1.INTRODUCTION

1.1 LIBRARY MANAGEMENT SYSTEM

Library management system is a project which aims in developing a computerized system to maintain all the daily work of library. This project has many features which are generally not available in normal library management system like user or faculty/student login. It has a faculty of admin login through which the admin can monitor the whole system. The user after logging in to his account i.e., admin account can generate various reports such as student details, book borrowed by students and returned, book entries such as name of the book, author and category. The aim of the project is to make the manual handling of library system in to computerized system which includes all above features.

The "Library Management System" has been developed to override the problems prevailing in the practicing manual system. This software is supported to eliminate and in some cases reduce the hardships faced by this existing system. Moreover
this system is designed for the particular need of the company to carry out operations in a smooth and effective manner. The application is reduced as much as possible to avoid errors while entering the data. It also provides error messages while entering invalid data. These systems will ultimately allow you to better manage resources.

1.2 OBJECTIVES

- A Library Management System gives access to and manage the resources in your library.
- A well-chosen system will increase your library's efficiency, save valuable administration time, lead to a better educational experience, information and to use them effectively.
- This project is helpful to track all the book and library information.
- The software will be able to handle all the necessary information.

2. SYSTEM ANALYSIS

Existing system:

In our existing system all the transactions of books are done manually, so taking more time for transaction like borrowing a book or returning a book and also for searching of books. Another major disadvantage is that to preparing the list of books borrowed and the available books in the library will take more time, currently it is doing as a one-day process for verifying all the records. So after conducting the feasibility study we decided to make a computerized library management system.

Proposed system:

Proposed system is an automated Library Management System. Through our software user can add members, add books, search members, search books, update information, edit information, borrow and return books in quick time. Our purposed system has the following advantages.

- User friendly interface
- Fast access to database
- Less error
- More storage capacity
- Quick transaction

All the manual difficulties in managing the library have been rectified by implementing computerization.

Module Description

Books:

This module consists the details of the book available in the library and their categories. It also contains the list of books in each category and their details.

Member Account:

To issue a book from the library, one should have an account in the library.

The registration contains all the details about the member like member id, name, address, contact number, roll number etc.

Book Request:

This module is used by the member to request a book from the library. The search can be performed by entering the book id.

Issue of books:

This module is used by the librarian to issue a book based on the request made by the member.

Returning books:

In this module the librarian maintains the details of the books returned by the member, which also includes the fine details.

Extend due date:

In this module the librarian can extend the due date if the students wish to return the issued book later than actual due date.

Reports:

This module includes the details about the issued books, returned books, member reports, fine reports and details of the book which are not returned.

3. HARDWARE & SOFTWARE REQUIREMENTS

3.1 HARDWARE INTERFACES

Processor	:	i3+
RAM	:	4GB
Hard Disk	:	80GB
Speed	:	1.2 GHz+

3.2 SOFTWARE INTERFACES

Operating System	:	Windows 7 or Higher
IDE	:	Visual Studio 2019
Language	:	Visual Basic
Framework	:	VB.NET 4.0
Back End	:	MY SQL

4.SOFTWARE REQUIREMENT SPECIFICATION

4.1 PRODUCT PERSPECTIVE

This is totally self-contained and works efficiently.it provides simple database rather than complex ones for high requirements and it provides good and easy graphical user interface new, naive as well as experienced users of the computers.

4.2 FUNCTIONAL REQUIREMENTS

1.NORMAL USER

1.1 USER LOGIN

DESCRIPTION OF FEATURE

This feature used by the user to login in to system. They are required to enter user id and password before they are allowed to enter the system. The user id and password will be verified and if invalid id is there user is allowed to not enter the system.

Functional requirements

- User id is provided when they register
- The system must only allow user with valid id and password to enter the system
- The system performs authorization process which decide what decided what user
- level can access to.
- The user must be able to logout after they finished using system.

1.2 REGISTER NEW USER

DESCRIPTION OF FEATURE

This feature can be performed by all users to register new user to create account.

Functional requirements

- System must be able to verify information.
- System must be able to delete information if the information is wrong.

1.3 REGISTER NEW BOOK

DESCRIPTION OF FEATURE

This feature allows to add new books to the library.

Functional requirements

- System must be able to verify information.
- System must be able to enter number of copies into table.
- System must be able to not allow two books having same book id.

1.4 SEARCH BOOK

DESCRIPTION OF FEATURE

This feature id found in book maintenance part. We can search book based on book id, book name, publication or by author name.

Functional requirements

- System must be able to search the database based on select search type.
- System must be able to filter books based on keyword entered.
- System must be able to show the filtered book in table view.

1.5 ISSUE BOOKS AND RETURN BOOKS

DESCRIPTION OF FEATURE

This feature allows to issue and return books and also view reports of books issued.

Functional requirements

- System must be able to enter issue information in database.
- System must be able to update number of books.
- System must be able to search if book is available or not before issuing books.
- System should be able to enter issue and return date information

1.6 EXTEND DUE DATE

DESCRIPTION OF FEATURE

This feature allows to extend the due date if the member wish to extend the due date of the book.

Functional requirements

- System must be able to check the due date to return the book.
- System must be able to extend the due date based on number of dates requested by the member

4.3 MICROSOFT VISUAL STUDIO

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms such as

Windows API, Windows Forms, Windows Presentation Foundation, Windows Store and Microsoft Silverlight. It can produce both native code and managed code.

Visual Studio includes a code editor supporting IntelliSense as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a code profiler, designer for building GUI applications, web designer, class designer, and database schema designer. It accepts plug-ins that expand the functionality at almost every level including adding support for source control systems and adding new toolsets like editors and visual designers for domain-specific languages or toolsets for other aspects of the software development lifecycle.

.NET FRAMEWORK SDK

The .NET framework is an integral windows component that supports building and running the next generation of applications and XML web services. The key component of the .NET frame work are the common language run time and the .NET frame work class library, which includes ADO.NET, ASP.NET and windows forms. The .NET framework provides a managed execution environment simplified development and deployment and integration with a wide variety of programming languages. This framework is made up of the following parts:

- The common language runtime (CLR).
- The base class libraries.
- Object oriented internet development with ASP.NET.
- Rich client user interface using windows forms.
- RAD for the internet using web forms.

OVERVIEW OF .NET FRAMEWORK

- The .NET framework is a new computing platform that simplifies application development in the highly distributed environment of the internet. The .NET framework is designed to fulfil following objectives:

  To provide a consistent object-oriented programming environment whether object code is stored and executed locally but internet- distributed or executed remotely.
- To provide a code execution environment that minimizes software deployment and versioning conflicts.
- To provide a code execution environment that guarantees safe execution of code, including code created by an unknown or semi trusted third party.
- To provide a code execution environment that eliminates the performance problem of scripted or interpreted environments.
- To make the developer experience consistent across widely types of application, such as windows-based applications and web-based applications.
- To build all communication on industry standards to ensure that code based on the .NET framework can integrate with any other code.

The .NET framework has two main components: the common language runtime and the .Net framework class library. The common language runtime is the foundation of the .NET framework. You can think of the runtime as an agent that manages code at execution time, and removing while also enforcing strict type safely and other forms of code accuracy that ensure security and robustness in fact the concept of code management is a fundamental principle of the runtime.

Code that targets the runtime is known as managed code, while code that does not target the runtime is known as un managed code. The class library, the other main component

of the .NET frameworks is a comprehensive, object-oriented collection reusable types that you can use to develop applications ranging from traditional command line or graphical user interface (FGUI) applications to application base d on the latest innovations provided by ASP.NET, such as web forms and XML web services.

The .NET framework can be hosted by unmanaged component that load the common language runtime into their processes and initiate the execution of managed code. ASP.NET works directly with the runtime to enable ASP.NET application and XML web services, both of which are discussed later in this topic, Internet explorer is an example of unmanaged application that hosts the runtime.

Using internet explorer to the host runtime enables you to embed managed components or windows forms controls in HTML documents. Hosting the runtime in this way makes mobile code 9similar to Microsoft Active Xr controls) possible, but with significant improvement that only managed code can offer, such as semi-trusted execution and secure isolated file storage.

The following illustration shows the relationship of the common language runtime and the class library to your application and to the overall system. The illustration also shows how managed code operated with in a larger architecture.

We can use the .NET framework to develop the following types of application and services:

- Console applications
- Window GUI application (Windows Forms) ASP.NET applications
- XML Web services
- Windows services

COMMON LANGUAGE RUNTIME (CLR)

The common language runtime (CLR) is responsible for runt-time services such as language integration; security enforcement; and memory, process and thread management. In addition, it has a roll at development time when features such as life cycle management strong type naming, cross-language exception handling, dynamic binding and so on, reduce the amount of code that a developer must write to turn the business logic the reusable component. The runtime can be hosted by high performance, server-side applications, such a s Microsoft Internet Information Services (IIS) for building web applications with ASP.NE and the next release of Microsoft SQL Server.

This infrastructure enables you to use code "managed "by the .NET framework to write your business logic, while still enjoying the superior performance of the industry's best enterprises servers that support runtime hosting.

VB .NET

VB.NET stands for Visual Basic.NET, and it is a computer programming language developed by Microsoft. It was first released in 2002 to replace Visual Basic 6. VB.NET is an object-oriented programming language. This means that it supports the features of object-oriented programming which include encapsulation, polymorphism, abstraction, and inheritance.

Visual Basic .ASP NET runs on the .NET framework, which means that it has full access to the .NET libraries. It is a very productive tool for rapid creation of a wide range of Web, Windows, Office, and Mobile applications that have been built on the .NET framework.

ADO .NET

ADO.NET provides consistent access to data sources such a Microsoft SQL Server and XML, as well as to data sources exposed through OLE DB and ODBC. Data sharing consumer applications can use ADO.NET to connect to these data sources and retrieve, manipulate and update the data that they contain. ADO.NET separates data access from

data manipulation into discrete components that can be used separately or in tandem. ADO.NET includes .NET Frame work data providers for connecting to a database, executing commands and retrieving results. Those results are either processed directly, placed in and ADO.NET Dataset objects in order to be exposed to the used in an ad hoc manner, combined with data from multiple sources or remote between tiers. The ADO.NET Dataset object can also be used independently of a .NET Framework data provider to manage data local to the application or sourced from XML.

The ADO.NET classes are found in System.Data.dll and are integrated with the XML classes found in System.Xml.dll. When compiling code that uses the System. Data, namespace reference both System.Data.dll and System.Xml.dll. ADO.NET provided functionality to developers writing managed code similar to the functionality provided to native component object model (COM) developers by ActiveX Data Objects (ADO).

ADO .NET COMPONENTS

There are two components of ADO.NET that you can use to access and manipulate data:

>>NET Framework data providers.

>> The Dataset

.NET FRAMEWORKS DATA PROVIDERS

The .NET Framework Data providers are components that have been explicitly designed for data manipulation and fast, forward-only, read-only access to data. The connection object provides connectivity to a data source. The command object enables access to database commands to return data, modify data, run stored procedures and send or retrieve parameter information. The Data Adapter provides a high-performance stream of data from the data source. Finally, the Data Adapter provides the bridge between the Dataset object and the data source.

THE DATASET

The ADO.NET Dataset is explicitly designed for data access independent of any data source. As a result, it can be used with multiple and differing data sources used with XML data or used to manage data local to the application. It contains a collection of one or more Data Table objects made up to rows and columns of data as well as primary key, foreign key, constraint and relation information about the data in the Data Table objects.

BENEFITS OF ADO .NET

ADO.NET offers several advantages over previous versions of ADO and over other data access components. These benefits fall into the following categories:

1. Interoperability & Maintainability
2. Programmability & Scalability

MICROSOFT DATA ACCESS COMPONENTS (MDAC)

Microsoft Data Access Components (MDAC) is a collection of core files provided to help applications by providing a means of accessing data. MDAC includes core files for Open Database Connectivity (ODBC), ActiveX Data Objects (ADO), OLEDB, Network libraries and client configuration tool for SQL Server. Depending on your data access strategy, you may need to ensure that MDAC is installed on the client computers, the business servers, the Web servers or the database servers. MDAC 2.6 or later is required by the .NET Framework and at least MDAC 2.7 SP1 is recommended

4.4 MY SQL

My SQL is an open-source relational database management system (RDBMS). MySQL works with an operating system to implement a relational database in a computer's storage system, manages users, allows for network access and facilitates testing database integrity and creation of backups, most frequently used My SQL tools are workbench, phpMyAdmin, HeidiSQL etc.

MySQL Workbench is a visual database design tool that integrates SQL development, administration, database design, creation and maintenance into a single integrated development environment for the MySQL database system. It is the successor to DB Designer 4 from fabFORCE.net, and replaces the previous package of software, MySQL GUI Tools Bundle.

phpMyAdmin is a free and open-source administration tool for MySQL and Maria DB. As a portable web application written primarily in PHP, it has become one of the most popular MySQL administration tools, especially for web hosting services.

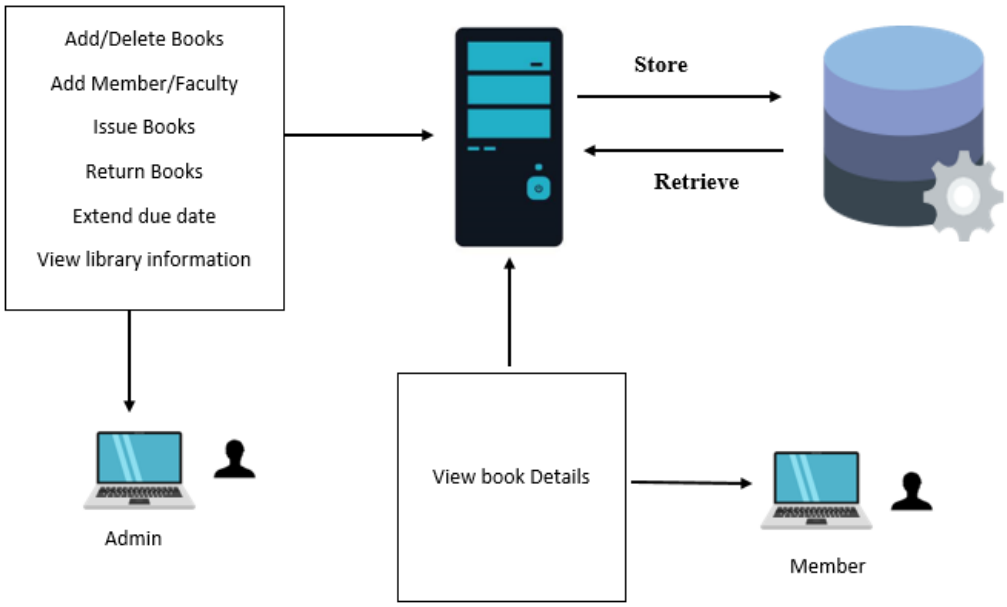
HeidiSQL is a useful and reliable tool designed for web developers using the popular MySQL server, Microsoft SQL databases and PostgreSQL. It enables you to browse and edit data, create and edit tables, views, procedures, triggers and scheduled events.

