192122**



**CERTIFICATE**

This is to certify that the project entitled

**LIBRARY MANAGEMENT SYSTEM**

Has been carried out by:

Pir Hilal Ahmad (MCA-21-25)

Zameer Ahmad Mir (MCA-21-61)

In partial fulfilment of the requirements for the award of the degree of

**Master of Computer Applications**

during the academic year 2023

**Dr Shabia Shabir**

(Assistant Professor, Deptt.  
of Computer Science, IUST)

**Dr Rumaan Bashir**

(HOD, Deptt.Of Computer  
Science, IUST)

### **Student Declaration/Certificate**

We Pir Hilal Ahmad and Zameer Ahmad Mir hereby declare that the work, which is being presented in the project entitled " **LIBRARY MANAGEMENT SYSYTEM**" in partial fulfilment of the requirement for the award of MASTER OF COMPUTER APPLICATION (MCA) degree in the session 2023, is an authentic record of our own work carried out under the supervision of **Dr Shabia Shabir**, Assistant Professor, Department of Computer Science,

The matter embodied in this project has not been submitted by us for the award of any other degree.

Date: \_\_\_\_\_

1. Pir Hilal Ahmad

2. Zameer Ahmad Mir

This is to certify that the above statements made by the candidates are correct to the best of my knowledge.

**Dr Shabia Shabir**

Assistant Professor

(Deptt. of Computer Science, IUST)

**Dr Rumaan Bashir**

H.O.D.

(Deptt. of Computer Science, IUST)

## **Acknowledgement**

*“Success is born from the fusion of perseverance and passion. Embrace challenges as opportunities, fuel your dreams with determination, and let resilience be the catalyst for your extraordinary journey.”*

**If it weren't for dedication, we wouldn't be here.**

The success and final outcome of this project required a lot of guidance and assistance from many people and we are extremely fortunate to have got this all along the completion of our project work. Whatever we have done is only due to such guidance and assistance and we would not forget to thank them.

We respect and thank Dr. Shabia Shabir for giving us the opportunity to do the project work in “LIBRARY MANAGEMENT SYSTEM.” and providing us all the support and guidance which made us to complete the project on time. We are extremely grateful to him for providing such a nice support and guidance though he had busy schedule managing the academics of the department.

We would not forget to remember all the teaching and non-teaching staff of Dept. of Computer Science for their enlisted encouragement and moreover for their timely support and guidance till the completion of our project work.

Lastly, we would like to extend our sincere regards to all our family and friends for their constant encouragement and constant source of inspiration during the preparation of the project work without which this project work could not have been possible.

Thanking you

## Table of Contents

<b>1. Introduction</b>	<b>8-11</b>
1.1 Overview	9
1.2 Goals & Objectives	10
1.3 Existing System	11
1.4 Drawbacks of Existing System	11
<b>2. Proposed System</b>	<b>12-31</b>
2.1 Overview	13
2.2 Features of Proposed System	16
2.3 Feasibility Study	22
2.3.1 Economic Feasibility	22
2.3.2 Technical Feasibility	22
2.3.3 Operational Feasibility	23
2.4 Project Plan	24
2.4.1 Project Planning Objectives	24
2.4.2 Project Estimation	24
2.4.2.1 Title & Scope of Project	25
2.4.2.2 Project Decomposition	26
2.4.2.3 LOC –based Estimation	27
2.4.3 Project Scheduling & Tracking	27
2.4.3.1 Project Scheduling and Tracking Steps	27
2.4.4.1 Risk Analysis Steps	28
2.4.3.1 Risks Identified	28
2.5 Software Engineering Paradigm	28
<b>3. System Analysis</b>	<b>31-43</b>
3.1 Overview	32
3.2 Requirement Analysis	32
3.3 Requirement Elicitation of Software	33
3.4 Analysis Principles	33
3.5 Requirement Gathering Techniques	33
3.5.1 Interview Technique	33
3.5.2 Joint Application Development Technique (JAD)	34
3.5.3 Survey Method	35

## Table of Content Continued.....

3.6 Requirement Gathering for Web Application	35
3.7.1 Data Modeling (ERD)	39
3.7.2 Functional Modeling (DFD)	40
3.7.3 Behavioral Modeling (STD)	41
3.7.4 UML Diagrams (Use-Cases, Activity diagram , Sequential Diagram)	42
<b>4. System Design</b>	<b>46-61</b>
4.1 Overview	46
4.2 System Design Steps	46
4.3 Software Design and Software Engineering	46
4.4 Design issues of Intranet Application Engineering	48
4.5 Interface Design	48
4.6 Architecture Design	48
<b>4 Coding</b>	<b>63-97</b>
<b>5 Testing</b>	<b>99-105</b>
<b>6 Implementation</b>	<b>109-109</b>
<b>7 Snapshots</b>	<b>111-115</b>
<b>8 Conclusion</b>	<b>117</b>
<b>9 Future Scope</b>	<b>119</b>
<b>10 Bibliography</b>	<b>121</b>

# *Chapter 1*

## *Introduction*



# **1.Introduction**

## **Overview**

The “Library Management System” is a software project developed in .NET to streamline and automate various library processes, offering an efficient and user-friendly platform for librarians and patrons. The system aims to enhance the overall management and accessibility of library resources, including books, journals, and multimedia, while providing essential features for administrators and users.

Library is place where all kind of books are available. Library Management System is a web-based application. This system contains list of all the books and can be accessed by remote users concurrently from anywhere in the campus. But for that users must be registered user. This system is three tier architecture.

Client sends requests, on receiving the request the server processes it and extracts the data from database and sends the result back to the client. This system provides separate interface and login for librarian, students and faculties. Librarian can modify database.

Users can search for books and renewal books online. They can recommend for new books by just sending messages to the librarian from anywhere in the college. They can view the issue and return dates of any book and due they have to pay. This system generates reports that can be used in analysing the library performance. Thus, the management can take appropriate steps to improve the facilities.

### **1.1.2 Goals and Objectives**

The primary objective of the proposed Library Management System in .NET is to develop a comprehensive and efficient software solution that revolutionizes the way libraries operate and interact with their users. The system aims to achieve the following specific objectives:

1. **Automation and Efficiency:** Streamline and automate various library processes, including cataloguing, borrowing, returning, and fine management, to reduce manual efforts and enhance overall operational efficiency.
2. **User-Friendly Interface:** Create an intuitive and user-friendly interface for both librarians and patrons, ensuring a seamless user experience while navigating the system and performing tasks.
3. **Real-Time Information:** Provide real-time updates on the availability of books, journals, and multimedia items to help users make informed decisions when selecting resources.
4. **Accurate Tracking and Reporting:** Enable librarians to generate various reports and

analytics related to library usage, popular books, inventory status, and member statistics to support data-driven decision-making.

5. Fine Management: Automate fine calculation and tracking for overdue items, ensuring timely notifications to patrons and efficient management of fines.
6. Renewal System: Facilitate easy and quick book renewal requests.
7. Data Security: Implement robust security measures to safeguard user data and maintain the confidentiality of sensitive information.

### 1.1.3 Existing System and its Drawbacks

The existing system here is the 'traditional library' which we use to see from the times before. The existing system here is completely manual run system. The burden of controlling the library entirely lies in the shoulder of the admin. More over the term "user friendly" doesn't have a place to stand in the existing library system, as the user have the heavy work of finding a desired book from the library.

Some of the major drawbacks the existing system faces is includes, wastage of time, man power, inconsistency, absence of secure mechanism, burden of record keeping & wastage of storage space, expensive.

When going deep into each;

**Time consuming:** Due to the lack of a computer based system, searching for a specific book by a specific user in the library is a consuming a lot of time. In the present system the user is only known about book the particular category of books that each shelf or rack in library holds. So for a particular book he has to search the shelf containing that category of books.

**Man power:** In the existing system all activities are human oriented. Every records relating to both books and users are created and maintained by the librarian manually. And if a search in record for a particular book or user or a transaction is needed it becomes a complex activity as the Librarian have to search the entire written records.

**Absence of secure mechanism:** There is no security present to check the identity of user or books and also records can be manipulated as all where in the written form.

**Record keeping:** In the existing system as all the records are kept in the written forms, each. year lot of new records are required when a new batch of new users comes in to the institution and also a lot of space is required for keeping the previous records and current records. So it's both a waste of time and money.

**Expensive:** Existing system is little bit expensive as the entire record keeping is a manual written activity; it consumes a lot of stationery and other resources.

**Report preparation:** It is a very difficult task to summarize all the records available as there are chances for alteration exists.

Since these much disadvantages exist with the existing manual library management it is better to run a changeover to a new system. Moving to a new computer based library management system will provide a lot of benefits to both the admin (Librarian) and the users.

Some other drawbacks are discussed as below:-

**Limited Features:** Some library management systems may lack essential features, such as comprehensive search options, user-friendly interfaces, and integration with other systems.

**Scalability Issues:** An existing system might face difficulties in handling a growing number of users, books, or transactions, leading to performance issues and slowdowns.

**Poor User Interface and Experience:** A suboptimal user interface can hinder the user experience, making it difficult for librarians, students, or other users to navigate the system efficiently.

**Lack of Mobile Responsiveness:** In an era where mobile devices are prevalent, a library management system that doesn't adapt well to different screen sizes and devices can limit accessibility.

**Inadequate Security Measures:** Security is crucial for a library management system, especially considering the sensitive nature of user and library data. Outdated or insufficient security measures could expose the system to potential threats.

**Limited Reporting and Analytics:** A system that lacks robust reporting and analytics capabilities may hinder administrators' ability to make informed decisions and optimize library operations.

**Ineffective Integration:** An inability to integrate with other systems, databases, or technologies can limit the system's overall functionality and interoperability.

# *Chapter 2*

## *Proposed System*

## **What is Library Management System?**

A library management system (LMS) is a software solution designed to facilitate and streamline the various tasks and processes involved in the efficient operation of a library. It serves as a comprehensive tool to manage the diverse aspects of library administration, from cataloging and book circulation to user management and reporting. The primary goal of an LMS is to automate and optimize routine library functions, enhancing the overall efficiency and effectiveness of library operations. At its core, a library management system typically includes features such as cataloging, which involves organizing and classifying library resources, be it books, journals, or multimedia items. The system maintains a centralized database of these resources, allowing for easy retrieval and tracking. The circulation module manages the borrowing and returning of materials, tracking due dates and sending notifications to patrons. User management features enable librarians to manage member profiles, issue library cards, and maintain records of patrons' borrowing history.

Moreover, an LMS often incorporates search and discovery functionalities, enabling users to locate resources efficiently. This is complemented by a user-friendly interface, contributing to an enhanced user experience. Some systems also offer advanced features like RFID technology for automated book tracking and self-checkout kiosks, further streamlining the borrowing process. Importantly, a library management system contributes to data-driven decision-making through its reporting and analytics capabilities. Librarians can generate reports on circulation patterns, popular items, and overdue materials, helping in collection development and resource allocation. Security measures are crucial, ensuring the protection of sensitive user information and safeguarding against unauthorized access. In essence, a well-designed library management system acts as a central hub for all library activities, fostering efficiency, accuracy, and improved services for both librarians and library patrons. It represents a crucial component in modernizing and digitizing library operations, aligning them with contemporary technological standards.

## **How It Works**

A Library Management System (LMS) works as a comprehensive and integrated software solution designed to facilitate the various processes involved in managing a library. Here's a breakdown of how an LMS typically works:

### **Cataloging and Classification:**

#### **Input of Library Resources:**

Librarians input information about books, journals, multimedia, and other resources into the system. This includes details like title, author, ISBN, publisher, genre, and more.

#### **Classification and Categorization:**

The system organizes resources based on predefined classification systems (e.g., Dewey Decimal Classification or Library of Congress Classification), making it easier to locate items.

### **User Management:**

#### **Member Registration:**

Librarians create and manage user profiles for library patrons. This includes personal information, contact details, and a unique identifier, often in the form of a library card.

### **Circulation Management:**

#### **Check-in and Check-out:**

When a patron borrows a book, the librarian uses the system to check out the item, associating it with the patron's account. When returned, the system updates the status accordingly.

#### **Overdue Tracking:**

The system monitors due dates and notifies patrons about overdue materials. It may also automate the process of blocking accounts or imposing fines for late returns.

### **Search and Discovery:**

#### **Online Catalog:**

Patrons use the system's search functionality to find available resources. This can include advanced search options, filters, and sorting mechanisms for a better user experience.

### **Reservation and Holds:**

#### **Patron Requests:**

Patrons can request or reserve items through the system, and librarians manage these requests, ensuring efficient resource allocation.

## **Reporting and Analytics:**

### **Data Analysis:**

The system generates reports on various aspects of library operations, such as circulation statistics, popular items, and user behavior. Librarians use this information for decision-making and collection development.

## **Security Measures:**

### **Access Control:**

The system incorporates access controls to ensure that only authorized personnel can perform specific functions, safeguarding sensitive user data and library resources.

## **Integration with Other Systems:**

### **Interoperability:**

Many LMSs can integrate with other institutional systems, such as student databases or financial systems, streamlining overall administrative processes.

## **Maintenance and Updates:**

### **Regular Upkeep:**

The LMS requires regular maintenance to address bugs, update information, and incorporate new features. System administrators ensure that the software operates smoothly.

In summary, a Library Management System works by automating and organizing the core functions of a library, providing librarians and patrons with efficient tools for resource management, circulation, and information retrieval. The system enhances the overall user experience and contributes to the effective administration of library resources.

## **Features of Library Management System**

Implementing a Library Management System (LMS) offers a range of benefits, enhancing the efficiency, effectiveness, and overall functionality of library operations. Here are some key advantages:

### **Automation of Manual Processes:**

LMS automates time-consuming manual processes such as cataloging, indexing, and circulation, reducing the workload on librarians and allowing them to focus on more strategic tasks.

### **Efficient Resource Management:**

The system helps in efficient management of library resources by providing tools for cataloging, tracking, and organizing books, journals, and multimedia items. This ensures accurate inventory control.

### **Enhanced User Experience:**

LMS provides patrons with user-friendly interfaces for searching and locating materials. It improves the overall user experience by offering advanced search options, recommendations, and easy access to information.

### **Quick and Accurate Information Retrieval:**

Users can quickly locate and retrieve information about available resources using the system's search functionalities, reducing the time spent searching for books or other materials.

### **Streamlined Circulation Processes:**

Automation of check-in, check-out, and reservation processes simplifies and expedites circulation tasks. This leads to faster service for patrons and more efficient management of library materials.

### **Effective Tracking of Borrowing History:**

LMS maintains a comprehensive borrowing history for each user, helping librarians track patterns, analyze preferences, and make informed decisions about collection development.

### **Improved Communication:**

The system facilitates communication between the library and patrons through features like email notifications for overdue items, reservation confirmations, and announcements about new acquisitions or events.



**Data-Driven Decision Making:**

LMS generates detailed reports and analytics on various aspects of library operations. Librarians can use this data to make informed decisions about resource allocation, collection development, and user engagement.

**Enhanced Security and Access Control:**

The system provides security features to protect sensitive user data and ensure that only authorized personnel have access to certain functions. This helps maintain the integrity and confidentiality of library information.

**Cost Savings:**

Automation and streamlining of processes can result in cost savings over time. Reduced manual labor, efficient use of resources, and improved management of collections contribute to overall cost-effectiveness.

**Remote Access and Mobile Capabilities:**

Many modern LMSs offer remote access and mobile capabilities, allowing users to access the library's resources from anywhere. This is especially important in the age of digital connectivity and mobile devices.

**Interoperability:**

LMS can integrate with other institutional systems, such as student databases or financial systems, fostering better coordination and interoperability within the broader organizational framework.

## **Problem Statement**

### **What is Library Management System and how it's Important to our Education System?**

A Library Management System (LMS) is a software solution that automates and organizes library operations. It is crucial for the education system as it enhances resource management, streamlines processes, and facilitates efficient access to information, contributing significantly to the overall academic and research experience.

### **Concept of Library Management System**

The concept of a Library Management System (LMS) revolves around a comprehensive software solution designed to automate and streamline the various tasks associated with library operations. It encompasses functions such as cataloging, circulation management, user registration, and resource tracking. The LMS serves as a centralized platform to efficiently manage library collections, enhance user experiences, and provide valuable insights through reporting and analytics. By automating manual processes, optimizing resource allocation, and improving accessibility to information, an LMS contributes to the overall effectiveness of library administration. The concept underscores the importance of leveraging technology to modernize libraries, making them more user-friendly, resource-efficient, and aligned with contemporary educational and research needs.

### **Purpose of Library Management System**

The purpose of a Library Management System (LMS) lies in revolutionizing and optimizing the management of library resources, fundamentally enhancing the efficiency and effectiveness of library operations. At its core, an LMS serves as a centralized platform for cataloging, organizing, and disseminating diverse library materials, including books, journals, and multimedia resources.

One of the primary purposes is to automate manual processes traditionally associated with libraries. By digitizing tasks like cataloging and circulation, an LMS reduces the burden on librarians, allowing them to allocate more time to strategic activities and improving overall workflow.

Furthermore, an LMS aims to enhance user experiences. With advanced search capabilities, user-friendly interfaces, and features like online catalogs, patrons can easily locate and access materials. The system's automation of circulation processes, including check-in, check-out, and reservation, not only expedites transactions but also contributes to a more seamless and user-friendly interaction. The LMS also plays a crucial role in resource management. It provides librarians with tools to efficiently track inventory, manage borrowing histories, and make informed decisions about collection development. The reporting and analytics features empower library administrators to assess usage patterns, identify popular materials, and strategically plan for the library's future needs.

Moreover, the purpose extends to fostering communication between the library and its patrons. Through notifications, announcements, and other communication tools, the LMS keeps users informed about overdue items, upcoming events, and new acquisitions, contributing to an engaged and well-informed community.

In summary, the purpose of an LMS is multifaceted: it automates manual tasks, improves user experiences, enhances resource management, and facilitates effective communication. By fulfilling these objectives, an LMS transforms libraries into dynamic, efficient, and user-centric hubs of information, aligning them with the evolving needs of modern educational and research environments.

### **Pace of Library Management System:**

The inherent flexibility of Library Management Systems (LMS) is pivotal for their effectiveness, allowing users to progress through resource exploration and utilization at their own pace. Unlike traditional library settings, LMS caters to diverse user needs, offering a personalized experience in accessing and managing library materials. This adaptability benefits all patrons, providing a tailored space for efficient resource utilization.

While LMS affords autonomy in resource management, it also integrates features such as cataloging, circulation, and user registration. These functions may have structured processes, ensuring organized and standardized library operations. The mention of potential time constraints or supervision in certain aspects of library management aligns with the need to maintain efficiency and adhere to specific operational standards.

In essence, while LMS grants flexibility in resource utilization, certain components may have defined parameters to ensure the integrity and efficiency of library operations.

## **Flexibility**

The flexibility of a Library Management System (LMS) lies in its ability to adapt and cater to the diverse needs of library operations. This flexibility encompasses several aspects:

### **1. Resource Accessibility:**

LMS allows users to access library resources at their convenience. Patrons can browse catalogs, check in/out materials, and explore digital resources from any location with internet access.

### **2. User-Friendly Interface:**

A flexible LMS is designed with a user-friendly interface, making it easy for both librarians and patrons to navigate and utilize its features effectively.

### **3. Customization of Catalogs:**

LMS provides flexibility in cataloging, allowing libraries to customize how they organize and classify materials. This adaptability ensures that the system aligns with the specific needs and preferences of the library.

### **4. Integration with Other Systems:**

Flexibility in integration enables LMS to connect with other institutional systems, enhancing interoperability and allowing seamless data exchange.

### **5. Adaptability to Changing Needs:**

An effective LMS should be adaptable to changing library needs. It should accommodate evolving technologies, new types of resources, and shifting user demands.

### **6. Data Management and Reporting:**

LMS flexibility extends to data management and reporting. Librarians can generate custom reports, track circulation patterns, and analyze data to inform decision-making processes.

### **7. Support for Different Library Sizes:**

Whether a small community library or a large academic institution, a flexible LMS can scale to meet the specific requirements of different library sizes and types.

### **8. User Permissions and Security:**

LMS provides flexibility in defining user permissions and security settings, ensuring that sensitive information is protected and that access is granted based on designated roles.

In essence, the flexibility of a Library Management System empowers libraries to adapt to their unique environments, optimize workflows, and meet the ever-changing demands of library services.

### **Interactivity**

The interactivity of a Library Management System (LMS) is crucial for cultivating a dynamic and engaging library environment. By fostering user engagement, the LMS transforms the library experience from a traditional, static resource repository into a vibrant, interactive space. Patrons can actively participate through personalized profiles, collaborative tools, and feedback mechanisms, creating a sense of community within the digital library.

This interactivity extends to the catalog, where users can intuitively navigate through resources using features like reviews, ratings, and multimedia elements. The LMS encourages users to contribute to community-driven recommendations, making the exploration of library materials a more interactive and enjoyable experience. Furthermore, collaborative tools, such as discussion forums and group study spaces, facilitate interaction among users, promoting knowledge-sharing and collaboration.

Additionally, an interactive LMS keeps users informed and engaged through real-time updates on new arrivals, events, and changes in library policies. This dynamic flow of information ensures that patrons stay connected with the latest developments within the library. Overall, the interactivity of an LMS enhances user participation, community building, and the overall efficacy of library services in the digital age.

### **Reducing cost**

Reducing costs in a Library Management System (LMS) can be achieved through strategic choices and operational optimizations. Adopting open-source or cloud-based LMS solutions minimizes upfront licensing costs and reduces the need for extensive infrastructure. Additionally, optimizing resource utilization, automating routine tasks, and embracing energy-efficient practices contribute to operational efficiencies, lowering both labor and environmental costs. Proactive system maintenance, updates, and collaborative agreements with vendors or other institutions further enhance cost-effectiveness, allowing libraries to achieve financial efficiency while maintaining the quality of services provided to patrons.

### **Gathering data**

Paperwork is both a hassle to manage and a significant impediment to efficient data gathering. The manual handling of paper-based records introduces complexities in organization, storage, and retrieval, leading to potential errors and inefficiencies in the process. To overcome these challenges, organizations increasingly turn to digital solutions, recognizing the limitations of paperwork in the modern landscape of information management. Leveraging digital tools and technologies not only streamlines data gathering but also enhances accuracy and accessibility,

paving the way for a more efficient and error-resistant approach to handling and utilizing valuable information.

## **2.1 Feasibility Study:**

A feasibility study is conducted so that an inadequate system can be identified early in the defining phase. But when it comes to system engineering, we pay particular attention to four key interest areas. This stage is crucial since it's crucial to determine whether the idea is feasible before moving forward with the actual construction of the system.

**Economic Feasibility:** A comparison between development costs and the eventual revenue or value from the created system.

**Technical Feasibility:** An investigation into how restrictions, performance, and other factors may affect the development of a workable system.

**Operational Feasibility:** An investigation on the system's operational features.

### **2.3.1 Economic Feasibility:**

Cost Benefit Analysis and an evaluation of the economic basis for a project including a computer-based system are two of the most crucial pieces of information in a feasibility study. Cost Benefit Analysis breaks down project development expenses and compares them to the system's direct and indirect advantages. The parameters that change depending on the qualities of the system to be produced, the relative scale of the project, and the expected return on investment desired as part of the company's strategic strategy complicate cost-benefit analyses. In addition, a lot of the advantages of a computer-based system are intangible (example, better design quality through iterative optimization, increased customer satisfaction through programmable control etc). This project doesn't need a significant sum of money, expensive equipment, or infrastructure because it is an intellectual endeavour for our university.

### **2.3.2 Technical Feasibility:**

In this step, we examine the technological feasibility of the proposed system, i.e., whether all the technologies needed to create the system are easily accessible or not. Technical feasibility examines if the organisation has the tools and expertise needed to complete the project, as well as how those skills and experience should be acquired. The following factors can make the system workable:

- Technology is available now to create the system.
- This method can be made even more flexible and versatile.
- This system can provide assurances on correctness, usability, dependability,
- and data security.

Technically speaking, our idea is feasible because we have access to all the necessary technologies.

- **Operating System:** Linux, Windows or Mac
- **Development Platform:** The Library Management System (LMS) is developed in ASP.NET using C# within the Visual Studio development environment, ensuring a robust and scalable web-based application.
- **Database System:** MS SQL
- **Documentation Tool :** MS Word.

### 2.3.3 Operational Feasibility:

The proposed system is undoubtedly entirely GUI based, very user-friendly, and all inputs are self-explanatory even to a layperson. Furthermore, in order to make the users feel at ease with the new system, a suitable training session has been held to explain the system's fundamentals. The project is operationally feasible. The users need only a web browser and an internet connection to gain access to the System which makes it operationally feasible.

### 2.3.4 Social Feasibility:

The proposed system is socially feasible because it does not pose any hindrance to any society or cultural community. Even this project does not produce any kind of material or substance that will be harmful to society.

## 2.2 Project Plan:

Software Project Management Begins with a set of activities that are collectively called **Project Planning**. Before the project begins, the manager and the software team must estimate the work to be done, the resources that will be required, and the time that will elapse from start to finish. Planning involves estimation—your attempt to determine how much money, how much effort, how many resources, and how much time it will take to develop a specific software-based system or product.

Estimate begins with a description of the scope of the product. Until the scope is “**bounded**” its not possible to develop a meaningful estimate. The problem is then decomposed into a set of smaller problems and each of these is estimated using historical data and experience as guides. It is advisable to generate your estimates using at least two different methods ( as a cross check). Problem complexity and risk are considered before a final estimate is make

### **2.3.1 Project Planning Objectives:**

**Software Scope:** The first activity in Software Project Planning is the determination of software scope. Software scope describes the data and control to be processed, function, performance, constraints, interfaces, and reliability.

**Feasibility:** Once the scope has been identified, it is reasonable to ask: “Can we build software to meet this scope? Is the project feasible?”

**Resources:** Another software planning task is the estimation of the resources required to accomplish the software development effort.

**Decomposition:** Software Project Estimation is a form of problem solving, and in most cases, the problem to be solved is too complex to be considered in one piece. For this reason, we decompose the problem, re-characterizing it as a set of smaller problems. Decomposition can be: decomposition of problem and decomposition of process.

### **2.4.1 Project Estimation:**

Project Estimation is an attempt to determine how much money, effort, resources, and time it will take to build a specific software-based system or product. Estimation begins with a description of the scope of the product. The problem is then decomposed into a set of smaller problems, and each of these is estimated using historical data and experience as guides. Problem complexity and risk are considered before a final estimate is made.



#### 2.4.2.1 Title & Scope of Project:

**Project Title:** LIBRARY MANAGEMENT SYSTEM

**Project Scope:** The Library Management System (LMS) project aims to develop a comprehensive software solution for efficient cataloging, circulation, and user management, enhancing the overall functionality and effectiveness of library operations.

**Project Code:** LMS

**Initial Activity:** Project Plan.

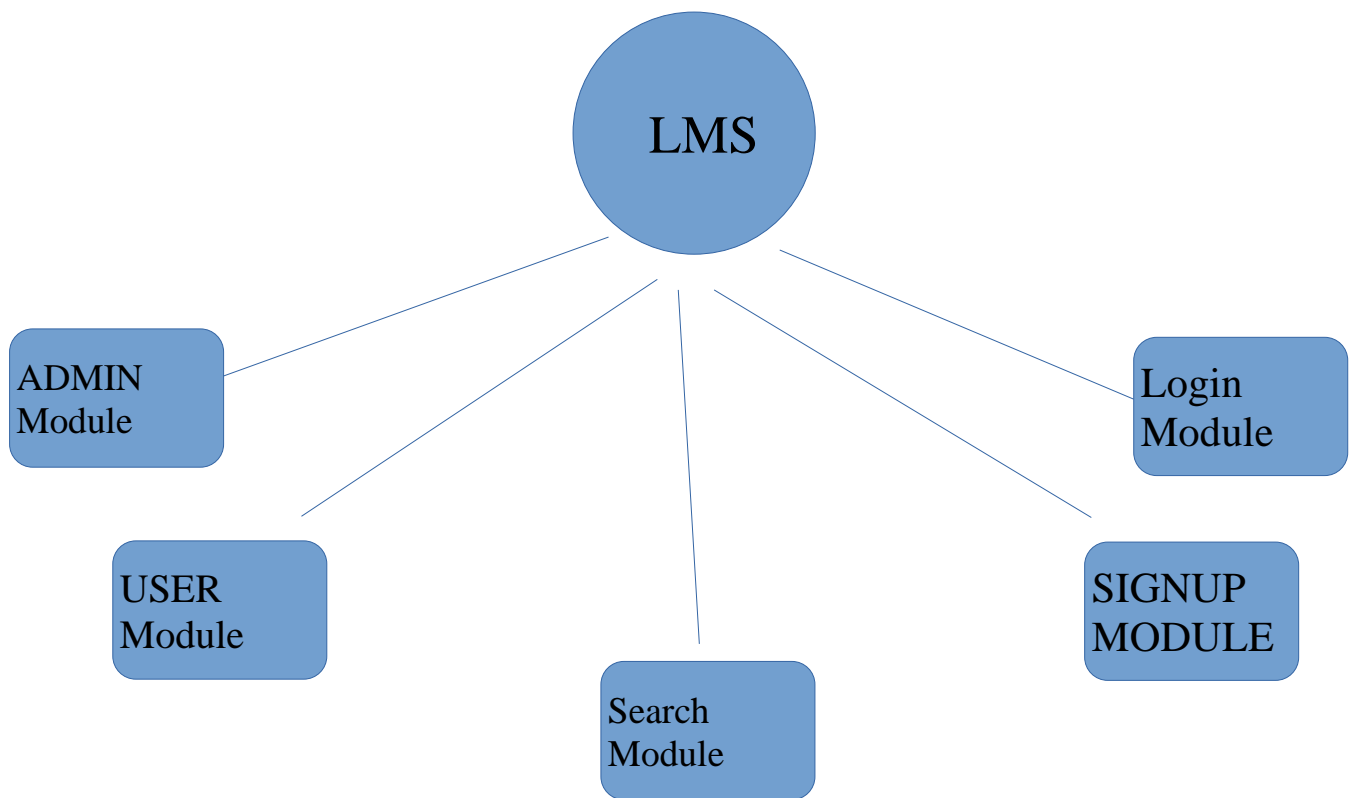
**Final Activity:** Project walkthrough.

#### Client Details:

- **Client Name:** Islamic University of Science & Technology.
- **Client Address:** 1-University Avenue, Awantipora, Kashmir, 192122. J&K, India.
- **Client Contact:** (01933) 247316 , (01933) 24726

#### 2.4.2.2 Project Decomposition:

Since it is a massive project therefore we have decomposed it into different modules for the purpose of easy estimation as shown in below fig:











### **2.4.1 Project Scheduling and Tracking:**

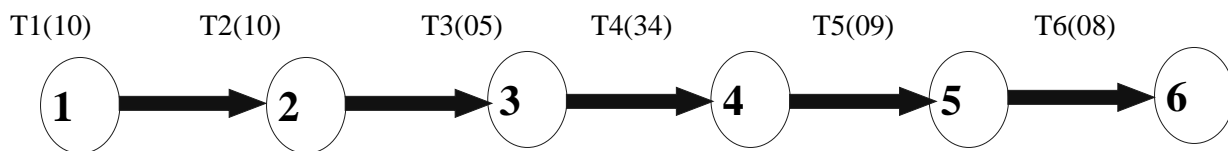
We have selected an appropriate process model, we have identified the software engineering tasks that have to be performed, we estimated the amount of work and the number of people, we know the deadline, we even considered risks. Now it is time to connect the dots. That is, we have to create a network of software engineering tasks that will enable us to get the job done on time. Once the network is created, we have to assign the responsibility for each task, make sure it gets done, and adapt the network as risks become reality. In a nutshell, that is Software Project Scheduling and Tracking.

#### **2.4.3.1 Project Scheduling and Tracking Steps:**

The software engineering tasks dictated by the software process model are refined for the functionality to be built. Effort and duration are allocated to each task and a task network (also called an “activity network”) is created in a manner that enables the software team to meet the delivery deadline established. The output of this phase is the project schedule and related information.

St ep s	Task Name	Start Date	Finish Date	Duration	September	October	Novemb er	Decemb er
01	Requirement Gathering	20-09-2023	30-09-2023	10				
02	Analysis	01-10-2023	10-10-2023	10				
03	Design	10-10-2023	15-10-2023	05				
04	Coding	16-10-2023	18-11-2023	34				
05	Testing	16-11-2023	25-11-2022	09				
06	Report Writing	24-11-2022	02-12-2022	08				

**Figure :- Gantt Chart Work Schedule**



**Figure:- Program Evaluation And Review Technique (PERT) CHART**

## **2.3 Software Engineering Paradigm (Framework):**

### **Rapid Application Development Model:**

Rapid application development is a software development methodology that involves methods like iterative development and software prototyping. According to Whitten (2004), it is a merger of various structured techniques, especially data-driven Information Engineering, with prototyping techniques to accelerate software systems development. In rapid application development, structured techniques and prototyping are especially used to define users' requirements and to design the final system. The development process starts with the development of preliminary data models and business process models using structured techniques. In the next stage, requirements are verified using prototyping, eventually to refine the data and process models. These stages are repeated iteratively; further development results in "a combined business requirements and technical design statement to be used for constructing new systems.

RAD approaches may entail compromises in functionality and performance in exchange for enabling faster development and facilitating application maintenance.

### **Four Phases of RAD:**

**1. Requirements Planning phase** - combines elements of the system planning and systems analysis phases of the System Development Life Cycle (SDLC), Users managers, and IT staff members discuss and agree on business needs, project scope, constraints, and system requirements like the video lectures that are required for the students and where from it can be available for free. It ends when the team agrees on the key issues and obtains management authorization to continue.

**2. User design phase** - During this phase, users interact with systems analysts and develop models and prototypes that represent all system processes, inputs, and outputs. The RAD groups or subgroups typically use a combination of Joint Application Development (JAD) techniques and CASE tools to translate user needs into working models. User Design is a continuous interactive process that allows users to understand, modify, and eventually approve a working model of the system that meets their needs.

**3. Construction phase** - Focuses on program and application development task similar to the SDLC. In RAD, however, users continue to participate and can still suggest changes or improvements as actual screens or reports are developed. Its tasks are programming and application development, coding, unit-integration and system testing. IDE like VS-Code, ATOM helps in implementing the project

**4. Cut-over phase** -Resembles the final tasks in the SDLC implementation phase, including data conversion, testing, changeover to the new system, and user training. Compared with traditional methods, the entire process is compressed. As a result, the new system is built, delivered, and placed in operation much sooner. Its tasks are data conversion, full-scale testing, system changeover, user training. In our project every module are being tested on completion

# ***Chapter 3 System Analysis***

*“How to gather details about the System and the related details”*

# 1. System Analysis

## 3.1 Overview:

System analysis is the process of studying the business processors and procedures generally referred to as business systems, to see how they can operate and whether improvement is needed. This may involve examining data movement and storage, machines and technology used in the system, programs that control the machines, people providing inputs, doing the processing and receiving the outputs.

## 3.2 Requirement Analysis:

Requirement analysis is a software engineering task that bridges the gap between system level requirement engineering and software design as shown in fig. 3.1 below:

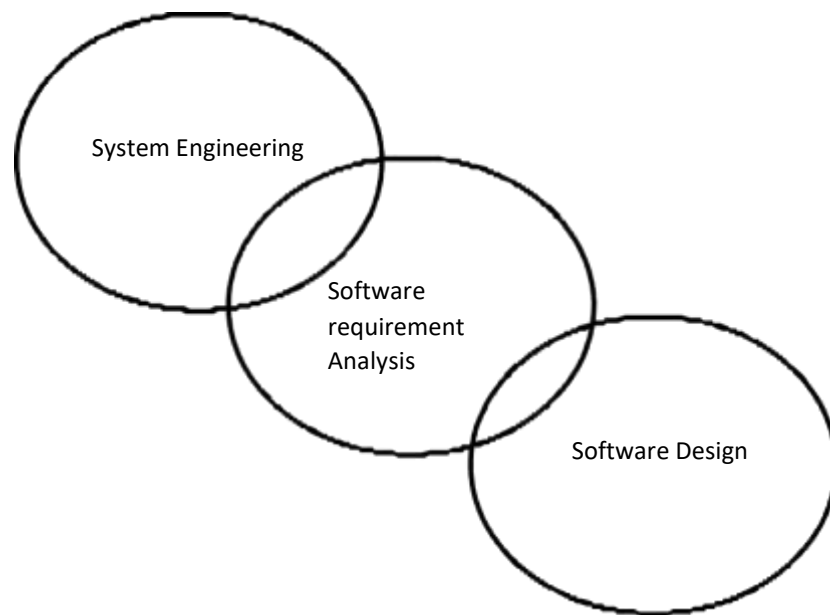


Fig. 3.1 Analysis as a bridge



### **3.3 Requirements Elicitation for Software:**

Before requirements can be analyzed, modeled or specified they must be gathered through an elicitation process. A customer has a problem that may be amenable to computer based solution. A developer responds to the customers request for help. Communication has begun.

In our project since our customer is our own University where we have studies for three years in the same system, we have already recognized the need for such type of system. We conducted a meeting with our project guide who is a user (faculty) of the current System. We discussed the problems of the current system and the solution for that. He forwarded this plan to the department and our proposal was approved.

### **3.4 Analysis Principles:**

Investigators have identified analysis problems and their causes and have developed a variety of modeling notations and corresponding sets of heuristics to overcome them. Each analysis method has a unique point of view. However, all analysis methods are related by a set of operational principles:

### **3.5 Requirement Gathering Techniques:**

Requirements gathering techniques provide project team members with a choice of methods for eliciting needs or requirements from stakeholders and for validating requirements with stakeholders. Certain techniques are appropriate in gathering stakeholder needs, while other techniques are most helpful in defining high-level and detailed requirements, or validating detailed requirements with the stakeholders.

The three recommended techniques are Interview, JAD Session, and Survey Method.

#### **Formal Interview Process Steps:**

1. Identify stakeholders to be interviewed
2. Obtain a general understanding of the customers business
3. Develop interview questions using open-ended questions
4. Set meeting time and location for the interview
5. Provide a set of questions to interviewees prior to the interview (if they will need to prepare for the interview)

6. Use one or more Recorders to accurately preserve results of the interview
7. Provide results to interviewees for confirmation of content

### **Informal Interview Process Steps:**

1. Identify stakeholders to be interviewed
2. Obtain a general understanding of the customers business
3. Develop interview questions (for interviewer's use only) to make sure certain questions are answered during the session
4. Set up a casual meeting or telephone conversation time for the interview.
5. Takes handwritten notes during the interview; avoid using electronic data capture.
6. Provide results to interviewee for confirmation of content

### **3.5.1 Joint Application Development (JAD) Technique:**

The Joint Application Development (JAD) technique is an extended, facilitated workshop. It involves collaboration between stakeholders and systems analysts to identify needs or requirements in a concentrated and focused effort.

#### **JAD Process Steps:**

1. **Define Session:** Define the purpose, scope, and objectives of the JAD session, selecting the JAD team, invite and obtain commitment to attend sessions from the appropriate stakeholders, and schedule the session. It is important to obtain management commitment to support the process and identify the appropriate stakeholders.
2. **Research Product:** Become more familiar with the product or service, gather preliminary information, obtaining any models.
3. **Prepare:** Prepare any visual aids, developing a realistic agenda, training the recorder, and preparing the meeting room.
4. **Conduct Session:** Follow agenda to gather and document the project needs and requirements. It is important to ensure all participants are given equal treatment during the process.
5. **Draft the Documents:** Prepare the formal documents. The information captured in the JAD session is further refined through analysis efforts, open questions or issues discovered through the sessions are resolved, and the final document is returned to stakeholders for review and validation.

### 3.5.2 Survey Method:

The Survey Method is an electronic or paper based method of soliciting needs or requirements from stakeholders. The survey method is a list of questions, directed at identifying stakeholder needs or requirements.

### 3.6 Requirement Gathering for Web Applications:

Although the requirements gathering activity for Web Engineering may be abbreviated, the overall requirements gathering objectives proposed for software engineering remain unchanged. Adapted for WebApps, these objectives become:

- Identify content requirements, like tools, package to develop the project
- Identify functional requirements.
- Define interaction scenarios for different classes of users.
- The following requirement gathering steps have been taken in order to gather requirements for this Project and to achieve the above objectives:
  - Asked department about the user categories and developed courses for each category.
  - Communicated with the department to define the basic WebApp requirements.
  - Thoroughly analyzed the information gathered and used the information to follow up with the department.
  - Defined use-cases that describe interaction scenarios for each user class.

### Performance Requirements

The following performance characteristics should be taken care of while developing the system:

**User friendliness:** The system should be easy to learn and understand so that novice user can also use the system effectively, without any difficulty.

**User satisfaction:** The system should meet user expectations.

**Response time:** The response time of all the operations should be low. This can be made possible by careful programming.

**Error handling:** Response to user errors and the undesired situations should be taken care of to ensure that the system operates without halting.