5. SYSTEM DESIGN

5.1 INTRODUCTION

The purpose of the design phase is to plan a solution of the problem specified by the requirements document. This phase is the first step in moving from the problem domain to the solution domain. In other words, starting with what is needed; design takes us toward how to satisfy the needs. The design of a system is perhaps the most critical factor affecting the quality of the software; it has a major impact on the later phases particularly testing and maintenance.

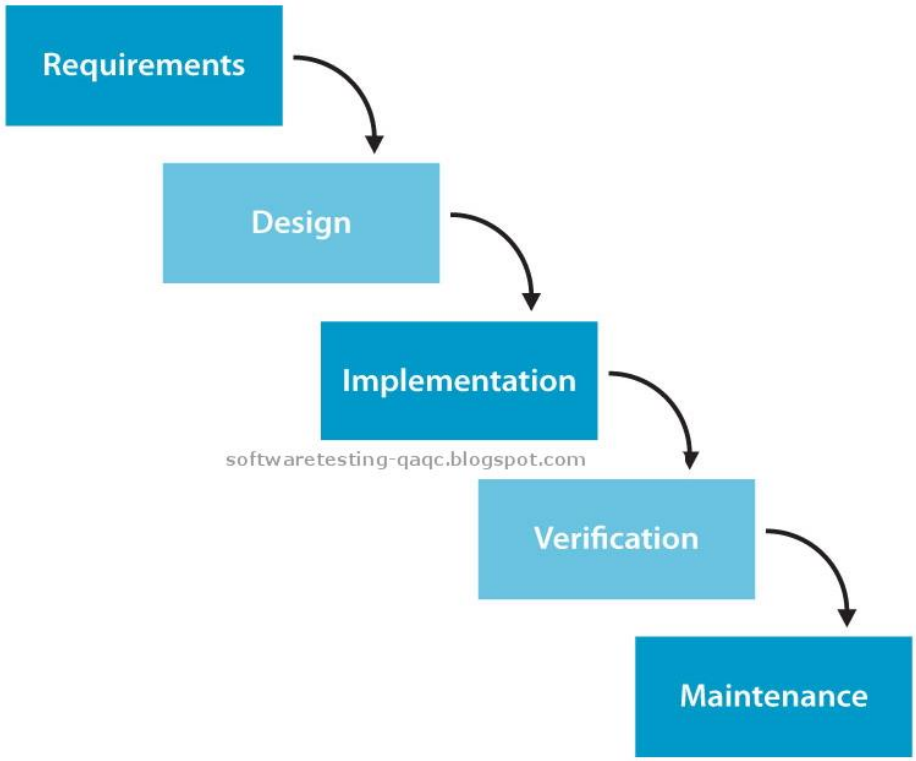
5.5.1 ARCHITECTURE DESIGN



SOFTWARE MODEL USED

WATERFALL MODEL

The waterfall model is a sequential (non-iterative) design process, used in software development processes, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of conception, initiation, analysis, design, construction, testing, production/implementation and maintenance. Despite the development of new software development process models, the Waterfall method is still the dominant process model with over a third of software developers still using it.



The waterfall development model originates in the manufacturing and construction industries: highly structured physical environments in which after-the-fact changes are prohibitively costly, if not impossible. Because no formal software development methodologies existed at the time, this hardware-oriented model was simply adapted for software development.

Advantages of waterfall model:

- This model is simple and easy to understand and use.
- It is easy to manage due to the rigidity of the model – each phase has specific deliverables and a review process.
- In this model phases are processed and completed one at a time. Phases do not overlap.
- Waterfall model works well for smaller projects where requirements are very well understood.

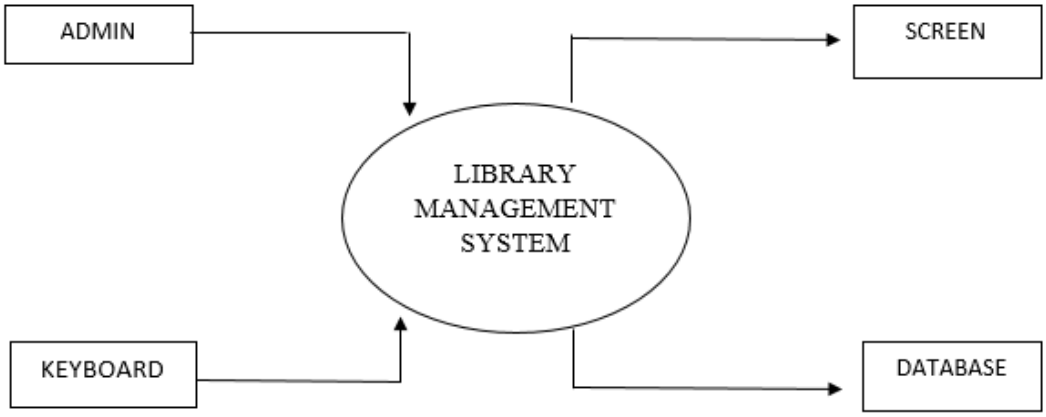
Disadvantages of waterfall model:

- Once an application is in the testing stage, it is very difficult to go back and change something that was not well-thought out in the concept stage.
- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Not a good model for complex and object-oriented projects.
- Poor model for long and ongoing projects.
- Not suitable for the projects where requirements are at a moderate to high risk of changing.

5.2 DATAFLOW DIAGRAM

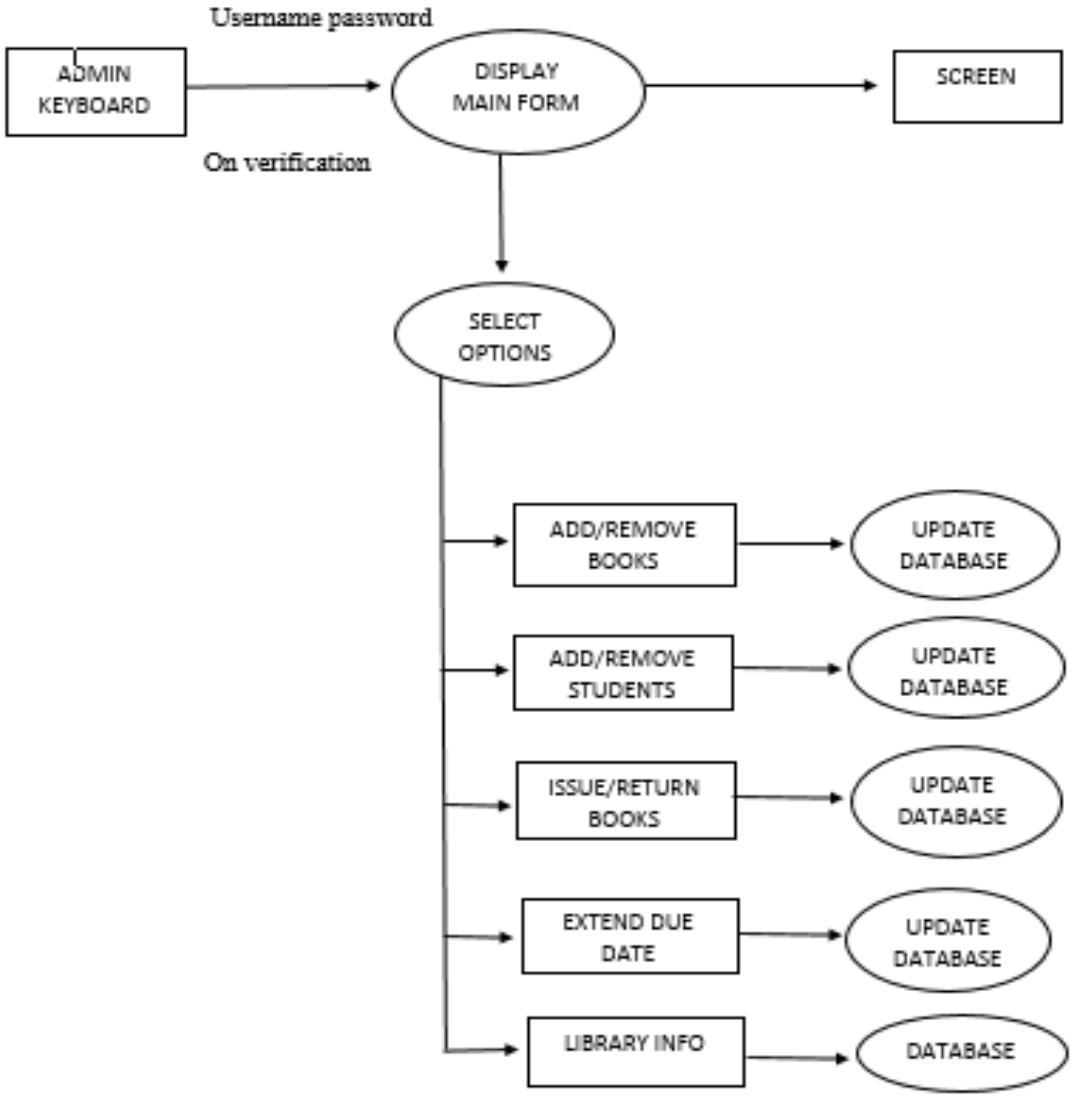
Level-0 DFD:

It is also known as a context diagram. It's designed to be an abstraction view, showing the system as a single process with its relationship to external entities. It represents the entire system as a single bubble with input and output data indicated by incoming/outgoing arrows.



Level-1 DFD:

This level shows all processes at the first level of numbering, data, stores, external entities and the data flows between them. The purpose of this level is to show the major and high level processes of the system and their interrelation.


Username password
ADMIN KEYBOARD
DISPLAY MAIN FORM
SCREEN
On verification
SELECT OPTIONS
ADD/REMOVE BOOKS
UPDATE DATABASE
ADD/REMOVE STUDENTS
UPDATE DATABASE
ISSUE/RETURN BOOKS
UPDATE DATABASE
EXTEND DUE DATE
UPDATE DATABASE
LIBRARY INFO
DATABASE

5.3 USE CASE DIAGRAM

Use case diagrams model the functionality of a system using actors and use cases. Use cases are services or functions provided by the system to its users**. Use case diagrams** are usually referred to as behavior diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors). Each use case should provide some observable and valuable result to the actors or other stakeholders of the system.

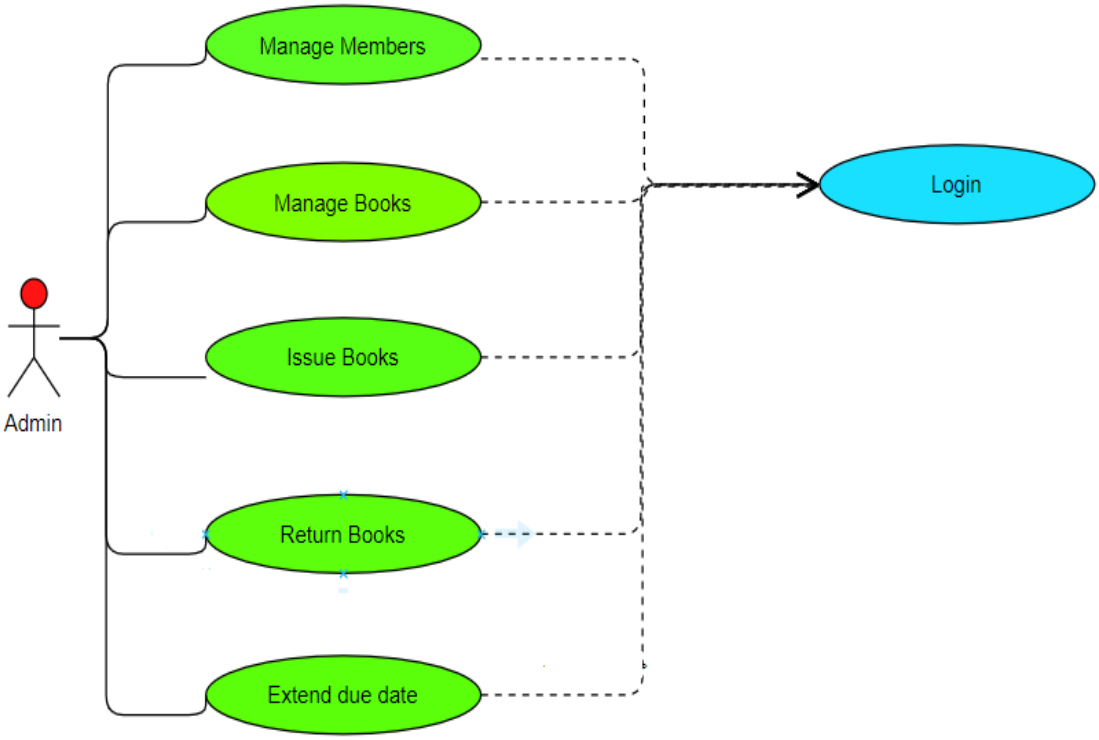
Use cases: A use case describes a sequence of actions that provide something of measurable value to an actor and is drawn as a horizontal ellipse.

Actors: An actor is a person, organization, or external system that plays a role in one or more interactions with your system. Actors are drawn as stick figures.

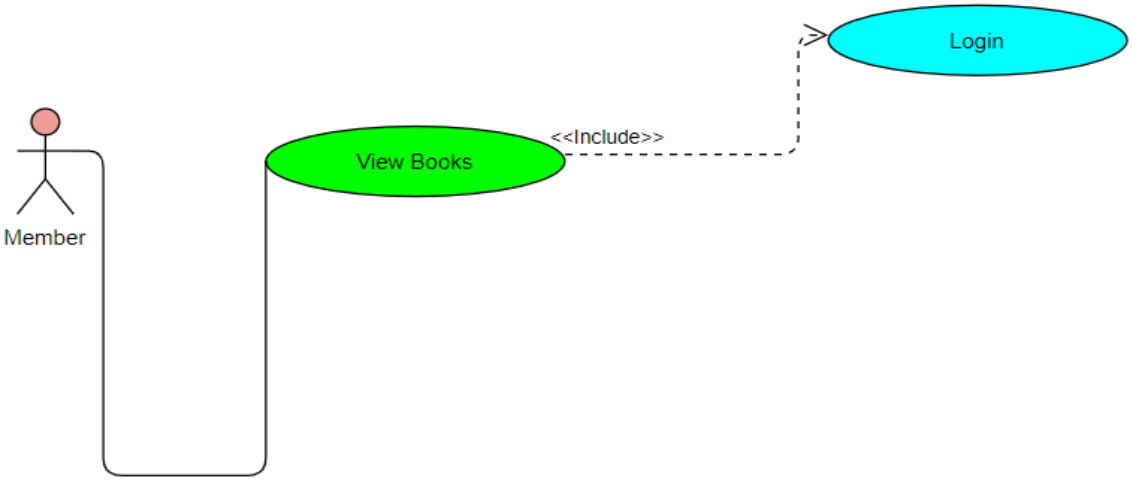
Associations: Associations between actors and use cases are indicated in use Case diagrams by solid lines. An association exists whenever an actor is involved with an interaction described by a use case.

System boundary boxes: You can draw a rectangle around the use cases, called the system boundary box, to indicate the scope of your system. Anything within the box represents functionality that is in scope.

5.3.1 Use Case Diagram for Admin



5.3.2 Use Case Diagram for Member

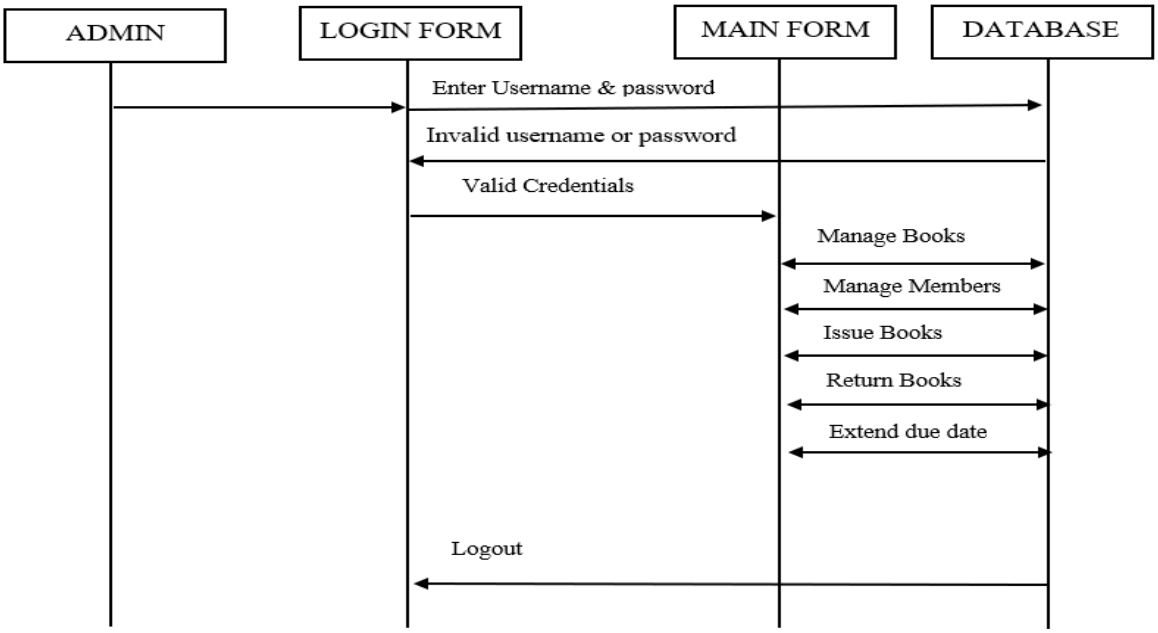


5.4 SEQUENCE DIAGRAM

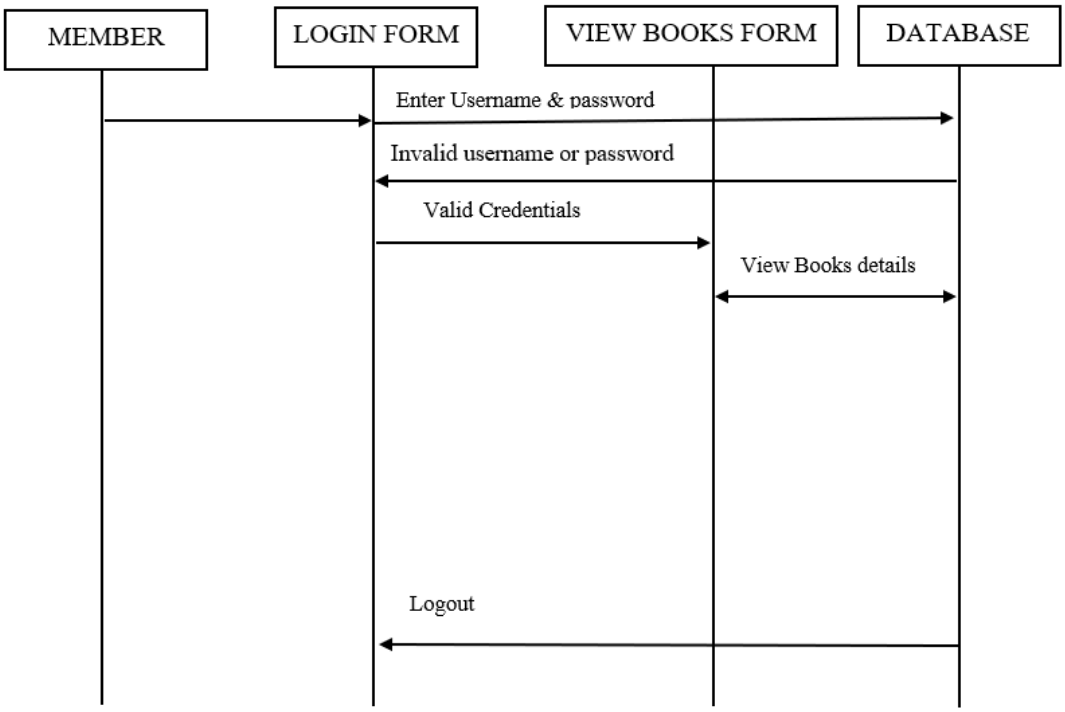
The Sequence Diagram models the collaboration of objects based on time. sequence. It shows how the objects interact with each other's in a particular scenario of a use case. With the advanced visual modelling capability, you can create complex sequence diagram in a few clicks. Besides, Visual paradigm can generate sequence diagram from the flow of events which you have defined in the use case description. The sequence diagram models the collaboration of objects based on a time sequence. It shows how the objects interacts with others in a particular scenario of a use case. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario.

- **Lifelines:** A sequence diagram shows, as parallel vertical lines which indicates different processes or objects that live simultaneously.
- **Message:** Messages written with horizontal arrows with the message name written above them, display interaction. The messages are written in order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.
- **Object/Activation Box/Process:** Activation boxes, or method-call boxes, are opaque rectangles drawn on top of lifelines to represent that processes are being performed in response to the message.

5.4.1 Sequence Diagram for Admin



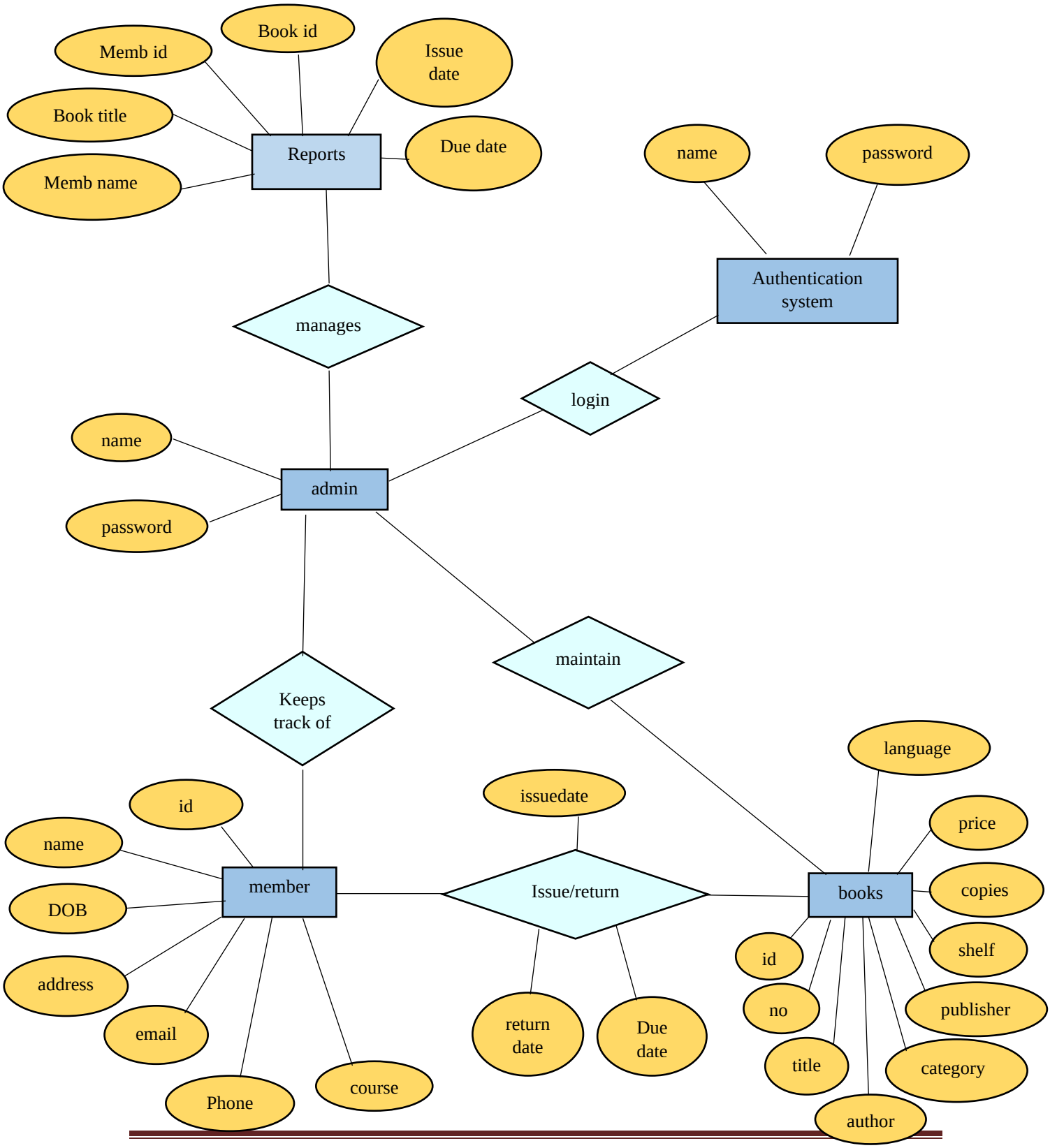
5.4.2 Sequence Diagram for Admin



5.4 ER DIAGRAM

Entity–relationship model (ER model) is a data model for describing the data or information aspects of a business domain or its process requirements, in an abstract way that lends itself to ultimately being implemented in a database such as a relational database. The main components of ER models are entities (things) and the relationships that can exist among them, and databases. ER Diagram is a graphical representation of entities and their relationships to each other, typically used in computing in regard to the organization of data within databases or information systems. An entity is a piece of data-an object or concept about which data is stored. A relationship is how the data is shared between entities. There are three types of relationships between entities:

- **One-to-One:** One instance of an entity is associated with one other instance of another entity.
- **One-to-Many:** One instance of an entity is associated with zero, one or many instances of another entity, but for one instance of entity B there is only one instance of entity A
- **Many-to-Many:** One instance of an entity (A) is associated with one, zero or many instances of another entity (B), and one instance of entity B is associated with one, zero or many instances of entity A.


Book id
Memb id
Issue date
Book title
Reports
Due date
Memb name
name
password
Authentication system
manages
login
name
admin
password
maintain
Keeps track of
language
id
issuedate
price
name
member
Issue/return
books
copies
DOB
shelf
id
address
no
publisher
email
return date
Due date
title
category
Phone
course
author

6. IMPLEMENTATION

6.1 INTRODUCTION

Features of object-oriented paradigm:

This web application is implemented using object-oriented programming language. Object oriented programming is an approach that provides a way of modularizing

programs by creating partitioned memory area for both data and functions that can be used as templates for creating copies of such modules on demand.

- Emphasis is on data rather than procedure.
- Programs are divided into what are known as objects.
- Data structures are designed such that they characterize the objects.
- Methods that operate on the data of an object are tied together in the data structure.
- Objects may communicate with each other through methods.
- New data and methods can be easily added whenever necessary.
- Follows bottom-up approach in program design.
- Data is hidden and cannot be accessed by external functions.
- This project is implemented using three tier architecture. VB.NET is used in the presentation layer, VB classes are used in the Business logic, Table adopter is used in the data tier and MY SQL (database) is used as the backend.

Module

A module is a logically separable partitioning of the program. In terms of common programming language constructs, module can be a macro procedure, a process or a package. Some criteria must be used to select modules so that module supports well-defined abstraction and solvable and modifiable separately.

6.2 Module Description

• **Register:** Customer has to first get registered to the application by entering First name, Last name, Pin, pan number, Email, Address, Contact number etc.

• **Login:** Customer can access to application by providing login details like Card number and a 4-digit Pin.

• **Withdraw:** Customer can withdraw amount if he has enough balance in his account linked to entered card number.

• **Deposit:** Customer can deposit amount to the account linked to entered card number.

• **Get/Return Loan:** Customer can request or pay loans, based on eligibility.

• **Transfer:** Customer can transfer amount to another account that exists in the database.

• **Admin:** Admin can do following activities

6.Update customer details: - Rectify errors in account details.

7.Block/Unblock card: - Block the card at certain situations and unblock when required to.

8.Approve cards: - Approve/Reject card number requested by customers.

9.Approve loans: - Approve/Reject loans requested by customer.

10.View customer details: - Admin can view details of customers using their card numbers.