**Safety:** The system should be able to avoid or tackle catastrophic behaviour.

**Robustness:** The system should recover from undesired events without human intervention.

## **Performance Requirements**

The following acceptance criteria were established for the evaluation of the new system:

**User friendliness:-** The system should meet user needs and should be easy to learn and use.

**Modularity:-** The system should have relatively independent and single function parts .

**Maintainability:-** The system should be such that future maintenance and enhancements times and efforts are reduced.

**Timeliness:-** The system should operate well under normal, peak and recovery conditions.

The system developed should be accurate and hence reliable i.e., the error rate should be minimized and the outputs should be consistent and correct.

Both the execution time and response time should be negligibly low.

The system should be efficient ie. the resources utilization should be optimal.

The system should have scope for future modifications and enhancements ie. it should be able to cope with the changes in future technology.

### **3.7 Analysis Modelling:**

Analysis modeling uses a combination of text and diagrammatic forms to depict requirement for data, function, behavior in a way that is relatively easy to understand, and more important straightforward to review for correctness, completeness and consistency.

#### **Analysis Modelling Steps:**

The data, functional and behavioral requirements are modeled using a number of different diagrammatic formats.

**Data modelling** defines data objects, attributes and relationships.

**Functional modelling** indicates how data are transferred within a system.

**Behavioural modelling** depicts the impact of events.

Once preliminary models are modeled, they are refined and analyzed to assess their clarity, completeness and consistency. A specification incorporating the model is created and then validated by both software engineers, customers/ users.

#### The Elements of the Analysis Modelling:

The analysis model must achieve three primary objectives:

1. to describe what the customer requires,
2. to establish a basis for the creation of software design and
3. to define a set of requirements that can be validated once the software is built.

To accomplish these objectives, the analysis model derived during structured analysis takes the form illustrated in fig. 3.7.

At the core of the model lies the **data dictionary**– a repository that contains description of all data objects consumed or produced by the software. Three different diagrams surround the core. The **entity relation diagram (ERD)** depicts relationships between data objects.

The ERD is the notation that is used to conduct the data modeling activity. The attributes of each data object noted in the ERD can be described using a data object description.

The **Data Flow Diagram (DFD)** serves two purposes: (1) to provide an indication of how data are transformed as they move through the system and (2) to depict the functions (and sub functions) that transform the data flow. The DFD provides additional information that is used during the analysis of the information domain and serves as a basis for the modeling of function. A description of each function presented in the DFD is contained in a *process specification*.

The **State Transition Diagram (STD)** indicates how the system behaves as a consequence of external events. To accomplish this, the STD represents the various modes of behavior (called *states*) of the system and the manner in which transitions are made from state to state. The STD serves as the basis for behavioral modeling. Additional information about the control aspects of the software is contained in the control specification.

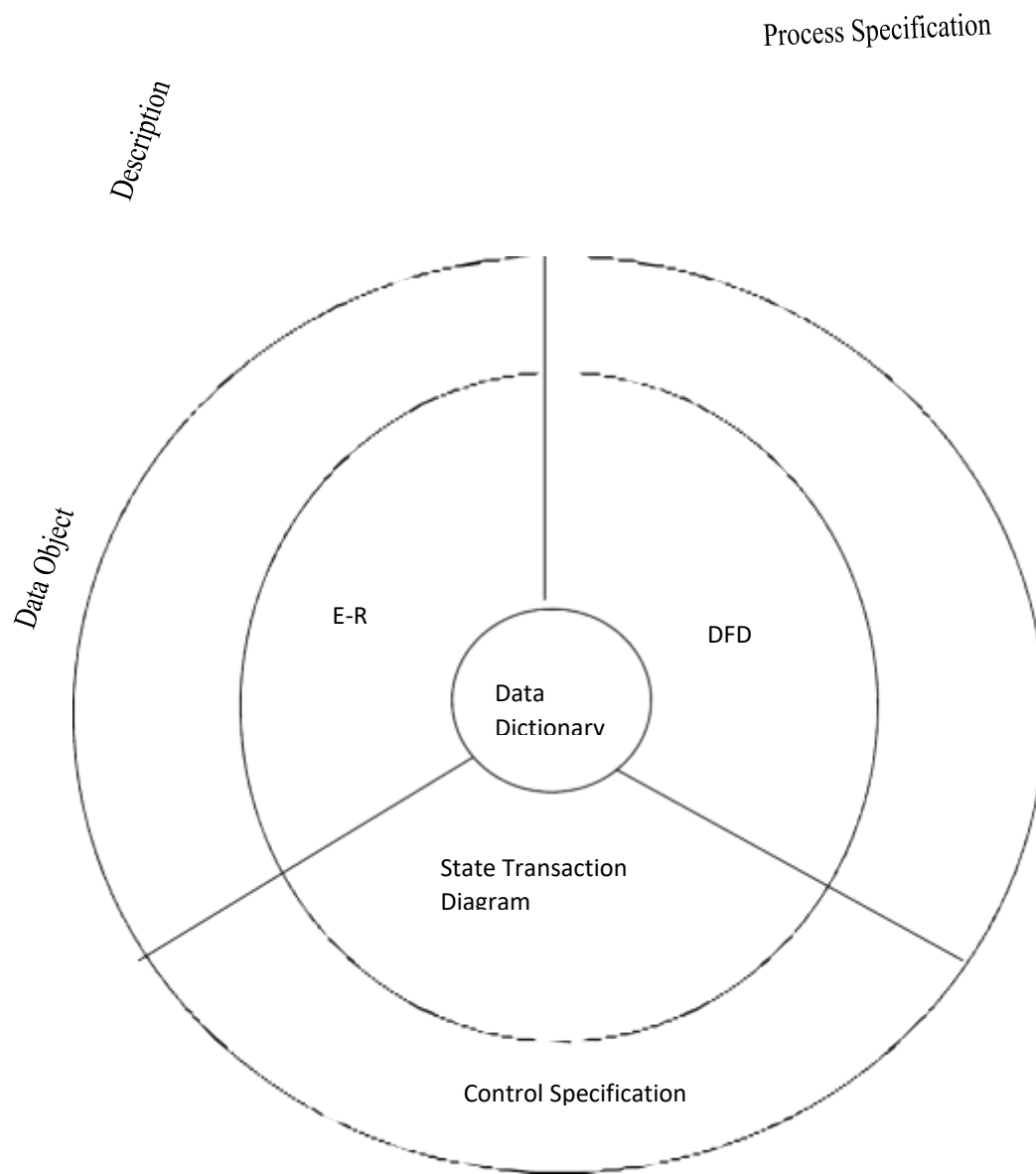
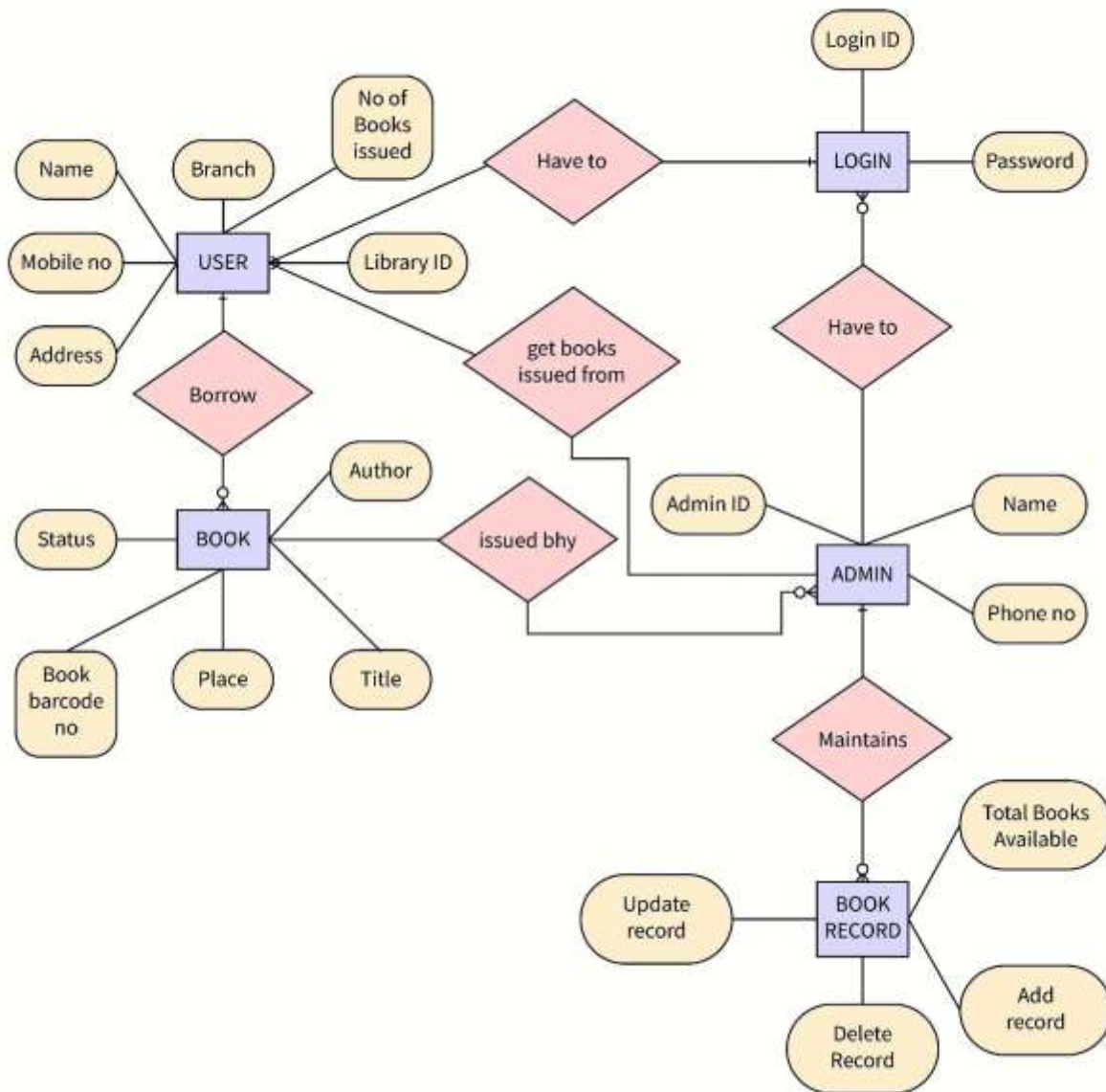


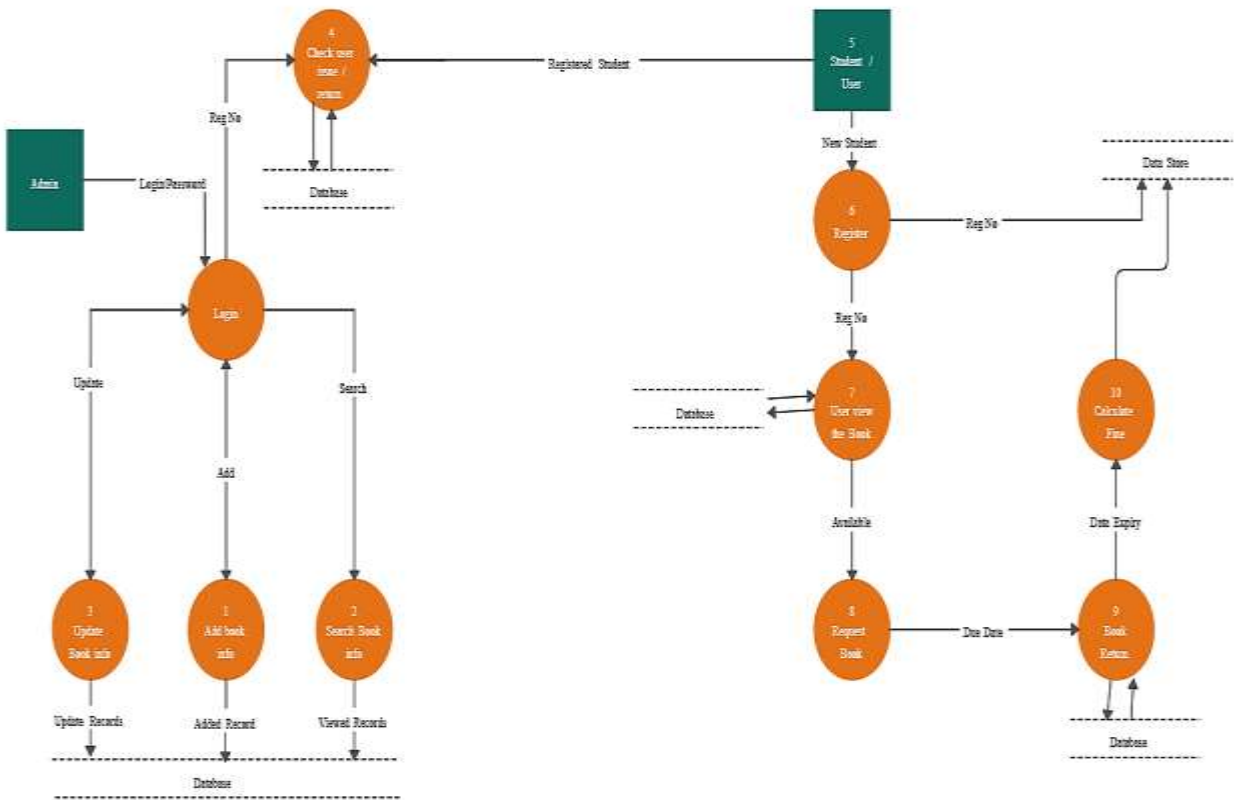
Fig. 3.7 Structure of Analysis Model

### 3.7.1 Data Modeling (ERD)



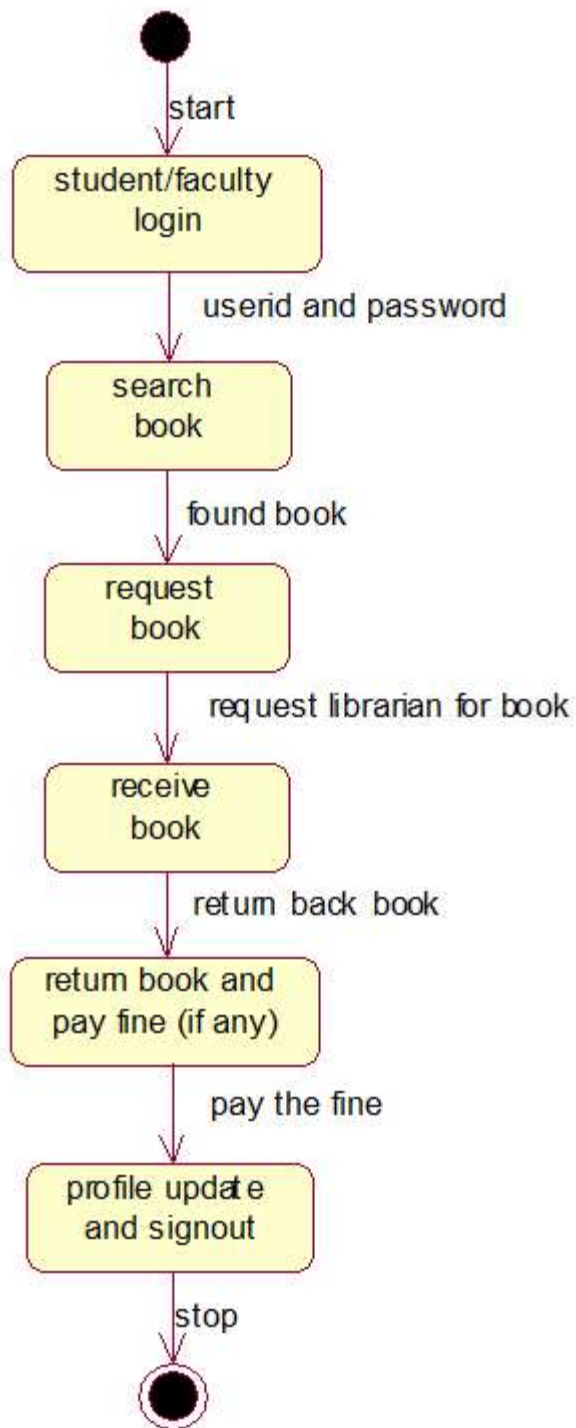
3.7.1 ER Diagram

### 3.7.2 Functional Modelling (DFD)



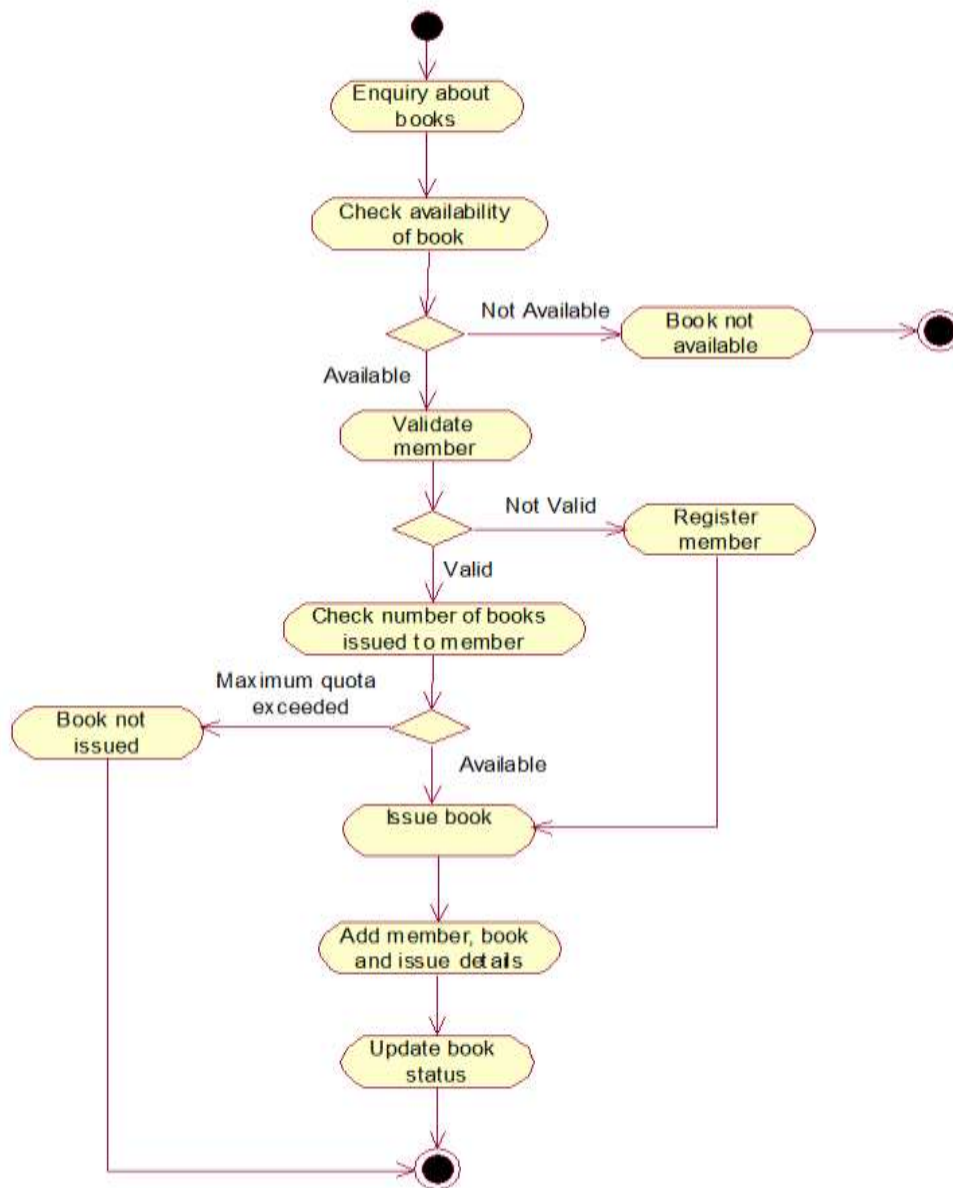
3.7.2 Level -1 DFD



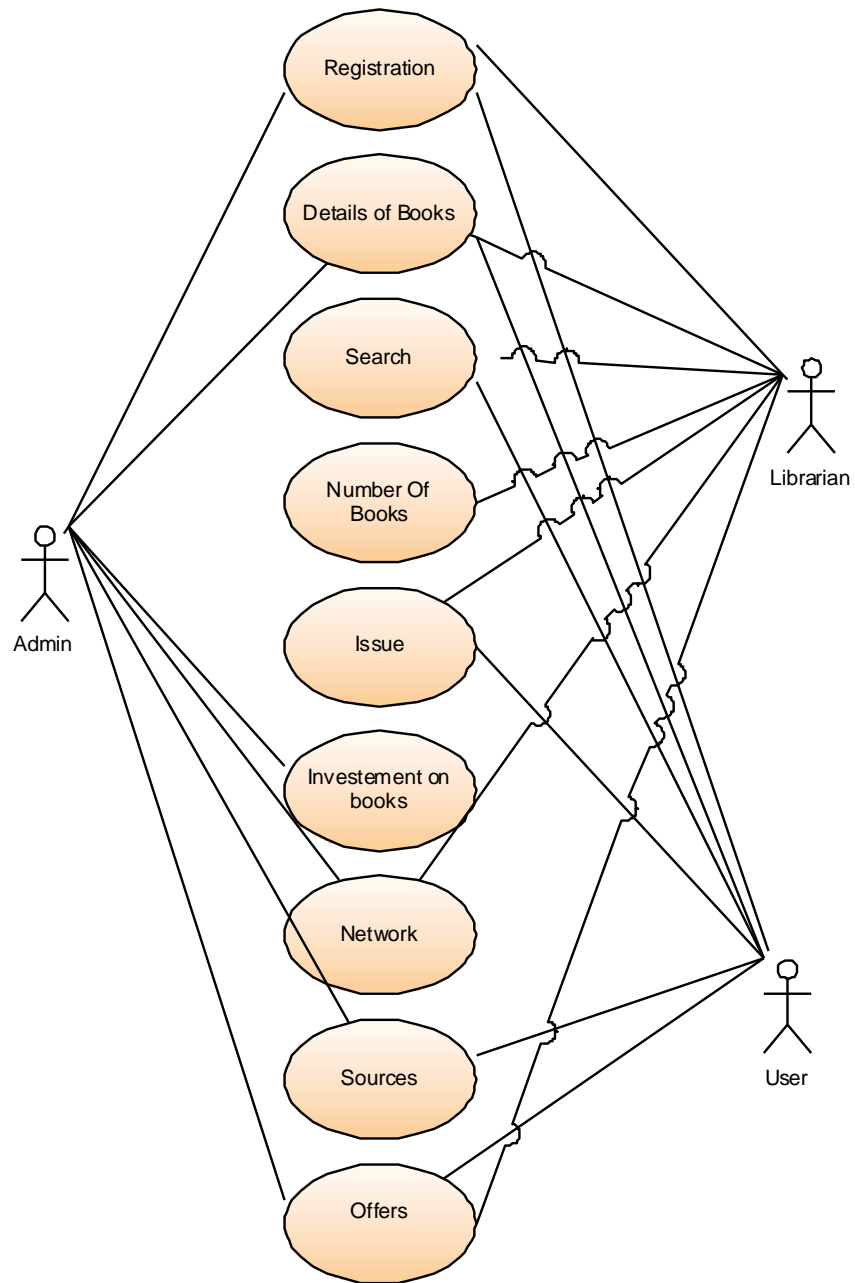


### 3.7.3 State Transition Diagram

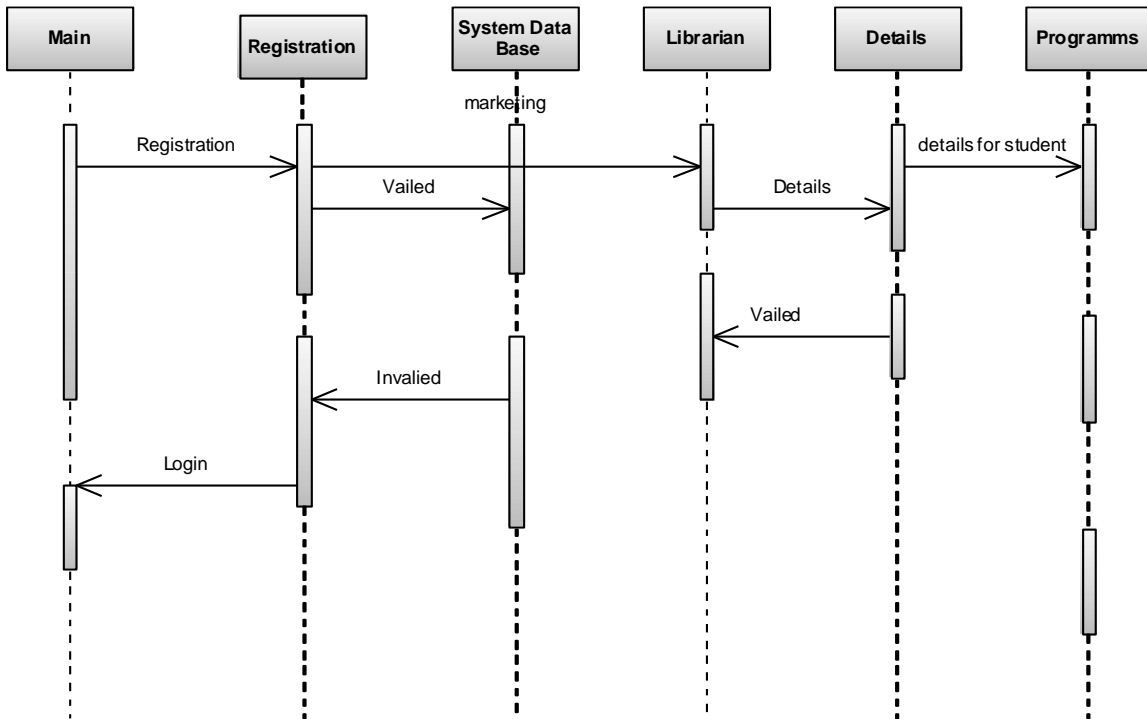
### 3.7.3.1 Activity diagram



### 3.7.3.3 USE-CASE DIAGRAM



### 3.7.4.5 Sequential Diagram



# ***Chapter 4 System Design***

***“Design is a meaningful engineering representation of something that is to be built.”***

## 4. System Design

### 4.1 Overview:

"Design" is a useful engineering term for a depiction of a future construction. It can be linked to a customer's specifications and quality-checked against a list of standards for "excellent" design at the same time. Design in software engineering is concentrated on four key areas: data, architecture, interfaces, and components.

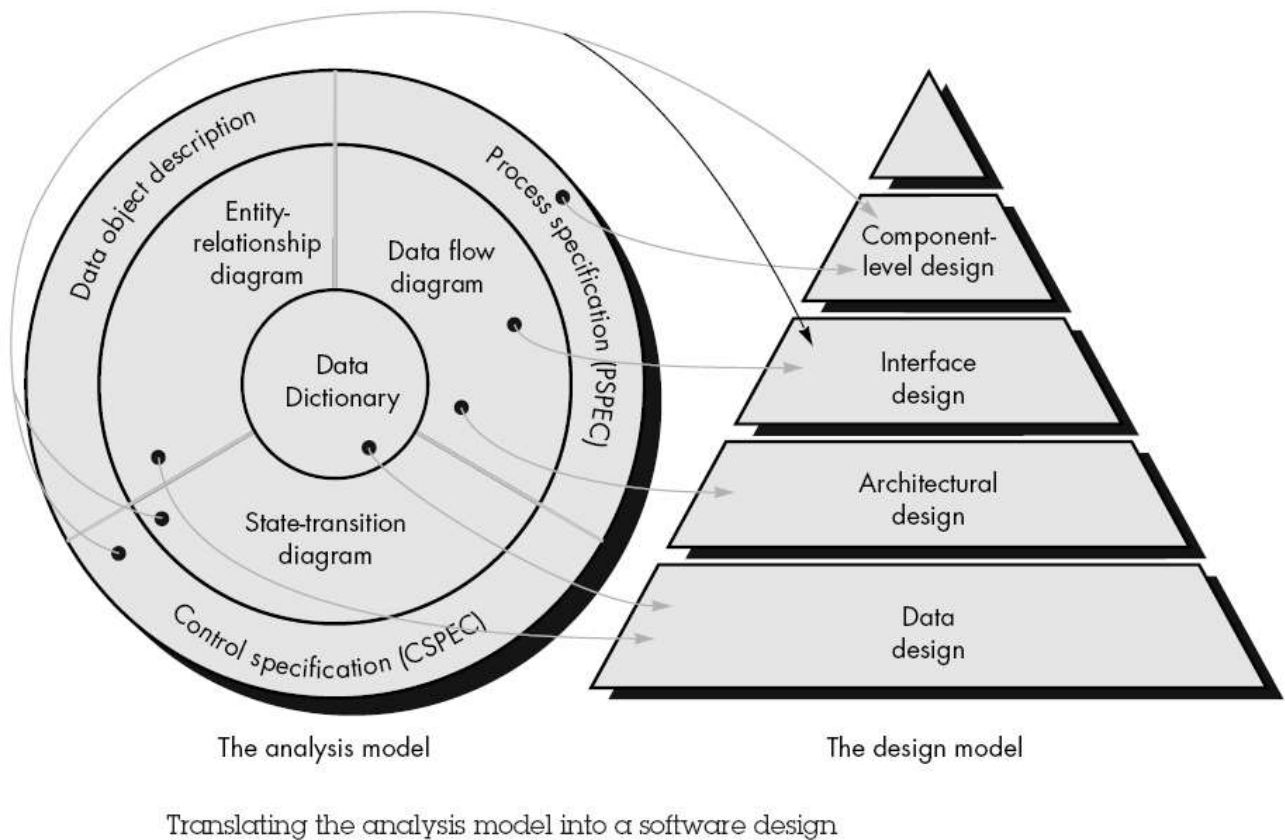
#### 4.1.1 System Design Steps:

Design begins with the requirements model. We work to transform this model into four levels of design detail: the **Data Structure**, the **System Architecture**, the **Interface Representation**, and the **Component Level Detail**. During each design activity, we apply basic concepts and principles that lead to high quality. The outcome of System Design is the Design Specification.

#### 4.1.2 Software Design and Software Engineering:

Software Design sits at the technical kernel of software engineering and is applied regardless of the software process model that is used. Beginning once software requirements have been analyzed and specified, software design is the first of three technical activities— design, code generation, and test—that are required to build and verify the software. Each activity transforms information in a manner that ultimately results in validated computer software.

Each of the elements of the analysis model provides information that is necessary to create the four design models required for a complete specification of design. The flow of information during software design is illustrated in fig.



### 4.1.3 Data Design:

The data design transforms the information domain model created during analysis into the data structures that will be required to implement the software. The data objects and relationships defined in the entity relationship diagram and the detailed data content depicted in the data dictionary provide the basis for the data design activity. Part of data design may occur in conjunction with the design of software architecture. More detailed data design occurs as each software component is designed.

### 4.1.4 Architecture Design:

The architectural design defines the relationship between major structural elements of the software, the “design patterns” that can be used to achieve the requirements that have been defined for the system, and the constraints that effect the way in which architectural design patterns can be applied.

**4.1.5 Interface Design:** The interface design describes how the software communicated within itself, with systems that interoperate with it, and with humans who use it. An interface implies a flow of information (e.g., data and / or control) and a specific type of behaviour. Therefore, data and control flow diagrams provide much of the information required for interface design.

## **4.2 Functional Classification:**

Our web application is divided into two modules:

1. Admin Module

2. User Module

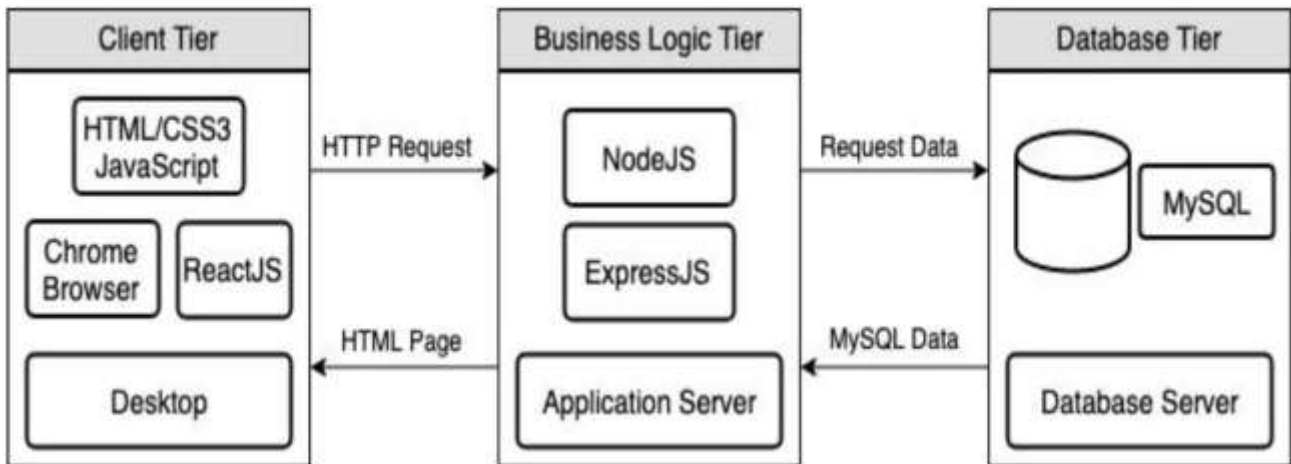
**1. Admin Module:** Admin can log in with their login information. The application's administrator has the most functionality; they have access to user details, manage user browsing history block/unblock specific user.

**2. User Module:** The user can login with his login information. In terms of security, the user can change his or her profile details; they can view and update their profile in the profile section.

**4.1 Architecture Design:** The architecture of our application is based on a typical MVC model. Our Client tier (View) will be written in Javascript, HTML, and CSS, using ReactJS as the framework. This level of the architecture is what the user will interact with to access the features of our application. The Business Logic Tier (Controller) will be written using NodeJs and ExpressJS, and this tier represents the Application Server that will act as a bridge of communication for the Client Tier and Database Tier. This tier will serve HTML pages to the user's device and accept HTTP requests from the user and follow with the appropriate response. Our Database Tier (Model) will be hosting MongoDB. This is where we will store all of the crucial data our application needs to function.



A 3-tier architecture is a web app architecture that is widely used around the world. It basically contains 3 tiers: Client, Server, and Database

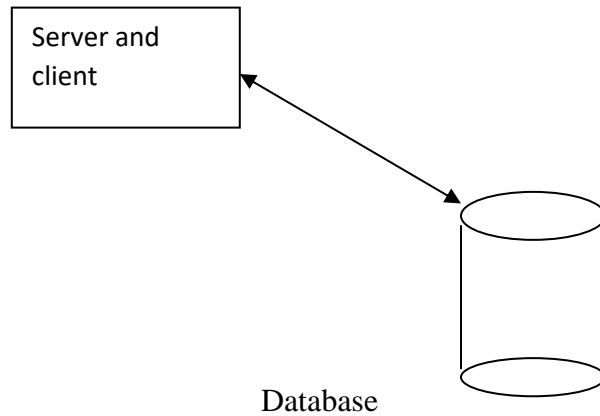


## **DATABASE MODELS**

ADO.NET and accessing the database through applets and ADO.NET API via an intermediate server resulted in a new type of database model which is different from the client-server model. Based on number of intermediate server through the request should go it is named as single tier, two tier and multi tier architecture

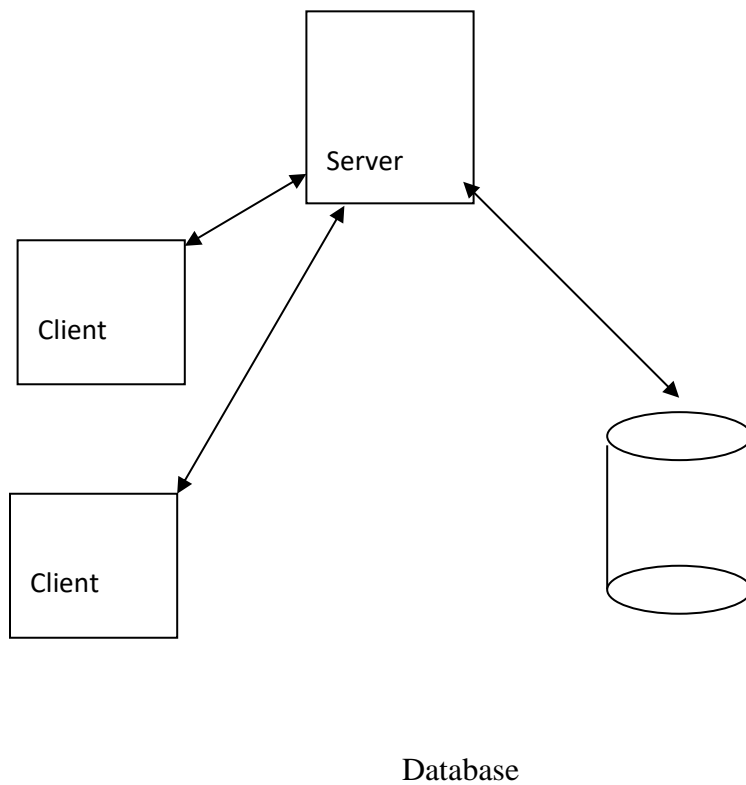
### **Single Tier**

In a single tier the server and client are the same in the sense that a client program that needs information (client) and the source of this type of architecture is also possible in java, in case flat files are used to store the data. However this is useful only in case of small applications. The advantage with this is the simplicity and portability of the application developed.



### Two Tier (client-server)

In two tier architecture the database resides in one machine and client in different machine they are connected through the network. In this type of architecture, a database management takes control of the database and provides access to clients in a network. This software bundle is also called as the server. Software in different machines, requesting for information are called as the clients



### **Three Tier and N-Tier**

In the three-tier architecture, any number servers can access the database that resides on server. Which in turn serve clients in a network. For example, you want to access the database using java applets, the applet running in some other machine, can send request only to the server from which it is down loaded. For this reason we will need to have a intermediate server which will accept the requests from applets and them to the actual database server. This intermediate server acts as a two-way communication channel also. This is the information or data from the database is passed on to the applet that is requesting it. This can be extended to make n tiers of servers, each server carrying to specific type of request from clients, however in practice only 3 tiers architecture is popular.

### **C# Language**

C# (pronounced C Sharp) is a multi-paradigm programming language that encompasses functional, imperative, generic, object-oriented (class-based), and component-oriented programming disciplines. It was developed by Microsoft as part of the .NET initiative and later approved as a standard by ECMA (**ECMA-334**) and ISO (**ISO/IEC 23270**). C# is one of the 44 programming languages supported by the .NET Framework's Common Language Runtime.

C# is intended to be a simple, modern, general-purpose, object-oriented programming language. Anders Hejlsberg, the designer of Delphi, leads the team which is developing C#. It has an object-oriented syntax based on C++ and is heavily influenced by other programming languages such as Delphi and Java. It was initially named Cool, which stood for "C like Object Oriented Language". However, in July 2000, when Microsoft made the project public, the name of the programming language was given as C#. The most recent version of the language is C# 3.0 which was released in conjunction with the .NET Framework 3.5 in 2007. The next proposed version, C# 4.0, is in development.

### **History :-**

In 1996, Sun Microsystems released the Java programming language with Microsoft soon purchasing a license to implement it in their operating system. Java was originally meant to be a

platform independent language, but Microsoft, in their implementation, broke their license agreement and made a few changes that would essentially inhibit Java's platform-independent capabilities. Sun filed a lawsuit and Microsoft settled, deciding to create their own version of a partially compiled, partially interpreted object-oriented programming language with syntax closely related to that of C++.

During the development of .NET, the class libraries were originally written in a language/compiler called Simple Managed C (SMC). In January 1999, Anders Hejlsberg formed a team to build a new language at the time called Cool, which stood for "C like Object Oriented Language". Microsoft had considered keeping the name "Cool" as the final name of the language, but chose not to do so for trademark reasons. By the time the .NET project was publicly announced at the July 2000 Professional Developers Conference, the language had been renamed C#, and the class libraries and ASP.NET runtime had been ported to C#.

C#'s principal designer and lead architect at Microsoft is Anders Hejlsberg, who was previously involved with the design of Visual J++, Borland Delphi, and Turbo Pascal. In interviews and technical papers he has stated that flaws in most major programming languages (e.g. C++, Java, Delphi, and Smalltalk) drove the fundamentals of the Common Language Runtime (CLR), which, in turn, drove the design of the C# programming language itself. Some argue that C# shares roots in other languages.

### **Features of C# :-**

By design, C# is the programming language that most directly reflects the underlying Common Language Infrastructure (CLI). Most of C#'s intrinsic types correspond to value-types implemented by the CLI framework. However, the C# language specification does not state the code generation requirements of the compiler: that is, it does not state that a C# compiler must target a Common Language Runtime (CLR), or generate Common Intermediate Language (CIL), or generate any other specific format. Theoretically, a C# compiler could generate machine code like traditional compilers of C++ or FORTRAN; in practice, all existing C# implementations target CIL.