6.3 DATABASE TABLES

Book details table:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
Book_ID	INT	✓	✓							
Book_no	VARCHAR(45)									NULL
Title	VARCHAR(45)									NULL
Author	VARCHAR(45)									NULL
Category	VARCHAR(45)									NULL
Publisher	VARCHAR(45)									NULL
Language	VARCHAR(45)									NULL
Year	VARCHAR(45)									NULL
Description	VARCHAR(45)									NULL
Image	LONGBLOB									NULL
Price	VARCHAR(45)									NULL
Shelf	VARCHAR(45)									NULL
Available	VARCHAR(45)									NULL
copies	VARCHAR(45)									NULL

Categories table:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
ID	INT	✓	✓					✓		
Category	VARCHAR(45)									NULL
Shelf	VARCHAR(45)									NULL

Course list table:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT	✓	✓					✓		
course	VARCHAR(45)									NULL

Faculty details table:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
member_id	INT	✓	✓							
name	VARCHAR(45)									NULL
address	VARCHAR(100)									NULL
phone_number	VARCHAR(45)									NULL
email	VARCHAR(45)									NULL
designation	VARCHAR(45)									NULL
dob	VARCHAR(45)									NULL
password	VARCHAR(45)									NULL
photo	LONGBLOB									NULL

Issue details table:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
Book_id	INT		✓							
Member_id	VARCHAR(45)		✓							
Book_Title	VARCHAR(45)									NULL
Member_Name	VARCHAR(45)									NULL
IssueDate	VARCHAR(45)									NULL
Due_Date	VARCHAR(45)									NULL
id	INT	✓	✓					✓		
return_status	VARCHAR(45)									NULL

Member details table:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
Member_id	INT	☑	☑	☐	☐	☐	☐	☐	☐	
Name	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
Address	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
Phonenumber	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
Dob	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
Email	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
Photo	LONGBLOB	☐	☐	☐	☐	☐	☐	☐	☐	NULL
Course	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
AdmissionYear	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
Rollno	VARCHAR(60)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
password	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
		☐	☐	☐	☐	☐	☐	☐	☐	

Submitted books table:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
BookId	INT	☐	☑	☐	☐	☐	☐	☐	☐	
MemberId	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
BookTitle	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
MemberName	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
IssueDate	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
DueDate	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
SubmitDate	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
Fine	DOUBLE(7,2)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
ID	INT	☑	☑	☐	☐	☐	☐	☑	☐	
return_status	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
		☐	☐	☐	☐	☐	☐	☐	☐	

User details table:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT	☑	☑	☐	☐	☐	☐	☑	☐	
Username	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
Password	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
Name	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
Address	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
PhoneNumber	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
Email	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
Type	VARCHAR(45)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
		☐	☐	☐	☐	☐	☐	☐	☐	

7. SOURCE CODE

Admin Login:

```
Imports MySql.Data.MySqlClient
Public Class Form1
    Dim sqlconnection As New MySqlConnection
    Dim command As New MySqlCommand
    Dim datareader As MySqlDataReader

    Private Sub LinkLabel1_LinkClicked(sender As Object, e As
LinkLabelLinkClickedEventArgs) Handles LinkLabel1.LinkClicked
        Me.Hide()
        Student_or_Faculty_Login.Show()
    End Sub

    Private Sub loginBtn_Click(sender As Object, e As EventArgs) Handles loginBtn.Click
        sqlconnection = New MySqlConnection
        sqlconnection.ConnectionString =
"server=localhost;user=root;password=manoj123;database=librarydb"

        Try
            sqlconnection.Open()
            Dim query As String
            query = "select * from librarydb.userdetails where username= '" & txtuser.Text &
"' and password='" & txtpass.Text & "'"
            command = New MySqlCommand(query, sqlconnection)
            datareader = command.ExecuteReader

            Dim count As Integer
            count = 0
            While datareader.Read
                count = count + 1
            End While

            If count = 1 Then
                Dim username = datareader.GetString("name")
                MsgBox("Welcome " & username & "  ")
                Me.Hide()
```

```
            Main_Form.Show()
            txtuser.Text = ""
            txtpass.Text = ""
        Else
            MsgBox("Wrong Username or Password!")
            txtuser.Text = ""
            txtpass.Text = ""
        End If

        sqlconnection.Close()



    Catch ex As MySqlException
        MsgBox(ex.Message)

    End Try
  End Sub

End Class
```

Student/Faculty Login:

```
Imports MySql.Data.MySqlClient
Public Class Student_or_Faculty_Login
conn = New MySqlConnection
    conn.ConnectionString                                                    =
("server=localhost;user=root;password=manoj123;database=librarydb")
  Dim command As New MySqlCommand
  Private Sub loginBtn_Click(sender As Object, e As EventArgs) Handles loginBtn.Click
    If cmbtype.Text = "Student" Then
      student_login()
    ElseIf cmbtype.Text = "Faculty" Then
      faculty_login()
    End If
  End Sub

  Private Sub faculty_login()

    Try
      conn.Open()
      Dim query As String
      Dim reader As MySqlDataReader
```

```
        Dim count As Integer = 0
        query = "select * from librarydb.faculty_details where name='" &
txtusername.Text & "' and password='" & txtpassword.Text & "'"
        command = New MySqlCommand(query, conn)
        reader = command.ExecuteReader
        While reader.Read
          count = count + 1
        End While
        If count = 1 Then
          MsgBox("Login succesful", MsgBoxStyle.Information, "Login")
          Me.Hide()
          viewBooks.Show()
        Else
          MsgBox("Wrong username or password", MsgBoxStyle.Critical, "Wrong
credentials")
        End If
      Catch ex As MySqlException
        MsgBox(ex.Message)
      End Try
  End Sub

  Private Sub student_login()
    Try
      conn.Open()
      Dim query As String
      Dim reader As MySqlDataReader
      Dim count As Integer = 0
      query = "select * from librarydb.memberdetails where rollno='" &
txtusername.Text & "' and password='" & txtpassword.Text & "'"
      command = New MySqlCommand(query, conn)
      reader = command.ExecuteReader
      While reader.Read
        count = count + 1
      End While
      If count = 1 Then
        MsgBox("Login succesful", MsgBoxStyle.Information, "Login")
        Me.Hide()
        viewBooks.Show()
      Else
        MsgBox("Wrong username or password", MsgBoxStyle.Critical, "Wrong
credentials")
      End If
    Catch ex As MySqlException
      MsgBox(ex.Message)
```

```
        End Try
    End Sub

End Class
```

Add Member:

```
Imports MySql.Data.MySqlClient
Imports System.IO
Public Class addMember
        sqlconnection = New MySqlConnection("Server=localhost;user=root;password=manoj123;database=librarydb")
  Dim sqlconnection As New MySqlConnection
    Dim command As New MySqlCommand

    Private Sub student()
        Try
            Dim query As String
            query = "insert into librarydb.memberdetails (member_id,name,address,phonenumber,email,course,dob,rollno,admissionyear,photo,password) values (@member_id,@name,@address,@phonenumber,@email,@course,@dob,@rollno,@admissionyear,@photo,@password)"
            Dim ms As New MemoryStream
            picPhoto.Image.Save(ms, picPhoto.Image.RawFormat)
            sqlconnection.Open()
            command = New MySqlCommand(query, sqlconnection)
            command.Parameters.AddWithValue("@member_id", txtMemId.Text)
            command.Parameters.AddWithValue("@Name", txtName.Text)
            command.Parameters.AddWithValue("@Address", txtAddress.Text)
            command.Parameters.AddWithValue("@Phonenumber", txtPhone.Text)
            command.Parameters.AddWithValue("@email", txtEmail.Text)
            command.Parameters.AddWithValue("@course", cmbCourse.Text)
            command.Parameters.AddWithValue("@dob", dateDob.Text)
            command.Parameters.AddWithValue("@Rollno", txtRollNo.Text)
            command.Parameters.AddWithValue("@admissionyear", txtAdmYear.Text)
            command.Parameters.AddWithValue("@Photo", ms.ToArray())
            command.Parameters.AddWithValue("@password", txtpassword.Text)
            command.ExecuteNonQuery()
            sqlconnection.Close()
            MsgBox("Student details stored succesfully", MsgBoxStyle.Information, "Add Member")
```

```
        Catch ex As MySqlException
            MsgBox(ex.Message)
        End Try
    End Sub

    Private Sub faculty()
        Try
            Dim query As String
            query = "insert into librarydb.faculty_details
(member_id,name,address,phone_number,email,dob,photo,designation,password) values
(@member_id,@name,@address,@phone_number,@email,@dob,@photo,@designation
,@password)"
            Dim ms As New MemoryStream
            picPhoto.Image.Save(ms, picPhoto.Image.RawFormat)
            sqlconnection.Open()
            command = New MySqlCommand(query, sqlconnection)
            command.Parameters.AddWithValue("@member_id", txtMemId.Text)
            command.Parameters.AddWithValue("@Name", txtName.Text)
            command.Parameters.AddWithValue("@Address", txtAddress.Text)
            command.Parameters.AddWithValue("@Phone_number", txtPhone.Text)
            command.Parameters.AddWithValue("@email", txtEmail.Text)
            command.Parameters.AddWithValue("@dob", dateDob.Text)
            command.Parameters.AddWithValue("@designation", txtRollNo.Text)
            command.Parameters.AddWithValue("@Photo", ms.ToArray())
            command.Parameters.AddWithValue("@password", txtpassword.Text)
            command.ExecuteNonQuery()
            sqlconnection.Close()
            MsgBox("Faculty details stored succesfully", MsgBoxStyle.Information, "Add
Member")
        Catch ex As MySqlException
            MsgBox(ex.Message)
        End Try
    End Sub


    Private Sub BtnAdd_Click(sender As Object, e As EventArgs) Handles BtnAdd.Click
        If cmbMember.Text = "Student" Then
            student()
        Else
            faculty()
        End If
    End Sub
```

End Class

Add Books:

```
Imports MySql.Data.MySqlClient
Imports System.IO
Public Class AddBooks
  sqlconnection = New MySqlConnection("Server=localhost;user=root;password=manoj123;database=librarydb")
   Dim command As New MySqlCommand

   Private Sub Btnadd_Click(sender As Object, e As EventArgs) Handles Btnadd.Click
      Try
            Dim query As String
        query = "insert into librarydb.book_details (book_id,book_no,title,author,category,publisher,language,year,description,image,price,shelf,available,copies) values (@book_id,@book_no,@title,@author,@category,@publisher,@language,@year,@description,@image,@price,@shelf,@available,@copies) "
        Dim ms As New MemoryStream
        picPhoto.Image.Save(ms, picPhoto.Image.RawFormat)
        sqlconnection.Open()
        command = New MySqlCommand(query, sqlconnection)
        command.Parameters.AddWithValue("@book_id", txtBid.Text)
        command.Parameters.AddWithValue("@book_no", txtBno.Text)
        command.Parameters.AddWithValue("@title", txtTitle.Text)
        command.Parameters.AddWithValue("@author", txtAuthor.Text)
        command.Parameters.AddWithValue("@category", cmbCategory.Text)
        command.Parameters.AddWithValue("@publisher", txtPublisher.Text)
        command.Parameters.AddWithValue("@language", txtLanguage.Text)
        command.Parameters.AddWithValue("@year", txtYear.Text)
        command.Parameters.AddWithValue("@description", txtDesc.Text)
        command.Parameters.AddWithValue("@price", txtPrice.Text)
```

```
            command.Parameters.AddWithValue("@shelf", txtshelf.Text)
            command.Parameters.AddWithValue("@copies", txtcopies.Text)
            command.Parameters.AddWithValue("@available", "Yes")
            command.Parameters.AddWithValue("@image", ms.ToArray())
            command.ExecuteNonQuery()
            sqlconnection.Close()
            MsgBox("Book added succesfully", MsgBoxStyle.Information, "Add Book")
            Me.Close()
                    sqlconnection.Close()
        Catch ex As MySqlException
            MsgBox(ex.Message)
        End Try
    End Sub

End Class
```

Issue Book:

```
Imports MySql.Data.MySqlClient
Public Class issue_books
    Dim sqlconnection As New MySqlConnection
    Dim command As New MySqlCommand
    Dim copies As String

    Private Sub BtnIssue_Click(sender As Object, e As EventArgs) Handles BtnIssue.Click
        sqlconnection = New MySqlConnection
        sqlconnection.ConnectionString                                          =
"server=localhost;user=root;password=manoj123;database=librarydb"
        Try
            sqlconnection.Open()
            Dim query As String
            query = "select * from librarydb.book_details where book_id='" & txtbid.Text &
"'"
            Dim datareader As MySqlDataReader
            command = New MySqlCommand(query, sqlconnection)
            datareader = command.ExecuteReader
            While datareader.Read
                copies = datareader.GetString("copies").ToString
            End While
            If copies <= 0 Then
                set_availability()
                checkavailability()
            Else
                checkavailability()
            End If
```

```
        Catch ex As MySqlException
            MsgBox(ex.Message)
        End Try
    End Sub

    Private Sub book_issuer()
        sqlconnection = New MySqlConnection
        sqlconnection.ConnectionString =
"server=localhost;user=root;password=manoj123;database=librarydb"
        Dim return_status As String = "No"
        Try
            sqlconnection.Open()
            Dim query As String
            query = "insert into librarydb.issue_details
(book_id,member_id,book_title,member_name,issuedate,due_date,return_status) values
('" & txtbid.Text & "','" & txtmemberid.Text & "','" & txtbooktitle.Text & "','" &
txtmemname.Text & "','" & dateBorrow.Text & "','" & dateDue.Text & "','" &
return_status & "') "
            command = New MySqlCommand(query, sqlconnection)
            Dim datareader As MySqlDataReader
            datareader = command.ExecuteReader
            MsgBox("Issue data stored", MsgBoxStyle.Information, "Issue")
                txtbid.Text = ""
            txtmemberid.Text = ""
            txtmemname.Text = ""
            txtbooktitle.Text = ""
            dateBorrow.Text = ""
            dateDue.Text = ""
            sqlconnection.Close()
        Catch ex As MySqlException
            MsgBox(ex.Message)
        End Try
    End Sub

    Private Sub checkavailability()
        sqlconnection = New MySqlConnection
        sqlconnection.ConnectionString =
"server=localhost;user=root;password=manoj123;database=librarydb"
        Dim return_status As String = "No"
        Try
            sqlconnection.Open()
            Dim query As String
            Dim datareader As MySqlDataReader
```

```
        Dim count As Integer = 0
        Dim available As String = "Yes"
        query = "select * from librarydb.book_details where book_id='" & txtbid.Text &
"' and available='" & available & "'"
        command = New MySqlCommand(query, sqlconnection)
        datareader = command.ExecuteReader
        While datareader.Read()
          count = count + 1
        End While

        If count = 1 Then
          book_issuer()
        Else
          MsgBox("Entered book is not available at the moment")
          txtbid.Text = ""
          txtmemberid.Text = ""
          txtmemname.Text = ""
          txtbooktitle.Text = ""
          dateBorrow.Text = ""
          dateDue.Text = ""
        End If
      Catch ex As MySqlException
        MsgBox(ex.Message,)
      End Try
    End Sub

    Private Sub set_availability()
      sqlconnection = New MySqlConnection
      sqlconnection.ConnectionString                                    =
"server=localhost;user=root;password=manoj123;database=librarydb"
      Try
        sqlconnection.Open()
        Dim query As String
        Dim available As String = "No"
        query = "update librarydb.book_details set available='" & available & "'where
book_id='" & txtbid.Text & "'"
        Dim datareader As MySqlDataReader
        Dim count As Integer = 0
        command = New MySqlCommand(query, sqlconnection)
        datareader = command.ExecuteReader
      Catch ex As MySqlException
        MsgBox(ex.Message)
      End Try
    End Sub
```

```
End Class
```

Return Book:

```
Imports MySql.Data.MySqlClient

Public Class Return_Books
    Dim sqlconnection As New MySqlConnection
    Dim command As New MySqlCommand
    Dim copies As String

Private Sub btnReturn_Click(sender As Object, e As EventArgs) Handles btnReturn.Click
        sqlconnection = New MySqlConnection
        sqlconnection.ConnectionString = "server=localhost;user=root;password=manoj123;database=librarydb"
        Dim ret_status As String = "Yes"
        Try
            sqlconnection.Open()
            Dim query As String
            query = "insert into librarydb.submittedbooks (bookid,memberid,booktitle,membername,issuedate,duedate,submitdate,fine,return_status) values ('" & txtBid.Text & "','" & txtmemberid.Text & "','" & txtbooktitle.Text & "','" & txtmemname.Text & "','" & dateBorrow.Text & "','" & dateDue.Text & "','" & SubmitDate.Text & "','" & txtfine.Text & "','" & ret_status & "')"
            command = New MySqlCommand(query, sqlconnection)
            Dim datareader As MySqlDataReader
            datareader = command.ExecuteReader

            MsgBox("Return data stored succesfully", MsgBoxStyle.Information, "Return")
            Me.Hide()
            Main_Form.Show()
            sqlconnection.Close()
        Catch ex As MySqlException
            MsgBox(ex.Message)
        End Try
    End Sub

End Class
```

Book Details:

```
Imports MySql.Data.MySqlClient
Imports System.IO
Public Class Book_details
   Dim sqlconnection As New MySqlConnection
   Dim command As New MySqlCommand
   Private Sub Book_details_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
   End Sub

   Private Sub searchBtn_Click(sender As Object, e As EventArgs) Handles
searchBtn.Click
         sqlconnection.ConnectionString =
"server=localhost;user=root;password=manoj123;database=librarydb"
      Dim datareader As MySqlDataReader
      Try
         sqlconnection.Open()
         Dim query As String
         query = "select * from librarydb.book_details where book_id='" &txtid.Text& "'"
         command = New MySqlCommand(query, sqlconnection)
         datareader = command.ExecuteReader
         Dim count As Integer=0
               While datareader.Read
            count = count + 1
         End While
         If count = 1 Then
            txtbid.Text = datareader.GetString("book_id")
            txtbno.Text = datareader.GetString("book_no")
            txttitle.Text = datareader.GetString("title")
            txtauthor.Text = datareader.GetString("author")
            txtdesc.Text = datareader.GetString("description")
            txtCat.Text = datareader.GetString("category")
            txtpublisher.Text = datareader.GetString("publisher")
            txtLang.Text = datareader.GetString("language")
            txtPrice.Text = datareader.GetString("price")
            txtYear.Text = datareader.GetString("year")
```

```
            txtCopies.Text = datareader.GetString("copies")
            txtSHelf.Text = datareader.GetString("shelf")
            txtavail.Text = datareader.GetString("available")
                End If
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try
    End Sub

End Class
```

Extend due date:

```
Imports MySql.Data.MySqlClient
Public Class extendDueDate
    Dim sqlconnection As New MySqlConnection
    Dim command As New MySqlCommand
    Dim d As String

    Private Sub extendDueDate_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
        dayscmb.Enabled = False
        dtpDue.Enabled = False
        extendBtn.Enabled = False
        dtpDue.Format = DateTimePickerFormat.Custom
        dtpDue.CustomFormat = "dd-MM-yyyy"

    End Sub

    Private Sub dayscmb_SelectedIndexChanged(sender As Object, e As EventArgs)
Handles dayscmb.SelectedIndexChanged
        dtpDue.ResetText()
        dtpDue.Value = DateTime.Parse(d).AddDays(dayscmb.Text)
    End Sub

    Private Sub searchBtn_Click(sender As Object, e As EventArgs) Handles
searchBtn.Click
        Dim datareader As MySqlDataReader
        sqlconnection = New MySqlConnection
        sqlconnection.ConnectionString                                    =
"server=localhost;user=root;password=manoj123;database=librarydb"
        Try
            sqlconnection.Open()
            Dim query As String
```

```
        query = "SELECT * FROM librarydb.issue_details where ( Book_id='" &
txtbid.Text & "' and  member_id=" & txtmemid.Text & " ) and return_status='No' "
        command = New MySqlCommand(query, sqlconnection)
        Dim count As Integer
        datareader = command.ExecuteReader
        While datareader.Read
            count = count + 1
        End While
        If count = 1 Then
            dtpDue.Value = datareader.GetString("due_date")

            dayscmb.Enabled = True
            d = dtpDue.Text.ToString
            extendBtn.Enabled = True
            txtbid.Enabled = False
            txtmemid.Enabled = False

        Else
            MsgBox("Entered data doesnt exists ", MsgBoxStyle.Critical, "Error")
            txtbid.Text = ""
            txtmemid.Text = ""
        End If
    Catch ex As MySqlException
        MsgBox(ex.Message)
    End Try
End Sub

Private Sub extendBtn_Click(sender As Object, e As EventArgs) Handles
extendBtn.Click
    saver()
End Sub

Private Sub saver()
    Dim datareader As MySqlDataReader
    sqlconnection = New MySqlConnection
    sqlconnection.ConnectionString                                          =
"server=localhost;user=root;password=manoj123;database=librarydb"
    Try
        sqlconnection.Open()
        Dim query As String
        query = " update librarydb.issue_details set due_date='" & dtpDue.Value & "'
where book_id ='" & txtbid.Text & "' and member_id='" & txtmemid.Text & "'"
        command = New MySqlCommand(query, sqlconnection)
        datareader = command.ExecuteReader
```

```
            MsgBox("Duedate updated", MsgBoxStyle.Information, "Update")
            Me.Close()
            Main_Form.Show()
        Catch ex As MySqlException
            MsgBox(ex.Message)
        End Try
    End Sub

End Class
```

Issued or returned report:

```
Imports MySql.Data.MySqlClient

Public Class Issued_book_details
    Dim sqlconnection As MySqlConnection
    Dim command As New MySqlCommand
    Dim sda As MySqlDataAdapter
    Dim dbd As DataTable
    Dim bsource As BindingSource
    Private Sub issueBtn_Click(sender As Object, e As EventArgs) Handles issueBtn.Click
        Dim dbd As New DataTable
        dbd.Clear()
        sqlconnection = New MySqlConnection
        sqlconnection.ConnectionString =
          "server=localhost;userid=root;password=manoj123;database=librarydb"
        Dim SDA As New MySqlDataAdapter
        Dim bSource As New BindingSource
        Dim retstat As String = "No"


        Try
            sqlconnection.Open()
            dbd.Clear()
            Dim Query As String
            Query = "select book_id as 'Book id',member_id as 'Member id',Book_title as
'Book title',Member_name as 'Member name',issuedate as 'Issue date',due_date as 'Due
date' from librarydb.issue_details where return_status='" & retstat & "'"
            command = New MySqlCommand(Query, sqlconnection)
            SDA.SelectCommand = command
            SDA.Fill(dbd)
```

```
            bSource.DataSource = dbd
            dgv.DataSource = bSource
            SDA.Update(dbd)
            sqlconnection.Close()
        Catch ex As MySqlException
            MessageBox.Show(ex.Message)
        Finally
            sqlconnection.Dispose()
        End Try
    End Sub

    Private Sub returnBtn_Click(sender As Object, e As EventArgs) Handles returnBtn.Click
        Dim dbd As New DataTable
        dbd.Clear()
        sqlconnection = New MySqlConnection
        sqlconnection.ConnectionString =
          "server=localhost;userid=root;password=manoj123;database=librarydb"
        Dim SDA As New MySqlDataAdapter
        Dim bSource As New BindingSource
        Dim retstat As String = "Yes"
        Try
            sqlconnection.Open()
            dbd.Clear()
            Dim Query As String
            Query = "select bookid as 'Book id',memberid as 'Member id',Booktitle as 'Book title',Membername as 'Member name',issuedate as 'Issue date',duedate as 'Due date' from librarydb.submittedbooks where return_status='" & retstat & "'"
            command = New MySqlCommand(Query, sqlconnection)
            SDA.SelectCommand = command
            SDA.Fill(dbd)
            bSource.DataSource = dbd
            dgv.DataSource = bSource
            SDA.Update(dbd)
            sqlconnection.Close()
        Catch ex As MySqlException
            MessageBox.Show(ex.Message)
        Finally
            sqlconnection.Dispose()
        End Try
    End Sub
End Class
```

Edit member details:

```
Imports MySql.Data.MySqlClient
Imports System.IO
Public Class EditMemberDetails
    Dim sqlconnection As New MySqlConnection
    Dim command As New MySqlCommand

Private Sub deleteBtn_Click(sender As Object, e As EventArgs) Handles deleteBtn.Click
        sqlconnection = New MySqlConnection
        sqlconnection.ConnectionString = "server=localhost;user=root;password=manoj123;database=librarydb"
        Try
            sqlconnection.Open()
            Dim datareader As MySqlDataReader
            Dim query As String
            query = "delete from librarydb.memberdetails where member_id='" & txtmemid.Text & "'"
            command = New MySqlCommand(query, sqlconnection)
            datareader = command.ExecuteReader
            MsgBox("Member details deleted", MsgBoxStyle.Information, "Delete")
            txtmemid.Text = ""
            txtname.Text = ""
            txtAddress.Text = ""
            txtEmail.Text = ""
            txtPhone.Text = ""
            txtRollno.Text = ""
            picPhoto.Image = Nothing
            cmbCourse.Text = ""
            txtadmyear.Text = ""
            dateDob.Text = ""

            sqlconnection.Close()
        Catch ex As MySqlException
            MsgBox(ex.Message)
        End Try

    End Sub

Private Sub updateBtn_Click(sender As Object, e As EventArgs) Handles updateBtn.Click
        If cmbType.Text = "Student" Then
            student_updater()
        ElseIf cmbType.Text = "Faculty" Then
            faculty_updater()
        End If
```

```
    End Sub

Private Sub student_updater()
        Try
            sqlconnection = New MySqlConnection
            sqlconnection.ConnectionString =
"server=localhost;user=root;password=manoj123;database=librarydb"
            Dim query As String
            query = "update librarydb.memberdetails set
name=@name,dob=@dob,address=@address,email=@email,phonenumber=@phonenu
mber,course=@course,admissionyear=@admissionyear,rollno=@rollno,photo=@photo
where member_id='" & txtmemid.Text & "'"
            Dim ms As New MemoryStream
            picPhoto.Image.Save(ms, picPhoto.Image.RawFormat)
            sqlconnection.Open()
            command = New MySqlCommand(query, sqlconnection)
            command.Parameters.AddWithValue("@Name", txtname.Text)
            command.Parameters.AddWithValue("@Address", txtAddress.Text)
            command.Parameters.AddWithValue("@Phonenumber", txtPhone.Text)
            command.Parameters.AddWithValue("@email", txtEmail.Text)
            command.Parameters.AddWithValue("@course", cmbCourse.Text)
            command.Parameters.AddWithValue("@dob", dateDob.Text)
            command.Parameters.AddWithValue("@Rollno", txtRollno.Text)
            command.Parameters.AddWithValue("@admissionyear", txtadmyear.Text)
            command.Parameters.AddWithValue("@Photo", ms.ToArray())
            command.ExecuteNonQuery()
            sqlconnection.Close()
            MsgBox("Member Account has been Updated sucessfully")
            sqlconnection.Close()
        Catch ex As MySqlException
            MsgBox(ex.Message)
        End Try
    End Sub


    Private Sub faculty_updater()
        Try
            sqlconnection = New MySqlConnection
            sqlconnection.ConnectionString =
"server=localhost;user=root;password=manoj123;database=librarydb"
            Dim query As String
            query = "update librarydb.faculty_details set
name=@name,dob=@dob,address=@address,email=@email,phone_number=@phone_n
```

```
umber,designation=@designation,photo=@photo where member_id='" & txtmemid.Text
& "'"
        Dim ms As New MemoryStream
        picPhoto.Image.Save(ms, picPhoto.Image.RawFormat)
        sqlconnection.Open()
        command = New MySqlCommand(query, sqlconnection)
        command.Parameters.AddWithValue("@Name", txtname.Text)
        command.Parameters.AddWithValue("@Address", txtAddress.Text)
        command.Parameters.AddWithValue("@Phone_number", txtPhone.Text)
        command.Parameters.AddWithValue("@email", txtEmail.Text)
        command.Parameters.AddWithValue("@dob", dateDob.Text)
        command.Parameters.AddWithValue("@designation", txtRollno.Text)
        command.Parameters.AddWithValue("@Photo", ms.ToArray())
        command.ExecuteNonQuery()
        sqlconnection.Close()
        MsgBox("Faculty Account has been Updated sucessfully")
        sqlconnection.Close()
      Catch ex As MySqlException
        MsgBox(ex.Message)
      End Try
    End Sub

End Class
```

Main Form:

```
Public Class Main_Form
    Private Sub BtnCategories_Click(sender As Object, e As EventArgs) Handles
BtnCategories.Click
      Me.Hide()
      book_categories.Show()
    End Sub

    Private Sub BtnCourse_Click(sender As Object, e As EventArgs) Handles
BtnCourse.Click
      Me.Hide()
      Course.Show()
    End Sub

    Private Sub BtnAdmins_Click(sender As Object, e As EventArgs) Handles
BtnAdmins.Click
      Me.Hide()
      Manage_admins.Show()
    End Sub
```

```
    Private Sub LibraryBtn_Click(sender As Object, e As EventArgs) Handles LibraryBtn.Click
        Me.Hide()
        Library_info.Show()
    End Sub

    Private Sub issueBtn_Click(sender As Object, e As EventArgs) Handles issueBtn.Click
        Me.Hide()
        issue_books.Show()
    End Sub

    Private Sub ReturnBtn_Click(sender As Object, e As EventArgs) Handles ReturnBtn.Click
        Me.Hide()
        Return_Books.Show()
    End Sub

    Private Sub ManageBtn_Click(sender As Object, e As EventArgs) Handles ManageBtn.Click
        Me.Hide()
        AddBooks.Show()
    End Sub

    Private Sub ManageMembersBtn_Click(sender As Object, e As EventArgs) Handles ManageMembersBtn.Click
        Me.Hide()
        Manage_Members.Show()
    End Sub

    Private Sub ExtendBtn_Click(sender As Object, e As EventArgs) Handles ExtendBtn.Click
        Me.Hide()
        extendDueDate.Show()
    End Sub

    Private Sub IssuedBtn_Click(sender As Object, e As EventArgs) Handles IssuedBtn.Click
        Me.Hide()
        Issued_book_details.Show()
    End Sub
```

```
    Private Sub BtnAbout_Click(sender As Object, e As EventArgs) Handles
BtnAbout.Click
        Me.Hide()
        About.Show()
    End Sub

    Private Sub btnLogout_Click(sender As Object, e As EventArgs) Handles
btnLogout.Click
        Me.Hide()
        Form1.Show()
    End Sub

    Private Sub BookAvailabilityBtn_Click(sender As Object, e As EventArgs) Handles
BookAvailabilityBtn.Click
        Me.Hide()
        Book_availability.Show()
    End Sub

End Class
```