Some notable C# distinguishing features are:

- There are no global variables or functions. All methods and members must be declared within classes. It is possible, however, to use static methods/variables within public classes instead of global variables/functions.
- Local variables cannot shadow variables of the enclosing block, unlike C and C++. Variable shadowing is often considered confusing by C++ texts.
- C# supports a strict Boolean data type, `bool`. Statements that take conditions, such as `while` and `if`, require an expression of a boolean type. While C++ also has a boolean type, it can be freely converted to and from integers, and expressions such as `if(a)` require only that `a` is convertible to `bool`, allowing `a` to be an `int`, or a pointer. C# disallows this "integer meaning true or false" approach on the grounds that forcing programmers to use expressions that return exactly `bool` can prevent certain types of programming mistakes such as `if (a = b)` (use of `=` instead of `==`).
- In C#, memory address pointers can only be used within blocks specifically marked as *unsafe*, and programs with unsafe code need appropriate permissions to run. Most object access is done through safe object references, which are always either pointing to a valid, existing object, or have the well-defined null value; a reference to a garbage-collected object, or to random block of memory, is impossible to obtain. An unsafe pointer can point to an instance of a value-type, array, string, or a block of memory allocated on a stack. Code that is not marked as unsafe can still store and manipulate pointers through the `System.IntPtr` type, but cannot dereference them.
- Managed memory cannot be explicitly freed, but is automatically garbage collected. Garbage collection addresses memory leaks. C# also provides direct support for deterministic finalization with the `using` statement (supporting the Resource Acquisition Is Initialization idiom).
- Multiple inheritance is not supported, although a class can implement any number of interfaces. This was a design decision by the language's lead architect to avoid complication, avoid dependency hell and simplify architectural requirements throughout CLI.

- C# is more type safe than C++. The only implicit conversions by default are those which are considered safe, such as widening of integers and conversion from a derived type to a base type. This is enforced at compile-time, during JIT, and, in some cases, at runtime. There are no implicit conversions between booleans and integers, nor between enumeration members and integers (except for literal 0, which can be implicitly converted to any enumerated type). Any user-defined conversion must be explicitly marked as explicit or implicit, unlike C++ copy constructors (which are implicit by default) and conversion operators (which are always implicit).
- Enumeration members are placed in their own scope.
- C# provides syntactic sugar for a common pattern of a pair of methods, accessor (getter) and mutator (setter) encapsulating operations on a single attribute of a class, in form of properties.
- Full type reflection and discovery is available.
- C# currently (as of 3 June 2008) has 77 reserved words.

### **Common Type system (CTS)**

C# has a unified type system. This unified type system is called Common Type System (CTS). A unified type system implies that all types, including primitives such as integers, are subclasses of the System.Object class. For example, every type inherits a ToString() method. For performance reasons, primitive types (and value types in general) are internally allocated on the stack.

Categories of datatypes

**CTS separates datatypes into two categories:**

- Value types
- Reference types

Value types are plain aggregations of data. Instances of value types do not have referential identity nor a referential comparison semantics - equality and inequality comparisons for value types compare the actual data values within the instances, unless the corresponding operators are overloaded. Value types are derived from System.ValueType, always have a default value, and can always be created and copied. Some other limitations on value types are that they cannot

derive from each other (but can implement interfaces) and cannot have a default (parameterless) constructor. Examples of value types are some primitive types, such as `int` (a signed 32-bit integer), `float` (a 32-bit IEEE floating-point number), `char` (a 16-bit Unicode codepoint), and `System.DateTime` (identifies a specific point in time with millisecond precision).

In contrast, reference types have the notion of referential identity - each instance of reference type is inherently distinct from every other instance, even if the data within both instances is the same. This is reflected in default equality and inequality comparisons for reference types, which test for referential rather than structural equality, unless the corresponding operators are overloaded (such as the case for `System.String`). In general, it is not always possible to create an instance of a reference type, nor to copy an existing instance, or perform a value comparison on two existing instances, though specific reference types can provide such services by exposing a public constructor or implementing a corresponding interface (such as `ICloneable` or `IComparable`). Examples of reference types are `object` (the ultimate base class for all other C# classes), `System.String` (a string of Unicode characters), and `System.Array` (a base class for all C# arrays).

Both type categories are extensible with user-defined types.

## **Boxing and unboxing**

*Boxing* is the operation of converting a value of a value type into a value of a corresponding reference type.

### **Example:**

```
int foo = 42;      // Value type...
object bar = foo;  // foo is boxed to bar.
```

Unboxing is the operation of converting a value of a reference type (previously boxed) into a value of a value type.

### **Example:**

```
int foo = 42;    // Value type.
object bar = foo; // foo is boxed to bar.
int foo2 = (int)bar; // Unboxed back to value type.
```

## **Features of C# 2.0**

New features in C# for the .NET SDK 2.0 (corresponding to the 3rd edition of the ECMA-334 standard) are:

### **Partial class**

Partial classes allow implementation of a class to be spread between several files, with each file containing one or more class members. It is primary useful when parts of a class are automatically generated. For example, the feature is heavily used by code-generating user interface designers in Visual Studio.

file1.cs:

```
public partial class MyClass
{
    public void MyMethod1()
    {
        // Manually written code
    }
}
```

file2.cs:

```
public partial class MyClass
{
    public void MyMethod2()
    {
        // Automatically generated code
    }
}
```

### **Generics**

Generics, or parameterized types, or parametric polymorphism is a .NET 2.0 feature supported by C#. Unlike C++ templates, .NET parameterized types are instantiated at runtime rather than by the compiler; hence they can be cross-language whereas C++ templates cannot. They support



some features not supported directly by C++ templates such as type constraints on generic parameters by use of interfaces. On the other hand, C# does not support non-type generic parameters. Unlike generics in Java, .NET generics use reification to make parameterized types first-class objects in the CLI Virtual Machine, which allows for optimizations and preservation of the type information.

## **Static classes**

Static classes are classes that cannot be instantiated or inherited from, and that only allow static members. Their purpose is similar to that of modules in many procedural languages.

A new form of iterator providing generator functionality

A new form of iterator that provides generator functionality, using a yield return construct similar to yield in Python.

```
// Method that takes an iterable input (possibly an array)
// and returns all even numbers.
public static IEnumerable<int> GetEven(IEnumerable<int> numbers)
{
    foreach (int i in numbers)
    {
        if (i % 2 == 0) yield return i;
    }
}
```

## **Anonymous delegates**

Anonymous delegates provide closure functionality in C#. Code inside the body of an anonymous delegate has full read/write access to local variables, method parameters, and class members in scope of the delegate, excepting out and ref parameters. For example:-

```
int SumOfArrayElements(int[] array)
{
```

```

int sum = 0;
Array.ForEach(
    array,
    delegate(int x)
    {
        sum += x;
    }
);
return sum;
}

```

### **Delegate covariance and contravariance**

Conversions from method groups to delegate types are covariant and contra variant in return and parameter types, respectively.

### **The accessibility of property accessors can be set independently**

Example:

```
string status = string.Empty;
```

```

public string Status
{
    get { return status; }           // anyone can get value of this property,
    protected set { status = value; } // but only derived classes can change it
}

```

### **Nullable types**

Nullable value types (denoted by a question mark, e.g. `int? i = null;`) which add null to the set of allowed values for any value type. This provides improved interaction with SQL databases, which can have nullable columns of types corresponding to C# primitive types: an SQL `INTEGER NULL` column type directly translates to the C# `int?` Nullable types received an eleventh-hour improvement at the end of August 2005, mere weeks before the official launch, to

improve their boxing characteristics: a nullable variable which is assigned null is not actually a null reference, but rather an instance of struct Nullable<T> with property HasValue equal to false. When boxed, the Nullable instance itself is boxed, and not the value stored in it, so the resulting reference would always be non-null, even for null values. The following code illustrates the corrected flaw:

```
int? i = null;
object o = i;
if (o == null)
    Console.WriteLine ("Correct behavior - runtime version from September 2005 or later");
else
    Console.WriteLine ("Incorrect behavior - pre-release runtime (from before September 2005)");
```

When copied into objects, the official release boxes values from Nullable instances, so null values and null references are considered equal. The late nature of this fix caused some controversy, since it required core-CLR changes affecting not only .NET2, but all dependent technologies (including C#, VB, SQL Server 2005 and Visual Studio 2005).

## **DATABASE TABLES:**

### **1.) User SignUp**

Full Name	nvarchar(MAX),,
DOB	nvarchar(50),
Email	nvarchar(50),
City	nvarchar(50),
Pincode	nvarchar(50),,
Full Address	nvarchar(MAX),,
Password	nvarchar(MAX),,
Registration No	nvarchar(50),,
Account_Status	nvarchar(MAX),,

## 2.) Book Issue Table

BOOKID	INT NULL,
Registration No	nvarchar(50)
Book ID	nvarchar(50)
User Name	nvarchar(50)
Book Name	nvarchar(50)
Issue Date	nvarchar(50)
Due Date	nvarchar(50)

## 3.) Book Inventory Table

Book Id	nvarchar(50)
Book Name	nvarchar(50)
Genre	nvarchar(50)
Author_Name	nvarchar(50)
Publisher_Name	nvarchar(50)
Language	nvarchar(50)
Edition	nvarchar(50)
Book_Cost	nvarchar(50)
No_of_Pages	nvarchar(50)
Book_Description	nvarchar(MAX)
Actual_Stock	nvarchar(50)
Current_Stock	nvarchar(50)
Book_img	nvarchar(MAX)

#### 4.) Publisher Table

Publisher Name	nvarchar(50),
Publisher Id	nvarchar(MAX),

#### 4.) Author Table

Author Name	nvarchar(50),
Author Id	nvarchar(MAX),

#### 4.) Admin Table

Username	nvarchar(50),
Password	nvarchar(MAX),

#### Database Connection

```
<configuration>
  <connectionStrings>
    <add name="con" connectionString="Data Source=ZAMEERMIR;Initial
Catalog=elibraryDB;Integrated Security=true" />
    <add name="elibraryDBConnectionString" connectionString="Data
Source=ZAMEERMIR;Initial Catalog=elibraryDB;Integrated Security=True"
providerName="System.Data.SqlClient" />
    <add name="elibraryDBConnectionString2" connectionString="Data
Source=ZAMEERMIR;Initial Catalog=elibraryDB;Integrated Security=True"
providerName="System.Data.SqlClient" />
  </connectionStrings>
```

# *Chapter 5 Coding*

*“Have a look at the hard code”*

## 5.0 Code:

### 5.1 Code for User Signup:

#### Front End

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site1.Master" AutoEventWireup="true"
CodeBehind="usersignup.aspx.cs" Inherits="Lms.usersignup" %>
<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat="server">
    <div class="container">
        <div class="row">
            <div class="col-md-8 mx-auto">

                <div class="card">

                    <div class="card-body">

                        <div class="row">

                            <div class="col">
                                <center>
                                    
                                </center>
                            </div>
                        </div>
                        <div class="row">

                            <div class="col">
                                <center>
                                    <h4>User Registration</h4>
                                </center>
                            </div>
                        </div>
                        <div class="row">

                            <div class="col">
                                <hr>
                            </div>
                        </div>
                        <div class="row">

                            <div class="col-md-6">
                                <div class="form-group">
                                    <label>Full Name</label>
                                    <asp:TextBox CssClass="form-control" ID="TextBox1"
runat="server" placeholder="Enter Your Full Name"></asp:TextBox>
                                </div>
                            </div>
                            <div class="col-md-6">
                                <div class="form-group">
                                    <label>Date Of Birth</label>
                                    <asp:TextBox CssClass="form-control" ID="TextBox2"
runat="server" placeholder="User Password" TextMode="Date"></asp:TextBox>
```

```

        </div>
    </div>
</div>
<div class="row">
    <div class="col-md-6">
        <div class="form-group">
            <label>Contact Number</label>
            <asp:TextBox CssClass="form-control" ID="TextBox3"
runat="server" placeholder="Contact No" TextMode="Number"></asp:TextBox>

        </div>
    </div>
    <div class="col-md-6">
        <div class="form-group">
            <label>Email</label>
            <asp:TextBox CssClass="form-control" ID="TextBox4"
runat="server" placeholder="Email ID" TextMode="Email"></asp:TextBox>

        </div>
    </div>
</div>

<div class="row">
    <div class="col-md-4">
        <div class="form-group">
            <label>State</label>
            <asp:DropDownList CssClass="form-control" ID="DropDownList1"
runat="server">
                <asp:ListItem Text="Select" Value="Select" />
                <asp:ListItem Text="Select" Value="select" />
                <asp:ListItem Text="Andhra Pradesh" Value="Andhra
Pradesh" />
                <asp:ListItem Text="Arunachal Pradesh" Value="Arunachal
Pradesh" />
                <asp:ListItem Text="Assam" Value="Assam" />
                <asp:ListItem Text="Bihar" Value="Bihar" />
                <asp:ListItem Text="Chhattisgarh" Value="Chhattisgarh" />
                <asp:ListItem Text="Rajasthan" Value="Rajasthan" />
                <asp:ListItem Text="Goa" Value="Goa" />
                <asp:ListItem Text="Gujarat" Value="Gujarat" />
                <asp:ListItem Text="Haryana" Value="Haryana" />
                <asp:ListItem Text="Himachal Pradesh" Value="Himachal
Pradesh" />
                <asp:ListItem Text="Jammu and Kashmir" Value="Jammu and
Kashmir" />
                <asp:ListItem Text="Jharkhand" Value="Jharkhand" />
                <asp:ListItem Text="Karnataka" Value="Karnataka" />
                <asp:ListItem Text="Kerala" Value="Kerala" />
                <asp:ListItem Text="Madhya Pradesh" Value="Madhya
Pradesh" />
                <asp:ListItem Text="Maharashtra" Value="Maharashtra" />
                <asp:ListItem Text="Manipur" Value="Manipur" />
                <asp:ListItem Text="Meghalaya" Value="Meghalaya" />
                <asp:ListItem Text="Mizoram" Value="Mizoram" />
                <asp:ListItem Text="Nagaland" Value="Nagaland" />
                <asp:ListItem Text="Odisha" Value="Odisha" />
                <asp:ListItem Text="Punjab" Value="Punjab" />
                <asp:ListItem Text="Rajasthan" Value="Rajasthan" />
                <asp:ListItem Text="Sikkim" Value="Sikkim" />
                <asp:ListItem Text="Tamil Nadu" Value="Tamil Nadu" />
                <asp:ListItem Text="Telangana" Value="Telangana" />

```



```

        <asp:ListItem Text="Tripura" Value="Tripura" />
        <asp:ListItem Text="Uttar Pradesh" Value="Uttar Pradesh"
/>

        <asp:ListItem Text="Uttarakhand" Value="Uttarakhand" />
        <asp:ListItem Text="West Bengal" Value="West Bengal" />
    </asp:DropDownList>

    </div>

</div>
<div class="col-md-4">
    <div class="form-group">
        <label>City</label>
        <asp:TextBox CssClass="form-control" ID="TextBox6"
runat="server" placeholder="City"></asp:TextBox>

    </div>
</div>
<div class="col-md-4">
    <div class="form-group">
        <label>Pincode</label>
        <asp:TextBox CssClass="form-control" ID="TextBox7"
runat="server" placeholder="Pincode" TextMode="Number"></asp:TextBox>

    </div>
</div>
</div>
<div class="col">
    <div class="form-group">
        <label>Full Address</label>
        <asp:TextBox CssClass="form-control" ID="TextBox5" runat="server"
placeholder="Enter your Full Address" TextMode="MultiLine" Rows="2"></asp:TextBox>

    </div>
</div>
<div class="row">
    <div class="col">
        <center>
            <span class="badge baddge-pill badge-dark">Login
Credentials</span>

        </center>
    </div>

</div>

<div class="row">
    <div class="col-md-6">
        <div class="form-group">
            <label>Registration No</label>
            <asp:TextBox Class="form-control" ID="TextBox8"
runat="server" placeholder="Registration NO"></asp:TextBox>

        </div>

    </div>
    <div class="col-md-6">
        <div class="form-group">
            <label>Password</label>
            <asp:TextBox Class="form-control" ID="TextBox9"
runat="server" placeholder="Password" TextMode="Password"></asp:TextBox>

        </div>
    </div>
</div>
<div class="row">

```

```

        <div class="col">

            <div class="form-group">
                <asp:Button CssClass="btn btn-primary btn-block btn-lg"
ID="Button1" runat="server" Text="Sign Up" OnClick="Button1_Click" />
            </div>

        </div>

    </div>
    <div class="row">
    </div>
</div>

    </div>
    <a href="homepage.aspx"><< Back to Home</a><br>
    <br>
</div>
</div>
</asp:Content>

```

## Back\_End

```

namespace Lms
{
    public partial class usersignup : System.Web.UI.Page
    {
        string strcon = ConfigurationManager.ConnectionStrings["con"].ConnectionString;
        protected void Page_Load(object sender, EventArgs e)
        {

        }
        //signup click event
        protected void Button1_Click(object sender, EventArgs e)
        {

            if (checkMemberExist())
            {
                Response.Write("<script> alert('User Already Exists.Please try by other
registration no');</script>");
            }
            else
            {
                SignupNewUser();
            }
        }
        //user defined methods
        bool checkMemberExist()
        {

            SqlConnection con = new SqlConnection(strcon);
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }

            SqlCommand cmd = new SqlCommand("select * from member_master_tbl where
registration_no='" + TextBox8.Text.Trim() + "'", con);

```

```

        SqlDataAdapter da = new SqlDataAdapter(cmd);
        DataTable dt = new DataTable();
        da.Fill(dt);
        if (dt.Rows.Count >= 1)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
}

void SignupNewUser()
{
    try
    {
        SqlConnection con = new SqlConnection(strcon);
        if (con.State == System.Data.ConnectionState.Closed)
        {
            con.Open();
        }
        string hash= HashPassword(TextBox9.Text.Trim());
        SqlCommand cmd = new SqlCommand("INSERT INTO
member_master_tbl(full_name,dob,contact_no,email,state,city,pincode,full_address,registration_no,
password,account_status)
values(@full_name,@dob,@contact_no,@email,@state,@city,@pincode,@full_address,@registration_no,@p
assword,@account_status)", con);
        cmd.Parameters.AddWithValue("@full_name", TextBox1.Text.Trim());
        cmd.Parameters.AddWithValue("@dob", TextBox2.Text.Trim());
        cmd.Parameters.AddWithValue("@contact_no", TextBox3.Text.Trim());
        cmd.Parameters.AddWithValue("@email", TextBox4.Text.Trim());
        cmd.Parameters.AddWithValue("@state", DropDownList1.SelectedItem.Value);
        cmd.Parameters.AddWithValue("@city", TextBox6.Text.Trim());
        cmd.Parameters.AddWithValue("@pincode", TextBox7.Text.Trim());
        cmd.Parameters.AddWithValue("@full_address", TextBox5.Text.Trim());
        cmd.Parameters.AddWithValue("@registration_no", TextBox8.Text.Trim());
        cmd.Parameters.AddWithValue("@password", hash);
        cmd.Parameters.AddWithValue("@account_status", "pending");

        cmd.ExecuteNonQuery();
        con.Close();
        Response.Write("<script>alert('Sign up Successful. Go to User
Login');</script>");
    }

    catch (Exception ex)
    {
        Response.Write("<script>alert('\" + ex.Message + '\" );</script>");
    }
}

string HashPassword(string password)
{
    using (SHA256 sha256 = SHA256.Create())
    {
        // Compute hash from the password bytes
        byte[] hashedBytes = sha256.ComputeHash(Encoding.UTF8.GetBytes(password));

        // Convert the byte array to a string representation
        StringBuilder builder = new StringBuilder();
        for (int i = 0; i < hashedBytes.Length; i++)
        {
            builder.Append(hashedBytes[i].ToString("x2"));
        }
    }
}

```

```

        return builder.ToString();
    }
}
}

```

## Code for Admin Login :

Front\_End

```

<%@ Page Title="" Language="C#" MasterPageFile="~/Site1.Master" AutoEventWireup="true"
CodeBehind="adminlogin.aspx.cs" Inherits="Lms.adminlogin" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat="server">
    <div class="container">
        <div class="row">
            <div class="col-md-6 mx-auto">

                <div class="card">

                    <div class="card-body">

                        <div class="row">

                            <div class="col">
                                <center>
                                    
                                </center>
                            </div>
                        </div>
                        <div class="row">

                            <div class="col">
                                <center>
                                    <h3>Admin Login</h3>
                                </center>
                            </div>
                        </div>
                        <div class="row">

                            <div class="col">
                                <hr>
                            </div>
                        </div>
                        <div class="row">

                            <div class="col">
                                <div class="form-group">
                                    <label>Admin ID</label>

```

```

        <asp:TextBox CssClass="form-control" ID="TextBox1"
runat="server" placeholder="Admin ID"></asp:TextBox>

        </div>

        <div class="form-group">
            <label>Password</label>
            <asp:TextBox CssClass="form-control" ID="TextBox2"
runat="server" placeholder="Password" TextMode="Password"></asp:TextBox>

        </div>
        <div class="form-group">
            <asp:Button CssClass="btn btn-primary btn-block btn-lg"
ID="Button1" runat="server" Text="Login" OnClick="Button1_Click" />

        </div>

    </div>

</div>
</div>
</div>
</div>
<a href="homepage.aspx"><< Back to Home</a><br>
<br>
</div>
</div>
</div>
</asp:Content>

```

## Back\_End