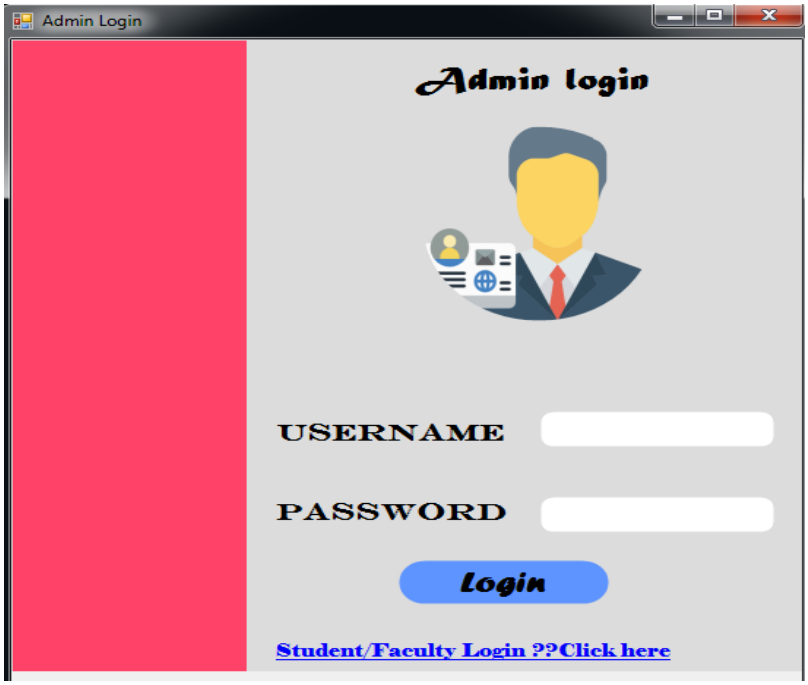
Library Information:

```
Imports MySql.Data.MySqlClient
Public Class Library_info
    Dim sqlconnection As New MySqlConnection
    Dim command As New MySqlCommand

    Private Sub Library_info_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
        queryloader()
    End Sub

    Private Sub queryloader()
        sqlconnection = New MySqlConnection
        sqlconnection.ConnectionString                                    =
"server=localhost;user=root;password=manoj123;database=librarydb"
        Try
            sqlconnection.Open()
            Dim query As String
            query = "select count(book_id) from librarydb.book_details"
            command = New MySqlCommand(query, sqlconnection)
```

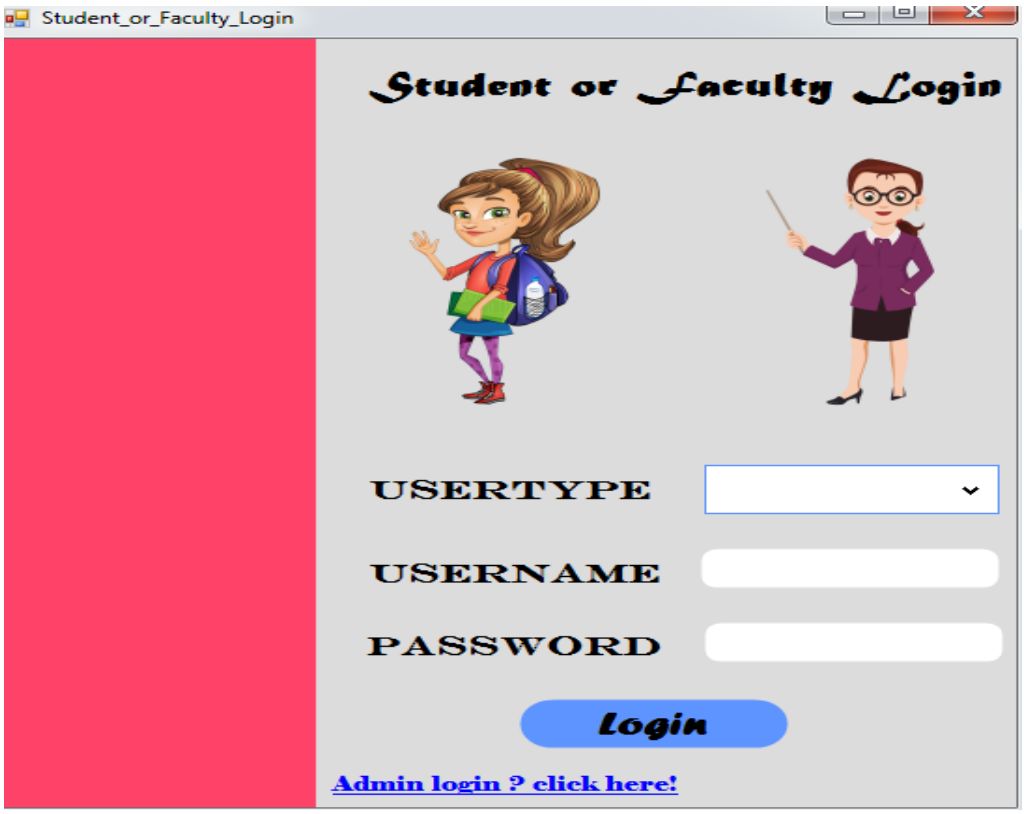
```
            txtTotalBooks.Text = command.ExecuteScalar.ToString

            query = "select sum(price) from librarydb.book_details"
            command = New MySqlCommand(query, sqlconnection)
            txtCostBooks.Text = command.ExecuteScalar.ToString

            query = "select count(book_id) from librarydb.issue_details"
            command = New MySqlCommand(query, sqlconnection)
            txtIssuedBooks.Text = command.ExecuteScalar.ToString

            query = "select count(book_id) from librarydb.book_details where
available='Yes'"
            command = New MySqlCommand(query, sqlconnection)
            txtAvailableBooks.Text = command.ExecuteScalar.ToString

            query = "select count(member_id) from librarydb.memberdetails "
            command = New MySqlCommand(query, sqlconnection)
            txtTotalMembers.Text = command.ExecuteScalar.ToString

            query = "select sum(price) from librarydb.book_details "
            command = New MySqlCommand(query, sqlconnection)
            Dim totalcost As String = command.ExecuteScalar.ToString
            txtCostBooks.Text = totalcost + " Rupees"



            query = "select sum(fine) from librarydb.submittedbooks "
            command = New MySqlCommand(query, sqlconnection)

            Dim totalfine As String = command.ExecuteScalar.ToString
            txtTotalFine.Text = totalfine + " Rupees"

        Catch ex As MySqlException
            MsgBox(ex.Message)
        End Try
    End Sub
End Class
```

8. SCREENSHOTS

Login form:

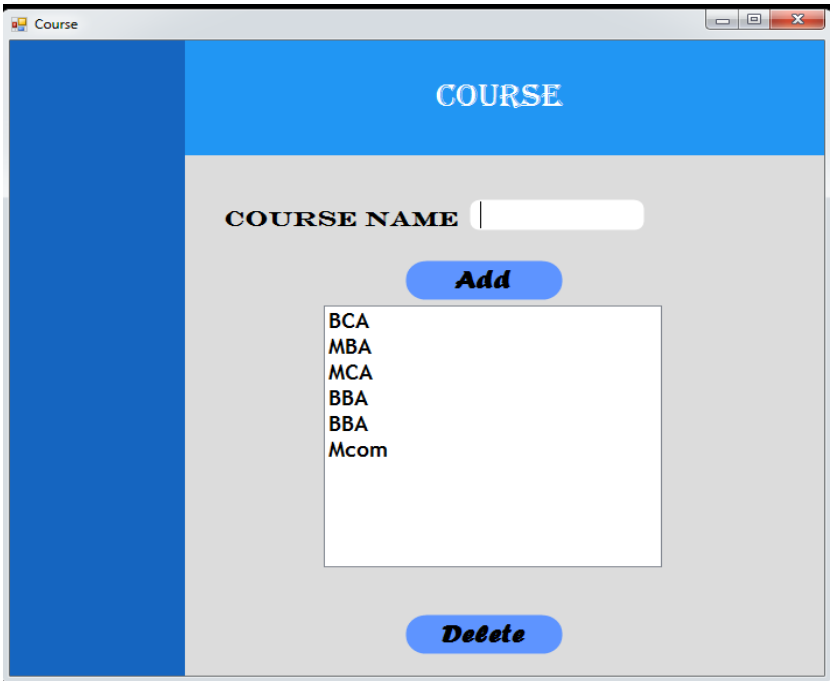


Student/faculty login form:

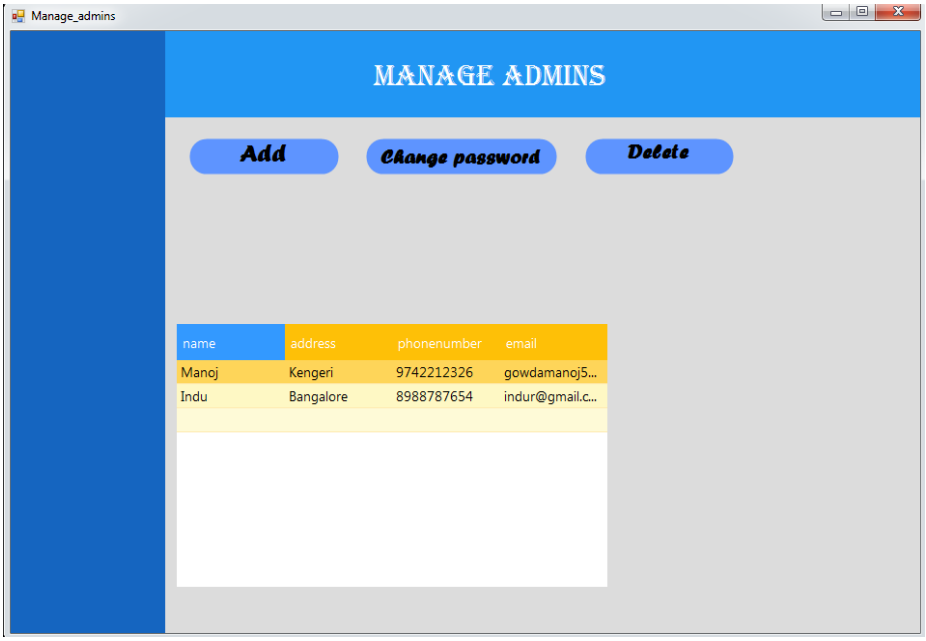


Books category form:

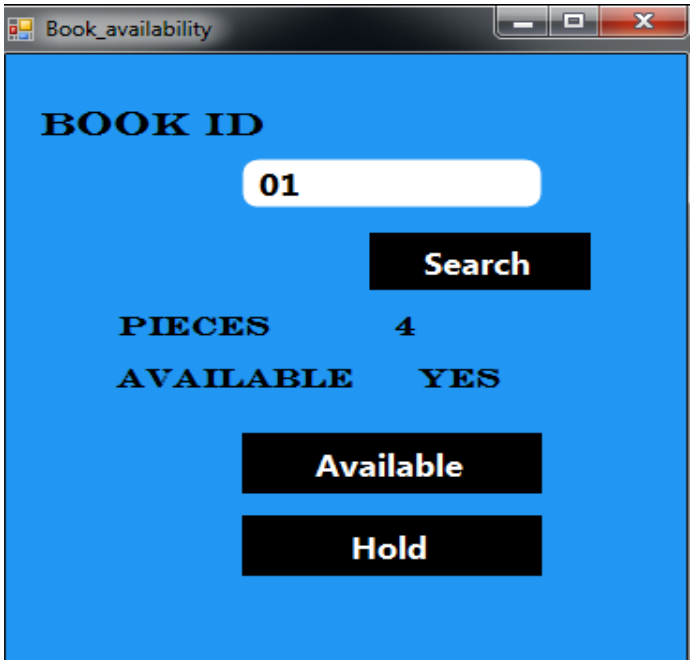
Course form:



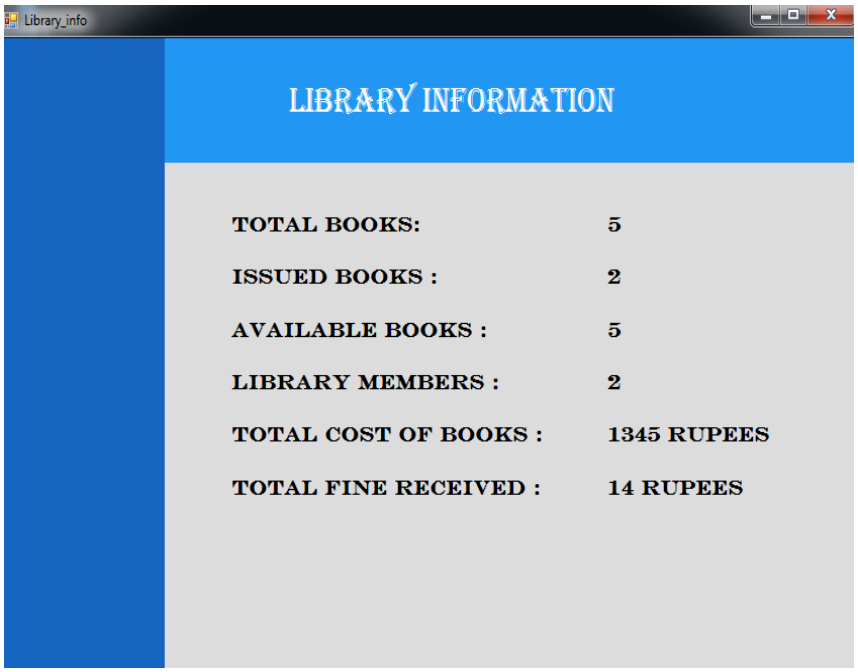
Manage admins form:



Book availability form:

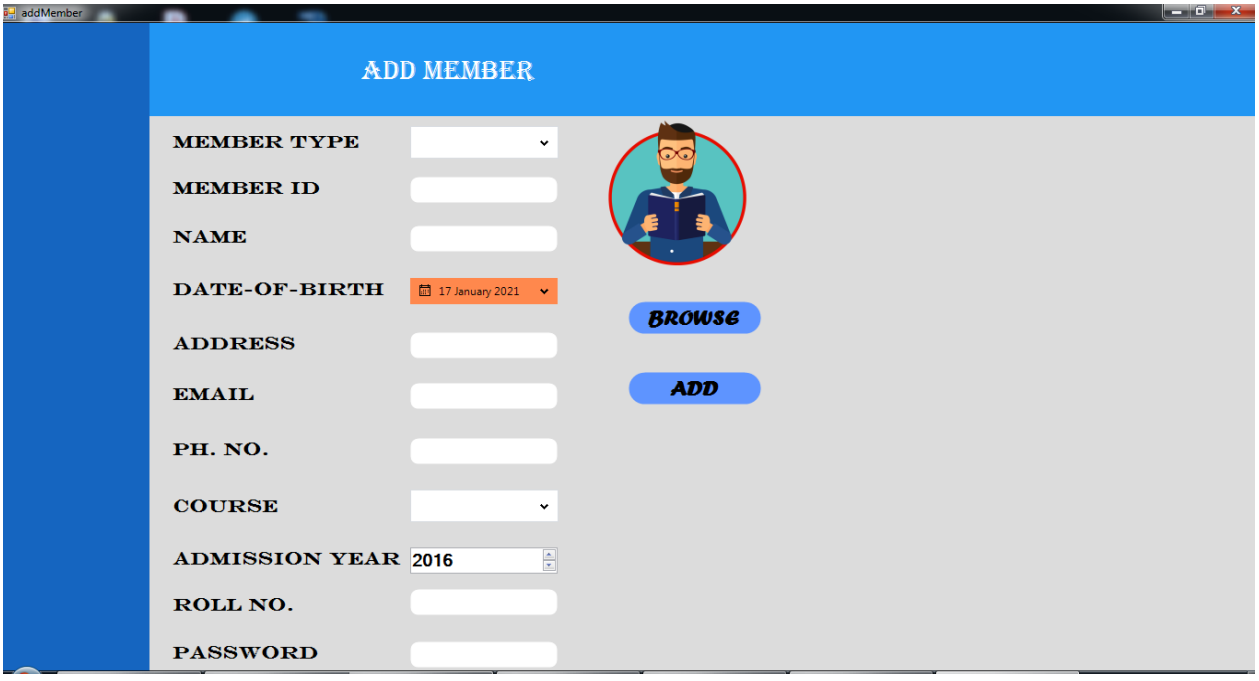


Library information form:

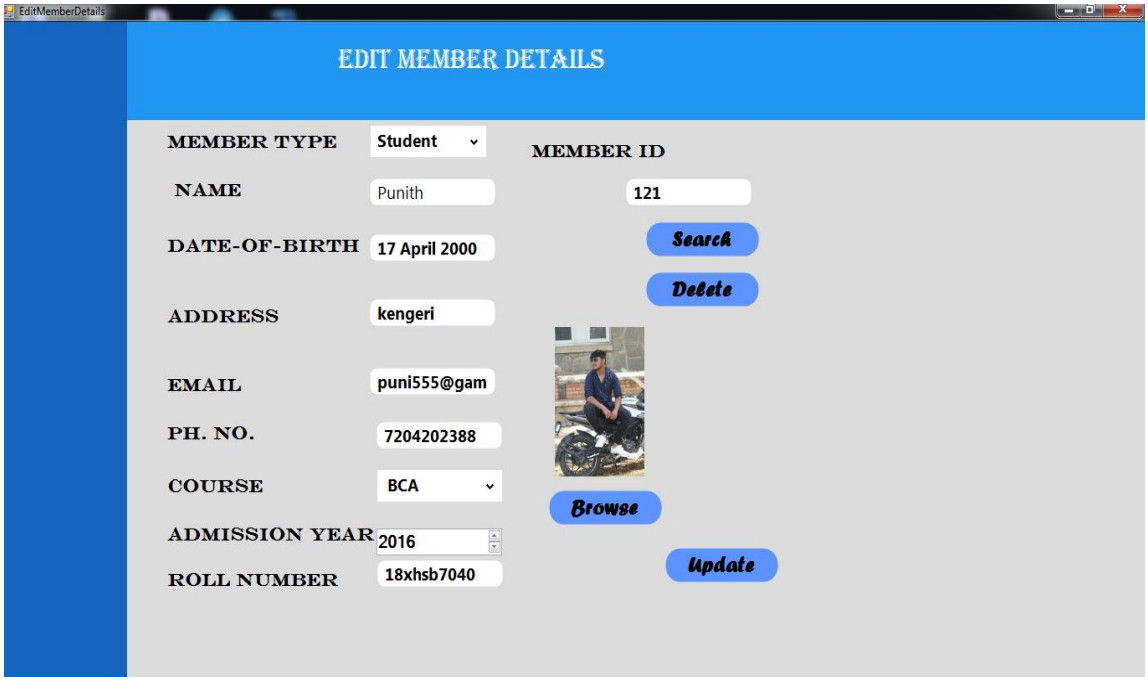


Manage members form:

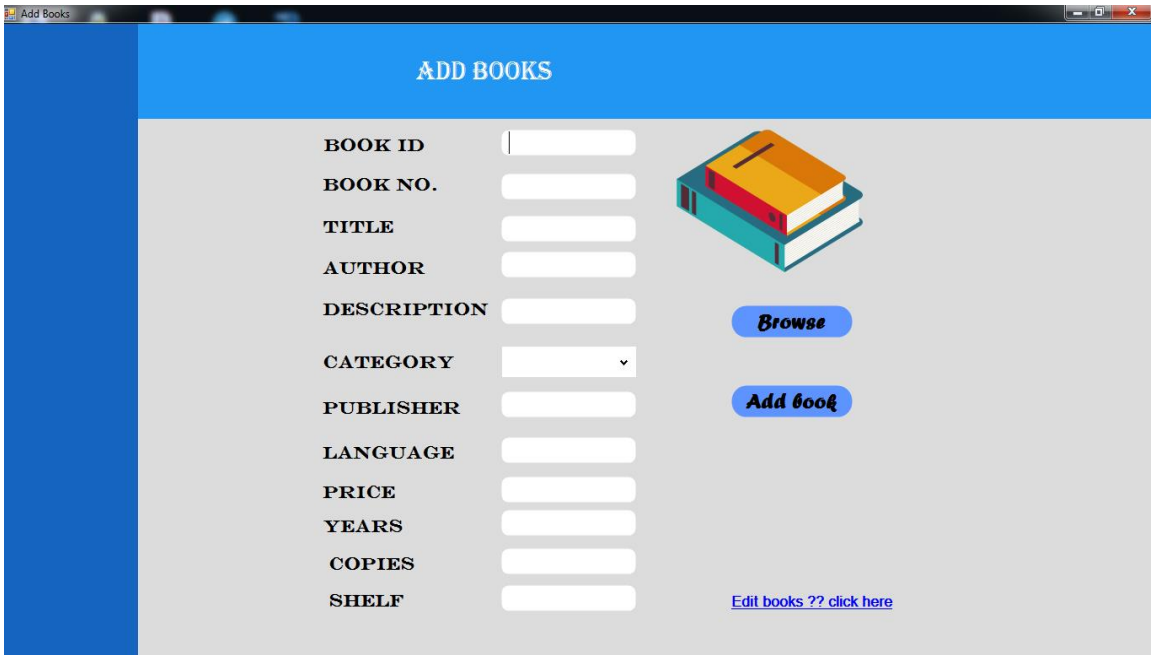
Add member form:



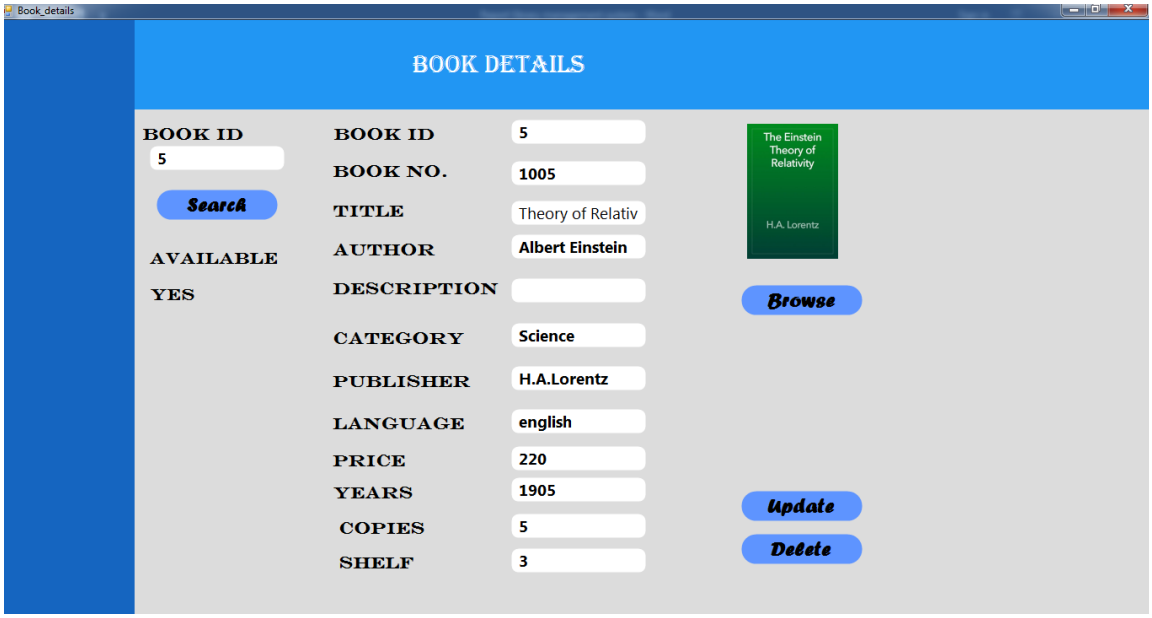
Edit member details:



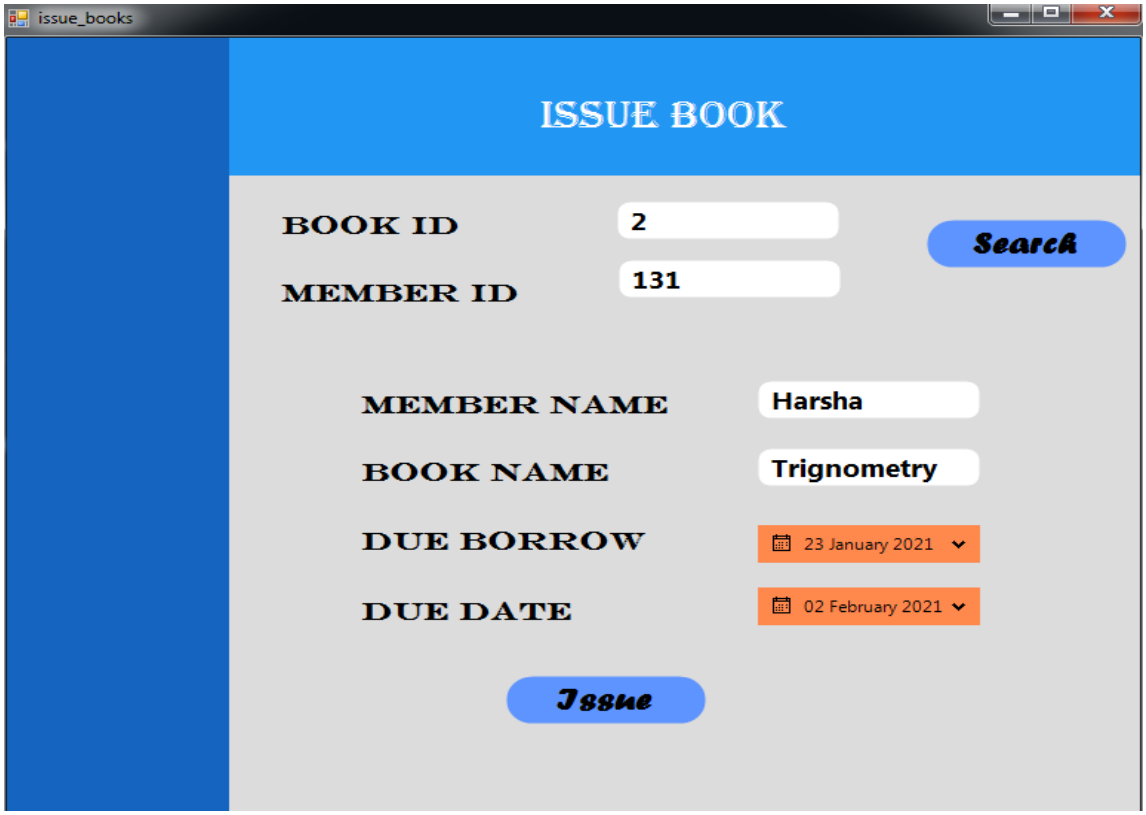
Add books form:



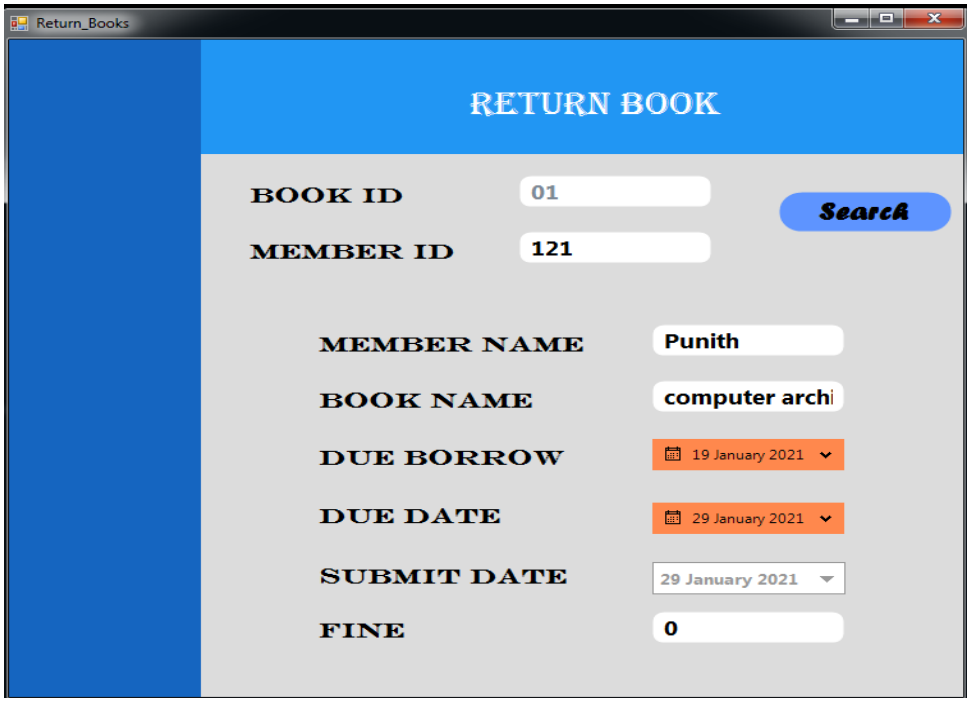
Book details form:



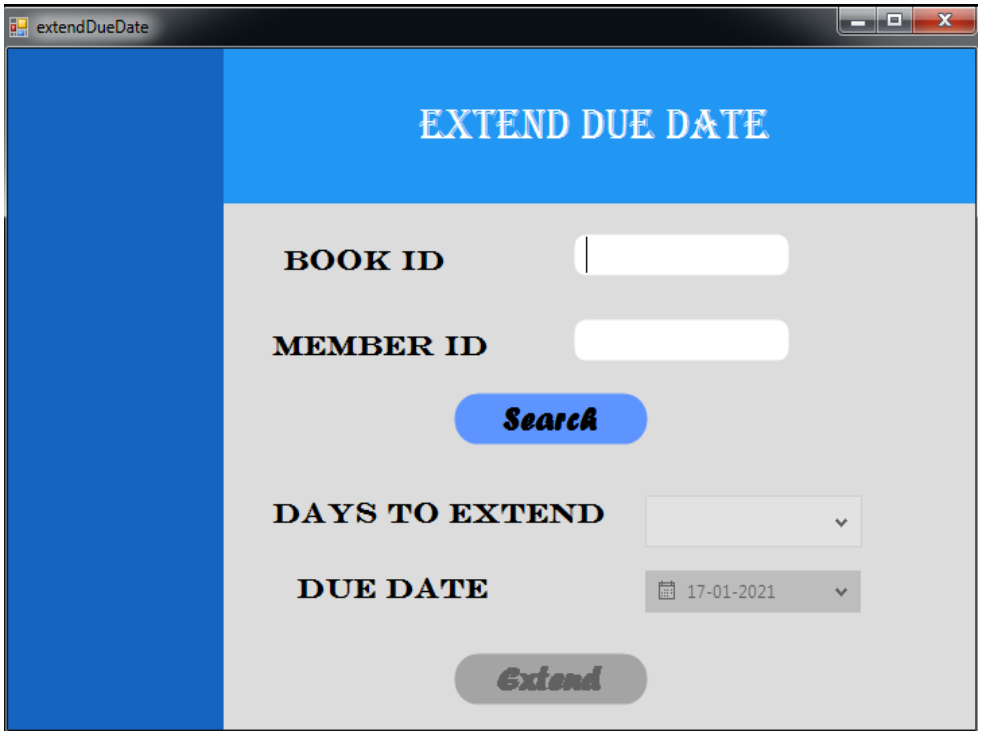
Issue books form:



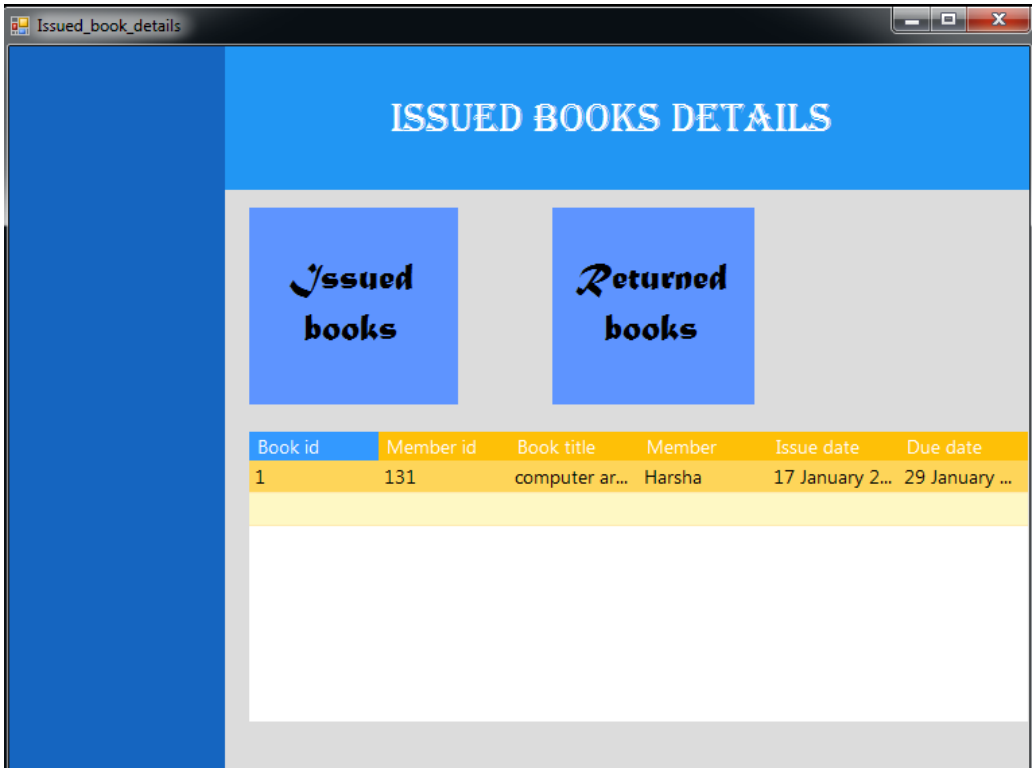
Return book form:



Extend due date form:



Issued/returned book details form:



9. TESTING

Software testing is performed to verify that the completed software package functions according to the expectations defined by the requirements/specifications. The overall objective to not to find every software bug that exists, but to uncover situations that could negatively impact the customer, usability and/or maintainability.

PURPOSE OF TESTING

- Finding defects in software which may get created by programmer while developing the software.
- To prevent defects.
- To make sure the end result meets business and user requirements.
- To ensure that it satisfies BRS that is Business Requirements Specification and SRS that is Software Requirements Specifications.
- To gain the confidence of the customers by providing them quality product.

9.1 Types of testing:

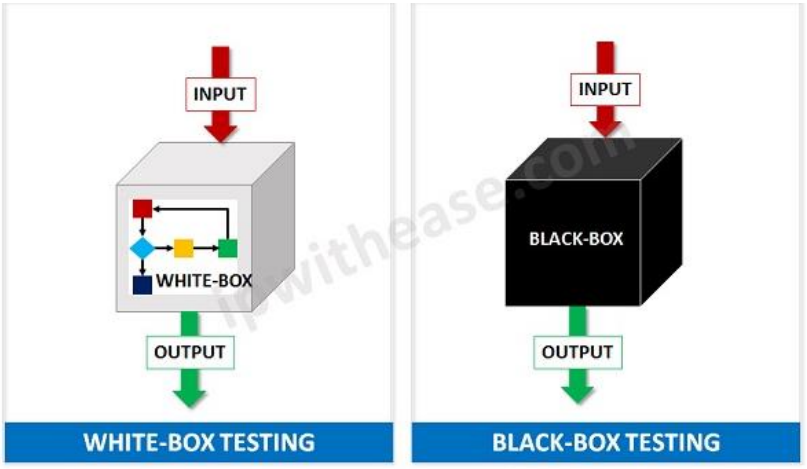
1. White Box Testing

It is a software testing method in which the internal structure/ design/ implementation of the item being tested is known to the tester. The tester chooses inputs to exercise paths through the code and determines the appropriate outputs. Programming know-how and the implementation knowledge is essential. This method is named so because the software program, in the eyes of the tester, is like a white/transparent box; inside which one clearly sees. Internal software and code

working should be known for this type of testing. Tests are based on coverage of code statements, branches, paths, conditions. Also known as structural testing and Glass box Testing.

2. Black box testing

Internal system design is not considered in this type of testing. Tests are based on requirements and functionality. This method is named so because the software program, in the eyes of the tester, is like a black box; inside which one cannot see. Black box testing is a testing technique that ignores the internal mechanism of the system and focuses on the output generated against any input and execution of the system. It is also called functional testing.



Levels of testing

UNIT TESTING

Unit Testing is a level of the software testing process where individual units/components of a software/system are tested. The purpose is to validate that each unit of the software performs as designed.

INTEGRATION TESTING

Integration Testing is a level of the software testing process where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units.

SYSTEM TESTING

This is the next level in the testing and tests the system as a whole. Once all the components are integrated, the application as a whole is tested rigorously to see that it meets Quality Standards. This type of testing is performed by a specialized testing team.

System Testing is a level of the software testing process where a complete, integrated system/software is tested. The purpose of this test is to evaluate the system's compliance with the specified requirements.

ACCEPTANCE TESTING

Acceptance testing or User Acceptance Testing is a level of the software testing process where a system is tested for acceptability. The purpose of this test is to evaluate the system's compliance with the business requirements and assess whether it is acceptable for delivery

9.3 TEST CASES

Test Scenario 1: Enter proper credentials, login as Admin and check for successful login.

Test Case 1:

Step#	Description	Input	Expected result	Actual result	Status
1	Open the Application	N/A	'Library Management System' Login form must be displayed	'Library Management System' Login form is displayed	Pass
2	Enter username. and password and click on login	Username: Manoj Password: xyzlm	Admin main form must be displayed	Admin main form is displayed	Pass

Test Scenario 2: Login as admin by leaving password field blank and check for error message

Test Case 2:

Step#	Description	Input	Expected result	Actual result	Status
1	Open the Application	N/A	'Library Management System' Login form must be displayed	'Library Management System' Login form is displayed	Pass

2	Enter username and password and click on login	Username: Manoj Password: xyzlm	Enter password message must be displayed	Enter password message is displayed	Pass

Test Scenario 3: Login as admin, Add Courses.

Test case 3:

Step#	Description	Input	Expected result	Actual result	Status
1	Login as admin	username: manoj Password: xyzlmnop	Login form must be displayed	Login form is displayed	Pass
2	Click on courses text box	N/A	Focus must be given to courses. text field	Focus is given to Courses text field	Pass
3	Enter proper details Click on Add	Enter a Course	Course successfully added message must be displayed	Course successfully added message is displayed	Pass

Test Scenario 4: Login as admin, add Categories.

Test case 4:

Step#	Description	Input	Expected result	Actual result	Status
1	Login as admin	username: manoj Password: xyzlmnop	Login form must be displayed	Login form is displayed	Pass
2	Click on categories text box	N/A	Focus must be given to categories text field	Focus is given to categories text field	Pass
3	Enter proper details Click on Add	Enter a Category	Category successfully added message must be displayed	Category successfully added message is displayed	Pass

10. CONCLUSION

The project **LIBRARY MANAGEMENT SYSTEM** is for computerizing the working in a library.

The software takes care of all the requirements of a library and is capable to provide easy and effective storage of information related to books & users.

The Library Management System allows the user to store the book details and the persons details.

This software allows storing the details of all the data related to library

The implementation of the system will reduce data entry time and provide readily calculated reports.

More advantages of library management system are:

- Decreases paperwork's.
- Less manual work.
- Increases accuracy in storing data.
- Saves time.
- Easy to check book availability.
- Keeps proper record of fine amount.

11. FUTURE ENHANCEMENT

Since user requirements keeps changing, the software has to be upgraded according to user needs. In future the purposed work can be enhanced in following ways.

- Since the project is based on object-oriented design, any future change can be easily adaptable.
- Based on future security issues, it can be improved using emerging technologies.
- SMS notifications can be added to alert users who haven't returned book.
- Use of more classes and modules to shot down the code.
- Changes made by any user can be monitor.
- Adding packages for members to borrow certain no. books by paying certain amount per month.
- Add online payment option while paying.
- More interactive UI.

BIBLIOGRAPHY

REFERRED TEXTBOOKS

- "Visual Programming" -by Padma Geetha B G & Srikanth S
- "Software Engineering a Practitioners Approach: TATA McGraw Hill series Fifth Edition by Roger S Pressman, Ph.D.
- "Database Management System" -by Dr. Rajiv Chopra.

REFERRED WEBSITES

- www.w3schools.com
- www.youtube.com
- www.vb.net-informations.com
- www.wikipedia.com
- www.tutorialspoint.com

APPENDIX(SYNOPSIS)

TITLE: LIBRARY MANAGEMENT SYSTEM

ABOUT THIS PROJECT

Library management system is a project which aims in developing a computerized system to maintain all the daily work of library. This project has many features which are generally not available in normal library management system like user or faculty login. It has a faculty of admin login through which the admin can monitor the whole system. The user after logging in to his account i.e , admin account can generate various reports such as student details, book borrowed by students and returned, book entries such as name of the book, author and category. The aim of the project is to make the manual handling of library system in to computerized system which includes all above features.

The "Library Management Sytem" has been developed to override the problems prevailing in the practicing manual system. This software is supported to eliminate and in some cases reduce the hardships faced by this existing system. Moreover this system is designed for the particular need of the company to carry out operations in a smooth and effective manner. The application is reduced as much as possible to avoid errors while entering the data. It also provides error messages while entering invalid data.No formal knowledge is needed for the user to use this system. Thus by this all it provesit is user-friendly. Library Management system, as described above, can lead to error free, secure, reliable and fast management system. It can assist the user to concentrate on their other activities rather to concentrate on the record keeping. Thus it will organization in better utilization of resources. Every organization, whether big or small has challenges to overcome and managing the information of books, student, address, and etc. These systems will ultimately allow you to better manage resources.

OBJECTIVE

A Library Management System gives access to and manage the resources in your library. A well-chosen system will increase your library's efficiency, save administration time, lead to a better educational experience, information and to use them effectively. This project is helpful to track all the book and library information. The software will be able to handle all the necessary information.

Existing system:

In our existing system all the transactions of books are done manually, so taking more time for transaction like borrowing a book or returning a book and also for searching of books. Another major disadvantage is that to preparing the list of books borrowed and the available books in the library will take more time, currently it is doing as a one-day process for verifying all the records. So after conducting the feasibility study we decided to make a computerized library management system.

Proposed system:

Proposed system is an automated Library Management System. Through our software user can add members, add books, search members, search books, update information, edit information, borrow and return books in quick time. Our purposed system has the following advantages.

- User friendly interface
- Fast access to database
- Less error
- More storage capacity
- Quick transaction

All the manual difficulties in managing the library have been rectified by implementing computerization.

1. ADMIN

•Log In:admin can login to the application by providing the valid credentials to access the application.

•Manage member:admin will add members with proper details.

•Manage book:admin will add book with proper details and information.

•Delete member:admin can delete member detrails.

•Delete book:admin can delete book details.

•Issue book:admin can issue book if the member requests for books.

•Return book:members can return books to admin.

•Extend due date:admin can extend the due date of the book.

•Manage course:admin can add/delete course.

•Manage categories:admin can add/delete book categories.

•Logout:admin can logout of the application.

2. MEMBER

•Login:member can login in to the application using username and password.

•view book:member can view books and details of it.

SOFTWARE REQUIREMENTS:

HARDWARE INTERFACES

Processor	:	i3+
RAM	:	4GB
Hard Disk	:	80GB
Speed	:	1.2 GHz+

SOFTWARE INTERFACES

Operating System	:	Windows 7 or Higher
IDE	:	Visual Studio 2019
Language	:	Visual Basic
Framework	:	VB.NET 4.0
Back End	:	MY SQL