```

namespace Lms
{
    public partial class adminlogin : System.Web.UI.Page
    {
        string strcon = ConfigurationManager.ConnectionStrings["con"].ConnectionString;
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void Button1_Click(object sender, EventArgs e)
        {
            try
            {
                SqlConnection con = new SqlConnection(strcon);
                if (con.State == System.Data.ConnectionState.Closed)
                {
                    con.Open();
                }

                SqlCommand cmd = new SqlCommand("SELECT * from admin_login_tbl where username='"
+ TextBox1.Text.Trim() + "'AND password='" + TextBox2.Text.Trim() + "'", con);

                SqlDataReader dr = cmd.ExecuteReader();
                if (dr.HasRows)
                {
                    while (dr.Read())
                    {
                        //Response.Write("<script>alert('" + dr.GetValue(0).ToString() +
"');</script>");

                        Session["username"] = dr.GetValue(0).ToString();
                        Session["fullname"] = dr.GetValue(2).ToString();
                        Session["role"] = "admin";
                    }
                }
            }
            catch { }
        }
    }
}

```

```

        }
        Response.Redirect("homepage.aspx");
    }
    else
    {
        Response.Write("<script>alert('Invalid Credentials');</script>");
    }
}
catch (Exception ex)
{
    Response.Write("<script>alert('" + ex.Message + "');</script>");
}
}
}
}
}

```

### 5.1.1 Code for USER Login:

#### Front\_end

```

<%@ Page Title="" Language="C#" MasterPageFile="~/Site1.Master" AutoEventWireup="true"
CodeBehind="userlogin.aspx.cs" Inherits="Lms.userlogin" %>
<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat="server">
    <div class="container">
        <div class="row">
            <div class="col-md-6 mx-auto">
                <div class="card">
                    <div class="card-body">
                        <div class="row">
                            <div class="col">
                                <center>
                                    
                                </center>
                            </div>
                        </div>
                        <div class="row">
                            <div class="col">
                                <center>
                                    <h3>User Login</h3>
                                </center>
                            </div>
                        </div>
                        <div class="row">
                            <div class="col">
                                <hr>
                            </div>
                        </div>
                        <div class="row">
                            <div class="col">
                                <div class="form-group">
                                    <label>Registration No</label>
                                    <asp:TextBox CssClass="form-control" ID="TextBox1"
runat="server" placeholder="Registration No"></asp:TextBox>
                                </div>
                                <div class="form-group">

```

```

                                <label>Password</label>
                                <asp:TextBox CssClass="form-control" ID="TextBox2"
runat="server" placeholder="User Password" TextMode="Password"></asp:TextBox>
                                </div>
                                <div class="form-group">
                                    <asp:Button CssClass="btn btn-primary btn-block btn-lg"
ID="Button1" runat="server" Text="Login" OnClick="Button1_Click" />
                                </div>
                                <div class="form-group">
                                    <a href="usersignup.aspx">
                                        <input class="btn btn-info btn-block btn-lg" id="Button2"
type="button" value="Sign Up" />
                                    </a>
                                </div>
                                <a href="forgetpassword.aspx"> Forget Password</a><br>
                            </div>
                        </div>
                    </div>
                </div>
                <a href="homepage.aspx"><<< Back to Home</a><br>
                <br>
            </div>
        </div>
    </div>
</asp:content>

```

## Back End

```

namespace Lms
{
    public partial class userlogin : System.Web.UI.Page
    {
        string strcon = ConfigurationManager.ConnectionStrings["con"].ConnectionString;

        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void Button1_Click(object sender, EventArgs e)
        {
            try
            {
                string hashedPassword = HashPassword(TextBox2.Text.Trim());

                SqlConnection con = new SqlConnection(strcon);
                if (con.State == System.Data.ConnectionState.Closed)
                {
                    con.Open();
                }

                SqlCommand cmd = new SqlCommand("SELECT * from member_master_tbl where
registration_no=@registration_no AND password=@password;", con);
                cmd.Parameters.AddWithValue("@registration_no", TextBox1.Text.Trim());
                cmd.Parameters.AddWithValue("@password", hashedPassword);
                SqlDataReader dr = cmd.ExecuteReader();
                if (dr.HasRows)
                {
                    while (dr.Read())
                    {
                        Response.Write("<script>alert(' " + dr.GetValue(8).ToString() +
"');</script>");

```

```

        Session["username"] = dr.GetValue(8).ToString();
        Session["fullname"] = dr.GetValue(9).ToString();
        Session["role"] = "user";
        Session["status"] = dr.GetValue(10).ToString();

    }
    Response.Redirect("userviewprofile.aspx");
}
else
{
    Response.Write("<script>alert('Invalid Credentials');</script>");
}
}
catch (Exception ex)
{
    Response.Write("<script>alert('" + ex.Message + "');</script>");
}
}
string HashPassword(string password)
{
    using (SHA256 sha256 = SHA256.Create())
    {
        byte[] hashedBytes = sha256.ComputeHash(Encoding.UTF8.GetBytes(password));

        StringBuilder builder = new StringBuilder();
        for (int i = 0; i < hashedBytes.Length; i++)
        {
            builder.Append(hashedBytes[i].ToString("x2"));
        }

        return builder.ToString();
    }
}
}
}
}

```

## Code for View Profile:

### Front\_End

```

<%@ Page Title="" Language="C#" MasterPageFile="~/Site1.Master" AutoEventWireup="true"
CodeBehind="userviewprofile.aspx.cs" Inherits="Lms.userviewprofile" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">

    <script type="text/javascript">

        $(document).ready(function () {

            //$(document).ready(function () {

            //$('#.table').DataTable();

            // });

            $(".table").prepend($(".<thead></thead>")).append($(this).find("tr:first")).dataTable();

```



```

        //$('.table1').DataTable();

    });

</script>
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat="server">
<div class="container-fluid">
    <div class="row">
        <div class="col-md-5">

            <div class="card">

                <div class="card-body">

                    <div class="row">

                        <div class="col">

                            <center>

                            </center>

                        </div>

                    </div>

                </div>

            </div>

        </div>

        <div class="col">

            <center>

                <h4>Your Profile</h4>

                <span>Account Status - </span>

                <asp:Label ID="Label1" runat="server" Text="Your Status"></asp:Label>

            </center>

        </div>
    </div>
</div>

```

```
</div>
```

```
<div class="row">
```

```
<div class="col">
```

```
<hr>
```

```
</div>
```

```
</div>
```

```
<div class="row">
```

```
<div class="col-md-6">
```

```
<div class="form-group">
```

```
<label>Full Name</label>
```

```
<asp:TextBox CssClass="form-control" ID="TextBox1" runat="server" placeholder="Enter Your Full Name"></asp:TextBox>
```

```
</div>
```

```
</div>
```

```
<div class="col-md-6">
```

```
<div class="form-group">
```

```
<label>Date Of Birth</label>
```

```
<asp:TextBox CssClass="form-control" ID="TextBox2" runat="server" placeholder="User Password" TextMode="Date"></asp:TextBox>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
<div class="row">
```

```
<div class="col-md-6">
```

```
<div class="form-group">
```

```
<label>Contact Number</label>
```

```
        <asp:TextBox CssClass="form-control" ID="TextBox3" runat="server" placeholder="Contact No"
TextMode="Number"></asp:TextBox>
```

```
    </div>
```

```
</div>
```

```
<div class="col-md-6">
```

```
    <div class="form-group">
```

```
        <label>Email</label>
```

```
        <asp:TextBox CssClass="form-control" ID="TextBox4" runat="server" placeholder="Email ID"
TextMode="Email"></asp:TextBox>
```

```
    </div>
```

```
</div>
```

```
</div>
```

```
<div class="row">
```

```
    <div class="col-md-4">
```

```
        <div class="form-group">
```

```
            <label>State</label>
```

```
            <asp:DropDownList CssClass="form-control" ID="DropDownList1" runat="server">
```

```
                <asp:ListItem Text="Select" Value="Select" />
```

```
                <asp:ListItem Text="Select" Value="select" />
```

```
                <asp:ListItem Text="Andhra Pradesh" Value="Andhra Pradesh" />
```

```
                <asp:ListItem Text="Arunachal Pradesh" Value="Arunachal Pradesh" />
```

```
                <asp:ListItem Text="Assam" Value="Assam" />
```

```
                <asp:ListItem Text="Bihar" Value="Bihar" />
```

```
                <asp:ListItem Text="Chhattisgarh" Value="Chhattisgarh" />
```

```
                <asp:ListItem Text="Rajasthan" Value="Rajasthan" />
```

```
                <asp:ListItem Text="Goa" Value="Goa" />
```

```
                <asp:ListItem Text="Gujarat" Value="Gujarat" />
```

```

<asp:ListItem Text="Haryana" Value="Haryana" />
<asp:ListItem Text="Himachal Pradesh" Value="Himachal Pradesh" />
<asp:ListItem Text="Jammu and Kashmir" Value="Jammu and Kashmir" />
<asp:ListItem Text="Jharkhand" Value="Jharkhand" />
<asp:ListItem Text="Karnataka" Value="Karnataka" />
<asp:ListItem Text="Kerala" Value="Kerala" />
<asp:ListItem Text="Madhya Pradesh" Value="Madhya Pradesh" />
<asp:ListItem Text="Maharashtra" Value="Maharashtra" />
<asp:ListItem Text="Manipur" Value="Manipur" />
<asp:ListItem Text="Meghalaya" Value="Meghalaya" />
<asp:ListItem Text="Mizoram" Value="Mizoram" />
<asp:ListItem Text="Nagaland" Value="Nagaland" />
<asp:ListItem Text="Odisha" Value="Odisha" />
<asp:ListItem Text="Punjab" Value="Punjab" />
<asp:ListItem Text="Rajasthan" Value="Rajasthan" />
<asp:ListItem Text="Sikkim" Value="Sikkim" />
<asp:ListItem Text="Tamil Nadu" Value="Tamil Nadu" />
<asp:ListItem Text="Telangana" Value="Telangana" />
<asp:ListItem Text="Tripura" Value="Tripura" />
<asp:ListItem Text="Uttar Pradesh" Value="Uttar Pradesh" />
<asp:ListItem Text="Uttarakhand" Value="Uttarakhand" />
<asp:ListItem Text="West Bengal" Value="West Bengal" />
</asp:DropDownList>

```

```

</div>

```

```

</div>

```

```

<div class="col-md-4">

```

```

<div class="form-group">

```

```

<label>City</label>

```

```

<asp:TextBox CssClass="form-control" ID="TextBox6" runat="server"

```

```

placeholder="City"></asp:TextBox>

```

```
</div>
```

```
</div>
```

```
<div class="col-md-4">
```

```
<div class="form-group">
```

```
<label>Pincode</label>
```

```
<asp:TextBox CssClass="form-control" ID="TextBox7" runat="server" placeholder="Pincode"
TextMode="Number"></asp:TextBox>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
<div class="col">
```

```
<div class="form-group">
```

```
<label>Full Address</label>
```

```
<asp:TextBox CssClass="form-control" ID="TextBox5" runat="server" placeholder="Enter your Full
Address" TextMode="MultiLine" Rows="2"></asp:TextBox>
```

```
</div>
```

```
</div>
```

```
<div class="row">
```

```
<div class="col">
```

```
<center>
```

```
<span class="badge baddge-pill badge-dark">login Credentials</span>
```

```
<br>
```

```
<br>
```

```
</center>
```

```
</div>
```

```
</div>
```

```
<div class="row">
```

```
<div class="col-md-4">

    <div class="form-group">

        <label>Registration NO</label>

        <asp:TextBox ReadOnly="true" Class="form-control" ID="TextBox8" runat="server"
placeholder="Registration NO"></asp:TextBox>

    </div>

</div>

<div class="col-md-4">

    <div class="form-group">

        <label>Old Password</label>

        <asp:TextBox ReadOnly="true" Class="form-control" ID="TextBox9" runat="server"
placeholder="Password" TextMode="Password"></asp:TextBox>

    </div>

</div>

<div class="col-md-4">

    <div class="form-group">

        <label>New Password</label>

        <asp:TextBox Class="form-control" ID="TextBox10" runat="server" placeholder=" New Password"
TextMode="Password"></asp:TextBox>

    </div>

</div>

</div>

<div class="row">

    <div class="col-8 mx-auto">

        <center>

            <div class="form-group">

                <asp:Button CssClass="btn btn-primary btn-block btn-info" ID="Button1" runat="server"
Text="Update" OnClick="Button1_Click" />

            </div>

        </center>

    </div>

</div>
```

```

        </div>

    </center>

</div>

</div>

<div class="row">
    <div>
        </div>

    </div>

</div>

<div>
    <a href="homepage.aspx"><<< Back to Home</a><br>
    <br>
</div>

<div class="col-md-7">
    <div class="card">

        <div class="card-body">

            <div class="row">

                <div class="col">
                    <center>
                        
                    </center>
                </div>

            </div>

        </div>

    </div>

</div>

```

```

        <div class="col">

            <center>

                <h4>Your Account Status</h4>

                <asp:Label CssClass="badge badge-pill badge-info" ID="Label2" runat="server" Text="Your
BooksInfo"></asp:Label>

            </center>

        </div>

    </div>

    <div class="row">

        <div class="col">

            <hr>

        </div>

    </div>

    <div class="row">

        <asp:SqlDataSource ID="SqlDataSource1" runat="server" ConnectionString="<%=
ConnectionStrings:elibraryDBConnectionString2 %>" SelectCommand="SELECT * FROM
[book_issue_tbl]"></asp:SqlDataSource>

        <div class="col">

            <<asp:GridView ID="GridView1" class="table table-striped table-bordered" runat="server"
AutoGenerateColumns="False" DataKeyNames="book_id" OnRowDataBound="GridView1_RowDataBound">

                <Columns>

                    <asp:BoundField DataField="registration_no" HeaderText="Reg No"
SortExpression="registration_no"></asp:BoundField>

                    <asp:BoundField DataField="book_id" HeaderText="Book ID" ReadOnly="True"
SortExpression="book_id"></asp:BoundField>

                    <asp:BoundField DataField="member_name" HeaderText="User Name"
SortExpression="member_name"></asp:BoundField>

                    <asp:BoundField DataField="book_name" HeaderText="Book Name"
SortExpression="book_name"></asp:BoundField>

```





```

    }

    //update button click
    protected void Button1_Click(object sender, EventArgs e)
    {
        if (Session["username"].ToString() == null || Session["username"] == null)
        {
            Response.Write("<script>alert('Session Expired Login Again');</script>");
            Response.Redirect("userlogin.aspx");
        }
        else
        {
            UpdateUserPersonalDetail();
        }
    }

    //user defined functions

    void UpdateUserPersonalDetail()
    {
        string newPassword = TextBox10.Text.Trim() == null ? TextBox9.Text.Trim() :
        TextBox10.Text.Trim();
        string newPasswordHash = HashPassword(newPassword);
        try
        {
            SqlConnection con = new SqlConnection(strcon);
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }

            SqlCommand cmd = new SqlCommand("update member_master_tbl set
            full_name=@full_name, dob=@dob, contact_no=@contact_no, email=@email, state=@state, city=@city,
            pincode=@pincode, full_address=@full_address, password=@password, account_status=@account_status
            WHERE registration_no='" + Session["username"].ToString().Trim() + "'", con);

            cmd.Parameters.AddWithValue("@full_name", TextBox1.Text.Trim());
            cmd.Parameters.AddWithValue("@dob", TextBox2.Text.Trim());
            cmd.Parameters.AddWithValue("@contact_no", TextBox3.Text.Trim());
            cmd.Parameters.AddWithValue("@email", TextBox4.Text.Trim());
            cmd.Parameters.AddWithValue("@state", DropDownList1.SelectedItem.Value);
            cmd.Parameters.AddWithValue("@city", TextBox6.Text.Trim());
            cmd.Parameters.AddWithValue("@pincode", TextBox7.Text.Trim());
            cmd.Parameters.AddWithValue("@full_address", TextBox5.Text.Trim());
            cmd.Parameters.AddWithValue("@password", newPasswordHash);
            cmd.Parameters.AddWithValue("@account_status", "pending");

            int result = cmd.ExecuteNonQuery();
            con.Close();
            if (result > 0)
            {
                Response.Write("<script>alert('Your Details Updated
                Successfully');</script>");
                GetUserPersonalDetails();
                getbookdata();
            }
            else
            {
                Response.Write("<script>alert('Invalid entry');</script>");
            }
        }
    }
}

```

```

        catch (Exception ex)
        {
            Response.Write("<script>alert('" + ex.Message + "');</script>");
        }
    }

    void GetUserPersonalDetails()
    {
        try
        {
            SqlConnection con = new SqlConnection(strcon);
            if (con.State == ConnectionState.Closed)
            {
                con.Open();
            }

            SqlCommand cmd = new SqlCommand("select * from member_master_tbl where registration_no='" + Session["username"].ToString() + "';", con);
            SqlDataAdapter da = new SqlDataAdapter(cmd);
            DataTable dt = new DataTable();
            da.Fill(dt);
            TextBox1.Text = dt.Rows[0]["full_name"].ToString();
            TextBox2.Text = dt.Rows[0]["dob"].ToString();
            TextBox3.Text = dt.Rows[0]["contact_no"].ToString();
            TextBox4.Text = dt.Rows[0]["email"].ToString();
            DropDownList1.Text = dt.Rows[0]["state"].ToString();
            TextBox6.Text = dt.Rows[0]["city"].ToString();
            TextBox7.Text = dt.Rows[0]["pincode"].ToString();
            TextBox5.Text = dt.Rows[0]["full_address"].ToString();
            TextBox8.Text = dt.Rows[0]["registration_no"].ToString();
            TextBox9.Text = dt.Rows[0]["password"].ToString();

            Label1.Text = dt.Rows[0]["account_status"].ToString().Trim();

            if (dt.Rows[0]["account_status"].ToString().Trim() == "active")
            {
                Label1.Attributes.Add("class", "badge badge-pill badge-success");
            }
            else if (dt.Rows[0]["account_status"].ToString().Trim() == "pending")
            {
                Label1.Attributes.Add("class", "badge badge-pill badge-warning");
            }
            else if (dt.Rows[0]["account_status"].ToString().Trim() == "deactive")
            {
                Label1.Attributes.Add("class", "badge badge-pill badge-danger");
            }
            else
            {
                Label1.Attributes.Add("class", "badge badge-pill badge-info");
            }

        }
        catch (Exception ex)
        {
            Response.Write("<script>alert('" + ex.Message + "');</script>");
        }
    }

    void getbookdata()
    {
        try
        {
            SqlConnection con = new SqlConnection(strcon);

```

```

        if (con.State == ConnectionState.Closed)
        {
            con.Open();
        }

        SqlCommand cmd = new SqlCommand("select * from book_issue_tbl where
registration_no='" + Session["username"].ToString() + "';", con);
        SqlDataAdapter da = new SqlDataAdapter(cmd);
        DataTable dt = new DataTable();
        da.Fill(dt);
        GridView1.DataSource = dt;
        GridView1.DataBind();

    }
    catch (Exception ex)
    {
        Response.Write("<script>alert('" + ex.Message + "');</script>");
    }
}
protected void GridView1_RowDataBound(object sender, GridViewRowEventArgs e)
{
    try
    {
        if (e.Row.RowType == DataControlRowType.DataRow)
        {
            DateTime dt = Convert.ToDateTime(e.Row.Cells[5].Text);
            DateTime today = DateTime.Today;
            if (today > dt)
            {
                e.Row.BackColor = System.Drawing.Color.PaleVioletRed;
            }
        }
    }
    catch (Exception ex)
    {
        Response.Write("<script>alert('" + ex.Message + "');</script>");
    }
}
string HashPassword(string password)
{
    using (SHA256 sha256 = SHA256.Create())
    {
        byte[] hashedBytes = sha256.ComputeHash(Encoding.UTF8.GetBytes(password));

        StringBuilder builder = new StringBuilder();
        for (int i = 0; i < hashedBytes.Length; i++)
        {
            builder.Append(hashedBytes[i].ToString("x2"));
        }

        return builder.ToString();
    }
}
}
}

```

## Code For BOOK Issue:

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site1.Master" AutoEventWireup="true"
CodeBehind="bookissuing.aspx.cs" Inherits="Lms.bookissuing" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
    <script type="text/javascript">
        $(document).ready(function () {

            //$(document).ready(function () {
            //$( '.table' ).DataTable();
            // });

            $( ".table" ).prepend( $( "<thead></thead>" ).append( $( this ).find( "tr:first" ) ) ).dataTable();
            //$( '.table1' ).DataTable();
        });
    </script>
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat="server">
    <div class="container-fluid">
        <div class="row">
            <div class="col-md-5">

                <div class="card">

                    <div class="card-body">
                        <div class="row">

                            <div class="col">
                                <center>
                                    <h4>Book Issuing</h4>
                                </center>
                            </div>
                        </div>
                    </div>
                    <div class="row">

                        <div class="col">
                            <center>
                                
                            </center>
                        </div>
                    </div>
                </div>

                <div class="row">

                    <div class="col">
                        <hr>
                    </div>
                </div>
                <div class="row">

                    <div class="col-md-6">
                        <div class="form-group">
                            <label>Registration No</label>
                            <asp:TextBox CssClass="form-control" ID="TextBox2"
runat="server" placeholder="Reg. No"></asp:TextBox>

                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
</asp:Content>
```

```

        <div class="col-md-6">
            <label>Book ID</label>
            <div class="form-group">
                <div class="input-group">
                    <asp:TextBox CssClass="form-control" ID="TextBox1"
runat="server" placeholder="Book ID"></asp:TextBox>
                    <asp:Button Class="btn btn-dark" ID="Button1"
runat="server" Text="Go" OnClick="Button1_Click" />
                </div>
            </div>
        </div>
    </div>
    <div class="row">
        <div class="col-md-6">
            <div class="form-group">
                <label>User Name</label>
                <asp:TextBox ReadOnly="true" CssClass="form-control"
ID="TextBox3" runat="server" placeholder="User Name"></asp:TextBox>
            </div>
        </div>
        <div class="col-md-6">
            <div class="form-group">
                <label>Book Name</label>
                <asp:TextBox ReadOnly="true" CssClass="form-control"
ID="TextBox4" runat="server" placeholder="Book Name"></asp:TextBox>
            </div>
        </div>
    </div>
    <div class="row">
        <div class="col-md-6">
            <div class="form-group">
                <label>Start Date</label>
                <asp:TextBox CssClass="form-control" ID="TextBox5"
runat="server" placeholder="Start Date" TextMode="Date"></asp:TextBox>
            </div>
        </div>
        <div class="col-md-6">
            <div class="form-group">
                <label>End Date</label>
                <asp:TextBox CssClass="form-control" ID="TextBox6"
runat="server" placeholder="End Date" TextMode="Date"></asp:TextBox>
            </div>
        </div>
    </div>
    <div class="row">
        <div class="col-6">
            <asp:Button class="btn btn-primary btn-lg btn-block" ID="Button2"
runat="server" Text="Issue" OnClick="Button2_Click" />
        </div>
    </div>

```

```

        </div>
        <div class="col-6">
            <asp:Button class="btn btn-success bnt-lg btn-block" ID="Button3"
runat="server" Text="Return" OnClick="Button3_Click" />
        </div>
    </div>
    <br>
    <div class="row">
        <div class="col-6">
            <asp:Button class="btn btn-danger bnt-lg btn-block" ID="Button5"
runat="server" Text="CheckFine" OnClick="Button5_Click" />
        </div>
        <div class="col-6">
            <asp:Button class="btn btn-block bnt-lg btn-warning" ID="Button4"
runat="server" Text="ReIssue" OnClick="Button4_Click" />
        </div>
    </div>

    <div class="row">
    </div>
</div>

<a href="homepage.aspx"><< Back to Home</a><br>
<br>
</div>

<div class="col-md-7">
    <div class="card">

        <div class="card-body">
            <div class="row">

                <div class="col">
                    <center>
                        <h4>Issued Book List</h4>
                    </center>

                </div>
            </div>
            <div class="row">

                <div class="col">
                    <hr>
                </div>
            </div>

            <div class="row">
                <asp:SqlDataSource ID="SqlDataSource1" runat="server"
ConnectionString="<?%$ ConnectionStrings:elibraryDBConnectionString2 %>" SelectCommand="SELECT *
FROM [book_issue_tbl]"></asp:SqlDataSource>

                <div class="col">
                    <asp:GridView class="table table-striped table-bordered"
ID="GridView1" runat="server" DataSourceID="SqlDataSource1" AutoGenerateColumns="False"
DataKeyNames="book_id" OnRowDataBound="GridView1_RowDataBound">
                        <Columns>
                            <asp:BoundField DataField="registration_no"
HeaderText="Reg No" SortExpression="registration_no"></asp:BoundField>
                            <asp:BoundField DataField="book_id" HeaderText="Book ID"
ReadOnly="True" SortExpression="book_id"></asp:BoundField>

```





```

        {
            Response.Write("<script>alert('This Memeber already has this
book');</script>");
        }
        else
        {
            DateTime issueDate = DateTime.Today;
            DateTime dueDate = issueDate.AddDays(15);
            Insert(issueDate, dueDate);
        }
    }
    else
    {
        Response.Write("<script>alert('Wrong Book ID OR Registration
Number');</script>");
    }
}
//return book
protected void Button3_Click(object sender, EventArgs e)
{
    if (checkbook() && checkmember())
    {
        if (issueentry())
        {
            returnbook();
        }
        else
        {
            Response.Write("<script>alert('This entry does not exist');</script>");
        }
    }
    else
    {
        Response.Write("<script>alert('Wrong Book ID OR Registration
Number');</script>");
    }
}

//user defined function
void returnbook()
{
    try
    {
        SqlConnection con = new SqlConnection(strcon);
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
        }
    }
}

```

```

        SqlCommand cmd = new SqlCommand("DELETE from book_issue_tbl where book_id='"
+ TextBox1.Text.Trim() + "' AND registration_no='" + TextBox2.Text.Trim() + "'", con);
        int result = cmd.ExecuteNonQuery();

        if (result > 0)
        {
            cmd = new SqlCommand("UPDATE book_master_tbl set
current_stock=current_stock+1 where book_id='" + TextBox1.Text.Trim() + "'", con);
            cmd.ExecuteNonQuery();
            con.Close();
            Response.Write("<script>alert('Book Returned  Successfully');</script>");
            GridView1.DataBind();
            con.Close();

        }
        else
        {
            Response.Write("<script>alert('Error _ Invalid Detail');</script>");
        }
    }
    catch (Exception ex)
    {
        Response.Write("<script>alert('" + ex.Message + "');</script>");
    }
}

void Insert(DateTime issueDate, DateTime dueDate)
{
    try
    {
        SqlConnection con = new SqlConnection(strcon);
        if (con.State == System.Data.ConnectionState.Closed)
        {
            con.Open();
        }
        if (IsUserActive(con))
        {
            SqlCommand cmd = new SqlCommand("SELECT COUNT(*) FROM book_issue_tbl
WHERE registration_no=@registration_no", con);
            cmd.Parameters.AddWithValue("@registration_no", TextBox2.Text.Trim());
            int issuedBooksCount = (int)cmd.ExecuteScalar();

            if (issuedBooksCount >= 3)
            {
                Response.Write("<script>alert('You have already issued the maximum
number of books (3).');</script>");
            }
            else

```

```

        {
            cmd = new SqlCommand("INSERT INTO
book_issue_tbl(registration_no,book_id,member_name,book_name,issue_date,due_date)
values(@registration_no,@book_id,@member_name,@book_name,@issue_date,@due_date)", con);
            cmd.Parameters.AddWithValue("@registration_no",
TextBox2.Text.Trim());
            cmd.Parameters.AddWithValue("@book_id", TextBox1.Text.Trim());
            cmd.Parameters.AddWithValue("@member_name", TextBox3.Text.Trim());
            cmd.Parameters.AddWithValue("@book_name", TextBox4.Text.Trim());
            cmd.Parameters.AddWithValue("@issue_date", issueDate.ToString("yyyy-
MM-dd"));
            cmd.Parameters.AddWithValue("@due_date", dueDate.ToString("yyyy-MM-
dd"));

            cmd.ExecuteNonQuery();
            cmd = new SqlCommand("UPDATE book_master_tbl set
current_stock=current_stock-1 where book_id='" + TextBox1.Text.Trim() + "'", con);
            cmd.ExecuteNonQuery();
            con.Close();
            Response.Write("<script>alert('Book
Issued Successfully');</script>");
            GridView1.DataBind();
        }
    }
    else
    {
        Response.Write("<script>alert('Your account is not active. You cannot
issue books.');

```

```

        da.Fill(dt);
        if (dt.Rows.Count >= 1)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
    catch (Exception ex)
    {
        Response.Write("<script>alert('" + ex.Message + "');</script>");
        return false;
    }
}

bool checkmember()
{
    try
    {
        SqlConnection con = new SqlConnection(strcon);
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
        }
        SqlCommand cmd = new SqlCommand("select full_name from member_master_tbl
where registration_no='" + TextBox2.Text.Trim() + "'", con);
        SqlDataAdapter da = new SqlDataAdapter(cmd);
        DataTable dt = new DataTable();
        da.Fill(dt);
        if (dt.Rows.Count >= 1)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
    catch (Exception ex)
    {
        Response.Write("<script>alert('" + ex.Message + "');</script>");
        return false;
    }
}

bool issueentry()
{
    try

```

```

    {
        SqlConnection con = new SqlConnection(strcon);
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
        }
        SqlCommand cmd = new SqlCommand("select * from book_issue_tbl where
registration_no='" + TextBox2.Text.Trim() + "' AND book_id='" + TextBox1.Text.Trim() + "'",
con);

        SqlDataAdapter da = new SqlDataAdapter(cmd);
        DataTable dt = new DataTable();
        da.Fill(dt);
        if (dt.Rows.Count >= 1)
        {
            return true;
        }
        else
        {
            return false;
        }
    }
    catch (Exception ex)
    {
        Response.Write("<script>alert('" + ex.Message + "');</script>");
        return false;
    }
}

void getnames()
{
    try
    {
        SqlConnection con = new SqlConnection(strcon);
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
        }
        SqlCommand cmd = new SqlCommand("select book_name from book_master_tbl where
book_id='" + TextBox1.Text.Trim() + "'", con);
        SqlDataAdapter da = new SqlDataAdapter(cmd);
        DataTable dt = new DataTable();
        da.Fill(dt);
        if (dt.Rows.Count >= 1)
        {
            TextBox4.Text = dt.Rows[0]["book_name"].ToString();
        }
        else
    }

```

```

        {
            Response.Write("<script>alert('Wrong Book ID');</script>");
        }

        cmd = new SqlCommand("select full_name from member_master_tbl where
registration_no='" + TextBox2.Text.Trim() + "'", con);
        da = new SqlDataAdapter(cmd);
        dt = new DataTable();
        da.Fill(dt);
        if (dt.Rows.Count >= 1)
        {
            TextBox3.Text = dt.Rows[0]["full_name"].ToString();
        }
        else
        {
            Response.Write("<script>alert('Invalid Registration Number');</script>");
        }
    }
    catch (Exception ex)
    {
        Response.Write("<script>alert('" + ex.Message + "');</script>");
    }
}

bool IsUserActive(SqlConnection con)
{
    try
    {
        SqlCommand cmd = new SqlCommand("SELECT account_status FROM member_master_tbl
WHERE registration_no=@registration_no", con);
        cmd.Parameters.AddWithValue("@registration_no", TextBox2.Text.Trim());
        string accountStatus = cmd.ExecuteScalar().ToString();

        return accountStatus == "active";
    }
    catch (Exception ex)
    {
        Response.Write("<script>alert('" + ex.Message + "');</script>");
        return false;
    }
}

protected void GridView1_RowDataBound(object sender, GridViewRowEventArgs e)
{
    try
    {
        if (e.Row.RowType == DataControlRowType.DataRow)
        {
            DateTime dt = Convert.ToDateTime(e.Row.Cells[5].Text);
            DateTime today = DateTime.Today;
            if (today > dt)
            {

```

```

        e.Row.BackColor = System.Drawing.Color.PaleVioletRed;
    }
}
}
catch (Exception ex)
{
    Response.Write("<script>alert('" + ex.Message + "');</script>");
}
}
private int CalculateTotalFine(string registrationNo, string bookId)
{
    int totalFine = 0;

    try
    {
        SqlConnection con = new SqlConnection(strcon);
        if (con.State == ConnectionState.Closed)
        {
            con.Open();
        }

        SqlCommand cmd = new SqlCommand("SELECT due_date FROM book_issue_tbl WHERE
registration_no = @registration_no AND book_id = @book_id", con);
        cmd.Parameters.AddWithValue("@registration_no", registrationNo);
        cmd.Parameters.AddWithValue("@book_id", bookId);

        SqlDataReader dr = cmd.ExecuteReader();

        while (dr.Read())
        {
            DateTime dueDate = Convert.ToDateTime(dr["due_date"]);
            int fineRate = 10;

            // Calculate fine for each book and add it to the total fine
            totalFine += CalculateFine(dueDate, DateTime.Now, fineRate);
        }

        dr.Close();
        con.Close();
    }
    catch (Exception ex)
    {
        Response.Write("<script>alert('" + ex.Message + "');</script>");
    }

    return totalFine;
}
// Add this method to calculate fine based on due date and current date
int CalculateFine(DateTime dueDate, DateTime currentDate, int fineRate)
{

```

```

        int fine = 0;

        if (currentDate > dueDate)
        {
            TimeSpan overdueDays = currentDate - dueDate;
            fine = fineRate * (int)overdueDays.TotalDays;
        }

        return fine;
    }
    protected void Button5_Click(object sender, EventArgs e)
    {
        string bookId = TextBox1.Text.Trim();
        string registrationNo = TextBox2.Text.Trim();

        // Call the method to calculate and display the total fine
        int totalFine = CalculateTotalFine(registrationNo, bookId);

        if (totalFine > 0)
        {
            // Display the total fine
            Response.Write("<script>alert('The Fine For This BOOK is: Rs" +
totalFine.ToString() + "');</script>");
        }
        else
        {
            // If there is no fine, you can display a message
            Response.Write("<script>alert('No fine found for this user and
book.');

```



```

cmd.Parameters.AddWithValue("@book_id", TextBox1.Text.Trim());
cmd.Parameters.AddWithValue("@registration_no", TextBox2.Text.Trim());

int result = cmd.ExecuteNonQuery();

if (result > 0)
{
    Response.Write("<script>alert('Book Reissued Successfully');</script>");
    GridView1.DataBind();
}
else
{
    Response.Write("<script>alert('Error in reissuing the book');</script>");
}

con.Close();
}
catch (Exception ex)
{
    Response.Write("<script>alert('" + ex.Message + "');</script>");
}
} //Reissue
protected void Button4_Click(object sender, EventArgs e)
{

    if (checkbook() && checkmember())
    {
        if (issueentry())
        {
            DateTime currentDate = DateTime.Today;
            DateTime dueDate = currentDate.AddDays(15);

            Reissue();
        }
        else
        {
            Response.Write("<script>alert('This entry does not exist');</script>");
        }
    }
    else
    {
        Response.Write("<script>alert('Wrong Book ID OR Registration
Number');</script>");
    }
}
}
}

```

# *Chapter 6 Testing*

*“Executing a Program with the intent of finding errors”*

## 6. Testing

Software testing is a critical element of the ultimate review of specification design and coding. Testing of software leads to the uncovering of errors in the software functional and performance requirements are met. Testing also provides a good indication of software reliability and software quality as a whole. The result of different phases of testing are evaluated and then compared with the expected results. If the errors are uncovered they are debugged and corrected. A strategy approach to software testing has the generic characteristics:

- Testing begins at the module level and works “outwards” towards the integration of the entire computer based system.
- Different testing techniques are appropriate at different points of time.
- Testing and debugging are different activities, but debugging must be accommodated in the testing strategy.

### **6.1 Goals & Objectives:**

Testing is a process of executing a program with the intent of finding an error. A good test case is one that has a probability of finding an as yet undiscovered error. A successful test is one that uncovers an as yet undiscovered error. Our Objective is to design test processes that systematically uncover different classes of errors and do so with minimum amount of time and effort.

### **6.1 Statement of Scope:**

A description of the scope of the software testing is developed. All the features to be tested are noted as follows. The basic principles that guides software testing are,

- All test cases should be traceable top customer requirements. The most severe defects from the customer’s point of view are those that cause the program to fail to meet its requirements.
- Test case should be planned long before testing begins. Testing plan can begin as soon as the requirement model is complete. Detailed definition of the test cases can begin as soon as the design is solidified. Therefore, the entire test can be planned before any code has been generated.

Testing should begin “in the small” and progress towards “in the large”. The first test planned and executed generally focus on the individual modules. As testing progresses testing shifts focus in an attempt to find errors in integrating clusters of modules and ultimately in the entire system.

## 6.1. Testing Principles:

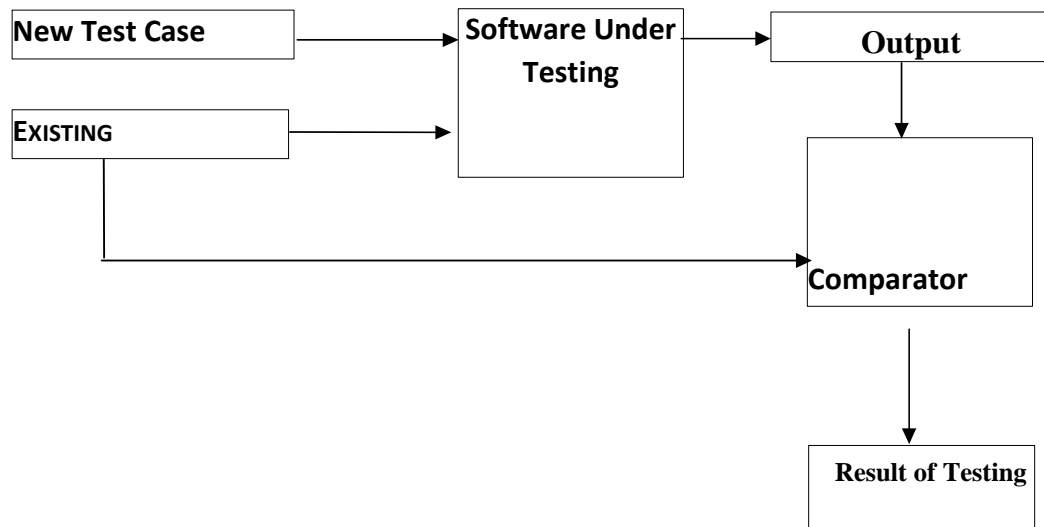
The basic principles that guide software testing are: -

- □ All test cases should be traceable to customer requirements. The most severe defects from the customer's point of view are those that cause the program to fail to meet its requirements.
- □ Test case should be planned long before testing begins. Testing plan can begin as soon as the requirement model is complete. Detailed definition of the test cases can begin as soon as the design is solidified. Therefore all the test can be planned before any code has been generated.
- □ The Pareto principle applies to software testing. Stated simply the Pareto principle implies that 80% of all errors uncovered during testing will likely be traceable to 20% of all program modules. The program of course is to isolate these suspect modules and to thoroughly test them.
- □ Testing should begin "in the small" and progress towards "in the large". The first test planned and executed generally focus on the individual modules. As testing progresses testing shifts focus in an attempt to find errors in integrating clusters of modules and ultimately in the entire system.
- □ Exhaustive testing is not possible. The number of paths permutations for even a moderately sized program is exceptionally large. For this reason it is impossible to execute every combination of path during testing. It is possible however to ensure that all conditions in the procedural design have been exercised
- To be most effective an independent third party should conduct testing. The third party has the highest probability of finding the errors.

## 6.1 Test Case:

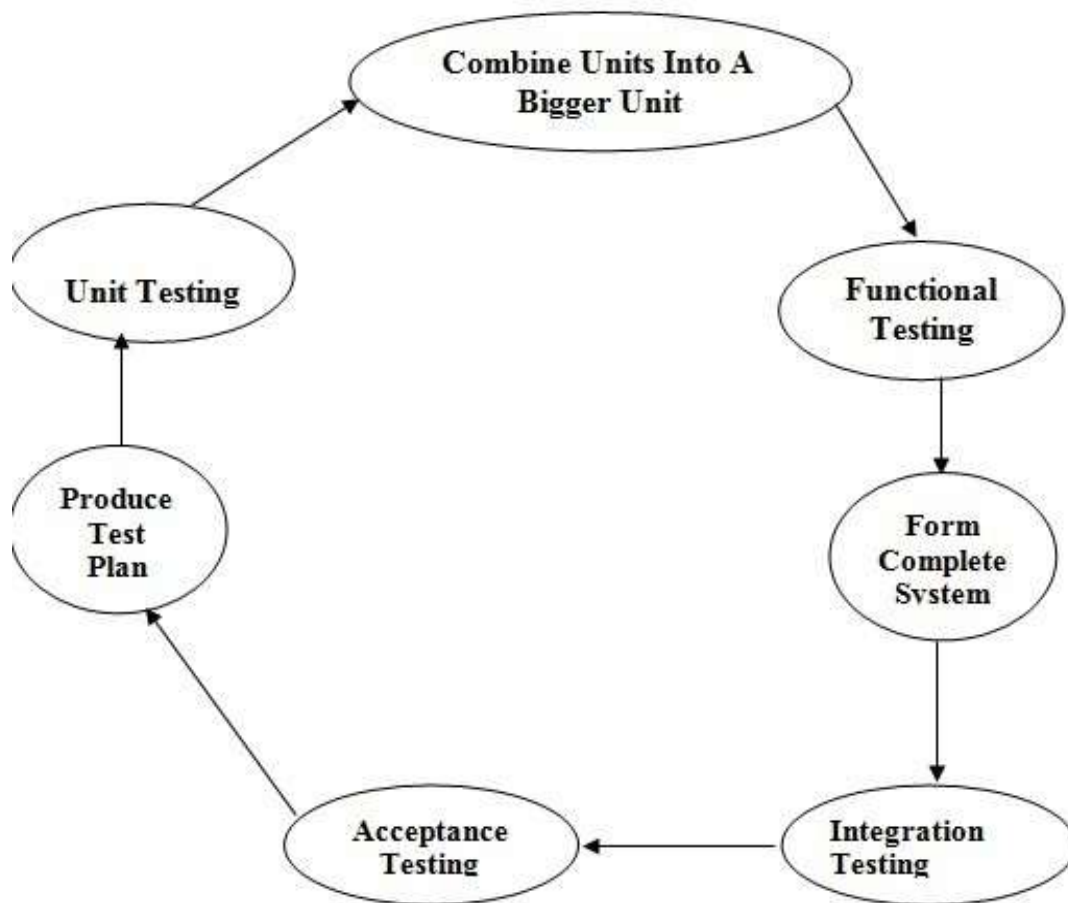
Before the project is released, it has to pass through a test cases suit, so that the required functionality is met and previous functionality of the system is also not broken to do this, there is an existing test cases which checks for the previous functionality. New test cases are prepared and added to this existing test suit to check for the added functionality.

A pictorial representation of this can be shown as follows



## 6.1 Testing Process:

The Testing Process can be shown as:



## 6.1 Testing Approaches Used in Project:

The module interface is tested to ensure that information properly flows into and out of the program unit under test. The unit testing is normally considered as an adjunct step to coding step. Because modules are not a standalone program, drivers and/or stubs software must be developed for each unit. A driver is nothing more than a “main program” that accepts test cases data and passes it to the module. A stub serves to replace the modules that are subordinate to the modules to be tested. A stub may do minimal data manipulation, prints verification of entry and returns.

### 6.0.1 Unit Testing:

**6.0.1.1 Functional Test:** Each part of the code was tested individually and the panels were tested individually on all platforms to see if they are working properly.

**6.0.1.1 Performance Test:** These determined the amount of execution time spent on various parts of units and the resulting throughput, response time given by the module.

**6.0.1.1 Stress Test:** A lot of test files were made to work at the same time in order to check how much workloads can the unit bear.

**6.0.1.1 Structure Test:** These tests were made to check the internal logic of the program and traversing particular execution paths.

**6.0.1.1 Integration Test:** “If they all work individually, they should work when we put them together.” The problem of course is “putting them together”. This can be done in two ways:

**6.0.0.1 Top down Integration:** Modules are integrated by moving downwards through the control hierarchy, beginning with main control module are incorporated into the structure in either a depth first or breadth first manner.

**6.0.0.1 Bottom up Integration:** It begins with construction and testing with atomic modules i.e. modules at the lowest level of the program structure. Because modules are integrated from the bottom up, processing required for the modules subordinate to a given level is always available and the need of stubs is eliminated.

#### **6.0.1 Validation Test:**

Validation succeeds when software functions in a manner that can be reasonably expected by the customer. It covers the following :-

**6.0.0.1 Validation Test Criteria::** Performance, functional characteristics and uncovered deviation from specification

**6.0.0.1 Configuration Review:** Ensures that all the elements of software configuration have been properly developed cataloged and have support for the maintenance phase of software life cycle

**6.0.1 Alpha-Beta Testing:** Alpha test is conducted by developer’s site by customer. Beta test is conducted at one or more customer site by software end user.

**6.0.1 Modular Integration Testing:** Modular integration testing is done to ensure that the module is working independently. The inputs as required by the module are given as required and the output is tested as per the specifications

Testing Phase	Functionality to be Tested	Scheduled Date	Actual Date
<b>Unit Testing:</b>			
Components	Check whether each method of a component results the expected values	10-Nov-2023	10- Nov -2023
Windows Form	Check whether all validations are performed properly and whether appropriate	15- Nov -2023	15- Nov -2023
Login, Log	Values are added to the database.	18- Nov -2023	18- Nov -2023
Data Insertion in modules	Check whether each module is taking the right data	20- Nov -2023	20- Nov -2023
Web Forms	Check whether the data is retrieved from database properly or not and displayed correctly on the web form	25- Nov -2023	25- Nov -2023
<b>Integration Testing:</b>			
	Check whether all independent modules work properly after integration	28- Nov -2023	28- Nov -2023

## 5.1 Validations to be Performed:

### Form: Login

1. The user name and password should exist in the *user* table.
2. Form: Generate New Password.
3. Checks whether any user has sent a forgot password request.
4. Generates the new Password accordingly.

### Form: Changing Password.

1. Checks whether the old password is correct.
2. Checks minimum length of the password is 8 characters.



3. Checks whether new password and confirm password are same.

Test cases form name	Testing name	description	results	remarks
Login form	Unit testing	Username= <a href="#">abc@gmail.co m</a> Password= 123pass@wor d	Login failed	Wrong username/pass word
New user	Unit testing	Enter password and confirm password different	Invalid email/passw ord	Not Allowed
verification	Unit testing	<a href="#">Username=xyz @gmail.com</a> Password= pass111word	Login successfully	Viewed Successfully

# ***Chapter 7 Implementation***

***“Tools used and tools required for the working of System”***

## 6.1 Implementation Requirements:

The implementation requirements for the product are listed below:

Hardware Requirements	
S. No	Description
Server Requirements	
1	A server with 1.0 GHZ processor, 20GB Hard disk, 512 MB RAM with networking capabilities having 256 kbps LAN connection.
Client Requirements	
1	Any medium range PC with networking capabilities having a 256 kbps LAN connection.

Software Requirements		
S. No	Software	Description
Server Requirements		
1	Operating System	Windows 2000/XP/Server (preferred for performance).
2	Application Server	IIS 5.0 or above
3	DBMS and Reporting Server	MS SQL
Client Requirements		
1	Operating System	Windows 2000/XP/Server (preferred for performance), Linux.
2	Web Browser	Any web browser capable of running java script, VB scripts, flash etc.

## 6.2 Tools and Software Used:

S. No	Tool	Purpose
1	VISUAL STUDIO	IDE used for the Development of the Product. It support for development operations like debugging,task running, and version control
2	MS SQL	For storing the information.
3	Microsoft Office 2007	For creating the documentation and files.
4	Microsoft PowerPoint	For Creating the Presentations

## 6.3 Why MS SQL Server

Selecting MS SQL Server for a .NET C# web forms project ensures seamless integration and compatibility, streamlining development. MS SQL's scalability, security features, and robust management tools contribute to system efficiency and reliability, supported by a large and active developer community.

### 6.3.1 Hardware Requirements of MS SQL: -

#### Processor (CPU):

Minimum: 1.4 GHz 64-bit processor

Recommended: 2.0 GHz or faster multi-core processor

#### RAM (Memory):

Minimum: 1 GB (2 GB for Express Edition)

Recommended: At least 4 GB or more

#### Hard Disk Space:

Minimum: 6 GB of available hard-disk space

Recommended: 10 GB or more for a typical installation

#### Operating System:

Windows Server (various editions) or Windows operating system

#### Additional Requirements:

DVD-ROM drive (for installation from DVD)

Super VGA (800x600) or higher resolution monitor

Internet connection for software downloads and updates

It's crucial to note that these are general requirements, and the actual hardware needs may vary based on factors such as the database size, the number of concurrent users, and the workload. Before installation, it's recommended to refer to the specific documentation for the version of MS SQL Server you are using for detailed and up-to-date hardware requirements.

# *Chapter 8 Snapshots*

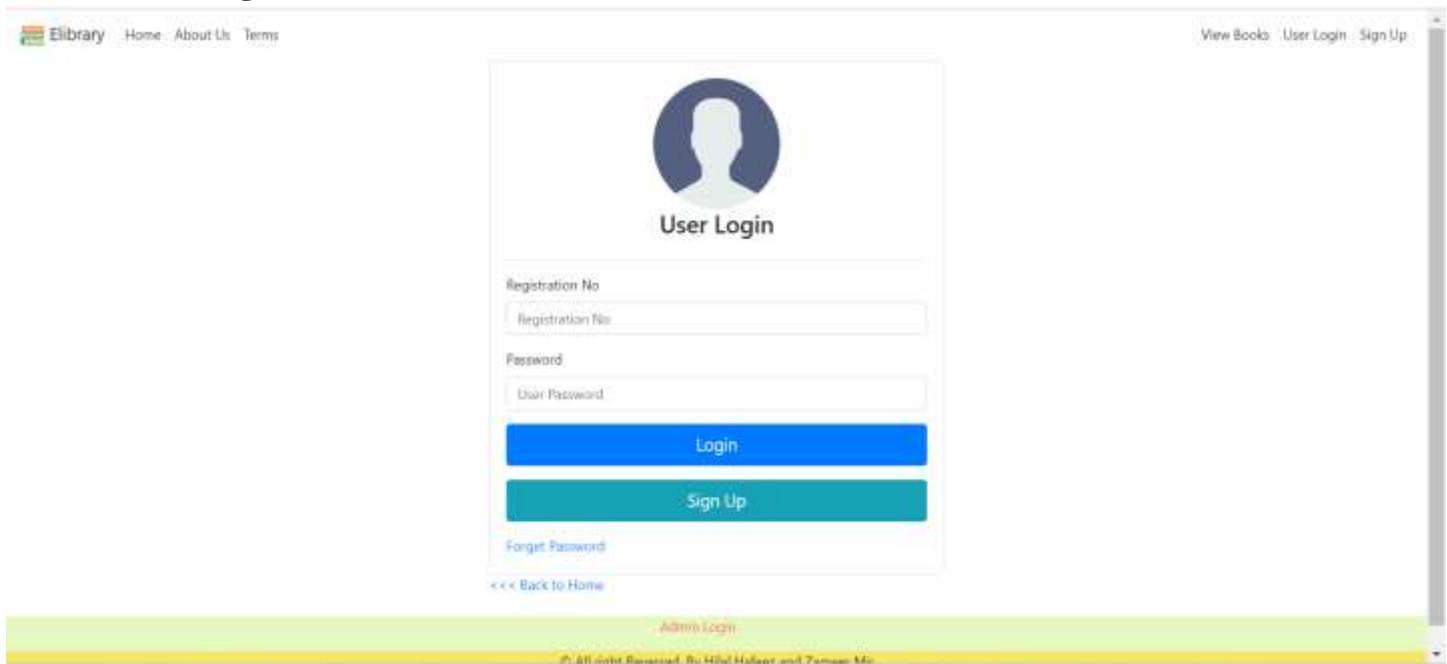
*“Have a look at the working System”*

## 8 Snapshots

### 8.1 Home Screen :




### 1.2 User Login Module



## 1.3 View Books Module:

Book Inventory List	
Show 10 entries	Search
ID	
1123	<b>Python WorkShop</b> Author - <b>Ahmad</b>   Genre - <b>Machine Learning</b>   Language - <b>English</b> Publisher - <b>Tufail</b>   Publish Date - <b>2023-10-05</b>   Pages - <b>1123</b>   Edition - <b>12</b> Cost - <b>123</b>   Actual Stock - <b>8</b>   Available Stock - <b>8</b> Description - <b>Avila</b>
12	<b>Think And Grow Rich</b> Author - <b>Ahmad</b>   Genre - <b>Machine Learning</b>   Language - <b>Urdu</b> Publisher - <b>Tufail</b>   Publish Date - <b>2023-10-03</b>   Pages - <b>12</b>   Edition - <b>12</b> Cost - <b>01</b>   Actual Stock - <b>7</b>   Available Stock - <b>6</b> Description - <b>think and grow rich</b>
22348	<b>Past Forward</b> Author - <b>Hilal</b>   Genre - <b>Machine Learning</b>   Language - <b>English</b> Publisher - <b>Peer</b>   Publish Date - <b>2023-11-16</b>   Pages - <b>234</b>   Edition - <b>13</b> Cost - <b>123</b>   Actual Stock - <b>8</b>   Available Stock - <b>6</b> Description - <i>Did you know that your mind has a 'mind' of its own? Yes! Without even realize our mind is often governed by another entity which is called sub-conscious mind(Zameer)</i>

## 1.4 SignUp Module:



User Registration

Full Name

Date Of Birth

Contact Number

Email

State

Select

City

Pincode

Full Address


Registration No

Password

Sign Up




## 8.5 Book Issue Module:


[Home](#)
[About Us](#)
[Terms](#)

[View Books](#)
[User Login](#)
[Sign Up](#)

### Book Issuing



Registration No

Book ID

User Name

Book Name

Start Date

End Date

[<<< Back to Home](#)

### Issued Book List

Show  entries


Search

Reg No	Book ID	Member Name	Book Name	Issue Date	Due Date
1122	2234E		Past Forward	2023-11-16	2023-12-01
	12		Thank And Grow Rich	2023-11-15	2023-11-30
	2234E		Past Forward	2023-11-16	2023-12-01

Showing 1 to 3 of 3 entries

[Admin Login](#)

## 8.6 User View Profile Module



### Your Profile

Account Status - Your Status

Full Name

Date Of Birth

Contact Number

Email

State

City

Pincode

Full Address

Registration NO

Old Password

New Password

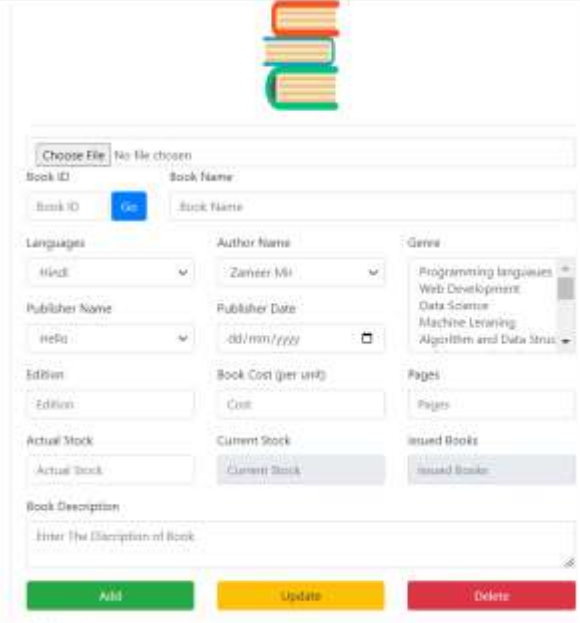
[<<< Back to Home](#)



### Your Account Status

[New Bookable](#)

## 8.6 Book Inventory Module:



**Book Inventory Form**

Choose File: No file chosen

Book ID:  Book Name:

Book ID:  Book Name:

Languages:  Author Name:  Genre:

Publisher Name:  Publisher Date:

Edition:  Book Cost (per unit):  Pages:

Actual Stock:  Current Stock:  Issued Books:

Book Description:

[Back to Home](#)

Show 10 entries Search

ID \*

1123 **Python WorkShop**  
 Author - **Ahmad** | Genre - **Machine Learning** | Language - **English**  
 Publisher - **Tufail** | Publish Date - **2023-10-05** | Pages - **1123** | Edition - **12**  
 Cost - **123** | Actual Stock - **8** | Available Stock - **8**  
 Description - **hello**

12 **Think And Grow Rich**  
 Author - **Ahmad** | Genre - **Machine Learning** | Language - **Urdu**  
 Publisher - **Tufail** | Publish Date - **2023-10-03** | Pages - **12** | Edition - **12**  
 Cost - **01** | Actual Stock - **7** | Available Stock - **6**  
 Description - **think and grow rich**


2234E **Past Forward**  
 Author - **Hilal** | Genre - **Machine Learning** | Language - **English**  
 Publisher - **Peer** | Publish Date - **2023-11-16** | Pages - **234** | Edition - **13**  
 Cost - **123** | Actual Stock - **8** | Available Stock - **6**  
 Description - **Did you know that your mind has a 'mind' of its own? Not Without even realize our mind is often governed by another entity which is called sub-conscious mind(Zameer)**

545W **How To Win Friends And Influence People**  
 Author - **Hilal** | Genre - **Machine Learning** | Language - **English**  
 Publisher - **Peer** | Publish Date - **2023-11-16** | Pages - **234** | Edition - **13**  
 Cost - **123** | Actual Stock - **8** | Available Stock - **8**  
 Description - **Did you know that your mind has a 'mind' of its own? Not Without even realize our mind is often**

## 8.7 Admin User Management Module:

**Elibrary** Home About Us Terms

**User Details:**



Reg. No:  Full Name:  Account Status:

Date Of Birth:  Contact No:  Email ID:

State:  City:  Pincode:

Full Address:

[Back to Home](#)

**User List**


Show 10 entries Search

Reg No	Name	Status	Contact No	Email ID	State	City
1122		active	098765432		Jammu and Kashmir	Sopore
1123		active			Jammu and Kashmir	Sopore


Showing 1 to 2 of 2 entries Previous 1 Next

[Admin Login](#)

## 8.7 Admin Publisher Management Module

 Elibrary [Home](#) [About Us](#) [Terms](#) [View Books](#) [User Login](#) [Sign Up](#)

### Publisher Details



Publisher ID:  ID  Publisher Name:  Publisher Name

[<<< Back to Home](#)

### Publisher List

Show  entries Search:


Publisher ID	Publisher Name
1122	Hello
56DE	W
ER12	A

Showing 1 to 3 of 3 entries Previous  Next


[Admin Login](#)

© All right Reversed. By Hilal Hafeez and Zameer Mir

## 8.9 Admin Author Management Module

 Elibrary [Home](#) [About Us](#) [Terms](#) [View Books](#) [User Login](#) [Sign Up](#)

### Author Details



Author ID:  ID  Author Name:  Author Name

[<<< Back to Home](#)

### Author List

Show  entries Search:

Author ID	Author Name
1123	
23E101	A
ERP112	d

Showing 1 to 3 of 3 entries Previous  Next

[Admin Login](#)

© All right Reversed. By Hilal Hafeez and Zameer Mir

# ***Chapter 9 Conclusion***

***“A view of the Developers”***

## **9.0 Conclusion**

This website provides a computerized version of library management system which will benefit the students as well as the staff of the library. It makes entire process online. Library Management System allows the user to store the book details and the customer details. This software package allows storing the details of all the data related to library. The system is strong enough to withstand regressive yearly operations under conditions where the database is maintained and cleared over a certain time of span. The implementation of the system in the organization will considerably reduce data entry, time and also provide readily calculated reports. Having an online platform for students and staff to access and manage library resources can indeed improve efficiency and convenience.

# ***Chapter 10 Future Scope***

***“What can be added to the system in the future”***

## **Future Scope:**

### **Artificial Intelligence Integration:**

Future Library Management Systems (LMS) are likely to incorporate artificial intelligence (AI) for tasks like personalized recommendations, predictive analytics for resource usage, and automated content categorization.

### **Enhanced User Analytics:**

LMS will evolve to provide more robust user analytics, allowing institutions to gain deeper insights into user behavior, preferences, and learning patterns. This data-driven approach will aid in tailoring services and resources to individual needs.

### **AI-Driven Personalization:**

Implement AI algorithms to personalize user experiences by recommending relevant resources based on individual preferences, learning history, and behavioral patterns.

### **To Implement email and SMS technology into the system**

Integrating email and SMS technology into the system enhances communication, enabling efficient notifications for overdue items, reservation confirmations, and important announcements, thereby fostering improved user engagement and interaction.

# ***Chapter 11 Bibliography***

*“Helping material—what and from where?”*



## **Bibliography**

The following books were referred during the analysis and execution phase of the project.

### **MICROSOFT .NET WITH C#**

Microsoft .net series

### **ASP .NET 2.0 PROFESSIONAL**

Wrox Publishers

### **ASP .NET WITH C# 2005**

Apress Publications

### **C# COOK BOOK**

O reilly Publications

### **PROGRAMMING MICROSOFT ASP .NET 2.0 APPLICATION**

Wrox Professional Guide

### **BEGINNING ASP .NET 2.0 E-COMMERCE IN C# 2005**

Novice to Professional.

### **WEBSITES:**

[www.google.com](http://www.google.com)

[www.microsoft.com](http://www.microsoft.com)

[W3Schools Online Web Tutorials](http://W3Schools Online Web Tutorials)

[GeeksforGeeks | A computer science portal for geeks](http://GeeksforGeeks | A computer science portal for geeks)